

Capelin

Fast Data-Driven Capacity
Planning for Cloud Datacenters

Georgios Andreadis

Capelin

Data-Driven Capacity Planning for Cloud Datacenters

by

Georgios Andreadis

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Thursday August 27, 2020 at 10:00 AM.

Student number: 4462254
Project duration: November 11, 2019 – August 27, 2020
Thesis committee: Prof. dr. ir. D.H.J. Epema, TU Delft, chair
Dr. G. Gousios, TU Delft, committee member
Prof. dr. ir. A. Iosup, VU Amsterdam and TU Delft, daily supervisor
Ir. V. van Beek, TU Delft and Solvinity, external expert
Dr. Z. Erkin, TU Delft, Board of Examiners

An electronic version of this thesis is available at <http://repository.tudelft.nl/>.

Abstract

Cloud datacenters provide a backbone to our digital society. Crucial to meeting increasing demand while maintaining efficient operation is the activity of capacity planning. Inaccurate capacity planning for cloud datacenters can lead to significant performance degradation, denser targets for failure, and unsustainable energy consumption. Although this activity is core to improving cloud infrastructure, relatively few comprehensive approaches and support tools exist, leaving many planners with merely rule-of-thumb judgement.

We propose Capelin, a data-driven, scenario-based capacity planning system for cloud datacenters. We design Capelin to address requirements we have derived from a unique survey of experts in charge of diverse datacenters in several countries. Capelin introduces the notion of portfolios of scenarios, which it leverages in its probing for alternative capacity-plans. At the core of the system, a trace-based, discrete-event simulator enables the exploration of different possible topologies, with support for scaling the volume, variety, and velocity of resources, and for horizontal (scale-out) and vertical (scale-up) scaling. The approach centers around a notion of portfolios of scenarios as a framework for probing alternative decisions and courses of events. Capelin gives detailed quantitative operational information for each scenario, which could facilitate human decisions in capacity planning.

We implement and open-source Capelin, and show through comprehensive trace-based experiments it can aid practitioners. Although Capelin is designed to work across many kinds of datacenters, in this work we focus on private-cloud, business-critical workloads, and on public-cloud operations. The results give evidence that choices that seem reasonable and common in practice could be worse by a factor of 1.5-2.0 than the best, in terms of performance degradation or energy consumption. We also show evidence of Capelin identifying meaningful choices that are different from the baseline proposed by a team of professional data-center engineers. We open-source Capelin and release data artifacts for public inspection and reuse.

Preface

This thesis is the culmination of five years of education and research at Delft University of Technology. It was a long and challenging journey, but I am very grateful to have been able to traverse it. Along the way, I have had the pleasure of working with many talented and kind people, some of whom I would like to thank especially here.

First, I would like to thank my two supervisors, Alexandru Iosup and Vincent van Beek. Alexandru, it was your course in the Bachelor that sparked my interest in Computer Science research. Throughout my time as Honours student and later as Master student, I have been incredibly fortunate to benefit from your inspiration, motivation, and feedback. I am also very grateful for the many doors you have opened for me, even as I was at the early beginning of my academic career. Vincent, your guidance and level-headed support has been invaluable to me. From real-world technical know-how, to feedback on my scientific endeavours, I learned a lot. Both of you also frequently reminded me not to forget to take break, which was greatly appreciated. Thank you both!

Second, I would like to thank the people at Solvinity for facilitating this research project and embedding me in their company. Few companies at Solvinity's line and scale of business facilitate such kind of open-ended, open-access research collaborations, and I am very grateful for the opportunity. Thank you to Bas Demmink and Jeffrey Minneboo, for lending your expertise. A special thanks to the team I was embedded in, for taking me in and making my Mondays at the office a welcome change from working from home (in pre-pandemic days).

Third, I would like to thank the practitioners that I was fortunate enough to interview for my community survey. I may not be able to list your names or affiliations here, due to the confidentiality of the study, but that does not diminish my gratitude. Thank you for your willingness to participate and for your openness.

Fourth, I extend my sincere thanks to the many researchers, junior and senior, that I have worked with in the @Large Research team. Thank you all for the many chats and brainstorm sessions. Thanks especially to Laurens for the use of your (amazing) suite of tools and Jesse for the many encouraging words. A big thank you to the whole of Team OpenDC. Any large software project is a team effort and I am glad to have been able to lead this particular one. To Leon and Matthijs: thank you for your tireless efforts and your companionship in the first stages of the project. To the many people working on and around the simulator: thank you for your many contributions to this project. I would like to extend a special thank you to Fabian Mastebroek, my co-author and core OpenDC team colleague. I am very grateful to have been able to collaborate with you on several projects. Your continued resolve, critical insights, and spot-on Kotlin skills were a source of inspiration for me. This project in particular and OpenDC as a whole wouldn't be where they are now, without your tireless efforts.

Last, but certainly not least, I would like to thank my friends and family. You never fail to put a smile on my face, and I am very grateful for that. Thank you for being there for me and for supporting me through this journey.

*Georgios Andreadis
Delft, August 2020*

Contents

1	Introduction	1
1.1	Primer on Capacity Planning for Cloud Infrastructure	2
1.2	Problem Statement	3
1.3	Research Questions	3
1.4	Approach	4
1.5	Development and Dissemination	5
1.6	Guidelines for Reading	5
2	State-of-the-Art	7
2.1	System Model for Datacenter Operations	7
2.2	Capacity Planning Across Domains	9
2.3	Capacity Planning for Computer Systems	10
2.4	Capturing Capacity Planning Practice.	14
3	Systematic Literature Survey of Capacity Planning	17
3.1	Overview	17
3.2	Method	17
3.3	Taxonomy of Capacity Planning Approaches	20
3.4	Systematic Map of the Capacity Planning Literature	21
3.5	Meta-Analysis of the Field.	25
3.6	Comparison to Theory	30
3.7	Discussion	31
4	Real-World Survey of Capacity Planning	35
4.1	Overview	35
4.2	Method	36
4.3	Main Observations from the Interviews.	37
4.4	Full Observations from the Interviews	39
4.5	Discussion	49
5	Design of Capelin: A Capacity Planning System for Cloud Infrastructure	51
5.1	Overview	51
5.2	Requirements Analysis	51
5.3	Overview of the Capelin Architecture	53
5.4	Portfolio Abstraction for Capacity Planning	55
5.5	Discussion	56
6	Evaluation of Capelin, through Experiments with a Real-World Prototype	59
6.1	Overview	59
6.2	Implementation of a Software Prototype	59
6.3	Experiments with Capelin.	66
6.4	Discussion	73
7	Conclusion and Future Work	75
7.1	Conclusion	75
7.2	Future Work.	76
A	Capacity Planning Interview Script	79
A.1	Part 1: Overview (15')	79
A.2	Part 2: The Process (15').	80
A.3	Part 3: Inside Factors (15').	80
A.4	Part 4: Outside Factors (10').	80
A.5	Part 5: Summary and Follow-Up (5')	81

B	External Validation of the Simulation	83
B.1	How to ensure simulator outputs are valid?	83
B.2	How to ensure simulator outputs are sound?	83
B.3	How to ensure no regression in subsequent simulator versions?	85
C	Full Visual Experiment Results	87
D	Full Tabular Experiment Results	111
	Bibliography	121



Introduction

Cloud datacenters are comprised of large numbers of powerful computers. They are part of the backbone of today's increasingly digital society [40, 42, 43], serving users across industry, government, and academia. These users have come to expect high quality of service and in particular near-infinite scalability, but at a low cost with near-zero unavailability. Similarly to other infrastructure that our society relies on, the cloud datacenter infrastructure requires careful capacity planning, lest too little or too much of it exists. Planning the capacity of the cloud infrastructure is a critical yet non-trivial optimization problem which could lead to significant service improvements, cost savings, and environmental sustainability [11]. Although many approaches to the capacity-planning problem have been published [23, 114, 139], companies still rely on rule-of-thumb reasoning for decisions, and possibly on in-house, closed-source tools. To minimize operational risks, many industry approaches currently lead to significant overprovisioning [45], or miscalculate the balance between underprovisioning and overprovisioning [98]. Both underprovisioning and overprovisioning are undesirable. In this work, we approach the problem of capacity planning for cloud datacenters with a semi-automated, specialized, data-driven tool for decision making.

The practice of capacity planning is not unique to the digital infrastructure domain. It appears in areas as diverse as planning the production capacity for a factory, the intervention capacity of a hospital or the national healthcare system, the operational infrastructure for the national train network, etc. Production and operations management research gives a theoretical foundation for the capacity planning processes involved in such industrial settings [89]. In the theory of capacity planning developed already in these fields, we observe that the solution is often a highly inter-disciplinary and multi-factor process for decision-making. Although this broader capacity planning research has preceded the field of cloud (and other service-oriented) computational infrastructure as a whole, in time, its awareness and findings do not seem to have permeated yet to the computational capacity planning process.

The main responsibility of a cloud capacity planner is to ensure that adequate infrastructure is in place for incoming workloads. These workloads consist of computational tasks submitted by users of the cloud, such as a Virtual Machine (VM) or even a workflow of smaller tasks. The capacity planner needs to acquire and install (*provision*) sufficient resources in advance, in order to be able to host all incoming workloads. This requires accurately estimating the needed capacity for years in advance and shaping the infrastructure topology to meet demand while keeping costs and environmental impact low.

The cloud offers many layers of services, from Infrastructure as a Service (IaaS) to Software as a Service (SaaS) [90]. Although important, SaaS currently seems very fragmented and difficult to address systematically. Instead, this thesis focuses on capacity planning for mid-tier (small to medium-scale) cloud infrastructure providers operating at the low- to mid-level tiers of the service architecture. Very similar to industrial capacity planning problems, these providers make decisions in highly multi-disciplinary and complex situations, as shown for example by our systematic survey of the state of the field (see Chapter 3) and our series of community interviews (see Chapter 4). Beyond the computational needs which are traditionally derived from high-level principles and workload-related considerations, the capacity of cloud infrastructure relates to many different aspects, including heterogeneous hardware environments, non-functional constraints, and contractual considerations. As our community interviews show, capacity planners find that existing tools do not match the complexity of the infrastructure they manage and the questions they need to ask. This can lead to a loss of service quality, or even to missed cost savings and environmentally unsustainable decisions.

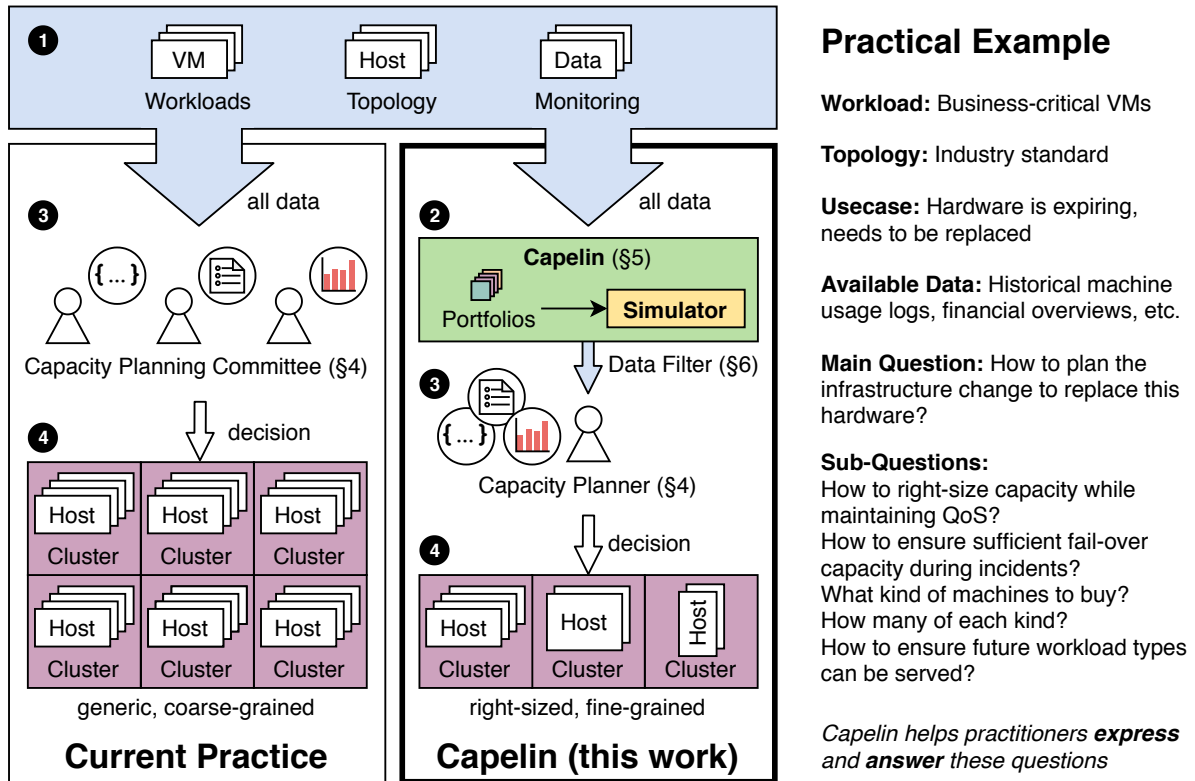


Figure 1.1: Capelin, a new, data-based capacity planning process for datacenters, compared against the current typical approach. Technical details in the practical example are introduced in §1.1 (general process) and §1.4 (our contribution), and detailed throughout the thesis. Note the current typical approach lacks step ②, which is a contribution in this thesis.

1.1. Primer on Capacity Planning for Cloud Infrastructure

The current capacity planning process in cloud infrastructures is typically conducted by a committee of stakeholders and experts. Figure 1.1 depicts this current practice alongside the approach proposed in this work. The process is conducted periodically and/or at certain events, such as pre-defined intervals in the lifecycle of resources (e.g., expected lifespan) or project milestones (e.g., the start and the middle review).

The capacity planning activity starts with modeling the current state of the infrastructure, typically using inputs such as workloads, current infrastructure topology, and large volumes of monitoring data (step ① in the figure). The workload consists of computational tasks submitted by users. The infrastructure topology is composed of one or more datacenters, each consisting of multiple clusters of machines provided by different hardware vendors. This topology is continuously monitored by hardware and software sensors, producing large volumes of monitoring data on the current status, utilization, and health of topology and workloads.

Then, a committee of various stakeholders (the *capacity planning committee*) extracts meaning from all the input data (step ③, noting step ② only appears in our approach and not in typical capacity planning processes). It typically consists of executive level representatives, technology experts, and depending on the nature of the meeting other domain experts and department representatives. In a series of meetings, this committee gathers requirements, reflects on past decisions, and forecasts future needs. Needs for Quality of Service (QoS) can at this point be expressed more concretely and with sufficient detail, as technical contracts specified as Service Level Agreements (SLAs) and Service Level Objectives (SLOs).

After a series of meetings, this leads to a decision about the infrastructure being managed (step ④). The stakeholders agree on how many new resources to provision, of which kind, and from which types of sources. For each machine, the decision could be to include a single but powerful CPU, and a single or multiple GPUs. These decisions are costly, because each CPU or GPU can cost upwards of hundreds of Euros [126]. While topology changes are at the core of this infrastructure decision, they can be augmented by operational decisions, such as strategies for how to respond to resource failures.

As depicted on the left-hand side of Figure 1.1, capacity planners are faced with large volumes of data and tasked with making complex, consequential decisions, often without dedicated tooling. As a result, topol-

ogy decisions are frequently coarse-grained and result in homogeneous clusters consisting of many identical machines. This can result in overprovisioning capacity and a waste of resources. A tool assisting capacity planners in formalizing and investigating the many possible decisions could help make more fine-grained, heterogeneous capacity decisions, custom-sizing clusters to fit the workload more closely. The right-hand side of Figure 1.1 presents such a tool-based approach, forming the core contribution of this thesis.

1.2. Problem Statement

We identify four key problems that emerge when capacity planning for cloud infrastructure providers operating at the low- to mid-level tiers of the service architecture, ranging from IaaS to Platform as a Service (PaaS) service tiers.

First, we observe a *lack of published knowledge about the current theory and practice of cloud capacity planning*. For a problem of such importance and long-lasting effects, it is surprising that no systematic literature survey of capacity planning approaches for cloud infrastructure exists. A systematic literature survey of the field could reveal the current state of the field and trends in its research directions. Equally surprising is that real-world practice is rarely surveyed. The only studies of how practitioners make and take capacity planning decisions are either over three decades old [79] or focus on non-experts deciding how to externally provision capacity for IT services [14]. A community survey of *expert* capacity planners could reveal new insights and current requirements for capacity-planning tools and techniques.

Second, we observe the *need for a flexible instrument for capacity planning, one that can address various operational scenarios*. State-of-the-art tools [52, 57, 132] and techniques [23, 44, 123] for capacity-planning operate on abstractions that match only one vendor or focus on simplistic problems. Although single-vendor tools, such as VMware's Capacity Planner [132] and IBM's Z Performance and Capacity Analytics tool [57], can provide good advice for the cloud datacenters equipped by that vendor, they do not support real-world cloud datacenters that are heterogeneous in both software [7][11, §2.4.1] and hardware [19, 37][11, §3]. Yet, to avoid vendor lock-in and licensing costs, cloud datacenters acquire heterogeneous hardware and software from multiple sources and could, for example, combine VMware's, Microsoft's, and open-source OpenStack+KVM virtualization management technology, and complement it with container technologies. Although linear programming [137], game theory [123], stochastic search [44], and other optimization techniques work well on simplistic capacity-planning problems, they do not address the many disciplines and dimensions that this problem consists of. Without adequate capacity planning tools and techniques, practitioners need to rely on rules-of-thumb calibrated with casual visual interpretation of the complex data provided datacenter monitoring. This state-of-practice likely results in overprovisioning of cloud datacenters, to avoid operational risks [45, 48]. Even then, evolving customers and workloads could make the planned capacity insufficient, leading to risks of not meeting SLAs [4, 16], inability to absorb catastrophic failures [11, p.37], and even unwillingness to accept new users.

Third, we identify the *need for comprehensive evaluations of capacity planning approaches, based on real-world data and scenarios*. The existing capacity planning tools and techniques have rarely been tested with real-world scenarios, and even more rarely with real-world operational traces that capture the detailed arrival and execution of user requests. Furthermore, for the few thus tested, the results are only rarely peer-reviewed [4, 114]. We advocate comprehensive experiments with real-world operational traces and diverse scaling scenarios to test capacity planning approaches. This allows us to recreate the real-world circumstances faced by capacity planners and put the strategies we propose to the test.

Fourth and last, we observe the *need for publicly available, comprehensive tools for capacity planning*. Access to decision support tools is critical for practitioners. However, few such tools are publicly available, and even fewer are open-source. From the available tools, none can model all the aspects needed to analyze cloud datacenters we present in Section 2.1. A tool can help practitioners match and test their intuitions with data-based insights. Especially in the face of large volumes of data, having a tool structure this data and simulate possible future scenarios can help capacity planners make better informed decisions with confidence.

1.3. Research Questions

The main objective of this thesis is to create a proof-of-concept capacity planning system for cloud infrastructure. Such a system would address the problem sketched in Section 1.2, and thus support practitioners in making decisions on the sizing and structure of their infrastructure. The problem of forecasting and decision support in this field is a non-trivial challenge. To address this challenge and achieve the main objective, we pose the following research questions in this thesis:

(RQ1) State-of-the-art: How to capture and assess the current state-of-the-art of capacity planning for cloud infrastructure?

(RQ1.1) How to systematically survey literature in the field of capacity planning?

(RQ1.2) How to design, conduct, and analyze community interviews on capacity planning with real-world practitioners?

(RQ2) Design of a system: How to design a capacity planning system for cloud infrastructure that responds to key issues faced in the community?

(RQ2.1) How to specify capacity planning problems and the known state of cloud infrastructure?

(RQ2.2) How to design a system that manipulates capacity planning problem specifications?

(RQ3) Evaluation of a system: How to evaluate a capacity planning system for cloud infrastructure?

(RQ3.1) How to design and implement a prototype meeting key aspects of the designed capacity planning system?

(RQ3.2) How to design, conduct, and analyze trace-based simulation experiments modeling cloud infrastructure?

1.4. Approach

Towards answering RQ1, we combine a literature-based theoretical analysis with a study of real-world experiences we collect from international experts. First, we conduct a systematic survey of capacity planning literature and present a taxonomy of the design space (RQ1.1). In the design of this survey, we need to choose a method of publication discovery. We see two main options: an exploratory, free traversal of literature and a systematic, structured search of public literature repositories. We choose the latter approach to ensure high recall of relevant results, and augment it with manual entry of items gathered through other means.

Second, we conduct interviews with several capacity planning practitioners from different backgrounds (RQ1.2). This brings a *human-centered* perspective to the discussion, by identifying the challenges and lessons learned of the practitioners who could be helped by a decision support system. This focus on the human practitioner is crucial, since any tool will not operate autonomously and will be used by a human operator. The choice of interview method in this interview study needs to take into account the trade-off between systematic exploration and flexibility. On the one end of the spectrum is a text survey, which is highly suited for a systematic study, but generally does not facilitate low-barrier individual follow-up questions. On the other end is an in-person interview without pre-defined questions that allows for full flexibility, but can result in incomplete results. We choose here a *general interview guide approach* [125], an approach in the middle of the spectrum, with interviews guided by a script.

Addressing RQ2, we propose a simulation-based capacity planning system that meets the requirements we gather from the interviews we conduct in RQ1.2: Capelin. First, answering RQ2.1, we design a conceptual framework for planners to specify their capacity planning problems: a portfolio of “what-if” scenarios. This represents the many possible future scenarios a capacity planner needs to take into account. Second, we design a decision support system that allows for exploration and manipulation of such scenarios, addressing RQ2.2. We depict this system (Capelin) in its context in Figure 1.1 (see step ②). The designed system builds upon the OpenDC platform for datacenter simulation [59], extending it with functionality tailored towards capacity planning practitioners. By extending OpenDC instead of building a stand-alone tool, we ensure that existing users of Capelin can benefit from future improvements to the core simulator without needing additional integration. Our design approach is guided by the AtLarge design process [61], with special emphasis on linking the requirements to concrete findings gathered in the series of community interviews, and on co-evolving the solution and the problem (i.e., by adding requirements as we understand the problem better, and iterating on the solution to include these new requirements).

To address RQ3, we evaluate Capelin by implementing a working prototype and conducting experiments with it. First, we implement the design system as an extension to OpenDC (RQ3.1), extending the simulator with many functionalities that benefit both the prototype and the broader user base of the simulation platform. Second, we consider a selection of common capacity planning questions and answer them with the designed tool. We conduct trace-based experiments, considering different topology dimensions, a range of workloads, and complex operational phenomena.

1.5. Development and Dissemination

This thesis has resulted in the following disseminated materials and developed software:

1. Articles under submission
 - (a) Article on Capelin to be submitted to a tier-1 journal in the field, as first author: G. Andreadis, F. Mastenbroek, V. van Beek, and A. Iosup, Capelin: Data-Driven Capacity Procurement for Cloud Datacenters using Portfolios of Scenarios, TPDS, to be submitted end-of-August, 2020.
 - (b) Survey on Capacity Planning to be submitted to a leading peer-reviewed journal, as first author: A Systematic Literature Survey of Capacity Planning Approaches for Cloud Infrastructure, ACM CSUR, to be submitted in September, 2020.
2. Published open science artifacts
 - (a) Publicly disseminated artifacts, following the FAIR principles for scientific data (Findable, Accessible, Interoperable, and Reusable) [135], published on the Zenodo Open Science platform [8].
 - (b) Free and Open Source Software (FOSS) software artifacts published on GitHub, for inspection and reuse: <https://github.com/atlarge-research/opendc>
3. Coordination of the development of a FOSS datacenter simulation platform (OpenDC¹), leading a team of 15+ members. This has resulted in a tested, trusted, and flexible simulator for the datacenter community. Some of the features used in this thesis include automated and repeatable execution of multiple instances for results with high statistical confidence, broad workload trace reading and modeling support, and high-performance simulation. Beyond, some of the OpenDC features being developed at the moment that complement Capelin well, are prefabricated components, realistic energy models, and simulation of serverless cloud systems.

1.6. Guidelines for Reading

The remainder of this thesis is structured as depicted in Figure 1.2. In Chapter 2, we lay out the current state-of-the-art of capacity planning theory. In Chapters 3 and 4, we systematically survey the current state of literature on the subject and the current state of practice as experienced by experts, respectively. We present the design of Capelin in Chapter 5 and evaluate the prototype of this system in Chapter 6. Finally, in Chapter 7 we summarize the contributions of this thesis and envision future work stemming from this project.

¹<https://opendc.org>

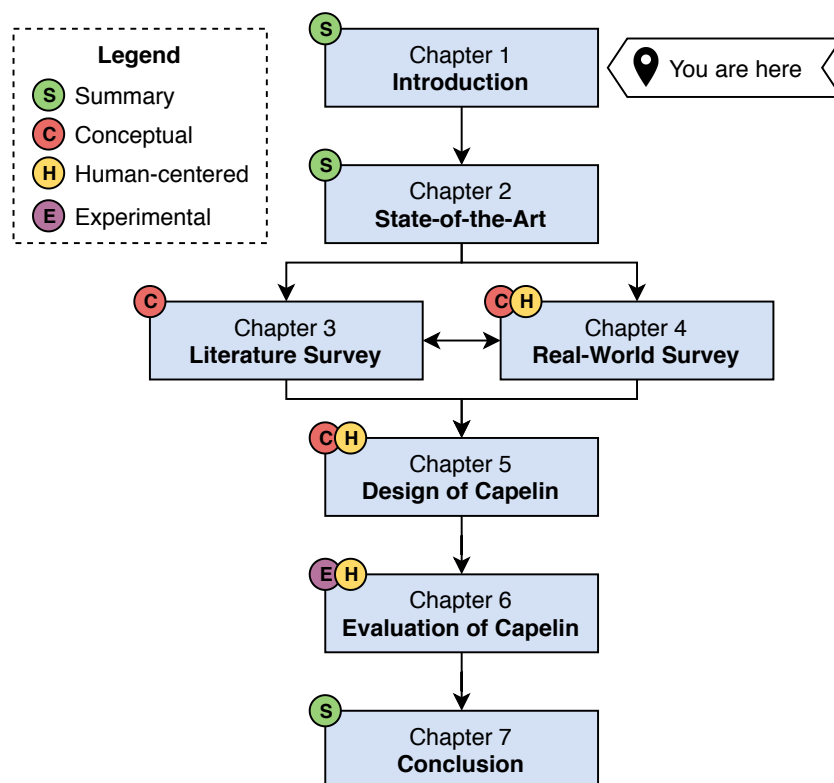


Figure 1.2: Structure of this thesis, with suggested reading flows.

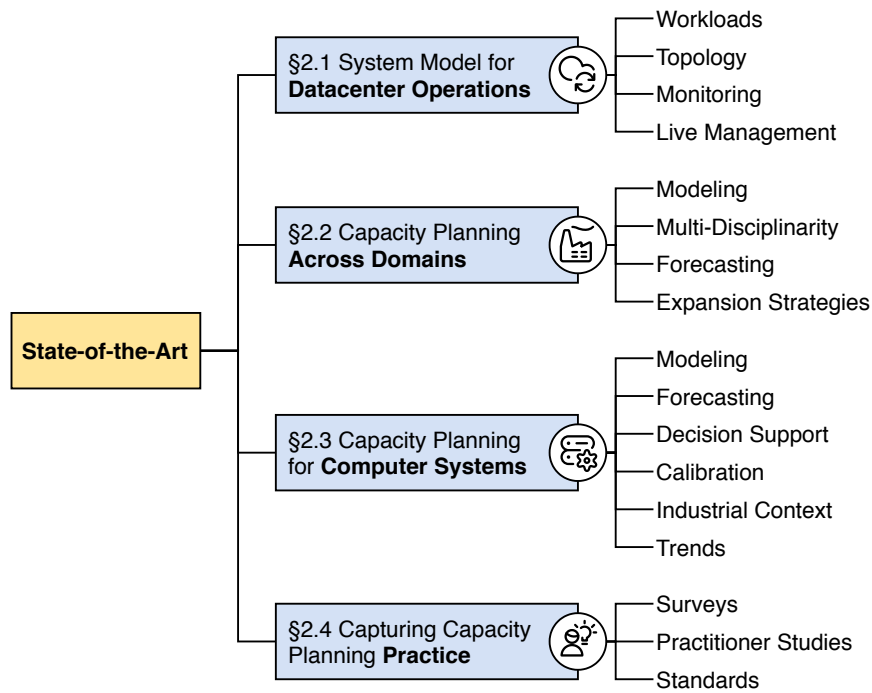


Figure 2.1: Overview of the structure of this chapter, presenting the state-of-the-art in capacity planning for cloud infrastructure. Keywords on the right-hand side represent important notions introduced in each section.

2

State-of-the-Art

In this chapter, we discuss the state-of-the-art in capacity planning theory for cloud infrastructure. This forms the foundation for an understanding of current literature in the field, answering the theoretical side of RQ1.1.

As Figure 2.1 depicts, we begin in Section 2.1, with a top-down perspective of datacenter operations, placing capacity planning in its context. Then, in Section 2.2, we discuss capacity planning as it is practiced across domains. In Section 2.3, we summarize the current theory of capacity planning for computer systems. Finally, we cover existing efforts to understand and standardize the state-of-the-art of capacity planning practice in Section 2.4.

2.1. System Model for Datacenter Operations

How does a datacenter work? There are many different types of datacenters, each with different operational models. To understand the work of a capacity planner, we first need to understand the context and capacity that the capacity planner works in and should manage. This context is encapsulated in a *model of datacenter structure and operation*. Such a model needs to cover many aspects, such as the *workload*, the *infrastructure*,

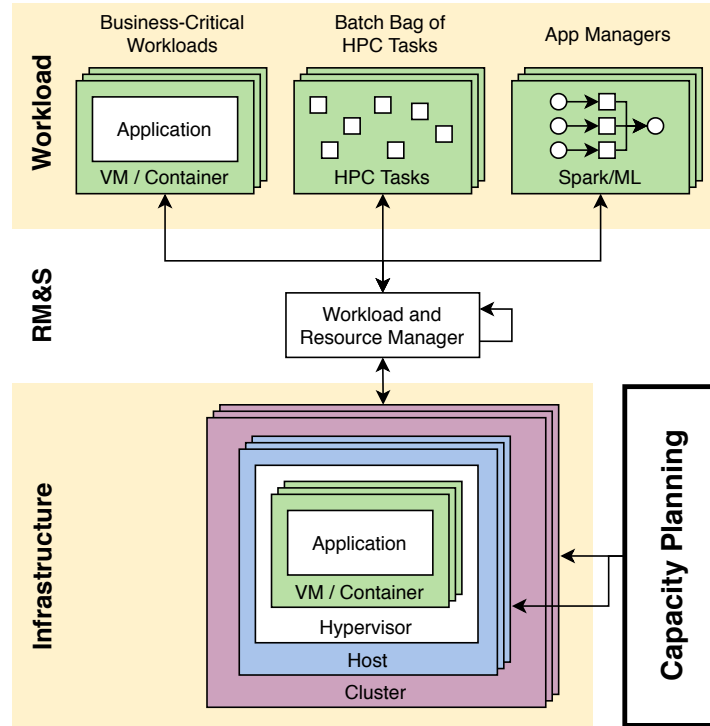


Figure 2.2: Generic model for datacenter operation. In this work, we focus on the capacity planning process.

and the *live resource management*. In this section, we provide such a model as context of our contributions.

We are not the first to provide such a system model. In other lines of research, system models have been proposed describing the context of their contributions, such as in the area of resource scheduling research in clouds [9, 130]. Each of these models portrays the parts of the environment that are relevant to that work and at a suitable level of abstraction. We are the first to provide such a system model for the capacity planning practice in datacenters. Below, we summarize the current state-of-the-art existing across these aspects and present our model for datacenter operations, depicted in Figure 2.2.

2.1.1. Workload

The workload consists of applications executing in *VMs* and *containers*. The emphasis of this study is on *business-critical workloads*, which are long-running, typically user-facing, and back-end enterprise services at the core of an enterprise’s business [120, 121]. Their downtime, or even just low QoS, can incur significant and long-lasting damage to the business. We also consider virtual *public cloud workloads* in this model, submitted by a wider user base with typically shorter duration.

The business-critical workloads we consider also include virtualized High Performance Computing (HPC) parts. These parts are primarily comprised of conveniently (embarrassingly) parallel tasks, e.g., Monte Carlo simulations, forming *batch bags-of-tasks*. Large HPC workloads, such as scientific workloads from the health-care sciences, also fit in our model. Our system model also considers app managers, such as the big data frameworks Spark and Apache Flink, and the machine learning framework TensorFlow, all of which orchestrate virtualized workflows and dataflows for their users.

2.1.2. Infrastructure

The workloads described earlier run on physical datacenter infrastructure. Our model views datacenter infrastructure as a set of physical clusters of possibly *heterogeneous hosts* (machines), each host being a node in a datacenter rack. A host can execute multiple VM or container workloads, managed by a *hypervisor*. The hypervisor allocates computational time on the CPU between the workloads that request it, through *time-sharing* (if on the same cores) or *space-sharing* (if on different cores).

We model the CPU usage of applications per discretized time slices. At each time slice, all workloads report requested CPU time to the hypervisor and receive the granted CPU time that the resources allow. We

assume a generic memory model, with memory allocation constant over the runtime of a VM. As is common in industry, we allow overcommissioning of CPU resources [12], but not of memory resources [120].

We also consider in this work operational phenomena, emerging in the complex hardware and software ecosystems at play. We focus on two well-known phenomena: performance variability caused by performance interference between collocated VMs [74, 78, 127] and correlated cluster failures [17, 38, 41].

2.1.3. Live Platform Management (RM&S)

We model a workload and resource manager that performs management and control of all clusters and hosts, and is responsible for the lifecycle of submitted VMs, including their placement onto the available resources [9]. The resource manager is configurable and supports various *allocation policies*, defining the distribution of workloads over resources. The devops team monitors the system and responds to incidents that the resource management system cannot self-manage [16].

2.1.4. Capacity Planning

Closely related with infrastructure and live platform management is the activity of *capacity planning*. This activity is conducted periodically and/or at certain events by a capacity planner (or committee). The activity typically consists of first *modeling* the current state of the system (including its workload and infrastructure) [91], *forecasting* future demand [27], deriving a capacity *decision* [138], and finally *calibrating and validating* the decision [72]. The latter is done for QoS, possibly expressed as detailed SLAs and SLOs. In Chapter 3 we discuss existing approaches in literature and in Chapter 4 we analyze the current state of real-world practice. We are not aware of analytical tools that can cope with such complexity. Although tools for VM simulation exist [21, 53, 100], few support CPU overcommissioning and none outputs detailed VM-level metrics; the same happens for infrastructure phenomena.

2.2. Capacity Planning Across Domains

We now focus on the activity of capacity planning. This activity is far from unique to the cloud domain: capacity planning is a key component of operations in many different domains, from production industries [56] to transport logistics [81]. In this section, we give an overview of the theories presented in literature and attempt to identify where they converge and where they diverge.

2.2.1. Modeling Capacity

We begin with a production and operations management perspective. Martinich gives a theoretical foundation for the operational processes at play in industrial settings, ranging from hospital sites to factories, with elaborate treatment of the capacity planning cycle [89]. The entity being planned, *capacity*, can be defined as “the rate at which output can be produced by an operating unit” [89, p. 252]. This notion can be divided into *design capacity*, the capacity of a unit under ideal conditions, and *effective capacity*, the capacity of a unit under normal conditions [89, p. 252]. Two key metrics can be derived from these quantities: *capacity utilization*, the ratio between actual output and design capacity, and *capacity efficiency*, the ratio between actual output and effective capacity [89, p. 253]. Both metrics can separately indicate the quality of a capacity planning process, although the combination can also be insightful: a system with high capacity efficiency but low capacity utilization indicates a large discrepancy between the ideal case and the normal case.

2.2.2. Capacity Planning: A Multi-Factor, Multi-Disciplinary Problem

With these definitions in mind, we move to the process of capacity planning itself. Martinich lists the following factors to be relevant in determining capacity: process design, product design, product variety, product quality, production scheduling, materials management, maintenance, and personnel management [89, pp. 253–255]. The high *diversity of factors* in this enumeration shows the highly interdisciplinary nature of the capacity planning process. The author also discusses the problem of (geographical) facility locality in relation to capacity planning [89, p. 266]. While we consider the topic of facility placement to be out of scope for this discussion, an interesting overlap between that topic and the field of capacity planning is formed by the challenge of how and whether to specialize these facility locations. For this purpose, Martinich identifies three dimensions on which the facilities can be specialized: according to products, processes, and/or markets [89, pp. 255–256]. This is closely related to the quantity and type of capacity planned at each facility.

2.2.3. Forecasting Future Demand

Concerning capacity planning itself, a key sub-process is “the art and science” of forecasting future demand [89, p. 102]. Martinich differentiates between *qualitative* and *quantitative* forecasting approaches [89, p. 106]. Qualitative approaches are subjective, generally based on expert knowledge and tacit rules-of-thumb, while quantitative approaches are data-based and more objective, generally based on the analysis of historical measurements. Although it may seem counter-intuitive to resort to qualitative approaches if quantitative historical data is available, there are cases when a qualitative forecast might be preferable, according to Martinich. If the environment is likely to be unstable, or the forecast has a time horizon longer than 3 to 5 years, qualitative forecasting may need to be considered [89, p. 107].

2.2.4. Capacity Expansion Strategies

Once a forecast has been constructed, planners can choose from a variety of *capacity expansion* heuristics: *demand leading* (over-provision ahead of demand), *demand trailing* (capacity lacks behind demand), *demand matching* (capacity matches demand as closely as possible), and *steady expansion* (expanding at regular time intervals) [89, pp. 257–260]. These expansion strategies can be applied in both qualitative and quantitative approaches, although the achieved precision of provisioning will most likely differ. If capacity plans are based on quantitative approaches, these policies can also be supported by more rigorous mathematical optimization models, such as (non-)linear programs. These can also enable automated “what if” analysis for decision support [89, p. 198].

Towards more complex, *multi-facility* expansion problems, Dynamic Programming has been used to minimize capacity costs of different public network, water resource, and process industry facilities [88]. This approach can be combined with a network flow model, exploiting extreme point solutions in the flow [82]. For a more complete overview of expansion modeling strategies, we refer to Lumbreras and Ramos [87]. The scope of either of these expansion models can be restricted to an *finite planning horizon* or extend to *infinite planning horizons*, depending on the longevity of the desired plans. In finite horizon planning, rolling schedules of fixed horizon lengths are one form of ensuring continuity between subsequent iterations [73]. This overlap between capacity decision time spans can also lead to *instability*, however, i.e. future decisions of capacity at a given time being reversed or amended in overlapping future schedules [73].

2.3. Capacity Planning for Computer Systems

We now explore the theories of capacity planning for computer systems. Although the computer as a capacity to be managed has a shorter history than many of the industrial capacities discussed previously, already a variety of approaches exist towards capacity planning for computer systems. In this section, we explore the sub-processes that are a part of the capacity planning process. We then discuss what sets capacity planning for computer systems apart from traditional capacity planning. Finally, we cover standardization efforts for capacity planning processes in this domain.

2.3.1. A Schematic Overview

To introduce the theory, we present the process models that we have found in the established literature. We summarize these models through a schematic overview, which we depict in Figure 2.3. Each author is represented by a column and each possible step in the process of capacity planning. Similarity between the models proposed by each author is indicated by rows where many cells are filled. This figure represents a synthesis of the field not available elsewhere, a synthesis of these process models made by the author of this thesis.

We find processes in cloud capacity planning to belong in four fundamental stages: *modeling*, *forecasting*, *decision support*, and *calibration and validation*. These are high abstractions, but serve a common denominator for each of the processes from literature, as seen in Figure 2.3. Although the order in which they are listed here is not arbitrary, the stages can be found interleaved in other capacity planning processes. In the following, we explore each of these stages.

Modeling

The start of the capacity planning process is one of the few aspects that most theories agree on. A capacity planner begins by understanding the environment and assessing current capacity [79, p. 6]. A *measurement base* is often built, with information on current system configurations and resource usage [20, p. 5]. This prior understanding of the managed system is augmented with more dynamic information: the workload and the

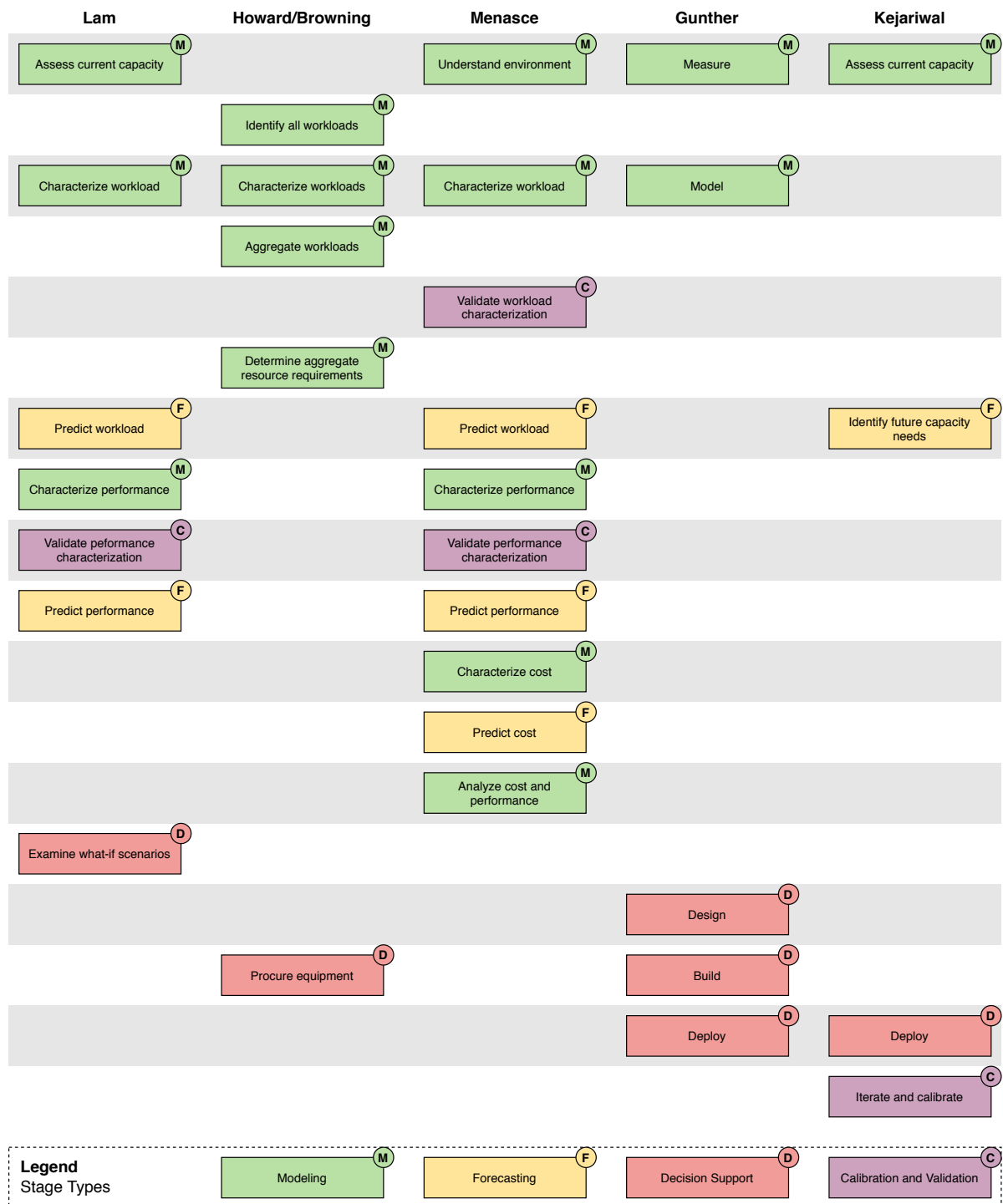


Figure 2.3: Schematic comparison of capacity planning process models presented in the theoretical literature. Sources: Lam and Chan [79, p. 92], [55] [55] (presented in [20, p. 7]), Gunther [49, p. 22], Menascé and Almeida [91, p. 179] (we flatten the model using a topological ordering), and Kejarawal and Allspaw [72, p. 4].

system's performance when faced with it.¹ A *workload model* describes the behavior of user applications in the system, or *what* the system will run, whereas a *performance model* takes the workload as input and combines it with knowledge from the environment to characterize the behavior of the system as a whole, or *how* the system will run the workload. Common dimensions in performance modeling can be divided into system, resource, and workload parameters [91, pp. 193] (the latter being taken from the workload model).

A modeling dimension that is often left out is the human cost involved in implementing and maintaining a capacity decision. In fact, new capacity plans can have significant *people costs* [20, p. 59]. These range from the effort needed to implement new configurations to getting a team trained for new capacity. Models for this dimension are not currently available, making it difficult to quantify and address its challenges.

Forecasting

"Planning without forecasting is futile" [20, p. 66]. There are two main classes of models for forecasting: *time-series models*, where measurements are related to points in time, and *causal models*, where measurements are related to other measurements or specifications [20, p. 79]. Forecasting can be performed both on workload and performance models, depending on the insights sought. What the modeled units may be, the relations are often not linear, but have trends and cycles on different timescales [20, p. 75]. Workload time series may even exhibit *self-similarity*, repeating itself across different scales [91, p. 252].

Forecasting is challenging: Lam and Chan call it the "weakest element in the computer capacity planning process" [79, p. 34]. One difficulty stems from the notion of *latent workload*, i.e., workload that is being held back due to lack of capacity and that might be added when capacity changes [79, p. 35]. There is a parallel to this from the history of economics: the Jevons' paradox of solving the capacity needs for coal at the start of Industrial Revolution in Victorian England, only to realize this triggered an Age of Steam where many "latent" needs for coal emerged. Both in this historical context and in the case of latent workload, the latency cannot be easily measured or predicted, adding another layer of unpredictability to plans.

For many capacity planners, forecasting requires access to existing workloads. For the workloads currently being run, this is straight-forward: they can simply check the dashboards also used by the datacenter's administrators. However, historical knowledge and understanding of emerging workloads raise the difficult challenge of access to relevant data. Such information, contained in an *operational* or *workload trace*, is rarely available, or difficult to parse due to size, etc. State-of-the-art data archives include the Parallel Workloads Archive², the Grid Workloads Archive [58], the Workflows Trace Archive [131], and the single-operator traces shared generously in the past years by organizations such as Google [111], Microsoft [33], and SURFsara [126].

Decision Support

Once a workload forecast has been made, the forecast needs to be transformed into actual *capacity planning decisions*. This is not as straight-forward as it may seem: trade-offs between horizontal and vertical scaling, heterogeneous environments, and many other domain-specific aspects make this a complex problem. This remains a subjective task where humans often still play the main role.

A capacity planning system can help with making decisions by providing suggestions and evaluating alternative decisions. Unfortunately, theories on how this should be done are difficult to find and only provide rough guidance. Lam and Chan give the most extensive explanation in this regard [79, p. 108]. They see the role of a capacity planning system and its performance model to serve the capacity planner in answering what-if questions ("examining alternatives"). The capacity planning system is given little autonomy in this, but this may be explained by the early date of publication of this publication (1987). The limited computational capabilities at the time prohibited more complex (e.g., machine learning) models.

Browning, Gunther, and Kejariwal and Allspaw all mention procurement (sometimes referred to as deployment) steps in their theory of capacity planning, but give little to no detail on how to decide on the actual capacity to be acquired in this step [20, 49, 72]. A relevant rule of thumb is provided by Kejariwal and Allspaw, however: generally, it seems to be more advantageous to acquire hardware later than sooner, because the cost of a certain quality of hardware tend to decrease over time [72, pp.104–105].

Another important notion is the *factor of safety* that many planners explicitly or implicitly employ [72, p. 121]. This is a theoretical margin of a resource's capacity allocated as a safety buffer, to ensure overflow capacity in case of workload spikes or larger failures. Although an important theoretical concept, its link to SLAs and SLOs is not addressed systematically in the theoretical literature.

¹This separation between workload and performance is not universal to all theories: only Menascé and Almeida, and Lam and Chan, specify that these concepts should be treated separately.

²<https://www.cs.huji.ac.il/labs/parallel/workload/>

Calibration and Validation

An important step in any process involving modeling or decision making is the retrospective view: a critical look at the validity and optimality of the findings and decisions made by the capacity planner. More importantly perhaps, there is potential for non-trivial feedback loops to emerge that might not be identified without a critical post-decision analysis: increased capacity can change user behaviors, changing the picture for future capacity plans [72, pp. 126–127].

