DOI: 10.15849/IJASCA.240730.01

Int. J. Advance Soft Compu. Appl, Vol. 16, No. 2, July 2024
Print ISSN: 2710-1274, Online ISSN: 2074-8523

Independent Task Scheduling in Cloud Computing Environment using Modified Orca Optimizer

Raja Masadeh, Ahmad sharieh, Mai Abu-Jazoh, Kholoud Alshqurat, Esra Masadeh and Nesreen Alsharman

Computer Science Department, The World Islamic Sciences and Education University
e-mail: raja.masadeh@wise.edu.jo
Computer Science Department, The University of Jordan
e-mail: sharieh@ju.edu.jo
Computer Science Department, The University of Jordan
e-mail: mai.abujazoh@gmail.com
Academic services Department, The World Islamic Sciences and Education University
e-mail: kholoud.shqurat@wise.edu.jo
Independent researcher
e-mail: esramasadeh@yahoo.com
Computer Science Department, German Jordanian University
e-mail: nesreen.alsharman@gju.edu.jo

Abstract

Cloud computing (CC) has become a hot study area as a result of the rapid development and movement of many services to the cloud environment. In order to minimize makespan, the task scheduling mechanism must allocate tasks to appropriate and obtainable virtual machines (VMs). In this research, a modified Orca optimization algorithm (MOOA) and a task scheduling method for CC based on MOOA are presented. The simulation results showed that MOOA scheduler outperformed other state-of-the-art schedulers in terms of makespan.

Keywords: *Cloud Computing, Makespan, Metaheuristic Algorithm, Modified Orca Optimization Algorithm, ORCA Optimization Algorithm, Task scheduling.*

1 Introduction

Task Scheduling in cloud computing (CC) is defined as assigning tasks to the appropriate and available resources (software applications, data, CPU, storage, memory, and bandwidth) that gains the highest possible level of performance and resources usage. It is one of the broadly researched issues in cloud computing where many cloud computing researchers have focused on the tasks scheduling methods in the efficient utilization of resources that reduces the response time and thus the execution of clients' tasks in the least possible time [1,2, 3, 30, 31]. The CC is one of the most prominent technologies in the Information Technology (IT) sector. It combines concepts of distributed and parallel computing to provide services to the internet users via service providers such as Google, Amazon, Microsoft, and Apple [4, 5, 6]. In CC, there are three essential ways for providing

Received 12 April 2024; Accepted 14 May 2024

the services to the internet users which are: Platform as a Service (PaaS), Software as a Service (SaaS), and Infrastructure as a Service (IaaS) [7, 8].

The rapid universal expands of Coronavirus 2019 (COVID-19) has grown up the vital of CC than ever. Throughout COVID-19 crisis, the CC provides platform for the companies and educational institutions to continue operating seamlessly [9, 10, 11]. The client can request the services he needs and pay for using them. Pay-as-you-use is the most recent CC payment model where the charges can be made based on the actual consumption of available resources during specific time [12, 13]. Therefore, it is important that clients are provided with an efficient task scheduling technique as clients need to finish their tasks as fast as possible in CC environments [12, 14].

Virtualization technique is mainly used to reduce using hardware, cost less, and energy saving techniques that is used by Cloud Service Provider (CSP). It enables several customers and organizations to share a single physical instance of an application or resource simultaneously. Virtualization term is often referred to hardware virtualization, which plays an essential role in delivering IaaS solutions for CC. In addition, virtualization technologies create a virtual environment for memory, networking, and storage, in addition to application execution. [15]. In CC, When the number of client requests rises at a given moment, it becomes a demanding job to handle the whole request in the most constrained response time and pleased nature of administration. CSP attempts to assign these incoming requests to an appropriate virtual machine (VM) in a way that avoids overloading the machine and maintains a load balance between these machines or resources. Therefore, good scheduling mechanism is required to handle the clients' requests by assigning appropriate resources and balancing the load among VMs. Moreover, the scheduling strategy is employed to improve the nature of administration attributes such as resource consumption, task dismissal percentage, vitality utilization, dependability, running cost, minimal running duration with limit makespan, and enhance throughput. [16].

Having an optimal scheduling solution in CC is regarded as an NP-complete problem because of the huge solution domain and it needs an extended amount of time to get an optimal solution within minimum response time. Many heuristic algorithms used to generate optimal or near to optimal solutions for a special complex optimization problem such as Optimal Workflow Scheduling (OWS), A compromised-Time-Cost (CTC), Heterogeneous Earliest Finish Time (HEFT) and Resource-Aware-Scheduling algorithm (RASA) [17]. In addition to that, different metaheuristic algorithms used to generate optimal or near to optimal schedules such as the scheduling technique based on Genetic Algorithm (GA), Ant Colony Optimization (ACO), Symbiotic Organisms Search (SOS), Grey Wolf Optimization (GWO), Whale Optimization Algorithm (WOA), Vocalization of Whale Optimization Algorithm (VWOA), and Sea Lion Optimization algorithm (SLnO) [1, 18].

In this study, a novel cloud computing task scheduling technique depending on Modified ORCA Optimization Algorithm (MOOA) is presented. Moreover, a new cloud scheduler environment depends on Orca Optimization Algorithm (OOA) is introduced. OOA is a metaheuristic algorithm inspired by the exceptional Orcas hunting strategy. It is suggested for complicated and nonlinear optimization problems, mostly for practical problems facing engineering. According to Authors in [19], the accuracy of OOA is high and convergence speed is very good and the performance of OOA is significantly superior than other comparable algorithms such as Imperialist Competitive Algorithm (ICA) and particle swarm algorithm (PSO). Moreover, the OOA has strong performance in multiple runs, which is a significant and necessary attribute for an optimization method. The proposed performance of MOOA is compared with original OOA. The results of simulation provided good outcomes in terms of reducing makespan.

The paper is organized as follows: Section two presents related work. Section three outlines the ORCA Optimization Algorithm (OOA). Section four describes in detail the proposed Modified ORCA Optimization Algorithm (MOOA). MOOA scheduler is described in section five. Section six presents complexity analysis of MOOA scheduler. The experiment findings and discussion are reported in section seven. Finally, Section eight concludes the paper.

