Инж. Шаттур Ходаа

Студент аспирантской подготовки,

Департамент системной инженерии и компьютерных сетей

Факультет информатики, Университет Аль-Баас,

Сирия, г. Хомс

Др. Аббуд Мохссен

научный руководитель, профессор кафедры « Департамент системной инженерии и компьютерных сетей »

Факультет информатики, Университет Аль-Баас,

Сирия, г. Хомс

ОПТИМИЗАЦИЯ ВРЕМЕНИ ОТВЕТА И ОБРАБОТКИ С ПОМОЩЬЮ АЛГОРИТМА FIREFLY В CLOUD ANALYST

Аннотация: *Это исследование направлено на улучшение балансировки нагрузки в туманных вычислениях, предложив гибридный алгоритм, сочетающий в себе работу алгоритмов активный и Файрфлай. Затем изучая оценку его производительности и сравнивая ее с рядом алгоритмов балансировки нагрузки в облачной аналитике, изучая влияние параметров производительности: среднего времени отклика для пользователей и времени обработки для центров обработки данных.*

Ключевые слова: *туманные вычисления, балансировка нагрузки, Облачный аналитик, Активный алгоритм, алгоритм Файрфлая*

Eng Houda Shattour
Department of Systems Engineering and Computer Networks
Faculty of Informatics Engineering Al-Baath University Homs
Dr Mohssen Abboud
Department of Systems Engineering and Computer Networks
Faculty of Informatics Engineering Al-Baath University Homs

OPTIMIZING RESPONSE AND PROCESSING TIMES USING FIREFLY ALGORITHM AT CLOUD ANALYST

Annotation: *This research aims to improve load balancing in fog computing, by proposing a hybrid algorithm, which combine the work of both Active, and Firefly algorithms. Then studying the evaluation of its performance and comparing it with a number of load balancing algorithms in the cloud analyst, by studying the effect of performance parameters: average response time for users and processing time for data centers.*

Keywords: *Fog computing, load balancing, cloud analyst, Active algorithm,* Firefly algorithm.

1. Introduction:

Fog computing has become a new trend in the domain of Internet of things and cloud computing applications. It is a new model to achieve the availability, flexibility and better responding time. In spite of that, there is so many challenges facing computing environments such as the misuse of the resources and load-balancing between them, which has a major effect on performance. The requirement of effective and robust load-balancing algorithms is one of the most important issues in this field.

Previous reference studies agreed to present a specific algorithm and compare it with other algorithms for evaluating results through the cloud analyst tool, study

[5] presented an effective algorithm for load balancing in fog computing, which is the Min Min algorithm, it showed close results to the RR algorithm.

Study[6] presented a solution to reduce the response time and delay within the structure of the smart electric system by applying FCFS algorithm (First Come First serve) but did not exceed in some stages the RR and Throttled algorithm, also in study [7] the modified Honey Bee algorithm was proposed, but without comparison with other algorithms.

In this study [8] an efficient load-balancing algorithm was proposed for real time tasks only, as for the study [9], another algorithm was proposed that works to duplicate data to balance the load, but it led to a high cost of data transfer rate.

2. Load Balancing Algorithms included in the simulator:

Round Robin:

It is one of the simplest algorithms. It distributes the load to virtual machines in a circular order regardless of the processing power when assigning the task [2].

Throttled:

The primary step is that the load balancer maintains an indexed table of the status of virtual machines in the data center (whether they are available or busy). When a new request is assigned, the load balancer assigns it to the first available virtual machine [3].

Active:

A dynamic algorithm maintains information about each virtual machine with the number of requests assigned to it at the current time. When a new request is assigned, the machine with the lowest number of requests is searched for, when there is more than one machine with the same request number.

