International Journal of Advances in Engineering and Management (IJAEM)
Volume 3, Issue 5 May 2021, pp: 919-923 www.ijaem.net ISSN: 2395-5252

Adaptive Decentralized Computation for Deep Reinforcement Multitak in Mobile Edge Computing Using DSRS

Prof.T.Geetha M.E.,M.B.A., Assistant Profesor,[1] P.Maheswari[2], C.Monisha,[3] A.Sathya,[4] S.Suruthi[5]

Dhanalakshmi Srinivasan Engineering College (Autonomous), Perambalur.

Submitted: 05-05-2021 Revised: 17-05-2021 Accepted: 20-05-2021

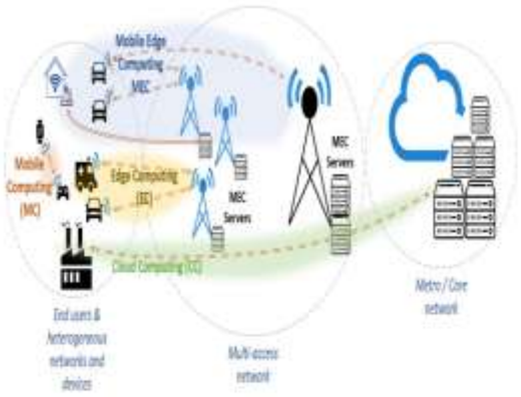
ABSTRACT: Nowadays, mobile devices are responsible for processing more and more Computational intensive tasks, such as data processing, artificial intelligence, and virtual reality. Despite the development of mobile devices, these devices may not be able to process all their tasks locally with a low latency due to their limited computational resources. To facilitate efficient task processing, mobile edge computing (MEC), also known as fog computing and multi-accessed gecomputing, is introduced. MEC facilitates mobile devices to offload their computational intensive tasks to near by edge nodes for processing in order to reduce the task processing delay. It can also reduce the ratio of dropped tasks for those delay - sensitive tasks. In MEC, there are two main questions related to task offloading. The first question is whether a mobile device should offload its task to an edge node or not. These condquestion is that if a mobile device decides toper form offloading, then which edge node should the device offload its task to. In this work, we consider non - divisible and delay - sensitive tasks as well as edge load dynamics, and formulate a task offloading problem to minimize the expected long-term cost. We propose a model - free deep reinforcement learning - based distributed algorithm, where each device an determine its offloading decision without knowing the task models and offloading decision of other devices. To improve the estimation of the long - term cost in the algorithm, we incorporate the long short - term memory (LSTM), dueling deep Q - network (DQN), and double DQN techniques. Simulation results with 50 mobile devices and five edge nodes show that the proposed algorithm can reduce the ratio of dropped tasks and average task delay by 86.4% − 95.4% and 18.0% −30.1%, respectively, when compared with several existing algorithms.

I. INTRODUCTION

MOBILE EDGE COMPUTING

The centralized form of remote computing resources is not well suited with massive traffic originated from geographically distributed edge devices. Therefore, pushing the servers to the edge of the network is becoming an essential trend. Propagation of the edge devices creates a necessity for the applications to process at least some of the data at the edge rather than carrying them to the remote data centres for minimizing not only the energy consumption but also service delay. Additionally, next generation devices are of small size with lower power, while the computation tasks are generally intensive and latency critical. Due to the limitation of physical size, computation capacity and low battery power, some computation applications cannot be performed smoothly by the mobile devices (MDs). So, enhancing the computing capability and prolonging the battery life of a mobile device is a key design challenge. Hence, energy-efficient data processing is obviously vital for battery-empowered MDs. To address the current issues, mobile-edge computing (MEC) has emerged as a promising technique, which offers computation capability within the radio access network in contrast to conventional cloud computing systems that use remote public clouds. By offloading the computation intensive tasks from the MDs to the nearby MEC servers, the quality of computation experience, including the latency and device energy consumption, could be greatly improved.

MEC Process



II. LITERATURE SURVEY

2.1 Title : A Survey On Mobile Edge Computing : The Communication Perspective

Author : Changsheng You, Jun Zhang, Kaibin Huang-2017

Driven by the visions of Internet of Things and 5G communications, recent years have seen a paradigm shift in mobile computing, from the centralized mobile cloud computing toward mobile edge computing (MEC). The main feature of MEC is to push mobile computing, network control and storage to the network edges (e.g., base stations and access points) so as to enable computation-intensive and latency-critical applications at the resource-limited mobile devices. MEC promises dramatic reduction in latency and mobile energy consumption, tackling the key challenges for materializing 5G vision. This paper provides a comprehensive survey of the state-of-the-art MEC research with a focus on joint radio-and-computational resource management. We also discuss a set of issues, challenges, and future research directions for MEC research, including MEC system deployment, cache-enabled MEC, mobility management for MEC, green MEC, as well as privacy-aware MEC.

