Serverless Computing for Big Data Analytics: Performance and Cost Analysis of AWS Lambda and Google Cloud Functions

Mohamed Amine Ben Ali[1]

[1]Université de Kairouan, Département d'Informatique, 25 Avenue Ibn Khaldoun, Kairouan, Tunisia

2023

Abstract

Serverless computing has emerged as a crucial paradigm for big data analytics, offering automated resource provisioning and event-driven execution environments that aim to reduce overhead and maintenance complexities. In this work, a detailed performance and cost analysis of two prominent serverless platforms, AWS Lambda and Google Cloud Functions, is performed in the context of big data workloads. By investigating the intricacies of invocation patterns, parallel execution, cold start latencies, and resource allocation models, this research identifies essential factors that influence efficiency and scalability. The discussion presents a comprehensive framework for analyzing task throughput, response times, memory usage, and concurrency limits under varying workloads. The interplay between performance metrics and cost structures is emphasized, highlighting how function duration, memory configurations, and request frequencies contribute to diverse pricing outcomes. To further refine these outcomes, mathematical models capturing task arrival rates, provisioning delays, and cost functions are introduced to elucidate optimal resource allocation decisions. The results presented in this work underscore the potential of serverless platforms to handle large-scale datasets effectively, while also illustrating critical trade-offs in cost, performance consistency, and architectural design. This analysis serves as a reference for practitioners seeking to implement big data processing pipelines, as well as for researchers aiming to extend the theoretical underpinnings of serverless computing in cloud-based analytics.

1 Introduction

Serverless computing has significantly altered the cloud computing landscape by decoupling capacity planning and server management from application development [1]. The notion of deploying functions directly to a provider-managed environment has transformed the focus of computation, moving it away from complex infrastructure considerations and toward rapid, event-triggered processing. This shift is particularly relevant for big data analytics, where ephemeral compute resources can be dynamically scaled to meet fluctuating demands without incurring substantial idle costs [2]. As organizations handle increasingly large volumes of data from diverse sources, the operational benefits of serverless computing become more pronounced, especially in the context of cost efficiency and operational simplicity. However, the performance characteristics of serverless platforms are not entirely uniform; differences in resource allocation, concurrency limits, cold start times, and underlying virtualization mechanisms introduce distinct performance profiles [3]. This research elucidates how AWS Lambda and Google Cloud Functions behave under the stress of large workloads, paying particular attention to concurrency scaling dynamics, function invocation throughput, and latency characteristics.

The expansion of serverless computing into analytics workflows is underpinned by its capacity to scale near-instantly in response to rapid surges in demand, a feature that aligns well with the bursty nature of data-intensive tasks [4]. Despite these advantages, the interplay of factors such as platform overhead, operational latencies, and cost structures must be analyzed comprehensively. Analytical models can capture essential performance and cost parameters, facilitating an understanding of how function-centric architectures compare to more conventional and container-based approaches in big data scenarios. The analysis provided here explores the threshold at which serverless solutions meet their limitations, whether through concurrency bottlenecks, memory constraints, or increased latency [5]. Such insights lead to mathematical characterizations of system throughput, queueing dynamics,

and cost functions that guide architectural decisions. Consequently, the research emphasizes a multifaceted view of serverless platforms, outlining both their strengths and limitations and providing a robust basis for designing large-scale data processing pipelines. [6]

2 Serverless Foundations and Architecture

The architecture of serverless platforms such as AWS Lambda and Google Cloud Functions rests upon a foundational principle of abstracting server management. Developers supply functions, often packaged with necessary dependencies, and the platform handles all aspects of provisioning and scaling [7]. Central to this model are ephemeral containers or micro-virtual machines that instantiate rapidly to handle incoming requests. Upon receiving an event trigger, the platform checks for available, warm function instances to minimize overhead; if none exist, a new instance is created. While this process can introduce cold start latencies, it also ensures that applications are not bound to idle resources when demand is low. [8]

From a structural perspective, the serverless execution model can be conceptualized using a multi-layered approach. At the outer layer, event sources, such as HTTP requests or streaming data services, trigger function invocations [9]. The core layer contains the functions themselves, which are self-contained pieces of code designed to handle a specific task. Beneath these layers, container orchestration and resource management subsystems monitor concurrency limits, memory usage, and the scheduling of new containers [10], [11]. A unifying factor of this layered approach is the reliance on short-lived containers or micro-VMs, which permit granular scaling at the function level. This granularity contrasts with traditional virtualization or container orchestration frameworks that scale entire applications or services.

Mathematically, let λ denote the average arrival rate of incoming events, measured in invocations per second [12]. Let X be the random variable representing the service time of a single function invocation. The serverless platform can be modeled akin to an M/M/k queue, though the value of k (the number of servers) is dynamically adjusted by the platform to match demand [13]. In an idealized scenario, one could frame the total system throughput as a function of λ, $E[X]$, and the concurrency capacity that the provider can scale to. Specifically, one might consider the probability of instance unavailability P_{queue} to be a function of concurrency constraints:

$$P_{\text{queue}} = \left(\frac{\lambda E[X]}{k}\right)^{n} \Phi(\lambda, k, E[X]),$$

where Φ is a function encoding how concurrency scaling aligns with the instantaneous arrival rate. Although such models can be idealized, they present a starting point for analyzing how ephemeral containers handle bursty workloads [14]. By capturing the interplay between arrival rates, function execution times, and concurrency scaling, these models help to characterize the system's capacity to maintain low latency under massive parallel requests.

The ephemeral nature of serverless resources also imposes constraints on certain classes of applications. Large-scale data analytics can demand significant CPU or memory resources, and short timeouts may complicate computations that require extended processing times [15]. Yet for workloads that can be partitioned into smaller tasks, the serverless approach can harness massive parallelism. This parallelism stems from the near-instantaneous replication of the function container to match the event arrival rate [16]. Understanding how ephemeral containers spawn, execute, and terminate is key to utilizing serverless computing for large-scale data analytics.

