Reactive vs Predictive Live Migration in Edge Cloud

*

Daisy Rani, Saptarshi Ghosh, Tasos Dagiuklas
Division of Computer Science
London South Bank University
London, U.K
email:mymail.daisy@gmail.com, ghos5,tdagiuklas@lsbu.ac.uk

***Abstract*—Migrating services in an edge-cloud environment poses unique challenges, including heterogeneous environments, potential failures, and uneven resource distribution. This paper studies and evaluate reactive and predictive migration approaches to support live migration in case of edge cloud computing failures. Telemetry information relate to edge cloud computing have been considered to trigger migration, whereas deadlock prevention algorithm has been used to determine and select the target device to migrate services. The paper evaluates these strategies by comparing resource utilization, assessing differences between predictive and reactive migration and handling multiple migrations for tenants hosting numerous applications. Experimental results have shown that predictive migration can reduce the downtime of the hosted services. Additionally, the total migration cost can be increased for both scenarios where the containers can be migrated to different edge devices due to lack of available resources.**

***Index Terms*—edge, migration, reactive, predictive, MADM, LSTM, banker's algorithm, edge computing**

I. Introduction

Edge computing is a decentralized computing paradigm that brings computation and data storage closer to the location where it is needed, data is processed and analyzed at or near the data source (the "edge") [20]. To maintain the availability and Qos of edge, several different techniques of resource allocation must be used to support migration.

Migration in edge computing can be a complex and challenging process due to the unique characteristics and constraints of edge environments as in edge there are more occurrence of multiple failure. Edge may also support multiple tenants, which requires isolation from each other for privacy. One tenant can host few application making the multiple container migration process rather complex. Migration can be either reactive or predictive. In the case of reactive, a failure is detected by measuring the computing resources. In the case of preventive, AI/ML models can be used to predict the occurrence of a failure. In both cases, hosted applications must be transferred to another edge device that has enough resources to support the transferred applications.

The aim of this paper is to compare reactive vs preventive migration when a tenant that may host several applications must carry out migration. Different metrics such as downtime, cost of migration and completion for different number of hosted applications have been used for this comparison.

The paper is organised as follows: Section 2 presents the state of the art and the paper contribution; Section 3 describes the methodology and both reactive and preventive algorithms that have been developed; Section 4 outlines the experimental methodology, scenarios and simulation results and Section 5 concludes the findings.

II. STATE-OF-THE-ART

In Edge network for optimal resource management and quality of service different algorithms are being used such as predictive, reactive and hybrid approach [8]. Other algorithms like load balancing algorithms distribute workloads evenly across edge nodes, by taking into account resource availability and network conditions for improved balance [9]. Deadlock avoidance algorithms prevent disruptions in edge migrations. A specialized deadlock avoidance algorithm ensures smooth migrations as discussed in [10]. Further in this section the current work on migration is discussed.

A. Edge-based Migration Techniques

The first research paper by T. G. Rodrigues, K. Suto, H. Nishiyama and N. Kato aims for low Service Delay, resulting in high Quality of Service by focusing on

both computation and communication elements to trigger migration [1]. This approach accommodates diverse services, using a mathematical model to configure virtual server migration and transmission power. Experimental results consistently favor this approach for minimizing Service Delay.

The Authors in [2] explore methods to optimize replica placement and migration in edge cloud environments, aiming to boost data transfer speed, reduce bandwidth usage, enhance reliability, and maintain load balance. The evaluation asserts that the proposed solution notably reduces downtime, even in scenarios with high state update volumes. The Authors in [3] propose an architecture that has cognitive capabilities and knowledge about the network environment to offer elastic cognitive computing services for energy efficiency and a superior Quality of Experience (QoE). Experimental results demonstrate that the architecture delivers ultra-low latency, an excellent user experience, improved service quality, high energy efficiency and resource savings. The Next research paper is efficiently scheduling live container migrations in large-scale edge computing environments for multi container migration. It offers a novel approach to address the complex issue of scheduling multiple live container migrations in large-scale edge computing environments to optimize resource utilization and minimize downtime in edge computing scenarios [4]. The Follow Me Cloud Prototype (FMC) facilitates seamless service migration between data centres, often acting as edge servers, influenced by factors like geographical distance and workload to balance migration cost and enhance mobile user Quality of Service (QoS). FMC architecture comprises an FMC controller and an edge server/gateway mapping entity working alongside mobile cloud components like data centres and gateways to manage distributed edge servers in mobile operator networks [5]. Markov Decision Processes (MDPs) guide service migration decisions. The one-dimensional MDP model suits linear mobile user movements, using states for distances from edge servers and actions for migration choices. Transition probabilities and rewards aid optimal decisions. A two-dimensional MDP model addresses 2D mobile user movement scenarios, optimizing service placement based on distance and network topology while managing state complexity [6]. Time window-based service migration minimizes costs over a specified period with predictive look-ahead windows. It determines optimal window sizes, finding service placement sequences based on prediction costs through a shortest-path approach, balancing prediction deviation and time division impacts [7].

