Journal of Computers Vol. 32 No. 5, 2021, pp. 222-231
doi:10.53106/199115992021103205019

Dynamic Node Link Model of Hierarchical Edge Computing

Sheng-Guo Wang, Yong-Gang Liu*, Tian-Wei Bai

China Shenhua Energy Company Limited Shenshuo Railway Branch, Shenmu, 719316, China
{shengguo.wang.e, yonggang.liu.r, tianwei.bai}@chnenergy.com.cn

Received 1 August 2021; Received 1 September 2021; Accepted 8 October 2021

Abstract. With the rise of the Internet of Things, edge computing has become one of the key technologies in Internet of Things solutions. In the context of the Industrial Internet of Things, hierarchical edge computing shows its advantages. This article focuses on hierarchical edge computing in the industrial Internet of Things scene, and studies the dynamic resource allocation of hierarchical edge computing networks. When using a hierarchical edge computing network with existing equipment, it is difficult to make changes to existing equipment. Therefore, this article uses queuing theory modeling analysis and proposes Dynamic Link Model based on Nodes Relation. Aiming at the hierarchical edge computing network, this model uses a method based on node connection relationship transfer to achieve load balancing of task flow and completes the dynamic allocation of computing resources in the network, and proposes a time-experienced priority queue offloading strategy. The paper uses Java to achieve a dynamic link model experiment based on the connection relationship between nodes. The results show that this scheme has significant advantages in the global average delay of the system, and ensure the loss probability is reasonable within a certain limit.

Keywords: edge computing, Industrial Internet of Things, network deployment

1 Introduction

1.1 Research Background

In recent years, the concept of edge computing has emerged. Edge computing focuses on the word "edge", which means that computing functions are deployed at the edge of the network. It is located at the edge of the network, so it has a better response time and can guarantee the delay requirements of low-latency applications.

In 2012, Flavio Bonomi et al. [1] proposed the concept of fog computing. S. Yi et al. [2] further discussed the definition, goals and challenges of fog computing, and carried out examples and practices based on actual scenarios. M. Chiang et al. [3] and A. V. Dastjerdi et al. [4] analyzed the advantages and problems of fog computing in the Internet of Things. With the development of fog computing research, its concept is getting closer and closer to edge computing. In many studies, the concept of fog computing and edge computing has gradually no longer separated a clear line. In future research, it remains to be seen whether fog computing and edge computing are still two different computing networks.

With the development of edge computing research, people's demand analysis in scenarios such as the Internet of Things, 5G, and Internet of Vehicles is also becoming more and more perfect. The development of edge computing provides solutions for these scenarios and further promotes the progress of related research such as the Internet of Things. The advancement of research related to these application scenarios has further put forward new demands and promoted related industries and industries to invest in edge computing research. Edge computing is no longer an independent technology, but has become one of the foundations of future networks, complementing and developing together with

* Corresponding Author

other technologies.

1.2 Typical Application Scenarios in Edge Computing

This section analyzes the typical application scenarios of edge computing. Edge computing has a variety of application scenarios, such as MEC, Internet of Vehicles, and Industrial Internet of Things, all of which have different requirements for edge computing.

Among them, the node is often inseparable from the base station. At the same time, a large number of MEC deployment problems are also carried out around base stations. In the current research background, scholars have adopted the practice of adding servers to the base station to make it a small edge cloud.

In the scene of the Internet of Vehicles, the edge nodes are often provided by the on-board computer or the equipment set up on the roadside. Due to the mobility of automobiles, the Internet of Vehicles also poses new challenges to the mobility of corresponding edge computing.

In the industrial Internet of Things, edge computing is mainly composed of smart gateways, smart routers or small servers. Due to the characteristics of the Industrial Internet of Things, its mobility is weaker than that of the Internet of Vehicles, but it has a large amount of data from non-smart sensors or smart terminals. This requires that the edge computing network having good carrying capacity and will not be congested.

Due to the characteristics of the Industrial Internet of Things, its various devices are often heterogeneous and located at different levels in the network. Therefore, in this scenario, hierarchical edge computing has a good deployment environment. When deploying a hierarchical edge computing network, if you use new equipment to deploy, the hierarchical characteristics should adapt to the structural characteristics of the industrial Internet of Things; and if you use existing equipment to deploy, then the hierarchical structure should just use industrial things. The characteristics of networking naturally realize the collaboration between multiple layers.

