**DADO FOR EFFICIENT DATA MIGRATION IN CLOUD INSTANCES**

S Durgaprasad1

sdpaug11@gmail.com

Chebolu SriCharan3

Department of Computer Science and Engineering,

Geethanjali College of Engineering and Technology, Department of CSE, Hyderabad,

India

charanchebolu7310@gmail.com

Arif Mohammad Abdul2

arif.software@gmail.com

Satyam Das4

Department of Computer Science and Engineering,

Geethanjali College of Engineering and Technology, Department of CSE, Hyderabad,

India

satyamdas0104@gmail.com

***Abstract***-**Cloud computing has transformed how computing resources are managed and utilized, delivering benefits such as optimized infrastructure use, increased flexibility, and faster deployment times. Despite these advantages, achieving seamless interoperability across diverse cloud environments remains a significant challenge, especially in terms of ensuring smooth resource sharing and access in heterogeneous systems. Containers have emerged as a lightweight virtualization solution that enhances scalability, portability, and flexibility in cloud services. The proposed solution employs the Adaptive Dragonfly Optimization (ADrO) algorithm for efficient data migration. While the traditional Dragonfly optimization algorithm is prone to local optima trapping due to loss of population diversity, this limitation is addressed by integrating a Levy flight strategy, enhancing population diversity and accelerating convergence toward optimal solutions. In the migration process, user tasks are collected, organized, and assigned to containers, ensuring efficient resource allocation. Additionally, load prediction is performed using an Actor-Critic Neural Network (ACNN) to further optimize migration decisions, accounting for predicted load and system capacity. A multi-objective function is developed to evaluate migration based on parameters such as predicted load transmission cost, demand, resource capacity, agility, reputation, migration time, and energy consumption.**

**Keywords— Container-based heterogeneous cloud, Data migration, Adaptive Dragonfly Optimization, Actor-Critic Neural Network**

1. **Introduction**

Cloud computing has become a vital component of modern technology, offering scalable and flexible solutions for a wide range of computational needs. However, managing data migration in heterogeneous cloud environments remains a complex challenge due to interoperability issues, dynamic workloads, and varying resource capacities. The project "DADO for Efficient Data Migration in Cloud Instances" aims to address these challenges by introducing an optimized data migration framework. The proposed solution leverages Adaptive Dragonfly Optimization (ADrO) combined with Levy flight strategies to avoid local optima trapping and ensure efficient migration. Additionally, Actor-Critic Neural Networks (ACNN) are used to predict system load and facilitate informed migration decisions. The project develops a multi-objective function to evaluate migration based on factors such as transmission cost, demand, resource capacity, and energy consumption. By integrating containerization for dynamic resource allocation, the system ensures improved scalability and adaptability to workload fluctuations. The implementation is designed to reduce downtime, improve energy efficiency, and enhance overall system performance.

1. **Related Work**

Research has been conducted in related fields, as evidenced by the studies that are reported in [1] to [7]. In this article, we discuss some recently published multi-objective problem-based data migration scheduling strategies in container-enabled heterogeneous cloud environments.

Tej.C. Hiremath et al. [1] suggested an energy-efficient data migration approach using a hybrid Taylor Lion-based Poor and Rich Optimization (Taylor Lion-based PRO) algorithm in a heterogeneous cloud. An Actor-Critic Neural Network (ACNN) predicts load to improve migration decisions. The approach optimizes migration time, energy consumption, and resource capacity while enhancing interoperability.

Manisha Malhotra et al. [2] proposes an Optimal Meta-Heuristic Elastic Scheduling (OMES) approach using the Artificial Bee Colony algorithm and flower pollination for VM resource allocation.The simulation on 1000 VMs shows improved resource utilization and reduced network traffic. The proposed method enhances load balancing.

Mohammad Shehab et al. [3] provided a comprehensive review of the Dragonfly Algorithm (DA), highlighting its effectiveness in solving various optimization problems in fields like engineering design, medical applications, image processing, power and energy systems, and economic load dispatch.

Muhammad Zakarya et al. [4] examines resource management in hybrid cloud platforms used by companies like Google, Rackspace, and AWS. The paper suggests that effective workload-aware resource management can improve energy and cost savings without compromising workload performance in hybrid cloud setups.

Mainak Adhikaria and Satish Narayana Sriramab et al. [5] propose an Energy-Efficient Container-based Scheduling (EECS) approach using Accelerated Particle Swarm Optimization (APSO) to optimize IoT and non-IoT task execution in cloud environments. It improves processing time, energy consumption, and resource utilization but lacks reliability due to unaddressed QoS trade-offs.