B. Contribution

Existing research works lack in making a comparison between reactive and preventive migration. We evaluate the complexity for both reactive and preventive migration in a new edge instance. The reactive migration raises the need to make decision by considering metrics such as available CPU and memory over the time and considering their dependencies. Multiple Attribute Decision Making (MADM) and Analytic Hierarchy Process (AHP) are used in decision-making [17]. MADM methods help in ranking or choosing alternatives based on these multiple attributes. AHP is one of the various methods used in MADM. It decomposes the decision problem into a hierarchy of criteria and sub-criteria, performs pairwise comparisons to determine the relative importance or weights of these criteria and then evaluates the alternatives against each criterion. At last, overall scores for each alternative based on the weighted criteria is calculated and the alternative with the highest score is selecting as the preferred choice. Predictive migration relies on historical data and machine learning to proactively forecast resource demands and improve large-scale edge environment migrations [11]. Reactive migration addresses real-time resource issues like failures, optimizing resource use and reducing service interruptions. It suits scenarios where predictive models may falter.

III. SYSTEM MODEL

A. Methodology and Design

The network structure consists of a series of edge instances that are connected to the edge cloud orchestration. The network uses both control flow and data flow paths. Data flow path is used among the network devices for usual data exchange, where the control flow path is used for the exchange of telemetry related information between the device and the Orchestration. The orchestration has been designed to continuously monitor the Edge network and get the telemetry of all devices, containers related to CPU, memory, network, and storage. Network data such as round-trip time are collected. All the observations are stored in a time series database. Two different migration decision algorithms have been implemented: reactive and predictive. Figure 1, illustrate the system model to support both reactive and predictive migration.

B. Preventive Migration Algorithm

LSTM trained model is used to predict the upcoming failure or overhead in the network of edge devices.

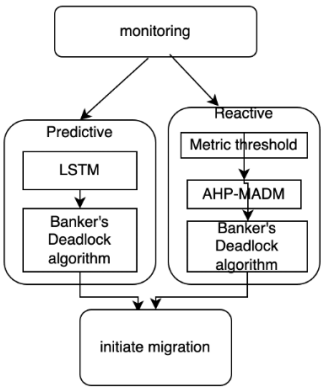

Fig. 1: System Model

LSTMs consist of memory cells that can store and retrieve information over long sequences [16]. These memory cells have three gates: an input gate, an output gate, and a forget gate. In this research study, LSTM model has been created using time-series CPU and memory data from a real-time experimentation. An experimental testbed with Raspberry PIs have been used to collect real-time data with respect to CPU and memory usage and track node's failures [18]. Historical data from 1 week have been used to train LSTM. The following formula has been used to determine resource utilisation:

The Equation used for selecting the destination Edge Device, where the victim container must be migrated :

$$f(E_d) = min(\lambda_1 \upsilon(E_i) + \lambda_2 RTT(E_i)) \quad (1)$$

where, $\upsilon : utilisation, RTT : Round - TripTime$, *Ed*: Destination node, *Ei*: Edge nodes,
$\lambda_1 + \lambda_2 = 1$ (2)

$$\upsilon = \upsilon(CPU) + \upsilon(nw) + \upsilon(storage) + \upsilon(mem) \quad (3)$$

Calculation of the expected required resource by the stressed container:

$$\upsilon(C_i) = \mu(C_i) + 2\sigma(C_i) \quad (4)$$

where
$\upsilon(C_i) : utilizationofcontainer, \mu : mean, \sigma : Standarddeviation$

Therefore, 80% of the data have been used to train the LSTM model and 20% of the data have been used to validate the model. The following Figure 2 illustrates the comparison between the actual vs the predicted (LSTM) value.

