In cloud infrastructure, it's a common reflex to optimize costs by selecting the cheapest virtual machine (VM) instance type available. However, this approach frequently misses the forest for the trees. The "cheapest" VM in terms of list price or on-demand hourly rate often leads to hidden costs — in operational overhead, burst charges, degraded performance, and ultimately, more expensive workloads. Drawing from experience across AWS, Azure, and Google Cloud platforms, this post explores why selecting a VM purely based on price can inflate your total cost of ownership (TCO), how cloud providers' shared CPU models influence performance and uptime, and how to better combine metrics and tooling for truly cost-effective workload placement.

The Fallacy of "Cheapest VM = Cheapest Workload"

When evaluating VM pricing, it's tempting to focus on nominal hourly rates in a single region without considering additional dimensions such as:

- **Region pricing variations:** Same VM types can cost significantly different amounts depending on geographic region.
- **Resource burst capabilities and burst charges:** "Cheap" VMs with shared CPUs or burstable performance modes may incur throttling or hidden costs during peak usage.
- **Operational overhead:** Extra engineering efforts for managing unstable or underperforming VMs increase your maintenance burden and reduce reliability.
- **Storage and egress costs:** Ignoring these often leads to incremental charges overshadowing compute savings.

Failing to holistically evaluate the true cost of running a workload on a given VM instance type can paradoxically increase your cloud bill.

Always-On Small Services: The Hidden Waste

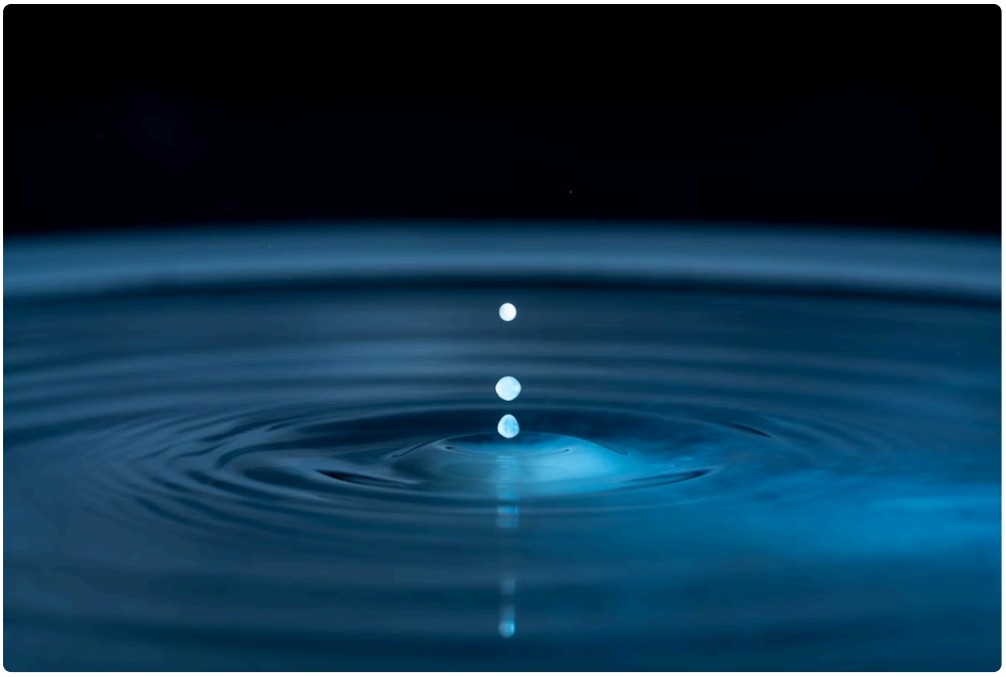
Many teams deploy small VMs for always-on internal tooling, monitoring agents, CI/CD runners, or low-traffic APIs, assuming "less is more" on cloud spend. But these constant steady-state resources contribute quietly yet significantly to waste:

- **Small VMs tend to have shared CPUs or burstable credit systems.** Performance is not guaranteed, leading to unpredictable latency spikes and retries.
- **Underprovisioned VMs create operational overhead.** DevOps and SRE teams spend disproportionate time troubleshooting flaky services, increasing indirect costs.
- **Multiple small VMs often add up to more than a few properly sized instances.** Fragmentation can inflate licensing, monitoring, and patching costs.

