International Journal of Computational and Experimental Science and ENgineering
(IJCESEN)

Vol. 11-No.3 (2025) pp. 6818-6825
http://www.ijcesen.com

ISSN: 2149-9144

Research Article

Fine-Grained Scaling in Stream Processing Systems: Hybrid CPU-Memory Autoscaling with Graph Neural Networks

Venkata Chandra Sekhar Sastry Chilkuri*

Independent researcher-USA
* **Corresponding Author Email:** venkatachandrach@gmail.com- **ORCID:** 0000-0002-5247-7855

Article Info:

DOI: 10.22399/ijcesen.3918
Received : 05 July 2025
Accepted : 29 August 2025

Keywords

Stream Processing,
Hybrid Autoscaling,
Graph Neural Networks,
Resource Optimization,
Cloud-Native Computing

Abstract:

Stream-processing engines in distributed environments require dynamic resource allocation, though traditional autoscaling treats CPU and memory as coupled units. Recent technological developments showcase disaggregated resource management that allows operator-specific allocation based on runtime computational demands. The Justin framework for Apache Flink demonstrates hybrid CPU/memory scaling through precise resource distribution matching individual operator requirements during execution. StreamTune represents another advancement, employing Graph Neural Networks trained on execution histories to optimize operator parallelism and identify bottlenecks dynamically. These innovations achieve resource consumption reductions while preserving throughput, crucial for cloud-native deployments requiring cost optimization. Such developments prove essential when building large-scale platforms handling millions of events per second with minimal latency and maximum resource efficiency. Graph Neural Network integration enables predictive scaling through pattern recognition from execution data, shifting from reactive to proactive resource management paradigms. Machine learning convergence with distributed stream processing opens pathways for intelligent infrastructure adapting to workload variations while maintaining performance guarantees. Production environments reveal practical importance as operational expenses maintain direct relationships with resource utilization performance. These innovations create the groundwork for autonomous systems to achieve self-optimization via real-time performance indicators and predictive computational models.

1. Introduction

Stream processing systems in distributed computing face increasing demands for efficient resource management while maintaining consistent performance across varying workload conditions. Traditional autoscaling approaches typically couple CPU and memory resources as unified allocation units, creating suboptimal resource distribution when streaming applications exhibit diverse computational requirements across different operators [1]. As organizations handle growing volumes of real-time data, the need for adaptive resource management strategies becomes critical for maintaining system reliability and cost effectiveness in production environments [9].

Streaming topologies present unique resource allocation challenges where individual operators demonstrate distinct computational and memory consumption patterns. Operators handling extensive state repositories need considerable memory resources, while processing-intensive transformation phases require substantial computational capacity. Traditional scaling systems lack the ability to differentiate between these varying resource demands, resulting in either inefficient over-allocation or performance-constraining under-provisioning. The gap between uniform scaling policies and heterogeneous operator needs highlights fundamental limitations in current resource management approaches.Recent developments in machine learning offer new possibilities for intelligent resource allocation that move beyond static threshold-based policies. Hybrid CPU-memory autoscaling enables independent scaling of processing and memory resources based on actual operator behavior rather than predetermined assumptions. This separation allows systems to allocate resources more precisely according to specific computational demands, potentially reducing waste while improving overall system efficiency.Graph Neural Networks present natural modeling approaches for stream processing topologies by representing operators as nodes and data flows as edges within computational graphs. This representation captures complex interdependencies between operators that influence resource allocation decisions. Machine learning models trained on historical execution data can identify patterns and predict future resource needs, enabling proactive scaling before performance issues occur.Cloud environments emphasize the importance of efficient resource utilization due to direct cost implications of resource consumption. Organizations handling substantial event throughput demand require scaling solutions that adapt rapidly to workload fluctuations while preserving stringent performance standards. The capacity to forecast resource demands and distribute them accurately becomes vital for managing operational costs without compromising service delivery. The progression toward anticipatory resource coordination signifies a transition from reactive threshold-based monitoring to proactive demand prediction. This evolution allows streaming platforms to foresee scaling needs and modify resources correspondingly, sustaining reliable performance while enhancing utilization effectiveness. Incorporation of machine learning methods with distributed stream processing generates possibilities for self-adjusting infrastructure that responds to evolving conditions with reduced manual oversight, enabling more dependable and economical streaming solutions.