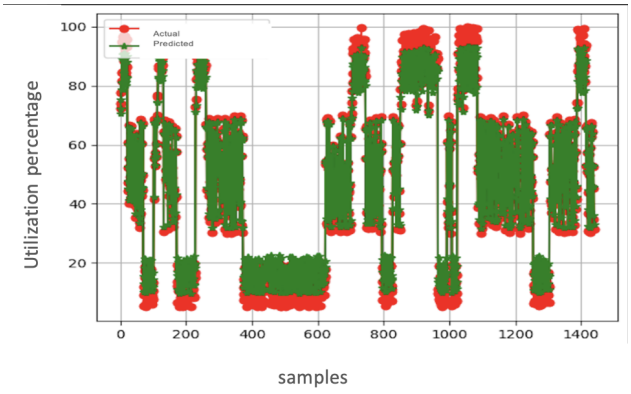

Fig. 2: predicted vs actual values

In this respect, Banker's algorithms has been used by taking into account the resource requirements of the container in terms of CPU and memory. In case of a tenant, an aggregate request is made from all hosted containers. If there is no available edge device, the migration will segment the containers separately. The algorithm for all the steps is presented in Algorithm 2.

C. Reactive Migration Algorithm

In the reactive algorithm, MADM-AHP has been considered to determine the unified metric from the computing resources (CPU, memory). Under MADM-AHP, resource attributes are defined as criteria, for each pair to determine their relative importance with respect to a threshold [17]. Similar to the previous case, Banker's algorithm has been used to select an edge device to migrate the container(s), so that deadlock avoidance criterion is met. Banker's algorithm is used for the migration selection. The Banker's algorithm is a deadlock avoidance algorithm [13]. Its primary purpose is to ensure that the resources can safely be requested and released without leading to a deadlock situation. The Banker's algorithm helps prevent such deadlocks by carefully allocating and managing resources.

Algorithm 1: Reactive Migration

```
Result: migration started
host list;
while 1 do
    each host  metric threshold;
    if migration decision then
        MADM for selecting destination;
        Banker's Algorithm for selecting
         destination Node;
        if Single node is sufficient for all
         container then
            trigger migration;
        else
            while all containers get selected do
                segment containers;
            end
            trigger migration;
        end
    else
        sleep 1;
    end
end
```

Algorithm 2: Predictive Migration

```
Result: migration started
host list;
while 1 do
    each host  use LSTM model on host data;
    if migration decision then
        Banker's Algorithm for selecting
         destination Node;
        if Single node is sufficient for all
         container then
            trigger migration;
        else
            while all containers get selected do
                segment containers;
            end
            trigger migration;
        end
    else
        sleep 1;
    end
end
```

IV. SCENARIOS AND RESULTS

A. Experimental set-up

The implementation set-up is shown in Figure 3. An edge network has been designed and developed using GNS-3 [14]. Some of the characteristics of the experimental set-up are the following: the number of edge devices have been customised and each node can run to 1 to 10 containers. Psutil [18] has been used to collect computing data from devices and store them in InfluxDB [15]. Both reactive and preventive migration algorithms have been implemented in Python. We have run three different scenarios with 10, 50 and 100 edge nodes. The orchestration application situated outside the edge network, establishes connections with all edge devices and comprises two modules: the monitoring agent and the migrating agent. Both these modules are linked to a database and a dashboard. The monitoring agent continuously gathers and stores data from all devices within the network, focusing on resource utilization of the devices and their contained containers. These observations are consistently recorded in a time-series database. The migrating agent plays a crucial role in the system, reading data from the database at predefined intervals, computing moving averages, and comparing the results with predefined thresholds.