2 Related Work

Scheduling of independent tasks is one of the primary issues that leads to CC system efficiency reduction. Improving the performance of cloud systems requires an effective task scheduling mechanism. Different techniques were used by different researchers to resolve task scheduling problems. Most of researchers have used metaheuristic algorithms to optimize task scheduling in CC and find an optimal or near to optimal distribution of available resources in CC, as Ant Colony Optimization (ACO)Algorithm, Genetic Algorithm (GA), firefly algorithm (FA), Particle Swarm Optimization (PSO), Grey Wolf Optimization (GWO) Algorithm and Whale Optimization Algorithm (WOA) Algorithm [1, 20, 21]. Where metaheuristic algorithms lead to an optimal or near to optimal solutions to a given issue at low computational cost when compared using the exact method of solving the same problem. This section includes some state-of-the-art metaheuristics algorithms were used to propose solutions of task scheduling problem in CC are demonstrated.

In [22], the researchers used WOA to develop technique that leads to less energy consumption and minimize the makespan for independent task scheduling in CC. The results of experiments showed better performance of energy consumption and makespan using the suggested approach is compared to the Min-Min algorithm. A novel task scheduling algorithm was introduced by the researchers in [23] which is called hybrid Firefly - Simulated Annealing Algorithm FA-SA algorithm. The FA-SA algorithm was produced by combining two optimization algorithms, FA, and the simulated annealing (SA) optimization algorithms in order to take advantage of the merits of both algorithms. Comparing the proposed algorithm with FA, simulated annealing, max-min, and min-min algorithms. The results showed that the performance of FA-SA algorithm in term of makespan is reduced applied on different scenarios using various numbers of VMs and tasks.

The researchers in [20] suggested a new technique for task scheduling in CC depending on a modified grey wolf optimization (MGWO) algorithm. The researchers obtained MGWO algorithm by modifying the fitness function of GWO algorithm and made this new algorithm based on multi objective fitness function rather than using the single makespan objective. Researchers simulated the proposed algorithm using Cloud-Sim tool and compared the proposed algorithm performance with the standard WOA and GWO algorithms. Results show better outcomes in minimization cost, degree of imbalance and makespan.

Jochsam et al. in [21] proposed a new solution for task scheduling issue in CC aimed to minimize the processing time of task scheduling and minimize the number of VMs used. Researchers used to achieve their goals combination of metaheuristic GA and static algorithm. In the first phase of the technique the GA algorithm was used to reduce tasks processing time then the static algorithm was applied in order to tackle the set partitioning problem. The outcomes proved that the enhanced GA led to a perfect performance in case of heavy loads in terms of reduced processing time in task scheduling.

Kumar et al. [24] suggested a novel hybrid task scheduling algorithm called GA-PSO algorithm. This algorithm was produced by combining two optimization algorithms GA and PSO algorithms. Compared to standard the GA, Max-Min and Min-Min algorithms, the experimental findings showed that the GA-PSO algorithm obtained more desirable results in terms of reducing the execution time.

The WOA is one of the metaheuristic algorithms that is used to handle task scheduling problem in CC. Raja et al. [25] proposed vocalization for humpback whale optimization (VWOA) algorithm. This new metaheuristic algorithm mimics humpback whales' vocalization behavior. The VWOA was used to improve task scheduling in CC using multi-objective mathematical model to enhance the cloud environment scheduling of independent tasks. Compared with the standard Round Robin (RR) and the WOA algorithms, the results reveal that the VWOA scheduler has preferable performance in terms of makespan, degree of imbalance cost, energy consumption and resource utilization.

In [26], relied on the WOA algorithm the researchers suggested a new task scheduling approach for CC in order to develop a system performance with certain resources computations. Furthermore, to make more improvements on the search efficiency of scheduling depends on the WOA algorithm, the researchers improved the WOA algorithm and proposed a new algorithm called Improved WOA for CC task scheduling (IWC). Compared with some commonly used metaheuristic algorithms such as PSO and ACO algorithms, the experimental results extracted that IWC algorithm is effective in terms of searching optimal scheduling plans. Therefore, using IWC lead to improve CC efficiency concerning the utilization cost of system resource and system load.

In [27], researcher suggested a novel technique for task scheduling in CC depending on the metaheuristic Sea Lion Optimization (SLnO) algorithm. The researchers aimed to minimize the overall completion time, power consumption and cost; and increase the resources utilization. Compared to other techniques such as WOA, VWOA, GWO and RR algorithms the results showed that the introduced technique performs better in terms of consumed energy, makespan, cost, degree of imbalance, and resources utilization. Table 1 summarizes the previous studies reviewed in this paper.

As mentioned earlier, many studies employed various metaheuristic optimization algorithms to deal with cloud task scheduling problems, however no study employed ORCA optimizer to solve the problem. Thus, this study presented a novel solution using ORCA technique to solve task scheduling in CC and single objective mathematical model considering makespan. Moreover, a Modified Orca Optimization Algorithm (MOOA) and a scheduling of independent task technique for CC based on MOOA are suggested. The simulation results show that MOOA scheduler had performed better than other state-of-the-art schedulers in term of makespan.

Table 1: Summary of evaluated articles relating to CC independent task scheduling.

Ref	Year	Applied Algorithm	Performance Criteria	Task Nature	Used Tools	Compared to
[20]	2019	Modified Grey Wolf Optimizer (MGWO)	• Makespan • Cost	Independent	CloudSim	Traditional Grey Wolf Optimizer

			• Degree of imbalance			(GWO) and Whale Optimization Algorithm (WOA),
[21]	2019	Hybrid Genetic Algorithm-Static (GSASTCC) Algorithm	• Makespan • Resources utilization •	Dependent	CloudSim	Particle Swarm Optimization PSO and First in First Out (FIFO) algorithms
[22]	2017	Whale Optimization Algorithm (WOA)	• Makespan • Energy consumption	Independent	MATLAB	min-min algorithm
[23]	2018	FA-SA algorithm, produced by combining two optimization algorithms, Firefly and Simulated Annealing Algorithms	• Makespan	Independent	MATLAB	Simulated Annealing (SA), firefly(FA), max-min , and min-min, algorithms.
[24]	2019	Hybrid Genetic Algorithm-Particle Swarm Optimization (GA-PSO) algorithm	• Execution time	Dependent	CloudSim	Standard Genetic Algorithm (GA), Min-Min and Max-Min algorithms
[25]	2019	Vocalization for humpback whale optimization Algorithm (VWOA)	• Makespan • Cost • Consumption of Energy • Utilization of Resources • Imbalance degree	Independent	CloudSim	Round Robin (RR) and Whale Optimization Algorithm (WOA)
[26]	2020	Whale Optimization Algorithm (WOA)	• Makespan • Resources utilization • Load balancing. • Cost	Independent	MATLAB	Particle Swarm Optimization (PSO) and Ant Colony Optimization (ACO) Algorithms
[27]	2021	Sea Lion Optimization (SLnO) Algorithm	• Makespan • Resources utilization • Load balancing.	Independent	CloudSim	Grey Wolf Optimization (GWO), Vocalization for humpback whale optimization Algorithm (VWOA),