2. Background and Related Work

Stream processing frameworks traditionally implement scaling strategies that treat computational resources as uniform entities across entire cluster deployments. This approach assumes similar resource consumption characteristics among all topology operators, yet practical streaming applications exhibit significant variations in processing requirements between different computational stages [4]. Early distributed platforms employed centralized monitoring that tracked aggregate metrics without differentiating individual operator needs, often resulting in either excessive provisioning or localized resource constraints within processing pipelines. Container orchestration platforms introduced improved resource coordination capabilities but remained limited by an insufficient understanding of streaming-specific operational requirements. Standard scaling policies use threshold-based triggers that activate resource changes after performance degradation begins, creating delays between workload shifts and system adjustments [8]. These reactive mechanisms prove particularly problematic in real-time processing scenarios where maintaining low latency requires anticipatory resource provisioning to prevent bottlenecks.Hierarchical scaling approaches emerged as intermediate solutions balancing allocation efficiency with operational complexity through layered policy implementation across system components. However, these frameworks still operate under assumptions linking processor and memory resources as coupled units, limiting optimization opportunities that could improve utilization effectiveness [8]. The continued treatment of computational and memory resources as inseparable components restricts scaling mechanism performance in modern streaming deployments.Machine learning integration has created new opportunities for intelligent resource coordination based on historical execution pattern recognition and future demand prediction. Graph-based modeling techniques offer intuitive representations of streaming topologies, enabling optimization algorithms that consider complex operator interdependencies when making scaling decisions. These developments represent shifts toward anticipatory resource management capable of predicting scaling needs before performance impacts occur.Neural network applications in distributed system optimization have shown promising results across various computational domains, suggesting potential for streaming environment deployment. Machine learning approaches can identify complex relationships between system performance indicators and resource allocation patterns that traditional rule-based methods cannot effectively capture. This capability becomes valuable when managing dynamic workloads with temporal variations and cyclical patterns requiring adaptive scaling strategies.The integration of predictive analytics with distributed processing represents an advancement in autonomous system management, enabling infrastructure components to optimize themselves through continuous learning from operational data. These technological developments establish foundations for intelligent scaling platforms that adapt to changing computational demands while maintaining performance guarantees essential for production deployments.Contemporary scaling mechanisms must address the fundamental mismatch between uniform resource allocation policies and heterogeneous operator requirements in streaming applications. The transition toward machine learning-driven resource management offers possibilities for more precise and efficient allocation strategies that can improve both performance and cost effectiveness in distributed stream processing environments.

3. Hybrid CPU-Memory Autoscaling Architecture

Disaggregated resource management fundamentally transforms how streaming systems approach computational and memory allocation by treating these components as independent, scalable entities rather than coupled units. This architectural shift enables precise resource distribution based on individual operator characteristics and runtime behavior patterns [1]. Traditional systems assume proportional scaling between processing power and memory capacity, yet streaming workloads demonstrate distinct resource consumption profiles where certain operators require memory-intensive operations while others demand computational acceleration without corresponding memory increases.The architectural foundation relies on continuous operator profiling that monitors resource utilization patterns across different workload conditions and execution phases. Each operator within the streaming topology maintains detailed performance metrics, including memory access patterns, computational complexity indicators, and state management requirements [2]. This granular monitoring enables the system to build comprehensive profiles of operator behavior that inform intelligent scaling decisions based on actual resource consumption rather than predetermined assumptions about uniform requirements.Resource allocation mechanisms operate through independent scaling controllers that manage CPU and memory provisioning separately according to operator-specific demands. Memory-intensive operators such as windowing functions and stateful aggregations receive targeted memory scaling without unnecessary computational resource increases. Conversely, mathematically complex transformation operators obtain additional processing capacity without memory allocation overhead. This separation eliminates the waste associated with uniform scaling while ensuring adequate resources for diverse operational requirements.Dynamic resource modification happens through continuous monitoring of operator performance metrics and instant allocation adjustments when resource limitations or surplus capacity are identified. The system constantly assesses operator throughput, latency properties, and resource utilization effectiveness to establish optimal allocation thresholds [1]. These modifications occur seamlessly without interrupting processing assurances or demanding manual oversight, facilitating smooth adaptation to shifting workload circumstances. Coordination with container orchestration platforms allows precise resource management at the infrastructure layer while preserving compatibility with current deployment architectures.The hybrid scaling architecture leverages container resource limits and requests to implement precise CPU and memory allocation according to operator requirements [2]. This integration ensures that scaling decisions translate effectively into actual resource availability while respecting infrastructure constraints and deployment policies.State management during scaling operations presents unique challenges that require sophisticated coordination between resource allocation and data consistency maintenance. The architecture implements checkpointing mechanisms that preserve operator state during resource modifications, ensuring processing continuity and exactly-once delivery guarantees. Memory scaling operations coordinate with state backend systems to migrate or redistribute state data according to new memory allocations without losing critical information.Performance optimization through hybrid scaling demonstrates significant improvements in resource utilization efficiency compared to traditional uniform allocation approaches. The architecture enables systems to maintain consistent performance levels while reducing overall resource consumption by eliminating unnecessary over-provisioning. This optimization becomes particularly valuable in cloud environments where resource costs directly impact operational expenses, and efficient allocation strategies provide competitive advantages through reduced infrastructure spending while maintaining service quality standards.