3 Performance Metrics and Scalability

Performance evaluation in serverless computing centers on multiple interrelated metrics [17]. These include invocation latency, cold start overhead, throughput, concurrency limits, and memory utilization. Invocation latency, in particular, may significantly vary due to cold starts, where a new container or micro-VM is launched and initialized [18]. In contrast, warm starts reuse existing containers, keeping latency relatively low. The disparity between cold start and warm start times becomes crucial when analyzing tight response time requirements.

To characterize performance, one can track the distribution of service times for both cold and warm invocations [19]. Let T_c denote the random variable representing the cold start overhead, and let T_w denote the random variable for the warm invocation overhead. The total response time for an invocation can then be modeled as: [20]

$$R = \begin{cases} T_c + P, & \text{with probability } p_c, \\ T_w + P, & \text{with probability } 1 - p_c, \end{cases}$$

where P is the execution time for the actual function payload, and p_c is the probability of a cold start. This probability depends on the arrival pattern of requests and the idle timeout of function instances [21]. By collecting samples of T_c, T_w, and P, one can build empirical or theoretical distributions to quantify the expected value

and variance of R. Such characterizations are valuable in comparing serverless platforms, as they can reveal how infrastructure choices and runtime optimizations influence latency profiles.

Beyond latency, concurrency scaling represents a vital dimension of performance [22]. Ideally, as the arrival rate increases, the platform replicates function instances proportionally. Nevertheless, practical limits exist in both AWS Lambda and Google Cloud Functions [23]. If $\Lambda_{\max}$ denotes the maximum concurrency for a given account or region, the probability of saturating concurrency resources rises as λ grows. Once concurrency is saturated, new invocations must wait, resulting in queueing delays or failures. In mathematical terms, if we define $k(\lambda)$ to be the dynamically allocated concurrency at arrival rate λ, then the maximum concurrency constraint imposes $k(\lambda) \leq \Lambda_{\max}$. When λ is sufficiently large, further arrival increases produce queuing behavior and degrade system responsiveness [24]. Understanding these saturation points helps clarify the maximum feasible throughput and informs decisions about partitioning data processing tasks or upgrading concurrency limits.

Additionally, memory settings impose another layer of complexity. In certain configurations, memory allocation is directly correlated with CPU resources [25]. For instance, a function configured with higher memory may access a more powerful CPU. This interplay can significantly affect execution times for compute-bound tasks [26]. One can model the memory allocation dimension by letting M be the assigned memory in megabytes, which influences the function's execution speed. A simplified approach introduces a function $f(M)$ that captures how the mean processing time $E[P]$ decreases with increasing memory and CPU resources. Then, the overall response time can be represented by an expression of the form: [27]

$$R(M) = \begin{cases} T_c(M) + \frac{c_1}{f(M)}, & \text{with probability } p_c, \\ T_w(M) + \frac{c_1}{f(M)}, & \text{with probability } 1 - p_c, \end{cases}$$

where c_1 is a constant reflecting the computational complexity of the function payload. This formulation underscores how memory and CPU allocation can influence not only throughput but also cold start times, as larger memory footprints may increase container initialization overhead [28]. Quantifying these effects through empirical measurements and theoretical models provides a systematic approach to assessing serverless performance across different workloads.

4 Cost Modeling and Resource Allocation

Cost analysis in a serverless environment is strongly coupled with the performance characteristics of the application [29]. Because providers usually charge per execution time and memory configuration, the choice of memory size and concurrency level can significantly impact billing. Let $C(\text{mem})$ denote the per-millisecond charge for a function assigned a memory size of mem megabytes. When a function invocation runs for t milliseconds, the cost of a single invocation can be expressed as: [30]

$$\text{Cost}_{\text{invocation}}(\text{mem}, t) = C(\text{mem}) \times t.$$

Further, each invocation carries an overhead in requests, typically with a cost per million invocations. If α is the cost per request, and $\beta(\text{mem})$ is the cost per millisecond for a given memory tier, then for N total invocations each taking t milliseconds, the total cost can be approximated as:

$$\text{Cost}_{\text{total}} = \alpha \times N + \beta(\text{mem}) \times N \times t.$$

Choosing the optimal memory allocation to minimize total cost while maintaining acceptable performance becomes a multi-objective optimization challenge. One might seek to minimize cost subject to latency constraints or, conversely, minimize latency under a given budget [31]. A multi-objective optimization approach can be adopted, where a function $L(M)$ describes latency as a function of memory, and a function $\Gamma(M)$ describes the total cost. One then looks for M^* such that: [32]

$$\min \Gamma(M) \quad \text{subject to} \quad L(M) \leq L_{\text{threshold}},$$

where $L_{\text{threshold}}$ is a specified performance requirement. Alternatively, one might define a Lagrangian of the form:

$$\mathcal{L}(M, \lambda) = \Gamma(M) + \lambda(L(M) - L_{\text{threshold}}),$$

with λ being a Lagrange multiplier [33]. Solving for $\frac{\partial \mathcal{L}}{\partial M} = 0$ could identify points where the trade-off between cost and latency is balanced. However, such analytical expressions can become complex due to the variability of cold starts, concurrency constraints, and ephemeral container initialization times.

Dynamic scenarios introduce further intricacies [34]. In real-time big data analytics, workloads may fluctuate, causing the arrival rate λ to shift rapidly. An adaptive strategy might continuously monitor performance metrics

and adjust memory allocations on the fly, aiming to balance cost and throughput in near-real-time. This leads to control-theoretic models, where the control variable M is updated based on feedback about latency, concurrency usage, and ongoing costs [35]. One might formulate a discrete-time controller:

$$M_{k+1} = M_k + \zeta\big(L_{\text{measured}}(k) - L_{\text{target}}\big),$$

where ζ is a gain factor [36]. The cost function would then be evaluated iteratively to ensure that as memory is increased or decreased, the overall system behavior remains within desired performance boundaries without causing prohibitive billing. The viability of such adaptive policies is contingent on the speed with which serverless platforms apply memory configuration changes and the accuracy of short-term predictions of incoming workload intensity. [37], [38]

In practice, evaluating cost efficiency also demands attention to provider-specific pricing structures. AWS Lambda, for example, uses a specific incremental scale for memory settings, whereas Google Cloud Functions applies a comparable but not identical pricing model. Calculating and comparing total expenditure under real-world conditions often reveals subtle distinctions between platforms [39]. Any cost model must thus be nuanced enough to account for the interplay of invocation frequencies, average execution durations, memory tiers, and concurrency limits.