			• Energy consumption • Degree of imbalance.			(WOA), and Round Robin (RR) Algorithms, Whale Optimization Algorithm

3 Orca Optimization Algorithm (OOA)

The researchers of [19] proposed one of the most advanced nature-inspired metaheuristic optimization algorithms: the ORCA Optimization Algorithm. It mimics the hunting habits of orcas, or killer whales. The following concepts serve as the foundation for this algorithm.

A. Orcas hunt using a unique wave washing strategy in which they wash the seal's ice floe with different energies depending on the distance between them and the ice floe. Each Orca's energy determines their ability to wash the ice floe and approach the seal. Therefore, the farther Orca has more energy than closest one, so farther Orca can impact on the ice floe more. This behavior is modeled in OOA algorithm using Equations (1) and (2)

$$e_i = F_i - F_s \qquad (1)$$

In Eq. (1) the energy of each Orca ($\boldsymbol{e_i}$) is calculated by subtraction the fitness of seal ($\boldsymbol{F_s}$) from the fitness of the Orca ($\boldsymbol{F_i}$). Then, the energy vector will be sorted to determine which Orca has the highest energy and which one has the lowest energy. Depending on the sorted energy vector to sorted energy vector, each Orca's energy is normalized using Equation (2).

$$\mathrm{E}_i = \frac{e_i - e_{min}}{e_{max} - e_{min}} \qquad (2)$$

B. According to OOA, the Orcas move toward the ice floe in order to hunt the seal, and they will be located in the highlighted area between the two circles as shown in Fig.1, where R indicates the radius of the first circle and R-d represents the radius of the second circle and d_i is calculated by Equation (3).

$$d_i = E_i * \mathrm{R} \qquad (3)$$

C. The researchers of OOA tried to make the algorithm escape from local minima by eliminating a percentage of Orca population (the worst Orcas) after each iteration. Algorithm 1 shows the Pseudo code of OOA.

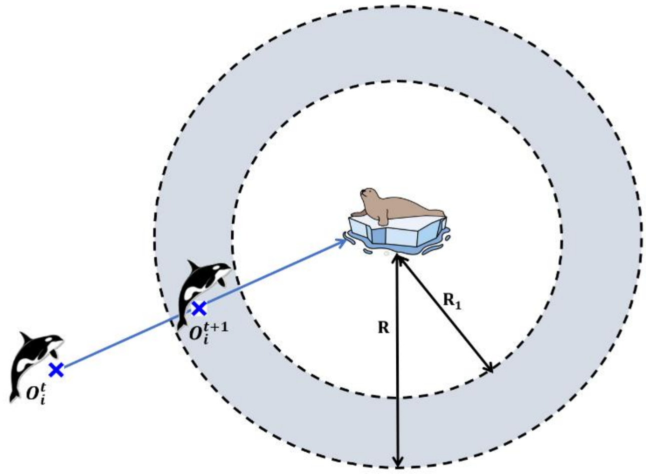

Fig. 1: Updating the position of search agent.

Algorithm.1: Pseudo code of OOA

Randomly initialize the Orcas population (N+1) in the search space.
Define the initial radius for an ice floe (R (ice floe))
Define the initial radius (Ri) for each Orca.
While (the halting condition is not satisfied.) do

- Allocate the best result as the Seal (Xs).
- Use Eqs. (1) and (2) to calculate the normalized energy (Ei) of each orca.
- Using the energy provided by Equation (3), move the orcas toward the direction of the seal and ice floe.
- The worst Orcas (P%) are eliminated, and new ones are chosen at random for the following cycle.

End while

4 Orca Optimization Algorithm (OOA)

This section begins by discussing the inspiration for the suggested algorithm. Then, the mathematical model is introduced.

4.1 Inspiration

Orca, often known as killer whale, is the ocean’s dominant predator. Orcas are highly social, and most live in pods, or social groups. This type of predator has a cunning strategy of hunting in a pod and creating tremendous waves to wash prey such as seals from floating pieces of ice. More precisely, the furthest Orca, with more energy, can melt a patch of ice that the prey is lying on, and get closer to it. The nearest Orca with lower energy will have little effect on the melting of an ice floe. At least two orcas remained near the prey against the ice floe, while four orcas moved away from the ice floe carrying the prey (seal). These four orcas return at the same time, swimming together with their left sides pointing to the ocean's surface. This produces a massive wave that flips over the ice floe towards the wave, causing the ice floe to twist and tilt in the other direction where the pursuing Orcas are waiting as illustrated in Fig.2. [19. 28].

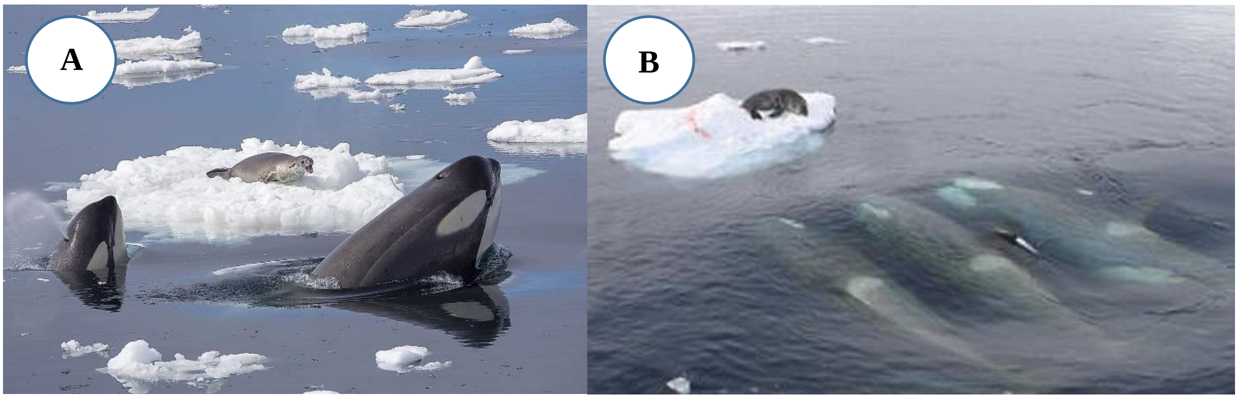

Fig.2: Hunting behavior of Orca whales: (A) Orca whales melt the ice floe that the prey lies on. (B) Four Orcas generate a huge wave that washes the prey off the ice floe

4.2 Mathematical Model and Optimization Algorithm

Orcas can detect the position of prey, especially seal that lies on ice floes and encircle it. The suggested algorithm (MOOA) considers the fittest solution as alpha (α) which has the highest energy. Accordingly, the second, third and fourth best solutions are assumed to be beta (β), gama (γ) and delta (δ), respectively. However, the rest of the pod (candidate solutions) are named omega (ω).

