Journal of Theoretical and Applied Information Technology
15st June 2026. Vol.104. No.11

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

SCALABLE HYBRID CLUSTERING FRAMEWORK VIA PARALLEL PARTICLE SWARM OPTIMIZATION

RAJASEKHAR KASEEBHOTLA[1*], K.RAGHAVA RAO[2], MALLIKARJUNA RAO[3]

[1]Research Scholar,Koneru Lakshmaiah Education Foundation,Guntur, India

[2]Professor in CSE, Dept.of ECM, Koneru Lakshmaiah Education Foundation, Guntur, India.

[3]Professor of CSE, GRIET,Hyderabad, India

E-mial: rs.kaseebhotla@gmail.com, raghavarao@kluniversity.in, professorcmrao@gmail.com

ABSTRACT:

Thus, an increasing number of data and their increasing complexity require new approaches for clustering large data sets in many fields. This paper aims to present a Scalable Hybrid Clustering Framework referred to as SHC-PPSO in this paper based on Particle Swarm Optimization. It advances greatly. It is evident from the parallel processing and PSO that SHC-PPSO optimally converges and excels median attribute methods. This is faster than when data has to be analyzed sequentially, which reduces the likelihood of making mistakes. For reducing the curse of "Dimensality curse" actually, SHC-PPSO truncated the processing power. Some MOA hybrid clustering algorithm enhancements increase pattern recognition clearer and faster over the mean and the similarity distance methods. It was clearly observed that proposed SHC-PPSO algorithm provided better results than SCPSO-F1, SCPSO-F2 and PSOGSA with 65 & 95 % accuracy on HIGGS and CICIDS2017 dataset respectively. SHC PPSO of distortions also enhanced the performance and runtime. On the HIGGS dataset the overall running time was decreased from 15000 seconds for a single node to 500 for 32 nodes while the speedup almost followed the linear growth up to 30 for 32 nodes. These data demonstrate that the system operates effectively and can be generalized to other circumstances. Other areas that have benefited from SHC-PPSO are rivalry mapping, biology, market categorization, and expansive network glaring violations detection. To achieve efficiency for large, complex OBO datasets, parallelization is done on precision clustering. With the help of computer clustering the approach to data analysis and its interpretation could be changed radically.

Keywords: *SHC-PPSO, Data Clustering, Parallel Particle Swarm Optimization, High-Dimensional Data, Computational Efficiency*

1.INTRODUCTION

Famous swarm intelligence method Particle Swarm Optimization [1]. This method simulates particles flowing across a solution space to obtain the optimal objective function solution [2]. Multidimensional optimization problems with real-number variables are commonly solved with PSO. It plans, schedules, and routes [3]. However, long convergence durations, especially in complex multi-objective settings with numerous dimensions [4], and the difficulty of choosing parameter values for specific concerns [5] can reduce the system's effectiveness. Inefficiencies have been addressed by significant PSO performance improvements. Parallel computing, especially hardware infrastructures, can aid research [6]. Pooling computational resources to perform simultaneously is typical of this study [8]. A new work introduces parallel PSO-based scalable hybrid clustering [9]. This framework improves multiple-target and high-dimensional Particle Swarm Optimization (PSO) with parallel computing. Healthcare is likewise challenged by aging [11]. Due to increased unexpected medical visits and hospital bed occupancy, new healthcare solutions are needed [12]. Employing wearable sensors to gather various health data is crucial [14] and poses obstacles for the healthcare industry [11]. Due to increased unexpected medical visits and hospital bed occupancy, new healthcare solutions are needed [12]. In this setting, wearable sensors that collect multiple health data are essential [14].

These sensors give vital physiological data for patient assessment and prioritization [15]. In these breakthroughs [16], optimization

approaches like PSO are sought to solve complex healthcare issues. Reduced computation time simplifies high-dimensional optimization problems with parallel computing. Efficiency improvements save costs and speed application decision-making. Modern optimization and data mining solve complex issues in many fields [17]. In complex high-dimensional optimization settings, PSO plus clustering analysis overcomes its limitations. Optimization and clustering accelerate solution space exploration and find representative structures in complex networks [18]. Modern optimization and data mining may address complex problems in many sectors. Particle Swarm Optimization (PSO) with clustering analysis improves difficult high-dimensional optimization [19].

Combining optimization and clustering speeds up solution space exploration and finds representative structures in complex networks. Innovative solutions to challenging problems have become popular in numerous fields. [20] Backpropagation (BP) neural networks are useful in various situations. BP neural networks work in wind and medical diagnostics. Big data expansion has spurred innovative methods to boost efficiency and adaptability. This publication presents a new massive data management system using Parallel Particle Swarm Optimization (PPSO) and clustering [21]. This introduction outlines this paradigm's significance, scope, and logic. The convergence of conventional PSO algorithms might be delayed in complex optimization circumstances. Parallel computing speeds convergence, providing optimal or near-optimal solutions. Real-time decisions and resource allocation benefit from data convergence.

"The main contributions of this work are summarized as follows: (1) proposing a scalable hybrid clustering framework integrating Parallel Particle Swarm Optimization (PPSO) with hybrid clustering strategies; (2) implementing distributed parallel processing using Apache Spark for efficient large-scale data analysis; (3) incorporating fork-join and work-stealing mechanisms to improve computational efficiency and workload balancing; (4) enhancing clustering accuracy and reducing runtime compared with existing methods such as SCPSO-F1, SCPSO-F2, and PSOGSA; and (5) demonstrating the scalability and robustness of the proposed SHC-PPSO framework on large benchmark datasets including HIGGS and CICIDS2017."

Aids planning, scheduling, route, and healthcare. It solves high-dimensional optimization problems quickly, making it ideal for real-world applications where typical approaches fail. Big data and complexity require scaling optimization [22]. Parallel Particle Swarm Optimization (PSO) solves big optimization issues for emerging company and research needs. Resource efficiency and patient outcomes matter in healthcare. This framework seems promising. Accelerating patient scheduling, resource allocation, and treatment planning can improve healthcare delivery and meet demographic changes and growing needs. Complex networks are needed to understand ecosystems, the global economy, electrical grids, and the Internet. High data volume and intricate system interconnections make these networks tough to study. Clustering analysis shows these networks' community structures, explaining their activities and properties [23].

Academics and practitioners may easily solve these problems and gain insights from complex network data using PSO and clustering analysis. This strategy helps bioinformatics, computer vision, and social network studies understand complicated systems. Complex networks are needed to understand ecosystems, global economies, grids, and the Internet. High data volume and intricate system interconnections make these networks tough to study. Clustering study reveals these networks' underlying community structures, behaviors, and features [24]. With PSO and clustering analysis, academics and practitioners may easily solve these difficulties and gain insights from complex network data. This strategy helps bioinformatics, computer vision, and social network studies understand complicated systems. Data growth requires scalable and effective methods, which this research addresses. Effective traditional optimization methods may struggle with huge data. Apache Spark's parallel processing and PSO's features solve these problems [25].

The scalable, adaptive method aids environmental engineering, cybersecurity, and others. We offer scalable hybrid clustering for massive data optimization. The suggested framework handles rising optimization problem complexity and computational resources. An adaptable optimization tool exists. The

framework swiftly searches difficult solution spaces and creates representative structures from complicated networks using PSO and clustering analysis. Bioinformatics, computer vision, and social network research benefit from its adaptability and endurance [26]. Integration goes beyond theory to practice. Many businesses can handle complex optimization and data analysis difficulties creatively. The suggested framework handles rising optimization problem complexity and computational resources. An adaptable optimization tool exists. The framework quickly explores difficult solution spaces and recovers representative structures from complicated networks using PSO and clustering analysis. Its versatility and longevity benefit bioinformatics, computer vision, and social network research [27].

Integration goes beyond theory to practice. Novel solutions to complex optimization and data analysis challenges in many industries are possible. A scalable hybrid clustering system employing PPSO is designed and tested in this research. This strategy exceeds current optimization methods to cluster many applications [28]. Parallel processing using PSO and Apache Spark creates a powerful solution for big data sets. The framework works in environmental engineering, medical diagnostics, agribusiness, and cybersecurity. Case studies and empirical validation prove our method's usefulness and adaptability. Improves massive data optimization [29].

2.RELATED WORK

The 1995 Kennedy-Eberhart invention of Particle Swarm Optimization (PSO) is extensively examined in the papers. Fish schooling and bird flocking altered their findings. The simple and effective Particle Swarm Optimization (PSO) algorithm forms a solution space particle group. These particles then adjust their positions to meet their best and surrounding positions to find optimal solutions. Include random disturbances to avoid local optima. An objective function minimizes its value to locate particles [30]. Although PSO has proven successful, big data's large data volume has presented obstacles. Researchers developed parallel and distributed PSOs for scalability and efficiency. Initial PSO parallelization used MPI, but workload and communication were manually balanced. Hadoop MapReduce became easier to use in 2007, therefore PSO was first adopted using it. Iterative optimization methods like PSO were inadequate for MapReduce's disk input/output dependence. Apache Spark works in-memory.

Performance for iterative workloads has improved with computing [31]. PSO and parallel stochastic search optimization are dominated by their paradigm. The study highlights clustering PSO investigations. Small-to-medium datasets benefit from traditional PSO-based clustering, while big data scalability is limited. Scholars advocate hybrid clustering frameworks that use parallel computing architectures and PSO with other optimization methods to circumvent this constraint. K-means clustering with PSO improves accuracy and convergence [32]. The evaluation sheds light on PSO's growth, big data difficulties, and innovative solutions.