D. A. Chekired et al. [20] focused on the advantages of hierarchical fog computing in industrial IoT data scheduling, and called it as the key of smart factories. They first analyzed the Industrial IoT scenarios in large factories and built a hierarchical computing network. Using probabilistic analysis, they concluded that the hierarchical structure is more efficient than the flat structure. They designed a queuing strategy and an offloading plan, and finally performed a simulation analysis using data from the real environment, which proved that compared with the traditional flat edge computing network, the performance of the hierarchical structure is better.

This article uses the Industrial Internet of Things as an application scenario, combined with the actual analysis of the performance of hierarchical edge computing, and study the problem of capacity allocation between layers and the dynamic connection of nodes.

1.3 The Main Content of This Article

This article is oriented to the industrial Internet of Things scene in edge computing, and studies two issues of static deployment and dynamic resource allocation of hierarchical edge computing networks. The main content of this paper is dynamic resource allocation problem of the hierarchical edge computing network. This paper proposes the Dynamic Link Model based on Nodes Relation, which realizes the load balancing of task flow and completes the dynamic allocation of computing resources in the network. Then it analyzes the impact of multiple offloads in the hierarchical edge computing network on the task delay, and proposes the elapsed time priority queue offloading strategy, which reduces the maximum task delay in the system.

2 Related Work and Theories

Queueing Theory is a branch of operations research that focuses on the working process of random service systems. The main content of queuing theory is to study the relationship between variables such as waiting time, service time, waiting queue length and the system structure, customer arrival rate, and service rate that the system can provide in a random service system. The analysis results of queuing theory with statistical characteristics provide accurate and good support for the analysis of system performance.

The content of queuing theory is complicated. This section only briefly introduces the more content involved in this article.

2.1 Little Theorem

The content of Little's theorem is that in a stable system, the number of customers in the system is equal to the product of customer arrival rate and their average stay time.

$$N = \lambda T \tag{1}$$

It is the number of customers when the system is stable, the arrival rate of customers, and the average stay time of customers in the system. For example, assuming that the task arrival rate in a system is 5 per second and the average stay time is 5 seconds, then the number of tasks contained in the system should be 25 when the system is stable. This result is also intuitive.

2.2 Kendall Notation

Kendall notation is used to describe the characteristics of modeling in queuing theory. The main format is X/Y/Z/A/B/C. There are special representation methods for symbols. A, B, and C can be omitted under normal circumstances. When omitted, it means that the system capacity is unlimited, the number of customer sources is unlimited, and the rule is First Come First Served.

For example, if a queuing model is G/D/1, it means that customer arrivals conform to the general distribution, and the general distribution means that the content of the distribution is determined by a specific expression; the service rate conforms to the definite distribution, that is, the service time of each customer is Determine the length; there is only one waiter in the system. When the waiter serves customers, other customers need to wait in line; the capacity in the system is unlimited, that is, there is no limit to the queue length; the total number of customers is unlimited, that is, customers will not be exhausted; service rules It is first-come-first-served, that is, queuing in the order of arrival and serving in the order in the queue.

The M/Er/n/c queuing model means that the arrival rate of customer obeys the Poisson distribution, and the service rate obeys the Erlang distribution. In the system, there are n waiters serving in parallel, and the queue length is limited to c. After the customer arrives, if they find that the queue is full, they will leave and will not accept the service.

2.3 Pasta Theorem

The PASTA theorem was originally proposed by Ronald W. Wolff, which states that if the customer's arrival process conforms to the Poisson distribution, and if the limit distribution exists. When a customer arrives, the distribution probability of each state in the system he observes is consistent with the result obtained when the outside world observes the system. According to the PASTA theorem, if the arrival of a customer obeys the Poisson distribution, then we can use the distribution result of the external observation system to replace the distribution result observed when the customer arrives, so as to help us get the probability of finding that the queue is full when the customer arrives in the system, namely Call loss rate.

2.4 Related Work

The current research focus of edge computing mainly lies in the aspects of resource scheduling, security and privacy protection, equipment energy saving, deployment strategy and so on. The resource scheduling method in edge computing is one of the research hotspots, and its purpose is to reduce time delay and reduce bandwidth occupation. Many scholars at home and abroad have conducted research in this area.

