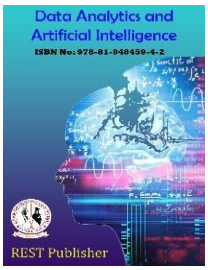
Data Analytics and Artificial Intelligence

Vol: 5(2), 2025

REST Publisher; ISBN: 978-81-948459-4-2

Website: http://restpublisher.com/book-series/daai/

DOI: https://doi.org/10.46632/daai/5/2/11



Direction Dependence Analysis A Framework to Test the Direction of Effects in Linear Models with an Implementation in SPSS

Gokul Gurunathaswamy

Manager, Commercial Technology and Intelligence, IDeaS - a SAS Company.

**Corresponding Author Email: gokulgurun@gmail.com*

Abstract: *High-performance computing (HPC) increasingly depends on distributed memory parallel architectures, such as iWarp cell networks, which combine high processing power with directly accessible memory to deliver substantial computational bandwidth. Optimized parallel computing algorithms are essential, not only for efficiently distributing tasks across multiple nodes but also for minimizing overhead from conventional fault tolerance methods like check pointing and replication. Techniques such as genetic programming and enhancements to High Performance Fortran (HPF) with task parallelism enable effective utilization of existing libraries while facilitating a smooth transition from sequential to parallel execution. Memory and communication latency continue to pose significant challenges in large-scale multiprocessor systems, as processor cycles are often spent on protocol handling and network interface operations rather than actual data transfer. Innovations in processor and network integration have improved performance, though industry trends favor parallel systems built on standard RISC microprocessors. Out-of-core memory systems using solid-state drives (SSDs) offer cost-effective and scalable memory extensions, supporting compute-intensive tasks with near-linear speedup on multicore processors. Frameworks like PJ2 provide robust support for shared memory, message passing, hybrid parallelism, GPU acceleration, and big data processing, allowing fine-grained control over parallel execution, scheduling, and result aggregation. This paper introduces language extensions and a subsystem that automatically converts imperative programs into efficient distributed computations for large clusters, addressing critical HPC challenges such as task parallelism, fault tolerance, memory optimization, and high-speed interprocessor communication.*

Key words: *Parallelism, Scalability, Multicore Processors, Distributed Systems, Message Passing Interface (MPI), Shared Memory Architecture, Load Balancing, Task Scheduling, GPU Acceleration, Fault Tolerance.*

1. INTRODUCTION

An iWarp cell is formed by connecting an iWarp component to local memory, providing up to 64 MB of directly accessible memory. A large network of iWarp cells offers exceptional computing bandwidth in distributed memory parallel systems. Thanks to its strong processing and communication features, along with its commercial accessibility, iWarp is anticipated to play a central role in various high-performance parallel computing systems.[1] Parallel computing algorithms are fundamental to high-performance computing (HPC) systems, as they determine how tasks are allocated among multiple nodes. Conventional fault tolerance (FT) techniques, including check pointing and replication, frequently cause performance slowdowns and bottlenecks (Cappello et al., 2009). Consequently, optimizing parallel computing algorithms is crucial to lower FT overhead while preserving the system's reliability.[2] Genetic programming starts by generating a large, random initial population of programs composed of elements from problem-specific function and terminal sets. Each program is assessed based on fitness, and these fitness values are then used to produce the next generation through genetic operations like reproduction, crossover, and mutation.[3] Enhancing HPF with task parallelism provides practical advantages compared to developing a new language. It allows continued use of existing sequential and data-parallel libraries, facilitates easier conversion of current programs to task and data parallelism, and promotes broader user acceptance. Moreover, it supports efficient compilation using current compiler technologies, with specific directives included to optimize code generation—though some may become unnecessary as compilers evolve. [4]

