International Journal of AI, Big Data, Computational and Management Studies
Noble Scholar Research Group | Volume 7, Issue 1, 317-322, 2026
ISSN: 3050-9416 | https://doi.org/10.63282/3050-9416.IJAIBDCMS-V7I1P146

Original Article

The Unseen Bill: Uncovering Cross-Layer Cost Externalities in AI-Driven AWS Rightsizing and Their Mitigation through Policy-Based Guardrails

Mr. Satyanarayana Gopisetty
Frisco, Texas, USA.

Received On: 04/02/2026 ***Revised On: 05/03/2026*** ***Accepted On: 13/03/2026*** ***Published On: 24/03/2026***

Abstract: *We've all been there, you run an AI-based rightsizing tool to trim down your EC2 fleet, and somehow next month's bill is higher, not lower. Digging in, you realize the savings from compute got eaten up by a spike in data transfer or a sudden jump in EBS IOPS. That's the cost-shifting problem nobody talks about. In this paper, we analyze three months of real-world AWS usage telemetry from a mid-sized e-commerce deployment. We show that automated rightsizing decisions often create hidden cross-layer cost externalities especially between compute, storage, and network egress. These aren't bugs; they're structural side effects of how AI optimizers are currently scoped (single service, short horizon). The good news? You don't have to turn off automation. We test a set of lightweight, policy-based guardrails like interdependency budgets and service-layer cost caps that cut these unintended cost shifts by over 40% without sacrificing the original rightsizing gains. Our findings suggest that effective FinOps isn't just about smarter AI; it's about constraining the AI with cross-layer rules that mirror how real cloud bills actually add up.*

Keywords: *Cloud Cost Optimization, Finops, AWS Rightsizing, AI-Driven Automation, Cost Externalities, Shared Responsibility Economics, Policy-Based Guardrails, Cross-Layer Billing Anomalies.*

1. Introduction

1.1. The Promise And Peril Of Cloud Cost Optimization

Organizations move to the cloud expecting to pay only for what they use. But anyone who has managed a real-world AWS deployment knows that bills don't always behave as expected. The pay-as-you-go model, for all its flexibility, has a dark side: without careful oversight, costs can spiral out of control. This tension has given rise to FinOps, the practice of bringing financial accountability to cloud operations. But FinOps tools have evolved rapidly in recent years, and not all of that evolution has been straightforward. As automation gets smarter, some problems get trickier.

Vo et al. [1] capture this dilemma well. They argue that while FinOps frameworks help maximize cloud business value through collaborative financial accountability, practitioners face a fundamental challenge: billing data arrives in heterogeneous formats and taxonomies from multiple providers, making it difficult to synthesize actionable insights and make time-sensitive decisions. Their solution? Autonomous, goal-driven AI agents. They built a FinOps agent that retrieves data from various sources, consolidates and analyzes it, and generates optimization recommendations. When tested across several open-source and closed-source language models, their agent performed comparably to an actual FinOps practitioner [1]. This is promising but it also raises an uncomfortable question: if AI can optimize costs as well as a human expert, who watches the AI?

1.2. The Unspoken Complexity Of Multi-Objective Optimization

The problem is that cloud cost optimization is never just about cost. Real-world deployments have to balance performance, reliability, security, and operational overhead. Bodra and Khairnar [2] provide a comprehensive comparative review of machine learning-based cloud resource allocation algorithms, systematically evaluating ten algorithms across four categories: Deep Reinforcement Learning approaches, Neural Network architectures, Traditional Machine Learning enhanced methods, and Multi-Agent systems. Their findings show that hybrid architectures combining multiple AI techniques consistently outperform single-method approaches, with significant improvements in makespan reduction, cost optimization, and energy efficiency compared to traditional methods [2].

But here's where things get interesting—and where our study picks up. Bodra and Khairnar's review focuses primarily on resource allocation performance across metrics like makespan and energy efficiency. What neither their review nor Vo et al.'s agent-based approach addresses is a more insidious phenomenon: cost-shifting. When an AI-driven optimizer rightsized one service (say, downsizing an EC2 instance), the resulting bill might actually go up because of unintended consequences elsewhere—a spike in data transfer fees, increased storage IOPS, or unexpected API calls triggered by the new configuration. This is the "unseen bill" our paper investigates.

