

Performance Measurements in the AI-Centric Computing Continuum Systems

Praveen Kumar Donta*, Qiyang Zhang[†] and Schahram Dustdar[‡]

*Department of Computer Systems and Sciences, Stockholm University, Stockholm 164 25, Sweden.

[†]Computer Science School, Peking University, Beijing 100876, China.

[‡]Distributed Systems Group, TU Wien, Vienna 1040, Austria and ICREA Barcelona 08002, Spain

praveen@dsv.su.se, qiyangzhang@pku.edu.cn, dustdar@dsg.tuwien.ac.at

Abstract—Over the Eight decades, computing paradigms have shifted from large, centralized systems to compact, distributed architectures, leading to the rise of the Distributed Computing Continuum (DCC). In this model, multiple layers such as cloud, edge, Internet of Things (IoT), and mobile platforms work together to support a wide range of applications. Recently, the emergence of Generative AI and large language models has further intensified the demand for computational resources across this continuum. Although traditional performance metrics have provided a solid foundation, they need to be revisited and expanded to keep pace with changing computational demands and application requirements. Accurate performance measurements benefit both system designers and users by supporting improvements in efficiency and promoting alignment with system goals. In this context, we review commonly used metrics in DCC and IoT environments. We also discuss emerging performance dimensions that address evolving computing needs, such as sustainability, energy efficiency, and system observability. We also outline criteria and considerations for selecting appropriate metrics, aiming to inspire future research and development in this critical area.

Index Terms—Internet of things; Performance Measurements; Distributed Computing Continuum; Artificial Intelligence; and Quality of Service

I. INTRODUCTION

COMPUTING paradigms have evolved significantly over the past few decades, moving from large, room-sized processors and memory units to compact and efficient computing nodes. At the same time, system architectures have shifted from centralized models to distributed frameworks. One of the important developments in this direction is the Distributed Computing Continuum (DCC), which reflects a major change in how computing systems are designed and utilized [1]. In contrast to earlier models, where data centers, cloud platforms, edge devices, and Internet of Things (IoT) systems operated independently, the DCC brings these layers together into a unified and collaborative environment [2]–[4]. This integration enables applications to choose the most suitable layer for executing tasks or processing data. Depending on the requirement, processing can take place in a central data center, on cloud infrastructure, at the network edge, or directly on resource-constrained devices such as sensors or IoT nodes [5]. In many cases, applications may

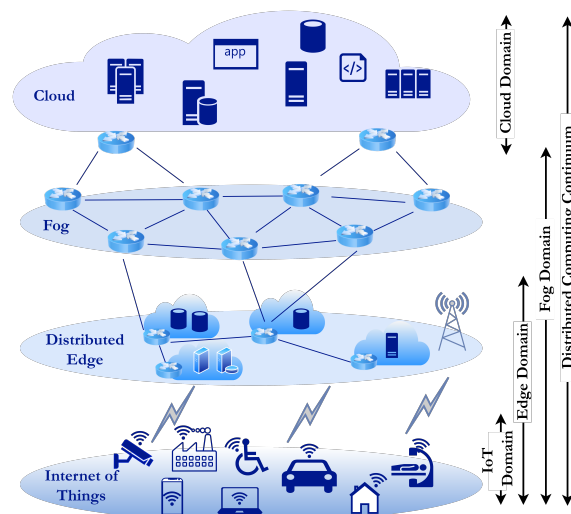


Fig. 1: Conceptual Framework of DCC Architecture

also distribute their workloads across multiple layers at the same time to utilize resources efficiently and achieve highest quality of service (QoS).

On the other side of this evolution, the emergence of Generative AI (GenAI) and large language models (LLMs) has introduced new dimensions to computing [6]–[8]. These models require significantly higher computational power compared to traditional applications, and their growing adoption has placed unprecedented demands on underlying infrastructures. As a result, there has been a noticeable shift towards high-performance computing systems, supported by advanced processors, specialized Graphics processing units (GPU), accelerators, and optimized memory and storage units [9]. Over the past few years, global investment in AI hardware has steadily increased [10], with leading data centers and research institutions upgrading their systems to handle larger workloads and improve throughput.