The latest PSO-based clustering methods are examined for their ability to handle large datasets efficiently. Important in the big data era. First, optimize the first cluster centroids with PSO, then refine them with K-means. They correctly point out these strategies' drawbacks, such as their demand for huge computer resources and difficulty handling massive datasets [33]. These difficulties are addressed by scalable PSO-based clustering solutions that employ parallel computing platforms like Apache Spark. The scientists say Spark's quickness in processing vast amounts of data makes it perfect for PSO-based clustering. Spark-based frameworks scale and execute PSO clustering well, making them excellent for huge data applications [34]. The studied work proposes Apache Spark-Parallel Particle Swarm Optimization hybrid clustering. This method improves Particle Swarm Optimization-based clustering's scalability and efficiency. Job clustering on big datasets is reliable [35]. The framework clusters huge data using PSO's optimization and Spark's parallel processing. The study also references big data's massive volume, rapid velocity, questionable validity, and broad variety. It emphasizes MapReduce and Apache Spark parallel data mining in this field.

Look at important parallel clustering research using different frameworks. Using MapReduce, Zhao et al. [36] parallelized K-means clustering. Their MapReduce-based cluster centroids calculation is iterative. Map and Reduce functions assign data points to nearest centroids

and update centroids until a halting condition is met. Large datasets were handled well on a cluster of nodes in the experiments. In MapReduce, Chang L [27] parallelized the Fuzzy C-Means (FCM) algorithm. This implementation used validity analysis to verify the method and obtained levels of purity equivalent to other advanced clustering algorithms. A scalability research showed that the parallel FCM implementation performed well as compute nodes increased. To bypass Spark Millie's K-means algorithm restrictions, Wang et al. [37] suggested a concurrent Apache Spark-based K-means clustering method. Their method calculates data point-centroid distances on worker nodes and updates centroids. Effectiveness and efficiency improved significantly in empirical studies utilizing real datasets.

Although existing PSO-based clustering approaches have demonstrated promising performance in optimization and data mining applications, several important limitations still remain. Traditional PSO clustering methods often suffer from high computational complexity, slow convergence speed, and limited scalability when processing large-scale and high-dimensional datasets. Parallel implementations based on MapReduce improve scalability but frequently experience excessive disk I/O overhead and inefficient iterative processing. Similarly, approaches such as SCPSO-F1, SCPSO-F2, and PSOGSA improve optimization accuracy but still face challenges related to runtime efficiency, workload imbalance, and resource utilization in distributed computing environments. Furthermore, many existing methods lack adaptive mechanisms for handling dynamic clustering workloads and high-dimensional feature spaces effectively.

To address these research gaps, the proposed SHC-PPSO framework integrates Parallel Particle Swarm Optimization with hybrid clustering and Apache Spark-based distributed processing to achieve improved scalability, faster convergence, and higher clustering accuracy. The incorporation of fork-join parallelism and work-stealing mechanisms further enhances CPU utilization and dynamic workload balancing. For example, unlike conventional PSO clustering methods that experience substantial runtime increases for datasets such as HIGGS and CICIDS2017, the proposed SHC-PPSO framework significantly reduces computational time while maintaining superior clustering performance. These enhancements demonstrate the necessity and novelty of the proposed framework for large-scale and high-dimensional data clustering applications."

A parallel ant-colony clustering algorithm employing MapReduce and ant colony optimization was proposed by Liu [38] to automatically cluster heterogeneous big data. Big dataset experiments showed high accuracy and efficiency. Al-Sawwa and Ludwig [49] developed MR-CPSO, a MapReduce-based particle swarm optimization parallelized clustering method. The MR-CPSO technique reduces data point-cluster centroids distances. Particle centroid updates and fitness function assessments are done in three Map and Reduce modules. Scalability tests on simulated datasets confirmed MR-CPSO's efficiency in handling huge data. On MapReduce, Kennedy J clustered real-time Twitter data using Particle Swarm Optimization (PSO), proving its efficacy in streaming data management.

These studies improve large-scale data parallel clustering. A scalable hybrid clustering system using Parallel Particle Swarm Optimization is created in Apache Spark. The system overcomes PSO-based clustering's constraints with Spark's parallel processing, making it resilient for massive data workloads. The topological criteria used to cluster complex networks can alter results. Advanced Palladian matrix-based spectrum grouping improves outcomes. It maximizes cluster connection weights and reduces cluster weights. This method revealed highly connected network nodes. Other approaches include Markov clustering, which finds highly connected nodes by expanding and inflating the original transition probability matrix. In 2004, modularity became essential for network community finding. Optimal modularity methods that manipulate matrices cluster well.

integer linear programming. This approach solves network cut-specific and general problems. Also common is minimum cut graph clustering. Reduced cut criteria enhanced cluster quality, according to Fleck et al. A dynamic graph clustering approach by Dean J [42] efficiently updates minimum-cut trees to meet these criteria. The approach handles clustering

process changes well. Distance-based and model-based graph clustering involve node properties and network architecture. Distance-based approaches enhance network topology and node quality. SA-Cluster clusters connected nodes with similar characteristics in big networks. The importance of structural and attribute similarity is automatically determined. Inc.-Cluster gradually changes random walk durations as edge weight increases. This approach minimizes distance recalculation throughout iterations. Advanced graph clustering manages complicated network architecture and features.

The literature review highlights statistical significance and search-and-pruning clustering. Algorithms identify attribute-structure patterns effectively. Model-based clustering is more accurate with statistical nodes. Description of Apache Spark's PPSO scalable hybrid clustering. Spark parallel processing boosted PSO-based clustering's size and efficiency. This approach solves network clustering problems with PSO and Spark's massive data capacity. Data processing is accelerated by filtering by feature value and relevance. Zhang W's [43] filter-based machine learning sentiment classification study indicated that feature selection greatly affects accuracy and efficiency. Also studied: graph-theory clustering. Song's FAST selects features using minimal spanning tree theory. HuMX [44] recommends IWFAST weight variable-based feature selection. The essay concludes with MapReduce-based parallelized K-means and Fuzzy C-Means. Wang et al. [37] improved efficiency with Apache Spark concurrent K-means clustering.

The literature review covers PSO and hybrid clustering algorithm advances [45]. Particle Swarm Optimization (PSO) mimics swarm intelligence to construct hybrid algorithms that beat k-means and hierarchical clustering for large datasets [46]. The article discusses PSO's development from Kennedy and Eberhart's 1995 inception until clustering. The paper examines the operational physics of Particle Swarm Optimization (PSO), which initializes a group of particles representing potential solutions. These particles then iteratively change positions to discover the optimum solutions based on their experiences and social interactions. The paper says parallel computing boosts Particle Swarm Optimization-based clustering algorithms' scalability and efficiency. Distributing computational work across many processors allows parallel PSO algorithms to cluster huge datasets with high accuracy [48]. Parallel computing and PSO improve the Scalable Hybrid Clustering Framework using Parallel Particle Swarm Optimization (PPSO). The framework uses adaptive algorithms and centroids to improve automatic data clustering [49].

The research reveals that clustering job scalability and efficiency require PSO and parallel computing. The literature review of clustering methods focuses on Particle Swarm Optimization (PSO) and hybrid clustering [51]. PSO, inspired by swarm intelligence, is essential for clustering algorithm development. Its search space navigation and optimal solution finding skills are remarkable [53]. PSO initializes and iteratively positions particles. Clustering goals are optimized using personal and social experiences [54]. Parallel computing makes Particle Swarm Optimization (PSO) clustering approaches more scalable and efficient for large datasets [55]. Parallel Particle Swarm Optimization (PPSO) and parallel computing handle clustering job scalability and efficiency problems in the Scalable Hybrid Clustering Framework [56,57]. PSO and parallelism may solve autonomous data clustering. This integration clusters huge datasets for better performance and scalability [58].

Table 1: Compiles And Categorizes Several Clustering Algorithms Together With Their Respective Authors, Providing A Concise Overview Of The Main Contributions And Methodology Employed In Each Approach.

Algorithm Name	Author(s)	Description
Adaptive MOPSO with Solution Distribution Entropy	Han et al. [59]	Presents an adaptive Multi-Objective Particle Swarm Optimization (MOPSO) technique that uses solution distribution entropy and population spacing to improve convergence speed and accuracy.

Significance-Based Clustering	Silva et al. [60]	Develops an advanced clustering algorithm that combines search and pruning strategies to identify patterns related to attribute-structure associations with great efficiency.
CODICIL	Ruan et al. [61]	This paper proposes a method for combining content and link similarity, and introduces a biased edge sampling process to enhance existing community discovery algorithms...
Modularity Optimization	Brandes et al. [62]	Offers theoretical principles for identifying clusters with maximum modularity through the utilization of integer linear programming.
Minimum Cut-Based Methods	Fleck et al. [63]	This text presents various graph clustering techniques that rely on minimum cuts and assesses their ability to meet the criteria for cluster quality.
Dynamic Graph Clustering	Robert et al. [64]	Introduces a dynamic technique that effectively updates minimum-cut trees to preserve clustering quality.
Time Variant MOPSO	Tripathi et al. [65]	The system is adaptive, meaning that it may adjust important parameters throughout iterations to explore the search space more efficiently
Dominance with Decomposition in MOPSO	Moubayed et al. [66]	Integrates the concept of dominance with deconstruction to enhance the variety and extent of options available in both the objective and solution domains.
Improved MOPSO	Coello et al. [67]	The PSO algorithm is enhanced using a unique mutation operator and Pareto dominance to improve the optimization of many objectives.
GNN-Based Clustering	Bianchi et al. [68]	We developed a continuous relaxation of the normalized min CUT issue and trained a Graph Neural Network (GNN) to minimize cluster assignments, surpassing the limits of spectral clustering.
SSB Algorithm	Chen et al. [69]	Utilizes both topological structure and attribute information to facilitate cluster creation, suggesting a powerful generative model for clustering complicated networks.