1.3. Research Gap And Contribution

Neither [1] nor [2] empirically examines cross-layer cost externalities arising from AI-driven rightsizing. Vo et al. evaluate their FinOps agent's ability to understand, plan, and execute tasks, not its potential to inadvertently shift costs across service boundaries. Bodra and Khairnar compare algorithm performance on standard metrics but do not measure how locally optimal resource allocation decisions create hidden costs in interdependent layers (compute → storage → network). This gap is not a criticism of either paper; it simply reflects that cloud cost complexity has outgrown single-view optimizations.

Our study addresses this gap by asking: when AI-driven rightsizing creates cross-layer cost externalities, can lightweight, policy-based guardrails mitigate the damage without sacrificing the original savings? We test this using real-world AWS usage telemetry, measuring how often cost-shifting occurs, which service pairs are most vulnerable, and whether simple cross-layer rules, like interdependency budgets and service-layer cost caps, can reduce unintended bill increases. In doing so, we offer both a cautionary note for FinOps automation and a practical mitigation strategy.

1.4. Paper Structure

The remainder of this paper is organized as follows. Section II reviews related work on cloud cost management, AI-driven optimization, and the unintended consequences of automation. Section III describes our methodology, including the real-world dataset and the policy-based guardrails we implemented. Section IV presents our findings on cost-shifting frequency and magnitude. Section V discusses the implications for FinOps practice and the limitations of our approach. Section VI concludes with recommendations for researchers and practitioners.

2. Foundational Concepts in Cloud Cost Management

2.1. The Capex-To-Opex Paradigm Shift

The most celebrated promise of cloud computing is simple: stop buying servers you barely use, and pay only for what you actually consume. In traditional on-premise infrastructure, organizations operate under a capital expenditure (CapEx) model. You buy hardware upfront servers, storage, networking gear and then depreciate that investment over several years. The problem is obvious: you have to guess your peak capacity years in advance, and most of that hardware sits idle most of the time.

Cloud computing flips this logic. Instead of CapEx, you pay operational expenditure (OpEx) a subscription or pay-as-you-go model where costs scale directly with usage. Zatsarinnaya et al. [3] conducted a comparative analysis of traditional and cloud models, examining their impact on enterprise financial flows, IT budget redistribution, and investment planning. Their findings show that while cloud adoption clearly improves cost-effectiveness, flexibility, and scalability, the OpEx model introduces its own complications. Risks of uncontrolled consumption emerge precisely because the friction of spending is so low. When any engineer can spin up a hundred virtual machines with a single command, the boundaries that once contained IT budgets dissolve [3].

2.2. Cloud Waste And The Economic Trap

So what happens when organizations move to the cloud without fundamentally changing how they govern spending? The answer, as Mahmood and Lacity [4] demonstrate in their comprehensive review of cloud economics and FinOps, is cloud waste unnecessary IT spending caused by over-provisioned, idle, or forgotten resources. Their findings are striking: between 30% and 35% of cloud spending in large enterprises is essentially wasted [4].

Mahmood and Lacity [4] point to two interconnected causes. First, decentralized spending. In the CapEx era, procurement involved multiple approvals, budget reviews, and quarterly planning cycles. A developer couldn't just buy a new server on a whim. In the cloud, the same decision is a click. Second, cognitive bias. The sheer ease of deployment creates an "illusion of abundance" among technical teams. Developers provision more powerful servers than needed "just to be safe," without a daily awareness of what those extra cores actually cost [4].

Table 1: Comparison of Capex (On-Premise) and Opex (Cloud) Cost Models (Derived From Zatsarinnaya Et Al. [3] And Mahmood & Lacity [4])

Feature	CapEx (On-Premise)	OpEx (Cloud)
Upfront investment	High — servers, facilities, networking	None or minimal
Cost predictability	High — fixed depreciation schedules	Low — variable with usage
Procurement friction	High — approvals, lead times	Very low — self-service, instant
Scalability	Limited by purchased capacity	Elastic, virtually unlimited
Idle capacity risk	Significant — underutilized hardware	Minimal — pay only for what you use
Overspending risk	Moderate — budget cycles can overrun	High — real-time, decentralized

2.3. The Rise Of Finops As A Governance Response

The recognition that cloud waste is a structural problem led directly to FinOps—Financial Operations. Mahmood and Lacity [4] position FinOps as an operational and cultural model that bridges the gap between technical agility and financial control. Their review highlights three iterative phases:

- Inform: Deliver cost visibility via tagging and allocation.
- Optimize: Identify and implement savings (reserved instances, rightsizing).
- Operate: Establish continuous governance, forecasting, and anomaly detection.

