Scalable Computing: Practice and Experience, ISSN 1895-1767, http://www.scpe.org
© 2024 SCPE. Volume 25, Issues 3, pp. 1833–1840, DOI 10.12694/scpe.v25i3.2743

RESEARCH ON HIGH-PERFORMANCE COMPUTING NETWORK SEARCH SYSTEM BASED ON COMPUTER BIG DATA

XIAOGANG CHEN [1]*AND DONGMEI LIU [2]†

Abstract. An efficient computing system for massively parallel systems is established. The stochastic Petri net abstracts and models the high-performance computer work scheduling system. The IB switch is modeled. Stochastic Petri nets are used for performance analysis. Finally, the proposed method is combined with InfiniBand interconnection architecture to evaluate the system's delay. The experimental results prove the feasibility of this algorithm.

Key words: Multi-cluster; Job scheduling; High-performance computing; Task sequencing; Performance evaluation

1. Introduction. The degree of digitalization in various industries is increasing, and the depth and breadth of its applications are gradually increasing. In scheme design, system simulation verification and optimization, digital design and analysis tools are widely used in engineering development. However, majors such as structure, strength and fluid are more likely to use digital simulation methods to solve various technical problems. The increasing size of analytical models, the increasing accuracy of calculations, and the increasing number of multidisciplinary iterations led to an explosive increase in the demand for computing power. High-performance computing systems delivering supercomputing power are already an essential digital foundation. It is already a significant indicator of Chinese overall competitiveness. It can effectively support and drive the research and development of China's primary science and technology projects and thus promote the development of science and technology. Because there is no interconnection among different HPC clusters, many computing tasks are challenging to execute in the clusters. As a result, the system's management complexity and resource efficiency are not well utilized. Using multiple HPC clusters to build a platform with logical consistency and fully use computing resources is a problem that needs to be solved.

Literature [1] reviews the research progress of high-performance computing at home and abroad. The research results of the high-performance computing ecosystem built by the Chinese Academy of Sciences are introduced. This lays a foundation for the research of high-performance computing in China. Literature [2] illustrates the challenges and problems faced in building HPC portals and the technical paths taken. Especially for aviation and other industries, the construction of high-performance computing has essential reference value. [3] Building efficient computing architecture. Literature [4] presents new challenges and development directions for high-performance computing in cloud environments. They research performance evaluation of high-performance computers. At present, the commonly used evaluation techniques include measurement method, reference method, simulation method, model evaluation method and so on. This paper presents a new performance evaluation method. This method has significant application value in performance prediction, capacity planning and hardware and software procurement. This project will start by constructing a random Petri net (GSPN) and conducting fine processing. This results in a higher-level random network. Then, the relevant performance evaluation is carried out.

2. System design ideas. This paper makes a detailed analysis of the distribution of multiple high-performance computer clusters in each laboratory. For example, each cluster uses existing scheduling software and storage systems and adopts a hierarchical scheduling mode [5]. Single-layer scheduling in the same room reconstructs a complete high-performance computing platform. Each cluster location is maintained at the same

*1. College of Computer Engineering, Henan Institute of Economics and Trade, Zhengzhou, Henan 450018, China; Corresponding author's e-mail: cxiaogang@126.com

†2. College of Management, Henan Institute of Economics and Trade, Zhengzhou, Henan 450018, China

