Arno Uhlig [1,2], Iris Braun [2], Matthias Wählisch [2]

[1] SAP SE, [2] TU Dresden

The SAP Cloud Infrastructure Dataset

A Reality Check of Scheduling and Placement of VMs in Cloud Computing

ACM IMC, Madison // October 28-21, 2025

Funded by the European Union
NextGenerationEU

Supported by:
Federal Ministry for Economic Affairs and Energy
on the basis of a decision by the German Bundestag

Agenda

Motivation

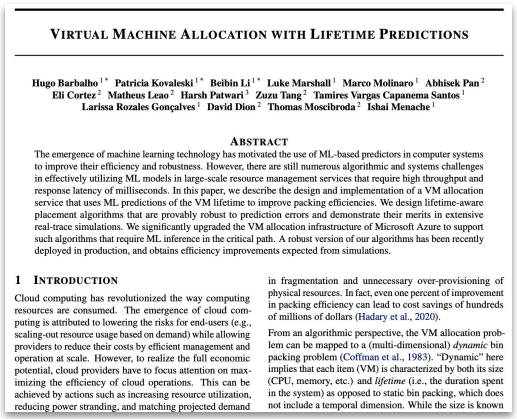
Virtual Machine (VM) placement and scheduling determines **where and when VMs run across physical hosts**

Efficient VM placement and scheduling remain **critical for optimizing resource utilization**

Growing **complexity, scale, and workload diversity** can provide **challenges for cloud infrastructure providers**

Usually, algorithms are evaluated based on synthetic data and lack real-world workload characteristics

Examples of research about resource allocation in cloud deployments



VIRTUAL MACHINE ALLOCATION WITH LIFETIME PREDICTIONS

Hugo Barbalho [1 *] Patricia Kovaleski [1 *] Beibin Li [1 *] Luke Marshall [1] Marco Molinaro [1] Abhisek Pan [2] Eli Cortez [2] Matheus Leao [2] Harsh Patwari [2] Zuzu Tang [2] Tamires Vargas Capanema Santos [1] Larissa Rozales Gonçalves [1] David Dion [2] Thomas Moscibroda [2] Ishai Menache [1]

ABSTRACT

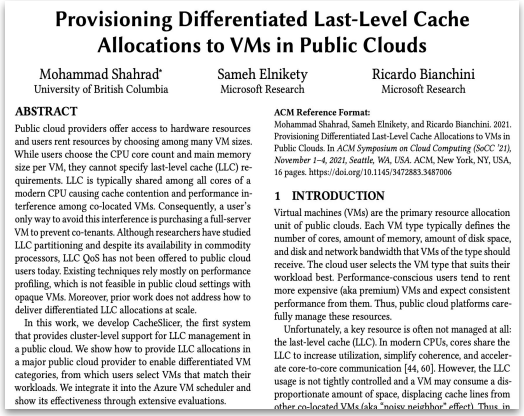
The emergence of machine learning technology has motivated the use of ML-based predictors in computer systems to improve their efficiency and robustness. However, there are still numerous algorithmic and systems challenges in effectively utilizing ML models in large-scale resource management services that require high throughput and response latency of milliseconds. In this paper, we describe the design and implementation of a VM allocation service that uses ML predictions of the VM lifetime to improve packing efficiencies. We design lifetime-aware placement algorithms that are provably robust to prediction errors and demonstrate their merits in extensive real-trace simulations. We significantly upgraded the VM allocation infrastructure of Microsoft Azure to support such algorithms that require ML inference in the critical path. A robust version of our algorithms has been recently deployed in production, and obtains efficiency improvements expected from simulations.

1 INTRODUCTION

Cloud computing has revolutionized the way computing resources are consumed. The emergence of cloud computing is attributed to lowering the risks for end-users (e.g., scaling-out resource usage based on demand) while allowing providers to reduce their costs by efficient management and operation at scale. However, to realize the full economic potential, cloud providers have to focus attention on maximizing the efficiency of cloud operations. This can be achieved by actions such as increasing resource utilization, reducing power stranding, and matching projected demand in fragmentation and unnecessary over-provisioning of physical resources. In fact, even one percent of improvement in packing efficiency can lead to cost savings of hundreds of millions of dollars (Hadary et al., 2020).

From an algorithmic perspective, the VM allocation problem can be mapped to a (multi-dimensional) *dynamic* bin packing problem (Coffman et al., 1983). "Dynamic" here implies that each item (VM) is characterized by both its size (CPU, memory, etc.) and *lifetime* (i.e., the duration spent in the system) as opposed to static bin packing, which does not include a temporal dimension. While the size is known

[MLSys '23]



Provisioning Differentiated Last-Level Cache Allocations to VMs in Public Clouds

Mohammad Shahrad* University of British Columbia

Sameh Elnikety Microsoft Research

Ricardo Bianchini Microsoft Research

ABSTRACT

Public cloud providers offer access to hardware resources and users rent resources by choosing among many VM sizes. While users choose the CPU core count and main memory size per VM, they cannot specify last-level cache (LLC) requirements. LLC is typically shared among all cores of a modern CPU causing cache contention and performance interference among co-located VMs. Consequently, a user's only way to avoid this interference is purchasing a full-server VM to prevent co-tenants. Although researchers have studied LLC partitioning and despite its availability in commodity processors, LLC QoS has not been offered to public cloud users today. Existing techniques rely mostly on performance profiling, which is not feasible in public cloud settings with opaque VMs. Moreover, prior work does not address how to deliver differentiated LLC allocations at scale.

In this work, we develop CacheSlicer, the first system that provides cluster-level support for LLC management in a public cloud. We show how to provide LLC allocations in a major public cloud provider to enable differentiated VM categories, from which users select VMs that match their workloads. We integrate it into the Azure VM scheduler and show its effectiveness through extensive evaluations.

ACM Reference Format:

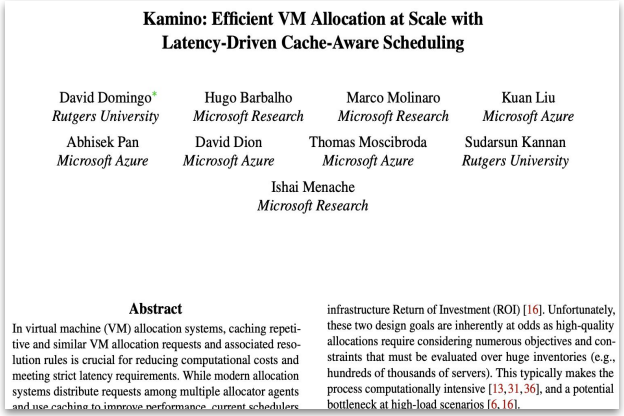
Mohammad Shahrad, Sameh Elnikety, and Ricardo Bianchini. 2021. Provisioning Differentiated Last-Level Cache Allocations to VMs in Public Clouds. In *ACM Symposium on Cloud Computing (SoCC '21), November 1–4, 2021, Seattle, WA, USA.* ACM, New York, NY, USA, 16 pages. https://doi.org/10.1145/3472883.3487006

1 INTRODUCTION

Virtual machines (VMs) are the primary resource allocation unit of public clouds. Each VM type typically defines the number of cores, amount of memory, amount of disk space, and disk and network bandwidth that VMs of the type should receive. The cloud user selects the VM type that suits their workload best. Performance-conscious users tend to rent more expensive (aka premium) VMs and expect consistent performance from them. Thus, public cloud platforms carefully manage these resources.