Copyright@ REST Publisher

The option to switch between various module versions offers a simple method to balance performance and accuracy. Table 1 illustrates the relative system performance across six different cache and network module combinations. The performance difference between the least and most accurate configurations can be up to tenfold. During development, simulations usually exceed one million simulated cycles per second since maximum accuracy is often not essential. At later stages, users can switch to more accurate modules without modifying their code. [5] Memory latency—the duration needed to initiate, process, and fulfill a memory request—has consistently posed a significant challenge in high-performance computing architectures, particularly in large multiprocessor systems. Beyond the actual memory access time, considerable processor cycles are lost as requests traverse distant memory modules. In scalable architectures, communication latency tends to increase with the number of modules because of deeper network structures and can become unpredictable due to network congestion. [6] The naturally high latency of interprocessor communication is often seen as a significant challenge in multiprocessing systems. However, in real-world scenarios—particularly in commercial parallel and distributed systems—communication overhead is largely driven by factors other than transport latency. The majority of processor cycles are consumed by network protocol handling and operating the network interface hardware, while only a small fraction is spent on the actual transmission of data packets between processors.[7] In comparison, research systems have achieved significantly higher communication performance by utilizing innovative processor designs with tightly integrated network interfaces. However, despite the appeal of these specialized architectures, they do not reflect the dominant industry trend, which favors parallel machines constructed from standard superscalar RISC microprocessors.[8] Miscomputing systems support these applications by tackling challenges like resource allocation, fault tolerance, security, and system heterogeneity. Although most of these systems are language-neutral and support multiple programming languages, there is increasing interest in architectures built around a single programming language. This approach may restrict certain applications, but it simplifies system design and ensures consistency by relying on a unified type system.[9] This paper's main contribution is a set of language extensions and a supporting system that explicitly and automatically transforms imperative programs in a general-purpose language into efficient distributed computations suitable for execution on large computer clusters.[10] All four physical cores offer the potential for linear performance improvements in computation-heavy workloads, with algorithms utilizing multiple cores simultaneously considered parallel.[11] Furthermore, solid-state drives (SSDs) provide fast read speeds and low seek latency, making them a practical and affordable way to expand main memory capacity. Systems that efficiently use disk storage as an extension of memory are referred to as out-of-core systems. [12] The four physical cores enable near-linear speedups for compute-intensive tasks when parallel algorithms are employed. Solid-state drives (SSDs), with their high read speeds and low seek latency, serve as affordable and scalable extensions of main memory. Systems that leverage disk storage in this way are commonly referred to as off-the-shelf solutions. [13] PJ2 is a versatile Java library designed for parallel computing on a variety of architectures. It implements shared memory parallelism on multicore systems, message passing on compute clusters, and hybrid models on clusters with multicore nodes. PJ2 also supports GPU acceleration and big data processing using the MapReduce model. Like OpenMP, it provides parallel loops and segments to facilitate parallel code writing. Developers can control how loop iterations are divided between threads using various scheduling strategies: fixed and leapfrog for symmetric tasks, and dynamic, guided, or proportional for non-symmetric workloads. It also provides reduction variables for efficient parallel result aggregation. [14] PJ2 is a versatile Java library designed for parallel computing on a variety of architectures. It implements shared memory parallelism on multicore systems, message passing on compute clusters, and hybrid models on clusters with multicore nodes. PJ2 also supports GPU acceleration and big data processing using the MapReduce model. Like OpenMP, it provides parallel loops and segments to facilitate parallel code writing. [15] Developers can control how loop iterations are divided between threads using various scheduling strategies: fixed and leapfrog for symmetric tasks, and dynamic, guided, or proportional for non-symmetric workloads. It also provides reduction variables for efficient parallel result aggregation.

2. MATERIAL AND METHOD

Input Parameters

Architecture Type: An architectural style denotes a cohesive set of design principles, characteristics, and techniques that define buildings or systems, influenced by historical context, culture, technology, and intended purpose. It encompasses both physical structures (e.g., Gothic, Modernism) and digital systems (e.g., Micro services, N-Tier), focusing on form, materials, construction methods, and organization to achieve functional, aesthetic, or structural goals.

Programming Model: A programming model is a conceptual structure offering abstractions and guidelines for creating and executing software. It connects hardware and programmers by defining rules for computation,

memory management, and interaction, simplifying development. Examples include object-oriented programming and MapReduce, which provide organized methods for problem-solving and code structuring.

Fault Tolerance Method: Fault tolerance is a design approach that allows a system to operate continuously even when some components fail. By implementing redundancy and failover mechanisms—such as backups and replicas—it safeguards business continuity, maintains data integrity, and minimizes downtime for essential applications.

Memory Type: Memory can be understood by considering its duration and the type of information it holds. It primarily consists of sensory memory, which briefly captures sensory input; short-term memory, which temporarily retains information; and long-term memory, which preserves information over extended periods. Long-term memory is divided into explicit (conscious) memory—covering episodic (personal experiences) and semantic (factual knowledge) memories—and implicit (unconscious) memory, which includes procedural memory (skills and routines).