ADVANTAGES

- Reduction in latency
- Reduce mobile energy consumption

DISADVANTAGES

- High complexity process
- Low performance

2.2 Title : Survey On Multi-Access Edge Computing For Internet Of Things Realization

Author : Pawani Porambage, Jude Okwuibe - 2018

The Internet of Things (IoT) has recently advanced from an experimental technology to what will become the backbone of future customer value for both product and service sector businesses. This underscores the cardinal role of IoT on the journey towards the fifth generation (5G) of wireless communication systems. IoT technologies augmented with intelligent and big data analytics are expected to rapidly change the landscape of myriads of application domains ranging from health care to smart cities and industrial automations. MEC will inspire the development of myriads of applications and services with demand for ultra low latency and high Quality of Service (QoS) due to its dense geographical distribution and wide support for mobility. MEC is therefore an important enabler of IoT applications and services which require real-time operations. In this survey, we provide a holistic overview on the exploitation of MEC technology for the realization of IoT applications and their synergies. We further discuss the technical aspects of enabling MEC in IoT and provide some insight into various other integration technologies therein.

ADVANTAGES

- High-bandwidth
- Low latency

DISADVANTAGES

- Less quality of service
- Poor bandwidth utilization

III. SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

Edge nodes may have limited processing capacities, so the processing capacity that an edge node allocated to a mobile device depends on the load level at the edge node. When a large number of mobile devices offload their tasks to the same edge node, the load at that edge node can be high, and hence those offloaded tasks may experience large processing delay.

Distributed task offloading algorithms considering the load levels at the edge nodes, where each mobile device makes its offloading decision in a decentralized manner. The designing such a distributed algorithm is challenging. This is because when a device makes an offloading decision, the device does not know a priori the load levels at the edge nodes, since the load also depends on the offloading decisions and task models of other mobile devices.

3.1.1 DISADVANTAGES

- Computational task offloading problem
- Limited computational resources

- Large processing delay
- It affect the load level dynamics

3.2 PROPOSED SYSTEM

A DRL-based distributed offloading algorithm for the MEC system is proposed. In the proposed algorithm, each mobile device can determine the offloading decision in a decentralized manner using the information observed locally, including the size of its task, the information of its queues, and the historical load levels at the edge nodes. In addition, the proposed algorithm can handle the time-varying system environments, including the arrival of new tasks, the computational requirement of each task, and the offloading decisions of other mobile devices.

3.2.1 CONTRIBUTION

Task Offloading Problem for the MEC System: We formulate a task offloading problem taking into account the load level dynamics at the edge nodes to minimize the expected long-term cost (considering the delay of the tasks and the penalties for those tasks being dropped).

DRL-based Task Offloading Algorithm: To achieve the expected long-term cost minimization considering the unknown load dynamics at the edge nodes, we propose a model-free DRL-based distributed offloading algorithm that enables each mobile device to make its offloading decision without knowing the task models and offloading decisions of other mobile devices.

Performance Evaluation: We perform simulations and show that when compared with the potential game based offloading algorithm (PGOA) and the user-level online offloading framework (ULOOF) our proposed DRL-based algorithm can better exploit the processing capacities of the mobile devices and edge nodes, and it can significantly reduce the ratio of dropped tasks and the average delay.

3.2.2 ADVANTAGES

- It can reduce the ratio of dropped tasks
- It can handle the time-varying system environments
- Reduce the average delay

IV. SYSTEM IMPLEMENTATION

4.2 MODULE DESCRIPTIONS

5.2.1 Decentralized Resource Analysis

The Resource Allocation aimed to find the amount of resources to be leased such that the operational cost could be minimized, Assuming that insufficient resources at any time instance could be dynamically and instantaneously allocated on-demand.The resource reservation plan included the lease period, types of VM (servers) and their quantity to be reserved.

4.2.2 Resource Provisioning

Resource on demand. In this scenario, the predicted demand (rp) exceeds the capacity of all reserved VMs (rr), thus the Resource Broker must operate the on-demand option to subscribe more VMs. Optimal solution of the on-demand VM configuration. A cloud broker is a third-party individual or business that acts as an intermediary between the purchaser of a cloud computing service and the sellers of that service.