M. Alrowaily et al. [5] reviewed the problems, challenges and opportunities of secure edge computing in IoT scenarios, and used two cases of smart parking and content delivery network (CDN, Content Delivery Network) to analyze the problems. K. Fan et al. [6] analyzed the cross-domain data sharing scheme in cooperative edge computing, mainly using RSA and attribute encryption based on password policy (CP-ABE). M. Caprolu et al. [7] revealed the limitations of some technologies in fog computing and edge computing, and provided a potential direction for future research on security issues. W. Zhou et al. [8] proposed an efficient and secure coding edge computing scheme using orthogonal vectors to improve data confidentiality. J. Wang et al. [9] designed efficient task allocation algorithms and coding calculation schemes in the context of heterogeneous edge computing devices. A. Liguori et al. [10] focused on the dynamic migration of containerized processes in a virtualized runtime environment, and analyzed the attacker's model. M. Bazm et al. [11] proposed a secure distributed computing solution using trusted Linux containers on untrusted fog infrastructure to alleviate the problem of untrustworthy third-party operating systems or hardware. N. Mäkitalo et al. [12] proposed a new programming model to alleviate security issues on the edge of the network.

The energy-saving requirements in edge computing mainly come from the power limit of mobile devices such as user terminals, and the cost of energy consumption when operators deploy edge computing networks.

D. Loghin et al. [13] proposed a new time-energy-cost analysis and compared the two models of Amazon EC2 cloud, Jetson TK1 and Jetson TX1. M. Yao et al. [14] proposed energy-saving cooperative edge computing under multi-source and multi-relay equipment, and designed the best total energy consumption algorithm and the best energy consumption allocation algorithm. C. -Y. Yu et al. [15] introduced the scenario of implementing energy-saving workload offloading in 5G vehicle edge computing. S. Xie et al. [16] designed the transmission power and offloading strategy, so that the MEC server working at THz frequency provides ultra-reliable end-to-end delay (URLLC) while minimizing the power consumption caused by communication. T. Zhao et al. [17] studied a solution to minimize energy consumption by allocating bandwidth and computing resources to mobile devices in the case of offloading tasks with heterogeneous clouds in a multi-user MEC system.

3 Dynamic Link Model Based on Node Connection Relationship

The existing inter-layer capacity allocation method of hierarchical in edge computing focuses on how much computing performance should be allocated to each layer under a certain total cost. In the end, we get a static model, and the allocation plan is also static allocation program. However, we cannot implement static strategies in all situations.

In this chapter, we propose a new dynamic model based on node connection relationship to adapt to the uneven distribution of task arrivals, so it can make adaptive adjustments according to task distribution, improve hardware resource utilization, and reduce system call loss and the global average delay of the system. At the same time, this chapter also proposes the unloading strategy of the priority queue over time, adjusts the distribution of task delay, reduces the maximum task delay in the system, and reduces the possibility of task overtime.

D. A. Chekired et al. [20] proposed that in hierarchical edge computing, there is only one node at the highest level, the system has the greatest resource utilization. Therefore, the first design goal is to make the highest-level node number of the system 1.

The highest-level nodes often need to bear some extra work brought about by the northbound interface and generate additional workloads. Therefore, it is stipulated that a single node at the highest level has a lower base load. When designing the system, base load should be taken into consideration.

By changing task-oriented decision-making to task-flow decision-making, the decision-making burden is shifted to a higher level, avoiding high-load nodes to do the main decision-making work. Due to the characteristics of the hierarchical structure, higher-level nodes will have more surplus, and the additional consumption of collecting other lower-level nodes will be less.

Fig. 1 shows the model structure.

Fig. 1 shows the relationship between the top two-level nodes in the edge computing network. In this section we have adopted the restriction strategy of only one top-level node, the N node in the figure represents the top-level node. n1, n2, and n3 represent the next-high-level nodes. There are three nodes in the next-high-level as an example. To avoid confusion, we will not use numbers to describe the number of layers, but the highest layer (N nodes) and the next highest layer (n1, n2, n3 nodes) will be used to describe each layer.

