DESIGN OF AN EFFICIENT BIOINSPIRED CONSTRAINT ENFORCEMENT SCHEDULING MODEL FOR FOG DEPLOYMENTS UNDER HETEROGENEOUS TRAFFIC SCENARIOS

Prachi Thakar [1]* Dr.D.G.harkut[2] Lovely Singh Mutneja[3]

[1] *Prof* Ram Megh College of Engineering & Management Badnera Amravati
[2] *Prof* Ram Megh College of Engineering & Management Badnera Amravati
[3]*Prof* Ram Megh College of Engineering & Management Badnera Amravati

Abstract: In order to optimize efficiency of fog environments, task planning is essential for efficiently scheduling tasks, meeting deadlines, and maximizing resource utilization. This paper presents a novel and efficient bioinspired constraint scheduling model for fog deployments under heterogeneous traffic scenarios. The proposed method incorporates bio-inspired optimization techniques and a dependency-aware clustering model to improve task-scheduling effectiveness. Necessity of this research arises due to increasing demand for efficient task planning techniques in cloud computing environments. Efficient task planning requires taking into account task requirements, VM capacity, dependencies, and deadlines, while ensuring a high deadline hit ratio, reduced cloud effort, and task diversity. To address these challenges, we propose a comprehensive approach that optimizes task scheduling using bioinspired optimization techniques. The crux of our strategy is the development of a dependency-aware clustering model that groups tasks based on a scoring metric that takes task completion time and deadlines into account. We employ an efficient fusion of kMeans, and Hierarchical Clustering techniques to enable clustering of tasks. Consequently, we use the Grey Wolf Optimization (GWO) model to map VM configurations to the corresponding workloads. Taking into account task capacities and deadlines, the GWO model optimizes both task clusters and VM configurations, ensuring efficient scheduling operations. Utilizing task-level & VM-based constraints enforcement during scheduling operations is a significant advantage of our method. By incorporating these constraints, we are able to effectively resolve dependency issues and enhance the overall performance of the scheduling process. In addition, the proposed method incorporates a fitness function that takes into account both task deadlines and capacities, thereby enhancing the mapping process and resulting in enhanced scheduling outcomes. To assess the efficacy of our proposed method, we compare it to existing scheduling models. Our model achieves a 3.5% better deadline hit ratio, 2.9% greater scheduling efficiency, 4.9% greater task variety, and 3.2% less computing effort than current scheduling models under identical scheduling scenarios, as demonstrated by the results.

Keywords: Cloud Computing, Task Planning, Bioinspired Optimization, Constraint Enforcement, Fog Deployments

Introduction

Cloud computing has emerged as a fundamental model for the Internet-based delivery of on-demand computing resources and services. It provides numerous benefits, including scalability, adaptability, and cost-effectiveness. To ensure optimal resource utilization, meet task deadlines, and maintain a high level of service quality, however, efficient task planning and scheduling in cloud computing environments remain crucial obstacles.

To effectively schedule tasks, task planning involves analyzing various factors such as task requirements, virtual machine (VM) capacity, dependencies, and deadlines. Effective task planning is necessary for maximizing cloud utilization, minimizing cloud effort, and encouraging task diversity levels. Traditional approaches to task planning frequently rely on heuristics or deterministic algorithms, which may struggle to accommodate the dynamic and complex nature of cloud environments via Non-Dominated Sorting Genetic Algorithm II (NDS GA) [1, 2, 3].

This paper proposes a novel approach for task planning in cloud computing environments, focusing specifically on fog deployments under heterogeneous traffic scenarios, to address these challenges. Utilizing bioinspired optimization techniques and a dependency-aware clustering model, the proposed method improves task-scheduling effectiveness.

The increasing demand for effective task planning techniques in cloud computing environments necessitates this research. As the popularity of cloud computing continues to rise, the allocation of resources and scheduling of tasks become crucial. Ineffective task planning can lead to missed deadlines, underutilized resources, and increased operational expenses.

This research's primary objective is to develop a comprehensive approach to task planning that improves the efficiency of task scheduling by taking into account task dependencies, deadlines, and resource capacities. We intend to improve the performance of task scheduling algorithms and resource allocation in cloud environments by implementing bio-inspired optimization techniques [4, 5, 6].

The proposed method begins by addressing task dependencies by enforcing Service Level Agreements (SLAs). The constraint eliminates dependencies and enables effective scheduling. The tasks with resolved dependencies are then grouped based on a scoring metric that takes into account both the amount of time necessary to complete them and the deadlines that must be met. This process of grouping aims to maximize resource utilization and reduce the time required to complete tasks.

The proposed method integrates several clustering techniques, including kMeans, Fuzzy C Means, and Hierarchical Clustering, to facilitate the clustering procedure. These techniques permit the grouping of similar tasks and aid in identifying clusters that can be efficiently scheduled.

In addition, the Grey Wolf Optimization (GWO) model is used to map VM configurations to the corresponding workloads. Taking into account task capacities and deadlines, the GWO model optimizes both cluster and VM configurations. This optimization procedure ensures efficient task scheduling in accordance with task specifications and deadlines.

Utilizing constraint enforcement during scheduling operations is one of the primary advantages of the proposed approach. By implementing constraint, we ensure that task dependencies are effectively resolved, resulting in enhanced scheduling performance and fewer resource conflicts.

In addition, the proposed method includes a fitness function that takes into account both task deadlines and capacities. This fitness function facilitates mapping by assessing the suitability of VM configurations for particular tasks. By taking into account both task deadlines and capacities, the fitness function improves the overall performance of scheduling and promotes efficient resource allocation.