However, these technological advancements also highlight a growing mismatch between modern application demands and traditional performance measurement strategies [11], [12]. Most conventional metrics were developed with static,

centralized systems in mind and often fall short in capturing the complexity, scale, and heterogeneity of today's distributed environments [13]. This challenge becomes even more relevant in environments like the DCC, where powerful servers must work in coordination with lightweight, resource-constrained IoT devices. The wide variation in processing power, network reliability, and energy limitations across these layers makes it difficult to use a single method to assess system performance. Therefore, it is important to revisit existing evaluation strategies and design performance metrics that reflect the continuous evolution of computing systems and the growing demands of modern applications.

In this context, we examine commonly used performance metrics and highlight where they may fall short in meeting the needs of today's diverse and changing computing environments. We also suggest new directions for performance measurement that are better suited to modern infrastructure and emerging application demands. In summary, the main contributions of this paper are as follows:

- We explore traditional performance metrics at the computing, network, and application or user level.
- We provide theoretical novel performance measurement approaches for the changing computational demands
- We discuss the criteria to choose performance metrics according to their goals and characteristics regarding the available resources and applications demands.

II. IMPORTANCE OF PERFORMANCE MEASUREMENTS

Measuring performance of computing systems is crucial for various reasons and this holds equally true for DCC. Firstly, it aids in the selection and architecture of computer systems by allowing practitioners to make informed choices based on performance and cost considerations. Performance evaluation also plays an important role in the design of systems and applications, enabling designers to choose configurations that offer the best performance. Moreover, it allows for the analysis of existing systems, identifying areas for improvement and optimization. In order to improve computing systems efficiency, it is essential to assess computer performance, which provides insights into component capacity, hardware, software, and operational procedures [14]. In addition, performance evaluations help determine compatibility with various application requirements.

As discussed earlier, DCC differ from regular computing systems in that they have distributed, interconnected, and heterogeneous computing nodes of different types connected over network. Due to these variations, measuring DCC performance is different compared to regular computing systems. Benchmarking and specific methods tailored to application domains must be used to diagnose a computing system's performance efficiently, this holds for DCC.

Systematic performance evaluation also enables stakeholders to make informed decisions regarding system design, deployment, and management. This process supports the

identification of strengths and weaknesses within the system, highlights opportunities for improvement, and ensures that resources are utilized effectively. Moreover, regular performance assessment helps maintain the reliability, scalability, and adaptability of DCCS in the face of changing workloads, technologies, and user requirements. Ultimately, robust performance measurement fosters innovation and continuous improvement, enabling distributed systems to meet the diverse and evolving needs of modern applications.

III. TRADITIONAL PERFORMANCE MEASUREMENTS

A. Common goals of performance metrics in DCC

Measuring DCC performance depends on the system's operating conditions and its intended purpose. Still, several objectives remain consistently important for both developers and users, such as performance optimization, scalability, maintaining QoS, early identification of bottlenecks, energy efficiency, cost reduction, and strengthening reliability and security [15]. Traditional performance optimization focuses on minimizing latency and maximizing resource utilization across the system. Scalability involves adapting to varying workloads without compromising performance, while QoS ensures that applications continue to meet expected service levels. In addition, system performance can be influenced by factors such as network latency, limited bandwidth, inefficient processing methods, or hardware constraints. Most IoT devices use low-power batteries, while cloud devices with high computing power demand huge energy, all in all, results in large energy costs and carbon emissions, which affect the sustainability of DCC. The operational costs of DCC are not only limited to energy but also to computing and memory resources. A more efficient algorithm might be used to perform the high computations within limited resources to minimize the overall infrastructure costs. Reliability aims to ensure a high rate of recovery from failures and provide uninterrupted services. Security is an essential goal and it can continuously identify vulnerabilities or security threats in the system [16]. In general, detecting and addressing these issues early is essential to maintain stability, prevent degradation, and ensure long-term system effectiveness.