4.2.3 Adaptive DSRS Resource Allocation

In the resource provisioning is insufficient to meet the workload demand QoS will be violated, which results in a penalty for the application provider. In the first phase, focused on determining the optimal long-term resource provisioning.

4.2.4 Offload decision Planning Process

In the Planning Process, Reservation User (Short Term Planning), On Demand User (Long Term Planning) Reservation User: In the short-term planning for dynamic resource allocation. After starting the reservation contract, the process enters the second phase.A cloud computing deployment allows an Agency to focus on its mission while someone else manages its computing infrastructure. Being able to provision unlimited computer resources as needed and right-size resources as required enables an Agency to respond quickly and more effectively deliver critical services.

4.2.5 Information Alert System

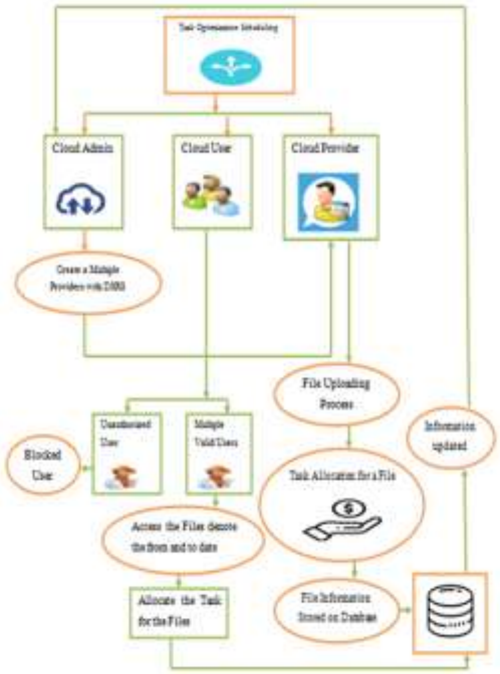
Furthermore, they also offer several pricing models. It raises an interesting issue to application providers on how to effectively provision or subscribe VM resources from an IaaS provider. In this paper, formulated the resource provisioning problem as a two phase resource planning problem.

V. SYSTEM DESIGN

5.1 ARCHITECTURE DIAGRAM

There are an increasing number of emerging mobile applications that will benefit from MEC, by offloading their Computation-intensive tasks to the MEC cloud servers for cloud execution. The capability of enhancing the privacy and security of mobile applications is also an attractive benefit brought by MEC compared to MCC.

International Journal of Advances in Engineering and Management (IJAEM)
Volume 3, Issue 5 May 2021, pp: 919-923 www.ijaem.net ISSN: 2395-5252



VI. SOFTWARE DESCRIPTION

6.1 Core JAVA

Java is a powerful but lean object-oriented, multi-threaded programming language. It is designed to be the small, simple and portable across different operating systems.

The powerful of java is due to its unique technology that is design on the basis of 3 key elements. They are the usage of applets, powerful programming language constructs and a rich set of significant object classes.

When a program is compiled it is translated in to machine code or processor instructions that are specific to the processor. In the java development environment there two parts:

1. Java Compiler - that generates byte code instead of machine code.
2. Java Interpreter - executes java program.

   The disadvantage of using byte code is the execution speed. Since system specific programs run directly on the hardware, they are faster than the java byte codes that are processed by the interpreter.

   Java is actually a platform consist of three components.

1. Java programming language.
2. Java library of classes and interfaces.
3. Java Virtual Machine.

JAVA Includes Library

The java platform includes an extensive class library so that programmers can use already existing classes as it is, create sub class to modify existing class or implement interfaces to augment the capabilities of classes.

Both classes and interfaces contain fields (data members) and methods (functions). In a class, fields may be either variables or constants and methods are implemented. In an interface, fields must be constant and methods are just prototypes with no implementations.

Interfaces let one to add the functionality to a class and give the great deal of flexibility in doing so. In the words, interfaces provide most of the advantage of multiple inheritances without its disadvantages.

VII. SYSTEM TESTING

7.1 TESTING PROCESS

When a system is developed, it is expected that it performs properly. In practice, however, some errors always occur. The main purpose of testing an information system is to find the errors and correct them. A successful test is one, which find an error. The main objectives of the system testing are :

1. To ensure during the operation that the system will perform as per specified in the design phase.
2. To make sure that the system meets user requirements during operations.
3. To verify that the controls incorporated in the system functions as intended.
4. To see that if correct inputs are fed into the system, it provides perfect output.
5. To verify that during operation incorrect input processing and output will be deleted.