Application Domain: An application domain refers to a particular field, industry, or environment—like healthcare, finance, or e-commerce—where a software system is designed to function. It focuses on addressing real-world problems, establishing the relevant terminology, users, and workflows, and enabling solutions specific to that domain, including the use of domain-specific languages. Technically (for instance, in .NET), an application domain also acts as **an** isolated execution environment, ensuring code runs securely and reliably, much like an operating system process.

Communication Network: A communication network consists of interconnected devices, people, or nodes that share information, facilitating the transfer of data, messages, and resources via established channels and protocols. These networks are vital for organizational operations—such as team coordination—and for global connectivity, including the Internet. They shape the patterns of information exchange, whether formal (like organizational hierarchies) or informal, and utilize wired or wireless links to enable collaboration and seamless connectivity.

Evaluation parameters

Performance Efficiency: Performance efficiency is the capability of a system, organization, or individual to achieve objectives using minimal resources—like time, CPU, memory, or energy—while maintaining responsiveness, effectiveness, and scalability under changing demands. It involves both doing the right things (performance) and doing things efficiently, necessitating continuous monitoring, adjustments, and adaptations to maximize resource use without negatively impacting the user experience.

Fault Tolerance Level: Fault tolerance is a system's capacity to operate correctly despite the failure of certain components. It relies on redundancy and failover strategies—like backups, mirrored data, or spare parts—to ensure high availability and avoid service disruptions. The system can either switch to backup resources seamlessly or reduce performance gradually (graceful degradation) instead of failing completely. Fault tolerance indicates the system's resilience to hardware, software, or network faults and is typically measured by how many failures it can endure without impacting users.

Scalability: Scalability refers to a system, process, or organization's ability to efficiently handle rising workloads or higher demand. It supports growth—by adding resources as necessary—without compromising performance, quality, or consistency across users, data, or markets. Scalable systems can adjust in size smoothly, preventing disruptions and expensive overhauls while maintaining responsiveness and efficiency.

Memory Utilization Efficiency: Memory utilization efficiency measures how well a system or application leverages its allocated memory, usually expressed as the percentage of actively used memory relative to the total available. It helps maintain optimal performance, reduce waste, and avoid bottlenecks, particularly on devices with limited resources. The aim is to maximize productive memory usage—such as during computations—while minimizing idle memory or fragmentation, often accomplished through efficient memory management strategies like garbage collection and the use of smart data structures.

Communication Overhead: Communication overhead is the extra time, data, and resources needed to handle communications—like routing, formatting, or synchronization—rather than executing the main task or transferring essential data. This overhead can lower overall performance in systems, networks, and collaborative projects. It reflects the "cost" of coordination, including tasks such as adding headers, ensuring delivery, and managing protocols, distinct from the actual data payload or hardware delays.