Overall, the subject of calibration and validation is only briefly discussed in literature. Menascé and Almeida incorporate it in the validation of their models workload and performance models. Their validation approach is to run a “synthetic workload” produced by the model and compare its performance with those from the actual workload [91, pp. 190–191]. Lam and Chan advocate a similar approach [79, p. 108]. For both works, the actual mechanisms and metrics used to execute this comparison are left unspecified. In practice, acquisition processes that include public procurement (such as those conducted by public scientific infrastructure) require applicants to provide benchmarking data and thus meticulously specify these mechanisms and metrics. We see that theoretical basis for these very common processes is lacking. Kejariwal and Allspaw add a step for iteration and calibration to their model [72, p. 4], but do not elaborate on how to execute this step. However, the recognition that calibration based on earlier decisions before making new decisions should occur still sets this work by Kejariwal and Allspaw apart.

2.3.2. Parallels with Industrial Capacity Planning

We have previously made explicit discussions on “traditional” industrial capacity planning and the more recent capacity planning for cloud infrastructure, but there have been implicit comparisons made on what the two share and what sets them apart. It is worth exploring this explicitly, to see where inspiration can be gained for either of the two fields and to explore whether there are fundamental differences between the two.

The strong focus on forecasting workload (or demand) is shared by both fields. Without it, the capacity cannot be accurately predicted. A more subtle common aspect is the trade-off between *economies of scale* and locality factors. In industrial settings, it is often more profitable from an operational point of view to concentrate production in one large facility [89, p. 50]. However, the distribution costs that arise from centralizing all supply prohibits this. Similarly, datacenters and cloud platforms often need to be distributed, to minimize latency of service for customers and, more recently, to adhere to emerging privacy regulations (e.g., General Data Protection Regulation (GDPR)).

We can contrast the two fields in the granularity of decisions. In one sense, the decisions for industrial settings seem to be of a larger kind: Martinich refers to a “Lumpiness of Capacity” that makes it relatively unprofitable to add capacity in small increments [89, pp. 50–51]. This is generally not shared by cloud infrastructure planning, which seems to operate on per-rack changes without prohibitive cost. This is further true when considering that *virtual* infrastructure can also be added on-demand, e.g., a cloud operator may “spill over” (migrate) workload to another rack temporarily, when its demand increases beyond capacity and threatens the SLA. In another sense, the industrial settings seem to have a longer time scale for decision making and planning in advance, sometimes of decades. This is related to the earlier point about the magnitude of decisions, but should be mentioned on its own, since it affects the precision needed for a forecast and therefore the margin for error.

2.3.3. Trends in Computer Capacity Planning

When we compare cloud capacity planning in the 1980s with the capacity planning of today, the field seems to still have the same theoretical foundations. The structure of the processes we see today closely resembles those of the past, even though platform specific aspects have changed, along with the pace and costs of procurement. Gunther even finds capacity planning to be less widely accepted today than in the past [49, p. 2], conjecturing that a drop in hardware price has led to a reduced sense of urgency to make precise capacity plans. Another profound change over time is the increase in managed complexity. Although this is hard to objectively measure, the quantity, heterogeneity, and expectations of the managed infrastructure seem to all have *increased* compared to earlier computer capacity planning approaches.

The lack of published research on more sophisticated methods such as simulation-based scenario exploration is cause for concern. Conventional methods may still apply in simple, restricted scenarios, but not in the comprehensive, multi-disciplinary capacity planning we frequently see in practice (see also Chapter 4 for our study of real world experiences on capacity planning, or the scenarios we exemplify through experiments in Chapter 6). Van Hoorn also argues for the integration of simulators in standard practice of offline capacity planning, and even proposes a simulation-based online capacity planning approach [128, p. 79].

The technology itself is also of concern: as technology moves forward, the nature of capacity planning might need to adapt, as it has done from mainframes to grid servers to early web platforms. Kejariwal and Allspaw points out that microservices and serverless services have “a direct impact on the capacity planning process” [72, p. x].

2.4. Capturing Capacity Planning Practice

Real-world practice can diverge from published theory. In this section, we discuss past efforts to understand the practice of capacity planning, both as published in literature and as experienced by practitioners. We also summarize key standards that have been established to describe and systematize this practice.

2.4.1. Surveys of Published Approaches

There are only a few meta-studies reviewing the approaches that have been published in this field. Unlike the work summarized in Section 2.3, none of them are both comprehensive and systematic in their mapping of publications to a designed model of the field. In the following, we discuss meta-studies adjacent to this survey, both independently published and within academic reports.

The thesis of Bauer gives an extensive summary of capacity planning processes in other fields, such as manufacturing, transport, and health care [15, pp. 13–18]. Information Technology (IT) capacity planning is treated in more depth: the author discusses standardized frameworks on the topic and a selection of scientific approaches [15, pp. 19–25]. A key insight given here is that the existing research “lack[s] insight into how organizations are performing the proposed frameworks, best practices, or recommendations” [15, p. 24]. This implies a potential gap between what research advises and what organizations do, in practice. We will act on this gap with a novel survey of real-world experts, in Chapter 4.

Prabath outlines main ingredients and metrics that are considered in the IT capacity planning process [107]. In their thesis, they divide the capacity planning field into two groups: *rule-based* capacity planning, often assisted by Machine Learning models or fuzzy logic models, and *model-based* capacity planning, often assisted by queuing theoretical models. They do not, however, support or validate the designed taxonomy with examples from academic publications.

In a study centered around cloud networking capacity management, Jiang and Sun explore the benefits and challenges of a comprehensive network capacity planning process [69]. The authors discuss both technical factors, such as the potential cost savings and operational risk management, and human factors, such as the better understanding and service of customer demands that a frequent and thorough capacity planning process can facilitate. Among the main challenges for capacity management for networks in particular, they list bottleneck analysis and the estimation of the network’s state at a point in time. The model given as solution framework, however, remains on a high level of abstraction and does not aim to accurately model a capacity planning.

Odun-Ayo et al. perform a systematic mapping study of cloud management in a broad sense, with capacity planning being one of the subjects analyzed [101]. They classify the surveyed works into types of research contributions, which for capacity planning reportedly fall mainly into models and processes. They find only few publications discussing metrics or proposing tools for capacity planning.

The work of Loboz is an exception to this list, in that their study does not attempt to comprehensively survey the field [84]. Rather, it gives a more critical perspective on the conventional approach to modeling and forecasting resource usage. In a meta-study, Loboz challenges the common assumption that resource utilization distributions are normally or exponentially distributed. The author observes that the distributions are in fact often heavy-tailed and rather follow power-law distributions. The high volatility common in traces of resource usages would therefore lead to erroneously predicted capacity buffer sizes, leading to a waste of resources and/or unexpectedly long queuing delays and other SLA violations. Loboz draws a parallel to stock market indices, which are reportedly several times less volatile than the distributions characterizing this domain [84]. In stock markets, the simple tooling claimed to be prevalent in resource usage modeling literature is not considered acceptable for the challenge. The work done by Loboz emphasizes the critical attitude that is needed towards commonly held beliefs around capacity planning.

Overall, we see the lack of a systematic overview of the published approaches surrounding capacity planning. Such an overview is crucial to understanding the state of the field and community, to ensure widespread knowledge of existing findings and learned lessons, especially for researchers aiming to advance the state of the art. We address this shortcoming in Chapter 3, where we present a systematic survey of the field.

2.4.2. Practitioner Studies

To our knowledge, only two works have surveyed the real-world practice of capacity planning in IT infrastructures. In the late-1980s, Lam and Chan conducted a written questionnaire survey [79] and, mid-2010s, Bauer and Bellamy conducted semi-structured interviews [14]. The target group of these studies differs from the practitioners, however, since both focus on practitioners from different industries planning the resources used by their IT department. We summarize both related works below.

Lam and Chan (1987) conduct a written survey with 388 participants [79, p. 142]. The survey consists of scaled questions where practitioners indicate how frequently they use certain strategies in different stages of the capacity planning process [79, p. 143]. Their results indicate that very few respondents believe that they use “sophisticated” forecasting techniques for their capacity planning activities, with visual trending being the most popular strategy at that time. They find that “many companies still rely on the simplistic, rules-of-thumb, or judgmental approach” to capacity planning [79, p. 8]. More importantly even, the authors believe that there is a “significant gap between theory and practice as to the usability of the scientific and the more sophisticated techniques”. These findings stress the need for a usable and comprehensive capacity planning system for today’s computer systems.

Bauer and Bellamy (2017) conduct 12 in-person interviews with “IT capacity-management practitioners” [14] in six different industries. Similar to our interviewing style, the interviews were “semi-structured”, guided by questions prepared in advance. The questions range from capacity planning process questions to more managerial questions around organizational structure. After manual evaluation of the interview transcripts, the authors find that practitioners often state that the number of capacity planning roles in organizations is decreasing, while the discipline is still very much relevant. The practitioners also find that “vendor-relationship management and contract management” are playing an increasing role in the capacity planning process, as well as redundancy and multi-cloud considerations. These results, even if for a different target group, underline our call for the need to focus on the capacity planning process as an essential part of resource management, and emphasize the multi-disciplinary, complex nature of the decisions needing to be taken.

For a problem of such importance, it is surprising that no up-to-date, expert-oriented community interview study exists on the field of cloud capacity planning. Understanding the real-world practice could help map the experiences of practitioners and guide the design of tools to assist them in their activity. We address the lack of such a study in Chapter 4, where we interview capacity planning experts from different cloud infrastructures.

2.4.3. Standardization Efforts

As is the case in most mature fields, standards have been composed for the capacity planning process in cloud infrastructure. International standards typically address the broader issue of IT service management, but often also include formalisms of processes for capacity planning. While often refraining from giving concrete advice, these can indicate what activities practitioners in industry consider and what terminology is used. Below, we list the major standards which specify capacity planning activities to some extent.

Capability Maturity Model Integration (CMMI) This framework features the most elaborate description of capacity planning processes [1]. Sub-activities of this framework include: (1) documentation of the current state, (2) forecasting of future needs, (3) development of a “strategy” for capacity, (4) documentation of involved costs, and (5) periodic or event-driven revisions. Notable is the emphasis on underutilization as one of the key indicators for the quality of a capacity strategy, and the integration of revision and reflection as a key component of the process.

Information Technology Infrastructure Library (ITIL) This framework divides capacity planning into four sub-processes: business capacity management, service capacity management, component capacity management, and capacity management reporting [64]. The differentiation between these activities mirrors the stack of abstractions that systems operate in. The framework also defines a distinction between a capacity plan and a capacity report, the former being forward-looking and the latter being an assessment of the current situation.

ISO 20000 The ISO standard on IT service management includes a section on capacity planning within the “service delivery process” [63]. It only sets requirements for the process, specifying that practitioners should use capacity monitoring strategies and should analyze the monitoring data with respect to human, technical, and financial aspects for potential improvements.

Microsoft Operations Framework (MOF) This framework places the capacity planning process in an “optimizing quadrant”, a circular combination of processes aiming to improve quality of service while reducing costs [93]. The documentation of this framework mentions many sub-activities within capacity planning, from sizing service capacity to setting SLA-compliant performance level targets. The process mentions the definition of “usage scenarios” as well as the consideration of “peak load characteristics” as key activities to meet these targets.

Control Objectives for Information and Related Technologies (COBIT) This framework identifies five key practices [62]. Most notable is its recommendation to define a “baseline” capacity. It also mentions monitoring and planning activities, focusing on deviations from the defined baseline.

The Open Group Architecture Framework (TOGAF) This framework gives a more managerial, high-level view on the topic [2]. It breaks the process into a definition of current capabilities, a design of “capability increments”, and resulting “building blocks”. Capability increments are defined on different dimensions, with each increment ideally contributing to improvements on each dimension.

We also observe large companies taking steps to standardize and document their own processes. Microsoft’s MOF (described above) is one example of this. Publication of the details of actually used processes is limited however. Published work by Google on their network capacity planning processes forms an exception to this [4, 95]. In these publications, they only focus on a subset of the problem and propose specific strategies for that subset, yet are also more concrete in the description of their approach than the standards listed above.

3

Systematic Literature Survey of Capacity Planning

This section presents a systematic survey of the cloud infrastructure capacity planning field. The methods and results presented in this section address research question RQ1.1.

3.1. Overview

The state-of-the-art in published approaches within a certain field typically spans over many tens (if not hundreds) of publications. Without an overview of these publications, gathered knowledge and learned lessons are not available to researchers new to the field without labor-intensive search and conceptualization. A systematic survey can make this tacit knowledge in the field explicit and serve as a map of the community to researchers and practitioners. In Section 2.4.1, we establish that the field of capacity planning for cloud infrastructure in particular is still in need of such a systematic overview.

The aim of this survey is to understand the current state-of-the-art of capacity planning literature for cloud infrastructure. We describe our method for constructing this survey and analyzing the results in Section 3.2. In Section 3.3, we present a taxonomy for the field. We map a diverse set of publications in the field to this taxonomy in Section 3.4. Using this set of mappings, we conduct a meta-analysis of the field and community in Section 3.5. In Section 3.6, we then compare our findings to the theory that we have presented in Chapter 2. Finally, we discuss our findings and their validity in Section 3.7.

We FAIRly disseminate the data artifacts [135] of this survey as well as the software artifacts used to analyze them, on the Zenodo Open Science platform [8]. These are publicly accessible and can serve as basis for external analysis.

3.2. Method

We begin by setting the precise scope for this survey in Section 3.2.1. We then describe our methods for collecting relevant publications in Section 3.2.2. We lay out our design process for a taxonomy of the field in Section 3.2.3. In Section 3.2.4, we discuss how to map collected publications to a designed taxonomy. Finally, in Section 3.5 we describe our meta-analysis to gather useful insights from the publications mapped to the taxonomy space.

3.2.1. Definition of a Scope

Before understanding and mapping the design space for capacity planning systems and approaches for cloud infrastructure, we first need to define what we consider as part of this process and what we exclude from it. In this work, we focus on static capacity planning, the process of making mid- to long-term decisions about the structure and scale of infrastructure in clouds. We do not limit the kind of decisions to the physical level, low in the hierarchy of abstractions in cloud infrastructure—we also address capacity planning at the higher levels, including at IaaS and PaaS levels. Deciding how many VMs to provision requires similar thought and analysis processes as deciding how many physical server units are needed. We exclude capacity planning processes that rely solely on dynamic, short-term decision-making (e.g., temporarily switching off unused machines),

such Google's Auxon system [16, §18]. While such dynamic processes need to be taken into account in conventional capacity planning activities, they are not the focus of this study, since the static decisions are still most likely to have the most significant environment and financial impact. They set the boundaries for dynamic strategies to operate in.

3.2.2. Acquisition and Selection of Publications

The survey process begins with an acquisition of relevant publications. We query three of the most popular and extensive online repositories for scientific work, with a search query composed according to the query language of each repository. The repositories are: Google Scholar, DBLP, and Scopus. After querying each repository, we collect the results in order of appearance, up to a per-repository threshold of 200 items. We set this limit of 200 following empirical evidence that the precision of results decreases significantly in listings after that order of item counts. Next to our searches in the mentioned repositories, we also include a small set of publications obtained otherwise, such as during exploratory searches or on external recommendation.

We now conduct a preliminary filter that excludes duplicate results, such as work published in multiple venues, and results that are not accessible to the authors, even with academic access programmes. After this preliminary filter, we analyze title and abstract of each result and determine whether it fits our definition. We do not accept theoretical books and surveys, since they typically do not propose a single approach to capacity planning and therefore are not fairly comparable to the other works. They are, however, taken into consideration in the broader study of related work to this thesis (see Chapter 2).

Google Scholar

The search query for this repository is:

```
capacity planning|management cloud|datacenter|"data center"|cluster
```

We exclude patents from the results. The date of access for this repository was November 21, 2019.

DBLP

The search query for this repository is:

```
capacity planning|management cloud|datacenter|cluster.
```

We use the Dagstuhl mirror. The date of access for this repository was November 22, 2019.

Scopus

The search query for this repository is:

```
capacity AND planning OR management AND cloud OR datacenter OR data center OR cluster.
```

We search only in title, to preserve relevance of results. We sort the results by relevance, as defined by the repository. The date of access for this repository was November 22, 2019.

3.2.3. Design of a Taxonomy

We now describe our process of creating a conceptual model for this field. We choose for a taxonomy as the model type, since the many sub-activities within capacity planning lend themselves well to a hierarchical overview. Our design follows an iterative process that is closely related to the mapping process described in Section 3.2.4. First, we gather potential dimensions of interest from the selected publications, theoretical works, and interviews conducted in the scope of this thesis. We then evaluate each potential dimension on its relevance towards characterizing capacity planning systems as described in literature. The selection that emerges from this process is a hierarchical tree of categories, with height 2. For each of the "leaf" categories (at the bottom of the tree), we collect the different instantiations (classes) that capacity planning systems can take for those specific categories. In each category, systems can belong to none, one, or multiple classes. When we find a previously unconsidered class in deeper analysis of one of the papers, we extend the corresponding category in the taxonomy with this class.

3.2.4. Mapping Publications to the Taxonomy

Given our selection of publications and our designed taxonomy, we now classify each publication along each category to provide a systematic overview of the field. For each publication, we begin by reading abstract and introduction to understand the context of that work and its main contribution(s). We then traverse the rest of the publication with the taxonomy in mind.

We use a text-based approach (using the YAML serialization format¹) to serialize the classification decisions for each category and each class. For each publication-class membership, we either assert or reject the membership based on what information we find in the publication, for each of the classes in the taxonomy. We only assert membership if the publication mentions that class clearly. The exact implementation or strategy possibly associated with that class does not have to be explained, but the class (or an equivalent formulation) has to be given. An absence of a class in the mapping does not mean that the work that was published does not belong in this class, but only that there is insufficient evidence to make that particular positive classification. When a publication presents a new, unforeseen class which the taxonomy does not already cover, we re-traverse the entire set of already mapped publications and investigate their class memberships in that particular category, anew.

It can occur that, during this mapping phase, we conclude that a paper should not be included in the selection of mapped systems. In that case, we note the rationale behind this decision and exclude the paper from the list of publications. A key criterion here is that a publication describes a system that fits the notion of capacity planning as we define it (see Section 3.2.1). Another criterion is that a publication needs to present a model or systematic process for capacity planning. Incidental experiments with informal analysis do not suffice, since their findings or processes cannot easily be extrapolated to other scenarios. Finally, we do not admit duplicates in the list of works. Even if published in different venues and with slightly different presentation, we exclude similar contributions and include only the most elaborate publication out of a set of duplicate publications.

3.2.5. Meta-Analysis of Mapped Publications

We conduct a series of meta-analyses on the set of selected publications and their mappings to the taxonomy space. Below, we present our methods for these analyses.

Year of Publication

We begin with an analysis of the distribution of publications over their year of publication. For this purpose, we analyze a histogram of the year of publications of all selected publications and investigate trends over time.

Inter-Class Correlations

We now investigate relations between different classes to see which classes co-occur often or, in the opposite case, mutually exclude each other. To numerically evaluate this, we can view the taxonomy as spanning a multi-dimensional, one-hot encoded design space, with each class being represented by a dimension. We then visualize these relations as correlation matrix of each class with each other class.

Clustering Analysis

Looking at the publication set from a landscape perspective, we want to explore communities of publications in an automated, systematic way. The dimensional space spanned by the taxonomy enables us to do so, through analysis of clusters in the space. By finding clusters of publications, we hope to uncover groups that are closely related in some aspect that may not be directly obvious. Using the `scikit-learn` package [106], we apply two different clustering methods on a dimensionality-reduced version of the data, using Principal Component Analysis (PCA). We reduce the full space to two dimensions, to facilitate visual inspection. For clustering, we use the K-means clustering method with 4 clusters and the DBSCAN algorithm [39] with $\epsilon = 0.4$. We exclude the year of publication and other metadata from the clustering input to ensure that the clustering is only sourced from the dimensions of the taxonomy.

Analysis Per Category

We also conduct more fine-grained analysis per category. We distinguish between two types of analysis: Class counts and class trends. Class counts indicate the “popularity” (frequency of a positive match) for each class, over the entire surveyed set. Class trends indicate these counts over time, aggregating them per year and thus giving an overview of their progression.

Trending Keyword Analysis

We also investigate the keywords present in abstracts of relevant publications, to see if the trends found there can be corroborated or contrasted against our findings in the systematic meta-analysis. For this purpose, we

¹<https://yaml.org/>

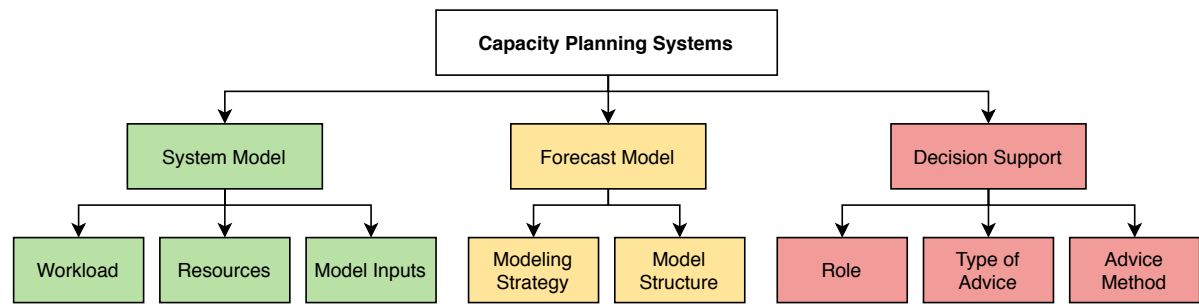


Figure 3.1: Taxonomy of capacity planning systems for cloud infrastructure.

turn to tooling developed by Versluis et al. [131] and perform global trending analysis on the keywords of our survey. We query a database of all literature from DBLP and Semantic Scholar between 2015 and 2019 (both inclusive) on abstract and title with the same query as the literature survey. We consider the 100 highest-ranking keywords by frequency of occurrence in these 5 years, in alphabetical order. These can be seen as trending keywords, due to their persistence in the high ranks in the popularity index. Stop-words are automatically filtered from the list.

3.3. Taxonomy of Capacity Planning Approaches

Following the process described in Section 3.2.3, we design a taxonomy for the field. The result can be seen in Figure 3.1, visualizing this hierarchical decomposition, and in Table 3.1, serving as a legend of classes for each of the categories of the taxonomy.

Our taxonomy divides capacity planning systems into three key components: (1) the system model, (2) the forecast model, and (3) the decision model they provide. Each of these is divided further into multiple categories. We now treat each of the categories of the taxonomy in the following subsections.

3.3.1. System Model

A crucial prerequisite to any capacity plan is an understanding of the context it should manage. This mainly consists of the workloads being served and the resources on which these workloads are served. Along with these two characterizations, it is important to identify what parts of the context are actually considered in the capacity planning process in literature, also described as the inputs of assumed capacity planning system model.

Workloads

This category concerns itself with the types of workloads that a capacity planning system takes into consideration. There is a close relationship between these workloads and the services that are provided by the system to be planned. Because of the layer-agnostic approach of this taxonomy, these workloads can themselves also serve as resources to other workloads or be directly served to end-users.

Resources

The resources offered to stakeholders and considered in capacity planning are covered by this category. We only include resources that are part of the capacity plan's decision in the mappings for this category. To illustrate: If a capacity planning system for VMs is deciding whether to add compute nodes to its cluster, but it also monitors their storage use without advising changes in that dimension, it is classified in the C (compute) class, not in the S (storage) class. Similarly, the classes E and H are reserved for systems making decisions about energy- and heat-control-related hardware, as differentiated from capacity planning systems that only regard power consumption or heat dissipation as a factor in their decision-making.

Model Inputs

A core part of any capacity planning is the abstractions it operates on. We consider the combinations of these abstractions the system model of the capacity planning system. This category represents the inputs that this model considers, such as historical data, human personnel costs, or SLAs. Resource lease contracts can also be considered inputs, although we limit ourselves to long-term lease contracts (on the order of months), excluding dynamic second/minute/hour-based leases of virtual resources.

3.3.2. Forecast Model

Once an understanding of the system under planning has been established, practitioners need a forecast for future demands to be able to form decisions. This forecast is of course optional, in theory, but in practice we see every publication listing a kind of forecast model, some more complex than others. Forecasts across capacity planning systems operate on different levels, with some directly predicting resource demand and others only predicting the workload's evolution over time, with other models in the system translating this to resource demand.

Modeling Strategy

This category characterizes the type of model used for forecasting demand. The main contenders here are analytical models (frequently based on concepts from Queuing Theory and Queuing Networks) and simulation models. Some publications execute experimental tests with real-world resources, as well.

A clarification at this point is needed on what simulation and experiments are considered part of the capacity planning system. Publications frequently execute experiments (in simulation or in the real-world) to validate their model. However, since this data is not available at the time of training the model, we do not consider it part of the capacity planning system. Experiments and simulations thus need to be explicitly informed by the capacity planning process and not (only) serve to evaluate it, afterwards, for them to be considered part of the modeling strategy. An interesting opportunity for further research would be to investigate the various strategies used for self-evaluation of capacity planning system, both with and without humans in the loop.

Model Structure

The uncertainty of future events means that accurate forecasts are inherently difficult to make. It is therefore often useful to design different scenarios that might unfold (in the form of "what-if" questions). This category differentiates between systems that only offer one forecast and systems that are built to support different scenarios and to predict future demand for each scenario.

3.3.3. Decision Support

Once the current system under planning has been understood and a forecast has been made for the future of this system, a capacity planning system can also aid practitioners in their decision making, e.g. through model-based optimization. The categories of this section concern these decision support mechanisms.

Role

In practice, not all capacity planning systems offer this last step of support. This category characterizes this by splitting the field into purely forecasting systems and systems which provide active decision support through actionable advice.

Type of Advice

If a system belongs to the latter category of the previous category, there are different kinds of advice it can give to practitioners. These include the number of resources to provision (often described with horizontal scaling), the type of resources needed (often described as vertical scaling), and the locality of these resources. The number of resources can be on the level of cores, machines, VMs, or even cooling units. Examples for the type of resource are the specifications of a machine or the provisioning-type of a VM (e.g. on a long-term vs. short-term lease). The locality of a resource concerns its placement in the considered infrastructure, such as the placement in one of the datacenter sites of a single operator or the placement in a hybrid cloud (the decision being between a public and private cloud offering).

Advice Method

Finally, the method of deriving this advice is treated in this category. The capacity planning problem is then typically formulated as an optimization problem with (multiple) fitness metric(s). A wide range of optimization algorithms can be chosen for this purpose, sometimes even in combination.

3.4. Systematic Map of the Capacity Planning Literature

Following the process outlined in the previous section, we select and map a set of publications on the topic to the taxonomy. The quantitative results of the selection process can be seen in Table 3.3. The 86 resulting publications are further filtered during the mapping process, resulting in 57 mapped publications, listed in Table 3.4.

Table 3.1: Overview of classes belonging to each category of the taxonomy.

Concern	Category	Abbreviation	Class
System Model	Workloads	VM	Virtual Machines
		DB	Databases
		S	Streaming Workloads
		BD	Big Data Frameworks
		WS	Web Service
		B	Batch Jobs
	Resources	C	Compute Hardware
		S	Storage Hardware
		N	Network Hardware
		E	Energy Hardware (Storage and Supply)
		H	Heat Control Hardware
	V	Virtualized Resources (VMs, containers, etc.)	
	Model Inputs	H	Historical Data
		RS	Resource Specifications
		B	(Micro)Benchmarks or Systematic Performance Tests
S		SLAs	
P		Pricing Data	
LC		Lease Contracts	
HP		Human Personnel-related Factors	
Forecast Model	Modeling Strategy	A	Analytical
		S	Simulation
		E	Real-world Experimentation
	Model Structure	U	Unconditional Extrapolation
		W	What-if Scenarios
Decision Support	Role	F	Forecast
		A	Adaptation Advice
	Type of Advice	N	Number of Resources
		T	Type of Resources
		L	Locality of Resources
	Advice Method	H	Heuristic
		R	Regression
		L	Local Search
		SS	Stochastic Search
		SP	Stochastic Programming
NN		Neural Network	
GT		Game Theory	
GA	Genetic Algorithm		
NLP	(Non)Linear Programming		

Table 3.3: Intermediate selection results per consulted source of publications. Counts of publications can overlap between sources.

Source	Initial results	After preliminary filter	After high-level content filter
Google Scholar	200	182	69
DBLP	55	46	33
Scopus	82	42	23
Unstructured Search	5	5	5

Table 3.4: Systematic summary of related work. Acronyms for classes within categories explained in Table 3.1. Dashes (-) indicate that no evidence was to be found for any of the possible classes.

Publication	Year	System Model			Forecast Model		Decision Support		
		Workloads	Resources	Model Inputs	Modeling Strategy	Model Structure	Role	Type of Advice	Advice Method
Kant and Youjip Won [71]	1999	WS	C, S, N	H, RS, B	A	W	E, A	N, T	R
Menascé et al. [92]	1999	WS	C, S, N	H, RS, B	A	W	E, A	N, T	R
Altiok and Gunduc [6]	2001	WS	C, S, N	RS	S	W	F	-	-
Almeida and Menasché [5]	2002	WS	C, S, N	H, P	A	U	F	N, T	-
Park and Kim [104]	2002	S	C, S, N	RS, B	A	W	F	-	-
Cherkasova et al. [30]	2004	S	C, S	H, RS, B, S	A	W	E, A	N	L
Rolia et al. [114]	2005	VM	C	H, RS, S	A, S	W	E, A	N, T	L
Gmach et al. [46]	2007	VM	C	H, S	A	W	E, A	N	R
Nakadai and Taniguchi [97]	2007	WS	C	H, RS, S	A	U	E, A	N, T	NLP
Rasmussen [110]	2007	-	C, E, H	H, RS, P	A	W	F	-	-
Zhang et al. [139]	2007	WS	C	H, RS, S	A	W	E, A	N	R
Jiang et al. [68]	2008	WS	C, S, N	H, RS	A	W	E, A	N, T	R
Lu et al. [86]	2008	B	C	H, RS	S	W	E, A	N	H, R, SVM
Risch et al. [113]	2008	-	V	H, RS, S, P, HP	A	W	E, A	N	SVM
Spellmann et al. [122]	2008	WS	C, S	RS, B, S, P	-	W	F	-	-
Zhang et al. [138]	2009	WS	C	RS, S, P	A	W	E, A	N	NLP
Lopes et al. [85]	2010	WS	V	H, RS, B, P	A	W	E, A	N, T	R
Mylavarapu et al. [96]	2010	VM	C	H, RS, S	A, S	W	E, A	N	GA
Ardagna et al. [10]	2011	VM	V	S, P	A	U	E, A	N, L	GT
Kounev et al. [77]	2011	DB	C, N	H, RS	S	W	E, A	N	L
Roy et al. [115]	2011	WS	C	H, RS, S	A	W	E, A	N	H
Shang et al. [119]	2011	VM	V	RS, S	A	U	E, A	N, T	R
Adinarayan [3]	2012	VM	C, S, N, V	H	A	W	E, A	N, L	-
Chaisiri et al. [25]	2012	VM	V	RS, P, LC	A	W	E, A	N, T	NLP
Jiang et al. [70]	2012	VM	C, S	H, S, P	A	W	E, A	N	R, NN, GA, SVM
Kouki and Ledoux [76]	2012	WS	V	H, S, P	A	W	E, A	N	L
Rao and Rao [109]	2012	WS	V	H, RS, S, P	A	W	E, A	N	NN
Ren et al. [112]	2012	-	E	H, RS, P	A	W	F	-	-
Wang et al. [134]	2012	VM	V	H, RS, P	A	W	E, A	N, T	H
Bashroush and Nouredine [13]	2013	S, WS	C, S, N	RS, B, S, P	A	W	E, A	N, T	R
Izurrieta and Geyer [65]	2013	BD	V	RS, B	A, E	W	F	-	-

Table 3.4: Systematic summary of related work (continued).

Publication	Year	System Model				Forecast Model			Decision Support	
		Workloads	Resources	Model Inputs	Modeling Strategy	Model Structure	Role	Type of Advice	Advice Method	
Pal and Hui [103]	2013	-	-	B, S, P	A	U	E, A	N	GT, NLP	
Patel et al. [105]	2013	-	C, N	H, RS, P, HP	A, S	W	F	-	-	
Chaisiri et al. [26]	2014	B	C	H, RS, P	A	U	E, A	N	SP	
Chen et al. [28]	2014	B	C	H, RS, S, P, HP	A	W	E, A	N, L	NLP	
Chiang and Ouyang [31]	2014	B	C	H, RS, S, P	A	W	E, A	N	H	
Dorsch and Häckel [36]	2014	-	C	H, RS, S, P	S	W	E, A	N, L	L	
Ghosh et al. [44]	2014	VM	C	H, RS, S, P, HP	A	W	E, A	N, T	SS	
Candeia et al. [22]	2015	WS	V	H, RS, S, P	S	W	E, A	N, T	H	
Goncalves et al. [47]	2015	WS	V	RS, S, P	E	W	E, A	N, T	H	
Sandar et al. [118]	2015	BD	C, S	H, RS, P	A	W	E, A	N	NLP	
Toosi et al. [124]	2015	VM	V	H, RS, S, P	A	W	E, A	N, T	H, SP	
Wang et al. [133]	2015	BD	C, S	H, RS, B, S	A	W	E, A	N	R	
Xu and Zhu [137]	2015	VM	C	H, RS, S, P	A	W	E, A	N	NLP	
Kong and Liu [75]	2016	B	E	H, RS, S, P	A	W	E, A	N, T	R	
Le et al. [80]	2016	B	C, E	H, RS, P	A	W	E, A	N, T	NLP	
Li et al. [83]	2016	VM	C	H, RS	A	W	E, A	N, T	SS	
Carvalho et al. [23]	2017	VM	C	H, RS, S	A	W	E, A	N	L	
Carvalho et al. [24]	2017	VM	C, S	H, RS, S	A	W	E, A	N	L	
Nazareth and Choi [98]	2017	-	C, S	H, RS, P, LC, HP	S	W	F	-	-	
Noreikis et al. [99]	2017	WS	C	RS, B, S	A, E	W	E, A	N	L	
Ostberg et al. [102]	2017	VM	C, S, N	H, RS, S, P	A, S	W	E, A	N, L	-	
Raei [108]	2017	VM	C	RS, S, P	A	W	E, A	N, L	SS	
Sadashiv et al. [116]	2017	-	C	H, RS, S, P	A	W	E, A	N, T	L	
Tang and Chen [123]	2017	VM	C	H, RS, S, P	A	W	E, A	N	GT, NLP	
Alipourfard et al. [4]	2019	BD, WS	N	H, RS, S, P	S	W	E, A	N, T, L	L	
Zheng et al. [140]	2019	DB	C	H, RS	A	U	E, A	N	NN	

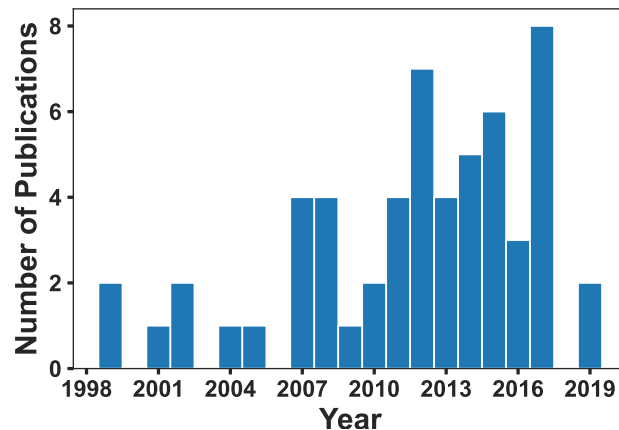


Figure 3.2: Histogram of the number of publications plotted against the year of publication.

3.5. Meta-Analysis of the Field

In this section, we present the results of our meta-analysis, following the method described in Section 3.2.5.

3.5.1. Year of Publication

We begin with a perspective on the distribution of years of publication. Figure 3.2 visualizes this perspective in the form of a histogram. Our main findings are:

SF1: The field of study has seen *significant increase in interest* in the last decade.

SF2: We also see a *sharp decline* in the number of publications recently published.

We observe that this field of study has been pursued since (at least) 1999 and has seen a significant increase in interest around 2007. Compared to the broader capacity planning practice across industries, research activities in this field begin decades later, but this can be attributed to the upcoming nature of computer systems, in general. We see the majority of work being published between 2012 and 2017. Only two publications have followed since that period, forming a sharp decline in publications in very recent years. This decrease can only partially be explained with publication and indexing delays: At the time of writing this chapter, 2 years have passed since the end of 2017, with no publication appearing in 2018.

3.5.2. Inter-Class Correlations

We now analyze correlations between classes. Figure 3.3 visualizes such a correlation matrix. Our main findings are:

SF3: We find that if systems consider *compute, storage, or network resources*, they are more likely to consider another resource type from this set, as well.

SF4: We observe a *strong separation between approaches* using *analytical* modeling and those using *simulation* modeling—the combination of both is rare.

Upfront, we observe two perpendicular line of non-correlation on the dimension of class F (forecasting) in the role category. This is mandated by the design of this survey: Our scope is limited to systems that do some flavor of forecasting (or modeling usable for forecasting).

We observe a number of positive correlations that stand out in magnitude. Classes C (compute), S (storage), and N (network) in the resources category appear positively correlated, indicating that when capacity planning works address one, they tend to address one or more others, as well. This is evidence for a perceived bisection of the field between compute-centered efforts and non-compute capacity planning. Classes E (energy) and H (heat control) also appear positively correlated. We see this stemming from the strong relationship the management of both resources has, with heat control being a major consumer of energy. In

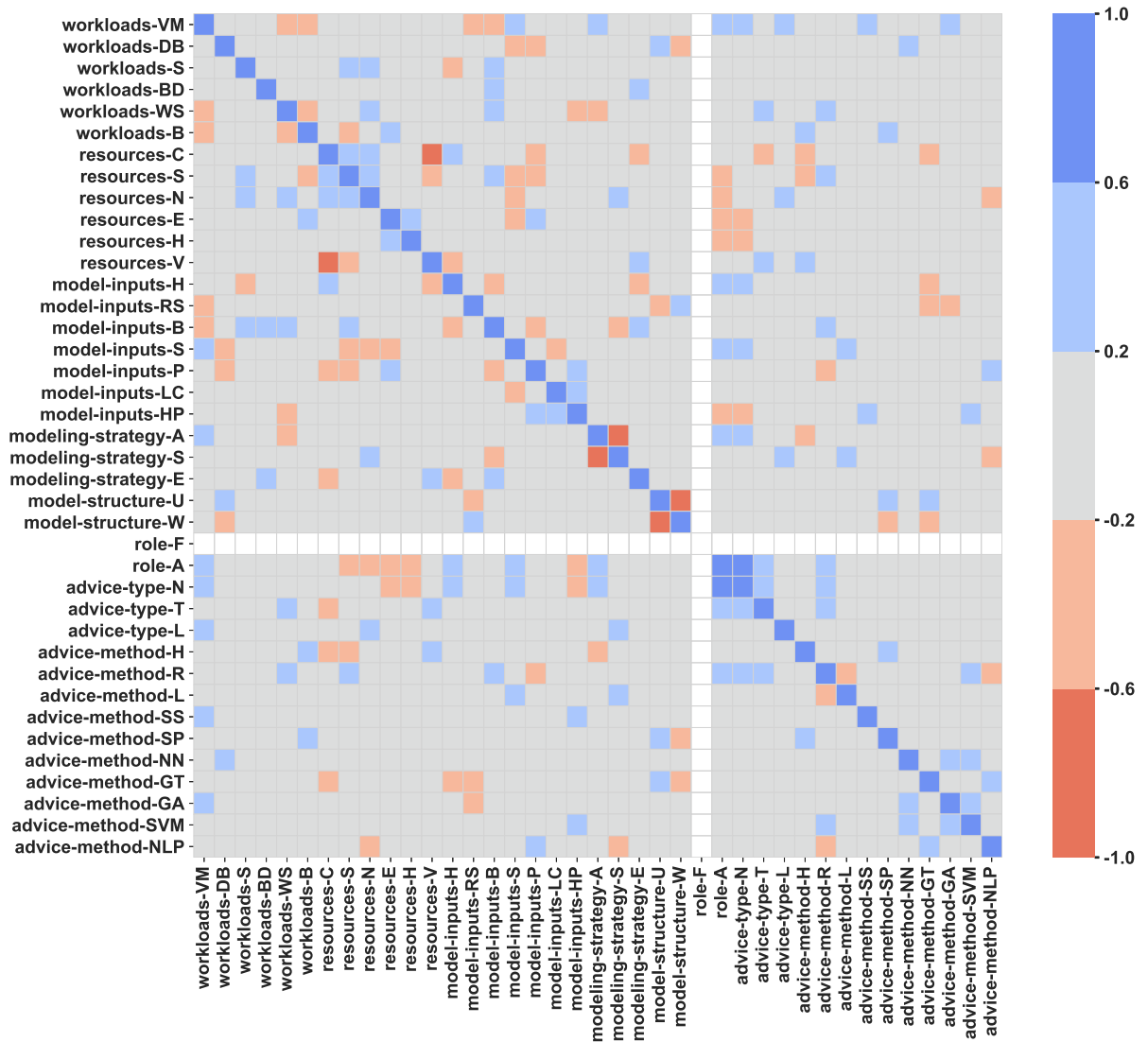


Figure 3.3: Pairwise correlation heatmap of all classes of the taxonomy.

the decision support phase, class A (advice) of the decision support role category and class N (number of resources) in the type of advice category appear strongly correlated. This indicates that advisory capacity planning systems tend to advise from a quantitative angle rather than a qualitative or locality angle, both of which are far less correlated with class A.

Significant negative correlations can be found between class V (virtual) and class C (compute) of the resources category. This result meets expectations, since a capacity planning approach tends to focus on either virtual or physical resources, not both. In modeling, the strategy also shows a partial mutual exclusion between analytical and simulation-based modeling strategies. The strong division observed here indicates that little work has gone into investigating cross-overs and hybrid strategies employing both. The model structure category is completely mutually exclusive, due to the definition of the two classes: U (unconditional) and W (what-if based) have no overlap in the definition of this taxonomy.

Except for these smaller clusters of activity, the design space appears rather weakly correlated. There are, however, fields of weak consistently positive/negative correlations in the advice method. It seems that most advice methods are negatively correlated with each other, which indicates that most system rely on only one method to optimize the proposed decision for the capacity planner.

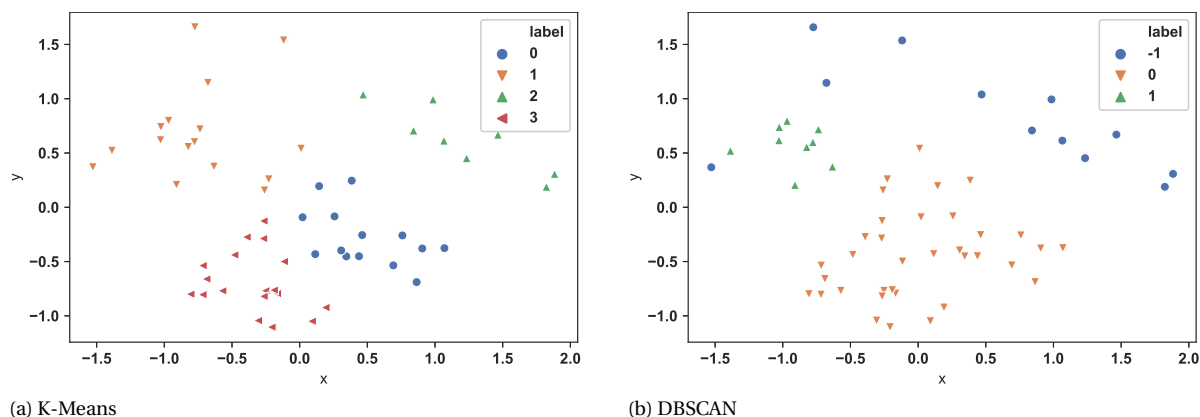


Figure 3.4: Visualization of clusters found with different clustering methods, in dimensionality-reduced space.

3.5.3. Clustering Analysis

We now proceed to a clustering analysis of the mapped publications. The clustering results of all algorithms are visualized in Figure 3.4. Our main findings are:

SF5: We find that clustering analysis of the taxonomy space can provide *insightful, unique perspectives* on the capacity planning community.

SF6: We provide evidence that approaches common in the community *differ significantly over time*.

We now discuss each of the clustering methods and their results. First, we apply K-means clustering. The clusters are listed in Table 3.5. For each category, we filter and sort the used classes by their frequency and then extract the two highest-scoring, meeting a lower threshold of 0.25. Looking at the aggregate meta-data of both clusters, the mean publication years differ, with cluster 2 being the oldest and cluster 3 the newest. We now discuss the clusters in chronological order. The environments in cluster 2 (2005) can be characterized as web services on conventional physical infrastructure. The capacity planning approaches in this cluster are primarily specification- and benchmark-based, with regression as advice method. The environments in cluster 1 (2012) can be seen as the evolution of the previous cluster: web services (and modern virtual services) running on virtual resources. We see that the advice is predominantly quantitative in this cluster, much more so than in the previous cluster. We conjecture that this is due to virtual resources coming in more easily quantifiable standardized units (less so than leased hardware). In cluster 0 (2012), we see a body of work focusing on compute services and primarily using historical data as inputs (next to specifications). This is also the only cluster with a sizable portion of publications reporting simulation efforts next to the dominating analytical approach. Quantitative advice is the main support angle here, with no others meeting the threshold. The last cluster (number 3, year 2014) is the most modern, featuring virtual workloads on compute resources. Again there is a strong focus on historical data, but notably this cluster consists of only advising approaches (none purely forecast). The most popular advice optimization method is Local Search, although the field is fragmented in this category (seen by the low value of L and the fact that it is the only one to meet the threshold).

We now apply the DBSCAN algorithm [39] to the dimensionality-reduced dataset. As can be seen in Table 3.6, this gives a significantly different clustering. A formal difference is the differentiation between labeled and unlabeled data, unlabeled data being publications that the clustering algorithm could not confidently assign to a cluster. All labeled data has the same publication year: 2012. This data is divided into two clusters, both with mainly virtual workloads. The difference between the two is that the one uses physical and the other uses virtual resources as levels of abstraction. Cluster 0 (physical) also shows more historical data being used, while cluster 1 (virtual) has more focus on pricing and SLAs. It is interesting to see that the choice of resource abstraction also reflects in the choice of model inputs used for capacity planning. Unlabeled data consists mainly of web service capacity planning efforts, which the clustering algorithm separates from the main two clusters. We see this as a greater trend in the field: Older publications tend to focus on web services on physical resources, while more recent publications become more virtualized.

Table 3.5: Top two most frequent classes per category for each cluster of the K-Means clustering method, above a threshold of 0.25. The relative frequency is listed in parentheses.

Category	Cluster 0	Cluster 1	Cluster 2	Cluster 3
Pub. Year: Mean	2012	2012	2005	2014
Pub. Year: Std.	4.39	3.14	5.61	3.54
Num. Publications	14	16	9	18
Workloads	–	WS (0.50), VM (0.31)	WS (0.78)	VM (0.67)
Resources	C (0.93), S (0.36)	V (0.69)	S (0.89), C (0.89)	C (1.00)
Model Inputs	RS (0.93), H (0.93)	P (0.88), S (0.81)	RS (0.89), B (0.67)	H (0.94), RS (0.89)
Modeling Strategy	A (0.79), S (0.29)	A (0.81)	A (0.78)	A (0.94)
Model Structure	W (0.93)	W (0.75), U (0.25)	W (0.89)	W (0.94)
Role	F (1.00), A (0.71)	F (1.00), A (1.00)	F (1.00), A (0.44)	F (1.00), A (1.00)
Advice Type	N (0.71)	N (1.00), T (0.62)	T (0.56), N (0.56)	N (1.00), T (0.28)
Advice Method	–	NLP (0.25), H (0.25)	R (0.44)	L (0.28)

Table 3.6: Top two most frequent classes per category for each cluster of the DBSCAN clustering method [39], above a threshold of 0.25. The relative frequency is listed in parentheses.

Category	Cluster 0	Cluster 1	Unlabeled
Pub. Year: Mean	2013	2012	2007
Pub. Year: Std.	4.15	2.29	6.05
Num. Publications	35	9	13
Workloads	VM (0.37)	VM (0.44)	WS (0.77)
Resources	C (0.94)	V (0.78)	S (0.62), C (0.62)
Model Inputs	RS (0.91), H (0.91)	P (0.89), S (0.78)	RS (0.85), P (0.54)
Modeling Strategy	A (0.86), S (0.26)	A (1.00)	A (0.69)
Model Structure	W (0.91)	W (0.78)	W (0.85)
Role	F (1.00), A (0.89)	F (1.00), A (1.00)	F (1.00), A (0.62)
Advice Type	N (0.89)	N (1.00), T (0.56)	N (0.69), T (0.62)
Advice Method	L (0.26)	–	R (0.38)

3.5.4. Analysis Per Category

We now analyze the state and trends of each category, in turn. Our main findings here are:

SF7: A large majority of publications (82%) consider *only one workload type* in their approach.

SF8: Similarly, a majority of all publications (63%) consider *only one resource type*.

SF9: We find that the *analytical modeling approach is currently dominant* in the field.

SF10: The *type of advice* that published capacity planning systems provide is *predominantly quantitative*. Other types of advice, such as locality and qualitative concerns, are less popular.

