Models and Control Strategies for Data Center Energy Efficiency

Submitted in partial fulfillment of the requirements for
the degree of
Doctor of Philosophy
in
Department of Electrical and Computer Engineering

Luca Parolini

Bachelor (Laurea Triennale) in Information Engineering,
Department of Information Engineering, University of Padua, Italy

M.Sc. (Laurea Specialistica) in Automation Engineering,
Department of Information Engineering, University of Padua, Italy

Carnegie Mellon University
Pittsburgh, PA

February 2012

Keywords: Data centers, Cyber-Physical systems, Energy Efficiency, Thermal management, Computer applications, Predictive control.

Abstract

As the foundation of the nation's information infrastructure, data centers have been growing rapidly in both number and capacity to meet the increasing demands for highly-responsive computing and massive storage. Data center energy consumption doubled from 2000 to 2006, reaching a value of 60 TWh/year (Tera Watt hour / year). Coupled with increasing power and cooling demands imposed by the Moore's law and with the quest for high density data centers, this trend has been rapidly raising the energy cost associated with data centers.

Data centers are large cyber-physical systems (CPSs) with hundreds of variables that can be measured and controlled. Dynamics of the controlled processes span multiple time scales: electricity costs can fluctuate hourly, temperatures evolve in the order of minutes, and CPU power states can be changed as frequent as milliseconds. Processes also differ in the spatial areas they influence: computer room air conditioners (CRAC) affect the inlet air temperatures of multiple servers, whereas CPU power states affect only single servers. The large number of constraints and their heterogeneity in nature make data center control a challenging research problem.

This dissertation considers data centers as CPSs, with a focus on run-time management and operating costs. The proposed modeling framework explicitly captures the cyber-physical nature of data centers and allows the development of models that represent both the computational and the thermal characteristics of a data center, as well as their interactions. The proposed control strategy attempts to manage both the computational and the thermal characteristics of a data center. The control strategy is based on a hierarchical/distributed control architecture that takes advantage of the modularity typically found in data centers. The hierarchy constitutes of three control levels. The lower levels of the hierarchy deal with fast dynamic processes, while the higher levels deal with the bulk thermal management and the coordination of the controllers at the lower levels. The focus of this dissertation is on controllers at the highest level of the hierarchy, which we call *data center level*. Three controllers are proposed for the data center level. Each controller takes advantage of the thermal-computational model in a different way. The performances of the controllers are compared in simulation under a variety of scenarios.

In the dissertation, it is shown that every data center has operating regimes for which simple control strategies can be as optimal as advanced ones that consider current and future, i.e., estimated, evolutions of the data center. The dissertation also discusses the existence of data center cases for which simple control strategies are always as optimal as advanced ones, regardless of the operating regime. An index, called the *cyber-physical index* (CPI) is proposed to provide a priori estimates of the savings that can be gained when using advanced control strategies that consider the coupling between the computational and the thermal characteristics of a data center, rather than simpler control strategies that do not account for this coupling. A possible interaction between data centers and the power-grid is also discussed. The interaction with the power-grid is designed so that data center controllers can take advantage of a time-varying electricity price. Finally, some results about the interactions between the controllers at the lower levels of the hierarchy and the controller at the data center level are discussed.

Acknowledgments

I express my deepest gratitude to my advisors Professor Bruce H. Krogh and Professor Bruno Sinopoli for their support, guidance and enthusiasm in my research. Working with them has been my greatest privilege and fortune. Sincere gratitude is extended to my dissertation committee members: Professor David Andersen, Dr. William Courtright, and Dr. Zhikui Wang for the thoughtful discussions and valuable comments.

Special thanks go to Claire Bauerle and Carolyn Patterson. Porter Hall B-level would not be that warm and pleasant place to work and study without them. I also would like to thank Elaine Lawrence. Her help in getting over all of the procedures for Ph.D. study has been invaluable.

A special thank also goes to the Parallel Data Laboratory (PDL) group of Carnegie Mellon University (CMU). The students and professors of the PDL group gave me valuable advises and feedback about my research work. I really appreciated their help. Among the many people of the PDL group, a special thank goes to Professor Greg Ganger, Karen Lindenfelser, Joan Digney, and Michael Stroucken.

I would like to acknowledge the technical support given by the Sustainable Ecosystems Research Group at HP Labs, CA. A special thank goes to Dr. Niraj Tolia former senior researcher at the Exascale Computing Lab at HP Labs, Palo Alto, CA.

I acknowledge the financial support given by the National Science Foundation and by the Pennsylvania Infrastructure Technology Alliance. This dissertation was supported in part by the National Science Foundation under Grant ECCS 0925964 and by the Pennsylvania Infrastructure Technology Alliance under Grant C000032167.

Friendship is one of the most valuable assets I had during the years of my PhD study. I am fortunate to have met Mohammadreza Aghajani, Matthias Althoff, Roberto Ambrosino, Nikos Arechiga, Dragana Bajovic, Kyri Baker, Ajinkya Bhave, Ellery Blood, Rohan Chabukswar, Milos Cvetkovic, Sanja Cvijic, Yue Ding, Emanuele Garone, Andrew Hsu, Dusan Jakovetic, Jhi-Young Joo, Yilin Mo, Sergio Pequito, Paolo Piovano, Andrea Qualizza, Akshay Rajhans, Aurora Schmidt, Marco Vincenzi, Ker-Jiun Wang, Jim Weimer, Le Xie, Xiaoqi Yin, Sabina Zejnilovic, Yangfang Zhou, and others too numerous to name.

I am in great debt to my family and to my girlfriend, for their love and support throughout my Ph.D. study. This dissertation is dedicated to them. My friends in Padova, Italy, deserve a special thank. We know each others since we were kids and despite the distance across the Atlantic ocean, they have been my constant source of encouragement.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Data centers are large facilities hosting massive numbers of servers and the associated support infrastructure. Servers can be used for several purposes, including interactive computation, batch computation, and real-time transaction. Data centers can be seen as a composition of information technology (IT) systems and the support infrastructure. The IT systems provide services to the end users and the support infrastructure supports the IT systems by supplying power and cooling. The IT systems include servers, storage devices, networking devices, middleware and software stacks, e.g., hypervisors, operating systems, and applications. The support infrastructure includes backup power generators, uninterruptible power supplies (UPSs), power distribution units (PDUs), batteries and the cooling technology (CT) systems. The CT systems include server fans, computer room air conditioners (CRACs), chillers, and cooling towers [13, 15, 21, 54, 57, 77].

Figures 1.1 and 1.2 show two examples of data centers. The data center in Fig. 1.1, a Facebook data center located in North Carolina, USA, is an example of a large-scale data center, where the number of servers is of the order of tens of thousands. The data center in Fig. 1.2 is an example of a medium-size data center, where the number of servers is of the order of thousand or less. This data center, located at the Parallel Data Laboratory (PDL)[1] of Carnegie Mellon University, Pittsburgh, PA, USA, is constructed following a modular

[1]`http://www.pdl.cmu.edu/`.

Figure 1.1: Aerial view of the Facebook data center in North Carolina, USA. The facility occupies an area of approximately 300,000 ft^2 [43].

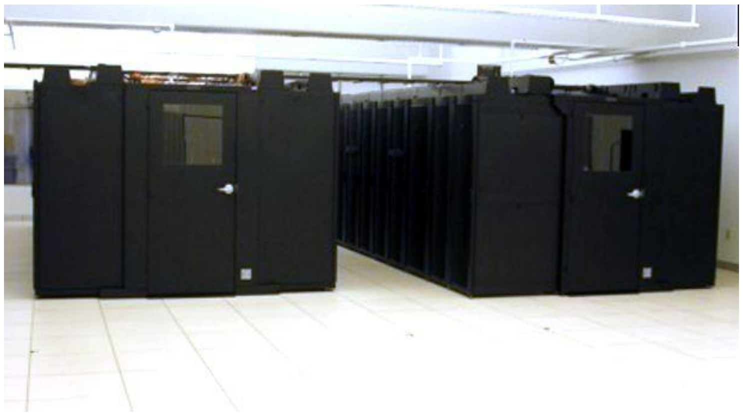

Figure 1.2: Zone 1 and zone 2, respectively on the left and on the right side of the picture, of the data center located at the Parallel Data Laboratory of Carnegie Mellon University, Pittsburgh, PA.

architecture. In a modular architecture, IT systems and the support infrastructure are grouped into multiple and independent zones. Every zone contains its own IT systems and support infrastructure. A modular architecture allows a reduction of the initial costs for building the data center and it also provides redundancy of power and cooling resources by design. Figure 1.3 shows the modular approach proposed by Sun Microsystems, Inc. In contrast to the approach used in the data center at Carnegie Mellon University, the modular approach to the data center construction proposed by Sun does not require a

Figure 1.3: Modular data center approach proposed by Sun Microsystems, Inc.

physical building to operate and hence, it further reduces the initial costs to build a data center.

In general, the goal of the data center operators is the maximization of the data center efficiency, subject to a set of operational constrains and to constraints on the reliability and on the availability of the provided services. The large number of constraints and their heterogeneity in nature, e.g., some constraints may be related to physical constraints of the data center subsystems and others may be related to software (cyber) constraints, make data center control a cumbersome task and a challenging research problem. We believe that control algorithms based on cyber-physical models of data centers can improve current control strategies by leveraging information about how the workload distribution affects the overall power consumption and the quality of service (QoS) provided to the users.

The following section discusses the trends in data center growth. Section 1.2 presents a classification of data centers based on the expected availability of power and cooling they can provide to the IT systems. Section 1.3 discusses an approach to quantify the contribution of different elements to the run-time cost of operating a data center. Section 1.4 discusses the constraints typically considered in data center control problems with a particular focus on the bounds on the temperature and humidity of the air at the inlet of IT systems. Section 1.5 discusses the factors that make data center control a challenging

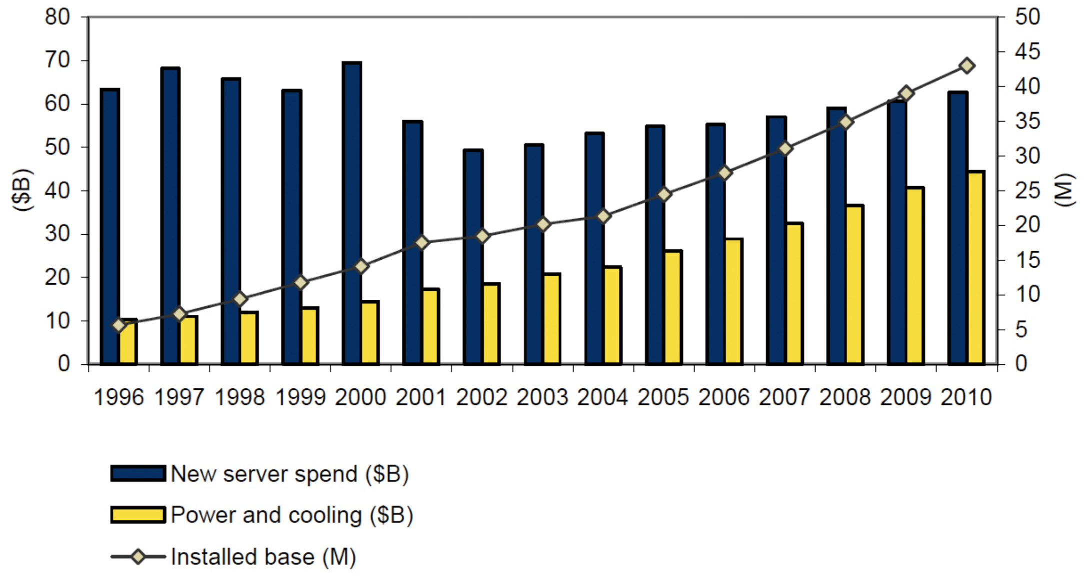

Figure 1.4: Data center spending trend from 1996 till 2010 [66].

research problem. Finally, Sec. 1.6 summarizes the contributions of this dissertation.

1.1 Trends in data center growth

The number of data centers is rapidly growing throughout the world, fueled by the increasing demand for remote storage and cloud computing services. Figure 1.4 shows the increase in data center expenditures for power, cooling, and new servers from 1996 until 2010. Along with the increase in the data center number, the yearly energy consumption has also increased. A report of the U.S. Environmental Protection Agency (EPA) shows that the average annual energy consumption of data centers in US doubled from 2000 to 2006, reaching a value of 60 TWh/yr (terawatt hour/year) [74]. Estimated future yearly energy consumption values are shown in Fig. 1.5 [14].

Recent research in the development in the powering and cooling infrastructures for data centers has allowed a drastic increase in the efficiency at which data centers operate. For example, the power consumed by non-IT devices in some data centers operated by Google Inc. accounts for less than 15% of the total data center power consumption.[2] Nevertheless,

[2]`http://www.google.com/about/datacenters/inside/efficiency/power-usage.html` .

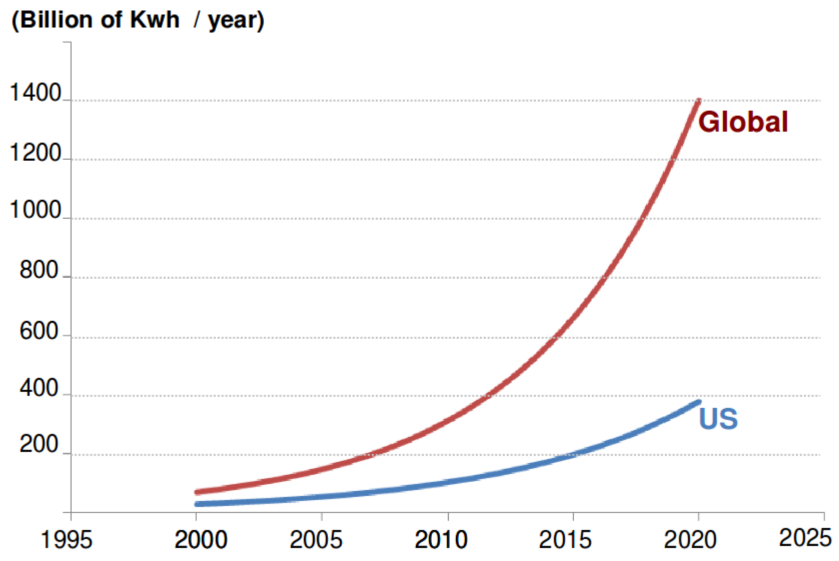

Figure 1.5: Expected yearly energy consumption of data centers in US and over the world [14]. Data after 2005 are predicted values.

modern and extremely efficient data centers are a minority in the global panorama of data centers and the development of software approaches toward the maximization of their efficiency is crucial.

Computational density has also been increasing over the years. Figure 1.6 shows the measured and the expected growth of the power consumption density in data center equipment from 1994 until 2014 [56]. Such a growth impacts data center cooling operations at every scale, from transistors on integrated circuits (ICs), to servers in racks, to racks in a room, leading to nearly equal costs for operating the IT systems and the CT systems [53, 61]. Available cooling capacity has in some cases become the limiting factor for the computational capacity [68].

The increasing demand for cloud computing applications does not explain by itself the large increase in the number of data centers. The development of a cloud infrastructure based on spatially distributed servers which interact over high speed networks is conceivable. There are however, at least three major points that make a data-center-based solution preferable to a solution in which servers are spatially distributed:

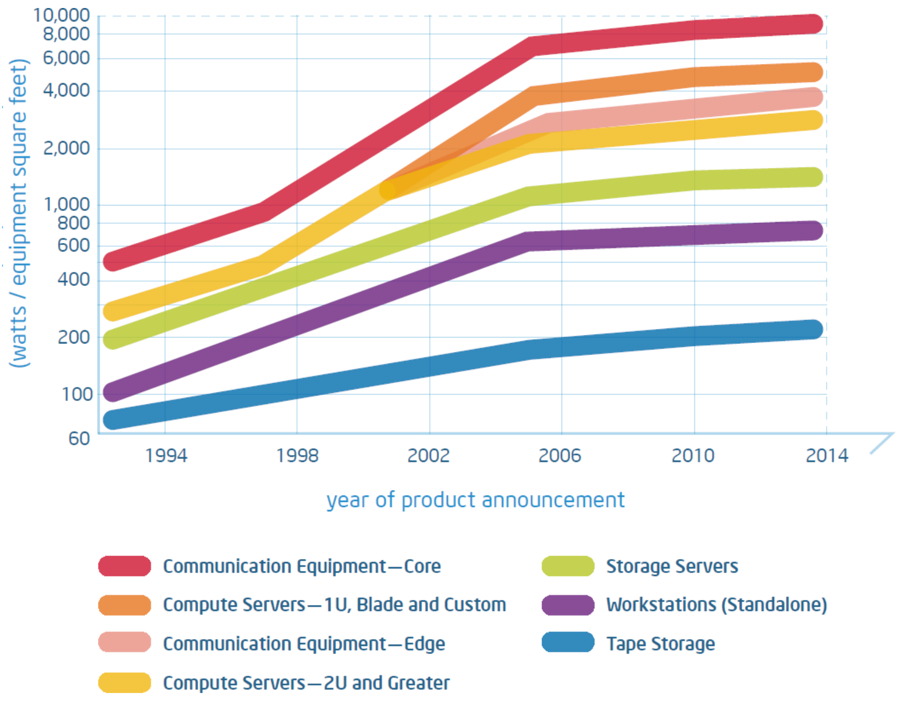

Figure 1.6: Expected increase in power consumption per equipment footprint over time [56]. Data after 2005 are predicted values. Power values are in a logarithmic scale.

- It is easier to maintain a massive number of servers when they are physically collocated, e.g., the Facebook's data center shown in Fig. 1.1 will be managed by less than 50 employees [43].
- When servers are collocated, it is possible to interconnect them with low delay and high bandwidth networks. Reducing the communication delay among servers can reduce the overall delay experienced by the end users, which typically improves the overall QoS.
- When servers are collocated, it is possible to guarantee a high level of data security and privacy. Aside from software security applications, physical barriers and anti-intrusion systems can be installed to prevent unauthorized access [59].

The fast trend at which their energy consumption has increased sets data centers apart from other electricity consumers. In 2007, a report of the EPA estimated that data center efficiency could be drastically improved if state-of-the art control techniques were em-

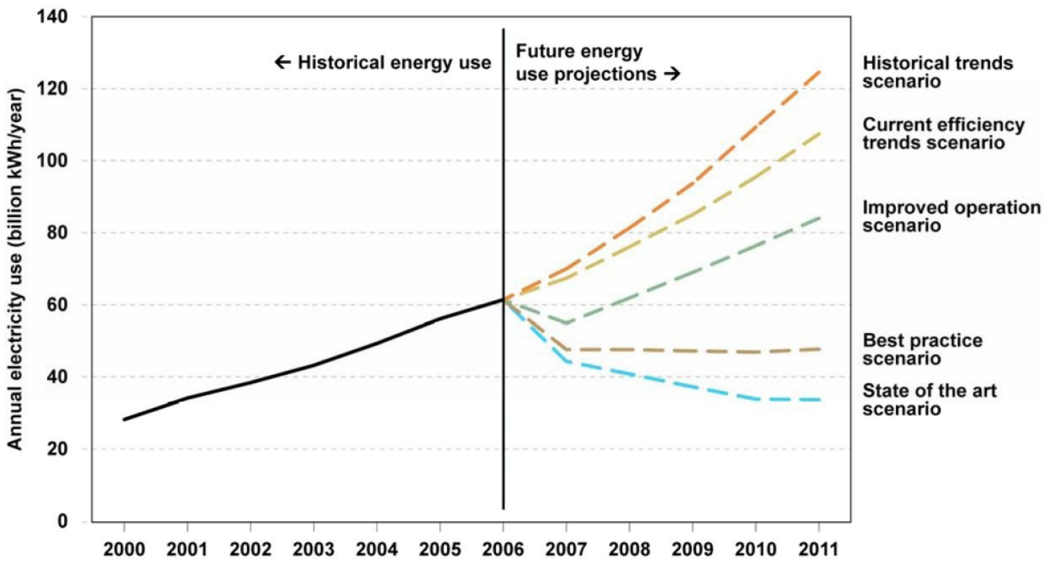

Figure 1.7: Estimated yearly energy consumption of data centers in US, based on different control approaches [74].

ployed [74]. Figure 1.7 shows multiple predicted growths of the yearly energy consumption of data centers, based on different approaches to the maximization of the data center energy efficiency. The report appeared in 2007 and refers to data collected up to 2006.

1.2 Uptime requirements for data centers

A metric often related with cloud computing is the expected availability of the services. It is common practice in industry to approximate such a metric with the annual uptime percentage, defined (a posteriori) as the ratio of the amount of time that a service was active over the whole year. For example, the Amazon Elastic Compute Cloud (EC2) service level agreement (SLA) states:[3]

> "[...] If the Annual Uptime Percentage for a customer drops below 99.95% for the Service Year, that customer is eligible to receive [...]"

Providing an a priori estimation of the annual uptime percentage is often impractical, since its value depends on both the availability of computational and power resources and on the whole software stack. It is often simpler to provide an upper bound on the annual uptime

[3]`http://aws.amazon.com/ec2-sla/` .

percentage. For example, the availability of powering and cooling resources sets an upper bound on the annual uptime percentage. Therefore, a classification of the data centers based on the expected availability of power and cooling resources gives an indication to the service providers of the maximum uptime percentage that they can guarantee to the users. The Telecommunication Industry Association (TIA) recognizes four data center tiers [1]. Tiers are numbered from 1 to 4, where higher numbers are associated with higher expected availability of powering and cooling. For example, a tier 1 data center is supposed to have an expected average annual downtime of 28.8 hours, which implies 99.671% availability of power and cooling. A tier 4 data center has an expected average annual downtime of 0.4 hours, which implies 99.995% availability of power and cooling [67].

1.3 Run-time cost of data centers

In 2008, James Hamilton proposed an approach to quantify the contributions of different elements to the run-time cost of a data center [30].[4] Software licensing and employee costs were not considered in his analysis, since they vary widely among data centers. Furthermore, data centers typically rely on open source and free software to manage the provided services.

Figure 1.8 shows the breakdown of the data center run-time costs based on the Hamilton's analysis. Table 1.1 shows the values of the parameters used in the analysis. *Server* cost represents the average monthly cost to replace servers and is the largest cost of operating a data center. The second largest cost, *power distribution and cooling*, represents the cost for creating the infrastructure to distribute power among the servers and to control the environment where servers operate. The cost of powering a data center, *powering cost*, is the third largest cost for data centers and it accounts for 13% of the total average monthly cost. The cost for the network infrastructure and the cost of infrastructure amortization,

[4]James Hamilton is vice president and distinguished engineer on the Amazon Web Services (AWS) team `http://www.mvdirona.com/jrh/work/` .

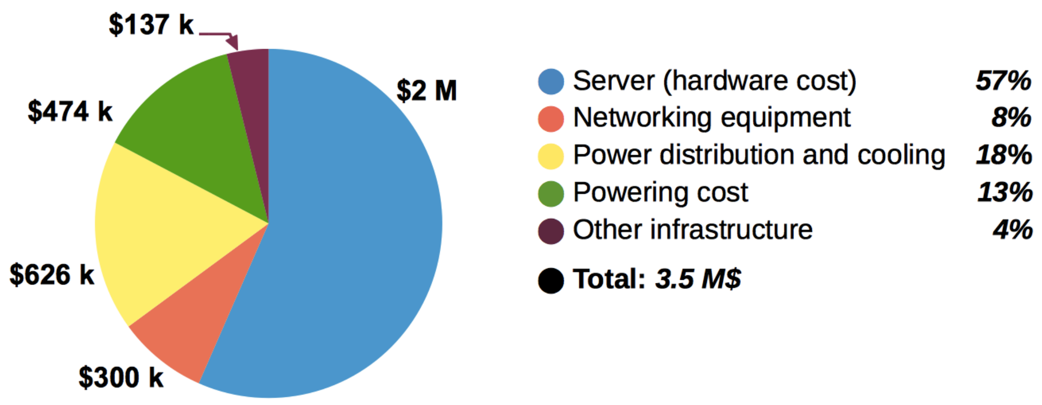

Figure 1.8: Average monthly data center operating costs. Analysis based on the approach proposed by J. Hamilton [30].

Table 1.1: Data used to compute the average monthly data center operating costs shown in Fig. 1.8.

Facility size	8	MW
Cost of power	0.07	$/kWh
Cost per critical watt	9	$/W
Facility amortization period	10	year
Cost per server	1,450.00	$
Server amortization period	3	year
Network equipment amortization period	4	year
Annual cost of money	5	%
Average critical load usage	80	%
Total power consumption over server power consumption, i.e., PUE	1.45	
Percentage of infrastructure that accounts for power and cooling	82	%

respectively, *network equipment* and *other infrastructure* costs, account for 12% of the total average monthly cost.

Although the cost of powering a data center is not the largest one, there are at least four points which make research on maximizing the efficiency of data centers extremely relevant for data center managers:

- The electricity cost is expected to increase over time, whereas the cost of servers is expected to decrease.

- The cost of creating the power and the cooling infrastructure depends on the maximum amount of electrical power consumed by the data center. Therefore, reducing the data center's peak power consumption reduces the cost of creating the physical infrastructure for powering and cooling.
- The peak power consumption of servers is expected to increase over time.
- Depending on the data center characteristics, the impact of maximizing its operating efficiency on the overall electricity consumption may be very large or negligible. However, even considering the case where only 3% of the average monthly electricity cost would be reduced for the case shown in Fig. 1.8, the electricity cost reduction would be of $14,220 per month. If we consider the "state of the art scenario" discussed in the EPA report [74], then the average energy reduction is about 70%, which implies a reduction in the powering cost of $331,000 per month.

1.4 Constraints in data center control

The large number of constraints that data center control algorithms have to enforce is due to multiple and heterogeneous factors. For example, the capacity of the power lines and the availability of cooling resources may bound the maximum amount of workload that can be executed in certain areas of the data center. Software licensing and data availability may also set constraints on which servers can execute certain kinds of workload.

Typically, the control constraints widely vary from one data center to another. An exception, in the case of air cooled data centers, is given by the constraints on the temperature and the humidity of air at the inlet of IT systems. Air is by far the most widely used medium for removing heat from IT systems. Although liquid cooling is a promising technology, particularly for high-density data centers, high installation costs and safety concerns have prevented the proliferation of this technology [32, 41]. Constraints on the

Table 1.2: ASHRAE recommendations in 2004 and in 2008. RH stands for relative humidity and DP stands for dew point.

	Recommendation in 2004 [4]	Recommendation in 2008 [5]
Low-end temperature	20 °C	18 °C
High-end temperature	25 °C	27 °C
Low-end moisture	40 %RH	5.5 °C (DP)
High-end moisture	55 %RH	69 % RH and 15 °C (DP)

inlet air impose limits on both the temperature and on the humidity of the air. If the air is too dry, static discharge can be generated among internal server components. On the other hand, if the air is too humid, moisture can be deposited on some areas of the servers. Bounds on the air temperature are set to prevent overheating of server components. In 2004, the American Society of Heating, Refrigerating and Air-conditioning Engineers (ASHRAE) defined limits on the humidity and temperature of the inlet air for IT systems [4]. The limits were relaxed in 2008 in order to allow cooling control algorithms to increase the CT efficiency [5]. Table 1.2 shows a comparison between the 2004 and the 2008 recommended environmental conditions for data centers.

1.5 Issues in data center control

The goal of data center control algorithms is the maximization of the data center performance with respect to a given metric. In contrast to the usual focus on closed-loop stability in control system design, here the focus is on the maximization of the given performance metric. Since data center control algorithms do not attempt to stabilize the state of the data center around a reference value, analysis and modeling of the data center around a given equilibrium point tend to be inappropriate and unimportant.

Data centers are large-scale systems. The number of state variables ranges in the order of ten of thousands for a medium-size data center. The number of sensors and actuators can also range in the order of thousands. As we later discuss, it is often the case that CPU temperatures and the amount of resources used in every server are part of a data center's state. If we only consider one CPU per server, one type of resource per server, and we examine the case of a medium-size data center, then the dimension of the state vector is about a few thousand elements. The large dimensionality of the system impacts both the synthesis of the control law and the estimation of the parameters for the model used by the controllers (if a model is used).

Data centers are also cyber-physical systems (CPSs). Both *how* and *where* the workload is executed affect the total energy consumption. The way workload is executed (how) impacts the efficiency at which IT operates. The location where the workload is executed (where) impacts the efficiency at which CT operates. The coupling between the cyber and the physical aspects is apparent at every scale in a data center, and the manipulation of one control variable can have unexpected and detrimental effects on multiple subsystems.

Processes in data centers span multiple time-scales, e.g., they range from few milliseconds at the CPU level to hours at the bulk thermal level. Furthermore, most processes are stochastic in nature. For example the workload arrival and its processing time are uncertain processes that drive almost all other data center operations, e.g., power consumption of the IT systems and the QoS provided to users.

Finally, we observe that no widely accepted model is available for data centers. Comparing different control algorithms is often impossible and the implementation of the proposed control algorithms on small testbeds is generally the only viable approach for demonstrating the performance of a proposed algorithm. Furthermore, the lack of publicly available data about data centers makes the development of models and simulators an arduous task.

1.6 Contributions of the dissertation

The contributions of this dissertation are:

- A modeling framework for data centers. The proposed framework facilitates the development of both simulation-oriented and control-oriented models of data centers.

- A collection of models that show how the proposed framework can be used in a variety of different data center cases.
- A control strategy that takes advantage of the modularity typically found in data centers.
- A collection of algorithms for controlling data centers at different time-scales. While we initially develop a control model to describe data center dynamics from a fine time-scale of few seconds till the slowest processes evolving over multiple hours, the main focus of this dissertation is on algorithms for controlling data centers at the higher level (slower time scale).
- A collection of routines, written in MATLAB, to simulate the behavior of a data center based on the proposed modeling framework. The simulator is used in this dissertation to compare the performance of different control algorithms under a variety of scenarios.

1.7 Dissertation organization

The following chapter reviews relevant work in data center control. The work is classified on the spatial-scale focus of the control algorithms. Chapter 3 discusses a modeling framework for data centers. This chapter also discusses a possible model for data centers and a mathematical formulation of the data center control problem. Chapter 4 discusses the proposed approach for controlling a data center. Three control algorithms are introduced in Chapter 4. The three algorithms rely on data center models in different ways. Chapter 5 compares the performance obtained by the three control algorithms under multiple data center scenarios. Chapter 6 discusses a possible interaction between a data center and the power-grid. Chapter 7 discusses a control algorithm for groups of collocated servers. The proposed control algorithm focuses on the management of the ON and OFF states of the servers. Chapter 8 discusses some results about the effects of model mismatches on a hierarchical control approach for data centers. Finally, we present the conclusions of this dissertation and discuss future work in Chapter 9.

Chapter 2

Literature review

This chapter reviews previous work on data center control. We group previous work based on the spatial scale to which the control algorithms apply. Three scales are considered: *server level*, *group level*, and *data center level*. At every level, controllers can base their control actions considering only the state of the local controlled subsystem, or they can take into consideration the effects on other subsystems. In the first case, the controller acts based on a local view of the system, whereas in the second case the controller acts based on a global view of the system. Control approaches are classified based on the scale of the controlled subsystem. For example, a controller at the server level manages the operations of a single server, even though the way the control actions are chosen may derive from a global-view of the data center.

2.1 Control at the server level

At this level, a large piece of research work has been focusing on controlling the power consumption of central processing units (CPUs) [2, 9, 19, 20, 25, 26, 33, 35, 45]. As in every complementary metal-oxide-semiconductor (CMOS) circuit, the power consumption of a CPU can be divided between its static part and its dynamic part. In a CMOS circuit, the static power consumption is due to leakage currents, whereas the dynamic power consumption is due to movements of the electric charge, induced by variations in the logic states of the circuit's gates [80]. As the size of the CMOS circuit shrinks, the static power consumption tends to increase. The dynamic power consumption increases with the voltage and the frequency at which the circuit operates. We first consider the dynamic energy consumption of a CMOS circuit. The dynamic energy consumption of a CMOS circuit can be approximated as

$$E = C_1 V^2, \tag{2.1}$$

where V is the voltage at which the circuit operates and C_1 is a coefficient whose value depends on the physical characteristics of the circuit [40]. The relationship between the maximum frequency f at which the circuit can operate and the voltage V can be approximated as

$$f = \frac{V - V_t}{C_2 V}, \tag{2.2}$$

where V_t is the threshold voltage and C_2 is a coefficient whose value depends on the physical characteristics of the circuit [40]. The operating voltage V must be greater than the threshold voltage V_t. From (2.1) it can be seen that reducing the operating voltage of a CMOS circuit reduces its dynamic energy consumption. However, the processing frequency also has to be reduced (2.2). The sole control of the frequency f affects the average power consumption of the CMOS circuit, but it does not affect the overall energy consumption.

Dynamic voltage and frequency scaling (DVFS) techniques attempt to modify the operating voltage and the processing frequency of CPUs based on the computational demand. DVFS techniques have been already implemented in many operating systems. For example, the CPU governors in Linux systems reduce the server power consumption while providing computational resources when required. Varma *et al.* [75] discuss a control-theoretic approach to DVFS. Cho *et al.* [19] discuss a control algorithm that varies both the clock frequency of a microprocessor and the clock frequency of the memory.

In order to minimize static power consumption of CPUs, techniques other than DVFS have to be used. One of these is gating, i.e., the zeroing of circuit voltage supply as discussed in the work of Leverich *et al.* [38].

2.2 Control at the group level

At the group level, the focus is on the control of groups of collocated servers. Management of server states, e.g., the ON and OFF state of every server, and migration of the virtual machine (VMs) are the control "knobs" typically considered at this level. An idle server consumes about 60% of its peak power consumption and, in the average, operates between 10% and 50% of its maximum utilization [10, 23, 31, 37]. In order to maximize the efficiency of the group, it would be desirable to operate the active servers at 100% of their maximum utilization and to turn off the unused ones. The main problem with such an approach is the time required to turn a server back on. Such a time, often called *setup time*, is of the order of the minutes and it is typically too large for interactive workload [2, 25].

Padala *et al.* [47] propose a control algorithm to allocate IT resources among servers when these are divided among different tiers. Gandhi *et al.* [25, 26] focus on workload scheduling policies and control of the transitions between the OFF and ON state of servers, as a means to minimize the average response time of a data center given a certain power budget. Wang *et al.* [78] propose a model predictive controller to minimize the total power consumption of the servers in an enclosure, subject to a given set of QoS constraints. The proposed feedback control algorithm manages the voltage and frequency of every CPU of every server and the migration of VMs among the servers in the enclosure. Chen *et al.* [16] discuss a server provision and load balancing algorithm that minimize the total server power consumption, with negligible impact to the QoS. The proposed controller is allowed to turn servers on and off and to change also the workload allocation among servers. The authors discuss a linear model that relates the expected average CPU utilization over a time interval, with the total number of active connection, and with the server and the login rate.

Tolia *et al.* [72] discuss a model-based control approach for coordinated workload performance, server power and thermal management of a server enclosure, where a set of blade servers shares the cooling capacity of a set of fans. The fans are controlled through a multi-input multi-output (MIMO) controller in order to minimize the aggregate power of the fans, while the location-dependent cooling efficiency of the fans is exploited so that the decisions of workload migration can result in least total server and fan power consumption.

Other authors have been working on the development of new IT architectures at the group level so as to maximize the efficiency of the group for certain classes of workload. For example, Vasudevan *et al.* discuss an architecture based on fast array of wimpy nodes (FAWN) [6, 76]. The goal of the proposed architecture is the maximization of the number of queries per joule. Authors focus on read-intensive workloads and on large-scale data-analysis. Via the development of ad hoc software algorithms for parallel sorting and data analysis, the authors are able to reduce the energy spent on massive data analysis by more than 50% [58].

2.3 Control at the data center level

At the data center level, the focus of control algorithms is on the resources shared by multiple IT systems such as cooling and powering. At this level, it is often the case that controllers generate set points for lower-level controllers, rather than directly manipulating the actual control variables. Aggregated views of the data center subsystems are typically employed at this level.

There are at least two reasons that make a data-center-wide control approach necessary. The first reason is that the workload may have to be migrated to servers that belong to different groups. The other reason is that power and cooling capacity has to be shared throughout the data center. For example, the cooling capacity in a raised-floor air-cooled data center can be generated by a chiller plant, distributed to the IT systems through a set of CRAC units, the common plenum under the floor, and the open space above the floor. Sharing the capacity makes the cooling management a data center level control problem [11, 12, 24].

Quershi *et al.* [60] discuss the savings that can be achieved by migrating workload to the data centers locations where the electricity cost is at its minimum. A similar problem is considered in the work of Rao *et al.* [62]. Tang *et al.* [69] discuss a control algorithm that allocates the workload among servers so as to minimize their peak inlet temperatures. This approach reduces the cooling power consumption by maximizing the efficiency of the cooling system.

Bash *et al.* [12] discuss a control algorithm that aims at maximizing the efficiency of the cooling infrastructure, while enforcing the thermal constraints of the IT. A network of temperature sensors are used to collect thermal data about the servers. Anderson *et al.* [7] consider a MIMO robust control approach for managing CT. The authors discuss how a MIMO robust control approach can offer large improvement in the efficiency of the CT with respect to other single-input single-output (SISO) control techniques such as proportional-integral (PI) controllers. Chen *et al.* [17] propose a holistic workload, power and cooling management framework in virtualized data centers through exploration of the location-dependent cooling efficiency in the data center level. Workload migration and consolidation through virtual machine (VM) migration is taken as the main approach for active power management. On the thermal side, the rack inlet temperatures are under active control of a cooling controller. The controller dynamically tunes the cooling air flow rates and temperatures from the CRAC units, in response to the "hot-spots" of the data center. On the cyber side, the authors consider multi-tier IT applications or workloads hosted in VMs that can span multiple servers, and be able to migrate VMs.

The control strategy discussed in this dissertation covers the control aspects considered by the algorithms in the group level and in the data center level. The proposed strategy can also be extended to cover the aspects considered by the controllers at the server level. Similarly to the work of Quershi *et al.* [60] and Rao *et al.* [62], we consider the cost of powering the data center. The focus of this dissertation, however, is on the control of a single data center and hence, the workload that is not executed in the data center is dropped.

Our approach closely aligns with the work of Chen *et al.* [17]. The goal of the proposed control strategy is to consider the efficiency of IT and CT in the same optimization problem and to find the best trade-off between the cost of powering the data center and the revenue obtained by executing the workload with a certain QoS. The major difference from the work of Chen *et al.* [17], is the thermal predictive model of the data center considered in our proposed control approach. The thermal predictive model allows a preemptive over-cooling of the data center so as to take advantage of a time-varying electricity price. Furthermore, the effects of the external environment can be included in the prediction of the thermal dynamic model of the data center.

Chapter 3

Data center modeling

The control problem tackled in this dissertation can be stated as:

> Given a data center, a performance metric, and a collection of constraints, find a control law that maximizes the performance of the data center, while enforcing the given constraints.

The first difficulty that arises in this context is the mathematical formulation of the control problem. This chapter discusses the modeling framework we propose in order to formulate the data center control problem; the following chapter discusses the proposed control approach.

The chapter is organized as follows. The following section discusses the proposed modeling framework. Section 3.2 introduces the notation for vectors and matrices used in the rest of the dissertation. Section 3.3 discusses how different elements of the data center can be represented within the proposed modeling framework. Section 3.4 collects the equations representing different elements of the data center and discusses the dynamics of the whole systems. Section 3.5 discusses a formulation of the data center control problem with respect to the proposed model. At the end of the chapter, Sec. 3.6, discusses possible extensions of the model.

3.1 A modeling framework for data centers

We propose to represent a data center as two coupled networks, as illustrated in Fig. 3.1. One network describes the process of data exchange within the data center and between the data center and the external users. This network is called *computational network*. The second network describes the energy exchange within the data center and between the data center and the external environment. This network is called *thermal network*. We call the components of the computational network *computational nodes* and the components of the thermal network *thermal nodes*. Computational nodes represent computational resources of the data center. For example, a virtual machine (VM) can be represented as a computational node, a software program responsible for handling HTTP requests can also be modeled as a computational node. If we only focus on the computational characteristics of IT systems, then we can represent them as pure computational nodes. Thermal nodes represent the physical elements of the data center, e.g., the silicon die over which a CPU core is implemented, or the hardware resources of a server.

Nodes are connected by directional links. A link from one computational node to another computational node represents the possibility for the first node to transfer data to the second one. A link from one thermal node to another thermal node represents the possibility for the first node to transfer energy to the second one. Links between the nodes of the two networks represent constraints on the dynamic evolution of the two nodes.

Nodes in the two networks communicate with the external environment. Nodes in the computational network receive from and send data to the external environment. The flow of data within the computational network and between the computational network and the external environment is called *workload*. Energy is exchanged among the nodes of the thermal network and between the nodes of the thermal network and the external environment. For example, electricity flows from the power-grid to the devices in the data center.

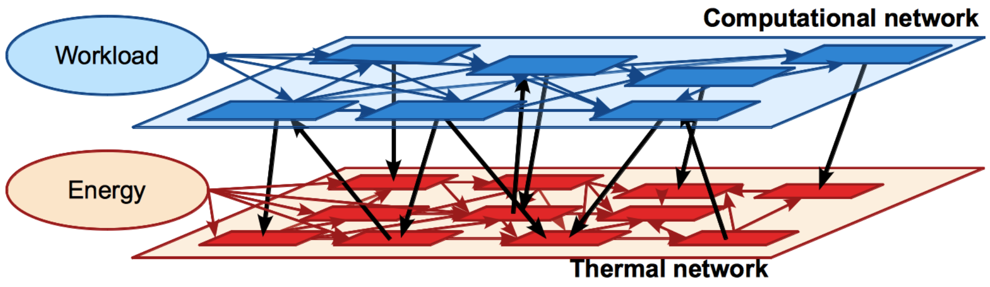

Figure 3.1: Networked model representation of a data center.

The proposed modeling framework highlights two properties of a data center that we believe are crucial for developing more efficient control algorithms:

- The cyber-physical nature of data centers. On the one hand, this point is explicitly captured by the division of data center's devices between computational and thermal nodes. On the other hand, this point is explicitly captured by the links that connect the computational and the thermal nodes.
- The networked nature of data centers. The computational and the thermal networks explicitly capture the fact that data centers are composed by a multitude of elements, which collectively operate in order to provide services to the users.

The first point allows the development of control strategies which focus on both the computational characteristics and the thermal characteristics of a data center. Controllers can now rely on unified, thermal-computational models of data centers and can take advantage of the additional information about how the computational and the thermal characteristics of different devices interact. The second point allows the development of distributed control strategies, where multiple algorithms, each controlling only a part of the data center, collaborate in order to maximize the global performance while enforcing the given constraints. The relevant feature of a distributed architecture is that it can reduce the complexity of a control algorithm that manages both the computational and the thermal characteristics of a data center.

3.2 Notation for vectors and matrices

In this dissertation, vectors are denoted with bold letters and matrices are denoted with capital letters. With $\mathbf{1}$ we denote the column vector whose elements are all 1. With $\mathbf{0}$ we denote the column vector whose elements are all 0. Let $\boldsymbol{x}$ be a $N \times 1$ vector. With x_i we denote the i^{th} element of the vector $\boldsymbol{x}$. Let $\boldsymbol{x}$ and $\boldsymbol{y}$ be two $N \times 1$ vectors. We say that $\boldsymbol{x} \leq \boldsymbol{y}$ if $x_i \leq y_i$ for all $i \in \{1, \ldots, N\}$. With $diag\{\boldsymbol{x}\}$ we denote the $N \times N$ diagonal matrix obtained by placing the elements of the vector $\boldsymbol{x}$ along the main diagonal. Let $\boldsymbol{x}$ be a $N \times 1$ vector and $\mathcal{I}$ be a finite set of ordered indexes whose values ranges between 1 and N. With $\mathcal{I}_j$ we denote the j^{th} element of the set $\mathcal{I}$ and with $|\mathcal{I}|$ we denote the number of elements in $\mathcal{I}$. With $\boldsymbol{x}_{\mathcal{I}}$ we denote the $|\mathcal{I}| \times 1$ vector, in which the j^{th} element of $\boldsymbol{x}_{\mathcal{I}}$ is the element in the $\mathcal{I}_j^{th}$ position of the vector $\boldsymbol{x}$. For example, if $\boldsymbol{x} = \begin{bmatrix} a & b & c & d & e \end{bmatrix}^T$ and $\mathcal{I} = \{2, 3, 5\}$, then $\boldsymbol{x}_{\mathcal{I}} = \begin{bmatrix} b & c & e \end{bmatrix}^T$.

With I we denote the identity matrix. Let X be a square matrix. With $diag_B\{X; y\}$ we denote the square block-diagonal matrix obtained by repeating the matrix X along the diagonal blocks y times. Let A be a $N \times M$ matrix and $\mathcal{I}_1$ and $\mathcal{I}_2$ be two sets of ordered indexes such that, indexes in $\mathcal{I}_1$ range between 1 and N and indexes in $\mathcal{I}_2$ range between 1 and M. With $A_{[\mathcal{I}_1, \mathcal{I}_2]}$ we denote the $|\mathcal{I}_1| \times |\mathcal{I}_2|$ matrix, in which the element in position (i, j) is the element in position $(\mathcal{I}_{1i}, \mathcal{I}_{2j})$ of the original matrix A. For example, if

$$A = \begin{bmatrix} a & b & c \\ d & e & f \end{bmatrix}$$

and $\mathcal{I}_1 = \{1, 2\}$ and $\mathcal{I}_2 = \{2\}$, then

$$A_{[\mathcal{I}_1, \mathcal{I}_2]} = \begin{bmatrix} d & e \end{bmatrix}.$$

3.3 An example of data center modeling

This section discusses how to derive a model of a data center within the proposed framework. In this example, computational nodes represent VMs and thermal nodes represent IT systems and the support systems. CRAC units are modeled as pure thermal nodes, i.e., we neglect the computational characteristics of the controllers embedded in the CRAC units. Other elements such as routers, switches, or storage units are treated as uncontrollable heat sources and modeled as pure thermal nodes. The model focuses on the case of air-cooled data centers. Data centers based on liquid-cooling can still be modeled using the proposed framework, but they may require a different model of heat exchange among the data center devices.