5 Implementation and Experimental Analysis

Implementing serverless analytics pipelines involves deploying functions in a provider's platform, configuring triggers for data intake, and integrating various services for orchestration [40]. The experimental methodology typically encompasses the development of test workloads that range from lightweight, CPU-bound functions to memory-intensive tasks such as parsing large files or performing transformations on streaming data [41]. Once the functions are deployed, an external driver can generate invocation requests at controlled rates to evaluate throughput, concurrency scaling, and response times. [42]

A critical phase in experimentation is assessing cold start frequency under different traffic patterns. For instance, sporadic invocation patterns may yield a higher proportion of cold starts. Conversely, sustained invocations at a moderate rate often keep containers warm, minimizing cold start overhead [43]. Gathering detailed logs allows for measuring distributional properties of latency, including tail latencies that might become significant during load spikes. The time-series data of invocation latencies offers insight into how the serverless system adapts to sudden bursts. [44]

Performance analysis also involves correlating concurrency usage with overall throughput. Observing metrics such as the number of active function instances at a given moment uncovers the elasticity and responsiveness of the platform [45]. When concurrency scaling lags behind large surges in invocation rate, temporary queuing may occur. By contrasting these metrics with predicted values from the theoretical M/M/k or M/M/∞ frameworks, one can evaluate the accuracy of queueing-based performance models in practical environments. In many cases, real systems deviate from idealized assumptions due to container re-initializations, memory variations, and network factors. [46]

Memory allocation experiments can illustrate the trade-offs described in previous sections. By deploying function variants with different memory sizes, one can measure changes in execution time, cold start latency, and cost per invocation [47]. Empirical results often show diminishing returns: beyond a certain point, increasing memory leads to marginal improvements in execution time but significantly increases cost. Conversely, minimal memory settings may cause the function to run slowly, elevating the total billed duration [48]. Such findings highlight the value of systematic experimentation in identifying near-optimal configurations.

Another dimension of experimentation revolves around data partitioning strategies [49]. Big data workloads often require subdividing datasets into manageable chunks for parallel processing across multiple function invocations. The granularity of these chunks affects concurrency and can influence latency. If too few chunks are used, concurrency is underutilized; if too many chunks are employed, overhead from orchestrating a large number of function invocations can become prohibitive [50]. Understanding how best to partition large datasets for serverless execution remains a key challenge. This issue can be approached mathematically by modeling partition sizes and arrival rates, then determining the partition size that optimally balances overhead and concurrency gains [51], [52]. For instance, let N represent the total number of records, and let ω be the partition size, so that the number of function invocations is $\frac{N}{\omega}$. One can then define a function $\Psi(\omega)$ representing the total elapsed time or cost. Minimizing $\Psi(\omega)$ with respect to ω yields insights into how to slice the data for parallel function execution. [53]

In deploying these analyses, real-world conditions introduce additional complexity. Network latencies, API call overheads, and ephemeral storage limits can all affect performance. Observing the interplay of these variables strengthens the validity of theoretical models, uncovering scenarios where adjustments are needed to account for platform-specific behaviors [54]. Experimental analysis thus remains a cornerstone of understanding how serverless

architectures behave in production-like contexts, where controlled theoretical assumptions give way to the subtleties of real systems.

6 Advanced Mathematical Modeling for Big Data

As big data workloads demand both large-scale parallelism and efficient resource utilization, advanced mathematical models can capture the evolving dynamics of serverless platforms [55]. One line of inquiry considers the interplay between arrival processes and concurrency expansions. An assumption of a Poisson arrival process can work as a first approximation, letting $\lambda(t)$ represent a time-varying arrival rate [56], [57]. We define the number of active function instances at time t as $k(t)$. Then a dynamic equation can be posited:

$$\frac{dk(t)}{dt} = \gamma\big(\lambda(t) - \mu k(t)\big),$$

where γ encapsulates the elasticity of the platform and μ is an effective service rate per instance [58]. In a highly idealized scenario, the serverless system adjusts $k(t)$ almost instantaneously to track $\lambda(t)/\mu$. In reality, platform-imposed limits and cold start times create delays in instance provisioning, producing a lag in concurrency scaling. [59]

Another sophisticated perspective involves coupling the concurrency adaptation with queueing models that incorporate state-dependent service times. Let $Q(t)$ be the queue length at time t [60]. When new events arrive and no warm instances are available, either new containers must be launched or the event must wait. If the system has an upper concurrency limit $\Lambda_{\max}$, the queue length evolves as:

$$\frac{dQ(t)}{dt} = \lambda(t) - \min(\Lambda_{\max}, k(t))\mu,$$

subject to a rule for how $k(t)$ changes. In discrete-event simulations, these equations become transitions between states [61], capturing the number of active containers, number of queued requests, and time since a container was last used [62]. Such models allow for quantitative analysis of how bursts of incoming data induce backlogs, how quickly concurrency can expand, and how cold starts reduce effective throughput.

For big data analytics, tasks can be large and may themselves be subdivided [63]. Suppose each invocation processes a batch of b records from a dataset of total size N. The time to complete one batch might depend not only on the function's core runtime but also on the overhead required to fetch data, decrypt or decompress it, and write back results [64]. A more complex model might define the service time X_b as:

$$X_b = \Omega\left(b; M, \delta\right),$$

where M is the allocated memory and δ incorporates external overhead factors [65]. The function Ω can be empirically derived from measurements. Then the total time to process the dataset becomes: [66]

$$T_{\text{total}} = \max_t \Big\{ \sum_{i=1}^{N/b} I_i(t) X_b(i) \Big\},$$

where $I_i(t)$ indicates that the ith batch is processed at time t. In parallel processing, multiple batches run simultaneously, and the concurrency limit impacts how many can execute concurrently [67]. A thorough mathematical exploration must account for the distribution of X_b across multiple invocations, potential data skew, and the cost model that accumulates billing per millisecond of function execution.