B. Commonly used Performance Measurements

Currently, researchers use a variety of performance metrics to evaluate DCC systems either as a whole or across specific layers, depending on the system designers. This paper categorizes these metrics into three groups: computing-level metrics, which capture system resource behavior; network-level metrics, which reflect communication efficiency; and application-level metrics, which assess user experience and data processing. The taxonomy of these performance metrics is illustrated in Fig. 2 and discussed in detail below.

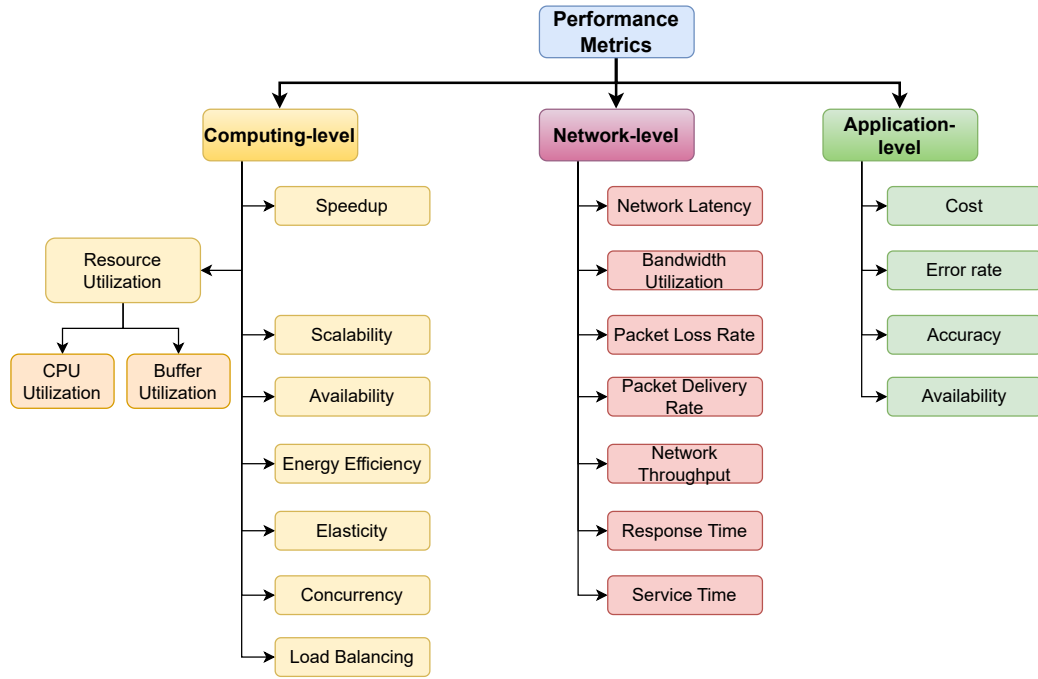


Fig. 2: A taxonomy of commonly used performance measurements in DCC and IoT Networks

1) *Computing-level Performance Measurements*: are essential for understanding a comprehensive view of system behavior. These resources span across various components such as CPUs, memory, storage, and specialized hardware like GPUs or FPGAs distributed through cloud servers, edge devices, and IoT nodes. Measuring performance in such heterogeneous and dynamic settings requires a nuanced understanding of several critical factors. Metrics such as **speedup** provide insight into how much faster a task executes when distributed across multiple nodes compared to a single processor, helping to gauge the benefits of parallelism. **Resource utilization** reveals the extent to which the available hardware (CPU, Memory, other peripherals) is being used, highlighting potential over- or under-utilization of critical components. **Scalability** measures the system's ability to handle increasing workloads by expanding resources, which is vital in scenarios where demand fluctuates rapidly. **Elasticity** captures how well the system adjusts resources in response to changing demands, maintaining cost-effectiveness and performance stability. **Availability** ensures that the system remains operational and accessible, an important aspect for applications requiring high reliability. In addition, **energy utilization** has become increasingly significant as the environmental impact and operational costs of large-scale deployments grow. Finally, **concurrency** reflects the system's capacity to perform multiple operations simultaneously to improve parallelism.