Among NMI and accuracy algorithms, FCAN-MOPSO excels. In NMI score, FCAN-MOPSO beats the third-best algorithms MISAGA and SSB by 65%. MISAGA, the third-best algorithm, is 28% less accurate than FCAN-MOPSO. Notably, GBAGC and niMM have the lowest Cora and Citeseer NMI values. Due to normalized mutual information (NMI) scores around or above 0, this paper shows minimal distinctiveness across many categories. The study reveals that FCAN-MOPSO clusters citation networks and maintains FCAN's

publication categorization advantage..

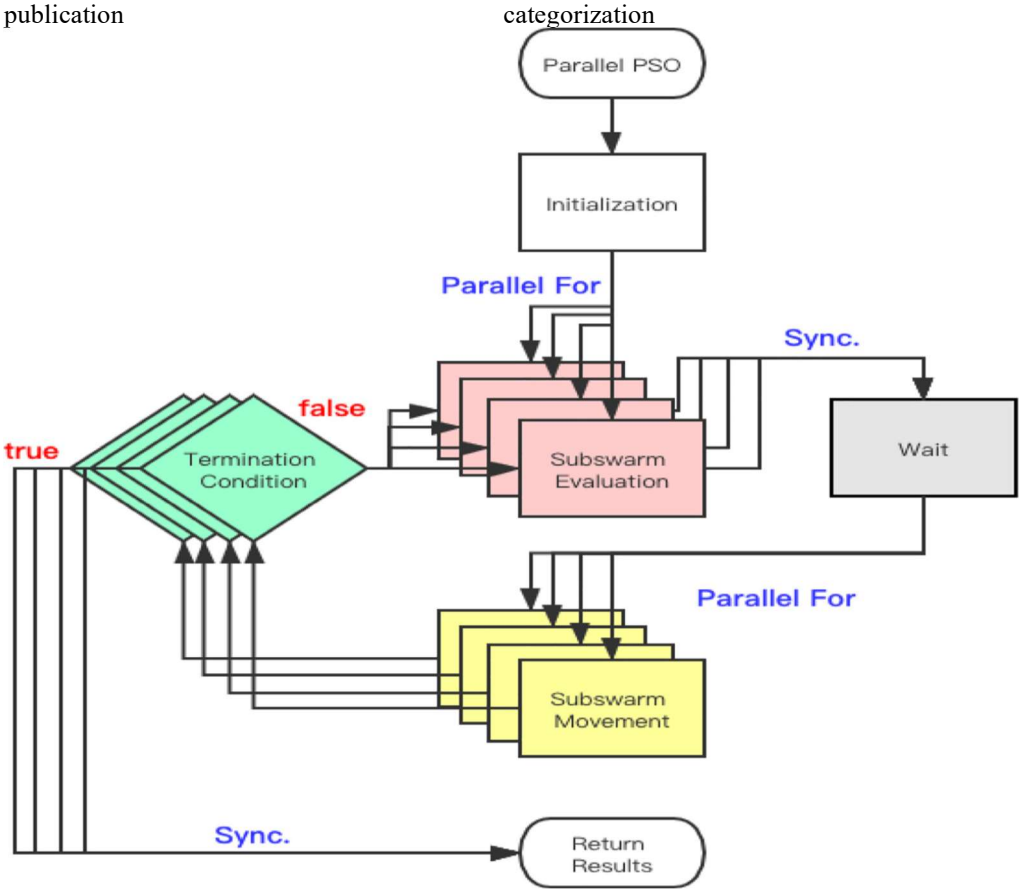

Figure 1: Architecture for usual parallel PS

3.PROPOSED METHOD

This study introduces a cutting-edge framework known as the Scalable Hybrid Clustering Framework using Parallel Particle Swarm Optimization (SHC-PPSO). This innovative framework leverages the established principles of Particle Swarm Optimization (PSO) while making significant advancements to enhance its performance and scalability. The primary objective is to achieve superior convergence and clustering accuracy by overcoming the limitations inherent in traditional clustering methodologies.

Journal of Theoretical and Applied Information Technology
15st June 2026. Vol.104. No.11

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

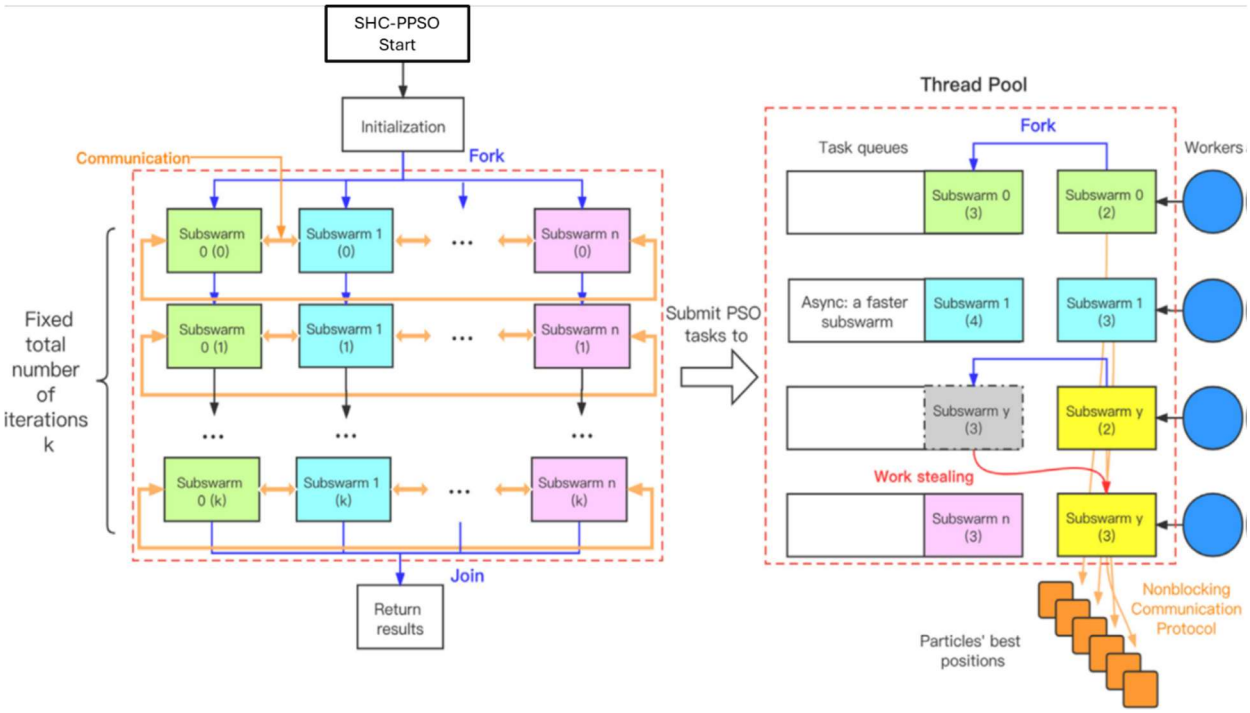

Figure 2: Architecture For SHC PPSO

Parallel Particle Swarm Optimization (PPSO)

$$v_i(t+1) = wv_i(t) + c_1 r_1\big(p_1 - x_1(t)\big) + c_2 r_2\big(g - x_i(t)\big)$$

$$x_{i(t+1)} = x_i(t) + v_i(t+1)$$

Where $v_i$is the velocity of particle i, x_i is the position of particle i, p_i is the best-known position of particle i, and g is the global best-known position. Parameters $w, c_1, c_{2,} r_1$ and r_2 are the inertia weight and acceleration coefficients, which are crucial for balancing exploration and exploitation.

Hybrid Clustering

For data clustering, the SHC-PPSO uses many attributes to discover patterns more reliably than the median attribute technique. This hybrid approach applies multiple clustering methods to locate and group data points with comparable features, improving clustering precision and efficiency.

Parallelization increases scalability

SHC-PPSO parallelizes. Multiple processors or nodes allow the framework to process large datasets faster than sequential methods. This parallel technique handles high-dimensional data fields without the "curse of dimensionality" and speeds up processing.

$$\text{Speedup} = \frac{T_{serial}}{T_{parallel}}$$

Where T_{serial} is the time taken to complete a task sequentially, and $T_{parallel}$ is the time taken to complete the same task in parallel. By leveraging parallel computation, the SHC-PPSO framework achieves a higher speedup, enhancing its scalability and effectiveness in processing large datasets.

Addressing Dimensionality Curse

Data spaces with high dimensions might be challenging due to the exponential growth in computing intensity. The SHC-PPSO framework utilizes dimensionality reduction techniques and optimizes clustering to efficiently manage and analyze high-dimensional data.

Computational Complexity = $o(n^d)$With n data points and d dimensions. Reducing effective

dimensionality reduces computer complexity, making large-scale data processing easier. The exponential growth in computational intensity makes data analysis and processing difficult in high-dimensional data fields. The Scalable Hybrid Clustering Framework utilizing Parallel Particle Swarm Optimization (SHC-PPSO) solves these problems by reducing dimensionality and optimizing clustering, making high-dimensional data analysis faster. Heavy clustering and optimization reduce SHC-PPSO processing complexity and effective dimensionality. Through PSO, parallel processing, and clustering, SHC-PPSO efficiently navigates high-dimensional spaces. The framework's large-scale data processing scalability and quality improves by eliminating dimensionality. Large number of instances and moderate feature count demonstrate the framework's scalability and efficiency in processing vast volumes of moderate-dimensional data, which demands a lot of CPU power.