In the following, the letter τ is the continuous-time variable and we consider $\tau \in \mathbb{R}$.[1] For the sake of an uncluttered discussion of the model, this model assumes that the number of nodes in the two networks is constant. The case where the number of nodes changes over time can be described, but it requires a more cumbersome notation.

Let V be the number of computational nodes, i.e., the number of virtual machines (VMs) allocated in the data center. We denote with $\mathcal{V}$ the set of indexes of the computational nodes, i.e., $\mathcal{V} = \{1, \dots, V\}$. Since every VM is univocally associated with a computational node, we address the VM represented by the i^{th} computational node as the i^{th} VM. Thermal nodes are divided into three classes: *server*, *CRAC*, and *environment.* Environment nodes represent those entities that affect the heat exchanged among CRAC units and servers, but that cannot be directly controlled. For example, servers over which no control can be imposed are environment nodes. Uninterruptible power supplies (UPSs), power distribution units (PDUs), switches and other network and support devices are considered here as uncontrollable elements and modeled as environment nodes. The environment surrounding the data center is also modeled as an environment node and

[1]With $\mathbb{R}$ we denote the set of real numbers.

hence, the heat exchanged between the data center and the environment is represented as heat exchanged among the nodes of the thermal network.

With N we denote the number of server nodes, with C we denote the number of CRAC nodes and with E we denote the number of environment nodes. The letter M is used to denote the total number of thermal nodes, i.e., $M = N + E + C$. Thermal nodes are ordered. Indexes from 1 to N represent thermal server nodes, indexes from $N+1$ to $N+C$ represent CRAC nodes, and indexes from $N+C+1$ to M represent environment nodes. We also define the sets of indexes $\mathcal{N} = \{1, \dots, N\}$, $\mathcal{C} = \{N+1, \dots, N+C\}$, and $\mathcal{E} = \{N+C+1, \dots, M\}$. Since every server is univocally associated with a thermal node, we address the server represented by the i^{th} server node as the i^{th} server. Similarly, we address the CRAC unit represented by the i^{th} CRAC node as the i^{th} CRAC unit. Note that the i^{th} CRAC node has index $N+i$. We also address the generic device represented by the i^{th} thermal node as the i^{th} device.

3.3.1 Modeling the heat exchange within the thermal network

In air cooled data centers, air flows into a device through the inlets, it exchanges heat with the device's internal component, and then it exits the device through the outlets. Fans force the air to move from the inlets to the outlets. We consider the case where every device has one inlet, one outlet, and one fan. The extension to the multiple inlets, outlets, or fans case is possible, but it requires a more complex notation.

Let $\mathsf{f}_i(\tau)$ (Kg/s) be the rate at which air flows into and out from the i^{th} device at time τ. Every device exchanges heat with the air and all air flows are greater than 0. The temperature of the air flowing into the i^{th} device is called the *input temperature* of the i^{th} thermal node and its value at time τ is denoted with $T_{\text{in},i}(\tau)$ (K).[2] The temperature of the air flowing out from the i^{th} device is called the *output temperature* of the i^{th} thermal node and its value at time τ is denoted with $T_{\text{out},i}(\tau)$ (K).

[2]With (K) we denote the Kelvin unit of measurement.

If we neglect the variations in the pressure of the air, then the amount of heat entering the i^{th} thermal node over the interval $[\tau, \tau + h)$ is

$$\int_{\tau}^{\tau+h} T_{\text{in},i}(t)\mathsf{f}_i(t)c_p \, dt, \tag{3.1}$$

where h is a nonnegative real number and c_p (J/ (Kg K)) is the specific heat of the air at ambient pressure. The amount of heat leaving the i^{th} thermal node over the same interval is

$$\int_{\tau}^{\tau+h} T_{\text{out},i}(t)\mathsf{f}_i(t)c_p \, dt. \tag{3.2}$$

According to the model proposed by Tang *et al.* [70], the airflow moving into the i^{th} device at time τ is the composition of the flows coming from the other devices and from the i^{th} device itself. The model also assumes that heat exchanged within the data center is solely due to the air movement. Other forms of heat transfer, such as radiation and conduction, are neglected. Let $\gamma_{i,j}(\tau)$ be the relative amount of air that moves from the j^{th} device to the i^{th} one at time τ. For all $i \in \{1, \dots, M\}$, we can write

$$\mathsf{f}_i(\tau) = \sum_{j=1}^{M} \gamma_{i,j}(\tau)\mathsf{f}_j(\tau). \tag{3.3}$$

The coefficients $\{\gamma_{i,i}(\tau)\}$ represent the relative amount of air that recirculates from the outlet of the i^{th} device to its own inlet at time τ. Figure 3.2 provides a graphical representation of the recirculation model.

Over the interval $[\tau, \tau + h)$, the total amount of heat received by the i^{th} thermal node is

$$\sum_{j=1}^{M} \int_{\tau}^{\tau+h} \gamma_{i,j}(t)T_{\text{out},j}(t)\mathsf{f}_j(t)c_p \, dt. \tag{3.4}$$

Eq. (3.4) represents the total amount of heat entering the i^{th} thermal node over the interval

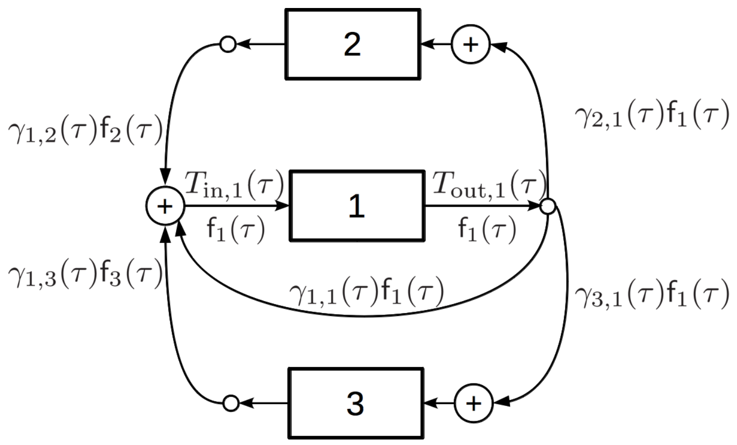
2
1
3
+
$\gamma_{1,2}(\tau)\mathsf{f}_2(\tau)$
$\gamma_{2,1}(\tau)\mathsf{f}_1(\tau)$
$T_{\text{in},1}(\tau)$
$\mathsf{f}_1(\tau)$
$T_{\text{out},1}(\tau)$
$\mathsf{f}_1(\tau)$
$\gamma_{1,3}(\tau)\mathsf{f}_3(\tau)$
$\gamma_{1,1}(\tau)\mathsf{f}_1(\tau)$
$\gamma_{3,1}(\tau)\mathsf{f}_1(\tau)$

Figure 3.2: Example of heat exchange among nodes in a thermal network composed by 3 nodes. In the figure are shown the flows related to thermal node 1 only.

$[\tau, \tau + h)$ and hence, we can write

$$\int_{\tau}^{\tau+h} T_{\text{in},i}(t)\mathsf{f}_i(t)c_p \, dt = \sum_{j=1}^{M} \int_{\tau}^{\tau+h} \gamma_{i,j}(t)T_{\text{out},j}(t)\mathsf{f}_j(t)c_p \, dt. \tag{3.5}$$

Since (3.5) holds for every $h > 0$, we can write

$$T_{\text{in},i}(\tau)\mathsf{f}_i(\tau)c_p = \sum_{j=1}^{M} \gamma_{i,j}(\tau)T_{\text{out},j}(\tau)\mathsf{f}_j(\tau)c_p \tag{3.6}$$

and (3.6) holds almost everywhere. In the rest of the dissertation, we assume that (3.6) holds for every $\tau \in \mathbb{R}$. For all $\tau \in \mathbb{R}$ and $i \in \{1, \ldots, M\}$, $\mathsf{f}_i(\tau) > 0$ and hence, the relationship between the input temperature of the i^{th} thermal node and the output temperatures of all the other thermal nodes can be written as

$$T_{\text{in},i}(\tau) = \sum_{j=1}^{M} \frac{\gamma_{i,j}(\tau)\mathsf{f}_j(\tau)}{\mathsf{f}_i(\tau)} T_{\text{out},j}(\tau). \tag{3.7}$$

For all $i, j \in \{1, \ldots, M\}$, we define the coefficient

$$\psi_{i,j}(\tau) \triangleq \frac{\gamma_{i,j}(\tau)\mathsf{f}_j(\tau)}{\mathsf{f}_i(\tau)} \tag{3.8}$$

and (3.7) can now be rewritten as

$$T_{\text{in},i}(\tau) = \sum_{j=1}^{M} \gamma_{i,j}(\tau) T_{\text{out},j}(\tau). \tag{3.9}$$

The coefficients $\{\psi_{i,j}(\tau)\}$ are functions of the variables $\{\mathsf{f}_i(\tau)\}$ and $\{\gamma_{i,j}(\tau)\}$. Therefore we should denote them as $\Big\{\psi_{i,j}\Big(\mathsf{f}_1(\tau), \dots, \mathsf{f}_M(\tau), \gamma_{1,1}(\tau), \dots, \gamma_{M,M}(\tau)\Big)\Big\}$. In order to reduce the complexity of the notation, we denote the coefficients $\{\psi_{i,j}(\tau)\}$ as if they had direct dependency over the time variable τ. It is meant however, that at any time τ, the values of the coefficients $\{\psi_{i,j}(\tau)\}$ are set by (3.8).

We define the $M \times 1$ vectors

$$\boldsymbol{T}_{\text{in}}(\tau) \triangleq \Big[\underbrace{T_{\text{in},1}(\tau) \quad \dots \quad T_{\text{in},N}(\tau)}_{\text{Server nodes}} \; \underbrace{T_{\text{in},N+1}(\tau) \quad \dots \quad T_{\text{in},N+C}(\tau)}_{\text{CRAC nodes}} \; \underbrace{T_{\text{in},N+C+1}(\tau) \quad \dots \quad T_{\text{in},M}(\tau)}_{\text{Environment nodes}} \Big]^T,$$

$$\boldsymbol{T}_{\text{out}}(\tau) \triangleq \Big[\underbrace{T_{\text{out},1}(\tau) \quad \dots \quad T_{\text{out},N}(\tau)}_{\text{Server nodes}} \; \underbrace{T_{\text{out},N+1}(\tau) \quad \dots \quad T_{\text{out},N+C}(\tau)}_{\text{CRAC nodes}} \; \underbrace{T_{\text{out},N+C+1}(\tau) \quad \dots \quad T_{\text{out},M}(\tau)}_{\text{Environment nodes}} \Big]^T,$$

$$\mathbf{f}(\tau) \triangleq \Big[\underbrace{\mathsf{f}_1(\tau) \quad \dots \quad \mathsf{f}_N(\tau)}_{\text{Server nodes}} \; \underbrace{\mathsf{f}_{N+1}(\tau) \quad \dots \quad \mathsf{f}_{N+C}(\tau)}_{\text{CRAC nodes}} \; \underbrace{\mathsf{f}_{N+C+1}(\tau) \quad \dots \quad \mathsf{f}_M(\tau)}_{\text{Environment nodes}} \Big]^T,$$

$$\mathsf{f}^{-1}(\tau) \triangleq \Big[\underbrace{\frac{1}{\mathsf{f}_1(\tau)} \;\cdots\; \frac{1}{\mathsf{f}_N(\tau)}}_{\text{Server nodes}} \; \underbrace{\frac{1}{\mathsf{f}_{N+1}(\tau)} \;\cdots\; \frac{1}{\mathsf{f}_{N+C}(\tau)}}_{\text{CRAC nodes}} \; \underbrace{\frac{1}{\mathsf{f}_{N+C+1}(\tau)} \;\cdots\; \frac{1}{\mathsf{f}_M(\tau)}}_{\text{Environment nodes}} \Big]^T,$$

and the matrices

$$\Gamma(\tau) = \begin{bmatrix} \gamma_{1,1}(\tau) & \dots & \gamma_{1,M}(\tau) \\ & & \\ \gamma_{M,1}(\tau) & \dots & \gamma_{M,M}(\tau) \end{bmatrix}, \qquad \Psi(\tau) = \begin{bmatrix} \psi_{1,1}(\tau) & \dots & \psi_{1,M}(\tau) \\ & & \\ \psi_{M,1}(\tau) & \dots & \psi_{M,M}(\tau) \end{bmatrix}.$$

The vector $\boldsymbol{T}_{\text{in}}(\tau)$ is the *input temperature vector*, the vector $\boldsymbol{T}_{\text{out}}(\tau)$ is the *output temperature vector*, and the vector $\mathsf{f}(\tau)$ is the *airflow vector*. Eqs. (3.3) and (3.7) can now be written as

$$\mathsf{f}(\tau) = \Gamma(\tau)\mathsf{f}(\tau) \tag{3.10}$$

and

$$\boldsymbol{T}_{\text{in}}(\tau) = \Psi(\tau)\boldsymbol{T}_{\text{out}}(\tau). \tag{3.11}$$

We now want to rewrite (3.8) using vectors. Toward this goal, we use the operator $diag\{\cdot\}$. The operator $diag\{\cdot\}$ transforms the generic vector $\boldsymbol{x}$ into a diagonal matrix, with the elements along the main diagonal equal to the elements of the vector $\boldsymbol{x}$. Eq. (3.8) can now be written as

$$\Psi(\tau) = diag\{\mathsf{f}^{-1}(\tau)\}\Gamma(\tau)diag\{\mathsf{f}(\tau)\}. \tag{3.12}$$

Eqs.(3.10) and (3.11) describe the heat exchange among the thermal server nodes. In this model, the links connecting the thermal server nodes are represented by the structure of the matrix $\Gamma(\tau)$, i.e., by the positions of the nonzero elements of the matrix. According to (3.11), the relationship between the input temperature of a node and all of the other output temperature is linear as long as the matrix $\Gamma(\tau)$ and the vector $\mathsf{f}(\tau)$ are constant.

If the matrix $\Gamma(\tau)$ and the vector $\mathbf{f}(\tau)$ are time-varying, then the relationship between the input and output temperature vectors is a nonlinear function of the vector $\mathbf{f}(\tau)$ and (3.12) has to be considered.

At any time τ, the vector $\mathbf{f}(\tau)$ is a right eigenvector of the matrix $\Gamma(\tau)$ associated with the eigenvalue 1. Therefore, variations of the airflow of the i^{th} thermal node may affect both the way heat is exchanged in the data center (changes in the $\Gamma(\tau)$ matrix) and the rate at which air moves through the physical devices (changes in the whole $\mathbf{f}(\tau)$ vector). Once the matrix $\Gamma(\tau)$ or the vector $\mathbf{f}(\tau)$ have changed their values, it may take a certain amount of time before a new equilibrium point, given by (3.10), is found. In this dissertation we neglect this transient and we assume that (3.10) holds for every $\tau \in \mathbb{R}$. Appendix A.1 discusses some notable properties of the matrices $\Gamma(\tau)$ and $\Psi(\tau)$.

3.3.2 Dynamics of thermal server nodes

This subsection discusses how multiple elements, typically considered for modeling the thermal dynamic of servers and their power consumption, can be described in the proposed framework. The section only focuses on the elements that will be later used in the dissertation. The elements are

- the temperature of the air at the inlet and at the outlet of the server;
- the rate at which air flows into the server;
- the state of the server, e.g., the ON or OFF state of the server;
- the power consumption of the server;
- the amount of computing resources used in the server.

The state of the n^{th} server at time τ is denoted with $\varphi_n(\tau)$. The set of states of the n^{th} server is denoted with $\mathcal{S}_n$ which we assume is composed by a finite number of state variables. The values of the states of all the servers is described by the $N \times 1$ vector $\boldsymbol{\varphi}(\tau)$

defined as

$$\boldsymbol{\varphi}(\tau) \triangleq \begin{bmatrix} \varphi_1(\tau) & \dots & \varphi_N(\tau) \end{bmatrix}^T.$$

The power consumption of the n^{th} server at time τ is denoted with $\mathsf{p}_n(\tau)$. Computing resources are divided into R classes and we denote with the $R \times 1$ vector $\boldsymbol{\rho}_n(\tau)$ the amount of computing resources used in the n^{th} server at time τ.

This subsection focuses on the case where the state of every server is a control variable and the time required to move from one state to another is negligible. Section 3.6 discusses how these additional assumptions can be lifted at the cost, however, of a more complex model of the server. We focus on the case where the state of the server bounds the maximum amount of computing resources available on the server. For example, a turned off server has no computing resources available for computation. With $\overline{\boldsymbol{r}}(n, \varphi_n(\tau))$ we denote the maximum amount of computing resources that the n^{th} server can use at time τ. At any time τ, the amount of computing resources used in the n^{th} server are bounded by

$$\mathbf{0} \leq \boldsymbol{\rho}_n(\tau) \leq \overline{\boldsymbol{r}}(n, \varphi_n(\tau)). \tag{3.13}$$

Specific models for the thermal dynamics of the thermal server node will be discussed in the next chapter. Here we retain a generic form of the dynamics of the thermal server nodes. A generic equation for the dynamics of the n^{th} sever can be written as

$$F_S(n, T_{\text{out},n}(\tau), T_{\text{in},n}(\tau), \mathsf{f}_n(\tau), \mathsf{p}_n(\tau), \varphi_n(\tau)) = 0, \tag{3.14}$$

where the function F_S can consider the time derivatives of the variables $T_{\text{out},n}(\tau)$, $T_{\text{in},n}(\tau)$, $\mathsf{f}_n(\tau)$, and $\mathsf{p}_n(\tau)$.

Measurements taken on a server in our laboratory and discussed in [49] show that a linear time-invariant (LTI) system is a good approximation of the evolution of the outlet temperature of a server, when the fans of the server rotate at constant speed. Figure 3.3

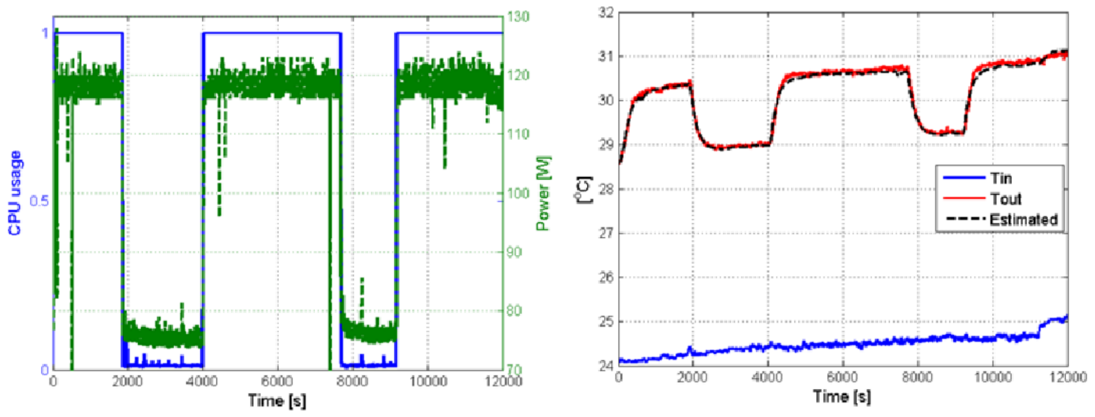

Figure 3.3: CPU utilization, power consumption, inlet and outlet temperature of a desktop computer, when fans rotate at constant speed.

shows the CPU utilization, power consumption, inlet, and outlet temperature of one of the desktop computer used for the experiment. In the following chapter, an LTI model is used to describe the thermal dynamics of a group of servers. Appendix A.3 discusses a model for the server thermal dynamics which also accounts for the temperature of the CPU.

The power consumption of the n^{th} server at time τ can be written as

$$\mathsf{p}_n(\tau) = f_{p,S}(n, \boldsymbol{\rho}_n(\tau), \varphi_n(\tau)), \tag{3.15}$$

where the subscript "p, S" specifies that the generic function f is related to the power consumption of the server nodes. Eq. (3.15) assumes that the state of the server and the amount of computing resources used in the server affect the server dynamics only via the power consumption. When necessary, other elements can be considered in the equation modeling the dynamics of the server. Appendix A.4 discusses a model of server power consumption which considers the amount of computing resources used in the server; the model focuses on the amount of CPU usage in the server.

3.3.3 Dynamics of CRAC nodes

We consider the case where a local controller is embedded in every CRAC unit. Every embedded controller has as external input, which we call *reference temperature.* With $T_{\text{ref},c}(\tau)$ we denote the reference temperature of the c^{th} CRAC unit at time τ and we define the $C \times 1$ vector

$$\boldsymbol{T}_{\text{ref}}(\tau) \triangleq \begin{bmatrix} T_{\text{ref},1}(\tau) & \dots & T_{\text{ref},C}(\tau) \end{bmatrix}^T .$$

It is common practice in industry to call the temperature of the air flowing into a CRAC unit the *return air temperature* and to call the air flowing out from a CRAC unit the *supplied air temperature.* In this dissertation, we prefer to be consistent with the terminology already introduced, so we call input temperature the return air temperature and output temperature the supplied air temperature. The input and the output temperatures of the c^{th} CRAC unit at time τ are denoted with $T_{\text{in},N+c}(\tau)$ and $T_{\text{out},N+c}(\tau)$ respectively.

Even though some CRAC units are able to increase the input air temperature, heating air in data center is rarely a necessity and will not be considered here. At every time τ the embedded controller of the c^{th} CRAC unit manages the internal component of a CRAC unit so that the output temperature of the CRAC units tends to the minimum between the reference temperature and the input temperature of the CRAC unit. A generic equation for the dynamic of the c^{th} CRAC unit dynamics can be written as

$$F_C(c, T_{\text{out},N+c}(\tau), T_{\text{in},N+c}(\tau), T_{\text{ref},c}(\tau), \mathsf{f}_{N+c}(\tau)) = 0, \tag{3.16}$$

where the function F_C can consider the time derivatives of the variables $T_{\text{out},n}(\tau)$, $T_{\text{in},n}(\tau)$, and $\mathsf{f}_n(\tau)$.

Besides controlling the output temperature, CRAC units also control the relative humidity of the output air. The current bounds on the minimum and on the maximum humidity of inlet air, set by ASHARE in 2008 and shown in Tab.1.2, are often enforced

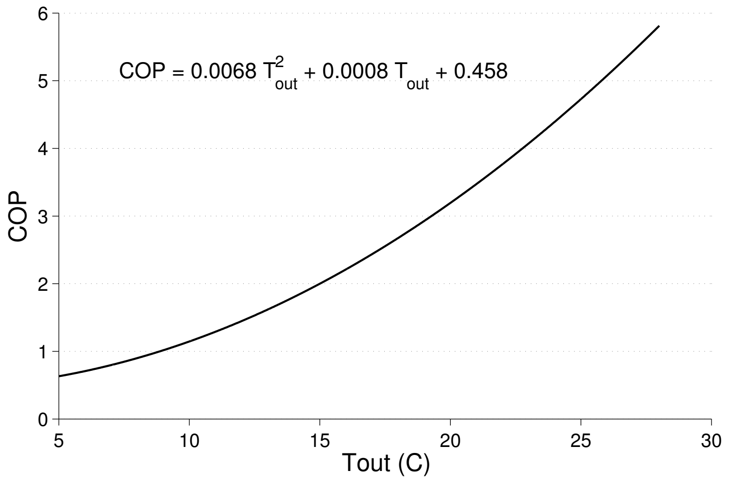

Figure 3.4: Coefficient of performance (COP) of a CRAC unit for different values of its output temperature. Data from [44].

without having to add or remove moisture from the air. Also, bounds on the relative humidity of the air can be mapped into bounds on the inlet air temperature when the absolute amount of humidity is constant. Therefore, the bounds and the control of the humidity are not explicitly considered in this dissertation.

The power consumption of a CRAC unit may depend upon multiple factors, such as the speed at which fans rotate, how compressors (if any) are used, and the specific technology used for the heat exchangers located inside the CRAC unit. A parameter often used to approximate and to predict the power consumption of a CRAC unit is the *coefficient of performance* (COP). The COP of a CRAC unit represents the efficiency at which a CRAC unit removes heat from the air and it is function of the CRAC unit's output temperature. Figure 3.4 shows an example of the relationship between the COP and the output temperature of a CRAC unit.

Assume that at time τ the input temperature of the c^{th} CRAC unit is greater than the output temperature of the unit, i.e., $T_{\text{in},N+c}(\tau) \geq T_{\text{out},N+c}(\tau)$. The rate at which heat is removed by the CRAC unit is

$$\Big(T_{\text{in},N+c}(\tau) - T_{\text{out},N+c}(\tau)\Big)\mathsf{f}_{N+c}(\tau)c_p. \tag{3.17}$$

According to the model proposed by Moore *et al.* [44] and, assuming that $T_{\text{in},N+c}(\tau) \geq T_{\text{out},N+c}(\tau)$, the power consumption of the c^{th} CRAC unit can be approximated by

$$\mathsf{p}_{N+c}(\tau) = \frac{\Big(T_{\text{in},N+c}(\tau) - T_{\text{out},N+c}(\tau)\Big)\mathsf{f}_{N+c}(\tau)c_p}{COP_c(T_{\text{out},c}(\tau))}. \tag{3.18}$$

Eq. (3.18) holds only when $T_{\text{in},N+c}(\tau) \geq T_{\text{out},N+c}(\tau)$.

To model the power consumption of a CRAC unit at every time τ, we consider a generic equation having the form of

$$\mathsf{p}_{N+c}(\tau) = f_{p,C}\Big(c, T_{\text{in},N+c}(\tau), T_{\text{out},N+c}(\tau), COP(T_{\text{out},N+c}(\tau))\Big). \tag{3.19}$$

3.3.4 Dynamics of environment nodes

Environment nodes are divided into two groups. Environment nodes in the first group represent those devices that consume power and for which meaningful input and output temperatures can be defined. Environment nodes in this group are described by

$$\begin{cases} F_{E_1,1}(e, T_{\text{out},N+C+e}(\tau), T_{\text{in},N+C+e}(\tau), \mathsf{f}_{N+C+e}(\tau), \mathsf{p}_{N+C+e}(\tau)) = 0 \\ \mathsf{p}_{N+C+e}(\tau) = f_{E_1,2}(e, \tau). \end{cases} \tag{3.20}$$

The model in (3.20) is similar to the model used for representing the dynamics of thermal server nodes. Environment nodes in the second group are used for those entities that are better described as pure heat sources, e.g., the external environment. Environment nodes in the second group are described by

$$\begin{cases} T_{\text{out},N+C+e}(\tau) & = & f_{E_2,1}(e,\tau) \\ \mathsf{p}_{N+C+e}(\tau) & = & 0. \end{cases} \tag{3.21}$$

Models in (3.20) and in (3.21) are based on a minimalistic representation of the environment nodes. However, due to the large variability of devices that can be included in the environment nodes we expect that, in general, multiple additional parameters will be considered when modeling the power consumption of environment nodes or their thermal evolution.

The number of environment nodes in the first group is E_1 and the number of environment nodes in the second group is E_2. We assume that environment nodes are ordered so that the first E_1 nodes are in the first group and the other nodes are in the second group. We also define the indexes sets $\mathcal{E}_1 = \{N+C+1,\ldots,N+C+E_1\}$ and $\mathcal{E}_2 = \{N+C+E_1+1,\ldots,N+C+E\}$. Indexes in $\mathcal{E}_1$ represent environment nodes in the first group and indexes in $\mathcal{E}_2$ represent environment nodes in the second group.

The input temperatures of the environment nodes having indexes in $\mathcal{E}_2$ are not relevant for control purposes and may be neglected. As a consequence, the time-varying coefficients $\{\gamma_{i,j}\}$ related to these input temperatures do not need to be estimated or known.

3.3.5 Modeling data exchange in the computational network

The data exchange among computational nodes is modeled as atomic computational requests that we call *jobs* and the workload is a *stream of jobs*. Jobs are divided among J classes and jobs in the same class have the same resource requirements. Job classes are used to represent different kind of user requests. For example, HTTP requests sent to a certain server may be collected into a class different from user requests sent to an FTP server. When necessary, it is possible to represent the variability of resource requirements among jobs in the same class, at the cost, however, of a more complex model and of a more complex notation.

A data center may have multiple entry points for jobs and multiple schedulers which manage the scheduling of jobs among the VMs. We represent the whole scheduling action as if it is accomplished by a single scheduler. Every job is assigned, at most, to one VM. Jobs that are not assigned to any VMs are not executed in the data center. We call such jobs *dropped jobs*. With $s_v^j(\tau)$ we denote the relative amount of jobs in class j that we want to schedule on the v^{th} VM at time τ. Let h be a positive real number. With $a^{W,j}(\tau, \tau+h)$ we denote the number of jobs in class j arrived to the data center over the interval $[\tau, \tau+h)$. With $a_v^j(\tau, \tau+h)$ we denote the relative amount of jobs in class j scheduled to be executed in the v^{th} VM over the interval $[\tau, \tau+h)$. Consider the case where the variables $\{s_v^j(\tau)\}$ are constant over the interval $[\tau, \tau+h)$. For all $v \in \mathcal{V}$ and $j \in \{1, \dots, J\}$, the scheduler operates so that

$$\lim_{a^{W,j}(\tau,\tau+h)\to+\infty} \frac{a_v^j(\tau, \tau+h)}{a^{W,j}(\tau, \tau+h)} = s_v^j(\tau). \tag{3.22}$$

Eq. (3.22) means that the relative amount of jobs in class j assigned to the v^{th} VM tends to $s_v^j(\tau)$ as the total number of jobs arrived at the data center ($a^{W,j}(\tau, \tau+h)$) increases. For all $v \in \mathcal{V}$ and $j \in \{1, \dots, J\}$, the product

$$s_v^j(\tau) a^{W,j}(\tau, \tau+h) \tag{3.23}$$

represents the number of jobs in class j that are required to be assigned to the v^{th} VM over the time $[\tau, \tau+h)$. The difference between $a_v^j(\tau, \tau+h)$ and $s_v^j(\tau)a^{W,j}(\tau, \tau+h)$ is due to the policy chosen by the scheduler in the allocation of the jobs among the VMs.

Example 1. Consider a computational network composed by 4 nodes and the number of job classes is 1. The scheduler is set to distribute jobs uniformly among the 4 nodes, i.e., $\boldsymbol{s}(\tau) = \begin{bmatrix} \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \end{bmatrix}^T$ for all $\tau \in \mathbb{R}$. Assume that over the interval $[\tau, \tau+h)$ 3 jobs arrive at the data center. The required number of jobs to be allocated to the computational nodes is $\frac{3}{4}$. The actual number of jobs allocated to 3 nodes will be 1 and one of the node will receive no jobs. Which of the four nodes will receive no jobs depends on the choices made by the scheduler. $\Diamond$

We define the $J \times 1$ vector

$$\boldsymbol{a}^W(\tau, \tau+h) \triangleq \begin{bmatrix} a^{W,1}(\tau, \tau+h) & \ldots & a^{W,J}(\tau, \tau+h) \end{bmatrix}^T$$

and the $(V \cdot J) \times 1$ vectors

$$\boldsymbol{a}(\tau, \tau+h) \triangleq \begin{bmatrix} a_1^1(\tau, \tau+h) & \ldots & a_1^J(\tau, \tau+h) & a_2^1(\tau, \tau+h) & \ldots & \ldots & a_V^J(\tau, \tau+h) \end{bmatrix}^T,$$

$$\boldsymbol{s}(\tau) \triangleq \begin{bmatrix} s_1^1(\tau) & \ldots & s_1^J(\tau) & s_2^1(\tau) & \ldots & \ldots & s_V^J(\tau) \end{bmatrix}^T.$$

The vector $\boldsymbol{a}^W(\tau, \tau+h)$ represents the number of jobs, divided per job class, arriving at the data center over the interval $[\tau, \tau+h)$. The vector $\boldsymbol{a}(\tau, \tau+h)$ represents the number of jobs, divided per job class, arrived at every computational node over the interval $[\tau, \tau+h)$.

We now want to rewrite (3.23) using the vectors defined earlier. Toward this goal, we have to align the elements of the vector $\boldsymbol{a}^W(\tau)$ with the elements of the vector $\boldsymbol{s}(\tau)$. We consider then the operator $diag_B\{X; y\}$, which creates a block matrix by placing y copies of the matrix X along the diagonal blocks and define the $(V \cdot J) \times (V \cdot J)$ diagonal matrix $\mathsf{A}^W(\tau, \tau+h)$ as

$$\mathsf{A}^W(\tau, \tau+h) = diag_B\{diag\{\boldsymbol{a}^W(\tau)\}; V\}. \tag{3.24}$$

The matrix $\mathsf{A}^W(\tau, \tau+h)$ can be written as

$$\begin{bmatrix} a^{W,1}(\tau,\tau+h) & & & & & \\ & \ddots & & & & \\ & & a^{W,J}(\tau,\tau+h) & & & \\ & & & a^{W,1}(\tau,\tau+h) & & \\ & & & & \ddots & \\ & & & & & a^{W,J}(\tau,\tau+h) \end{bmatrix}.$$

Eq. (3.23) can now be rewritten as $\mathsf{A}^W(\tau,\tau+h)\boldsymbol{s}(\tau)$. The vector $\mathsf{A}^W(\tau,\tau+h)\boldsymbol{s}(\tau)$ takes value in $\mathbb{R}^{NV}$, whereas the vector $\boldsymbol{a}(\tau,\tau+h)$ takes value in $\mathbb{N}^{NV}$.[3] The difference between the vector $\boldsymbol{a}(\tau,\tau+h)$ and the vector $\mathsf{A}^W(\tau,\tau+h)\boldsymbol{s}(\tau)$ is due to the sum between the vector $\mathsf{A}^W(\tau,\tau+h)\boldsymbol{s}(\tau)$ and a random vector $\tilde{\boldsymbol{a}}(\tau,\tau+h)$. We can write

$$\boldsymbol{a}(\tau,\tau+h) = \tilde{\boldsymbol{a}}(\tau,\tau+h) + \mathsf{A}^W(\tau,\tau+h)\boldsymbol{s}(\tau). \tag{3.25}$$

The vector $\tilde{\boldsymbol{a}}(\tau,\tau+h)$ models the policy of the scheduler on the allocation of jobs among the computational nodes.

3.3.6 Dynamics of computational nodes

Computational nodes are queueing systems. With $l_v^j(\tau)$ we denote the number of jobs in class j in the v^{th} VM at time τ. For all $v \in \mathcal{V}$, $j \in \{1,\ldots,J\}$ and $h > 0$, we have

$$l_v^j(\tau+h) = l_v^j(\tau) + a_v^j(\tau,\tau+h) - d_v^j(\tau,\tau+h), \tag{3.26}$$

where $d_v^j(\tau,\tau+h)$ is the number of jobs in class j that leaves the v^{th} node over the interval $[\tau,\tau+h)$.

[3]With $\mathbb{N}$ we denote the set of natural numbers.

Eq. (3.26) considers the case where all jobs entering a node will eventually leave the node, i.e., the computational nodes do not generate and do not destroy jobs.

For all $v \in \mathcal{V}$ and for all $\tau, h \in \mathbb{R}$ with $h > 0$, we can define the $J \times 1$ vectors

$$\boldsymbol{l}_v(\tau) = \begin{bmatrix} l_v^1(\tau) & \dots & l_v^J(\tau) \end{bmatrix}^T,$$

$$\boldsymbol{a}_v(\tau, \tau + h) = \begin{bmatrix} a_v^1(\tau, \tau + h) & \dots & a_v^J(\tau, \tau + h) \end{bmatrix}^T,$$

and

$$\boldsymbol{d}_v(\tau, \tau + h) = \begin{bmatrix} d_v^1(\tau, \tau + h) & \dots & d_v^J(\tau, \tau + h) \end{bmatrix}^T.$$

Eq. (3.26) can now be written as

$$\boldsymbol{l}_v(\tau + h) = \boldsymbol{l}_v(\tau) + \boldsymbol{a}_v(\tau, \tau + h) - \boldsymbol{d}_v(\tau, \tau + h). \tag{3.27}$$

Eq. (3.27) describes the evolution of all of the queues on the v^{th} vector. The evolution of all the nodes of the computational network is described by

$$\boldsymbol{l}(\tau + h) = \boldsymbol{l}(\tau) + \boldsymbol{a}(\tau, \tau + h) - \boldsymbol{d}(\tau, \tau + h), \tag{3.28}$$

where

$$\boldsymbol{l}(\tau) = \begin{bmatrix} l_1^1(\tau) & \dots & l_1^J(\tau) & l_2^1(\tau) & \dots & \dots & l_V^J(\tau) \end{bmatrix}^T$$

and

$$\boldsymbol{d}(\tau) = \begin{bmatrix} d_1^1(\tau) & \dots & d_1^J(\tau) & d_2^1(\tau) & \dots & \dots & d_V^J(\tau) \end{bmatrix}^T.$$

3.3.7 Coupling between computational and thermal nodes

The coupling between computational and thermal nodes is captured by the amount of computing resources used by the thermal server nodes. The amount of jobs processed by the computational nodes sets the amount of computing resources used by thermal server nodes to which they are linked. The state of a thermal server node sets the maximum amount of resources that can be used by the computational nodes linked to the thermal server nodes.

In a real data center, VMs can be migrated from one server to another. In this model, the migration of VM is modeled via the variables $\{\theta_v(\tau)\}$. The v^{th} VM is associated with the n^{th} server node at time τ if $\theta_v(\tau) = n$. The associations of all the computational nodes are given by the $V \times 1$ vector $\boldsymbol{\theta}(\tau)$ defined as

$$\boldsymbol{\theta}(\tau) \triangleq \begin{bmatrix} \theta_1(\tau) & \dots & \theta_V(\tau) \end{bmatrix}^T .$$

In order to simplify the following notation, we introduce the function $\Theta^{-1}(\cdot,\cdot)$. The function Θ^{-1} has domain in $\mathcal{N} \times \mathbb{R}$ and it takes values in $\mathcal{P}(\mathcal{V})$, where $\mathcal{P}(\mathcal{V})$ is the power set of $\mathcal{V}$.[4] With $\Theta^{-1}(n, \tau)$ we denote the set of indexes of computational nodes associated with the n^{th} thermal server node at time τ.

The amount of resources assigned to the v^{th} VM at time τ is a controllable variable denoted with the $R \times 1$ vector $\boldsymbol{\omega}_v(\tau)$. The number of jobs departing from the v^{th} vector over the interval $[\tau, \tau + h)$ depends on the number of jobs in the VM at time τ, the number of jobs arrived at the VM over the interval $[\tau, \tau+h)$ and the amount of computing resources used by the VM over the same interval. We can write

$$\boldsymbol{d}_v(\tau, \tau + h) = f_{d,VM}(v, \boldsymbol{l}_v(\tau), \boldsymbol{a}_v(\tau, \tau + h), [\boldsymbol{\omega}_v(t)]_{\tau}^{\tau+h}), \tag{3.29}$$

where $[\boldsymbol{\omega}_v(t)]_{\tau}^{\tau+h}$ denotes the amount of computing resources assigned to the v^{th} VM at every time t for $t \in [\tau, \tau + h)$.

The amount of resources actually used by the v^{th} VM at time τ is denoted by the $R \times 1$

[4]Given a set A, the power set $\mathcal{P}(A)$ is the set of all subsets of A.

vector $\boldsymbol{\pi}_v(\tau)$. The total amount of computing resources used on the n^{th} server is

$$\boldsymbol{\rho}_n(\tau) \triangleq \sum_{v \in \Theta^{-1}(n,\tau)} \boldsymbol{\pi}_v(\tau) \tag{3.30}$$

and the following inequalities hold for all $v \in \mathcal{V}$ and $n \in \mathcal{N}$

$$\mathbf{0} \leq \quad \boldsymbol{\rho}_n(\tau) \quad \leq \overline{\boldsymbol{r}}(n, \varphi_n(\tau)), \tag{3.31}$$

$$\mathbf{0} \leq \quad \boldsymbol{\pi}_v(\tau) \quad \leq \boldsymbol{\omega}_n(\tau). \tag{3.32}$$

Eq. (3.31) means that the total amount of computing resources used in a server cannot exceed the maximum amount of resources available on the server. Eq. (3.32) means that VMs cannot use more resources than those allocated to them.

The amount of computing resources used by a VM depends upon multiple elements, some of which cannot be observed, such as the internal state of the software running within the VM. We model the amount of resources used by the v^{th} VM at time τ as a random variable, whose distribution is parametrized by the couple $(\boldsymbol{l}_v(\tau), \boldsymbol{\omega}_v(\tau))$.

3.4 Data center dynamics

Collecting together the models from the previous sections, for all $\tau, h \in \mathbb{R}$ and $h \geq \tau$, the dynamics of the data center is given by

$$
\begin{cases}
F_S(n, T_{\text{out},n}(\tau), T_{\text{in},n}(\tau), \mathsf{f}_n(\tau), \mathsf{p}_n(\tau), \varphi_n(\tau)) = 0 & \text{for } n \in \mathcal{N} \\
F_C(c - N, T_{\text{out},c}(\tau), T_{\text{in},c}(\tau), T_{\text{ref},c-N}(\tau), \mathsf{f}_c(\tau)) = 0 & \text{for } c \in \mathcal{C} \\
F_{E_1,1}(e - N - C, T_{\text{out},e}(\tau), T_{\text{in},e}(\tau), \mathsf{f}_e(\tau), \mathsf{p}_e(\tau)) = 0 & \text{for } e \in \mathcal{E}_1 \\
T_{\text{out},e}(\tau) = f_{E_2,1}(e - N - C, \tau) & \text{for } e \in \mathcal{E}_2 \\
\mathsf{p}_n(\tau) = f_{p,S}(n, \boldsymbol{\rho}_n(\tau), \varphi_n(\tau)) & \text{for } n \in \mathcal{N} \\
\mathsf{p}_c(\tau) = f_{p,C}(c - N, T_{\text{in},c}(\tau), T_{\text{out},c}(\tau), T_{\text{ref},c-N}(\tau), \mathsf{f}_c(\tau)) & \text{for } c \in \mathcal{C} \\
\mathsf{p}_e(\tau) = f_{E_1,2}(e - N - C, T_{\text{out},e}(\tau), T_{\text{in},e}(\tau), \mathsf{f}_e(\tau)) & \text{for } e \in \mathcal{E}_1 \\
\mathsf{p}_e(\tau) = 0 & \text{for } e \in \mathcal{E}_2 \\
\boldsymbol{T}_{\text{in}}(\tau) = \Psi(\tau) \boldsymbol{T}_{\text{out}}(\tau) \\
\Psi(\tau) = diag\{\mathbf{f}^{-1}(\tau)\} \Gamma(\tau) diag\{\mathbf{f}(\tau)\} \\
\boldsymbol{a}(\tau, \tau + h) = \tilde{\boldsymbol{a}}(\tau, \tau + h) + \mathsf{A}^W(\tau, \tau + h) \boldsymbol{s}(\tau) \\
\boldsymbol{l}(\tau, \tau + h) = \boldsymbol{l}(\tau) + \boldsymbol{a}(\tau, \tau + h) - \boldsymbol{d}(\tau, \tau + h) \\
\boldsymbol{d}_v(\tau, \tau + h) = f_{d,VM}(v, \boldsymbol{l}_v(\tau), \boldsymbol{a}_v(\tau, \tau + h), [\boldsymbol{\omega}_v(\tau)]_\tau^{\tau+h}) & \text{for } v \in \mathcal{V} \\
\boldsymbol{\rho}_n(\tau) = \sum_{v \in \Theta^{-1}(n,\tau)} \boldsymbol{\pi}_n(\tau) & \text{for all } n \in \mathcal{N}
\end{cases}
\tag{3.33}
$$

As discussed in Ch.1, the model derived in this section attempts to provide a complete description of the data center dynamics. The model is nonlinear. Even though most of the equations are not specified yet, the relationship between the matrix $\Psi(\tau)$ and the vector $\mathbf{f}(\tau)$ and the matrix $\Gamma(\tau)$ is nonlinear. This also implies that the relationship between the input and the output temperature vector is nonlinear.

The model also captures the stochastic nature of data centers. The stochasticity in the evolution of the data center is captured by the vectors $\boldsymbol{a}^W(\tau, \tau+h)$, $\tilde{\boldsymbol{a}}(\tau, \tau+h)$, and $\{\boldsymbol{\pi}_n(\tau)\}$. The vector $\boldsymbol{a}^W(\tau, \tau+h)$ represents the uncertainty in the number of jobs arriving at the data center over the interval $[\tau, \tau+h)$. The vector $\tilde{\boldsymbol{a}}(\tau, \tau+h)$, models the unknown policy of the scheduler, and the vectors $\{\boldsymbol{\pi}_n(\tau)\}$ model the amount of resources used by the VMs over time. We note that the power consumption of environment nodes in the first group ($\mathbf{p}_{\mathcal{E}_1}(\tau)$) and the output temperatures of the environment nodes in the second group ($\boldsymbol{T}_{\text{out}\mathcal{E}_2}(\tau)$) can also be represented as random variables and hence, they can increase the amount of uncertainty in the evolution of the data center. The model provides a cyber-physical description of a data center, since it describes both the computational and the thermal characteristics of the data center as well as their interaction.

