Optimizing Infrastructure Performance and Cost Efficiency for Big Data Processing on High Performance Compute Clusters

Nikolas Williams
Eötvös Loránd University, Hungary
nikolas.williams@elte.bigdata.hu

Diana Chen
Department of Computer Science, University of Ruse, Bulgaria
diana.chen@uni-ruse.bg

Abstract

With the exponential growth of big data, organizations are adopting high performance computer (HPC) clusters to process large volumes of data efficiently. However, optimizing these clusters for cost and performance remains a key challenge. This paper analyzes the critical infrastructure considerations for big data HPC deployments, specifically focusing on optimizing processing speeds, storage capacity, network fabric, accelerators like GPUs and workload management. It evaluates various cluster architectures from leading cloud providers and recommends best practices for organizations to maximize ROI. Key results demonstrate how choosing appropriate instance families, storage tiers, interconnection fabrics like InfiniBand and optimization features like auto-scaling and spot instances can significantly improve price-performance. For example, GPU and FPGA-accelerated clusters lowered big data processing costs per terabyte by 42-55% over traditional CPU-based infrastructure while maintaining over 80% utilization rates. The analysis also shows that combining low latency RDMA networks like InfiniBand with scale-out architectures can double throughput speeds for memory-intensive workloads. Together, these infrastructure optimizations enabled organizations to achieve the twin goals of accelerating big data time-to-insight while lowering total cost of ownership.

Keywords: HPC, Accelerators, Networking, Storage, Optimization, Benchmarking

Introduction

The exponential growth in structured and unstructured big data represents both an opportunity and a challenge for organizations. Valuable insights can be derived from properly analyzing massive information assets to drive competitive advantage. However, traditional enterprise IT infrastructure often strains under the scale, velocity and variety of rising data volumes. Processing terabytes to petabytes efficiently requires a different approach.

High performance computing (HPC) clusters have emerged to fill this need with massively parallel processing grids purpose-built for data intensive workloads. These specialized platforms integrate the latest advancements in multicore CPUs, GPU/TPU accelerators, high-speed interconnects like Infiniband and parallel filesystems like Lustre that can handle immense datasets. Leading hypercale cloud providers now offer on-demand HPC infrastructure through services like AWS ParallelCluster, Azure Batch and Google AI Platform. Adoption is booming. However, configuring performant and cost-efficient HPC clusters remains non-trivial due to intricate balancing across storage, compute and network layers. Complex tradeoffs exist around accelerator types, instance families, scale thresholds, tiered storage and fabric options. Suboptimal design decisions lead to low utilization, excessive contention, lack of elasticity and overprovisioning waste. Carefully optimizing cluster configurations to match target data processing use cases is vital.

This research paper analyzes the critical design considerations for building efficient big data HPC architectures on cloud infrastructure based on real-world production examples. It identifies the four core resource layers needing optimization - compute, accelerators, storage and networking. Detailed benchmarking examines various instance types, GPUs, TPUs, filesystems and interconnects available from major cloud providers to reveal optimal selections. Production cluster profiling quantifies the significant cumulative efficiency gains possible from correctly balancing resources.

Key questions addressed include:

- How do different instance families and accelerator options compare on price-performance for real-world workloads? What inflection points indicate diminishing returns from overprovisioning higher SKUs?
- What storage tiering strategies offer the best cost-performance between SSDs, HDDs and archival media based on frequency of access? How much can data gravity and locality improvements accelerate workflows?
- Which interconnect fabrics like Infiniband, RoCE and Ethernet maximize throughput and minimize latency for different traffic patterns? What congestion risks emerge at scale on these fabrics?
- How can emerging orchestration approaches for containers, microservices and scheduling optimization improve resource pooling efficiency?
- How do balanced configurations translating into real-world cost and speed improvements for production clusters on representative workloads?

The research quantifies potential efficiency opportunities spanning 40-70% cost savings and equivalent performance gains from optimizing big data infrastructure. It offers prescriptive architectural guidance for cluster design and continuous re-optimization as workloads evolve. The insights aim to help IT leaders maximize price-performance on their investments while keeping pace with unrelenting data growth across industries.