Copyright@ REST Publisher

SPSS METHOD

In the modern era, digital media technologies are driving an unparalleled global shift in industry and technology. The emergence of big data interconnectivity, artificial intelligence, and other sophisticated digital tools is pushing traditional sectors toward deep digitalization and intelligent upgrading. This evolution not only redefines the structure and functionality of products but also introduces heightened expectations for industrial design. [16] Geography is fundamentally an interdisciplinary discipline, and the data it examines frequently extend across multiple fields. The intricate nature of spatial information and geographic phenomena—combined with their inherent spatial interdependencies—creates major challenges for conventional data-mining methods. These complexities underscore the necessity for theoretical progress and the development of novel techniques that can effectively extract valuable insights from extensive and heterogeneous spatial datasets.[17] Research refers to the structured gathering of information through field studies, experimentation, or observation, and the use of this knowledge to address issues across different domains. In the present era, computerization has become integral even to social science research, serving as an indispensable tool for carrying out and organizing research processes.[18] Conventional statistical techniques can assess the magnitude of hypothesized effects, yet they provide little insight into the *direction* of those effects—specifically, whether x influences y or y influences x. The statistical framework outlined in this article allows researchers to infer causal direction between variables. [19] This work centers on observational (non-experimental) data, as such data serve as the main foundation for applying the directional principles presented.[20] Standard data analysis methods—including Pearson correlation, t-tests, ANOVA, MANOVA, and multiple regression—rely on the assumption that observations are independent and randomly sampled from the population of interest. In reality, interdependence is common in both human and animal behaviour, and this pervasive connectedness frequently poses challenges for applying these traditional analytical techniques.[21] Mediation analysis explores the mechanisms that clarify how an exposure variable influences an outcome variable by considering the role of a third factor—the mediator. Instead of positing a solely direct causal link between the independent and dependent variables, a mediation model suggests that the exposure impacts the mediator, which subsequently shapes the outcome.[22] Physicians routinely depend on statistical and epidemiological data in their everyday practice. Golditz and Emerson found that over 40% of the 700 medical articles they examined employed advanced statistical techniques such as regression and MANOVA. Balteil et al. [23] note that contemporary medical decision-making demands strong communication abilities in addition to data-analysis expertise. To build these competencies—and those required for conducting and interpreting clinical research—medical students should be trained in evidence-based medicine (EBM), epidemiology, and biostatistics. [24] Accurate and consistent measurement is crucial in nearly all empirical psychological research. Classical test theory, item response theory, and generalizability theory (G theory) constitute three major and overlapping approaches for evaluating measurement reliability. Although specialists have long advocated greater adoption of item response theory and G theory, classical test theory remains by far the most frequently employed method in empirical studies.[25] Historically, cluster analysis methods, including hierarchical and k-means clustering, have been commonly employed, though they are now increasingly complemented by probabilistic (Bayesian) techniques such as latent class analysis (LCA). Traditional clustering approaches first assess the similarity or dissimilarity between individuals (e.g., using Euclidean distance) and then uncover the underlying subgroup structure by reducing within-group variability while enhancing between-group differences.[26] As China's economy has rapidly developed, the logistics and distribution industry has attracted growing interest from scholars across multiple disciplines. Research has consistently shown that distribution is a critical component of the logistics system. [27] In the fields of education and psychology, measurement reliability has conventionally been evaluated using classical test theory (CTT). CTT posits that an observed score reflects the true score, without isolating an explicit error component. Nevertheless, this assumption has notable limitations, particularly in performance-based assessments where the likelihood of error is considerable. [28] Additionally, when the sample size is large enough, the central limit theorem can often be applied to support the asymptotic normality of the sampling distributions of the relevant statistics. Parametric tests also tend to be robust against departures from normality. These factors collectively justify the use of parametric tests even when the data are not perfectly normally distributed.[29] Surveys are an essential research method applied across fields like social, political, and educational studies, each with specific goals and approaches. [30] They function as instruments for gathering data, offering researchers, businesses, and policymakers valuable insights into human behaviour, opinions, and preferences. In market research, surveys play a crucial role by helping businesses gain a deeper understanding of consumer needs, choices, and market dynamics.27 it is first assumed that the two dichotomous variables under study originate from unobserved continuous variables. The tetra choric correlation estimates the Pearson correlation that would exist between these continuous variables, assuming their joint distribution is bivariate normal. [31]

Copyright@ REST Publisher

3. RESULT AND DISCUSSION

TABLE 1. Reliability Statistics

Reliability Statistics		
Cronbach's Alpha	Cronbach's Alpha Based on Standardized Items	N of Items
.979	.987	5

Table 1 Reliability testing shows exceptionally strong internal consistency, highlighting the impressive stability of the measurement scale. A Cronbach's alpha of .979—rising to .987 with standardized items—indicates that all five items align extremely well, demonstrating the instrument's high reliability and precision in measuring the targeted construct.

TABLE 2. Reliability Statistic individual

	Cronbach's Alpha if Item Deleted
Performance Efficiency	.975
Fault Tolerance Level	.974
Scalability	.982
Memory Utilization Efficiency	.969
Communication Overhead	.969

Table 2 Impulsive behavior involves acting without adequate self-control or moderation, often resulting in hasty decisions, strong emotional reactions, or uncontrolled actions. These tendencies can hinder performance, strain relationships, and weaken one's credibility in important situations. Becoming aware of and regulating impulsive reactions supports balance, stability, and reliable performance in both personal and professional settings.

TABLE 3. Descriptive Statistics

Descriptive Statistics											
	N	Range	Minimum	Maximum	Sum	Mean		Std. Deviation	Variance	Kurtosis	
	Statistic	Statistic	Statistic	Statistic	Statistic	Statistic	Std. Error	Statistic	Statistic	Statistic	Std. Error
Performance Efficiency	6	4.79	.78	5.57	11.81	1.9683	.73757	1.80668	3.264	5.039	1.741
Fault Tolerance Level	6	3	1	4	12	1.98	.454	1.111	1.234	2.363	1.741
Scalability	6	3.40	1.23	4.63	15.09	2.5150	.56854	1.39263	1.939	-1.291	1.741
Memory Utilization Efficiency	6	4.34	.98	5.32	12.74	2.1240	.65738	1.61025	2.593	4.886	1.741
Communication Overhead	6	3.36	1.21	4.57	13.42	2.2367	.51214	1.25449	1.574	2.679	1.741
Valid N (list wise)	6										

Table 3 Impulsive behavior signifies insufficient restraint or moderation, often resulting in exaggerated reactions, inconsistent decisions, and erratic performance. These tendencies can hinder collaboration, weaken trust, and generate avoidable conflict. Developing self-control and managing intense responses supports steadiness, professionalism, and reliable performance, especially in challenging or high-pressure environments.