4.2.1 Encircling prey (Seal)

As mentioned above, Orcas encircle the seal during the hunting technique. As shown in Fig.3, Orcas that stay far away from the chunk of ice that the seal lies on, have higher energy and velocity when arrive the ice floe and wash it, causing it to melt and shrink in size. However, Orcas that are near to the chunk of ice have reduced energy and velocity, thus they have a smaller influence on melting the ice floe. This section contains the suggested formulas in this regard.

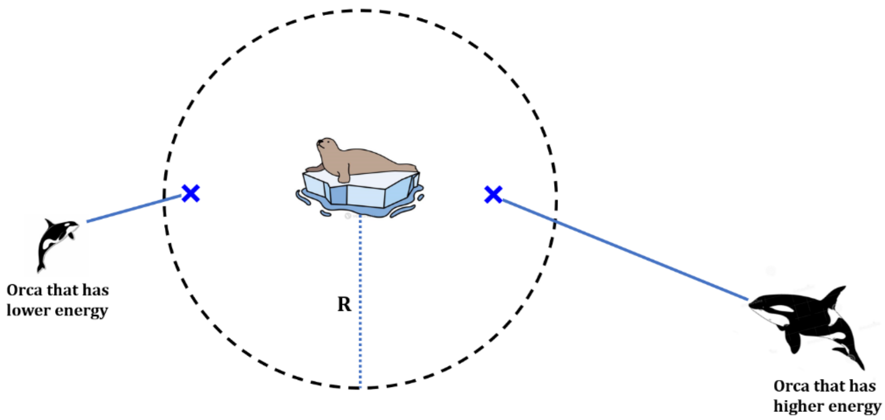

Fig. 3: Orcas wash the ice floe to minimize its size.

The energy of each search Orca (ΔP_i) can be computed using Equation (4).

$$\Delta P_i = L_i - L_s \qquad (4)$$

Where the energy(ΔP_i) of a specific Orca calculated by subtraction the current location (fitness) of seal ($\boldsymbol{L_s}$) from the current location (fitness) of this Orca ($\boldsymbol{L_i}$) Where, as mentioned earlier, Orcas that are stayed far away from the seal have more energy than those who stay close to the seal.

Then sort the calculated energy of each Orca in descending order and define the Orca with highest or lowest energy. Then, calculate the normalized values of search agents' energies using Equation (5).

$$P_i = (P_i - P_{min})/(P_{max} - P_{min}) \qquad (5)$$

After the best search agents have been determined, the other search agents will adjust their positions toward the prey. shown in Fig.1, the Orca moves approaches the seal's ice floe and updates its location based on the area of melted ice that the Orcas removed and reduced. More precisely, Orca stays on the highlighted area between co-circles of radius $\boldsymbol{R}$ and radius $\boldsymbol{R1}$, where $\boldsymbol{R}$ indicates the radius of ice floe before melting and $\boldsymbol{R1}$ represents the radius of ice floe after melting. Moreover, each Orca has its own $\boldsymbol{R}$ which displays the distance between this Orca and the ice floe, which means it specifies the current location of this Orca. Update the Orcas' positions is calculated using Equations (6) and (7).

$$d_i(t) = P_i * R_i(\mathrm{t}) \qquad (6)$$

Where $\boldsymbol{d_i}$ denotes the amount of change in distance of the search agent (Orca(i)), $\boldsymbol{t}$ represents the current iteration, $\boldsymbol{P_i}$ normalized energy value of the search agent Orca(i), and $\boldsymbol{R_i}(\mathrm{t})$ indicates the current radius of the search agent Orca(i). Equation (7) calculates the new radius of the search agent after moving near the seal $\boldsymbol{R_i(t+1)}$.

$$R_i(t+1) = R_i(\mathrm{t}) - d_i(t) \qquad (7)$$

Where $(\boldsymbol{t+1})$ indicates the next iteration, $\boldsymbol{R_i}(\mathrm{t})$ represents the radius of the search agent $(\boldsymbol{i})$ before movement and $\boldsymbol{R_i(t+1)}$ is the radius of search agent $(\boldsymbol{i})$ after movement.

4.2.2 Hunting

As displayed in Fig.4, hunting mechanism is usually guided by the four best search Orcas that have the highest energies. When the floe reaches a tiny size and is directed into open water, four Orcas launch a coordinated attack, causing a breaking wave that tipped the ice and displaced the prey. According to the four killer whales' approach, when the ice floe the whales started to swim on their sides; mostly to keep their dorsal fins protected and also to allow them to be very close from the bottom of the ice floe. Swimming on their sides enhances the plane of their bodies, deflecting the water and maybe increasing the magnitude of the wave they are making. The mathematical model is as follows in Equation (8).

$$Wave\ energy = \mathrm{Avg}(\textstyle\sum_{i \in S} P_i) \qquad (8)$$

Where $S\epsilon\{O_\alpha, O_\beta, O_\gamma, O_\delta\}$, $\boldsymbol{P_i}$ represents the normalized energy value of $O_\alpha, O_\beta, O_\gamma,$ and O_δ which is employed to generate the huge wave. The radius of ice floe $\boldsymbol{R_{ice\ floe}}$ will be affected by the wave energy then become melted and broken. To calculate the amount of decrease in radius ($\boldsymbol{R_{ice\ floe}}$), Equations (9) and (10) are employed in MOOA.

$$d_{ice\ floe}(t) = Wave\ energy * R_{ice\ floe}(\text{t}) \quad (9)$$

Where $\boldsymbol{d_{ice\ floe}(t)}$ denotes the amount of change in $\boldsymbol{R_{ice\ floe}}$, $\boldsymbol{(t)}$ represents the current iteration, and $\boldsymbol{R_{ice\ floe}}(\text{t})$ indicates the current radius of the ice floe. Therefore, Equation (10) calculates the new radius of the ice floe after breaking by the four most powerful Orcas $\boldsymbol{R_{ice\ floe}(t+1)}$.