We now divide the variables of the model between input, output, and state variables. The input variables are further divided between controllable input variables and uncontrollable input variables.

- **Input variables**
  - *Controllable*
    - The rate at which air flows into every device, $\mathbf{f}(\tau)$.
    - The state of every server node, $\boldsymbol{\varphi}(\tau)$.
    - The reference temperatures of the CRAC nodes, $\boldsymbol{T}_{\text{ref}}(\tau)$.
    - The relative number of jobs assigned to the v^{th} VM at time τ, $\boldsymbol{s}(\tau)$.
    - The amount of computing resources assigned to the VMs at time τ, $\{\boldsymbol{\omega}_n(\tau)\}$.
    - The association between VMs and server, $\boldsymbol{\theta}(\tau)$.
  - *Uncontrollable*
    - The way air flows mix in the data center, $\Gamma(\tau)$.
    - The power consumption values of the environment nodes modeled by (3.20),

$\mathsf{p}_{\mathcal{E}_1}(\tau)$.

– The output temperatures of the environment nodes modeled by (3.21), $\boldsymbol{T}_{\text{out}\mathcal{E}_2}(\tau)$.

– The number of jobs arriving at the data center over the interval $[\tau, \tau+h)$, $\boldsymbol{a}^W(\tau, \tau+h)$.

– The specific policy of the scheduler in allocating jobs among the VMs, $\tilde{\boldsymbol{a}}(\tau, \tau+h)$.

– The amount of computing resources used by the VMs, $\{\boldsymbol{\pi}_v(\tau)\}$.

- **State variables**
  - The output temperatures of server thermal nodes, CRAC nodes, and environment nodes in the first group, $\boldsymbol{T}_{\text{out}\mathcal{N}\cup\mathcal{C}\cup\mathcal{E}_1}(\tau)$.
  - The number of jobs in every VM, $\boldsymbol{l}(\tau)$.
- **Output variables**
  - The input temperatures of thermal nodes $\boldsymbol{T}_{\text{in}}(\tau)$.
  - The power consumption of server and CRAC nodes, $\{\mathsf{p}_i(\tau)\}$, $i \in \{1, \dots, N+C\}$.
  - The number of jobs departing from VMs over the interval $[\tau, \tau+h)$, $\boldsymbol{d}(\tau, \tau+h)$.

Even though the distributions of the variables $\{\boldsymbol{\pi}_v(\tau)\}$ are function of the state variables and of the input control variables, the value at time τ of the variables $\{\boldsymbol{\pi}_v(\tau)\}$ cannot be chosen by the control algorithm. For this reason, we put the variables $\{\boldsymbol{\pi}_v(\tau)\}$ in the uncontrollable input group.

3.5 Optimal control of data centers

Once the performance metric is chosen and the constraints on the control variables are given, the optimization problem can be formulated. The dynamic evolution of the data center given in (3.33) generates a collection of constraints for the optimal control problem.

Given the state of the system at time τ, the model in (3.33) allows the control algorithms to predict future values of the state vectors and output vectors based on the chosen values for the input variables.

A closed-form solution of the optimization problem may be available for some specific and highly simplified cases, but in general, we expect it will be necessary to use numerical methods to solve the optimal control problem. Independently from the specific chosen performance metric, the optimization problem tackled by the data center controller is non-convex and hence, numerical methods may not be able to guarantee global optimality. Furthermore, since the control variables $\boldsymbol{\varphi}(\tau)$ and $\boldsymbol{\theta}(\tau)$ take values over discrete sets, the optimization problem belongs to the class of a mixed integer nonlinear programs (MINLP).

Receding-horizon (RH) approaches are typically considered when the performance of a nonlinear system has to be maximized. In a RH approach, the control problem is transformed into an optimization problem at each time step. The optimization problem considers future values of the control input and of the state and output of the system under control. The solution of the optimization problem returns a sequence of control inputs. Only the first element of the sequence is applied to the system and, at the next time-step, a new optimization problem is instantiated based on the measured outputs of the system and the procedure is repeated. The most common RH control approach is *model predictive control* (MPC). Some of the controllers discussed in the next chapter are based on the MPC approach. An introduction to MPC can be found in [63], as well as many other references.

The strength of RH approaches resides in their ability to compensate for inaccuracies in the predictive model by sampling the state of the system at the beginning of every control interval. However, the time required to solve the optimization problem has to be much smaller than the length of the controller's sampling time. Even though numerical algorithms have largely improved over the last 20 years, MINLP are still hard problems to solve. Furthermore, the number of variables and the number of constraints that have to be considered in the optimization problem grows linearly with the number of future steps considered by the control algorithm. As discussed in the following example, RH control approaches are not a viable solution for data center control due to the large number of servers that have to be controlled.

Example 2. Consider the case of a data center with 1000 servers, each having two states: ON and OFF. The set of states of the n^{th} server is $\mathcal{S}_n = \{\text{ON}, \text{OFF}\}$. The maximum number of VMs active at any time is 5000, i.e., $V = 5000$. In such a case, the number of discrete variables is 6000, i.e., the vector $\boldsymbol{\theta}(\tau)$ is a $5000\times$ 1 vector and the vector $\boldsymbol{\varphi}(\tau)$ is a 1000×1 vector. Furthermore, the length of the vector $\mathbf{f}(\tau)$ is at least 5000×1, i.e., one air flow per server. The length of the vector $\mathbf{f}(\tau)$ sets the number of nonlinear equations associated with the temperature evolution. Solving an MILNP with 6000 integer variables and more than 1000 nonlinear equality constraints can take weeks even on modern machines executing state-of-the-art algorithms. $\diamondsuit$

There are however structures and properties of the data center models that can be exploited in order to reduce the complexity of the control problem. As the number of servers increases, we expect that associating a VM with one server or with a nearby server has negligible effects on the global performance. Similarly, we expect that as the number of servers increases, the state of each single server becomes irrelevant and only the state of large group of servers becomes important. The following chapter further discusses how control algorithms can leverage these properties.

3.6 Extensions to the model

In subsection 3.3.2 we assume that the state of the thermal server nodes is an input control variable. In general, this assumption does not hold. For example, a turned off server takes a certain amount of time before it is able to process workload. A refined model which considers the time required by a server to move from one state to another is discussed in this section.

Consider the case where an order is defined over the sets $\{\mathcal{S}_n\}$. Since the sets are composed by a finite number of elements, an order can always be defined. The time required by the n^{th} server to migrate from its i^{th} state to its j^{th} state is denoted with $\zeta_n(i,j)$ (s). If the variables $\{\zeta_n(i,j)\}$ are only partially known, then they can be represented as random variables.

With $\kappa_n(\tau)$ we denote the desired server state of the n^{th} thermal server node at time τ. If at time τ, the server node is at the state $\varphi_n(\tau) \neq \kappa_n(\tau)$, then the server will start the transition that will take it to the state $\kappa_n(\tau)$.

In general, from a single state, a server can move to multiple states. For example, if a server has multiple active states, then from the TURNING-ON state, it may move to any of the active states. Furthermore, a server may have states that are reachable only from a subset of other states. For example, the OFF state may be reached only from the TURNING-OFF state. In order to describe the paths allowed from one state to another, we can consider the graph $G_{S,n}(\mathcal{S}_n, E_{S,n})$, where $E_{S,n}$ is the set of directed edges which connect the states of the n^{th} server. An edge connects the i^{th} state with the j^{th} state, where $i, j \in \mathcal{S}_n$, if and only if the n^{th} server is allowed to migrate from the i^{th} state to the j^{th} one.

Furthermore, some of the states of a server may not be stable, i.e., the server can stay on those states only for a limited amount of time. We say that a state is stable if it is not unstable and we denote with $\bar{\mathcal{S}}_n$ the set of stable states of the n^{th} server. In such a case, the variable $\kappa_n(\tau)$ takes value in $\bar{\mathcal{S}}_n$. The following example helps clarifying the notation introduced in this section.

Example 3. Consider the case of a medium-size data center composed by 1008 servers, e.g., the data center is composed by 24 racks, each containing 42 servers. In this example $N = 1008$ and $\mathcal{N} = \{1, \ldots, 1008\}$. All servers can either be on, or off. When a server is off, we say it is in the OFF state and when a server is on, we say it is in the ON state. For

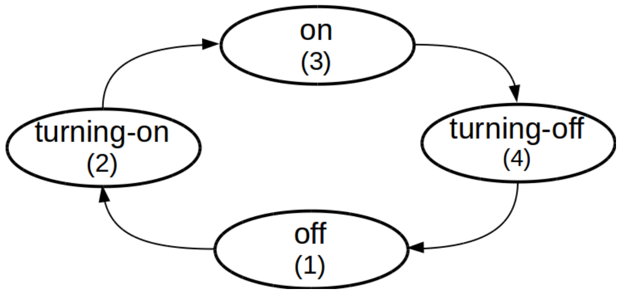

Figure 3.5: Graphical representation of the state and allowed transition among states for the servers in Ex. 3. In brackets we show the chosen order for the states.

all $n \in \mathcal{N}$ the sets of stable states is {OFF, ON}. To move from the OFF state to the ON state, servers require a certain amount of time to initialize their software and hardware. The sequence of operations necessary for a server to move from the OFF state to the ON state is represented by the TURNING-ON state. Servers take also time to move from the ON state to the OFF state. The sequence of operations necessary to a server to move from the ON state to the OFF state is represented by the TURNING-OFF state. For all $n \in \mathcal{N}$ the sets $\mathcal{S}_n$ are {OFF, TURNING-ON, ON, TURNING-OFF}. The sets $\{\mathcal{S}_n\}$ are ordered. The state OFF is always the first state, the state TURNING-ON is the second state, the state ON is the third state, and the state TURNING-OFF is the fourth state. Figure 3.5 provides a graphical representation of $G_{S,n}(\mathcal{S}_n, E_{S,n})$ for all $n \in \{1, \ldots, 1008\}$. From the figure it is possible to derive a mathematical formalization of the set $E_{S,n}$.

Suppose servers 1 to 504 take 1 min to turn on and they take 2 min to turn off and servers 505 to 1008 take 3 min to turn off and they also take 3 min to turn on. Let us focus on server 1. The time required by server 1 to turn on can be modeled, for example, by assuming that the server moves immediately from the OFF state to the TURNING-ON state. The server then takes 60 s to move from the TURNING-ON state to the ON state. Similarly, we can assume that the server takes no time to move from the ON state to the TURNING-OFF state and that it takes 120 s to move from the TURNING-OFF state to the OFF state. A similar approach can be used for the other servers. The migration time of the servers is as follow[5]

For $n \in \{1, \ldots, 504\}$ For $n \in \{505, \ldots, 1008\}$

$$\zeta_n(i,j) = \begin{cases} 0 & \text{if } i = 1, j = 2 \\ 60 & \text{if } i = 2, j = 3 \\ 0 & \text{if } i = 3, j = 4 \\ 120 & \text{if } i = 4, j = 1 \end{cases} \qquad \zeta_n(i,j) = \begin{cases} 0 & \text{if } i = 1, j = 2 \\ 180 & \text{if } i = 2, j = 3 \\ 0 & \text{if } i = 3, j = 4 \\ 180 & \text{if } i = 4, j = 1 \end{cases}$$

In order to guarantee that every server reaches a stable state before a new migration is requested, the variables $\{\kappa_n(\tau)\}$ can be set to be piecewise constant over time-intervals longer than 180 s. $\Diamond$

The data center model discussed in this section is now represented via a hybrid system, where a collection of discrete-dynamic models and continuous-dynamics model interact. The description and the analysis of hybrid systems require notation that is more complex than notation for systems with just continuous dynamics, since the allowed discrete state transitions have to be considered.

The section introduced the notation and the concepts that will be used in Chapter 7. The example shows how adding details to the data center model can drastically increase the complexity of the overall model.

3.7 Summary

In this chapter, we discuss the control problem tackled in the dissertation. A modeling framework is proposed to formalize the data center control problem. The chapter discusses also the development of a data center model and discusses how these assumptions can be lifted at the cost of a more complex model of the data center.

The data center model discussed in this chapter is nonlinear and leads to optimization

[5]Time values are expressed in seconds.

problems which belong to the class of mixed-integer nonlinear programming (MINLP). The complexity of the optimization problem and the large number of optimization variables and nonlinear constraints prevent the use of receding horizon (RH) control techniques. The chapter concludes with an example of how the time required by servers to turn on and off can be included in the data center model.

Chapter 4

Control approach

This chapter discusses the proposed control approach. The proposed hierarchical/distributed architecture takes advantage of the modularity typically found in data centers and it allows the use of optimization algorithms at run-time. The hierarchy is composed of three levels of control and the focus of this dissertation is on controllers for the highest level of the hierarchy.

The chapter is organized as follows. The following section discusses the control architecture we propose. Section 4.2 introduces three control algorithms for the highest level of the control hierarchy. Section 4.3 discusses a continuous-time model of the data center that controllers at the data center level use. The model discussed in Sec. 4.3 stems from the model discussed in the previous chapter, but it adds multiple additional hypothesis which simplify the synthesis of the control law for the data center controllers. Section 4.4 discusses the dynamics of the whole data center in the new continuous-time model. Section 4.5 discusses the existence, under certain additional hypotheses, of a unique stable equilibrium point for the data center. Section 4.6 discusses the predictive discrete-time model used by data center controllers. Section 4.7 discusses the constraints that control variables have to enforce. Finally, Sec. 4.8 formalizes the three control algorithms proposed for the highest level of the hierarchy.

4.1 Proposed control architecture

We propose a hierarchical/distributed control architecture in order to solve, at run-time, the optimization problem discussed in the previous chapter. The proposed hierarchy is composed of three control levels. At the highest level of the hierarchy the focus is on processes having dynamics of the order of tens of minutes or more. The highest level of the hierarchy is called *data center level*. A single controller operates at this level and it directs the lower level controllers. The controller at the data center level is the only controller that defines the set points for the CRAC units, i.e., it defines the values of the reference temperature vector at every time τ. At the second level of the hierarchy the focus is on processes having dynamics of the order of the minutes. The second level of the hierarchy is called *zone level*. Multiple controllers operate at this level and they manage only those controllers and systems related to their zone. The controllers of the group level, discussed in Sec. 2.2, can be thought as controllers at the zone level. The third level of the hierarchy focuses on processes having dynamics of the order of few seconds or less. The third level of the hierarchy is called *intra-zone level* and multiple controllers operate at this level. For example, the CPU governors and the control algorithms discussed in Sec. 2.1 are examples of intra-zone level controllers.

Figure 4.1 provides a graphical representation of the proposed hierarchical control strategy. In the figure, we included a block called *disturbance* which represents the source of the uncontrollable inputs that affect the dynamics of the data center. A *predictor* block is also included in the figure. The predictor block provides estimation about future values of the disturbances to the data center controller. The data center controller may pass data about the predicted disturbance values to the lower level controllers.

The proposed control architecture reduces the complexity of each control algorithm by taking advantage of the modularity typically found in data centers. The sizes of the control problems considered by the control algorithms allow now the use of predictive models. The

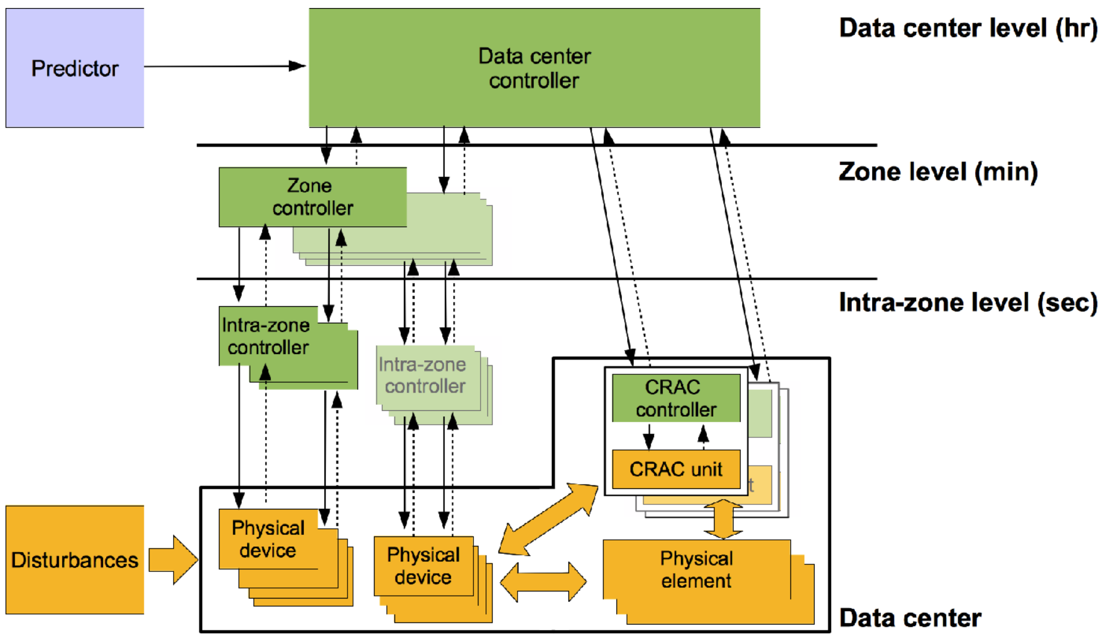

Figure 4.1: Proposed hierarchical and distributed control architecture.

architecture however, also has some drawbacks. For example, the optimality of the overall control action, i.e., the control action generated by the totality of the controllers, is not guaranteed. Also, the use of aggregate models may induce prediction errors and these may lead to poor control performances. The amount of performance loss due to the use of aggregate models depends on both the amount of information lost at the aggregation step and on the way lower level controllers operate. Chapter 8 provides some results about the performance loss, for a specific data center case, when the controller at the data center level uses inaccurate models to predict the behavior of the zone level controllers.

The focus of this dissertation is on controllers at the data center level and on processes ranging in the minutes to hours time scale. In the following, we consider the case where the airflow vector $\mathbf{f}(\tau)$ and the matrix $\Gamma(\tau)$ are constant, i.e., $\mathbf{f}(\tau) = \mathbf{f}$ and $\Gamma(\tau) = \Gamma$ for all $\tau \in \mathbb{R}$. This assumption accounts for multiple issues related to the modeling of air flows in data centers. The way air moves in a data center is described by nonlinear differential equations which take into account, among others, the air pressure variations in the data center. Solving these equations typically requires hours of computations, even on modern machines executing state-of-the-art software. To the best of our knowledge, there are no available algorithms which can model the movements of the air flows with reasonable accuracy and are fast enough to be used for real-time control. A first step toward the development of these algorithms can be found in the work of Toulouse *et al.* [73]. In this dissertation, we prefer to focus on the case where flows of air are constant in the data center and therefore, thermal predictions can be made within few seconds [70]. Under the assumption of constant vector $\mathbf{f}(\tau)$ and of a constant matrix $\Gamma(\tau)$, the matrix $\Psi(\tau)$ is constant and the relationship between the output temperature vector and the input temperature vector is linear. If accurate and fast algorithms able to consider the effects of airflow variations are available, then these can be included in the overall model and considered in the control algorithm.

4.2 Control at the data center level

A controller at the data center level manages lower level controller by fixing set-points for

- the scheduling of jobs among the zone;
- the migration of VMs among the zone;
- the rate at which jobs have to be executed by every zone;
- the reference temperature of every CRAC unit.

The goal of a data center control algorithm is the maximization of the data center performance, subject to the given operational constraints. In order to understand the impact of neglecting the cyber-physical nature of data centers, we study the performance of three different control strategies: *baseline*, *uncoordinated*, and *coordinated*. The control strategies are abstractions of three different control approaches that can be implemented at the data center level.

- **Baseline strategy.** The baseline strategy represents those control algorithms where the set points are fixed and their values are chosen so as to satisfy the control constraints for the worst-case scenario. A baseline controller operates in open-loop and provides indications about the lowest performance we can expect from other control algorithms attempting to maximize the data center performance, either by considering or neglecting the cyber-physical nature of data centers. The baseline control strategy is a reasonable control approach when the power consumed by the CT is much smaller than the total power consumed by IT and when the IT is efficiently managed by lower level controllers.

- **Uncoordinated strategy.** The uncoordinated strategy represents those control approaches where the efficiencies of IT and CT are managed by two independent controllers. An uncoordinated controller neglects the cyber-physical nature of data centers and it provides a performance reference for the coordinated controllers, where the efficiencies of IT and CT are managed by the same algorithm. The uncoordinated control strategy can be typically found in modern data centers. An uncoordinated controller is able to enforce constraints on the IT system, but it may not guarantee the enforcement of the thermal constraints.
- **Coordinated strategy.** The coordinated strategy represents those control approaches where the efficiencies of IT and CT are controlled using a single optimization problem. A coordinated controller takes full advantage of a cyber-physical model of a data center.

The performances of the three controllers are discussed in the next chapter. With respect to the complexity of the control algorithm, a baseline controller is one of the simplest algorithm to implement. No computations are executed by the controller at run-time and a fixed control law is used. An uncoordinated controller is more complex than a baseline controller, but typically, less complex than a coordinated algorithm.

We focus on the case where the uncoordinated and the coordinate controllers are based on a model-predictive-control (MPC) approach. An introduction to MPC can be found, among others, in [63].The coordinated and the uncoordinated controllers use a discrete-time version of the model discussed in the following section.

4.3 Plant model at the data center level

At the data center level, the focus is on processes having dynamics of the order of tens of minutes or more. Therefore, only the mean values of the predicted processes (later specified) are considered. It is left to the controllers at the zone and at the intra-zone level to define strategies to counteract the effects of the variations around the mean values.

At the data center level, the computational nodes represent the computational characteristics of the zones, whereas the thermal nodes represent the physical characteristics of the zones, CRAC units, and uncontrollable elements. Thermal nodes are still divided among three classes: servers, CRAC, and environment nodes. Nodes in the server class represent the thermal characteristics of the zones. CRAC nodes represent the thermal characteristics of CRAC units, and environment nodes represent the thermal characteristics of aggregated, uncontrollable, devices.

Every computational node is associated with a single thermal server node and N denotes both the number of thermal server nodes and the number of computational nodes. The migration of VMs among servers of different zones is represented by the migration of jobs from one zone to another. Due to the use of aggregate models, the number of computational and of thermal nodes managed by a data center controller is largely reduced.

Example 4. Consider the case of a data center composed by 24 racks. Every racks contains 42 servers. Racks are grouped into 8 rows of 3 racks reach. Consider the case where every row is a zone. The data center controller has to manage only 8 zones, whereas the number of servers in the data center is 1008. In this example, every zone-level controller has to manage 126 servers. $\Diamond$

In [51], a simplified version of the data center model presented in this section is discussed.

4.3.1 Thermal model of zones

We use a first-order LTI system to model the evolution of the output temperature of a zone. The model for the i^{th} zone can be written as

$$\dot{T}_{\mathrm{out},i}(\tau) = -\mathsf{k}_i T_{\mathrm{out},i}(\tau) + \mathsf{k}_i T_{\mathrm{in},i}(\tau) + \mathsf{c}_i \mathsf{p}_i(\tau), \tag{4.1}$$

where $\frac{1}{\mathsf{k}_i}$ is the time constant of the temperature of i^{th} node, c_i is the coefficient that maps power consumption into output temperature variation, and $\mathsf{p}_i(\tau)$ is the power consumption of the i^{th} node at time τ. The model in (4.1) specifies the generic model for thermal server nodes considered in (3.15). Appendix A.2 discusses how the generic term c_i is related to the airflow rate of the i^{th} zone and to the time constant of the zone by

$$\mathsf{c}_i = \frac{\mathsf{k}_i}{c_p \mathsf{f}_i},$$

where c_p (J/ (Kg K)) is the specific heat of the air at ambient pressure.

Eq. (4.1) represents the thermal dynamic of the i^{th} zone through a first order LTI system. In general however, higher order models can be considered. We expect the use of black-box modeling or model reduction techniques to derive the values of the coefficients in (4.1). An introduction about system identification techniques can be found in the work of Ljung [39], as well as many other references. A survey of model reduction techniques can be found, among others, in the work of Antoulas *et al.* [8].

In order to derive an equation for the power consumption of the i^{th} zone, we need to introduce two variables that shall be further discussed in Sec. 4.3.5. With $l_i^j(\tau)$ we denote the number of jobs in class j being executed at time τ in the i^{th} zone and with $\eta_i^j(\tau)$, we denote the rate at which jobs in class j leaves the i^{th} zone at time τ. We define the $J \times 1$

vectors

$$\boldsymbol{l}_i(\tau) \triangleq \begin{bmatrix} l_i^1(\tau) & \dots & l_i^J(\tau) \end{bmatrix}^T$$

and

$$\boldsymbol{\eta}_i(\tau) \triangleq \begin{bmatrix} \eta_i^1(\tau) & \dots & \eta_i^J(\tau) \end{bmatrix}^T.$$

Zone controllers are assumed to manage the zones so that the overall power consumption and job execution grow proportionally with each other. We can write

$$\mathsf{p}_i(\tau) = \boldsymbol{\alpha}_i^T \boldsymbol{\eta}_i(\tau) + \boldsymbol{\beta}_i^T \boldsymbol{l}_i(\tau), \qquad \text{for all } i \in \mathcal{N}, \tag{4.2}$$

where $\boldsymbol{\alpha}_i$ and $\boldsymbol{\beta}_i$ are the $J \times 1$ vectors that map the effects of job execution to the zone power consumption. Eq. (4.2) specifies the generic relationship written in (3.15).

When necessary, the power consumption model can be extended to include nonlinear functions which may account, for example, for the ON and OFF state of every server.

4.3.2 Thermal model of CRAC units

As discussed in Sec. 3.3.3, the output temperatures of the CRAC units can be modeled as

$$\dot{T}_{\text{out},i}(\tau) = -\mathsf{k}_i T_{\text{out},i}(\tau) + \mathsf{k}_i \min\{T_{\text{in},i}(\tau), T_{\text{ref},i-N}(\tau)\}, \tag{4.3}$$

where $T_{\text{ref},i}(\tau)$ represents the reference temperature of the CRAC node having index i. The min operator in (4.3) ensures that the output temperature of the CRAC node is always lower than, or equal to, the input temperature of the node. Eq. (4.3) specifies the generic model for the dynamics of CRAC units given in (3.16). According to (3.18), the power consumption of the i^{th} CRAC unit can be written as

$$\mathsf{p}_i(\tau) = \begin{cases} \mathsf{f}_i c_p \dfrac{T_{\text{in},i}(\tau) - T_{\text{out},i}(\tau)}{COP(T_{\text{out},i}(\tau))} & T_{\text{in},i}(\tau) \geq T_{\text{out},i}(\tau) \\ 0 & T_{\text{in},i}(\tau) < T_{\text{out},i}(\tau). \end{cases} \tag{4.4}$$

In (4.4), it is assumed that a CRAC unit consumes no power when it provides no cooling, i.e., when $T_{\text{in},i}(\tau) < T_{\text{out},i}(\tau)$. There are two elements that made us set to zero the power consumption of a CRAC unit when $T_{\text{in},i}(\tau) < T_{\text{out},i}(\tau)$:

- We assume the speed at which fans rotate is constant and hence, we expect the power consumption of the fans is well approximated by a constant.
- We expect the power consumption of a CRAC unit to be continuous over the variables $T_{\text{out},i}(\tau)$ and $T_{\text{in},i}(\tau)$.

In order to maintain the continuity of the power consumption over the variables $T_{\text{out},i}(\tau)$ and $T_{\text{in},i}(\tau)$, any constant used for the case $T_{\text{in},i}(\tau) < T_{\text{out},i}(\tau)$ has to be considered also for the case $T_{\text{in},i}(\tau) \geq T_{\text{out},i}(\tau)$. Since the power consumption of the CRAC units is only accounted in the cost function, any constant added to the power consumed by a CRAC unit can be disregarded at the time of optimizing the data center.

4.3.3 Thermal model of environment nodes

At the data center level, environment nodes may represent single uncontrollable devices, or aggregation of uncontrollable devices. The choice depends on the characteristics of the control problem and on the required accuracy. Environment nodes in the first group are modeled as

$$\dot{T}_{\text{out},i} = -\mathsf{k}_i T_{\text{out},i}(\tau) + \mathsf{k}_i T_{\text{in},i}(\tau) + \mathsf{c}_i \mathsf{p}_i(\tau), \tag{4.5}$$

where $i \in \mathcal{E}_1$. Eq. (4.5) is similar to the equation used to model the evolution of thermal server nodes and it specifies the generic model described in (3.20). The power consumption of the i^{th} environment nodes (of the first group) is an uncontrollable input.

Environment nodes in the second group do not have a dynamic equations and their output temperature is an uncontrollable input which directly affects the dynamics of the overall system. The power consumption of the i^{th} environment node of the second group is set to be 0, i.e., $\mathsf{p}_i(\tau) = 0$ for all $i \in \mathcal{E}_2$ and $\tau \in \mathbb{R}$.

4.3.4 Dynamics of the thermal network

The dynamic evolution of all of the nodes in the thermal network can be written as

$$\begin{cases} \dot{T}_{\text{out},i}(\tau) = -\mathsf{k}_i T_{\text{out},i}(\tau) + \mathsf{k}_i T_{\text{in},i}(\tau) + \mathsf{c}_i \mathsf{p}_i(\tau) & \text{for } i \in \mathcal{N}, \\ \dot{T}_{\text{out},i}(\tau) = -\mathsf{k}_i T_{\text{out},i}(\tau) + \mathsf{k}_i \min\{T_{\text{in},i}(\tau), T_{\text{ref},i-N}(\tau)\} & \text{for } i \in \mathcal{C}, \\ \dot{T}_{\text{out},i}(\tau) = -\mathsf{k}_i T_{\text{out},i}(\tau) + \mathsf{k}_i T_{\text{in},i}(\tau) + \mathsf{c}_i \mathsf{p}_i(\tau) & \text{for } i \in \mathcal{E}_1, \\ T_{\text{in},i}(\tau) = \sum_{j=1}^{M} \psi_{i,j} T_{\text{out},i}(\tau) & \text{for } i \in \mathcal{N} \cup \mathcal{C} \cup \mathcal{E}_1, \\ \mathsf{p}_i(\tau) = \begin{cases} \mathsf{f}_i c_p \dfrac{T_{\text{in},i}(\tau) - T_{\text{out},i}(\tau)}{COP(T_{\text{out},i}(\tau))} & T_{\text{in},i}(\tau) \geq T_{\text{out},i}(\tau) \\ 0 & T_{\text{in},i}(\tau) < T_{\text{out},i}(\tau) \end{cases} & \text{for } i \in \mathcal{C}. \end{cases} \tag{4.6}$$

The state of the thermal network is the vector $\boldsymbol{T}_{\text{out}\mathcal{N}\cup\mathcal{C}\cup\mathcal{E}_1}(\tau)$. Outputs of the thermal network are the vectors $\boldsymbol{T}_{\text{in}\mathcal{N}\cup\mathcal{C}\cup\mathcal{E}_1}(\tau)$ and $\mathbf{p}_{\mathcal{C}}(\tau)$. Inputs of the thermal network are the vectors $\mathbf{p}_{\mathcal{N}}(\tau)$, $\mathbf{p}_{\mathcal{E}_1}(\tau)$, $\boldsymbol{T}_{\text{out}\mathcal{E}_2}(\tau)$, and $\boldsymbol{T}_{\text{ref}}(\tau)$. The only controllable input is the vector is $\boldsymbol{T}_{\text{ref}}(\tau)$. The equations in (4.6) model the evolution of the thermal network and specify the generic relationships given in (3.33). Even though the relationship between the input and the output temperature vector is now linear, the dynamics of the thermal network is still described by a nonlinear system. One of the nonlinearities affects the dynamic evolution of the output temperature of the CRAC units. The other non linearity affects the power consumption of the CRAC units.

The nonlinearity introduced by the min operator in the evolution of the output temperature of the CRAC nodes can be removed if we assume $\boldsymbol{T}_{\text{ref}}(\tau) \leq \boldsymbol{T}_{\text{in},\mathcal{C}}(\tau)$, for all $\tau \in \mathbb{R}$, or if we assume $\boldsymbol{T}_{\text{ref}}(\tau) > \boldsymbol{T}_{\text{in},\mathcal{C}}(\tau)$, for all $\tau \in \mathbb{R}$. We define the $(N + C + E_1) \times 1$ vectors

$$\mathbf{k} \triangleq \begin{bmatrix} \mathsf{k}_1 & \dots & \mathsf{k}_N & \mathsf{k}_{N+1} & \dots & \mathsf{k}_{N+C} & \mathsf{k}_{N+C+1} & \dots & \mathsf{k}_{N+C+E_1} \end{bmatrix}^T$$

and

$$\mathsf{c} \triangleq \begin{bmatrix} \mathsf{c}_1 & \dots & \mathsf{c}_N & 0 & \dots & 0 & \mathsf{c}_{N+C+1} & \dots & \mathsf{c}_{N+C+E_1} \end{bmatrix}^T.$$

We consider at first the case $\boldsymbol{T}_{\text{ref}}(\tau) \leq \boldsymbol{T}_{\text{in},\mathcal{C}}(\tau)$, for all $\tau \in \mathbb{R}$. Under this hypothesis, the state evolution of the thermal network can be written as

$$\dot{\boldsymbol{T}}_{\text{out},\mathcal{N}\cup\mathcal{C}\cup\mathcal{E}_1}(\tau) \;=\; A_{T,ct}\boldsymbol{T}_{\text{out}\mathcal{N}\cup\mathcal{C}\cup\mathcal{E}_1} + B_{1_{T,ct}} \begin{bmatrix} \mathbf{p}_{\mathcal{N}}(\tau) \\ \boldsymbol{T}_{\text{ref}}(\tau) \end{bmatrix} + B_{2_{T,ct}} \begin{bmatrix} \mathbf{p}_{\mathcal{E}}(\tau) \\ \boldsymbol{T}_{\text{out},\mathcal{E}_2}(\tau) \end{bmatrix} \tag{4.7}$$

where

$$A_{T,ct} = diag\{-\mathbf{k}\} + \begin{bmatrix} diag\{\mathbf{k}_{\mathcal{N}}\} & & \\ & diag\{\mathbf{0}\} & \\ & & diag\{\mathbf{k}_{\mathcal{E}_1}\} \end{bmatrix} \Psi_{[\mathcal{N}\cup\mathcal{C}\cup\mathcal{E}_1,\mathcal{N}\cup\mathcal{C}\cup\mathcal{E}_1]},$$

$$B_{1_{T,ct}} = \begin{bmatrix} diag\{\mathsf{c}_{\mathcal{N}}\} & 0 \\ 0 & diag\{\mathbf{k}_{\mathcal{C}}\} \\ 0 & 0 \end{bmatrix}$$

$$B_{2_{T,ct}} = \begin{bmatrix} 0 & diag\{\mathbf{k}_{\mathcal{N}}\}\Psi_{[\mathcal{N},\mathcal{E}_2]} \\ 0 & 0 \\ diag\{\mathsf{c}_{\mathcal{E}_1}\} & diag\{\mathbf{k}_{\mathcal{E}_1}\}\Psi_{[\mathcal{E}_1,\mathcal{E}_2]} \end{bmatrix},$$

With a slight notation abuse, we denoted with 0 the null matrix, i.e., the matrix whose elements are all 0. Note that the power consumption of CRAC units is still described by a nonlinear equation.

Under the hypothesis $\boldsymbol{T}_{\text{ref}}(\tau) \geq \boldsymbol{T}_{\text{in}\mathcal{C}}(\tau)$ for all $\tau \in \mathbb{R}$, the dynamic evolution of the thermal network is linear and the matrices $A_{T,ct}$ and $B_{1_{T,ct}}$ are

$$A_{T,ct} = diag\{-\mathbf{k}\} + \begin{bmatrix} diag\{\mathbf{k}_{\mathcal{N}}\} & & \\ & diag\{\mathbf{k}_{\mathcal{C}}\} & \\ & & diag\{\mathbf{k}_{\mathcal{E}_1}\} \end{bmatrix} \Psi_{[\mathcal{N}\cup\mathcal{C}\cup\mathcal{E}_1,\mathcal{N}\cup\mathcal{C}\cup\mathcal{E}_1]},$$

$$B_{1_{T,ct}} = \begin{bmatrix} diag\{\mathbf{c}_{\mathcal{N}}\} & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix},$$

$$B_{2_{T,ct}} = \begin{bmatrix} 0 & diag\{\mathbf{k}_{\mathcal{N}}\}\Psi_{[\mathcal{N},\mathcal{E}_2]} \\ 0 & 0 \\ diag\{\mathbf{c}_{\mathcal{E}_1}\} & diag\{\mathbf{k}_{\mathcal{E}_1}\}\Psi_{[\mathcal{E}_1,\mathcal{E}_2]} \end{bmatrix}.$$

We focus now on the stability properties of the thermal network. In the rest of the dissertation the case $\boldsymbol{T}_{\text{ref}}(\tau) \leq \boldsymbol{T}_{\text{in},\mathcal{C}}(\tau)$ will result more relevant than the case $\boldsymbol{T}_{\text{ref}}(\tau) > \boldsymbol{T}_{\text{in},\mathcal{C}}(\tau)$ and hence, we only consider the former case in the stability analysis.

According to the notation earlier introduced and considering the index set $\mathcal{Z} = \mathcal{N} \cup \mathcal{E}_1$, we can define the vectors

$$\boldsymbol{T}_{\text{out}\mathcal{Z}} \triangleq \begin{bmatrix} \boldsymbol{T}_{\text{out}\mathcal{N}} \\ \boldsymbol{T}_{\text{out}\mathcal{E}_1} \end{bmatrix}, \ \boldsymbol{T}_{\text{in}\mathcal{Z}} \triangleq \begin{bmatrix} \boldsymbol{T}_{\text{in}\mathcal{N}} \\ \boldsymbol{T}_{\text{in}\mathcal{E}_1} \end{bmatrix}, \ \mathbf{k}_{\mathcal{Z}} \triangleq \begin{bmatrix} \mathbf{k}_{\mathcal{N}} \\ \mathbf{k}_{\mathcal{E}_1} \end{bmatrix}, \ \mathbf{c}_{\mathcal{Z}} \triangleq \begin{bmatrix} \mathbf{c}_{\mathcal{N}} \\ \mathbf{c}_{\mathcal{E}_1} \end{bmatrix}, \ \mathbf{p}_{\mathcal{Z}} \triangleq \begin{bmatrix} \mathbf{p}_{\mathcal{N}} \\ \mathbf{p}_{\mathcal{E}_1} \end{bmatrix}.$$

and matrices

$$\Psi_{[\mathcal{Z},\mathcal{Z}]} \triangleq \begin{bmatrix} \Psi_{[\mathcal{N},\mathcal{N}]} & \Psi_{[\mathcal{N},\mathcal{E}_1]} \\ \Psi_{[\mathcal{E}_1,\mathcal{N}]} & \Psi_{[\mathcal{E}_1,\mathcal{E}_1]} \end{bmatrix}, \quad \Psi_{[\mathcal{Z},\mathcal{C}]} \triangleq \begin{bmatrix} \Psi_{[\mathcal{N},\mathcal{C}]} \\ \Psi_{[\mathcal{E}_1,\mathcal{C}]} \end{bmatrix},$$

$$\Psi_{[\mathcal{Z},\mathcal{E}_2]} \triangleq \begin{bmatrix} \Psi_{[\mathcal{N},\mathcal{E}_2]} \\ \Psi_{[\mathcal{E}_1,\mathcal{E}_2]} \end{bmatrix}, \qquad \Psi_{[\mathcal{C},\mathcal{Z}]} \triangleq \begin{bmatrix} \Psi_{[\mathcal{C},\mathcal{N}]} & \Psi_{[\mathcal{C},\mathcal{E}_1]} \end{bmatrix}.$$

Under the assumptions

- $\boldsymbol{T}_{\text{in}\mathcal{C}}(\tau) \geq \boldsymbol{T}_{\text{ref}}(\tau)$, for all $\tau \in \mathbb{R}$;
- all of the elements on the vector $\mathbf{k}$ are greater than 0;
- all of the elements of the vector $\mathbf{c}$ are nonnegative;
- the sum of the elements of $\Psi_{[\mathcal{Z},\mathcal{Z}]}$ along every row is lower than 1;

then the thermal network has an unique asymptotically stable equilibrium point given by

$$\begin{cases} \boldsymbol{T}_{\text{out}\mathcal{Z}} &= (I-\Psi_{[\mathcal{Z},\mathcal{Z}]})^{-1}\Psi_{[\mathcal{Z},\mathcal{C}]}\boldsymbol{T}_{\text{ref}} + (I-\Psi_{[\mathcal{Z},\mathcal{Z}]})^{-1}diag\{\mathbf{k}_{\mathcal{Z}}\}^{-1}diag\{\mathbf{c}_{\mathcal{Z}}\}\mathbf{p}_{\mathcal{Z}} + \\ & \quad (I-\Psi_{[\mathcal{Z},\mathcal{Z}]})^{-1}\Psi_{[\mathcal{Z},\mathcal{E}_2]}\boldsymbol{T}_{\text{out}\mathcal{E}_2} \\ \boldsymbol{T}_{\text{out}\mathcal{C}} &= \boldsymbol{T}_{\text{ref}}, \end{cases} \tag{4.8}$$

where we neglected the dependency of the variables from time variable τ. The proof of the invertibility of the matrix $(I-\Psi_{[\mathcal{Z},\mathcal{Z}]})^{-1}$ is given in App. A.5. The proof of the stability of the thermal network is given in App. A.6.

Requiring the sum along the rows of the elements of $\Psi_{[\mathcal{Z},\mathcal{Z}]}$ to be lower than 1, corresponds to constrain the input temperatures of all of the thermal server nodes and environment nodes in the first group, to be affected by the output temperature of at least one CRAC node, or by one environment node of the second group.

4.3.5 Computational network

We model the computational network using a fluid approximation of the workload execution and arrival processes, i.e., the workload is represented by job flow rates rather than as discrete jobs. The proposed workload modeling approach represents a first-order approximation of a queuing system. The strength of the proposed approach resides in its simplicity which allows an uncluttered discussion of the model and of the control approach. On the other hand, a certain amount of computational details of a data center are not included in the model. We consider the approximation provided by the proposed model adequate for the goal of the data center controller. However, when additional details of the system are relevant, more refined approximations can be considered [36].

Due to the fluid approximation, the number of jobs in class j arriving at the data center over the interval $[\tau, \tau + j)$ belongs to the set of the real numbers and we can write

$$a^{W,j}(\tau, \tau + h) = \int_{\tau}^{\tau+h} \lambda^{W,j}(t)dt,$$

where $\lambda^{W,j}(\tau)$ is the rate of arrival of jobs in class j at time τ. In the rest of the dissertation we assume that the variables $\{\lambda^{W,j}(\tau)\}$ are continuous. Consider the case where the variables $\{s_i^j(\tau)\}$ are constant. Due to the fluid approximation and to the assumptions about the scheduler behavior given in (3.22), for all $\tau, h \in \mathbb{R}$ and $h \geq 0$, $i \in \mathcal{N}$, and $j \in \{1, \ldots, J\}$, we have

$$a_i^j(\tau, \tau + h) = a^{W,j}(\tau, \tau + h)s_i^j(\tau). \tag{4.9}$$

With respect to (3.25), we observe that the fluid approximation removed the uncertainty related to the policy of the scheduler on the allocation of jobs among the zones, represented by the random vector $\tilde{\boldsymbol{a}}(\tau, \tau + h)$.

The number of jobs in class j arriving at the i^{th} computational node over the interval $[\tau, \tau + j)$ belongs now to the set of the real numbers and we can write

$$a_i^j(\tau, \tau + h) = \int_{\tau}^{\tau+h} \lambda_i^j(t)dt,$$

where $\lambda_i^j(\tau)$ is the rate of arrival of jobs in class j at time τ to the i^{th} computational node. Since (4.9) holds for every $h \geq 0$, we have

$$\lambda_i^j(\tau) = \lambda^{W,j}(\tau)s_i^j(\tau), \quad \text{for all } i \in \mathcal{N}, \text{ and } j \in \{1, \ldots, J\} \tag{4.10}$$

and (4.10) holds almost everywhere. In the rest of the dissertation, we assume that (4.10) holds for every $\tau \in \mathbb{R}$.

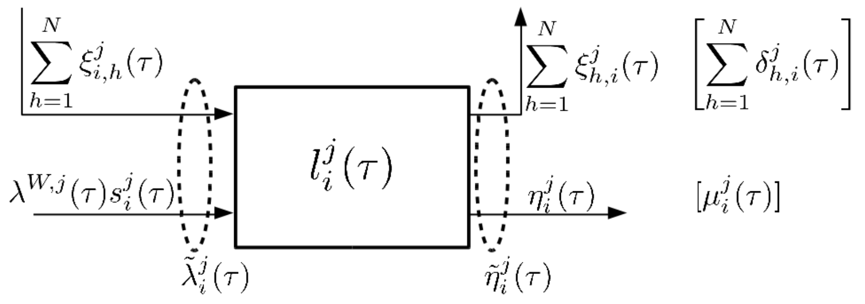
$\sum_{h=1}^{N} \xi_{i,h}^{j}(\tau)$
$\lambda^{W,j}(\tau)s_i^j(\tau)$
$\tilde{\lambda}_i^j(\tau)$
$l_i^j(\tau)$
$\sum_{h=1}^{N} \xi_{h,i}^{j}(\tau)$
$\left[\sum_{h=1}^{N} \delta_{h,i}^{j}(\tau)\right]$
$\eta_i^j(\tau)$
$[\mu_i^j(\tau)]$
$\tilde{\eta}_i^j(\tau)$

Figure 4.2: Components of the input, state, and output variables related to jobs in class j of the i^{th} computational node.