Copyright@ REST Publisher

TABLE 4. Frequencies Statistics

Statistics						
		Performance Efficiency	Fault Tolerance Level	Scalability	Memory Utilization Efficiency	Communication Overhead
N	6	6	6	6	6	6
	0	0	0	0	0	0
Mean		3.83	4.17	4.00	3.83	4.50
Median		1.3250	1.90	2.2200	1.5900	1.8500
Mode		.78[a]	1[a]	1.23[a]	.98[a]	1.21[a]
Std. Deviation		1.80668	1.111	1.39263	1.61025	1.25449
Percentiles	25	.8935	.99	1.3200	1.1385	1.3075
	50	1.3250	1.90	2.2200	1.5900	1.8500
	75	4.25	5.00	5.00	4.25	5.00
a. Multiple modes exist. The smallest value is shown						

Table 4 Impulsive behavior indicates limited self-regulation, frequently leading to exaggerated responses, inconsistent decisions, and heightened tension in challenging situations. These actions can weaken cooperation, diminish trust, and negatively affect overall performance. Developing emotional steadiness and practicing restraint support clearer judgment, stronger teamwork, and more reliable outcomes in both professional and high-pressure contexts.

HISTOGRAM:



FIGURE 1. Performance Efficiency

Figure 1 the distribution of performance efficiency reveals marked instability, demonstrated by noticeable fluctuations and unpredictable variations across observations. These inconsistent patterns suggest a system with limited stability and inadequate control. Such variability lowers predictability, undermines reliability, and emphasizes the need for stronger optimization measures to ensure steady, balanced, and dependable computational performance.

Copyright@ REST Publisher

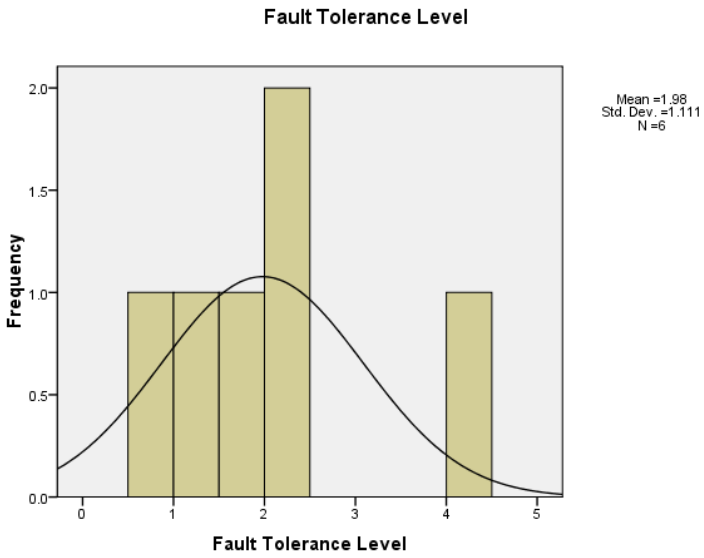


FIGURE 2. Fault Tolerance Level

Figure 2 the fault-tolerance levels exhibit considerable variation, indicating system instability across different conditions. The distribution reflects irregular performance, with some values concentrated at the lower end and others rising unexpectedly. This variability suggests inadequate resilience, underscoring the need for stronger mechanisms to sustain reliability, minimize operational risks, and support consistent performance in high-demand computing environments.