4. Graph Neural Network-Based Optimization

Graph Neural Networks provide natural modeling frameworks for stream processing topologies by representing operators as computational nodes and data flow connections as edges within structured graphs that capture complex interdependencies [3]. This representation enables sophisticated optimization algorithms that consider operator relationships, performance characteristics, and resource requirements when making scaling decisions. The graph structure naturally encodes the topology's computational dependencies, allowing machine learning models to understand how changes in one operator's resource allocation might affect downstream components.

Feature extraction processes collect comprehensive operator performance data, including processing latency distributions, throughput measurements, memory usage patterns, and state size growth characteristics over time. These features serve as input vectors for neural network training, capturing both individual operator behavior and system-wide performance indicators [7]. The feature engineering process transforms raw performance metrics into meaningful representations that highlight critical scaling decision factors such as bottleneck indicators, resource contention signals, and workload variation patterns.Neural network architecture incorporates attention mechanisms that identify critical performance bottlenecks and resource allocation opportunities within the streaming topology. The model learns to focus on operators that most significantly impact overall system performance and prioritize scaling decisions accordingly [3]. Attention weights help the system understand which operators require immediate resource adjustments and which can tolerate temporary resource constraints without affecting end-to-end processing guarantees.Training methodologies utilize historical execution traces spanning diverse workload scenarios and operational conditions to build robust predictive models. The training dataset includes operator performance metrics, resource allocation decisions, and resulting system outcomes across different traffic patterns, seasonal variations, and fault conditions [7]. This comprehensive training enables the model to generalize across various operational scenarios and make accurate predictions for previously unseen workload patterns.Prediction generation occurs in real-time as the system continuously processes incoming performance data and generates resource allocation recommendations for each operator in the topology. The model outputs probabilistic predictions of future resource requirements based on current system state and historical patterns, enabling proactive scaling before performance degradation occurs [3]. These predictions include confidence intervals that help the system balance between aggressive optimization and conservative resource provisioning to maintain reliability.Decision optimization algorithms translate neural network predictions into concrete resource allocation actions while respecting system constraints and operational policies. The optimization process considers factors such as resource availability, scaling velocity limits, and cost constraints when determining final allocation decisions [7]. This conversion guarantees that machine learning findings translate into actionable scaling operations that enhance system performance within operational constraints. Adaptive learning frameworks allow the system to modify its predictive models according to observed scaling results and evolving workload patterns. The neural network constantly integrates fresh performance information and scaling outcomes to enhance its comprehension of operator behavior and resource demands. This continuous learning capability enables the system to strengthen prediction precision progressively and adjust to changing application behaviors, infrastructure modifications, and operational demands that might not have existed in the original training datasets.