System Model: Workloads

In the workload category, most studies (82%) only consider one workload type. Only two publications consider multiple, which is remarkable considering the heterogeneous deployments prevalent today. As can be seen in Figure 3.5a, virtual and web workloads dominate the total class counts. In Figure 3.5b, we observe an increasing trend in VM considerations in the past decade, while web services have a longer, more stable history.

System Model: Resources

63% of all studies consider only one resource type, while 18% and 16% consider 2 and 3 types of resources, respectively. While still predominantly single-dimension, it seems that more publications look into combinations of multiple resource types than multiple workload types. We conjecture that this is evidence for capacity planning being done on a per-workload basis. The compute resource class dominates the field, as shown in Figure 3.5c. In Figure 3.5d, we observe that this type of resource sees increased interest over time, having a large share especially recent years. Virtual resources started gaining traction in the field of capacity planning around 2010, which is also when we see it gaining momentum as a standard mode of deployment in clouds. Among the other classes, storage remains relatively constant in its interest over time. This seems to be a resource of interest for a long time, in this field.

System Model: Model Inputs

We find that most models use 3 inputs (at 40%), followed by 4 inputs (30%) and 2 inputs (19%). As seen in Figure 3.5f, there are four types of inputs leading the publication counts over time: resource specifications, pricing, SLAs, and historical information. This is a combination of both purely technical (specifications and historical information) and more service-centered factors (pricing and SLAs), further emphasizing the multi-disciplinary nature of the capacity planning activity. These findings are confirmed by the total class counts in Figure 3.5e. There seems to be little work focusing on the human personnel factors involved in the process.

Forecast Model: Modeling Strategy

In the strategy used for creating a forecast, we observe in Figure 3.6a that the analytical approach seems to be the standard in the field. If we look back to the development of this over time (see Figure 3.6b), this predominance seems to have been less clear in the beginning, but becomes more firmly established in the past decades. Simulation sees a slight increase in popularity in the past years, possibly related to the increasing complexity of modeled systems. The entry of experiment-based approaches from 2013 on could support this conjecture, since experiments are another way around the complexity limitations of analytical experiments.

Forecast Model: Model Structure

In terms of the structure of models, we see only very few models being unconditional in Figure 3.6c. This indicates a broad understanding that capacity planning models need to be conditional and open-ended in their construction to allow for different plausible future consequences.

Decision Support: Role

In terms of the role that capacity planning approaches take, most (84%) seem to not only be forecasting but also advisory (see Figure 3.7a). This indicates that most researched systems provide a suggested capacity decision to the practitioner.

Decision Support: Type of Advice

The type of advice that published capacity planning systems provide is predominantly on the quantity dimension. Qualitative and locality aspects are far less frequent. This can be seen in Figure 3.7c. We observe that a significant portion of systems give two types of advice (44%) while most others give only one type of advice (40%). This indicates that many published approaches actually provide two-dimensional decision support. When looking at the trends over time (see Figure 3.7d), we observe that locality has become a topic of interest only in the last decade. This could be related to the growing popularity of multi-site datacenter operations.

Decision Support: Advice Method

The field of methods used to deduce advice to the practitioner seems more diverse. Most use only one advice method (72%), with only very few using more than one. As can be seen in Figure 3.7e, the spread of method usages is quite high. Still, regression, Non-Linear Programming and heuristic-based approaches dominate total counts. In Figure 3.7f, we can observe little clear trends over time. We see neural-network-based approaches emerging around 2012, which can be related to the increasing popularity this method has experienced in other fields.

Overall, the field of capacity planning still seems to mainly rely on simple, heuristics-based approaches. This can either indicate that the field is not complex enough to warrant more complex analysis, or that more complex analysis is simply not tractable for the scale at which this is needed. Although hard proof for this is not easily obtainable, we conjecture that the latter is the case.

3.5.5. Trending Keyword Analysis

We close off with an analysis of keyword trends over time. We list the results of this search below (emphasis is ours). Our main finding:

SF11: We find that trending keywords such as *penalties*, *underutilization*, and *multilevel* can indicate upcoming trends in the capacity planning community.

benefit, better, block, building, collected, derived, economy, edge, fog, google, highlevel, ie, implemented, internet, iot, limit, maximize, monitoring, **multilevel**, **penalty**, prevalent, price, publicly, reclaimed, **region**, revenue, rigorous, sell, short, size, **slas**, stochastic, studied, theoretical, **underutilization**, **unused**, variation, wireless, would

We find a number of keywords corresponding with interesting trends in the field, both inside and outside of capacity planning. There seems to be increasing focus on *penalties* and *SLAs*, as cloud infrastructure pricing models become more established and capacity planners likely become more aware of their influence on capacity plans. We also observe that *underutilization* (and synonym *unused*) is a trending keyword. This could be explained by a growing need for more accurate capacity plans. We also see multi-facility capacity problems gaining traction (*multilevel* and *region*). This could be linked with the increasingly distributed and globalized nature of cloud operations, leading capacity planners to increase the scope of capacity.

3.6. Comparison to Theory

We now compare the theory presented in Section 2 to the findings of this survey. Overall, the workflow that theories adhere to (model–predict–decide) is roughly followed by the majority of the publications. In terms of modeling strategy, the predominance of analytical models in theoretical literature is reflected in the more practical published literature on capacity planning approaches. The substantial portion of web service workloads in surveyed publications also can be correlated with the attention these workloads receive in analyzed books.

We also observe significant discrepancies. First, we see that far from all capacity planning approaches use historical measurements, while literature indicates that this is an integral part of the process. A plausible explanation for this could be that these are not always readily available in real-world environments, or that their size is too large to allow for quick analysis. Second, the times of publication also differ between theory and practice, although perhaps understandably so: While the majority of relevant capacity planning books was published before 2010, the majority of relevant capacity planning approaches were published after 2010. It is not uncommon for practice to follow theory in this fashion, but the absence of a standard theory for cloud capacity planning is noticeable. Finally, we do not include a category for calibration or validation in

this taxonomy. Even though it is a crucial step in our view, it simply has not been integrated as a major stage in the publications we find. We estimate that there still is much to be gained from a thorough investigation of long-term calibration/validation efforts in capacity planning.

3.7. Discussion

We now summarize the contributions of this chapter and discuss possible threats to their validity.

3.7.1. Summary

The state-of-the-art of literature in a field can be difficult to grasp without an overview. We provide such an overview in the form of a systematic literature survey for capacity planning in cloud infrastructures. We collect and filter a large number of publications and design a taxonomy to provide a structured view of published approaches. We also demonstrate novel meta-analysis methods to uncover trends and groupings in the research community and their approaches.

3.7.2. Threats to Validity

We discuss potential threats to internal validity, construct validity, and external validity.

Internal Validity

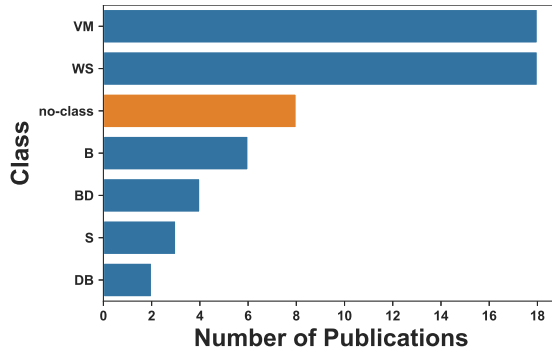
While we present many correlations in this study, we take care to mark suspected causal claims as such. By limiting the causal claims made and maintaining an observational role, we ensure the internal validity of our study.

Construct Validity

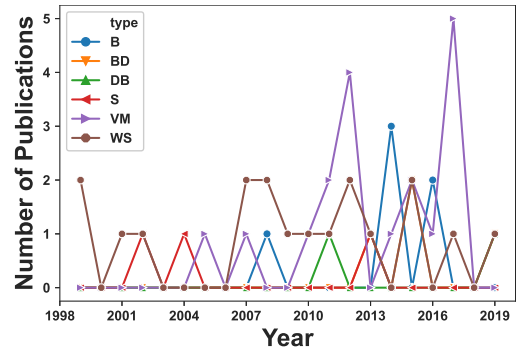
The inherently ambiguous nature of the mapping process is a threat to the construct validity. Even though the author took great care to classify the works as objectively as possible, the classifications are inherently subjective as they rely on interpretation of (frequently ambiguous) free-form text. Misinterpretations can therefore stem from the author of this thesis misreading passages of the text. Another source of ambiguity can be an incomplete portrayal by the author of the work published in the publication, be it due to omissions of sensitive details or a selective perspective to focus the reader's attention. These inherent ambiguities could be reduced through interviews with the authors, although these interviews would also introduce their own bias. A different approach would be to duplicate mapping efforts across a panel of experts and compare their results, although this can only partly address the issue.

External Validity

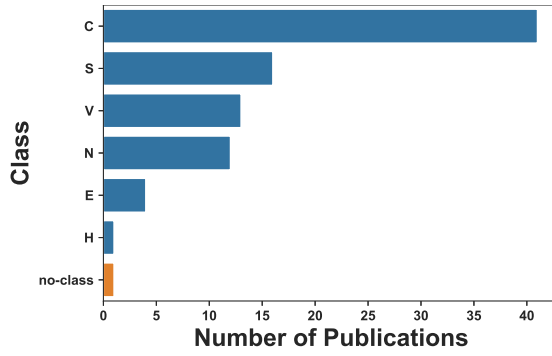
The results of this survey are likely skewed in favor of approaches that are more easily published about than actually put to practice. This threat to external validity is inherent to a survey that only regards published work. The interviews conducted in Chapter 4 of this thesis complement this literature-based approach with a practice-driven one to assess the current real-world practice in capacity planning and uncover possible discrepancies between both.



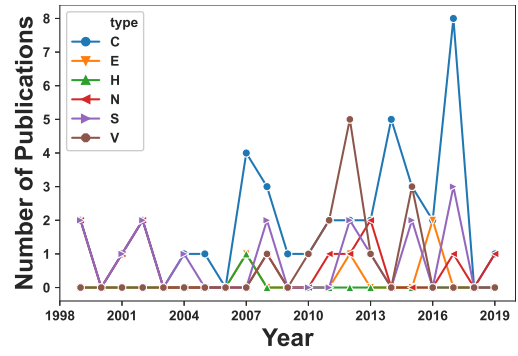
(a) Workloads: class counts



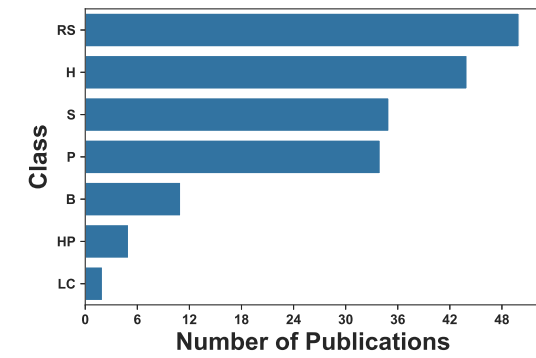
(b) Workloads: class trends



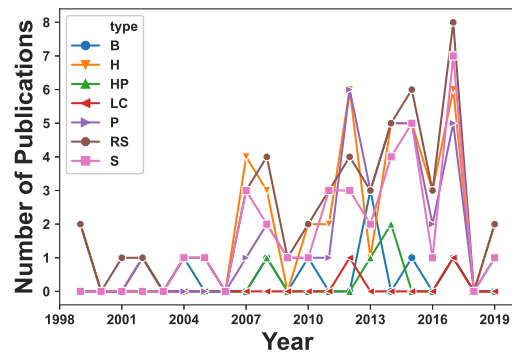
(c) Resources: class counts



(d) Resources: class trends

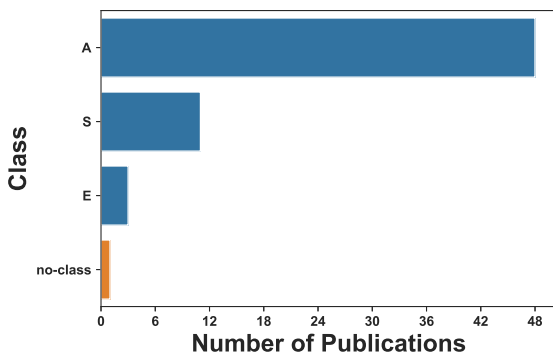


(e) Model Inputs: class counts

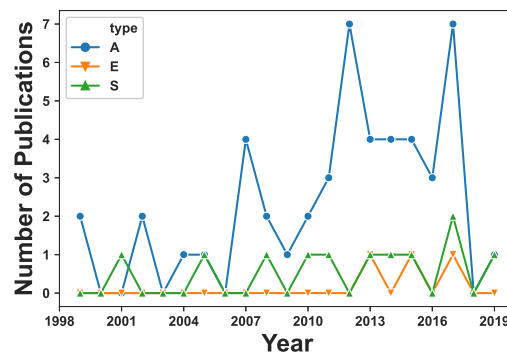


(f) Model Inputs: class trends

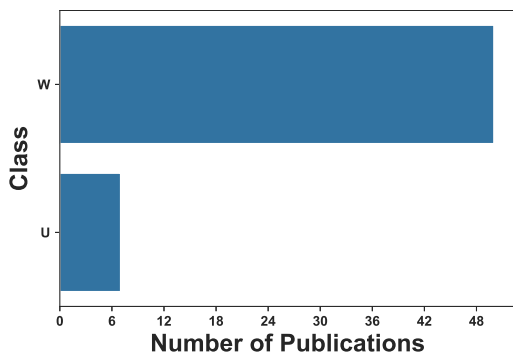
Figure 3.5: Class counts and trends for the System Model section per category. An additional class type (no-class) is introduced to indicate the number of publications without any class, when such publications exist in that category.



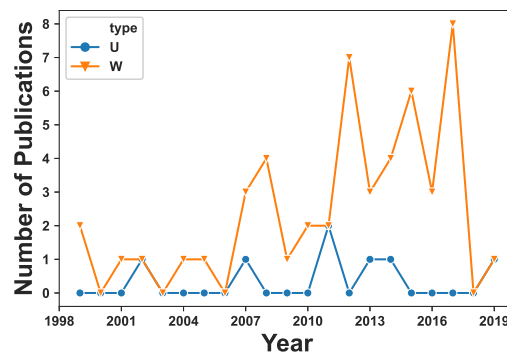
(a) Modeling Strategy



(b) Modeling Strategy



(c) Model Structure



(d) Model Structure

Figure 3.6: Class counts and trends for the Forecast Model section per category. An additional class type (no-class) is introduced to indicate the number of publications without any class, when such publications exist in that category.

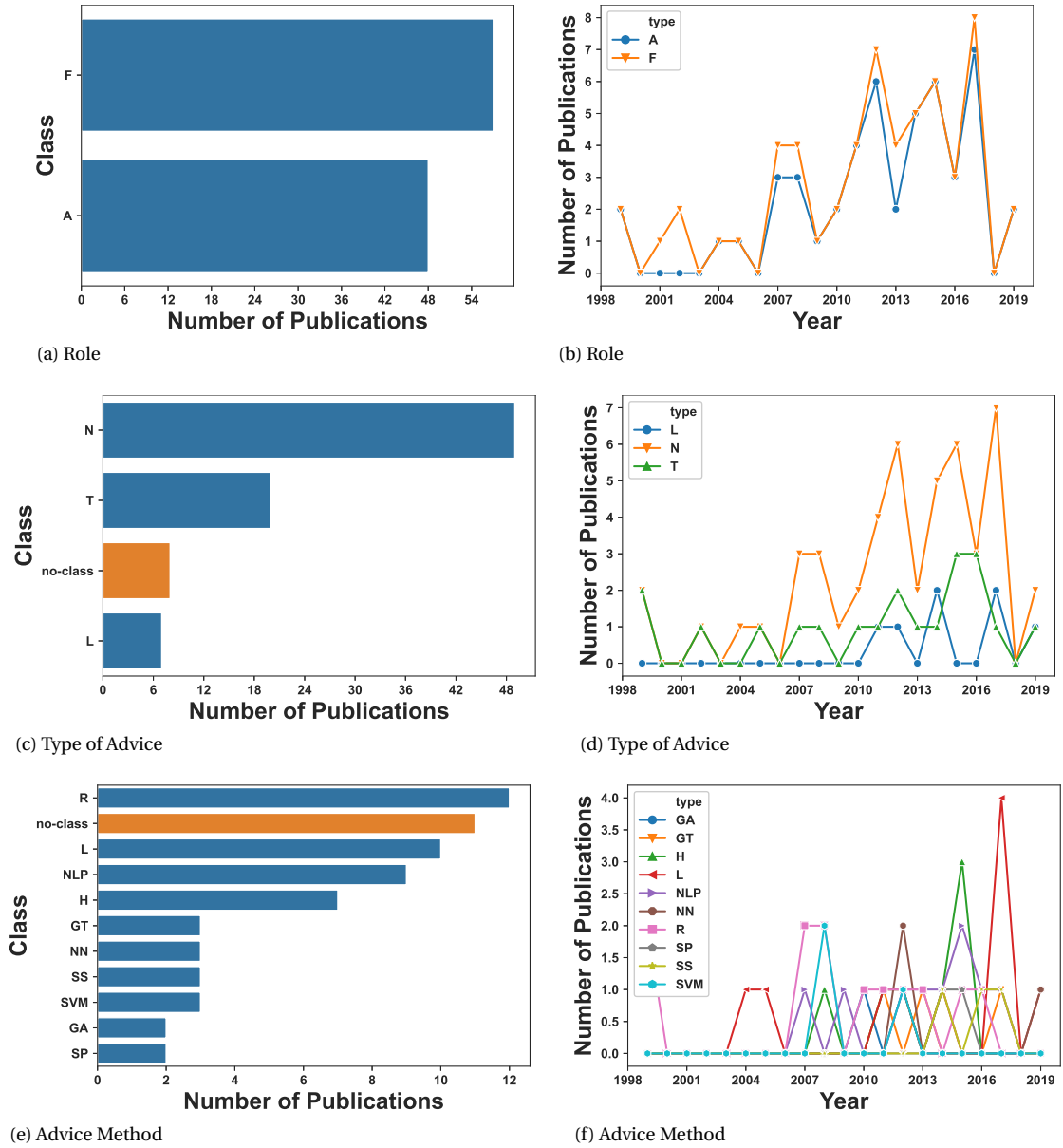


Figure 3.7: Class counts and trends for the Decision Support section per category. An additional class type (no-class) is introduced to indicate the number of publications without any class, when such publications exist in that category.

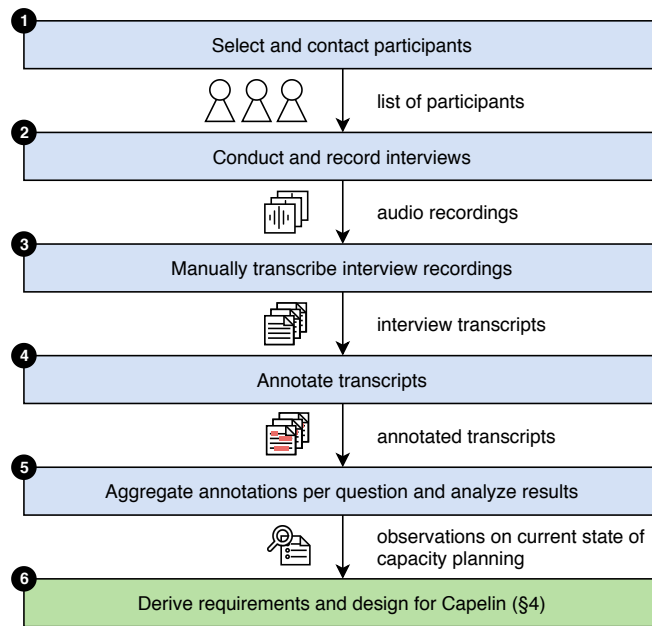


Figure 4.1: Our process for conducting and analyzing interviews with capacity planners.

4

Real-World Survey of Capacity Planning

Real-world practice can deviate significantly from published theories and strategies. In this chapter, we conduct and analyze interviews with 8 practitioners from a wide range of backgrounds and multiple countries, to assess whether this is the case in the field of capacity planning. Our method and results address research question RQ1.2.

4.1. Overview

As detailed in Section 2.4.2, we observe a severe lack of published knowledge on the actual real-world practice of capacity planning in datacenters. The two works that can be related to this are either decades old or focus on a management perspective. This is surprising for a problem of this importance, because any research attempting to improve the state-of-the-art in this field needs to be aware of existing challenges and any tacit knowledge.

In this chapter, we address this lack of knowledge by conducting and analyzing interviews with practitioners in the field, using a technique that is infrequently applied in computer systems research. The remainder of this chapter is structured as follows. We present the interview and analysis method in Section 4.2. Section 4.3 lists a selection of our main observations made in analyzing the interviews. In Section 4.4, we provide the full set of observations. Finally, we summarize our contributions and discuss threats to validity in Section 4.5.

4.2. Method

Our goal is to collect real-world experiences from practitioners systematically and without bias, yet also leave room for flexible, personalized lines of investigation. To this end, we adapt a classic method to our specific goals. Figure 4.1 depicts the data collection process. The remainder of this section details it.

4.2.1. Interview Type

The choice of interview type is guided by the trade-off between the systematic and flexible requirements. A text survey, for example, is highly suited for a systematic study, but generally does not allow for low-barrier individual follow-up questions or even conversations. An in-person interview without pre-defined questions allows full flexibility, but can result in unsystematic results.

We use the *general interview guide approach* [125], a semi-structured type of interview that ensures certain key topics are covered but permits deviations from the script. We conduct in-person interviews with a prepared script of ranked questions, and allow the interviewer the choice of which scripted questions to use and when to ask additional questions. Such additional questions can follow up on unanticipated statements made by interlocutors or focus on aspects specific to the interlocutor's situation (e.g., specific mentioned processes or experiences).

4.2.2. Data Collection

Our data collection process involves three steps. Firstly, we *selected and contacted* a broad set of prospective interviewees representing various kinds of datacenters, with diverse roles in the process of capacity planning, and with diverse responsibilities in the decisions.

Secondly, we *conducted and recorded interviews*. Each interview is conducted in person or via video call connection by the author of this thesis and digitally recorded with the consent of the interlocutor. For each interview, the language that is most comfortable for both participants is chosen, resulting in 3 different languages in the different conducted conversations. Interviews last between 30 and 60 minutes, depending on availability of the interlocutors and complexity of the discussion. To help the interviewer select questions and fit in the time-limits imposed by each interviewee, we rank questions by their importance and group questions broadly into 5 categories: (1) introduction, (2) process, (3) inside factors, (4) outside factors, and (5) summary and followup. The choice between questions is then dynamically adjusted to give precedence to higher-priority questions and to ensure each category is covered at least briefly. The script itself is listed in Appendix A.

Thirdly, the recordings are *manually transcribed* into a full transcript to facilitate easy analysis. Because matters discussed in these interviews may reveal sensitive operational details about the organisations of our interviewees, all interview materials are handled *confidentially*. No information that could reveal the identity of the interlocutor or that could be confidential to an organization's operations is shared without the explicit consent of the interlocutor. In addition, all raw records (recordings, transcripts, and any analysis material derived from the transcripts bearing confidential material) will be destroyed directly after this study.

4.2.3. Analysis of Interviews

Due to the unstructured nature of the chosen interview approach, we combine a question-based aggregated analysis with incidental findings. Our approach is inspired by the Grounded Theory strategy set forth by Coleman and O'Connor [32], and has two steps. First, for each transcript, we *annotate* each statement made based on which questions it is relevant to. This may be a sub-sentence remark or an entire paragraph of text, frequently overlapping between different questions. We augment this systematic analysis with more general findings, including comments on topics unanticipated by our script or specific to the domain of expertise of the interlocutor. Secondly, we traverse all transcripts for each question and form *aggregate observations for each question* in the transcript. From these, we synthesize Capelin requirements (see Section 5.2).

In total, we transcribed over 35,000 words in 3 languages, which is a very large amount of raw interview data. We conducted interviews with 8 practitioners from commercial and academic datacenters, with roles ranging from capacity planners, to datacenter engineers, to managers. One of the interviews was conducted with 2 practitioners in parallel, leading to a total of 7 interviews.

Table 4.1: Summary of interviews. (Notation: TTD = Time to Deploy, CP = Cloud Provider, MDC = Multi-Datcenter, SDC = Single-Datcenter, M = Monitoring, m/y = month/year, NIT = National IT Infrastructure Provider, SA = Spreadsheet Analysis.)

Int.	Role(s)	Backgr.	Scale	Scope	Tooling	Workload	Frequency	TTD
1	Researcher	CP	rack	MDC	M	combined	3m, ad-hoc	?
2	Board Member	NIT	iteration	MDC	–	combined	4–5y	12–18m
3	Manager, Eng.	CP	rack	MDC	M	combined	ad-hoc	4–5m
4	Manager	NIT	iteration	SDC	M	benchmark	6–7y	18m
5	Hardware Eng.	NIT	iteration	SDC	M	benchmark	6y	18m
6	Researcher	NIT	rack	MDC	M	separate	6m	12m
7	Manager	NIT	iteration	MDC	M, SA	combined	5y	3.5-4y

4.3. Main Observations from the Interviews

Table 4.1 summarizes the results of the interviews. We summarize here our main observations, forward referencing the detailed findings we provide in Section 4.4. Each observation is detailed here and indexed as O_x , e.g., $O1$. Observations are based on individual findings, which can be seen as smaller observations. Individual findings are further detailed in Section 4.4 and indexed as IF_x , e.g., $IF16$ related to $O1$.

O1: A majority of practitioners find that the process involves a *significant amount of guesswork and human interpretation* (see detailed finding $IF16$ in §4.4). Interlocutors managing commercial infrastructures emphasize multi-disciplinary challenges such as lease and support contracts, and personnel considerations ($IF19$, $IF18$).

O2: In all interviews, we notice the *absence of any dedicated tooling* for the capacity planning process ($IF44$). Instead, the surveyed practitioners rely on visual inspection of data, through monitoring dashboards ($IF45$). We observe two main reasons for not using dedicated tooling: (1) tools tend to under-represent the complexity of the real situation, and (2) have high costs with many additional, unwanted features ($IF47$).

O3: The organizations using these capacity planning approaches provide a *range of digital services*, ranging from general IT services to specialist hardware hosting ($IF2$). They run VM workloads, in both commercial and scientific settings, and batch and HPC workloads, mainly in scientific settings ($IF3$).

O4: A *large variety of factors* are taken into account when planning capacity ($IF34$). The three named in a majority of interviews are (1) the use of historical monitoring data, (2) financial concerns, and (3) the lifetime and aging of hardware ($IF35$).

O5: *Success and failure* in capacity planning are underspecified. Definitions of success differ: two interviewees see the use of new technologies as a success ($IF6$), and one interprets the absence of total failure events as a success ($IF5$). Challenges include chronic underutilization ($IF8$), increasing complexity ($IF9$), and small workloads ($IF10$). Failures include decisions taking long ($IF12$), misprediction ($IF13$), and new technology having unforeseen consequences ($IF11$).

O6: *The frequency of capacity planning processes seems correlated with the duration of core activities using it:* commercial clouds deploy within 4-5 months from the start of capacity planning, whereas scientific clouds take 1–1.5 years ($IF39$, $IF40$).

O7: We found three *financial and technical factors* that play a role in capacity planning: (1) funding concerns, (2) special hardware requests, and (3) the cost of new hardware ($IF74$). In two interviews, interlocutors state that financial considerations prime over the choice of technology, such as the vendor and model ($IF78$).

O8: The *human aspect* of datacenter operations is emphasized in 5 of the 7 interviews ($IF85$). The data-center administrators need training ($IF81$), and wrong decisions in capacity planning lead to stress within the operational teams ($IF83$). Users also need training, to leverage heterogeneous or new resources ($IF81$).

O9: We observe a *wide range of requirements and wishes expressed by interlocutors* about custom tools for the process. Fundamentally, the tool should help manage the increasing complexity faced by capacity planners (IF97). A key requirement for any tool is interactivity: practitioners want to be able to interact with the metrics they see and ask questions from the tool during capacity planning meetings (IF95). The tool should be affordable and usable without needing the entire toolset of the vendor (IF96). One interviewee asks for support for infrastructure heterogeneity, to support scientific computing (IF98).

O10: Two interviewees detail “*what-if*” scenarios they would like to explore with a tool, using several dimensions (IF101): (1) the *topology*, in the form of the computational and memory capacity needed, or new hardware arriving; (2) the *workload*, and especially emerging kinds; and (3) the *operational phenomena*, such as failures and the live management of the platform (e.g., scheduling and fail-over scenarios).

Table 4.3: An index of the detailed interview findings listed in Section 4.4. (Notation: CP = Capacity Planning, WL = Workload.)

ID	Description	ID	Description	ID	Description
IF1	importance of availability	IF37	commercial factors	IF73	special regulations
IF2	wide range of services	IF38	margin for error	IF74	main fin. and tech. factors
IF3	services per domain	IF39	frequency of plans	IF75	fin. factors in comm. settings
IF4	scope of success	IF40	time to deployment	IF76	fin. factors in scientific settings
IF5	absence of failures as success	IF41	frequency in special cases	IF77	relevance of factors in finances
IF6	benefits of new technology	IF42	historic data sources	IF78	role of finances in tech. choice
IF7	abundance of failure stories	IF43	other data sources	IF79	investment and energy
IF8	underutilization	IF44	absence of tooling	IF80	main human factors
IF9	complexity	IF45	prevalence of visual formats	IF81	need for training
IF10	role of small deployments	IF46	monitoring products	IF82	listening to users
IF11	failures of new technology	IF47	tool issues	IF83	role of CP in team stress
IF12	long decision processes	IF48	frequency of errors	IF84	proportionality of personnel size
IF13	misprediction of resource ratios	IF49	severity of errors	IF85	importance of the human factor
IF14	CP as afterthought	IF50	error reporting	IF86	scope of CP
IF15	periodicity of CP	IF51	timing of incident recollections	IF87	more fine-grained decisions
IF16	guesswork and interpretation	IF52	number of service types	IF88	separating prod. and dev.
IF17	non-computational factors	IF53	commercial services considered	IF89	treating everything as prod.
IF18	financial vs. human factors	IF54	scientific services considered	IF90	what-if scenarios with factors
IF19	lease and support contracts	IF55	CP for business critical WLs	IF91	ideal process: flexibility
IF20	use of benchmarks	IF56	CP for big data WLs	IF92	ideal process: speed
IF21	role of big users	IF57	CP for serverless WLs	IF93	ideal process: attention to detail
IF22	scientific infra. perceptions	IF58	CP for HPC WLs	IF94	ideal process: training
IF23	yes/no: what-if	IF59	seeing WLs as one	IF95	ideal tool: interactivity
IF24	yes/no: hybrid clouds	IF60	combining WLs in benchmarks	IF96	ideal tool: availability
IF25	yes/no: share data	IF61	risk of combining WLs	IF97	ideal tool: complexity
IF26	yes/no: human factors	IF62	popularity of serverless	IF98	ideal tool: heterogeneity
IF27	common stakeholders	IF63	characteristics of serverless	IF99	ideal tool: multi-disciplinarity
IF28	level of the final decision	IF64	CP for serverless: expectations	IF100	ideal tool: trend analysis
IF29	commercial stakeholders	IF65	CP for serverless: current state	IF101	ideal tool: what-if scenarios
IF30	scientific stakeholders	IF66	lack of data on serverless	IF102	absence of direct links
IF31	time scale in general	IF67	what-if scenarios in the process	IF103	links to scheduling
IF32	time scale in commercial settings	IF68	role of regulatory constraints	IF104	inherent uncertainty
IF33	time scale in scientific settings	IF69	constraints in banking	IF105	regional differences
IF34	plurality of factors	IF70	constraints due to GDPR	IF106	coherence of capacity plans
IF35	main factors	IF71	competitive dialogue procedure		
IF36	scientific factors	IF72	role of security in regulation		

4.4. Full Observations from the Interviews

We now list the full set of interview findings from which the main set of observations discussed in the previous section were derived. We aggregate these findings per question in the interview script (see Appendix A). Table 4.3 serves as a tabular index for all findings, to ease navigation.

Category 1 - Q1: Importance of Availability

IF1: We observe that availability appears to be critical in commercial cloud environments. In scientific cloud infrastructure, availability generally appears to be perceived as less important than in commercial cloud environments.

Category 1 - Q2: Services

IF2: The organizations for which the interlocutors work provide a wide range of services, from general IT services to specialist hardware hosting. The interlocutors themselves are mainly concerned with compute cloud services. These can be divided into virtualized offerings (VM hosting), batch workload services, and specialized HPC hosting.

IF3: Batch workloads and HPC hosting are only seen in the scientific infrastructure surveyed. VM hosting is dominant in the commercial infrastructure space and can only be found in half of the surveyed scientific infrastructures.

Category 1 - Q3: Success in Capacity Planning

IF4: The success stories being told vary from large installations with significant upfront effort to more flexible, iterative installations. Flexibility is still often valued even in large scale designs, in the form of strategies leaving room for adaptations later-on.

IF5: One interview characterizes the absence of total-failure scenarios in the past as a success story for the capacity planning team.

IF6: The utilization of new hardware with beneficial features, such as competitive pricing or an increase in parallelism, is a success story recounted in two of the interviews.

Category 1 - Q4: Insuccess in Capacity Planning

IF7: A first observation is the abundance of failure stories, especially compared to the number of success stories. A possible explanation is that the process could be largely taken for granted. The practices is mainly revisited when suboptimal situations arise.

Challenges Capacity Planners Face

We find that capacity planning practitioners face many challenges in the process.

IF8: Most interlocutors see and are discontent with a pervasive under-utilization of their system. This under-utilization can be caused by operational risk minimization taking precedence, newly installed resources not being directly used, and delays in the process.

IF9: We observe the challenge of increasing complexity, both in the managed resources and the tooling needed to correctly monitor them. The heterogeneity of hardware, especially in HPC domains, is also mentioned in accounts of this.

IF10: Some remark that supporting the small to medium workload deployments is much more difficult than planning for the larger deployments. While the larger units each have a larger financial impact, the small units tend to be neglected and found in need of sufficient leftover capacity.

Failure Stories

We summarize the most common failure types below.

IF11: In some cases, the adoption of new technologies can have unforeseen negative consequences. A failure story of the adoption of an (unnamed) new processor architecture ends in an entire rollback of the installation, due to users having difficulties properly utilizing the new hardware, and due to high power consumption.

IF12: Some capacity planning decisions take so long that technological perceptions have changed due to the rapidly changing nature of the field. This leads to hardware choices needing to be implemented that, at the time of actual installation, are considered suboptimal.

IF13: A notorious challenge seems to be the prediction of future ratios between different resource types (mainly number of cores, memory units, storage capacity). We observe a number of failure stories surrounding the misprediction of how different resource type might relate in the future, resulting in significant parts being underutilized or certain capacity dimensions running out of capacity far faster than others. This last consequence can lead to reduced QoS.

IF14: We observe cases where capacity planning is only seen as an afterthought. A representative example is the fast onboarding of a new client where available capacity or time to acquire new capacity is judged too optimistically.

Category 1 - Q5: Typical Process

IF15: The typical processes we see have two shapes: a periodic process, typically centered around the lifecycle of topology resources, and a more ad-hoc process, triggered by less predictable events such as the arrival of new users. The former is dominant in most surveyed scientific clouds, while the latter is more common in commercial clouds. One scientific cloud in the set has a combination of both.

IF16: A sentiment expressed in the majority of interviews is that guessing and human interpretation of monitoring data are a big part of the process. The tooling in the area seems underutilized, further discussed in Q13.

IF17: Next to computational performance of resources, electricity and cooling also play a significant role in the equation. This both impacts and is impacted by the choice of hardware.

Commercial Infrastructures

We observe a difference in the typical process and challenges involved in this process between commercial and scientific infrastructures. Below, we outline the findings for commercial infrastructures.

IF18: We see all interlocutors from commercial backgrounds facing a dilemma between combining purchases and spreading them. The former can lead to significant cost savings but periods of more intense effort for employees, while the latter has the opposite advantages and drawbacks. A possible generalization of this is the competition and interplay between financial and human factors, both impacting the capacity planning process in different ways.

IF19: We also see some interlocutors with commercial cloud backgrounds describing lease and support contracts as being especially important in the set of factors taken into account in a typical capacity planning process. The timing and duration of these conditions can have significant impact on the decision taken.

Scientific Infrastructures

We now summarize the main findings for the typical process scientific infrastructures.

IF20: Most scientific clouds seem to follow a typical public competitive dialogue process for their resource selection. In at least half of the surveyed scientific infrastructures, this includes a benchmark-driven process.

This entails providing hardware contractors with a set of benchmark applications that need to be optimized by the contractor. Results for applications in this benchmark are often weighted by importance or frequency of deployment.

IF21: We observe that most scientific clouds take their biggest users as the main indicator of the needs of a platform. Half of scientific clouds also take into account a broader user feedback survey, reaching users regardless of size or importance.

IF22: Almost all scientific cloud interlocutors perceive their process as fundamentally different from the commercial capacity planning process. The main perceived difference is the mode of operation, which they believe to be budget-centered rather than demand-centered. These interlocutors also consider their budget-centered to have less capacity planning efforts than commercial efforts. Whether this assessment is accurate is difficult to objectively judge, although analysis from other questions seems to indicate that there are more aspects of “traditional” capacity planning in their process than commonly perceived.

Category 1 - Q6: Yes/No Questions

IF23: What-if scenarios do not seem to be established practice currently. Some unstructured examples fitting the format (e.g. fail-over scenarios, clients arriving) are mentioned informally, but not in a structured form.

IF24: Most of the scientific clouds have exploratory projects running where they investigate the possibility of offloading demands to a public cloud. This indicates increased interest in hybrid infrastructure offerings.

IF25: A significant portion of interlocutors is willing to share historical data with the interviewer. This could signal interest in academia and industry for more research being conducted in topics surrounding their capacity planning decision making.

IF26: The majority of interlocutors considers human personnel in some form in the capacity planning process. We further analyze this topic in Q22.

Category 2 - Q7: Stakeholders

IF27: All processes surveyed seem to have an executive board at the head of the process. This board seek out advice from experts in the domains relevant to their decision. These can include technical advisors or scientific experts.

IF28: For all surveyed instances, the final decision seems to be at board level. While the input of domain experts is sought out, the final decision is made by the management.

IF29: The interlocutors with commercial background also have an engineer in charge of the capacity planning process, monitoring the current situation and coordinating the lifecycle-based planning process. The process in this case also includes input from the hardware contract administration for contracts. Occasionally, the sales department is involved, if the decision affects the shaping or pricing of services provided to customers.

IF30: In scientific environments, an important part of the set of stakeholders tend to be scientific partners and (governmental) funding agencies. Half of the surveyed processes here also take into account user input through a user questionnaire.

Category 2 - Q8: Time and Infrastructure Scale

IF31: Unlike the frequency or trigger of capacity planning processes, the time scale of the decisions made seem to be roughly uniform across interlocutors. We observe that the aging and thus deterioration of hardware is seen as the most important factor here, with a mean of 5 years until the next decision for a specific

machine/rack. Commercial environments seem to tend towards faster replacement (3-5 years), while scientific environments seem to replace less quickly (4-7 years).

IF32: The infrastructure scale of decisions for commercial environments tends to be a single rack. Making multi-rack decisions is desired, due to potential cost savings, but not always possible.

IF33: In scientific environments, the infrastructure scale of decisions seems to be larger, with most surveyed infrastructures working on scales of entire cluster/site iterations. One infrastructure works on a smaller scale, making single-machine or single-rack decisions.

Category 2 - Q9: Factors

IF34: The number of factors is remarkable, with more than 25 distinct factors being named in the full set of interviews.

IF35: Nevertheless, the factors that span across a majority of interviews are few. Only three factors are named in more than half of the interviews: the use of historical monitoring data, financial concerns (such as budget size), and the lifetime of hardware. These are followed up by a set of four factors mentioned in slightly less than half of the interviews: user demand, new technologies, incoming projects, and the benchmark performance of different solutions.

IF36: We observe a number of factors particular to scientific infrastructures but not being mentioned in the commercial set. The most important here are the benchmark performance of solutions (which is often required by public competitive acquisition processes) and user demands. One surveyed infrastructure optimizes for throughput here, meaning the number of times the benchmark can be run in a certain time frame.

IF37: Similarly, we observe a number of factors unique to commercial infrastructures. The most prominent are lease contracts, current offerings that the provider has, and personnel capacities.

Category 2 - Q10: Margin for Error

IF38: The margin for error is difficult to objectively measure, due to the multi-faceted nature of this process. Two main consequences of errors are mentioned by interlocutors. First, financial losses can occur due to overestimation of the demand of a certain resource, such as specific accelerators or storage capacity, or due to underestimation, as can happen if the ratio of resource types is mispredicted. Second, personnel can come under pressure, due to available capacity being smaller than expected, starting a search for spare capacity in any of the managed clusters.

Category 2 - Q11: Frequency and Time to Deployment

IF39: While interlocutors from commercial backgrounds report a frequency of at least once per three months (depending on an ad-hoc component), counterparts from scientific infrastructures generally report a frequency upwards of four years. There is one notable exception to this rule, with one of the scientific clouds which takes a decision twice a year. In general, we observe a separation between fast-paced commercial planning cycles and longer cycles in scientific clouds.

IF40: Similar to the frequency of planning events, the time from start of the event to deployment is determined largely by the background of the infrastructure. Commercial clouds tend to finish deployment within 4-5 months, while scientific clouds tend to take 1-1.5 years to deploy. We see a positive correlation with the frequency of planning instances, meaning that a higher frequency trends to be paired with a shorter time to deployment.

IF41: In some scientific clusters, we see a part of the topology containing specialized hardware, such as accelerators, getting a special process with more rapid cycles than the rest of the architecture. This could be due to the faster pace of evolution that these kinds of technologies experience.

Category 2 - Q12: Data Sources

IF42: With the exception of one infrastructure, historic utilization data from monitoring agents is universally reported to be used in the process.

IF43: Next to historic utilization data, we see operational data such as lease contracts and maintenance periods being involved in the process. We also observe some interlocutors explicitly mention taking global market developments into account.

Category 2 - Q13: Tooling

IF44: The main observation here is that none of the surveyed infrastructures have dedicated tooling for the capacity planning of their infrastructures. They use monitoring tools (with dashboards) and/or spreadsheets, combined with human interpretation of the results. Decisions are, in one infrastructure, being preserved in minutes and mails.

IF45: We observe that planners typically consume the data they receive from monitoring in visual formats, in plots over time. Being able to visually investigate and interpret developments plays an important role here.

IF46: The most commonly used tool for monitoring seems to be Grafana, which allows teams to build custom dashboards with the information they see as relevant. NetApp monitoring tools are mentioned as being used by one commercial party. One scientific infrastructure reports basing their results on custom SQL queries of monitoring data. Another scientific infrastructure uses spreadsheets as the primary medium for analysis.

IF47: We identify two key issues being raised explaining the absence of dedicated tooling. First, tools tend to be too platform specific or work only in one layer of the hierarchy and thus return misleading results. We see the issue being the mismatch between the complexity of the reality on the ground and the complexity that these tools assume of the topology. Second, tools tend to have high cost and carry a number of additional features that planners reportedly do not find useful, meaning that the high price is not justified by the value the planners receive out of these tools.

Category 2 - Q14: Errors

IF48: We observe several occurrences of failures being mentioned, although the perceived frequency varies. One interlocutor believes that (slightly) erroneous plans are made constantly, since it is not possible to predict accurately what will be needed in the future, while another interlocutor claims the errors made are not very frequent. On average, the frequency is perceived as low, drawing contrast to the failures being mentioned in the rest of the interview.

IF49: The severity of an error is hard to measure objectively if not actively monitored. The (subjective) descriptions of how severe errors vary from losing potential income, to having underutilized hardware, to hitting storage limits. This raises a different point, surrounding the definition of errors or failures in the field of capacity planning. An underutilized new cluster may be seen as a minor error, since service is typically not affected and the only cost seems to be additional power usage and environmental footprint.

IF50: We did not observe any structured approach to recording errors in the process. Whether they only remain tacit team knowledge or are still recorded somewhere is not clear, although our interpretation indicates the former.

IF51: While most interlocutors seem to describe negative capacity planning incidents as being infrequent and having low severity, the examples being given in response to other questions tend to be from the most recent (if not one of the last) iterations. This is partly explainable with more recent memories being more readily accessible, but also might indicate a more structural underappreciation of the possibility for failures or suboptimal choices in the process.

Category 3 - Q15: Services and Infrastructure Part of Process

IF52: We observe that the majority of interlocutors considers only one type of service as part of their capacity planning process and a minority considers two or more.

IF53: In the commercial settings we survey, we see that VMs holding business-critical workloads are most universally considered as part of the process. One interlocutor mentions a new container platform as also being part of the process, although it is internally approximated as a VM while planning.

IF54: In the scientific settings we survey, we see that batch workloads, HPC workloads, VM workloads, and baremetal hosting services are equally popular. One provider also mentions shared IT services (more general IT functionality) as also being a part of the process.

Category 3 - Q16: Processes for Specific Workloads

IF55: The instances running business critical workloads report two special aspects that they consider for these workloads: special redundancy requirements and live management concerns (primarily migration and offloading).

IF56: We do not observe any special processes being mentioned for Big Data workloads.

IF57: The processes for serverless workloads are still very much in a stage of infancy, as most interlocutors having container or Function as a Service (FaaS) solutions only host them as experimental pilot projects. In terms of capacity planning, one interlocutor points out that for the container platform they currently build they only approximate the containers with VMs in their reasoning. However, they acknowledge that the density and characteristics of this new workload might be very different and that they may need to have special process for this in the future.

IF58: Two of the interlocutors reporting that HPC is a part of their process, state that capacity planning for HPC workloads is even more challenging than for conventional workloads, due to the increased heterogeneity in the hardware platforms needed for this domain.

Category 3 - Q17: Combining Workloads

IF59: All interlocutors, with one exception, consider all workloads combined in one process. The interlocutor forming the exception states that certain different workload types in their cloud are hosted on different infrastructure and separated entirely, with no synchronization occurring between the different efforts.

IF60: A popular approach in scientific infrastructures seems to be to combine workloads through a weighted benchmark suite. This scores topologies by running important representatives from each workload type and combining the scores into a single score.

IF61: One interlocutor with commercial background points out that there is a trade-off between combining processes, thus gaining efficiency but also increasing the risk of failure, and keeping processes separate, thus losing efficiency but also reducing the risk.

Category 3 - Q18: Serverless Workloads

IF62: We observe that, with one exception, all interviews describe introduction of (pilot) serverless programs in their services. One interlocutor sees serverless as a fast growing business, but another interlocutor contrasts this with an observation that the demand for it is still limited.

IF63: Interlocutors see a number of differences with traditional workloads. They observe differing usage patterns with finer granularity of execution units. This leads to higher fluctuation of the load and faster deployment patterns.

IF64: Three of the interviews detail expectations on how their capacity planning will change with serverless workloads becoming more prevalent. They expect impacts on the resource needs, such as the CPU to memory ratio and the allowed overcommission ratio. One also states that guaranteeing workload isolation is likely to become significantly more difficult.

IF65: Currently, none of the interlocutors state having a special subprocess for serverless in their capacity planning approach. They agree, however, that this might need to change in the future, as serverless workloads increase in popularity.

IF66: We observe two key issues hindering the specialization of (parts of) the capacity planning process towards new workloads such as serverless. First, not yet enough information is available on this new workload type and its behavior. This makes reasoning about its capacity needs more difficult, at least with conventional capacity planning methods. Second, interlocutors report a lack of personnel to dedicate to research into effective and efficient hosting of this new workload type.

Category 3 - Q19: What-If Scenarios

IF67: This question was asked infrequently due to time constraints. One interlocutor answered that scenarios they look at indirectly are customer-based scenarios (if a certain customer needs to be hosted) and new hardware releases and acquirements (with new specifications and properties). See also IF101 for requested what-if scenarios in tooling.

Category 4 - Q20: Regulatory Constraints

IF68: We gather a number of laws and standards relevant to the capacity planning process. We conclude that regulatory constraints can definitely play a role in capacity planning.

IF69: Financial institutions tend to have strict standards for the capacity they acquire, such as a guaranteed amount of fail-over capacity. This requires a capacity planning process (and recorded trail of that process) that meets these standards.

IF70: We observe that privacy regulations such as the GDPR are only of limited concern in the capacity planning process. One interlocutor managing a scientific infrastructure states that GDPR only affects the placement and planning of privacy-critical applications on their platform, in the form of preventing public cloud offloads for these specific applications. Another interlocutor mentions the storage of logs could be affected by GDPR, as its introduction leads to less log storage demands and thus less storage capacity being needed for that purpose.

IF71: In scientific infrastructures, we observe the competitive dialogue procedure playing a big role in shaping the process. Publicly funded institutions need to shape their acquirement processes around a public tender with competitive dialogue, which limits how and which hardware components can be selected.