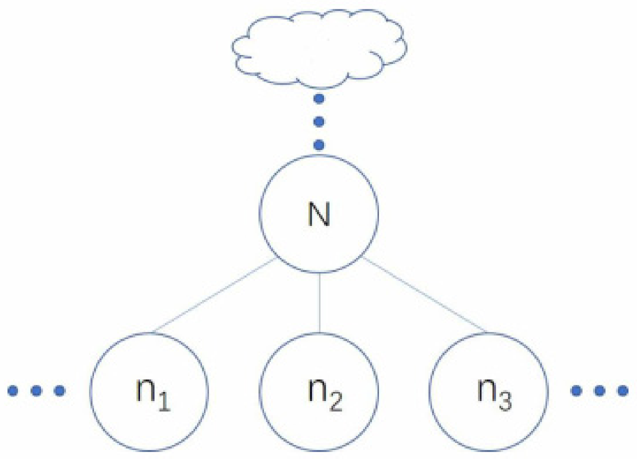

Fig. 1. Dynamic Link Model based on Node Relation

This model only focuses on the top two layers, the two layers of nodes closest to the cloud, and transfers the connection between these two layers. In the two-layer and multi-layer model, we still give priority to the connection relationship between the top two layers, and this model can be extended to the multi-layer structure. The priority to pay attention to high-level connection relationships is because the data aggregation degree of high-level nodes is greater, and the load pressure for task processing is generally lower, and there is more margin for computing resources. And when task is not uninstalled in the lower layer, it is difficult for the lower node to judge the subsequent uninstallation of the task. This knowledge is difficult to judge by itself. But for the high-level nodes, they will naturally receive the information fed back from the low-level nodes, and will be able to judge the conditions of the low-level nodes at a lower cost. For example, for a high-level node, it will accept the task offloading generated by the connected low-level node, then accepting the task offload indicates that the low-level node is already busy. The high-level node only needs to care about the low-level node that does not have task offloading. Additional query load will be generated for high-load low-level nodes. On the other hand, it means that low-level nodes that do not have task offloading are likely to have surplus computing capacity to process query requests from high-level nodes.

While using the highest level and the next level to describe each level, we also use parent nodes and child nodes to describe each node. Because we only focus on the top two layers this time, the parent node represents node N, and the child nodes represent nodes n1, n2, and n3. Table 1 describes the significance of each variable in this model.

Table 1. Variables in DLMNR

variable name	unit	Annotation
B	IPS (Instruction Per Second)	Due to the existence of the northbound interface, the basic load brought to the parent node N
O_k	IPS	The load of the offload task generated by the kth child node
W_k	IPS	The load of the node, the subscript indicates the corresponding node
W_e	IPS	Rated load, the normal load expected by the node，$W_e<C$
C	IPS	The total computing capacity of the node
β	/	Transfer sensitivity factor

Among them, the CPU’s instruction per second (IPS) is used to represent the load.

4 Simulation and Result Analysis

4.1 Simulation and Analysis of Priority Queue Offloading Strategy

The simulation in this section is mainly aimed at the unloading strategy of the priority queue over time. Considering the two cases of using elapsed time priority queues and inapplicable priority queues respectively, we focused on simulating the average task delay and maximum task delay in edge computing networks. The independent variable is the link delay in the system, in seconds. The delay calculation method is:

The high-priority task needs to experience its own processing delay and the queuing delay of the same high-priority task before it. The low-priority task needs to experience processing delay, two-priority queuing delay, and the queuing delay caused by the newly arrived high-priority task jumping in the queue during the waiting process. As a result, the respective retention delays of the two priority tasks can be eliminated.

$$w_1 = \frac{1}{\mu - \lambda_1} \tag{2}$$

$$w_2 = \frac{1 + \lambda_1 w_1}{\mu - \lambda_{2\varepsilon} - \lambda_1} \tag{3}$$

Therefore, we can express the average delay and the maximum delay of the queue unloading strategy using the experienced time priority.

$$E(t) = \frac{\lambda_{2\varepsilon}}{\lambda_1} \cdot w_2 + \frac{\lambda_1}{\lambda}(w_1 + t_L) + \frac{\lambda_{off}}{\lambda}(w_3 + t_L) \tag{4}$$

$$Max(t) = \max(w_2, (w_1 + t_L), (w_3 + t_L)) \tag{5}$$

The three terms in the formula respectively represent the delay when the task is a local task and processed locally, the delay when the task is unloaded from other nodes and processed locally, and the delay when the task is unloaded outward by the node. They are weighted according to their respective probabilities. Obtain the expectation of the global delay.

For the control group, the non-priority queue strategy is similarly expressed as follows.

$$\lambda = \lambda_1 + \lambda_2 \tag{6}$$

$$\lambda_\varepsilon = \frac{\mu}{\gamma} \tag{7}$$

$$w = \frac{1}{\mu - \lambda_\varepsilon} \tag{8}$$

$$\lambda_{off} = \lambda - \lambda_\varepsilon \tag{9}$$

$$E(t') = \frac{\lambda_\varepsilon}{\lambda_1}\left(\frac{\lambda_1}{\lambda}(w + t_L) + \frac{\lambda_2}{\lambda} \cdot w\right) + \frac{\lambda_{off}}{\lambda}\left(\frac{\lambda_1}{\lambda}(2t_L + w_3) + \frac{\lambda_2}{\lambda}(t_L + w_3)\right) \tag{10}$$

$$Max(t') = \max((w + t_L), (2t_L + w_3)) \tag{11}$$

The parameters are set to: $\lambda_1 = \lambda_2 = 120$, $\mu = 240$, $\gamma = 1.2$, $w_3 = 0.03$.