5. Performance Evaluation and Cost Analysis

Experimental evaluation of hybrid CPU-memory autoscaling demonstrates substantial improvements in resource utilization efficiency across diverse streaming workloads when compared to traditional uniform scaling approaches. Benchmark testing using representative event processing scenarios reveals consistent resource consumption reductions while maintaining comparable throughput and latency characteristics [6]. The evaluation methodology incorporates various workload patterns, including bursty traffic, seasonal variations, and sustained high-volume processing to assess scaling algorithm robustness under realistic operational conditions.Resource utilization measurements indicate significant efficiency gains through disaggregated scaling, particularly for workloads containing operators with heterogeneous resource requirements. Memory-intensive operators such as large window aggregations benefit from targeted memory scaling without unnecessary CPU provisioning, while compute-heavy transformation stages receive appropriate processing power without memory overhead [8]. These targeted allocations eliminate waste associated with uniform scaling policies that provision resources according to the most demanding operator requirements across the entire topology.Latency analysis demonstrates that hybrid scaling maintains consistent processing delays while optimizing resource allocation. The overhead introduced by dynamic resource adjustment operations remains minimal, typically adding negligible latency to end-to-end processing times [6]. Performance measurements show that scaling decisions occur rapidly enough to prevent bottleneck formation while avoiding excessive resource provisioning that could increase operational costs without corresponding performance benefits.Cost evaluation in cloud environments reveals significant economic advantages through precise resource allocation that aligns expenses with actual computational demands. The capacity to scale CPU and memory separately allows more precise cost forecasting and budget management compared to conventional methods that over-allocate resources to handle peak-demand situations [8]. These cost savings become especially significant for extended-duration streaming applications where minor efficiency enhancements accumulate into considerable financial benefits over prolonged operational timeframes.Throughput stability measurements confirm that hybrid scaling preserves processing guarantees while optimizing resource consumption. The evaluation demonstrates that systems can maintain required event processing rates with reduced resource footprints through intelligent allocation based on operator-specific requirements [6]. This stability proves essential for production deployments where consistent performance must be maintained regardless of underlying resource allocation changes.Edge computing deployment scenarios present additional evaluation dimensions where resource constraints amplify the benefits of efficient allocation strategies. Testing in resource-limited edge environments demonstrates that hybrid scaling enables deployment of streaming applications that would otherwise exceed available capacity with traditional scaling approaches [6]. These results highlight the particular value of disaggregated scaling in environments where resource efficiency directly impacts deployment feasibility and operational sustainability.Performance benchmarking across different infrastructure configurations confirms that hybrid scaling benefits translate across diverse deployment scenarios, including on-premises clusters, public cloud instances, and hybrid environments. The consistent performance improvements across various infrastructure types validate the general applicability of disaggregated resource management for streaming systems regardless of underlying hardware characteristics or deployment models [8].

6. Implementation Challenges and Future Directions

Production deployment of hybrid CPU-memory autoscaling presents several technical complexities that require careful consideration in enterprise environments [2]. Container orchestration platforms must provide fine-grained resource allocation capabilities, including fractional CPU assignments and dynamic memory limits, which may not be universally supported across all infrastructure providers. Integration with existing monitoring and deployment pipelines requires substantial system modifications and operational procedure updates that can introduce temporary instabilities during transition periods [9].State management during resource reallocation operations demands sophisticated coordination mechanisms that preserve data consistency while enabling seamless scaling transitions [2]. The system must implement advanced checkpointing strategies that minimize service disruption during scaling events while maintaining exactly-once processing guarantees essential for many streaming applications. Memory scaling operations particularly challenge state preservation as operator memory allocation changes require careful state migration without data loss or processing interruption [9].Machine learning model integration introduces additional operational complexity related to model versioning, training data management, and fallback mechanisms when predictions prove inaccurate [2]. Production systems require robust monitoring of model performance and automated retraining capabilities to maintain prediction accuracy as workload patterns evolve. The integration must provide graceful degradation to traditional scaling mechanisms when machine learning components experience failures or produce unreliable predictions [9].Cross-platform compatibility remains a significant challenge as different infrastructure providers implement varying levels of support for fine-grained resource controls [2]. The scaling architecture must accommodate diverse container runtimes, orchestration frameworks, and cloud service interfaces while maintaining consistent behavior across deployment environments. This compatibility requirement often necessitates platform-specific adaptations that increase implementation complexity and maintenance overhead [9].Future advancements will probably concentrate on federated learning methodologies that allow multi-tenant streaming platforms to gain from collective optimization insights while preserving tenant separation and privacy standards [2]. Reinforcement learning approaches provide encouraging pathways for adaptive policy optimization that can modify scaling strategies based on observed results rather than depending exclusively on supervised learning from historical information. These approaches could enable more sophisticated optimization that balances multiple objectives, including performance, cost, and resource efficiency [9].Edge computing integration represents an emerging area where hybrid scaling principles could provide significant value in resource-constrained environments [2]. The extension of these techniques to edge-cloud hybrid deployments requires addressing additional challenges related to network connectivity, resource heterogeneity, and distributed coordination across geographically dispersed infrastructure components. Research into hardware-accelerated optimization engines could reduce decision latency and enable real-time scaling responses essential for modern streaming applications [9].