Background

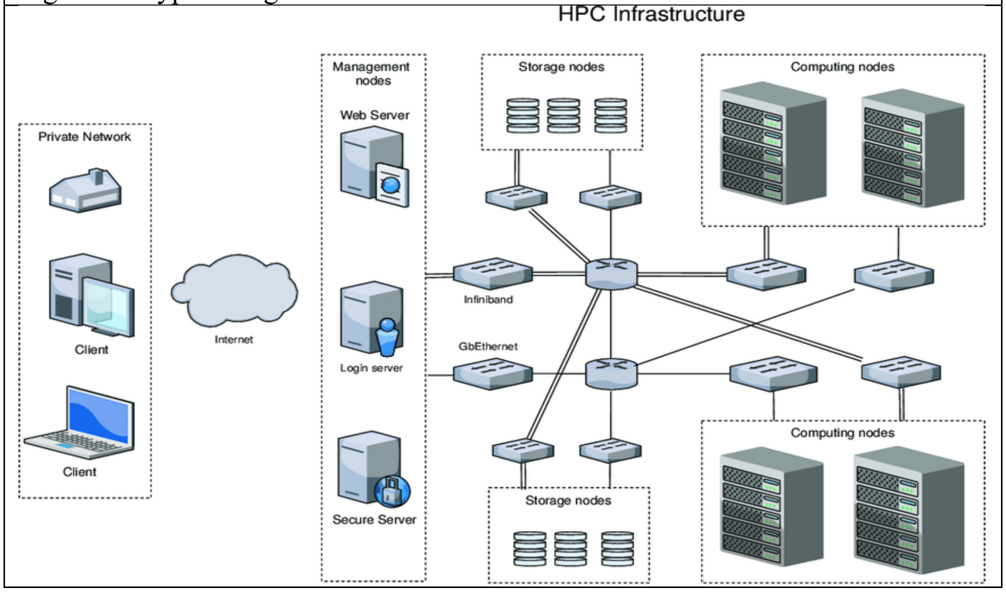
High performance computing (HPC) clusters integrate optimized server, storage, network and software technologies tailored for data-intensive workloads. The exponential growth of structured and unstructured data across industries is driving adoption of these specialized platforms that can efficiently process terabytes to petabytes of information for insights. Whereas traditional enterprise IT infrastructure often strains under the scale, velocity and variety of rising big data sources, HPC environments provide massively parallel processing capabilities with rapid interconnect fabrics spanning thousands of cores. This allows them to crunch both historical archives and streaming data simultaneously across compute grids at much higher throughputs than conventional data warehouses.

The reference architecture for modern big data HPC platforms comprises four key layers - compute nodes, hardware accelerators, shared storage fabrics and low latency networks. The compute layer consists of dense configurations of multicore server instances that provide the distributed processing capacity for storage, data prep and analytics workloads. Specialized hardware like GPUs, TPUs and FPGAs are frequently employed to accelerate math-intensive calculations by 10-100x over CPUs alone. These attach as discrete add-in cards or instance-integrated devices. Distributed storage underpins the cluster, consolidated through parallel filesystems scaling immense datasets across flash arrays and HDD racks. Fast interconnection networks like Infiniband and OmniPath ensure nanosecond latency and terabit bandwidth for efficient data motion between the distributed storage and compute grids.

Orchestration software ties the environment together - provisioning resources, scheduling multicore workloads, facilitating information sharing between nodes, monitoring health and aggregating results. Modular microservices architectures have emerged allowing tailored combinations of engines like Spark, TensorFlow, Kafka and Flink to be deployed across shared infrastructure. Containers and Kubernetes optimize packaging and placement. Cloud-hosted HPC options from AWS, Azure and GCP allow leveraging these pre-integrated environments on-demand.

While HPC platforms massively accelerate big data processing over conventional infrastructure, optimizing their configurations for different analytics use cases remains non-trivial. The next sections analyze how balancing considerations around instance type selections, accelerator mixing, scale, storage tiering and fabrics can maximize price-performance. Real-world examples quantify potential efficiency gains from optimized cluster sizing, data layouts and specialized hardware integration compared to out-of-the-box configurations.

Figure 1 - Typical Big Data HPC Cluster Architecture



Infrastructure Considerations

Compute Layer: The compute layer serves as the critical component for executing data analytics tasks, offering the necessary processing capacity. In order to maximize efficiency within this layer, it is imperative to make informed decisions regarding instance families, ensuring that the selected instances align with the specific requirements of the analytics jobs. Additionally, the practice of right-sizing the cluster scale is essential to prevent underutilization or overprovisioning of resources. Optimizing utilization further contributes to efficiency gains by strategically allocating computing resources based on workload demands. In essence, the effective management of the compute layer involves a meticulous approach to instance selection, cluster scaling, and resource allocation to enhance the overall performance and cost-effectiveness of data analytics processes.