Unfortunately, a key resource is often not managed at all: the last-level cache (LLC). In modern CPUs, cores share the LLC to increase utilization, simplify coherence, and accelerate core-to-core communication [44, 60]. However, the LLC usage is not tightly controlled and a VM may consume a disproportionate amount of space, displacing cache lines from other co-located VMs (aka "noisy neighbor" effect). Thus, in

[SoCC '21]



Kamino: Efficient VM Allocation at Scale with Latency-Driven Cache-Aware Scheduling

David Domingo* *Rutgers University*

Hugo Barbalho *Microsoft Research*

Marco Molinaro *Microsoft Research*

Kuan Liu *Microsoft Azure*

Abhisek Pan *Microsoft Azure*

David Dion *Microsoft Azure*

Thomas Moscibroda *Microsoft Azure*

Sudarsun Kannan *Rutgers University*

Ishai Menache *Microsoft Research*

Abstract

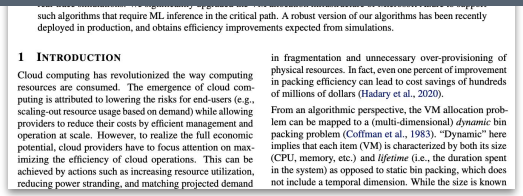
In virtual machine (VM) allocation systems, caching repetitive and similar VM allocation requests and associated resolution rules is crucial for reducing computational costs and meeting strict latency requirements. While modern allocation systems distribute requests among multiple allocator agents and use caching to improve performance, current schedulers infrastructure Return of Investment (ROI) [16]. Unfortunately, these two design goals are inherently at odds as high-quality allocations require considering numerous objectives and constraints that must be evaluated over huge inventories (e.g., hundreds of thousands of servers). This typically makes the process computationally intensive [13, 31, 36], and a potential bottleneck at high-load scenarios [6, 16].

[OSDI '25]

Usually, algorithms are evaluated based on synthetic data and lack real-world workload characteristics.

Examples of research about resource allocation in cloud deployments

A data-driven evaluation is important for a realistic assessment of scheduling and placement algorithms.


such algorithms that require ML inference in the critical path. A robust version of our algorithms has been recently deployed in production, and obtains efficiency improvements expected from simulations.

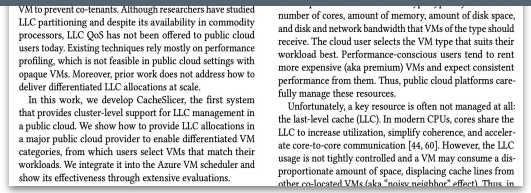
1 INTRODUCTION

Cloud computing has revolutionized the way computing resources are consumed. The emergence of cloud computing is attributed to lowering the risks for end-users (e.g., scaling-out resource usage based on demand) while allowing providers to reduce their costs by efficient management and operation at scale. However, to realize the full economic potential, cloud providers have to focus attention on maximizing the efficiency of cloud operations. This can be achieved by actions such as increasing resource utilization, reducing power stranding, and matching projected demand

in fragmentation and unnecessary over-provisioning of physical resources. In fact, even one percent of improvement in packing efficiency can lead to cost savings of hundreds of millions of dollars (Hadary et al., 2020).

From an algorithmic perspective, the VM allocation problem can be mapped to a (multi-dimensional) *dynamic* bin packing problem (Coffman et al., 1983). "Dynamic" here implies that each item (VM) is characterized by both its size (CPU, memory, etc.) and *lifetime* (i.e., the duration spent in the system) as opposed to static bin packing, which does not include a temporal dimension. While the size is known

[MLSys '23]

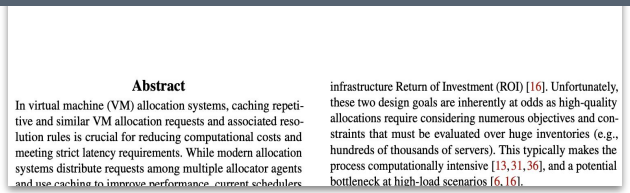

VM to prevent co-tenants. Although researchers have studied LLC partitioning and despite its availability in commodity processors, LLC QoS has not been offered to public cloud users today. Existing techniques rely mostly on performance profiling, which is not feasible in public cloud settings with opaque VMs. Moreover, prior work does not address how to deliver differentiated LLC allocations at scale.

In this work, we develop CacheSlicer, the first system that provides cluster-level support for LLC management in a public cloud. We show how to provide LLC allocations in a major public cloud provider to enable differentiated VM categories, from which users select VMs that match their workloads. We integrate it into the Azure VM scheduler and show its effectiveness through extensive evaluations.

number of cores, amount of memory, amount of disk space, and disk and network bandwidth that VMs of the type should receive. The cloud user selects the VM type that suits their workload best. Performance-conscious users tend to rent more expensive (aka premium) VMs and expect consistent performance from them. Thus, public cloud platforms carefully manage these resources.

Unfortunately, a key resource is often not managed at all: the last-level cache (LLC). In modern CPUs, cores share the LLC to increase utilization, simplify coherence, and accelerate core-to-core communication [44, 60]. However, the LLC usage is not tightly controlled and a VM may consume a disproportionate amount of space, displacing cache lines from

[SoCC '21]


Abstract

In virtual machine (VM) allocation systems, caching repetitive and similar VM allocation requests and associated resolution rules is crucial for reducing computational costs and meeting strict latency requirements. While modern allocation systems distribute requests among multiple allocator agents and use caching to improve performance, current schedulers

infrastructure Return of Investment (ROI) [16]. Unfortunately, these two design goals are inherently at odds as high-quality allocations require considering numerous objectives and constraints that must be evaluated over huge inventories (e.g., hundreds of thousands of servers). This typically makes the process computationally intensive [13, 31, 36], and a potential bottleneck at high-load scenarios [6, 16].

[OSDI '25]

Related Work. Common Cloud Datasets and SAP.

Dataset	Resources					Workload	Vantage Points and Data Sampling
	CPU	Memory	Network	Storage	GPU		
Google Alibaba Philly Atlas MIT							
Azure							
SAP (this work)							

Related Work. Common Cloud Datasets and SAP.

Dataset	Resources					Workload	Vantage Points and Data Sampling
	CPU	Memory	Network	Storage	GPU		
Google	✓	✓	✘	✘	✘		
Alibaba	✓	✓	✘	✓	✓		
Philly	✓	✓	✓	✘	✓		
Atlas	✓	✓	✓	✘	✓		
MIT	✓	✓	✓	✓	✓		
Azure	✓	✓	✓	✓	✘		
SAP (this work)	✓	✓	✓	✓	✘		

Related Work. Common Cloud Datasets and SAP.

Dataset	Resources					Workload		Vantage Points and Data Sampling
	CPU	Memory	Network	Storage	GPU	Batch Jobs	VMs	
Google	✓	✓	✗	✗	✗	✓	✗	
Alibaba	✓	✓	✗	✓	✓	✓	✗	
Philly	✓	✓	✓	✗	✓	✓	✗	
Atlas	✓	✓	✓	✗	✓	✓	✗	
MIT	✓	✓	✓	✓	✓	✓	✗	
Azure	✓	✓	✓	✓	✗	✗	✓	
SAP (this work)	✓	✓	✓	✓	✗	✗	✓	

Related Work. Common Cloud Datasets and SAP.

Dataset	Resources					Workload			Vantage Points and Data Sampling
	CPU	Memory	Network	Storage	GPU	Batch Jobs	VMs	Lifetime	
Google	✓	✓	✘	✘	✘	✓	✘	sec–days	
Alibaba	✓	✓	✘	✓	✓	✓	✘	min–days	
Philly	✓	✓	✓	✘	✓	✓	✘	min–weeks	
Atlas	✓	✓	✓	✘	✓	✓	✘	n/a	
MIT	✓	✓	✓	✓	✓	✓	✘	min–days	
Azure	✓	✓	✓	✓	✘	✘	✓	min–weeks	
SAP (this work)	✓	✓	✓	✓	✘	✘	✓	min–years	

Related Work. Common Cloud Datasets and SAP.