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding. If the testing conducted successfully, it will uncover errors in the software. As a secondary benefit, testing demonstrates that the software functions appear to be working according to specification and that performance requirements appear to have been made.The scope of the system test should include both manual operations and computer operations system testing is comprehensive evaluation of the programs, manual procedures, computer operations and controls.System testing is the process of checking if the developed system is working according to the original objectives and requirements. All testing needs to be conducted in accordance to the test conditions specifies earlier.This will ensure that the test coverage meets the requirement and that testing is done in semantic manner. System testing

accounts for the largest percentage of technical effort in the software development phase.
There are four aspects of system testing:

- Unit testing
- Integration testing
- Validation testing
- Verification testing

VIII. CONCLUSION AND FUTURE ENHANCEMENT

8.1 CONCLUSION

In this work, we studied the computational task offloading problem with non-divisible and delay-sensitive tasks in the MEC system and designed a distributed offloading algorithm that enables mobile devices to make their offloading decisions in a decentralized manner. The proposed algorithm can address the unknown load level dynamics at the edge nodes, and it can handle the time-varying system environments (e.g., the arrival of new tasks, the computational requirement of each task). Simulation results showed that when compared with several benchmark methods, our proposed algorithm can reduce the ratio of dropped tasks and average delay. The benefit is especially significant when the tasks are delay-sensitive or the load levels at the edge nodes are high.

8.2 FUTURE ENHANCEMENT

For future work, it is interesting to enable mobile devices to learn their optimal offloading policies cooperatively through taking advantage of the trained neural networks of other mobile devices. Through the cooperative learning, the training process of the DRL-based algorithm may be accelerated, and the performance may be improved.

REFERENCE :

[1]. Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, “A survey on mobile edge computing: The communication perspective,” IEEE Commun. Surveys & Tuts., vol. 19, no. 4, pp. 2322–2358, Aug. 2017.

[2]. P. Porambage, J. Okwuibe, M. Liyanage, M. Ylianttila, and T. Taleb, “Survey on multi-access edge computing for Internet of things realization,” IEEE Commun. Surveys & Tuts., vol. 20, no. 4, pp. 2961–2991, Jun. 2018.

[3]. C. Wang, C. Liang, F. R. Yu, Q. Chen, and L. Tang, “Computation offloading and resource allocation in wireless cellular networks with mobile edge computing,” IEEE Trans. Wireless Commun., vol. 16, no. 8, pp. 4924–4938, May 2017.

[4]. S. Bi and Y. J. Zhang, “Computation rate maximization for wireless powered mobile-edge computing with binary computation offloading,” IEEE Trans.Wirel. Comm., vol. 17, no. 6, pp. 4177–4190, Apr. 2018.

[5]. N. Eshraghi and B. Liang, “Joint offloading decision and resource allocation with uncertain task computing requirement,” in Proc. IEEE INFOCOM, Paris, France, Apr. 2019.

[6]. X. Lyu, H. Tian, W. Ni, Y. Zhang, P. Zhang, and R. P. Liu, “Energyefficient admission of delay-sensitive tasks for mobile edge computing,”IEEE Trans. Commun., vol. 66, no. 6, pp. 2603–2616, Jun.2018.

[7]. M. Chen and Y. Hao, “Task offloading for mobile edge computing in software defined ultra-dense network,” IEEE J. Sel. AreasCommun., vol. 36, no. 3, pp. 587–597, Mar. 2018.

[8]. K. Poularakis, J. Llorca, A. M. Tulino, I. Taylor, and L. Tassiulas, “Joint service placement and request routing in multi-cell mobileedge computing networks,” in Proc. IEEE INFOCOM, Paris,France, Apr. 2019.

[9]. X. Lyu, W. Ni, H. Tian, R. P. Liu, X. Wang, G. B. Giannakis, and A. Paulraj, “Distributed online optimization of fog computing for selfish devices with out-of-date information,” IEEE Trans. Wirel. Comm., vol. 17, no. 11, pp. 7704–7717, Sep. 2018.

[10]. L. Li, T. Q. Quek, J. Ren, H. H. Yang, Z. Chen, and Y. Zhang, “An incentive-aware job offloading control framework for multi-access edge computing,” IEEE Trans. Mobile Comput., 2019 (Early Access).