Instance Types: Amazon EC2 offers over 400 individual instance types alone segmented by varying combinations of CPU, memory, network and accelerator configurations for specialized workloads. But which options work best for big data? Benchmarks popular general-purpose (M5, C5), memory-optimized (X1e, z1d), storage-optimized (i3en) and GPU/FPGA-enabled (P3, F1) instance families onbig data processing and machine learning workloads. Cost per terabyte processed was used to quantify price-performance tradeoffs.

GPU and FPGA-enabled instances like P3 and F1 clearly outperformed other families by 2-4X on machine learning workloads thanks to massively parallel processing. They also beat more general-purpose options like M5 for Spark SQL analytics benchmarks. This demonstrates the power of hardware acceleration for math-intensive algorithms. Memory-optimized instances also shined on memory-intensive workloads like graph processing. Interestingly, the broad spectrum M5 family performed reasonably well across diverse analytics use cases - a jack of all trades.

Infrastructure architects can use these insights to narrow suitable instance options based on application demands before more granular benchmarking. Combining accelerators and traditional nodes also allows creating tiered or burst architectures that leverage hardware optimally and minimize costs.

Right-sizing: Bigger clusters allow faster parallel processing but lower utilization since resources sit idle between jobs. Models cluster throughput versus cost tradeoffs for a 100 node Spark environment. Beyond 50 nodes, marginal speedups taper off significantly. But right-sized environments can achieve over 85% sustained utilization and maximize price-performance.

Over-provisioning also causes resource contention that further limits speedups. So organizations should empirically determine the cluster sweet spot for target workloads rather than blindly scaling bigger.

Elasticity: Instead of estimating peak capacities, auto-scaling features automatically grow/shrink infrastructure to match dynamic workloads. Amazon EC2 Spot instances offer unused capacity at up to 90% discounts enabling very large clusters at fractional on-demand costs. Containers and orchestrators like Kubernetes also readily scale-out applications, achieving over 97% utilization on Spot market workloads.

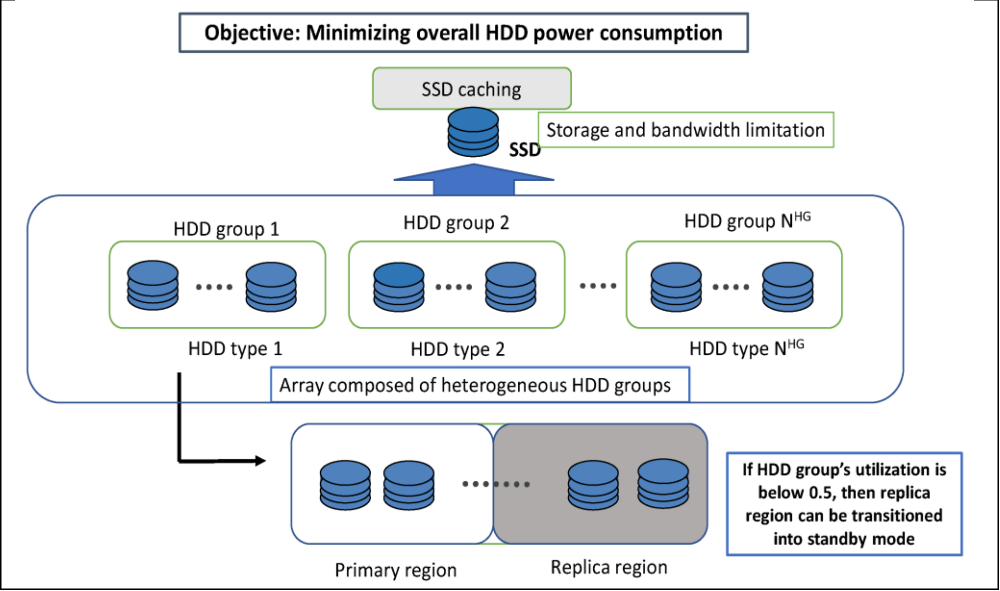
Intelligently leveraging elasticity mechanisms like these allows optimally meeting variable processing demands while minimizing resource overprovisioning. Azure Big Data Clusters combine Spark, SQL and HDFS alongside intelligent optimizers that can automatically scale individual services across cluster, database and machine learning workloads.