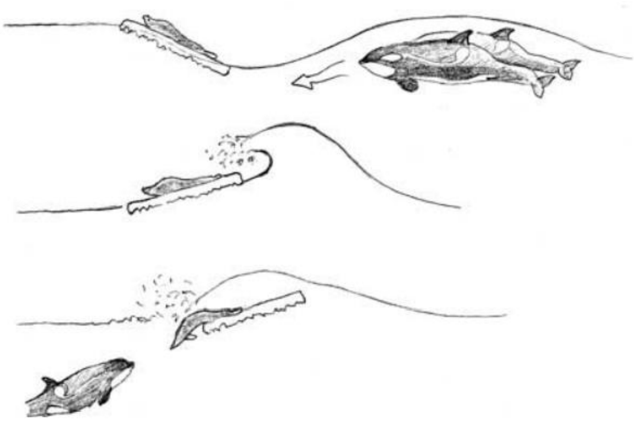

Fig.4: Four Orcas generated a huge wave that washed the seal and rotated the ice floe and titled it in the other direction.

Orcas end their hunting by attacking the prey (Seal) when the radius ($R_{ice\ floe}$) is less than the threshold $\boldsymbol{C}$ which is equal to one meter. More precisely, when $\boldsymbol{R_{ice\ floe}}$ is greater than one meter, it indicates that the prey is still far away from Orcas. Whereas, when $\boldsymbol{R_{ice\ floe}}$ is less than one meter, it means that the prey is close to Orcas and the Orcas can easily attack this prey. The value of $\boldsymbol{C}$ is assumed based on the fact that in [28] where the mature seals reach a length of up to 2.64 and 2.77 meters, thus when the radius of ice floe is less than one, the seal cannot stay on the ice floe and falls into the water. Algorithm 2 presents the pseudocode of proposed MOOA.

Algorithm 2: Pseudo code of MOOA

1. Randomly initialize the Orcas population (N+1) in the search space.
2. Define the initial radius for an ice floe ($R_{ice\ floe}$)
3. Define the initial radius (Ri) for each Orca.
4. While (the halting condition is not satisfied.)
   - If $R_{ice\ floe} > \mathrm{C}$
     {
   - Compute the fitness of each search agent (Orca) and sort them according to their fitness level.
   - Allocate the best result as the Seal (Xs)
   - $O_\alpha = The\ best\ search\ agent$ (Farthest Orca from ice floe)
   - $O_\beta = The\ 2^{nd}\ best\ search\ agent$
   - $O_\gamma = The\ 3^{rd}\ best\ search\ agent$
   - $O_\delta = The\ 4^{th}\ best\ search\ agent$
   - For each search agent (Orca) Compute the normalized energy by Eq. (4) and Eq. (5)
   - Update the search agent's (Orca) location towards the prey (S) and ice floe according to their energy by Eqs. (6) and (7)
   - Calculate the power of the wave movement that is generated by the four best search agents who have the most energies by Eq. (8)
   - Compute $R_{ice\ floe}$ after melting by Eqs. (9) and (10)
   - Generate them randomly for the next iteration.
     }

. End While

5 MOOA Scheduler

MOOA algorithm begins with random population of solutions. Thus, the first four current solutions are considered as the first four best candidate solutions which are called alpha, beta, delta, and gamma, respectively. Thus, keeps the iteration process based on the first current solution (alpha). This process is continued until it finds the optimal solution. The main phases of MOOA are presented as follows:

- ***Initialization stage***: In this phase, search agents' population O_j $(j = 1, 2, 3 \ldots n)$ is generated randomly.
- ***Fitness calculating stage***: to calculate the fitness function, makespan is taken into consideration. The best search agent O* is selected based on evaluation.
- ***Encircling prey stage***: In this step, the location of prey (seal) is supposed to be stationary on the ice floe. Therefore, Orcas surrounded the prey (seal), believing that the present solution is the best location. Other search agents will adjust their placements based on the position of the present best agent.
- ***Exploitation stage***: In this stage, the search agents terminate the chase by assaulting the prey when the ice floe melted and catch the prey.
- ***Exploration Stage*****:** In this phase, search agents often search based on the locations of alpha, beta, delta, and gamma. They move away from each other to look for prey, then converge to pursue the prey. The behavior diverges, depending on the value of $\boldsymbol{R}$. In case $|\boldsymbol{R}| > \boldsymbol{C}$, the search agent gets away from the prey; While in case $|\boldsymbol{R}| < \boldsymbol{C}$, the search agent moves toward the prey.

In MOOA Scheduler, independent task scheduling problem in CC environment is tackled as an optimization problem. At the beginning, a random population of Orcas is generated.

Each Orca indicates to a cloud task which calculates its fitness value in the calculation fitness phase. Throughout the course of iterations, the first four best solutions (alpha, beta, delta, and gamma Orcas) have the first maximum fitness values. The MOOA scheduler stops when the fittest solutions are reached.

Fig.5 introduces the abstraction for the presented MOOA scheduler for CC task scheduling. The primary goal of the proposed scheduler is to optimally distribute tasks into available and appropriate VMs.