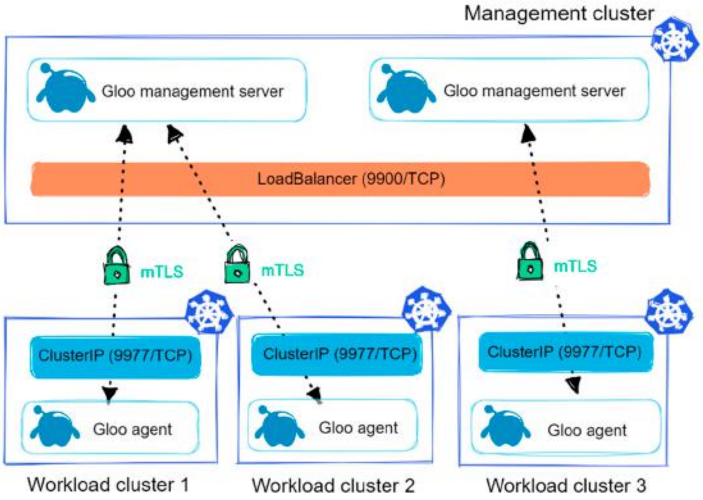

Fig. 3.1: A shared architecture diagram for multiple clusters.

level, considering existing conditions, compute and storage capabilities, and scalability. The following ideas are adopted for construction:

(1) Unified entrance: Users can use efficient computing resources through the unified entrance.
(2) Unified user: log in to the website with a unified ID and perform efficient calculations.
(3) Integrated scheduling: Effective integration and allocation of high-performance computing tasks through a unified scheduling system. In this way, high-performance computing resources can be fully utilized.
(4) Integrated storage: comprehensive integration of scattered storage on each entity. This ensures maximum utilization of storage resources.
(5) Integrated monitoring: Through the comprehensive monitoring and statistics of the operation and utilization of high-performance computing tasks, resource utilization, license, etc., the reasonable allocation of computing resources is realized. This improves the efficiency of operation management.

3. Technical architecture.

3.1. Basic Principles.

(1) High scalability: it can access multiple high-performance computers simultaneously.
(2) High security: The information security in the system can be reliably transmitted and saved after the cluster is networked.
(3) High ease of use: it can quickly and effectively use high-efficiency computing resources.

3.2. System Architecture. Construct an efficient computing system based on a distributed system and integrate and share it. The management center mainly manages user access, user management, unified work arrangement and platform monitoring [6]. Figure 3.1 shows the HPC platform architecture (image referenced in High availability and disaster recovery).

The high-performance computing platform consists of several functional modules:

(1) Access portal: Users and administrators can access, use and manage high-performance computers through this portal.
(2) User management: Authentication of user credentials by integrating with the existing certificate issuance and verification system. Through the integration with the central database to achieve the collection of enterprise-related information [7]. The active table of the College Network Administration Center is used to authenticate operating system users.
(3) Task allocation: Each computing center builds efficient task clusters to complete task allocation. In a distributed environment, the computing tasks of each node are transmitted in real-time [8]. The

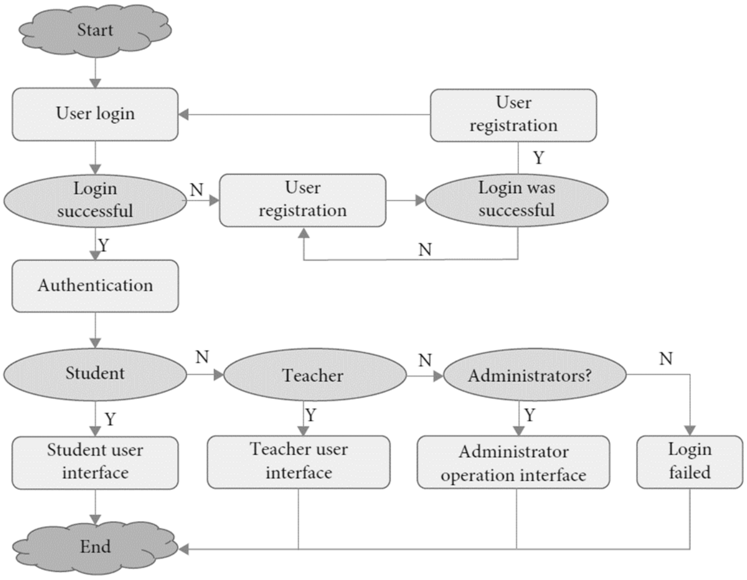

Fig. 3.2: User management diagram.

computing resources of other nodes are used to compute, and the corresponding input and output results are forwarded in real-time.