The paper proposed by Tarik A Rashid et al. [6] reviews the Dragonfly Algorithm (DA), its variants, and hybrid forms. It compares DA with other algorithms like PSO and GA, showing better exploration and convergence rates. DA’s applications in machine learning, image processing, and networking are discussed. The study highlights DA’s strengths, weaknesses, and potential improvements.

Cigdem Aci and Hakan Gulcan et al. [7] proposes a modified Dragonfly Algorithm (DA) using Brownian motion to improve the randomization stage. It addresses the limitations of Levy flight and enhances search efficiency. The modified DA showed up to 90% improvement over the original algorithm in solving optimization problems.

1. **SYSTEM MODEL**

Cloud computing provides fast access to shared software and hardware resources via the internet. Virtualization handles cloud resources, where VMs deliver services and containers enhance deployment and resource management. Efficient resource allocation based on user demand remains a key challenge. Containers start quickly and use fewer resources, ensuring better synchronization between service providers and users. VM performance depends on factors like memory, processing power, and storage capacity. Inefficient allocation can reduce system performance, highlighting the need for effective migration strategies. This model aims to optimize resource allocation across containers in a cloud platform consisting of multiple physical machines (Pmac’s).The physical mahines are represented as Pmac, virtual machines as VT Mac and the containers are represented as Cont in the Figure1 .

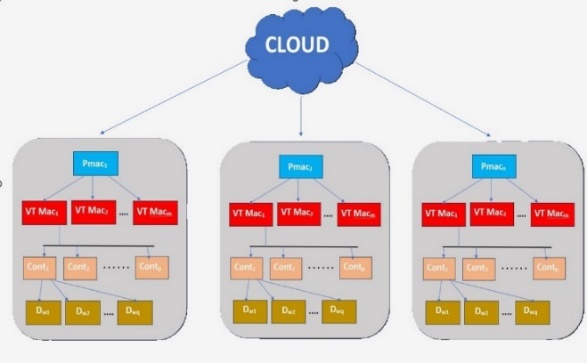


Figure 1: Cloud system model

1. **PROPOSED METHODOLOGY**

This research focuses on enhancing data migration in containerized heterogeneous cloud environments by improving resource allocation and migration efficiency. An Actor-Critic Neural Network (ACNN) predicts workload patterns, supplying key inputs to the proposed Adaptive Dragonfly Optimization (ADrO) algorithm. ADrO incorporates a Levy flight mechanism to enhance the migration process, minimizing energy use, transmission expenses, and migration time. This approach boosts system operational reputation, and resource utilization. By predicting workload patterns accurately, the system adapts more efficiently to dynamic workloads. Ultimately, this improves overall migration performance and cloud service efficiency.

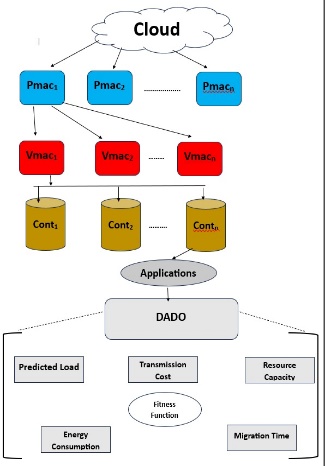
****

Figure 2: Framework Architecture

**4.1. Multi Objective problem design**

The study introduces a multi-objective function aimed at optimizing cost, migration time, and energy consumption during data migration in cloud environments. It factors in predicted load, transmission costs, resource demand, and migration time while considering system agility, reputation, and capacity. The goal is to balance these variables to ensure efficient workload migration and resource utilization. By optimizing this balance, the system enhances overall cloud performance and reduces operational overhead. This approach ensures that data migration is both cost-effective and time-efficient.

The Actor-Critic Neural Network (ACNN) uses reinforcement learning to predict workload patterns and improve migration decisions. The actor generates control policies based on estimated loads and adjusts them using Temporal Difference (TD) errors to enhance performance. The critic evaluates these policies by estimating value functions based on environmental feedback. Policy updates follow a gradient-based approach for stable convergence, supported by a Gaussian probability distribution for a stochastic strategy. Eligibility traces further improve learning efficiency by assigning credit or blame for TD errors.

Transmission cost reflects the expense of data movement within containers, calculated by the ratio of transmitted workloads to the total number of workloads. Migration time measures how long it takes to transfer data between cloud locations, depending on data size and migration efficiency. Energy consumption reflects the power used during migration, influenced by the number of VMs/containers and their resource utilization. Both active and idle states contribute to the total energy cost. Optimizing these factors improves overall system efficiency and resource management.