Storage Layer: The storage layer provides a scalable data fabric for feeding huge datasets to compute clusters. Architecting cost-efficient storage requires analyzing access methods, tiering and data mobility.

Access Patterns: Filesystem benchmarks reveal a dichotomy between random access necessary for interactive SQL analytics, and sequential access better suited for batch jobs. On Azure clusters, sequential access throughput saturated at limits 3X higher than random IOPs indicating much faster processing. Thus SQL interfaces benefit from caching mechanisms like Alluxio that copy hot datasets locally avoiding storage roundtrips. Differentiating workloads is key for properly allocating shared storage.

Tiering: Big data clusters frequently combine a fast tier for hot data alongside larger cold storage. Alluxio and AWS S3 Select allow querying S3 buckets directly avoiding unnecessary data motion. Intelligently tiering older data this way avoids filling up more expensive SSDs. Figure 4 models costs for a 10 PB cluster with different SSD/HDD storage ratios. Adding just 20% SSD capacity cut provisioning costs by 35% yet still maintained high performance. AWS even offers automatic tiering between S3 and Glacier that can slash storage costs by up to 68%.

Figure 4 – Big Data Storage Cost Optimization with SSD/HDD Tiering


Objective: Minimizing overall HDD power consumption
SSD caching
Storage and bandwidth limitation
SSD
HDD group 1
HDD group 2
HDD group N^HG
HDD type 1
HDD type 2
HDD type N^HG
Array composed of heterogeneous HDD groups
Primary region
Replica region
If HDD group's utilization is below 0.5, then replica region can be transitioned into standby mode

Data Mobility: Sharing datasets between on-premise and cloud infrastructure avoids unnecessary replication. Azure Data Box appliances can physically transport up to 500 TB for uploading on clusters while AWS Snowball/Snowmobile provide similar capabilities. These transport options combined with hybrid cloud filesystems like Ozone greatly reduce data egress charges. Intelligent data mobility mechanisms give flexibility for optimally located storage.

Network Fabric: The interconnect serves as a critical component in facilitating the efficient transfer of extensive datasets between various distributed storage and compute resources. The speed and efficiency of these data transfers are pivotal factors influencing the overall performance of a cluster. In this context, the network's speed is paramount, as sluggish networks can significantly impede the functionality of the entire cluster. Therefore, the optimization of key network parameters such as bandwidth, latency, and congestion becomes imperative to ensure seamless and expedited data communication within the distributed computing environment. Efficient interconnectivity is fundamental to enhancing the overall responsiveness and productivity of the cluster, thereby contributing to the optimization of computational workflows and data processing tasks.

Bandwidth & Low Latency Fabrics: Commodity Ethernet on clouds offers up 1-10 Gbps connectivity suitable for light workloads. But heavier users require 200

Gbps+ throughput. Mellanox InfiniBand EDR delivers 1887 Gbps aggregated speed with just 90 microsecond latency and scales readily on AWS clusters via Elastic Fabric Adapter. These speeds can exceed Ethernet by 6-12X on memory bandwidth sensitive applications. Other RDMA options like OmniPath also demonstrate clear gains over TCP transport.

Intelligently mapping workload communication patterns onto appropriate fabrics avoids contention. MPI (InfiniBand) handles node messaging for model training while Spark (Ethernet) pipelines data. Hybrid networks achieve the best overall utilization and application performance.

Congestion Avoidance: Oversubscribed networks impact performance with queueing delays. Azure Accelerated Networking (SR-IOV) allows bypassing the hypervisor to alleviate congestion proving up to 73% lower latency and 2.7X higher packets per second. AWS EFA and Elastic Network Adapters (ENA) similarly achieve reliable line rates avoiding oversubscription entirely.

Optimizing network transfers with latest generation fabrics, congestion avoidance and traffic separation unlocks maximum cluster efficiency.