Input: Tasks ($T_1, T_2, T_3, \ldots, T_n$), Virtual machines ($V_1, V_2, \ldots, V_k$).
Output: Assign independent tasks to the VMs.
BEGIN
// Initialization phase

1. Population size
2. Initialize radius (R=R0) of ice floe.
3. **While** (the halting condition is not satisfied)
   If $R_{ice\ floe} > C$
   {

// Calculation fitness phase

- Compute the search agent's fitness (for each Orca)
- Assign the best result as the Seal (Xs)
- $O_\alpha = The\ best\ search\ agent$
- $O_\beta = The\ 2^{nd}\ best\ search\ agent$
- $O_\gamma = The\ 3^{rd}\ best\ search\ agent$
- $O_\delta = The\ 4^{th}\ best\ search\ agent$

// Optimization phase

- For each search agent (Orca) Compute the normalized energy by Eqs. (4) and (5)
- Update the search agent's (Orca) location towards the prey (S) and ice floe according to their energy by Eqs. (6) and (7)
- Compute the power of the wave movement that is generated by the four best search agents who have the most energies by Eq. (8)
- Compute $R_{ice\ floe}$ after melting by Eq. (9) and Eq. (10)
- Generate them randomly for the next iteration.
  }

End While

Fig.5: Abstraction of MOOA scheduler

6 Complexity Analysis of MOOA Scheduler

MOOA scheduler includes three main phases: initialization, calculation fitness and optimization phases. Throughout this section, ***n*** indicates the population size and ***t*** represents the maximum number of iterations. The total time complexity of MOOA scheduler is represented in the following Equation (11).

$$O\ (MOOA\ Scheduler) = O\ (Initialization\ function) + O\ (calculation\ fitness\ function)$$

$$+ O\ (Optimization\ function) \qquad (11)$$

The initialization stage focuses on creating a large number of individuals in order to create a population that includes full solutions. While the population size is based on both the number of CC tasks and the number of VMs. The run time complexity of initialization function is $O\ (n)$.

$$O\ (Initialization\ function)\ =\ O\ (n) \qquad (12)$$

In the initial calculation fitness stage, each cloud task computes its single objective function that depends on makespan. The orcas that have the first four maximum values are considered the best four solutions (alpha, beta, delta, and gamma). Consequently, this function assists assigning each task to an available and suitable VM, then remove these cloud tasks from the queue. Therefore, the time complexity of the initial calculation fitness function is equal to O (n).

In the next iteration, the fitness values of overall search agents are calculated except the fittest solutions in the first iteration. Thus, the run time complexity of the calculation fitness function is equal to $O\ (n)$, Besides the number of independent tasks and VMs, maximum number of iterations is considered. The overall complexity of calculation fitness function is shown in Equation (13).

$$O\ (Calculation\ fitness\ function)\ =\ O\ (t\ *\ n) \qquad (13)$$

The time complexity of optimization function is determined by the number of orcas (population size) and the maximum number of iterations. Consequently, the time complexity of optimization function is represented in Equation (14).

$$O\ (Optimization\ function)\ =\ O\ (t\ *\ n) \qquad (14)$$

The total run time complexity of MOOA scheduler is expressed in the following Equation:

$$O\ (MOOA\ Scheduler)\ =\ O\ (n\ +\ t\ *\ n\ +\ t\ *\ n)\ =\ O\ (\ t\ *\ n)$$

7 Experiments Results and Discussion

The performance of MOOA algorithm, OOA algorithm, Sea Lion Optimization (SLnO) algorithm, and Round Robin (RR) algorithms for independent CC task scheduling is experimentally evaluated. SLnO is one of metaheuristic algorithms that simulated hunting manners of the sea lions in nature. Whereas RR is a traditional algorithm for scheduling various tasks. Most researchers compare their suggested scheduling strategies with the RR algorithm since it is based on starvation-free, where all jobs get a fair number of VMs. These algorithms were simulated with Eclipse IDE 4.4 and CloudSim Toolkit 3.0.3. In terms of makespan, the MOOA and OOA algorithms' results are compared to the findings of the existing SLnO and RR approaches. Makespan is the overall time required by the resources to finish performing all independent tasks in the CC environment [1]. Minimum makespan means that tasks in CC were efficiently scheduled which led to better performance of CC therefore the clients of CC can finish their tasks quickly in CC environments. Assume that $\{T1, T2, T3, T4, \ldots..Tk\}$ is the execution time of $\{t1, t2, t3, \ldots.ti\}$ tasks on $\{VM1, VM2, VM3, \ldots\ldots..VMj)$ VM. Thus, the makespan can be calculated using Equation (15) [20].

$$Makespan^{min} = Max\{T1 \dots .TK\} \quad (15)$$

This study compared the performance of the MOOA and OOA algorithms with the performance of SLnO and RR scheduling algorithms over different number of independent tasks and VMs. The setup includes 200,300,400, 500 Independent tasks with 4, 8, and 16 VMs. All experiments are validated on a computer with Intel Core i-7 10th generation processor, 16 GB RAM, and Windows 10 operating system. Each scenario was examined and implemented ten times, and the average of the scenarios was calculated. The findings are displayed in Fig. 6–8. Where these results showed that the MOOA outperforms SLnO, OOA and RR in terms of makespan.

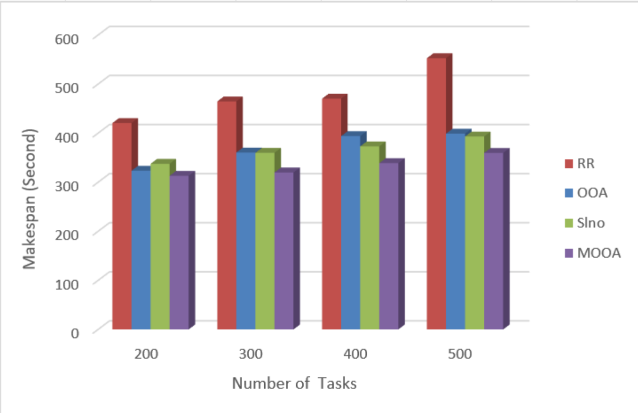

Fig.6: Makespan of various numbers of independent tasks when VMs equal to 4

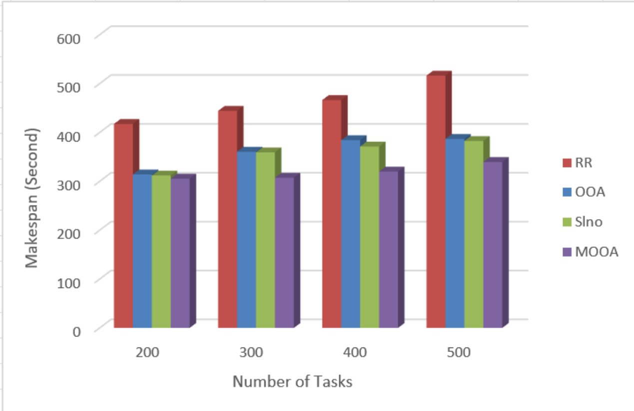

Fig. 7: Makespan of various numbers of independent tasks when VMs equal to 8

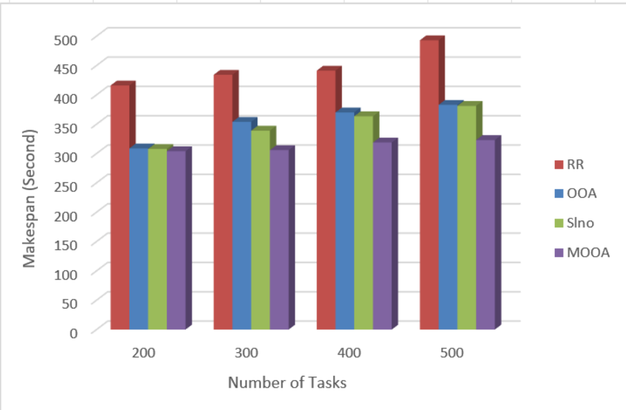

Fig. 8: Makespan of various numbers of independent VMs tasks when VMs equal to 16.