IF72: Security standards can also steer the choice of certain technologies in the capacity planning process. We observe one case where reported exploits in a container platform limit a quick deployment of that technology.

IF73: Special regulations that hold in the country of origin of a certain hardware vendor can also play a role. We hear one example of a supercomputer manufacturer prohibiting its hardware being used by personnel and users having the nationality of a certain set of countries set by the government of the manufacturer.

Category 4 - Q21: Financial and Technical Aspects

IF74: Overall, there are three most frequently cited financial and technical aspects across interviews. The first is funding concerns, looking at the source of funds for future expansions and maintenance. The second consists of special hardware requests from users. The third is the cost of new hardware, in line with global

cost developments on the market.

IF75: For commercial infrastructures, there are a wide range of similarly frequent financial and technical factors that are taken into consideration. Noteworthy are the timing and costs of lease contracts, which receive special attention, and historical sale and usage models for existing services.

IF76: For scientific infrastructures, the size of publicly funded grants is a factor that was mentioned in all interviews. Special hardware requests are the second most frequent factor here, followed up by the total cost of ownership (including electricity and cooling costs) and new hardware developments.

IF77: One interlocutor with commercial background made an observation that we believe resonates with statements by other interlocutors, as well: The variables in the equation that are most relevant are the factors around the hardware, not the cost of the hardware itself. Support contracts, lease contracts, personnel cost – all these factors play a significant role.

IF78: In two interviews, we observe the point being made that their choice of technology is inferior to the financial considerations they make. This underlines the importance of financial aspects in the process.

IF79: One interview raises an interesting relation, between financial investment and the energy consumed by resources that can be acquired with this investment. It observes a trend in which constant investment can lead to increasing energy costs, due to the falling cost of computational resources when compared by energy usage.

Category 4 - Q22: Human Factors

IF80: Overall, we observe two human factors being mentioned most frequently: the need to account for personnel capacity and time to install and set up hardware and software, and the usage patterns that users exhibit when using the infrastructures (each of these is mentioned in 3 interviews).

IF81: Strongly present in the commercial sphere is an awareness of the need for training personnel, especially when switching technologies. In scientific infrastructures, the focus seems to rest more on end-users. Their usage demands and their abilities (and training) are most frequently raised as factors in this category.

IF82: Listening to users and their demands has its limits, however, as one interlocutor points out. They state that if administrators ask users if they would like more computing power, the answer will likely often be “yes”.

IF83: One interlocutor points out that improper capacity planning can lead to stress in the team, because it can lead to short term remedial actions becoming necessary, such as gathering left-over capacity from the entire resource pool.

IF84: One interlocutor observes that personnel does not grow proportionally to the size of the managed resource pool, but that specialization and having specialized staff for certain technologies is the deciding factor. We see this sentiment being shared in many of the interviews.

IF85: We conclude that the human factor plays a significant role in the process, for most surveyed infrastructures. 5 out of the 7 interviews place special emphasis on this. Hiring costs and personnel hours can add up especially, as one interlocutor points out.

Category 4 - Q23: Multi-Datacenter or Local Level

IF86: We observe that the majority of surveyed clouds takes decisions on a multi-datacenter level. The interlocutors that report single-datacenter decision making cite differences in architecture and requirements between different sites as the main cause.

IF87: One interlocutor points out that while the scope of decision making is multi-datacenter, the scope of single decisions still focuses on single racks.

Category 4 - Q24: Production vs. Development

IF88: 3 out of the 7 interviews mention capacity planning differently for development and testing resources than for production resources. For one of these interlocutors, this involves setting aside a dedicated set of testing nodes per cluster. For another, this means splitting workloads into different datacenters. For a third, this involves having lower redundancy on certain development machines.

IF89: We see a set of 2 other interviews claiming that every resource in their topology is considered production, even if it is in fact used as a development or testing machine.

Category 4 - Q25: What-If Scenarios

IF90: This question was asked infrequently due to time constraints. One interlocutor names scenarios where costs are conditioned against certain amounts of cooling, with different cooling types. Another interlocutor points out that their process currently does not contain any what-if scenarios, but believes that they should in the future. This would create a better understanding of possible future outcomes, using factors such as the timing of deployments or new, unknown workload patterns.

Category 5 - Q26: Ideal Process

IF91: An aspect shared by 5 out of the 7 interviews is the call for a more flexible, fast-paced process. Planners across academia and industry believe the current process they follow is not always able to adequately keep up with the latest hardware trends, with special mention of accelerators. Most of them also see another issue arising from this: procured hardware is often idle for a (relatively long) time before it is utilized. An ideal process would address these two issues by adding hardware more flexibly, i.e. in smaller batches. This is not straightforward, due to the economies of scale that sometimes only come into effect at larger batch sizes.

IF92: One interlocutor mentions that their ideal process would also require less time than it currently does. The years of analysis and discussions should be reduced to months.

IF93: We observe two interlocutors mentioning a preference for smaller-scale decision-making, with more attention to detail. One interlocutor mentions this with respect to topologies, having per-rack decisions replace multi-datacenter decisions. Another interlocutor mentions this with respect to the application domain, separating different domains into different sub-processes due to the difficulty of capacity planning large heterogeneous environment.

IF94: One interlocutor expresses the desire for an increased focus in the process on training of users in order to exploit the full potential of new hardware once it arrives.

Tooling

Especially from interlocutors with commercial backgrounds, we hear a wide variety of requests for better tooling for their activities. We list them below, grouped into categories.

IF95: In 3 of the 7 interviews, we observe a demand for capacity planning tools, helping the practitioners rely less on their intuition. A key request is interactivity, with answers wanted within a maximum of two weeks. Getting immediate answers during a meeting would be even better. One interlocutor describes this as an “interactive dashboard”. One interlocutor also states that having the tool answer questions at different levels of accuracy (increasing over time) would also be beneficial, to facilitate quick estimates upfront and more detailed analysis over a longer period, e.g. between two meetings.

IF96: We observe the requirement that tools should be affordable. Interlocutors state that high prices of an existing tooling platform that might help this activity are mainly due to the many other features in that

platform being packaged along with the capacity planning functionality needed.

IF97: Interlocutors also express the need for tools that help in addressing complexities in the process. Being able to track the details of current capacity and being able to predict needed capacity would be a first step in this direction.

IF98: One interlocutor managing a scientific infrastructure points out that any tool should for this activity should support making heterogeneous decisions, which are more difficult to make but are yet still necessary, especially in the academic domain. The request for heterogeneous capabilities is repeated by interlocutors from commercial backgrounds.

IF99: Interlocutors managing commercial infrastructure call for tools that are aware of multi-disciplinary aspects in the process. This includes lifecycle processes (such as aging and maintenance) and lease contracts.

IF100: One interlocutor also expresses the wish for workload trend analysis capabilities in any tool for this activity.

IF101: Two interlocutors list a number of what-if scenarios that they would like to explore with a capacity planning tool. We list the questions underlying these scenarios here, in no particular order.

1. Deciding how much more capacity is needed after certain decisions are taken, given projected CPU usage, memory commission, and overbooking ratios.
2. Seeing the impact different new kinds of workloads have before they become common.
3. Deciding when to buy new hardware.
4. Modeling fail-over scenarios.
5. Deciding whether special user requests can be granted before responding to users.
6. Choosing the best lease duration.
7. Deciding in which cluster to place new workloads.
8. Choosing the best overlap duration between acquiring new hardware and decommissioning old hardware.

Category 5 - Q27: Other Processes Linked With Capacity Planning

IF102: One interlocutor sees no direct link between capacity planning and other processes, although an indirect link is present. Capacity planning, according to the interlocutor, tends to be on the end of the pipeline: only after acquiring new projects is the challenge of finding capacity for these projects considered.

IF103: One interlocutor sees a close relationship between the process and resource management strategies and research. The live management of the infrastructure can have significant impact on the needed capacity, just as the capacity can have consequences for the management strategies that should be employed. Migration and consolidation approaches need special attention here.

Category 5 - Q28: Other Aspects Shared

IF104: One interlocutor states that, no matter what one plans, the future always looks (slightly) differently. This does eliminate the need for planning, but underlines the need to be flexible and plan for unforeseen changes down the road.

IF105: One interlocutor (from a scientific background) points out the difference of speed in capacity planning processes between Europe and the United States. They believe that infrastructures in the U.S. have a faster pace of capacity planning than comparable infrastructures in Europe. They find this disadvantageous,

due to new hardware improvements being slower to arrive.

IF106: Two interlocutors observe that it is easier to obtain grants for a proposal of an infrastructure addressing one coherent need or project. The surveyed scientific infrastructure tends to serve a far more heterogeneous set of use cases, which makes acquiring sufficient funds more difficult.

4.5. Discussion

We now summarize the contributions of this chapter and discuss possible threats to their validity.

4.5.1. Summary

In this chapter, we survey a set of experts with diverse background, using a flexible data collection method. Our analysis of the large amount of raw interview transcript data yields novel results on the capacity planning practice in cloud datacenters. We are the first to map the capacity planning practice in datacenters, as it is experienced first-hand by practitioners. Among many other observations, we find there is a significant amount of guesswork within the capacity planning process, and that a flexible, comprehensive, and realistic tool is lacking.

4.5.2. Threats to Validity

We discuss potential threats to internal validity, construct validity, and external validity.

Internal Validity

A threat to internal validity we see is introduced by the ambiguity and subjectivity of *interpretation of source transcripts*. Confidentiality limits us from sharing these source transcripts. The majority of statements made are however of factual nature, with little room for interpretation. To minimize the possibility of the few, more general statements being misaggregated, the process used is meticulously described and the full findings are objectively presented. We take special care to objectively formulate these findings and refrain from adding personal interpretations, unless specifically marked as such.

Construct Validity

An interview study such as ours might have different results than a study conducted while present at *capacity planning meetings*, accompanying a capacity planner from start to finish. This is a potential threat to construct validity, since what capacity planners state might be different from their actual practice. However, we see several factors limiting this threat. First, we see that many of the observations we make are corroborated by multiple accounts, from interlocutors who were not aware of the identities of the other interlocutors. This reduces the risk of taking incorrect conclusions from the omissions of individual accounts. Second, our script and follow-ups include several questions which approach different parts of the process from different angle, increasing our coverage of the actual real-world practice.

External Validity

The *limited sample size* of our study presents a threat to the external validity of our interview findings (the degree to which results hold outside of the study). In general, this threat is difficult to address, due to the labor-intensive transcription and analysis conducted already in this study: For the existing sample set, we have already spent many months and the data collected alone counts over 35,000 words in size. This massive collection has no parallel in the community thus far. As Table 4.1 shows, our sample of interlocutors is also highly diverse, including multiple countries, infrastructure backgrounds, and roles.

Furthermore, the process of *open* interviews simply has a different methodology [32] than the process of regular, multiple-choice surveys. It is also the only good instrument to collect *new* requirements, which are not all foreseen in advance. Follow-up studies could build on the results of our study by conducting a textual survey with a wider user base, harnessing our exploratory results and requiring less time investment per interlocutor.

5

Design of Capelin: A Capacity Planning System for Cloud Infrastructure

In this chapter, we address the second research question (RQ2). We synthesize requirements for Capelin, design an abstraction for practitioners to formulate capacity problems (addressing RQ2.1), and design a capacity planning approach for cloud infrastructure meeting the requirements (addressing RQ2.2).

5.1. Overview

The capacity planning practice lacks a flexible decision support tool that can reduce the complexity and volume of data coming into the decision process. We propose Capelin, a scenario-based capacity planning system that helps practitioners understand the impact of alternatives. Underpinning this process, we propose as core abstraction the *portfolio of capacity planning scenarios*.

The remainder of this chapter is structured as follows. We synthesize requirements for Capelin in Section 5.2. We then propose a design for the Capelin architecture in Section 5.3. In Section 5.4 we discuss the core abstraction forming the foundation for Capelin’s architecture. Finally, we summarize our contributions of this chapter and discuss threats to validity in Section 5.5.

5.2. Requirements Analysis

In this section, from the findings of Chapter 4, we synthesize the core functional and non-functional requirements addressed by Capelin. This activity falls in stage (1) of the AtLarge design process [61]. Instead of aiming for full automation—a future objective that is likely far off for the field of capacity planning—the emphasis here is on human-in-the-loop decision support [60, P2].

5.2.1. Functional Requirements

(FR1) Model a cloud datacenter environment (see O2, O3, O7): The system should enable the user to model the datacenter topology and virtualized workloads introduced in Section 2.1. Without FR1, Capelin could not answer arbitrary what-if scenario queries (FR3).

(FR2) Enable expression of what-if scenarios (see O2, O10): The system should enable the user to express what-if scenarios concerning workload, topology, and phenomena. For each of these three domains, the system should provide the user with a range of parameters to express the hypothetical system state. The system should then execute the specified what-if scenario(s) and produce and justify a set of user-selected QoS metrics. Without FR2, Capelin could not provide insight into hypothetical future scenarios that later might adversely impact the user’s services.

(FR3) Enable expression of QoS requirements (see O2, O5, O9): The system should enable the user to express QoS requirements, in the form of SLAs, consisting of several SLOs. These requirements are formulated as thresholds or ranges of acceptable values for a set of metrics selected by the user from the metrics offered by the system. Without FR3, Capelin could not differentiate between acceptable and unacceptable scenario states and thus provide less useful suggestions (FR5).

- (FR4) Suggest a portfolio of what-if scenarios** (see O2, O10): The system should suggest a portfolio of what-if scenarios to the user based on the selected workloads, the given topology, and specified QoS requirements. This significantly reduces the user's responsibility to identify the most applicable scenarios. Without FR4, Capelin would rely on the user to be aware of the full capabilities of the scenario modeling capabilities of the tool and to identify the most important scenarios upfront.
- (FR5) Provide and explain a capacity plan** (see O2, O9): The system should provide a capacity plan suggestion, optimizing for minimal capacity within acceptable QoS levels, as specified by FR3. The system should provide a brief explanation and visualize data sources it used to derive the given suggestion. Without FR5, Capelin would not be able to proactively advise the user. If the suggestions were to be provided without explanation, the suggestions could be less likely to be implemented by practitioners.
- (FR6) Model workload behavior** (see O4, O9): The system should be able to model workload behavior, given a limited training set of past behavior. It should recognize patterns and trends in this behavior at different timescales. As an outcome, Capelin should also support synthesizing derivative workloads and forecast future workload behavior. Without FR6, the queries that users could submit to the system would be limited to past, already recorded workloads, severely limiting the forward-looking activity of capacity planning.
- (FR7) Provide a library of reusable components** (see O8, O9): The system should provide a prebuilt library of cloud components, topologies, and cloud scenarios to the user. This enables users to quickly query the system on demand, without having to spend large manual effort upfront. This library should consist of commonly used items that can be configured to reasonable extent and be combined with each other and manually added items. Without FR7, Capelin is less likely to be used frequently, since the barrier to creating scenarios and getting capacity plan suggestions could otherwise be too high.

5.2.2. Non-Functional Requirements

- (NFR1) Provide fast initial coarse estimate and precise same-day estimate** (see O1): The system should generally be prompt in answering what-if queries (FR2) and providing capacity planning suggestions (FR6) to the user. However, such speed of response would come at a cost of accuracy. Therefore, the system should provide a initial coarse estimate within seconds to (at most) minutes, with a self-assessed degree of confidence in this estimate. The system should then provide a same-day precise estimate with increased degree of confidence. Without NFR1, Capelin could not easily be used in dynamic meeting-room settings, where rapid results and interactivity are crucial aspects of a decision support system.
- (NFR2) Intuitive what-if portfolio design and analysis inspection** (see O8, O9, O10): The interface should be designed in a way that facilitates intuitive design of scenario portfolios and scenario analysis. As a decision support system, this layer of human-computer interaction is important to optimize. Without NFR2, Capelin is less likely to be used by practitioners, limiting its impact in capacity planning decision processes.

5.2.3. Evolution of Requirements

The requirement analysis process we follow is iterative [61], meaning that our requirements have evolved with our understanding of the problem. We illustrate this here by giving examples of how and why we adapted key requirements over time.

For requirement FR2 (expression of what-if scenarios), we first only considered workload and topology as the ingredients of a what-if scenario. While these are crucial ingredients, interviews (see O4 in §4.3) and internal discussions show that the capacity planning problem is intricately linked with far more aspects of a datacenter, most importantly its dynamic operation. To capture this dynamic component and the phenomena that arise here, we add the notion of phenomena to each what-if scenario. This allows us to model the current situation and candidate scenarios with increased realism.

Requirement FR6 (modeling workload behavior) first exclusively addressed the forecasting of future workload behavior based on detected historical patterns. We realized that workload behavior can be modeled and synthesized in more ways, such as synthesis of different workloads or manual specification of load levels. Restricting the scope to forecasting also requires a firm foundation of historical data. Such data is not always readily available, especially for emerging workload types and changing business requirements. These considerations led us to broaden the scope of this functional requirement to these other mechanisms for workload modeling and synthesis.

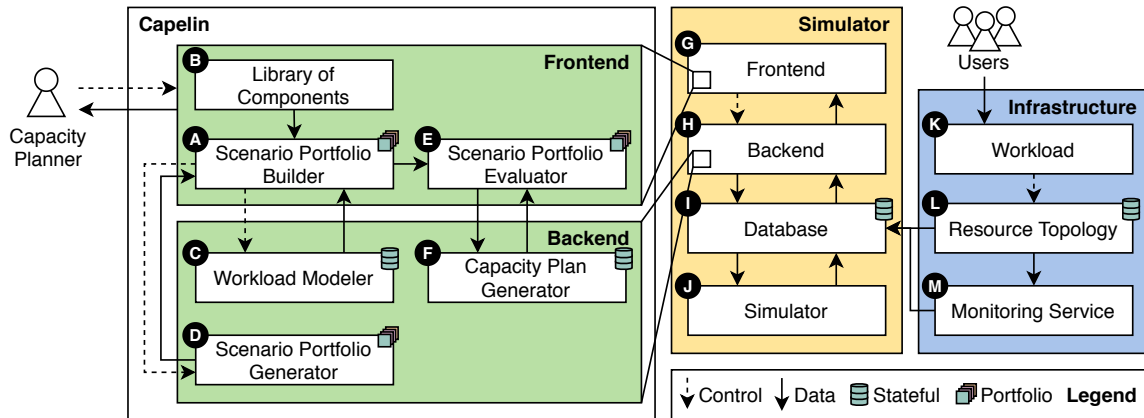


Figure 5.1: An overview of the architecture of Capelin. Capelin is provided information on the current state of the infrastructure and assists the capacity planner in making capacity planning decisions. Labels indicate the order of traversal by the capacity planner (e.g., the first step is to use component **A**, the scenario portfolio builder).

Non-functional requirement NFR1 (fast initial coarse estimate response) first only specified response time for FR6 (capacity plan suggestion). Automatic capacity plan suggestion is indeed a time-intensive operation and merits this constraint. However, while reflecting the significant efforts undertaken in Chapter 6 to reduce the runtime of the simulator, we realize that FR2 (answering what-if queries) also requires a prompt (if less accurate) response. If Capelin is to be used in meetings as an interactive question-answering tool, it needs to be quick in responding, even if its outputs may not yet be completely final. For this reason, we add FR2 to the list of functionality that needs a fast initial response.

5.3. Overview of the Capelin Architecture

Figure 5.1 depicts an overview of the Capelin architecture. Capelin extends OpenDC, an open-source, discrete event simulator with multiple years of development and operation [59]. This architecture is designed iteratively, following the AtLarge design process [61]. Stages (3) (bootstrapping the creative process) and (4) (high-level and low-level design) of that process are particularly applicable, but (2) (understanding alternatives) is also involved in the deliberations made.

We now discuss each main component of the Capelin architecture, taking the perspective of a capacity planner. We outline the abstraction underpinning this architecture, the capacity planning portfolios, in Section 5.4.

5.3.1. The Capelin Process

The frontend and backend of Capelin are embedded in OpenDC. This enables Capelin to leverage the simulator’s existing platform for datacenter modeling and allows for inter-operability with other tools as they become part of the simulator’s ecosystem. Making Capelin a standalone tool may give greater independence of design and evolution, unconstrained by the parent platform, but would require greater effort to continue evolving with the state-of-the-art in datacenter modeling.

The capacity planning practitioner interacts with the frontend of Capelin, starting with the *Scenario Portfolio Builder* (component **A** in Figure 5.1), addressing FR2 and FR3. This component allows the user to visually specify portfolios of scenarios, setting workload, topology, and phenomena for each scenario. The first scenario specified in each portfolio is considered to be the base scenario, forming the baseline for all other scenarios. The builder also makes the construction of SLO constraints on a portfolio-level possible (FR3), enabling users to compare scenarios on real-world terms fitting the targets existing within their infrastructure.

The aforementioned scenarios can be composed using pre-built components from the *Library of Components* (**B**), addressing FR7. This library contains workload, topology, and operational building blocks, facilitating fast and intuitive composition of scenarios. Such a library is pre-populated by the system with a set of industry-standard components (using for example the Open Compute Project¹ as starting point), but can be augmented by the user with platform-specific components. The choice to also allow user-populated

¹<https://www.opencompute.org/>

components benefits ease-of-use: If practitioners are to include this in their frequent practice, any obstacles to quick interactive querying should be minimized.

If the (human) planner wants to modify historical workload behavior or anticipate future trends, the *Workload Modeler* (C) can model workload traces and can synthesize custom loads (FR6). This component provides a set of pre-defined operations on existing workloads that allow for straight-forward workload modeling with little knowledge needed about the underlying mechanisms.

The planner might also not always be aware of the full range of possible scenarios during the planning phase. The *Scenario Portfolio Generator* (D) addresses this by suggesting customized scenarios extending the given base-scenario (FR4, NFR2). It augments the builder with suggested, common scenarios relevant to that situation. The generation and selection of scenarios in this component can be informed by the base scenario exclusively, or even the entire set of existing scenarios in the portfolio. A set of fixed suggestions (e.g., large-scale failures, workload stress tests) combined with custom, tailored suggestions (e.g., topology expansions, workload fluctuations) can provide a solid base of reliable but also relevant suggestions.

The portfolios built in the builder can be explored and evaluated in the *Scenario Portfolio Evaluator* (E). Graphical overviews of key selected metrics across scenarios, augmented with SLOs markers, give users quick access to the outcomes of the simulation of built scenarios. Possible automated curation of the full results, showing the most relevant or interesting results first, could further reduce the time needed for practitioners to gather the desired insights from their overview. A key focus here is to refrain from overwhelming the user with the deluge of data that capacity planners already face in the conventional capacity planning process, but to visualize and curate a custom overview that reduces the presented information to crucial insights. Not properly ensuring this can actually lead to an *increase* in information the capacity planner has to deal with, possibly with adverse impact.

Finally, based on the results from this evaluation, the *Capacity Plan Generator* (F) suggests plans to the planner (FR5). Such suggested plans primarily consist of a topology meeting SLOs and not dominated by other topologies on the spectrum of performance vs. cost-of-ownership. To constrain the domain that a design space search approach would need to explore here, the scenarios in the selected portfolio could be taken as boundary points in terms of the allowed cost. Between these boundaries, the capacity plan generator could search for solutions meeting requirements while minimizing cost (both upfront and in operation).

In earlier designs, this last component was combined with the scenario portfolio generator, since at first glance, both provide similar functionality (generation of capacity plans). One might even consider the capacity plan generator to address a subset of the functionality of the scenario portfolio generator, since it “only” generates candidate topologies. However, there are notable differences that motivated splitting the components into two. First, the aims differ: the scenario portfolio generator aims to provide large coverage of scenarios, while the capacity plan generator aims to find the best scenario(s) in the design space. Second, the input data differs: The scenario portfolio generator takes in only the existing portfolio definition, while the capacity plan generator also considers the performance that different scenarios have exhibited in simulation. Given these two considerations, the subset relation is no longer satisfied and their responsibilities differ, leading to the decision to split the two into their own components.

5.3.2. The OpenDC Simulator

OpenDC is the simulation platform backing Capelin, enabling the capacity planner to model (FR1) and experiment (FR2) with the cloud infrastructure, interactively. The software stack of this platform is composed of a web app frontend, a web server backend, a database, and a discrete-event simulator. This kind of simulator offers a good trade-off between accuracy and *performance*, especially at the scale of datacenters and long-term workloads. In contrast, cycle-accurate emulation or real-time simulators would be too slow.

The *Frontend* (G) serves as the user portal through which stakeholders of an infrastructure can interact with its models and experiments. The *Backend* (H) responds to frontend requests, acting as intermediary and business-logic between frontend, and database and simulator. The *Database* (I) manages the state, including topology models, historical data, simulation configurations, and simulation results. It receives inputs from the real-world topology and monitoring service, in the form of workload traces. The *Simulator* (J) evaluates the configurations stored in the database and reports the simulation results back to the database.

5.3.3. Infrastructure

The cloud infrastructure is at the foundation of this architecture, forming the system to be managed and planned. We consider three components within this infrastructure: The *workload* (K) submitted by users (e.g., VMs), the (logical or physical) *resource topology* (L), and a *monitoring service* (M). The infrastructure

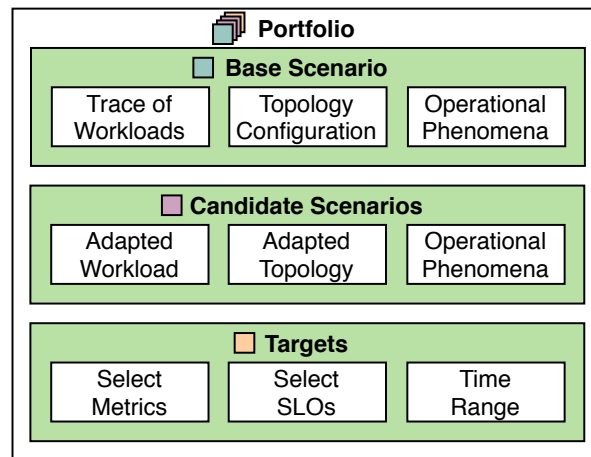


Figure 5.2: Abstraction of a capacity planning portfolio, consisting of a base scenario, a number of candidate scenarios, and comparison targets.

as a whole follows the industry-standard system model described in Section 2.1.

Monitoring services in particular can be provided by a wide range of existing vendor products. VMware’s vSphere tool can provide monitoring insights for the logical and physical layers of the topology². While this is a tool operating both on physical and logical layers, some tools also focus only on one: IBM’s cloud infrastructure monitoring and reporting tool³ reports on the physical health of monitored machines. Our system can be fed many of the output metrics of these different tools, as demonstrated in Chapter 6.

5.4. Portfolio Abstraction for Capacity Planning

In this section, we propose a new abstraction, which organizes multiple scenarios into a *portfolio* (see Figure 5.2). Each portfolio includes a base scenario, a set of candidate scenarios given by the user and/or suggested by Capelin, and a set of targets to compare scenarios. In contrast, most capacity planning approaches in published literature are tailored towards a *single* scenario—a single potential hardware expansion, a single workload type, one type of service quality metrics. We believe that this approach does not adequately cover the complexities that capacity planners currently face. The multi-disciplinary and multi-dimensional nature of capacity planning call for a novel approach, based on *multiple* scenarios.

5.4.1. Scenarios

A scenario represents a point in the capacity planning (datacenter design) space to explore. It consists of a combination of workload, topology, and a set of *operational phenomena*. Phenomena can include correlated failures [41], performance variability [74, 127], security breaches [29], etc. Considering such phenomena allows scenarios to more accurately capture the real-world operations and hardships related to them. Such phenomena are often hard to predict intuitively during capacity planning, due to emergent behavior that can arise at scale.

The baseline for comparison in a portfolio is the *base scenario*. It represents the status quo of the infrastructure or, when planning infrastructure from scratch, it consists of very simple base workloads and topologies. The other scenarios in a portfolio, called *candidate scenarios*, represent changes to the configuration that the capacity planner could be interested in. Dividing scenarios into these two categories ensures that any comparative insights that the tool provides are meaningful within the context of the current architecture. It does, however, also require the capacity planner to always specify such a base scenario, even in the case of an infrastructure project starting from scratch. Nevertheless, even a simple baseline can serve as a good comparative baseline for other scenarios, and the benefits of assuming such a baseline outweigh the potential drawbacks of always needing to specify a certain base case.

We define changes to configurations to be effected in one of the following four dimensions:

²<https://docs.vmware.com/en/VMware-vSphere/7.0/com.vmware.vsphere.monitoring.doc/GUID-A8B06BE0-E5FC-435C-B12F-A31618B21E2C.html>

³<https://www.ibm.com/nl-en/cloud/infrastructure/monitoring>

- (1) *Variety*: qualitative changes to the workload or topology (e.g., different arrival patterns, or resources with more capacity)
- (2) *Volume*: quantitative changes to the workload or topology (e.g., more workloads or more resources)
- (3) *Velocity*: speed-related changes to workload or topology (e.g., faster resources)
- (4) *Vicissitude*: combining (1)–(3) over time

This approach to derive candidate scenarios is *systematic* and addresses the core need of FR2: a structured way of specifying what-if scenarios. Although abstract, it allows approaching many of the practical problems discussed by capacity planners. For example, an ongoing discussion is horizontal scaling (scale-out) vs. vertical (scale-up) [117]. Horizontal scaling, which is done by adding clusters and commodity machines, contrasts to vertical scaling, which is done by acquiring more expensive, “beefy” machines. Horizontal scaling is typically cheaper for the same performance, and offers a broader failure-target (except for cluster-level failures). Yet, vertical scaling could lower operational costs, due to fewer per-machine licenses, fewer switch-ports for networking, and smaller floor-space due to fewer racks. Experiment 6.3.5 explores this dichotomy.

One particular form of candidate scenario that Capelin could generate and thus help practitioners explore and understand is an *adversarial scenario*. Adversarial incidents such as sudden significant spikes in resource demand, catastrophic cascading failures, and purposefully orchestrated usage patterns to fool automated anomaly detection mechanisms [34] can severely impact datacenter operations. Adding automatically generated adversarial scenarios to Capelin’s Scenario Portfolio Generator could help practitioners prepare for these cases, which do happen in practice [136].

5.4.2. Targets

A portfolio also has a set of targets that prescribe on what grounds the different scenarios should be compared. Targets include the metrics that the practitioner is interested in and their desired granularity, along with relevant SLOs [94]. Following the taxonomy defined by the performance organization SPEC [51], we support both *system-provider metrics* (such as operational risk and resource utilization) and *organization metrics* (such as SLO violation rates and performance variability). This addresses requirement FR3 by enabling the user express QoS targets and ultimately helps in addressing FR5 (providing a meaningful capacity plan).

The targets also include a time range over which these metrics should be recorded and compared. This time range is at least as long as the base workload trace requires and can extend beyond it to include future workload behaviour predictions.

To address NFR1, the granularity of targeted metrics should also allow for a fast initial coarse estimate and a more precise later estimate. This could be offered through of a series of runs, each increasing the accuracy of and confidence in the results.

5.5. Discussion

We now summarize the contributions of this chapter and discuss possible threats to their validity.

5.5.1. Summary

We propose a novel capacity planning decision support system that answers RQ2 by addressing key findings from our community interview study. Capelin is embedded in the OpenDC platform, leveraging its existing feature set and extending it in key areas. Our tool is based on a new abstraction for capacity planning: portfolios of scenarios. Combined with the auto-generating capabilities of Capelin, this abstraction allows for structured discourse and evaluation of capacity decisions.

5.5.2. Threats to Validity

We discuss potential threats to internal validity, construct validity, and external validity.

Internal Validity

A key decision early in the design process is to focus on what-if scenarios as the main abstraction for this process. While this claim is supported by evidence gathered from several interviews (discussed in Section 4.3, O10), this choice is still a potential threat to internal validity. We claim that a what-if scenario structure can help practitioners, but this claim needs to be verified in Chapter 6, in discussion with stakeholders.

Construct Validity

A potential threat to construct validity is whether the designed tool addresses the requirements. To minimize this threat, we link back from each feature to the requirement that inspired it and explain how the feature or component addresses that requirement. Nevertheless, full validation of a design with respect to its requirements is difficult without an implementation, which will be discussed in Chapter 6.

External Validity

The main threat to the external validity of this system design is whether an implementation of it can successfully be applied in context. We demonstrate an implementation in Chapter 6 and gather initial feedback from key stakeholders. This provides evidence towards a broader, external applicability of the design.

6

Evaluation of Capelin, through Experiments with a Real-World Prototype

In this chapter, we evaluate Capelin in real-world scenarios, addressing research question RQ3. We present our prototype implementation of Capelin (RQ3.1) and evaluate it with a set of trace-based experiments (RQ3.2).

6.1. Overview

We implement a working prototype of Capelin, realising key features of the design. This process begins with extending OpenDC’s modeling capabilities to cover infrastructures and workloads common in the state-of-the-art. We then discuss the implementation of dedicated capacity planning functionality, presenting designed algorithms and policies as well as interaction models. We describe the details of our prototype in Section 6.2. Given the implemented prototype, we explore how Capelin can be used to answer capacity planning questions. We conduct extensive experiments using Capelin and data derived from operational traces collected long-term from private and public cloud datacenters. Our experiments explore key trade-off portfolios common to the capacity planning domain, exploring alternative topologies, diverse workloads, and emerging operational phenomena. We present our experimental design and results in Section 6.3. Finally, we summarize our contributions and discuss their validity in Section 6.4.

Our implementation has resulted in a working prototype, tested with real-world scenarios and workloads. It features a low-barrier user interface and provides reproducible results. The tool can also be used in live meetings, due to its high performance simulation capabilities providing answers to practitioners in minutes even in realistic, complex situations. We publish Capelin as FOSS, for the community to use. Discussions with partners in academia and industry are promising.

Our experiments show interesting findings, supporting our claim for a need for data-based capacity planning and the validity of our design. The results of our experiments are externally validated. From the computational power demands and time requirements needed to run our experiments, we conclude that these simulation-based experiments are much more environment-friendly than real-world experiments, differing by several orders of magnitude in time and resource cost.

6.2. Implementation of a Software Prototype

In this section, we describe our implementation of a working prototype for Capelin. We begin with presenting an overview of extensions made to the existing simulation model of OpenDC (§6.2.1). We address different aspects of this model in subsequent sections, discussing workload (§6.2.2), topology (§6.2.3), allocation policies (§6.2.4), operational phenomena (§6.2.5), and metrics (§6.2.6). We then discuss the implementation of Capelin’s capacity planning functionality (§6.2.7). Finally, we describe the public release of our software artifacts for Capelin and refer to efforts designed and used to test their validity.

6.2.1. Extensions to the OpenDC Model

We extend the open-source OpenDC simulation platform [59] with capabilities for modeling and simulating the *virtualized workloads* prevalent in modern clouds. We model the CPU and memory usage of each VM

Algorithm 1 Sampling procedure for selecting VMs in a trace based on a desired total load fraction.

```

1: procedure SAMPLETRACE( $vms, fraction, totalLoad$ )
2:    $selected \leftarrow \emptyset$  ▷ The set of selected VMs
3:    $load \leftarrow 0$  ▷ Current total load (FLOP)
4:   while  $|vms| > 0$  do
5:      $vm \leftarrow$  Randomly removed element from  $vms$ 
6:      $vmLoad \leftarrow$  Total load of  $vm$ 
7:     if  $\frac{load+vmLoad}{totalLoad} > fraction$  then
8:       return  $selected$ 
9:     end if
10:     $load \leftarrow load + vmLoad$ 
11:     $selected \leftarrow selected \cup \{vm\}$ 
12:  end while
13:  return  $selected$ 
14: end procedure

```

Algorithm 2 Sampling procedure for combining different traces A and B , at the total load of A .

```

1: procedure SAMPLEMULTIPLETRACES( $vmsA, fractionA, vmsB, fractionB$ )
2:   Ensure the traces of  $vmsA$  and  $vmsA$  have the same duration
3:    $totalLoad \leftarrow$  Total CPU load of the trace  $A$ 
4:    $vmsASelected \leftarrow$  SAMPLETRACE( $vmsA, fractionA, totalLoad$ )
5:    $vmsBSelected \leftarrow$  SAMPLETRACE( $vmsB, fractionB, totalLoad$ )
6:   return  $vmsASelected \cup vmsBSelected$ 
7: end procedure

```

along with hypervisors deployed on each managed node. Each hypervisor implements a fair-share scheduling model for VMs, granting each VM at least a fair share of the available CPU capacity, but also allowing them to claim idle capacity of other VMs. The scheduler permits *overprovisioning of CPU resources, but not of memory resources*, as is common in industry practice. We also model a workload and resource manager that controls the deployed hypervisors and decides based on configurable allocation policies to which hypervisor to allocate a submitted VM.

6.2.2. Workload Modeling

The number of virtual workload traces that are available to the general public are limited. Modeling available workloads allows us to synthesize different workloads and greatly increase the set of workloads available to experimentation. It can even allow practitioners to create workloads from existing data that may resemble future workloads of a cloud, to investigate possible future scenarios.

Modeling and synthesizing such workloads is typically a task reserved for experts, involving manual effort and domain knowledge. Capelin features functionality for practitioners and novices alike to take existing traces and synthesize new traces from them with minimal effort. By pre-defining a set of strategies for workload modeling that can be generally applied, Capelin makes the process simpler and much more accessible.

One such strategy that we support is to sample VMs from existing traces at different target loads, resulting in workloads with a fraction of the original load. To *sample*, Capelin randomly takes VMs from the full trace and adds their entire load, until the resulting workload has enough load. We illustrate this in pseudocode, in Algorithm 1.

We also implement a strategy for two traces to be combined. This allows practitioners to examine how different workloads would interact when executed by the same datacenter. Two traces with equal duration, denoted A and B here, are combined by sampling each set of VMs at given fractions of their load, using the previously described sampling procedure. The total load of the trace A is used as the total load to which the two fractions are applied. The full procedure is listed in Algorithm 2.

Finally, we allow practitioners to manipulate the composition of a workload consisting of multiple different workload types. Since workload traces are scarce, up- or down-sampling certain VM types in a trace can help investigate what the impact of more or less VMs of a certain type would be on certain metrics. We implement this for VM traces containing a certain set of VMs hosting HPC workloads. We facilitate two forms

Algorithm 3 Sampling procedure for increasing or reducing the fraction of HPC VMs in a trace consisting of both HPC and non-HPC VMs.

```

1: procedure SAMPLEHPCINTRACE(vmsHPC, vmsNonHPC, hpcFraction, sampleOnLoad)
2:   totalLoad  $\leftarrow$  total original load of entire trace
3:   if sampleOnLoad then ▷ Sample on load
4:     load  $\leftarrow$  0 ▷ Current total load (FLOP)
5:     selected  $\leftarrow$   $\emptyset$  ▷ The set of selected VMs
6:     while load < totalLoad * hpcFraction do
7:       vm  $\leftarrow$  Randomly drawn from vmsHPC ▷ Without rep., until all drawn, then with rep.
8:       vmLoad  $\leftarrow$  Total load of vm
9:       if  $\frac{\textit{load} + \textit{vmLoad}}{\textit{totalLoad} * \textit{hpcFraction}} > 1$  then
10:        break
11:       end if
12:       load  $\leftarrow$  load + vmLoad
13:       selected  $\leftarrow$  selected  $\cup$  {vm}
14:     end while
15:     while load < 1 do
16:       vm  $\leftarrow$  Randomly drawn from vmsNonHPC ▷ Without rep., until all drawn, then with rep.
17:       vmLoad  $\leftarrow$  Total load of vm
18:       if  $\frac{\textit{load} + \textit{vmLoad}}{\textit{totalLoad}} > 1$  then
19:        return selected
20:       end if
21:       load  $\leftarrow$  load + vmLoad
22:       selected  $\leftarrow$  selected  $\cup$  {vm}
23:     end while
24:     return selected
25:   else ▷ Sample on VM count
26:     totalNumVMs  $\leftarrow$  |vmsHPC| + |vmsNonHPC|
27:     vmsHPCSampled  $\leftarrow$  (hpcFraction * totalNumVMs) of elements taken from vmsHPC
28:     vmsNonHPCSampled  $\leftarrow$  ((1 - hpcFraction) * totalNumVMs) of elements taken from vmsNonHPC
29:     return vmsHPCSampled  $\cup$  vmsNonHPCSampled
30:   end if
31: end procedure

```

of sampling: by load and by count. To sample by load means to sample for a certain total load, regardless of the number of VMs (as is done in the previous sampling methods). To sample by count means to sample for a certain VM count, regardless of the total load this carries with it. We illustrate this process in pseudocode, in Algorithm 3.

6.2.3. Topology Definition

OpenDC enables practitioners to define a datacenter topology in an interactive, visual way. Such a topology consists of clusters, each consisting of multiple racks, which in turn consist of a number of machines. Each machine has a set of computational units (CPUs and GPUs) as well as memory and storage units. For each of these units, detailed specifications are provided. The user interface makes the definition of such topologies accessible to users of many different skill levels, from the enthusiast to the skilled datacenter engineer.

Before Capelin's extensions, the simulator only allowed for one topology to be defined per project. Topologies also could also only be defined through user interface interactions. With Capelin, we extend OpenDC with the capability to define multiple different topologies within the same project. This allows for comparative studies with multiple topologies. Figure 6.1 features the visual definition of multiple topologies, in OpenDC with Capelin's extensions.

We also enable the textual definition of topologies. To streamline the definition of topologies, we redefine the way topologies are specified and stored in the platform. We denormalize topology structures towards single-object topologies in v2.x, meaning that one topology is now represented as one object in the database, as opposed to a linked tree of many smaller objects spread across tables. The base format we use here is JSON, an industry-standard for text encoding of objects. This choice of format allows for textual definition

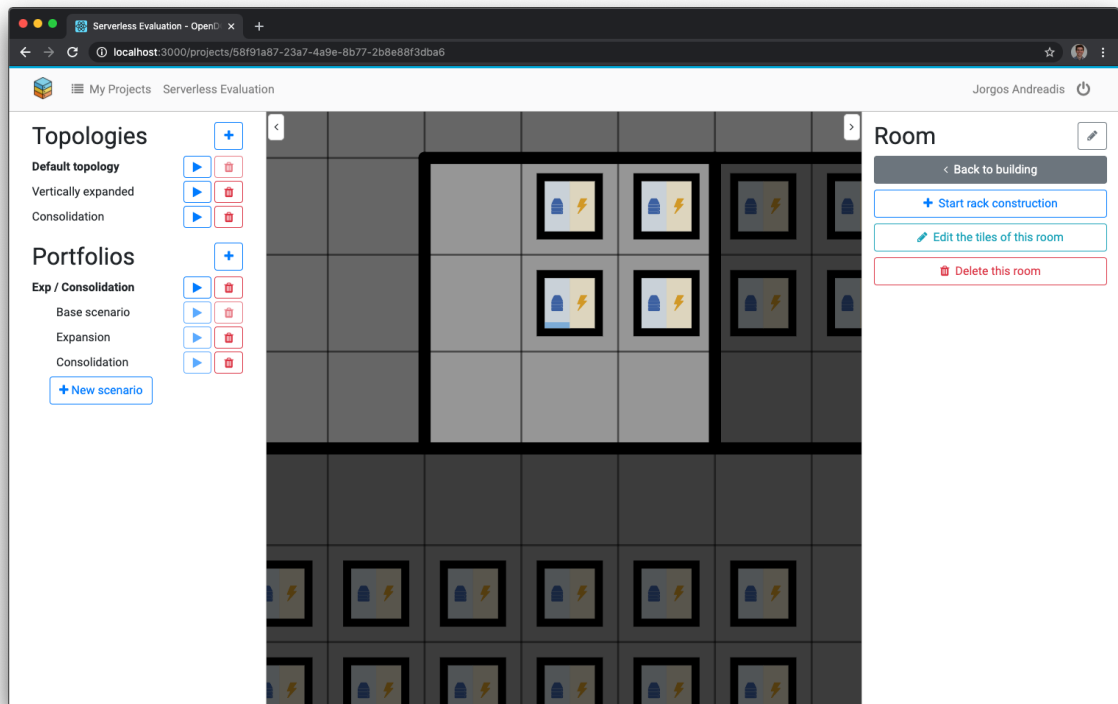


Figure 6.1: Definition of multiple topologies in the OpenDC interface, as extended by Capelin.

of topologies and straightforward duplication of already existing topologies (such as “export to clipboard” functionality), reducing the barrier to sharing and adapting topologies. This leads to significant gains in portability and flexibility, both for Capelin and the simulator platform as a whole.

6.2.4. Allocation Policies

The allocation policy is a key component of live infrastructure management. When a new VM is submitted, a machine needs to be selected to allocate this VM to. Capelin supports several policies that describe how they should be placed:

- (1) The `mem` policy: prioritizing by available memory, meaning that the resource with most available memory is chosen.
- (2) The `core-mem` policy: prioritizing by available memory per CPU core, ranking all resources by this ratio.
- (3) The `active-servers` policy: prioritizing by number of active VMs already on this machine.
- (4) The `replay` policy: prioritizing by mimicking the original placement strategy of a trace, if this placement data is available.
- (5) The `random` policy: randomly placing VMs on hosts (with the randomness seeded for reproducibility).

For each of the comparative policies (`mem`, `core-mem`, and `active-servers`) we use two variants. The Worst-Fit variant selects the resource with the *most* available resource of that policy. The Best-Fit variant is the inverse of this previous variant, thus selecting the *least* available resource, labeled as `-inv`.

6.2.5. Operational Phenomena

Each capacity planning scenario can include operational phenomena. In these experiments, we consider two such phenomena, (1) performance variability caused by performance interference between collocated VMs, and (2) correlated cluster failures. Both are enabled, unless otherwise mentioned.

Table 6.1: Parameters for the lognormal failure model we use in experiments. We use the normal logarithm of each value.

Parameter [Unit]	Scale	Shape
Inter-arrival time [hour]	24×7	2.801
Duration [minute]	60	60×8
Group size [machine-count]	2	1

We assume a common model [74, 127] of performance interference, with a *score* from 0 to 1 for a given set of collocated workloads, with 0 indicating full interference between VMs contending for the same CPU, and 1 indicating non-interfering VMs. We derive the value from the *CPU Ready* fraction of a VM time-slice: the fraction of time a VM is ready to use the CPU but is not able to, due to other VMs occupying it. We mine the placement data of all VMs running on the base topology and collect the set of collocated workloads along with their mean score, defined as the mean CPU ready time fraction subtracted from 1, conditioned by the total host CPU load at that time, rounded to one decimal. At simulation time, this score is then activated if a VMs is collocated with at least one of the others in the recorded set and the total load level on the system is at least the recorded load. The score is then applied to each collocated VMs with probability $1/N$, where N is the number of collocated VMs, by multiplying its requested CPU cycles with the score and granting it this (potentially lower) amount of CPU time.

The second phenomenon we model are cluster failures, which are based on a common model for space-correlated failures [41] where a failure may trigger more failures within a short time span; these failures form a *group*. We consider in this work only hardware failures that crash machines (full-stop failures), with subsequent recovery after some duration. We use a lognormal model with parameters for failure inter-arrival time, group size, and duration, as listed in Table 6.1. The failure duration is further restricted by a minimum of 15 minutes, since faster recoveries and reboots at the physical level are rare. The choice of parameter values is inspired by GRID'5000 [41] (public trace also available [66]) and Microsoft Philly [67], scaled to a commercial mid-tier cloud's topology, in discussion with experts.

6.2.6. Metrics

We would like to ensure that practitioners get a complete picture of the datacenter. This means having many, diverse perspectives and views on the same issue. Doing so allows practitioners to inspect involved trade-offs and phenomena, which are often multi-metric and require context provided by many different metrics to be visible [54]. To address this, Capelin supports a wide array of metrics, listed below.

- (1) The *total requested CPU cycles* (in MFLOPs) of all VMs. A VM requests a certain amount of cycles during each time slice. This metric is the summation of all of these requests over all slices, for all VMs.
- (2) The *total granted CPU cycles* (in MFLOPs) of all VMs. Sometimes, when requested computational capacity exceeds existing capacity, not all requested cycles can be granted. This metric is the summation of all actually granted cycles over all slices, for all VMs.
- (3) The *total overcommitted CPU cycles* (in MFLOPs) of all VMs, defined as the sum of CPU cycles that were requested but not granted. This difference is summed over all slices of all VMs. This metric serves as an indicator of the CPU wait time a VM would experience when run in a real-world setting: higher overcommitment means that VMs have to wait longer to be serviced on the CPU.
- (4) The *total interfered CPU cycles* (in MFLOPs) of all VMs, defined as the sum of CPU cycles that were requested but could not be granted due to performance interference. This metric will always be lower or equal than the *overcommitted* cycles, since interference is one of the factors contributing to overcommitted cycles.
- (5) The *total power consumption* (in Wh) of all machines. We use a linear model based on machine load [18], with an idle baseline of 200 W and a maximum power draw of 350 W. This metric represents the sum of recorded power consumption per cycle, multiplied by the time over which it is summed, summed over all machines.