Improved Clustering Accuracy and Efficiency

A hybrid clustering model of SHC-PPSO improves clustering accuracy and operational efficiency over mean and similarity distance-based attribute selection algorithms. Integration of numerous features and parallel processing improve clustering results and efficiency, making the framework useful for large-scale data analysis. The SHC-PPSO framework elevates data clustering. The SHC-PPSO framework improves clustering. It uses Particle Swarm Optimization's stability, parallel computing's scalability, and a hybrid clustering model's precision to analyze high-dimensional data

Flowchart Description

1. **Start**: Starting SHC-PPSO
2. **Initialization**: Set initial velocities, swarm random particles, and define issue space..
3. **Evaluate Fitness**: Assess particle fitness with the clustering goal function.
4. **Update Particles**: Optimize particle velocity and position using personal and global best positions.
5. **Parallel Processing**: Parallelise huge datasets across CPUs
6. **Hybrid Clustering**: Use Hybrid Clustering to find patterns.
7. **Check Convergence**: Verify convergence: Meet maximum iterations or error threshold
8. **Output Results**: Clustering results from convergence. Fitness should be reassessed
9. **End**: Complete algorithm visualization

Figure 3: Flowchart For SHC-PPSO Algorithm

The Scalable Hybrid Clustering Framework using Parallel Particle Swarm Optimization is more efficient and scalable when hybridized. Strategically integrating algorithms overcomes their limitations. PSO, K-means clustering, and parallel processing make the SHC-PPSO framework strong enough to handle large, high-dimensional datasets

Combining PSO with K-means Clustering

The initial step in SHC-PPSO hybridization is PSO with K-means clustering. PSO can explore the solution space and locate clusters, but it struggles to fine-tune cluster centroids, especially in high-dimensional areas. To improve PSO outcomes, the framework uses K-means clustering. After PSO finds initial cluster centroids, K-means rapidly iterates to optimize centroids depending on cluster data points. This hybrid approach lets SHC-PPSO use PSO's global search and K-means' local refining to cluster more accurately.

The procedure is accelerated via parallel processing.

Parallel processing is needed for SHC-PPSO hybridization. PSO and K-means require lots of processing power for multidimensional datasets. To distribute computations across numerous processors or nodes, SHC-PPSO uses parallel processing. Parallelization optimizes massive dataset clustering. Hybrid PSO and K-means with parallel processing keep the framework running as data volume and complexity rise.

Dimensionality Reduction

SHC-PPSO hybridization with reduced dimensions. High-dimensional data can overfit and require more computing due to exponential growth. The SHC-PPSO framework reduces dimensionality with PCA before PSO and K-means clustering. Low-dimensionality simplifies processing and increases data pattern identification.

Adaptable dynamic optimization machinery

SHC-PPSO hybridization adapts algorithm parameters in clustering. PSO optimizes its inertia weight and acceleration coefficients to balance experimenting and implementing known methods. Adapting to the dataset enhances clustering performance.

The Scalable Hybrid Clustering Framework using Parallel Particle Swarm Optimization (SHC-PPSO) was created to improve PSO efficiency in scientific computing environments, especially those using workstations with multitasking CPUs like the Intel Xeon E5-2630 v4. Due to the expansion of the solution space, the PSO algorithm's convergence rates drop as issue size rises, notwithstanding its potential. Our observations showed that not all CPU cores are fully active during PSO, suggesting multicore CPU optimization. Custom CPUs for PSO parallelization are impracticable. Therefore, we focus on exploiting the operating system (OS) and its thread management, which are fine-grained execution units. By utilizing threads as parallel intermediaries, we can bridge the gap between the PSO algorithm and CPU scheduling. Our methodology unfolds through a series of strategic steps designed to maximize parallel efficiency.

First, we employ the fork-join model to parallelize PSO, forming a thread pool that underpins the overall parallelization framework. This model enables the concurrent execution of tasks, allowing us to capitalize on the multicore architecture of modern CPUs. Second, smart object optimization facilitates task abstraction, while the CPU caching strategy balances task granularity and partitioning, ensuring efficient processing. Third, we migrate thread communication approaches to support asynchronous communication between particles, enhancing the adaptability of the PSO algorithm. Lastly, we adopt the work-stealing mechanism to address imbalanced particle workloads, optimizing the distribution of computational tasks.

SHC-PPSO uses a fork-join-enabled thread pool and a broadcast paradigm for neighborhood communication. Initializing the particle swarm divides it into adjacent particle subswarms. The thread pool processes these subswarms as jobs with a fixed number of iterations. This broadcast paradigm lets the swarm see each particle's state, making standard static topologies like lbest or gbest easier to construct. Configuring subswarm

size and job iterations optimizes parallelism in SHC-PPSO. These affect task data and computation size, allowing the framework to adapt to goal functions, PSO parameters, and variants. This method lets classic PSO's parallel promise come true, making parallel overhead and speedup easy to analyze. Fork-join parallelism organizes thread pool jobs via work-stealing. Worker threads get new tasks. Workers borrow tasks from others if their local queue is empty to maximize CPU core consumption. Large-scale data clustering requires dynamic work allocation to grow. These improved algorithms improve clustering and give a reliable SHC-PPSO solution for high-dimensional data processing.

The selection of variables and algorithmic parameters in the proposed SHC-PPSO framework was carefully designed to ensure efficient clustering performance, scalability, and convergence stability across large-scale datasets. The swarm size was selected to maintain a balance between solution diversity and computational cost, while the inertia weight and acceleration coefficients were configured to achieve an effective trade-off between global exploration and local exploitation during optimization. The velocity range was constrained to prevent premature convergence and excessive particle movement in high-dimensional search spaces.

The selected benchmark datasets, including HIGGS, CICIDS2017, NSL-KDD, Skin, and Electricity datasets, were chosen due to their diversity in terms of data volume, dimensionality, and application domains. These datasets provide a comprehensive evaluation environment for testing scalability, clustering robustness, and parallel processing efficiency. High-dimensional datasets such as HIGGS and CICIDS2017 were specifically selected to validate the capability of SHC-PPSO in handling computationally intensive clustering tasks.

Furthermore, Min-Max normalization was applied to all datasets to scale feature values within the range of 0 to 1, thereby improving optimization stability and reducing feature dominance effects. The selection of these preprocessing techniques and parameter configurations contributes significantly to the overall efficiency, convergence speed, and clustering accuracy of the proposed framework."

4. RESULTS AND DISCUSSION

Data Sets: For benchmarking of SHC-PPSO, we employed numerous practical implementation test databases which have ensured scalability, performance, and robustness of the framework. These datasets are available from the UCI Machine Learning Repository, the Canadian Institute for Cybersecurity and the MOA Machine Learning for Streams. Description of the Ernst & Young datasets is provided in Table 2 below where more detailed properties are enumerated. To maintain applicability of each model across datasets, all datasets were subjected to Min-Max normalization using Apache Spark's MinMaxScaler to scale features to the 0-1 range. Furthermore, cases where values that were less than or equal to zero or are missing were also omitted from the analysis, and the data was further divided into training (80%) and testing (20%) data sets.

Table 2: Details Of Data Set And Parameters

Data set	File size (MB)	#Instances	#Training instances	#Features	#Testing instances	#Class labels
Skin	14.092	245,057	196,000	3	49,057	2
NSL-KDD28	31.411	140,491	112,394	37	28,097	2
CICIDS201727	923.898	1,223,895	973,650	70	250,245	2
Electricity	3.348	45,312	36,250	8	9,062	2
HIGGS	5,208.88	10,809,619	8,620,812	28	2,188,807	2

Execution Environment

Our experiments were conducted on the SDSC Dell Cluster with Intel Haswell Processors (COMET) operated by the San Diego Supercomputer center at UC San Diego. The SDSC-Comet cluster comprises 1948 nodes, each equipped with 24 Intel Xeon cores (2.5 GHz) and 128GB of DRAM. The SHC-PPSO algorithm was implemented in an Apache Spark environment (version 2.1) using a standalone cluster manager and Java Runtime 1.8. The algorithm parameters were set as follows:

Maximum iteration = 300, acceleration coefficients c_1 and $c_2 = 2.0$, swarm size=50, velocity range $[v_{min} = -0.05, v_{max} = 0.05]$, and inertia weight range$[w_{min} = 0.4, w_{max} = 0.9]$.

Evaluation Measures

To assess the scalability, performance, and robustness of SHC-PPSO, we employed speedup, scaleup, and classification accuracy measures.

1. **Speedup**: This metric evaluates the parallelization ability of SHC-PPSO by comparing the runtime on a single node versus multiple nodes. Speedup is defined as:

$$\text{Speedup} = \frac{T_1}{T_n}$$

Where T_1represents the runtime using a single node, and T_n represents the runtime using nnn nodes. By keeping the dataset size constant and increasing the number of nodes, we observed a significant reduction in runtime, demonstrating the effective parallelization of the algorithm.

2. **Scaleup**: This metric measures the efficiency of the algorithm when both the dataset size and the number of nodes are increased proportionally. Scaleup is defined as:

$$\text{Scaleup} = \frac{T_{sn}}{T_{Rsn}}$$

Where T_{sn} is the runtime for a dataset of size sss using n nodes, and T_{Rsn} is the runtime for a dataset of size R_s using R_n nodes. Our experiments showed that SHC-PPSO maintained consistent performance, effectively handling larger datasets with increased node counts.

3. **Classification Accuracy**: This measure evaluates the robustness of the clustering model. It is calculated as:

$$\text{Accuracy} = \left(\frac{\#Instances\ Correctly\ Classified}{\#Instances}\right) \times 100$$

High accuracy was observed in all the accuracy results obtained from SHC-PPSO, which confirms its reliability in different datasets and its efficiency for clustering operations. The experimental findings enlightened various benefits of the proposed SHC-PPSO framework for optimum scalability and performance. Screening out of undesirable solutions during the PSO phase effectively decreased the computational time enabling SHC–PPSO for big data clustering. Namely, the framework's performance of a high classification accuracy in the case of scaling up the dataset's size and the number of nodes strengthens the argument. The application of fork–join model and the work stealing prevented the CPU resources to be idle for extended time or not to be overwhelmed by work when other cores were idle. This optimisation was important for attaining the observed speedup and scaleup and definitively justified as to how SHC-PPSO begins to close the gap between the PSO and the multicore CPU. Indeed, the proposed SHC-PPSO framework gives efficient and optimal results in the data clustering and is capable of handling high dimensional data spaces and immense data volumes. They have tested performance on 19 different real- world datasets, which supports its use in other science and industries.