(4) File system: Use a unified file system to save all kinds of intermediate data and calculation results generated by each operation center during the operation process.

(5) Resource monitoring: various machine-generated information is collected by each computing center. The dispatching center comprehensively processes the operation center, and the overall statistics, analysis and charges are carried out.

3.3. Portal Design. The access portal is configured in the management center of the system to realize the efficient utilization of each system. Access portals are configured in clusters [9]. This can balance the network load and ensure the system's high availability. The implementation of the system includes task management, data management, graphic interaction, compilation and debugging, third-party system integration, web page customization and so on. Through the portal, administrators can manage clusters, tasks, users, permissions, projects, etc. This portal allows Users to submit, monitor, and manage work and data. This system is based on B/S architecture. Access the entry using a browser.

3.4. Multi-User Cluster Management and Scheduling.

3.4.1. User and License Management. The recognition and control of users are realized through system control. The authentication of user credentials is realized by integrating with the existing authentication system [10]. Through the integration with the central database to achieve the collection of enterprise-related information. The paper uses LDAP technology to authenticate computing resources. Figure 3.2 shows a schematic of user management (image cited in Wireless Communications and Mobile Computing, 2022, 2022.).

Through the hierarchical authorization method, the unified management of all kinds of users is realized, while the computer system administrator can only manage the corresponding permissions of users.

3.4.2. Job Scheduling and Software Management. The administrative center is responsible for coordinating and arranging the work. The manager can set an upper limit for CPU time, memory size, runtime, etc., required to perform the task. It can adjust the priority of tasks and perform operations such as pause and resume. You can configure the task schedule according to the following scheduling strategy:

(1) First come, then calculate: the calculation task distribution method is ”first come, then calculate.” By its

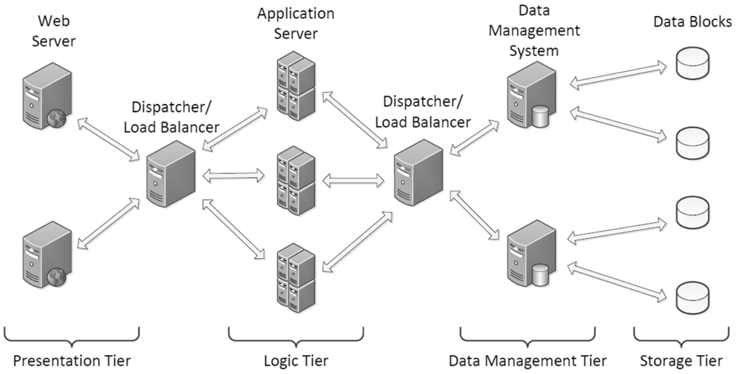

Fig. 3.3: Installation diagram of multi-node simulation computing software.

order in the queue to determine. Users or managers can change the order of issuance by modifying the priority of computation work.

(2) Fairness: Distribute different data to different user groups for different needs. This enables fair access to different types of data streams. The fair sharing mechanism can ensure the fair and reasonable use of the system by distributing the resources allocated to each person or a particular group. If the workload is insufficient, other human computing tasks can use the extra resources for other people's computing [11]. This makes full use of the system. If a user submits more computing tasks, its computing tasks will be completed with higher priority. A method based on fair sharing is proposed. In this way, a reasonable allocation is made to specific users.

(3) Limited time constraint: Resource restriction Scheduling policies can restrict the use of resources. When the number of resources occupied by a computing task exceeds the specified number, it is labeled, or its priority is reduced. The queue parameter is set to limit the resources available for the computation job. Resource constraints determine the number of resources that an arithmetic task can use.