Dataset	Resources					Workload			Vantage Points and Data Sampling
	CPU	Memory	Network	Storage	GPU	Batch Jobs	VMs	Lifetime	Scale
Google	✓	✓	✘	✘	✘	✓	✘	sec–days	672,074 jobs
Alibaba	✓	✓	✘	✓	✓	✓	✘	min–days	~4k nodes
Philly	✓	✓	✓	✘	✓	✓	✘	min–weeks	117,325 jobs
Atlas	✓	✓	✓	✘	✓	✓	✘	n/a	96,260 jobs
MIT	✓	✓	✓	✓	✓	✓	✘	min–days	441–9k nodes
Azure	✓	✓	✓	✓	✘	✘	✓	min–weeks	>1M VMs
SAP (this work)	✓	✓	✓	✓	✘	✘	✓	min–years	1.8k nodes, 48k VMs

Related Work. Common Cloud Datasets and SAP.

Dataset	Resources					Workload			Vantage Points and Data Sampling		
	CPU	Memory	Network	Storage	GPU	Batch Jobs	VMs	Lifetime	Scale	Duration	Sampling
Google	✓	✓	✘	✘	✘	✓	✘	sec–days	672,074 jobs	29 days	5 min
Alibaba	✓	✓	✘	✓	✓	✓	✘	min–days	~4k nodes	8 days	n/a
Philly	✓	✓	✓	✘	✓	✓	✘	min–weeks	117,325 jobs	75 days	1 min
Atlas	✓	✓	✓	✘	✓	✓	✘	n/a	96,260 jobs	90–1,800 days	1 min
MIT	✓	✓	✓	✓	✓	✓	✘	min–days	441–9k nodes	90–180+ days	n/a
Azure	✓	✓	✓	✓	✘	✘	✓	min–weeks	>1M VMs	14 days	5 min
SAP (this work)	✓	✓	✓	✓	✘	✘	✓	min–years	1.8k nodes, 48k VMs	30 days	30s–300s

Related Work. Common Cloud Datasets and SAP.

Dataset	Resources					Workload			Vantage Points and Data Sampling			
	CPU	Memory	Network	Storage	GPU	Batch Jobs	VMs	Lifetime	Scale	Duration	Sampling	Public
Google	✓	✓	✘	✘	✘	✓	✘	sec–days	672,074 jobs	29 days	5 min	✓
Alibaba	✓	✓	✘	✓	✓	✓	✘	min–days	~4k nodes	8 days	n/a	✓
Philly	✓	✓	✓	✘	✓	✓	✘	min–weeks	117,325 jobs	75 days	1 min	✓
Atlas	✓	✓	✓	✘	✓	✓	✘	n/a	96,260 jobs	90–1,800 days	1 min	✓
MIT	✓	✓	✓	✓	✓	✓	✘	min–days	441–9k nodes	90–180+ days	n/a	✓
Azure	✓	✓	✓	✓	✘	✘	✓	min–weeks	>1M VMs	14 days	5 min	✘
SAP (this work)	✓	✓	✓	✓	✘	✘	✓	min–years	1.8k nodes, 48k VMs	30 days	30s–300s	✓

We present the first public dataset providing VM workload of a major cloud infrastructure.

Context and Research Objectives

Context

Large-scale, real-world cloud computing environment of SAP

Diverse workload from lightweight to resource-intensive tasks

Research Objectives

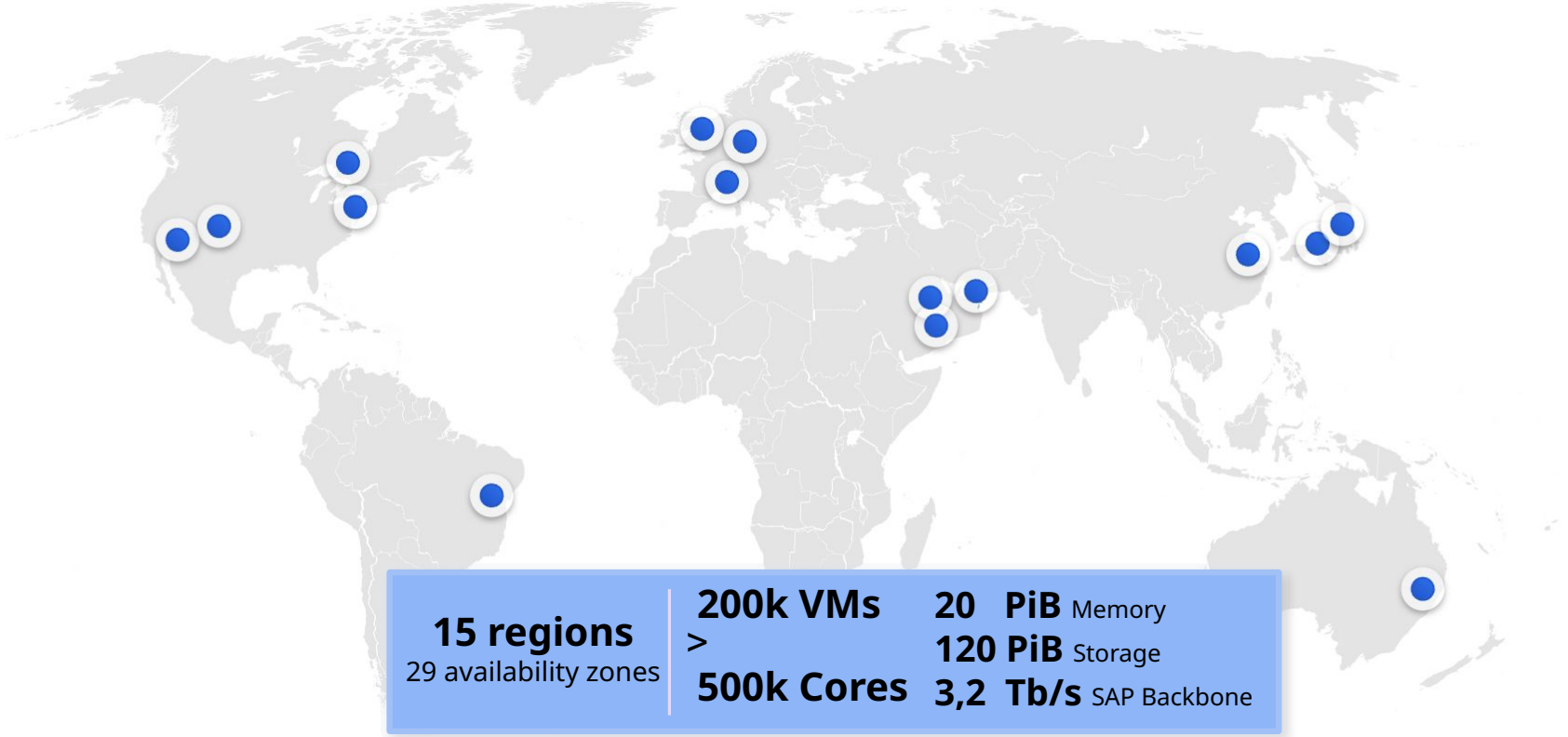
Analyse compute workload of the SAP Cloud infrastructure