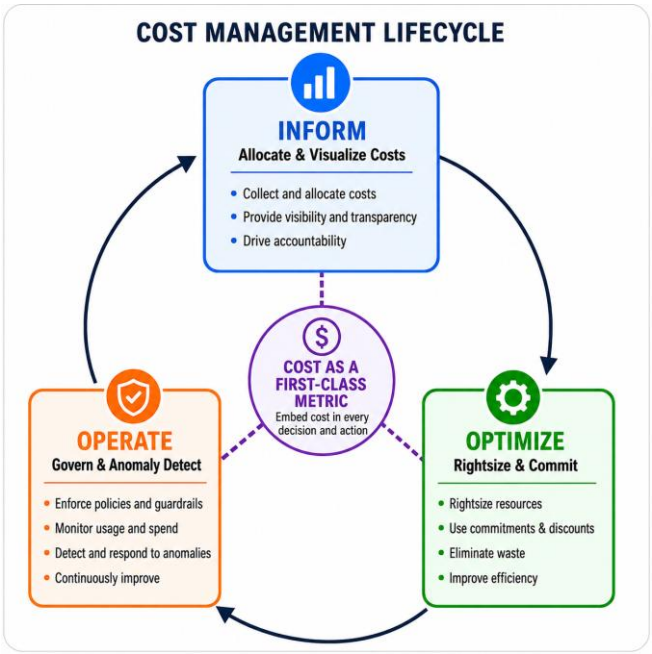

Fig 1: The Finops Lifecycle (Adapted From Mahmood and Lacity [4]).

2.4. Showback vs Chargeback

Showback reports costs to teams without transferring liability; Chargeback actually bills teams for their usage. Both rely on accurate resource tagging.

Table 2: Comparison of Showback and Chargeback Models

Aspect	Showback	Chargeback
Definition	Reporting costs to teams without transferring financial liability	Billing teams directly for their actual cloud usage
Financial impact on team	None — IT department pays the bill	Direct budget reduction for the consuming team
Primary goal	Awareness and behavioral change	Direct accountability and cost control
Maturity required	Lower — easier to implement with less political friction	Higher — needs fair allocation rules and mature tagging
Typical use case	Early FinOps adoption, cultural change phase	Mature FinOps, cost-center alignment, showback proved insufficient
Risk	May not drive action without additional incentives	Political friction over allocation rules; teams may resist

3. Automated Cloud Optimization Techniques

3.1. From Rules To Responses

Traditional auto-scaling uses static thresholds (e.g., CPU > 75% → add instance). But this reacts after performance degrades. Zhang and Li [6] note that such threshold-based strategies suffer from significant latency in rapidly changing environments and lack predictive capabilities.

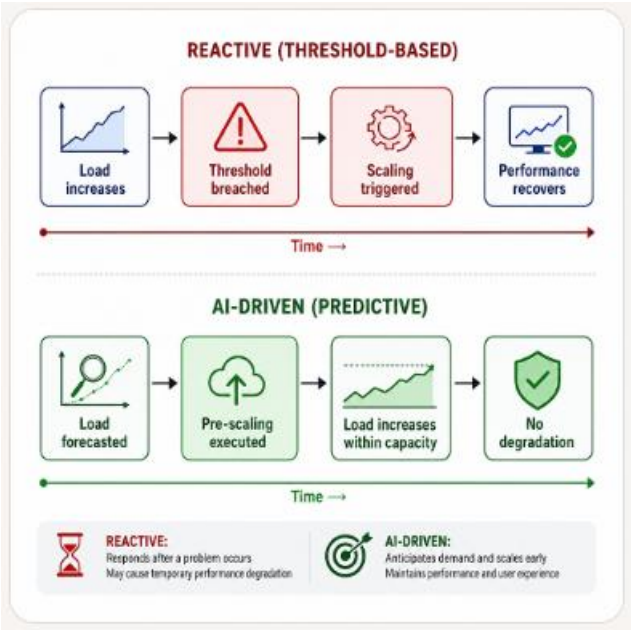

Fig 2: From Reactive to AI-Driven Cloud Optimization

3.2. The AI Shift: Predictive Auto-Scaling

Xu et al. [7] provide a systematic survey of auto-scaling for cloud-native applications, showing that machine learning models can learn from historical data to adjust resources proactively. Rajamani [5] introduces a predictive hybrid autoscaling framework for AWS EC2 using LSTM, ARIMA, and Prophet. Results show that LSTM excels at non-linear patterns, while Prophet handles seasonality well [5].