Stochastic fluid models can also be introduced [68]. In a fluid model, the workload is treated as a continuous flow instead of discrete tasks, and concurrency expansion is viewed as a rate at which the system can process the fluid. Such models can be beneficial for approximating behavior under large-scale parallelism, where discrete queueing analyses become unwieldy. By defining a rate of data ingestion $\rho(t)$ and a processing capacity $\kappa(t)$, one can write: [69]

$$\frac{dW(t)}{dt} = \rho(t) - \kappa(t),$$

where $W(t)$ is the amount of outstanding workload. When $\kappa(t)$ can spike up to a maximum concurrency multiplied by a per-instance rate, the fluid model reveals system bottlenecks [70]. For instance, if concurrency cannot scale quickly enough, $W(t)$ grows during surges, leading to potential timeouts if certain tasks must complete within strict deadlines.

An additional avenue for rigorous modeling lies in optimal concurrency management [71]. Even though serverless platforms often abstract these details, advanced users can configure concurrency settings or partition tasks in a

manner that effectively shapes concurrency behavior. One approach might define a cost function for concurrency overshoot. If concurrency is ramped up too aggressively, overhead from launching and maintaining ephemeral containers might outweigh the performance gains [72]. Thus, an objective function can be formulated to trade off completion time against overhead costs:

$$J = \int_0^T \Big(w_1 \cdot \max(0, W(t)) + w_2 \cdot c\big(k(t)\big) \Big) dt, \text{[73]}$$

where w_1 is a weight penalizing outstanding workload, w_2 is a weight penalizing concurrency-related costs, and $c(\cdot)$ is a cost function reflecting the overhead of maintaining $k(t)$ instances. Minimizing J yields a control strategy for concurrency expansion [74]. Although actual serverless environments may not expose concurrency as a fine-grained control variable, such modeling suggests strategies for partitioning tasks and scheduling invocations that approach optimal trade-offs in real deployments.

Applying these advanced models in practice demands careful calibration and validation against empirical data. Each of these mathematical formalisms, from queueing-based approaches to fluid approximations and cost optimization, highlights different facets of serverless computing [75]. By integrating them with robust experimental data, researchers and practitioners can more precisely anticipate how AWS Lambda or Google Cloud Functions behave under complex, large-scale analytic workloads, and how to tailor configurations to achieve optimal performance and cost profiles.

7 Conclusion

The exploration of serverless computing for big data analytics has revealed a multifaceted landscape where performance, scalability, and cost-efficiency intertwine [76]. The comparative analysis of AWS Lambda and Google Cloud Functions highlights the significance of cold start latencies, concurrency thresholds, and memory configurations in determining throughput and responsiveness. Even as these platforms abstract many complexities through automated resource provisioning, the underlying architecture exerts substantial influence on workload performance, particularly when task intensities fluctuate or when large datasets demand extensive parallel processing [77]. Mathematical modeling contributes to a deeper understanding of these dynamics by offering frameworks for analyzing how function invocations, queueing delays, and capacity scaling combine to shape real-time responsiveness.

The interplay between performance and cost underscores the value of optimizing memory allocations and partitioning strategies for specific workload profiles. Detailed experimental evaluations show how marginal gains in execution speed may be offset by elevated billing, while overly conservative resource settings risk prolonging runtimes or creating concurrency bottlenecks [78]. Advanced models, including queueing systems, fluid approximations, and control-theoretic perspectives, provide a lens through which to predict bottlenecks, quantify overhead, and propose fine-grained adjustments to concurrency. By grounding these models in empirical data [79], one obtains a robust toolkit for designing serverless architectures that are both cost-effective and high-performing. [80]

Challenges remain in translating the insights of theoretical models to production-scale applications, given the complexities of real-time event handling, ephemeral container initialization, and evolving workloads. Nonetheless, the paradigm of serverless computing continues to evolve as providers refine their platforms and developers refine strategies to mitigate cold starts and resource constraints [81]. The findings presented here aim to guide future experimentation and modeling work, illuminating the opportunities and constraints inherent in serverless environments. As big data continues to grow in volume and velocity, serverless platforms will likely play an increasingly important role in data analytics pipelines. Comprehending their behaviors, limitations, and optimization avenues is a critical step toward fully harnessing their potential for agile, cost-efficient computation. [82], [83]

References

[1] O. Kreidieh, J. Whitaker, C. J. Thurber, *et al.*, "Utility of a cloud-based lesion data collection software to record, monitor, and analyze an ablation strategy.," *Heart rhythm O2*, vol. 3, no. 3, pp. 319–322, Apr. 6, 2022. DOI: 10.1016/j.hroo.2022.03.006.

[2] L. Haghnegahdar, S. S. Joshi, and N. B. Dahotre, "From iot-based cloud manufacturing approach to intelligent additive manufacturing: Industrial internet of things—an overview," *The International Journal of Advanced Manufacturing Technology*, vol. 119, no. 3-4, pp. 1461–1478, Jan. 3, 2022. DOI: 10.1007/s00170-021-08436-x.

[3] J. M. Tien, "Toward the fourth industrial revolution on real-time customization," *Journal of Systems Science and Systems Engineering*, vol. 29, no. 2, pp. 127–142, Apr. 11, 2020. DOI: 10.1007/s11518-019-5433-9.

[4] V. Navale, D. von Kaeppler, and M. McAuliffe, "An overview of biomedical platforms for managing research data," *Journal of Data, Information and Management*, vol. 3, no. 1, pp. 21–27, Jan. 23, 2021. DOI: 10.1007/s42488-020-00040-0.