(4) Preemptive scheduling: This method allows high-priority tasks to occupy a smaller space and be executed immediately under tight conditions. When two arithmetic tasks compete for the same arithmetic resources, the arithmetic task in execution is suspended. Currently, the scheduling of work parts is mainly based on the combination of first access calculation and limited resource constraints. By default, the first commit computation task has a higher priority and terminates the configuration of the user's resources if the user reaches a limit. In this way, the dynamic adjustment of the emergency operation task is realized. The simulation analysis software is uniformly installed and configured (Figure 3.3).

Integrate access points and task plans. Use a variety of simulation analysis software to complete user tasks.

3.4.3. multi-computing center planning and monitoring. The schedule for the cross-cluster is shown in Figure 3.4.

(1) Computing network connection: the existing computing network is divided into a management network, computing network and monitoring network. The network management system implements cluster management. The computing network realizes the interconnection of each computing server. The monitoring network can monitor and control the hardware. The network management, computing, and monitoring network are interconnected through networking. The task scheduler is executed on the management server. The task is executed on the computing server. Computing tasks are then assigned to specific computing centers.

(2) File transfer: It is responsible for transmitting working data. And enter the file into the data buffer directory of the working server. Set parameters and compress them before sending. In the data transfer section, the paper added the function of resumable breakpoint. It can prevent the retransmission of big data due to network failure. To ensure that the communication between the client and the client is not interfered with by the outside world, the access of the data between the client and the server must be authenticated. API is used to verify the data in the system and ensure the correctness of the system information. When data is

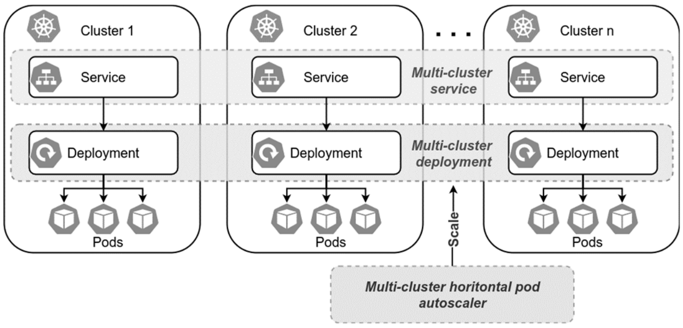

Fig. 3.4: Schematic diagram of cross-cluster scheduling.

transferred, it is transferred in blocks and finally integrated into the overall file. At the same time, it provides the function of a resumable breakpoint [12]. It can effectively prevent data resending caused by network failure. Authorization differential management is carried out on operation data to ensure the security of operation data. Users can only access and manage their operation data but cannot see the operation information of other users.

The cluster monitoring module retrieves each cluster periodically to obtain the related data of each cluster. The specific implementation method of cluster monitoring is:

(1) Real-time fine-cluster load monitoring and analysis: monitor and analyze the CPU core ratio, task ratio, CPU core usage, memory usage, etc., at each stage.
(2) File system load and health monitoring: monitoring file system space utilization, IOPS, etc.
(3) Load analysis of multi-dimensional spatial clusters: The data is studied from multiple levels, such as units.
(4) Real-time monitoring of the license: the usage of each component in the license is monitored.
(5) Statistics and early warning of system records: analysis and early warning of abnormal data.
(6) Email alert: In any abnormal situation, the email is sent to the user's mobile phone to play a role in prompting. When no task is executed, it will be merged into the queue. This allows multi-dimensional monitoring of the queue for computing tasks.

4. Abstract model of the job scheduling system. This paper abstracts it based on the analysis of LSF work plan theory. The corresponding random Petri net model is established [13]. The model contains only one queue g in the LSF where the default is reached. A separate CPU does each task. The workpiece is assigned to a specific arithmetic node (a_i) for the first time upon arrival. A specific $CPU(s_{ij})$ is then assigned to that arithmetic node to complete the job. Repositories and changes in this pattern include the following (Fig.4. 1):

g indicates a work queue for temporary storage tasks that have not been specified. β_i represents the task that has been temporarily assigned to compute node i. g_i represents the task waiting queue for computing node i. A task temporarily stored in arithmetic node i that is not assigned to a processor. v_{ij} represents the work wait queue, the processor$j$ used to compute node i. It is used to store the artifacts that the processor processes. z represents the process in which the task is hosted by the client. a_i indicates that the workpiece is assigned to the arithmetic node i according to a particular scheduling strategy. w_i indicates that the operation node i adds the planned work program to its wait queue. s_{ij}represents a processor j that assigns a job to an arithmetic node i according to a scheduling strategy. r_{ij} represents the working process in the processor.