**4.2. Multi-Objective Data Migration Using Deep Adaptive Dragonfly Optimization**

The Adaptive Dragonfly Optimization (ADO) algorithm enhances data migration efficiency by organizing container-to-VM and VM-to-PM assignments, inspired by dragonfly swarm behavior. Dragonflies use two key strategies: hunting (local search) and migration (global search). The ADO algorithm applies these strategies to improve search accuracy and resource distribution during migration.

1. Initial Setup

The algorithm starts by creating a population of potential solutions, where each solution is represented as a matrix showing container-to-VM assignments. These solutions define how containers are distributed among VMs and PMs, and are generated randomly to cover a wide search space.

1. Performance Evaluation

Each solution’s effectiveness is measured based on energy usage, migration time, and transmission cost. The goal is to minimize both operational cost and energy consumption while improving overall system performance.

1. Behavioral Adjustment

* The algorithm updates the search process using five key behaviors:
* Spacing – Prevents solutions from clustering too closely.
* Synchronization – Aligns solutions to move in the same direction.
* Grouping – Guides solutions toward the average position of the population.
* Attraction – Directs solutions toward the most promising outcomes.
* Repulsion – Pushes solutions away from poor-performing options.

1. Levy Flight Mechanism

To prevent stagnation, the algorithm introduces Levy Flight, allowing solutions to take large, random steps across the search space. This helps avoid getting stuck in suboptimal solutions and encourages broader exploration.

1. Completion

The algorithm stops when it reaches a preset iteration limit or finds a solution that meets the optimization targets. The final result provides an efficient container-to-VM and VM-to-PM mapping, reducing costs and energy while improving performance.

|  |
| --- |
| **Adaptive Dragonfly Optimization Algorithm** |
| Input: Number of VM, Count of PM, Count of containers and Count of population size  Output: Best solutions |
| Initialize:  Generate initial population of dragonflies  Initialize velocity vector  Set learning factors: w, s, a, c, f, e  Define maximum iterations  Train Actor-Critic Neural Network:  Define and train Actor model for load prediction  Define and train Critic model for value estimation  Outer Dragonfly Algorithm (Physical Machine Level):  While termination criteria not met:  For each dragonfly:  Calculate fitness using load, energy, migration, and variance  Update food source (best solution) and enemy (worst solution)  Update velocity and position based on separation, alignment, cohesion, attraction, distraction, and levy flight  If new fitness is better → Update best solution  If no improvement → Stop iterations  Select best physical machine  Middle Dragonfly Algorithm (VM Level):  Initialize population for VM search within best machine  While termination criteria not met:  For each dragonfly:  Calculate fitness using load, energy, and migration  Update velocity and position based on search factors  If new fitness is better → Update best solution  If no improvement → Stop iterations  Select best VM  Inner Dragonfly Algorithm (Container Level):  Initialize population for container search within best VM  While termination criteria not met:  For each dragonfly:  Calculate fitness using load, energy, and migration  Update velocity and position based on search factors  If new fitness is better → Update best solution  If no improvement → Stop iterations  Select best container allocation  Dynamic Load Adjustment:  Introduce load fluctuation based on real-world simulation  Adjust fitness based on updated load  Output Results:  Return best physical machine, VM, container allocation, and performance metrics |

1. **Findings and Analysis**

The Dragonfly Algorithm successfully optimized resource allocation across virtual machines, as evidenced by the decreasing fitness score over iterations. Predicted loads were well-balanced, with one VM identified as having the least loaded container. Energy consumption varied across VMs, with some experiencing reduced usage due to efficient resource placement. Transmission costs and migration times followed a similar trend, decreasing where resources were allocated optimally. The algorithm also demonstrated effective energy and cost conservation, leading to improved overall efficiency. Visual analysis through plots confirmed these findings, highlighting variations in energy, cost, and load across different VMs, reinforcing the impact of the optimization process.