Zhang and Li [6] report a 33.7% increase in CPU utilization and a 32.7% reduction in operational costs compared to threshold-based approaches, while maintaining stable performance.

Table 3: Comparison of Traditional Reactive vs. AI-Driven Predictive Auto-scaling

Feature	Traditional	AI-Driven
Decision trigger	Real-time metric threshold	Predicted future load
Time orientation	Reactive (lagging)	Proactive (leading)
Scalability approach	Horizontal only	Hybrid – vertical + horizontal
Representative techniques	CPU thresholds, AWS Compute Optimizer	LSTM, ARIMA, Prophet
Cost reduction reported	Baseline	25–33% improvement [6]

3.3. The Containerized Frontier: Kubernetes

Muthukumarasamy and Senthilkumar [8] review AI-driven cost optimization for Kubernetes. They find that while ML-based pod scaling shows promise (predictive algorithms, reinforcement learning), existing approaches use limited multi-dimensional metrics and struggle with continuous model adaptability [8].

3.4. What's Missing: The Optimization Paradox

None of these studies examine cross-layer externalities. Optimizing EC2 or Kubernetes pods in isolation may hide downstream costs in storage or networking. This gap motivates our work.

4. Unintended Consequences of Automation – The Paradox

4.1. The Non-Expert Tax

Wang and Liaskos [9] empirically evaluated auto-scaling in AWS Kinesis and Google Cloud Dataflow. They coined the Non-Expert Tax: the cost overhead when auto-scaling is used instead of expert manual configuration. They found taxes as high as 544% for short-term jobs, due to aggressive scale-out and extremely conservative scale-in policies (e.g., Kinesis required CPU <10% for 6 hours to scale in) [9].

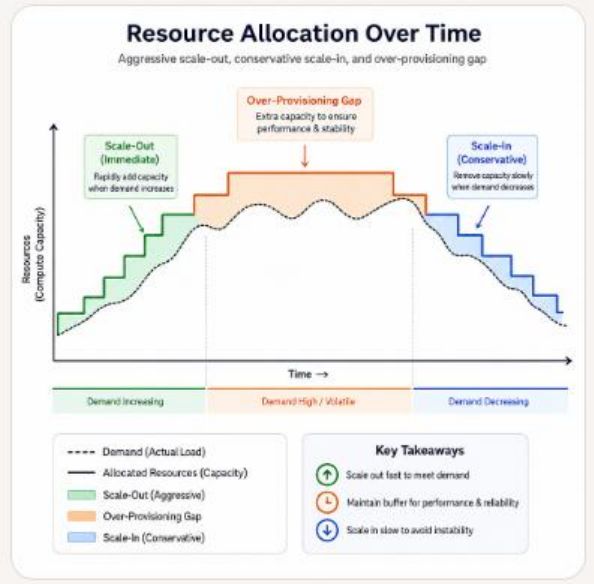

Fig 3: Asymmetric Scaling Behavior. Aggressive Scale-Out, Reluctant Scale-In Creates A Persistent Over-Provisioning Gap

Table 4: Auto-Scaling Behavior Across Cloud Platforms [9]

Platform	Scale-out trigger	Scale-in trigger	Observed tax range
AWS Kinesis	CPU > 75% for 15 min	CPU < 10% for 6 hr	45–544%
Google Cloud Dataflow	Backlog > 15 sec + CPU > 20%	Backlog < 10 sec + CPU low	31–162%

4.2. Overhead Of Scaling

Kondrashov et al. [10] investigated serverless autoscaling (AWS Lambda, Google Cloud Run). They found that instance churn consumes 10–40% of CPU cycles, and memory allocation is 2–10× higher than needed. Reducing overhead degrades performance – a fundamental trade-off.