4.1. Mode refinement and analysis. The original mathematical modeling method is divided into several independent sub-models. Each submodule represents $Y/Y/1$ queue system, in which transition s_{ij}, library v_{ij} and transition r_{ij} respectively represent the workpiece arrival process (arrival rate μ_{ij}), the workpiece waiting queue (queue length d_{ij}) and the workpiece processing process (processing rate η_{ij}) of the queue system of the node i processor j.

4.2. Scheduling method of operation nodes. This project aims at a global minimum average latency. The task is assigned to the nodes with the global minimum mean delay by the comprehensive minimum mean

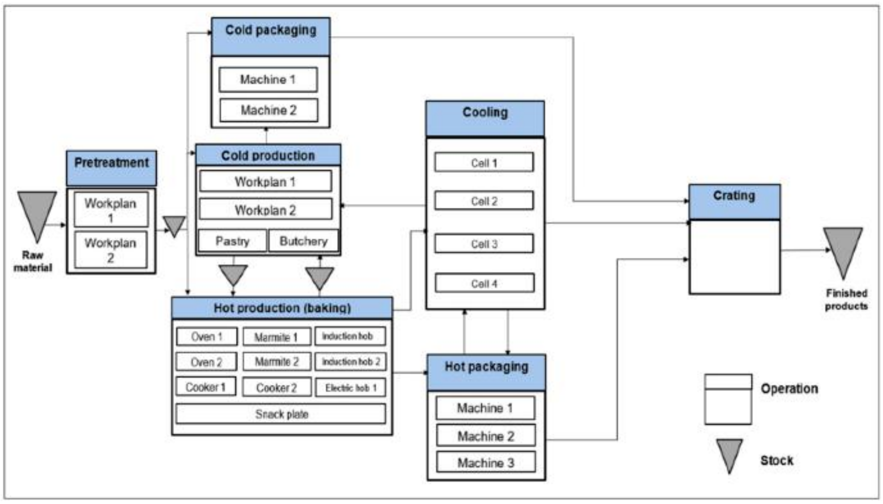

Fig. 4.1: Abstract model of job scheduling.

delay optimization algorithm [14]. The executable predicate y_{a_i} of change a_i is used to restrict change $a_i$and determine whether it can be executed. If the processor queue for the compute node i is not satisfied for the current task to be dispatched, then either the processor i requires the slightest delay or the other compute node's processors are full. Currently, tasks are assigned to the compute node$i$ according to the scheduling algorithm. The conditions that can be implemented can be expressed in the following expressions:

$$\begin{aligned} y_{a_i} : (\sum_{y=1}^{n} Y(v_{iy}) < \sum_{y=1}^{n} d_{iy})\Lambda((\sum_{y=1}^{n} \frac{Y(v_{iy}}{\eta_{iy}} = \min(\sum_{y=1}^{n} \frac{Y(v_{1y})}{\eta_{1y}}, \cdots, \sum_{y=1}^{n} \frac{Y(v_{my})}{\eta_{my}})) \\ \vee (\forall \zeta \neq i, \sum_{y=1}^{n} Y(v_{\zeta y}) = \sum_{y=1}^{n} d_{\zeta y})) \end{aligned} \tag{4.1}$$

Type a_i random switching u_{a_i} represents the possibility of changing the realization of a_i, that is, the possibility of changing the realization of a_i if it can be realized at multiple operation nodes [15]. The following expression can express this possibility

$$u_{a_i}(Y) = \begin{cases} \frac{1}{|OSEDR(Y)|}, & if\, i \in OSEDR(Y) \\ 0, & otherwise \end{cases} \tag{4.2}$$