In Fig. 2, the average delay of the non-priority queue coincides with the average delay of the elapsed time priority queue. Therefore, there are 5 lines, and the 5th line is not obvious because of the overlap in the figure. We used the cyan dashed line to indicate the non-priority average delay, and the pink solid line to indicate the average delay of the elapsed time priority queue.

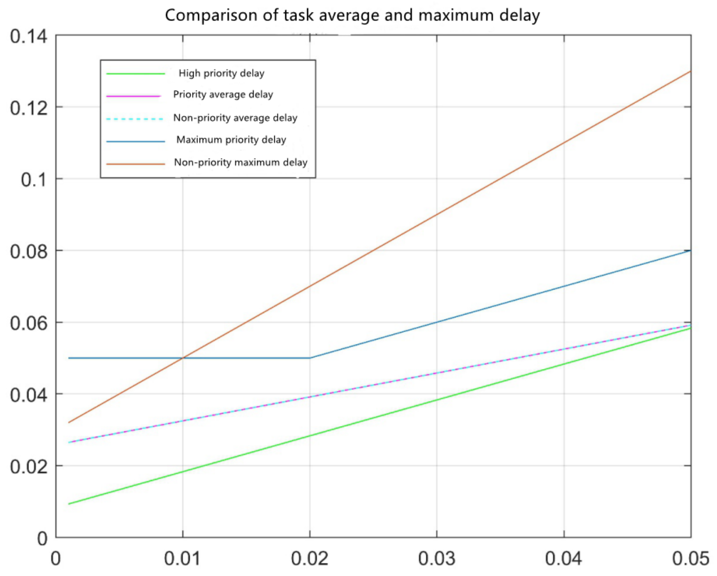

Fig. 2. Elapsed time priority queue unloading strategy simulation results

Obviously, using the elapsed time priority queue offloading strategy will not cause loss to the average delay. When the impact of link delay is small, using the elapsed time priority queue offloading strategy will slightly increase the maximum task delay in the system, which is caused by the low priority queue time. However, as the impact of link delay increases, the elapsed time priority queue unloading strategy can effectively and significantly reduce the maximum delay in the system, which further reduces the possibility of task timeout. The experimental results show that when the link delay has a significant impact, using the elapsed time priority queue offloading strategy can effectively improve the system performance.

4.2 Analysis of Results

The simulation data is exported to Microsoft Excel for drawing. Fig. 3 shows a line graph of the average delay of a task as the load changes.

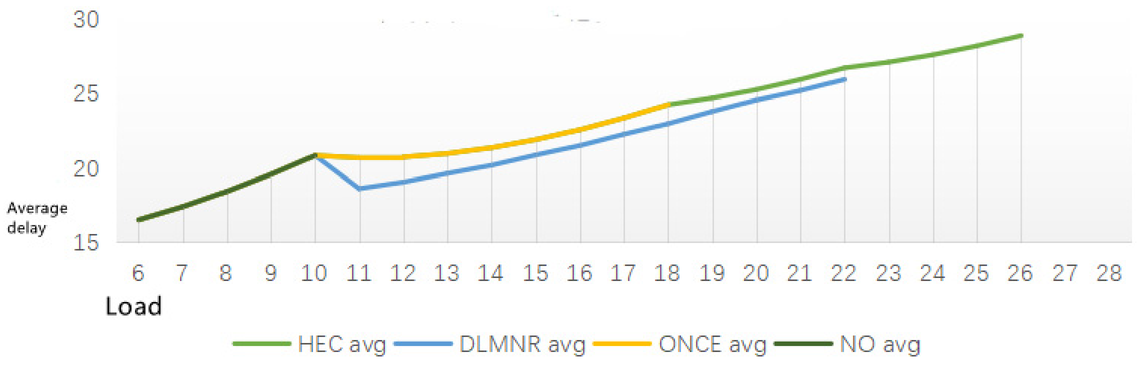

Fig. 3. Global Average Delay Trend with Node Load

The end points of the curves in Fig. 3 are different. Because after the system call loss occurs, the measurement of time delay does not have sufficient practical significance, so the end position of each curve is the location where the call loss occurs in each network. It can be clearly seen that the use of a dynamic link network based on the connection of nodes has a lower average delay.