***Table 1**: Traditional vs Hybrid Scaling Comparison [1,2]*

Traditional Scaling Approach	Hybrid CPU-Memory Scaling Approach
Unified resource allocation	Disaggregated resource management
Coarse-grained scaling policies	Fine-grained operator-level scaling
Reactive threshold-based triggers	Proactive predictive scaling
Manual configuration requirements	Automated intelligent allocation
Limited operator differentiation	Operator-specific resource profiling
Static resource provisioning	Dynamic resource adjustment
High resource waste overhead	Optimized utilization efficiency
Single scaling dimension	Multi-dimensional resource control

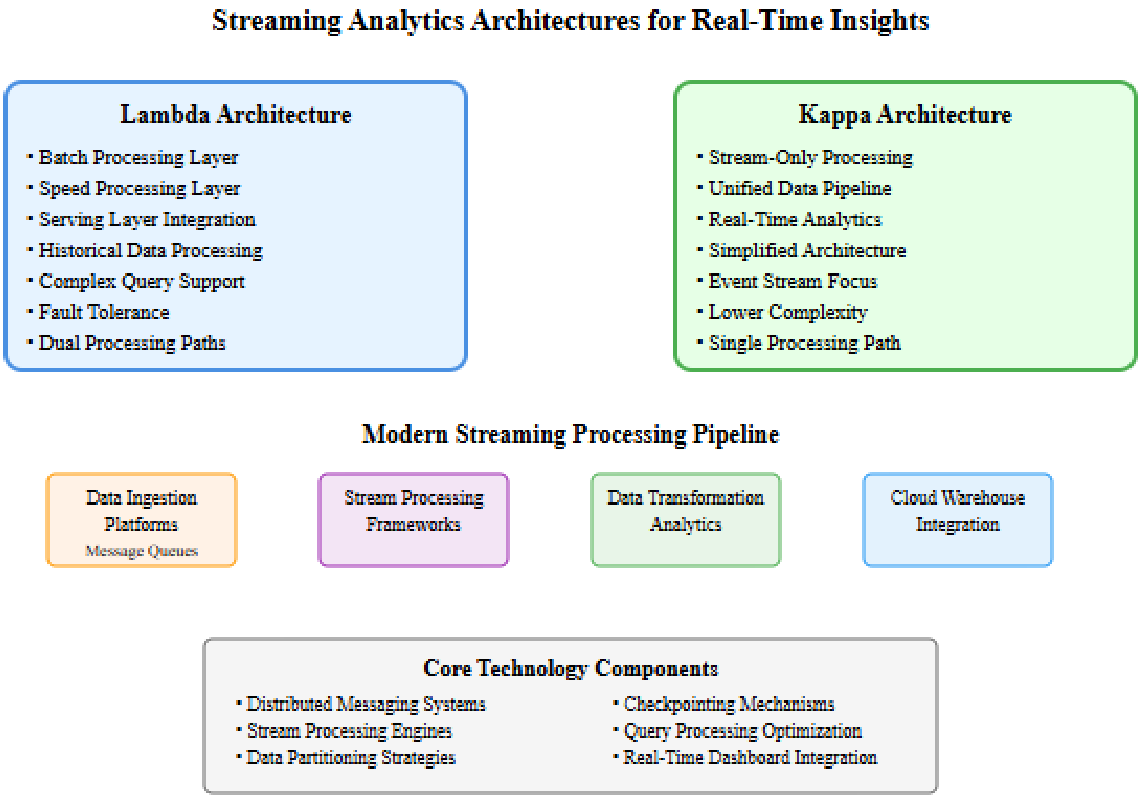
Streaming Analytics Architectures for Real-Time Insights
Lambda Architecture
• Batch Processing Layer
• Speed Processing Layer
• Serving Layer Integration
• Historical Data Processing
• Complex Query Support
• Fault Tolerance
• Dual Processing Paths
Kappa Architecture
• Stream-Only Processing
• Unified Data Pipeline
• Real-Time Analytics
• Simplified Architecture
• Event Stream Focus
• Lower Complexity
• Single Processing Path
Modern Streaming Processing Pipeline
Data Ingestion Platforms
Message Queues
Stream Processing Frameworks
Data Transformation Analytics
Cloud Warehouse Integration
Core Technology Components
• Distributed Messaging Systems
• Stream Processing Engines
• Data Partitioning Strategies
• Checkpointing Mechanisms
• Query Processing Optimization
• Real-Time Dashboard Integration