Among them,

$$\begin{aligned} OSEDR(Y) = \{\zeta | \sum_{y=1}^{n} \frac{Y(v_{\zeta y})}{\eta_{\zeta y}} = \\ \min(\sum_{y=1}^{n} \frac{Y(v_{1y})}{\eta_{1y}}, \cdots, \sum_{y=1}^{n} \frac{Y(v_{my})}{\eta_{my}} \wedge \sum_{y=1}^{n} Y(v_{\zeta y}) < \sum_{y=1}^{n} d_{\zeta y}\} \end{aligned} \tag{4.3}$$

4.3. Processor scheduling algorithm. This project is based on the principle of minimum delay. The task is assigned to the processor with the most minor delay by sorting the given processor. The executable predicate $y_{s_{ij}}$ of change $s_{ij}$is used to limit the change and determine whether it can be executed [16]. The task assigned to processor j can be assigned to processor j if it has the shortest expected delay among the tasks currently to be assigned. What can be achieved can be expressed in the following expressions:

$$y_{s_{ij}} : (Y(v_{ij}) < d_{ij}) \wedge ((\forall\zeta \neq j, \frac{Y(v_{ij})}{\eta_{ij}} \leq \frac{Y(v_{i\zeta})}{\eta_{i\zeta}} \\ \vee (\forall\zeta \neq j, Y(v_{i\zeta}) = d_{i\zeta})) \tag{4.4}$$

Random switching $u_{s_{ij}}$ in change $s_{ij}$is the possibility of changing the implementation of s_{ij}, that is, the possibility of changing the implementation of s_{ij} when multiple processors can implement it.

$$u_{s_{ij}}(Y) = \begin{cases} \frac{1}{|SEDR(Y)|}, & if\, j \in SEDR(Y) \\ 0, & otherwise \end{cases}$$

Among them, $SEDR(Y) = \{\zeta | \frac{Y(v_{i\zeta})}{\eta_{i\zeta}} = \min(\frac{Y(v_{i1})}{\eta_{i1}}), \cdots, \frac{Y(v_{in})}{\eta_{id}}) \wedge Y(v_{i\zeta}) < d_{i\zeta}\}$.

4.4. Perform a scheduling system. Let $U(Y)$ represent the steady-state probability of identifying Y. Suppose that the warehouse v is a queue with capacity$d$, then the average number of tokens in this queue represents the average number of artifacts $S(v)$ in the queue [17]. It can be expressed in terms of $S(v) = \sum_{y=1}^{d} y * U(Y(v) = y)$. The usage degree $A(t)$ of the change t is equal to the sum of the stable probabilities of all the identifiers that make the change executable. If the change is the processor running, then the change in usage is the processor usage. It can be expressed by $A(t) = \sum_{Y \in E} U(Y)$. Here is the entire accessible identification set that can execute t. The productivity of change $T(t)$ is the product of the efficiency with which the change is used and the efficiency with which it is executed, and can be expressed by $T(t) = A(t) * \mu$. Here μ is the execution rate t. According to the queuing theory, the waiting time ST_{ij} is equal to the number of waiting workpieces/represents the number of transferred processes that exit the queue. It can be expressed as $ST_{ij} = S(v_{ij})/T(r_{ij})$, where $S(v_{ij})$ is the average token number of the warehouse v_{ij}. Where $T(r_{ij})$ is the output of r_{ij} in the transfer process. The lag time $ST_{scheduling}$ of the whole plan can be expressed by $ST_{scheduling} = (\sum_{i=1}^{m} \sum_{j=1}^{n} S(v_{ij}))/ \sum_{i=1}^{m} \sum_{j=1}^{n} T(r_{ij})$. The paper takes the probability of queuing up to the maximum time as the task loss rate LR. During the queuing process, the workpiece is discarded, and the loss rate LR of the workpiece can be expressed by $LR = U(Y(v) = d)$. Using the above mathematical expression, we can use SPNP to calculate the system's performance.