[5] M. Ayachi, H. Nacer, and H. Slimani, "Cooperative game approach to form overlapping cloud federation based on inter-cloud architecture," *Cluster Computing*, vol. 24, no. 2, pp. 1551–1577, Mar. 6, 2021. DOI: 10.1007/s10586-021-03253-z.

[6] R. Q. Cao, D. G. Schniederjans, and V. C. Gu, "Stakeholder sentiment in service supply chains: Big data meets agenda-setting theory," *Service Business*, vol. 15, no. 1, pp. 151–175, Feb. 5, 2021. DOI: 10.1007/s11628-021-00437-w.

[7] A. Klimentov, M. Grigorieva, A. Kiryanov, and A. Zarochentsev, "Bigdata and computing challenges in high energy and nuclear physics," *Journal of Instrumentation*, vol. 12, no. 06, pp. C06044–C06044, Jun. 29, 2017. DOI: 10.1088/1748-0221/12/06/c06044.

[8] M. Hernandez-de-Menendez, C. A. E. Díaz, and R. Morales-Menendez, "Engineering education for smart 4.0 technology: A review," *International Journal on Interactive Design and Manufacturing (IJIDeM)*, vol. 14, no. 3, pp. 789–803, Jul. 29, 2020. DOI: 10.1007/s12008-020-00672-x.

[9] X. Zeng, S. Garg, M. Barika, *et al.*, "Sla management for big data analytical applications in clouds: A taxonomy study," *ACM Computing Surveys*, vol. 53, no. 3, pp. 1–40, Jun. 12, 2020. DOI: 10.1145/3383464.

[10] H. Yuan, J. Bi, and M. Zhou, "Temporal task scheduling of multiple delay-constrained applications in green hybrid cloud," *IEEE Transactions on Services Computing*, vol. 14, no. 5, pp. 1558–1570, Sep. 1, 2021. DOI: 10.1109/tsc.2018.2878561.

[11] M. Kansara, "A structured lifecycle approach to large-scale cloud database migration: Challenges and strategies for an optimal transition," *Applied Research in Artificial Intelligence and Cloud Computing*, vol. 5, no. 1, pp. 237–261, 2022.

[12] D. Sigdel, V. Kyi, A. Zhang, *et al.*, "Cloud-based phrase mining and analysis of user-defined phrase-category association in biomedical publications.," *Journal of visualized experiments : JoVE*, no. 144, Feb. 23, 2019. DOI: 10.3791/59108.

[13] Y. Yang, X. Zheng, V. Chang, Y. Shaozhen, and C. Tang, "Lattice assumption based fuzzy information retrieval scheme support multi-user for secure multimedia cloud," *Multimedia Tools and Applications*, vol. 77, no. 8, pp. 9927–9941, Mar. 28, 2017. DOI: 10.1007/s11042-017-4560-x.

[14] T. S. Kolmykova, A. S. Obukhova, S. V. Klykova, P. N. Mashegov, A. G. Zaitsev, and O. V. Popova, "Features and benefits of digital technologies in agricultural enterprises," *E3S Web of Conferences*, vol. 247, pp. 01 018–, Apr. 5, 2021. DOI: 10.1051/e3sconf/202124701018.

[15] R. Cottrell, C. Fang, A. Hanushevsky, W. Kreuger, and W. Yang, "High performance data transfer," *Journal of Physics: Conference Series*, vol. 898, no. 6, pp. 062 025–, Nov. 22, 2017. DOI: 10.1088/1742-6596/898/6/062025.

[16] J. N. Pritikin, J. E. Schmitt, and M. C. Neale, "Cloud computing for voxel-wise sem analysis of mri data.," *Structural equation modeling : a multidisciplinary journal*, vol. 26, no. 3, pp. 470–480, Oct. 2, 2018. DOI: 10.1080/10705511.2018.1521285.

[17] H. Song, xiu-ying Han, C. E. Montenegro-Marin, and S. Krishnamoorthy, "Secure prediction and assessment of sports injuries using deep learning based convolutional neural network," *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, no. 3, pp. 3399–3410, Mar. 15, 2021. DOI: 10.1007/s12652-020-02560-4.

[18] M. K. Tageldeen, S. A. N. Gowers, C. L. Leong, M. G. Boutelle, and E. M. Drakakis, "Traumatic brain injury neuroelectrochemical monitoring: Behind-the-ear micro-instrument and cloud application," *Journal of neuroengineering and rehabilitation*, vol. 17, no. 1, pp. 1–16, Aug. 21, 2020. DOI: 10.1186/s12984-020-00742-x.

[19] A. Gupta, A. V. Deokar, L. S. Iyer, R. Sharda, and D. Schrader, "Big data & analytics for societal impact: Recent research and trends," *Information Systems Frontiers*, vol. 20, no. 2, pp. 185–194, Mar. 18, 2018. DOI: 10.1007/s10796-018-9846-7.

[20] P. Cai and Q. Jiang, "Gis spatial information sharing of smart city based on cloud computing," *Cluster Computing*, vol. 22, no. 6, pp. 14 435–14 443, Mar. 19, 2018. DOI: 10.1007/s10586-018-2311-z.

[21] Y. Chen, M. Lei, W. Ren, Y. Ren, and Z. Qu, "Rofa: A robust and flexible fine-grained access control scheme for mobile cloud and iot based medical monitoring," *Fundamenta Informaticae*, vol. 157, no. 1-2, pp. 167–184, Jan. 24, 2018. DOI: 10.3233/fi-2018-1624.

[22] F. N. Al-Obeidat, A. Bani-Hani, O. Adedugbe, M. Majdalawieh, and E. Benkhelifa, "A microservices persistence technique for cloud-based online social data analysis," *Cluster Computing*, vol. 24, no. 3, pp. 2341–2353, Mar. 30, 2021. DOI: 10.1007/s10586-021-03244-0.

[23] B. Fang, Y. Jia, X. Li, A. Li, and X. Wu, "Big search in cyberspace," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 9, pp. 1793–1805, Sep. 1, 2017. DOI: 10.1109/tkde.2017.2699675.