Identify scheduling inefficiencies

Provide data-driven insights for new scheduling solutions

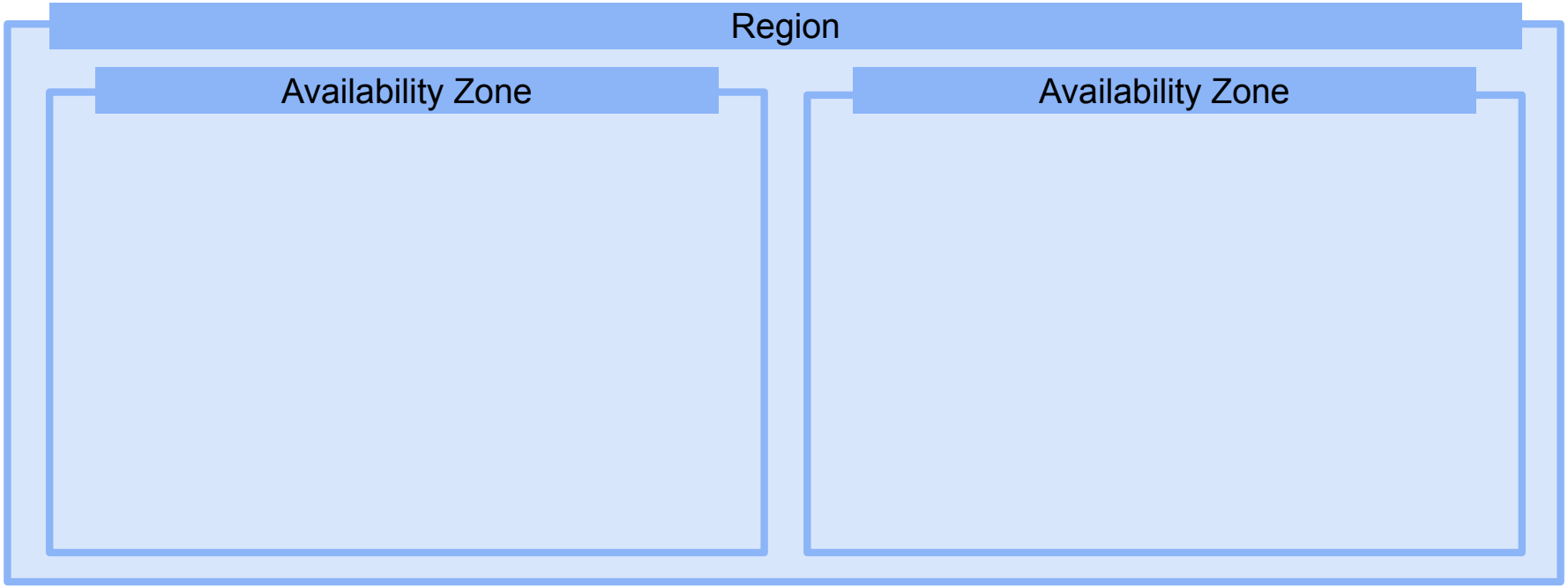
Agenda

The SAP Cloud Infrastructure

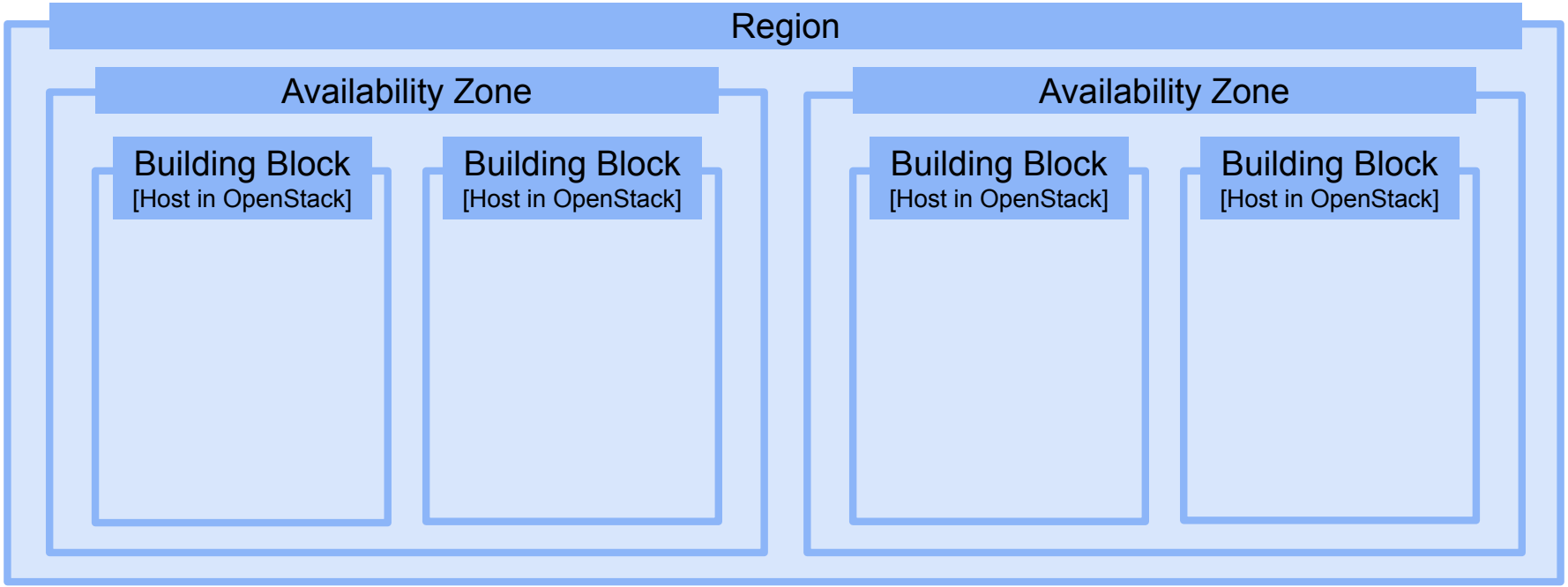


Hierarchical Abstractions

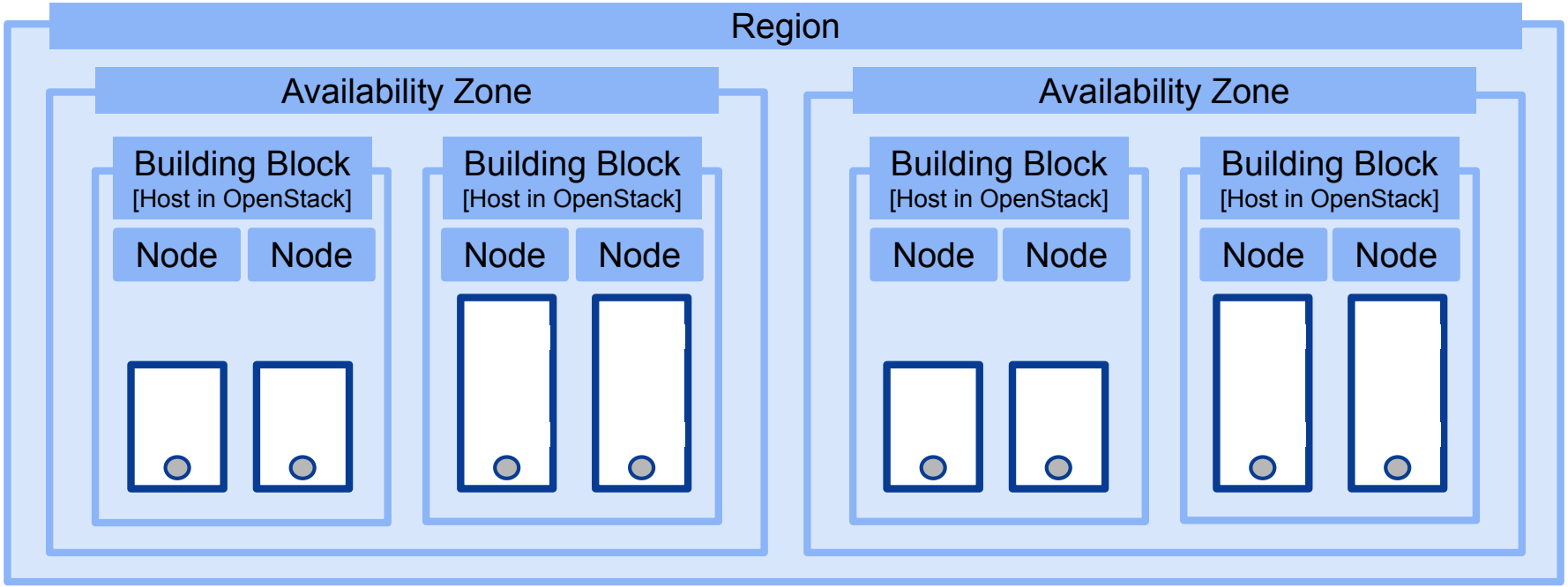
Hierarchical Abstractions



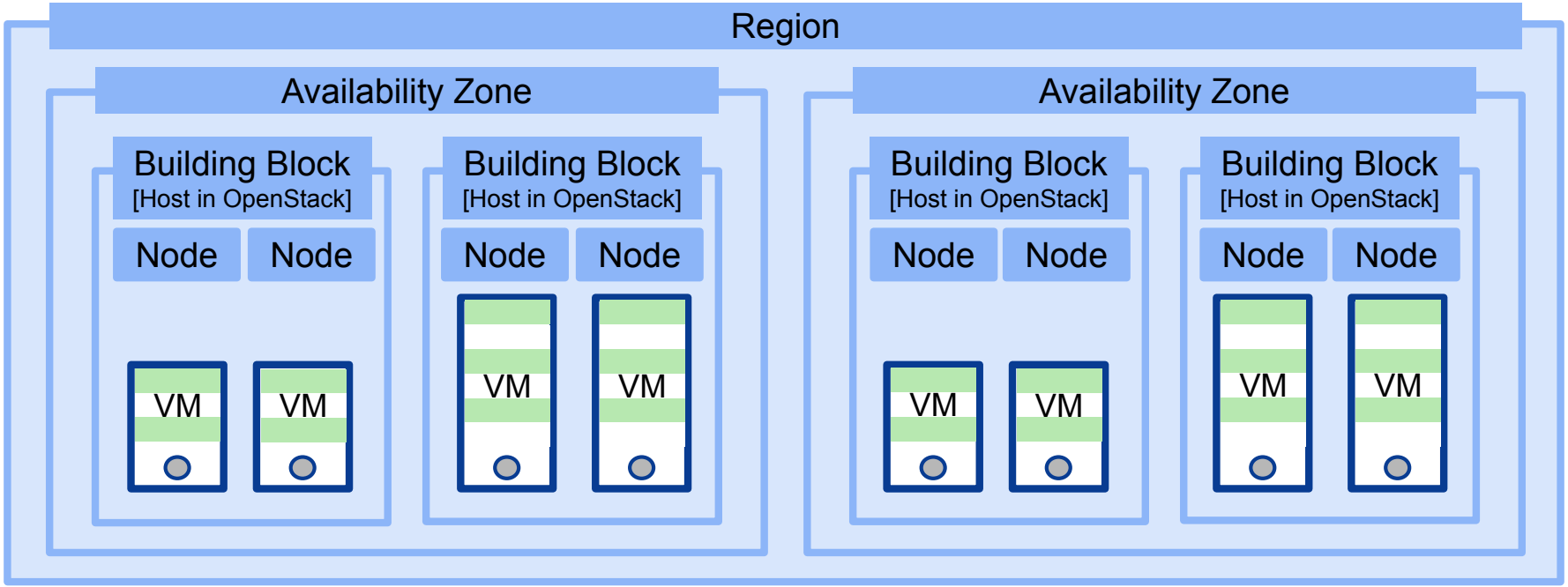
Hierarchical Abstractions



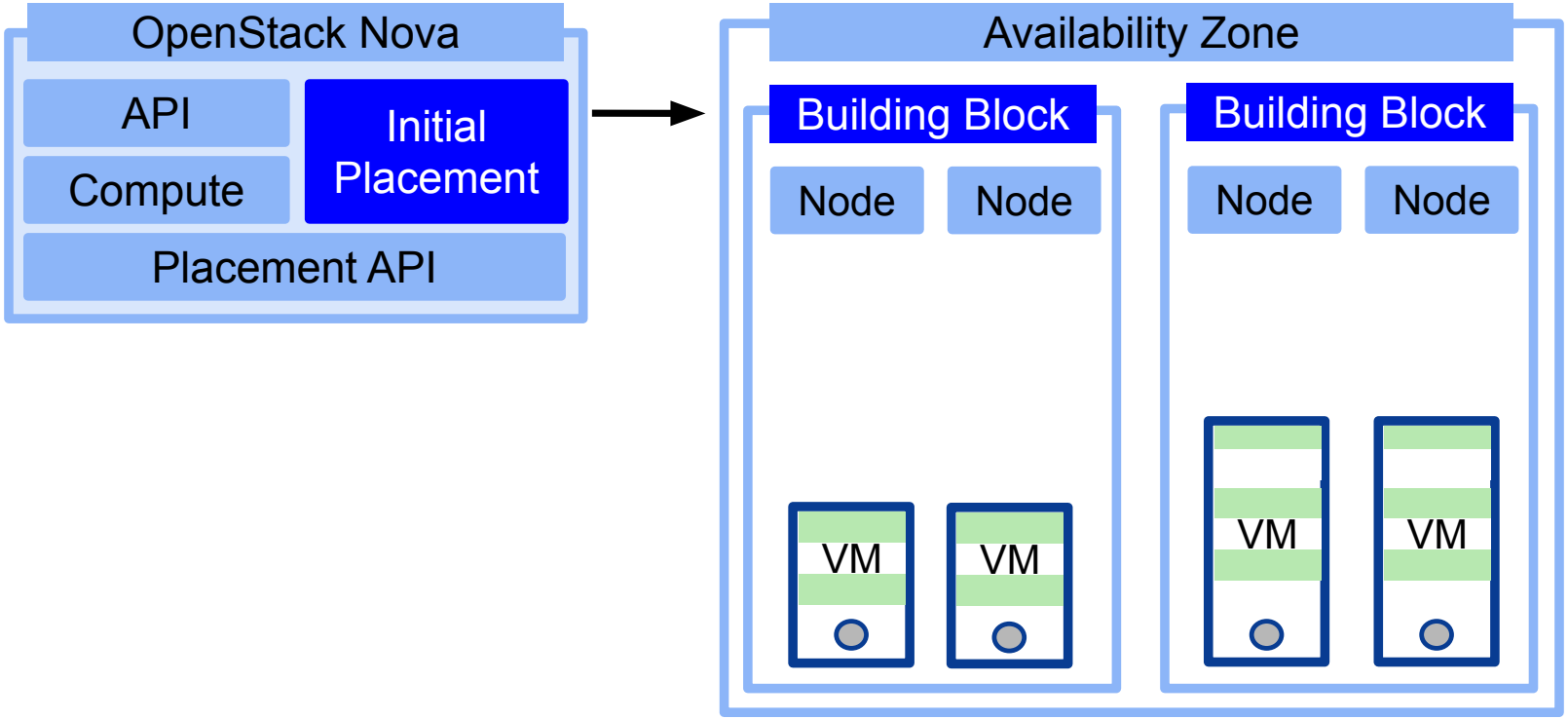
Hierarchical Abstractions



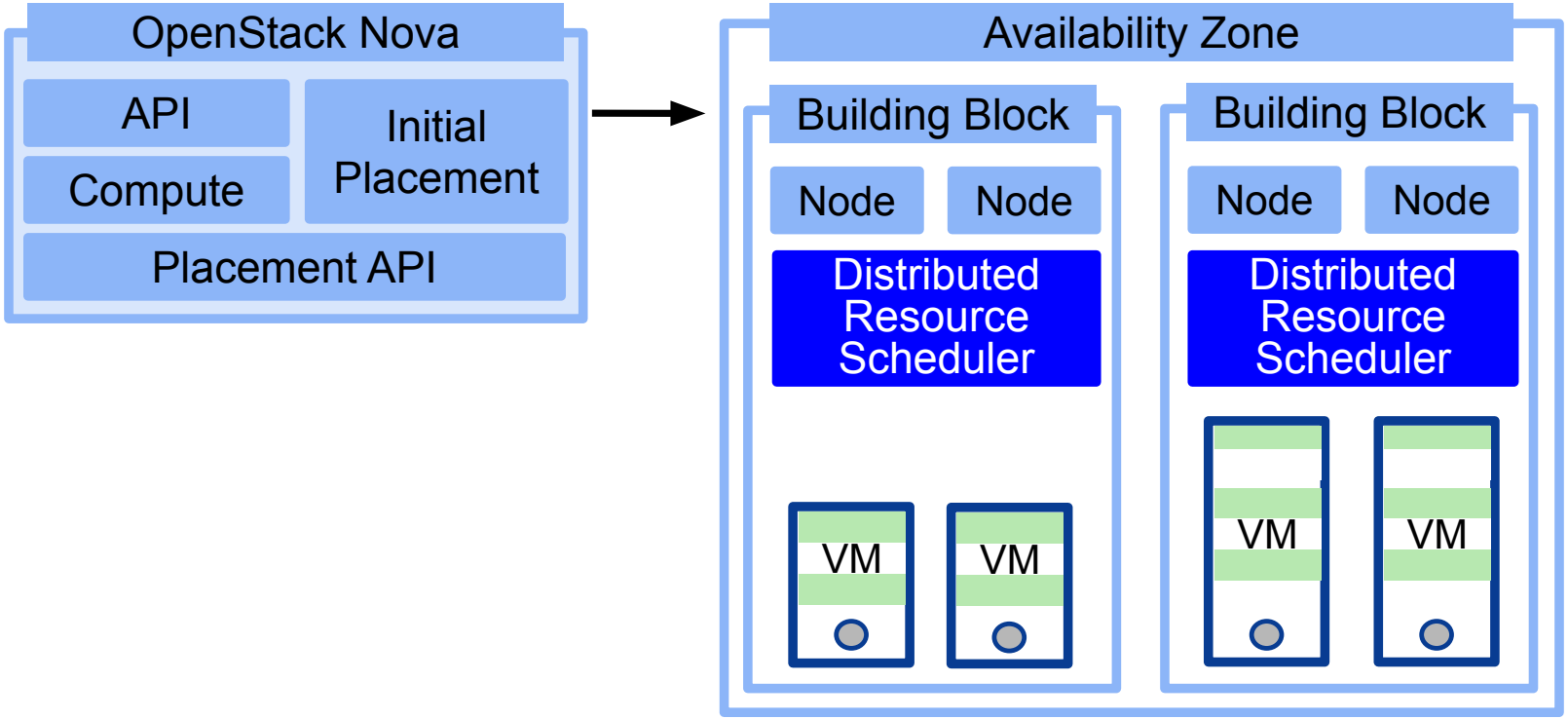
Hierarchical Abstractions



Placement and Scheduling of Components



Placement and **Scheduling** of Components



Agenda

1 Motivation and Background

2 The SAP Cloud Infrastructure

3 The SAP Cloud Infrastructure Dataset

4 Key Findings

5 Conclusion and Outlook

Data Corpus. Publicly Available.

Data Sources and Collection

VMware telemetry (vCenter, vROps)

Prometheus monitoring stack + Thanos for long-term storage

Custom exporters and other tools

DOI: 10.5281/zenodo.17141306

Data Corpus. Publicly Available.

Data Sources and Collection

VMware telemetry (vCenter, vROps)

Prometheus monitoring stack + Thanos for long-term storage

Custom exporters and other tools

Coverage

30-day continuous observation period

Sampling interval from 30s-300s

~48,000 VMs across ~1,800 hypervisors in a single region

DOI: 10.5281/zenodo.17141306

Data Corpus. Publicly Available.

Data Sources and Collection

VMware telemetry (vCenter, vROps)