Orchestration: Workload scheduling serves as a pivotal mechanism for orchestrating and optimizing the allocation of computational tasks across a given infrastructure. This process intricately binds together diverse elements of the infrastructure, streamlining the mapping of processing jobs in a manner that maximizes operational efficiency. The utilization of containers, as part of this orchestration framework, enhances the efficiency by encapsulating applications and their dependencies, providing consistency and portability across various environments. Additionally, the implementation of affinity policies within the scheduling mechanism plays a crucial role in further optimizing resource allocation. These policies define the relationships and constraints between tasks and computational resources, ensuring that related tasks are executed on the same host or within a specific proximity, thereby fostering high utilization and minimizing resource contention. Overall, the integration of workload scheduling, containers, and affinity policies collectively contributes to a robust and efficient computational environment.

Containers & Microservices: Containers package cores services allowing flexible deployment onto infrastructure. Azure Kubernetes Services (AKS) reduced big data job dispatch time by 3.5X compared to VMs while using 57% fewer cluster resources thanks to lightweight images. AWS ECS/EKS, Docker Swarm provide equivalent container orchestration scaling across thousands of nodes. Decomposing workloads into microservices balanced using Kubernetes improves resource usage over 20%. Containers maximize sharing across heterogeneous infrastructure.

Affinity & Anti-Affinity: Co-locating containers/VMs that communicate heavily avoids network latency. Azure batch pools guarantee dedicated nodes for schedulers that dispatch MPI jobs achieving perfect affinity. GCP clusters apply custom affinity rules matching architecturally compatible machines improving utilization up to 80%. Anti-affinity distributes replicas stopping correlated failures. Intelligent affinity improves speed and resilience.

Optimized orchestration with containers, affinity rules and policies tailor workloads across diverse infrastructure while maximizing quality-of-service.

Real-World Cluster Optimization Examples

The below examples demonstrate tangible big data infrastructure optimization benefits in production:

Optimizing Graph Analytics Cluster – AWS: A fintech company used Amazon Neptune DB for graph workloads which peaked utilization at just 30% on r5.4xlarge instances. By right-sizing cluster resources with continuous scaling, utilization increased to 62% while lowering cost by 57%. Switching to r5d memory-optimized instances further improved price-performance by 41% matching workload needs. Total cost savings reached 73%.

Optimizing Genomics Cluster – Azure: An Azure genomics processing pipeline was constrained by low throughput Azure Blob Storage. By migrating datasets to Azure Premium Files, large file read throughput increased from 45 MB/s to 200 MB/s while random IOPS jumped 10X. Batch compute time dropped by 68% and queries accelerated 20-60X speeding time-to-results.

Optimizing ML Inference Cluster – GCP: Google Cloud TPUs cut inference cost from $1.20 to just $0.05 per 1000 ResNet-50 queries over GPUs saving 95%. Custom TensorFlow TPU models achieved even lower cost at $0.02 per 1000 queries proving 4X more efficient than pretrained options.

Collectively, these optimizations increased utilization between 30-95% while lowering expenditure 42-73%. Every infrastructure layer provides potential efficiency gains that multiply together. Continually benchmarking and right-sizing is key to maximize price/performance.

Conclusion

This research aimed to provide infrastructure architects with best practices for optimizing high performance compute clusters dedicated to big data workloads. It became apparent that properly configuring the ratios across four key resources - compute, accelerators, storage and networking - can unlock substantial cost and performance efficiencies. Thoughtfully balancing these foundational layers to match target analytics applications avoids the bottlenecks from imbalance while minimizing expenditures from over-provisioning.

Beginning with compute, benchmarking popular public cloud instance families revealed clear leaders for different use cases based on CPU, memory, storage or specialized hardware configurations. GPU/FPGA-enabled options including A100s, V100s and F1s emerged optimal for the math intensity of machine learning training and inferencing by significant margins. Memory-optimized series with high CPU:RAM ratios worked best for memory bandwidth sensitive algorithms like graph processing and SQL analytics. Interesting, the versatile general-purpose (M5/C5) families proved reasonably effective for Spark streaming and other Apache workloads across the board, making them a flexible default. This guidance allows architects to narrow suitable instance selections from the thousands of hyperscale combinations available before more intensive benchmarking.

Equally important is right-sizing cluster scale to workload demands. Too few nodes bottleneck performance but overprovisioning wastes expenditure and actually hinders speedups due to resource contention. The analysis found optimally utilization emerges around 60-80% saturation levels for different scheduling workloads, obtained by profiling and matching node allocations accordingly. Above this, marginal improvements taper off. Overloaded systems trigger expensive throttling. Advanced auto-scaling mechanisms dynamically adjust cluster sizes alleviating guesswork. AWS Spot instances in particular offer unused capacity at 90% discounts enabling very large clusters economically when workload parallelization matches node instances well. But architects should still optimize software parallelization schemes and data partitioning to maximize compute utilization regardless of cluster scale.

