Performance Benchmarking of AWS EC2 Instance Types

Course: Cloud Computing

Student Name: Longjie Su

Student ID: 225040158

1．Overview

In today's cloud service environment, providers offer hundreds of instance configurations, making it extremely challenging to select the optimal instance for specific applications. Poor instance selection can lead to serious consequences, including performance bottlenecks and application delays due to underprovisioned resources, or cost inefficiencies resulting from overprovisioned resources. Moreover, choosing instance types that do not align with workload characteristics can trigger scalability issues.

To address this challenge, this project aims to systematically evaluate and compare three common EC2 instance types in Amazon AWS through empirical performance testing: t3.medium (general-purpose with burstable performance), m5.large (general-purpose with consistent performance), and c5.large (optimized for compute-intensive workloads). Our research seeks to answer the following core questions: How do different instance families (general-purpose vs. compute-optimized) perform in fundamental performance metrics? What are the performance-to-cost trade-offs among t3.medium, m5.large, and c5.large? And which instance type is best suited for specific workload patterns, such as CPU-intensive, memory-intensive, I/O-intensive, and network-intensive tasks?

The current testing scope of this project will focus on the three aforementioned instance types and conduct benchmarks across four key performance dimensions: CPU (computational throughput), memory (read/write bandwidth), disk I/O (IOPS and throughput under various access patterns), and network (TCP bandwidth with multiple parallel streams). Through this research, we aim to provide data-driven decision-making support for cloud instance selection, thereby striking an optimal balance between performance and cost.

2. Timeline：

2.1 Completed Tasks (Weeks 1-3)

The project has successfully completed its initial phase, encompassing infrastructure setup, benchmark execution, and preliminary data analysis. Key accomplishments include the creation of AWS EC2 instances, development of automated testing scripts, and the execution of a comprehensive benchmark suite covering CPU, memory, disk I/O, and

network performance. The collected data has been processed through a Python-based analysis pipeline to generate comparative visualizations and performance summaries.

2.2 In Progress (Week 4)

The current phase is focused on deepening the analysis. This involves conducting multiple test runs to ensure statistical significance by calculating confidence intervals and standard deviations, and performing a cost-performance analysis by integrating AWS pricing data to compute cost-efficiency metrics.

2.3 Future Tasks (Weeks 5-6)

Week 5: Advanced Analysis and Real-World Application Testing

- Real-World Application Benchmarks
- Performance Degradation Analysis, measure performance variance over time, especially for t3.medium burst credits

Week 6: Advanced Features and Final Deliverables

- Comprehensive Performance Atlas
- Cost Performance Analysis
- Prepare Final Material

3. Methodology

3.1 Testing Environment and Experimental Design

All testing was conducted within a controlled Amazon Web Services (AWS) environment in the US East (N. Virginia) - us-east-1 region, utilizing a single Availability Zone to ensure consistency and minimize network latency variations. The experiments were performed on a standardized Ubuntu Server 24.04 LTS operating system. To guarantee reliable measurements, each benchmark was configured to run for a sustained duration of 30 to 60 seconds. The experimental design emphasized isolation, with each benchmark executed independently to prevent resource contention, and full automation through Bash scripts to ensure consistent test parameters and reproducible results across all instance types. All result data was systematically collected in JSON format for subsequent programmatic analysis.

3.2 Benchmark Tools and Configurations

A suite of industry-standard benchmarking tools was employed to evaluate performance across critical dimensions. For CPU performance, Sysbench (v1.0.x) was used to conduct

prime number calculations (limit of 20,000) over 60 seconds, with thread counts set to 1 and 2 to match the available vCPUs; performance is measured in events per second. Memory performance was assessed using Sysbench's sequential memory access test with a 10 GB data size and 2 threads, with results recorded as bandwidth in MiB/sec. Disk I/O performance was evaluated using FIO, testing a combination of random and sequential read/write patterns across block sizes of 4KB, 64KB, and 1MB with a queue depth of 64 and direct I/O enabled, reporting metrics in IOPS and throughput (KB/s). Finally, network performance was measured using iperf3 to gauge TCP throughput between instances in the same AZ, utilizing 1, 4, and 8 parallel streams over 30-second tests, with the resulting bandwidth reported in Gbps.