<pre>Portfolio: type: object properties: _id: type: string projectId: type: string name: type: string scenarioIds: type: array items: type: string targets: type: object properties: enabledMetrics: type: array items: type: string repeatsPerScenario: type: integer</pre>	<pre>Scenario: type: object properties: _id: type: string portfolioId: type: string name: type: string simulation: type: object properties: state: type: string results: type: object trace: type: object properties: traceId: type: string loadSamplingFraction: type: number topology: type: object properties: topologyId: type: string operational: type: object properties: failuresEnabled: type: boolean performanceInterferenceEnabled: type: boolean schedulerName: type: string</pre>
--	---

Figure 6.2: JSON schema of the portfolio and scenario data types, listed in the YAML format for legibility.

- (6) The *number of time slices a VM is in a failed state*, summed across all VMs. When a host fails, we count the number of slices it fails and multiply this by the VM count on that host, at that time. This metric is the sum of these counts over all hosts.
- (7) The *mean CPU usage* (in MHz), defined as the mean number of *granted* cycles per second per machine, averaged across machines. This represents the mean utilization per machine, over time.
- (8) The *mean CPU demand* (in MHz), defined as the mean number of *requested* cycles per second per machine, averaged across machines. Compared to *mean CPU usage*, this metric represents requested load, not granted load.
- (9) The *mean number of deployed VM images* per host. At each slice, we report the number of VMs on each host. This metric is the mean of these reported counts.
- (10) The *maximum number of deployed VM images* per host. We compute the maximum VM count per host, and take the overall maximum over all machines, over all slices.
- (11) The *total number of submitted VMs*. This is the number of incoming VMs from the trace entering the system. It serves as a validation check, ensuring all VMs are deployed.
- (12) The *maximum number of queued VMs* in the system at any point in time. Before a VM can be deployed, it briefly is placed in a queue. This metric reports the maximum length of this queue over the entire duration of the simulation.
- (13) The *total number of finished VMs*. This metric serves as a validation check and should match the *submitted VMs* metric, since all deployed VMs from the trace should also finish.

- (14) The *total number of failed VMs*. This is the number of VMs that did not manage to finish. This metric serves as a validation check and should not exceed 0.

While no explicit SLO metrics are present in this list, we have two proxies (both Service Level Indicators (SLIs)) from which we can infer violations of such higher-level metrics. Incidents such as overcommitted CPU cycles and failed VM slices directly affect the availability of the service. High prevalence of these incidents in a short time-span even cause unavailability of services, violating SLOs and thus propagating into SLO metric that could be formulated here.

6.2.7. Capelin Functionality

The models we describe in preceding sections benefit this study, but also the broader OpenDC community. We now focus on extensions made specifically for Capelin capacity planning functionality. At this prototype stage, we focus on components **A** (Scenario Portfolio Builder), **C** (Workload Modeler), and **E** (Scenario Portfolio Evaluator). We see these components as the most crucial to form a minimum viable product that has added value as support system to a capacity planner.

Extending the Backend

We extend the OpenDC database model and web server API specification with a notion of portfolios and scenarios. We list the data type schemas of these concepts in Figure 6.2. Our API extensions cover all CRUD (Create, Read, Update, and Destroy) operations of these options and are documented in the updated API specification document. We extend the Python web server according to the new API specification. Finally, we connect the updated simulator (as described in previous sections) with this new backend. An external watcher periodically polls the database for new scenarios and launches simulations on them as they come in.

The simulator outputs its periodic measurements to Parquet files. The choice of this file format is preceded by experiments with text-based and database-contained formats. We found each of the latter to fail more rigorous performance tests, simply unable to facilitate the dataset sizes we are working with (in the hundreds of GBs) within reasonable time. Even with a heavily compressed format such as Parquet, our simulations result in the very large file sizes. We have experimented with different ways of handling this, from external orchestration with Python to internal orchestration with Kotlin code. Our final solution involves the setup of a Spark big data pipeline to automatically aggregate the results from Parquet files into a per-repeat overview. Overall, these optimisations have allowed us to run thousands of repeats of complex 1-month traces in at most hours, while previous approaches would require months or even years to run, as well as special hardware.

Extending the Frontend

The frontend of OpenDC v1.x was focused on a live replay of simulated loads. This simulation mode is, while illustrative for demonstrations, less valuable for practitioners needing to extract meaningful, aggregate information from the simulation. In our extensions and work towards v2.x, we rebuild this interface to focus on meaningful simulation insights. We make portfolios of scenarios a core notion of the OpenDC simulation notion, enabling users to experiment with datacenters in a structured, iterative way. We unify the construction and simulation interfaces into one page, navigable through a context-sensitive sidebar.

For portfolios and scenarios, specifically, we facilitate easy creation through pop-over dialogues and setting of common default values. We replace the previous simulation replay mode with a graph overview of aggregate plots, on portfolio and scenario level. This facilitates direct comparison between different results, across different metrics.

Adding a Library of Components

In parallel to these implementation efforts, the author of this thesis is supervising an external project to design and implement a library of components in Capelin (conforming to component **B** in the design). This project will add functionality to save and reuse common topology components in OpenDC. It will also pre-populate this library of components with a set of industry-standard components from different domains.

6.2.8. Release and Validation of Artifacts

We release our extensions of the open-source OpenDC codebase and the analysis software artifacts on GitHub¹. The process our artifacts go through before they can be published to this repository consists of a number of

¹<https://github.com/atlarge-research/opendc>

Table 6.3: Experiment configurations. A legend of topology dimensions is provided below. (Notation: PI = Performance Interference, pub = public cloud trace, pri = private cloud trace, AP = Allocation Policy, AS = active-servers.)

Experiment	Candidate Topologies				Workloads		Op. Phen.		
	Mode	Quality	Direction	Variance	Trace	Loads	Failures	PI	AP
§6.3.5 Hor. / Ver.			\leftrightarrow \updownarrow		pri	sample	✓	✓	AS
§6.3.6 Velocity			\updownarrow		pri	sample	✓	✓	AS
§6.3.7 Op. Phen.	-	-	-	-	pri	original	X/✓	X/✓	all
§6.3.8 Pub. / Pri.			\leftrightarrow \updownarrow		pri/pub	sample	✓	X	AS
§6.3.9 HPC			\leftrightarrow \updownarrow		pri/hpc	sample	✓	✓	AS

Mode	Direction	Quality	Variance
replace expand	\leftrightarrow horizontal \updownarrow vertical	volume velocity	homogeneous heterogeneous

steps. All code artifacts need to be properly documented and readable. This includes both comments inside the code base and in central places such as the README document and other documentation files. In addition, we formally specify key communication protocols using open standards, such as the OpenAPI² specification of our API.

We adhere to a set of modern coding standards and enforce this adherence through the use of automated checks. We also check for common mistakes with the help of linting tools. All checks and all written test suites need to pass at any point of publication, as enforced by our continuous integration setups. Next to automated checks, we have a manual review policy in place in our version control system. This review policy requires at least one peer to review submitted code changes before they can be integrated into the main codebase.

We conduct thorough validation and tests of both the core OpenDC and our additions. For external validity, we commission a third party validation, detailed in Appendix B, and use its results.

6.3. Experiments with Capelin

We use our prototype of Capelin to construct an experiment pipeline and verify the reproducibility of its results and that it can be run within the expected duration of a capacity planning session (§6.3.1). All experiments use *long-term, real-world traces as input*.

Our experiment design, which Table 6.3 summarizes, is comprehensive and addresses key questions such as: Which input workload (§6.3.2)? Which datacenter topologies to consider (§6.3.3)? Which operational phenomena (§6.2.5)? Which allocation policy (§6.2.4)? Which user- and operator-level performance metrics to use, to compare the scenarios proposed by the capacity planner (§6.2.6)?

The most important decision for our experiments is which scenarios to explore. Each experiment takes in a capacity planning portfolio (see Section 5.4), starts from a base scenario, and aims to extend the portfolio with new candidate scenarios and its results. The baseline is given by expert datacenter engineers, and has been validated with hardware vendor teams. Capelin creates new candidates by modifying the base scenario along dimensions such as variety, volume, and velocity of any of the scenario-components. In the following, we experiment systematically with each of these.

6.3.1. Execution and Evaluation

The *reproducibility* of a study is a crucial component of its internal and external validity. A study is reproducible if its methods are described in such manner that externals can reproduce its results, ideally with exactly the same outcomes. Our results are fully reproducible, regardless of the physical host running them. Each source of non-determinism is seeded by the current repetition, ensuring reproducibility of our measurements.

All setups are repeated 32 times. The results, in files amounting to hundreds of GB in size due to the large workload traces involved, are evaluated statistically and verified independently. Factors of randomness

²<https://swagger.io/specification/>

(e.g., random sampling, policy decision making if applicable, and performance interference modeling) are seeded with the current repetition to ensure deterministic outcomes, and for fairness are kept consistent across scenarios.

Capelin could be used during capacity planning meetings. A single evaluation takes 1–2 minutes to complete, enabled by many technical optimizations we added to the simulator. The full set of experiments is conveniently parallel and takes around 1 hour and 45 minutes to complete, on a “beefy” but standard machine with 64 cores and 128GB RAM; parallelization across multiple machines would reduce this to minutes. Queries within meetings, requiring less scenarios and repeats than our experiments, could quickly be answered by Capelin, at an increasing degree of accuracy, depending on the available time. Increasing the number of repeats in this way allows the implementation to partly meet NFR1: A lower number of repeats allows for a short response time but lower accuracy, while a higher number of repeats increases response time but also increases accuracy.

6.3.2. Workload

We experiment with a business-critical workload trace from Solvinity, a *Dutch private cloud provider* [120]. The anonymized version of this trace has been published in a public trace archive [58]. We were provided with the full, deanonymized data artifacts of this trace, which consists of more than 1,500 VMs along with information on which physical resources were used to run the trace and which VMs were allocated to which resources. We cannot release these full traces due to confidentiality, but release the summarized results.

The *full trace* includes a range of VM resource-usage measurements, aggregated over 5-minute-intervals over three months. It consumes 3,063 PFLOPs (*exascale*), with the mean CPU utilization on this topology of 5.6%. This low utilization is in line with industry, where utilization levels below 15% are common [129], and reduce the risk of not meeting SLAs.

For all experiments, we consider the full trace, and further generate three other kinds of workloads as samples (fractions) of the original workload, as described in Section 6.2.2. These workloads are sampled from the full trace, resulting, in turn, to 306 PFLOPs (0.1 of the full trace), 766 (0.25), and 1,532 (0.5).

For the §6.3.8 experiment, we further experiment with a public cloud trace from Azure [33]. We use the most recent release of the trace. The formats of the Azure and the Solvinity traces are very similar, indicating a de facto standard has emerged across the private and public cloud communities. One difference in the level of anonymity of the trace requires an additional assumption. Whereas the Solvinity trace expresses CPU load as a frequency (MHz), the Azure trace expresses it as a utilization metric ranging from 0 to the number of cores of that VM. Thus, for the Azure trace, in line with Azure VM types on offer we assume a maximum frequency of 3 GHz and scale each utilization measurement by this value. The Azure trace is also shorter than Solvinity’s full trace, so we shorten the latter to Azure’s length of 1 month. We present aggregate metrics of both the Azure and the Solvinity trace, in Table 6.5.

We combine for the §6.3.8 experiment the two traces and investigate possible phenomena arising from their interaction. We disable here performance interference, because we can only derive it for the Solvinity trace (see §6.2.5). To combine the two traces, we first take a random sample of 1% from the (very large) Azure trace, which results in 26,901 VMs running for one month. We then further sample this 1%-sample, using the same method as for Solvinity’s full trace. The full procedure is described in Section 6.2.2.

For the §6.3.9 experiment, we manipulate the composition of the trace. The cloud provider (Solvinity) has supplied us with a (confidential) list of names of VMs from the original trace which host HPC workloads. Given this subset, we can now up- or down-sample the number of HPC experiments in the trace, using the method described in Section 6.2.2. We cannot provide the classifications of VMs or the fraction of the workload that they represent, but we can say that they represent a significant portion of the trace.

6.3.3. Datacenter topology

For all experiments we set the topology that ran Solvinity’s original workload (the full trace in §6.3.2) as the base scenario’s topology. This topology is very common for industry practice. It is a subset of the complete topology of the Solvinity when the full trace was collected, but we cannot release the exact topology or the entire workload of Solvinity due to confidentiality.

From the base scenario, Capelin derives candidate scenarios as follows. First, it creates a temporary topology by choosing half of the clusters in the topology, consisting of average-sized clusters and machines, compared to the overall topology. Second, it varies the temporary topology, in four dimensions: (1) the *mode of operation*: replacement (removing the original half and replacing it with the modified version) and expansion (adding the modified half to the topology and keeping the original version intact); (2) the *modified quality*:

Table 6.5: Aggregate statistics for both workloads used in this study. (Notation: AP = Solvinity.)

Characterization		AP	Azure
VM submissions per hour	Mean ($\times 10^{-3}$)	31.836	4.547
	CoV	134.605	17.188
VM duration [days]	Mean	20.204	2.495
	CoV	0.378	3.072
CPU load [TFLOPs]	Mean ($\times 10^2$)	9.826	64.046
	CoV	2.992	4.654

volume (number of machines/cores) and velocity (clock speed of the cores); (3) the *direction of modification*: horizontal (more machines with fewer cores each) and vertical (fewer machines with more cores each); and (4) the *kind of variance*: homogeneous (all clusters in the topology-half modified in the same way) and heterogeneous (two thirds in the topology-half being modified in the designated way, the remaining third in the opposite way, on the dimension being investigated in the experiment).

Each dimension is varied to ensure cores and machine counts multiply to (at least) the same total core count as before the change, in the modified part of the topology. For volume changes, we differentiate between a horizontal mode, where machines are given 28 cores (a standard size for machines in current deployments), and vertical modes, where machines are given 128 cores (the largest CPU models we see being commonly deployed in industry). For velocity changes, we differentiate between the clock speed of the base topology and a clock speed that is roughly 25% higher. Because we do not investigate memory-related effects, the total memory capacity is preserved. Due to confidentiality, we can only describe the derived topologies in relative terms.

6.3.4. Listing of Full Results

In the subsections below, we will highlight a small selection of the key metrics for each experiment. For full transparency, we present the entire set of metrics for each experiment in the appendices. Appendix C visualizes the full results for all metrics and Appendix D lists the full results for the two most important metrics in tabular form.

6.3.5. Horizontal vs. Vertical Resource Scaling

Our main findings from this experiment are:

MF1: Capelin enables the exploration of a complex trade-off portfolio of multiple metrics and capacity dimensions.

MF2: Vertically scaled topologies can improve power consumption (median lower by 1.47x-2.04x) but can lead to significant performance penalties (median higher by 1.53x-2.00x) and increased chance of VM failure (median higher by 2.00x-2.71x, which is a high risk!).

MF3: Capelin reveals how correlated failures impact various topologies. Here, 147k–361k VM-slices fail.

The scale-in vs. scale-out decision has historically been a challenge across the field [117][50, §1.2]. We investigate this decision in a portfolio of scenarios centered around horizontally (symbol \leftrightarrow) vs. vertically (\updownarrow) scaled resources (see §6.3.3). We also vary: (1) the decision mode, by replacing the existing infrastructure (\leftrightarrow) vs. expanding it (\rightarrow), and (2) the kind of variance, homogeneous resources (\boxplus) vs. heterogeneous (\boxminus). On these three dimensions, Capelin creates candidate topologies by *increasing* the volume (\uparrow) and compares their performance using four workload intensities, two of which are shown in this analysis. We consider three metrics for each scenario: Figure 6.3 (top) depicts the overcommitted CPU cycles, Figure 6.3 (middle) depicts the power consumption, and Figure 6.3 (bottom) depicts the number of failed VM time slices.

Our key *performance* indicator is overcommitted CPU cycles, that is, the count of CPU cycles requested by VMs but not granted, either due to collocated VMs requesting too many resources at once, or due to

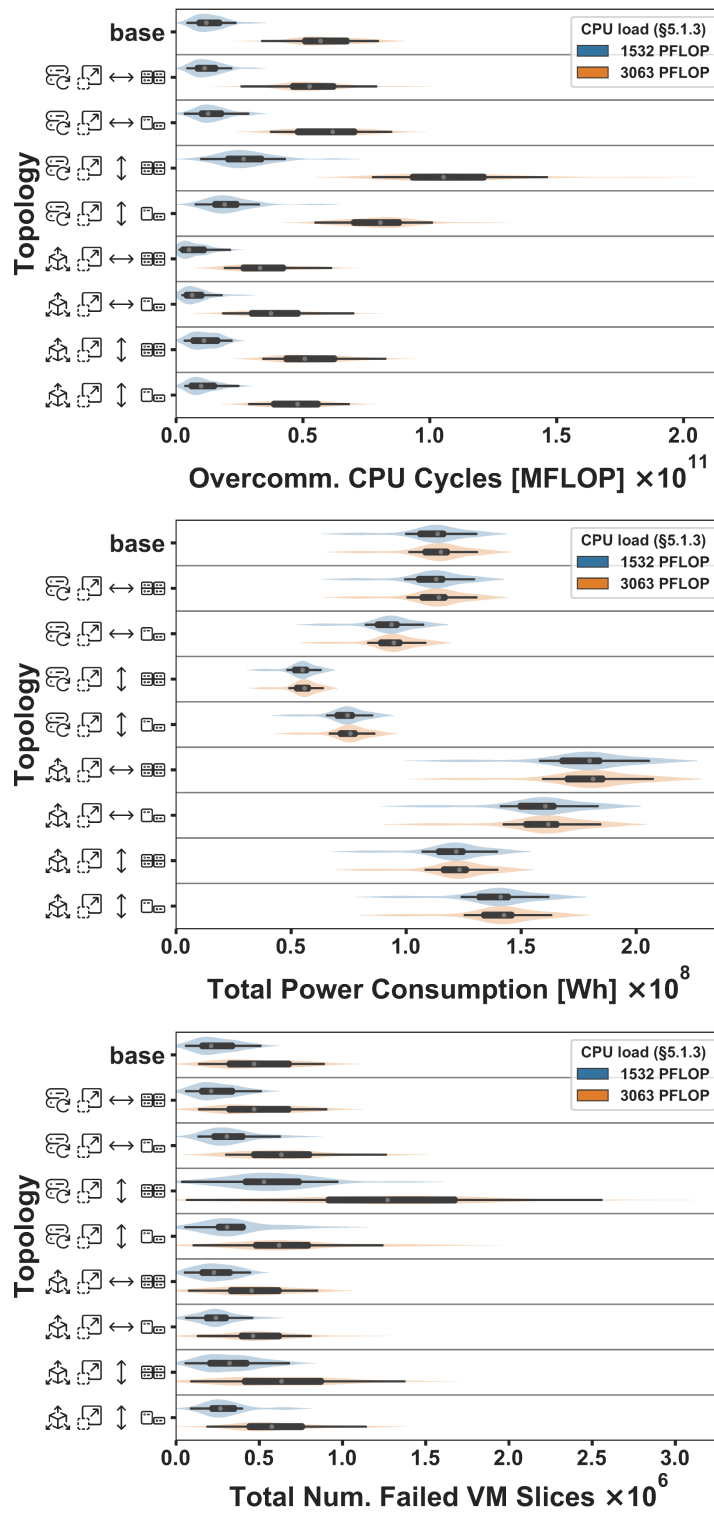


Figure 6.3: Results for a portfolio of candidate topologies and different workloads (§6.3.5): (top) overcommitted CPU cycles, (middle) total power consumption, (bottom) total number of time slices in which a VM is in a failed state. Table 6.3 describes the symbols used to encode the topology.

performance interference effects taking place. We observe in Figure 6.3 (top) that vertically scaled topologies (symbol \Downarrow) have significantly higher overcommission (lower performance) than their horizontally scaled counterparts (\Leftarrow , the other three symbols identical). The median value is higher for vertical than for horizontal scaling, for both replaced (\Leftarrow) and expanded (\Leftarrow) topologies, by a factor of 1.53x–2.00x (calculated

as the ratio between medians of different scenarios at full load). This is a large factor, suggesting that vertically scaled topologies are more susceptible to overcommission, and thus lead to higher risk of performance degradation. The decrease in performance observed in this metric is mirrored by the granted CPU cycles metric in Figure C.1b, which decreases for vertically scaled topologies. Among replaced topologies (all combinations including \mathbb{C}), the horizontally scaled, homogeneous topology ($\mathbb{C}\downarrow\mathbb{H}$) yields the best performance, and in particular the lowest median overcommitted CPU. We also observe that expanded topologies (\mathbb{H}) have lower overcommission than the base topology, so adding machines is worthwhile. We observe all these effects strongly for the full trace (3,063 PFLOPs), but less pronounced for the lower workload intensity (1,531 PFLOPs).

But performance is not the only criterion for capacity planning. We turn to *power consumption*, as a proxy for cost analysis and environmental concerns. We see here that vertically scaled topologies (\downarrow) drastically improve power consumption, for median values by a factor of 1.47x–2.04x, contrasting their worse performance compared to horizontal scaling (\leftrightarrow). As expected, all expanded topologies (\mathbb{H}), which have more machines, incur higher power-consumption than replaced topologies (\mathbb{C}). Higher workload intensity (i.e., for the 3,063 PFLOPs results) incurs higher power consumption, although less pronounced than earlier.

We also consider the *amount of failed VM time-slices*. Each failure here is full-stop (§6.2.5), which typically escalates an alarm to engineers. Thus, this metric should be minimized. We observe significant differences here: the median failure time of a homogeneous vertically scaled topology ($\downarrow\mathbb{H}$) is between 2.00x–2.71x higher than the base topology. This metric shows similarities qualitatively with the overcommitted CPU cycles. Vertical scaling is correlated not only with worse performance, but also with higher failure counts. We see that vertical scaling leads to a significant increase in the maximum number of deployed images per physical host (Figure C.2b), which leads to larger failure domains and thus potentially higher failure counts. The effect is less pronounced when making heterogeneous compared to homogeneous procurement.

Our findings show that Capelin gives practitioners the possibility to *explore a complex trade-off portfolio* of dimensions such as power consumption, performance, failures, workload intensity, etc. Optimization questions surrounding horizontal and vertical scaling can therefore be approached *with a data-driven approach*. We find that decisions including heterogeneous resources can provide meaningful compromises between more generic, homogeneous resources; they also lead to different decisions related to personnel training (not shown here). We show significant differences between candidate topologies in all metrics, translating to very different power costs, long-term. We conclude that *Capelin can help test intuitions and support complex decision making*.

6.3.6. Expansion: Velocity

Our main findings from this experiment are:

MF4: Capelin enables exploring a range of resource dimensions frequently considered in practice, such as component velocity.

MF5: Increasing velocity can reduce overcommitted CPU cycles by 3.3%.

MF6: Expanding a topology by velocity can improve performance by 1.54x, compared to volume expansion.

In vertical horizontal scaling, practitioners are also faced with the decision of which qualities to scale. This experiment varies the velocity of resources both homogeneously and heterogeneously, while replacing or expanding the existing topology. Figure 6.4 depicts the explored scenarios and their performance, in the form of overcommitted CPU cycles.

We find that in-place, homogeneous vertical scaling of machines with higher velocity leads to slightly better performance, by a percentage of 3.3% (compared to the base scenario, by median). In this dimension, performance varies only slightly between homogeneously and heterogeneously scaled topologies, for all metrics (see also Appendix C). Expanding the topology homogeneously ($\mathbb{H}\downarrow\mathbb{H}$) with a set of machines with higher CPU frequency helps reduce overcommission more drastically, also improving it beyond the lowest overcommission reached by the homogeneous vertical expansion explored in the previous experiment, in Figure 6.3. When expanding, this cross-experiment comparison *shows an improvement of performance of a factor of 1.54x*.

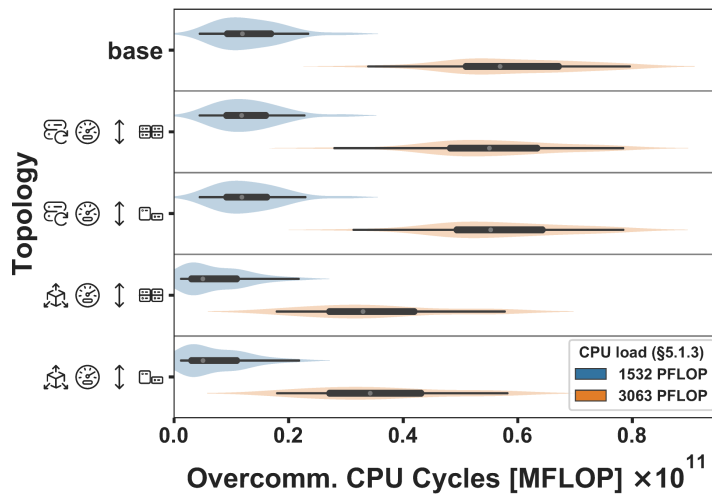


Figure 6.4: Overcommitted CPU time for a portfolio of candidate topologies and different workloads, for Experiment 6.3.6.

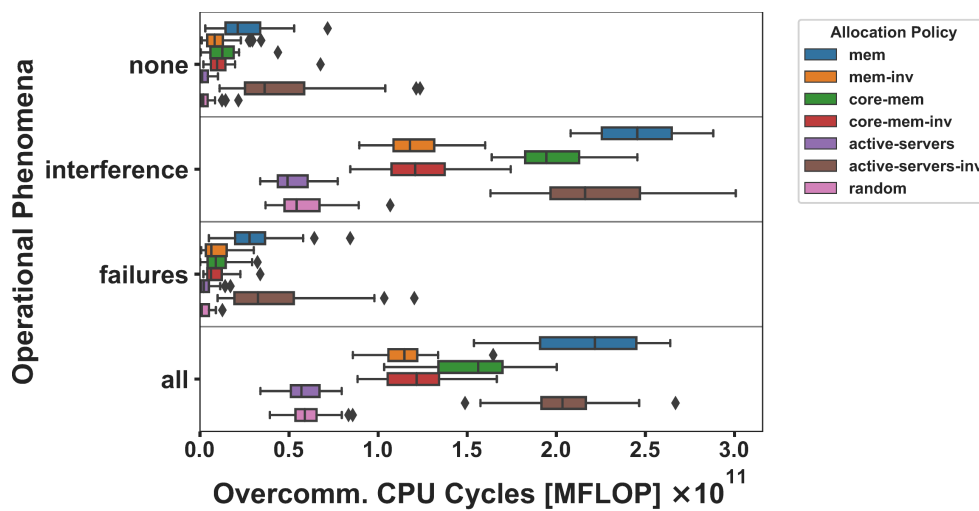


Figure 6.5: Overcommitted CPU cycles for a portfolio of *operational phenomena* (the “none” through “all” sub-plots), and *allocation policies* (legend), for Experiment 6.3.7.

6.3.7. Impact of Operational Phenomena

Our main findings from this experiment are:

MF7: Capelin enables the exploration of diverse allocation policies and operational phenomena, both of which lead to important differences in capacity planning.

MF8: Modeling performance interference can explain 80.6%—94.5% of the overcommitted CPU cycles.

MF9: Different allocation policies lead to different performance interference intensities, and median over-committed CPU cycles different by factors of 1.56x–30.3x compared to the best policy–high risk!

This experiment addresses operational factors in the capacity planning process. We explore the impact of better handling of physical machine failures, the impact of (smarter) scheduler allocation policies, and the impact of (the absence of) performance interference on overall performance. Figure 6.5 shows the impact of different operational phenomena on performance, for different allocation policies. We observe that performance interference has a strong impact on overcommission, dominating it compared to the “failures” sub-plot, where only failures are considered, or with the “none” sub-plot, where no failures or interference are considered. Depending on the allocation policy, it represents between 80.6% and 94.5% of the overcommission recorded in simulation for the “all” sub-plot, where both failures and interference are considered.

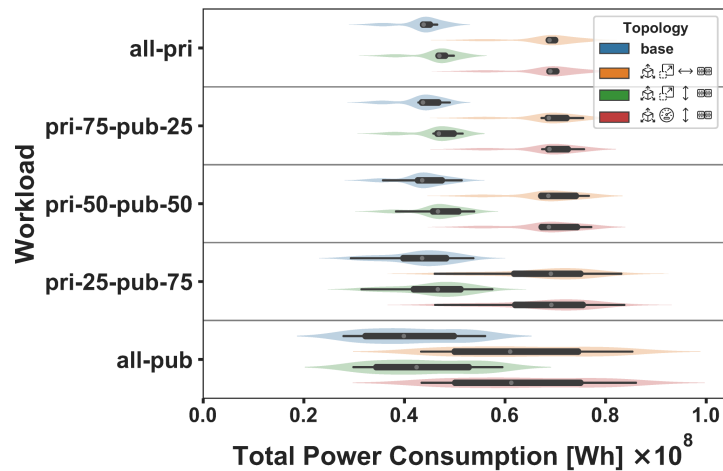


Figure 6.6: Total power consumption for a portfolio of candidate topologies (legend), subject to *different workloads* (the “all-pri” to “all-pub” sub-plots), for Experiment 6.3.8.

This is visualized more in detail in Figure C.9d, which plots the interference itself, separately.

We also see the large impact that live resource management (in this case, the allocation policy) can have on QoS. Median ratios vary between 1.56x and 30.3x vs. the best policy, with *active-servers* (see §6.2.4) generally best-performing. While the *random* policy is second-best in terms performance, its performance is much worse for other metrics, such as power consumption (Fig. C.9e), failures (Fig. C.9f), and maximum number of VMs per machine (Fig. C.10a). Finally, we observe that enabling failures increases the colocation ratio of VMs (see Figure C.10a).

We conclude *Capelin can help model aspects that are important but typically not considered for capacity planning.*

6.3.8. Impact of Public vs. Private Cloud Workloads

Our main findings from this experiment are:

MF10: Capelin enables exploring what-if scenarios with new workloads as they become available.

MF11: Power consumption can vary significantly more in all-private vs. all-public cloud scenarios, with the range higher by 4.79x–5.45x.

This experiment explores the impact that a new workload type can have if added to an existing workload, an exercise capacity planners have to consider often, e.g., for new customers. We combine here the 1-month Solvinity and Azure traces (see §6.3.2).

Figure 6.6 shows the power consumption for different combinations of both workloads and different topologies. We observe the unbiased variance of results [35, p. 32] is positively correlated with the fraction of the workload taken from the public cloud (Azure). Depending on topology, the variance increase with this fraction ranges from 4.78x to 5.45x. Expanding the volume horizontally (horizontal expansion) leads to the lowest increase in variance. The workload statistics listed in Table 6.5 show that the Azure trace has far fewer VMs, with higher load per VM and shorter duration, thus explaining the increased variance. Last, all candidate topologies have a higher power consumption than the base topology.

We also observe performance degrading with increasing public workload fraction (see Figure C.13c), calling for a different topology or more sophisticated provisioning policy to address the differing needs of this new workload. We see that horizontal volume expansion (horizontal expansion) provides the best performance in the majority of workload transition scenarios. We conclude *Capelin can support new workloads as they appear, so before they are deployed.*

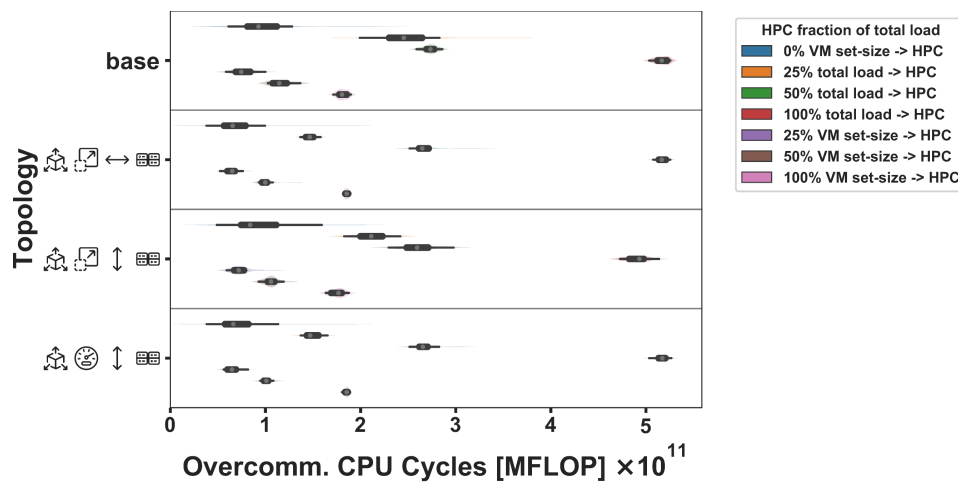


Figure 6.7: Overcommitted CPU cycles for a portfolio of *sampled HPC workloads* on a series of candidate topologies, for Experiment 6.3.9.

6.3.9. Impact of HPC Workloads

Our main findings from this experiment are:

MF12: Capelin enables exploring what-if scenarios that include emerging workloads within existing workloads, such as HPC workloads.

MF13: A transition from no HPC to only HPC, with constant load, can adversely impact performance by a factor of up to 5.58x.

This experiment explores the impact that a differing subset of a workload can have on the performance of the larger enclosing workload. We focus on HPC as the differing workload. Such questions are common to capacity planning around workload compositions. These can change significantly over time, e.g. with certain types of workloads becoming more popular. We use the 1 month Solvinty trace here, since it consists of a subset of HPC VMs as well as other “conventional” VMs. We up-sample and down-sample the HPC fraction, according to the method described in Section 6.3.2.

We observe that changing the composition of a workload consisting of both HPC and non-HPC components can have significant impact on the performance of the system, even if the total load is kept constant. We see a decrease by a factor of 5.58x in the worst case, when the total load is kept the same but the composition is changed from non-HPC to full-HPC. Amongst candidate topologies, we see vertical volume scaling performing best in scenarios with higher HPC fractions, while horizontal scaling seems to perform best in scenarios devoid of any HPC workload fraction.

We also observe that the sampling method can have significant impact on performance. If workload composition is altered by count-sampling, the resulting overcommission is up to 2.86x times less severe compared to by alteration by load-sampling. We see a potential link with the alignment of burst patterns that can lead to a higher chance of burst request collisions when sampled on load, since the average total load of a HPC VM appears to be lower but also more concentrated in certain regions.

6.4. Discussion

We now summarize the contributions of this chapter and discuss possible threats to their validity.

6.4.1. Summary

We present a working software prototype of Capelin with key features of the design outlined in Chapter 5. Our results show that Capelin can support capacity planning processes, exploring changes from a baseline scenario alongside four dimensions. We found that capacity plans common in practice could potentially lead to significant performance degradation, e.g., 1.5x–2.7x. We also gave evidence of the important, but often discounted, role that operational choices and operational phenomena play in capacity planning.

6.4.2. Threats to Validity

We discuss potential threats to internal validity, construct validity, and external validity.

Internal Validity

The *absence of public experiment data artifacts* can be seen as a threat to internal validity, since causal and aggregated findings that we claim may in theory not be reflected by full, confidential data artifacts. The confidentiality of the trace and topology we use in simulation prohibits the release of detailed artifacts and results. However, an anonymized version of the trace is available in a public trace archive [58], which can be used to explore a restricted set of the workload. The HPC classifications cannot be made public, since they would reveal sensitive information on the nature of the anonymized workloads. The Azure traces used in the experiment in §6.3.8 are public, along with our sampling logic for their use, and can therefore be locally used along with the codebase.

Construct Validity

A threat to validity to construct validity could be seen in the *validity of the outputs* of the (extensions to the) simulator itself. This threat also partially extends to external validity, since it impacts both the quality and reliability of measurements in our study as well as the degree to which findings can be extrapolated. We cover this threat extensively in Appendix B, commissioning a third party report of validation efforts taken to ensure validity and soundness of the simulator.

External Validity

Building topologies in practice requires consideration of many *different kinds of resources*. In our study, we only actively explore the CPU resource dimension in the capacity planning process, to restrict the scope. Adding or removing CPUs to/from a machine however can relate to different types of memory or network becoming applicable or necessary. This can have impacts on costs and energy consumption, impacting the quality of decision support provided and limit external applicability of the tool. This could therefore be seen as a threat to external validity. Nevertheless, the performance should suffer only minimal impact from this, since CPU consumption can be regarded as the critical factor in these considerations. In addition, future extensions to the simulation model OpenDC will directly become available to planners using Capelin.

7

Conclusion and Future Work

Accurately planning cloud datacenter capacity is key to meeting the needs of the 2020s society whilst saving costs and ensuring environmental sustainability. Although capacity planning is crucial, the current practice has not been analyzed in decades and publicly available tools to support practitioners are scarce. In this work, we have progressed towards addressing this shortcoming with a systematic overview of the practice and a novel approach to capacity planning. This section summarizes the contributions of this thesis and looks forward to future lines of research emerging from it.

7.1. Conclusion

In this work, we address three main research questions related to the problem of capacity planning. Having detailed the capacity planning problem in Chapter 1 and surveyed the state-of-the-art in knowledge related to it in Chapter 2, and having presented our contributions in Chapters 3 through 6, we can now address the questions, individually.

(RQ1) State-of-the-art: *How to capture and assess the current state-of-the-art of capacity planning for cloud infrastructure?*

This work presents a systematic overview of the literature and uncovers the current state of the field. A series of meta-analyses helps derive unique insights from the surveyed publications. We have also conducted a unique community survey, using guided interview with diverse practitioners from a variety of backgrounds, whose results led us to synthesize five functional requirements.

(RQ2) Design of a system: *How to design a capacity planning system for cloud infrastructure that responds to key issues faced in the community?*

We have designed Capelin to meet the requirements synthesized from our survey of interviews. Our decision support tool features the ability to model datacenter topologies and virtualized workloads, to express what-if scenarios and QoS requirements, to suggest scenarios to evaluate, and to evaluate and explain capacity plans. Capelin uses a novel abstraction, the capacity planning portfolio, to represent, explore, and compare a variety of scenarios.

(RQ3) Evaluation of a system: *How to evaluate a capacity planning system for cloud infrastructure?*

We have implemented a working prototype of Capelin, extending the OpenDC codebase significantly in the process. Experiments based on real-world workload traces collected from private and public clouds demonstrate Capelin's capabilities. Results show that Capelin can support capacity planning processes, exploring changes from a baseline scenario alongside four dimensions. We found that capacity plans common in practice could potentially lead to significant performance degradation, e.g., 1.5x–2.7x. We also gave evidence of the important, but often discounted, role that operational choices (e.g., the allocation policy) and operational phenomena (e.g., performance interference) play in capacity planning.

We have released Capelin as FOSS for capacity planners to use. Discussions with partners in academia and industry are promising and show interest in the adoption of Capelin. Solvinity, the cloud provider that has offered data artifacts for experimentation, sees large potential in Capelin:

“Predicting the future has always been one of the big wishes in life. Predicting demand in IT has always been the wish of the CIO. The Capelin initiative brings this wish a bit closer to reality. Its simulation capabilities based on big data gives IT leaders a tool to predict capacity demand for their datacenters. This solution is highly promising for many.”

Bas Demmink – CTO, Solvinity

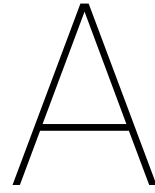
7.2. Future Work

This work provides the foundation for research and development for a field that has previously received insufficient attention and currently lacks innovation. We identify five directions of future work building on the contributions of this thesis:

1. We intend to conduct a *structured survey*, in the form of a textual questionnaire, to reach a larger base of capacity planning practitioners and augment the initial findings made in our interview study. Such a textual questionnaire would consist of a set of qualitative and predominantly quantitative questions, answered and submitted remotely by a large set of practitioners. The choice for flexibility in the approach taken in our initial survey has allowed us to quickly adapt to what questions and topics we see being raised and considered relevant. Now that an initial survey exists, a more structured survey could be formulated and would be able to reach a far wider audience, since per-interlocutor efforts would be significantly lower. The analysis of such a survey could build a more systematic foundation for the frame of findings found in the initial survey.
2. We plan to *deepen and engineer Capelin* to be fit for deployment in a production context. Improvements in this context can be divided into two categories: development of the OpenDC platform, and of Capelin in particular. First, we are continuously working to improve and extend the capabilities of the OpenDC simulation platform. This includes research into realistic resource models, upcoming workload types, and resource management strategies. Any improvements in this category are directly applicable to Capelin users, as well. Second, we plan to implement Capelin’s full feature set (including capacity plan generation and portfolio suggestion components) and make it widely adoptable by engineering fast, interactive components and including a broad library of pre-built topologies and workloads. We also plan to add dedicated integrations between Capelin and leading monitoring systems and topology definition software, from vendors such as VMware and open-source monitoring software projects such as Grafana. This will make it easier for practitioners to input their current situation into the tool and start work directly from that point. Such an integration could for example serve as a link with a monitoring database that directly imports usage data on the last one month of operation into the Capelin database.
3. Building on the previous area of future work, we are *investigating the use of Machine Learning and Artificial Intelligence search techniques* to make the *Capacity Plan Generator component* more capable of exploring the enormous design-space. While capacity planning will likely remain a human-centered activity with tools only providing support, a tool can significantly alleviate the load on the human practitioner. Automatic design space exploration can further assist capacity planners by evaluating many different points in the design space and presenting key trade-offs to the planner. Multi-objective optimization algorithms, found both in conventional Artificial Intelligence techniques and in Machine Learning approaches, should be evaluated on this problem specifically. A user study of chosen approaches compared against proposed approaches could validate the quality of proposed plans.
4. We see opportunities for research into *cloud user behavior* when emerging resources are deployed, a factor especially relevant in scientific clouds. We find three main opportunities: First, we find from interviews that emerging resource types (such as new kinds of accelerator hardware) are sometimes only slowly adopted by a user base. Research into automatic porting, widespread education, and assistance tools could help ease and speed up this adoption process, to ensure higher usage utilization and reduce waste. Second, we find that user behavior when provided with new capacity is still poorly understood. We find in interviews that there is a variable delay between the expansion of available resources and

the scaling up of demand to meet those resources, resulting in frequent resource waste. Future work could investigate this behavior and the role that cloud administrators and capacity planners can play in reducing this delay, such as through more fine-grained capacity plans. Third, we see potential for synergy between the previous two directions and a workload forecasting component driven by Machine Learning techniques. The behavior may be better understood and forecast with the help of pattern recognition and extrapolation algorithms.

5. We also plan to include *more workload types*, such as virtualized FaaS workloads. A tool such as Capelin has the potential to allow capacity planners to investigate new workloads far before they even enter their infrastructure. It can also point out changes that might be needed to facilitate those new workloads. This kind of foresight can help inform decision-making and long-term planning, especially with workloads on which intuition might still be difficult to form, due to their novel and emerging nature.



Capacity Planning Interview Script

In this chapter, we list the interview script used for the interviews described in Chapter 4. To encode instructions to the reviewer, we use the following notation. The font determines the type of instruction: Questions are written in standard fonts, emphasis indicates instructions to be read to the interlocutor by the interviewer (not necessarily verbatim), and mono-space font represents instructions for the reviewer. The questions are numbered for cross-reference and divided into 5 category. Each question is assigned one out of three priority levels, indicated by asterisks (*, **, and ***), with three stars indicating the highest priority. Each category (section) of questions is allocated a number of minutes, listed between parentheses.

A.1. Part 1: Overview (15')

Thanks for agreeing to meet with me. To get the most out of this conversation, would you allow me to record the conversation for note-taking purposes? I will not share the recording with anyone else but you, if you want a copy, and I will delete the recording at the end of my thesis project.

If you have any concerns about sharing this kind of information with me, let us talk quickly and openly about it. We hope you will share openly. Rest assured, we want more to learn about the issues around capacity planning than to publish on it.

I will transcribe the recording for myself, and for any use of a snippet of your words, I will ask you specifically for approval to release, with due reference, unless you want me to keep the author anonymous, of course.

We are interested in learning more about how businesses think about having IT infrastructure and services always available and plentiful. We call this capacity planning, and know we are referring here only to IT and the IT team, and not to other types of “capacity”. We want to learn and share with you what processes are used, what challenges exist, and how can we help solve them.

- (Q1) *** How important is it to have IT infrastructure and services always available and plentiful in your business?
- (Q2) *** What kind of services do you provide? How important is it for your different services?
- (Q3) *** Can you give us an example of a success in capacity planning? Share with us a good idea, a good process, some situation when all worked well?
- (Q4) *** Can you give us an example of an insuccess in capacity planning? Share with us a mistake, an erroneous process, some situation when many things failed to worked well or took much more to get through than expected?
- (Q5) *** What does the typical process for capacity planning look like at your company? You can start with an overview, or even from a concrete example, like how to get a new cluster in operation.
- (Q6) ** A few yes or no questions:
1. Do you have “what if” scenarios?
 2. Do you consider hybrid or public clouds to be part of your capacity planning process?

3. Would you be willing to share historical data on capacity planning?
4. Do you consider human personnel (availability, experience, training) when planning for new capacity?

A.2. Part 2: The Process (15')

- (Q7) *** Who are the stakeholders? Who gets to take the decision? Who gets to give input? Is this a board-level decision? Is it left to operations?
- (Q8) *** On what time and infrastructure scale are your typical decisions?
- (Q9) *** What factors do you take into account when making capacity planning decisions? Does this differ per stakeholder; if so, how?
- (Q10) ** What is the margin for error in the decision making process?
- (Q11) * How frequently are capacity planning decisions made? Also, how long does a decision take?
- (Q12) * What kind of data sources do you consult in your capacity planning process?
- (Q13) * What kind of tools do you use in your capacity planning process? For planning, recording, sharing information at different levels in the organization, etc.
- (Q14) ** How are errors or issues about capacity planning preserved? How frequent/severe are the errors that are made? How do people learn from these issues?

A.3. Part 3: Inside Factors (15')

- (Q15) *** What kinds of IT services and infrastructure are part of your capacity planning processes?
- (Q16) ** I will ask the same question about four kinds of workloads.
 What are your capacity planning processes for business-critical workloads?
 What are your capacity planning processes for big data workloads?
 What are your capacity planning processes for serverless workloads?
 What are your capacity planning processes for high performance computing workloads?
- (Q17) *** How do you try to combine multiple workloads in the same capacity planning process? Shared infrastructure? Shared services? What role do hybrid or public cloud offerings play in your capacity planning process?
- (Q18) *** Because serverless workloads are so new, I'd like to ask a couple more questions about them. With such fine-granularity and variable workloads, how do you reason about needs? Do you reason differently about them than about other (more traditional) workloads? How do you reason about workload isolation (performance, availability, security, etc.)?
- (Q19) * What are some typical "what if" scenarios?

A.4. Part 4: Outside Factors (10')

- (Q20) *** What regulatory constraints (laws, standards; e.g. GDPR, concerning where you get the capacity) play a role in the decision process?
- (Q21) *** What financial aspects (costs of resources, personnel, etc.) or technical aspects (new generation of hardware/software/IT paradigms) play a role in the decision process?
- (Q22) *** How and which human factors are involved in your decision making on resource capacity planning?
- (Q23) ** Do you make capacity planning decisions on a multi-datacenter level, or on a local level?
- (Q24) *** Do you do capacity planning specifically for production, development, and/or test?
- (Q25) * What are some typical "what if" scenarios?

A.5. Part 5: Summary and Follow-Up (5')

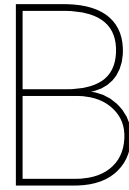
- (Q26) *** What would your ideal process for capacity planning look like? Something that is not already there?
- (Q27) ** Which other processes do you see capacity planning linked with? For example, managing change, evolution of business requirements, etc.?
- (Q28) ** What other aspects would you like to share?

Follow-Up Points

Explain what I will do with the information.

Ask if they want a summary or report related to my thesis project.

If they answered "yes" to sharing historical data, follow up here.



External Validation of the Simulation

We have commissioned a third-party to report on validation steps taken to ensure the results of the simulator are valid, sound, and reliable. This report is written by Fabian Mastenbroek, core simulation engineer on the OpenDC team. The report is reproduced below.

We discuss here the validity of the outputs of the (extensions to the) simulator. Capelin uses datacenter-level simulation using real-world traces to evaluate portfolios of capacity planning scenarios. Although real-world experimentation would provide more realistic outputs, evaluating the vast amount of scenarios generated by Capelin on physical infrastructure is prohibitively expensive, hard to reproduce, and cannot capture the scale of modern datacenter infrastructure, notwithstanding environmental concerns. Alternatively, we can use mathematical analysis, where datacenter resources are represented as mathematical models (e.g., hierarchical and queuing models). However, this approach is limited because its accuracy relies on pre-existing data from which the models are derived. Further considering the complexity and responsibilities of modern datacenters, this approach becomes infeasible.

Given that the effectiveness of Capelin depends heavily on (the correctness of) simulator outputs, we have worked very carefully and systematically to ensure the validity of the simulator. For the validity of the simulator, we consider three main aspects: (1) validity of results, (2) soundness of results, and (3) reliability of results. Below, we discuss for each of these aspects our approach and results.

B.1. How to ensure simulator outputs are valid?

We consider simulator outputs valid if a realistic base model (e.g., the datacenter topology) with the addition of a workload and other assumptions (e.g., operational phenomena) can reflect realistically real-world scenarios based on the same assumptions.