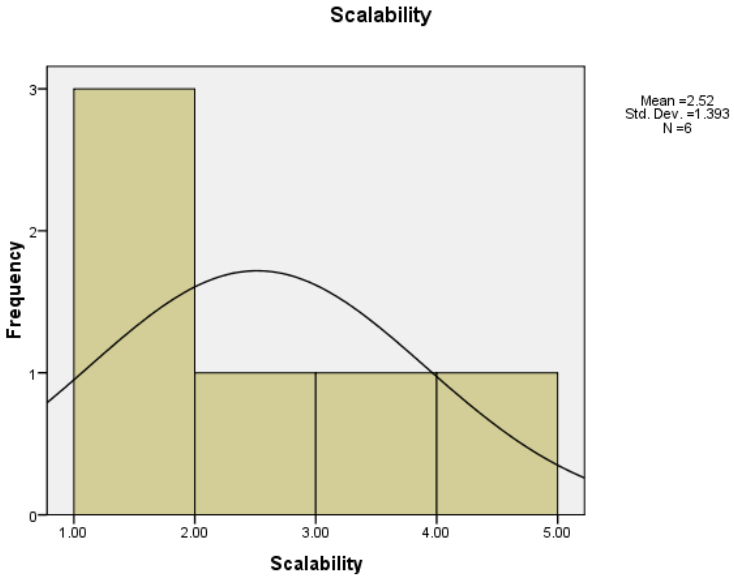


FIGURE 3. Scalability

Figure 3 the scalability distribution exhibits considerable irregularity, evidenced by unpredictable frequency patterns and substantial variation across values. This instability reflects inconsistent system scaling, suggesting that performance does not increase reliably with higher workloads. Such variability underscores a lack of adaptability and emphasizes the need for robust optimization to ensure stable and dependable scaling in demanding computing environments.

Copyright@ REST Publisher

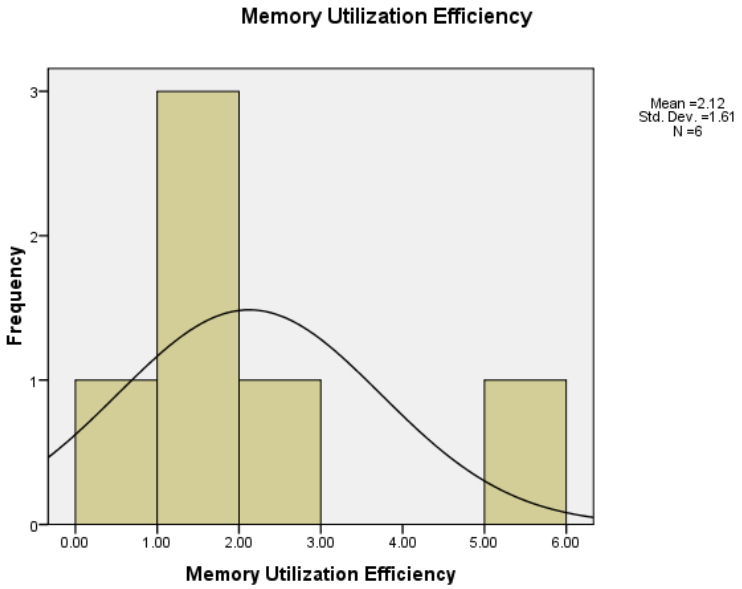


FIGURE 4. Memory Utilization Efficiency

Figure 4 the distribution of memory utilization efficiency demonstrates significant irregularity, with dispersed frequency patterns and unpredictable fluctuations. This variability reflects inefficient or unstable memory usage, compromising system reliability under load. Such inconsistency underscores the importance of robust memory management and optimization strategies to achieve stable, consistent, and high-performing computing environments.

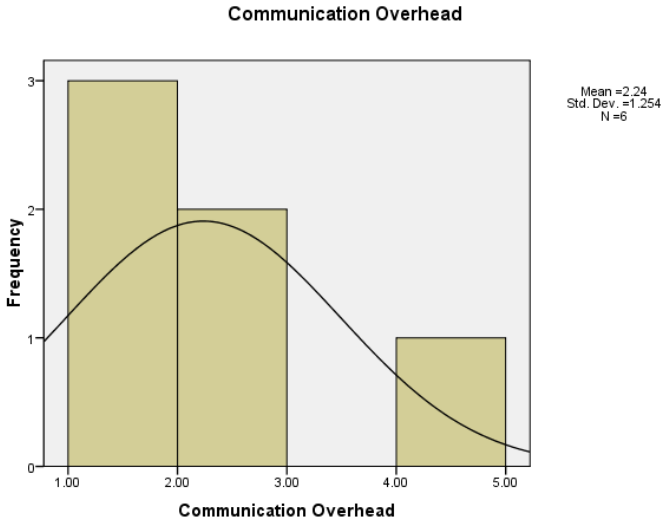


FIGURE 5. Communication Overhead

Figure 5 the communication overhead distribution exhibits considerable irregularity, characterized by uneven frequency patterns and unpredictable fluctuations. This variability reflects inefficiencies in data transfer and network management, potentially impairing overall system performance. Such inconsistency emphasizes the need for optimized communication protocols and strategies to ensure stable, reliable, and high-performing operations in parallel computing environments.