4. Preliminary Results & Feasibility

4.1 CPU Performance

Table 4.1: CPU Performance Comparison

Instance Type	1 Thread (events/s)	2 Threads (events/s)	Speedup	Scaling Efficiency
c5.large	447.65	701.82	1.57×	78.4%
m5.large	407.71	638.67	1.57×	78.3%
t3.medium	407.15	643.79	1.58×	79.1%

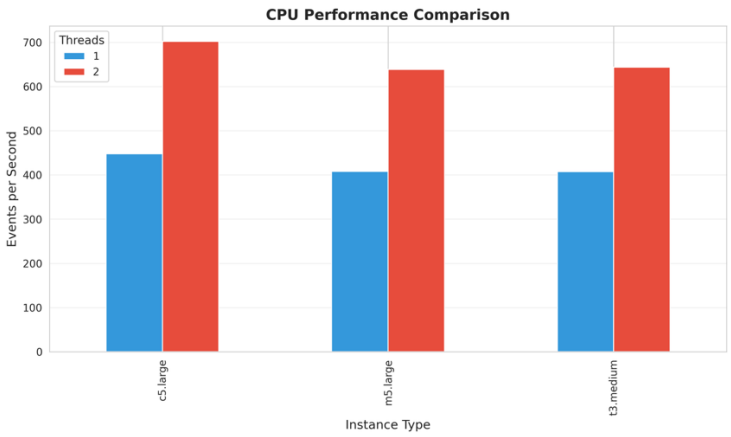

Figure 4.1: CPU Performance Comparison

The c5.large instance demonstrates superior CPU performance, achieving 10% higher throughput (447.65 events/s) in single-thread tests compared to t3.medium (407.15) and m5.large (407.71). This advantage persists in multi-threaded scenarios, where c5.large reaches 701.82 events/s versus 638-644 for the other instances. All three instance types

exhibit similar scaling efficiency (~78-79%) when utilizing both vCPUs, indicating minor overhead from thread management and shared cache contention. The c5.large's compute-optimized processor architecture provides consistent performance advantages for CPU-intensive tasks, making it the optimal choice for computational workloads such as scientific computing, video encoding, and batch processing.

4.2 Memory Performance

Table 4.2: Memory Bandwidth Performance

Instance Type	Read Bandwidth (MiB/s)	Write Bandwidth (MiB/s)	Read/Write Ratio
c5.large	11,132.70	7,578.41	1.47
m5.large	10,009.55	6,848.76	1.46
t3.medium	10,204.20	6,968.24	1.46