We ensure validity of simulator outputs by tracking a wide variety of metrics (see Section 6.2.6) during the execution of simulations in order to validate the behavior of the system. This selection is comprised of metrics of interest which we analyze in our experiments, but also fail-safe metrics (e.g., total requested burst) that we can verify against known values.

Moreover, we employ step-by-step inspection using the various tools offered by the Java ecosystem (e.g., Java Debugger, Java Flight Recorder, and VisualVM) to verify the state of individual components on a per-cycle basis.

B.2. How to ensure simulator outputs are sound?

While the simulator may produce valid outputs, for them to be useful, these outputs must also be realistic and applicable to users of Capelin. That is, the assumptions that support the datacenter model must hold in the real world, for the simulator outputs to be sound and in turn be useful. Concretely, a particular choice of scheduling policy might produce valid results, yet may not reflect reality.

To address this, we have created “replay experiments” that replicate the resource management decisions made by the original infrastructure of the traces, based on placement data from that time. We do not support live migration of VMs that occurs in the placement data, since VM placements are currently fixed over time

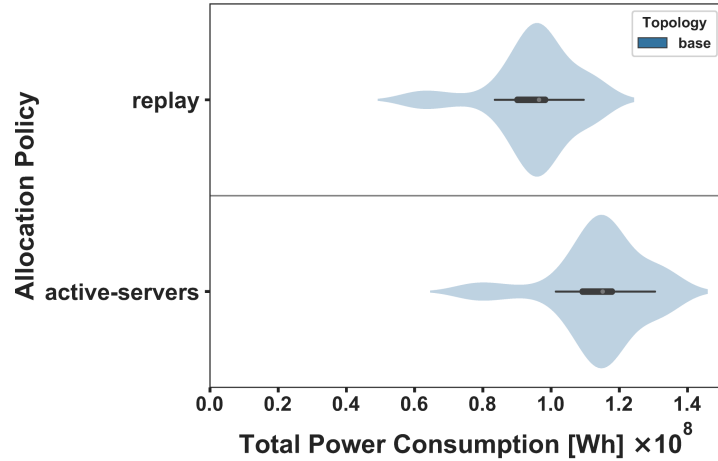


Figure B.1: Validation with a replay policy, copying the exact cluster assignment of the original deployment. For a legend of topologies, see Table 6.3.

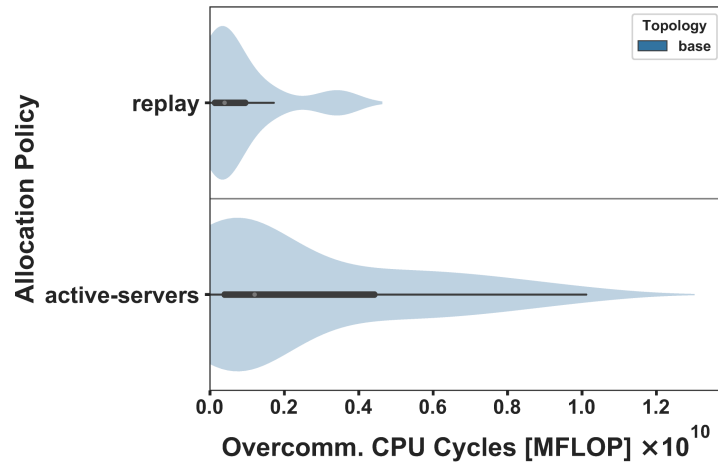


Figure B.2: Validation with a replay policy, copying the exact cluster assignment of the original deployment. For a legend of topologies, see Table 6.3.

in OpenDC. However, the majority of VMs do not migrate at all. Capacity issues due to not supporting live migration are resolved by scheduling VMs on other hosts in the cluster based on the mem policy.

The “replay experiments” are run in an identical setup to the experiments in Section 6.3 and its results are compared to the `active-servers` allocation policy. We find that:

1. The total overcommitted burst shows distributions that are similar in shape but differ in scale, for both policies. This can be explained by the fact that `active-servers` policy is not as effective as the manual placements on the original infrastructure in addition to the influence of performance interference (Figure B.2)
2. Other metrics exhibit very similar distributions. Small differences may be accounted to the number of VMs being slightly smaller in the “replay experiments” due to missing placement data (Figure B.1).

Furthermore, we have had several meetings with both industry and domain experts to discuss the simulator outputs in depth, validate our models and assumptions, and spot inconsistencies. Moreover, we have had proactive communication with the experts about possible issues with the simulator that arose during development, such as unclear observations.

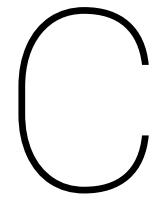
B.3. How to ensure no regression in subsequent simulator versions?

Although we may at one point trust the simulator to produce correct outputs, the addition or modification of functionality in subsequent versions of the simulator may inadvertently affect the output compared to previous versions.

We safeguard against such issues by means of snapshot testing. With snapshot testing, we capture a snapshot of the system outputs and compare it against the outputs produced by subsequent simulator versions. For this test, we consider a downsized variant of the experiments run in this work and capturing the same metrics. These tests execute after every change and ensure that the validity of the simulator outputs is not affected. In case some output changes are intentional, the test failures serve as a double check.

Furthermore, we use assertions in various parts of the simulator to validate internal assumptions. This includes verifying that messages in simulation are not delivered out-of-order and validating that simulated machines do not reach invalid states.

Finally, we employ industry-standard development practices. Every change to the simulator or its extensions requires an independent code review before inclusion in the main code base. In addition, we automatically run for each change static code analysis tools (e.g. linting) to spot common mistakes.



Full Visual Experiment Results

In the figures on the following pages, we visualize the full set of metrics (§6.2.6) for each experiment. We differentiate between select overviews (depicting only the results of a subset of workloads) and summary overviews (aggregating over all workloads).

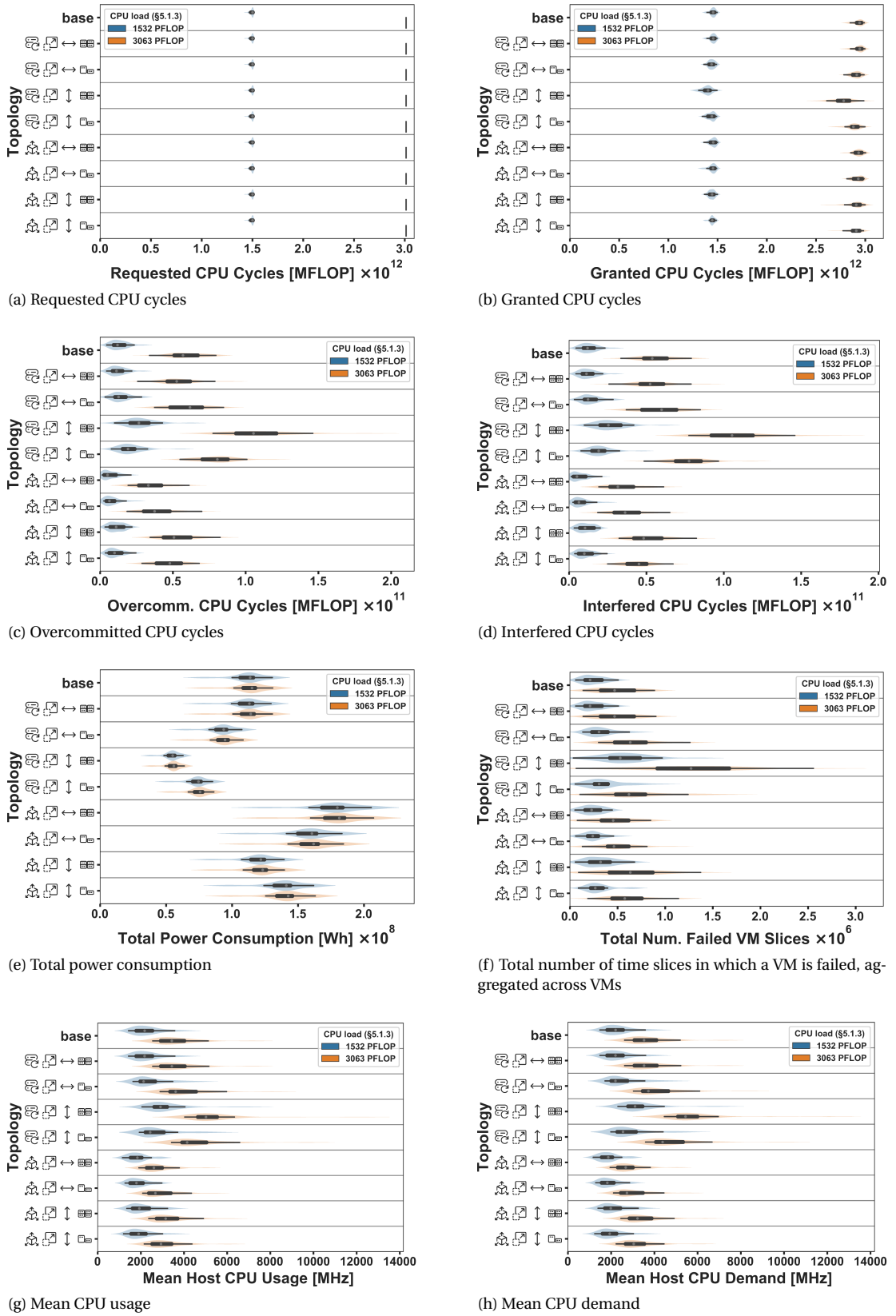
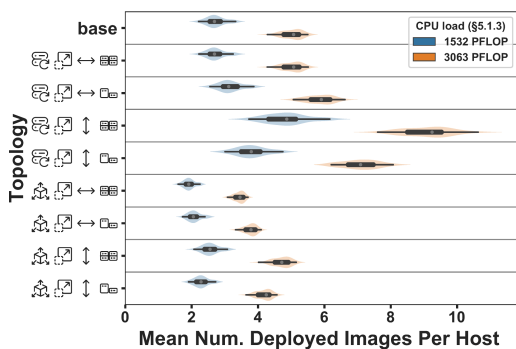
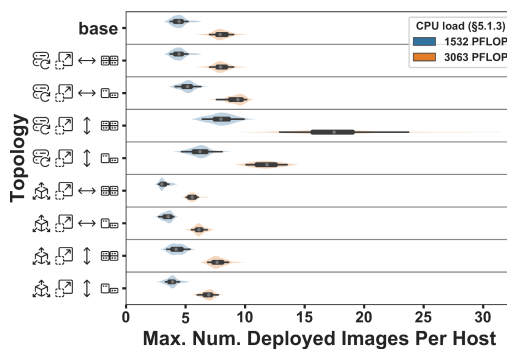


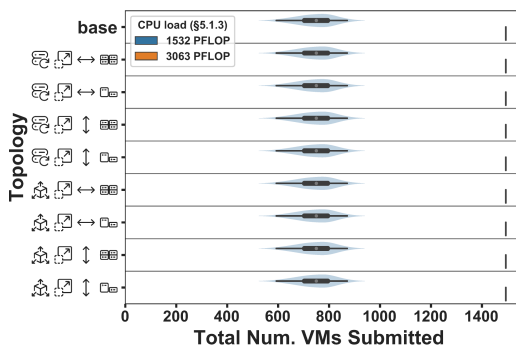
Figure C.1: Performance of different horizontally and vertically expanded topologies, compared across workloads. For a legend of topologies, see Table 6.3. Continued in Figure C.2.



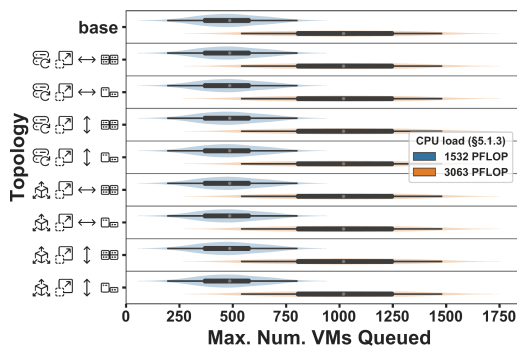
(a) Mean number of VMs per host



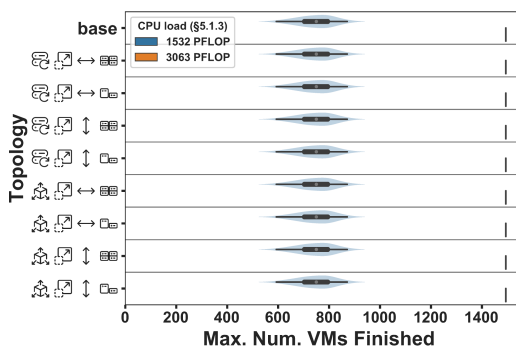
(b) Max number of VMs per host



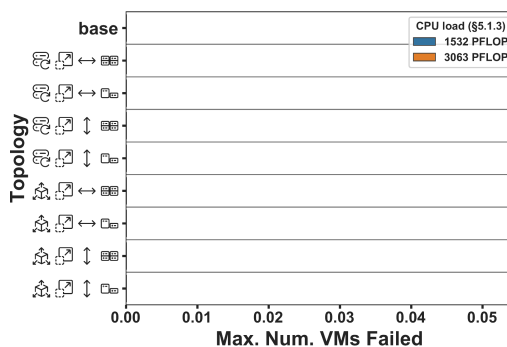
(c) Total VMs Submitted



(d) Total VMs Queued



(e) Total VMs Finished



(f) Total VMs Failed

Figure C.2: Performance of different horizontally and vertically expanded topologies, compared across workloads. For a legend of topologies, see Table 6.3.

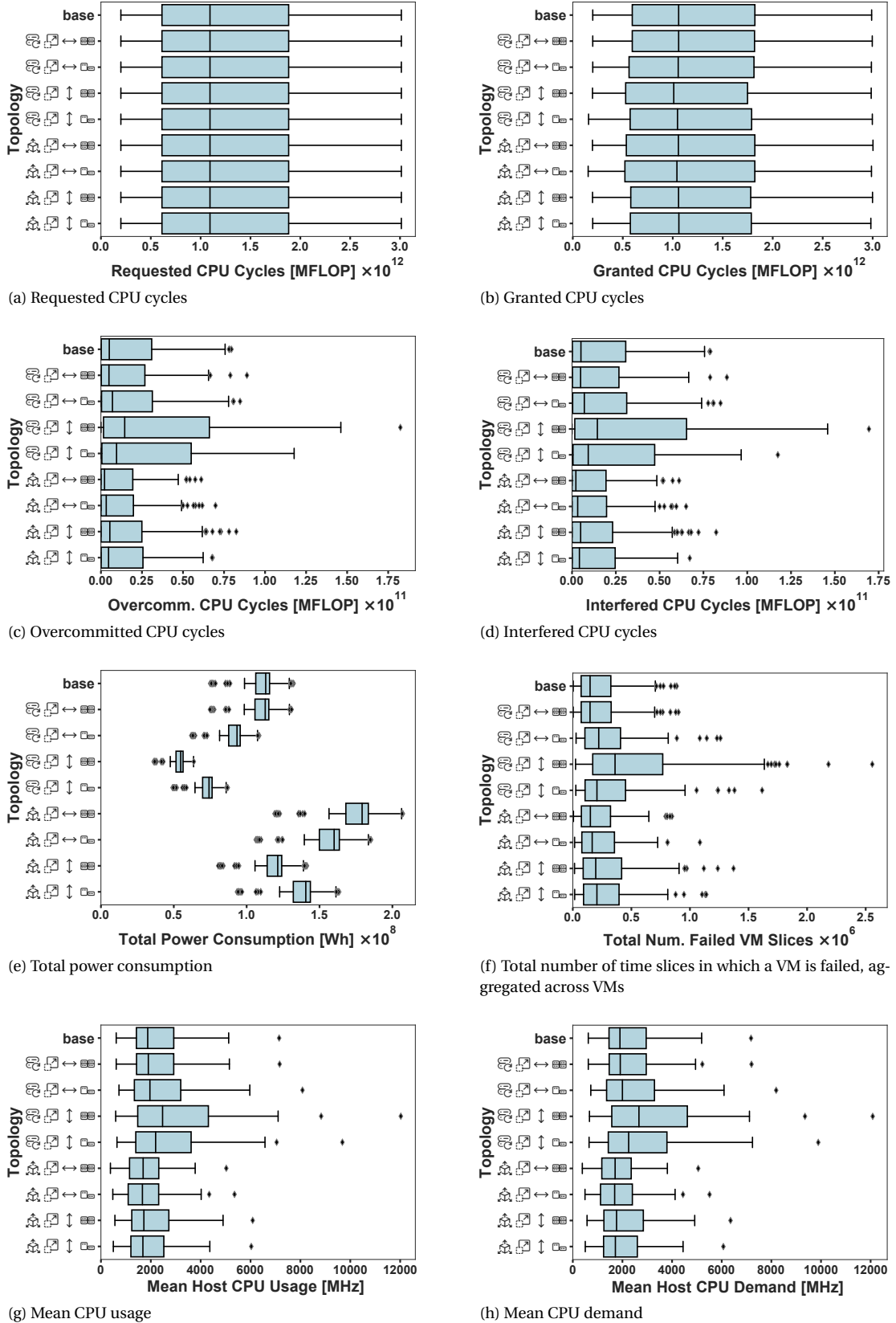
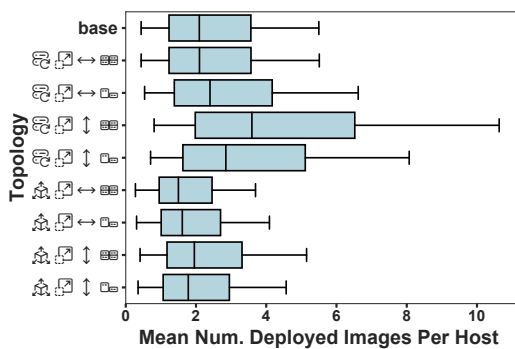
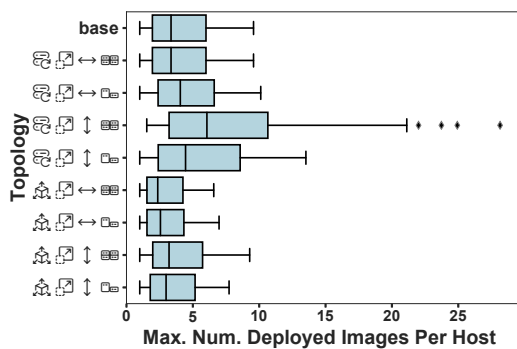


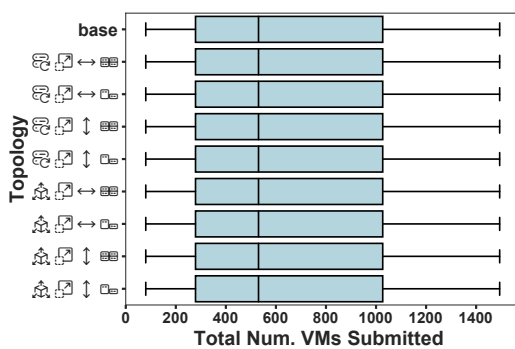
Figure C.3: Performance of different horizontally and vertically expanded topologies, compared across workloads. Results aggregated across the full set of workloads, including workloads not displayed in the more detailed figure. For a legend of topologies, see Table 6.3. Continued in Figure C.4.



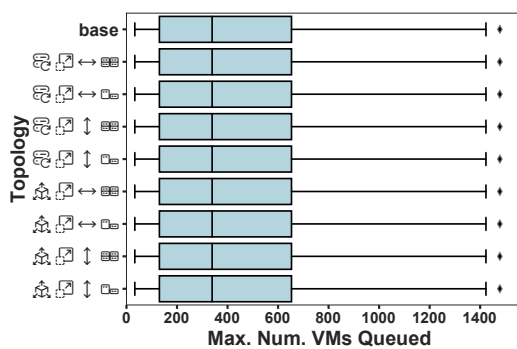
(a) Mean number of VMs per host



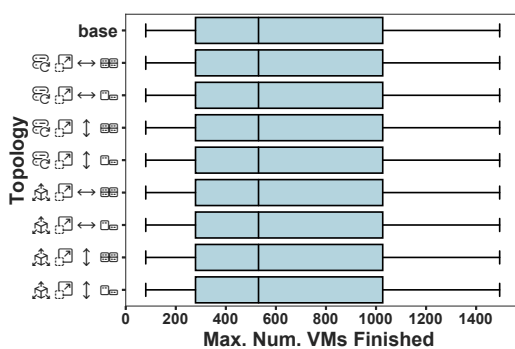
(b) Max number of VMs per host



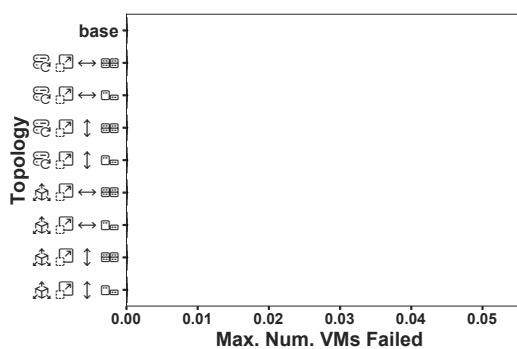
(c) Total VMs Submitted



(d) Total VMs Queued

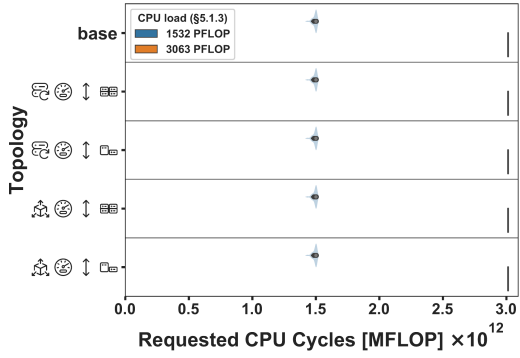


(e) Total VMs Finished

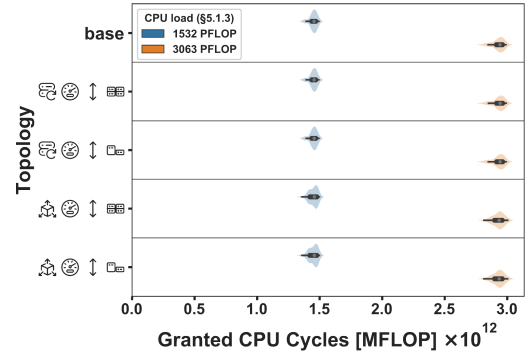


(f) Total VMs Failed

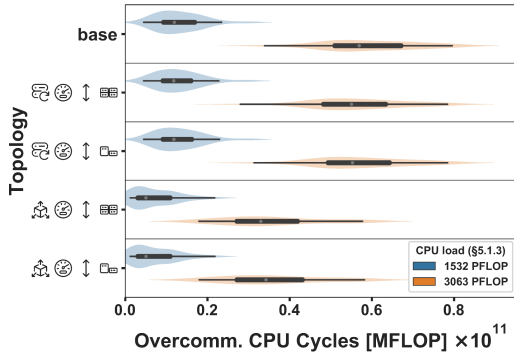
Figure C.4: Performance of different horizontally and vertically expanded topologies, compared across workloads. Results aggregated across the full set of workloads, including workloads not displayed in the more detailed figure. For a legend of topologies, see Table 6.3.



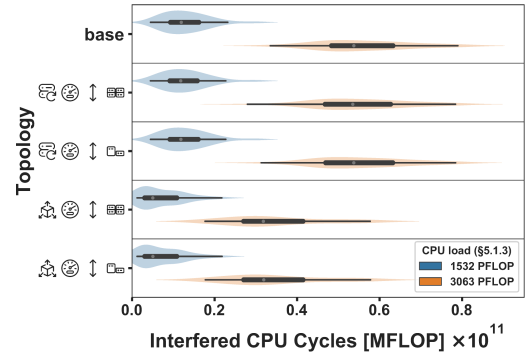
(a) Requested CPU cycles



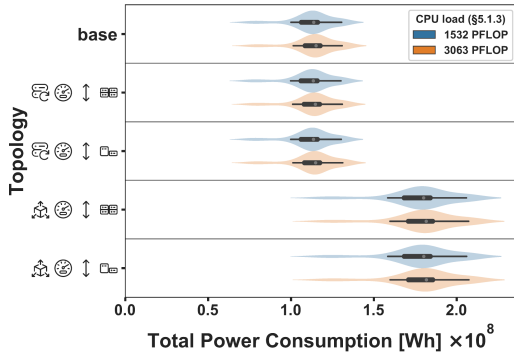
(b) Granted CPU cycles



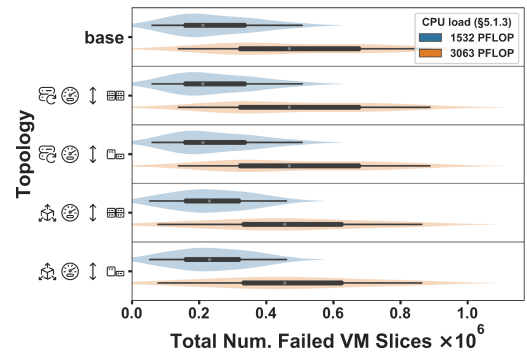
(c) Overcommitted CPU cycles



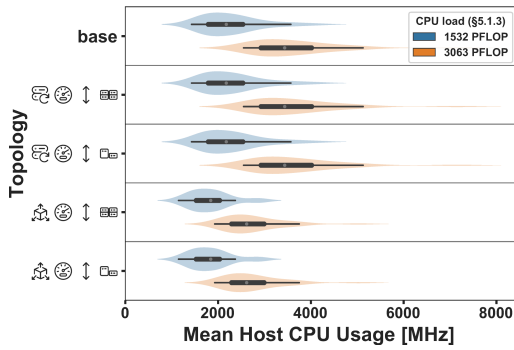
(d) Interfered CPU cycles



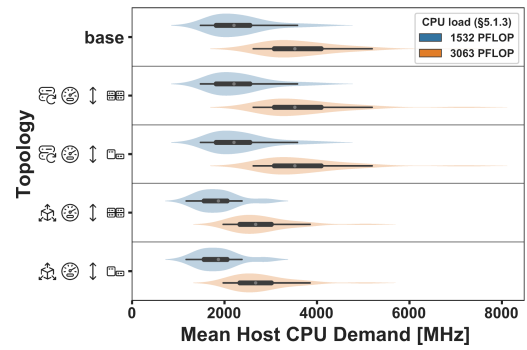
(e) Total power consumption



(f) Total number of time slices in which a VM is failed, aggregated across VMs

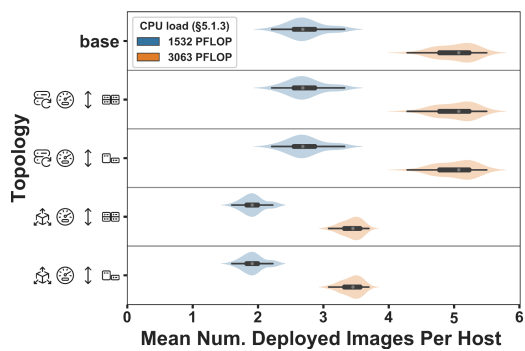


(g) Mean CPU usage



(h) Mean CPU demand

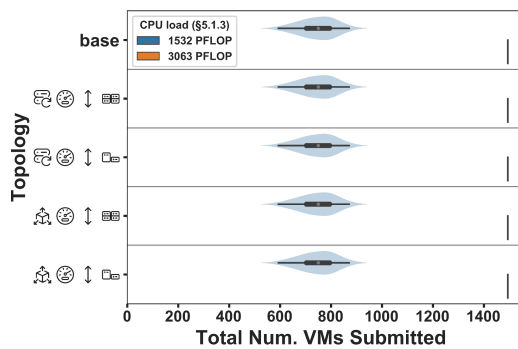
Figure C.5: Performance of different topologies expanded on velocity, compared across workloads. For a legend of topologies, see Table 6.3. Continued in Figure C.6.



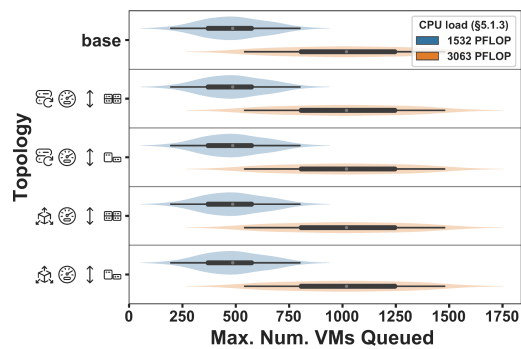
(a) Mean number of VMs per host



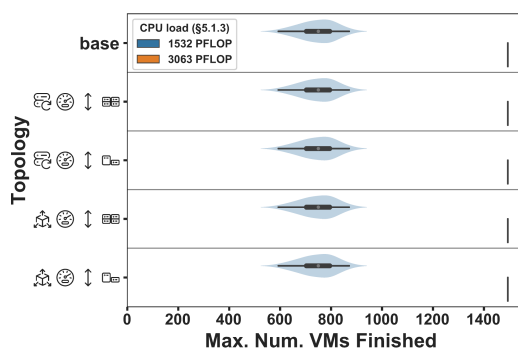
(b) Max number of VMs per host



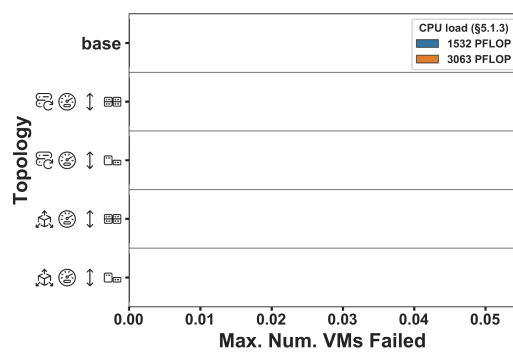
(c) Total VMs Submitted



(d) Total VMs Queued



(e) Total VMs Finished



(f) Total VMs Failed

Figure C.6: Performance of different topologies expanded on velocity, compared across workloads. For a legend of topologies, see Table 6.3.

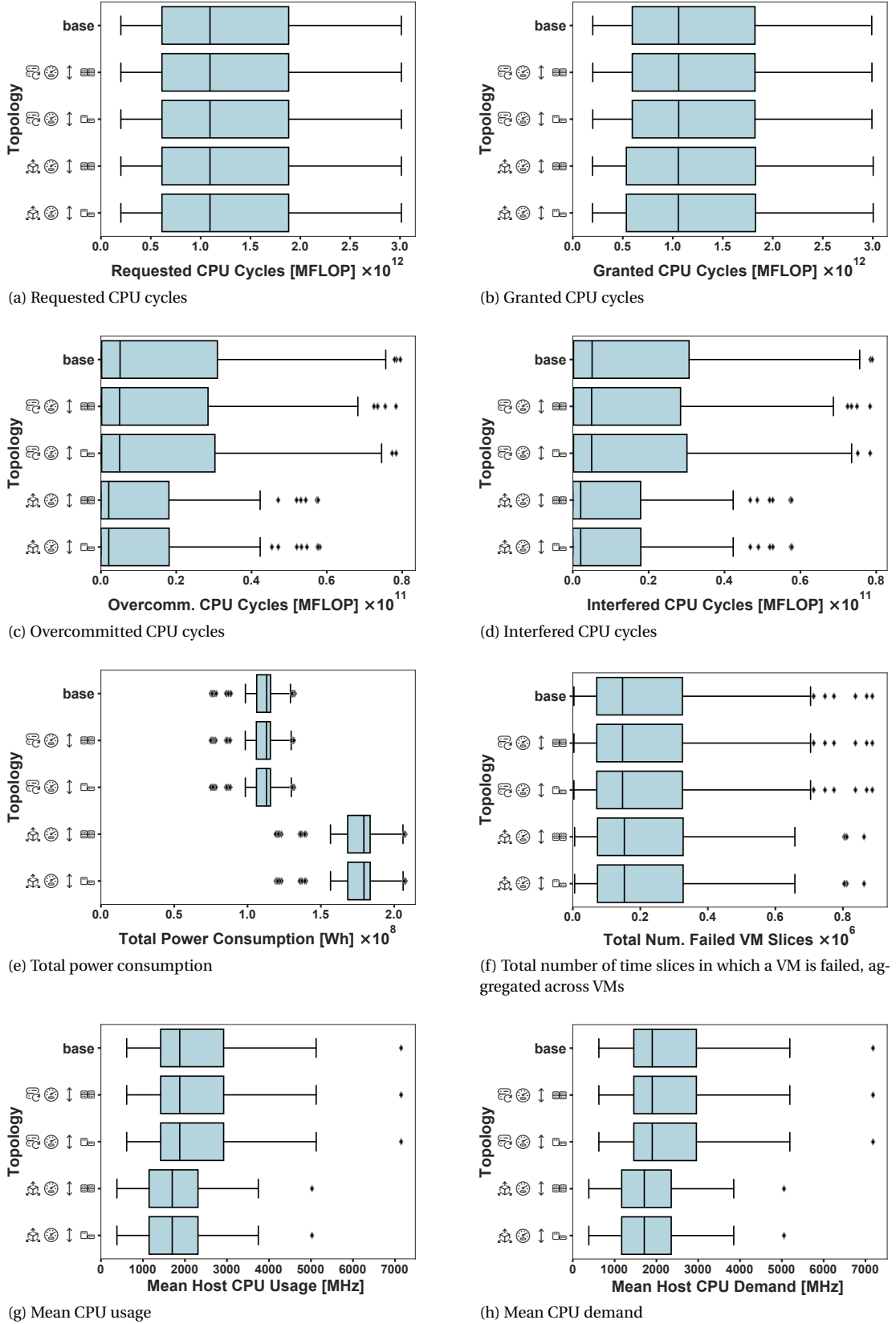
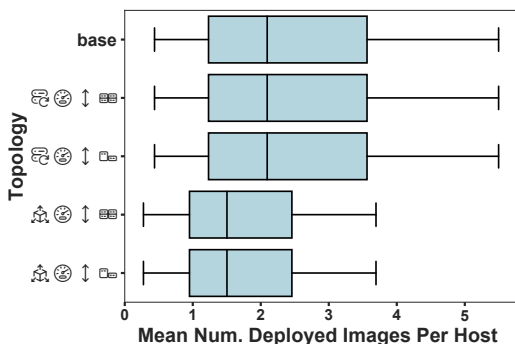
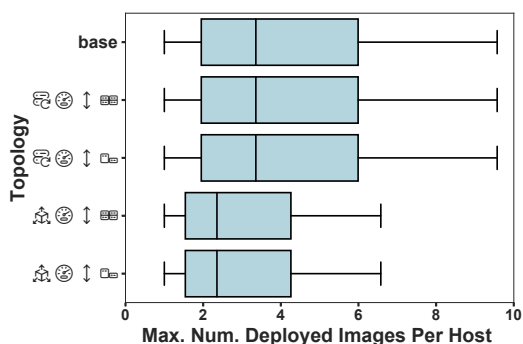


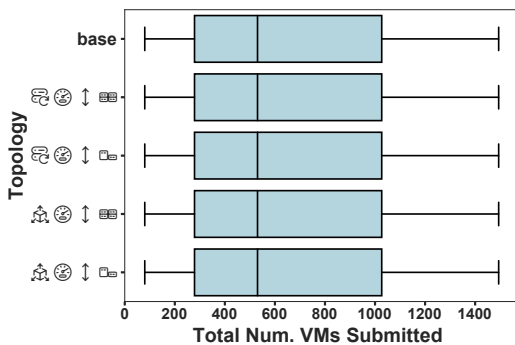
Figure C.7: Performance of different topologies expanded on velocity, compared across workloads. Results aggregated across the full set of workloads, including workloads not displayed in the more detailed figure. For a legend of topologies, see Table 6.3. Continued in Figure C.8.



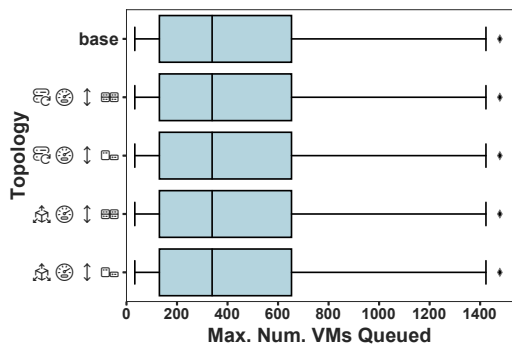
(a) Mean number of VMs per host



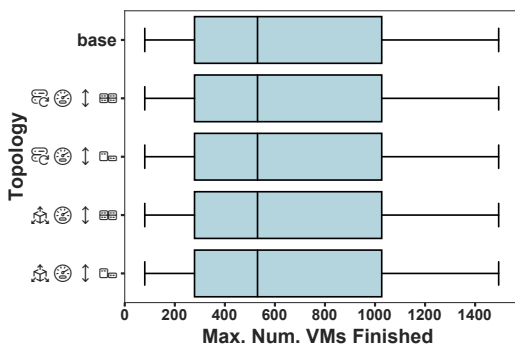
(b) Max number of VMs per host



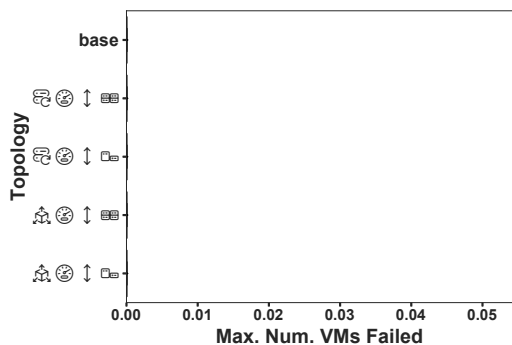
(c) Total VMs Submitted



(d) Total VMs Queued



(e) Total VMs Finished



(f) Total VMs Failed

Figure C.8: Performance of different topologies expanded on velocity, compared across workloads. Results aggregated across the full set of workloads, including workloads not displayed in the more detailed figure. For a legend of topologies, see Table 6.3.

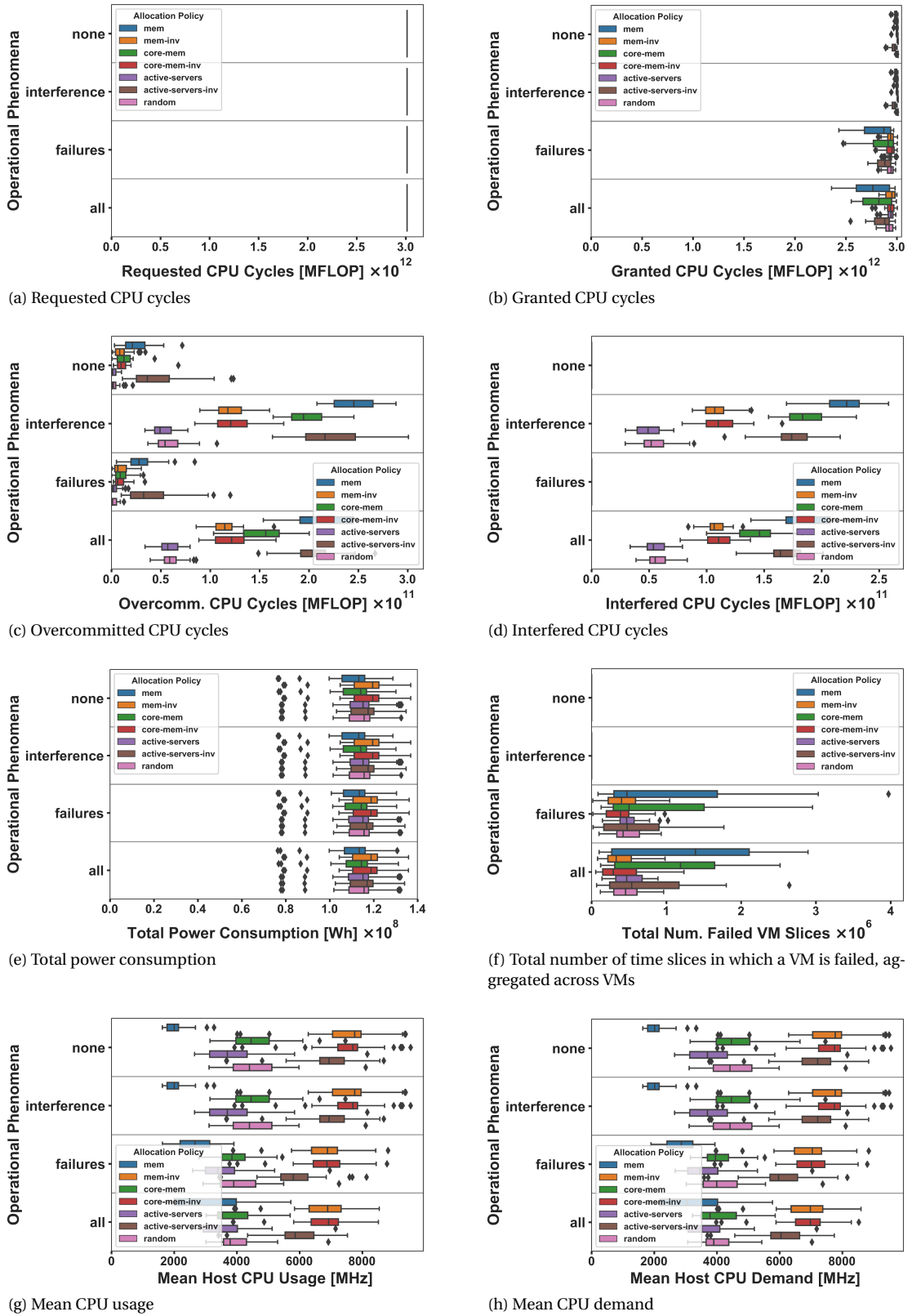
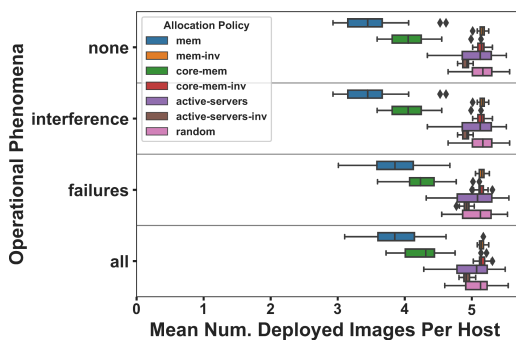
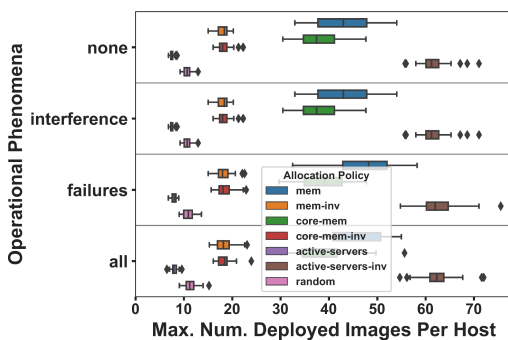


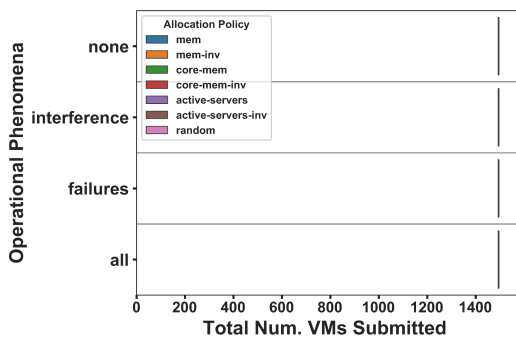
Figure C.9: Impact of operational phenomena and different allocation policies on the base topology. For a legend of topologies, see Table 6.3. Continued in Figure C.10.



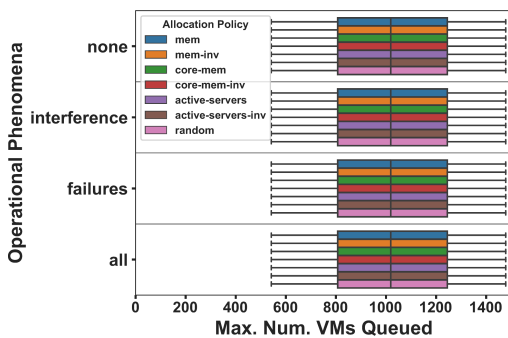
(a) Mean number of VMs per host



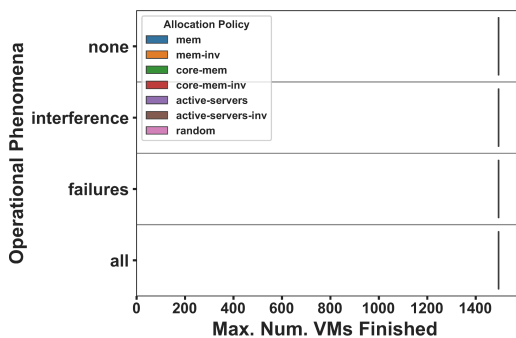
(b) Max number of VMs per host



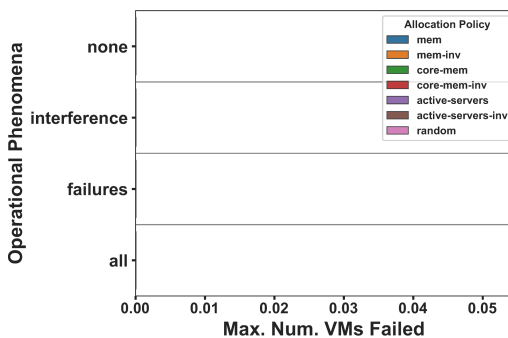
(c) Total VMs Submitted



(d) Total VMs Queued



(e) Total VMs Finished



(f) Total VMs Failed

Figure C.10: Impact of operational phenomena and different allocation policies on the base topology. For a legend of topologies, see Table 6.3.

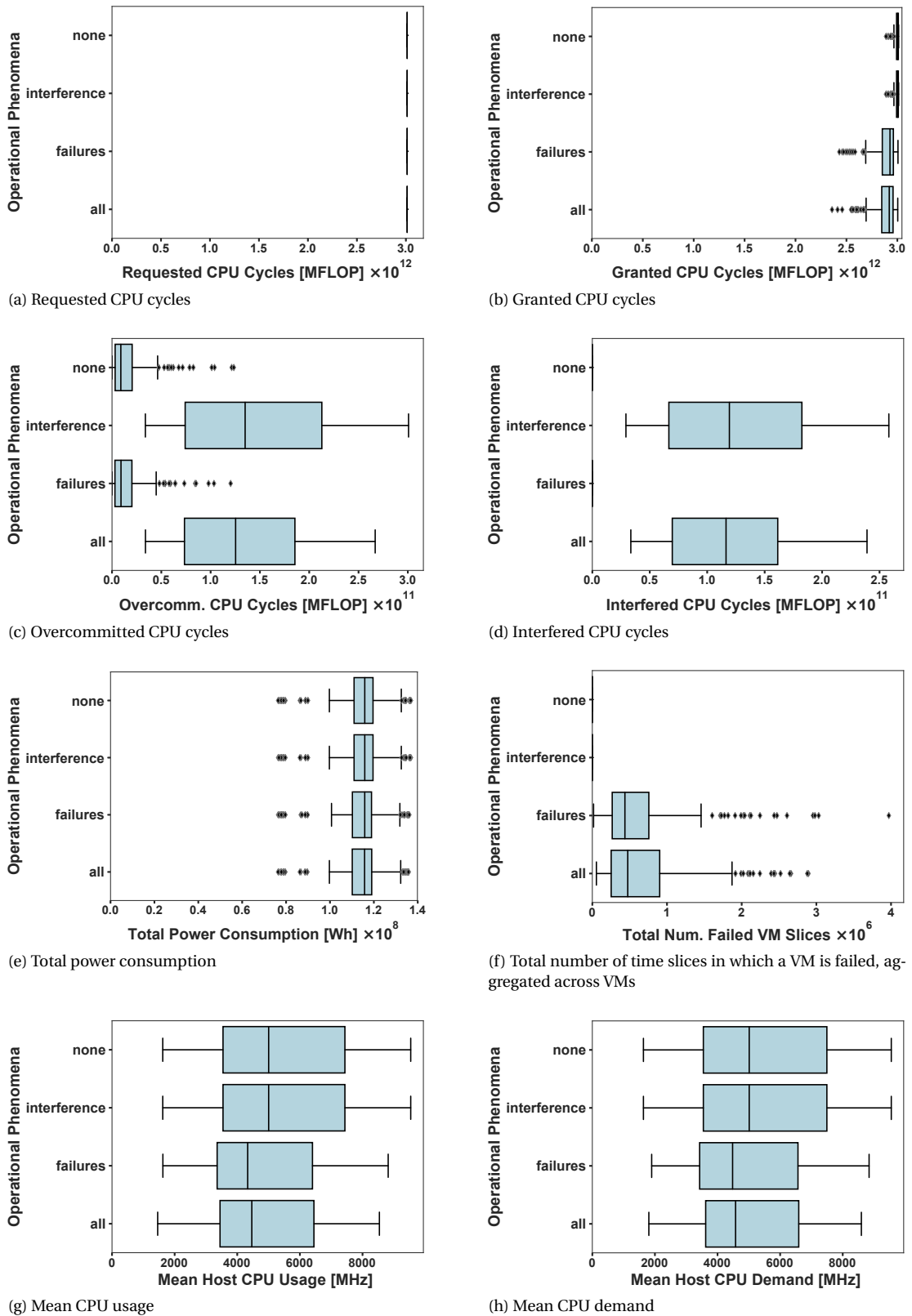
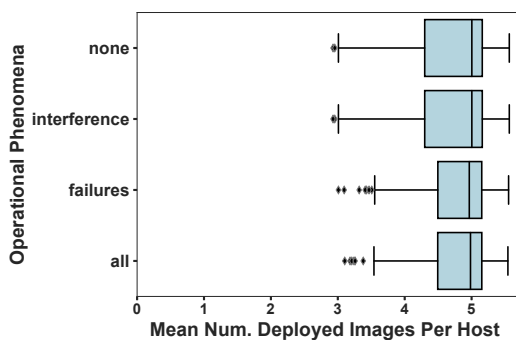
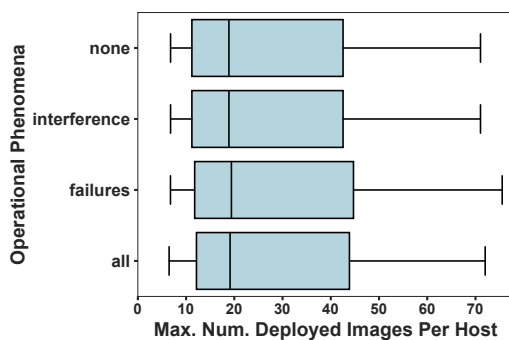


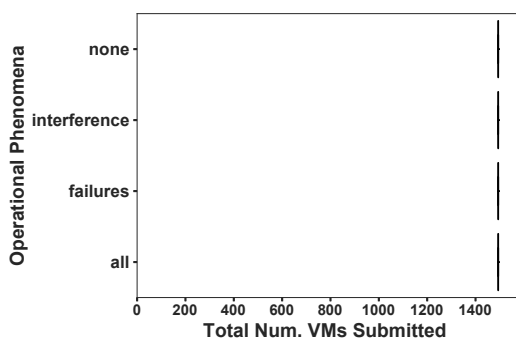
Figure C.11: Impact of operational phenomena and different allocation policies on the base topology. For a legend of topologies, see Table 6.3. Continued in Figure C.12.