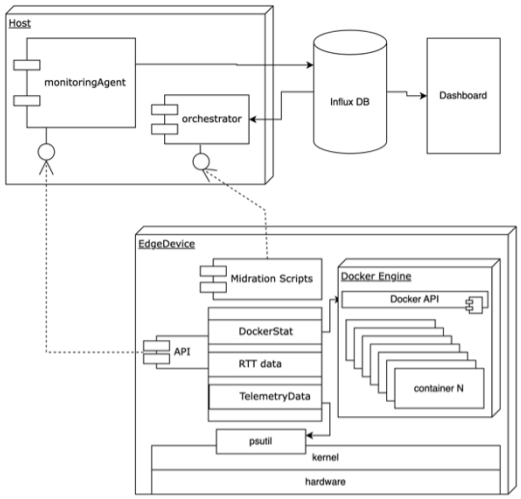

Fig. 3: Implementation set-up

B. Scenarios

The experimental performance metrics include the comparison between reactive and predictive migration for different number of edge nodes increases, running different number of containers. We are considering the following parameters:

- Downtime: In predictive mode, the downtime aligns with the migration time. In contrast, for the reactive mode, downtime can increase if a container fails

during the monitoring cycle, extending the downtime to one complete cycle time (This is the time to deploy the container plus the time required to start monitoring it). This results in a minimum downtime equivalent to the migration time and a maximum downtime equivalent to the migration time plus one monitoring cycle time.

- Cost of migration: The cost of migration measures the time required for saving a container image, transferring it to the destination node, and restoring the container to the new destination. This measurement is consistent for both the predictive and reactive methods, and it's evaluated with the number of nodes set at 5, 10, and 20. We analyze the cost of migration relative to the number of nodes, assessing the failure rate as the number of containers within an edge node grows, and investigating the impact of tenant hosting multiple containers by comparing the migration of containers to a single node versus distributing them across multiple nodes.

C. Experimental Results Discussion

Figure 5 illustrates the downtime for both algorithms for different number of nodes (10, 50, 100). It is shown (as expected), that the preventive algorithm detects in advance the potential failure, providing a reduction to downtime for all scenarios, by approximately 37%. Figure 4 and Figure 7, illustrates the total average migration costs for both scenarios. As expected, the transfer time of the containers contributes mainly (67%) to the total migration cost. As the number of edge nodes increase, the migration time gets larger since more traffic contention takes places within the backhauling links. The following Figure 6 illustrates the average migration cost for single versus segmented container migration for both algorithms. As the number of containers increase, the migration costs for segmented the containers to multiple instances goes higher due to the complexity of Banker selection process.

V. CONCLUSION

This paper makes a comparison between reactive vs preventive live migration foe edge by using parameters such as migration cost, downtime and tenant migration complexity. For this purpose, an edge-based experimental testbed has been built in GNS-3 integrating telemetry, monitoring and assessment along with database tools. The migrations have been tested in both small and complex network scenarios, demonstrating its scalability. A comparison between reactive and predictive approaches revealed the advantages of predictive migration in terms

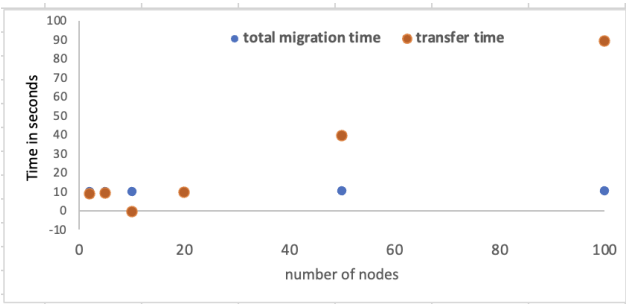

Fig. 4: migration time along with increase of node

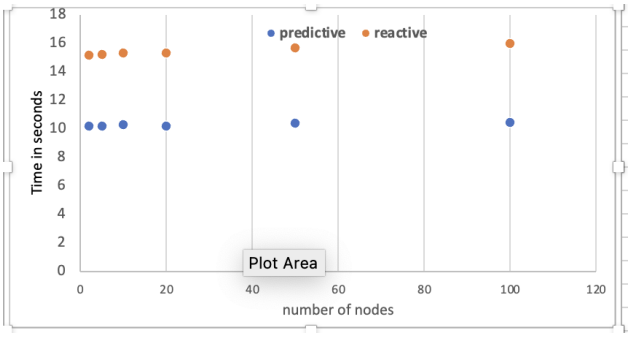

Fig. 5: maximum downtime in predictive vs reactive migration

of resource utilization. The impact of window size on prediction accuracy was observed, emphasizing the need for an optimal balance of loss function and time for migration. Less containers goes to unstable state in predictive migration. Both reactive and predictive migration strategies proved effective, with a focus on minimizing image transmission time and achieving stable container states. Multiple container migration focused for supporting multi-application tenants and addressing edge computing challenges on multi failure. Overall, this research