3. The proposed algorithm:

The firefly algorithm is one of the swarm intelligence algorithms that depend on the behavior of the swarm like fish, insects and birds in nature. The firefly produces short and rhythmic flashes through the bioluminescence process to attract partners or prey, attraction is measured by two factors: (Light intensity and distance), whereas, the intensity of light decreases as the distance between the source and distention increases, because the light is absorbed by the conducting medium and

```
Begin
    1) Objective function: f(x),   x = (x_1, x_2, ..., x_d);
    2) Generate an initial population of fireflies x_i   (i = 1, 2, ..., n);.
    3) Formulate light intensity I so that it is associated with f(x)
       (for example, for maximization problems, I ∝ f(x) or simply I = f(x);)
    4) Define absorption coefficient γ

    While (t < MaxGeneration)
        for i = 1 : n (all n fireflies)
            for j = 1 : i (n fireflies)
                if (I_j > I_i),
                    Vary attractiveness with distance r via exp(−γ r);
                    move firefly i towards j;
                    Evaluate new solutions and update light intensity;
                end if
            end for j
        end for i
        Rank fireflies and find the current best;
    end while

    Post-processing the results and visualization;

end
```

Figure 1 Firefly algorithm pseudo code

thus attraction decreases [1][4].

3.1 Description of the proposed algorithm:

The proposed algorithm works with the following steps:

Active algorithm runs to get a virtual machine with the fewest number of current requests regardless of machine capacity (vm1).

Then Firefly algorithm runs to get the best virtual machine at the current time, (vm2), both results may be equal, otherwise the randomization function is chosen between vm1 and vm2.

The proposed algorithm combines the characteristics of the two algorithms, as it has information on the number of current requests and the ability of the virtual machine in terms of bandwidth to achieve effective load balancing and reduce the disadvantages of Active algorithm as mentioned previously.

4. Experimental

The first step is to study the problem with the delay factor in the network by configuring the user base settings with the data center. Each one of them in a separate region. Ten virtual machines were identified within one data center and six users; using average response time were used for all Users and average processing time for the data center for comparison and results evaluation processes.

Table 1. data center parameters

Data Center	#VMs	Image Size	Memory	BW
DC1	10	10000	215	1000

Table 2. user bases parameters

Name	Region	Time	start	End
UB1	1	GMT-6:00	11	12
UB2	1	GMT-4:00	12	11
UB3	5	GMT-1:00	51	55
UB4	1	GMT-6:00	1	1
UB5	4	GMT-2:00	51	51
UB6	2	GMT-10:00	9	11

5. Results and Discussion:

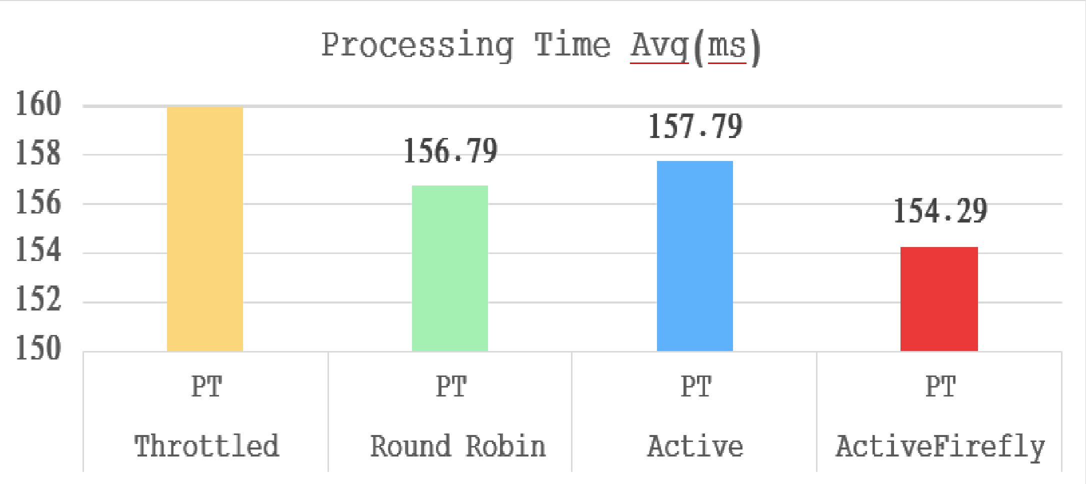

Figure 2: Algorithm Results Comparison (Average Processing Time)