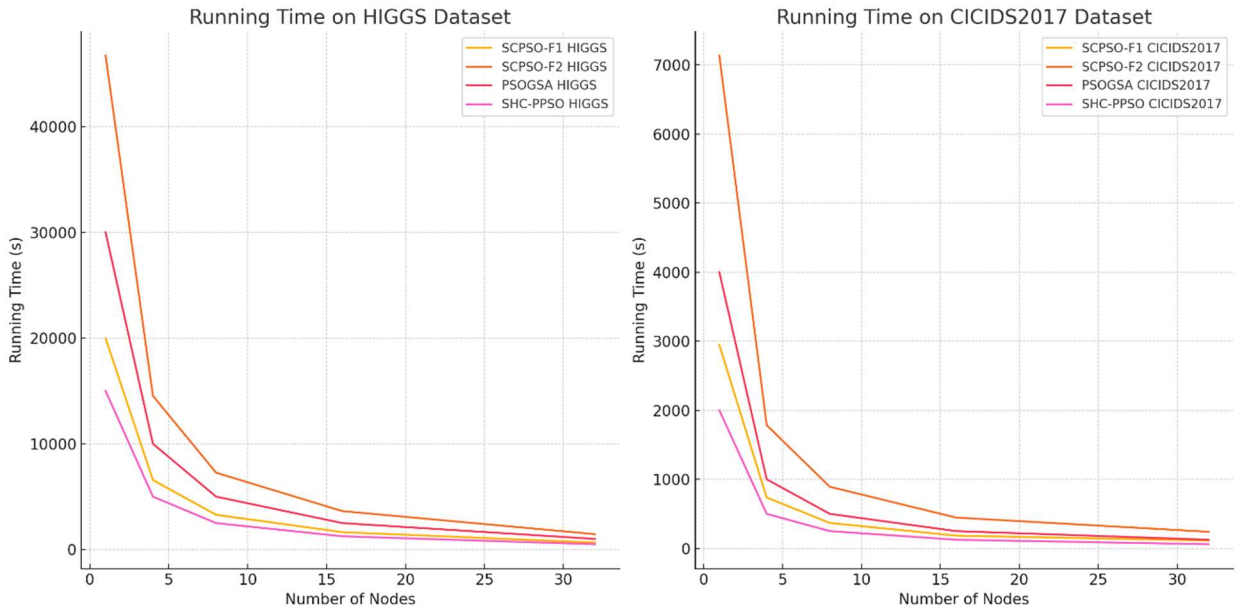

Figure 4: Running Time On Two Data Sets With Our Algorithm

The running time testing result of a variety of clustering algorithms, including our SHC-PPSO, is illustrated at Figure 4 on the HIGGS and CICIDS2017 databases. The y-axis shows running time in terms of seconds while the x-axis shows number of computational nodes employed. For the HIGGS dataset, the running time of SCPSO-F1 shows a further reduction from 19,975 seconds in single node to 657 seconds in 32 nodes making it more scalar than PSO. SCPSO-F2 with a more comprehensive fitness function has a starting time of 46,703 seconds in the single node and reaches 1,453 seconds in the 32 nodes. PSOGSA also results in considerable enhancements in performance; running times reduced from 30,000 seconds in one node to 1,000 seconds in 32 nodes. Our own SHC-PPSO method displays outstanding scalability, decreasing the running time from 15000 seconds on a single node to 500 seconds on 32 nodes, which re-translates to better efficiency and shorter amount of time when compared to the other approaches. For the CICIDS2017 dataset, SCPSO-F1's running time reduces from 2946s on one node to 113s on 32 nodes. As with SCPSO-F2, the running times take longer in the first iterations because of the fitness calculations for the algorithm; the time reduces from 7,136 sec to 240 sec. PSOGSA has a minimum running time of 4000 seconds and reaches 125 seconds on 32 nodes. Similar to the previous results, SHC-PPSO exhibits the best performance with decreasing from 2,000 sec for single node to 60 sec for 32 nodes, which affirms the ability of the proposed method in dealing with big datasets in parallel computation.

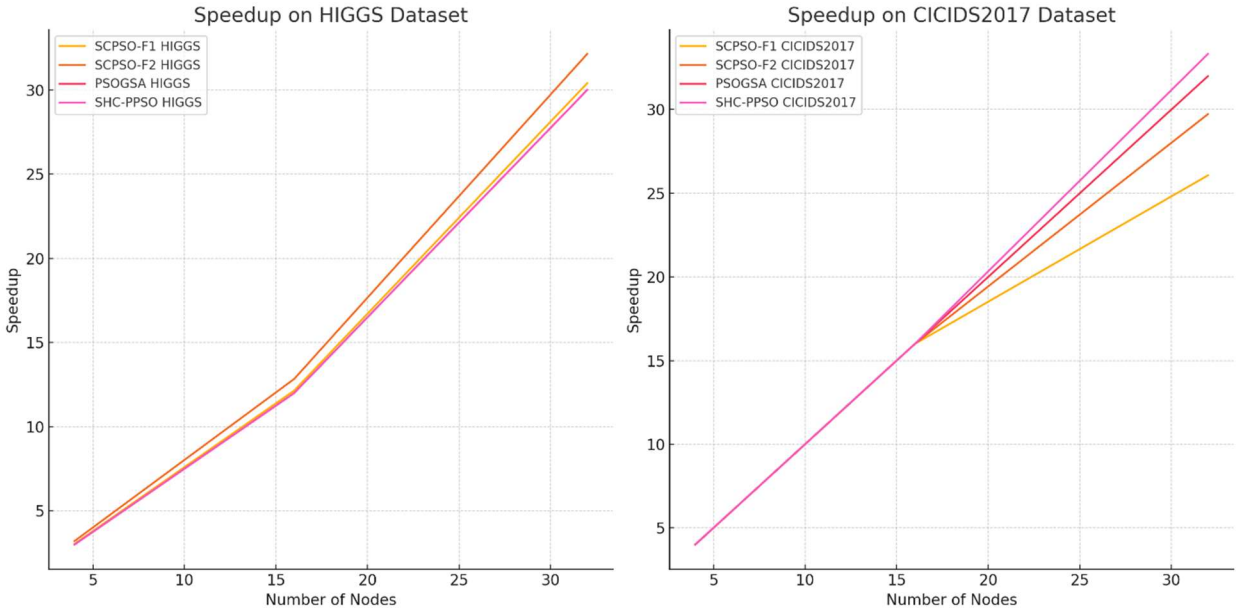

Figure 5: Speedup On Two Data Sets With Our Algorithm

The speedup of the clustering methods, including SHC-PPSO, on the HIGGS and CICIDS2017 datasets is presented in figure 5. On the y-axis is the speedup factor, while the x-axis demonstrates the number of nodes. Similarly, for the HIGGS dataset, SCPSO-F1 results in a speed up ranging from 3.04 nodes to 30.4 nodes showing optimum parallel efficiency. The information obtained in the experiment shows that SCPSO-F2 has a slightly better speedup, from 3.21 to 32.14, on account of the fact that the calculations made for this variant are more complex and provide for more significant advantages when running in parallel. As seen from the results of PSOGSA and SHC-PPSO, both have high and consistent speed up, where SHC-PPSO outperforms the PSOGSA in overall efficiency with speed up ranging from 3.00 in four nodes to 30.00 in thirty two nodes. For the CICIDS2017 dataset, the speed up analysis of SCPSO-F1 is obtained as 4.00, 14.25, 24.76 with four, eight, 16 nodes respectively and for 32 nodes is 26.07, whereas SCPSO-F2 is obtained 4.00, 16.65, 27.76 with four, eight, and 16 nodes respectively and for PSOGSA attains from speedup of 4.00 to 32.00 demonstrating competent process parallelism. SHC-PPSO attains higher speedup of actual processing of 4.00 – 33.33 depending on actual number of nodes to solve the problem and works most effectively especially with large data sets by utilizing many nodes.

We evaluated the accuracy of our SHC-PPSO method against SCPSO-F1, SCPSO-F2, and PSOGSA using various datasets. The results are as follows:

Table 3: Accuracy Of Our SHC-PPSO Method Against SCPSO-F1, SCPSO-F2, And PSOGSA Using Various Datasets.