The number of jobs in class j departing from the i^{th} computational node over the interval $[\tau, \tau + j)$ can be written as

$$d_i^j(\tau, \tau + h) = \int_{\tau}^{\tau+h} \eta_i^j(t)dt,$$

where $\eta_i^j(\tau)$ is the rate at which the jobs in class j depart, after being executed, from the i^{th} node at time τ. The migration of VMs from the servers in the i^{th} zone, to servers in the h^{th} zone, is modeled via the migration of jobs from the i^{th} computational node to the h^{th} computational node. With $\xi_{h,i}^j(\tau)$ we denote the rate at which jobs in class j migrates from the i^{th} computational node to the h^{th} computational node at time τ. Figure 4.2 illustrates the variables related to the i^{th} computational node. In the figure, we only show the variables related to jobs in class j.

The total rate at which jobs in class j arrive at the i^{th} node at time τ is

$$\tilde{\lambda}_i^j(\tau) \triangleq \lambda_i^j(\tau) + \sum_{h=1}^{N} \xi_{i,h}^j(\tau). \tag{4.11}$$

The total rate at which jobs in class j leave the i^{th} node at time τ is

$$\tilde{\eta}_i^j(\tau) \triangleq \eta_i^j(\tau) + \sum_{h=1}^{N} \xi_{h,i}^j(\tau). \tag{4.12}$$

For all $j \in \{1, \ldots, J\}$, the evolution of the amount of workload at the i^{th} computational node is given by

$$\dot{l}_i^j(\tau) = \tilde{\lambda}_i^j(\tau) - \tilde{\eta}_i^j(\tau). \tag{4.13}$$

The rate at which the length of the queue of the i^{th} node grows, depends on the difference between the total rate at which jobs arrive at the node and the rate at which jobs depart from the node.

The data center controller generates the set-points for the zone level controllers about the desired rate at which jobs have to be processed. With $\mu_i^j(\tau)$ we denote the desired workload execution rate of jobs in class j at the i^{th} node at time τ and with $\delta_{h,i}^j(\tau)$ we denote the required migration rate of jobs in class j from the i^{th} computational node to the h^{th} computational node at time τ. The variables $\{\mu_i^j(\tau)\}$ and $\{\delta_{h,i}^j(\tau)\}$ are used by the zone-level controllers to decide the state of every thermal server node and the association between server thermal nodes and computational nodes, i.e., the values of the vectors $\boldsymbol{\varphi}(\tau)$ and $\boldsymbol{\theta}(\tau)$.

With $\tilde{\nu}_i^j(t)(\tau)$ we denote the rate at which jobs in class j are required to depart from the i^{th} zone at time τ. The variables $\{\tilde{\nu}_i^j(t)(\tau)\}$ are defined as

$$\tilde{\nu}_i^j(\tau) \triangleq \mu_i^j(\tau) + \sum_{h=1}^{N} \delta_{h,i}^j(\tau), \qquad \text{for all } j \in \{1, \ldots, J\}, i \in \mathcal{N}, \tau \in \mathbb{R}. \tag{4.14}$$

For all $i \in \mathcal{N}$ and $j \in \{1, \ldots, J\}$, we have

$$\eta_i^j(\tau) = \begin{cases} \mu_i^j(\tau) & \text{if } l_i^j(\tau) > 0 \text{ or } \tilde{\lambda}_i^j(t) > \mu_i^j(t) \\ \tilde{\lambda}_i^j(t) & \text{otherwise} \end{cases} \tag{4.15}$$

$$\xi_{h,i}^j(\tau) = \begin{cases} \delta_{h,i}^j(\tau) & \text{if } l_i^j(\tau) > 0 \text{ or } \tilde{\lambda}_i^j(t) > \tilde{\nu}_i^j(t) \\ \frac{\delta_{h,i}^j(\tau)}{\sum\limits_{h=1}^{N} \delta_{h,i}^j(\tau)} (\tilde{\lambda}_i^j(\tau) - \eta_i^j(\tau)) & \text{otherwise} \end{cases} . \tag{4.16}$$

Eq. (4.16) considers the case where every computational node processes as many jobs as possible. When the desired departure rate is higher than the rate at which jobs enter the node, then no jobs are sent to other computational nodes. This specific policy for the computational node simplifies the model used by the controller to predict the rate at which jobs are processed by every node. When necessary however, other policies may be considered. The proposed model for describing the job execution rates in a zone is sufficient for our purposes, but we note that it can be extended to include different migration policies, workload classes, hardware requirements, and interactions among different types of workloads [48, 50, 52].

4.3.6 Dynamics of the computational network

We now want to derive a compact form of the equations discussed in this subsection using vectors. The nonlinear relationships in (4.15) and (4.16) cannot be written in a simple vectorized form and we shall keep them in their current forms.

We define the $J \times 1$ vector of job arrival rate at the data center

$$\boldsymbol{\lambda}^W(\tau) \triangleq \begin{bmatrix} \lambda^{W,1}(\tau) & \ldots & \lambda^{W,J}(\tau) \end{bmatrix}^T$$

and the $(N \cdot J) \times 1$ vectors

$$\boldsymbol{s}(\tau) \triangleq \begin{bmatrix} s_1^1(\tau) & \dots & s_1^J(\tau) & s_2^1(\tau) & \dots & \dots & s_N^J(\tau) \end{bmatrix}^T.$$

$$\boldsymbol{\lambda}(\tau) \triangleq \begin{bmatrix} \lambda_1^1(\tau) & \dots & \lambda_1^J(\tau) & \lambda_2^1(\tau) & \dots & \dots & \lambda_N^J(\tau) \end{bmatrix}^T.$$

The vector $\boldsymbol{s}(\tau)$ is the vector of relative job allocation among the zones and $\boldsymbol{\lambda}(\tau)$ is the vector of job arrival rate to each of the zone. Note that $\boldsymbol{\lambda}(\tau)$ does not consider the exchange of jobs among the zones.

Using the operators $diag\{\cdot\}$ and $diag_B\{\cdot;\cdot\}$, already introduced in Sec. 3.3.5, we can transform the vector $\boldsymbol{\lambda}^W(\tau)$ into the matrix $\Lambda(\tau)$ defined as $\Lambda(\tau) \triangleq diag_B\{diag\{\boldsymbol{\lambda}^W(\tau)\}; N\}$. Eq. (4.10) can now be written as

$$\boldsymbol{\lambda}(\tau) = \Lambda^W(\tau)\boldsymbol{s}(\tau).$$

Let us define the $(N \cdot J) \times 1$ vectors

$$\tilde{\boldsymbol{\lambda}}(\tau) \triangleq \begin{bmatrix} \tilde{\lambda}_1^1(\tau) & \dots & \tilde{\lambda}_1^J(\tau) & \tilde{\lambda}_2^1(\tau) & \dots & \dots & \tilde{\lambda}_N^J(\tau) \end{bmatrix}^T,$$

$$\boldsymbol{\eta}(\tau) \triangleq \begin{bmatrix} \eta_1^1(\tau) & \dots & \eta_1^J(\tau) & \eta_2^1(\tau) & \dots & \dots & \eta_N^J(\tau) \end{bmatrix}^T,$$

$$\tilde{\boldsymbol{\eta}}(\tau) \triangleq \begin{bmatrix} \tilde{\eta}_1^1(\tau) & \dots & \tilde{\eta}_1^J(\tau) & \tilde{\eta}_2^1(\tau) & \dots & \dots & \tilde{\eta}_N^J(\tau) \end{bmatrix}^T,$$

$$\tilde{\boldsymbol{\nu}}(\tau) \triangleq \begin{bmatrix} \tilde{\nu}_1^1(\tau) & \dots & \tilde{\nu}_1^J(\tau) & \tilde{\nu}_2^1(\tau) & \dots & \dots & \tilde{\nu}_N^J(\tau) \end{bmatrix}^T,$$

and

$$\boldsymbol{l}(\tau) = \begin{bmatrix} l_1^1(\tau) & \dots & l_1^J(\tau) & l_2^1(\tau) & \dots & \dots & l_N^J(\tau) \end{bmatrix}^T.$$

We also define the $(N^2 \cdot J) \times 1$ vector

$$\boldsymbol{\xi}(\tau) \triangleq \begin{bmatrix} \xi^1_{1,1}(\tau) & \xi^1_{1,2}(\tau) & \dots & \xi^J_{1,N}(\tau) & \xi^1_{2,1}(\tau) & \dots & \dots & \xi^J_{N,N}(\tau) \end{bmatrix}^T .$$

$$\boldsymbol{\delta}(\tau) \triangleq \begin{bmatrix} \delta^1_{1,1}(\tau) & \delta^1_{1,2}(\tau) & \dots & \delta^J_{1,N}(\tau) & \delta^1_{2,1}(\tau) & \dots & \dots & \delta^J_{N,N}(\tau) \end{bmatrix}^T .$$

Using two matrices B_λ and B_η, which consider the correct sets of elements of the vectors $\boldsymbol{\xi}(\tau)$ and $\boldsymbol{\delta}(\tau)$, we can rewrite (4.11), (4.12), and (4.14) as

$$\begin{aligned} \tilde{\boldsymbol{\lambda}}(\tau) &= \boldsymbol{\lambda}(\tau) + B_\lambda \boldsymbol{\xi}(\tau) & (4.17) \\ \tilde{\boldsymbol{\eta}}(\tau) &= \boldsymbol{\eta}(\tau) + B_\eta \boldsymbol{\xi}(\tau) & (4.18) \\ \tilde{\boldsymbol{\nu}}(\tau) &= \boldsymbol{\mu}(\tau) + B_\eta \boldsymbol{\delta}(\tau). & (4.19) \end{aligned}$$

Due to the fluid approximation, we can now describe the evolution of the length of the queues of all of the zones in the data center via the rate at which jobs arrive and depart. Eq. (3.28) can now be rewritten as

$$\dot{\boldsymbol{l}}(\tau) = \tilde{\boldsymbol{\lambda}}(\tau) - \tilde{\boldsymbol{\eta}}(\tau). \qquad (4.20)$$

The dynamic evolution of all of the nodes in the computational network is given by

$$\begin{cases} \boldsymbol{\lambda}(\tau) & = & diag_B\{diag\{\boldsymbol{\lambda}^W(\tau)\}; N\}\boldsymbol{s}(\tau) \\ \tilde{\boldsymbol{\lambda}}(\tau) & = & \boldsymbol{\lambda}(\tau) + B_\lambda \boldsymbol{\xi}(\tau) \\ \tilde{\boldsymbol{\eta}}(\tau) & = & \boldsymbol{\eta}(\tau) + B_\eta \boldsymbol{\xi}(\tau) \\ \tilde{\boldsymbol{\nu}}(\tau) & = & \boldsymbol{\mu}(\tau) + B_\eta \boldsymbol{\delta}(\tau) \\ \dot{\boldsymbol{l}}(\tau) & = & \tilde{\boldsymbol{\lambda}}(\tau) - \tilde{\boldsymbol{\eta}}(\tau) \\ & & \text{For all } i \in \mathcal{N},\ j \in \{1, \dots, J\} \\ \eta_i^j(\tau) & = & \begin{cases} \mu_i^j(\tau) & \text{if } l_i^j(\tau) > 0 \text{ or } \tilde{\lambda}_i^j(t) > \mu_i^j(t) \\ \tilde{\lambda}_i^j(t) & \text{otherwise} \end{cases} \\ \xi_{h,i}^j(\tau) & = & \begin{cases} \delta_{h,i}^j(\tau) & \text{if } l_i^j(\tau) > 0 \text{ or } \tilde{\lambda}_i^j(t) > \tilde{\nu}_i^j(t) \\ \dfrac{\delta_{h,i}^j(\tau)}{\sum\limits_{h=1}^{N} \delta_{h,i}^j(\tau)}(\tilde{\lambda}_i^j(\tau) - \eta_i^j(\tau)) & \text{otherwise} \end{cases} \end{cases} . \tag{4.21}$$

The state of the computational network is the vector $\boldsymbol{l}(\tau)$. The outputs of the computational networks are the vectors $\boldsymbol{\xi}(\tau)$ and $\boldsymbol{\eta}(\tau)$. The inputs of the computational networks are the vectors $\boldsymbol{\lambda}^W(\tau)$, $\boldsymbol{s}(\tau)$, $\boldsymbol{\delta}(\tau)$, and $\boldsymbol{\mu}(\tau)$. The controllable inputs are the vectors $\boldsymbol{s}(\tau)$, $\boldsymbol{\delta}(\tau)$, and $\boldsymbol{\mu}(\tau)$. The dynamic of the computational network is nonlinear. The nonlinearity are related to the way the values of the vectors $\boldsymbol{\eta}(\tau)$ and $\boldsymbol{\xi}(\tau)$ are computed.

4.4 Dynamics of the data center

The dynamics of the data center is obtained by considering the dynamics of the thermal network, the dynamics of the computational network, and the way jobs execution affects thermal server node power consumption. We can write

$$
\left\{
\begin{array}{lcl}
\boldsymbol{\lambda}(\tau) & = & diag_B\{diag\{\boldsymbol{\lambda}^W(\tau)\}; N\}\boldsymbol{s}(\tau) \\
\tilde{\boldsymbol{\lambda}}(\tau) & = & \boldsymbol{\lambda}(\tau) + B_\lambda \boldsymbol{\xi}(\tau) \\
\tilde{\boldsymbol{\eta}}(\tau) & = & \boldsymbol{\eta}(\tau) + B_\eta \boldsymbol{\xi}(\tau) \\
\tilde{\boldsymbol{\nu}}(\tau) & = & \boldsymbol{\mu}(\tau) + B_\eta \boldsymbol{\delta}(\tau) \\
\dot{\boldsymbol{l}}(\tau) & = & \tilde{\boldsymbol{\lambda}}(\tau) - \tilde{\boldsymbol{\eta}}(\tau) \\
 & & \\
 & & \text{For all } i \in \mathcal{N},\ j \in \{1, \ldots, J\} \\
\eta_i^j(\tau) & = & \begin{cases} \mu_i^j(\tau) & \text{if } l_i^j(\tau) > 0 \text{ or } \tilde{\lambda}_i^j(t) > \mu_i^j(t) \\ \tilde{\lambda}_i^j(t) & \text{otherwise} \end{cases} \\
\xi_{h,i}^j(\tau) & = & \begin{cases} \delta_{h,i}^j(\tau) & \text{if } l_i^j(\tau) > 0 \text{ or } \tilde{\lambda}_i^j(t) > \tilde{\nu}_i^j(t) \\ \dfrac{\delta_{h,i}^j(\tau)}{\sum\limits_{h=1}^{N} \delta_{h,i}^j(\tau)}(\tilde{\lambda}_i^j(\tau) - \eta_i^j(\tau)) & \text{otherwise} \end{cases} \\
 & & \\
\mathsf{p}_{\mathcal{N}}(\tau) & = & \mathsf{A}_\alpha \boldsymbol{\eta}(\tau) + \mathsf{B}_\beta \boldsymbol{l}(\tau) \\
\dot{\boldsymbol{T}}_{\text{out},\mathcal{N}}(\tau) & = & diag\{-\mathbf{k}_{\mathcal{N}}\}\boldsymbol{T}_{\text{out},\mathcal{N}}(\tau) + \\
 & & diag\{\mathbf{k}_{\mathcal{N}}\}\boldsymbol{T}_{\text{in},\mathcal{N}}(\tau) + diag\{\mathbf{c}_{\mathcal{N}}\}\mathbf{p}_{\mathcal{N}}(\tau) \\
\dot{\boldsymbol{T}}_{\text{out},\mathcal{C}}(\tau) & = & diag\{-\mathbf{k}_{\mathcal{C}}\}\boldsymbol{T}_{\text{out},\mathcal{C}}(\tau) + diag\{\mathbf{k}_{\mathcal{C}}\} \min \Big\{\boldsymbol{T}_{\text{in},\mathcal{C}}(\tau), \boldsymbol{T}_{\text{ref}}(\tau)\Big\} \\
\dot{\boldsymbol{T}}_{\text{out},\mathcal{E}_1}(\tau) & = & diag\{-\mathbf{k}_{\mathcal{E}_1}\}\boldsymbol{T}_{\text{out},\mathcal{E}_1}(\tau) + \\
 & & diag\{\mathbf{k}_{\mathcal{E}_1}\}\boldsymbol{T}_{\text{in},\mathcal{E}_1}(\tau) + diag\{\mathbf{c}_{\mathcal{E}_1}\}\mathbf{p}_{\mathcal{E}_1}(\tau) \\
 & & \\
\boldsymbol{T}_{\text{in}\mathcal{N}\cup\mathcal{C}\cup\mathcal{E}_1}(\tau) & = & \Psi_{[\mathcal{N}\cup\mathcal{C}\cup\mathcal{E}_1, \mathcal{N}\cup\mathcal{C}\cup\mathcal{E}_1]}\boldsymbol{T}_{\text{out}\mathcal{N}\cup\mathcal{C}\cup\mathcal{E}_1}(\tau) + \Psi_{[\mathcal{N}\cup\mathcal{C}\cup\mathcal{E}_1, \mathcal{E}_2]}\boldsymbol{T}_{\text{out},\mathcal{E}_2}(\tau) \\
 & & \\
 & & \text{For all } i \in \mathcal{C} \\
\mathsf{p}_i(\tau) & = & \begin{cases} \mathsf{f}_i c_p \dfrac{T_{\text{in},i}(\tau) - T_{\text{out},i}(\tau)}{COP(T_{\text{out},i}(\tau))} & T_{\text{in},i}(\tau) \geq T_{\text{out},i}(\tau) \\ 0 & T_{\text{in},i}(\tau) < T_{\text{out},i}(\tau) \end{cases}
\end{array}
\right. \tag{4.22}
$$

where we defined the $N \times (N \cdot J)$ matrices A_α and B_β as

$$\mathsf{A}_\alpha = \begin{bmatrix} \boldsymbol{\alpha}_1^T & & \\ & \ddots & \\ & & \boldsymbol{\alpha}_N^T \end{bmatrix}, \qquad \mathsf{B}_\beta = \begin{bmatrix} \boldsymbol{\beta}_1^T & & \\ & \ddots & \\ & & \boldsymbol{\beta}_N^T \end{bmatrix}.$$

The variables at the data center level can be divided as

- **Input variables**
  - *Controllable*
    - Desired job execution rate, $\boldsymbol{\mu}(\tau)$.
    - The desired rate of job migration among zones, $\boldsymbol{\delta}(\tau)$.
    - The relative amount of jobs assigned to every node, $\boldsymbol{s}(\tau)$.
    - The reference temperature vector of CRAC nodes, $\boldsymbol{T}_{\text{ref}}(\tau)$.
  - *Uncontrollable*
    - The power consumption values of the environment nodes modeled by (3.20), $\mathbf{p}_{\mathcal{E}_1}(\tau)$.
    - The output temperatures of the environment nodes modeled by (3.21), $\boldsymbol{T}_{\text{out}\mathcal{E}_2}(\tau)$.
    - The rate at which jobs arrive at the data center at time τ, $\boldsymbol{\lambda}^W(\tau)$.
- **State variables**
  - The output temperature of thermal nodes, $\boldsymbol{T}_{\text{out}\mathcal{N}\cup\mathcal{C}\cup\mathcal{E}_1}(\tau)$.
  - The number of jobs in VMs, $\boldsymbol{l}(\tau)$.
- **Output variables**
  - The input temperature of thermal nodes, $\boldsymbol{T}_{\text{in}\mathcal{N}\cup\mathcal{C}\cup\mathcal{E}_1}(\tau)$.
  - The power consumption of server and CRAC nodes, $\mathbf{p}_{\mathcal{C}}(\tau)$,.

- The rate at which jobs depart from VMs after being executed, $\boldsymbol{\eta}(\tau)$.

The model in (4.22) does not contain discrete-valued variables. Also, since only the mean values of the random variables is considered, the model is deterministic. The model is still nonlinear, however the lack of discrete-valued variables and the use aggregated model make now possible the use of MPC approaches.

4.5 Equilibrium point and stability analysis

In Sec. 4.3.4, we discussed the stability of the thermal network as if its evolution was separated and independent from the evolution of the computational network. In this section we want to study the stability of the whole system, where the evolution of the computational and of the thermal network are coupled.

Assume that

- $\boldsymbol{T}_{\text{in}\mathcal{C}}(\tau) \geq \boldsymbol{T}_{\text{ref}}(\tau)$ for all $\tau \in \mathbb{R}$;
- all of the elements on the vector $\mathbf{k}$ are greater than 0;
- all of the elements of the vector $\mathbf{c}$ are nonnegative;
- the sum of the elements of $\Psi_{[\mathcal{Z},\mathcal{Z}]}$ along every row is lower than 1;
- For all $\tau \in \mathbb{R}$, $\boldsymbol{\mu}(\tau) > \boldsymbol{\lambda}$, $\boldsymbol{\delta}(\tau) = 0$, and $\boldsymbol{s}(\tau)$ is constant.

Under these hypotheses, the data center has an asymptotically stable equilibrium point at

$$\begin{cases} \boldsymbol{l} & = \mathbf{0} \\ \boldsymbol{T}_{\text{out}\mathcal{Z}} & = (I - \Psi_{[\mathcal{Z},\mathcal{Z}]})^{-1}\Psi_{[\mathcal{Z},\mathcal{C}]}\boldsymbol{T}_{\text{ref}} + (I - \Psi_{[\mathcal{Z},\mathcal{Z}]})^{-1} diag\{\mathbf{k}_{\mathcal{Z}}\}^{-1} diag\{\mathbf{c}_{\mathcal{Z}}\} \begin{bmatrix} \mathbf{p}_{\mathcal{N}} \\ \mathbf{p}_{\mathcal{E}_1} \end{bmatrix} + \\ & \quad (I - \Psi_{[\mathcal{Z},\mathcal{Z}]})^{-1}\Psi_{[\mathcal{Z},\mathcal{E}_2]}\boldsymbol{T}_{\text{out}\mathcal{E}_2} \\ \boldsymbol{T}_{\text{out}\mathcal{C}} & = \boldsymbol{T}_{\text{ref}}. \end{cases}$$

and the other variables have values

$$\boldsymbol{\lambda} = diag_B\{diag\{\boldsymbol{\lambda}^W\}; N\}\boldsymbol{s}, \qquad \tilde{\boldsymbol{\lambda}} = \boldsymbol{\lambda},$$

$$\tilde{\boldsymbol{\eta}} = \boldsymbol{\eta}, \qquad \tilde{\boldsymbol{\nu}} = \boldsymbol{\mu}, \qquad \boldsymbol{\eta} = \boldsymbol{\lambda},$$

$$\boldsymbol{\xi} = \mathbf{0}, \qquad \mathsf{p}_{\mathcal{N}}(\tau) = diag\{\boldsymbol{\alpha}\}\boldsymbol{\lambda},$$

$$\boldsymbol{T}_{\mathrm{in}\mathcal{N}\cup\mathcal{C}\cup\mathcal{E}_1} = \Psi_{[\mathcal{N}\cup\mathcal{C}\cup\mathcal{E}_1,\mathcal{N}\cup\mathcal{C}\cup\mathcal{E}_1]}\boldsymbol{T}_{\mathrm{out}\mathcal{N}\cup\mathcal{C}\cup\mathcal{E}_1} + \Psi_{[\mathcal{N}\cup\mathcal{C}\cup\mathcal{E}_1,\mathcal{E}_2]}\boldsymbol{T}_{\mathrm{out},\mathcal{E}_2}$$

and for all $i \in \mathcal{C}$

$$\mathsf{p}_i = \mathsf{f}_i c_p \frac{T_{\mathrm{in},i} - T_{\mathrm{out},i}}{COP(T_{\mathrm{out},i})}.$$

Note that the condition $\boldsymbol{T}_{\mathrm{ref}} \leq \boldsymbol{T}_{\mathrm{in}\mathcal{C}}$ has to hold also at the equilibrium point. Appendix A.7 discusses the proof of the stability (in the Lyapunov sense) of the equilibrium point.

4.6 Predictive discrete-time model of the data center

The coordinated and the uncoordinated controllers use a discrete-time version of the continuous time model previously discussed. We introduce at first the notation for discrete-time variables and then, we introduce the notation used in the predictive model.

Let $x(\tau)$ be a generic continuous-time variable and let t_0 and Δ be two real numbers with $\Delta > 0$. The value assumed by $x(\tau)$ at the beginning of the k^{th} interval is $x(t_0 + \Delta k)$ and we define $x(k) \triangleq x(t_0 + \Delta k)$ for all $k \in \mathbb{Z}$. Let $x(k)$ be a discrete-time variable and let h and k be two integer numbers with $h \geq k$. With $\hat{x}(h|k)$ we denote the estimated value of $x(h)$ based on the information available up to the beginning of the k^{th} interval.

The presence of nonlinearities in the data center model makes the discretization of (4.22) a nontrivial step. We discretize the thermal part of the data center considering an Euler discretization under the hypothesis that $\boldsymbol{T}_{\mathrm{ref}}(\tau) \leq \boldsymbol{T}_{\mathrm{in}\mathcal{C}}(\tau)$ for all $\tau \in \mathbb{R}$. In order to simplify the notation, in the discrete-time model, job departure and arrival rates represent the number of jobs arrived and departed during the k^{th} interval. In such a case, the evolution (over the discrete-time) of the number of jobs in the computational network can be written as

$$\boldsymbol{l}(k+1) = \boldsymbol{l}(k) + \tilde{\boldsymbol{\lambda}}(k) - \tilde{\boldsymbol{\eta}}(k). \tag{4.23}$$

Over the discrete time, (4.15) and (4.14) are approximated as

$$\eta_i^j(k) = \begin{cases} \mu_i^j(k) & \text{if } l_i^j(k) + \tilde{\lambda}_i^j(k) \geq \mu_i^j(k) \\ l_i^j(k) + \tilde{\lambda}_i^j(k) & \text{otherwise} \end{cases} \tag{4.24}$$

$$\xi_{i,h}^j(k) = \begin{cases} \delta_{i,h}^j(k) & \text{if } l_i^j(k) + \tilde{\lambda}_i^j(k) \geq \tilde{\nu}_i^j(k) \\ \dfrac{\delta_{h,i}^j(k)}{\sum_{h=1}^N \delta_{h,i}^j(k)} \Big(l_i^j(k) + \tilde{\lambda}_i^j(k) - \eta_i^j(k) \Big) & \text{otherwise.} \end{cases} \tag{4.25}$$

Note that the vector $\boldsymbol{\eta}(k)$ represents now the number of jobs departed from the zones over the k^{th} interval whereas the vector $\boldsymbol{\eta}(\tau)$ represents the rate at which jobs depart from the zones. In such a case, the coefficients of the vectors $\{\boldsymbol{\alpha}_i\}$ defined in (4.2) have to be scaled appropriately so that we can maintain the relationship

$$\mathbf{p}_{\mathcal{N}}(k) = \mathsf{A}_\alpha \boldsymbol{\eta}(k) + \mathsf{B}_\beta \boldsymbol{l}(k).$$

Since the coordinated and the uncoordinated controller observe the data center only over the discrete time, the constraints considered by the controllers are enforced only at the beginning of every control interval.

4.7 Constraints on the control variables

The desired rate at which jobs should be processed by a zone is bounded between 0 and the maximum rate at which the servers in the zone can process jobs. With $\bar{\mu}_i^j$ we denote the maximum number of jobs in the j^{th} class that the i^{th} zone can process over a control time interval. We define the $(N \cdot J) \times 1$ vector $\bar{\boldsymbol{\mu}} \triangleq \begin{bmatrix} \bar{\mu}_1^1 & \dots & \bar{\mu}_N^J \end{bmatrix}^T$. The bounds on the variables $\{\mu_i^j(k)\}$ can be written as

$$\mathbf{0} \leq \boldsymbol{\mu}(k) \leq \bar{\boldsymbol{\mu}}. \tag{4.26}$$

The desired number of jobs to be migrated from one zone to another depends on the number of jobs available in the zone at the beginning of the time interval and, on the number of jobs that will arrive to the zone over the interval. We set the bounds for the vector $\boldsymbol{\delta}(k)$ as

$$\mathbf{0} \leq \boldsymbol{\delta}(k) \leq \boldsymbol{l}(k) + \boldsymbol{\lambda}(k) + B_\lambda \boldsymbol{\xi}(k) \tag{4.27}$$

As earlier discussed, the elements of the vector $\boldsymbol{s}(\tau)$ are all bounded in the interval $[0, 1]$ and, for all $j \in \{1, \dots, J\}$ we have $\sum_{i=1}^N s_i^j(\tau) \leq 1$. Using the vectors earlier introduced, we can write

$$\mathbf{0} \leq \boldsymbol{s}(k) \leq \mathbf{1}, \qquad B_s \boldsymbol{s}(k) \leq \mathbf{1}, \tag{4.28}$$

where the $J \times (N \cdot J)$ matrix B_s is used to correctly align the elements of $\boldsymbol{s}(k)$.

The reference temperature vector is bounded by physical limits imposed by the CRAC unit. Let $\overline{T}_{\text{ref},i}$ and $\underline{T}_{\text{ref},i}$ be the upper and the lower bound of $T_{\text{ref},i}(\tau)$, respectively and define the $C \times 1$ vectors

$$\underline{\boldsymbol{T}_{\text{ref}}} \triangleq \begin{bmatrix} \underline{T}_{\text{ref},1} \\ \vdots \\ \underline{T}_{\text{ref},C} \end{bmatrix}, \qquad \overline{\boldsymbol{T}_{\text{ref}}} \triangleq \begin{bmatrix} \overline{T}_{\text{ref},1} \\ \vdots \\ \overline{T}_{\text{ref},C} \end{bmatrix}.$$

The constraints on the vector $\boldsymbol{T}_{\text{ref}}(k)$ can now be written as

$$\underline{\boldsymbol{T}_{\text{ref}}} \leq \boldsymbol{T}_{\text{ref}}(k) \leq \overline{\boldsymbol{T}_{\text{ref}}}. \tag{4.29}$$

4.8 Formulation of the data center control strategies

In this section we discuss a formulation of the three data center controllers earlier introduced: *baseline*, *uncoordinated*, and *coordinated*. In this note the controllers are derived assuming that performance metric is the minimization of a combination of the total expected data center power consumption and the expected number of jobs dropped. The combination can be written as

$$\sum_{h=k}^{k+\mathsf{T}-1} \left(\sum_{i=1}^{N} \hat{\mathsf{p}}_i(h|k) \right) - \boldsymbol{b}^T \hat{\boldsymbol{s}}(h|k), \tag{4.30}$$

where T is the number of steps in the future that the controllers consider and $\boldsymbol{b}$ is a nonnegative vector. This particular performance metric is not specific to the controllers and others will be considered in the following of the dissertation.

The uncoordinated and the coordinated subsections discuss also the feasibility and the stability properties of the two controllers. The feasibility of the controller refer to the ability to solve the optimization problem at future steps, given that a solution was found for the current step. The stability of the closed-loop system deals with the existence of equilibrium points for the closed-loop systems and on their stability properties. The analysis is developed assuming that the predictions of future job arrival rate are perfect. When predictions errors have to be considered, then robust version of the proposed results must be searched.

4.8.1 Baseline controller

The baseline controller sets the control variables to

$$\begin{aligned} \boldsymbol{\mu}(k) &= \overline{\boldsymbol{\mu}}, & \boldsymbol{\delta}(k) &= \boldsymbol{0}, \\ \boldsymbol{s}(k) &= \boldsymbol{1}\frac{1}{N}, & \boldsymbol{T}_{\text{ref}}(k) &= \underline{\boldsymbol{T}_{\text{ref}}}. \end{aligned} \tag{4.31}$$

The baseline controller does not attempt to tune the control strategy based on the current state of the data center, or on the chosen performance metric.

4.8.2 Uncoordinated controller

The uncoordinated controller synthesizes its control law by solving two optimization problems. The first optimization problem only considers the discrete-time evolution of the computational network. The second optimization problem only considers the discrete-time evolution of the thermal network. We assume the two optimization problems have the same horizon length, i.e., they consider the evolution of the two networks over the same number of steps. Let T be the time horizon considered by the two optimization problems.

With

$$\mathcal{M} = \{\hat{\boldsymbol{\mu}}(k|k), \ldots, \hat{\boldsymbol{\mu}}(k+\mathsf{T}-1|k)\}$$

we denote the set of variables $\{\hat{\boldsymbol{\mu}}(h|k)\}$ over which the optimization problem is considered. Similarly, we define the sets

$$\mathcal{S} = \{\hat{\boldsymbol{s}}(k|k), \ldots, \hat{\boldsymbol{s}}(k+\mathsf{T}-1|k)\},$$

$$\mathcal{D} = \{\hat{\boldsymbol{\delta}}(k|k), \ldots, \hat{\boldsymbol{\delta}}(k+\mathsf{T}-1|k)\},$$

and

$$\mathcal{T}_{\text{ref}} = \{\hat{\boldsymbol{T}}_{\text{ref}}(k|k), \ldots, \hat{\boldsymbol{T}}_{\text{ref}}(k+\mathsf{T}-1|k)\}.$$

At the beginning of every interval, the uncoordinated controller solves at first the following optimization problem

$$\begin{aligned}
\textbf{1.}\quad & \min_{\mathcal{M},\mathcal{S},\mathcal{D}} \sum_{h=k}^{k+\mathsf{T}-1} \left(\sum_{i=1}^{N} \mathsf{p}_i(h|k) \right) - \tilde{\boldsymbol{b}}^T \hat{\boldsymbol{s}}(h|k) \\
& \textbf{s.t.} \quad \text{for all } h = k, \dots, k+\mathsf{T}-1 \\
& \qquad \text{computational dynamics} \\
& \qquad \boldsymbol{0} \le \hat{\boldsymbol{\delta}}(h|k) \le \hat{\boldsymbol{l}}(h|k) + \hat{\boldsymbol{\lambda}}(h|k) + B_\lambda \hat{\boldsymbol{\xi}}(h|k) \\
& \qquad \boldsymbol{0} \le \hat{\boldsymbol{\mu}}(h|k) \le \overline{\boldsymbol{\mu}}, \\
& \qquad \boldsymbol{0} \le \hat{\boldsymbol{s}}(h|k) \le \boldsymbol{1}, \\
& \qquad B_s \hat{\boldsymbol{s}}(h|k) \le \boldsymbol{1} \\
& \qquad \hat{\boldsymbol{l}}(k|k) = \boldsymbol{l}(k).
\end{aligned} \tag{4.32}$$

Based on the solution obtained for the optimization in (4.32), the second part of the uncoordinated controller generates and solves the following optimization problem

$$\begin{aligned}
\textbf{2.}\quad & \min_{\mathcal{T}_{ref}} \sum_{h=k}^{k+\mathsf{T}-1} \sum_{i=N+1}^{N+C} \hat{\mathsf{p}}_i(h|k) \\
& \textbf{s.t.} \quad \text{for all } h = k, \dots, k+\mathsf{T}-1 \\
& \qquad \text{thermal dynamics} \\
& \qquad \underline{\boldsymbol{T}_{\text{ref}}} \le \hat{\boldsymbol{T}}_{\text{ref}}(h|k) \le \overline{\boldsymbol{T}_{\text{ref}}} \\
& \qquad \hat{\boldsymbol{T}}_{\text{in}}(h+1|k) \le \overline{\boldsymbol{T}_{\text{in}}} \\
& \qquad \hat{\boldsymbol{T}}_{\text{out}}(k|k) = \boldsymbol{T}_{\text{out}}(k)
\end{aligned} \tag{4.33}$$

The uncoordinated controller attempts to minimize the total average data center consumption by minimizing, at first, the expected power consumption of the zones and then, by minimizing the expected power consumption of the CRAC nodes. Furthermore, since CRAC nodes are not considered and controlled during the first part of the optimization problem, the vector $\boldsymbol{b}$ has to be rescaled in order to account for the missing contribution to the average power consumption given by the CRAC nodes. The specific formulation of the uncoordinated controller shows that the chosen performance metric may not be exactly represented by the controller and approximations have to be considered.

The uncoordinated control strategy is typically considered in modern data centers where the management of IT and of CT are demanded to two independent controllers. The first controller attempts to maximize the efficiency of IT subject to a given set of constraints about the QoS provided to the users. The second controller attempts to maximize the efficiency of the CT subject to a given set of constraints about the inlet temperature of IT. We show in the next chapter how this suboptimal approach may be, in some cases, as optimal as the solution obtained by a coordinated controller where IT and CT are managed in the same optimization problem.

Feasibility. The current form of the uncoordinated optimization problem always allow the solution

$$\hat{\boldsymbol{s}}(h|k) = \mathbf{0}, \;\; \hat{\boldsymbol{\mu}}(h|k) = \overline{\boldsymbol{\mu}}, \;\; \hat{\boldsymbol{\delta}}(h|k) = \mathbf{0}, \qquad \text{for all } h = k, \ldots, k + \mathsf{T} - 1.$$

If for all admissible values of the vectors $\{\hat{\mathbf{p}}_{\mathcal{N}}(h|k)\}$, the second optimization problem admits a solution, then the feasibility of the uncoordinated controller is guaranteed. From a practical point of view, this condition means that the number and the position of the CRAC units are adequate for cooling the servers for any admissible amount of heat that the servers can generate every second.

Stability. Stability is not guaranteed for the closed-loop system.

4.8.3 Coordinated controller

The coordinated control strategy is based on a discrete-time MPC approach and it manages the IT and the CT resources in a single optimization problem. The sets $\mathcal{M}$, $\mathcal{S}$, $\mathcal{D}$, and T_{ref} are defined as in the uncoordinated controller case. At the beginning of every interval, the coordinated controller solves the following optimization problem

$$
\begin{aligned}
\min_{\mathcal{M},\mathcal{S},\mathcal{D},\mathsf{T}_{ref}} \; & \sum_{h=k}^{k+\mathsf{T}-1} \left(\sum_{i=1}^{N+C} \mathsf{p}_i(h|k) \right) - \boldsymbol{b}^T \hat{\boldsymbol{s}}(h|k) \\
\textbf{s.t.} \quad & \text{for all } h = k, \ldots, k+\mathsf{T}-1 \\
& \quad \text{computational dynamics} \\
& \quad \text{thermal dynamics} \\
& \quad \mathbf{0} \le \hat{\boldsymbol{\delta}}(h|k) \le \hat{\boldsymbol{l}}(h|k) + \hat{\boldsymbol{\lambda}}(h|k) + B_\lambda \hat{\boldsymbol{\xi}}(h|k) \\
& \quad \mathbf{0} \le \hat{\boldsymbol{\mu}}(h|k) \le \overline{\boldsymbol{\mu}} \\
& \quad \mathbf{0} \le \hat{\boldsymbol{s}}(h|k) \le \mathbf{1} \\
& \quad B_s \hat{\boldsymbol{s}}(h|k) \le \mathbf{1} \\
& \quad \underline{\boldsymbol{T}_{\text{ref}}} \le \hat{\boldsymbol{T}}_{\text{ref}}(h|k) \le \overline{\boldsymbol{T}_{\text{ref}}} \\
& \quad \hat{\boldsymbol{T}}_{\text{in}}(h+1|k) \le \overline{\boldsymbol{T}_{\text{in}}} \\
& \quad \hat{\boldsymbol{l}}(k|k) = \boldsymbol{l}(k) \\
& \quad \hat{\boldsymbol{T}}_{\text{out}}(k|k) = \boldsymbol{T}_{\text{out}}(k) \\
& \quad \hat{\mathbf{p}}_{\mathcal{N}}(k|k) = \mathsf{A}_\alpha \boldsymbol{\eta}(k) + \mathsf{B}_\beta \boldsymbol{l}(k).
\end{aligned}
\tag{4.34}
$$

A drawback of the coordinated controller is the complexity of the optimization problem that has to be solved. Typically the optimization problem is non-convex and large. Local optimal solutions may yield control strategies that are worse than those obtained by an uncoordinated controller.

Feasibility. Note that in the current formulation of the coordinated controller, the control inputs

$$
\hat{\boldsymbol{s}}(h|k) = \mathbf{0}, \;\; \hat{\boldsymbol{\mu}}(h|k) = \overline{\boldsymbol{\mu}}, \;\; \hat{\boldsymbol{\delta}}(h|k) = \mathbf{0}, \qquad \text{for all } h = k, \ldots, k+\mathsf{T}-1
$$

enforce the control constraints. Such sequence of input variables sets to zero the power consumption of the zones. The feasibility of the coordinated controller is proved if, for every vector $\boldsymbol{T}_{\text{out}}(k)$ such that $\boldsymbol{T}_{\text{in}}(k) \leq \overline{\boldsymbol{T}_{\text{in}}}$, there exists a sequence of vectors $\hat{\boldsymbol{T}_{\text{ref}}}(h|k)$ such that

$$\hat{\boldsymbol{T}_{\text{in}}}(h|k) \leq \overline{\boldsymbol{T}_{\text{in}}},$$

for all $h = k+1, \ldots, \mathsf{T} + k$.

Stability. The coordinated controller does not guarantee the stability of the close-loop system.

4.9 Summary

This chapter discusses the proposed control approach. The approach is based on a hierarchical/distributed control strategy. The hierarchy is composed of three control levels and the chapter focuses on the controllers at the highest level of the hierarchy.

Three controllers are discussed: a baseline controller, an uncoordinated controller, and a coordinated controller. The controllers are representative approaches of control strategies that can be implemented in real data centers. The controllers also take advantage of the unified, thermal-computational model of the data center in different ways. The baseline sets the values of the control inputs so as to satisfy the control constraints for the worst-case scenario, but it does not attempt to maximize the performance of the data center. The uncoordinated and the coordinated controllers attempt to maximize the performance of the data center via a MPC approach. The uncoordinated neglects the coupling between the computational and the thermal network, whereas the coordinated controller considers such a coupling. The coordinated and the uncoordinated controllers do not guarantee the stability of the system. As discussed in Sec. 4.5, there are conditions under which the data center, when it operates without a controller, has a unique and asymptotically stable equilibrium point. The coordinated and the uncoordinated controllers however, in the attempt to maximize the performance of the data center, may destabilize the system.

Chapter 5

Performance analysis of data-center-level controllers

This chapter discusses the analysis of the data-center-level controllers under a variety of scenarios. The results discussed in this chapter have been obtained by simulating the behavior of multiple data centers under a variety of scenarios. The following section gives a brief introduction to the data center simulator we developed in MATLAB. Section 5.2 discusses the data center layout used for developing most of the simulations. Section 5.3 analyzes the performance of the baseline, uncoordinated, and coordinated controller for multiple, constant, workload arrival rates. Section 5.4 discusses the index proposed to characterize the savings that can be obtained by using a coordinated controller with respect to an uncoordinated controller. Finally, Sec. 5.5 summarizes the results discussed in this chapter.

5.1 Introduction to the MATLAB simulator

The simulations hereafter discussed are developed using a collection of integrated MATLAB routines, created by us, which we call *data center simulator*. The simulator is developed following a modular approach. Figure 5.1 shows the blocks which compose the data center simulator. Every block represents a collection of MATLAB files which implement a certain

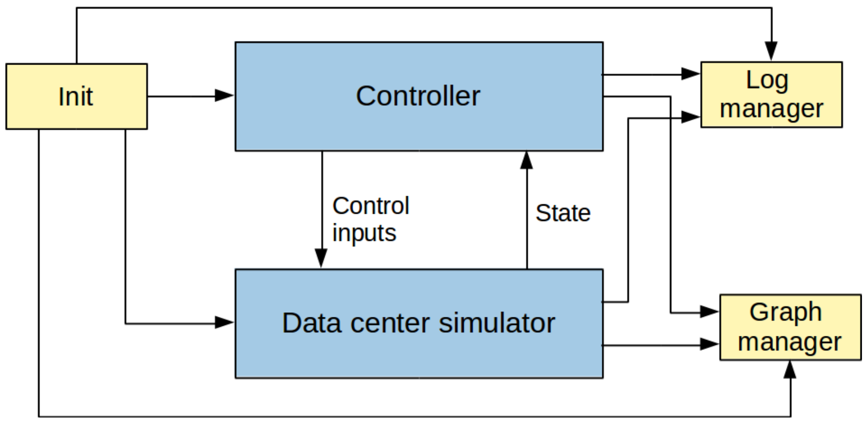

Figure 5.1: Functional blocks that compose the data center simulator.

function of the simulator.

The block `Init` is responsible for calling, in the right order, the initialization functions of the other blocks. For example, the `Data center simulator` blocks initializes all of the data structures necessary to run the simulation, the `Controller` blocks generates the optimization problem of the data center controllers (when an optimization problem is used), the `Log manager` opens the file which will store simulation logs, and the `Graph manager` blocks generate the graphs where the plots will be drawn.

The `Data center simulator` block is responsible for simulating the evolution of the data center over time. When necessary, the `Data center simulator` block passes all of its data to the `Controller` block in order to generate a new control input. User messages, such as error and warnings, are managed by the `Log manager` block. The `Log manager` block formats the data uniformly and, among other data, adds a time-stamp to each of the message written on the log file. The `Graph manager` block is responsible to handle the plots. The plots are updated every few seconds so that the evolution of the data center can be observed while the simulation runs.