Tools like AWS Compute Optimizer and Azure Advisor can help identify these hidden inefficiencies by analyzing the actual utilization over time and recommending right-sized VMs or autoscaling.

Shared CPU Definitions Differ By Provider — What Exactly Are You Buying?

A core misunderstanding is treating vCPU counts as performance guarantees. In reality, the definition of a vCPU and how CPU resources are shared vary widely between AWS, Azure, and Google Cloud:



Cloud Provider vCPU Definition Shared CPU Model Implications AWS One hyperthread of a dedicated core (Intel/AMD) Many instance types are directly allocated. Burstable types (T-series) use CPU credit system. Burst credits allow short spikes; once depleted, throttling leads to latency. Azure One hyperthread assigned to the VM Burstable VMs with Baseline CPU percentage and credit accumulation. Similar burst credit behavior; baseline may only be ~20%, impacting always-on workloads. Google Cloud One logical CPU core Shared core models exist but are not market dominant; sustained use discounts apply for steady workloads. More transparent CPU allocation; sustained discounts beneficial for consistent usage.

Treating vCPUs as "cores" without regard for the provider's CPU sharing model leads to incorrect sizing and potential performance surprises.

Measure Peaks With the Right Observation Window

One of the most annoying anti-patterns I encounter is deciding VM size or instance family based on *average* CPU utilization over short sampling windows (5–15 minutes) during non-peak periods. The result? VM undersizing, throttling, and ballooning costs from burst charges.

To optimize properly, your monitoring and capacity planning must:

- **Focus on the P95 and P99 usage percentiles, not averages.** The 95th and 99th percentile CPU metrics better represent the workload's true peak consumption.
- **Use long enough observation windows (days to weeks).** This ensures you catch daily peaks, weekly batch jobs, or unexpected spikes in demand.
- **Analyze spike durations, not just peak values.** Short bursts of high CPU may be tolerable on burstable instances, but sustained spikes require dedicated CPU resources.

Only after understanding these characteristics can you make cost-effective decisions and select instance types that balance price with performance.

How AWS Compute Optimizer and Azure Advisor Can Help

Both AWS and Azure provide tailored tools designed to analyze historical utilization data and recommend VM instance types that align with actual workload patterns:

- **AWS Compute Optimizer**: Leverages CloudWatch metrics to recommend right-sizing across CPU, memory, and network. Particularly useful to identify inefficient burstable instances and oversized large instances.
- **Azure Advisor**: Offers cost and performance recommendations including underutilized or oversized VMs, and flags burstable VMs exceeding baseline usage.

These tools recommend instance families and sizes based computingforgeeks.com on P95 and P99 metrics, rather than averages, helping reduce both waste and risk.

Region Pricing and Burst Charges: Hidden Cost Factors

Finally, when selecting your VM sizes and regions, keep in mind:

- **Region pricing variability:** The same VM in us-east-1 can cost 10–30% less than in europe-north or asia regions. Ignoring region price differentials can invalidate cost savings from instance selection.
- **Burst charges and throttling penalties:** Burstable VM types like AWS T3/T4 or Azure B-series initially appear cheap but may incur performance degradation during peak CPU usage or extra charges on some platforms.
- **Operational overhead:** Managing many smaller or burstable instances can increase patching, monitoring, and troubleshooting effort, which often does not show up in raw compute cost metrics.

Conclusion: Optimize Workloads, Not Just Instance Price

Selecting the “cheapest” VM based solely on listed price and nominal specs is a costly mistake. Instead, approach cloud cost optimization with these principles:

1. Understand your workload’s CPU usage distribution and peak patterns using high-percentile metrics (P95, P99) over meaningful intervals.
2. Recognize that vCPU counts are not equal across cloud providers; differentiate between dedicated and shared CPU models to properly size your VMs.
3. Leverage tools like AWS Compute Optimizer and Azure Advisor to inform sizing decisions using actual utilization data, not averages.
4. Account for region-specific pricing and burst or throttling effects that can increase costs beyond hourly rates.
5. Factor operational overhead from distributed small instances into your cost analysis to avoid hidden labor expenses.

By focusing on optimizing your workload’s resource profile and operational impact rather than chasing the lowest sticker price on VM instances, you’ll avoid expensive surprises and drive real cloud cost savings.


104
103
102
101
100
99
98
97
96
95
94
86
85
84
83