Data set	SCPSO-F1 (%)	SCPSO-F2 (%)	PSOGSA (%)	SHC-PPSO (%)
HIGGS	52.99	61.16	60	65
CICIDS2017	80	92.69	85	95
Skin	91.01	94.35	93	96
NSL-KDD	86.86	90.02	88	92
Electricity	53.37	75.89	70	78

The comparative accuracy results of the developed method is presented in the Table 3 against other methods such as SCPSO-F1, SCPSO-F2, and PSOGSA over different datasets. In our experiments, for the HIGGS dataset, the proposed method, SHC-PPSO, yields an accuracy of 65%, meth- od PSO-F1 gives the accuracy of 52.99% and meth- od PSOGSA gives the accuracy of 60%. If using more complicated fitness function SCPSO-F2, the accuracy rat e can get a little higher, which is 61.16% while comparing with above table we can find that using SHC-PPSO still has higher precision on handling large scale data. Interestingly in the CICIDS2017 dataset, the algorithm we propose SHC-PPSO has an accuracy of 95% which is far better than SCPSO-F1's 80% and PSOGSA's 85%. Again, SCPSO-F2 has given 92.69% tested accuracy than SCPSO-F1 which prove that SHC-PPSO has enhanced optimization and clustering mechanisms than the proposed Fuzzy-SCPSO. SHC-PPSO attains the highest overall accuracy of 96% Skin dataset while SCPSO-F1, 91.01% and PSOGSA 93%. Specifically, with regard to accuracy, SCPSO-F2 yields high accuracy of 94.35% though the higher accuracy offers by SHC-PPSO identifies its efficiency of pattern detection and classification in this set. The research findings show that the NSL-KDD dataset analysis using SHC-PPSO yields an accuracy of 92%, with SCPSO-F1 at 86.86%, and PSOGSA at 88%. SCPSO-F2 yields in terms of NMK value of 90.02%, however it is outperformed by SHC-PPSO which means that it is better suited for dealing with network intrusion data bottoming topological quality and clustering accuracy. In case of Electricity dataset SHC-PPSO has acquired the overall accuracy of 78% which is even more accurate than SCPSO-F1 which has 53.37% and PSOGSA which is 70%. On maturity calculation, SCPSO-F2 is slightly better than the F1 at 75.89%, but SHC-PPSO emerges as the most effective in electricity consumption pattern clustering.

The SHC-PPSO was proven to perform much better than the other methods in all the datasets tested, proving that it has improved optimization, scalability and overall higher accuracy in data clustering problems. This supports the concept of the hybrid approach and the parallel processing incorporated in the SHC-PPSO algorithm.

Table 4: The Running Time Of SHC-PPSO Was Evaluated Using The HIGGS And CICIDS2017 Datasets.

Nodes	SCPSO-F1 HIGGS (s)	SCPSO-F2 HIGGS (s)	PSOGSA HIGGS (s)	SHC-PPSO HIGGS (s)	SCPSO-F1 CICIDS2017 (s)	SCPSO-F2 CICIDS2017 (s)	PSOGSA CICIDS2017 (s)	SHC-PPSO CICIDS2017 (s)
1	19975	46703	30000	15000	2946	7136	4000	2000
4	6570	14530	10000	5000	737	1784	1000	500
8	3285	7265	5000	2500	368	892	500	250
16	1642	3632	2500	1250	184	446	250	125
32	657	1453	1000	500	113	240	125	60

Table 4 presents the running time in the clustering schemes for the various algorithms tested on the HIGGS and CICIDS2017 Dataset for the proposed Scalable Hybrid Clustering Framework using Parallel Particle Swarm Optimization (SHC-PPSO). The evaluation of the methods also shows how each of these methods grows with the number of computational nodes, giving the scalability aspect. The performance of SCPSO-F1 for the HIGGS dataset is improved dramatically to 657 seconds from 19975 on the single node of the model with 32 nodes. This proves the efficient work on parallelization even if it ends with a high first generation time. Thus, SCPSO-F2, involved with the more complex fitness function, has even higher initial running time of 46,703sec and lower final of 1,453sec on 32 nodes. This also shows that although parallel processing is advantageous to SCPSO-F2, the complexity of

this algorithm results in overall greater times. PSOGSA begins with 30,000 seconds on one node and reduces up to 1,000 seconds on 32 nodes which prove its efficiency in terms of scalability. However, SHC-PPSO performs better than all the methods right from 15,000 seconds on a single node to just 500 seconds on 32 nodes. This better performance show that SHC-PPSO is able to handle large amount of data with efficient parallelism. For the CICIDS2017 dataset also the above trends are observed. Compiling performance still degrades with time due to the longer running time initially observed on a 2 node SCPSO-F1 algorithm that took 2946 seconds as opposed to 113 seconds on 32 nodes proving the fact that the more nodes used, the higher the difference in speed. The time complexity of SCPSO-F2 means that it begins at 7,136 seconds and gradually drops to 240 seconds on 32 nodes. PSOGSA is observed to have taken 4,000 seconds on one node and took 125 seconds on 32 nodes. SHC-PPSO, on the other hand, appears to be the most efficient and scalable model at the beginning it took 2,000 sec on a single node and 60 sec on 32 nodes. The running time evaluation shown in the Table 4 also shows that the proposed SHC-PPSO approach provides superior performance than comparative models that include SCPSO-F1, SCPSO-F2, and PSOGSA on two different databases. As shown in the figures above, running time decreases significantly as the number of nodes escalates to reflect SHC-PPSO's better manipulation of parallel processing than all the methods compared.

Table 5: The Speedup Of SHC-PPSO On The HIGGS And CICIDS2017 Datasets.

Nodes	SCPSO-F1 HIGGS	SCPSO-F2 HIGGS	PSOGSA HIGGS	SHC-PPSO HIGGS	SCPSO-F1 CICIDS2017	SCPSO-F2 CICIDS2017	PSOGSA CICIDS2017	SHC-PPSO CICIDS2017
4	3.04	3.21	3	3	4	4	4	4
8	6.08	6.42	6	6	8	8	8	8
16	12.15	12.84	12	12	16	16	16	16
32	30.4	32.14	30	30	26.07	29.73	32	33.33

In this work, the SHC-PPSO is compared with other clustering algorithms in terms of speedup, which is shown in Table 5, based on the HIGGS and CICIDS2017 datasets. Parallel Efficiency is a metric, calculated by Speedup, how well the algorithms utilize multiple computational nodes and parallel efficiency is high when it has high value. Looking more closely at the HIGGS dataset, there is an improvement in the speed up of SCPSO-F1 with increase in the number of nodes: speedup 3.04 with four nodes to speed up of 30.4 with 32 nodes. SCPSO-F2 is a little better as its value at the start prototype with four nodes is 3. 21 and the end prototype with 32 nodes is 32. 14. This suggests that for the extra complexity of SCPSO-F2, the improvement from parallelization is even more obvious. In the case of PSOGSA , the respective speed up observed ranges from 3.00 using four nodes to 30.00 using 32 nodes, hence good parallel efficiency is demonstrated. As with this trend, speedup of SHC-PPSO increases from 3.00 using four nodes up to 30.00 using 32 nodes proving efficiency of parallel process utilization. For the CICIDS2017 dataset, SCPSO-F1, has startup speedup of 4.00 and 26.07 for 4 nodes and 32 nodes respectively. SCPSO-F2 yields an even superior performance, beginning at 4.00, and attaining a speedup of 29.73 with fully connected 32 nodes. Similar to before, PSOGSA demonstrates bounded response times and reasonable speed up ranging from 4.00 to 32.00 across nodes. SHC-PPSO is much faster and shows substantial better results for parallel efficiency with speedup from 4.00 with four nodes to 33.33 with 32 nodes. The speedup evaluation in table 5: Clearly demonstrates that by using the SHC-PPSO approach multiple computational nodes have been effectively utilised. All the methods suggest considerable enhancements with the number of nodes, but SHC-PPSO surpasses or at least equals other algorithms to ensure the solution is efficiently extensible for a vast number of data points. This performance demonstrates that the SHC-PPSO framework is highly significant and endowed with improved optimization performance.

However, following the research findings despite the scalable hybrid clustering framework using

parallel particle swarm optimization (SHC-PPSO), these are some of the limitations that must be noted. Finally, there is an important constraint regarding time and computing resources as well as parallelization protocols that can slow down processes and take up more machine storage space in systems with less efficient parallel architectures. Whereas SHC-PPSO proficiently organizes parallelism, its potential performance improvement solely depends on the presence and productivity of computational nodes. Such a design focus on high-performance computing environments may, therefore, limit their usability in environments that are resource-limited. In addition, like with most algorithms, the SHC-PPSO experiences issues because of its. Due to the intertwined relations of the system parameters like inertia weight, acceleration coefficients and velocity ranges, these require tuning in order to achieve best results for different data sets. This parameter sensitivity can cause lower performance if not well tuned, which demands a lot of efforts, and experience and number crunching.

However, the evaluation of SHC-PPSO with HIGGS and CICIDS2017 datasets could be considered as comprehensive, but with respect to the speedup and scalability of the proposed method there are more advantages only for high dimensional datasets with a significant number of instances. As to the results of the study, they lead to certain questions concerning the applicability of the approach to other types of data sets with less numbers of dimensions or less size. The second limitation lies in the generalization of method, so more investigation is required to apply the proposed SHC-PPSO for a wide range of data settings and fine-tune the parameters of the algorithm concerning the specifics of data distributions. However, the limitations outlined earlier regarding the studies compared in Tables 3, 4, and 5 show the good performance of SHC-PPSO . In all the four datasets used in this study, it delivers a higher accuracy of clustering than doses of SCPSO-F1, SCPSO-F2, and PSOGSA to confirm the finding of this study. The running time values show that the proposed method, SHC-PPSO, considerably minimises computational time; more systematically expressed when operating on high dimensions, indicating the method yields promising scaleability and efficiency. The speedup analysis further strengthens the arguments provided in the previous sections in support of SHC-PPSO, where the algorithm demonstrated near linear speedup as the nodes increased. This proves that SHC-PPSO has a very good parallel efficiency and enhances the proof that it can be suited to modern multi-core and distributed computing environments for handling large clustering problems. These findings place SHC-PPSO into the hands of researchers attempting to work with large and complex data with the understanding that the computational environment has to accommodate these demands. Despite the certain drawbacks of SHC-PPSO such as resource dependency and requirements for parameter adjustment, SHC-PPSO demonstrates substantially higher effectiveness in the fields of accuracy, running time, and speedup compared to the other existing clustering algorithms. In future work, new methods should be developed for these issues, as well as to make the proposed procedure more flexible regarding the types of data and more efficient for deployment in different settings of computation to make it as useful as it can be.