All of the data center data are stored in a struct data type called `Parameters`. This allows (but not always guarantee) that the data are passed by reference among the multiple functions used to simulate the data center and to synthesize the control inputs. Such an

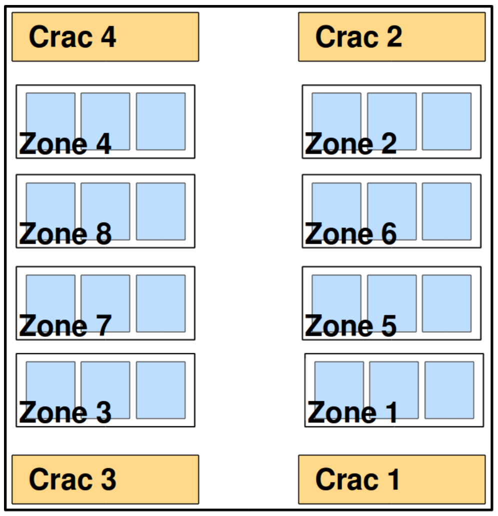

Figure 5.2: Data center layout: 8 server nodes (blue) and 4 CRAC nodes (orange).

approach reduces the amount of data that have to be copied from one function to another and speed-up the execution of the simulation.

The data center simulator is developed considering three design goals: modularity, performance, and code readability. The blocks in Fig. 5.1 are composed by multiple functions, each focused on a single goal. Functions and variables have meaningful names so that they can be easily maintained and used by multiple users. The functions also avoid the use of loops as much as possible, without however, compromising the readability of the code. If necessary, part of the data center simulator code can be reimplemented using `mex` files. This would take fully advantage of the performance tools available in the MATLAB language.

5.2 Data center layout

The simulation results discussed in this chapter refer to the data center layout shown in Fig. 5.2. The picture represents a small data center containing 32 racks and 4 CRAC units. The CT comprises 4 CRAC units which cool the servers through a raised-floor architecture. Racks are grouped into eight zones. Three racks are contained in every zone and every rack contains 42 servers. Under the same workload, servers in zones 5 to 8 consume 10%

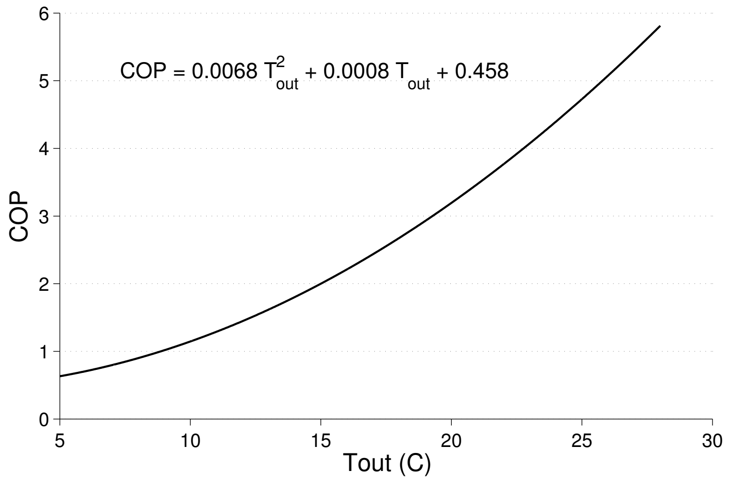

Figure 5.3: Coefficient of performance of the CRAC units for different output temperature values [44].

less power than other servers, but they are not cooled as efficiently as the servers in zones 1 to 4, which are closer to the CRAC units. The maximum power consumption of the servers in zones 5 to 8 is 270 (W) and the maximum power consumption of the servers in zones 1 to 4 is 300 (W). It is assumed that every zone has a local controller that forces the zone to behave as postulated in the data center model, i.e., the amount of power consumed by every zone is proportional to the workload execution rate. CRAC units are identical and their efficiency, i.e., their COP, increases quadratically with respect to their output temperatures. The relationship between the output temperatures of the CRAC units and their COP is shown in Fig. 5.3. The values of all of the parameters used in the simulations enforce the constraints discussed in Sec. 4.5. This implies that, if the controllers use constant input control values that enforce all of the given constraints and those additional constraints discussed in Sec. 4.5, then the existence of an equilibrium point and its stability are guaranteed. The maximum input temperature allowed for every zone is 27 °C, i.e., $\overline{\boldsymbol{T}_{\text{in}}} = 27$. This constraint reflects the environmental guidelines given by the ASHRAE [5].

The specific layout was chosen because it highlights the trade-off that controllers have to make between using energy-efficient servers and efficiently cooled servers. Furthermore, we decide to limit the number of zones and CRAC units in the simulation in order to facilitate the analysis of the results.

The simulations are obtained via the MATLAB simulator. The TomSym language is used to code the optimization problems into the MATLAB simulator. The numerical solver used to solve the optimization problems is Knitro 7.0.[1]

5.3 Analysis under constant workload arrival rates

This section discusses the performance of the baseline, uncoordinated, and coordinated control algorithm under the assumption of a constant workload arrival rate. The performance of the three controllers are evaluated in the ideal case where the controllers have perfect knowledge about the data center. The performance metric for all of the three controllers is the minimization of the long-term average data center power consumption. Aside from the constraints on the inlet temperature of the zones, the controllers have to enforce the following additional constraints:

- The desired job execution rate of every zone must be greater than, or equal to the job arrival rate.
- No jobs can be dropped.
- No jobs can be migrated from one zone to another.

The values of the parameters used for the simulation are set so as to guarantee that the three constraints do not lead to an unfeasible optimization problem. For example, the parameter values are set so that the maximum job execution rate is greater than the maximum rate at which jobs arrive to the zone. Also, the maximum cooling capacity of the CRAC units is set to be greater than the maximum amount of cooling required by the zones.

[1]`http://tomsym.com/` and `http://www.ziena.com/knitro.html` .

In the simulations we compare the *power usage effectiveness* (PUE) obtained by the three controllers. The PUE is a measure, typically used in industry, of data center efficiency and it is defined as the ratio between the total data center power consumption over the power consumption of the IT [71]. Since no environments nodes are considered in the simulation, the PUE of a data center at time τ can be written as

$$PUE(\tau) \triangleq \frac{\sum_{i=1}^{N+C} \mathsf{p}_i(\tau)}{\sum_{i=1}^{N} \mathsf{p}_i(\tau)} = 1 + \frac{\sum_{i=N+1}^{N+C} \mathsf{p}_i(\tau)}{\sum_{i=1}^{N} \mathsf{p}_i(\tau)}. \tag{5.1}$$

The second equality in (5.1), shows that the PUE values increase with the ratio between the total amount of power consumed by CRAC units and the total amount of power consumed by the zones. A PUE of 1.0 indicates that all of the data center power consumption is due to the IT. Data centers based on state-of-the-art cooling and load balancing technology can reach PUE values of 1.1, i.e., 90.9% of the total data center power consumption is consumed by IT [51].[2] A drawback of PUE is that it does not take into account IT equipment efficiency. In this simulation however, the zone power consumption grows linearly with respect to the rate at which jobs are processed. Therefore, the efficiency of IT is known and constant. In such a case, the PUE is an informative index and it helps us to compare the cooling efficiency obtained by the three control algorithms under the same job arrival rate.

The total data center power consumption and the PUE values are shown with respect to the average data center utilization. The average data center utilization is defined as the mean values of the ratios $\frac{\eta_i(\tau)}{\bar{\mu}_i}$ for $i = 1, \ldots, 8$. When the average utilization is 0 then zones process no jobs. When the average utilization is 1 then zones process jobs at the maximum allowable rate. The behavior of the three controllers is not considered for small

[2]Data from `http://www.google.com/corporate/datacenter/efficiency-measurements.html`

Table 5.1: Average and standard deviation values of $\sum_j \psi_{i,j}$. The coefficients refer to the graphs in Fig. 5.4 and Fig. 5.5.

$j \rightarrow$	Zones $1-4$		Zones $5-8$		CRACs	
$\downarrow i$	Avg.	Std.	Avg.	Std.	Avg.	Std.
Zones $1-4$	0.04	2.6e-6	0.03	2.2e-6	0.93	4.8e-6
Zones $5-8$	0.05	9.9e-7	0.52	4.8e-5	0.43	4.8e-5
CRACs	0.63	2.0e-5	0.25	4.3e-5	0.12	2.3e-5

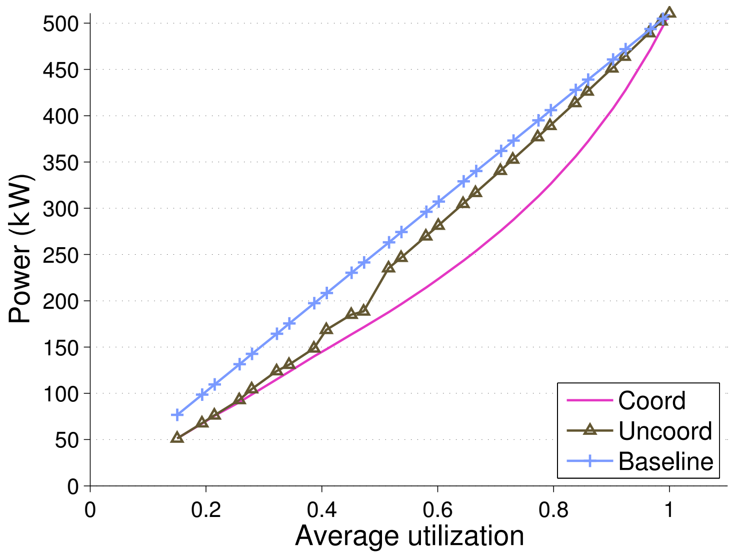

Figure 5.4: Average data center power consumption for different utilization values.

average utilization values since, at very low utilization values, nonlinearities of the IT and CT neglected in the proposed model become relevant. The coefficients relating input and output temperatures of zones and CRAC units are summarized in Table 5.1. Since in this study we focus solely on the average total data center power consumption, we assume that all of the jobs belong to the same class, i.e., $J = 1$.

Figure 5.4 shows the total data center power consumption obtained by the three controllers for different average utilization values.

- **Baseline controller.** The total data center power consumption obtained by the baseline controller grows proportionally with the average data center utilization. The

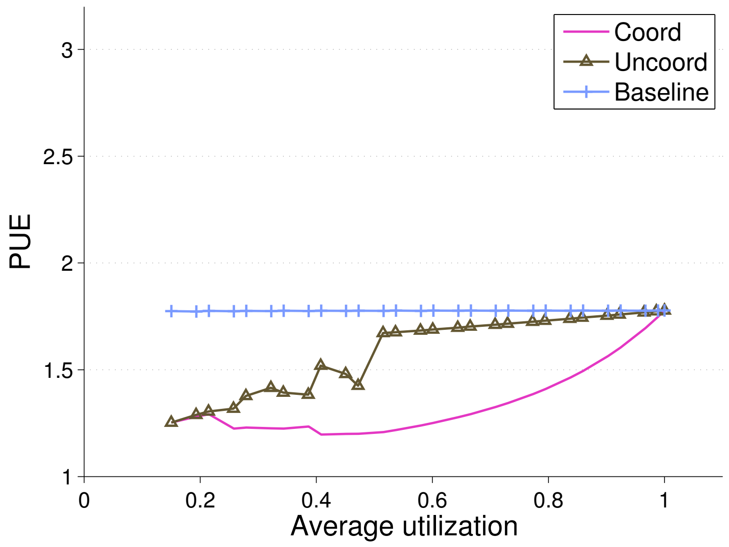

Figure 5.5: PUE values obtained by the baseline, uncoordinated, and coordinated controller.

proportional growth is due to two factors. The first factor is the assumption that rack-level and server-level controllers make the power consumption of every zone to grow proportionally with the amount of workload that the zones process. The second factor is that the reference temperature of every CRAC unit is fixed and always lower than or equal to the input temperature of the CRAC unit. In such a case, the CRAC units operate at a constant efficiency and their total power consumption is proportional to the amount of power that the zones consume. The absence of environment nodes in the first group forces the power consumption curve to tend to 0 as the average utilization value tends to 0. The PUE values obtained by the baseline controller are shown in Fig. 5.5.

- **Uncoordinated controller.** The total data center power consumption obtained by the uncoordinated controller is always lower than the power consumption obtained by the baseline controller. This happens because the uncoordinated controller assigns as much workload as possible to the most energy-efficient servers, i.e., those servers located in zones 5 to 8, and it tries to maximize the efficiency of the CT by setting the reference value of every CRAC unit to the largest value that still enforces the thermal constraints. The uncoordinated controllers attempt to maximize the efficiency of the CRAC units when solving the second part of its optimization problem. Therefore, as shown in Fig. 5.5, the uncoordinated controller is able to obtained lower PUE values than the baseline controller.

- **Coordinated controller.** The additional reduction in the data center power consumption obtained by the coordinated controller are due to the coordinated management of the zones and CRAC units. In particular, the coordinated controller, depending on the amount of workload that the zones have to process, decides whether it is more efficient to allocate workload on the energy-efficient servers, i.e., to the servers in zones 5 to 8, or on the efficiently cooled servers, i.e., to the servers in zones 1 to 4. Taking full advantage of the cyber-physical of the data center, the coordinated controller is able to further improve the efficiency of the CRAC units with respect to the uncoordinated controller. Such additional efficiency is reflected in the lower values of the PUE, as shown in Fig. 5.5.

The power consumption and PUE curves obtained by the uncoordinated and by the coordinated controller are not smooth because, for some values of the workload arrival rate, the controllers are unable to locate the true minimum of the optimization problems they solve. In these cases, the control actions are based on a local minimum that may be different from the control actions chosen for nearby values of the workload arrival rate.

The results obtained in this first simulation depend, among others, on the particular thermal coupling between CRAC units and zones and between the zones themselves. We want now to study how the performance of the coordinated and of the uncoordinated controller change when different thermal coupling exist in the data center. We study therefore,

Table 5.2: Average and standard deviation values of $\sum_j \psi_{i,j}$. The coefficients refer to the graphs in Fig. 5.6 and Fig. 5.7.

$j \rightarrow$	Zones $1-4$		Zones $5-8$		CRACs	
$\downarrow i$	Avg.	Std.	Avg.	Std.	Avg.	Std.
Zones $1-4$	0	0	0	0	1	0
Zones $5-8$	0.3	2.9e-5	0.4	8.0e-6	0.30	2.9e-5
CRACs	0.51	5.6e-5	0.34	3.4e-5	0.15	2.5e-5

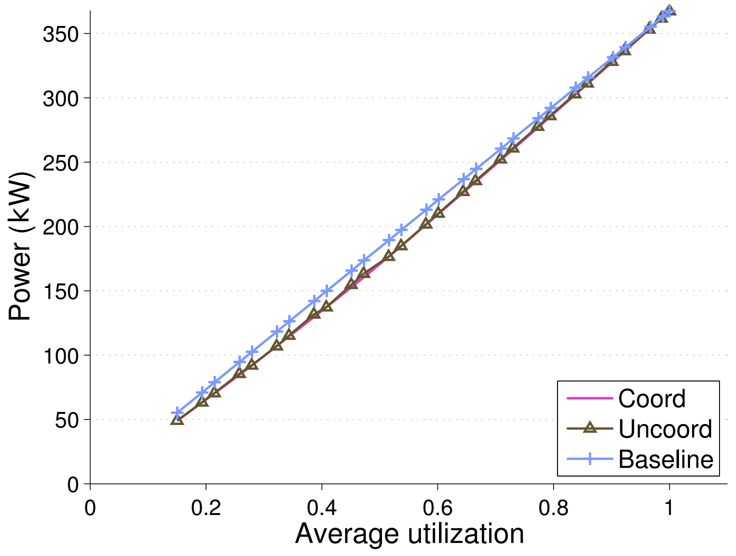

Figure 5.6: Average data center power consumption for different utilization values. All of the zones are efficiently cooled.

the performance of the three controllers under other data center cases.

In the second simulation, we focus on a case where the inlet temperatures of the servers in zones 1 to 4 equal the supplied air temperatures of the CRAC units, i.e., $\sum_j \psi_{i,j} \simeq 1$ when i is the index of the zones 1 to 4 and j represents the CRAC units only. Also, in the second simulation the servers in zones 1 to 4 are subject to less air-recirculation than in the first simulation case and their inlet temperatures depends more on the output temperatures of the servers in zones 5 to 8. In this new simulation, where all zones are efficiently cooled, the coordinated and the uncoordinated have the same performance. Therefore, for data

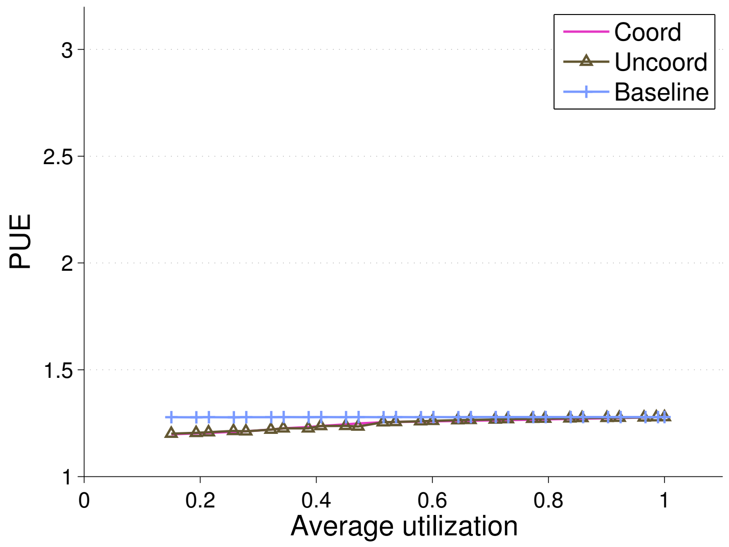

Figure 5.7: PUE values obtained by the baseline, uncoordinated, and coordinated controller. All of the zones are efficiently cooled.

centers that fall into this case, an uncoordinated controller is a viable control approach since it is as optimal as the coordinated one and simpler to implement. Figure 5.6 shows the total data center power consumption obtained by the three controllers in the second simulation. Table 5.2 summarizes the values of the coefficients relating input and output temperatures of zones and CRAC units for this simulation. The other parameters did not change. The PUE values obtained by the three controllers for this simulation are shown in Fig. 5.7.

In the third simulation we focus on the case where large variability exists between the efficiency at which servers in zone 1 to 4 are cooled and the efficiency at which servers in zone 5 to 8 are cooled. Table 5.3 summarizes the values of the coefficients relating input and output temperatures of zones and CRAC units for this simulation. The other parameters did not change. Figure 5.8 shows the total data center power consumption obtained by the three controllers in the third simulation. In this new simulation case, the PUE values obtained by the coordinated and by the uncoordinated controller, shown in

Table 5.3: Average and standard deviation values of $\sum_j \psi_{i,j}$. The coefficients refer to the graphs in Fig. 5.8 and Fig. 5.9.

$j \rightarrow$	Zones $1-4$		Zones $5-8$		CRACs	
$\downarrow i$	Avg.	Std.	Avg.	Std.	Avg.	Std.
Zones $1-4$	0.08	0	0.08	0	0.84	0
Zones $5-8$	0.08	7.0e-8	0.66	4.8e-5	0.26	4.8e-5
CRACs	0.57	5.4e-5	0.18	9.0e-5	0.25	4.1e-5

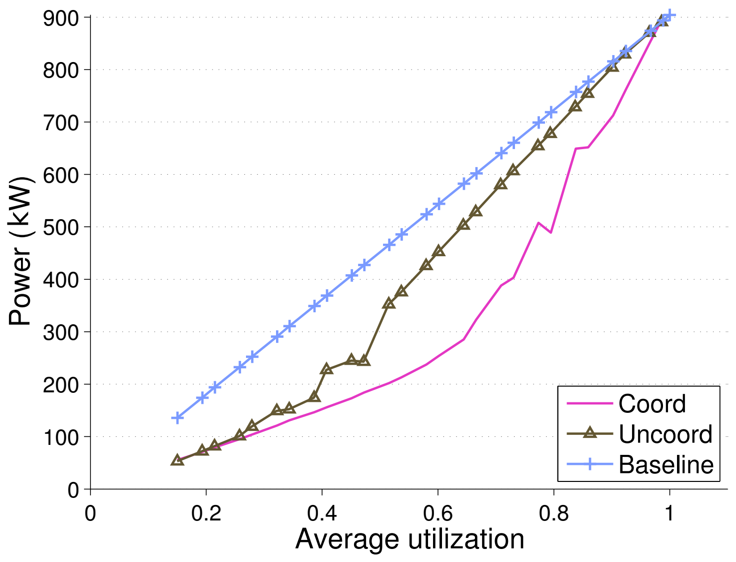

Figure 5.8: Average data center power consumption for different utilization values. Large variability among cooling efficiency of the zones.

Fig. 5.9, strongly depend on the average utilization of the data center.

The simulations discussed in this section show that at very high or at very low level of data center utilization, different control approaches yield the same performance. In these cases, the use of the baseline control strategy is a viable solution. The largest performance improvements, i.e., the largest reduction of the total data center average power consumption, are obtained when data center utilization is about 0.5. Such a results was expected. When no servers are used, they generate no heat (we assume a linear relationship between job execution rate and zone power consumption) and CRAC units remove no heat.

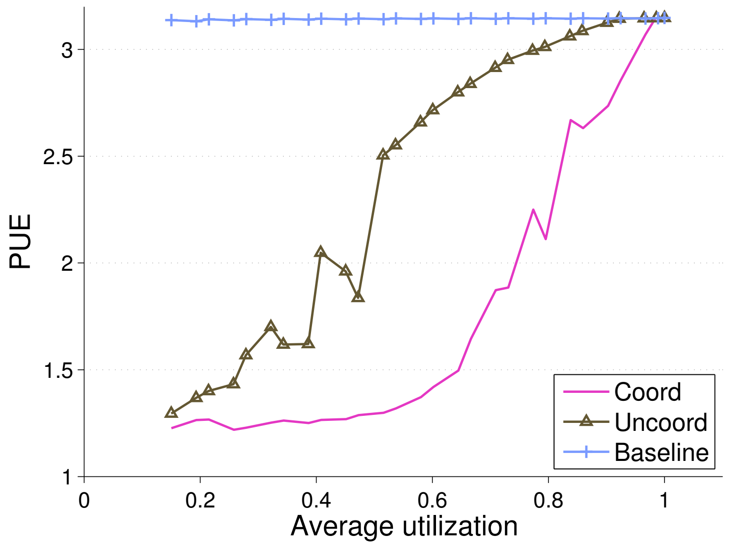

Figure 5.9: PUE values obtained by the baseline, uncoordinated, and coordinated controller. Large variability among the cooling efficiency of the zones.

When instead, all of the server resources are used, then controllers are forced to use all of the available cooling and computational resources of the data center and they cannot improve over the baseline strategy. Only when the controllers can choose where to allocate jobs among zones and how to cool the zones, then the control performances of coordinated, uncoordinated, and baseline controllers are different from each other. The assumption about the proportionality between the job execution rate and the zone power consumption is fundamental for these simulations. Such an assumption is reasonable when zone level controllers can turn-off unused servers and hence, remove the idle power consumption of servers.

5.4 A cyber-physical index for data centers

Given a data center, it would be useful to estimate a priori how much energy it could be saved by using a coordinated controller rather than an uncoordinated one. Towards this end, we define *relative efficiency* the ratio between the area below power consumption curve obtained by the uncoordinated controller and the power consumption curve obtained by the coordinated controller. With the appropriate weights, the relative efficiency can be mapped into the average monthly, or average yearly energy savings obtained by using a coordinated controller with respect to an uncoordinated controller.

Consider a data center at its thermal and computational equilibrium and assume that all of the constraints satisfied. Under the hypothesis that no environment nodes are in the data center, (4.8) leads to

$$\begin{aligned} \boldsymbol{T}_{\text{in},\mathcal{N}} &= (I - \Psi_{[\mathcal{N},\mathcal{N}]})^{-1} diag\{\mathbf{k}_{\mathcal{N}}\}^{-1} diag\{\boldsymbol{c}_{\mathcal{N}}\} \mathsf{A}_{\alpha} \boldsymbol{\eta} + \Psi_{[\mathcal{N},\mathcal{C}]} \boldsymbol{T}_{\text{ref}} \\ &= \boldsymbol{\mathcal{L}} \boldsymbol{\eta} + \Psi_{[\mathcal{N},\mathcal{C}]} \boldsymbol{T}_{\text{ref}}, \end{aligned} \tag{5.2}$$

where $\boldsymbol{\mathcal{L}} = (I - \Psi_{[\mathcal{N},\mathcal{N}]})^{-1} diag\{\mathbf{k}_{\mathcal{N}}\}^{-1} diag\{\boldsymbol{c}_{\mathcal{N}}\} \mathsf{A}_{\alpha}$.

The variations of the input temperatures of the thermal nodes with respect to a variation of the workload execution rate and with respect to a variation of the reference temperature vector can be written as

$$\frac{\partial \boldsymbol{T}_{\text{in},\mathcal{N}}}{\partial \boldsymbol{\eta}} = \boldsymbol{\mathcal{L}}, \qquad \frac{\partial \boldsymbol{T}_{\text{in},\mathcal{N}}}{\partial \boldsymbol{T}_{\text{ref}}} = \Psi_{[\mathcal{N},\mathcal{C}]}. \tag{5.3}$$

The inlet temperature of an efficiently cooled server largely depends on the reference temperature of the CRAC units and marginally on the execution rate of other servers. Let i be the index of a thermal node representing a zone. We say the i^{th} node is efficiently cooled if [3]

$$\left\| \frac{\partial T_{\text{in},i}}{\partial \boldsymbol{T}_{\text{ref}}} \right\|_2 \gg \left\| \frac{\partial T_{\text{in},i}}{\partial \boldsymbol{\eta}} \right\|_2. \tag{5.4}$$

The problem with (5.4) is that, with an opportune choice of the scaling coefficient, the norm of $\frac{\partial T_{\text{in},i}}{\partial \boldsymbol{\eta}}$ can be set to any desired (nonnegative) value. We consider therefore the vector $\boldsymbol{z} = \begin{bmatrix} \boldsymbol{T}_{\text{ref}}{}^T & \boldsymbol{\eta}^T \end{bmatrix}^T$ and define the *relative sensitivity index* of the i^{th} node as

$$\mathcal{S}_i = \left\| \frac{\partial T_{\text{in},i}}{\partial \boldsymbol{T}_{\text{ref}}} \right\|_2 \Big/ \left\| \frac{\partial T_{\text{in},i}}{\partial \boldsymbol{z}} \right\|_2.$$

[3]We focus on 2-norm, but other norms can be considered.

When the relative sensitivity index of the i^{th} zone equals 1, the input temperature of the i^{th} zone uniquely depends on the reference temperature of the CRAC nodes, whereas when the relative sensitivity index equals 0, then the input temperature of the i^{th} zone only depends on the workload execution rate of the other zones. For all $i \in \mathcal{N}$, the index $\mathcal{S}_i$ takes values in the interval $[0, 1]$ independently from the scaling coefficient used for $\frac{\partial T_{\text{in},i}}{\partial \boldsymbol{\eta}}$.

We collect the relative sensitivity indexes in the vector $\boldsymbol{S}$ and define as *cyber-physical index* (CPI) of a data center the normalized standard variation of $\boldsymbol{S}$

$$CPI = k \cdot std(\boldsymbol{S}), \tag{5.5}$$

where k is the normalizing coefficient and std is the standard deviation of the elements of the vector argument.

Figure 5.10 shows the relative efficiency obtained by the coordinated controller for different data center cases. When the CPI values are larger than about 0.55, the uncoordinated controller is unable to find a cooling strategy that satisfies the thermal constraints for large values of the average data center utilization and hence, we do not show here data center cases having index larger than 0.55. When the CPI is almost 0, the relative efficiency of a coordinated controller is almost 0. As the CPI increases, the efficiency of a coordinated controller grows exponentially fast. The simulation cases discussed in the previous section correspond to a CPI of 0.33, 0.04, and 0.52 respectively.

The exponential growth of the relative efficiency as a function of the CPI suggests that, in order to minimize the power consumption of a data center controlled by an uncoordinated controller, a careful positioning of the servers should be made. Different locations of a data center are subject to different cooling efficiency. Given a data center, the relocation of some of its servers may move the CPI of the data center toward smaller values so that an uncoordinated controller will be as efficient as a coordinated controller.

The proposed CPI is a first step toward the characterization of data center with respect

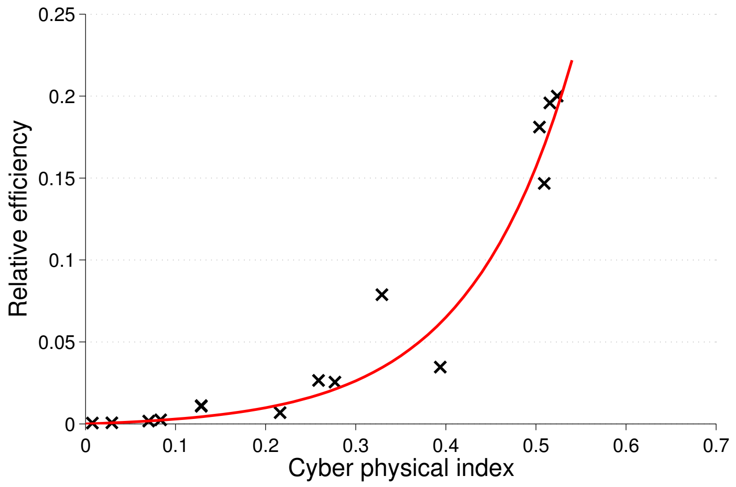

Figure 5.10: Relative savings of the coordinated controller with respect to the uncoordinated controller.

to their cyber-physical features. We recognize however, that the proposed index does not capture all of the variability in a data center. For example, the efficiency of different CRAC units is not accounted in the index. Nonetheless, one of the strengths of the CPI is that its value can be estimated via multiple tests executed on a data center. In order to derive the index it is necessary to repeatedly change the workload rate and the cooling of different zones and CRAC units and then, to measure the variation of the input temperature of the zones at the equilibrium. These tests, should be execute for different values of the average data center utilization.

5.5 Summary

This chapter compares the performance obtained by the baseline, the uncoordinated, and the coordinated controller under the assumption of a constant workload arrival rate. Multiple data center cases are considered in order to understand how the three controllers behave in a variety of scenarios.

The results discussed in the chapter show that every data center has operating regimes for which the baseline and the uncoordinated controllers are as optimal as the coordinated controller. Furthermore, the results show the existence of data center cases for which the uncoordinated controller is as optimal as the coordinated one, independently from the operating regime. An index, called *cyber-physical index* (CPI), is proposed in order to characterize how much savings can be obtained by using a coordinated controller with respect to an uncoordinated controller. The proposed CPI is a first step toward the characterization of data center with respect to their cyber-physical features. The exponential growth of the relative efficiency as a function of the CPI suggests that, in order to minimize the power consumption of a data center controlled by an uncoordinated controller, a careful positioning of the servers should be made.

Chapter 6

Interaction with the smart-grid

In this chapter, an interaction between a data center and the power-grid is discussed. The goal of the control algorithms is the minimization of the run-time operating cost of the data center. Such a cost depends on the cost of powering the data center and on the revenue generated by the execution of jobs with a certain QoS. A baseline controller is not considered in this chapter, since its open-loop strategy, that does not attempt to maximize the performance of the data center, would yield the same results already discussed in the previous chapter.

The chapter is divided as follows. Section 6.1 introduces and motivates the interaction between a data center and the power-grid. Section 6.2 formulates the run-time operating cost of the data center used in this chapter. Section 6.3 formulates the optimization problems of the uncoordinated and of the coordinated controller. Section 6.4 discusses the parameters used in the simulations. Section 6.5 compares the performance obtained by the uncoordinated and by the coordinated control algorithm, when a very high cost of dropping jobs is used in the run-time cost of the data center. Section 6.6 discusses the performance obtained by two different coordinated controllers. Finally, Sec. 6.7 summarizes the results obtained in this chapter.

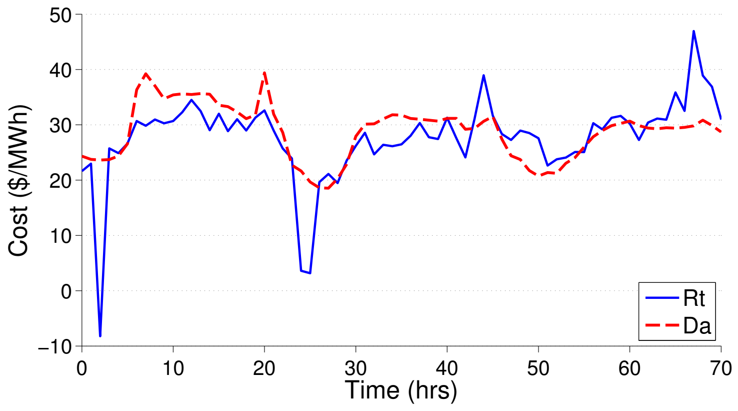

Figure 6.1: Examples of real-time and day-ahead cost of electricity.

6.1 Data centers as smart-nodes of the power-grid

Data centers can play an important role in the smart-grid scenario because of their high power consumption density. A low-density data center can have a peak power consumption of 800 W/m^2 (75 W/sq.ft.), whereas a high-density data center can reach 1.6 KW/m^2 (150 W/sq.ft.) [29, 56, 68]. These values are much higher than residential loads, where the peak power consumption is about a few watt per squared meter [42, 46].

We consider the case of a deregulated electricity market. In a deregulated electricity market, the price of electricity changes over time and over different areas [3]. In the U.S., two electricity markets can be used to purchase electricity: the *day-ahead* and the *real-time* market. Electricity price varies over time and over different areas in both markets and typically, electricity price oscillates much more in the real-time market than in the day-ahead market. Figure 6.1 shows an example of real-time and day-ahead costs of the electricity for the north-east area of U.S.[1] A feasible approach to reduce the variability of the electricity cost and also, to tackle critical situations such as network failure, is to require to some costumers of the power-grid to cap their power consumptions upon request from the grid. Such a technique, already applied by some independent system operators

[1]Data from PJM `http://www.pjm.com/` .

(ISOs), is called *demand response program* (DRP).[2]

Data centers may take advantage of DRPs. For example, a data center can receive electrical energy at a discounted price for participating into a DRP and, when required by the power-grid, it caps its power consumption below a given threshold. We focus on the more flexible case, where a data center can choose whether to cap its power consumption or not. If the data center does not cap its power consumption it pays a penalty to the power-grid operators. The choice between capping its power consumption, or paying a fee to the grid, depends on what is more profitable for the data center. If the data center caps its power consumption, then it may have to drop jobs or to process them with a low QoS which may induce a monetary loss. For example, an increase of 100 ms on the time to complete a user search requests induces a 1% of sales loss to Amazon, [17]. If the data center does not cap its power consumption, then it has to pay a fee to the power-grid, which may overshadow the revenue obtained by executing user requests with high QoS.

We formulate this problem assuming that, in order to minimize the operating cost, a data center level controller can leverage over two *service level agreements* (SLAs): SLA_U with the data center users and SLA_G with the grid. SLA_U sets the income for the data center when it executes the required workload with a certain quality of service (QoS), e.g., within a certain delay. If the QoS exceeds a certain threshold, then the income is negative. SLA_G sets the cost of electricity at over time and for different levels of power consumption. As long as the power consumption is lower than a time-varying threshold, the electricity price is kept at a reduced value. When the data center power consumption exceeds the given threshold, then the additional electricity is provided at a much higher price. To maximize the profit, a data center controller has to predict the future power consumption of the data center and decide if it is more profitable to drop or to delay the execution of jobs, or to buy energy at a high price and process all of the jobs with a high QoS. Additional details about the results presented in this section can be found in [50].

[2] `http://pjm.com/markets-and-operations/demand-response.aspx`

Aside from the cost of powering the data center and from the revenue obtained by executing jobs with a certain QoS, we also consider the cost of migrating jobs from one zone to another. The additional component of the cost function is added to reduce the amount of jobs migrated among the zones. The three components considered in the data center cost function are:

- cost of powering the data center, regulated by SLA_G;
- revenue induced by executing jobs with a certain QoS, regulated by SLA_U;
- cost of migrating jobs from one zone to another.

6.2 Data center operating cost

In the following subsections we define the cost function that the coordinated and the uncoordinated controller aims at minimizing. We prefer to consider the negative value of the revenue induced by executing jobs with a certain QoS and hence, we formulate the *cost of QoS*.

6.2.1 Cost of powering the data center.

Let $\bar{\mathsf{p}}(k)$ be the threshold which allows the data center to buy electricity at a reduced price and let $\mathsf{p}(k)$ be the total average data center power consumption during the k^{th} interval. The data center buys electricity at the reduced cost $\alpha_e(k)$ as long as $\mathsf{p}(k) \leq \bar{\mathsf{p}}(k)$. The energy exceeding $\bar{\mathsf{p}}(k)$ is paid at the higher cost $\beta(k)$. The fee that the data center pays to the power-grid is included in the higher cost of the electricity price. Let $c_e(k)$ be the cost of powering the data center during the k^{th} interval. We can write

$$c_e(k) = \begin{cases} \alpha_e(k)\mathsf{p}(k) & \mathsf{p}(k) \leq \bar{\mathsf{p}}(k) \\ \alpha_e(k)\bar{\mathsf{p}}(k) + \beta_e(k)(\mathsf{p}(k) - \bar{\mathsf{p}}(k)) & \mathsf{p}(k) > \bar{\mathsf{p}}(k). \end{cases} \tag{6.1}$$

6.2.2 Revenue induced by executing jobs with a certain QoS.

To formulate SLA_G, we consider the approach discussed in the work of Zhu *et al.* [81]. Let $c_i^{j,\star}$ be the desired ratio between the total number of jobs in class j that are in the i^{th} zone during the k^{th} interval and, the rate at which jobs are executed. The revenue obtained by executing jobs in class j at the i^{th} zone can be written as

$$\mu_i^j(k) - (\lambda_i^j(k) + l_i^j(k))c_i^{j,\star}. \tag{6.2}$$

We rotate the terms in (6.2) so that we can reason in terms of *cost* of QoS. The total QoS cost obtained in the data center is

$$c_{Qos}(k) = \sum_{j=1}^{J}\sum_{i=1}^{N}\Big((l_i^j(k) + \lambda_i^j(k))c_i^{j,\star} - \mu_i^j(k)\Big). \tag{6.3}$$

6.2.3 Cost of migrating jobs.

In order to favorite control actions that reduce the amount of migrating jobs and also, to reduce the number of dropped jobs, we consider the following cost

$$c_{\delta s}(k) = \sum_{i=1}^{N}\sum_{j=1}^{J}\Big(\sum_{h=1}^{N}\delta_{i,h}^j(k)\Big) - \mathsf{c}_{s,i}^j s_i^j(k), \tag{6.4}$$

where the coefficients $\{\mathsf{c}_{s,i}^j\}$ are used for scaling purposes.

6.2.4 Data center cost function.

The expected total cost at time ν given the estimation at time k is

$$J(\nu|k) = \mathsf{c}_Q\hat{c}_{Qos}(\nu|k) + \hat{c}_e(\nu|k) + \mathsf{c}_{\delta,s}\hat{c}_{\delta s}(\nu|k). \tag{6.5}$$

The cost function (6.5) represents the opposite in sign of the performance metric we want to maximize in the data center. We call it the total data center cost.

The coordinated controller is able to perfectly represent the data center cost function in its optimization problem, i.e., the cost function used in the optimization problem of the coordinated controller is identical to the cost function discussed in (6.5). This, however, is not true for the uncoordinated controller. The uncoordinated controller is unable to predict the future total data center power consumption and hence, it cannot represent exactly the cost of powering the data center as represented in (6.5). The following notes discuss how the coordinated and the uncoordinated controller represent the desired cost function in their optimization problems.

6.3 Formulation of the uncoordinated and coordinated optimization problems

This section discusses the formulation of the uncoordinated of the coordinated optimization problems. With respect to the optimization problems discussed in the previous chapter, in this case we consider different cost functions and operating constraints.

Coordinated controller. At the beginning of every control interval, the coordinated controller solves the following optimization problem

$$
\begin{aligned}
&\min_{\mathcal{M},\mathcal{S},\mathcal{D},\mathcal{T}_{\text{ref}}} \sum_{\nu=k}^{\mathsf{T}+k} J(\nu|k) \\
\textbf{s.t.}& \\
&\text{for all } \nu = k,\dots,k+\mathsf{T} \\
&\text{computational network dynamics} \\
&\text{thermal network dynamics} \\
&\text{control constraints} \\
&\hat{\mathbf{p}}_{\mathcal{N}}(\nu|k) = \mathsf{A}_\alpha \hat{\boldsymbol{\eta}}(\nu|k) \\
&\hat{\boldsymbol{l}}(k|k) = \boldsymbol{l}(k), \\
&\hat{\boldsymbol{T}}_{\text{in}}(\nu+1|k) \le \overline{\boldsymbol{T}_{\text{in}}} \\
&\underline{\boldsymbol{T}_{\text{ref}}} \le \hat{\boldsymbol{T}}_{\text{ref}}(\nu|k) \le \overline{\boldsymbol{T}_{\text{ref}}} \\
&\hat{\boldsymbol{T}}_{\text{out}}(k|k) = \boldsymbol{T}_{\text{out}}(k).,
\end{aligned}
\tag{6.6}
$$

The objective function used in the coordinated controllers is identical to the run-time data center function cost defined in (6.5).

Uncoordinated controller. At every time k the uncoordinated controller solves two optimization problems. In the first step, the uncoordinated controller chooses a control action that minimizes the sum of the expected QoS cost, the server powering cost, and the cost of the chosen control action

$$
\begin{aligned}
&\min_{\mathcal{M},\mathcal{S},\mathcal{D}} \sum_{\nu=k}^{\mathsf{T}+k} J_{U,1}(\nu|k) \\
\textbf{s.t.}& \\
&\text{for all } \nu = k,\dots,k+\mathsf{T} \\
&\text{computational network dynamics} \\
&\text{control constraints} \\
&\hat{\mathbf{p}}_{\mathcal{N}}(\nu|k) = \mathsf{A}_\alpha \hat{\boldsymbol{\eta}}(\nu|k) \\
&\hat{\boldsymbol{l}}(k|k) = \boldsymbol{l}(k),
\end{aligned}
\tag{6.7}
$$

where

$$J_{U,1}(\nu|k) = \mathsf{c}_Q \hat{c}_{Qos}(\nu|k) + \mathsf{c}_{\delta,s} \hat{c}_{\delta s}(\nu|k) + \hat{\alpha}_e(\nu|k) \sum_{i=1}^{N} \hat{\mathsf{p}}_i(\nu|k). \tag{6.8}$$

In (6.8), the cost of powering the data center is approximated by $\hat{\alpha}_e(\nu|k) \sum_{i=1}^{N} \hat{\mathsf{p}}_i(\nu|k)$, i.e., the first optimization problem used by the uncoordinated controller only considers the discounted electricity cost.

In the second step, the uncoordinated controller, based on the solution obtained at the first step, solves the following optimization problem to select the best sequence of vectors $\{\hat{\boldsymbol{T}}_{\text{ref}}(h|k)\}$ that minimizes the expected cost of powering the CRAC nodes and that also enforces the thermal constraints

$$\begin{aligned}
&\min_{\mathcal{T}_{\text{ref}},} \sum_{\nu=k}^{\mathsf{T}+k} \hat{\alpha}_e(\nu|k) \sum_{i=N+1}^{N+C} \hat{\mathsf{p}}_i(\nu|k) \\
\textbf{s.t.}& \\
&\text{for all } \nu = k, \dots, k+\mathsf{T} \\
&\text{thermal network dynamics} \\
&\hat{\boldsymbol{T}}_{\text{in}}(\nu+1|k) \le \overline{\boldsymbol{T}_{\text{in}}} \\
&\underline{\boldsymbol{T}_{\text{ref}}} \le \hat{\boldsymbol{T}}_{\text{ref}}(\nu|k) \le \overline{\boldsymbol{T}_{\text{ref}}} \\
&\hat{\boldsymbol{T}}_{\text{out}}(k|k) = \boldsymbol{T}_{\text{out}}(k).
\end{aligned}$$

6.4 Parameters used in the simulations

The uncoordinated controller cannot compute the expected cost of powering the data center and hence it cannot leverage on SLA$_G$. In order to create a fair comparison between the actions chosen by the coordinated controller and the actions chosen by the uncoordinated controller, we consider a high cost of dropping jobs. In this case, the two algorithms are forced to processes almost every incoming job. In the simulation, the ratio between the number of dropped jobs and the number of jobs arrived at the data center during a control actions was less than $6.8 \cdot 10^{-5}$ for both of the controllers and no jobs were lost due to the migration actions. Two job classes are considered in the simulation. Job arrivals are subject

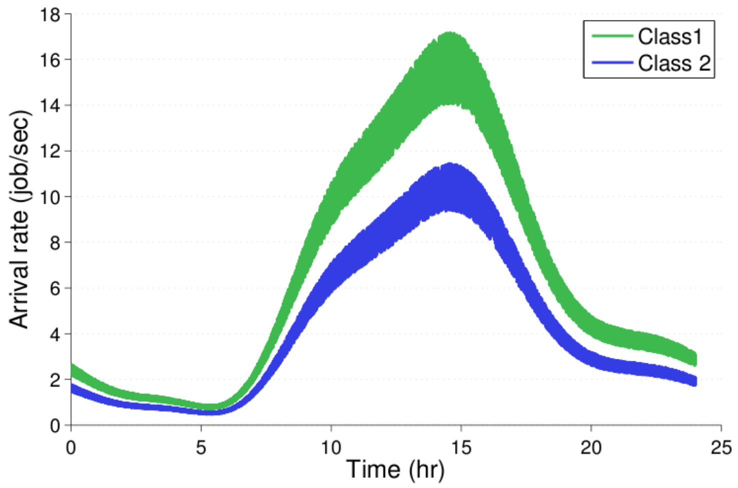

Figure 6.2: Job arrival rates at the data center.