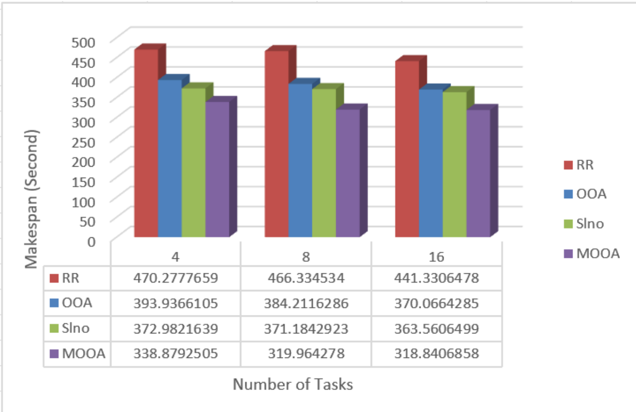

Fig. 9: Makespan of various number of number of independent tasks equal to 300.

Fig.6- Fig.9 shown that MOOA has better performance than the original OOA, SLnO, and RR algorithms in term of makespan. The makespan is employed in the fitness function of the MOOA algorithm in order to evaluate and choose the solutions which will lead to improved independent task scheduling and directly affect the overall makespan. Moreover, MOOA depends on selecting the best solution on the first four minimum makespan values which are considered form the Orcas (alpha, beta, delta, and gamma). The worst performance is for RR algorithm. This is due to the quantum length which is used in RR algorithm to switch between performing the different tasks. Where, if there are numerous long jobs to implement, the extended waiting time caused by the repeated switching between these tasks will consume a significant amount of time.

As shown in Fig. 6, 7, and 8, when the number of the VMs remains constant, while the number of tasks increases, the makespan increases. That is due increasing the amount of tasks to be distributed over a certain number of VMs complicates the task scheduling process. As a result, achieving global optimum scheduling will be more complicated and time-consuming.

On the contrary, it is possible to easily obtain an ideal global scheduling when the number of accessible VMs increases but the number of tasks remains constant, as seen in Fig. 8. Therefore, when the number of VMs increases, Makespan decreases while the number of tasks remains constant.

However, as shown in Fig. 6, when number of tasks is 300, the total makespan of scheduling, the tasks of MOOA is reduced by 9.14%, 13.97% and 27.94% compared with SLnO, OOA and RR schedulers, respectively. Table 2 illustrates the Improvement in percentage of MOOA over RR, OOA, and SLnO improvement percentage in all other scenarios.

Table 2: Improvement in percentage of MOOA over RR, OOA, and SLnO

		MOOA is reduced makespan compared		
Number of VMs	**Number of Tasks**	**with RR (%)**	**with OOA (%)**	**with SLnO (%)**
4	200	25.58005876	3.178000302	7.246398588
	300	31.1560579	11.25423604	11.134635
	400	27.94061829	13.97619784	9.14330945
	500	34.92149555	9.819533266	8.492036181
8	200	26.82254652	2.766521525	2.071760756
	300	30.81662012	14.76373705	14.47075985
	400	31.38739368	16.72186521	13.79907915
	500	34.25064911	12.21824562	11.2585887
16	200	26.93814635	1.543615848	1.26528603
	300	29.5654525	13.5980163	9.783820779
	400	27.75469201	13.84231014	12.30055127
	500	34.48189489	15.64356631	15.26073983

8 Conclusion

One of the major issues impacting the performance of cloud systems is task scheduling. Task scheduling means assigning tasks to available resources in an efficient way. Many studies attempted to overcome this problem in various ways. One of these ways is to employ a metaheuristic algorithm to create a suitable task scheduler. Therefore, in this study, the performance of OOA is improved by introducing a Modified Orca optimization algorithm (MOOA) algorithm. Moreover, to evaluate the performance of MOOA in solving the independent task scheduling problem, a new scheduler for CC based on MOOA algorithm is suggested.

The performance of the MOOA and OOA algorithms in handling CC task scheduling problem has been experimentally evaluated and compared to performance of SLnO and RR scheduling algorithms. The simulation results proved that the MOOA scheduler performed better than the OOA, SLnO, and RR schedulers in terms of makespan.

In this research, MOOA and OOA schedulers were evaluated using random number of populations. Therefore, in future work, the selected data set will be utilized. It is recommended to use the dataset presented in [29]. Moreover, in future, MOOA scheduler can be extended to other objectives in CC environment such as cost and power consumption.

References

[1] Masadeh, R., Mahafzah, B. A., & Sharieh, A. (2019). Sea lion optimization algorithm. *International Journal of Advanced Computer Science and Applications*, 10(5).

[2] Houssein, E. H., Gad, A. G., Wazery, Y. M., & Suganthan, P. N. (2021). Task scheduling in cloud computing based on meta-heuristics: review, taxonomy, open challenges, and future trends. " *Swarm and Evolutionary Computation* 62 (2021).

[3] Abdulraheem, W. K. (2022). Performance Comparison of Xen AND Hyper-V in Cloud Computing While Using Cryptosystems. *International Journal of Advances in Soft Computing & Its Applications*, 14(3).

[4] Leymann F, Fritsch D. Cloud computing: The next revolution in IT. *Proceedings of the 52th Photogrammetric Week*. 2009 Sep 7, pp.3-12.

[5] Almezeini, N., & Hafez, A. (2017). Task scheduling in cloud computing using lion optimization algorithm. *International Journal of Advanced Computer Science and Applications*, 8(11).