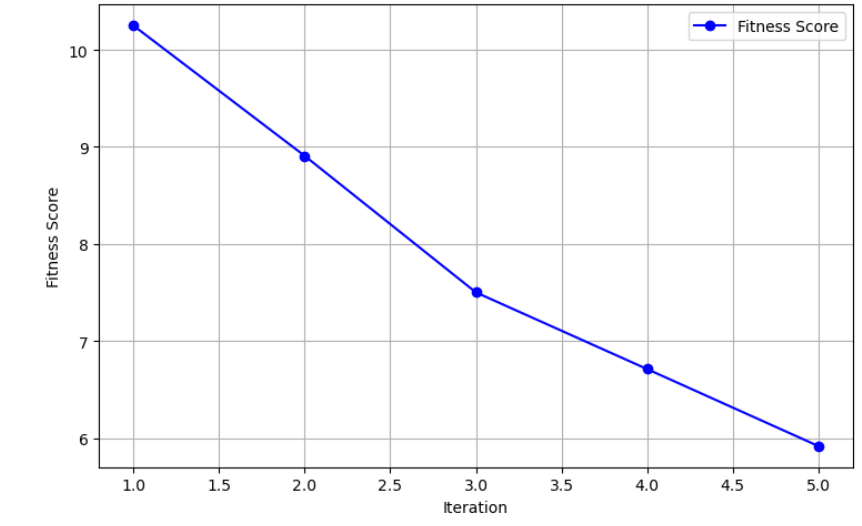
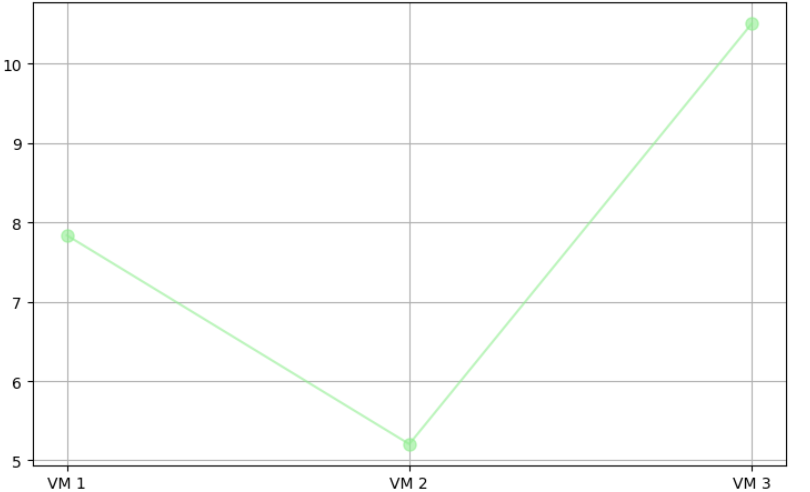
Figure 3: Fitness Score over Iteration