2) *Network-level Performance Measurement*: evaluate the communication infrastructure to detect communication bot-

tlenecks, fine-tune data routing, balance workloads (checking congestion), and maintain QoS. As cloud servers, edge devices, and IoT nodes in DCC depends on continuous data exchange to function cohesively, understanding network behavior becomes crucial. Since DCC is widely distributed and inherently heterogeneous environments, maintaining stable and efficient network performance to ensuring reliable system operation. One of the core considerations is **network latency**, which refers to the time it takes for data to travel between two nodes. **Bandwidth** represents the maximum data capacity of the network, while **throughput** reflects the actual volume of data successfully transferred over time. Sustained high throughput is vital for handling large-scale data and maintaining synchronization across distributed nodes. **Packet loss rate** captures the percentage of data packets that fail to reach their destination, potentially degrading both performance and reliability. In contrast, the **packet delivery rate** indicates the proportion of successfully received packets and serves as a measure of communication stability. **Network utilization** shows how much of the available capacity is in use, to monitor operational efficiency.

3) *Application or User-level Performance Measurement*: focus on how effectively services and applications function from the perspective of the end user within a DCC. These metrics help evaluate service quality, reliability, and the system's ability to meet specific application demands. **Response time**, which includes processing, queuing, and network delays, reflects how quickly the system responds to user requests and is key to ensuring smooth and interactive

experiences. **Service time**, on the other hand, measures how long it takes to complete a given task, offering insight into the system's processing efficiency. **Cost** metrics are used to understand the economic impact of running applications, including the use of computing resources and overall operational overhead. **Reliability** is captured through the error rate, indicating how often applications fail or produce incorrect results, while accuracy measures the correctness of output. **Availability** tracks whether services are accessible when users need them.

IV. NOVEL PERFORMANCE MEASUREMENTS

A. Novel Metrics

Many performance measures exist in the literature to evaluate computing systems. However, the continuous growth of these systems and the evolving demands of applications make it clear that *new and more relevant evaluation* methods are needed to accurately assess system performance.

1) *CO₂ emissions*: GenAI and LLMs require large amounts of computing power, which leads to high electricity use and serious environmental impact. Training a single advanced model can consume as much electricity as several hundred homes use in a year, producing hundreds of tons of carbon (CO₂) emissions. After training, running these models for daily user requests continues to use significant computational power (which further use energy), with each query often consuming more power than a normal web search. Data centers that support these systems also need large volumes of water for cooling, which can affect local water availability. As demand increases, emissions from data centers may triple by 2030, potentially reaching 2.5 billion tons annually. In addition, production of specialized hardware like GPUs further adds to the environmental impact.

Just imagine the impact when IoT and edge devices begin integrating GenAI into their operations. With millions of such devices deployed worldwide—ranging from smart home assistants and wearable health monitors to industrial sensors and autonomous vehicles—the scale of energy consumption could increase dramatically. Unlike centralized data centers, these edge devices often operate in environments with limited power and cooling infrastructure, making energy efficiency a critical concern. Running even lightweight AI models locally requires significant processing power, which could lead to frequent battery drain, increased hardware wear, and added maintenance needs. Moreover, the cumulative energy demand of millions of devices performing AI tasks, even intermittently, could place additional pressure on electricity grids and raise global CO₂ emissions. Therefore, it is important to consider CO₂-related metrics when designing new system models.