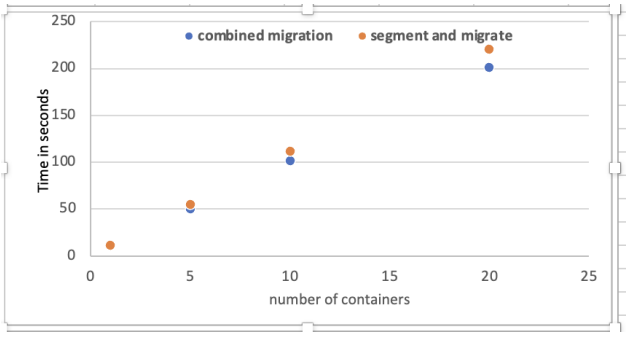

Fig. 6: migration with segment

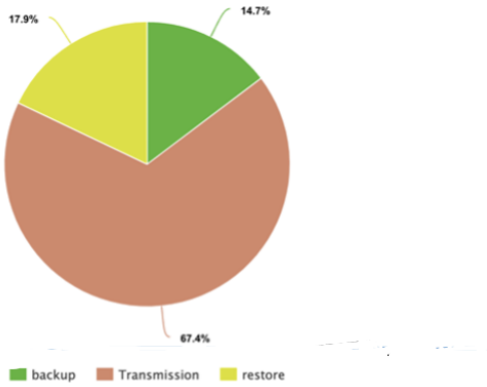

Fig. 7: Migration cost

contributes to enhancing quality of service and resource management in edge computing environments. As part of continuous work, this study can be expanded to encompass various container types. Additionally, exploring concurrent migration processes for scenarios involving multiple containers can enhance migration efficiency.

REFERENCES

[1] T. G. Rodrigues, K. Suto, H. Nishiyama and N. Kato, "Hybrid Method for Minimizing Service Delay in Edge Cloud Computing Through VM Migration and Transmission Power Control," in IEEE Transactions on Computers, vol. 66, no. 5, pp. 810-819, 1 May 2017, doi: 10.1109/TC.2016.2620469.

[2] C. Li, Y. Wang, H. Tang, and Y. Luo, "Dynamic multi-objective optimized replica placement and migration strategies for SaaS applications in edge cloud," Future Generation Computer Systems, vol. 100, pp. 921–937, Nov. 2019, doi: https://doi.org/10.1016/j.future.2019.05.003.

[3] M. Chen, W. Li, G. Fortino, Y. Hao, L. Hu, and I. Humar, "A Dynamic Service Migration Mechanism in Edge Cognitive Computing," ACM Transactions on Internet Technology, vol. 19, no. 2, pp. 1–15, Apr. 2019, doi: https://doi.org/10.1145/3239565.

[4] T. He, A. N. Toosi, and R. Buyya, "Efficient Large-Scale Multiple Migration Planning and Scheduling in SDN-enabled Edge Computing," arXiv.org, Nov. 17, 2021. https://arxiv.org/abs/2111.08936

[5] T. Taleb, A. Ksentini and P. A. Frangoudis, "Follow-Me Cloud: When Cloud Services Follow Mobile Users," in IEEE Transactions on Cloud Computing, vol. 7, no. 2, pp. 369-382, 1 April-June 2019, doi: 10.1109/TCC.2016.2525987.

[6] A. Ksentini, T. Taleb and M. Chen, "A Markov Decision Process-based service migration procedure for follow me cloud," 2014 IEEE International Conference on Communications (ICC), Sydney, NSW, Australia, 2014, pp. 1350-1354, doi: 10.1109/ICC.2014.6883509.