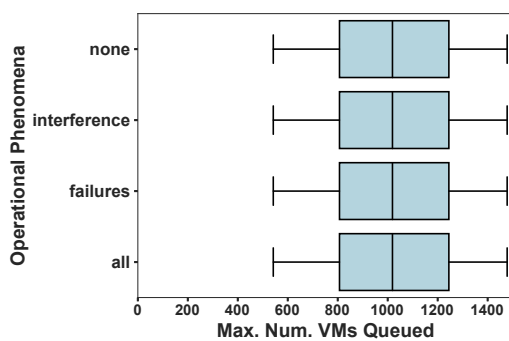
(a) Mean number of VMs per host



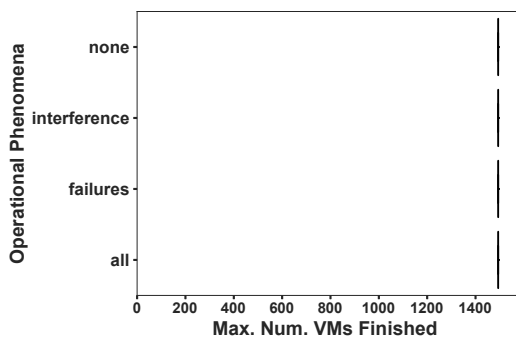
(b) Max number of VMs per host



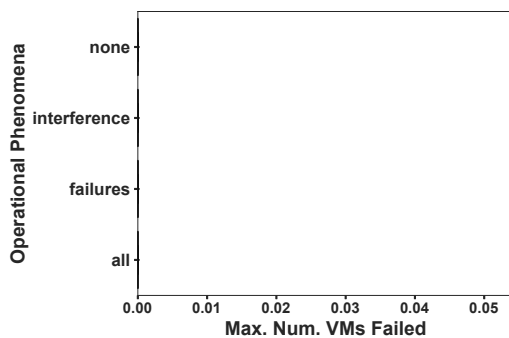
(c) Total VMs Submitted



(d) Total VMs Queued



(e) Total VMs Finished



(f) Total VMs Failed

Figure C.12: Impact of operational phenomena and different allocation policies on the base topology. For a legend of topologies, see Table 6.3.

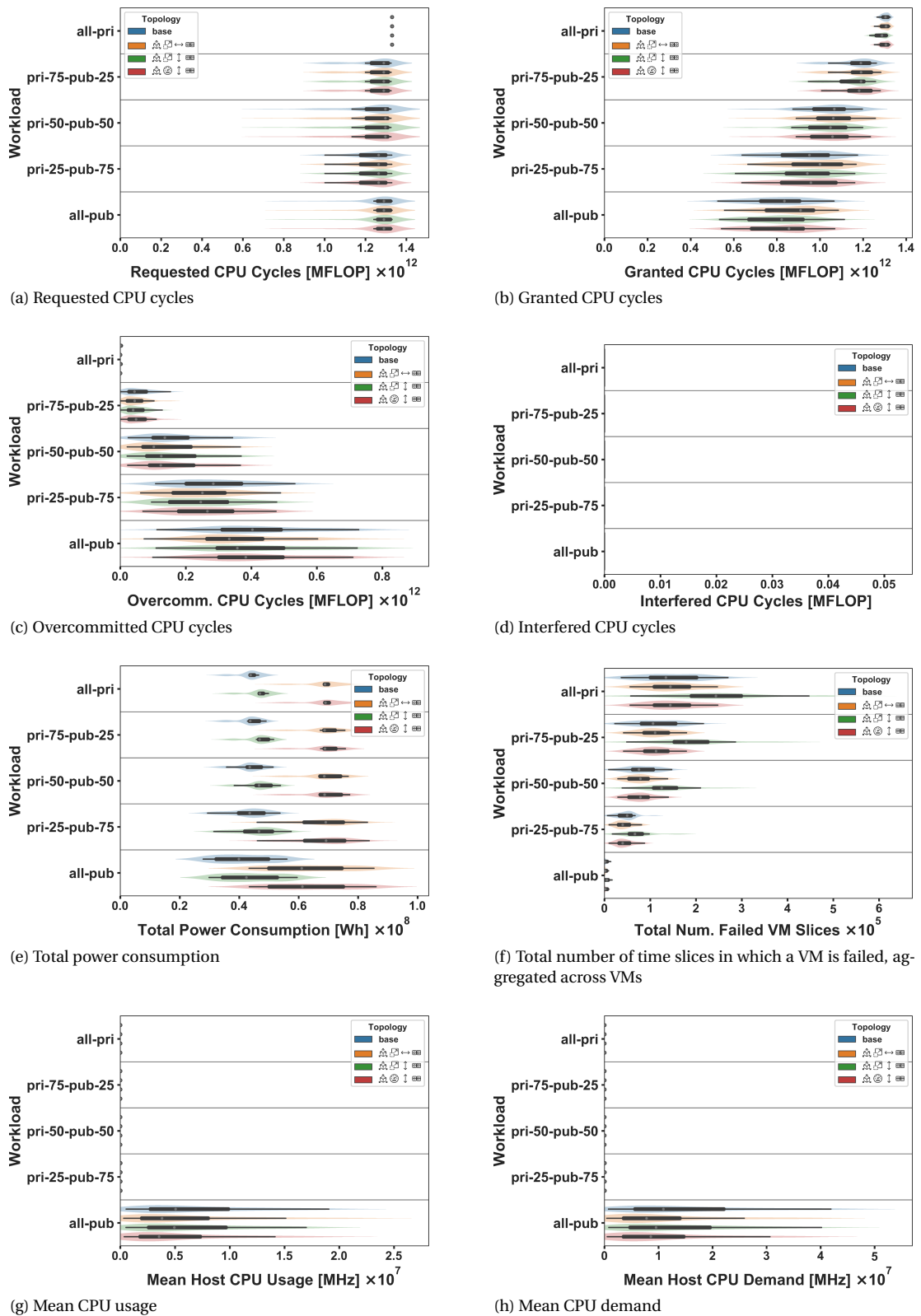
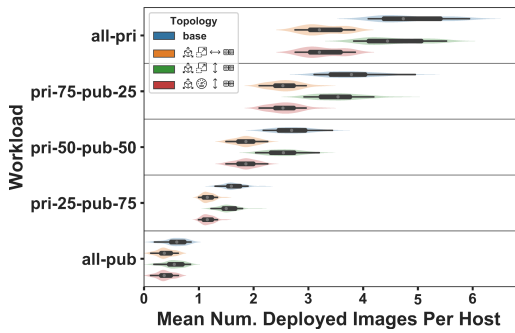
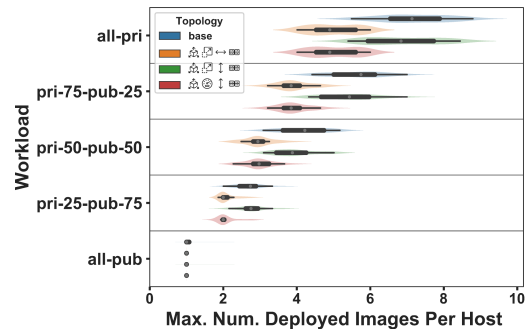


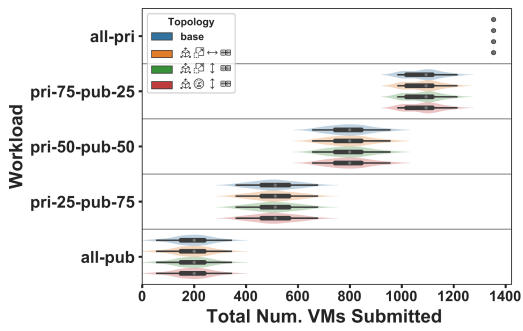
Figure C.13: Impact of a composite workload (consisting of private and public workloads) on different topologies. For a legend of topologies, see Table 6.3. Continued in Figure C.14.



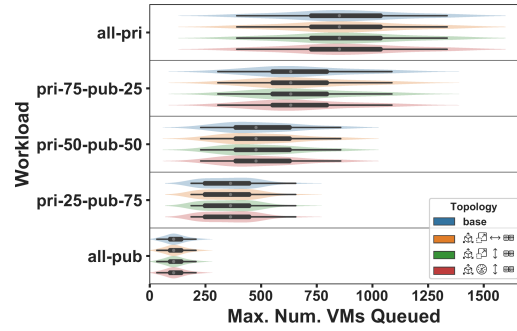
(a) Mean number of VMs per host



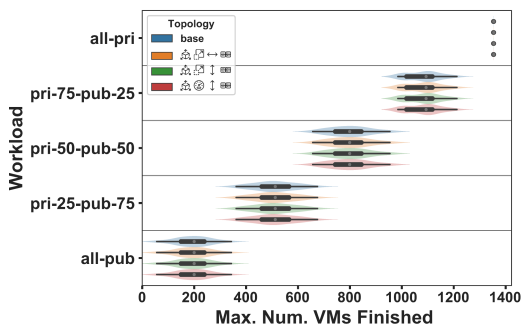
(b) Max number of VMs per host



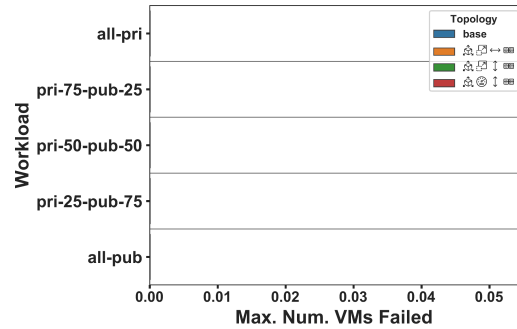
(c) Total VMs Submitted



(d) Total VMs Queued



(e) Total VMs Finished



(f) Total VMs Failed

Figure C.14: Impact of a composite workload (consisting of private and public workloads) on different topologies. For a legend of topologies, see Table 6.3.

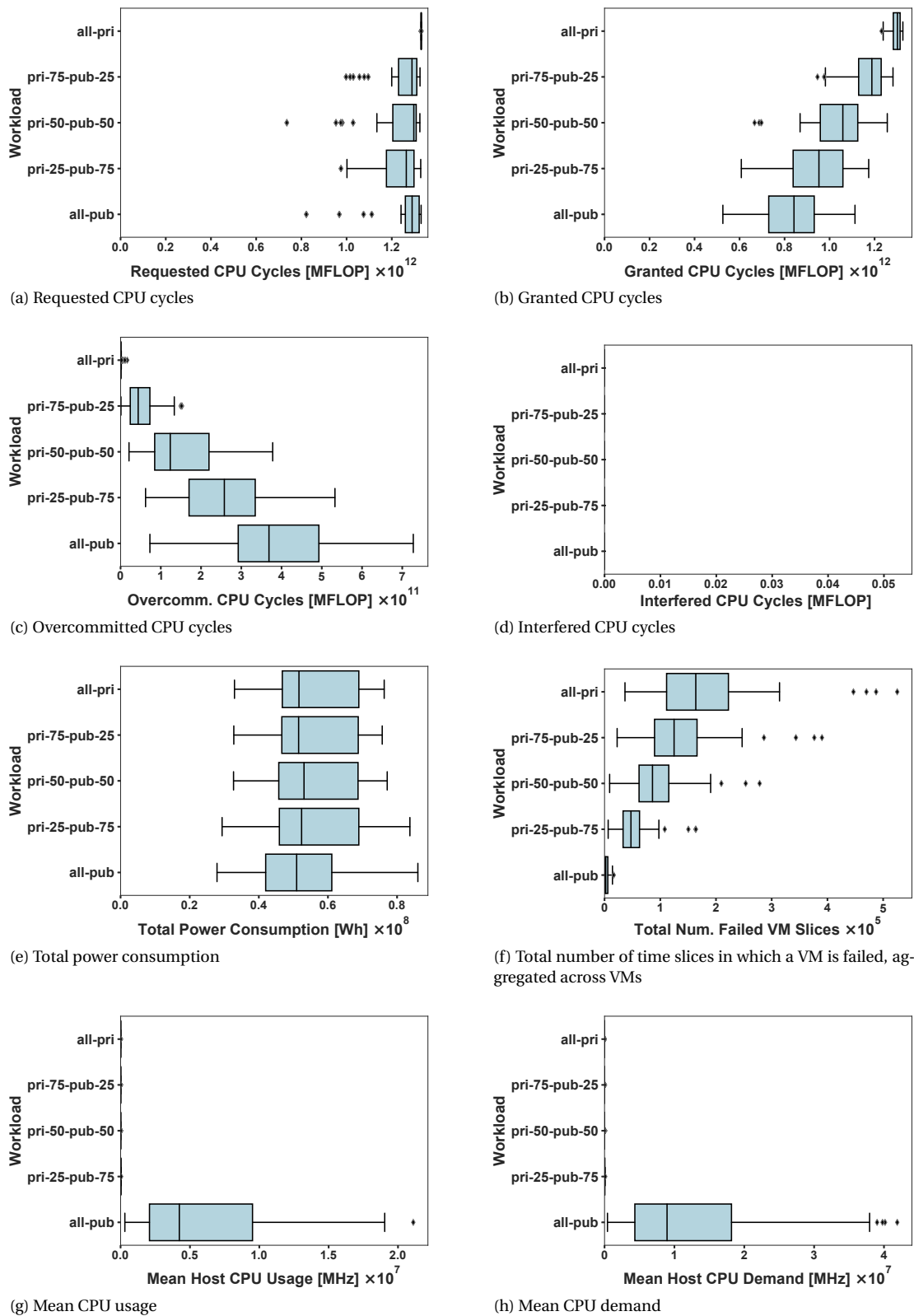
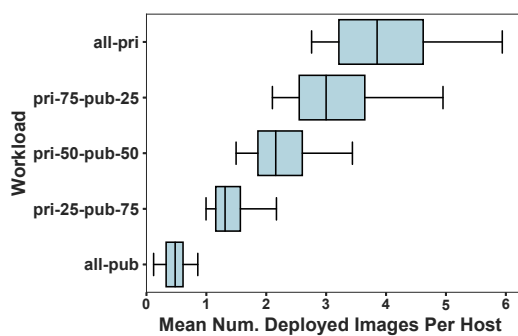
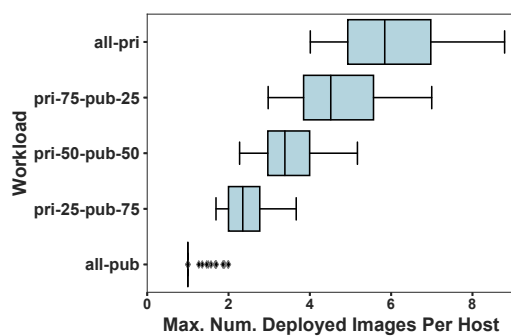


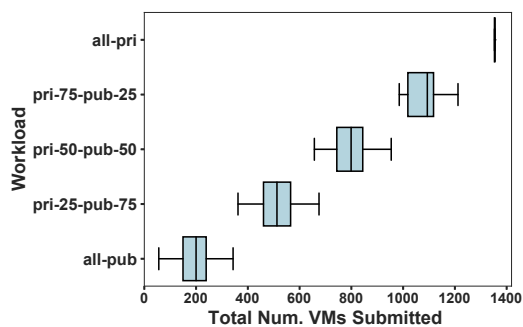
Figure C.15: Impact of a composite workload (consisting of private and public workloads) on different topologies. For a legend of topologies, see Table 6.3. Continued in Figure C.16.



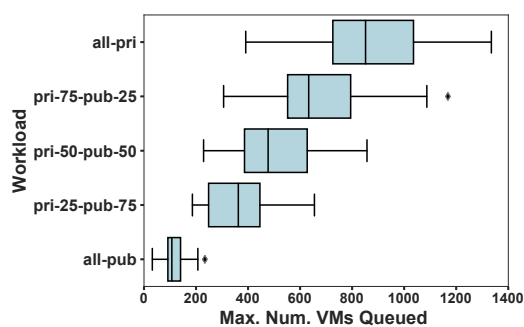
(a) Mean number of VMs per host



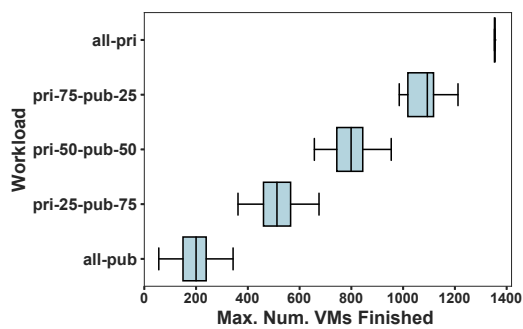
(b) Max number of VMs per host



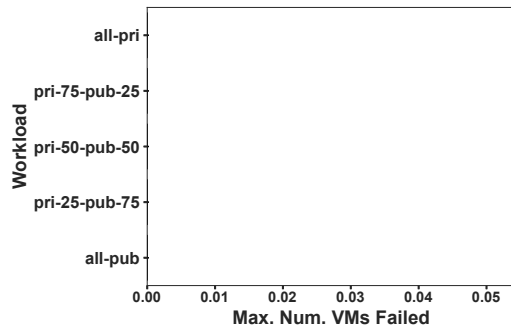
(c) Total VMs Submitted



(d) Total VMs Queued



(e) Total VMs Finished



(f) Total VMs Failed

Figure C.16: Impact of a composite workload (consisting of private and public workloads) on different topologies. For a legend of topologies, see Table 6.3.

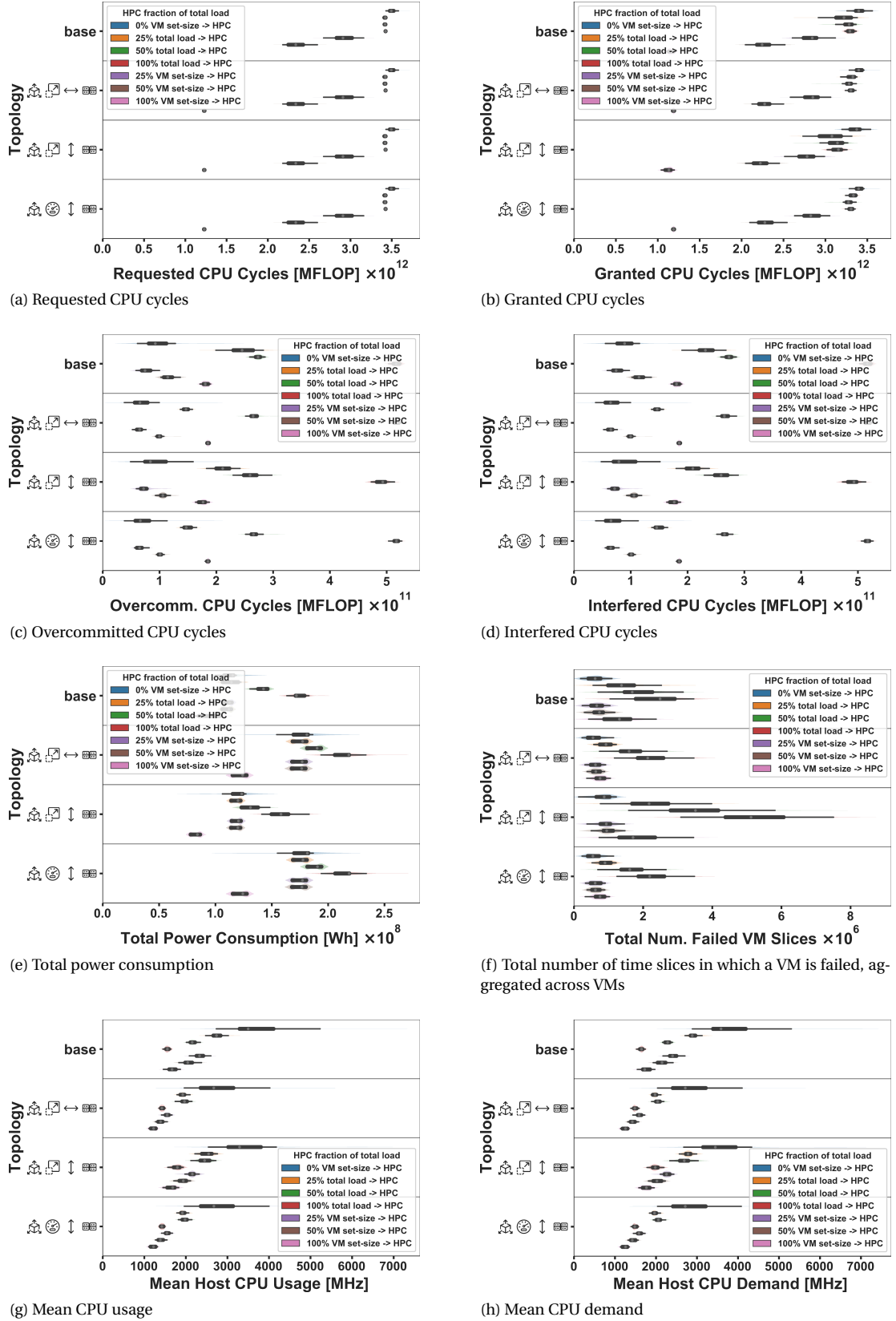
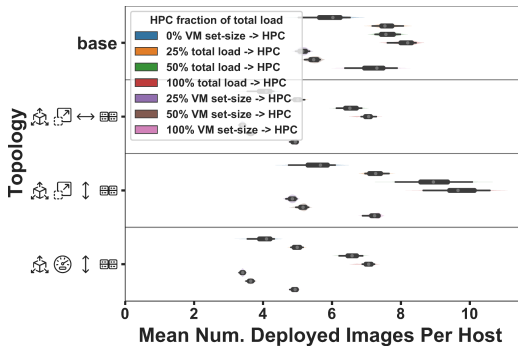
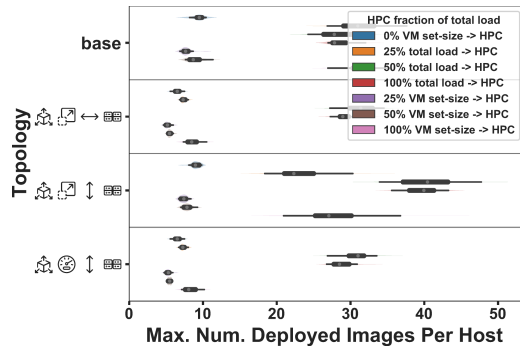


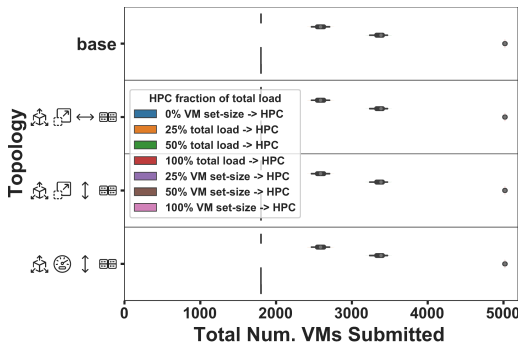
Figure C.17: Performance of increasing an increased relative intensity of HPC workloads compared to others, on different topologies. For a legend of topologies, see Table 6.3. Continued in Figure C.18.



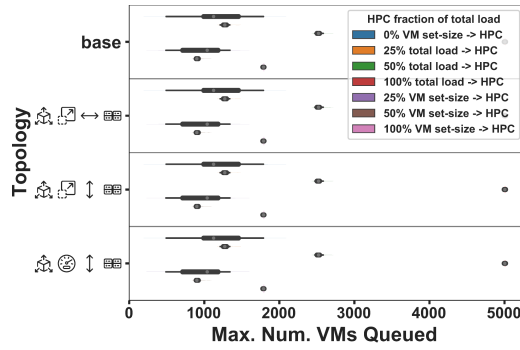
(a) Mean number of VMs per host



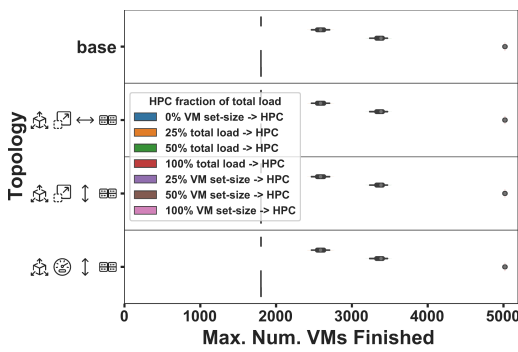
(b) Max number of VMs per host



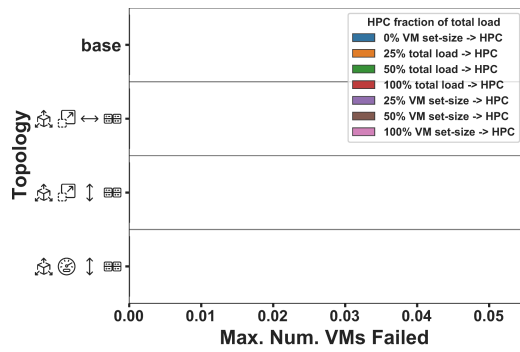
(c) Total VMs Submitted



(d) Total VMs Queued



(e) Total VMs Finished



(f) Total VMs Failed

Figure C.18: Performance of increasing an increased relative intensity of HPC workloads compared to others, on different topologies. For a legend of topologies, see Table 6.3.

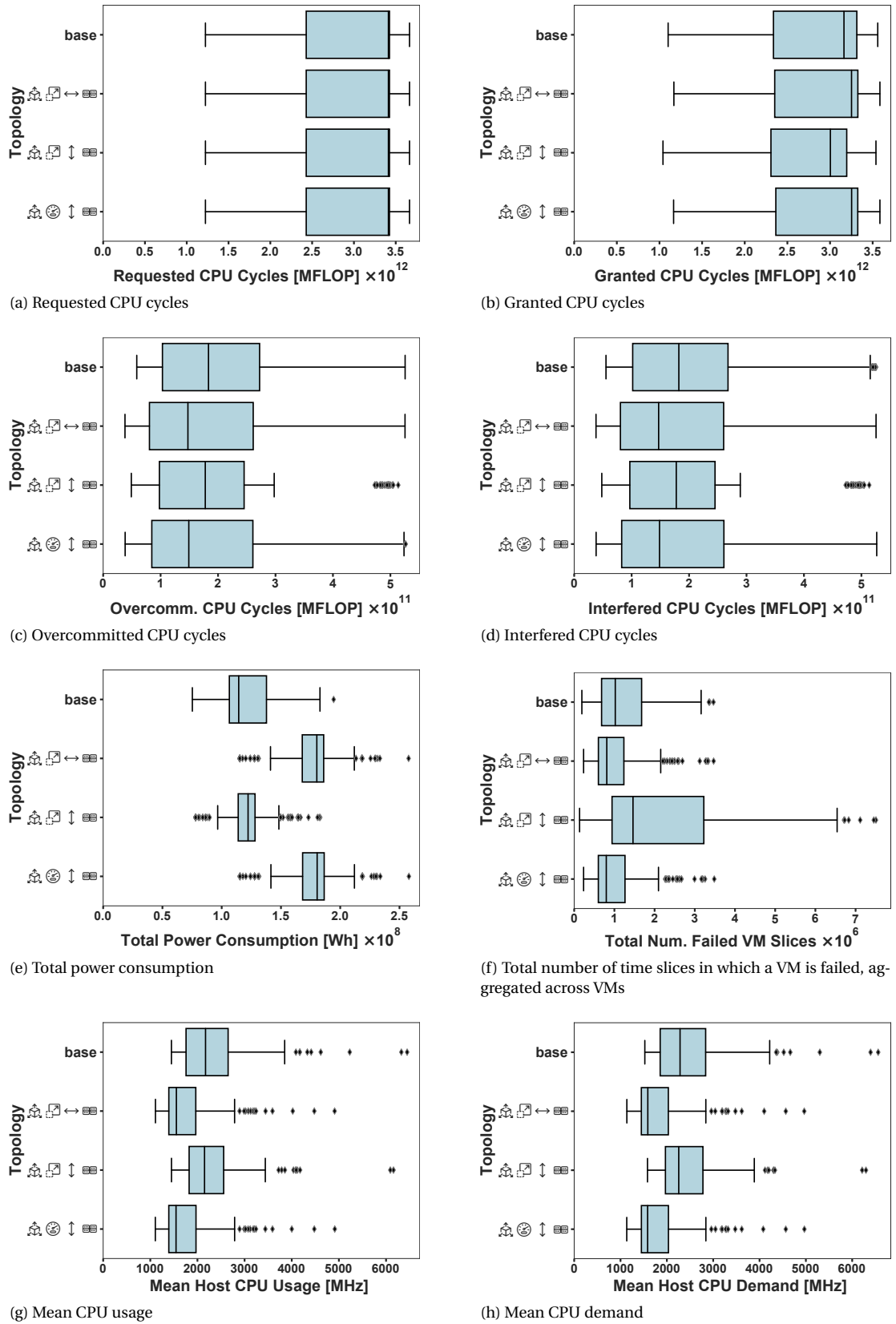
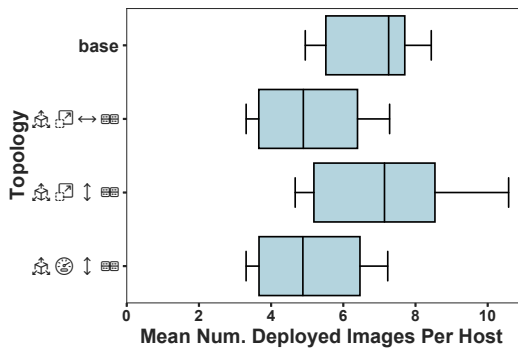
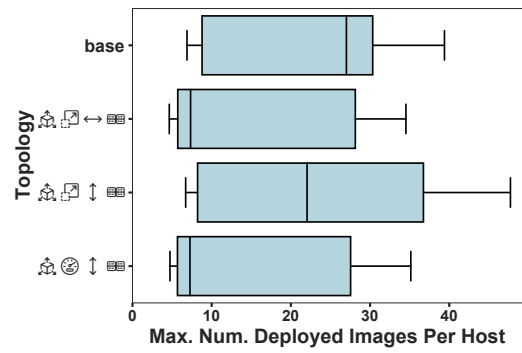


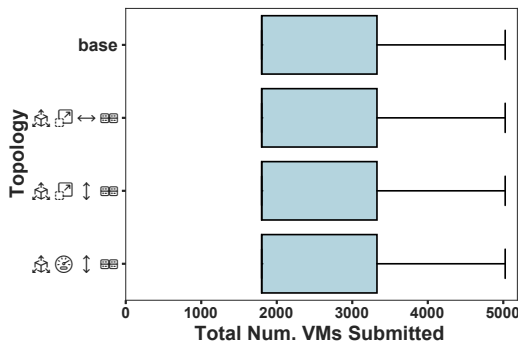
Figure C.19: Performance of increasing an increased relative intensity of HPC workloads compared to others, on different topologies. For a legend of topologies, see Table 6.3. Continued in Figure C.20.



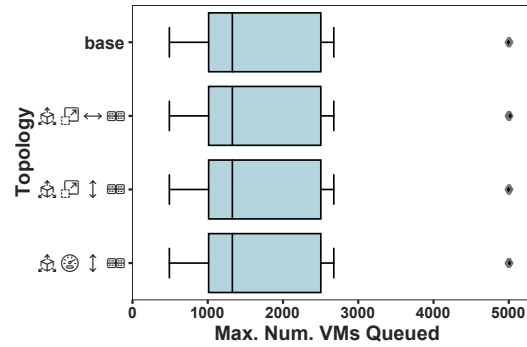
(a) Mean number of VMs per host



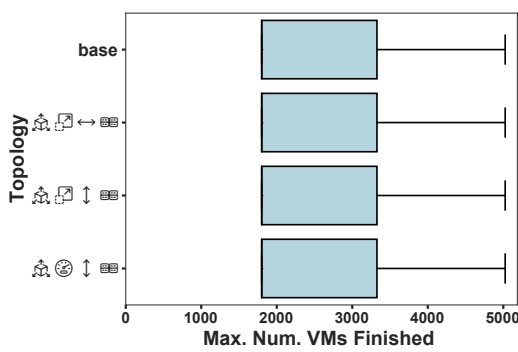
(b) Max number of VMs per host



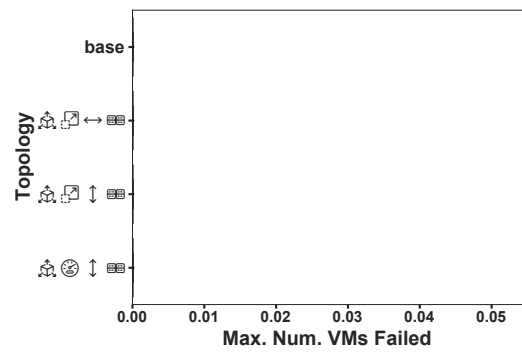
(c) Total VMs Submitted



(d) Total VMs Queued

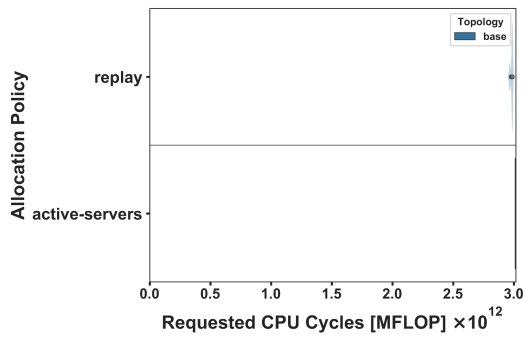


(e) Total VMs Finished

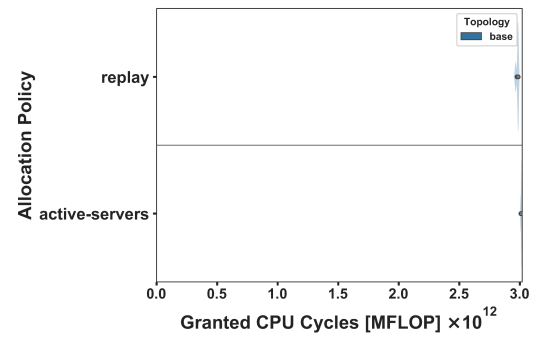


(f) Total VMs Failed

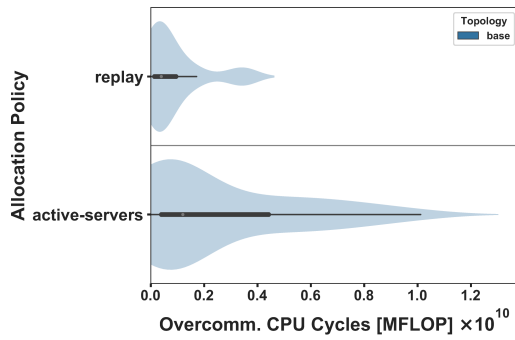
Figure C.20: Performance of increasing an increased relative intensity of HPC workloads compared to others, on different topologies. For a legend of topologies, see Table 6.3.



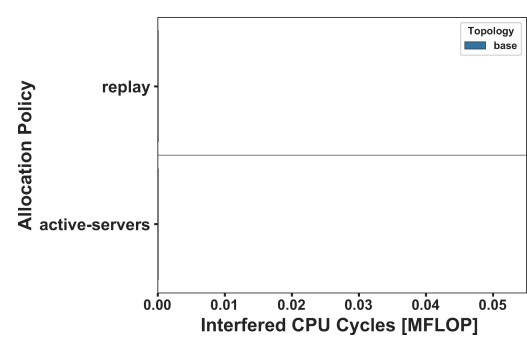
(a) Requested CPU cycles



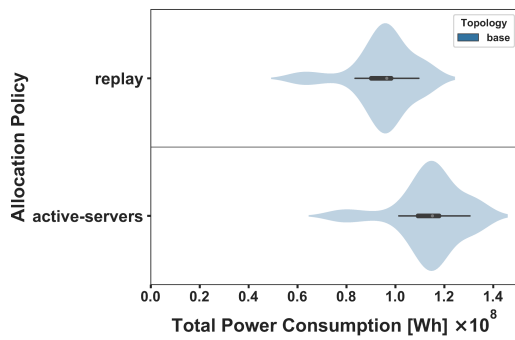
(b) Granted CPU cycles



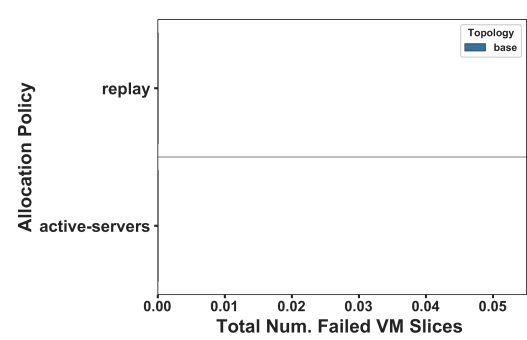
(c) Overcommitted CPU cycles



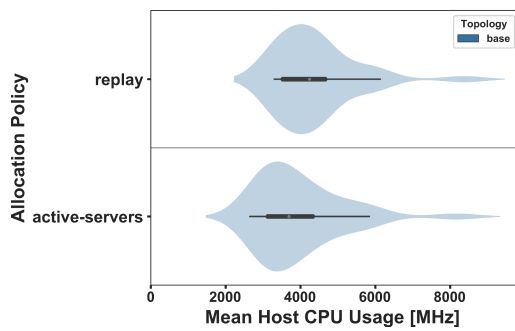
(d) Interfered CPU cycles



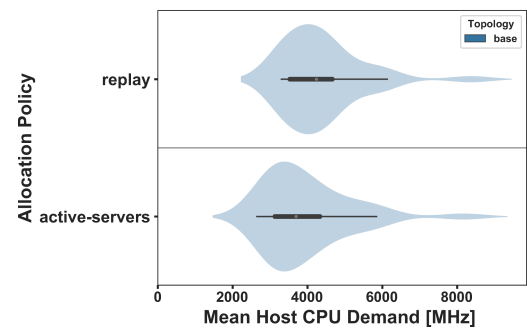
(e) Total power consumption



(f) Total number of time slices in which a VM is failed, aggregated across VMs

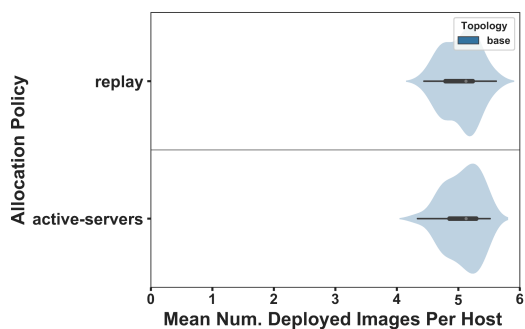


(g) Mean CPU usage

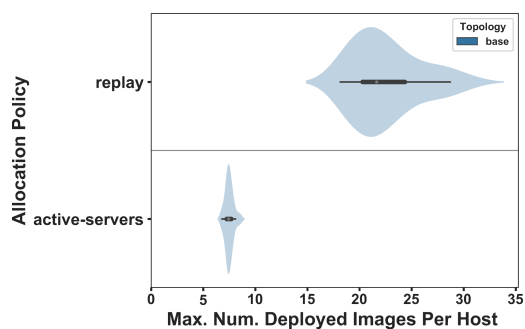


(h) Mean CPU demand

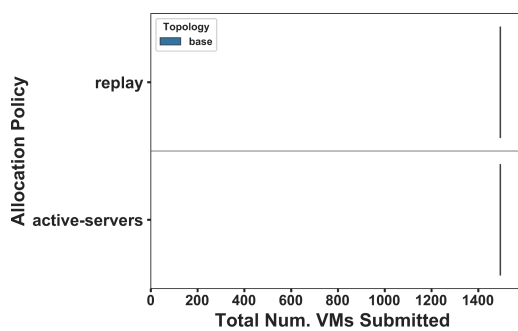
Figure C.21: Validation with a replay policy, copying the exact cluster assignment of the original deployment. For a legend of topologies, see Table 6.3. Continued in Figure C.22.



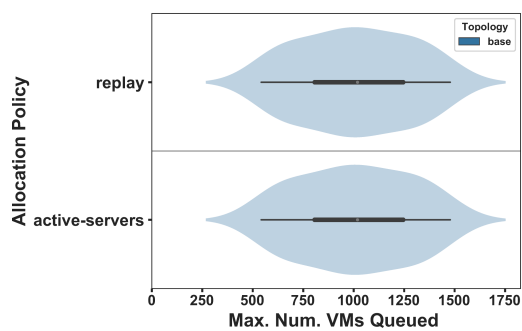
(a) Mean number of VMs per host



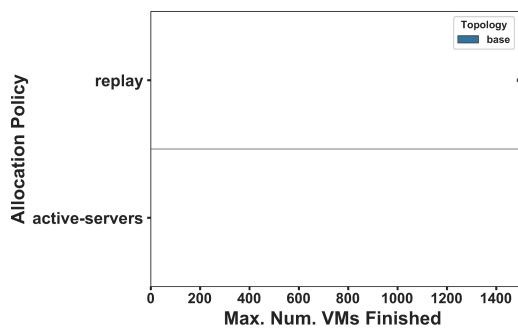
(b) Max number of VMs per host



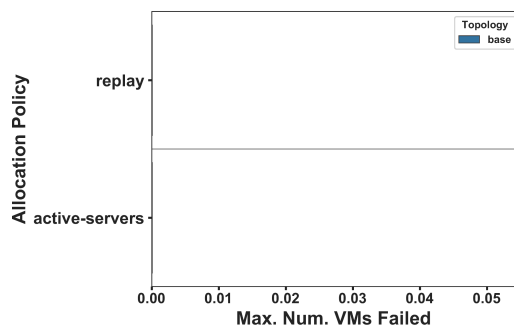
(c) Total VMs Submitted



(d) Total VMs Queued

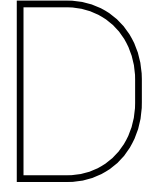


(e) Total VMs Finished



(f) Total VMs Failed

























Figure C.22: Validation with a replay policy, copying the exact cluster assignment of the original deployment. For a legend of topologies, see Table 6.3.



Full Tabular Experiment Results

In the tables on the following pages, we list the full results of two of the most important metrics (defined in §6.2.6) for each experiment: overcommission and power consumption.

Table D.1: Results of the horizontal vs. vertical scaling experiment, with key metrics and their mean and standard deviation.

Topology	Workload	Op. Phen.	Alloc. Policy	Overcommission [MFLOP]		Power Consumption [Wh]	
				median	std	median	std
base	306 PFLOP	all	active-servers	0	7.653e+07	1.106e+08	1.343e+07
base	766 PFLOP	all	active-servers	1.059e+09	1.514e+09	1.128e+08	1.288e+07
base	1532 PFLOP	all	active-servers	1.192e+10	5.622e+09	1.137e+08	1.32e+07
base	3063 PFLOP	all	active-servers	5.691e+10	1.131e+10	1.151e+08	1.32e+07
	306 PFLOP	all	active-servers	0	7.655e+07	1.105e+08	1.341e+07
	766 PFLOP	all	active-servers	8.603e+08	1.492e+09	1.125e+08	1.285e+07
	1532 PFLOP	all	active-servers	1.124e+10	5.613e+09	1.131e+08	1.314e+07
	3063 PFLOP	all	active-servers	5.261e+10	1.212e+10	1.142e+08	1.315e+07
	306 PFLOP	all	active-servers	7.288e+05	2.182e+08	9.139e+07	1.109e+07
	766 PFLOP	all	active-servers	1.678e+09	3.129e+09	9.311e+07	1.064e+07
	1532 PFLOP	all	active-servers	1.263e+10	6.379e+09	9.365e+07	1.087e+07
	3063 PFLOP	all	active-servers	6.172e+10	1.377e+10	9.466e+07	1.086e+07
	306 PFLOP	all	active-servers	1.466e+08	1.348e+09	5.327e+07	6.447e+06
	766 PFLOP	all	active-servers	5.407e+09	4.606e+09	5.443e+07	6.197e+06
	1532 PFLOP	all	active-servers	2.661e+10	1.035e+10	5.498e+07	6.326e+06
	3063 PFLOP	all	active-servers	1.054e+11	2.293e+10	5.585e+07	6.351e+06
	306 PFLOP	all	active-servers	4.594e+07	4.369e+08	7.239e+07	8.774e+06
	766 PFLOP	all	active-servers	3.174e+09	3.518e+09	7.388e+07	8.423e+06
	1532 PFLOP	all	active-servers	1.915e+10	8.997e+09	7.455e+07	8.615e+06
	3063 PFLOP	all	active-servers	8.054e+10	1.323e+10	7.575e+07	8.638e+06
	306 PFLOP	all	active-servers	0	0	1.757e+08	2.133e+07
	766 PFLOP	all	active-servers	2.351e+08	2.375e+09	1.79e+08	2.048e+07
	1532 PFLOP	all	active-servers	5.07e+09	5.265e+09	1.798e+08	2.093e+07
	3063 PFLOP	all	active-servers	3.311e+10	1.195e+10	1.813e+08	2.095e+07
	306 PFLOP	all	active-servers	0	6.543e+04	1.567e+08	1.904e+07
	766 PFLOP	all	active-servers	3.597e+08	3.238e+09	1.597e+08	1.826e+07
	1532 PFLOP	all	active-servers	6.354e+09	5.345e+09	1.605e+08	1.866e+07
	3063 PFLOP	all	active-servers	3.74e+10	1.284e+10	1.618e+08	1.868e+07

Continued on next page

Table D.1: Results of the horizontal vs. vertical scaling experiment, with key metrics and their mean and standard deviation.

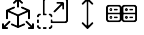
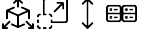
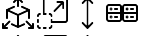
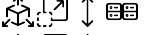
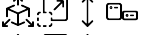
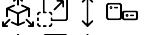
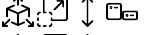
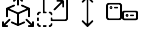
Topology	Workload	Op. Phen.	Alloc. Policy	Overcommission [MFLOP]		Power Consumption [Wh]	
				median	std	median	std
	306 PFLOP	all	active-servers	0	4.804e+06	1.187e+08	1.442e+07
	766 PFLOP	all	active-servers	1.265e+09	2.186e+09	1.21e+08	1.383e+07
	1532 PFLOP	all	active-servers	1.098e+10	5.14e+09	1.218e+08	1.412e+07
	3063 PFLOP	all	active-servers	5.072e+10	1.248e+10	1.232e+08	1.414e+07
	306 PFLOP	all	active-servers	0	3.796e+07	1.377e+08	1.673e+07
	766 PFLOP	all	active-servers	8.684e+08	1.667e+09	1.403e+08	1.606e+07
	1532 PFLOP	all	active-servers	9.854e+09	5.417e+09	1.411e+08	1.641e+07
	3063 PFLOP	all	active-servers	4.787e+10	1.087e+10	1.425e+08	1.644e+07

Table D.2: Results of velocity experiment, with key metrics and their mean and standard deviation.

