***Figure 1:** Streaming Analytics Architectural Paradigms and Processing Pipeline [2,5]*

***Table 2:** Implementation Framework Characteristics [6,8]*

Framework Feature	Implementation Aspect
Edge processing support	Distributed deployment capabilities
Autoscaling mechanisms	Dynamic resource allocation methods
Heterogeneous resources	Multi-platform compatibility
Runtime adaptation	Live system modification support
Policy configuration	Hierarchical scaling rule management
Performance monitoring	Real-time metrics collection
Cost optimization	Resource efficiency maximization
Fault tolerance	System resilience mechanisms

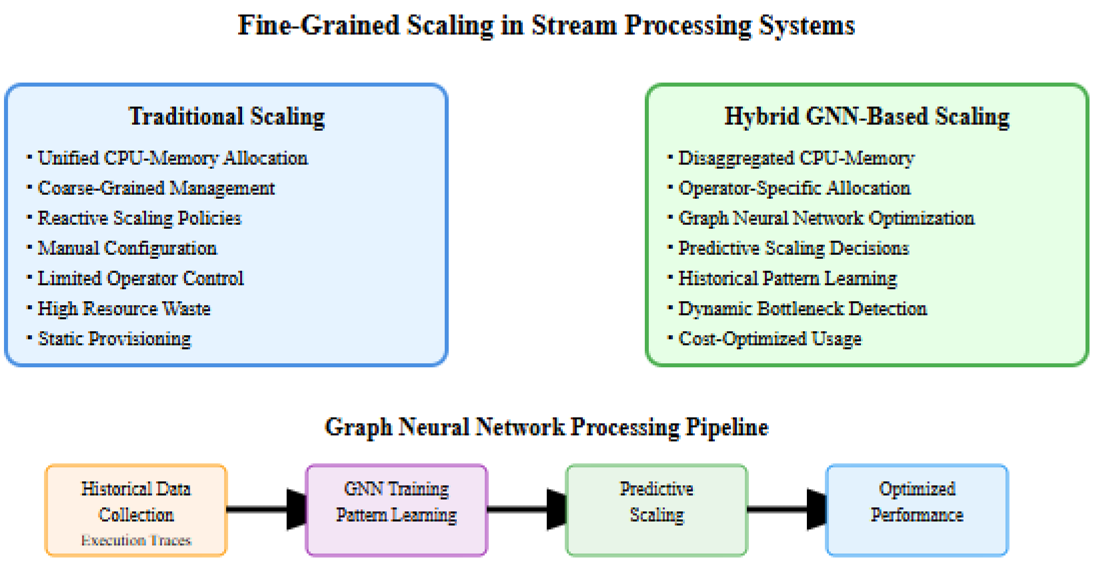
Fine-Grained Scaling in Stream Processing Systems
Traditional Scaling
• Unified CPU-Memory Allocation
• Coarse-Grained Management
• Reactive Scaling Policies
• Manual Configuration
• Limited Operator Control
• High Resource Waste
• Static Provisioning
Hybrid GNN-Based Scaling
• Disaggregated CPU-Memory
• Operator-Specific Allocation
• Graph Neural Network Optimization
• Predictive Scaling Decisions
• Historical Pattern Learning
• Dynamic Bottleneck Detection
• Cost-Optimized Usage
Graph Neural Network Processing Pipeline
Historical Data Collection
Execution Traces
GNN Training Pattern Learning
Predictive Scaling
Optimized Performance

Figure 2: *Fine-Grained Scaling Architecture and GNN Processing Pipeline [1,3]*

Table 3: *Implementation Challenges and Future Directions [2,9]*

Challenge	Future Direction
Container resource granularity	Enhanced orchestration integration
State preservation during scaling	Advanced checkpointing mechanisms
Cross-platform compatibility	Standardized scaling protocols
Model integration complexity	Automated ML pipeline management
Multi-tenant resource isolation	Federated learning strategies
Edge deployment constraints	Hybrid cloud-edge architectures
Real-time decision latency	Hardware-accelerated optimization
Operational overhead	Automated configuration tools