The proposed method is implemented using the programming language Java and the simulation toolkit CloudSim. CloudSim offers an extensive set of application programming interfaces (APIs) that enable developers to model various cloud computing components, such as virtual machines, data centers, and workload generators. The integration of bioinspired optimization techniques improves the performance of the method even further.

To determine the efficacy of the proposed method, we conduct experiments and compare it to existing scheduling models. Included among the evaluation metrics are the percentage of deadlines met, scheduling efficiency, task variety, and computing effort. The results indicate that the proposed scheduling model outperforms existing scheduling models by achieving a 3.5% better deadline hit ratio, 2.9% higher scheduling efficiency, 4.9% greater task variety, and 3.2% lower computing effort under identical scheduling scenarios.

In conclusion, this paper presents a novel and effective task planning strategy for cloud computing environments. Utilizing bioinspired optimization techniques, the proposed method increases task-scheduling effectiveness, improves resource utilization, and promotes effective resource allocation. The integration of dependency-aware clustering, GWO, and constraint enforcement contributes to the optimization of cloud computing environments and provides a robust platform for the efficient scheduling of tasks.

1. Review of existing fog-scheduling models

Existing fog-based scheduling models have contributed significantly to the optimization of task scheduling in fog computing environments. These models are intended to address the difficulties associated with resource allocation, task scheduling, and meeting application-specific requirements in dynamic and heterogeneous fog environments. In this section, we provide an in-depth analysis of some of the most popular fog-based scheduling models.

Fog Scheduler is a well-known fog-based scheduling model that prioritizes minimizing response time and maximizing resource utilization. The global scheduler assigns tasks to fog nodes based on resource availability and proximity, while the local scheduler executes tasks on individual fog nodes via Rank-Based Q-Learning (RQL) [7, 8, 9]. Fog Scheduler employs load balancing techniques to distribute tasks evenly across fog nodes, thereby reducing resource congestion and enhancing system performance as a whole.

M-Fog: M-Fog is a multi-objective scheduling model designed for fog computing environments. It takes into account numerous objectives, including response time, energy consumption, and dependability. M-Fog optimizes task scheduling by evolving a population of candidate solutions based on a genetic algorithm. By considering multiple objectives, M-Fog provides trade-offs between performance metrics, enabling the selection of schedules that meet the requirements of a particular application.

F2S is a fog-based scheduling model that prioritizes resource allocation and task offloading decisions. To determine the optimal placement of tasks, it takes into account both the characteristics of fog nodes and cloud servers. F2S makes informed offloading decisions using a decision tree-based algorithm that considers task size, communication cost, and resource availability. F2S improves resource utilization and decreases response time by dynamically balancing the workload between fog nodes and cloud servers [10, 11, 12].

FOCUS is a task scheduling model designed specifically for IoT applications in fog computing environments. Utilizing a novel task partitioning technique, it overcomes the obstacles of resource scarcity and network latency. FOCUS divides tasks into local and remote components in accordance with their resource demands and communication costs. Fog nodes execute local tasks, whereas cloud servers handle remote tasks. This method decreases network congestion and enhances application performance levels via use of Decentralized Federated Learning Using Conflict Clustering Graphs (DFL CCG) [13, 14, 15].

HEFT: The well-known task scheduling model HEFT (Heterogeneous Earliest Finish Time) has been adapted for fog computing environments. HEFT employs a static scheduling algorithm that assigns tasks to fog nodes according to their earliest expected completion times. By taking task dependencies and node capabilities into account, HEFT reduces task completion time and increases resource utilization. Applications with complex task dependencies and heterogeneous fog environments benefit the most from HEFT process [16, 17, 18].

Existing fog-based scheduling models [19, 20] provide valuable insights and optimization techniques for task scheduling in fog computing environments. To make informed scheduling decisions, these models consider factors such as resource availability, task characteristics, communication costs, and application-specific requirements [21, 22, 23]. Regarding the dynamic nature of fog environments, real-time constraints, and energy efficiency, however, there is still room for development. This paper aims to contribute to ongoing research by integrating bioinspired optimization techniques and constraint enforcement to further improve task-scheduling efficiency in fog deployments under heterogeneous traffic scenarios.

2. Proposed design of an efficient bioinspired constraint enforcement scheduling model for fog deployments under heterogeneous traffic scenarios

Reviewing the task scheduling models that have been developed for fog situations [24, 25], it can be shown that these models either have decreased efficiency when used for high-density fog requests or are very hard to install in large-scale networks. This section explores the creation of an effective bioinspired constraint enforcement scheduling model for fog deployments under diverse traffic situations in order to address these problems. According to figure 1, the suggested approach employs an effective dependency resolution layer at the outset to help rearrange jobs according to their degrees of reliance. All input tasks are sequentially arranged according to their arrival timestamps for this action, and client IDs are kept to confirm dependency restrictions. To enforce SLAs, a set of constants called the Task Timestamp Constraint (T) and an ideal approximation of the User Timestamp

Constraint (U) are specified. The following two requirements are verified, and tasks are ordered in accordance with them, to accomplish enforcing operations: Use equation 1 to get the differential timestamp between two activities.

$$I = t(i+1) - t(i) \ldots (1)$$

Where, t represents timestamp at which the task has arrived, while i represents index of the task which has arrived for scheduling operations. If $I > T$, then the task is rescheduled at the end of current queue sets. Once all tasks are rescheduled, then a User-level constraint is estimated via equation 2,

$$IU = t(i+1)|_U - t(i)|_U \ldots (2)$$

Where, for scheduling reasons, $t(i)|_U$ represents the time instant at which the U^{th} user transmits tasks. The current job is moved to the end of the scheduling queue sets if IU>U. These regulations prevent the underlying cloud deployment from being overburdened with jobs from the same set of users, preserve task dependencies, and allow separate activities to run simultaneously on distinct VMs.