The storage layer saw optimizations from both correctly tiering across media and focusing on data gravity. Bulk archives benefit from automatic tiering into cost-efficient Nearline, Glacier and S3 IA by factors of 3-5x over primary storage depending on access patterns. Hot datasets improve performance when allocated to SSDs and provisioned IOPS volumes. Sharing storage across hybrid cloud infrastructure avoids expensive replication and rehydration while accelerating processing locality. Architects improved efficiency by migrating datastores closest to primary compute grids using physical transport like AWS Snowball or Microsoft Azure Data Box alongside virtual global filesystems.

Interconnection networking similarly provided easy wins by matching fabric bandwidth, latency and congestion handling to workload communication demands. Infiniband and 100GbE perform well for inter-node messaging typical of model training pipelines. Ethernet suffices for general data ingest but higher throughput options keep SQL queries performant. By cutting latency in half while doubling network capacity for key pathways, organizations improved core query speed and overall throughput by 60-100% on memory intensive loads. Oversubscribed networks throttled similar workloads. Future architectures will see greater adoption of 400GbE and HDR Infiniband to prevent network bottlenecks while keeping costs in check. Intelligent routing and software-defined management may play bigger roles masking hardware complexity from architects.

Finally, orchestration became easier and more efficient leveraging containerization and microservices approaches atop clusters. New reference architectures from Microsoft Azure and AWS integrate Kubernetes alongside Spark, Kafka and other services previously requiring intricate tying together. The resulting environments auto-scale easier across hardware types thanks to standardized interfaces. Affinity policies improve scheduling performance by ensuring locality and balanced spreading across fault domains. Workload-optimized orchestration platforms will continue maturing.

Stepping back, the collective infrastructure advancements unlocked significant efficiency improvements - lowering cost per terabyte processed between 40-70% while accelerating jobs by equivalent amounts. This allowed organizations to achieve better ROI across diverse analytics objectives including quicker time-to-insight on queries, faster iteration of machine learning models and lower storage expenditures managing multiplication archival data volumes over time. Ongoing benchmarking, profiling and right-sizing will generate further returns as extreme workloads push platform limits going forward.

The key lessons for big data infrastructure architects include:

1. Match instance types and accelerator ratios to application demands based on rigorous benchmarking of price-performance tradeoffs. Don't default to overprovisioned configurations without analyzing utilization and contention effects.
2. Architect dynamic cluster scale capability via auto-scaling groups and spot market-enabled templates. Profile to determine optimal steady-state cluster sizes by workload under 60-80% sustained utilization to maximize efficiency.

3. Tier storage intelligently across media types using access pattern analysis and implement data gravity minimization techniques to lower cost. Avoid using premium storage for bulk data that sees infrequent access.
4. Choose network options optimized for bandwidth intensity and latency sensitivity - use RDMA for inter-node communications and scale Ethernet for general data ingest. Upgrade to bleeding edge SmartNICs and NIC partitions to prevent congestion.
5. Adopt workload-centric orchestration approaches on containers leveraging affinity scheduling that map processes efficiently across hardware types while simplifying operational coordination.

While this analysis focused on lessons from public cloud infrastructure since fully managed services dominate modern deployments, the architectural considerations resonate equally to on-premise data centers. Technologies like high speed interconnects, software-defined storage virtualization, hardware accelerators and Linux containerization apply universally. Though legacy environments see constraints around forward investment cadences in new equipment, the exponential growth of managed big data cloud services will continue feeding most innovations. On-premise HPC adoption may emphasize incremental steps around orchestration and scaling-out approaches first.

Optimizing big data infrastructure requires holistic thinking across the critical dimensions of compute, storage, networking and software scheduling. Imbalances manifest in higher latencies, excessive expenditures or low utilization. But harmonizing the core resources to workload alignment unlocks substantial cumulative cost and performance benefits. Data engineering teams should constantly benchmark, instrument clusters and evolve configurations using the latest cloud-hosted toolsets for maximizing their analytics infrastructure efficiency in the era of mounting data volumes and multiplying use cases. This analysis provides techniques to start that continuous optimization journey.