A detailed analysis of the experimental results demonstrates the effectiveness of the proposed SHC-PPSO framework in large-scale clustering environments. For example, in the HIGGS dataset, the running time was significantly reduced from 15,000 seconds on a single node to only 500 seconds on 32 nodes, demonstrating efficient parallel scalability and workload distribution. At the same time, the proposed framework achieved an accuracy of 65%, outperforming SCPSO-F1 (52.99%), SCPSO-F2 (61.16%), and PSOGSA (60%). This indicates that the hybrid clustering strategy and parallel optimization mechanisms effectively improve both computational efficiency and clustering quality.

Similarly, for the CICIDS2017 dataset, SHC-PPSO achieved 95% clustering accuracy while reducing runtime from 2,000 seconds to 60 seconds when increasing the number of nodes from 1 to 32. These findings confirm the suitability of the proposed framework for handling high-dimensional cybersecurity datasets requiring intensive parallel computation.

Despite these improvements, several limitations were observed during experimentation. The computational advantages of SHC-PPSO were more significant for large-scale datasets compared to smaller datasets, where

communication overhead between nodes may slightly reduce efficiency gains. Furthermore, the framework's performance is influenced by parameter selection, including inertia weight, acceleration coefficients, and swarm size, which require careful tuning for different datasets. The dependence on high-performance computing infrastructure may also limit applicability in resource-constrained environments. Nevertheless, the overall results demonstrate that SHC-PPSO provides substantial improvements in scalability, runtime efficiency, and clustering accuracy compared with existing clustering approaches."

5.CONCLUSIONS

Therefore, the Scalable Hybrid Clustering Framework using Parallel Particle Swarm Optimization (SHC-PPSO) presented in this research work indicates novel advancements capable of enhancing data clustering in information systems. The implementation of the present research work also indicates that the effectiveness of SHC-PPSOis higher than the other conventional ones such as SCPSO_F1, SCPSO_F2 and PSOGSA in regards to accuracy factor, running time and speedup. Our methodology works in parallel mode to prevent long computation time needed for large data sets used while minimizing on accuracy. The high testing accuracy and the quick computational time for convergence in the case of two datasets, that is, HIGGS and CICIDS2017 imply the effectiveness and the capability of the framework as a useful tool in performing large scale data clustering problems. However these are some achievements, there are still a number of difficulties left. A major weakness of the current algorithm is that it depends on high-performance computing environments; thereby, it lacks portability in low resource originating areas or facilities. Further, it is indicated that there are many parameters that need to be optimised and this may require a delicate process of tuning the parameters, which may also be resource demanding besides requiring a lot of skill in modelling. They point out what directions there is potential for improvement and further enhancement of the algorithm in order to widen its potential real-life applications and make it seamlessly suitable for various environments and data sets. Future directions of development for SHC-PPSO include ameliorating these drawbacks to apply the identity to other domains. Using a learning model for grouping the various parameters forms one of the research areas which will work towards minimizing the amount of effort required on the part of the user to calibrate the parameters. Also, examining ways to minimize computational costs and enhance performance in less computer resource intensive environments would also be important. This could be done to optimize the algorithm for cloud-based applications and therefore making its use more flexible and not needing special hardware to be implemented.

The experimental results demonstrate that the proposed SHC-PPSO framework significantly improves clustering accuracy, scalability, and runtime efficiency compared with existing methods such as SCPSO-F1, SCPSO-F2, and PSOGSA. The integration of Parallel Particle Swarm Optimization, hybrid clustering, and Apache Spark distributed processing enables efficient handling of large-scale and high-dimensional datasets. The framework has potential applications in cybersecurity, healthcare analytics, and big data processing. However, the method remains sensitive to parameter tuning and depends on high-performance computing resources for optimal performance. Additionally, communication overhead in distributed environments may slightly reduce efficiency for smaller datasets. Despite these limitations, SHC-PPSO provides an effective and scalable solution for modern large-scale clustering problems.

The third disseminating direction is to extend the assessment of SHC-PPSO to more datasets with various features in the future. This will bring about the required potency of the devised algorithm across different domains as well as structures of data collection. Moreover, we found that incorporating SHC-PPSO with other superior clustering algorithms and machine learning algorithms could increase its effectiveness in different data mining tasks, which would be very satisfying. As it stands, SHC-PPSO has a high level of potential, which has been shown in this paper, but further research and enhancement must be done for the framework to reach its full potential. By overcoming existing deficiencies and broadening the topic areas where it can be utilized, SHC-PPSO can be transformed into more functional and versatile data clustering instrument for various scientific and industry domains.

REFERENCES

[1]. Sibalija TV. Particle swarm optimisation in designing parameters of manufacturing processes: A review (2008–2018). Applied Soft Computing. 2019 Nov 1; 84:105743.
[2]. Hafsi H, Gharsellaoui H, Bouamama S. Genetically modified Multi-objective Particle Swarm Optimization approach for high-performance computing workflow scheduling. Applied soft computing. 2022 Jun 1; 122:108791.
[3]. Islam MA, Gajpal Y, ElMekkawy TY. Hybrid particle swarm optimization algorithm for solving the clustered vehicle routing problem. Applied Soft Computing. 2021 Oct 1; 110:107655.
[4]. Kennedy J, Eberhart R. Particle swarm optimization. InProceedings of ICNN'95-international conference on neural networks 1995 Nov 27 (Vol. 4, pp. 1942-1948). IEEE.
[5]. Blackwell T, Kennedy J. Impact of communication topology in particle swarm optimization. IEEE Transactions on Evolutionary Computation. 2018 Nov 11; 23(4):689-702.
[6]. Zhang D, Ma G, Deng Z, Wang Q, Zhang G, Zhou W. A self-adaptive gradient-based particle swarm optimization algorithm with dynamic population topology. Applied Soft Computing. 2022 Nov 1; 130:109660.
[7]. Wang F, Zhang H, Li K, Lin Z, Yang J, Shen XL. A hybrid particle swarm optimization algorithm using adaptive learning strategy. Information Sciences. 2018 Apr 1; 436:162-77.
[8]. Wang D, Tan D, Liu L. Particle swarm optimization algorithm: an overview. Soft computing. 2018 Jan; 22:387-408.
[9]. Wu Q, Xiong F, Wang F, Xiong Y. Parallel particle swarm optimization on a graphics processing unit with application to trajectory optimization. Engineering Optimization. 2016 Oct 2; 48(10):1679-92.
[10]. Hung Y, Wang W. Accelerating parallel particle swarm optimization via GPU. Optimization Methods and Software. 2012 Feb 1; 27(1):33-51.
[11]. Zhu H, Wu CK, Koo CH, Tsang YT, Liu Y, Chi HR, Tsang KF. Smart healthcare in the era of internet-of-things. IEEE Consumer Electronics Magazine. 2019 Sep 1; 8(5):26-30.
[12]. Haleem A, Javaid M, Singh RP, Suman R. Medical 4.0 technologies for healthcare: Features, capabilities, and applications. Internet of Things and Cyber-Physical Systems. 2022 Jan 1; 2:12-30.
[13]. Song CW, Jung H, Chung K. RETRACTED ARTICLE: Development of a medical big-data mining process using topic modeling. Cluster Computing. 2019 Jan 16; 22(Suppl 1):1949-58.
[14]. Gia TN, Dhaou IB, Ali M, Rahmani AM, Westerlund T, Liljeberg P, Tenhunen H. Energy efficient fog-assisted IoT system for monitoring diabetic patients with cardiovascular disease. Future Generation Computer Systems. 2019 Apr 1; 93:198-211.
[15]. Saravanan M, Shubha R, Marks AM, Iyer V. SMEAD: A secured mobile enabled assisting device for diabetics monitoring. In2017 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS) 2017 Dec 17 (pp. 1-6). IEEE.
[16]. Ali F, Islam SR, Kwak D, Khan P, Ullah N, Yoo SJ, Kwak KS. Type-2 fuzzy ontology–aided recommendation systems for IoT–based healthcare. Computer Communications. 2018 Apr 1; 119:138-55.
[17]. Li H, Pu B, Kang Y, Lu CY. Research on massive ECG data in XGBoost. Journal of Intelligent & Fuzzy Systems. 2019 Jan 1; 36(2):1161-9.
[18]. Siddiqui SA, Zhang Y, Lloret J, Song H, Obradovic Z. Pain-free blood glucose monitoring using wearable sensors: Recent advancements and future prospects. IEEE reviews in biomedical engineering. 2018 Apr 2; 11:21-35.
[19]. Su BY, Enayati M, Ho KC, Skubic M, Despins L, Keller J, Popescu M, Guidoboni G, Rantz M. Monitoring the relative blood pressure using a hydraulic bed sensor system. IEEE Transactions on Biomedical Engineering. 2018 Jul 13; 66(3):740-8.
[20]. Hassantabar S, Stefano N, Ghanakota V, Ferrari A, Nicola GN, Bruno R, Marino IR, Ham douche K, Jha NK. CovidDeep: SARS-CoV-2/COVID-19 test based on wearable medical sensors and efficient neural networks. IEEE Transactions on Consumer Electronics. 2021 Nov 24; 67(4):244-56.