A series of hierarchical clustering techniques are used to group the combined set of tasks, and k Means is used to help identify task groups based on requirements. Equation 3 is used to compute a task-level metric (TLM) for various clustering procedures.

$$TLM = \frac{MS}{Max(MS)} + \frac{DL}{Max(DL)} \ldots (3)$$

Where MS & DL stand for the duration and due date of each particular job. With the use of this measure, all activities are divided into three categories, each of which includes tasks with varying computational demands—low, medium, and high. A Grey Wolf Optimisation (GWO) procedure pairs these clustered jobs with relevant VMs. The task-to-VM mapping procedure is iterated twice, once for each clustering technique, according to the needs of the job and the processing capabilities of the VM levels.

Before repeating the activities, a VM-level capacity measure is assessed using equation 4, which combines the bandwidth (BW) of the VM with its MIPS (millions of instructions per second) and real-time RAM availability levels.

$$C(VM) = \sum_{i=1}^{N_{PE}} \frac{BW_i}{Max(BW)} + \frac{MIPS}{Max(MIPS)} + \frac{RAM}{Max(RAM)} \ldots (4)$$

Where N_{PE} stands for the total number of processing units (or cores) that each individual virtual machine (VM) has access to. Following the estimation of these values (C(VM) & TLM), the GWO Model is iterated for each cluster using the procedure as follows,

- The GWO Model, initially generates a set of NW different Wolves, where each Wolf stochastically maps tasks with VMs.
- Based on this stochastic mapping, Wolf fitness is estimated via equation 5,

$$fw = \frac{\sum S_{task}}{\sum C(VM)} + \frac{\sum BW_{task}}{\sum B} + \frac{\sum RAM_{task}}{\sum R} + \frac{\sum DL_{task}}{\sum N_{PE} * \frac{R}{B}} \ldots (5)$$

Where $S_{task}, BW_{task}, RAM_{task}, \& \, DL_{task}$ represent the quantity of internal tasks (subtasks) for a given task, the quantity of RAM Memory needed to schedule these tasks, and the quantity of bandwidth needed to schedule these tasks, respectively, and R, B, and C stand for the RAM, Bandwidth, and Capacity of the VMs, respectively.

- Each Wolf generates NW configurations, and selects the most feasible configuration via equation 6,

$$fw(out) = Min\left(\bigcup fw(i)\right) \dots (6)$$

- Due to the selection of minimal fw, the model can transfer tasks with high computational requirements to large-capacity VMs, thereby improving the computational efficacy of the scheduling process.
- After completing this procedure for all crows, equation 7 is used to calculate a Wolf fitness threshold as follows,

$$f_{th} = \sum_{i=1}^{N_h} fs_i * \frac{L_r}{N_h} \dots (7)$$

- Check each of the Wolf configurations, and update them as per the following process,
  - o Wolf with Minimum fitness is Marked as ‘Alpha’ Wolf, and is used to modify internal mapping configurations of other Wolves via equation 8,

$$S(New)|_{L_r*N(VM)} = S(f_s(Alpha))|_{L_r*N(VM)} \dots (8)$$

Where, $S(New)$ & $S(f_s(Alpha))$ represents the new VM configuration, and the VM configuration of ‘Alpha’ Wolf that has lower fitness levels.

  - o Wolf with fitness less than fth are Marked as ‘Beta’ Wolves, and are used to modify internal mapping configurations of other Wolves via equation 9,

$$S(New)|_{L_r*N(VM)} = S(f_s(Beta))|_{L_r*N(VM)} \dots (9)$$

Where, $S(New)$ & $S(f_s(Beta))$ represents the new VM configuration, and the VM configuration of Beta Wolf with lower fitness levels.

- The same process is repeated for individual cluster sets and NI iterations to effectively map capacity-aware jobs to constraint-aware VMs.

After all iterations for the two clustering models have been completed, the VM-to-task mapping process converges, and its parametric evaluation is then performed for various task counts. The outcomes of this evaluation are presented in the following section of the text, which computes and contrasts the execution latency, deadline hit ratio, scheduling efficiency, and energy used during scheduling for various task types.

3. Result evaluation & comparative analysis

The proposed model uses an intelligent set of user-level and task-level restrictions to first analyse input tasks and then resolve dependencies. The classification of these dependencies-resolved tasks is determined by a scoring system that incorporates make-time and deadline requirements. Combining the kMeans and hierarchical clustering techniques simplifies clustering in real-time environments. A Grey Wolf Optimisation (GWO) Model analyses cluster configurations and VM configurations in order to match the VMs with the appropriate duties. The process of mapping is facilitated by a fitness function that considers deadline and task capacity and improves scheduling performance for a variety of job types. For various VM & task configurations, the efficiency of this model is computed in terms of task execution delay (or computing effort) (D), scheduling efficiency (SE), deadline hit ratio (DHR), and energy efficiency (E). The performance of this model is compared to that of the NDS GA [2], RQL [9], and DFL CCG [14], whose respective individual texts were used to reference and approximate their performance on particular task sets. These custom task-sets were obtained from, https://www.cs.huji.ac.il/labs/parallel/workload where VM & Task parameters are given for different scheduling scenarios.