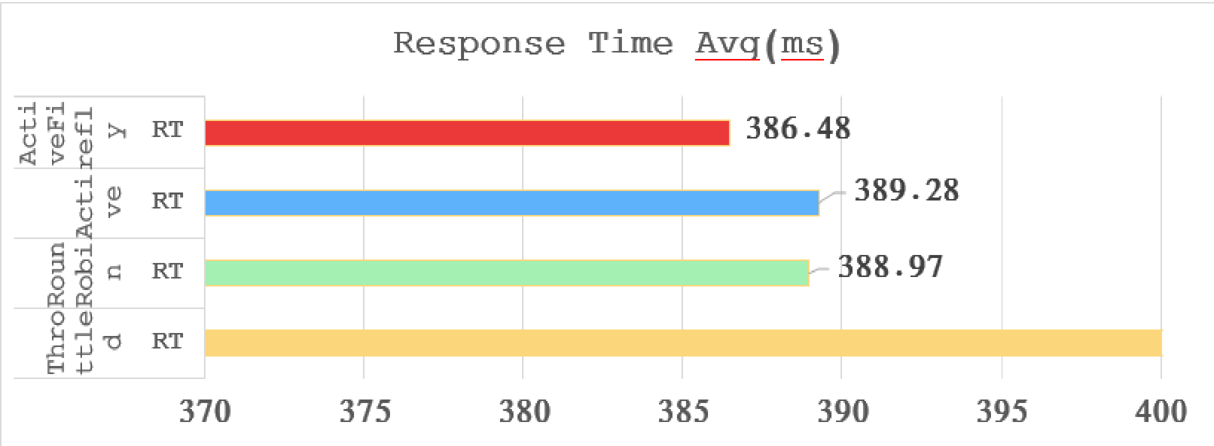

Figure3: Algorithm Results Comparison (Average Response Time)

Results:

The proposed algorithm scored the best result for the average response time for a group of users (386.48 ms) because the tasks were less likely to wait (wait time) as the tasks became evenly distributed among the virtual machines.

The proposed algorithm scored the best result for the average processing time of the data center of (154.29 ms), because no machine is overloaded with more tasks than others are.

References:

1. Alabajee, M. (2013). The Investigation of Software Reliability Models through Swarm Intelligence.

2. Buyya, R. (2009). CloudAnalyst: A CloudSim-based tool for modelling and analysis of large-scale cloud computing environments. Distrib. Comput. Proj. Csse Dept., Univ. Melb. 433-659.

3. Ivanisenko, I. (2015). Methods and Algorithms of load balancing. In International Journal “Information technologies and knowledge” (Vol. 9, No. 4, pp. 340-375). ITEA, Sofia.

4. Kaur, G., & Kaur, K. (2017). An adaptive firefly algorithm for load balancing in cloud computing. In Proceedings of sixth international conference on soft computing for problem solving (pp. 63-72). Springer, Singapore.

5. Manju, A. B., & Sumathy, S. (2019). Efficient load balancing algorithm for task preprocessing in fog computing environment. In Smart Intelligent Computing and Applications (pp. 291-298). Springer, Singapore.

6. Saeed, F., Javaid, N., Zubair, M., Ismail, M., Zakria, M., Ashraf, M. H., & Kamal, M. B. (2018, September). Load Balancing on Cloud Analyst Using First Come First Serve Scheduling Algorithm. In International Conference on Intelligent Networking and Collaborative Systems (pp. 463-472). Springer, Cham.

7. Verma, M., & Yadav, N. B. A. K. (2015). An architecture for load balancing techniques for Fog computing environment. International Journal of Computer Science and Communication, 8(2), 43-49.

8. Verma, M., Bhardwaj, N., & Yadav, A. K. (2016). Real time efficient scheduling algorithm for load balancing in fog computing environment. Int. J. Inf. Technol. Comput. Sci, 8(4), 1-10.

9. Verma, S., Yadav, A. K., Motwani, D., Raw, R. S., & Singh, H. K. (2016, March). An efficient data replication and load balancing technique for fog computing environment. In 2016 3rd International Conference on Computing for Sustainable Global Development (indiacom) (pp. 2888-2895). IEEE.