[6] Al-Dwairi, R., & Jditawi, W. (2022). The Role of Cloud Computing on the Governmental Units Performance and E-Participation (Empirical Study). *International Journal of Advances in Soft Computing & Its Applications*, 14(3).

[7] Ibrahim, I. M. (2021). Task scheduling algorithms in cloud computing: A review. *Turkish Journal of Computer and Mathematics Education (TURCOMAT),* 12(4), 1041-1053.

[8] Mumtaz, R., Samawi, V., Alhroob, A., Alzyadat, W., & Almukahel, I. (2022). PDIS: A Service Layer for Privacy and Detecting Intrusions in Cloud Computing. *International Journal of Advances in Soft Computing & Its Applications,* 14(2).

[9] Alashhab, Z. R., Anbar, M., Singh, M. M., Leau, Y. B., Al-Sai, Z. A., & Alhayja'a, S. A. (2021). Impact of coronavirus pandemic crisis on technologies and cloud computing applications. *Journal of Electronic Science and Technology,* 19(1), 100059.

[10] Agrawal, S. (2021, February). A survey on recent applications of Cloud computing in education: Covid-19 perspective. In *Journal of Physics: Conference Series* (Vol. 1828, No. 1, p. 012076). IOP Publishing.

[11] Masadeh, R., Almomani, O., Masadeh, E., & Masa'deh, R. E. (2023). Secure CoAP Application Layer Protocol for the Internet of Things Using Hermitian Curves. In The *Effect of Information Technology on Business and Marketing Intelligence Systems* (pp. 1869-1884). *Cham: Springer International Publishing*.

[12] Kaur C. The cloud computing and internet of things (IoT). *Int. J. Sci. Res. Sci. Eng. Technol*. 2020 Jan:19-22.

[13] Masadeh, R., AlSaaidah, B., Masadeh, E., Al-Hadidi, M. D. R., & Almomani, O. (2022). Elastic Hop Count Trickle Timer Algorithm in Internet of Things. *Sustainability*, 14(19), 12417.

[14] Almomani, O. (2020). A feature selection model for network intrusion detection system based on PSO, GWO, FFA and GA algorithms. *Symmetry*, 12(6), 1046.

[15] Alboaneen, D. A., Tianfield, H., & Zhang, Y. (2017, March). Glowworm swarm optimisation based task scheduling for cloud computing. *In Proceedings of the Second International Conference on Internet of things, Data and Cloud Computing* (pp. 1-7).

[16] Kumar, M., & Sharma, S. C. (2018). PSO-COGENT: Cost and energy efficient scheduling in cloud environment with deadline constraint. *Sustainable Computing: Informatics and Systems*, 19, 147-164.

[17] Almezeini, N., & Hafez, A. (2016, April). An Enhanced Workflow Scheduling Algorithm in Cloud Computing. *In CLOSER* (2) (pp. 67-73).

[18] Natesan, G., & Chokkalingam, A. (2020). An improved grey wolf optimization algorithm based task scheduling in cloud computing environment. *Int. Arab J. Inf. Technol*., 17(1), 73-81.

[19] Golilarz, N. A., Gao, H., Addeh, A., & Pirasteh, S. (2020, December). ORCA optimization algorithm: A new meta-heuristic tool for complex optimization problems. *In 2020 17th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)* (pp. 198-204). IEEE.

[20] Alzaqebah, A., Al-Sayyed, R., & Masadeh, R. (2019, October). Task scheduling based on modified grey wolf optimizer in cloud computing environment. *In 2019 2nd International Conference on new Trends in Computing Sciences (ICTCS)* (pp. 1-6). IEEE.

[21] Matos, J. G. D., Marques, C. K. D. M., & Liberalino, C. H. (2019). Genetic and static algorithm for task scheduling in cloud computing. *International Journal of Cloud Computing*, 8(1), 1-19.

[22] Sharma, M., & Garg, R. (2017, December). Energy-aware whale-optmized task scheduler in cloud computing. *In 2017 International Conference on Intelligent Sustainable Systems (ICISS)* (pp. 121-126). IEEE.

[23] Fanian, F., Bardsiri, V. K., & Shokouhifar, M. (2018). A new task scheduling algorithm using firefly and simulated annealing algorithms in cloud computing. *International Journal of Advanced Computer Science and Applications*, 9(2).

[24] Kumar, A. S., Parthiban, K., & Shankar, S. S. (2019, February). An efficient task scheduling in a cloud computing environment using hybrid Genetic Algorithm-Particle Swarm Optimization (GA-PSO) algorithm. *In 2019 International Conference on Intelligent Sustainable Systems (ICISS)* (pp. 29-34). IEEE.

[25] Masadeh, R., Sharieh, A., & Mahafzah, B. (2019). Humpback whale optimization algorithm based on vocal behavior for task scheduling in cloud computing. *International Journal of Advanced Science and Technology*, 13(3), 121-140.

[26] Chen, X., Cheng, L., Liu, C., Liu, Q., Liu, J., Mao, Y., & Murphy, J. (2020). A WOA-based optimization approach for task scheduling in cloud computing systems. *IEEE Systems journal*, 14(3), 3117-3128.

[27] Masadeh, R., Alsharman, N., Sharieh, A., Mahafzah, B. A., & Abdulrahman, A. (2021). Task scheduling on cloud computing based on sea lion optimization algorithm. *International Journal of Web Information Systems*, 17(2), 99-116.

[28] Visser, I., Smith, T., Bullock, I., Green, G., Carlsson, O. L., & Imberti, S. (2008). Antarctic peninsula killer whales (Orcinus orca) hunt seals and a penguin on floating ice. *Marine Mammal Science*, 24(1), 225-234.

[29] Muhammad Ibrahim, February 26, 2020, "Dataset for Task scheduling in Cloud using CLoudsim", *IEEE Dataport*, doi: https://dx.doi.org/10.21227/px5b-b729.

[30] Almomani, O., Almaiah, M. A., Madi, M., Alsaaidah, A., Almomani, M. A., & Smadi, S. (2023, October). Reconnaissance attack detection via boosting machine learning classifiers. *In AIP Conference Proceedings* (Vol. 2979, No. 1). AIP Publishing..

[31] Almomani, O., Alsaaidah, A., Shareha, A. A. A., Alzaqebah, A., & Almomani, M. (2024). Performance Evaluation of Machine Learning Classifiers for Predicting Denial-of-Service Attack in Internet of Things. *Performance Evaluation*, 15(1).