[7] S. Wang, R. Urgaonkar, T. He, K. Chan, M. Zafer and K. K. Leung, "Dynamic Service Placement for Mobile Micro-Clouds with Predicted Future Costs," in IEEE Transactions on Parallel and Distributed Systems, vol. 28, no. 4, pp. 1002-1016, 1 April 2017, doi: 10.1109/TPDS.2016.2604814.

[8] T. Liu, S. Ni, X. Li, Y. Zhu, L. Kong and Y. Yang, "Deep Reinforcement Learning Based Approach for Online Service Placement and Computation Resource Allocation in Edge Computing," in IEEE Transactions on Mobile Computing, vol. 22, no. 7, pp. 3870-3881, 1 July 2023, doi: 10.1109/TMC.2022.3148254.

[9] S. P. Singh, R. Kumar, A. Sharma, and A. Nayyar, "Leveraging energy-efficient load balancing algorithms in fog computing," Concurrency and Computation: Practice and Experience, Jul. 2020, doi: https://doi.org/10.1002/cpe.5913.

[10] E. E. Ugwuanyi, S. Ghosh, M. Iqbal and T. Dagiuklas, "Reliable Resource Provisioning Using Bankers' Deadlock Avoidance Algorithm in MEC for Industrial IoT," in IEEE Access, vol. 6, pp. 43327-43335, 2018, doi: 10.1109/ACCESS.2018.2857726.

[11] T. He, A. N. Toosi, and R. Buyya, "Efficient Large-Scale Multiple Migration Planning and Scheduling in SDN-enabled Edge Computing," arXiv.org, Nov. 17, 2021. https://arxiv.org/abs/2111.08936

[12] V. Tsakanikas et al "An intelligent model for supporting edge migration for virtual function chains in next generation internet of things. Scientific reports, 13(1), p.1063."

[13] S. . -D. Lang, "An extended banker's algorithm for deadlock avoidance," in IEEE Transactions on Software Engineering, vol. 25, no. 3, pp. 428-432, May-June 1999, doi: 10.1109/32.798330.

[14] P. Gil, G. J. Garcia, A. Delgado, R. M. Medina, A. Calderón and P. Marti, "Computer networks virtualization with GNS3: Evaluating a solution to optimize resources and achieve a distance learning," 2014 IEEE Frontiers in Education Conference (FIE) Proceedings, Madrid, Spain, 2014, pp. 1-4, doi: 10.1109/FIE.2014.7044343.

[15] A. Aggoune and Z. Benratem, "ECG Data Visualization: Combining the power of Grafana and InfluxDB," 2023 International Conference on Advances in Electronics, Control and Communication Systems (ICAECCS), BLIDA, Algeria, 2023, pp. 1-6, doi: 10.1109/ICAECCS56710.2023.10104857.

[16] H. Jing et al., "LSTM-Based Service Migration for Pervasive Cloud Computing," IEEE Xplore, Jul. 01, 2018. https://ieeexplore.ieee.org/abstract/document/8726684 (accessed Sep. 13, 2023).

[17] M. K. Moghadam, S. Bonsall, J. Wang, and A. Wall, "Application of Multiple Attribute Decision-Making (MADM) and Analytical Hierarchy Process (AHP) Methods in the Selection Decisions for a Container Yard Operating System," Marine Technology Society Journal, vol. 43, no. 3, pp. 34–50, Aug. 2009, doi: https://doi.org/10.4031/mtsj.43.3.3.

[18] Preeth E N, F. J. P. Mulerickal, B. Paul and Y. Sastri, "Evaluation of Docker containers based on hardware utilization," 2015 International Conference on Control Communication Computing India (ICCC), Trivandrum, India, 2015, pp. 697-700, doi: 10.1109/ICCC.2015.7432984.

[19] B. Balon and M. Simić, "Using Raspberry Pi Computers in Education," 2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), Opatija, Croatia, 2019, pp. 671-676, doi: 10.23919/MIPRO.2019.8756967.

[20] W. Shi, J. Cao, Q. Zhang, Y. Li and L. Xu, "Edge Computing: Vision and Challenges," in IEEE Internet of Things Journal, vol. 3, no. 5, pp. 637-646, Oct. 2016, doi: 10.1109/JIOT.2016.2579198.