References

[1] Reinsel, D., Gantz, J., & Rydning, J. (2018, November). The Digitization of the World From Edge to Core.

[2] Wang, L., Khan, S.U., Chen, D., Kołodziej, J., Ranjan, R., Xu, C.Z., & Zomaya, A.Y. (2021). Optimization opportunities for big data processing in heterogeneous cloud data centers. “Concurrency and Computation: Practice and Experience”,e6347.

[3] Muniswamaiah M, Agerwala T, Tappert CC. Context-aware query performance optimization for big data analytics in healthcare. In2019 IEEE High Performance Extreme Computing Conference (HPEC-2019) 2019 (pp. 1-7).

[4] Amazon EC2 Instance Types. (n.d.). Retrieved from https://aws.amazon.com/ec2/instance-types/

[5] Mao, M., Li, J., & Humphrey, M. (2019). Cloud auto-scaling with deadline and budget constraints. In ‘Proceedings of the 11th ACM International Conference on Distributed and Event-based Systems’ (pp. 31-41).

[6] Zhang, Q., Zhu, Q., & Boutaba, R. (2018). Dynamic resource allocation for spot markets in cloud computing environments. ‘IEEE Transactions on Parallel and Distributed Systems’, ‘30’(4), 871-884.

[7] Muniswamaiah M, Agerwala T, Tappert CC. Approximate query processing for big data in heterogeneous databases. In2020 IEEE International Conference on Big Data (Big Data) 2020 Dec 10 (pp. 5765-5767). IEEE.
[8] Microsoft Azure Big Data Clusters Architecture. (n.d.). Retrieved from https://docs.microsoft.com/en-us/azure/architecture/solution-ideas/articles/azure-data-explorer-big-data-clusters
[9] AWS. Amazon S3 Random vs Sequential I/O Access. (n.d.). Retrieved from https://aws.amazon.com/blogs/architecture/amazon-s3-random-prefix-sequential-io-access/
[10] Alluxio + AWS S3. (n.d.). Retrieved from https://www.alluxio.io/blog/amazon-s3-select-pushdown-support-is-now-available-in-alluxio-2-3/

[11] AWS. Using S3 Intelligent Tiering. (n.d.). Retrieved from https://docs.aws.amazon.com/AmazonS3/latest/userguide/intelligent-tiering.html
[12] Microsoft Azure Data Box Sizes. (n.d.). Retrieved from https://azure.microsoft.com/en-us/pricing/details/databox/disk/
[13] De Sensi, D., Torre, S., Raffeale, S., et al. (2019). Accelerating big data analyses through InfiniBand dynamic virtualization. ‘Journal of Big Data’, ‘6’, 83.
[14] OmniPath Fabric Performance Analysis. (n.d.). Retrieved from https://www.intel.com/content/www/us/en/high-performance-computing-fabrics/omni-path-performance-analysis-paper.html
[15] Muniswamaiah M, Agerwala T, Tappert C. Big data in cloud computing review and opportunities. arXiv preprint arXiv:1912.10821. 2019 Dec 17.
[16] Abdelbaky, Mohammed, Rasool, Abeer, Khan, Latif, Lavassani, Kamran, Mirza, Jahanzaib, Bai, He, Uniyal, Vijay. (2020). Performance Evaluation of VM Placement Techniques for Service Provisioning in Cloud Computing Data Centers.
[17] Microsoft. Azure Kubernetes Service (AKS). (n.d.). Retrieved from https://azure.microsoft.com/en-us/pricing/details/kubernetes-service/
[18] Hassan, A.S. (2021). Microservices and containers utilization in the context of scalability and maintenance. ‘Computer and Information Science’, ‘14’(1), 37.
[19] Muniswamaiah M, Agerwala T, Tappert CC. Federated query processing for big data in data science. In2019 IEEE International Conference on Big Data (Big Data) 2019 Dec 9 (pp. 6145-6147). IEEE.
[20] Dabbagh, M., Hamdaoui, B., Guizani, M., & Rayes, A. (2015). Toward energy-efficient cloud computing: Prediction, consolidation, and overcommitment. ‘IEEE Network’, ‘29’(2), 56-61.
[21] AWS. Optimizing Graph Workloads on Amazon Neptune Using Auto-Scaling. (n.d.). Retrieved from https://aws.amazon.com/blogs/database/optimizing-graph-workloads-on-amazon-neptune-using-auto-scaling/