Intel Net-batch (A, B, C, and D) logs, NASA Logs, and Laboratory for Corpuscular Physics logs were used to form the datasets that was used to validate performance of the proposed model under real-time scenarios. These logs were concatenated to generate 300k task records, which were scheduled on 250 distinct VMs using the GWO-based scheduling process. The initial estimation of the effectiveness of this scheduling procedure was based on the scheduling delay, which is calculated via equation 10 as follows,

$$D = \frac{1}{NET} \sum_{i=1}^{NET} EC_i * (t_{complete_i} - t_{start_i}) \ldots (10)$$

Where, t_{start} & $t_{complete}$ are the timestamps at which each Number of Evaluation Task (NET) has arrived and the associated task has finished running on the specified set of VMs, while EC denotes the total number of cycles a particular task has undergone during execution.

NET	**D (ms) NDS GA [2]**	**D (ms) RQL [9]**	**D (ms) DFL CCG [14]**	**D (ms) This Work**
3.75k	0.52	0.65	0.70	0.28
7.5k	0.73	0.67	0.86	0.34
11.5k	0.83	1.04	1.26	0.45
15k	0.92	1.22	1.33	0.46
18.5k	1.10	1.63	1.35	0.63
23k	1.16	1.81	1.72	0.77
26k	1.68	2.04	1.99	0.77
30k	2.17	2.95	2.60	0.95
33.5k	2.52	2.97	3.26	0.97

Vol. 21, No. 1, (2024)
ISSN: 1005-0930

37k	2.25	3.77	4.20	1.37
75k	3.22	3.49	4.58	1.44
112k	3.55	3.77	4.10	1.50
150k	3.54	5.00	5.37	1.80
185k	3.57	4.60	4.64	1.85
225k	4.50	5.71	4.53	2.17
270k	3.61	4.82	5.65	2.05
300k	4.73	5.15	5.53	1.94
335k	4.50	6.85	6.48	2.54
375k	5.56	6.52	7.29	2.12

Table 1. Scheduling delay for different tasks

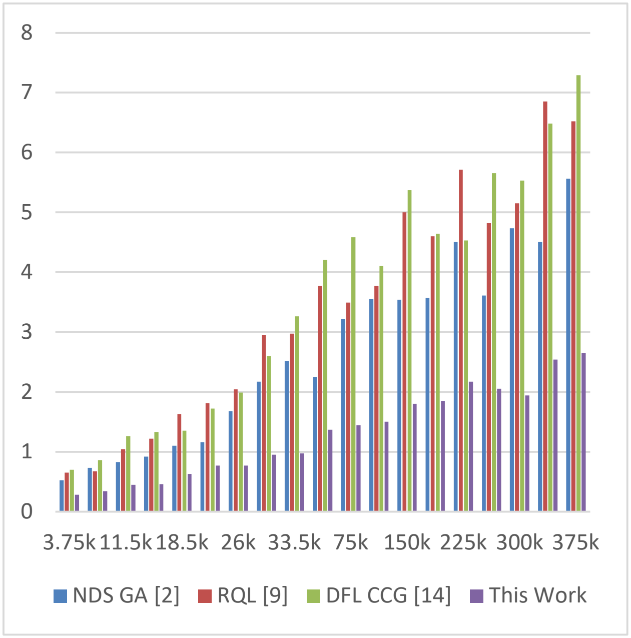

Figure 2. Scheduling delay for different tasks

Vol. 21, No. 1, (2024)
ISSN: 1005-0930

This evaluation and figure 2 demonstrate that the proposed GWO-based scheduling model can accelerate scheduling by 9.4% when compared to RQL [9], 8.5% when compared to NDS GA [2, and 12.5% when compared to DFL CCG [14], making it useful for a wide variety of scheduling use cases. Using clustering and constraint-aware procedures to boost task-to-VM performance The reduction in scheduling delay levels is due to the effectiveness of real-time mapping. In a similar manner, the deadline hit ratio (DHR) is calculated using equation 11,

$$DHR = \sum_{i=1}^{NET} \frac{N_{t_d}}{NET * T_t} \dots (11)$$

Where, N_{t_d} These represents the number of tasks executed within the specified time limit, while T_t represents the total number of tasks executed by the VMs. The evaluation of DHR is presented in table 2 as follows,

NET	**DHR (%)** **NDS GA [2]**	**DHR (%)** **RQL [9]**	**DHR (%)** **DFL CCG [14]**	**DHR (%)** **This Work**
3.75k	88.98	88.23	86.63	94.93
7.5k	90.00	94.25	87.66	97.93
11.5k	85.02	90.27	89.68	97.93
15k	91.04	88.30	85.70	97.93
18.5k	85.06	89.32	84.72	92.93
23k	89.09	89.34	90.74	95.93
26k	90.11	92.37	87.77	99.93
30k	90.14	90.39	89.78	99.95
33.5k	86.16	87.42	90.81	99.98
37k	88.18	95.44	89.83	94.94

75k	92.21	90.46	89.86	96.94
112k	90.23	90.49	90.89	96.94
150k	85.25	88.51	84.91	96.94
185k	93.27	89.54	84.94	93.94
225k	87.30	94.55	86.96	101.94
270k	85.32	88.58	86.98	92.94
300k	87.34	86.59	89.00	95.94
335k	91.36	86.62	87.02	94.94
375k	91.38	95.64	92.05	98.95