[24] S. V. G. Subrahmanya, D. K. Shetty, V. Patil, *et al.*, "The role of data science in healthcare advancements: Applications, benefits, and future prospects.," *Irish journal of medical science*, vol. 191, no. 4, pp. 1–11, Aug. 16, 2021. DOI: 10.1007/s11845-021-02730-z.

[25] J. H. Moore, "Ten important roles for academic leaders in data science," *BioData mining*, vol. 13, no. 1, pp. 18–, Oct. 26, 2020. DOI: 10.1186/s13040-020-00228-5.

[26] E. G. M. Petrakis, F. Antonopoulos, S. Sotiriadis, and N. Bessis, "Ipacs: A physical access control system as a service and mobile application," *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, no. 3, pp. 929–943, Jan. 23, 2019. DOI: 10.1007/s12652-019-01205-5.

[27] Y. Zheng, "Urban computing: Enabling urban intelligence with big data," *Frontiers of Computer Science*, vol. 11, no. 1, pp. 1–3, Apr. 7, 2017. DOI: 10.1007/s11704-016-6907-2.

[28] L. Qian, J. Yu, G. Zhu, *et al.*, "Overview of cloud computing," *IOP Conference Series: Materials Science and Engineering*, vol. 677, no. 4, pp. 042 098–, Dec. 1, 2019. DOI: 10.1088/1757-899x/677/4/042098.

[29] L. Duan and L. D. Xu, "Data analytics in industry 4.0: A survey.," *Information systems frontiers : a journal of research and innovation*, vol. 26, no. 6, pp. 1–17, Aug. 24, 2021. DOI: 10.1007/s10796-021-10190-0.

[30] R. Dubey, D. Bryde, G. Graham, C. Foropon, S. Kumari, and O. K. Gupta, "The role of alliance management, big data analytics and information visibility on new-product development capability," *Annals of operations research*, vol. 333, no. 2-3, pp. 1–25, Nov. 19, 2021. DOI: 10.1007/s10479-021-04390-9.

[31] Z.-H. Jin, H. Shi, Y.-X. Hu, L. Zha, and X. Lu, "Cirrodata: Yet another sql-on-hadoop data analytics engine with high performance," *Journal of Computer Science and Technology*, vol. 35, no. 1, pp. 194–208, Jan. 17, 2020. DOI: 10.1007/s11390-020-9536-z.

[32] Z. Kuang, H. Zhou, D. Zhou, J.-p. Zhou, and K. Yang, "A non-group parallel frequent pattern mining algorithm based on conditional patterns," *Frontiers of Information Technology & Electronic Engineering*, vol. 20, no. 9, pp. 1234–1245, Oct. 18, 2019. DOI: 10.1631/fitee.1800467.

[33] S. Dolev, P. G. S. Florissi, E. Gudes, S. Sharma, and I. Singer, "A survey on geographically distributed big-data processing using mapreduce," *IEEE Transactions on Big Data*, vol. 5, no. 1, pp. 60–80, Mar. 1, 2019. DOI: 10.1109/tbdata.2017.2723473.

[34] S. Bebortta, S. K. Das, M. Kandpal, R. K. Barik, and H. Dubey, "Geospatial serverless computing: Architectures, tools and future directions," *ISPRS International Journal of Geo-Information*, vol. 9, no. 5, pp. 311–, May 7, 2020. DOI: 10.3390/ijgi9050311.

[35] S. Latif, J. Qadir, S. Farooq, and M. Imran, "How 5g wireless (and concomitant technologies) will revolutionize healthcare," *Future Internet*, vol. 9, no. 4, pp. 93–, Dec. 11, 2017. DOI: 10.3390/fi9040093.

[36] K. Sheng-Kai, "Integrating travel history via big data analytics under universal healthcare framework for disease control and prevention in the covid-19 pandemic.," *Journal of clinical epidemiology*, vol. 130, pp. 147–148, Sep. 23, 2020. DOI: 10.1016/j.jclinepi.2020.08.016.

[37] M.-C. Chuang, C.-C. Yen, and C.-J. Hung, "Bandwidth-aware rescheduling mechanism in sdn-based data center networks," *Electronics*, vol. 10, no. 15, pp. 1774–, Jul. 24, 2021. DOI: 10.3390/electronics10151774.

[38] M. Kansara, "A comparative analysis of security algorithms and mechanisms for protecting data, applications, and services during cloud migration," *International Journal of Information and Cybersecurity*, vol. 6, no. 1, pp. 164–197, 2022.

[39] Z. Zhang and M. J. C. Crabbe, "Management of environmental streaming data to optimize arctic shipping routes.," *Arabian Journal of Geosciences*, vol. 14, no. 15, pp. 1–8, Jul. 20, 2021. DOI: 10.1007/s12517-021-07782-0.

[40] T. Kanwal, A. Anjum, and A. Khan, "Privacy preservation in e-health cloud: Taxonomy, privacy requirements, feasibility analysis, and opportunities," *Cluster Computing*, vol. 24, no. 1, pp. 293–317, Apr. 22, 2020. DOI: 10.1007/s10586-020-03106-1.

[41] R. Avula, "Overcoming data silos in healthcare with strategies for enhancing integration and interoperability to improve clinical and operational efficiency," *Journal of Advanced Analytics in Healthcare Management*, vol. 4, no. 10, pp. 26–44, 2020.

[42] B. Ahmad, Z. Maroof, S. McClean, D. Charles, and G. Parr, "Economic impact of energy saving techniques in cloud server," *Cluster Computing*, vol. 23, no. 2, pp. 611–621, Jun. 7, 2019. DOI: 10.1007/s10586-019-02946-w.

[43] A. Collins and A. R. Jones, "Phpms: A php-based mass spectrometry utilities library.," *Journal of proteome research*, vol. 17, no. 3, pp. 1309–1313, Feb. 13, 2018. DOI: 10.1021/acs.jproteome.7b00783.