4. Conclusions

Fine-grained scaling mechanisms transform how stream processing systems handle computational resources through operator-specific optimization. Hybrid CPU-memory autoscaling enables precise resource allocation beyond traditional unified models, delivering measurable efficiency improvements. Graph Neural Network integration provides predictive capabilities, facilitating proactive resource management through historical pattern recognition. These advances achieve substantial resource utilization improvements while preserving essential performance characteristics for cloud-native environments. Justin and StreamTune implementations demonstrate practical feasibility in production settings, showing cost reductions without throughput or latency compromise. Machine learning-driven optimization defines future directions for intelligent infrastructure, adapting to workload variations autonomously. Neural network convergence with distributed processing creates self-optimizing systems utilizing real-time metrics and predictive modeling. Streaming workloads increasing in complexity necessitate fine-grained scaling adoption for competitive advantages while managing operational expenses. The technological foundation established enables autonomous resource management based on learned execution patterns and performance predictions. Cloud computing environments benefit significantly from these innovations through reduced operational costs and improved resource efficiency. Anticipated technological evolution will extend these principles into edge computing environments and multi-cloud deployment configurations. The transition from reactive to predictive resource management constitutes a significant breakthrough in distributed systems enhancement.

Author Statements:

- **Ethical approval:** The conducted research is not related to either human or animal use.
- **Conflict of interest:** The authors declare that they have no known competing financial interests or personal relationships that could

have appeared to influence the work reported in this paper

- **Acknowledgement:** The authors declare that they have nobody or no-company to acknowledge.
- **Author contributions:** The authors declare that they have equal right on this paper.
- **Funding information:** The authors declare that there is no funding to be acknowledged.
- **Data availability statement:** The data that support the findings of this study are available on request from the corresponding author. The data are not publicly available due to privacy or ethical restrictions.

References

[1] Yunfan Qing and Wenli Zheng, "Towards Fine-Grained Scalability for Stateful Stream Processing Systems," ResearchGate, Mar. 2025.https://www.researchgate.net/publication/389895054_Towards_Fine-Grained_Scalability_for_Stateful_Stream_Processing_Systems

[2] Ziliang Wang et al., "DeepScaling: microservices autoscaling for stable CPU utilization in large-scale cloud systems," in SoCC '22: ACM Symposium on Cloud Computing, ResearchGate, Nov. 2022.https://www.researchgate.net/publication/365223689_DeepScaling_microservices_autoscaling_for_stable_CPU_utilization_in_large_scale_cloud_systems

[3] Theodoros Theodoropoulos et al., "Graph neural networks for representing multivariate resource usage: A multiplayer mobile gaming case-study," International Journal of Information Management Data Insights, ScienceDirect, Feb. 2023.https://www.sciencedirect.com/science/article/pii/S2667096823000058

[4] Federico Lombardi, "PASCAL: An architecture for proactive auto-scaling of distributed services," Future Generation Computer Systems, ScienceDirect, Apr. 2019.https://www.sciencedirect.com/science/article/abs/pii/S0167739X18303728

[5] Tong Li et al., "Towards Fine-Grained Explainability for Heterogeneous Graph Neural Network," Proceedings of the AAAI Conference on Artificial Intelligence, Association for the Advancement of Artificial Intelligence, Jun. 2023. https://ojs.aaai.org/index.php/AAAI/article/view/26040

[6] Eugene Armah and Linda Amoako Bannning, "Towards a Proactive Autoscaling Framework for Data Stream Processing at the Edge using GRU and Transfer Learning," arXiv, Jul. 2025.https://arxiv.org/abs/2507.14597v1

[7] Christina Giannoula et al., "Accelerating Graph Neural Networks on Real Processing-In-Memory Systems," arXiv, Feb. 2024.https://arxiv.org/html/2402.16731v1

[8] Gabriele Russo Russo et al., "Hierarchical Auto-scaling Policies for Data Stream Processing on Heterogeneous Resources," ACM Digital Library, Oct. 2023.https://dl.acm.org/doi/10.1145/3597435

[9] Valeria Cardellini et al., "Runtime Adaptation of Data Stream Processing Systems: The State of the Art," ACM Digital Library, Sep. 2022.https://dl.acm.org/doi/abs/10.1145/3514496