Copyright@ REST Publisher

TABLE 5. Correlations

Correlations					
	Performance Efficiency	Fault Tolerance Level	Scalability	Memory Utilization Efficiency	Communication Overhead
Performance Efficiency	1	.958**	.844*	.999**	.977**
Fault Tolerance Level	.958**	1	.923**	.964**	.978**
Scalability	.844*	.923**	1	.844*	.921**
Memory Utilization Efficiency	.999**	.964**	.844*	1	.979**
Communication Overhead	.977**	.978**	.921**	.979**	1
*. Correlation is significant at the 0.05 level (2-tailed).					

Figure 5 the variables in this high-performance computing system show very strong correlations, with many coefficients above .95. This tight interdependence suggests that performance, fault tolerance, scalability, memory utilization, and communication overhead are closely linked. While this reflects system coherence, it also highlights potential vulnerabilities, underscoring the need for balanced optimization across all metrics.

4. CONCLUSION

This study underscores the evolving landscape of high-performance computing (HPC) and the widespread adoption of digital technologies across industrial and research domains. Distributed memory parallel architectures, such as iWarp cell networks, offer remarkable computational bandwidth, support efficient task distribution, and reduce overhead associated with traditional fault-tolerant methods. Developments in parallel computing algorithms, task parallelism in High-Performance Fortran (HPF), and genetic programming enable smooth transitions from sequential to parallel execution while enhancing overall system performance. Key challenges, including memory and communication latency, are addressed through innovations in processor-network integration, off-chip memory systems utilizing SSDs, and multicore parallelism, providing scalable and cost-effective solutions. Frameworks like PJ2 further facilitate flexible parallel computing across diverse architectures, supporting shared memory, message passing, GPU acceleration, and big data processing. Beyond HPC, computational and statistical tools play an increasingly critical role in research, education, and industry. Techniques such as mediation analysis, cluster analysis, tetrachoric correlation, and parametric testing improve the reliability and rigor of data-driven insights. Surveys, statistical modelling, and evidence-based training enhance decision-making and operational efficiency across fields including medicine, psychology, geography, and logistics. The integration of HPC, advanced algorithms, and robust data analytics frameworks addresses core challenges in computational efficiency, system reliability, and actionable insights, highlighting the synergy of technological innovation and methodological rigor in advancing research and practical applications across diverse sectors.

REFERENCES

[1]. Borkar, Shekhar, Robert Cohn, George Cox, Sha Gleason, Thomas Gross, Hsing-Tsung Kung, Monica Lam et al. "Warp: an integrated solution of high-speed parallel computing." In Conference on High Performance Networking and Computing: Proceedings of the 1988 ACM/IEEE conference on Supercomputing, vol. 12, no. 17, pp. 330-339. 1988.

[2]. Migdalas, Athanasios, Gerardo Toraldo, and Vipin Kumar. "Nonlinear optimization and parallel computing." Parallel Computing 29, no. 4 (2003): 375-391.

[3]. Stoffel, Kilian, and Lee Spector. "High-performance, parallel, stack-based genetic programming." Genetic Programming (1996): 224-229.

[4]. Gross, Thomas, David R. O'Hallaron, and Jaspal Subhlok. "Task parallelism in a High Performance Fortran framework." IEEE Parallel and Distributed Technology 2, no. 3 (1994): 16-26.

[5]. Brewer, Eric A., Chrysanthos N. Dellarocas, Adrian Colbrook, and William E. Weihl. "Proteus: A high-performance parallel-architecture simulator." Acm Sigmetrics Performance Evaluation Review 20, no. 1 (1992): 247-248.

[6]. Saavedra-Barrera, Rafael H., David E. CulIer, and Thorsten von Eicken. "Analysis of Multithreaded Architectures for Parallel."