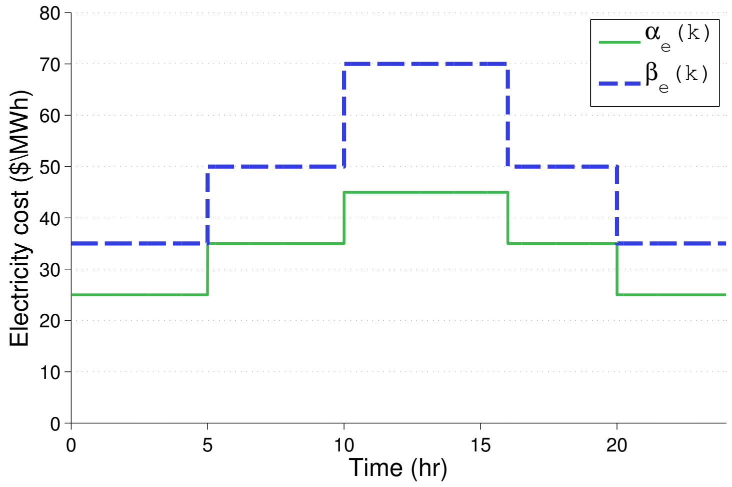

Figure 6.3: Electricity cost over time.

to a random noise uniformly distributed having zero mean and a variance proportional to the mean arrival rate. Job arrival rate at the data center is shown in Fig. 6.2. Job arrival rates represent a scaled version of the request rate arrived to an EPA server on Aug. 30th 1995.[3] The arrival rate values are scaled so that the peak arrival rate induce an average data center utilization of about 0.7. The cost of electricity over time is shown in Fig. 6.3.

The time-step for the simulation is 30 s and the controller closes the loop every 10 min. The optimization problem is formulated using a horizon of 6 steps (one hour).

[3]Source: The Internet Traffic Archive `http://ita.ee.lbl.gov/` .

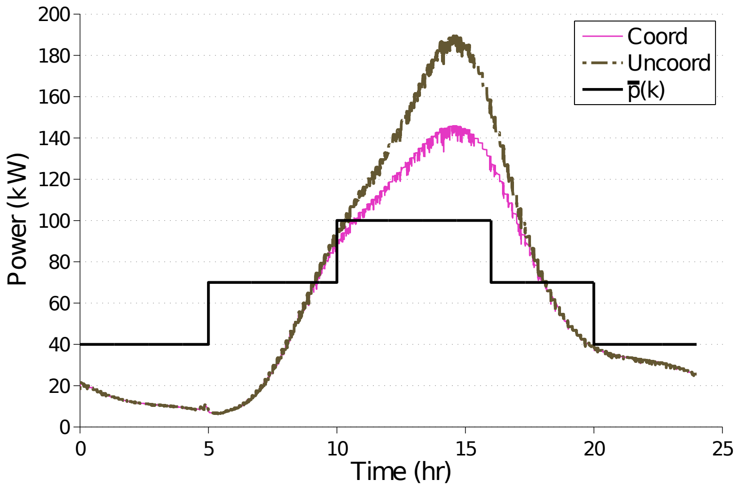

Figure 6.4: Total data center power consumption.

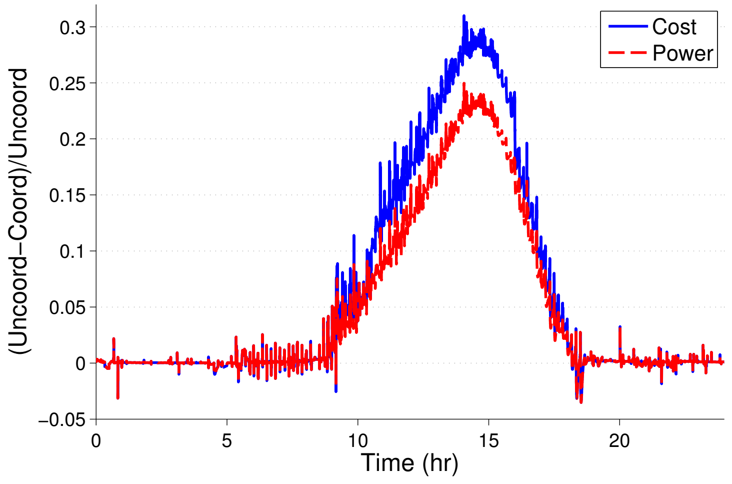

Figure 6.5: Relative difference of the cost of powering and of the total power consumption.

6.5 Comparison of the coordinated and the uncoordinated control strategy

Figure 6.4 shows the total data center power consumption obtained by the coordinated and by the uncoordinated controller. The power threshold value $\bar{\mathsf{p}}(k)$ is also shown in Fig. 6.4. The relative ratio between cost of powering the data center for the uncoordinated and the coordinated controller is shown in Fig. 6.5. The relative ratio between the power consumption of the data center when the two controllers are used is also shown in Fig. 6.5.

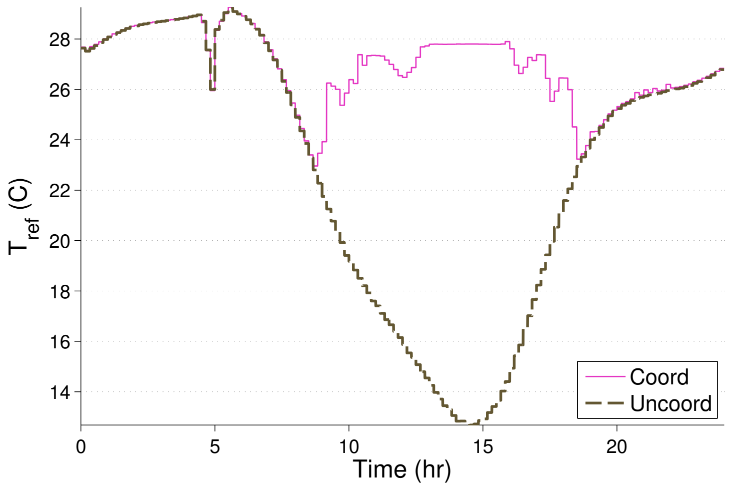

Figure 6.6: Average reference temperature.

Both controllers keep, on the average, the optimal ratio between the amount of hardware resources allocated and the amount of hardware resources required. The oscillations around the mean value had a standard deviation of 0.05. We expect that lower level controllers, not implemented in this simulation, can reduce the oscillation around the optimal value.

Figure 6.6 shows the average CRAC reference temperature for the coordinated and the uncoordinated controller case. Both controllers over-cool the servers just before the variation of the electricity cost at time 5 hr and they are able to reduce almost to zero the usage of CRAC unit for the following 30 min. Figure 6.7 shows the total power consumption of server and of the CRAC nodes obtained by the coordinated and the uncoordinated controllers. The coordinated controller, even though it obtains slightly higher values for the total server power consumption, is able to maintain a higher level of cooling efficiency. Therefore, it is able to largely reduce the power consumption of the CRAC units with respect to the uncoordinated controller. From Fig. 6.6 and 6.7 it can also be observed how the actions of the two controllers lead to the same server and CRAC power consumption as well as to the same average reference temperatures until the average arrival rate is lower than a certain threshold, e.g., just before time 10 hr and a little before time 20 hr. As discussed in the previous chapter, this particular behavior shows that the uncoordinated

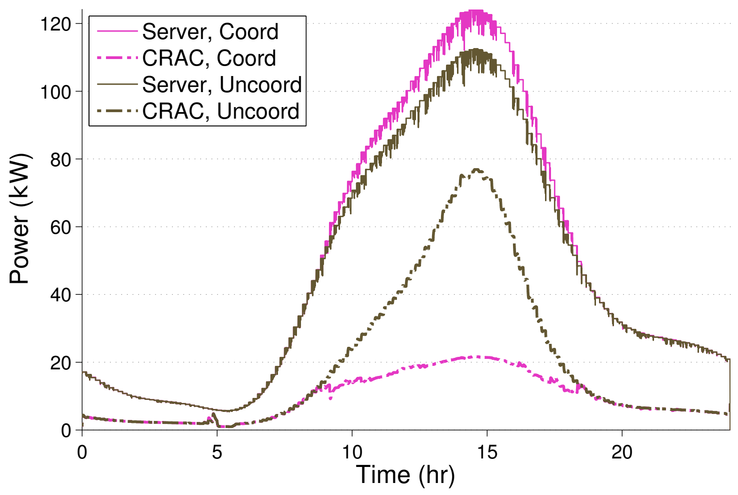

Figure 6.7: Power consumption values of server nodes and CRAC nodes.

policy is as optimal as the coordinated policy long as the average value of the arrival rate is reduced.

6.6 Comparison of two different coordinated control strategies

We consider now the case of two coordinated controllers. The first controller uses the same cost function discussed in the previous section. The first and the second coordinated controller consider the same kind of cost function, i.e., a combination of powering cost, of QoS cost, and of job migrating cost. However, the parameters used in the cost function of the second coordinated controller make it more favorable the drop of jobs.

Figure 6.8 shows the total data center power consumption when two coordinated controllers are used. The actions of the two controllers are different only when the cost of the current is over a certain value. Figure 6.9 shows the average percentage of jobs dropped over the time by the two controllers. We note that the second coordinated controller starts to drop jobs when the total power consumption of the data center exceeds the time-varying threshold. When the threshold value is so low that the data center cannot reduce its power consumption without incurring in a very high cost of QoS, it suddenly increases its power consumption and processes jobs at the maximum allowed rate. This behavior can be seen

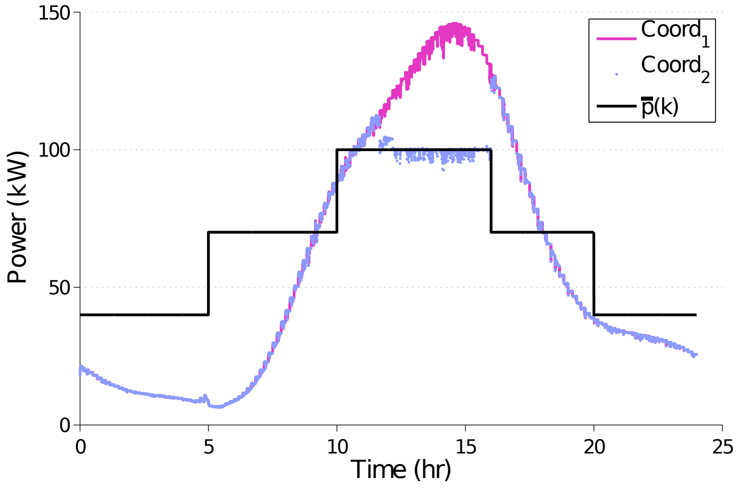

Figure 6.8: Total data center power consumption.

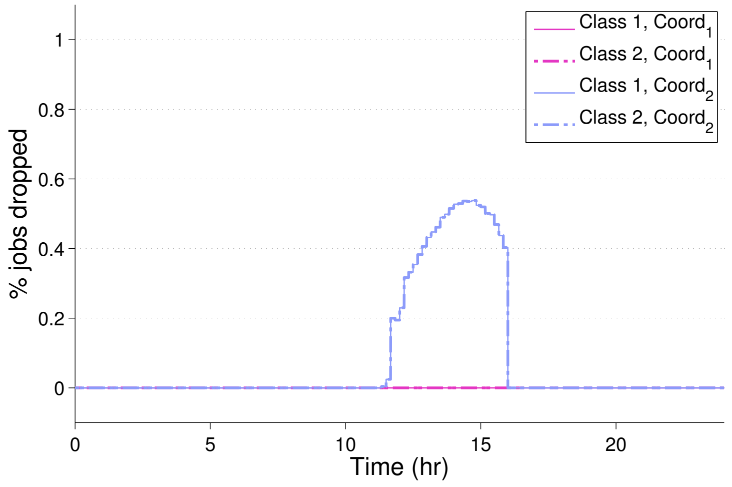

Figure 6.9: Relative percentage of dropped jobs.

in Fig. 6.8 around the 16 hr time.

The sudden increase in the total power consumption can lead to network imbalances and can generate large oscillations on the electricity price. Furthermore, the decision to cap the data center power consumption depends on both the electricity cost and on the predicted job arrival rate. This latter value is unknown to the power-grid and hence, the power-grid is unable to predict if and in which measure the data center will decide to cap its power consumption.

6.7 Summary

This chapter discusses a possible interaction between a data center and the power-grid. The high power consumption density of data centers motivates the study of possible interactions between data centers and the power-grid.

This chapter shows that data centers can take advantage of service level agreements (SLAs) with the power grid, but, depending on the formulation of the SLA, a data center may destabilize the power grid, rather than reducing the fluctuation of energy demand. Moreover, the chapter discusses how an uncoordinated controller can take advantage of a time-varying electricity price, even though it is unable to include the SLA with the power grid in the formulation of its optimization problem.

Chapter 7

Results on zone-level control strategies

The previous chapters assume that the zone level controllers are able to manage their zones so that the overall power consumption of every zone is proportional to the rate at which jobs are executed. Specifically, this implies that the zone level controllers turn off the unused servers of their zone. In this chapter we study when such an assumption is reasonable. We are interested in the analysis of the relationship between the variability of the job arrival rate and the time required to turn servers on. Results presented in this section are further discussed in the work of Aghajani *et al.* [2].

In this chapter, the goal of the proposed zone-level controller is the minimization of the power consumption of the zone. The QoS is given by the delay that a job arriving at the zone at the beginning of the k^{th} interval can expect. Bounding the maximum delay is equivalent to bound the minimum admissible QoS. For the sake of an uncluttered notation, we consider the case where all jobs belong to the same class, i.e., $J = 1$ and every server processes one job per sampling time.

The chapter is divided as follows. The following section discusses the model used by the proposed zone-level controller to predict the power consumption of the servers. Section 7.2 discusses the computational dynamics of the zone. Section 7.3 discusses the model used by the zone controller to predict the expected delay observer by jobs. Section 7.4 formulates the zone-level control algorithm. Section 7.5 discusses the stochastic model used to generate

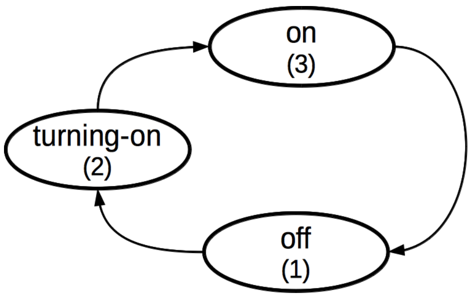

Figure 7.1: Graphical representation of the states and allowed transitions among states. In brackets we show the chosen order for the states.

time-varying job arrival rates. Section 7.6 discusses the simulation results obtained in this chapter. Finally, Sec. 7.7 summarizes the results obtained in this chapter.

7.1 Model of the server power consumption

Servers are modeled as a couple of computational and thermal nodes. However, the model used in this section focuses on the power consumption of the servers and it neglects their thermal evolution. With N we denote the number of servers in the zone. Servers have three states: ON, OFF, and TURNING-ON. Figure 7.1 shows the allowed transitions among the server states and the order chosen for the set of server states. The stable set of server states is composed by the states ON, OFF. According to the notation introduced in Sec. 3.6, we can write $\mathcal{S}_n = \{\text{ON, TURNING-ON, OFF}\}$ and $\bar{\mathcal{S}}_n = \{\text{ON, OFF}\}$.

We focus on the time required to move a server from the OFF state to the ON one and assume that all of the other transition times are negligible. Assume that the time required to move a server from the OFF state to the ON is multiple of the controller sampling time and let Δ_τ (s) be the zone level sampling time. For all $n \in \{1, \dots, N\}$, we can write

$$\zeta_n(i,j) = \begin{cases} T_s\Delta_\tau & \text{if } i = 1, j = 2 \\ 0 & \text{otherwise.} \end{cases} \tag{7.1}$$

We focus on a single hardware resource of the server and therefore $R = 1$. When servers are in the OFF or in the TURNING-ON state they cannot process any jobs. When servers are in the ON state, they process one job per sampling time. For all $n \in \{1, \dots, n\}$, we can write

$$\overline{\boldsymbol{r}}_n(\varphi_n(k)) = \begin{cases} 0 & \text{if } \varphi_n(k) = 1(\text{OFF}) \\ 0 & \text{if } \varphi_n(k) = 2(\text{TURNING-ON}) \\ \overline{\boldsymbol{r}} & \text{if } \varphi_n(k) = 3(\text{ON}), \end{cases} \tag{7.2}$$

where $\overline{\boldsymbol{r}} > 0$ and $\varphi_n(k)$ is the state of the n^{th} server during the k^{th} interval.

Servers consume no power when they are in the OFF state. The amount of power consumed in the TURNING-ON and in the ON states are equal. These conditions can be observed, for example, when the idle power consumption of a server equals the peak power consumption of the server. Also, these conditions correspond to a worst-case scenario for a zone-level controller that attempts to make the total zone power consumption proportional to the rate at which jobs are processed in the zone. The generic equation related to the server power consumption discussed in (3.15), can be specified as

$$\mathsf{p}_n(k) = \begin{cases} 0 & \text{if } \varphi_n(k) = 1(\text{OFF}) \\ \bar{\mathsf{p}} & \text{if } \varphi_n(k) = 2(\text{TURNING-ON}) \\ \bar{\mathsf{p}} & \text{if } \varphi_n(k) = 3(\text{ON}). \end{cases} \tag{7.3}$$

Let $c(k)$ denote the number of servers in the ON state at the beginning of the k^{th} interval and, let $c_{t,on}(k)$ denote the number of servers that are in the TURNING-ON state at the beginning of the k^{th} interval. The total power consumption of the zone is given by

$$\mathsf{p}(k) = \bar{\mathsf{p}} c_{t,on(k)} + \bar{\mathsf{p}} c(k). \tag{7.4}$$

7.2 Computational dynamics of the zone

With $l(k)$ we denote the total number of jobs in the zone at the beginning of the k^{th} interval and with $\lambda(k)$ we denote the number of jobs arriving at the zone during the k^{th} interval. Servers process one job per sampling time and no jobs are migrated among the servers. In this case, the equation that governs the evolution of $l(k)$ can be written as

$$l(k+1) = l(k) + \lambda(k) - \min\Big\{l(k),\, c(k)\Big\}. \tag{7.5}$$

The dynamic model in (7.5) corresponds to the dynamic model discussed in (4.23), where we set the total departure rate (over discrete time) of the zone equal to $\min\{l(k),\, c(k)\}$.

7.3 Job expected delay

Servers process exactly one job per sampling interval time. Therefore, it is possible to model every server as a $G/D/1$ queue, i.e., a queue where the arrival process is described by a generic distribution, the service time is deterministic, and only one server is available in the queue [28].

Consider the case where the number of servers in the ON state is constant and large enough so that every job is executed in a finite amount of time with probability 1. Assume also that the distribution of job arrival is stationary. With c we denote the number of servers in the ON state. The whole zone can be modeled as a $G/D/c$ queue and the distribution of jobs in the zone is stationary and its probability generating function $\mathcal{L}(z)$ (PGF) is

$$\mathcal{L}(z) = c\left(1 - \frac{\lambda}{c}\right)\frac{(z-1)\Lambda(z)}{z^c - \Lambda(z)}\prod_{i=1}^{c-1}\frac{z - z_i}{1 - z_i}, \tag{7.6}$$

where $\Lambda(z)$ is PGF of the distribution of job arrival rate, λ is the expected rate at which jobs arrive, and $\{z_i\}$ are the zeros of $z^c - \Lambda(z)$ inside the unit disk $\{z : |z| < 1\}$, [27, 28].

From (7.6), we can derive the expected number of jobs in the zone as

$$\bar{l} \triangleq \mathcal{L}'(1) = \lambda - \frac{(c-1) - \rho^2 c}{2(1-\rho)} + \sum_{i=1}^{c-1} \frac{1}{1-z_i}. \tag{7.7}$$

where we defined $\rho \triangleq \frac{\lambda}{c}$. Given the expected number of jobs in the zone, via the *Little's* theorem, we can compute the average delay that every jobs observe

$$\bar{D} = \frac{\bar{l}}{\lambda} = 1 - \frac{1}{\lambda}\frac{(c-1) - \rho^2 c}{2(1-\rho)} + \frac{1}{\lambda}\sum_{i=1}^{c-1} \frac{1}{1-z_i}. \tag{7.8}$$

7.4 Zone control algorithm

The value of the couple $(\bar{l},\, \bar{D})$ is computed under the hypothesis that the job arrival rate is constant. In the case studied in this section however, the job arrival rate changes over time. We denote with $\bar{l}(k)$ the expected number of jobs in the zone and with $\bar{D}(k)$ the expected job delay when $\lambda(k)$ $c(k)$ are used in place of λ and c in (7.8).

We assume that at every time k, the zone-level controller has the knowledge about the expected rate at which jobs will arrive over the next T_s time steps. For example, the data center controller can send this information to the zone-level controller. In [2] an algorithm for the data center level controller, for predicting future job arrival rates of every zone, is discussed. Let $\hat{\lambda}(k+T_s|k)$ be the expected job arrival rate to the zone T_s steps ahead. With $\hat{l}(k+T_s|k)$ we denote the expected number of jobs in the zone computed via (7.8), when the job arrival rate is $\hat{\lambda}(k+T_s|k)$ and the number of servers in the ON state is $\hat{c}(k+T_s|k)$. At the beginning of every control interval, the zone level controller finds the minimum number of servers that should be in the ON state, so that the expected average delay is lower than the given bound. With $\hat{c}^\star(k+T_s|k)$ we denote the optimal number of servers in the ON state computed by the zone controller at time k and with $\hat{l}^\star(k+T_s|k)$ we denote the optimal number of jobs in the zone computed by the zone controller at time k.

The number of servers that should be turned on or off at time k is computed via the following feed-back law

$$u(k) = \Big\lfloor \alpha_c \Big(\hat{c}^{\star}(k + T_s|k) - c(k) \Big) + \beta_l \Big(\hat{l}^{\star}(k + T_s|k) - l(k) \Big) \Big\rfloor , \tag{7.9}$$

where α_c and β_l are coefficients that can be appropriately tuned and $\lfloor \cdot \rfloor$ is the floor operator. In (7.9), $u(k)$ is the switching command, i.e., the number of servers which should be turned on or off, according to the sign of $u(k)$. The actual number of servers that will be turned on and off depends on the number of available servers in the zone. For example, if all of the servers are already in the ON state and $u(k) = 5$, no additional servers will be turned on.

7.5 Modeling the process of job arrival rate

In order to model the variability of the job arrival rate, we represent the process generating the job arrivals via a discrete-time Markov Modulated Poisson Process (MMPP). The discrete-time MMPP consists of a finite-state discrete-time Markov chain with a transition matrix P and a set of arrival rates $\{\lambda_1, \ldots, \lambda_n\}$. Let i be the state of the Markov chain during the k^{th} time slot. Then, the distribution of jobs arriving at the zone during the same slot has a Poisson distribution with rate λ_i. In the following, we study the performance of the zone-level controller under different values of the variability of the expected job arrival rate. The strength of the proposed model for the job arrival rate resides in the fact that the expected time over which the job arrival rate is constant can be computed in a closed form.

The change in the expected job arrival rate can be measured by the mean run time T_λ of the Markov chain, i.e., the expected time that the Markov chain stays in the same state before jumping to another. In order to compute T_λ we first have to compute the steady-state distribution of the Markov chain.

Let $p_i(k)$ be the probability that the chain is in the i^{th} state during the k^{th} interval. The $\{p_i(k)\}$ values can be collected in the $n \times 1$ vector $\boldsymbol{p}(k) \triangleq \begin{bmatrix} p_1(k) & \dots & p_n(k) \end{bmatrix}^T$. Given the knowledge of the vector $\boldsymbol{p}(k)$, the probability of the Markov chain of being in a certain state j at the beginning of the $(k+1)^{th}$ interval is $\sum_{i=1}^{N} p_i(k) P_{ij}$. Using vectors, we can write

$$\boldsymbol{p}(k+1)^T = \boldsymbol{p}(k)^T P. \tag{7.10}$$

Let us assume that the Markov chain has a stationary distribution and that it is unique. Then, the stationary distribution can be computed as $\boldsymbol{p}^T = \boldsymbol{p}^T P$.

Assume that at the beginning of the k^{th} interval, the chain is on the j^{th} state. The number of successive time steps that the Markov chain rate stays in the same state has a geometric distribution with parameter $1 - P_{jj}$. The mean run time T_{λ_j} for the j^{th} state is

$$\frac{1}{1 - P_{jj}.}$$

Taking the average over all of the states, we have that the mean run time T_λ of the Markov chain is

$$T_\lambda \triangleq \sum_{j=1}^{M} p_j T_{\lambda_j} = \sum_{j=1}^{M} \frac{p_j}{1 - p_{jj}}, \tag{7.11}$$

where p_j is the j^{th} element of the steady-state distribution $\boldsymbol{p}$ of the Markov chain.

As the value of T_λ increases, the Markov chain tends to stay more on the same state and therefore, the variability of the job arrival rate is reduced. When T_λ decreases, the Markov chain tends to move from one state to another at the beginning of every interval and therefore, the rate at which jobs arrive at the zone tends to be more unsteady.

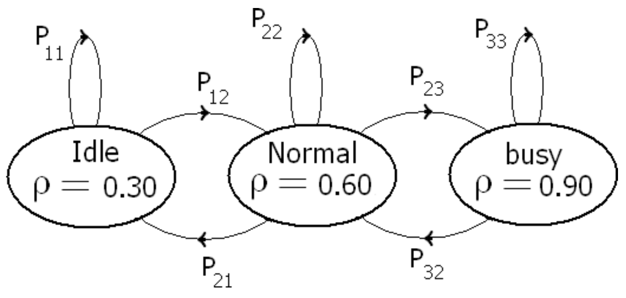

Figure 7.2: Markov chain rate in incoming job MMPP model.

7.6 Simulation Results

In simulation we compare the performance of the proposed zone control algorithm against one that fixes the number of active servers. The fixed number of servers is optimal for the long-term average job arrival rate. Figure 7.2 shows the normalized arrival rates of the Markov chain under different states. As ρ tends to 1, the job arrival rate tends to the maximum rate at which jobs can be processed, per sampling time, by the zone. The Markov chain rate has three states corresponding to $\rho = 0.3$, $\rho = 0.6$, and $\rho = 0.9$. The maximum allowed expected delay is 3 sampling time.

Figure 7.3 shows mean delay time obtained by the zone algorithm for different values of the ratio of $\frac{T_s}{T_\lambda}$. As $\frac{T_s}{T_\lambda}$ increases, then servers take more time to turn on with respect to the average time over which the job arrival rate can be expected to be constant. As $\frac{T_s}{T_\lambda}$ tends to 0, then the time to turn a server on becomes negligible with respect to the variability of the job arrival rate. As depicted in Fig. 7.3, the mean delay is successfully kept below the threshold approximately for $\frac{T_s}{T_\lambda} < 0.5$, but the algorithm fails for larger ratio values as predicted above.

The improvement in the average power consumption provided by the dynamic algorithm is shown in Fig. 7.4. The proposed dynamic power allocation algorithm offers the same mean delay time with 30% less consumed average power, when the incoming rate changes are sufficiently slower than the dynamics of the servers ($\frac{T_s}{T_\lambda} \ll 1$). The energy consumption decreases as the workload rate changes becomes faster and finally for very abrupt changes

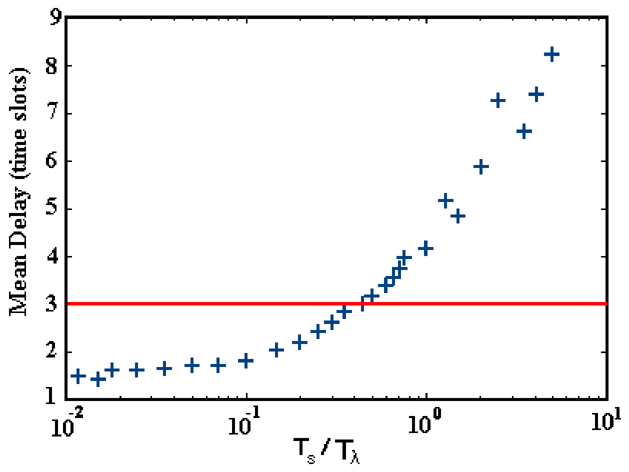

Figure 7.3: Mean delay time under dynamic power allocation for different values of $\frac{T_s}{\tau}$ ratio. The allowable delay time is set to be 3 sampling time.

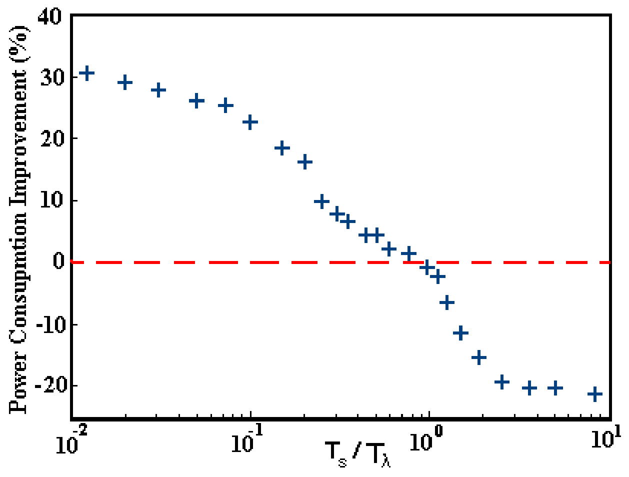

Figure 7.4: Improvement in total energy consumption made by proposed dynamic algorithm comparing to optimal static algorithm for different values of $\frac{T_s}{T_\lambda}$ ratio. when the same QoS level is attained.

in arrival rate, i.e., $\frac{T_s}{T_\lambda} \gg 1$, the static power allocation strategy might be preferable.

The behavior of the system can be explained by considering the energy saved when turning off idle servers and the energy spent by starting up servers which cannot immediately process jobs. When the changes on the arrival job rate are slow compared to the server start-up time, then the saved energy is dominant so the dynamic power allocation algorithm provides better performance than the static one. On the other hand, when the changes on the arrival job rate are on the same scale of the server start-up time, the energy spent on starting up servers is dominant and this causes the performance of the dynamic algorithm to degrade considerably compared to its static counterpart.

7.7 Summary

This chapter discusses a possible controller for the zone level. The goal of the controller is the minimization of the total zone power consumption subject to the constraints on the maximum expected job delay.

The results discussed in this chapter show that when the workload arrival rate changes too quickly, then keeping servers always active is the best strategy. The meaning of “too quickly” is related to the ratio between time required to turn servers on and the average time over which job arrival rate can be considered constant.

Chapter 8

Effects of model mismatches on the hierarchical control of data centers

In this chapter we study the effects of using wrong aggregate models of the zones at the data center level. In order to maintain the problem manageable, we consider here a simplified case where a data center controller provides set-points only about the amount of resources that every zone should allocate to the job execution and the amount of cooling that each CRAC unit should provide. Control of the job scheduling and of the job migration are neglected. Results presented in this chapter are discussed with further details in [48].

Three optimal control problems are discussed. The first optimal control problem is related to the management of the data center, without the use of aggregated models or of a hierarchical control strategy. This optimal control problem uses a prediction horizon of one step. We call this problem the *original optimization problem.* The second optimization problem is related to the data center level control. This problem only considers the effect of the data center controller, neglecting the effects of the zone level controllers. We call this problem the *data center optimization problem.* The third optimization problem is related to the use of both the data center and the zone level controllers. In this case, the effects of the zone controllers, acting according to the set-points defined by the data center controller, are considered. When the data center controller and the zone controllers operate together, multiple optimization problems have to be solved. We call this set of optimization problems the *hierarchical optimization problems*.

The performance of the different controllers is measured via the value of the cost function they obtain. Higher values of the cost function implies lower performance. The performance obtained by the original optimization problem provides us an upper bound on the performance we can expect from the hierarchical control approach. Considering the performance of the data center controller with and without the effects of zone level controllers allows us to better understand the impact of lower level controllers in the overall hierarchical.

This chapter is divided as follows. The following subsection discusses the simplified data center model. Subsection 8.2 formulates the original optimization problem. Subsection 8.3 formulates the data center optimization problem and the hierarchical optimization problems. Finally, subsection 8.4 discusses the effects of wrong model aggregation to the performance of the data center controller and to the performance of the whole hierarchy.

8.1 Data center model

Every server is represented by a couple of computational node and thermal server node. No environment nodes are considered. Every rack is a zone and the number of zone (racks) is denoted with R. With S we denote the total number of racks and CRAC units, i.e., $\mathsf{S} = \mathsf{R} + C$. The number of servers in the data center is N. The total number of thermal nodes is $M = N+C$. Note however, that the data center controller only manages $\mathsf{S} = R+C$ thermal nodes. With $\mathcal{R}_i$ we denote the set of server indexes which are located in the i^{th} rack.

As QoS metric, we consider the average job sojourn time of the jobs.[1] The goal of this section is to discuss how, under certain hypothesis, the dynamic of the data center

[1]The job sojourn time is defined as the difference between the time when a job arrives at a server and the time when it leaves the data center.

can be described by a collection of linear subsystems interacting with each other via linear relationships. Although we discuss a computational and a thermal model of servers, in this section the focus is on the thermal interaction among servers and CRAC units.

8.1.1 Server thermal model.

A discrete-time version of the thermal server model discussed in Sec. 4.3.1 is considered here. The evolution of the output temperature of the i^{th} server can be written as

$$T_{\text{out},i}(k+1) = (1 - \mathsf{k}_i)T_{\text{out},i}(k) + \mathsf{k}_i T_{\text{in},i}(k) + \mathsf{c}_i \mathsf{p}_i(k). \tag{8.1}$$

where k_i is the (discrete-time) thermal coefficient of the i^{th} server.

8.1.2 Thermal model of CRAC units.

We consider a discrete-time model for CRAC units similar to the one discussed in Sec. 4.3.2. Under the assumption that the reference temperature is always smaller than or equal to the input temperature of the CRAC unit, we can write

$$\begin{cases} T_{\text{out},i+N}(k+1) = (1 - \mathsf{k}_{i+N})T_{\text{out},i+N}(k) + \mathsf{k}_{i+N} T_{\text{in},i+N}(k) + \mathsf{k}_{i+N}\Delta T_{\text{ref},i}(k) \\ \Delta T_{\text{ref},i}(k) \leq 0, \end{cases} \tag{8.2}$$

where $\Delta T_{\text{ref},i}(k) = T_{\text{ref},i}(k) - T_{\text{in},i+N}(k)$ is a fictitious reference signal that requires the collocated CRAC controller to keep the supplied air temperature $\Delta T_{\text{ref},i}(k)$ below the CRAC inlet air temperature. The model in (8.2) is a discrete version of the model showed in (4.3), but we used a slightly different notation here. The notation was changed to simplify the formulation of the three control problems.

8.1.3 Heat exchange model.

According to the notation earlier introduced, the vectors $\boldsymbol{T}_{\text{in}\mathcal{R}_i}(k)$ and $\boldsymbol{T}_{\text{out}\mathcal{R}_i}(k)$ represent the vector of the input temperatures and the vector of the output temperatures in the i^{th}

rack respectively. A reduced order version of the vector $\boldsymbol{T}_{\text{in}\mathcal{R}_i}(k)$ and of the vector $\boldsymbol{T}_{\text{out}\mathcal{R}_i}(k)$ is also considered. With $\tilde{\boldsymbol{T}}_{\text{in},\mathcal{R}_i}(k)$ we denote the reduced order version of the vector $\boldsymbol{T}_{\text{in}\mathcal{R}_i}(k)$ and with $\tilde{\boldsymbol{T}}_{\text{out},\mathcal{R}_i}(k)$ we denote a reduced order version of the vector $\boldsymbol{T}_{\text{out}\mathcal{R}_i}(k)$. For example, the two vectors can represent the measured temperatures at the bottom, middle, and top level of the inlet and of the outlet of the i^{th} rack. The reduced order versions of the vectors $\boldsymbol{T}_{\text{in}\mathcal{R}_i}(k)$ and $\boldsymbol{T}_{\text{out}\mathcal{R}_i}(k)$ can be computed via

$$\tilde{\boldsymbol{T}}_{\text{in},\mathcal{R}_i}(k) = G_{in,i}\boldsymbol{T}_{\text{in}\mathcal{R}_i}(k), \tag{8.3}$$

$$\tilde{\boldsymbol{T}}_{\text{out},\mathcal{R}_i}(k) = G_{out,i}\boldsymbol{T}_{\text{out}\mathcal{R}_i}(k), \tag{8.4}$$

where the matrices $G_{in,i}$ and $G_{out,i}$ are full row rank. The elements of the matrices $G_{in,i}$ and $G_{out,i}$ depend on the position of the temperature sensors on the rack and on the rate at which air flows into every server of the rack.

The input temperature constraints can now be written as

$$\tilde{\boldsymbol{T}}_{\text{in},\mathcal{R}_i}(k) \leq \overline{\tilde{\boldsymbol{T}}_{\text{in},\mathcal{R}_i}}, \tag{8.5}$$

where $\overline{\tilde{\boldsymbol{T}}_{\text{in},\mathcal{R}_i}}$ is the upper bound for the temperature vector of the i^{th} rack.

Let v be the rack index where the i^{th} server is located. The relationship between the input temperature of the i^{th} server and the output temperatures of all other servers and CRAC units is approximated as

$$T_{\text{in},i}(k) = \sum_{j\in\mathcal{R}_v} \psi_{i,j}T_{\text{out},j}(k) + \sum_{j\in\mathcal{C}} \psi_{i,j}T_{\text{out},j}(k) + \sum_{\substack{j=1\\ j\neq v}}^{\mathsf{R}} \tilde{\boldsymbol{\psi}}^T_{i,\mathcal{R}_j}\tilde{\boldsymbol{T}}_{\text{out},\mathcal{R}_j}(k), \tag{8.6}$$

where $\tilde{\boldsymbol{\psi}}_{i,\mathcal{R}_j}$ is the vector which represents the relative global effect of the j^{th} rack on the i^{th} server. In (8.6), we approximate the effects of the output temperature of all of the servers in the other racks with the effects of the sole reduced order output temperature vector of the racks. The possibility to represent the effects of the output temperature of the servers located in other racks, with the aggregated output temperature of the rack and, the possibility to enforce the thermal constraints only on the reduced order input temperature vectors, are two key assumptions in this chapter.

8.1.4 Computational model of servers.

In this section we use the variables $\{\rho_i^j(k)\}$ to represent the relative desired job execution rate of jobs in class j of the i^{th} server at time k.[2] The introduction of the variables $\rho_i^j(k)$ allows us to simplify the notation of the three optimization problems. The desired job execution rate of jobs in class j of the i^{th} server at time k is written as $\mu_i^j(k) = \rho_i^j(k)\overline{\mu}_i^j$. For all $i \in \{1, \ldots, N\}$, $j \in \{1, \ldots, J\}$ and for all $k \in \mathbb{Z}$

$$\rho_i^j(k) \in [0,1], \qquad \sum_{j=1}^{J} \rho_i^j(k) \leq 1.$$

The controller considered in this section approximates the average job sojourn time with the difference, at every time k, between $\mu_i^j(k)$ and $\lambda_i^j(k)$. The idea behind the proposed approximation stems from the analysis of the expected sojourn time in the M/M/1 queuing systems: when the expected service rate μ is larger than the expected arrival rate λ, then the expected sojourn time is given by $(\mu - \lambda)^{-1}$ and it equals the long-run average sojourn time of jobs in the queue. In such a case, minimizing the average sojourn time is equivalent to maximize the difference between μ and λ.

Let $c_{q,i}^j(k)$ be the QoS cost obtained at the i^{th} server during the k^{th} interval for jobs in class j, that is,

$$c_{q,i}^j(k) = \mathsf{c}_{q,i}^j(\lambda_i^j(k) - \bar{\mu}_i^j \rho_i^j(k)), \tag{8.7}$$

[2]The variables $\{\rho_i^j(k)\}$ defined in this section, are different from the variables $\boldsymbol{\rho}_i(\tau)$ defined in Sec.3.3.2.

where $\mathsf{c}_{q,i}^{j}$ is a non-negative constant. The total data center QoS cost is defined as

$$c_q(k) = \sum_{i=1}^{N}\sum_{j=1}^{J} c_{q,i}^{j}(k) = \mathbf{c}_q^T(\boldsymbol{\lambda}(k) - diag\{\bar{\boldsymbol{\mu}}\}\boldsymbol{\rho}(k)), \tag{8.8}$$

where $\mathbf{c}_q^T$ is the vector that collects the $\mathsf{c}_{q,i}^{j}$ coefficients and

$$\bar{\boldsymbol{\mu}} = \begin{bmatrix} \bar{\mu}_1^1 & \dots & \bar{\mu}_1^J & \bar{\mu}_2^1 & \dots & \dots & \bar{\mu}_N^J \end{bmatrix}^T.$$

We assume the average power consumed by a the i^{th} server during the k^{th} interval follows a quadratic relationship between the power consumed by each job type and the arrival rate for each job type

$$\mathsf{p}_i(k) = \boldsymbol{\lambda}_i^T(k)\mathsf{C}_i\boldsymbol{\rho}_i(k), \tag{8.9}$$

where $\boldsymbol{\rho_i}(k) = \begin{bmatrix} \rho_i^1(k) & \dots & \rho_i^J(k) \end{bmatrix}^T$, $\boldsymbol{\lambda}_i(k) = \begin{bmatrix} \lambda_i^1(k) & \dots & \lambda_i^J(k) \end{bmatrix}^T$, and C_i is a $J \times J$ positive-definite matrix.

8.2 Formulation of the original data control problem

In order to simplify the formulation of the original data control problem, we introduce at first a common notation for the thermal evolution of servers and CRAC units. Each rack and each CRAC unit is represented as a subsystem of the data center. With $\boldsymbol{x}_i(k)$ we denote the state of the i^{th} subsystem and with $\boldsymbol{z}_i(k)$ we denote the output of the i^{th} subsystem. For $i = 1, \dots, \mathsf{R}$ (racks) $\boldsymbol{x}_i(k) = \boldsymbol{T}_{\text{out}\mathcal{R}_i}(k)$ and $\boldsymbol{z}_i(k) = \tilde{\boldsymbol{T}}_{\text{out},\mathcal{R}_i}(k)$, whereas for $i = \mathsf{R}+1, \dots, \mathsf{R}+C$ (CRACs), $\boldsymbol{x}_i(k) = T_{\text{out},i}(k)$ and $\boldsymbol{z}_i(k) = T_{\text{out},i}(k)$. The dynamics of both servers and CRAC units can now be written as

$$\begin{cases} \boldsymbol{x}_i(k+1) &= A_i\boldsymbol{x}_i(k) + B_i(k)\boldsymbol{u}_i(k) + \sum\limits_{\substack{j=1 \\ j\neq i}}^{\mathsf{R}+C} B_{i,j}\boldsymbol{z}_j(k) \\ \boldsymbol{z}_i(k) &= G_i\boldsymbol{x}_i(k), \end{cases} \tag{8.10}$$

where for $i = 1, \dots, \mathsf{R}$

$$A_i = I - diag\{\mathbf{k}_{\mathcal{R}_i}\} + diag\{\mathbf{k}_{\mathcal{R}_i}\}\Psi_{[\mathcal{R}_i,\mathcal{R}_i]}, \tag{8.11}$$

$$B_i(k) = diag_B\{\mathsf{c}_{\mathsf{p},i}\boldsymbol{\lambda}_i^T(k)\mathsf{C}_i\}, \tag{8.12}$$

$$B_{i,j} = diag\{\mathbf{k}_{\mathcal{R}_i}\}\tilde{\Psi}_{[\mathcal{R}_i,\mathcal{R}_j]}, \tag{8.13}$$

$$\boldsymbol{u}_i(k) = \begin{bmatrix} \boldsymbol{\rho}_{i_1}(k)^T & \dots & \boldsymbol{\rho}_{i_{n_i}}(k)^T \end{bmatrix}^T, \tag{8.14}$$

$$G_i = G_{out,i}, \tag{8.15}$$

and for $i = \mathsf{R}+1, \dots, \mathsf{R}+C$, we have

$$A_i = 1 - \mathsf{k}_i(1 - \psi_{i,i}), \tag{8.16}$$

$$B_i(k) = \mathsf{k}_i, \tag{8.17}$$

$$B_{i,j} = \mathsf{k}_i diag\{\tilde{\boldsymbol{\psi}}_{i,\mathcal{R}_j}\} \qquad \text{for all } j \in \{1, \dots, \mathsf{R}\}, \tag{8.18}$$

$$B_{i,j} = \mathsf{k}_i \psi_{i,j} \qquad \text{for all } j \in \{\mathsf{R}+1, \dots, \mathsf{S}\}, \tag{8.19}$$

$$\boldsymbol{u}_i(k) = \Delta T_{\text{ref},i}(k), \tag{8.20}$$

$$G_i = 1. \tag{8.21}$$

In the above equation, the matrix $\tilde{\Psi}_{[\mathcal{R}_i,\mathcal{R}_j]}$ is obtained by superimposing the vectors $\{\tilde{\boldsymbol{\psi}}_{i,\mathcal{R}_j}^T\}$. We can observe in this new form the data center is represented by a network of subsystems, where the output of every system (the variables $\{\boldsymbol{z}_i(k)\}$) is an input to the other subsystems.