Figure 4: Energy Consumption vs VMs

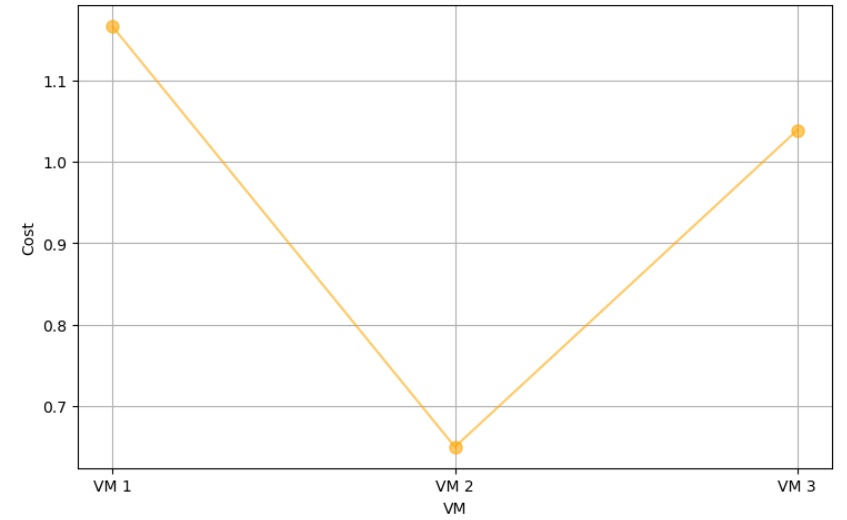
****

Figure 5: Cost vs VMs

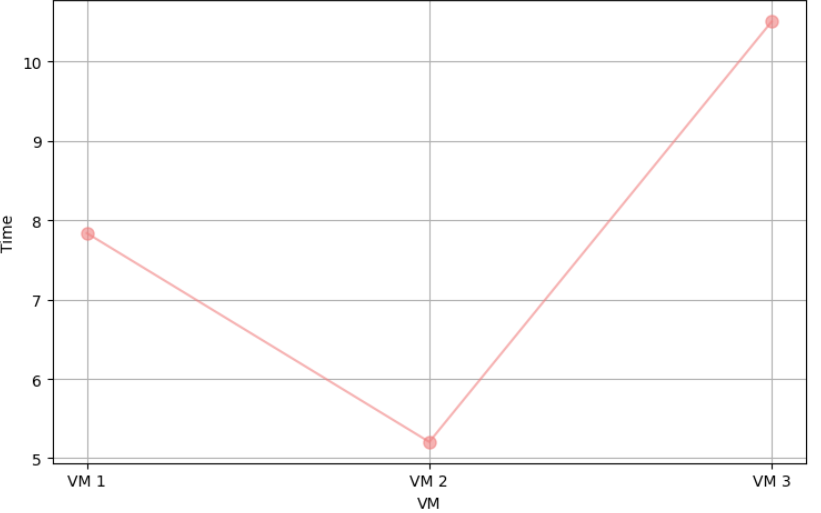
****

Figure 6: Time vs VMs

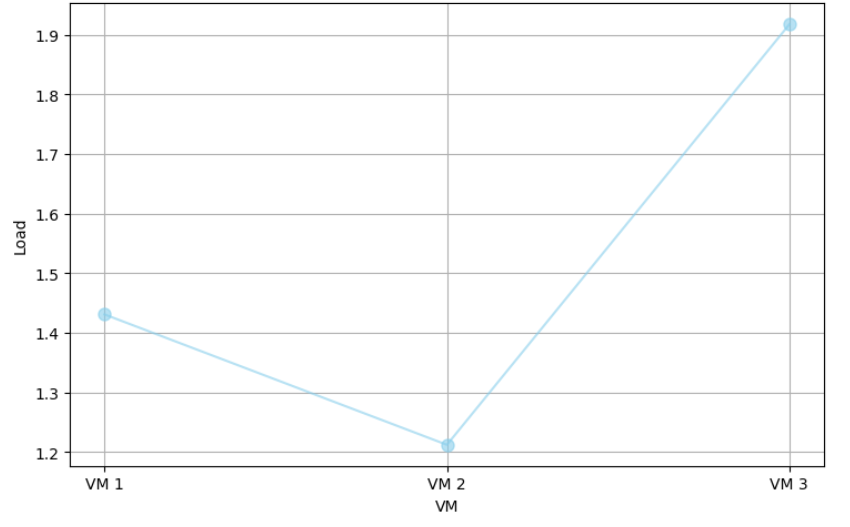
****

Figure 7: Load vs VMs

1. **Conclusions and Future Plan**

This research introduces an Adaptive Dragonfly Optimization (ADrO) framework to enhance data migration efficiency in container-based heterogeneous cloud environments. The inclusion of a Levy flight strategy helps avoid getting stuck in local optima, leading to more effective migration paths. An Actor-Critic Neural Network (ACNN) improves load forecasting, enabling smarter migration decisions. The framework’s multi-objective function focuses on balancing energy consumption, migration speed, operational cost, and resource capacity. Performance tests using Python demonstrated that ADrO outperforms Genetic Algorithm (GA) and Particle Swarm Optimization (PSO). The results showed significant improvements in energy efficiency, quicker migration, cost reduction, and better resource utilization. This approach provides a scalable and adaptive solution for cloud-based data management.

**References**

[1] Tej.C and K.S. Rekha, 2023. Energy Efficient Data Migration Concerning Interoperability Using Optimized Deep Learning in Container-Based Heterogenous Cloud Computing.

[2] Manisha Malhotra and Krishnan Tuli, 2023. Optimal Meta Heuristic Elastic Scheduling (OMES) for VM selection and migration in cloud computing.

[3] Mohammed Shehab, Mohammed Al Shinwan and Lalith Abualigah, 2021. Dragonfly algorithm: a comprehensive survey of its results, variants and applications.

[4]Mohammed Zakarya and Ayaz Ali Khan, 2020. HeporCloud: An energy and performance efficient resource orchestrator for hybrid heterogenous cloud computing environments.

[5] Adhikari, M. and Srirama, S.N., 2019. Multi-objective accelerated particle swarm optimization with a container-based scheduling for Internet-of-Things in cloud environment. Journal of Network and Computer Applications, 137, pp.35-61.

[6] Tarik A Rashid, Chnoor M Rahman, 2019. Dragonfly Algorithm and its Applications in Applied Science--Survey.

[7] Cigdem Aci, Hakan Gulcan, 2019. A Modified Dragonfly Optimization Algorithm for Single- and Multiobjective Problems Using Brownian Motion.