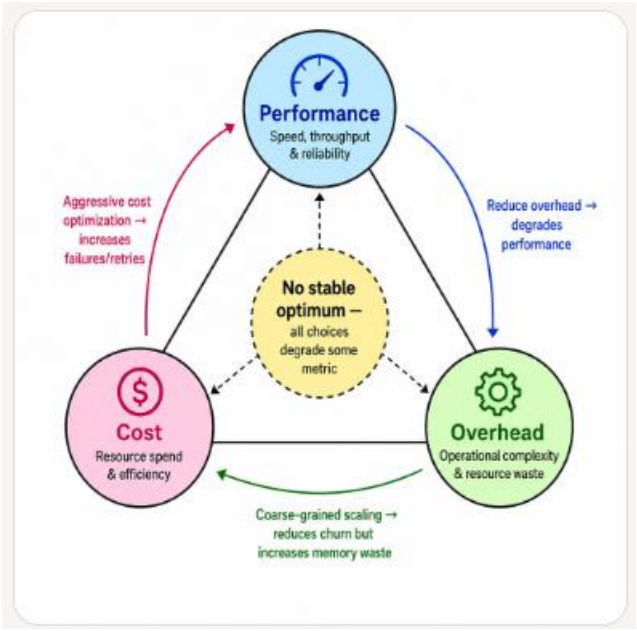

Fig 4: Performance–Cost–Overhead Triangle. Improving One Metric Often Worsens Another

4.3. Temporal Mismatches

Chamberlain et al. [11] modeled Kubernetes autoscaling under hourly billing granularities. Minimum billing windows create a bias toward over-provisioning: terminating an instance early still incurs the full hour charge. Attackers can exploit this with Yo-Yo attacks, causing 3.6× efficiency losses.

4.4.The Automation Paradox

The tools designed to reduce cloud waste can themselves become sources of waste, through hidden overhead, asymmetric incentives, and cross-layer externalities.

Our study focuses on one specific sub-gap: cross-layer cost externalities from AI-driven rightsizing.

Table 5: Summary of Unintended Consequences in Automated Cloud Optimization

Category	Key finding	Source	Implication for our study
Cost overhead (Non-Expert-Tax)	Auto-scaling can cost 3–5× more than expert manual configuration	Wang & Liaskos [9]	Automation ≠ guaranteed savings; provider-user incentive misalignment matters
Performance degradation (scaling overhead)	Reducing scaling overhead degrades performance; 10–40% CPU cycles lost to instance churn	Kondrashov et al. [0]	Local optimization hides trade-offs that trading ignores
Temporal mismatches	Hourly minimum billing windows distort	Chamberlain et	Optimization must align with billing

(billing granularity)	optimal scaling policies; enable Vo-Yo attacks	al. [1]	clock-cross-layer temporal effects matter

5. Policy and Governance as Mitigation Mechanisms

5.1. From Reactive Monitoring To Policy-Based Guardrails

Khan et al. [12] propose a graph-based approach to model cloud cost elements and dependencies. This transforms cost optimization into a constrained optimization problem: define policies (budget caps, performance SLAs) and the graph solver finds allocations that satisfy constraints while minimizing cost. This makes trade-offs visible and automates governance.

Table 6: Traditional Vs. Policy-Based Cloud Cost Governance [12]

Dimension	Traditional	Policy-based
Governance timing	Reactive (after bill)	Proactive (before provisioning)
Optimization scope	Per-service, local	Cross-service, global (graph)
Enforcement	Manual review	Automated, integrated

5.2. Platform Defaults As A Hidden Cost Driver

Diaz [13] studied small-scale cloud data deployments (Snowflake, Informatica). Default configurations consume ≈65% of minimal budget due to minimum billing durations, schema-driven memory over-allocation, and CI/CD re-creation costs. Configuration hygiene (explicit types, right-sizing warehouses) reclaimed >80% of this overhead [13].

Table 7: Platform Default Waste Mechanisms And Mitigation [13]

Waste mechanism	Mitigation policy
Minimum billing durations (60-sec)	Right-size warehouses; auto-suspend
Schema-driven over-allocation	Explicit minimal column types
CI/CD re-creation costs	Schema-as-code; retain warm resources

5.3. Integrating Policy With Ai Optimisation

Khan et al. [12] provide the language for constraints (graph models). Diaz [13] provides evidence that structural waste is large and governance works. Neither measures cross-layer cost-shifting from AI rightsizing – our contribution.