The dynamic link network curve showed a significant delay drop after the load increased to 10. Because the connection transfer mechanism of the dynamic link network is triggered at this time. After the offload, the number of tasks in the cache queue of each node is more even. This shortens the queue length of high-load nodes. Due to the shortened queuing delay, the average delay has a short-term downward trend. Although it seems that the delay has been greatly reduced, we should also take the system cost of connection transfer into consideration at the same time, and cannot blindly increase the sensitivity of connection transfer in order to pursue the reduction of delay. The call loss rate is shown in Fig. 4.

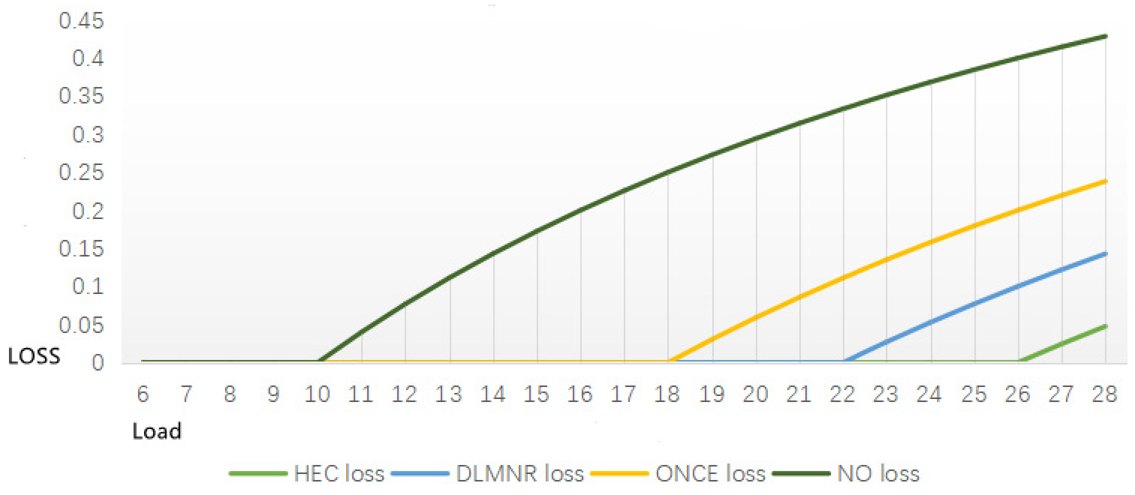

Fig. 4. Loss Trend with Node Load

The utilization of hardware resources in the dynamic link network is not as good as that of the home edge computing hierarchical load balancing scheme, and the appearance of call loss will be slightly earlier. However, when the node capacity is 10, the overload tolerance provided by the dynamic link network is also under normal circumstances. It is sufficient to meet the needs of the scene, and has a smaller average delay, as shown in Fig. 5.