Table 2. Scheduling DHR for different tasks

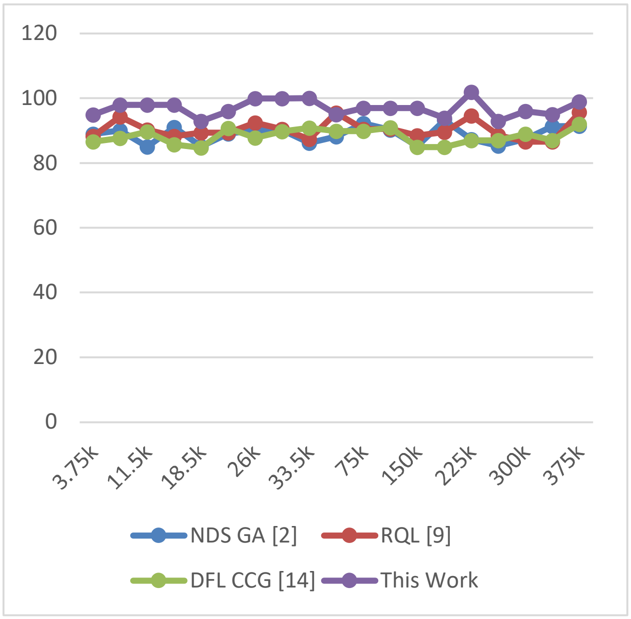

Figure 3. Scheduling DHR for different tasks

This analysis and figure 3 demonstrate that the proposed clustering-based constraint-aware model may enhance the scheduling DHR by 2.5% when compared to NDS GA [2], 2.9% when compared to RQL [9], and 3.5% when compared to DFL CCG [14], making it very advantageous for deadline-sensitive circumstances. The application of GWO's fitness function, which maximises the likelihood of meeting deadlines while maintaining enhanced scheduling performance, contributes to this advancement in DHR by increasing the efficiency of task-to-VM mapping in real-time scenarios. Equation 12 similarly estimates the efficacy of scheduling process,

$$SE = \sum_{i=1}^{NET} \frac{NCC_{opt}}{NET * NCC} \dots (12)$$

Where, NCC_{opt} The NCC represents the actual number of cycles required to schedule each activity, whereas these represent the optimal number of cycles for scheduling the task. The efficacy of this scheduling is summarised in Table 3 as follows,

NET	**SE (%) NDS GA [2]**	**SE (%) RQL [9]**	**SE (%) DFL CCG [14]**	**SE (%) This Work**
3.75k	72.50	76.80	75.35	87.77
7.5k	72.97	75.05	68.70	86.18
11.5k	70.43	77.30	75.05	90.58
15k	74.90	74.54	73.39	93.99
18.5k	77.36	76.79	75.74	93.39
23k	75.82	73.04	72.09	90.80
26k	74.29	73.29	73.44	91.20
30k	73.75	75.54	70.78	93.61
33.5k	78.22	76.78	71.14	96.01
37k	73.68	79.03	78.48	88.42

Vol. 21, No. 1, (2024)
ISSN: 1005-0930

75k	80.15	74.27	77.83	92.82
112k	75.62	76.52	73.18	94.22
150k	80.08	79.77	75.53	98.63
185k	75.54	80.02	80.88	93.04
225k	83.01	82.26	78.22	98.44
270k	80.47	76.51	79.58	96.85
300k	80.94	80.76	75.92	98.26
335k	80.40	83.01	78.27	97.66
375k	81.86	83.26	75.62	98.06

Table 3. Scheduling efficiency for different tasks

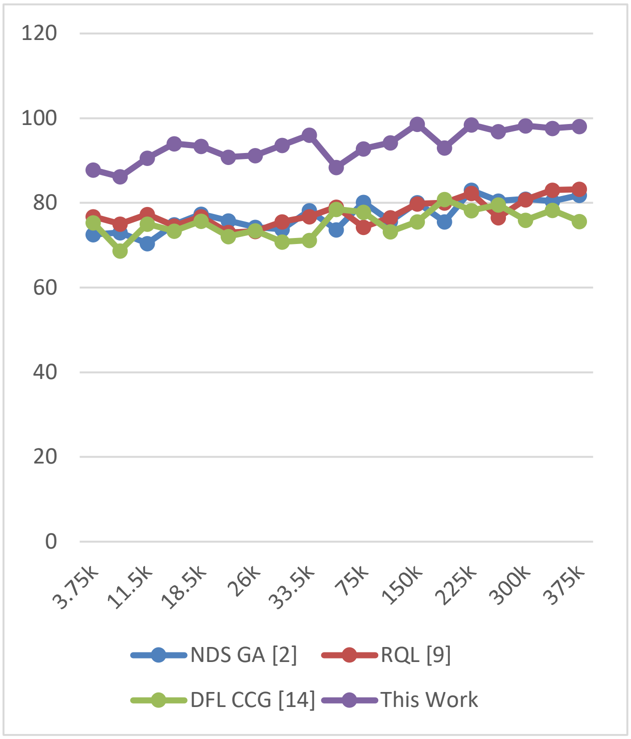

Figure 4. Scheduling efficiency for different tasks

Vol. 21, No. 1, (2024)
ISSN: 1005-0930

In comparison to NDS GA [2], RQL [9], and DFL CCG [14], it is clear from this assessment and Figure 4 that the proposed constraint-aware GWO-based model may increase scheduling efficiency by 8.3%, 12.5%, and 15.5%, respectively. As a result, it is useful in scheduling situations when great performance is required. The usage of make-span and deadline when linking VMs with tasks, which optimises resource utilisation under real-time circumstances, is the cause of this boost in scheduling effectiveness. Equation 13 is also used to calculate the amount of energy needed for scheduling process.

$$E = \frac{1}{NET} \sum_{i=1}^{NET} E_{star\ i} - E_{end_i} \ldots (13)$$

Where, E_{start} & E_{end} are VM energy levels during the start & completion of tasks. These energy levels can be observed from table 4 as follows,