8.2.1 Cost function.

We now want to rewrite the cost function with the newly defined variables. Let $\mathsf{c}_e(k)$ represent the average electricity cost over the k interval. The total server electricity cost is given by

$$c_{\mathsf{p}}(k) = \mathsf{c}_e(k) \sum_{i=1}^{N} \mathsf{p}_i(k) = \boldsymbol{e}(k)^T \boldsymbol{\rho}(k), \tag{8.22}$$

where c_e is the electricity cost and $\boldsymbol{e}(k)$ is a time-varying vector which includes also the effects of the job arrival rate on the server power consumption.

The power consumption of a CRAC unit is in general a nonlinear function of CRAC inlet and outlet temperatures and reflects the fact that power efficiency increases as the outlet air temperature increases [44]. In order to force the CRAC units to keep their outlet temperature at the highest value that does not violate the temperature constraints of servers, we consider the following cost

$$c_{T_{\text{ref}}}(k) = \mathbf{c}_{T_{\text{ref}}}^T \Delta \boldsymbol{T}_{\text{ref}}, \tag{8.23}$$

where $\mathbf{c}_{T_{\text{ref}}}^T \leq 0$.

The total data center operating cost can now be expressed as

$$\begin{aligned} c_q(k) + c_{\mathsf{p}}(k) + c_{T_{\text{ref}}}(k) = \mathbf{c}_q^T \boldsymbol{\lambda}(k) - \boldsymbol{c}_q^T diag\{\overline{\boldsymbol{\mu}}\} \boldsymbol{\rho}(k) + \mathbf{e}^T(k) \boldsymbol{\rho}(k) + \\ \boldsymbol{c}_{T_{\text{ref}}}^T(k) \Delta \boldsymbol{T}_{\text{ref}}(k). \end{aligned} \tag{8.24}$$

8.2.2 Control constraints.

Constraints on each of the $\boldsymbol{u}_i(k)$ variables can be written as

$$0 \leq \boldsymbol{u}_i(k) \leq 1, \quad i = 1, \ldots, \mathsf{R}, \tag{8.25}$$

$$\mathbf{1}^T \boldsymbol{u}_i(k) \leq 1, \quad i = 1, \dots, \mathsf{R}, \tag{8.26}$$

$$\boldsymbol{u}_i(k) \leq 0, \quad i = \mathsf{R}+1, \dots, \mathsf{R}+C. \tag{8.27}$$

8.2.3 Thermal constraints.

For all $i = 1, \dots, \mathsf{R}$, constraints on the vector of the rack inlet air temperatures can be written as

$$G_i \Psi_{[\mathcal{R}_i, \mathcal{R}_i]} \boldsymbol{x}_i(k) + G_i \sum_{\substack{j=1 \\ j \neq i}}^{\mathsf{R}+C} B_{i,j} \boldsymbol{z}_j(k) \leq \overline{\boldsymbol{T}_{\text{in}\mathcal{R}_i}}. \tag{8.28}$$

8.2.4 The original data center control problem

Let T be the horizon over which we consider the evolution of the data center. The optimization problem we want to solve is

$$\begin{aligned}
&\min_{\mathcal{U}_{1,\mathsf{T}}} \sum_{i=1}^{\mathsf{S}} \sum_{h=0}^{\mathsf{T}-1} \boldsymbol{c}_{i,u}^T(k) \hat{\boldsymbol{u}}_i(k+h|k) \\
&\textbf{s.t.} \\
&\hat{\boldsymbol{x}}_i(k+h+1|k) = A_i \hat{\boldsymbol{x}}_i(k+h|k) + \\
&\qquad\qquad B_i(k+h) \hat{\boldsymbol{u}}_i(k+h|k) + \\
&\qquad\qquad \textstyle\sum_{\substack{j=1 \\ j \neq i}}^{\mathcal{S}} B_{i,j} \hat{\boldsymbol{z}}_j(k+h|k) && j = 0, \dots, \mathsf{T}-1,\ i = 1, \dots, \mathsf{S} \\
&\hat{\boldsymbol{z}}_i(k+h|k) = G_i \hat{\boldsymbol{x}}_i(k+h|k) && h = 0, \dots, \mathsf{T}-1,\ i = 1, \dots, \mathsf{S} \\
&\underline{\boldsymbol{u}}_i \leq \hat{\boldsymbol{u}}_i(k+h|k) \leq \overline{\boldsymbol{u}}_i, && h = 0, \dots, \mathsf{T}-1,\ i = 1, \dots, \mathsf{S} \\
&H_i \hat{\boldsymbol{u}}_i(k+h|k) \leq 1, && h = 0, \dots, \mathsf{T}-1,\ i = 1, \dots, \mathsf{S} \\
&G_i \Psi_{[\mathcal{R}_i],[\mathcal{R}_i]} \hat{\boldsymbol{x}}_i(k+h|k) + \\
&\qquad G_i \sum_{\substack{j=1 \\ j \neq i}}^{\mathsf{R}+C} B_{i,j} \hat{\boldsymbol{z}}_j(k+h|k) \leq \overline{\boldsymbol{T}_{\text{in}\mathcal{R}_i}} && h = 1, \dots, \mathsf{T},\ i = 1, \dots, \mathsf{S},
\end{aligned} \tag{8.29}$$

where $\mathcal{U}_{1,\mathsf{T}} = \left\{ \hat{\boldsymbol{u}}_1(k|k), \dots, \hat{\boldsymbol{u}}_S(k+\mathsf{T}-1|k) \right\}$.

We focus here on the case where optimization problem uses a time horizon of only one step. Although it may seem restrictive, optimizing only over the one step ahead prediction can be an appropriate solution when the predictive values of future job arrival rate have a large variability and hence the predictive cost values have weak relevance compared to the current estimated one.

The *original optimization problem* is

$$\begin{aligned}
&\min_{\mathcal{U}_1} \sum_{i=1}^{\mathsf{S}} \boldsymbol{c}_{i,u}^T(k)\hat{\boldsymbol{u}}_i(k|k) \\
&\textbf{s.t.} \\
&\underline{\boldsymbol{u}}_i \leq \hat{\boldsymbol{u}}_i(k|k) \leq \overline{\boldsymbol{u}}_i, && i = 1, \ldots, \mathsf{S} \\
&H_i\hat{\boldsymbol{u}}_i(k|k) \leq \mathbf{1} \\
&F_iB_i(k)\hat{\boldsymbol{u}}_i(k|k) + \sum_{\substack{j=1 \\ j\neq i}}^{\mathsf{S}} F_{i,z_j}G_jB_j(k)\hat{\boldsymbol{u}}_j(k|k) + \mathbf{k}_i(k) \leq \overline{\boldsymbol{z}}_i, && i = 1, \ldots, \mathsf{S} \\
&\mathcal{U}_1 = \Big\{\hat{\boldsymbol{u}}_1(k|k), \ldots, \hat{\boldsymbol{u}}_S(k|k)\Big\}.
\end{aligned} \tag{8.30}$$

The matrices F_{i,z_j}, F_i, and the constant vector $\mathbf{k}_i$ are obtained by rearranging and reordering the matrices $B_{i,j}$ and $\Psi_{[\mathcal{R}_i,\mathcal{R}_i]}$. In [48] we provide the details about how the optimization problem in (8.29) is transformed into the optimization problem in (8.30).

The relevant feature of the optimization problem in (8.30) is that the only part of the i^{th} subsystem that affects all of the other subsystems is $G_iB_i\hat{\boldsymbol{u}}_i(k|k)$. Therefore, *the contribution of the i^{th} subsystem to the evolution of all of the other subsystems lives in a space of dimension m_i which is much smaller than the dimension of the i^{th} system input.* We exploit this feature in the hierarchical control strategy.

8.3 Hierarchical control strategy

We introduce here the optimization problem solved by the data center and by the zone controllers.

Data center optimization problem. As we discussed in the previous subsection, a data center can be seen as a collection of subsystems which are coupled via their outputs. The key point is that the dimension of the output of every zone is much smaller than the dimension of the state of the zone. The data center controller can take advantage of this particular structure of the zone and consider a control strategy only over the dimension of the output of the subsystems.

Assume that for every i and k, $G_iB_i(k)$ is a full row rank matrix. Then, for each of the i subsystems we define a $p_i \times m_i$ matrix $M_i(k)$ such that $G_iB_i(k)M_i(k) = I$. The matrices $\{M_i(k)\}$ represent how the data center controller expects the zone level controller to operate. The optimization problem solved by the data center controller can be written as

$$\begin{aligned}
&\min_{\mathcal{V}_1} \sum_{i=1}^{\mathsf{S}} \boldsymbol{c}_{i,u}^T(k) M_i(k) \hat{\boldsymbol{v}}_i(k|k) \\
&\textbf{s.t.} \\
&\underline{\boldsymbol{u}}_i \leq M_i \hat{\boldsymbol{v}}_i(k|k) \leq \overline{\boldsymbol{u}}_i, && i = 1, \dots, \mathsf{S} \\
&H_i M_i(k) \hat{\boldsymbol{v}}_i(k|k) \leq 1 && i = 1, \dots, \mathsf{S} \\
&F_i B_i(k) M_i(k) \hat{\boldsymbol{v}}_i(k|k) + F_{i,z} \hat{\boldsymbol{v}}(k|k) + \mathbf{k}_i(k) \leq \overline{\boldsymbol{z}}_i && i = 1, \dots, \mathsf{S}, \\
&\mathcal{V}_1 = \{\hat{\boldsymbol{v}}_1(k|k), \dots, \hat{\boldsymbol{v}}_{\mathsf{S}}(k|k)\}
\end{aligned} \tag{8.31}$$

where $\hat{\boldsymbol{v}}^*(k|k) = \begin{bmatrix} \hat{\boldsymbol{v}}_1^{*T}(k|k) & \dots & \hat{\boldsymbol{v}}_S^{*T}(k|k) \end{bmatrix}^T$. Using of the matrices $\{M_i(k)\}$, the data center controller optimizes over a reduces optimization space whose dimension equals the dimension of the output of every subsystem.

Zone level controllers. The vector $\hat{\boldsymbol{v}}^*(k|k)$, solution of (8.31), is then broadcasted

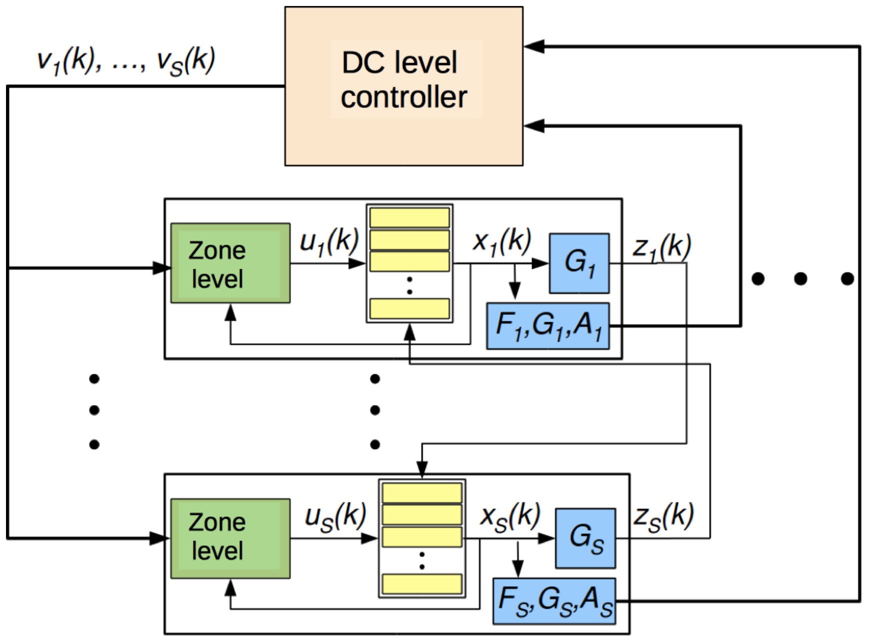

Figure 8.1: Hierarchy architecture and variables considered in the optimization problems.

to each of the i^{th} subsystems, which solve optimization problems having the form:

$$\begin{aligned}
&\min_{\hat{\boldsymbol{u}}_i(k|k)} \boldsymbol{c}_{i,u}^T(k)\hat{\boldsymbol{u}}_i(k|k) \\
&\textbf{s.t.} \\
&\underline{\boldsymbol{u}}_i \leq \hat{\boldsymbol{u}}_i(k|k) \leq \overline{\boldsymbol{u}}_i, \\
&H_i\hat{\boldsymbol{u}}_i(k|k) \leq \mathbf{1} \\
&F_iB_i(k)\hat{\boldsymbol{u}}_i(k|k) + \sum_{\substack{l=1 \\ l\neq i}}^{\mathsf{S}} F_{i,z_l}\hat{\boldsymbol{v}}_l(k|k) + \mathbf{k}_i(k) \leq \overline{\boldsymbol{z}}_i, \\
&\hat{\boldsymbol{v}}_i^*(k|k) = G_iB_i(k)\hat{\boldsymbol{u}}_i(k|k),
\end{aligned} \tag{8.32}$$

where the last constraint ensures the coherence among the different optimization problems. Figure 8.1 shows the architecture of the hierarchy and the role played by each variable.

The goal of the data center controller is to define the external behavior of the zones and of the CRAC units. The zone level controllers then, define the local control strategy of every zone so as to realize the external behavior required by the data center controller. The particular hierarchy control strategy defined in this subsection has multiple properties, which are discussed in the following propositions.

Proposition 1. The following properties hold true:

- The minimum cost of (8.31) is always greater than or equal to the minimum cost of (8.30).
- if (8.32) is feasible for every i, then the sum of the minimum costs of the optimization problem of each subsystem is grater than or equal to the minimum cost of (8.30).
- if the optimization problem (8.31) is feasible, then (8.32) is feasible for all $i = 1, \dots, \mathsf{S}$.
- the minimum cost of the i^{th} sub-problem in (8.32) is smaller than or equal to $\boldsymbol{c}_{i,u}^T(k) M_i(k) \hat{\boldsymbol{v}}_i^\star(k|k)$, where $\hat{\boldsymbol{v}}_i^\star(k|k)$ is the i^{th} sub-vector of the solution to (8.31).

Proof. Follows by construction. □

The first two points of Prop. 1 state that the performance of the hierarchical control approach may only attempt to be as good as the performance obtained by a controller solving the original control problem, but it will never be better. This result was to be expected. The third point of Prop. 1 states that the performance predicted by the data center level controller can actually be improved by the zone level controllers. The fourth point of Prop. 1 states that if the data center controller is able to solve its control problem, then it is guaranteed that all of the zone level controllers will also be able to solve their optimization problem. This point has relevant practical applications, since it implies that the data center controller does not need to verify that every zone level controller solve its own optimization problem. Once the data center controller has found a solution to its optimization problem, it is guaranteed that all of the other controllers will also be able to solve their optimization problem and therefore, a control input which enforces all of assigned constraints can be synthesized.

Proposition 2. For all $i, j = 1, \dots, \mathsf{S},\ i \neq j$, the condition $\hat{\boldsymbol{v}}_i^\star(k|k) = G_i B_i(k) \hat{\boldsymbol{u}}_i(k|k)$ in

(8.32) can be replaced by

$$F_{j,z_i} M_i(k)\hat{\boldsymbol{u}}_i(k|k) \leq F_{j,z_i} \hat{\boldsymbol{v}}_i^\star(k|k) \tag{8.33}$$

The proof is given in App. A.8

Proposition 3. Let $\tilde{\boldsymbol{u}}_i(k|k)$ and $\hat{\boldsymbol{v}}_i(k|k)$ be such that $G_i B_i(k)\tilde{\boldsymbol{u}}_i(k|k) = \hat{\boldsymbol{v}}_i(k|k)$, then there exists a vector $\boldsymbol{\xi}_i \in \mathcal{N}(G_i B_i(k))$ such that $\tilde{\boldsymbol{u}}_i(k|k) = M_i(k)\hat{\boldsymbol{v}}_i(k|k) + \boldsymbol{\xi}_i$, where $\mathcal{N}(G_i B_i(k))$ is the right null space of $G_i B_i(k)$.

Proof. Define $\boldsymbol{\xi}_i = \tilde{\boldsymbol{u}}_i(k|k) - M_i(k)\hat{\boldsymbol{v}}_i(k|k)$. Since $M_i(k)$ is the right inverse of $G_i B_i(k)$ the result follows. □

Proposition 4. Let $M_i(k)$ be a right inverse matrix of $G_i B_i(k)$. The optimization problem in (8.30) is equivalent to the following

$$\begin{aligned}
&\min_{\mathcal{V}_1,\Xi} \sum_{i=1}^{\mathsf{S}} \left(\boldsymbol{c}_{i,u}^T(k) M_i(k)\hat{\boldsymbol{v}}_i(k|k) + \boldsymbol{c}_{i,u}^T(k)\boldsymbol{\xi}_i \right) \\
&\textbf{s.t.} \\
&\underline{\boldsymbol{u}}_i \leq M_i(k)\hat{\boldsymbol{v}}_i(k|k) + \boldsymbol{\xi}_i \leq \overline{\boldsymbol{u}}_i, && i = 1, ...\mathsf{S} \\
&H_i(M_i(k)\hat{\boldsymbol{v}}_i(k|k) + \boldsymbol{\xi}_i) \leq \mathbf{1}, && i = 1, ...\mathsf{S} \\
&F_i B_i(k) M_i(k)\hat{\boldsymbol{v}}_i(k|k) + F_i B_i(k)\boldsymbol{\xi}_i + \\
&\qquad \sum_{\substack{j=1 \\ j\neq i}}^{\mathsf{S}} F_{i,z_j} G_j B_j M_i(k)\hat{\boldsymbol{v}}_i(k|k) \leq \overline{\boldsymbol{z}}_i - \mathsf{k}_i(k), && i = 1, ..., \mathsf{S} \\
&\mathcal{V}_1 = \{\hat{\boldsymbol{v}}_1(k|k), \dots, \hat{\boldsymbol{v}}_S(k|k)\} \\
&\Xi = \{\boldsymbol{\xi}_1, \dots, \boldsymbol{\xi}_S\} \\
&\boldsymbol{\xi}_i \in \mathcal{N}(G_i B_i(k)), && i = 1, \dots, \mathsf{S}.
\end{aligned} \tag{8.34}$$

Proof. Due to Prop. 3, any feasible point $\hat{\boldsymbol{u}}(k|k)$ for (8.30) can be written as a feasible point $[\hat{\boldsymbol{v}}(k|k)\}, \boldsymbol{\xi}]$ for (8.34) with the same cost. Similarly for any feasible point $[\hat{\boldsymbol{v}}(k|k), \boldsymbol{\xi}]$ exists a feasible point $\hat{\boldsymbol{u}}(k|k)$ for (8.30) which leads to the same cost. □

The meaning of Prop. 4 is that it always exists a way to consider the aggregated effects of zone level controllers, such that the overall hierarchical control approach is as optimal as solving the original optimization problem. The problem is how to verify if the chosen collection of matrices $\{M_i(k)\}$ is optimal.

Let $\mathcal{M}_i^\star(k)$ be the set of right inverse matrices of $\{G_i B_i(k)\}$ such that when $M_i(k) \in \mathcal{M}_i^\star(k)$ for $i = 1, \ldots, \mathsf{S}$ the minimum cost of (8.31) equals the minimum of (8.32). In general, given a choice of matrices $\{M_i(k)\}$, $i = 1, \ldots, \mathsf{S}$, we cannot test whether or not $M_i(k) \in \mathcal{M}_i^\star(k)$ for $i = 1, \ldots, \mathsf{S}$. However, a partial characterization of the set $\mathcal{M}_i^\star(k)$ is possible through Prop. 1: if the minimum cost of (8.31) is strictly greater than the sum of the minimum costs obtained for each of the (8.32) problems, then at least one of the chosen $\{M_i(k)\}$ matrix does not belong to the set $\mathcal{M}_i^\star(k)$. A good selection of $\{M_i(k)\}$ matrices is the one for which the minimum cost of (8.31) equals the sum of the minimum costs of (8.32).

We want now to study the effects of a wrong selection of a wrong set of matrices $\{M_i(k)\}$ on the hierarchy performance. The following subsection discusses the performance of the data center controller and, the performance of the whole hierarchy when a wrong selection of the matrices $\{M_i(k)\}$ is made.

8.4 Effects of a wrong selection of the $\{M_i(k)\}$ matrices

Consider a data center composed by 6 racks, each having 42 servers and 3 CRAC units. Racks and CRAC units are placed as in Fig. 8.2. Servers are identical each others and CRAC units are also identical each others. Servers have a weak thermal interaction among them and a strong thermal interaction with the CRAC units. Jobs are divided among 6 classes, and arrivals are evenly distributed among servers, so that $\lambda_{i_1}^j(k) = \lambda_{i_2}^j(k)$ for all $i_1, i_2 = 1, \ldots, 252$ and $j = 1, \ldots, 6$. We define this setup as the *nominal model*. For the nominal model, the optimal selection of matrices $\{M_i(k)\}$ is known.

The plots in Fig. 8.3 are: the relative difference between the cost computed by the

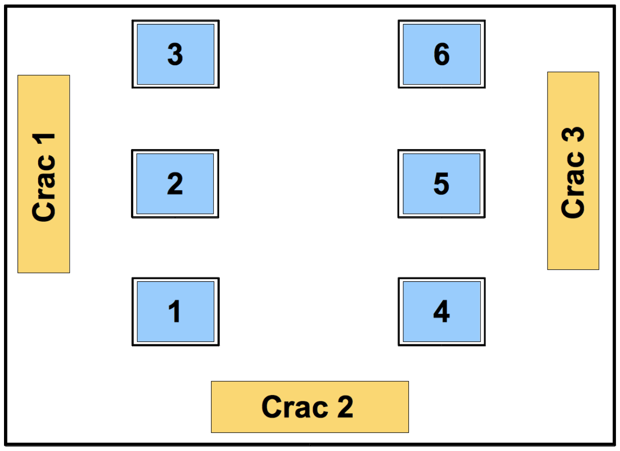

Figure 8.2: Data center layout.

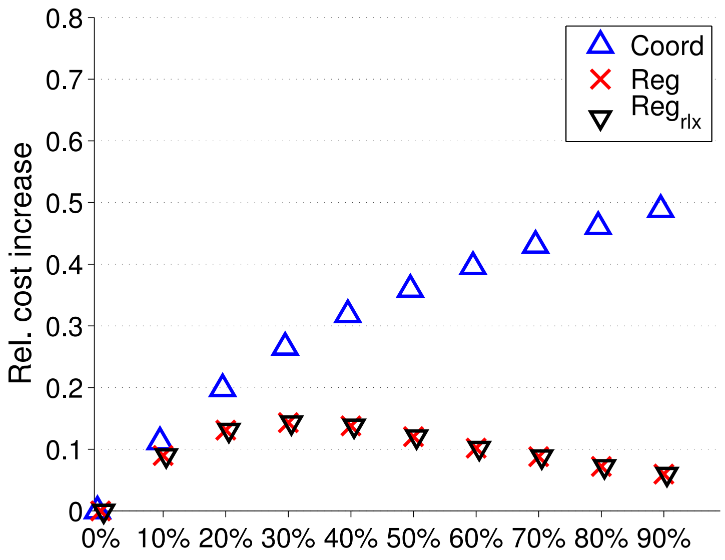

Figure 8.3: Relative cost increase for different values of the coefficient of perturbation.

data center controller and the cost of a controller solving (8.30) (Coord, in the figure), the relative difference between the sum of the cost obtained by the whole hierarchy solving (8.30) (Reg, in the figure). In Fig. 8.3 we also show the cost obtained by whole hierarchy when zone level controller solves optimization problem with weakened constraints discussed in Prop. 2 (Reg_{rlx}, in figure). Every point in the graph represent the average value over value is computed over 500 different simulations.

As the value of the coefficient of perturbation increases, the relative difference between the minimum cost found by the coordinator and the one computed by a controller solving (8.30) increases. When the sum of the costs is obtained by the local regulators instead, their optimal solutions induce a cost function increase of about 10%.

Our simulations suggest that the approach may be effective, but several research directions should be pursued to fully evaluate the approach and to extend it to more general situations. When the arrivals are quite predictable, it will be useful to be able to optimize the economic parameters over a long temporal horizon.

8.5 Summary

This chapter discusses the performance loss of a hierarchical control system, when the data center controller use wrong aggregated model of the zones. The analysis is performed over a simplified data center case, so that the optimal control law of the data center controller can be synthesized without using aggregated models.

The results discussed in this chapter suggest that the effects of a model mismatches can be mitigated by the zone-level controllers. The relevant assumption made in this chapter is that the zones are thermally coupled only via the average temperature of their servers.

Chapter 9

Conclusion and future work

This chapter summarizes the results obtained and the contributions made in this dissertation and suggests several directions for future research. This dissertation considers data centers as CPSs, with a focus on run-time management and operating costs. A modeling framework is proposed in order to explicitly capture the cyber-physical nature of data centers. The framework facilitates the development of models that represent both the computational and the thermal characteristics of a data center, as well as their interactions. The proposed control strategy, based on a hierarchical/distributed approach, attempts to manage both the computational and the thermal characteristics of a data center.

The main results obtained in the dissertation can be summarized in the following points

- Every data center has operating regimes for which the baseline and the uncoordinated controllers are as optimal as the coordinated controller.
- There are cases, for which the uncoordinated control strategy is as optimal as the coordinated one, independently from the operating regime. An index is proposed in order to characterize how much savings can be obtained by using a coordinated controller with respect to an uncoordinated controller.
- Data centers can take advantage of service level agreements (SLAs) with the power grid. It is possible however, that a data center destabilizes the power grid rather

than reducing the fluctuation of energy demand.

- Turning off servers is the usual approach considered to make the power consumption of a group of servers proportional to the amount of workload they process. We show that if the workload arrival rate changes too quickly, then keeping servers always on is the best strategy. The meaning of "too quickly" is related to the ratio between time required to turn servers on and the average time over which job arrival rate can be considered constant.
- The use of aggregated model at the data center level reduces the performance of the overall control strategies. However, under certain assumptions, typically verified in a data center, the overall control performance is still within few percentage of the optimal one obtained without the use of aggregated models.

9.1 Contributions

This dissertation makes the following contributions in the area of data center control:

- **A framework for the development of cyber-physical models of data centers.** We believe that data center control algorithms can take advantage of a cyber-physical representation of a data center. The proposed modeling framework explicitly captures the cyber-physical nature of data centers by means of two interacting networks. One network, called computational network, describes the process of data exchange within the data center and between the data center and the external users. The other network, called thermal network, describes the energy exchange within the data center and between the data center and the external environment. The coupling between the two networks yields the cyber-physical representation of the data center.

  The proposed framework is sufficient to the scope of this dissertation. However, we recognize that there may be cases which do not easily fit into the proposed framework. For example, the representation of software licenses, data storage, and human-in-the-loop issues may benefit from an increase in the descriptive capabilities of the modeling framework.

- **A collection of control-oriented models.** Multiple models are proposed in the dissertation. The models describe different elements of the data center with different level of accuracy and for different purposes. Large part of the dissertation is focused on the modeling of servers and CRAC units. Under mild assumptions, the dissertation shows that data centers are large-scale systems whose evolution is driven by stochastic processes such as the the job arrival and the job departure.

- **A control approach.** The hierarchical/distributed control strategy proposed in this dissertation takes advantage of the modularity typically found in data centers. The proposed hierarchy is composed by three control levels. The highest level of the hierarchy deals with processes having dynamics of the order of tens of minutes or more. As we descend the hierarchy, the time scale of the controlled processes increases. The highest level of the hierarchy is called data center level and it is the main subject of this dissertation. A single controller operates at the data center level and it directs the lower level controllers. The goal of lower level controllers is to minimize the effects of variations of disturbances around their predicted average values.

  Three controllers are proposed for the data center level: baseline, uncoordinated, and coordinated. The three controllers take advantage of a unified thermal-computational models of the data center in different ways. The baseline controller operates in open-loop. The uncoordinated controller manages the nodes of the computational and of the thermal networks as if their evolutions were decoupled. The coordinated controller manages the nodes of the computational and of the thermal networks considering their couping.

  A cyber-physical index (CPI) is proposed in the dissertation. The index aims at characterizing how much savings can be obtained by using a coordinated controller with respect to an uncoordinated controller.

- **A collection of MATLAB routines to simulate the behavior of data centers.** The collection of MATLAB routines are used throughout the dissertation to simulate the behavior of the data center under a variety of scenarios and with different controllers. The simulator is developed following a modular approach, so that it can be extended to represent cases not tackled in this dissertation. Functions and variables have meaningful names so that they can be easily maintained by multiple users. In order to reduce the time required to execute a simulation, the functions avoid the use of loops as much as possible.

9.2 Future work

There are multiple possible research directions in the future.

- **Model validation.** Validation of the proposed models is surely one of the most important research direction to pursue. In this dissertation, multiple models have been proposed, but only few of them have been validated against real data. Furthermore, even those models that we already validated, require to be tested against a larger set of cases. Multiple server architecture have to be tested in order to understand which server power consumption model is effective in which case. We do not expect that a single model will fit every case, but rather that a collection of data center models will have to be identified along with the cases to which they are applicable to.

  Toward this goal, the collection of data in multiple data centers is essential.

- **Data collection.** Collecting data in data centers is not a trivial task. Not only temperature and humidity of the air have to be measured, but also the statistics about the use of hardware resources, power consumption of servers, speed at which fans rotate, and the number and the type of user requests per second. In a medium-size data center, this can require the collection of thousand of samples at the beginning of every sampling time. The number of samples is not the only problem related to data collection in data centers. The accuracy of the sensors used for collecting data and the time-constant of the sensors are also relevant. Data collected with the temperature probes used at the data center observatory (DCO) at Carnegie Mellon University (CMU), show that the time constant of the probes is about 50 s.[1] Such a time constant is too large to measure the dynamic evolutions of the air at the outlet and at the inlet of the servers.

Data storage and retrieval with such a large pool of data is a challenging problem. Sensor Andrew, a research project developed at Carnegie Mellon University, can be used to to collect, store, and distribute the massive amount of data collected in the DCO among the users.[2]

- **Management of data storage.** Extension of the control algorithm in order to manage the data allocation is also an interesting research direction. A large body of research has been working on the development of storage algorithms for distributed data storage [18, 22, 34, 55, 64, 65]. This dissertation considers the problem of data allocation as if it could be solved independently from the management of the workload scheduling and migration among the servers. Including data storage strategies into the data center control algorithms may increase the possibility to turn off unused servers, while guaranteeing data availability and data redundancy.
- **Further development of the simulator.** Extending the data center simulator developed in this dissertation is also an interesting research direction. The development of data center simulator, which can be used by the research community, would allow the comparison of different control algorithms under the same test cases. Even though the accuracy of the simulations might not be adequate for every purposes, the

[1]The temperature probes used are APC NetBotz TS100 external temperature sensors, `www.apc.com` .
[2]`http://sensor.andrew.cmu.edu` .

possibility to study how different control algorithms behave under the same scenarios, overwhelms the inaccuracies generated by testing the controllers in simulation, rather than in a real environment.

Bibliography

[1] ADC Telecommunication Inc. TIA-942 data center standards overview. Technical report, ADC Telecommunication Inc., 2006.

[2] M. Aghajani, L. Parolini, and B. Sinopoli. Dynamic power allocation in server farms: a real time optimization approach. In *Proc. of the 49th IEEE Conference on Decision and Control (CDC)*, pages 3790–3795, Dec. 2010.

[3] E. Allen, J. Lang, and M. Ilic. A combined equivalenced-electric, economic, and market representation of the northeastern power coordinating council U.S. electric power system. *Power Systems, IEEE Transactions on*, 23(3):896–907, Aug. 2008.

[4] American Society of Heating, Refrigerating and Air-Conditioning Engineers. Thermal guidelines for data processing environments. Technical report, ASHRAE, 2004.

[5] American Society of Heating, Refrigerating and Air-Conditioning Engineers. Environmental guidelines for datacom equipment. Expanding the recommended environmental envelope. Technical report, ASHRAE, 2008.

[6] D. G. Andersen, J. Franklin, M. Kaminsky, A. Phanishayee, L. Tan, and V. Vasudevan. FAWN: A fast array of wimpy nodes. In *Proc. of the 22nd ACM Symposium on Operating Systems Principles (SOSP)*, pages 1–14, 2009.

[7] M. Anderson, M. Buehner, P. Young, D. Hittle, C. Anderson, J. Tu, and D. Hodgson. MIMO robust control for HVAC systems. *IEEE Transactions on Control Systems Technology*, 16(3):475–483, 2008.

[8] A. C. Antoulas, D. C. Sorensen, and S. Gugercin. A survey of model reduction methods for large-scale systems. *Contemporary Mathematics*, 280:193–219, 2001.

[9] H. Aydin and D. Zhu. Reliability-aware energy management for periodic real-time tasks. *IEEE Transactions on Computers*, 58(10):1382–1397, 2009.

[10] L. A. Barroso and U. Holzle. The case for energy-proportional computing. *Computer*, 40(12):33–37, 2007.

[11] C. Bash and G. Forman. Cool job allocation: Measuring the power savings of placing jobs at cooling-efficient locations in the data center. In *Proc. of USENIX Annual Technical Conference*, number 29 in 6, pages 363–368, Jun. 2007.

[12] C. E. Bash, C. D. Patel, and R. K. Sharma. Dynamic thermal management of air cooled data centers. In *Proc. of the 10th Intersociety Conf. Thermal and Thermomechanical Phenomena in Electronics Systems (ITHERM)*, pages 445–452, May 2006.

[13] M. H. Beitelmal and C. D. Patel. Model-based approach for optimizing a data center

centralized cooling system. Technical report, Hewlett Packard Laboratories, Apr. 2006.

[14] C. L. Belady. Projecting annual new datacenter construction market size. Technical report, Global foundation services, Mar. 2011.

[15] T. J. Breen, E. J. Walsh, J. Punch, A. J. Shah, and C. E. Bash. From chip to cooling tower data center modeling: Part I influence of server inlet temperature and temperature rise across cabinet. In *Proc. of the 12th IEEE Intersociety Conf. Thermal and Thermomechanical Phenomena in Electronic Systems (ITHERM)*, pages 1–10, Jun. 2010.

[16] G. Chen, W. He, J. Liu, S. Nath, L. Rigas, L. Xiao, and F. Zhao. Energy-aware server provisioning and load dispatching for connection-intensive internet services. In *Proc. of the 5th USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, pages 337–350, 2008.

[17] Y. Chen, D. Gmach, C. Hyser, Z. Wang, C. Bash, C. Hoover, and S. Singhal. Integrated management of application performance, power and cooling in data centers. In *Proc. of the 12th IEEE/IFIP Network Operations and Management Symposium (NOMS)*, pages 615–622, Apr. 2010.

[18] G. Cherubini, C. C. Chung, W. C. Messner, and S. O. R. Moheimani. Control methods in data-storage systems. *Control Systems Technology, IEEE Transactions on*, 20(2):296–322, Mar. 2012.

[19] Y. Cho and N. Chang. Energy-aware clock-frequency assignment in microprocessors and memory devices for dynamic voltage scaling. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 26(6):1030–1040, 2007.

[20] P. Choudhary and D. Marculescu. Power management of voltage/frequency island-based systems using hardware-based methods. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 17(3):427–438, 2009.

[21] R. C. Chu, R. E. Simons, M. J. Ellsworth, R. R. Schmidt, and V. Cozzolino. Review of cooling technologies for computer products. *IEEE Transactions on Device and Materials Reliability*, 4(4):568–585, 2004.

[22] B. Fan, H. Lim, D. G. Andersen, and M. Kaminsky. Small cache, big effect: Provable load balancing for randomly partitioned cluster services. In *Proc. of the 2^rd ACM Symposium on Cloud Computing (SOCC)*, pages 1–12, Oct. 2011.

[23] X. Fan, W.-D. Weber, and L. A. Barroso. Power provisioning for a warehouse-sized computer. In *International Symposium on Computer Architecture*, pages 13–23, Jun. 2007.

[24] E. Ferrer, C. Bonilla, C. Bash, and M. Batista. Data center thermal zone mapping. White paper, Hewlett-Packard, 2007.

[25] A. Gandhi, M. Harchol-Balter, and I. Adan. Server farms with setup costs. *Performance Evaluation*, 67:1123–1138, 2010.

[26] A. Gandhi, M. Harchol-Balter, R. Das, and C. Lefurgy. Optimal power allocation in

server farms. In *ACM SIGMETRICS*, pages 157–168, Jun. 2009.

[27] P. Gao, S. Wittevrongel, and H. Bruneelg. Delay and partial system contents for a discrete-time G-D-c queue. *4OR: A Quarterly Journal of Operations Research*, 6(3):279–290, 2008.

[28] P. Gao, S. Wittevrongel, J. Walraevens, M. Moeneclaey, and H. Bruneel. Calculation of delay characteristics for multiserver queues with constant service times. *European Journal of Operational Research*, 199(1):170–175, 2009.

[29] R. A. Greco. High density data centers fraught with peril. Slides, EYP Mission Critical Facilities Inc., 2003.

[30] J. Hamilton. Cost of power in large-scale data centers. `http://perspectives.mvdirona.com`, Nov. 2008.

[31] A. Hawkins. Unused servers survey results analysis. White Paper 28, The Green Grid, 2010.

[32] Hewlett-Packard. Hp modular cooling system: water cooling technology for high-density server installations. Technical report, Hewlett-Packard, 2007.

[33] Z. Jian-Hui and Y. Chun-Xin. Design and simulation of the cpu fan and heat sinks. *IEEE Transactions on Components and Packaging Technologies*, 31(4):890–903, 2008.

[34] S. Khan and I. Ahmad. A pure nash equilibrium-based game theoretical method for data replication across multiple servers. *Knowledge and Data Engineering, IEEE Transactions on*, 21(4):537–553, Apr. 2009.

[35] J. Kim, S. Yoo, and C.-M. Kyung. Program phase-aware dynamic voltage scaling under variable computational workload and memory stall environment. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 30(1):110–123, 2011.

[36] H. Kobayashi and B. L. Mark. *System Modeling and Analysis: Foundations of System Performance Evaluation*. Prentice Hall Press, 2008.

[37] C. Lefurgy, X. Wang, and M. Ware. Server-level power control. In *Proc. Fourth Int. Conf. Autonomic Computing ICAC*, page 4, Jun. 2007.

[38] J. Leverich, M. Monchiero, V. Talwar, P. Ranganathan, and C. Kozyrakis. Power management of datacenter workloads using per-core power gating. *Computer Architecture Letters*, 8(2):48–51, 2009.

[39] L. Ljung. *System identification: theory for the user*. Prentice-Hall information and system sciences series. Prentice-Hall, 2nd edition, 1999.

[40] J. Mao, Q. Zhao, and C. G. Cassandras. Optimal dynamic voltage scaling in power-limited systems with real-time constraints. In *Proc. CDC Decision and Control 43rd IEEE Conf*, volume 2, pages 1472–1477, 2004.

[41] G. I. Meijer. Cooling energy-hungry data centers. *Science*, 328(5976):318–319, 2010.

[42] D. Meisegeier, M. Howes, D. King, and J. Hall. Potential peak load reductions from residential energy efficient upgrades. White paper, ICF International, 2002.

[43] R. Miller. Facebook to build second data center in NC, Oct. 2011.

[44] J. Moore, J. Chase, P. Ranganathan, and R. Sharma. Making scheduling "Cool": temperature-aware workload placement in data centers. In *USENIX Annual Technical Conference*, pages 61–75, Apr. 2005.

[45] A. Mutapcic, S. Boyd, S. Murali, D. Atienza, G. De Micheli, and R. Gupta. Processor speed control with thermal constraints. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 56(9):1994–2008, 2009.

[46] National Association of Home Builders (NAHB) Research Center, Inc. Review of residential electrical energy use data. White paper, NAHB Research Center, Inc., 2001.

[47] P. Padala, K. G. Shin, X. Zhu, M. Uysal, Z. Wang, S. Singhal, A. Merchant, and K. Salem. Adaptive control of virtualized resources in utility computing environments. *SIGOPS Oper. Syst. Rev.*, 41:289–302, Mar. 2007.

[48] L. Parolini, E. Garone, B. Sinopoli, and B. H. Krogh. A hierarchical approach to energy management in data centers. In *Proc. of the 49th IEEE Conference on Decision and Control (CDC)*, pages 1065–1070, Dec. 2010.

[49] L. Parolini, B. Sinopoli, and B. H. Krogh. Reducing data center energy consumption via coordinated cooling and load management. In *Workshop on Power Aware Computing and Systems (HotPower)*, pages 14–18, Dec. 2008.

[50] L. Parolini, B. Sinopoli, and B. H. Krogh. Model predictive control of data centers in the smart grid scenario. In *Proc. of the 18th International Federation of Automatic Control (IFAC) World Congress*, Aug. 2011.

[51] L. Parolini, B. Sinopoli, B. H. Krogh, and Z. Wang. A cyber-physical systems approach to data center modeling and control for energy efficiency. *Proceedings of the IEEE*, 100(1):254–268, 2011.

[52] L. Parolini, N. Tolia, B. Sinopoli, and B. H. Krogh. A cyber-physical systems approach to energy management in data centers. In *First international conference on cyber-physical systems (ICCPS)*, pages 168–177, Apr. 2010.

[53] C. D. Patel and A. J. Shah. Cost model for planning, development and operation of a data center. Technical report, Internet Systems and Storage Laboratory, HP Laboratories Palo Alto, Jun. 2005.

[54] C. D. Patel, R. K. Sharma, C. Bash, and M. Beitelmal. Thermal considerations in cooling large scale high compute density data centers. In *Proc. of the 8th Intersociety Conf. Thermal and Thermomechanical Phenomena in Electronics Systems (ITHERM)*, pages 767–776, May 2002.

[55] S. Patil and G. Gibson. Scale and concurrency of giga+: file system directories with millions of files. In *Proceedings of the 9th USENIX conference on File and stroage technologies*, FAST'11, pages 13–29, Feb. 2011.

[56] M. Patterson, D. Costello, P. F. Grimm, and M. Loeffler. Data center TCO; a comparison of high-density and low-density spaces. White paper, Intel Corporation, 2007.

[57] M. K. Patterson and D. Fenwick. The state of data center cooling. White paper, Intel Corporation, Mar. 2008.

[58] P. Pillai, M. Kaminsky, M. A. Kozuch, V. Vasudevan, L. Tan, and D. G. Andersen. FAWNSort: Energy-efficient sorting of 10gb, 2011. Winner of 2011 10GB Joulesort Daytona and Indy categories.

[59] Processor.com. Data center physical security. pp.30, Oct. 2011.

[60] A. Qureshi, R. Weber, H. Balakrishnan, J. Guttag, and B. Maggs. Cutting the electric bill for internet-scale systems. In *Proc. of the ACM SIGCOMM Conference on Data communication*, pages 123–134, Aug. 2009.

[61] S. Rahman. Power for the internet. *Computer Applications in Power, IEEE*, 14(4):8–10, 2001.

[62] L. Rao, X. Liu, L. Xie, and W. Liu. Minimizing electricity cost: Optimization of distributed internet data centers in a multi-electricity-market environment. In *Proc. of the 29th IEEE International Conference on Computer Communications (INFOCOM)*, pages 1–9, Mar. 2010.

[63] J. B. Rawlings. Tutorial overview of model predictive control. *Control Systems Magazine, IEEE*, 20(3):38–52, Jun. 2000.

[64] W. Ren and P. Zhou. Research on the data storage and access model in distributed environment. In *International Conference on Computer Engineering and Technology (ICCET)*, volume 1, pages 134–136, Jan. 2009.

[65] S. Sakr, A. Liu, D. Batista, and M. Alomari. A survey of large scale data management approaches in cloud environments. *Communications Surveys Tutorials, IEEE*, 13(3):311–336, 2011.

[66] J. Scaramella. Worldwide server power and cooling expense 2006-2010 forecast. International Data Corporation (IDC), Sep. 2006.

[67] U. I. P. Services. Data center site infrastructure tier standard: Topology. Technical report, Uptime Institute, 2009.

[68] R. K. Sharma, C. E. Bash, C. D. Patel, R. J. Friedrich, and J. S. Chase. Balance of power: Dynamic thermal management for internet data centers. *IEEE Internet Computing*, 9:42–49, 2005.

[69] Q. Tang, S. K. S. Gupta, and G. Varsamopoulos. Energy-efficient thermal-aware task scheduling for homogeneous high-performance computing data centers: A cyber-physical approach. *IEEE Transactions on Parallel and Distributed Systems*, 19(11):1458–1472, 2008.

[70] Q. Tang, T. Mukherjee, S. K. S. Gupta, and P. Cayton. Sensor-based fast thermal evaluation model for energy efficient high-performance data centers. In *Fourth International Conference on Intelligent Sensing and Information Processing*, pages 203–208, Oct. 2006.

[71] The Green Grid. The green grid data center power efficiency metrics: PUE and DCiE. White paper, Technical Committee, 2007.