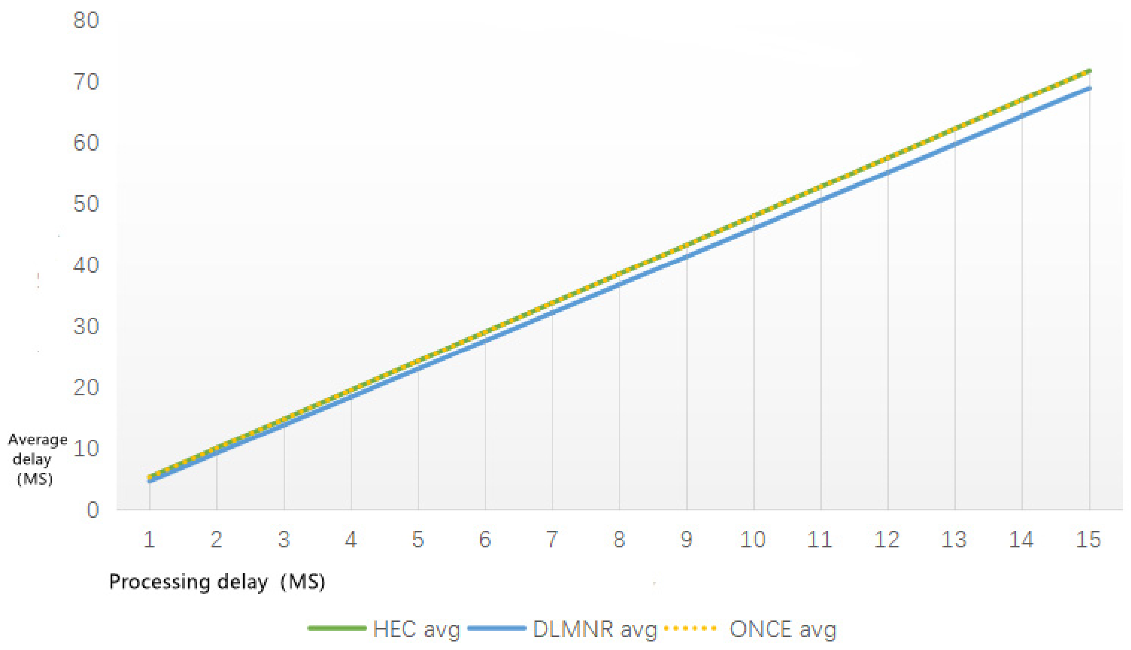

Fig. 5. The average task delay varies with the processing delay

The maximum delay change is shown in Fig. 6 below.

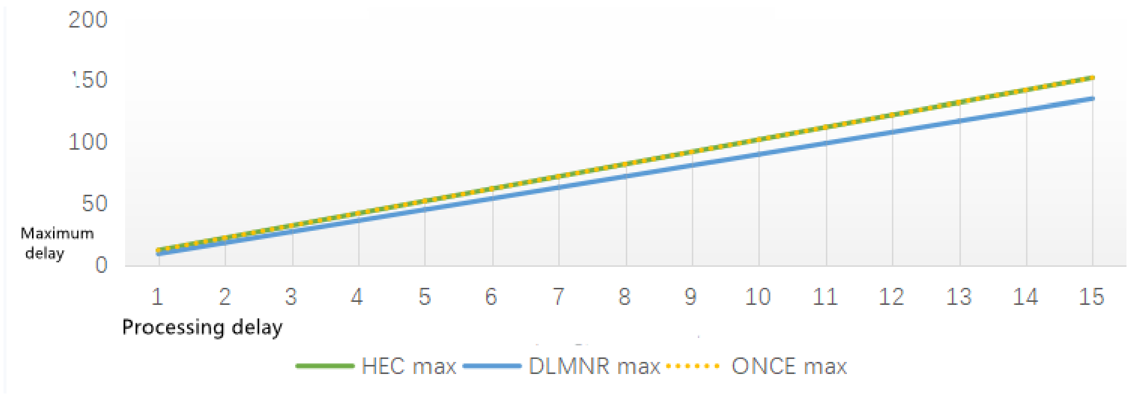

Fig. 6. Global Max Delay Trend with Process Delay

5 Summary and Outlook

In the context of the Industrial Internet of Things, a dynamic link model based on node connection relationship is proposed for the dynamic resource allocation problem of hierarchical edge computing.

Aiming at the problem of dynamic resource allocation of hierarchical edge computing, a dynamic link model based on node connection relationship is proposed to realize dynamic resource allocation during system operation. This model designs a brand-new mechanism and connection relationship management method, analyzes the queue strategy that needs to be selected, and proposes a strategy for task experience time priority. Finally, in the simulation experiment, this model was compared with the unloading model proposed by A. Kiani et al. [19] and C. S. M. Babou et al. [18]. Experimental results show that this model significantly reduces the average delay of the system based on the better carrying capacity of the system.

Finally, clustering is implemented based on task characteristics, and corresponding unloading decisions can be made based on various characteristics. Whether this decision can be compatible with the load balancing method based on the node connection relationship, there is still room for full discussion. A global fine-grained feature extraction method is proposed to improve the efficiency of internal threat detection by increasing the number of examples extracted. In this method, a variational auto-encoder is combined to propose an LVE internal threat detection algorithm. That is, the latent vector of the original data is generated by the LSTM encoder, so that the model retains the characteristics of the original data to the greatest extent, and the LSTM decoder is used to reconstruct the original features to optimize the LVE model. The simulation results show that compared with the traditional isolated forest method, this algorithm has a higher recall rate.

In the future, further research can be conducted in the following directions:

The representativeness and validity of the data set are critical to the model. Although the internal threat detection method proposed in this article has a certain effect, this article only uses the commonly used CMU-CERT internal threat data set detection, and hopes to use the real data collected from the enterprise to detect the effectiveness of the LVE algorithm.

Acknowledgements

This research was supported by the Technology innovation project of Shenshuo Railway Branch, China.

References

[1] F. Bonomi, R. Milito, J. Zhu, S. Addepalli, Fog computing and its role in the internet of things, in: Proceedings of the first edition of the MCC workshop on Mobile cloud computing, 2012.