Vahdat et al. [17] proposed a performance evaluation metric to estimate carbon dioxide and equivalent emissions, referred to as goodput. Their approach incorporates established standards such as the Greenhouse Gas Protocol

and ISO 14040 and 14044 to ensure accurate and reliable measurements. In the context of DCC and IoT, this metric is essential due to the rising importance of environmental responsibility and the drive toward sustainability.

2) *Heat Dissipation*: Similar to CO₂ emissions, the heat dissipation metric is closely linked to the environmental impact of DCC and IoT systems. Heat dissipation refers to the amount of heat produced and released by devices during their operation. This metric influences not only energy efficiency but also cooling needs and the overall sustainability of the system. To measure this metric, a popular approach, *Newton's law of cooling* (shown in Eq. (1)), can be used [18]. In this, the rate of heat loss of a device is proportional to the difference in temperature between the device and its environment. It helps in designing cooling strategies for data centers, edge devices, and IoT nodes by predicting temperature variations and optimizing cooling resources.

$$T(t) = T_e + (T_0 - T_e)e^{-kt} \quad (1)$$

where $T(t)$ is the device temperature at time t ; T_e denotes the ambient (environmental) temperature of the location where the device is deployed; T_0 represents the initial temperature of the device; k is a positive constant that depends on the properties of the device and the cooling medium (i.e., it determines the rate of heat loss); and e denotes Euler's number (approximately 2.718), which is the base of the natural logarithm.

3) *Bottlenecks*: In DCC, resource usage is often uneven. Some devices may be heavily loaded with tasks while others remain underutilized. Similarly, energy consumption can be much higher on certain nodes, and bandwidth availability may fluctuate across different parts of the network. These conditions create potential bottlenecks that are often hidden when relying on average metrics, as such measures tend to smooth out the underlying imbalances. To effectively identify and measure these hidden bottlenecks, the **fairness index** provides a valuable tool for assessing how evenly resources are distributed across the system. *Jain's fairness index* is one of the most popular metric to measure it [19], and shown in Eq. (2).

$$\mathcal{F}(x_1, x_2, \dots, x_n) = \frac{(\sum_{i=1}^n x_i)^2}{n \sum_{i=1}^n x_i^2} \quad (2)$$

where x_i represents the resource metric for the i^{th} node—such as CPU load, energy consumption, or bandwidth usage—and n is the total number of nodes. The index produces a value between 0 and 1, where 1 indicates perfect no bottlenecks, and values closer to 0 indicate increasing bottlenecks. Several fairness index variants based on Jain's approach have been proposed; Donta et al. [20] offers an alternate formulation.

4) *Measure Observability*: by making the processes and interactions within the DCC transparent and understandable,

explainability empowers real-time monitoring, debugging, and continuous improvement, ensuring that complex, heterogeneous, and autonomous systems remain accountable, reliable, and adaptable. Explainability enhances observability by enabling stakeholders to move beyond basic monitoring and gain meaningful insights into the rationale behind automated system behaviors and resource allocation decisions. This interpretability allows operators to trace the origins of actions and outcomes across cloud, edge, and IoT nodes, facilitating more effective diagnosis of issues and optimization of system performance. The observability score for a DCC is calculated as Eq. (3).

$$\mathcal{E} = \frac{\sum_{i=1}^N \gamma_i E_{\text{local}}^i}{N} \cdot \left(1 - \frac{\sum_{i \neq j} |C_{ij} - C_{ji}|}{\sum_{i,j} C_{ij}} \right) \quad (3)$$