[72] N. Tolia, Z. Wang, P. Ranganathan, C. Bash, M. Marwah, and X. Zhu. Unified power and cooling management in server enclosures. In *InterPACK*, pages 721–730, Jul. 2009.

[73] M. M. Toulouse, G. Doljac, V. P. Carey, and C. Bash. Exploration of a potential-flow-based compact model of air-flow transport in data centers. In *American Society Of Mechanical Engineers (ASME)*, pages 41–50, Nov. 2009.

[74] U.S. Environmental Protection Agency. Report to congress on server and data center energy efficiency. Technical report, ENERGY STAR Program, Aug. 2007.

[75] A. Varma, B. Ganesh, M. Sen, S. R. Choudhury, L. Srinivasan, and B. Jacob. A control-theoretic approach to dynamic voltage scheduling. In *International conference on compilers, architecture and synthesis for embedded systems*, pages 255–266, Oct. 2003.

[76] V. Vasudevan, J. Franklin, D. Andersen, and A. P. L. Tan. FAWNdamentally power-efficient clusters. In *Proc. of the 12th Workshop on Hot Topics in Operating Systems (HotOS XII)*, 2009.

[77] E. J. Walsh, T. J. Breen, J. Punch, A. J. Shah, and C. E. Bash. From chip to cooling tower data center modeling: Part II influence of chip temperature control philosophy. In *Proc. of the 12th IEEE Intersociety Conf. Thermal and Thermomechanical Phenomena in Electronic Systems (ITHERM)*, pages 1–7, Jun. 2010.

[78] X. Wang and Y. Wang. Coordinating power control and performance management for virtualized server clusters. *IEEE Transactions on Parallel and Distributed Systems*, 22(2):245–259, 2011.

[79] Z. Wang, C. Bash, N. Tolia, M. Marwah, X. Zhu, and P. Ranganathan. Optimal fan speed control for thermal management of servers. In *Proc. of the ASME/Pacific Rim Technical Conference and Exhibition on Packaging and Integration of Electronic and Photonic Systems, MEMS, and NEMS (InterPACK)*, 2009.

[80] N. Weste and D. Harris. *CMOS VLSI Design: A Circuits and Systems Perspective.* Addison-Wesley, 4th edition, 2010.

[81] X. Zhu, D. Young, B. Watson, Z. Wang, J. Rolia, S. Singhal, B. McKee, C. Hyser, D. Gmach, R. Gardner, T. Christian, and L. Cherkasova. 1000 islands: Integrated capacity and workload management for the next generation data center. In *International Conference on Autonomic Computing*, pages 172–181, Jun. 2008.

Appendix

A.1 Notable properties of the matrices $\Gamma(\tau)$ and $\Psi(\tau)$

Let $\mathsf{f}_i(\tau)$ (Kg/s) be the rate at which air moves through the i^{th} device at time τ. According to the recirculation model discussed in the work of Tang *et al.* [70], $\mathsf{f}_i(\tau)$ is the result of the composition of M airflows, each coming from a different device. For all $\tau \in \mathbb{R}$ and for all $i \in \{1, \dots, M\}$, we have

$$\mathsf{f}_i(\tau) = \sum_{j=1}^{M} \gamma_{i,j} \mathsf{f}_j(\tau). \tag{A.1}$$

and we assume that every airflow is strictly greater than 0. For all $j \in \{1, \dots, M\}$, we have

$$\sum_{i=1}^{M} \gamma_{i,j}(\tau) = 1. \tag{A.2}$$

Using the airflow vector $\mathbf{f}(\tau)$ and the matrix $\Gamma(\tau)$ defined in Sec. 3.3.1, (A.1) and (A.2) can be rewritten as

$$\mathbf{f}(\tau) = \Gamma(\tau)\mathbf{f}(\tau), \tag{A.3}$$

$$\mathbf{1}^T\Gamma(\tau) = \mathbf{1}^T, \tag{A.4}$$

where $\mathbf{1}$ is the column vector of appropriate dimension, whose elements are all 1. The matrix $\Gamma(\tau)$ is a left stochastic matrix and hence, it has a nonnegative vector $\boldsymbol{x}(\tau)$ associated with the eigenvalue 1. Furthermore for the Perron-Frobenius theorem, if all of the elements of the matrix $\Gamma(\tau)$ are strictly positive, then the eigenvector associated with the eigenvalue 1 is unique and can be obtained by

$$\lim_{k\to\infty}\left[\Gamma(\tau)^k\right]_{i,j}, \tag{A.5}$$

where $\left[\Gamma(\tau)^k\right]_{i,j}$ is the elements in the (i,j) position of the matrix $\Gamma(\tau)^k$.

For all $i,j \in \{1,\dots,M\}$ and $\tau \in \mathbb{R}$ we define the coefficients

$$\psi_{i,j}(\tau) \triangleq \frac{\gamma_{i,j}(\tau)\mathsf{f}_j(\tau)}{\mathsf{f}_i(\tau)} \tag{A.6}$$

Since for all $i,j \in \{1,\dots,M\}$, the variabels $\{\mathsf{f}_i(\tau)\}$ and $\{\gamma_{i,j}(\tau)\}$ are nonnegative, we have that all of the variables $\{\psi_{i,j}(\tau)\}$ are nonnegative.

Proposition 5. For all $i \in \{1,\dots,M\}$

$$\sum_{j=1}^{M}\psi_{i,j}(\tau) = 1. \tag{A.7}$$

Proof.

$$\sum_{j=1}^{M}\psi_{i,j}(\tau) = \sum_{j=1}^{M}\frac{\gamma_{i,j}(\tau)\mathsf{f}_j(\tau)}{\mathsf{f}_i(\tau)} = \frac{\mathsf{f}_i(\tau)}{\mathsf{f}_i(\tau)} = 1.$$

The last equality follows from the assumption that all of the airflows are strictly greater than 0. □

It easy to verify that all of the variables $\{\psi_{i,j}(\tau)\}$ are bounded in the interval $[0,1]$. We can conclude that the matrix $\Psi(\tau)$ is right stochastic. This implies that, at any time τ the value of the i^{th} element of the input temperature vector ($\boldsymbol{T}_{\text{in}}(\tau)$) is obtained via convex linear combination of all the elements of the output temperature vector ($\boldsymbol{T}_{\text{out}}(\tau)$). Therefore, the input temperature of the i^{th} thermal node is never greater than the maximum output temperature of any other thermal node and also, the input temperature of the i^{th} thermal node is never be smaller than the minimum output temperature of any other thermal node.

A.2 Thermal serve nodes, analysis at the equilibrium point

Consider the thermal server node model introduced Sec. 4.3.1. The model for the i^{th} thermal server node can be written as

$$\dot{T}_{\text{out},i}(\tau) = -\mathsf{k}_i T_{\text{out},i}(\tau) + \mathsf{k}_i T_{\text{in},i}(\tau) + \mathsf{c}_i \mathsf{p}_i(\tau), \tag{A.8}$$

where $\frac{1}{\mathsf{k}_i}$ is the time constant of the temperature of i^{th} node and c_i is the coefficient that maps power consumption into output temperature variation. Assume that $\mathsf{k}_i > 0$ and consider the system at its equilibrium point. The output temperature of the i^{th} server can be written as

$$T_{\text{out},i} = T_{\text{in},i} + \frac{\mathsf{c}_i}{\mathsf{k}_i}\mathsf{p}_i, \tag{A.9}$$

where we removed the dependency from the time variable.

At the equilibrium, the rate at which energy flows into the system equals the rate at which energy flows out from the system. The rate at which energy flows into the i^{th} thermal server node is

$$\dot{Q}_{I,i} = c_p \mathsf{f}_i T_{\text{in},i} + \mathsf{p}_i,$$

where the first addendum represents the heat transported by the air and the second addendum represents the electrical contribution. The rate at which energy flows out from the i^{th} thermal server node is

$$\dot{Q}_{O,i} = c_p \mathsf{f}_i T_{\text{out},i} = c_p \mathsf{f}_i h\Big(T_{\text{in},i} + \frac{\mathsf{c}_i}{\mathsf{k}_i}\mathsf{p}_i\Big),$$

where the second equality is due to (A.9). Since $\dot{Q}_{I,i} = Q_{O,i}$, we can conclude

$$\mathsf{c}_i = \frac{\mathsf{k}_i}{c_p \mathsf{f}_i}. \tag{A.10}$$

A.3 Relationship between inlet, outlet, and CPU temperature of a single core server

In this section we discuss a server model which considers the temperature of the air the inlet of the server, the temperature of the CPU, and the temperature of the air at the outlet of the server.

Without loss of generality, we focus on the case where a server has a single CPU. With $T_{\text{in},i}(\tau)$ we denote the temperature of the air at the inlet of the i^{th} server at time τ, with $T_{\text{out},i}(\tau)$ we denote the temperature of the air at the outlet of the server at time τ, with $T_{\text{cpu,i}}(\tau)$ we denote the temperature of the CPU at time τ, and with $\mathsf{p}_i(\tau)$ we denote the power consumption of the server at time τ. We focus on the case where the majority of the power consumption of the server is due to the CPU and we approximate the power consumption of the CPU with the power consumption of the server.

As discussed in the work of Wang *et al.* [79], the rate at which heat transfer from the CPU to the air depends on the *thermal resistance* of the CPU, which we denote with $R_{\text{CPU},i}(\tau)$ (K s / J). The rate at which the CPU exchanges heat with the air is

$$\dot{Q}_i(\tau) = \frac{T_{\text{cpu,i}}(\tau) - T_{\text{in},i}(\tau)}{R_{\text{CPU},i}(\tau)}. \tag{A.11}$$

The relationship between the thermal resistance and the speed of the airflow can be written as

$$R_{\text{CPU},i}(\tau) = \frac{C_3}{\mathsf{f}_i(\tau)^{n_R}} + C_4, \tag{A.12}$$

where C_3 (K Kg^{n_R}/ (J s$^{n_R-1}$)) and C_4 ($\frac{K\,s}{J}$) are non-negative coefficients whose values depend on multiple factors among which the geometry of the server, $\mathsf{f}_i(\tau)$ (Kg/s) is the rate at which air moves through the server and n_R is a coefficients which depends on the degree of turbulence of the air. The coefficient n_R typically ranges in the interval [1, 2].

The variation of the temperature of the CPU is proportional to the difference between the amount of electrical power consumed by the CPU, i.e., $\mathsf{p}_i(\tau)$, and the rate at which the CPU exchange heat with the air, i.e., $\dot{Q}_i(\tau)$. We can write

$$C_1\dot{T}_{\text{cpu},i}(\tau) = \mathsf{p}_i(\tau) - \dot{Q}_i(\tau), \tag{A.13}$$

where C_1 (J/K) is a non-negative coefficient. Substituting (A.11) and (A.12) in (A.13), we obtain

$$\dot{T}_{\text{cpu},i}(\tau) = -\frac{\mathsf{f}_i(\tau)^{n_R}}{C_3 + C_4\mathsf{f}_i(\tau)^{n_R}}\frac{1}{C_1}T_{\text{cpu,i}}(\tau) + \frac{\mathsf{f}_i(\tau)^{n_R}}{C_3 + C_4\mathsf{f}_i(\tau)^{n_R}}\frac{1}{C_1}T_{\text{in},i}(\tau) + \frac{1}{C_1}\mathsf{p}_i(\tau). \tag{A.14}$$

Eq. (A.11) shows the rate at which heat is transferred from the CPU to the airflow. The rate at which heat is transferred to the air at the outlet of the server is

$$\dot{Q}_{out}(\tau) = \dot{Q}_{in}(\tau) + \dot{Q}_i, \tag{A.15}$$

where $\dot{Q}_{in}(\tau)$ (J/s) is the rate at which heat enters the server via the inlet air. Let c_p be the specific heat of the air at constant pressure. We can write

$$\mathsf{c}_p\mathsf{f}_i(\tau)T_{\text{out},i}(\tau) = c_p\mathsf{f}_i(\tau)T_{\text{in},i}(\tau) + \dot{Q}_i. \tag{A.16}$$

Considering (A.11) and (A.12), we can rewrite (A.16) as

$$\begin{aligned} T_{\text{out},i}(\tau) \quad = \quad & \left[1 - \frac{\mathsf{f}_i(\tau)^{n_R-1}}{c_p}\frac{1}{C_3 + C_4\mathsf{f}_i(\tau)^{n_R}}\right]T_{\text{in},i}(\tau) + \\ & \left[\frac{\mathsf{f}_i(\tau)^{n_R-1}}{c_p}\frac{1}{C_3 + C_4\mathsf{f}_i(\tau)^{n_R}}\right]T_{\text{cpu,i}}(\tau). \end{aligned} \tag{A.17}$$

The thermal evolution of a server having a single CPU can then be written as

$$\begin{cases} \dot{T}_{\text{cpu},i}(\tau) &= -\dfrac{\mathsf{f}_i(\tau)^{n_R}}{C_3 + C_4\mathsf{f}_i(\tau)^{n_R}} \dfrac{1}{C_1} T_{\text{cpu,i}}(\tau) + \dfrac{\mathsf{f}_i(\tau)^{n_R}}{C_3 + C_4\mathsf{f}_i(\tau)^{n_R}} \dfrac{1}{C_1} T_{\text{in},i}(\tau) + \\ & \quad \dfrac{1}{C_1}\mathsf{p}_i(\tau) \\ T_{\text{out},i}(\tau) &= \left[1 - \dfrac{\mathsf{f}_i(\tau)^{n_R-1}}{c_p} \dfrac{1}{C_3 + C_4\mathsf{f}_i(\tau)^{n_R}}\right] T_{\text{in},i}(\tau) + \\ & \quad \left[\dfrac{\mathsf{f}_i(\tau)^{n_R-1}}{c_p} \dfrac{1}{C_3 + C_4\mathsf{f}_i(\tau)^{n_R}}\right] T_{\text{cpu,i}}(\tau). \end{cases} \tag{A.18}$$

Remarks:

- The state of the system is the temperature of the CPU, i.e., $T_{\text{cpu,i}}(\tau)$.
- The system is stable since $\frac{\mathsf{f}_i(\tau)^{n_R}}{C_3+C_4\mathsf{f}_i(\tau)^{n_R}} \frac{1}{C_1} > 0$ for all $\mathsf{f}_i(\tau) > 0$.
- The inputs of the system are the temperature of the air at the inlet of the server ($T_{\text{in},i}(\tau)$), the power consumption of the server ($\mathsf{p}_i(\tau)$), and the rate at which air flows into the server ($\mathsf{f}_i(\tau)$).
- The output of the system is the temperature of the air at the outlet of the server $T_{\text{out},i}(\tau)$.
- The temperature of the air at the inlet of the server affects without any delay the temperature of the air at the outlet of the server, i.e., one of the input is directly coupled with the output of the system. This is due to the fact that we neglect the time required by the airflow to move from the inlet to the outlet of the server.
- The system is nonlinear. However, if the airflow rate is constant, then the system is linear and time-invariant.
- at the equilibrium, the temperature of the CPU is given by

$$T_{\text{cpu,i}} = T_{\text{in},i} + \left[\frac{C_3 + C_4\mathsf{f}_i^{n_R}}{\mathsf{f}_i^{n_R}}\right]\mathsf{p}_i \tag{A.19}$$

and the temperature of the air at the outlet of the server is given by

$$T_{\text{out},i} = T_{\text{in},i} + \frac{1}{c_p \mathsf{f}_i} \mathsf{p}_i. \tag{A.20}$$

- At the thermal equilibrium, the temperature of the air at the outlet of the server equals the temperature predicted by the simplified model discussed in Sec. 4.3.1.
- The temperature of the CPU and the temperature of the air at the outlet of the server are inversely proportional to the air speed.
- As f_i tends to zero, the temperature of the CPU and the temperature of the output tend to infinity;
- As f_i tends to infinity, the temperature of the CPU tends to

  $$T_{\text{in},i} + C_4 \mathsf{p}_i$$

  and the temperature of the airflow at the outlet tends to $T_{\text{in},i}$.

At every time τ, the coefficient

$$1 - \frac{\mathsf{f}_i(\tau)^{n_R - 1}}{c_p} \frac{1}{C_3 + C_4 \mathsf{f}_i(\tau)^{n_R}}$$

is positive. This implies the following bound on $\mathsf{f}_i(\tau)$

$$C_3 \mathsf{f}_i(\tau)^{1-n_R} + C_4 \mathsf{f}_i(\tau) \geq \frac{1}{c_p}. \tag{A.21}$$

If $n_R = 1$ then (A.21) can be rewritten as

$$\mathsf{f}_i(\tau) \geq \frac{1}{C_4} \left[\frac{1}{c_p} - C_3 \right].$$

If $n_R = 2$ then (A.21) leads to the following system of inequalities

$$\begin{cases} 0 & \leq \mathsf{f}_i(\tau) \leq \quad f_{i,1} \\ \max\{0, \mathsf{f}_{i,2}\} & \leq \mathsf{f}_i(\tau) \end{cases},$$

where

$$\mathsf{f}_{i,1,2} = \frac{\frac{1}{c_p} \pm \sqrt{\Delta}}{2C_4}, \qquad \Delta = \frac{1}{c_p^2} - 4C_4C_3.$$

A.4 Estimating the server power consumption

In this section we discuss a linear model for estimating the power consumption of a server. The results discussed in this section are obtained in a joint work with Anshul Gandhi and Kevin Woo.

Server power consumption model. Let $\mathsf{p}(k)$ be the average power consumption of a server during the k^{th} interval. With $u_i(k)$ we denote the relative amount of time that the CPU spent in the i^{th} frequency during the k^{th} interval. The model we proposed can be written as

$$\mathsf{p}(k) = \mathsf{p}_C + \sum_{i=1}^{N} \alpha_i u_i(k), \tag{A.22}$$

where $\mathsf{p}_{C,i}$ is the constant part of the server power consumption and $\{\alpha_i\}$ are the coefficients which maps CPU utilization to server power consumption.

Servers used for the experiments. Two servers are used for the experiments: cn007 and PH

- **cn007** is a DELL PowerEdge 1950 1U server equipped with two Quad-Core Intel Xeon 5355, two 300GB hard disks SEAGATE ST3300555SS spinning at 15Krpm, and 8GB of memory at 667MHz (1.5ns) divided into 4 banks of 2GB each. Network connectivity is provided by two Broadcom NetXtreme II BCM5708 1000Base-T (B2) chips. The installed OS is Linux 64bit, kernel version 2.6.24 compiled against gcc

4.2.3.

- **PH** is a SuperMicro X6DHR-8G2 1U server equipped with a two Intel Xeon@3GHz, 250GB four hard disks SEAGATE ST3250823AS spinning at 7200 rpm, and 2GB of memory at 333MHz (3.0ns). Network connectivity is provided by two Intel PRO/1000 chips. The OS is a 64bit Linux kernel version 2.6.24 patched with perfmon2 support.

A collection of bash script is used to stress the use of different components of the servers, e.g., memory, network, CPU, and hard disks. The average utilization of the components is measured during every sample time, but only the CPU statistics are used for estimating the total server power consumption.

The power consumption of the servers is measured via two Watts up? PRO power meters produced by Electronic Educational Devices.[3] The power meters are capable of measuring power consumption within an accuracy of 1.5 W at 210V.

Experimental results. We observed that a relatively small delay in the interval considered for measuring the average server power consumption and the interval considered for measuring the CPU statics, can significantly impact the power estimation error. To counteract such a delay, we modified the model in (A.22) as follow

$$\mathsf{p}(k) = \mathsf{p}_C + \sum_{i=1}^{N} \alpha_i u_i(k) + \sum_{i=1}^{N} \beta_i u_i(k-1). \tag{A.23}$$

After a calibration step, we obtained an estimation error of 4.03 (W) at the 90^{th} percentile. Table A.1 summarizes the statistics about the estimation error. Figure A.1 shows the estimated power consumption of cn007 and the measured one. Figure A.2 shows the sampled power density function of the estimation error and Tab. A.1 shows the statics of the estimation error.

Results obtained in these experiment are not conclusive, since we were unable to test the power consumption of the server under a larger variety of workload cases and to test

[3]`www.wattsupmeters.com` .

Table A.1: Estimation error statistics. Values expressed in [W].

Mean	Std. deviation	90^{th} percentile
0	3.67	4.03

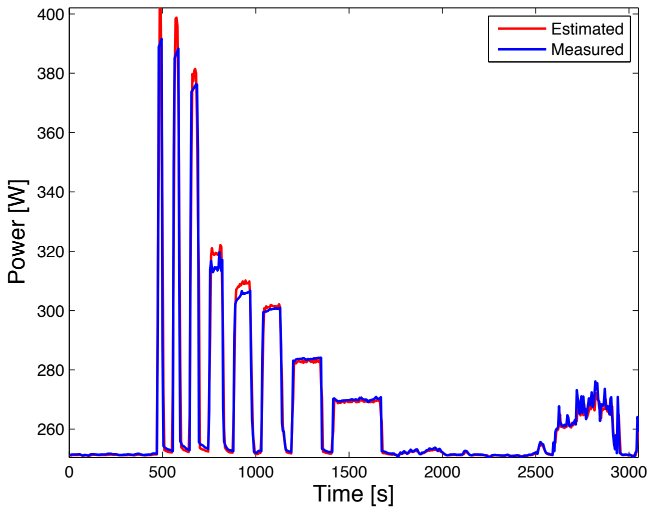

Figure A.1: Measured and estimated server power consumption.

the model power consumption for multiple servers. However, the results we obtained suggest that the sole observation of the CPU utilization is sufficient to provide accurate estimation of the total server power consumption. The utilization of the CPU have to include the percentage of the time the CPU is used at different frequencies. Also, since the measure collected by the power meter and the statistics about CPU utilization may be unsynchronized, the model has also to account for the CPU utilization measured one step in the past.

A.5 Invertibility of the matrix $(I - \Psi_{[\mathcal{Z},\mathcal{Z}]})$

This section discusses the proof of the invertibility of the matrix $(I - \Psi_{[\mathcal{Z},\mathcal{Z}]})$ defined in Sec. 4.3.4. Towards this goal, we consider a square matrix X, whose elements are bounded in the interval $[0, 1]$ and such that the sum of its elements along every row is lower than 1. Let us denote with N the number of rows and columns of X. With $x_{i,j}$ we denote the

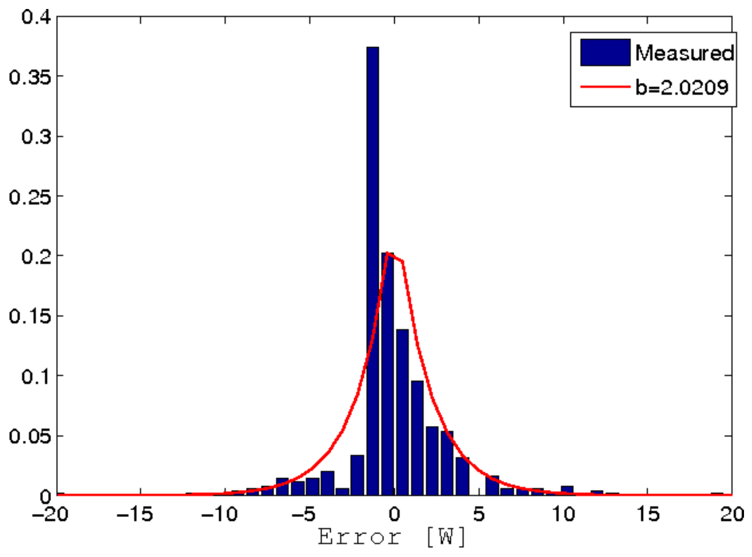

Figure A.2: Probability density function of the estimation error.

element in the i^{th} row and in the j^{th} column of X.

Proposition 6. The absolute value of every eigenvalue of X is lower than 1.

Proof. Let us denote with $D(x_{i,i}, r_i)$ the disc centered in $x_{i,i}$ and whose radius is r_i, where $r_i \geq 0$. For every $i \in \{1, \dots, N\}$ we set $r_i = \sum_{\substack{j=1 \\ j \neq i}}^{N} |x_{i,j}| = \sum_{\substack{j=1 \\ j \neq i}}^{N} x_{i,j}$, where the second equality is due to the fact that every element of X is nonnegative. The sum of the elements of X along every row is lower than 1 and hence, $r_i < 1 - x_{i,i}$ for all $i \in \{1, \dots, N\}$. For the Gershgorin theorem, we can conclude that no eigenvalue of X can have absolute value larger than or equal to 1. $\square$

Lemma 1. The matrix $(I - X)$ is invertible.

Proof. The eigenvalues of $(I - X)$ equals the eigenvalues of $-X$ shifted by 1. Since all of the eigenvalues of the matrix X have absolute value lower than 1, none of the eigenvalues of $(I - X)$ may be 0 and therefore, the matrix $(I - X)$ is invertible. $\square$

To prove that the matrix $(I - \Psi_{[\mathcal{Z},\mathcal{Z}]})$ is invertible, it is sufficient to note that the matrix $\Psi_{[\mathcal{Z},\mathcal{Z}]}$ enforces the assumptions about the matrix X discusses at the beginning of the section.

A.6 Stability of the matrix $A_{T,ct}$

As discussed in Sec. 4.3.4, the matrix $A_{T,ct}$ is defined as

$$A_{T,ct} = diag\{-\mathbf{k}\} + \begin{bmatrix} diag\{\mathbf{k}_{\mathcal{N}}\} & & \\ & diag\{\mathbf{0}\} & \\ & & diag\{\mathbf{k}_{\mathcal{E}_1}\} \end{bmatrix} \Psi_{[\mathcal{N}\cup\mathcal{C}\cup\mathcal{E}_1,\mathcal{N}\cup\mathcal{C}\cup\mathcal{E}_1]} \qquad \text{(A.24)}$$

Assume that

- the coefficients $\{\mathbf{k}_i\}$ are positive;
- the sum along every row of the elements of matrix $\Psi_{[\mathcal{N}\cup\mathcal{E}_1,\mathcal{N}\cup\mathcal{E}_1]}$ is lower than 1.

Proposition 7. All of the eigenvalues of the matrix $A_{T,ct}$ are lower than 0.

Proof. Consider a change of coordinates, so that we can write

$$\tilde{A}_{T,ct} = diag\{-\mathbf{k}\} + \begin{bmatrix} diag\{\mathbf{k}_{\mathcal{N}}\} & & \\ & diag\{\mathbf{k}_{\mathcal{E}_1}\} & \\ & & diag\{\mathbf{0}\} \end{bmatrix} \begin{bmatrix} \Psi_{[\mathcal{N},\mathcal{N}]} & \Psi_{[\mathcal{N},\mathcal{E}_1]} & \Psi_{[\mathcal{N},\mathcal{C}]} \\ \Psi_{[\mathcal{E}_1,\mathcal{N}]} & \Psi_{[\mathcal{E}_1,\mathcal{E}_1]} & \Psi_{[\mathcal{E}_1,\mathcal{C}]} \\ \Psi_{[\mathcal{C},\mathcal{N}]} & \Psi_{[\mathcal{C},\mathcal{E}_1]} & \Psi_{[\mathcal{C},\mathcal{C}]} \end{bmatrix} \qquad \text{(A.25)}$$

The stability properties of $A_{T,ct}$ and $\tilde{A}_{T,ct}$ are the same. We define the vector

$$diag\{\mathbf{k}_{\mathcal{Z}}\} \triangleq \begin{bmatrix} diag\{\mathbf{k}_{\mathcal{N}}\} & \\ & diag\{\mathbf{k}_{\mathcal{E}_1}\} \end{bmatrix}$$

and the matrices

$$\Psi_{[\mathcal{Z},\mathcal{Z}]} \triangleq \begin{bmatrix} \Psi_{[\mathcal{N},\mathcal{N}]} & \Psi_{[\mathcal{N},\mathcal{E}_1]} \\ \Psi_{[\mathcal{N},\mathcal{E}_1]} & \Psi_{[\mathcal{E}_1,\mathcal{E}_1]} \end{bmatrix}, \quad \Psi_{[\mathcal{Z},\mathcal{C}]} \triangleq \begin{bmatrix} \Psi_{[\mathcal{N},\mathcal{C}]} \\ \Psi_{[\mathcal{E}_1,\mathcal{C}]} \end{bmatrix}.$$

The matrix $\tilde{A}_{T,ct}$ can now be written as

$$\tilde{A}_{T,ct} = \begin{bmatrix} diag\{\mathsf{k}_{\mathcal{Z}}\}(\Psi_{[\mathcal{Z},\mathcal{Z}]} - I) & diag\{\mathsf{k}_{\mathcal{Z}}\}\Psi_{[\mathcal{Z},\mathcal{C}]} \\ \\ 0 & diag\{-\mathsf{k}_{\mathcal{C}}\} \end{bmatrix}.$$

The matrix $\tilde{A}_{T,ct}$ is block diagonal and its eigenvalues are the eigenvalues of its diagonal matrices. Since $\mathsf{k}_i > 0$ for $i \in \mathcal{C}$, then the matrix $diag\{-\mathsf{k}_{\mathcal{C}}\}$ has all negative eigenvalues. We can focus on the matrix $diag\{\mathsf{k}_{\mathcal{Z}}\}(\Psi_{[\mathcal{Z},\mathcal{Z}]} - I)$. From the hypothesis, the sum along any row i of the elements of the matrix $\Psi_{[\mathcal{Z},\mathcal{Z}]}$ is lower than 1 and therefore $\sum_{j=1}^{N+E_1} \psi_{i,j} < 1$, which holds for every $i \in \{1, \ldots, N + E_1\}$. For the Gershgorin theorem we can conclude that all of the eigenvalues of $diag\{\mathsf{k}_{\mathcal{Z}}\}(\Psi_{[\mathcal{Z},\mathcal{Z}]} - I)$ are strictly negative. Therefore, the matrix $\tilde{A}_{T,ct}$ and the matrix $A_{T,ct}$ have all negative eigenvalues.

$\square$

A.7 Proof of the stability the equilibrium point discussed in Sec. 4.5

We consider the case where no environment nodes are in the data center. Considering environment nodes in the data center does not change the proof, but it makes necessary the use of a more complex notation. Under the hypothesis discussed in Sec. 4.5, the data center dynamics can be written as

$$\begin{cases} \dot{\boldsymbol{l}}(\tau) & = & \boldsymbol{\lambda} - \boldsymbol{\eta}(\tau) \\ \begin{bmatrix} \dot{\boldsymbol{T}}_{\text{out},\mathcal{N}}(\tau) \\ \dot{\boldsymbol{T}}_{\text{out},\mathcal{C}}(\tau) \end{bmatrix} & = & \mathsf{A}_T \begin{bmatrix} \boldsymbol{T}_{\text{out}\mathcal{N}}(\tau) \\ \boldsymbol{T}_{\text{out}\mathcal{C}}(\tau) \end{bmatrix} + \mathsf{B}_T \begin{bmatrix} \mathbf{p}_{\mathcal{N}}(\tau) \\ \boldsymbol{T}_{\text{ref}} \end{bmatrix} \\ \mathbf{p}_{\mathcal{N}}(\tau) & = & \mathsf{A}_\alpha \boldsymbol{\eta}(\tau) + \mathsf{B}_\beta \boldsymbol{l}(\tau) \\ & & \text{For all } i \in \mathcal{N},\ j \in \{1, \dots, J\} \\ \eta_i^j(\tau) & = & \begin{cases} \mu_i^j & \text{if } l_i^j(\tau) > 0 \\ \lambda_i^j & \text{if } l_i^j(\tau) = 0 \end{cases} \end{cases}, \tag{A.26}$$

where we removed the explicit dependency over the time for those variables that are constant. The matrix A_T and B_T are defined as

$$\mathsf{A}_T = \begin{bmatrix} diag\{-\mathbf{k}_{\mathcal{N}}\}(I - \Psi_{[\mathcal{N},\mathcal{N}]}) & diag\{\mathbf{k}_{\mathcal{N}}\}\Psi_{[\mathcal{N},\mathcal{C}]} \\ & diag\{-\mathbf{k}_{\mathcal{C}}\} \end{bmatrix}$$

and

$$\mathsf{B}_T = \begin{bmatrix} diag\{\mathbf{c}_{\mathcal{N}}\} & \\ & diag\{\boldsymbol{T}_{\text{ref}}\} \end{bmatrix}.$$

The equilibrium point of the data center is

$$\begin{cases} \bar{\boldsymbol{l}} & = & \mathbf{0} \\ \overline{\boldsymbol{T}_{\text{out}\mathcal{N}}} & = & (I - \Psi_{[\mathcal{N},\mathcal{N}]})^{-1}\Psi_{[\mathcal{N},\mathcal{C}]}\boldsymbol{T}_{\text{ref}} + \\ & & (I - \Psi_{[\mathcal{N},\mathcal{N}]})^{-1} diag\{\mathbf{k}_{\mathcal{N}}\}^{-1} diag\{\mathbf{c}_{\mathcal{N}}\}\mathbf{p}_{\mathcal{N}} \\ \overline{\boldsymbol{T}_{\text{out}\mathcal{C}}} & = & \boldsymbol{T}_{\text{ref}}. \end{cases} \tag{A.27}$$

Let us define the vectors

$$\boldsymbol{x}(\tau) \triangleq \begin{bmatrix} \boldsymbol{l}(\tau) \\ \boldsymbol{T}_{\text{out}\mathcal{N}}(\tau) \\ \boldsymbol{T}_{\text{out}\mathcal{C}}(\tau) \end{bmatrix}, \qquad \bar{\boldsymbol{x}} \triangleq \begin{bmatrix} \bar{\boldsymbol{l}} \\ \overline{\boldsymbol{T}_{\text{out}\mathcal{N}}} \\ \overline{\boldsymbol{T}_{\text{out}\mathcal{C}}} \end{bmatrix}.$$

Since no environment nodes are considered in the data center, the vector $\boldsymbol{x}(\tau)$ and $\bar{\boldsymbol{x}}$ can also be written as

$$\boldsymbol{x}(\tau) \triangleq \begin{bmatrix} \boldsymbol{l}(\tau) \\ \boldsymbol{T}_{\text{out}}(\tau) \end{bmatrix}, \qquad \bar{\boldsymbol{x}}(\tau) \triangleq \begin{bmatrix} \bar{\boldsymbol{l}} \\ \overline{\boldsymbol{T}_{\text{out}}}(\tau) \end{bmatrix}.$$

The vector $\boldsymbol{x}(\tau)$ is the state of the data center and the vector $\bar{\boldsymbol{x}}$ is the equilibrium point, whose stability properties we want to analyze. Let us denote with D_0 the set of initial points that satisfy the hypothesis discussed in Sec. 4.5. In order to prove that $\bar{\boldsymbol{x}}$ is an asymptotically stable equilibrium point of (A.26) we have to prove that

- $\bar{\boldsymbol{x}}$ is an equilibrium point;
- as τ tends to infinity and for all initial points $\boldsymbol{x}(0) \in D_0$, $\boldsymbol{x}(\tau)$ tends to $\bar{\boldsymbol{x}}$, ;
- given a bound for the maximum distance between $\boldsymbol{x}(\tau)$ and $\bar{\boldsymbol{x}}(\tau)$, there exist a neighborhood $N(\bar{\boldsymbol{x}})$ of $\bar{\boldsymbol{x}}$, such that for all $\boldsymbol{x}(0) \in N(\bar{\boldsymbol{x}})$ the distance between $\boldsymbol{x}(\tau)$ and $\bar{\boldsymbol{x}}$ never exceeds the given bound.

To prove that (A.27) is an equilibrium point for (A.26), it is sufficient to note that, if $\boldsymbol{x}(0) = \bar{\boldsymbol{x}}$, then $\dot{\boldsymbol{x}}(\tau) = 0$ for all $\tau \geq 0$.

To prove the second point we consider the solution of the the system of differential equations in (A.26). Under the hypothesis discusses in Sec. 4.5, the system of differential equations in (A.26) has a unique solution, which can be written as

$$\left\{\begin{array}{lcl}
\begin{bmatrix}\boldsymbol{T}_{\text{out}\mathcal{N}}(\tau)\\ \boldsymbol{T}_{\text{out}\mathcal{C}}(\tau)\end{bmatrix} & = & e^{\mathsf{A}_T\tau}\begin{bmatrix}\boldsymbol{T}_{\text{out}\mathcal{N}}(0)\\ \boldsymbol{T}_{\text{out}\mathcal{C}}(0)\end{bmatrix}+\int_0^\tau e^{\mathsf{A}_T(\tau-t)}\mathsf{B}_T\begin{bmatrix}\mathbf{p}_{\mathcal{N}}(\tau)\\ \boldsymbol{T}_{\text{ref}}\end{bmatrix}dt \\
 & & \text{For all } i\in\mathcal{N},\ j\in\{1,\ldots,J\} \\
l_i^j(\tau) & = & \begin{cases}(\lambda_i^j-\mu_i^j)\tau+l_i^j(0) & 0\le\tau<\dfrac{l_i(0)^j}{\mu_i^j-\lambda_i^j}\\ 0 & \dfrac{l_i^j(0)}{\mu_i^j-\lambda_i^j}\le\tau\end{cases} \\
\mathsf{p}_i(\tau) & = & \sum_{j=1}^J \mathsf{p}_i^j(\tau) \\
\mathsf{p}_i^j(\tau) & = & \begin{cases}\alpha_i^j\mu_i^j+\beta_i^j l_i^j(0)-\beta_i^j(\mu_i^j-\lambda_i^j)\tau & 0\le\tau<\dfrac{l_{i,0}^j}{\mu_i^j-\lambda_i^j}\\ \alpha_i^j\lambda_i^j & \dfrac{l_i^j(0)}{\mu_i^j-\lambda_i^j}\le\tau\end{cases}
\end{array}\right. . \qquad \text{(A.28)}$$

It is easy to verify that the state variables in (A.28) tend to $\bar{\boldsymbol{x}}$. Since for all $\boldsymbol{x}(0)\in D_0$, (A.28) is the only solution of (A.26), then $\lim_{\tau\to+\infty}\boldsymbol{x}(\tau)\to\bar{\boldsymbol{x}}$ for all $\boldsymbol{x}(0)\in D_0$.

We now discuss a proof for the last point. Consider a new set of coordinates such that $\bar{\boldsymbol{x}}$ is at the origin. Let $\tilde{\boldsymbol{x}}(\tau)$ be the value of the vector $\boldsymbol{x}(\tau)$ in the new set of coordinates. For example, $\tilde{\boldsymbol{x}}(\tau)=\boldsymbol{x}(\tau)-\bar{\boldsymbol{x}}$. Note that, since $\bar{\boldsymbol{l}}=\mathbf{0}$, then $\tilde{\boldsymbol{l}}(\tau)=\boldsymbol{l}(\tau)$. Let $d(\tau)$ be the distance between $\boldsymbol{x}(\tau)$ and $\bar{\boldsymbol{x}}$ at time τ. In the new set of coordinates, $d(\tau)$ equals $||\tilde{\boldsymbol{x}}(\tau)||$ and we can write

$$d(\tau)=||\boldsymbol{l}(\tau)||+||\tilde{\boldsymbol{T}}_{\text{out}}(\tau)||.$$

The maximum value of $||\boldsymbol{l}(\tau)||$ can be made arbitrarily small by an appropriate choice of $\boldsymbol{l}(0)$. We now have to prove that also the maximum of $d_T(\tau)\triangleq||\tilde{\boldsymbol{T}}_{\text{out}}(\tau)||$ can be made arbitrarily small.

Toward this goal, we show that

- $d_T(\tau)$ has a maximum of finite value for some $\tau\ge 0$. We call $\bar{d}_T$ such a maximum.

- $\bar{d}_T$ is Lipschitz continuous in $\tilde{\boldsymbol{x}}(0)$. Hence, given a bound on $\bar{d}_T$, we can always find a neighborhood $N(\mathbf{0})$ such that for all $\tilde{\boldsymbol{x}}(0) \in N(\mathbf{0})$, $\bar{d}_T$ is smaller than the given bound.

From (A.28), we note that all of the functions, but one, composing $\boldsymbol{T}_{\text{out}}(\tau)$ are continuous over τ and bounded. The function $\begin{bmatrix} \mathbf{p}_{\mathcal{N}}(\tau) \\ \boldsymbol{T}_{\text{ref}} \end{bmatrix}$ is piece-wise continuous over τ and bounded. Since $\begin{bmatrix} \mathbf{p}_{\mathcal{N}}(\tau) \\ \boldsymbol{T}_{\text{ref}} \end{bmatrix}$ affects $\boldsymbol{T}_{\text{out}}(\tau)$ through a convolution operation, we can still conclude that $\boldsymbol{T}_{\text{out}}(\tau)$ is continuous over τ and bounded. This implies that also $d_T(\tau)$ is continuous over τ and bounded. Hence $d_T(\tau)$ has a maximum of finite value for some $\tau \geq 0$. Let $\bar{d}_T$ be the maximum of $d_T(\tau)$ for some $\tau \geq 0$.

For all $\tau \geq 0$, $\boldsymbol{T}_{\text{out}}(\tau)$ is a continuous function of the vector $\boldsymbol{x}(0)$. Therefore, $\tilde{\boldsymbol{T}}_{\text{out}}(\tau)$ is a continuous function of $\tilde{\boldsymbol{x}}(0)$. This implies that $\bar{d}_T$ is a continuous over $\tilde{\boldsymbol{x}}(0)$ and hence, Lipschitz continuous in $\tilde{\boldsymbol{x}}(0)$. This is sufficient to prove that given a bound on $\bar{d}_T$, we can always find a neighborhood $N(\mathbf{0})$ of $\mathbf{0}$, such that for all $\tilde{\boldsymbol{x}}(0) \in N(\mathbf{0})$, $\bar{d}_T$ is smaller than the given bound. This proves that $\mathbf{0}$ is a stable equilibrium point for the evolution of $\tilde{\boldsymbol{x}}(\tau)$ and hence, $\bar{\boldsymbol{x}}$ is a stable equilibrium point for the evolution of $\boldsymbol{x}(\tau)$.

A.8 Proof of Proposition 2 in Sec. 8.3

The following is the proof of proposition 2.

Proof. Let $\hat{\boldsymbol{v}}(k|k)^{\star}$ be a solution of (8.31) and consider $\tilde{\boldsymbol{u}}(k|k) = [\tilde{\boldsymbol{u}}_1^T(k|k), \ldots, \tilde{\boldsymbol{u}}_{\mathsf{S}}^T(k|k)]^T$

such that each sub-vector $\tilde{\boldsymbol{u}}_i^T(k|k)$ belongs to the feasible set of

$$\begin{aligned}
&\min_{\hat{\boldsymbol{u}}_i(k|k)} \boldsymbol{c}_{i,u}^T(k)\hat{\boldsymbol{u}}_i(k|k)\\
&\textbf{s.t.}\\
&\underline{\boldsymbol{u}}_i \leq \hat{\boldsymbol{u}}_i(k|k) \leq \overline{\boldsymbol{u}}_i,\\
&H_i\hat{\boldsymbol{u}}_i(k|k) \leq \mathbf{1}\\
&F_iB_i(k)\hat{\boldsymbol{u}}_i(k|k) + \textstyle\sum_{\substack{j=1\\j\neq i}}^{\mathsf{S}} F_{i,z_j}\hat{\boldsymbol{v}}_j^\star(k|k) + \mathbf{k}_i(k) \leq \overline{\boldsymbol{z}}_i,\\
&F_{j,z_i}M_i(k)\hat{\boldsymbol{u}}_i(k|k) \leq F_{j,z_i}\hat{\boldsymbol{v}}_i^*(k|k)\\
&\qquad\qquad\qquad\qquad \text{for all } i,j = 1,\ldots,\mathsf{S},\ i \neq j.
\end{aligned} \tag{A.29}$$

Therefore, for all $i = 1,\ldots,\mathsf{S}$

$$F_iB_i(k)\tilde{\boldsymbol{u}}_i(k|k) + \sum_{\substack{j=1\\j\neq i}}^{\mathsf{S}} F_{i,z_j}G_jB_j\tilde{\boldsymbol{u}}_j^\star(k|k) + \mathbf{k}_i(k) \leq \overline{\boldsymbol{z}}_i. \tag{A.30}$$

This implies that the vector $\tilde{\boldsymbol{u}}(k|k)$ is a feasible point for (8.30).

We now prove that the feasible set of (8.32) is contained in the feasible set of (A.29). Let $\bar{\boldsymbol{u}}(k|k) = [\bar{\boldsymbol{u}}_1^T(k|k),\ldots,\bar{\boldsymbol{u}}_{\mathsf{S}}^T(k|k)]^T$ such that each sub-vector $\bar{\boldsymbol{u}}_i^T(k|k)$ is a feasible point for the ith problem in (8.32). Then we have

$$\begin{aligned}
&\underline{\boldsymbol{u}}_i \leq \hat{\boldsymbol{u}}_i(k|k) \leq \overline{\boldsymbol{u}}_i\\
&H_i\hat{\boldsymbol{u}}_i(k|k) \leq \mathbf{1}\\
&F_iB_i(k)\hat{\boldsymbol{u}}_i(k|k) + \sum_{\substack{j=1\\j\neq i}}^{\mathsf{S}} F_{i,z_j}G_jB_j(k)\hat{\boldsymbol{u}}_j(k|k) \leq \overline{\boldsymbol{z}}_i - \mathbf{k}_i(k)\\
&\hat{\boldsymbol{v}}_i^*(k|k) = G_iB_i(k)\hat{\boldsymbol{u}}_i(k|k)
\end{aligned}$$

and hence, every $\bar{\boldsymbol{u}}_i^T(k|k)$ is a feasible point for (A.29). □