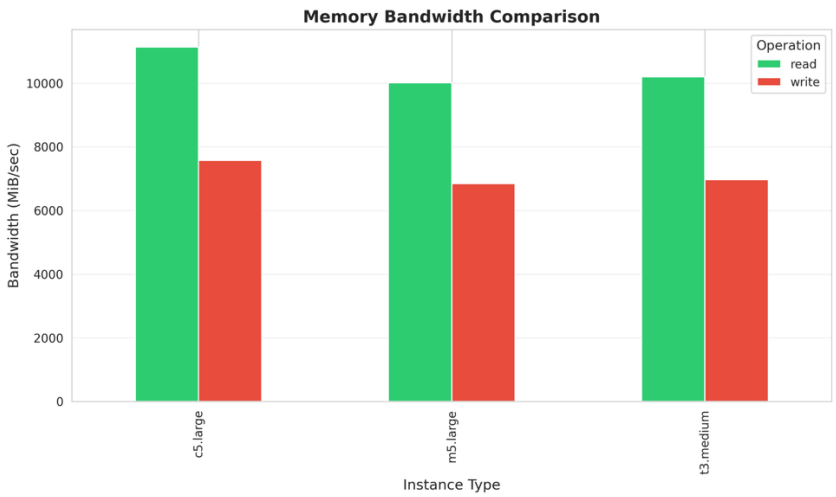

Figure 4.2: Memory bandwidth comparison

Memory bandwidth measurements reveal that c5.large achieves the highest read bandwidth at 11,132.70 MiB/s, approximately 9% superior to m5.large (10,009.55 MiB/s) and 9% higher than t3.medium (10,204.20 MiB/s). All instances demonstrate read-biased performance characteristics with consistent read-to-write ratios around 1.46-1.47, typical of modern memory subsystems optimized for read-heavy workloads. Notably, despite m5.large's doubled memory capacity (8GB versus 4GB), bandwidth differences remain modest, suggesting that for pure memory-bound operations, the choice of instance type has minimal impact. This consistency indicates that memory bandwidth is less differentiated across instance families compared to CPU performance, making it a less critical factor in instance selection unless memory capacity itself is the limiting constraint.

4.3 Disk I/O Performance

Table 4.3: Disk I/O Performance (4KB Block Size)

Instance Type	Random Read IOPS	Random Write IOPS	Sequential Read IOPS	Sequential Write IOPS
c5.large	787.70	937.10	725.73	817.23
m5.large	775.92	784.32	821.87	541.39
t3.medium	826.74	764.79	741.02	793.97

Table 4.4: Disk I/O Bandwidth (4KB Block Size, KB/s)

Instance Type	Random Read	Random Write	Sequential Read	Sequential Write
c5.large	3,150	3,748	2,902	3,268
m5.large	3,103	3,137	3,287	2,165
t3.medium	3,306	3,059	2,964	3,175

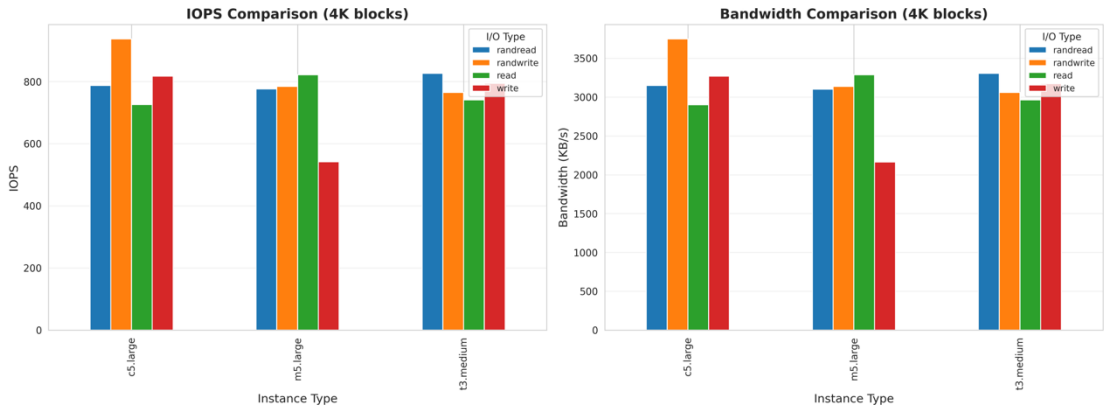

Figure 4.3: Disk I/O performance comparison for IOPS (left) and bandwidth (right)

Disk I/O performance exhibits significant variation across instance types, with results revealing workload-specific strengths. For 4KB random read operations, t3.medium achieves the highest IOPS at 826.74, representing a 5% improvement over c5.large (787.70). However, c5.large excels in random write scenarios with 937.10 IOPS, 23% superior to t3.medium (764.79) and 19% better than m5.large (784.32). An unexpected finding is m5.large's significantly lower sequential write performance (541.39 IOPS), approximately 33% below the other instances, warranting further investigation. These variations likely stem from different EBS volume configurations and network bandwidth limits affecting storage I/O. The c5.large's superior random write performance makes it particularly suitable for database workloads with high write requirements, while t3.medium's strong random read performance benefits read-intensive applications such as content delivery and caching systems.