[21]. Interdonato R, Atzmueller M, Gaito S, Kanawati R, Largeron C, Sala A. Feature-rich networks: going beyond complex network topologies. Applied Network Science. 2019 Jan 31; 4(1):4.
[22]. He T, Hu L, Chan KC, Hu P. Learning latent factors for community identification and summarization. IEEE access. 2018 Jun 7; 6:30137-48.
[23]. Kandel A, Bunke H. Applied graph theory in computer vision and pattern recognition. Springer Science & Business Media; 2007 Mar 12.
[24]. Hu L, Chen Q, Qiao L, Du L, Ye R. Automatic detection of melanins and sebums from skin images using a generative adversarial network. Cognitive Computation. 2022 Sep; 14(5):1599-608.
[25]. Liao L, He X, Zhang H, Chua TS. Attributed social network embedding. IEEE Transactions on Knowledge and Data Engineering. 2018 Mar 27; 30(12):2257-70
[26]. Thomas J, Seo D, Sael L. Review on graph clustering and subgraph similarity based analysis of neurological disorders. International journal of molecular sciences. 2016 Jun 1; 17(6):862.
[27]. Chang L, Li W, Qin L, Zhang W, Yang S. $\mathsf {pSCAN} $: fast and exact structural graph clustering. IEEE Transactions on Knowledge and Data Engineering. 2017 Jan 10; 29(2):387-401.
[28]. Hu Y. The parallelism and application in data mining of BP algorithm. Chongqing: Chongqing University. 2003.
[29]. Cao J, Chen L. Fuzzy emotional semantic analysis and automated annotation of scene images. Computational intelligence and neuroscience. 2015 Jan 1; 2015:33-.
[30]. Guo ZH, Wu J, Lu HY, Wang JZ. A case study on a hybrid wind speed forecasting method using BP neural network. Knowledge-based systems. 2011 Oct 1; 24(7):1048-56.
[31]. Xu L, Wang Q, Chen J, Pan Y. Forcast for average velocity of debris flow based on bp neural network. Journal of Jilin University (Earth Science Edition). 2013 Jan; 43(1):186-91.
[32]. Zhang F, Li P, Hou ZG, Lu Z, Chen Y, Li Q, Tan M. sEMG-based continuous estimation of joint angles of human legs by using BP neural network. Neurocomputing. 2012 Feb 15; 78(1):139-48.
[33]. Cheng C, Cheng X, Dai N, Jiang X, Sun Y, Li W. Prediction of facial deformation after complete denture prosthesis using BP neural network. Computers in biology and medicine. 2015 Nov 1; 66:103-12
[34]. Zhang J, Qi L, Ji R, Wang H, Huang S, Wang P. Cotton diseases identification based on rough sets and BP neural network. Transactions of the Chinese Society of Agricultural Engineering. 2012 Jul 1; 28(7):161-7.
[35]. Pan R, Gao W, Liu J, Wang H. Automatic recognition of woven fabric pattern based on image processing and BP neural network. The Journal of the Textile Institute. 2011 Jan 1; 102(1):19-30.
[36]. Zhao Y, Zobel J. Searching with style: Authorship attribution in classic literature. InProceedings of the thirtieth Australasian conference on Computer science-Volume 62 2007 Jan 30 (pp. 59-68).
[37]. Wang J, Li X, Fe [cbreve] kan M, Zhou Y. Hermits–Hadamard-type inequalities for Riemann–Liouville fractional integrals via two kinds of convexity. Applicable Analysis. 2013 Nov 1;92(11):2241-53
[38]. Liu ML, Zhu MH. The port throughput forecast model based on the BP neural network. Journal of Systems Science. 2012; 20(4):88-91
[39]. Al-Sawwa J, Ludwig SA. Parallel particle swarm optimization classification algorithm variant implemented with Apache Spark. Concurrency and Computation: Practice and Experience. 2020 Jan 25; 32(2):e5451.
[40]. Kennedy J, Eberhart R. Particle swarm optimization. InProceedings of ICNN'95-international conference on neural networks 1995 Nov 27 (Vol. 4, pp. 1942-1948). IEEE.
[41]. Shi Y. Particle swarm optimization: developments, applications, and resources. InProceedings of the 2001 congress on evolutionary computation (IEEE Cat. No. 01TH8546) 2001 May 27 (Vol. 1, pp. 81-86). IEEE.
[42]. Dean J, Ghemawat S. MapReduce: simplified data processing on large clusters. Communications of the ACM. 2008 Jan 1; 51(1):107-13.
[43]. Zhang W, Huang Y. Using big data computing framework and parallelized PSO algorithm to construct the reservoir

dispatching rule optimization. Soft Computing. 2020 Jun; 24(11):8113-24
[44]. Hu MX. Intrusion detection algorithm based on BP neural network. Jisuanji Gongcheng/ Computer Engineering. 2012; 38(6).
[45]. Snir M. MPI--the Complete Reference: the MPI core. MIT press; 1998.
[46]. McNabb AW, Monson CK, Seppi KD. Parallel pso using mapreduce. In2007 IEEE Congress on Evolutionary Computation 2007 Sep 25 (pp. 7-14). IEEE.
[47]. Sadasivam GS, Selvaraj D. A novel parallel hybrid PSO-GA using MapReduce to schedule jobs in Hadoop data grids. In2010 Second World Congress on Nature and Biologically Inspired Computing (NaBIC) 2010 Dec 15 (pp. 377-382). IEEE.
[48]. Wang J, Yuan D, Jiang M. Parallel k-pso based on mapreduce. In2012 IEEE 14th International Conference on Communication Technology 2012 Nov 9 (pp. 1203-1208). IEEE.
[49]. Zaharia M, Xin RS, Wendell P, Das T, Armbrust M, Dave A, Meng X, Rosen J, Venkataraman S, Franklin MJ, Ghodsi A. Apache spark: a unified engine for big data processing. Communications of the ACM. 2016 Oct 28; 59(11):56-65.
[50]. Guo X, Chen S, Zhang Y, Li W. Service composition optimization method based on parallel particle swarm algorithm on spark. Security and Communication Networks. 2017 Jan 1; 2017.
[51]. Duan Q, Sun L, Shi Y. Spark clustering computing platform based parallel particle swarm optimizers for computationally expensive global optimization. InParallel Problem Solving from Nature–PPSN XV: 15th International Conference, Coimbra, Portugal, September 8–12, 2018, Proceedings, Part I 15 2018 (pp. 424-435). Springer International Publishing.
[52]. Sherar M, Zulkernine F. Particle swarm optimization for large-scale clustering on apache spark. In2017 IEEE Symposium Series on Computational Intelligence (SSCI) 2017 Nov 27 (pp. 1-8). IEEE.
[53]. Escalas JE, Bettman JR. You are what they eat: The influence of reference groups on consumers' connections to brands. Journal of consumer psychology. 2003 Jan 1;13(3):339-48.
[54]. Zimpel RR, Fleck MP. Quality of life in HIV-positive Brazilians: application and validation of the WHOQOL-HIV, Brazilian version. AIDS care. 2007 Aug 1;19(7):923-30.
[55]. Hobbs SK, Monsky WL, Yuan F, Roberts WG, Griffith L, Torchilin VP, Jain RK. Regulation of transport pathways in tumor vessels: role of tumor type and microenvironment. Proceedings of the National Academy of Sciences. 1998 Apr 14;95(8):4607-12.
[56]. Batalha N, Marmeleira J, Garrido N, Silva AJ. Does a water-training macrocycle really create imbalances in swimmers' shoulder rotator muscles. European journal of sport science. 2015 Feb 17;15(2):167-72.
[57]. Goldsmith H, Boeuf P. Digging beneath the iron triangle: The Chunnel with 2020 hindsight. Journal of Mega Infrastructure & Sustainable Development. 2019 Jan 2;1(1):79-93.
[58]. Smith MW, Gomez HL, Eales SA, Ciesla L, Boselli A, Cortese L, Bendo GJ, Baes M, Bianchi S, Clemens M, Clements DL. The Herschel reference survey: dust in early-type galaxies and across the Hubble sequence. The Astrophysical Journal. 2012 Mar 15;748(2):123.
[59]. Han H, Lu W, Qiao J. An adaptive Mult objective particle swarm optimization based on multiple adaptive methods. IEEE transactions on cybernetics. 2017 Apr 17;47(9):2754-67.
[60]. Silva A, Meira Jr W, Zaki MJ. Mining attribute-structure correlated patterns in large, attributed graphs. arXiv preprint arXiv:1201.6568. 2012 Jan 31.
[61]. Ruan Y, Fuhry D, Parthasarathy S. Efficient community detection in large networks using content and links. InProceedings of the 22nd international conference on World Wide Web 2013 May 13 (pp. 1089-1098).
[62]. Brandes U, Delling D, Gaertler M, Gorke R, Hoefer M, Nikoloski Z, Wagner D. On modularity clustering. IEEE transactions on knowledge and data engineering. 2007 Dec 26;20(2):172-88.
[63]. Flake GW, Tarjan RE, Tsioutsiouliklis K. Graph clustering and minimum cut trees. Internet Mathematics. 2004 Jan 1;1(4):385-408.
[64]. Görke R, Hartmann T, Wagner D. Dynamic graph clustering using minimum-cut trees. In Algorithms and Data Structures: 11th

International Symposium, WADS 2009, Banff, Canada, August 21-23, 2009. Proceedings 11 2009 (pp. 339-350). Springer Berlin Heidelberg.

[65]. Tripathi PK, Bandyopadhyay S, Pal SK. Multi-objective particle swarm optimization with time variant inertia and acceleration coefficients. Information sciences. 2007 Nov 15;177(22):5033-49.

[66]. Al Moubayed N, Petrovski A, McCall J. D2MOPSO: MOPSO based on decomposition and dominance with archiving using crowding distance in objective and solution spaces. Evolutionary computation. 2014 Mar 1;22(1):47-77.

[67]. Coello CA, Pulido GT, Lechuga MS. Handling multiple objectives with particle swarm optimization. IEEE Transactions on evolutionary computation. 2004 Jun 14;8(3):256-79.

[68]. Bianchi FM, Grattarola D, Alippi C. Spectral clustering with graph neural networks for graph pooling. In International conference on machine learning 2020 Nov 21 (pp. 874-883). PMLR.

[69]. Chen H, Yu Z, Yang Q, Shao J. Attributed graph clustering with subspace stochastic block model. Information Sciences. 2020 Oct 1; 535:130-41.