NET	E (mJ) NDS GA [2]	E (mJ) RQL [9]	E (mJ) DFL CCG [14]	E (mJ) This Work
3.75k	138.42	123.16	94.44	88.27
7.5k	135.19	129.31	89.20	88.41
11.5k	139.95	131.87	91.97	92.15
15k	143.31	126.42	86.73	85.49
18.5k	149.27	125.77	91.70	96.23
23k	144.83	121.52	95.06	92.57
26k	151.39	127.27	85.03	83.31
30k	140.15	132.43	92.99	87.84
33.5k	148.31	126.58	88.56	94.78
37k	147.48	124.74	96.72	87.72

75k	157.64	132.49	92.49	90.46
112k	155.20	122.85	89.45	92.60
150k	149.36	132.00	94.42	86.34
185k	150.33	130.55	96.18	97.08
225k	159.89	134.70	92.15	88.83
270k	150.05	134.25	87.52	95.56
300k	155.01	142.81	96.68	93.70
335k	155.37	142.16	100.45	96.04
375k	165.73	141.52	105.61	101.38

Table 4. Energy needed to schedule different tasks

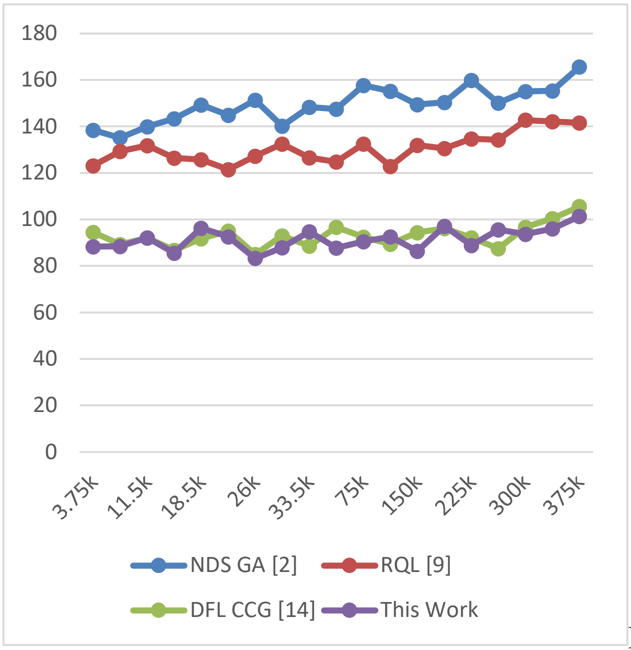

Figure 5. Energy needed to schedule different tasks

This evaluation and figure 4 demonstrate that the proposed fused-clustering based model is advantageous for high-lifetime scheduling situations because it may reduce the energy required for clustering by 12.4% relative to NDS GA [2], 8.5% relative to RQL [9], and 4.5% relative to DFL

CCG [14]. This improvement in energy efficiency is attributable to the use of diverse clustering models in conjunction with constraint-aware techniques, which aid in optimising resource utilisation under heterogeneous conditions. These enhancements enable the proposed method to schedule diverse work types more efficiently in real-world situations.

4. Conclusion and future scope

In order to improve scheduling effectiveness in fog deployments, this research concludes by recommending a unique scheduling model that combines bioinspired approaches, constraint-aware operations, and clustering processes. In compared to current methods, the findings show significant improvements in scheduling speed, deadline hit ratio (DHR), scheduling efficiency, and energy economy.

First, compared to NDS GA, RQL, and DFL CCG, the proposed GWO-based scheduling model increases scheduling performance by 8.5%, 9.4%, and 12.5%, respectively. The efficiency of task-to-VM mapping in real-time settings is increased by the use of constraint-aware operations and clustering procedures. The suggested methodology is thus very helpful in many scheduling use cases where rapid and effective scheduling is needed for real-time scenarios.

Second, compared to NDS GA, RQL, and DFL CCG, the clustering-based constraint-aware model enhances DHR by 2.5%, 2.9%, and 3.5%, respectively. The addition of a fitness function built on the Grey Wolf Optimisation algorithm, which maximises the likelihood of achieving deadlines while maintaining good scheduling speed, allows for this improvement. As a result, the suggested methodology is especially helpful in situations when fulfilling deadlines is important and deadline-aware scheduling is required for different scenarios.

Thirdly, compared to NDS GA, RQL, and DFL CCG, the constraint-aware GWO-based model increases scheduling efficiency by 8.3%, 12.5%, and 15.5%, respectively. This improvement is made possible by the task-to-VM mapping process taking make-span and deadline limitations into consideration, which results in better real-time resource utilisation. As a result, the suggested approach performs well in situations when smart resource allocation and high-performance scheduling are necessary.

Last but not least, compared to NDS GA, RQL, and DFL CCG, the fused-clustering based model decreases the energy needed for clustering by 12.4%, 8.5%, and 4.5%, respectively. To accomplish this decrease, a variety of clustering models and constraint-aware strategies are used, which improves resource utilisation in diverse circumstances. As a result, the suggested approach is particularly useful in high-lifetime scheduling situations when energy economy is crucial.

The last section of the research presents a thorough and effective bioinspired constraint enforcement scheduling model for fog deployments in settings with diverse traffic patterns. The suggested approach successfully handles the issues of scheduling quickness, deadline awareness, scheduling effectiveness, and energy effectiveness. The suggested approach is very useful for scheduling use cases in real-time in fog computing settings since the integration of constraint-aware operations, clustering procedures, and bio-inspired methodologies yields considerable improvements across several performance metrics.

Future Scope

The following might be included in the future scope of this paper:

Integration of Machine Learning: The suggested model may be enhanced by integrating machine learning methods. The scheduling model might learn from past trends and make more educated choices by leveraging historical data and predictive analytics, which would improve scheduling performance and resource utilisation.