where, N is the total number of nodes, γ_i is the weight for node i , and $E_{\text{local}}^i = (\alpha\mathbb{F} + \beta\mathbb{R})\forall(\alpha + \beta = 1)$ is the local explainability score of node i , and it depend on various factors [21] whereas our formula considered *Fidelity* ($\mathbb{F} = \mathbb{E}_{\mathbf{P}} \left[(\mathbf{P}^T \phi(\mathbf{x}) - (f(\mathbf{x}) - f(\mathbf{x} - \mathbf{P})))^2 \right]$) and *Robustness* ($\mathbb{R} = \text{sim}(\phi(\mathbf{x}), \phi(\mathbf{x} + \epsilon))$). Here, where $\phi(\mathbf{x})$ is the explanation vector at input $\mathbf{x}$, f is the model, $\mathbf{P}$ denotes the perturbation vector, ϵ represents a small noise perturbation applied to $\mathbf{x}$, and $\text{sim}(\cdot, \cdot)$ is a similarity/correlation function. C_{ij} represents causal influence from node i to node j , which can be estimated using techniques such as Granger causality or other causal inference methods [22]. In short, the first term in Eq. (3) captures the average explainability across nodes, while the second term penalizes asymmetric or unclear inter-node interactions.

5) *Adaptivity Quotient*: is a metric designed to quantify how effectively and rapidly a DCC system can sense, respond to, and recover from changes or disruptions across the continuum. In DCC environments, adaptation is not limited to simple resource scaling; it also encompasses dynamic workload migration, service offloading, quality adjustment, and reconfiguration in response to fluctuating workloads, failures, or environmental changes. As recent research introduces adaptive frameworks, the ability to adapt is crucial for ensuring service continuity, meeting application requirements, and optimizing resource use across the continuum. This metric is mathematically defined as shown in Eq.(4).

$$\mathcal{Q} = \frac{1}{K} \sum_{k=1}^K \left(\frac{P_k^{\text{post}}}{P_k^{\text{base}}} \cdot \frac{1}{T_k^{\text{adapt}}} \right) \quad (4)$$

where, K represents the total number of adaptation events observed in DCC. P_k^{before} is the metric (such as throughput or latency) before adaptation event k , and P_k^{after} is the metric after the system has adapted. T_k is the time taken to complete the adaptation for event k .

6) *Maintain Equilibrium*: Scaling up computational resources comes at a **cost**, leading to higher operational

expenses, increased energy demands (projected to consume 7% of global power by 2030), and greater *environmental impact* [23]. While investing in infrastructure enhances QoS—ensuring lower latency and better reliability—it also amplifies resource consumption and financial overhead. For instance, increasing replication factors to improve data availability demands additional storage and processing power, while low-latency techniques like indexing and caching require more computational resources. Conversely, reducing computational power or storage capacity affects QoS, creating a self-perpetuating challenge: more resources improve performance but increase costs, while cost-cutting compromises system reliability. These investments raise the question of how effectively increased resources translate into real-world performance gains. Notably, the growth of **personalized LLMs** may not require such large computing environments but necessitates careful control over infrastructure due to the associated costs, especially for small organizations (e.g., universities or public services). For example, Amdahl's Law (Eq. (5), where the variable S represents speedup and F represents fraction), which predicted this situation [24], highlights the limitation of diminishing returns: performance gains are constrained by the fraction of the workload that remains unoptimized. Therefore, the challenge for large-scale AI infrastructure initiatives lies not just in scaling hardware, but in efficiently using available resources to maximize QoS.

$$S_{\text{overall}} = \left(1 - F_{\text{enhanced}} + \frac{F_{\text{enhanced}}}{S_{\text{enhanced}}} \right)^{-1} \quad (5)$$

The interplay between Cost (C), Resources (R), and QoS (Q) creates a complex optimization challenge. Increasing resources enhances QoS but raises costs, while reducing costs can degrade performance. ($\max_{R,C} Q(R, C)$ subjected to: $C \leq C_{\text{max}}$, $R \geq R_{\text{min}}$) formalizes this trade-off, aiming QoS while adhering to cost and resource constraints. Current AI-driven infrastructure management lacks adaptive frameworks that can dynamically respond to workload fluctuations while maintaining **equilibrium**. Existing methods often rely on static provisioning or reactive scaling, leading to inefficiencies. In this context, further research should focus on AI-driven, self-optimizing platform that continuously learns from system fluctuations to achieve near-optimal equilibrium across cost, resources, and QoS, satisfying $\max_{R,C} Q(R, C)$.