Topology	Workload	Op. Phen.	Alloc. Policy	Overcommission [MFLOP]		Power Consumption [Wh]		
				median	std	median	std	
	base	306 PFLOP	all	active-servers	0	7.653e+07	1.106e+08	1.343e+07
	base	766 PFLOP	all	active-servers	1.059e+09	1.514e+09	1.128e+08	1.288e+07
	base	1532 PFLOP	all	active-servers	1.192e+10	5.622e+09	1.137e+08	1.32e+07
	base	3063 PFLOP	all	active-servers	5.691e+10	1.131e+10	1.151e+08	1.32e+07
		306 PFLOP	all	active-servers	0	7.656e+07	1.105e+08	1.342e+07
		766 PFLOP	all	active-servers	1.059e+09	1.511e+09	1.126e+08	1.286e+07
		1532 PFLOP	all	active-servers	1.179e+10	5.547e+09	1.135e+08	1.317e+07
		3063 PFLOP	all	active-servers	5.504e+10	1.134e+10	1.147e+08	1.318e+07
		306 PFLOP	all	active-servers	0	7.657e+07	1.106e+08	1.343e+07
		766 PFLOP	all	active-servers	1.059e+09	1.514e+09	1.127e+08	1.287e+07
		1532 PFLOP	all	active-servers	1.183e+10	5.608e+09	1.135e+08	1.318e+07
		3063 PFLOP	all	active-servers	5.527e+10	1.131e+10	1.149e+08	1.319e+07
		306 PFLOP	all	active-servers	0	0	1.758e+08	2.135e+07
		766 PFLOP	all	active-servers	2.803e+08	2.374e+09	1.791e+08	2.05e+07
		1532 PFLOP	all	active-servers	5.02e+09	5.349e+09	1.8e+08	2.096e+07
		3063 PFLOP	all	active-servers	3.298e+10	1.203e+10	1.816e+08	2.097e+07
		306 PFLOP	all	active-servers	0	4.703e+04	1.758e+08	2.135e+07
		766 PFLOP	all	active-servers	2.803e+08	2.374e+09	1.792e+08	2.05e+07
		1532 PFLOP	all	active-servers	5.043e+09	5.352e+09	1.801e+08	2.097e+07
		3063 PFLOP	all	active-servers	3.423e+10	1.221e+10	1.817e+08	2.098e+07

Table D.3: Results of operational phenomena experiment, with key metrics and their mean and standard deviation.

Topology	Workload	Op. Phen.	Alloc. Policy	Overcommission [MFLOP]		Power Consumption [Wh]	
				median	std	median	std
base	306 PFLOP	none	mem	0	0	1.102e+08	1.335e+07
base	306 PFLOP	none	mem-inv	5.662e+07	3.309e+09	1.108e+08	1.347e+07
base	306 PFLOP	none	core-mem	0	0	1.102e+08	1.335e+07
base	306 PFLOP	none	core-mem-inv	8.319e+07	2.172e+09	1.106e+08	1.344e+07
base	306 PFLOP	none	active-servers	0	6.543e+04	1.106e+08	1.346e+07
base	306 PFLOP	none	active-servers-inv	1.026e+08	2.163e+09	1.104e+08	1.342e+07
base	306 PFLOP	none	random	0	7.099e+04	1.105e+08	1.345e+07
base	306 PFLOP	interference	mem	1.951e+09	1.786e+09	1.102e+08	1.335e+07
base	306 PFLOP	interference	mem-inv	8.42e+09	5.136e+09	1.108e+08	1.347e+07
base	306 PFLOP	interference	core-mem	1.222e+09	1.487e+09	1.102e+08	1.335e+07
base	306 PFLOP	interference	core-mem-inv	1.09e+10	3.629e+09	1.106e+08	1.344e+07
base	306 PFLOP	interference	active-servers	0	1.573e+06	1.106e+08	1.346e+07
base	306 PFLOP	interference	active-servers-inv	1.203e+10	6.329e+09	1.104e+08	1.342e+07
base	306 PFLOP	interference	random	1.36e+08	1.069e+09	1.105e+08	1.345e+07
base	306 PFLOP	failures	mem	0	0	1.102e+08	1.335e+07
base	306 PFLOP	failures	mem-inv	9.672e+07	2.497e+09	1.107e+08	1.346e+07
base	306 PFLOP	failures	core-mem	0	0	1.102e+08	1.335e+07
base	306 PFLOP	failures	core-mem-inv	1.045e+08	2.164e+09	1.106e+08	1.344e+07
base	306 PFLOP	failures	active-servers	0	4.703e+04	1.106e+08	1.343e+07
base	306 PFLOP	failures	active-servers-inv	1.026e+08	2.165e+09	1.104e+08	1.342e+07
base	306 PFLOP	failures	random	0	3.939e+07	1.105e+08	1.341e+07
base	306 PFLOP	all	mem	1.423e+09	1.917e+09	1.102e+08	1.335e+07
base	306 PFLOP	all	mem-inv	8.388e+09	5.075e+09	1.107e+08	1.346e+07
base	306 PFLOP	all	core-mem	1.502e+09	2.269e+09	1.102e+08	1.335e+07
base	306 PFLOP	all	core-mem-inv	1.079e+10	3.501e+09	1.106e+08	1.343e+07
base	306 PFLOP	all	active-servers	0	7.653e+07	1.106e+08	1.343e+07
base	306 PFLOP	all	active-servers-inv	1.148e+10	5.756e+09	1.104e+08	1.342e+07
base	306 PFLOP	all	random	2.613e+08	1.797e+09	1.105e+08	1.341e+07
base	766 PFLOP	none	mem	0	0	1.121e+08	1.279e+07
base	766 PFLOP	none	mem-inv	1.734e+09	5.145e+09	1.137e+08	1.306e+07
base	766 PFLOP	none	core-mem	0	0	1.121e+08	1.279e+07
base	766 PFLOP	none	core-mem-inv	7.918e+08	3.135e+09	1.135e+08	1.301e+07
base	766 PFLOP	none	active-servers	0	1.242e+05	1.128e+08	1.293e+07
base	766 PFLOP	none	active-servers-inv	2.481e+09	8.611e+09	1.13e+08	1.296e+07
base	766 PFLOP	none	random	0	2.897e+06	1.13e+08	1.294e+07
base	766 PFLOP	interference	mem	1.901e+10	6.425e+09	1.121e+08	1.279e+07
base	766 PFLOP	interference	mem-inv	2.65e+10	9.494e+09	1.137e+08	1.306e+07
base	766 PFLOP	interference	core-mem	1.838e+10	7.025e+09	1.121e+08	1.279e+07
base	766 PFLOP	interference	core-mem-inv	2.905e+10	7.354e+09	1.135e+08	1.301e+07
base	766 PFLOP	interference	active-servers	5.298e+08	1.934e+09	1.128e+08	1.293e+07
base	766 PFLOP	interference	active-servers-inv	4.045e+10	1.41e+10	1.13e+08	1.296e+07
base	766 PFLOP	interference	random	2.085e+09	3.228e+09	1.13e+08	1.294e+07
base	766 PFLOP	failures	mem	0	2.956e+07	1.121e+08	1.279e+07
base	766 PFLOP	failures	mem-inv	1.499e+09	3.789e+09	1.135e+08	1.301e+07
base	766 PFLOP	failures	core-mem	0	8.401e+06	1.121e+08	1.279e+07
base	766 PFLOP	failures	core-mem-inv	1.077e+09	3.287e+09	1.132e+08	1.298e+07
base	766 PFLOP	failures	active-servers	0	3.987e+07	1.127e+08	1.289e+07
base	766 PFLOP	failures	active-servers-inv	2.134e+09	8.564e+09	1.128e+08	1.295e+07
base	766 PFLOP	failures	random	0	1.11e+08	1.128e+08	1.289e+07
base	766 PFLOP	all	mem	1.859e+10	6.785e+09	1.121e+08	1.278e+07
base	766 PFLOP	all	mem-inv	2.603e+10	7.593e+09	1.135e+08	1.303e+07
base	766 PFLOP	all	core-mem	2.069e+10	6.969e+09	1.121e+08	1.278e+07
base	766 PFLOP	all	core-mem-inv	2.878e+10	7.465e+09	1.134e+08	1.298e+07
base	766 PFLOP	all	active-servers	1.059e+09	1.514e+09	1.128e+08	1.288e+07
base	766 PFLOP	all	active-servers-inv	3.754e+10	1.48e+10	1.129e+08	1.295e+07
base	766 PFLOP	all	random	3.214e+09	2.526e+09	1.129e+08	1.289e+07
base	1532 PFLOP	none	mem	2.108e+06	1.128e+08	1.124e+08	1.302e+07
base	1532 PFLOP	none	mem-inv	5.12e+09	7.258e+09	1.157e+08	1.36e+07
base	1532 PFLOP	none	core-mem	1.023e+06	1.213e+09	1.124e+08	1.303e+07
base	1532 PFLOP	none	core-mem-inv	2.448e+09	4.48e+09	1.155e+08	1.358e+07
base	1532 PFLOP	none	active-servers	4.9e+05	4.845e+08	1.137e+08	1.325e+07
base	1532 PFLOP	none	active-servers-inv	1.242e+10	1.189e+10	1.148e+08	1.347e+07
base	1532 PFLOP	none	random	2.81e+06	1.536e+09	1.14e+08	1.329e+07
base	1532 PFLOP	interference	mem	7.042e+10	1.328e+10	1.124e+08	1.302e+07

Continued on next page

Table D.3: Results of operational phenomena experiment, with key metrics and their mean and standard deviation.

Topology	Workload	Op. Phen.	Alloc. Policy	Overcommission [MFLOP]		Power Consumption [Wh]	
				median	std	median	std
base	1532 PFLOP	interference	mem-inv	5.968e+10	1.333e+10	1.157e+08	1.36e+07
base	1532 PFLOP	interference	core-mem	7.529e+10	1.181e+10	1.124e+08	1.303e+07
base	1532 PFLOP	interference	core-mem-inv	5.868e+10	1.052e+10	1.155e+08	1.358e+07
base	1532 PFLOP	interference	active-servers	9.837e+09	4.374e+09	1.137e+08	1.325e+07
base	1532 PFLOP	interference	active-servers-inv	8.973e+10	1.804e+10	1.148e+08	1.347e+07
base	1532 PFLOP	interference	random	1.326e+10	6.45e+09	1.14e+08	1.329e+07
base	1532 PFLOP	failures	mem	2.4e+07	4.724e+08	1.124e+08	1.302e+07
base	1532 PFLOP	failures	mem-inv	3.479e+09	6.233e+09	1.154e+08	1.351e+07
base	1532 PFLOP	failures	core-mem	6.982e+07	1.22e+09	1.124e+08	1.302e+07
base	1532 PFLOP	failures	core-mem-inv	2.127e+09	4.865e+09	1.151e+08	1.349e+07
base	1532 PFLOP	failures	active-servers	1.613e+06	1.18e+09	1.137e+08	1.319e+07
base	1532 PFLOP	failures	active-servers-inv	8.15e+09	1.143e+10	1.142e+08	1.338e+07
base	1532 PFLOP	failures	random	4.024e+06	2.341e+09	1.139e+08	1.324e+07
base	1532 PFLOP	all	mem	7.815e+10	1.253e+10	1.124e+08	1.302e+07
base	1532 PFLOP	all	mem-inv	5.715e+10	9.696e+09	1.152e+08	1.353e+07
base	1532 PFLOP	all	core-mem	7.581e+10	1.721e+10	1.124e+08	1.302e+07
base	1532 PFLOP	all	core-mem-inv	5.885e+10	1.096e+10	1.153e+08	1.349e+07
base	1532 PFLOP	all	active-servers	1.192e+10	5.622e+09	1.137e+08	1.32e+07
base	1532 PFLOP	all	active-servers-inv	8.486e+10	1.848e+10	1.145e+08	1.339e+07
base	1532 PFLOP	all	random	1.432e+10	6.048e+09	1.139e+08	1.323e+07
base	3063 PFLOP	none	mem	2.118e+10	1.58e+10	1.132e+08	1.303e+07
base	3063 PFLOP	none	mem-inv	8.24e+09	9.127e+09	1.195e+08	1.433e+07
base	3063 PFLOP	none	core-mem	1.254e+10	8.967e+09	1.141e+08	1.322e+07
base	3063 PFLOP	none	core-mem-inv	9.598e+09	1.125e+10	1.195e+08	1.434e+07
base	3063 PFLOP	none	active-servers	1.201e+09	2.916e+09	1.151e+08	1.327e+07
base	3063 PFLOP	none	active-servers-inv	3.632e+10	3.072e+10	1.174e+08	1.4e+07
base	3063 PFLOP	none	random	1.872e+09	4.763e+09	1.156e+08	1.336e+07
base	3063 PFLOP	interference	mem	2.455e+11	2.342e+10	1.132e+08	1.303e+07
base	3063 PFLOP	interference	mem-inv	1.178e+11	1.717e+10	1.195e+08	1.433e+07
base	3063 PFLOP	interference	core-mem	1.944e+11	2.155e+10	1.141e+08	1.322e+07
base	3063 PFLOP	interference	core-mem-inv	1.208e+11	2.077e+10	1.195e+08	1.434e+07
base	3063 PFLOP	interference	active-servers	4.914e+10	1.23e+10	1.151e+08	1.327e+07
base	3063 PFLOP	interference	active-servers-inv	2.161e+11	3.762e+10	1.174e+08	1.4e+07
base	3063 PFLOP	interference	random	5.421e+10	1.603e+10	1.156e+08	1.336e+07
base	3063 PFLOP	failures	mem	2.779e+10	1.701e+10	1.135e+08	1.315e+07
base	3063 PFLOP	failures	mem-inv	6.364e+09	8.511e+09	1.188e+08	1.412e+07
base	3063 PFLOP	failures	core-mem	8.847e+09	8.367e+09	1.143e+08	1.317e+07
base	3063 PFLOP	failures	core-mem-inv	6.153e+09	7.126e+09	1.187e+08	1.413e+07
base	3063 PFLOP	failures	active-servers	2.111e+09	4.357e+09	1.151e+08	1.319e+07
base	3063 PFLOP	failures	active-servers-inv	3.255e+10	2.869e+10	1.168e+08	1.38e+07
base	3063 PFLOP	failures	random	9.814e+08	3.34e+09	1.156e+08	1.328e+07
base	3063 PFLOP	all	mem	2.217e+11	3.067e+10	1.134e+08	1.332e+07
base	3063 PFLOP	all	mem-inv	1.147e+11	1.401e+10	1.188e+08	1.407e+07
base	3063 PFLOP	all	core-mem	1.561e+11	2.525e+10	1.144e+08	1.327e+07
base	3063 PFLOP	all	core-mem-inv	1.216e+11	1.887e+10	1.188e+08	1.405e+07
base	3063 PFLOP	all	active-servers	5.691e+10	1.131e+10	1.151e+08	1.32e+07
base	3063 PFLOP	all	active-servers-inv	2.035e+11	2.452e+10	1.17e+08	1.383e+07
base	3063 PFLOP	all	random	5.885e+10	1.098e+10	1.156e+08	1.325e+07

Table D.4: Results of composite workload experiment, with key metrics and their mean and standard deviation.

Topology	Workload	Op. Phen.	Alloc. Policy	Overcommission [MFLOP]		Power Consumption [Wh]	
				median	std	median	std
base	all-pri	failures	active-servers	5.356e+08	2.428e+09	4.382e+07	3.95e+06
base	pri-75-pub-25	failures	active-servers	4.324e+10	3.961e+10	4.369e+07	4.089e+06
base	pri-50-pub-50	failures	active-servers	1.358e+11	9.609e+10	4.353e+07	4.548e+06
base	pri-25-pub-75	failures	active-servers	2.838e+11	1.168e+11	4.352e+07	6.325e+06
base	all-pub	failures	active-servers	4.032e+11	1.541e+11	3.988e+07	9.219e+06
	all-pri	failures	active-servers	1.094e+07	5.261e+08	6.885e+07	6.188e+06
	pri-75-pub-25	failures	active-servers	4.378e+10	3.267e+10	6.872e+07	6.294e+06
	pri-50-pub-50	failures	active-servers	1.022e+11	9.59e+10	6.862e+07	6.703e+06
	pri-25-pub-75	failures	active-servers	2.505e+11	1.061e+11	6.911e+07	9.373e+06
	all-pub	failures	active-servers	3.325e+11	1.444e+11	6.102e+07	1.355e+07
	all-pri	failures	active-servers	2.181e+08	3.535e+09	4.694e+07	4.228e+06
	pri-75-pub-25	failures	active-servers	3.912e+10	3.111e+10	4.684e+07	4.359e+06
	pri-50-pub-50	failures	active-servers	1.25e+11	1.008e+11	4.667e+07	4.733e+06
	pri-25-pub-75	failures	active-servers	2.456e+11	1.116e+11	4.664e+07	6.696e+06
	all-pub	failures	active-servers	3.572e+11	1.698e+11	4.242e+07	9.665e+06
	all-pri	failures	active-servers	1.862e+07	5.246e+08	6.901e+07	6.204e+06
	pri-75-pub-25	failures	active-servers	4.804e+10	3.466e+10	6.883e+07	6.318e+06
	pri-50-pub-50	failures	active-servers	1.234e+11	9.76e+10	6.872e+07	6.752e+06
	pri-25-pub-75	failures	active-servers	2.658e+11	1.131e+11	6.922e+07	9.444e+06
	all-pub	failures	active-servers	3.829e+11	1.559e+11	6.125e+07	1.369e+07

Table D.5: Results of HPC experiment, with key metrics and their mean and standard deviation.

Topology	Workload (HPC)	Op. Phen.	Alloc. Policy	Overcommission [MFLOP]		Power Consumption [Wh]	
				median	std	median	std
	base 0% VM set-size	all	active-servers	9.257e+10	3.891e+10	1.153e+08	1.316e+07
	base 25% total load	all	active-servers	2.454e+11	2.889e+10	1.142e+08	6.333e+06
	base 50% total load	all	active-servers	2.735e+11	6.93e+09	1.426e+08	5.504e+06
	base 100% total load	all	active-servers	5.166e+11	6.571e+09	1.718e+08	6.85e+06
	base 25% VM set-size	all	active-servers	7.443e+10	1.021e+10	1.124e+08	3.687e+06
	base 50% VM set-size	all	active-servers	1.145e+11	1.019e+10	1.119e+08	3.729e+06
	base 100% VM set-size	all	active-servers	1.805e+11	5.062e+09	8.978e+07	9.291e+06
	0% VM set-size	all	active-servers	6.552e+10	3.524e+10	1.811e+08	2.095e+07
	25% total load	all	active-servers	1.465e+11	5.719e+09	1.773e+08	5.881e+06
	50% total load	all	active-servers	2.648e+11	1.575e+10	1.902e+08	5.968e+06
	100% total load	all	active-servers	5.166e+11	4.903e+09	2.183e+08	1.336e+07
	25% VM set-size	all	active-servers	6.5e+10	5.853e+09	1.769e+08	5.87e+06
	50% VM set-size	all	active-servers	9.971e+10	6.943e+09	1.765e+08	5.913e+06
	100% VM set-size	all	active-servers	1.853e+11	1.925e+09	1.241e+08	5.926e+06
	0% VM set-size	all	active-servers	8.389e+10	3.47e+10	1.233e+08	1.412e+07
	25% total load	all	active-servers	2.112e+11	1.678e+10	1.2e+08	4.002e+06
	50% total load	all	active-servers	2.594e+11	1.784e+10	1.311e+08	7.406e+06
	100% total load	all	active-servers	4.931e+11	1.062e+10	1.582e+08	1.065e+07
	25% VM set-size	all	active-servers	7.193e+10	1.027e+10	1.204e+08	3.941e+06
	50% VM set-size	all	active-servers	1.058e+11	7.143e+09	1.199e+08	4.002e+06
	100% VM set-size	all	active-servers	1.767e+11	6.772e+09	8.401e+07	4.051e+06
	0% VM set-size	all	active-servers	6.617e+10	3.526e+10	1.815e+08	2.097e+07
	25% total load	all	active-servers	1.47e+11	7.966e+09	1.776e+08	5.883e+06
	50% total load	all	active-servers	2.655e+11	1.287e+10	1.905e+08	5.97e+06
	100% total load	all	active-servers	5.172e+11	5.641e+09	2.186e+08	1.317e+07
	25% VM set-size	all	active-servers	6.479e+10	6.759e+09	1.772e+08	5.862e+06
	50% VM set-size	all	active-servers	1.009e+11	4.881e+09	1.768e+08	5.926e+06
	100% VM set-size	all	active-servers	1.854e+11	2.108e+09	1.243e+08	5.936e+06

Table D.6: Results of the replay experiment (see Appendix B), with key metrics and their mean and standard deviation.

Topology	Workload	Op. Phen.	Alloc. Policy	Overcommission [MFLOP] median	std	Power Consumption [Wh] median	std
base	3063 PFLOP	none	replay	393283398	1.076e+09	9.646e+07	1.225e+07
base	3063 PFLOP	none	active-servers	1200611704	2.916e+09	1.151e+08	1.327e+07

Acronyms

- CMMI** Capability Maturity Model Integration. 15
- COBIT** Control Objectives for Information and Related Technologies. 16
- FaaS** Function as a Service. 44, 77
- FOSS** Free and Open Source Software. 5, 59, 76
- GDPR** General Data Protection Regulation. 13, 45, 80
- HPC** High Performance Computing. 8, 37, 39, 44, 60, 61, 67, 73, 74
- IaaS** Infrastructure as a Service. 1, 3, 17
- IT** Information Technology. 14, 15, 79, 80
- ITIL** Information Technology Infrastructure Library. 15
- MOF** Microsoft Operations Framework. 16
- PaaS** Platform as a Service. 3, 17
- PCA** Principal Component Analysis. 19
- QoS** Quality of Service. 2, 8, 9, 40, 51, 52, 56, 72, 75
- SaaS** Software as a Service. 1
- SLA** Service Level Agreement. 2, 3, 9, 12–14, 16, 20, 27, 29, 30, 51, 67
- SLI** Service Level Indicator. 65
- SLO** Service Level Objective. 2, 9, 12, 51, 53, 54, 56, 65
- TOGAF** The Open Group Architecture Framework. 16
- VM** Virtual Machine. 1, 8, 9, 17, 20–22, 37, 39, 44, 54, 59–65, 67–70, 72, 73, 88–109

Bibliography

- [1] CMMI v2.0, 2018. URL <https://cmmiinstitute.com/cmmi>.
- [2] The TOGAF Standard, 2018. URL <https://publications.opengroup.org/c182>.
- [3] Geetha Adinarayan. Monitoring and capacity planning of private clouds: the challenges and the solution. *IEEE Cloud Computing for Emerging Markets, CCEM 2012 - Proceedings*, pages 180–182, 2012.
- [4] Omid Alipourfard et al. Risk based planning of network changes in evolving data centers. *Proceedings of the 27th ACM Symposium on Operating Systems Principles*, page 414–429, 2019.
- [5] Virgílio A.F. Almeida and Daniel A. Menascé. Capacity planning: An essential tool for managing Web services. *IT Professional*, 4(4):33–38, 2002.
- [6] Tayfur Altiok and Mesut Gunduc. A Capacity Planning Tool for the Tuxedo Middleware Used in Transaction Processing Systems. In *Winter Simulation Conference*, pages 502–507, 2001.
- [7] George Amvrosiadis et al. On the diversity of cluster workloads and its impact on research results. In *2018 USENIX Annual Technical Conference, USENIX ATC 2018, Boston, MA, USA, July 11-13, 2018*, pages 533–546, 2018.
- [8] Georgios Andreadis. Literature on Cloud Capacity Planning – Dataset, 2020. URL <https://doi.org/10.5281/zenodo.3989102>.
- [9] Georgios Andreadis et al. A reference architecture for datacenter scheduling: Design, validation, and experiments. *SC'2018*, pages 478–492, 2018.
- [10] Danilo Ardagna, Barbara Panicucci, and Mauro Passacantando. A game theoretic formulation of the service provisioning problem in cloud systems. *Proceedings of the 20th International Conference on World Wide Web, WWW 2011*, pages 177–186, 2011.
- [11] Luiz André Barroso, Urs Hölzle, and Parthasarathy Ranganathan. *The Datacenter as a Computer: Designing Warehouse-Scale Machines*. Synthesis lectures on computer architecture. Morgan and Claypool, 2018. 3rd Edition.
- [12] Salman A Baset, Long Wang, and Chunqiang Tang. Towards an understanding of oversubscription in cloud. *2nd USENIX Workshop on Hot Topics in Management of Internet, Cloud, and Enterprise Networks and Services*, 2012.
- [13] Rabih Bashroush and Moustafa Noureddine. A cost effective cloud data centre capacity planning method based on modality cost analysis. *International Journal of Communication Networks and Distributed Systems*, 11(3):250–261, 2013.
- [14] Joe Bauer and Al Bellamy. Latent Effects of Cloud Computing on IT Capacity Management Structures. *International Journal of Computer and Communication Engineering*, 6(2):111–126, 2017.
- [15] Joseph Frederick Bauer. *Understanding How Organizations Operate Their IT Capacity-Management Processes*. PhD thesis, 2015.
- [16] Betsy Beyer et al. *Site Reliability Engineering: How Google Runs Production Systems*. O'Reilly Media, 2016.
- [17] Robert Birke et al. Failure analysis of virtual and physical machines: Patterns, causes and characteristics. *44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, DSN 2014*, pages 1–12, 2014.

- [18] Mark Blackburn and Green Grid. Five ways to reduce data center server power consumption. *The Green Grid*, 42:12, 2008.
- [19] Raphael Bolze et al. Grid'5000: A large scale and highly reconfigurable experimental grid testbed. *Int. J. High Perform. Comput. Appl.*, 20(4):481–494, 2006.
- [20] Tim Browning. *Capacity Planning for Computer Systems*. Academic Press, 1994.
- [21] Rodrigo N. Calheiros et al. Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Softw. Pract. Exp.*, 41(1):23–50, 2011.
- [22] David Candeia, Ricardo Araujo Santos, and Raquel Lopes. Business-Driven Long-Term Capacity Planning for SaaS Applications. *IEEE Transactions on Cloud Computing*, 3(3):290–303, 2015.
- [23] Marcus Carvalho, Daniel A. Menascé, and Francisco Brasileiro. Capacity planning for IaaS cloud providers offering multiple service classes. *Future Generation Computer Systems*, 77:97–111, 2017.
- [24] Marcus Carvalho et al. Multi-dimensional admission control and capacity planning for IaaS clouds with multiple service classes. *Proceedings - 2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, CCGRID 2017*, pages 160–169, 2017.
- [25] Sivadon Chaisiri, Bu-Sung Lee, and Dusit Niyato. Optimization of Resource Provisioning Cost in Cloud Computing. *IEEE Transactions on Services Computing*, 5(2):164–177, 2012.
- [26] Sivadon Chaisiri, Dusit Niyato, and Bu Sung Lee. Capacity planning for data center to support green computing. *2014 11th Int. Joint Conf. on Computer Science and Software Engineering: "Human Factors in Computer Science and Software Engineering" - e-Science and High Performance Computing: eHPC, JCSSE 2014*, pages 152–157, 2014.
- [27] Mark Chamness. Capacity forecasting in a backup storage environment. *USENIX Large Installation System Administration Conference (LISA)*, 2011.
- [28] Shuang Chen, Yanzhi Wang, and Massoud Pedram. Concurrent placement, capacity provisioning, and request flow control for a distributed cloud infrastructure. *2014 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 1–6, 2014.
- [29] Yanpei Chen, Vern Paxson, and Randy H Katz. What's new about cloud computing security. *University of California, Berkeley Report No. UCB/EECS-2010-5 January*, 20, 2010.
- [30] Ludmila Cherkasova, Wenting Tang, and Sharad Singhal. An SLA-oriented capacity planning tool for streaming media services. *Proceedings of the International Conference on Dependable Systems and Networks*, pages 743–752, 2004.
- [31] Yi Ju Chiang and Yen Chieh Ouyang. Profit optimization in SLA-aware cloud services with a finite capacity queuing model. *Mathematical Problems in Engineering*, 2014, 2014.
- [32] Gerry Coleman and Rory O'Connor. Using grounded theory to understand software process improvement: A study of Irish software product companies. *Information and Software Technology*, 49(6):654–667, 2007.
- [33] Eli Cortez et al. Resource central: Understanding and predicting workloads for improved resource management in large cloud platforms. In *Proceedings of the 26th Symposium on Operating Systems Principles*, pages 153–167, 2017.
- [34] Pratyush K. Deka et al. Adversarial impact on anomaly detection in cloud datacenters. In *2019 IEEE 24th Pacific Rim International Symposium on Dependable Computing (PRDC)*, pages 188–18809. IEEE, 2019.
- [35] Jay L. Devore. *Probability and Statistics for Engineering and the Sciences, 7Ed*. Brooks Cole Cengage Learning, 2009.

- [36] Christoph Dorsch and Björn Häckel. Combining models of capacity supply to handle volatile demand: The economic impact of surplus capacity in cloud service environments. *Decision Support Systems*, 58(1):3–14, 2014.
- [37] Dmitry Duplyakin et al. The design and operation of cloudlab. *2019 USENIX Annual Technical Conference, USENIX ATC 2019, Renton, WA, USA, July 10-12, 2019*, pages 1–14, 2019.
- [38] Nosayba El-Sayed, Hongyu Zhu, and Bianca Schroeder. Learning from failure across multiple clusters: A trace-driven approach to understanding, predicting, and mitigating job terminations. In *37th IEEE International Conference on Distributed Computing Systems, ICDCS 2017*, pages 1333–1344, 2017.
- [39] Martin Ester et al. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, pages 226–231, 1996.
- [40] Flexera. State of the Cloud Report. Tech. Rep., 2020.
- [41] Matthieu Gallet et al. A model for space-correlated failures in large-scale distributed systems. In *Euro-Par 2010 - Parallel Processing, 16th International Euro-Par Conference, Ischia, Italy, August 31 - September 3, 2010, Proceedings, Part I*, volume 6271, pages 88–100, 2010.
- [42] Gartner Inc. Gartner Forecasts Worldwide Public Cloud Revenue to Grow 17% in 2020. Press Release, 2019.
- [43] Frank Gens. Worldwide and Regional Public IT Cloud Services 2019–2023 Forecast. Tech. Rep. by IDC, Doc. #US44202119, Aug 2019.
- [44] Rahul Ghosh et al. Stochastic model driven capacity planning for an infrastructure-as-a-service cloud. *IEEE Transactions on Services Computing*, 7(4):667–680, 2014.
- [45] James Glanz. Data Centers Waste Vast Amounts of Energy, Belying Industry Image. *N.Y. Times*, 2012.
- [46] Daniel Gmach et al. Capacity management and demand prediction for next generation data centers. *Proceedings - 2007 IEEE International Conference on Web Services, ICWS 2007*, (July):43–50, 2007.
- [47] Marcelo Goncalves et al. Performance Inference: A Novel Approach for Planning the Capacity of IaaS Cloud Applications. *Proceedings - 2015 IEEE 8th International Conference on Cloud Computing, CLOUD 2015*, pages 813–820, 2015.
- [48] Albert G. Greenberg et al. The cost of a cloud: research problems in data center networks. *Computer Communication Review*, 39(1):68–73, 2009.
- [49] Neil J. Gunther. *Guerrilla Capacity Planning*. Springer, 2007.
- [50] Mor Harchol-Balter. *Performance Modeling and Design of Computer Systems: Queueing Theory in Action*. Cambridge University Press, Online, 2013.
- [51] Nikolas Herbst et al. Quantifying cloud performance and dependability: Taxonomy, metric design, and emerging challenges. *ACM Transactions on Modeling and Performance Evaluation of Computing Systems*, 3(4), 2018.
- [52] Hewlett-Packard Development Company. HP Capacity Advisor Version 7.4, 2014. URL https://support.hpe.com/hpsc/doc/public/display?docId=emr_na-c04453044. [Online; accessed 25-May-2020].
- [53] Takahiro Hirofuchi, Adrien Lebre, and Laurent Pouilloux. Simgrid VM: virtual machine support for a simulation framework of distributed systems. *IEEE Trans. Cloud Computing*, 6(1):221–234, 2018.
- [54] David Hixson et al. Capacity Planning. *USENIX ;login*, February, 2015.
- [55] Phillip C. Howard. IS Capacity Management Handbook Series-Volume 1-Capacity Planning. *Institute for Computer Capacity Management*, 1992.

- [56] Woonghee Tim Huh, Robin O. Roundy, and Metin Çakanyildirim. A general strategic capacity planning model under demand uncertainty. *Naval Research Logistics*, 53(2):137–150, 2006.
- [57] IBM. IBM Z Performance and Capacity Analytics tool, 2019. URL <https://www.ibm.com/us-en/marketplace/z-decision-support-for-capacity-planning>. [Online; accessed 25-May-2020].
- [58] Alexandru Iosup et al. The Grid Workloads Archive. *Future Generation Computer Systems*, 24(7):672–686, 2008.
- [59] Alexandru Iosup et al. The opendc vision: Towards collaborative datacenter simulation and exploration for everybody. *ISPDC'17*, pages 85–94, 2017.
- [60] Alexandru Iosup et al. Massivizing computer systems: A vision to understand, design, and engineer computer ecosystems through and beyond modern distributed systems. *Proceedings - International Conference on Distributed Computing Systems*, 2018-July:1224–1237, 2018.
- [61] Alexandru Iosup et al. The AtLarge Vision on the Design of Distributed Systems and Ecosystems. In *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, pages 1765–1776. IEEE, 2019.
- [62] ISACA. *COBIT 2019 Framework - Governance and Management Objectives*. 2019.
- [63] ISO/IEC JTC 1/SC 40 IT Service Management and IT Governance. *Iso/Iec 20000-1:2018*, 2018. URL <https://www.iso.org/standard/70636.html>.
- [64] ITIL. *ITIL Foundation*. AXELOS, 2019.
- [65] Iván Carrera Izurieta and Cláudio Resin Geyer. Impressionism in cloud computing a position paper on capacity planning in cloud computing environments. *ICEIS 2013 - Proceedings of the 15th International Conference on Enterprise Information Systems*, 2:333–338, 2013.
- [66] Bahman Javadi et al. The failure trace archive: Enabling the comparison of failure measurements and models of distributed systems. *J. Parallel Distributed Comput.*, 73(8):1208–1223, 2013.
- [67] Myeongjae Jeon et al. Analysis of large-scale multi-tenant GPU clusters for DNN training workloads. In *2019 USENIX Annual Technical Conference, USENIX ATC 2019, Renton, WA, USA, July 10-12, 2019*, pages 947–960, 2019.
- [68] Guofei Jiang, Haifeng Chen, and Kenji Yoshihira. Profiling services for resource optimization and capacity planning in distributed systems. *Cluster Computing*, 11(4):313–329, 2008.
- [69] Haokun Jiang and Xiaotong Sun. Understanding Networking Capacity Management in Cloud Computing. In *International Conference on Smart Computing and Communication*, pages 516–526, 2017.
- [70] Yexi Jiang, Chang Shing Perng, Tao Li, and Rong Chang. Self-adaptive cloud capacity planning. *Proceedings - 2012 IEEE 9th International Conference on Services Computing, SCC 2012*, pages 73–80, 2012.
- [71] K. Kant and Youjip Won. Server capacity planning for web traffic workload. *IEEE Transactions on Knowledge and Data Engineering*, 12(1):141–141, 1999.
- [72] Arun Kejariwal and John Allspaw. *The Art of Capacity Planning: Scaling Web Resources in the Cloud*. O'Reilly, 2017.
- [73] A. Kimms. Stability measures for rolling schedules with applications to capacity expansion planning, master production scheduling, and lot sizing. 1998.
- [74] Younggyun Koh et al. An analysis of performance interference effects in virtual environments. In *IEEE International Symposium on Performance Analysis of Systems & Software*, pages 200–209, 2007.
- [75] Fanxin Kong and Xue Liu. GreenPlanning: Optimal Energy Source Selection and Capacity Planning for Green Datacenters. *2016 ACM/IEEE 7th International Conference on Cyber-Physical Systems, ICCPS 2016 - Proceedings*, pages 1–10, 2016.

- [76] Yousri Kouki and Thomas Ledoux. SLA-driven capacity planning for Cloud applications. *CloudCom 2012 - Proceedings: 2012 4th IEEE International Conference on Cloud Computing Technology and Science*, pages 135–140, 2012.
- [77] Samuel Kounev et al. Automated simulation-based capacity planning for enterprise data fabrics. *SIMU-Tools 2011 - 4th International ICST Conference on Simulation Tools and Techniques*, pages 27–36, 2011.
- [78] Rouven Krebs, Christof Momm, and Samuel Kounev. Metrics and techniques for quantifying performance isolation in cloud environments. *Science of Computer Programming*, pages 116–134, 2014.
- [79] Shui F. Lam and K. Hung Chan. *Computer capacity planning: theory and practice*. Academic Press, 1987.
- [80] Tan N. Le et al. Joint capacity planning and operational management for sustainable data centers and demand response. *Proceedings of the 7th International Conference on Future Energy Systems, e-Energy 2016*, 2016.
- [81] Loo Hay Lee et al. Vehicle capacity planning system: A case study on vehicle routing problem with time windows. *IEEE Transactions on Systems, Man, and Cybernetics Part A: Systems and Humans.*, 33(2): 169–178, 2003.
- [82] Sang Bum Lee and Hanan Luss. Multifacility-Type Capacity Expansion Planning: Algorithms and Complexities. *Operations Research*, 35(2):249–253, 1987.
- [83] Weiling Li et al. On Stochastic Performance and Cost-Aware Optimal Capacity Planning of Unreliable Infrastructure-as-a-Service Cloud. In *International Conference on Algorithms and Architectures for Parallel Processing*, pages 644–657, 2016.
- [84] Charles Loboz. Cloud Resource Usage-Heavy Tailed Distributions Invalidating Traditional Capacity Planning Models. *Journal of Grid Computing*, 10(1):85–108, 2012.
- [85] Raquel Lopes, Francisco Brasileiro, and Paulo Ditarso Maciel. Business-driven capacity planning of a cloud-based IT infrastructure for the execution of web applications. *Proceedings of the 2010 IEEE International Symposium on Parallel and Distributed Processing, Workshops and Phd Forum, IPDPSW 2010*, pages 1–8, 2010.
- [86] Baochuan Lu, Linh Ngo, Hung Bui, Amy Apon, Nathan Hamm, Larry Dowdy, Doug Hoffman, and Denny Brewer. Capacity Planning of a Commodity Cluster in an Academic Environment: A Case Study. *9th LCI International Conference on High-Performance Clustered Computing*, 2008.
- [87] Sara Lumbreras and Andrés Ramos. The new challenges to transmission expansion planning. Survey of recent practice and literature review. *Electric Power Systems Research*, 134:19–29, 2016.
- [88] Hanan Luss. Operations Research and Capacity Expansion Problems: a Survey. *Operation Research*, 30(5):907–947, 1982.
- [89] Joseph S. Martinich. *Production and Operations Management*. John Wiley & Sons, 1997.
- [90] Peter Mell and Timothy Grance. The nist definition of cloud computing. Technical Report 800-145, National Institute of Standards and Technology (NIST), 2011.
- [91] Daniel A. Menascé and Virgílio A.F. Almeida. *Capacity Planning for Web Services: metrics, models, and methods*. Prentice Hall, 2001.
- [92] Daniel A. Menascé, Robert Peraino, Nikki Dinh, and Quan T. Dinh. Planning the capacity of a web server: an experience report. In *Computer Measurement Group Conference*, 1999.
- [93] Microsoft. Microsoft © Operations Framework Version 4.0, 2008. URL <https://docs.microsoft.com/en-us/previous-versions/tn-archive/cc506049%28v%3Dtechnet.10%29>.
- [94] Jeffrey C Mogul and John Wilkes. Nines are not enough: meaningful metrics for clouds. pages 136–141, 2019.

- [95] Jeffrey C Mogul et al. Experiences with modeling network topologies at multiple levels of abstraction. In *17th USENIX Symposium on Networked Systems Design and Implementation (NSDI20)*, pages 403–418, 2020.
- [96] Swarna Mylavarapu, Vijay Sukthankar, and Pradipta Banerjee. An optimized capacity planning approach for virtual infrastructure exhibiting stochastic workload. *Proceedings of the ACM Symposium on Applied Computing*, pages 386–390, 2010.
- [97] Shinji Nakadai and Kunihiro Taniguchi. Server capacity planning with priority allocation for service level management in heterogeneous server clusters. *10th IFIP/IEEE International Symposium on Integrated Network Management 2007, IM '07*, pages 753–756, 2007.
- [98] Derek L. Nazareth and Jae Choi. Capacity management for cloud computing: a system dynamics approach. *AMCIS 2017 - America's Conference on Information Systems: A Tradition of Innovation*, 2017-Augus:1–10, 2017.
- [99] Marius Noreikis, Yu Xiao, and Antti Yla-Jaaiski. QoS-oriented capacity planning for edge computing. *IEEE International Conference on Communications*, 2017.
- [100] Alberto Nuñez et al. iCanCloud: A flexible and scalable cloud infrastructure simulator. *J. Grid Comput.*, 10(1):185–209, 2012.
- [101] Isaac Odun-Ayo, Olasupo Ajayi, Rowland Goddy-Worl, and Jamaiah Yahaya. A Systematic Mapping Study of Cloud Resources Management and Scalability in Brokering, Scheduling, Capacity Planning and Elasticity. *Asian Journal of Scientific Research*, 12(2):151–166, 2019.
- [102] Per Olov Ostberg et al. Reliable capacity provisioning for distributed cloud/edge/fog computing applications. *EuCNC 2017 - European Conference on Networks and Communications*, pages 1–6, 2017.
- [103] Ranjan Pal and Pan Hui. Economic models for cloud service markets: Pricing and Capacity planning. *Theoretical Computer Science*, 496:113–124, 2013.
- [104] Kiejn Park and Sungsoo Kim. A capacity planning model of unreliable multimedia service systems. *Journal of Systems and Software*, 63(1):69–76, 2002.
- [105] Jayneel Patel, Shahram Sarkani, and Thomas Mazzuchi. Knowledge based data center capacity reduction using sensitivity analysis on causal Bayesian belief network. *Information Knowledge Systems Management*, 12(2):135–148, 2013.
- [106] Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
- [107] Frank W A D Prabath. *Workload and Performance Aware Capacity Planning*. PhD thesis, 2017.
- [108] Hassan Raei. Capacity planning framework for mobile network operator cloud using analytical performance model. *International Journal of Communication Systems*, 30(17):1–12, 2017.
- [109] Venkateshwar Rao and Sarika Rao. Application of artificial neural networks in capacity planning of cloud based IT infrastructure. *IEEE Cloud Computing for Emerging Markets, CCEM 2012 - Proceedings*, pages 38–41, 2012.
- [110] Neil Rasmussen. Power and Cooling Capacity Management for Data Centers. *APC White Paper #150, (Rev 0):18*, 2007.
- [111] Charles Reiss, John Wilkes, and Joseph L Hellerstein. Google cluster-usage traces: format and schema. *Google Inc., White Paper*, pages 1–14, 2011.
- [112] Chuangang Ren, Di Wang, Bhuvan Urgaonkar, and Anand Sivasubramaniam. Carbon-aware energy capacity planning for datacenters. *Proceedings of the 2012 IEEE 20th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems, MASCOTS 2012*, pages 391–400, 2012.

- [113] Marcel Risch, Yannis Makrypoulias, and Sergios Soursos. Economics-Aware Capacity Planning for Commercial Grids. Technical report, 2008.
- [114] Jerry Rolia et al. A capacity management service for resource pools. *Proceedings of the Fifth International Workshop on Software and Performance, WOSP'05*, pages 229–237, 2005.
- [115] Nilabja Roy, Abhishek Dubey, Aniruddha Gokhale, and Larry Dowdy. A capacity planning process for performance assurance of component-based distributed systems. *ICPE'11 - Proceedings of the 2nd Joint WOSP/SIPEW International Conference on Performance Engineering*, pages 259–270, 2011.
- [116] Naidila Sadashiv, S. M. Dilip Kumar, and R. S. Goudar. Cloud capacity planning and HSI based optimal resource provisioning. *Proceedings of the 2017 2nd IEEE International Conference on Electrical, Computer and Communication Technologies, ICECCT 2017*, pages 0–5, 2017.
- [117] Semih Salihoglu and M. Tamer Özsu. Response to "scale up or scale out for graph processing". *IEEE Internet Comput.*, 22(5):18–24, 2018.
- [118] Nay Myo Sandar, Lin Min Min Myint, and Sivadon Chaisiri. An optimization approach to capacity planning of aggregators and resource provisioning in cloud providers for meteorological sensor network. *Proceedings of the 2015 12th International Joint Conference on Computer Science and Software Engineering, JCSSE 2015*, pages 195–200, 2015.
- [119] S Shang, Y Wu, J Jiang, and W Zheng. An Intelligent Capacity Planning Model for Cloud Market. *Journal of Internet Services and Information Security*, 1(1):37–45, 2011.
- [120] Siqi Shen, Vincent Van Beek, and Alexandru Iosup. Statistical characterization of business-critical workloads hosted in cloud datacenters. *Proceedings - 2015 IEEE/ACM 15th International Symposium on Cluster, Cloud, and Grid Computing, CCGrid 2015*, pages 465–474, 2015.
- [121] Zhiming Shen et al. CloudScale: Elastic resource scaling for multi-tenant cloud systems. *Proceedings of the 2nd ACM Symposium on Cloud Computing, SOCC 2011*, 2011.
- [122] Amy Spellmann, Richard Gimarc, and Charles Gimarc. Green capacity planning: Theory and practice. *34th International Conference Computer Measurement Group*, 2008.
- [123] Ling Tang and Hao Chen. Joint Pricing and Capacity Planning in the IaaS Cloud Market. *IEEE Transactions on Cloud Computing*, 5(1):57–70, 2017.
- [124] Adel Nadjaran Toosi, Kurt Vanmechelen, Kotagiri Ramamohanarao, and Rajkumar Buyya. Revenue Maximization with Optimal Capacity Control in Infrastructure as a Service Cloud Markets. *IEEE Transactions on Cloud Computing*, 3(3):261–274, 2015.
- [125] Daniel W Turner. Qualitative Interview Design: A Practical Guide for Novice Investigators. *Qualitative Report*, 15(3):7, 2010.
- [126] Alexandru Uta et al. An MRI-Like View into the Life of a Datacenter. *USENIX ;login*, Fall, 2020.
- [127] Vincent van Beek, Giorgos Oikonomou, and Alexandru Iosup. A CPU Contention Predictor for Business-Critical Workloads in Cloud Datacenters. In *HotCloudPerf*, pages 56–61, 2019.
- [128] André van Hoorn. *Model-Driven Online Capacity Management for Component-Based Software Systems*. 2014.
- [129] Arunchandar Vasan et al. Worth their watts? - an empirical study of datacenter servers. *HPCA16*, pages 1–10, 2010.
- [130] Abhishek Verma et al. Large-scale cluster management at google with borg. pages 1–17, 2015.
- [131] Laurens Versluis, Roland Mathá, Sacheendra Talluri, Tim Hegeman, Radu Prodan, Ewa Deelman, and Alexandru Iosup. The Workflow Trace Archive: Open-Access Data from Public and Private Computing Infrastructures-Technical Report. 2019.

- [132] VMware. VMware Capacity Planner, 2009. URL <https://www.vmware.com/nl/products/capacity-planner.html>. [Online; accessed 25-May-2020].
- [133] Jinru Wang, Meina Song, Qian Chang, and Qin Shu. Capacity Planning for Telecom Operation Support System Cloud Migration. In *International Conference on Human Centered Computing*, pages 427–440, 2015.
- [134] Wei Wang, Baochun Li, and Ben Liang. Towards optimal capacity segmentation with hybrid cloud pricing. *Proceedings - International Conference on Distributed Computing Systems*, pages 425–434, 2012.
- [135] Mark D. Wilkinson et al. The FAIR Guiding Principles for scientific data management and stewardship. *Nature SciData*, 3, 2016.
- [136] Peng Xiao et al. Detecting ddos attacks against data center with correlation analysis. *Computer Communications*, 67:66–74, 2015.
- [137] Jie Xu and Chenbo Zhu. Optimal Pricing and Capacity Planning of a New Economy Cloud Computing Service Class. *Proceedings - 2015 International Conference on Cloud and Autonomic Computing, ICCAC 2015*, pages 149–157, 2015.
- [138] Chun Zhang et al. An optimal capacity planning algorithm for provisioning cluster-based failure-resilient composite services. *SCC 2009 - 2009 IEEE International Conference on Services Computing*, pages 112–119, 2009.
- [139] Qi Zhang et al. R-capriccio: A capacity planning and anomaly detection tool for enterprise services with live workloads. *ACM/IFIP/USENIX 2007 International Conference on Middleware*, page 244–265, 2007.
- [140] Ningxin Zheng, Quan Chen, Yong Yang, Jin Li, Wenli Zheng, and Minyi Guo. POSTER : Precise Capacity Planning for Database Public Clouds. *2019 28th International Conference on Parallel Architectures and Compilation Techniques (PACT)*, pages 457–458, 2019.