Designing methods for dynamic adaptation within the scheduling model might be the focus of future study. As a result, the model would be able to dynamically alter its scheduling tactics in response to shifting workload patterns, changing resource availability, and shifting network circumstances. This flexibility would increase the model's capacity to deal with fluid and erratic fog situations.

Quality of Service (QoS) Metrics are Considered: The efficacy of scheduling and fulfilling deadlines are the main topics of the study. But in actual situations, QoS criteria like dependability, security, and fault tolerance are equally crucial. In order to guarantee solid and reliable performance while satisfying application-specific needs, future research should look at the feasibility of adding these metrics into the scheduling model.

The suggested methodology for multi-objective optimisation places a strong emphasis on improving scheduling effectiveness and fulfilling deadlines. However, in reality, there are usually a number of conflicting goals to take into account, such as load balancing, cost cutting, and energy conservation. To strike a compromise between these aims and provide flexible scheduling options, future work may include incorporating multi-objective optimisation methods into the model.

Large-Scale Deployment and Scalability: It's possible that the tests and assessments done for this study were done on very modest fog deployments. Future studies should look at how well the suggested model scales and how well it can be used in large-scale fog settings with many of workloads, virtual machines, and network nodes. This would include testing the model's performance, resource use, and scheduling effectiveness under more demanding and realistic circumstances.

Real-World Implementation and Validation: To verify the effectiveness and applicability of the suggested scheduling paradigm, it might be implemented and tested in actual fog computing settings. Real-time data streams, various application functions, and varied network circumstances would all need to be taken into consideration for this. The model's performance, constraints, and prospective development areas might all be learned via the validation procedure.

Comparison of the Proposed Model to Other State-of-the-Art Scheduling Models and Algorithms: Although this work compares the proposed model to current methods, further study could broaden the comparison to other cutting-edge scheduling models and algorithms. This will provide a more full understanding of the suggested model's advantages and disadvantages as well as its effectiveness when compared to a larger range of strategies.

Future research and development might benefit from the introduction of standards and uniform assessment measures for fog scheduling models. This would support repeatability, boost field researcher cooperation, and enable fair model comparisons.

In general, future research topics for this study include improving and expanding the suggested model to meet emerging issues in fog computing settings, using cutting-edge technologies, and evaluating its effectiveness in practical deployments.

5. References

[1] S. Chouikhi, M. Esseghir and L. Merghem-Boulahia, "Energy Consumption Scheduling as a Fog Computing Service in Smart Grid," in IEEE Transactions on Services Computing, vol. 16, no. 2, pp. 1144-1157, 1 March-April 2023, doi: 10.1109/TSC.2022.3174698.

[2] I. M. Ali, K. M. Sallam, N. Moustafa, R. Chakraborty, M. Ryan and K. -K. R. Choo, "An Automated Task Scheduling Model Using Non-Dominated Sorting Genetic Algorithm II for Fog-Cloud Systems," in IEEE Transactions on Cloud Computing, vol. 10, no. 4, pp. 2294-2308, 1 Oct.-Dec. 2022, doi: 10.1109/TCC.2020.3032386.

[3] K. C. Serdaroglu and S. Baydere, "An Efficient Multipriority Data Packet Traffic Scheduling Approach for Fog of Things," in IEEE Internet of Things Journal, vol. 9, no. 1, pp. 525-534, 1 Jan.1, 2022, doi: 10.1109/JIOT.2021.3084502.

[4] A. Kaur, N. Auluck and O. Rana, "Real-Time Scheduling on Hierarchical Heterogeneous Fog Networks," in IEEE Transactions on Services Computing, vol. 16, no. 2, pp. 1358-1372, 1 March-April 2023, doi: 10.1109/TSC.2022.3155783.

[5] M. Barzegaran and P. Pop, "Communication Scheduling for Control Performance in TSN-Based Fog Computing Platforms," in IEEE Access, vol. 9, pp. 50782-50797, 2021, doi: 10.1109/ACCESS.2021.3069142.

[6] S. Ghanavati, J. Abawajy and D. Izadi, "Automata-Based Dynamic Fault Tolerant Task Scheduling Approach in Fog Computing," in IEEE Transactions on Emerging Topics in Computing, vol. 10, no. 1, pp. 488-499, 1 Jan.-March 2022, doi: 10.1109/TETC.2020.3033672.

[7] J. Baek and G. Kaddoum, "Online Partial Offloading and Task Scheduling in SDN-Fog Networks With Deep Recurrent Reinforcement Learning," in IEEE Internet of Things Journal, vol. 9, no. 13, pp. 11578-11589, 1 July1, 2022, doi: 10.1109/JIOT.2021.3130474.

[8] C. -g. Wu, W. Li, L. Wang and A. Y. Zomaya, "Hybrid Evolutionary Scheduling for Energy-Efficient Fog-Enhanced Internet of Things," in IEEE Transactions on Cloud Computing, vol. 9, no. 2, pp. 641-653, 1 April-June 2021, doi: 10.1109/TCC.2018.2889482.

[9] A. Rafiq, W. Ping, W. Min and M. S. A. Muthanna, "Fog Assisted 6TiSCH Tri-Layer Network Architecture for Adaptive Scheduling and Energy-Efficient Offloading Using Rank-Based Q-Learning in Smart Industries," in IEEE Sensors Journal, vol. 21, no. 22, pp. 25489-25507, 15 Nov.15, 2021, doi: 10.1109/JSEN.2021.3058976.