B. Metrics Selection Criteria

In general, it is not necessary to evaluate all of them in every case. Deciding on appropriate or effective performance metrics is challenging, however, some characteristics can make this task easier. Next, we provide helpful guidelines to decide the set of metrics that best characterize the system. Ideally, the metric should be related to DCC objectives and

goals. It is important that the metric be *sensitive* enough to detect changes in the system's performance over time. Even minor behavior changes should be able to be detected by it. In order to be *consistent*, the units of the performance metric and its precise definition must be the same across different systems or configurations of the same system. A performance metric is *repeatable* if it is measured the same way each time an experiment is conducted. An easy-to-understand and *easy-to-use* metric is essential and maintain the results should be clear and concise. All aspects of DCC performance should be captured by the metric. In other words, it needs to take into account all relevant aspects of DCC behavior, such as resource utilization, scalability, reliability, and energy efficiency. Performance metrics should be *independent* of outside influences. When all these requirements are met, a performance metric can be easily determined as to its accuracy and relevance are considered.

V. CONCLUSIONS

This paper has explored currently used performance measurements within AI-centric DCC systems across computing, network, and user-centric metrics, highlighting their foundational role. These metrics often fall short in addressing the continuously changing computational demands and application requirements, particularly in the context of GenAI and LLM workloads. To bridge this gap, we propose a set of novel performance metrics-including CO_2 emissions, heat dissipation, bottleneck fairness, observability, adaptivity quotient, and equilibrium maintenance-that collectively enable a more holistic and nuanced evaluation of DCC systems. These metrics incorporate critical dimensions such as sustainability, energy efficiency, adaptive responsiveness, and system transparency, which are essential for guiding the optimization of resource use and service quality in DCC. In addition, we outline criteria for selecting appropriate metrics based on relevance, sensitivity, and independence, supporting researchers in aligning measurement practices with operational goals. This paper aims to inspire the adoption of comprehensive, forward-looking performance measurements that will foster the development of efficient, robust, and sustainable AI-driven DCC systems capable of meeting the needs of next-generation applications. However, there remains significant scope for developing additional metrics; we plan to introduce several other measurements in the near future with empirical evaluations conducted on real-time use cases.

ACKNOWLEDGMENT

Many thanks to *Alaa Saleh*, PhD student at University of Oulu, Finland for helping to draw Figures in this paper.