[2] S. Yi, Z. Hao, Z. Qin, Q. Li, Fog Computing: Platform and Applications, in: Proc. 2015 Third IEEE Workshop on Hot Topics in Web Systems and Technologies, 2015.

[3] M. Chiang, T. Zhang, Fog and IoT: An Overview of Research Opportunities, IEEE Internet of Things Journal 3(6)(2016) 854-864.

[4] A.V. Dastjerdi, R. Buyya, Fog Computing: Helping the Internet of Things Realize Its Potential, Computer 49(8)(2016) 112-116.

[5] M. Alrowaily, Z. Lu, Secure Edge Computing in IoT Systems: Review and Case Studies, in: Proc. 2018 IEEE/ACM Symposium on Edge Computing, 2018.

[6] K. Fan, Q. Pan, J. Wang, T. Liu, H. Li, Y. Yang, Cross-Domain Based Data Sharing Scheme in Cooperative Edge Computing, in: Proc. 2018 IEEE International Conference on Edge Computing, 2018.

[7] M. Caprolu, R. Di Pietro, F. Lombardi, S. Raponi, Edge Computing Perspectives: Architectures, Technologies, and Open Security Issues, in: Proc. 2019 IEEE International Conference on Edge Computing, 2019.

[8] W. Zhou, J. Wang, L. Li, J. Wang, K. Lu, X. Zhou, An Efficient Secure Coded Edge Computing Scheme Using Orthogonal Vector, in: Proc. 2019 IEEE Intl Conference on Parallel & Distributed Processing with Applications, 2019.

[9] J. Wang, C. Cao, J. Wang, K. Lu, A. Jukan, W. Zhao, Optimal Task Allocation and Coding Design for Secure Edge Computing with Heterogeneous Edge Devices, IEEE Transactions on Cloud Computing (2020) 1-17.

[10] A. Liguori, P. Schoo, M. Winandy, Mind the Shift: Secure Migration of Containerized Processes in Edge Computing, in: Proc. 2019 6th IEEE International Conference on Cyber Security and Cloud Computing, 2019.

[11] M. Bazm, M. Lacoste, M. Südholt, J. Menaud, Secure Distributed Computing on Untrusted Fog Infrastructures Using Trusted Linux Containers, in: Proc. 2018 IEEE International Conference on Cloud Computing Technology and Science 2018.

[12] N. Mäkitalo, A. Ometov, J. Kannisto, S. Andreev, Y. Koucheryavy, T. Mikkonen, Safe, Secure Executions at the Network Edge: Coordinating Cloud, Edge, and Fog Computing, IEEE Software 35(1)(2018) 30-37.

[13] D. Loghin, L. Ramapantulu, Y.M. Teo, On Understanding Time, Energy and Cost Performance of Wimpy Heterogeneous Systems for Edge Computing, in: Proc. 2017 IEEE International Conference on Edge Computing, 2017.

[14] M. Yao, L. Chen, T. Liu, J. Wu, Energy Efficient Cooperative Edge Computing with Multi-Source Multi-Relay Devices, in: Proc. 2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems, 2019.

[15] C.-Y. Yu, C.K. Chang, W. Zhang, A Situation Enabled Framework for Energy-Efficient Workload Offloading in 5G Vehicular Edge Computing, in: Proc. 2020 IEEE World Congress on Services, 2020.

[16] S. Xie, H. Li, L. Li, Z. Chen, S. Li, Reliable and energy-aware job offloading at terahertz frequencies for mobile edge computing, China Communications 17(12)(2020) 17-36.

[17] T. Zhao, S. Zhou, L. Song, Z. Jiang, X. Guo, Z. Niu, Energy-optimal and delay-bounded computation offloading in mobile edge computing with heterogeneous clouds, China Communications 17(5)(2020) 191-210.

[18] C.S.M. Babou, D. Fall, S. Kashihara, Y. Taenaka, Y. Kadobayashi, Hierarchical Load Balancing and Clustering Technique for Home Edge Computing, IEEE Access 8(2020) 127593-127607.

[19] A. Kiani, N. Ansari, A. Khreishah, Hierarchical Capacity Provisioning for Fog Computing, IEEE/ACM Transactions on Networking 27(3)(2019) 962-971.

[20] D.A. Chekired, L. Khoukhi, H.T. Mouftah, Industrial IoT Data Scheduling Based on Hierarchical Fog Computing: A Key for Enabling Smart Factory, IEEE Transactions on Industrial Informatics 14(10)(2018) 4590-4602.