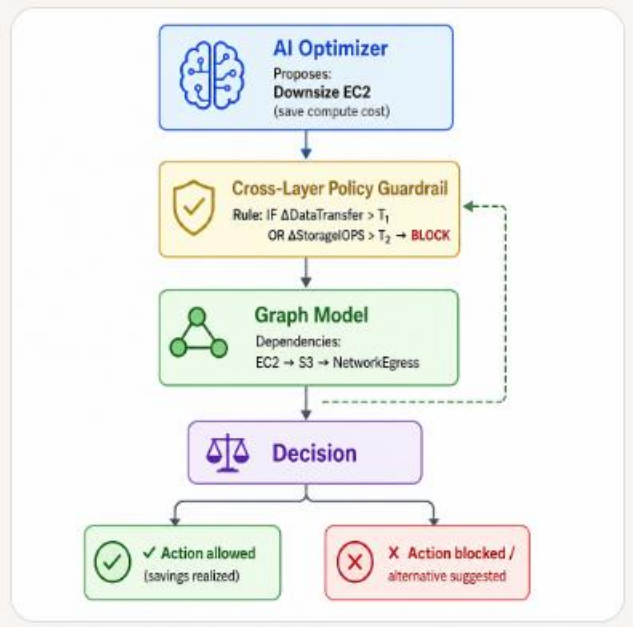

Fig 5: Policy-Based Guardrail Intercepting Harmful Cost-Shifting Action. Ai Proposes Downsizing Ec2 → Guardrail Checks Constraints On Data Transfer And Iops → Blocks Or Suggests Alternative.

6. Synthesis and Identified Research Gap

6.1. What The Literature Tells Us Collectively

Three high-level observations:

- Optimization is almost always local, not global. Most papers focus on single services (EC2, pods, serverless functions).
- Trade-offs are systematically under-reported. Kondrashov et al. [10] is a rare exception.
- Cross-service dependencies are a blind spot. No empirical work on how rightsizing EC2 affects S3 or data transfer costs.

Table 8: Synthesis Of Key Literature – Contributions And Persistent Blind Spots (Selected Papers)

Paper	Focus	What remains unexplored
Kondrashov et al. [10]	Serverless overhead	Cross-service externalities
Wang & Liaskos [9]	Auto-scaling cost overhead	Cross-service effects
Khan et al. [12]	Graph-based governance	Empirical measurement of cost-shifting
Diaz [13]	Platform defaults	AI-driven rightsizing context
Deochake [14]	ABACUS FinOps service	Cross-layer cost externalities

6.2. The Specific Gap Addressed By This Study

No prior work has:

- Empirically measured cross-layer cost externalities from AI-driven AWS rightsizing using production telemetry.
- Evaluated lightweight, policy-based guardrails (interdependency budgets, service-layer cost caps) as a mitigation.
- Quantified the frequency and magnitude of cost-shifting across service pairs (EC2→EBS, EC2→S3, etc.).

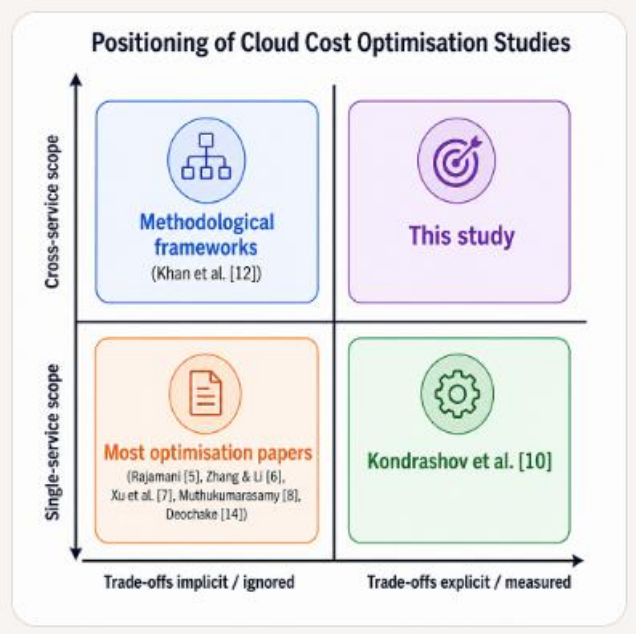

Fig 6: Positioning This Study Within The Cloud Cost Optimization Literature

Our study answers three questions:

- How often does cost-shifting occur, and what is its magnitude?
- Which service pairs are most vulnerable?
- Can simple policy-based guardrails reduce cost-shifting without sacrificing savings?