[44] S. Luo, G. Zhang, C. Wu, S. U. Khan, and K. Li, "Boafft: Distributed deduplication for big data storage in the cloud," *IEEE Transactions on Cloud Computing*, vol. 8, no. 4, pp. 1199–1211, Oct. 1, 2020. DOI: 10.1109/tcc.2015.2511752.

[45] J. H. Kim, M. Yoo, K. N. Lee, and H. Seo, "The innovation of the internet: A semantic network analysis of the internet of things," *Asian Journal of Technology Innovation*, vol. 25, no. 1, pp. 129–139, Jan. 2, 2017. DOI: 10.1080/19761597.2017.1302549.

[46] M. S. Wajid, H. Terashima-Marin, P. N. P. Rad, and M. A. Wajid, "Violence detection approach based on cloud data and neutrosophic cognitive maps," *Journal of Cloud Computing*, vol. 11, no. 1, Nov. 30, 2022. DOI: 10.1186/s13677-022-00369-4.

[47] L. Wei, C. H. Foh, B. He, and J. Cai, "Towards efficient resource allocation for heterogeneous workloads in iaas clouds," *IEEE Transactions on Cloud Computing*, vol. 6, no. 1, pp. 264–275, Jan. 1, 2018. DOI: 10.1109/tcc.2015.2481400.

[48] R. Madduri, K. Chard, M. D'Arcy, *et al.*, "Reproducible big data science: A case study in continuous fairness.," *PloS one*, vol. 14, no. 4, pp. 0 213 013–, Apr. 11, 2019. DOI: 10.1371/journal.pone.0213013.

[49] M. Hanif, E. Kim, S. Helal, and C. Lee, "Sla-based adaptation schemes in distributed stream processing engines," *Applied Sciences*, vol. 9, no. 6, pp. 1045–, Mar. 13, 2019. DOI: 10.3390/app9061045.

[50] M. Shahin, F. F. Chen, H. Bouzary, and K. Krishnaiyer, "Integration of lean practices and industry 4.0 technologies: Smart manufacturing for next-generation enterprises," *The International Journal of Advanced Manufacturing Technology*, vol. 107, no. 5, pp. 2927–2936, Mar. 28, 2020. DOI: 10.1007/s00170-020-05124-0.

[51] A. Vanky, S. K. Verma, T. K. Courtney, P. Santi, and C. Ratti, "Effect of weather on pedestrian trip count and duration: City-scale evaluations using mobile phone application data," *Preventive medicine reports*, vol. 8, pp. 30–37, Jul. 27, 2017. DOI: 10.1016/j.pmedr.2017.07.002.

[52] R. Avula, "Architectural frameworks for big data analytics in patient-centric healthcare systems: Opportunities, challenges, and limitations," *Emerging Trends in Machine Intelligence and Big Data*, vol. 10, no. 3, pp. 13–27, 2018.

[53] L. Abualigah and A. Diabat, "A novel hybrid antlion optimization algorithm for multi-objective task scheduling problems in cloud computing environments," *Cluster Computing*, vol. 24, no. 1, pp. 205–223, Mar. 12, 2020. DOI: 10.1007/s10586-020-03075-5.

[54] P. Liu, X. Wang, F. Wen, *et al.*, "Development and application of big data platform for "bohai granary"," *Wireless Personal Communications*, vol. 103, no. 1, pp. 275–293, Feb. 9, 2018. DOI: 10.1007/s11277-018-5441-y.

[55] W.-d. Xie and X. Cheng, "Imbalanced big data classification based on virtual reality in cloud computing," *Multimedia Tools and Applications*, vol. 79, no. 23, pp. 16 403–16 420, Feb. 20, 2019. DOI: 10.1007/s11042-019-7317-x.

[56] H. Mao, Q. Zhengwei, J. Duan, and X. Ge, "Cost-performance modeling with automated benchmarking on elastic computing clouds," *Journal of Grid Computing*, vol. 15, no. 4, pp. 557–572, Oct. 23, 2017. DOI: 10.1007/s10723-017-9412-4.

[57] M. Kansara, "Cloud migration strategies and challenges in highly regulated and data-intensive industries: A technical perspective," *International Journal of Applied Machine Learning and Computational Intelligence*, vol. 11, no. 12, pp. 78–121, 2021.

[58] I. Pintye, E. Kail, P. Kacsuk, and R. Lovas, "Big data and machine learning framework for clouds and its usage for text classification," *Concurrency and Computation: Practice and Experience*, vol. 33, no. 19, Dec. 21, 2020. DOI: 10.1002/cpe.6164.

[59] V. Arora, F. Nawab, D. Agrawal, and A. E. Abbadi, "Janus: A hybrid scalable multi-representation cloud datastore," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 4, pp. 689–702, Apr. 1, 2018. DOI: 10.1109/tkde.2017.2773607.

[60] H. Bouzary, F. F. Chen, and M. Shahin, "Using machine learning for service candidate sets retrieval in service composition of cloud-based manufacturing," *The International Journal of Advanced Manufacturing Technology*, vol. 115, no. 3, pp. 941–948, Jan. 7, 2021. DOI: 10.1007/s00170-020-06381-9.

[61] A. Sharma and K. M. Goolsbey, "Simulation-based approach to efficient commonsense reasoning in very large knowledge bases," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 1360–1367.

[62] S. Hamdan, M. Ayyash, and S. Almajali, "Edge-computing architectures for internet of things applications: A survey," *Sensors (Basel, Switzerland)*, vol. 20, no. 22, pp. 6441–, Nov. 11, 2020. DOI: 10.3390/s20226441.

[63] X. Yao, J. Zhou, Y. Lin, Y. Li, H. Yu, and Y. Liu, "Smart manufacturing based on cyber-physical systems and beyond," *Journal of Intelligent Manufacturing*, vol. 30, no. 8, pp. 2805–2817, Dec. 28, 2017. DOI: 10.1007/s10845-017-1384-5.