REFERENCES

- [1] A. Lapkovskis, B. Sedlak, S. Magnússon, S. Dustdar, and P. K. Donta, "Benchmarking dynamic slo compliance in distributed computing continuum systems," *arXiv preprint arXiv:2503.03274*, 2025.
- [2] P. K. Donta, B. Sedlak, I. Murturi, V. Casamayor-Pujol, and S. Dustdar, "Human-based distributed intelligence in computing continuum systems," *IEEE Internet Computing*, vol. 29, no. 2, 09 2025.
- [3] D. Praveen Kumar, M. Ilir, C. P. Victor, S. Boris, and D. Schahram, "Exploring the potential of distributed computing continuum systems," *Computers*, pp. 1–29, 10 2023.
- [4] V. C. Pujol, P. K. Donta, A. Morichetta, I. Murturi, and S. Dustdar, "Edge intelligence—research opportunities for distributed computing continuum systems," *IEEE Internet Computing*, vol. 27, no. 4, pp. 53–74, 2023.
- [5] V. Casamayor Pujol, P. K. Donta, A. Morichetta, I. Murturi, and S. Dustdar, "Distributed computing continuum systems—opportunities and research challenges," in *International Conference on Service-Oriented Computing*. Springer, 2022, pp. 405–407.
- [6] F. Mangione, C. Savaglio, and G. Fortino, "Generative artificial intelligence for internet of things computing: A systematic survey," *arXiv preprint arXiv:2504.07635*, 2025.
- [7] A. Saleh, R. Morabito, S. Dustdar, S. Tarkoma, S. Pirttikangas, and L. Lovén, "Towards message brokers for generative ai: Survey, challenges, and opportunities," *ACM Comput. Surv.*, Jun. 2025, just Accepted. [Online]. Available: <https://doi.org/10.1145/3742891>
- [8] R. Morabito and S. Jang, "Smaller, smarter, closer: The edge of collaborative generative ai," *IEEE Internet Computing*, 2025.
- [9] C. Silvano, D. Ielmini, F. Ferrandi, L. Fiorin, S. Curzel, L. Benini, F. Conti, A. Garofalo, C. Zambelli, E. Calore *et al.*, "A survey on deep learning hardware accelerators for heterogeneous hpc platforms," *ACM Computing Surveys*, vol. 57, no. 11, pp. 1–39, 2025.
- [10] L. T. Ha, "An investigation of dynamic connectedness between robotic, artificial intelligence development, and carbon risk by quantile spillovers," *Clean Technologies and Environmental Policy*, vol. 26, no. 9, pp. 3051–3067, 2024.
- [11] Heidelbergberger and Lavenberg, "Computer performance evaluation methodology," *IEEE Transactions on Computers*, vol. 100, no. 12, pp. 1195–1220, 1984.
- [12] P. Calingaert, "System performance evaluation: survey and appraisal," *Communications of the ACM*, vol. 10, no. 1, pp. 12–18, 1967.
- [13] H. Wang, X. Yuan, K. Zhang, X. Lang, H. Chen, H. Yu, and S. Li, "Performance evaluation and optimization of data center servers using single-phase immersion cooling," *International Journal of Heat and Mass Transfer*, vol. 221, p. 125057, 2024.
- [14] K. Kant, *Introduction to computer system performance evaluation*. International Edition, 1992.
- [15] R. Jain, *The art of computer systems performance analysis*. John Wiley & Sons, 1990.
- [16] D. J. Lilja, *Measuring computer performance: a practitioner's guide*. Cambridge university press, 2005.
- [17] A. Vahdat, X. Ma, and D. Patterson, "New computer evaluation metrics for a changing world," *Communications of the ACM*, vol. 67, no. 10, pp. 31–33, 2024.
- [18] R. Winterton, "Newton's law of cooling," *Contemporary Physics*, vol. 40, no. 3, pp. 205–212, 1999.
- [19] R. Jain, A. Duresi, and G. Babic, "Throughput fairness index: An explanation," in *ATM Forum contribution*, vol. 99, 1999, pp. 1–13.
- [20] P. K. Donta, B. S. P. Rao, T. Amgoth, C. S. R. Annavarapu, and S. Swain, "Data collection and path determination strategies for mobile sink in 3d wsns," *IEEE Sensors Journal*, vol. 20, no. 4, pp. 2224–2233, 2020.
- [21] M. Nauta, J. Trienes, S. Pathak, E. Nguyen, M. Peters, Y. Schmitt, J. Schlöterer, M. Van Keulen, and C. Seifert, "From anecdotal evidence to quantitative evaluation methods: A systematic review on evaluating explainable ai," *ACM Computing Surveys*, vol. 55, no. 13s, pp. 1–42, 2023.
- [22] C. W. Granger, "Investigating causal relations by econometric models and cross-spectral methods," *Econometrica: journal of the Econometric Society*, pp. 424–438, 1969.
- [23] S. Chen, "Data centres will use twice as much energy by 2030-driven by ai," *Nature*, 2025.
- [24] G. M. Amdahl, "Computer architecture and Amdahl's law," *Computer*, vol. 46, no. 12, pp. 38–46, 2013.