4.4 Network Performance

Table 4.5: Network Bandwidth Performance (Gbps)

Instance Type	1 Stream	4 Streams	8 Streams	Peak Performance	Scaling Behavior
c5.large	32.84	36.51	33.96	36.51	+11.2%
m5.large	29.41	34.12	31.70	34.12	+16.0%
t3.medium	39.41	30.37	26.15	39.41	-33.7%

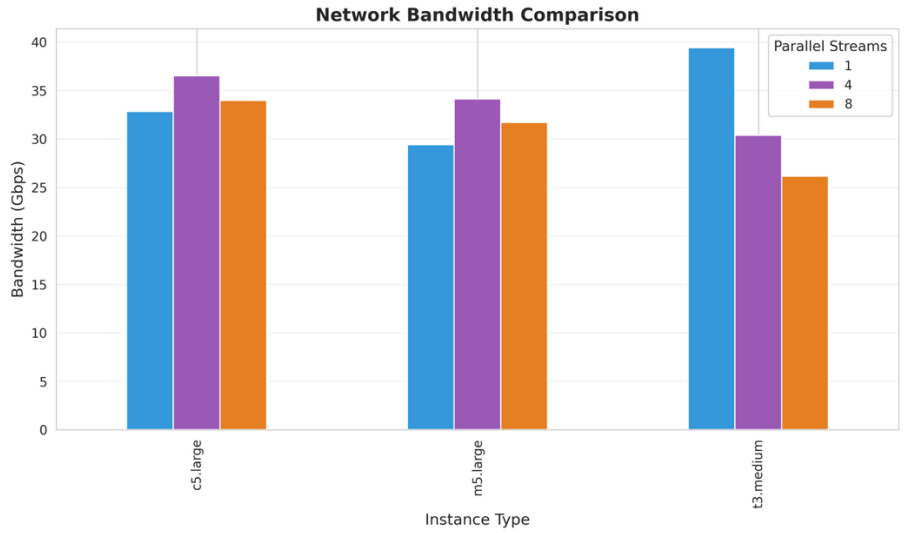

Figure 4.4: Network bandwidth comparison across different parallel stream configurations

Network performance testing reveals distinctly different characteristics across instance types, with implications for sustained high-throughput applications. The t3.medium achieves the highest single-stream bandwidth at 39.41 Gbps, suggesting aggressive network burst capabilities. However, this performance exhibits significant degradation with increased parallelism, dropping 34% to 26.15 Gbps with 8 parallel streams. In contrast, c5.large demonstrates the most stable and predictable behavior, achieving its peak performance of 36.51 Gbps with 4 parallel streams (11.2% improvement over single stream) and maintaining high throughput across all configurations. The m5.large shows moderate scaling behavior, improving 16% from 29.41 Gbps to 34.12 Gbps with 4 streams. The t3.medium's performance degradation pattern is consistent with burst-based networking, where initial performance is high but throttles under sustained load as network credits deplete. This makes c5.large and m5.large more suitable for applications requiring consistent high-throughput network performance, such as API servers, real-time data processing, and streaming services, while t3.medium may be appropriate for applications with intermittent network demands.

4.5 Overall Summary

Table 4.6: Consolidated Performance Summary

Metric	c5.large	m5.large	t3.medium	Best Performer
CPU (events/s)	701.82	638.67	643.79	**c5.large**
Memory Read (MiB/s)	11,132.7	10,009.5	10,204.2	**c5.large**
Memory Write (MiB/s)	7,578.4	6,848.8	6,968.2	**c5.large**
Disk IOPS (random read)	787.7	775.9	826.7	**t3.medium**
Network Peak (Gbps)	36.5	34.1	39.4	**t3.medium**
Memory Capacity (GB)	4	8	4	**m5.large**

The consolidated performance analysis reveals that no single instance type dominates across all metrics, emphasizing the critical importance of workload-specific instance selection. The c5.large emerges as the most balanced high-performance option, leading in CPU operations (701.82 events/s) and memory bandwidth (11,132.70 MiB/s read), making it optimal for compute-intensive applications. The t3.medium offers excellent burst performance in both network (39.41 Gbps) and disk operations (826.74 IOPS random read) at the lowest cost point, though with potential throttling under sustained load. The m5.large provides consistent mid-range performance across all categories with the advantage of doubled memory capacity, positioning it as a reliable choice for memory-intensive workloads despite not leading in any single performance metric. These findings validate the importance of matching instance characteristics to specific application requirements rather than pursuing the highest raw performance, as cost-performance trade-offs vary significantly across workload types.