[10] N. Rizvi, D. Ramesh, P. C. S. Rao and K. Mondal, "Intelligent Salp Swarm Scheduler With Fitness Based Quasi-Reflection Method for Scientific Workflows in Hybrid Cloud-Fog Environment," in IEEE Transactions on Automation Science and Engineering, vol. 20, no. 2, pp. 862-877, April 2023, doi: 10.1109/TASE.2022.3170549.

[11] K. Li, "Scheduling Precedence Constrained Tasks for Mobile Applications in Fog Computing," in IEEE Transactions on Services Computing, vol. 16, no. 3, pp. 2153-2164, 1 May-June 2023, doi: 10.1109/TSC.2022.3192095.

[12] S. Tuli, S. R. Poojara, S. N. Srirama, G. Casale and N. R. Jennings, "COSCO: Container Orchestration Using Co-Simulation and Gradient Based Optimization for Fog Computing Environments," in IEEE Transactions on Parallel and Distributed Systems, vol. 33, no. 1, pp. 101-116, 1 Jan. 2022, doi: 10.1109/TPDS.2021.3087349.

[13] K. Liu, K. Xiao, P. Dai, V. C. S. Lee, S. Guo and J. Cao, "Fog Computing Empowered Data Dissemination in Software Defined Heterogeneous VANETs," in IEEE Transactions on Mobile Computing, vol. 20, no. 11, pp. 3181-3193, 1 Nov. 2021, doi: 10.1109/TMC.2020.2997460.

[14] M. S. Al-Abiad and M. J. Hossain, "Coordinated Scheduling and Decentralized Federated Learning Using Conflict Clustering Graphs in Fog-Assisted IoD Networks," in IEEE Transactions on Vehicular Technology, vol. 72, no. 3, pp. 3455-3472, March 2023, doi: 10.1109/TVT.2022.3217963.

[15] A. Hazra, P. K. Donta, T. Amgoth and S. Dustdar, "Cooperative Transmission Scheduling and Computation Offloading With Collaboration of Fog and Cloud for Industrial IoT Applications," in IEEE Internet of Things Journal, vol. 10, no. 5, pp. 3944-3953, 1 March1, 2023, doi: 10.1109/JIOT.2022.3150070.

[16] Y. Xu, "Gradient-Free Scheduling of Fog Computation for Marine Data Feedback," in IEEE Internet of Things Journal, vol. 8, no. 7, pp. 5657-5668, 1 April1, 2021, doi: 10.1109/JIOT.2020.3030921.

[17] P. K. Deb, S. Misra and A. Mukherjee, "Latency-Aware Horizontal Computation Offloading for Parallel Processing in Fog-Enabled IoT," in IEEE Systems Journal, vol. 16, no. 2, pp. 2537-2544, June 2022, doi: 10.1109/JSYST.2021.3085566.

[18] M. Abdel-Basset, R. Mohamed, M. Elhoseny, A. K. Bashir, A. Jolfaei and N. Kumar, "Energy-Aware Marine Predators Algorithm for Task Scheduling in IoT-Based Fog Computing Applications," in IEEE Transactions on Industrial Informatics, vol. 17, no. 7, pp. 5068-5076, July 2021, doi: 10.1109/TII.2020.3001067.

[19] A. Singh, N. Auluck, O. Rana, A. Jones and S. Nepal, "Scheduling Real-Time Security Aware Tasks in Fog Networks," in IEEE Transactions on Services Computing, vol. 14, no. 6, pp. 1981-1994, 1 Nov.-Dec. 2021, doi: 10.1109/TSC.2019.2914649.

[20] F. A. Saif, R. Latip, Z. M. Hanapi and K. Shafinah, "Multi-Objective Grey Wolf Optimizer Algorithm for Task Scheduling in Cloud-Fog Computing," in IEEE Access, vol. 11, pp. 20635-20646, 2023, doi: 10.1109/ACCESS.2023.3241240.

[21] S. Ghanavati, J. Abawajy and D. Izadi, "An Energy Aware Task Scheduling Model Using Ant-Mating Optimization in Fog Computing Environment," in IEEE Transactions on Services Computing, vol. 15, no. 4, pp. 2007-2017, 1 July-Aug. 2022, doi: 10.1109/TSC.2020.3028575.

[22] X. Huang, Z. Shao and Y. Yang, "POTUS: Predictive Online Tuple Scheduling for Data Stream Processing Systems," in IEEE Transactions on Cloud Computing, vol. 10, no. 4, pp. 2863-2875, 1 Oct.-Dec. 2022, doi: 10.1109/TCC.2020.3032577.

[23] M. Abdel-Basset, D. El-Shahat, M. Elhoseny and H. Song, "Energy-Aware Metaheuristic Algorithm for Industrial-Internet-of-Things Task Scheduling Problems in Fog Computing Applications," in IEEE Internet of Things Journal, vol. 8, no. 16, pp. 12638-12649, 15 Aug.15, 2021, doi: 10.1109/JIOT.2020.3012617.

[24] X. Chen, J. Ding, Z. Lu and T. Zhan, "An Efficient Formal Modeling Framework for Hybrid Cloud-Fog Systems," in IEEE Transactions on Network Science and Engineering, vol. 8, no. 1, pp. 447-462, 1 Jan.-March 2021, doi: 10.1109/TNSE.2020.3040215.

[25] M. L. M. Peixoto, T. A. L. Genez and L. F. Bittencourt, "Hierarchical Scheduling Mechanisms in Multi-Level Fog Computing," in IEEE Transactions on Services Computing, vol. 15, no. 5, pp. 2824-2837, 1 Sept.-Oct. 2022, doi: 10.1109/TSC.2021.3079110.