[64] A. Dargazany, P. Stegagno, and K. Mankodiya, "Wearabledl: Wearable internet-of-things and deep learning for big data analytics—concept, literature, and future," *Mobile Information Systems*, vol. 2018, pp. 1–20, Nov. 14, 2018. DOI: 10.1155/2018/8125126.

[65] J. A. Delgado, N. M. Short, D. P. Roberts, and B. Vandenberg, "Big data analysis for sustainable agriculture on a geospatial cloud framework," *Frontiers in Sustainable Food Systems*, vol. 3, Jul. 16, 2019. DOI: 10.3389/fsufs.2019.00054.

[66] C. Hesselman, P. Grosso, R. Holz, *et al.*, "A responsible internet to increase trust in the digital world," *Journal of Network and Systems Management*, vol. 28, no. 4, pp. 882–922, Sep. 7, 2020. DOI: 10.1007/s10922-020-09564-7.

[67] T. Castrignanò, S. Gioiosa, T. Flati, *et al.*, "Elixir-it hpc@cineca: High performance computing resources for the bioinformatics community," *BMC bioinformatics*, vol. 21, no. 10, pp. 1–17, Aug. 21, 2020. DOI: 10.1186/s12859-020-03565-8.

[68] B. Zhu, J. Sun, J. Qin, and J. Ma, "Fuzzy matching: Multi-authority attribute searchable encryption without central authority," *Soft Computing*, vol. 23, no. 2, pp. 527–536, Sep. 26, 2017. DOI: 10.1007/s00500-017-2849-3.

[69] C.-T. Yang, T.-Y. Chen, E. Kristiani, and S. F. Wu, "The implementation of data storage and analytics platform for big data lake of electricity usage with spark," *The Journal of Supercomputing*, vol. 77, no. 6, pp. 5934–5959, Nov. 13, 2020. DOI: 10.1007/s11227-020-03505-6.

[70] D. Gupta, S. Rani, and S. H. Ahmed, "Icn-edge caching scheme for handling multimedia big data traffic in smart cities," *Multimedia Tools and Applications*, vol. 82, no. 25, pp. 39 697–39 717, Aug. 5, 2022. DOI: 10.1007/s11042-022-13518-3.

[71] A. Adeel, J. Ahmad, H. Larijani, and A. Hussain, "A novel real-time, lightweight chaotic-encryption scheme for next-generation audio-visual hearing aids," *Cognitive Computation*, vol. 12, no. 3, pp. 589–601, Nov. 13, 2019. DOI: 10.1007/s12559-019-09653-z.

[72] H. A. Shabeer, P. P. Ramaswamy, H. A. Zubar, and R. S. D. W. Banu, "Editorial: Green cloud computing and communication," *Mobile Networks and Applications*, vol. 25, no. 4, pp. 1287–1289, Jun. 2, 2020. DOI: 10.1007/s11036-020-01553-z.

[73] H. Cui, C. Wang, and H. Liu, "Monitoring and analysis of distributed new energy resources based on the internet of things," *IOP Conference Series: Earth and Environmental Science*, vol. 714, no. 4, pp. 042 025–, Mar. 1, 2021. DOI: 10.1088/1755-1315/714/4/042025.

[74] S. Shafqat, M. Fayyaz, H. A. Khattak, *et al.*, "Leveraging deep learning for designing healthcare analytics heuristic for diagnostics.," *Neural processing letters*, vol. 55, no. 1, pp. 1–27, Feb. 2, 2021. DOI: 10.1007/s11063-021-10425-w.

[75] Y. Sabri, F. Bahja, and H. Pet, "Integration of remote sensing data in a cloud," *International Journal of Electronics and Telecommunications*, pp. 167–172, Dec. 7, 2021. DOI: 10.24425/ijet.2022.139864.

[76] K. Börner, F. N. Silva, and S. Milojević, "Visualizing big science projects," *Nature Reviews Physics*, vol. 3, no. 11, pp. 753–761, Sep. 28, 2021. DOI: 10.1038/s42254-021-00374-7.

[77] S. S. Muhammad, B. L. Dey, and V. Weerakkody, "Analysis of factors that influence customers' willingness to leave big data digital footprints on social media: A systematic review of literature," *Information Systems Frontiers*, vol. 20, no. 3, pp. 559–576, Oct. 15, 2017. DOI: 10.1007/s10796-017-9802-y.

[78] M. R. Kamdar and M. A. Musen, "An empirical meta-analysis of the life sciences linked open data on the web," *Scientific data*, vol. 8, no. 1, pp. 24–24, Jan. 21, 2021. DOI: 10.1038/s41597-021-00797-y.

[79] R. Avula, "Optimizing data quality in electronic medical records: Addressing fragmentation, inconsistencies, and data integrity issues in healthcare," *Journal of Big-Data Analytics and Cloud Computing*, vol. 4, no. 5, pp. 1–25, 2019.

[80] Y. Wu, Z. Zhang, C. Wu, C. Guo, Z. Li, and F. C. M. Lau, "Orchestrating bulk data transfers across geo-distributed datacenters," *IEEE Transactions on Cloud Computing*, vol. 5, no. 1, pp. 112–125, Jan. 1, 2017. DOI: 10.1109/tcc.2015.2389842.

[81] R. G. Baraniuk, D. L. Donoho, and M. Gavish, "The science of deep learning.," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 117, no. 48, pp. 30 029–30 032, Nov. 23, 2020. DOI: 10.1073/pnas.2020596117.

[82] Á. M. García-Vico, P. González, C. J. Carmona, and M. J. del Jesus, "A big data approach for the extraction of fuzzy emerging patterns," *Cognitive Computation*, vol. 11, no. 3, pp. 400–417, Jan. 4, 2019. DOI: 10.1007/s12559-018-9612-7.

[83] S. Shekhar, "Integrating data from geographically diverse non-sap systems into sap hana: Implementation of master data management, reporting, and forecasting model," *Emerging Trends in Machine Intelligence and Big Data*, vol. 10, no. 3, pp. 1–12, 2018.