[7]. Coddington, Paul D. "Random number generators for parallel computers." (1997).
[8]. Papadopoulos, Gregory M., G. Andrew Boughton, Robert Greiner, and Michael J. Beckerle. "T: Integrated building blocks for parallel computing." In Proceedings of the 1993 ACM/IEEE conference on Supercomputing, pp. 624-635. 1993.
[9]. Henri, Rob van Nieuwpoort Jason Maassen, E. Bal, and Thilo Kielmann Ronald Veldema. "Wide-area parallel computing in Java."
[10]. Fetterly, Yuan Yu Michael Isard Dennis, Mihai Budiu, Úlfar Erlingsson, and Pradeep Kumar Gunda Jon Currey. "DryadLINQ: A system for general-purpose distributed data-parallel computing using a high-level language." Proc. LSDS-IR 8 (2009).
[11]. Nickolls, John R. "The design of the MasPar MP-1: a cost effective massively parallel computer." In Compcon, pp. 25-28. 1990.
[12]. Khoshlessan, Mahzad, Iowan's Paraskevakos, Shantenu Jha, and Oliver Beckstein. "Parallel Analysis in MDAnalysis using the Dask Parallel Computing Library." In SciPy, pp. 64-72. 2017.
[13]. Nagendra, M., and M. Chandra Sekhar. "Performance improvement of advanced encryption algorithm using parallel computation." International Journal of Software Engineering and Its Applications 8, no. 2 (2014): 287-296.
[14]. Kaminsky, Alan. "Parallel Java Library." In Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, New Orleans, LA, USA, pp. 16-21. 2014.
[15]. Yu, Chengzhi, and Yiming Tang. "Exploration of design driven innovative education model based on SPSS analysis: Taking industrial design as an example." In Proceedings of the 2025 2nd International Conference on Informatics Education and Computer Technology Applications, pp. 279-286. 2025.
[16]. Devi, B. Renuka, K. Rao, S. Setty, and M. Rao. "Disaster prediction system using IBM SPSS data mining tool." International Journal of Engineering Trends and Technology (IJETT) 4, no. 8 (2013).
[17]. Abu-Bader, Soleman H. Using statistical methods in social science research: With a complete SPSS guide. Oxford University Press, 2021.
[18]. Wiedermann, Wolfgang, and Xintong Li. "Direction dependence analysis: A framework to test the direction of effects in linear models with an implementation in SPSS." Behavior Research Methods 50, no. 4 (2018): 1581-1601.
[19]. Nimon, Kim. "Regression commonality analysis: Demonstration of an SPSS solution." Multiple Linear Regression Viewpoints 36, no. 1 (2010): 10-17.
[20]. O'Connor, Brian P. "SPSS and SAS programs for addressing interdependence and basic levels-of-analysis issues in psychological data." Behavior Research Methods, Instruments, & Computers 36, no. 1 (2004): 17-28.
[21]. Valeri, Linda, and Tyler J. VanderWeele. "Mediation analysis allowing for exposure–mediator interactions and causal interpretation: theoretical assumptions and implementation with SAS and SPSS macros." Psychological methods 18, no. 2 (2013): 137.
[22]. Perry, Zvi H., Aricha-Tamir Barak, Lily Neumann, and Amalia Levy. "Computer-Based Learning: The Use of SPSS Statistical Program for Improving Biostatistical Competence of Medical Students." Journal of Biomedical Education 2014, no. 1 (2014): 298140.
[23]. Godfrey, A. Jonathan R., and M. Theodor Loots. "Statistical software (R, SAS, SPSS, and Minitab) for blind students and practitioners." Journal of Statistical Software 58 (2014): 1-25.
[24]. Gelbard, Roy, Orit Goldman, and Israel Spiegler. "Investigating diversity of clustering methods: An empirical comparison." Data & Knowledge Engineering 63, no. 1 (2007): 155-166.
[25]. Zhang, Zipeng. "A principal component analysis using SPSS for multi-objective decision location allocation problem." WSEAS Trans Computers 14 (2015): 698-76.
[26]. Pelz, Corey J., and John P. Klein. "Analysis of survival data: A comparison of three major statistical packages (SAS, SPSS, BMDP)." Unpublished manuscript retrieved from: http://www. mcw. Edu/FileLibrary/Groups/Biostatistics/TechReports/tr017. Pdf (1996).
[27]. Zajić, Jelena S. Osmanović, and Jelena Ž. Maksimović. "The efficiency of the application of SPSS in higher education teaching: An experimental study." Proceedings of CBU in Social Sciences 2 (2021): 273-278.
[28]. Rodgers, Paul. "Multi-Response permutation procedures." Computer and Quantitative Methods in Archaeology 7957 (1987): 45-54.

Copyright@ REST Publisher