Prometheus monitoring stack + Thanos for long-term storage

Custom exporters and other tools

Coverage

30-day continuous observation period

Sampling interval from 30s-300s

~48,000 VMs across ~1,800 hypervisors in a single region

Key Metrics

CPU, memory, network, storage, workload composition

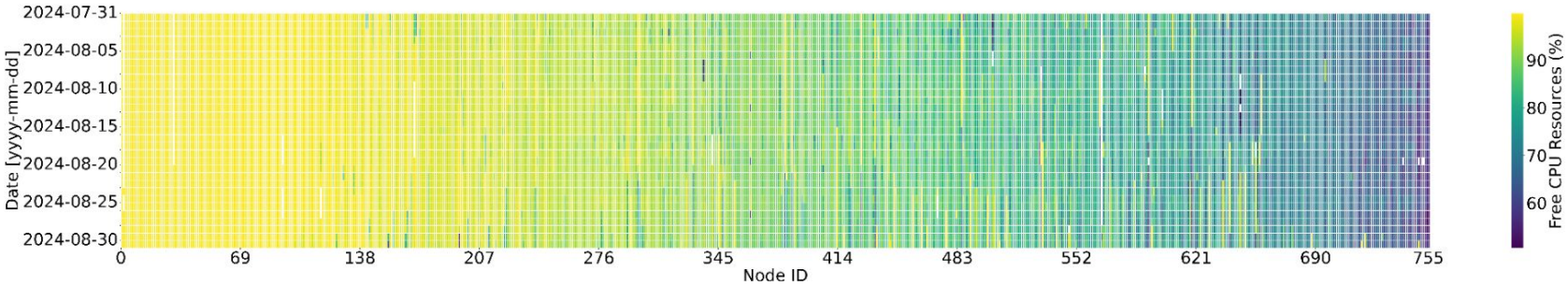
VM- and hypervisor node-level metrics

DOI: 10.5281/zenodo.17141306

Agenda

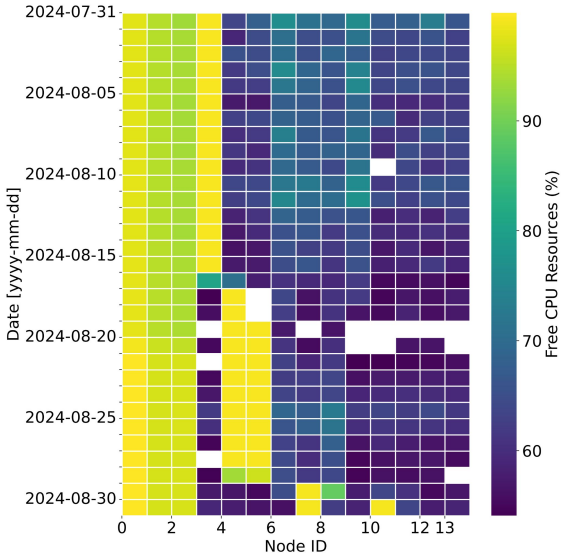
Free CPU Utilization per Node

Across availability zones

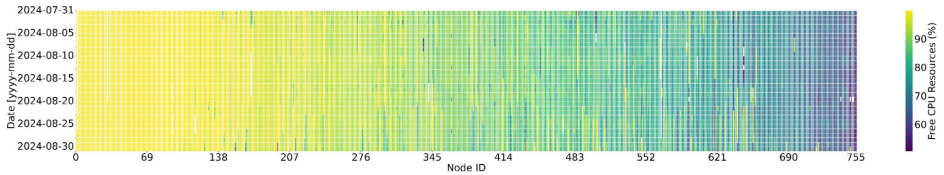


Free CPU Utilization per Node

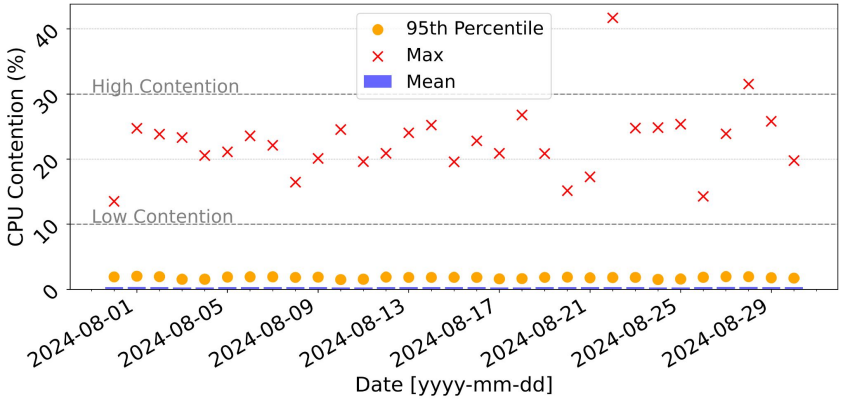
Within a building block



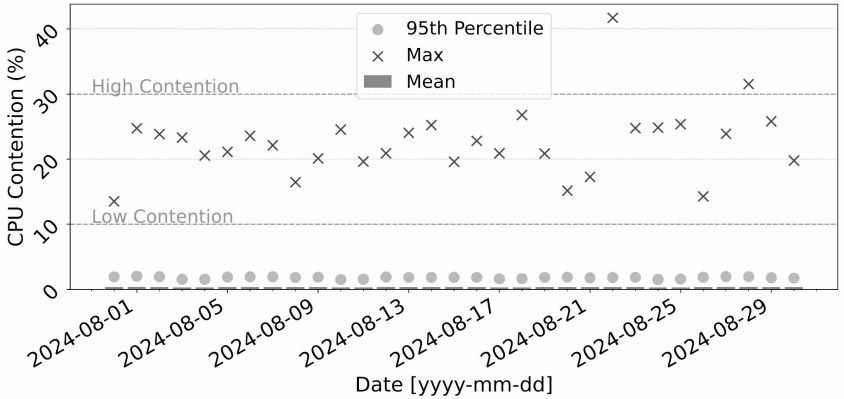
Across availability zones

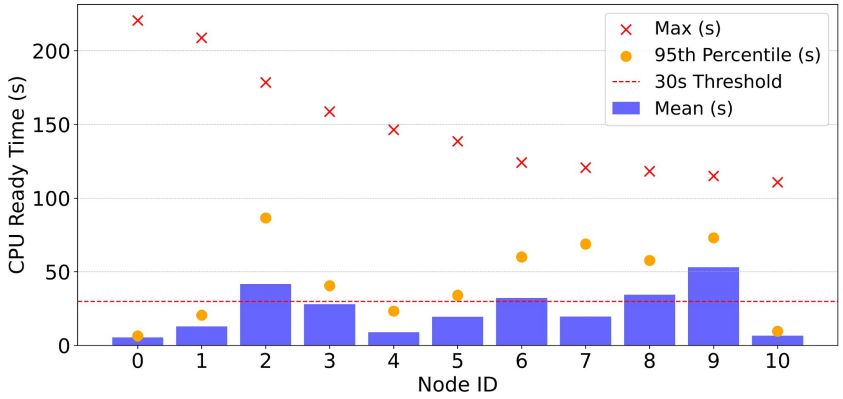


CPU Contention and Ready Time Across Nodes



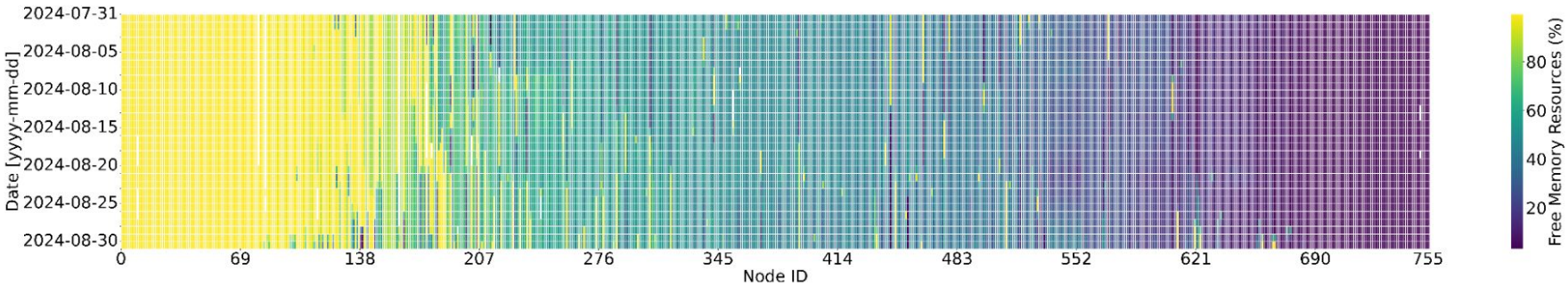
CPU Contention and **Ready Time** Across Nodes