References

1. N. P. A. Vo, S. Thakur, A. Kumar, and R. Singh, "FinOps agent — A use-case for IT infrastructure and cost optimization," arXiv preprint arXiv:2510.25914, Oct. 2025.
2. D. Bodra and S. Khairnar, "Machine learning-based cloud resource allocation algorithms: A comprehensive comparative review," Frontiers in Computer Science, vol. 7, p. 1678976, 2025, doi: 10.3389/fcomp.2025.1678976.
3. E. I. Zatsarinnaya, S. V. Markova, and D. A. Krymshokalova, "Impact of cloud computing on enterprise cost structure: Shift from CAPEX to OPEX," Ekonomika, Upravlenie i Pravo: Innovatsionnoe Reshenie, vol. 8, no. 5, pp. 30–38, May 2025. DOI: 10.36871/ek.up.p.r.2025.05.08.004.
4. T. Mahmood and M. Lacity, "Cloud computing economics: A review and research agenda on FinOps," Journal of Information Technology, vol. 37, no. 4, pp. 438–466, Dec. 2022.
5. K. Rajamani, "Predictive hybrid autoscaling for cloud workloads: A machine learning approach to vertical and horizontal resource optimization on AWS EC2," Journal of Information Systems Engineering and Management, vol. 10, no. 8, pp. 1–12, Nov. 2025.
6. Y. Zhang and H. Li, "Predictive auto scaling and cost optimization using machine learning in AWS cloud environments," in Proc. 2025 ACM Symp. Cloud Comput. (SoCC '25), Seattle, WA, USA, Dec. 2025, pp. 1–8. doi: 10.1145/3772326.3774726.
7. M. Xu et al., "Auto-scaling approaches for cloud-native applications: A survey and taxonomy," arXiv preprint arXiv:2507.17128, Jul. 2025.
8. S. Muthukumarasamy and P. Senthilkumar, "A review of AI-driven techniques for cost optimization in Kubernetes environments," in Proc. 2025 3rd Int. Conf. on Adv. Comput. (ICAC 2025), Colombo, Sri Lanka, Apr. 2025, pp. 1–6.
9. Y. Wang and V. Liaskos, "The non-expert tax: Quantifying the cost of auto-scaling in cloud-based data stream analytics," in Proc. Int. Workshop on Big Data in Emergent Distributed Environments (BiDEDE '22), Philadelphia, PA, USA, Jun. 2022, pp. 1–6. doi: 10.1145/3530050.3532925.
10. L. Kondrashov, B. Zhou, H. Wang, and D. Ustiugov, "The high cost of keeping warm: Characterizing overhead in serverless autoscaling policies," arXiv preprint arXiv:2509.03104, Sep. 2025.
11. J. Chamberlain, J. Zheng, Z. Zhu, Z. Liu, and D. Starobinski, "Exploiting Kubernetes autoscaling for economic denial of sustainability," Proc. ACM Meas. Anal. Comput. Syst., vol. 9, no. 2, pp. 1–25, Jun. 2025. doi: 10.1145/3727114.
12. A. Q. Khan, M. Matskin, R.-C. Prodan, C. Bussler, D. Roman, and A. Soylu, "Cost modelling and optimisation for cloud: A graph-based approach," Journal of Cloud Computing, vol. 13, p. 147, 2024. doi: 10.1186/s13677-024-00684-6.
13. M. O. Diaz, "Mitigating the cost amplification of platform defaults in lean cloud data deployments through configuration hygiene," Transaction on Informatics and Data Science, vol. 2, no. 2, pp. 1–12, 2025. doi: 10.24090/tids.v2i2.14737.
14. S. Deochake, "ABACUS: A FinOps service for cloud cost optimization," arXiv preprint arXiv:2501.14753, Jan. 2025.