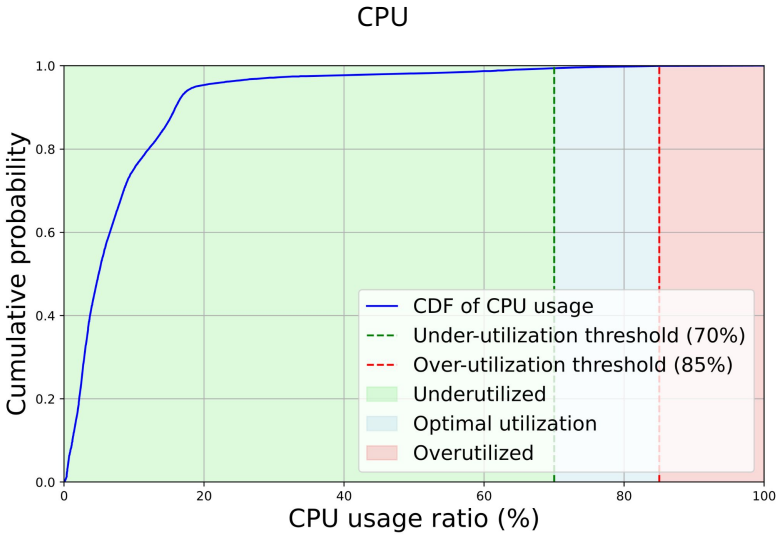


Free Memory Utilization per Node

Across availability zone



Over- and Underprovisioning of VMs

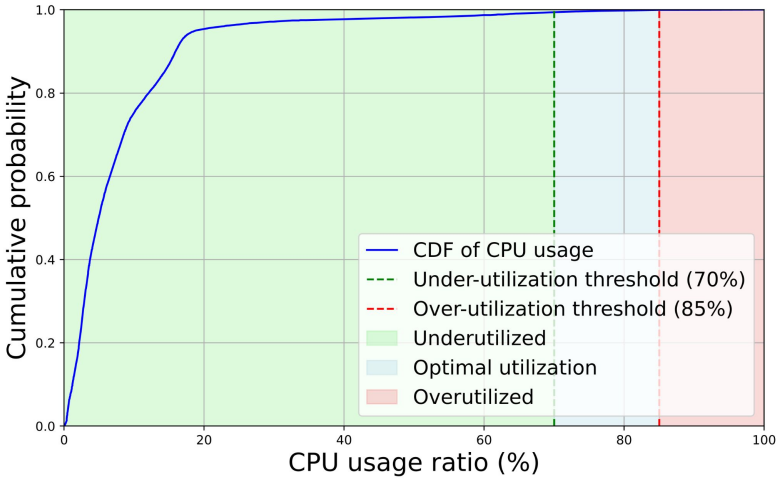


Key Findings

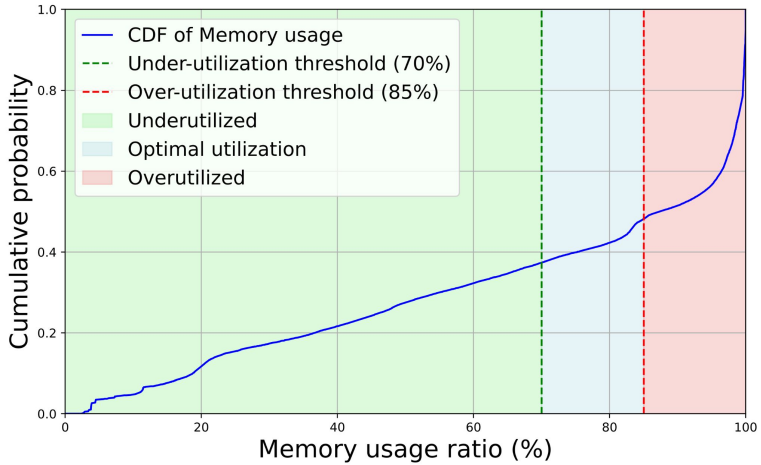
1) Virtual machines are predominantly overprovisioned in terms of CPU resources.

Over- and Underprovisioning of VMs

CPU



Memory

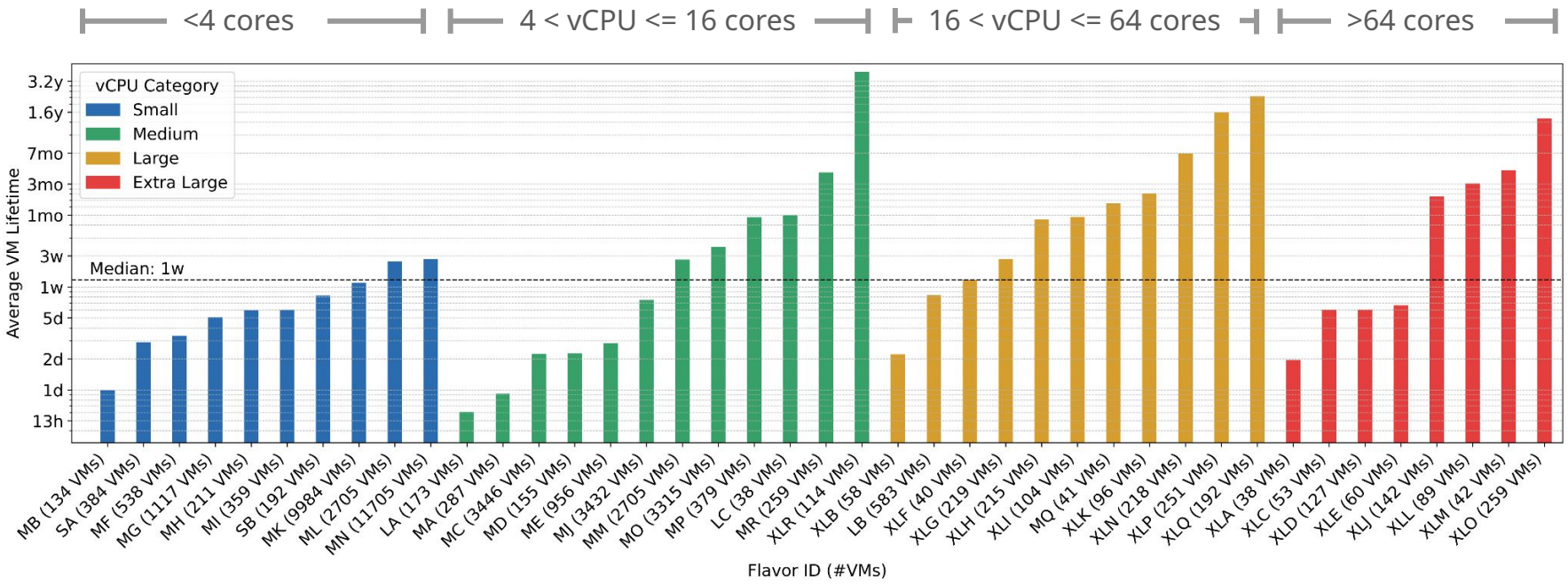


Key Findings

1) Virtual machines are predominantly overprovisioned in terms of CPU resources.

2) Virtual machines are often underprovisioned in terms of memory resources.

VM Lifetime Grouped by vCPU Count



Key Findings - Implications for Scheduling

Combine placement and dynamic rescheduling

Static, one-time placement is often insufficient
Continuous monitoring and selective migrations can mitigate long-term imbalances

Key Findings - Implications for Scheduling

Combine placement and dynamic rescheduling

Static, one-time placement is often insufficient
Continuous monitoring and selective migrations can mitigate long-term imbalances

Consider dominant resources per VM

This helps to avoid inefficiencies
CPU vs. memory trade-off

Key Findings - Implications for Scheduling

Combine placement and dynamic rescheduling

Static, one-time placement is often insufficient
Continuous monitoring and selective migrations can mitigate long-term imbalances

Consider dominant resources per VM

This helps to avoid inefficiencies
CPU vs. memory trade-off

Periodic assessment may be beneficial

Temporal and spatial variability exist because load fluctuates over time and across hypervisor nodes

Key Findings - Implications for Scheduling

Combine placement and dynamic rescheduling

Static, one-time placement is often insufficient
Continuous monitoring and selective migrations can mitigate long-term imbalances

Consider dominant resources per VM

This helps to avoid inefficiencies
CPU vs. memory trade-off

Periodic assessment may be beneficial

Temporal and spatial variability exist because load fluctuates over time and across hypervisor nodes

Workload lifetime data is important

Long-lived VMs dominate capacity
Short-lived VMs increase churn
Awareness improves stability and reduces fragmentation

Key Findings - Implications for Scheduling

Combine placement and dynamic rescheduling

Static, one-time placement is often insufficient
Continuous monitoring and selective migrations can mitigate long-term imbalances

Consider dominant resources per VM

This helps to avoid inefficiencies
CPU vs. memory trade-off

Periodic assessment may be beneficial

Temporal and spatial variability exist because load fluctuates over time and across hypervisor nodes

Workload lifetime data is important

Long-lived VMs dominate capacity
Short-lived VMs increase churn
Awareness improves stability and reduces fragmentation

Data-driven feedback loops are key

Real-time telemetry data should be considered by the scheduling logic
Enables adaptive consolidation and policy adjustments

Agenda

1 Motivation and Background

2 The SAP Cloud Infrastructure

3 The SAP Cloud Infrastructure Dataset

4 Key Findings

5 Conclusion and Outlook

Conclusion and Outlook

Conclusion

Resource scheduling in cloud infrastructures is an open challenge

Real-world data is needed but existing datasets lack focus on VMs

Our data highlights inefficiencies in current VM placement and scheduling

High utilization on some hypervisors while others are underutilized

Resource contention (predominantly CPU) remains a problem

All artifacts, including our dataset, are publicly available

DOI: 10.5281/zenodo.17141306

Conclusion and Outlook

Conclusion

Resource scheduling in cloud infrastructures is an open challenge

Real-world data is needed but existing datasets lack focus on VMs

Our data highlights inefficiencies in current VM placement and scheduling

High utilization on some hypervisors while others are underutilized

Resource contention (predominantly CPU) remains a problem

All artifacts, including our dataset, are publicly available

DOI: 10.5281/zenodo.17141306

Outlook

Improve initial placement, e.g., using historical measures

Implement dynamic scheduling (re-balancing) to improve load distribution