5. Conclusion. This paper studies a design scheme of multi-cluster high-performance computing on cluster architecture. Build a high-performance computing platform with unified access and resource sharing to provide users with efficient computing and software support, thereby improving resource utilization efficiency.

REFERENCES

[1] Chen, X., Zhang, J., Lin, B., Chen, Z., Wolter, K., & Min, G. (2021). Energy-efficient offloading for DNN-based intelligent IoT systems in cloud-edge environments. IEEE Transactions on Parallel and Distributed Systems, 33(3), 683-697.

[2] Lv, Z., Lou, R., Li, J., Singh, A. K., & Song, H. (2021). Big data analytics for 6G-enabled massive internet of things. IEEE Internet of Things Journal, 8(7), 5350-5359.

[3] Abualigah, L., Diabat, A., Sumari, P., & Gandomi, A. H. (2021). Applications, deployments, and integration of internet of drones (iod): a review. IEEE Sensors Journal, 21(22), 25532-25546.

[4] Luo, Q., Hu, S., Li, C., Li, G., & Shi, W. (2021). Resource scheduling in edge computing: A survey. IEEE Communications Surveys & Tutorials, 23(4), 2131-2165.

[5] Ghosh, A., Edwards, D. J., & Hosseini, M. R. (2021). Patterns and trends in Internet of Things (IoT) research: future applications in the construction industry. Engineering, Construction and Architectural Management, 28(2), 457-481.

[6] Ding, Y., Jin, M., Li, S., & Feng, D. (2021). Smart logistics based on the internet of things technology: an overview. International Journal of Logistics Research and Applications, 24(4), 323-345.

[7] Yazdeen, A. A., Zeebaree, S. R., Sadeeq, M. M., Kak, S. F., Ahmed, O. M., & Zebari, R. R. (2021). FPGA implementations for data encryption and decryption via concurrent and parallel computation: A review. Qubahan Academic Journal, 1(2), 8-16.

[8] Cao, K., Hu, S., Shi, Y., Colombo, A. W., Karnouskos, S., & Li, X. (2021). A survey on edge and edge-cloud computing assisted cyber-physical systems. IEEE Transactions on Industrial Informatics, 17(11), 7806-7819.

[9] Ouyang, F., Zheng, L., & Jiao, P. (2022). Artificial intelligence in online higher education: A systematic review of empirical research from 2011 to 2020. Education and Information Technologies, 27(6), 7893-7925.

[10] Humayun, M., Jhanjhi, N. Z., Alsayat, A., & Ponnusamy, V. (2021). Internet of things and ransomware: Evolution, mitigation and prevention. Egyptian Informatics Journal, 22(1), 105-117.

[11] Sun, Y., Liu, J., Yu, K., Alazab, M., & Lin, K. (2021). PMRSS: privacy-preserving medical record searching scheme for intelligent diagnosis in IoT healthcare. IEEE Transactions on Industrial Informatics, 18(3), 1981-1990.

[12] Chen, J. I. Z., & Lai, K. L. (2021). Deep convolution neural network model for credit-card fraud detection and alert. Journal of Artificial Intelligence and Capsule Networks, 3(2), 101-112.

[13] Zhou, X., Liang, W., She, J., Yan, Z., Kevin, I., & Wang, K. (2021). Two-layer federated learning with heterogeneous model aggregation for 6g supported internet of vehicles. IEEE Transactions on Vehicular Technology, 70(6), 5308-5317.

[14] Yang, L., Moubayed, A., & Shami, A. (2021). MTH-IDS: A multitiered hybrid intrusion detection system for internet of vehicles. IEEE Internet of Things Journal, 9(1), 616-632.

[15] Ageed, Z. S., Zeebaree, S. R., Sadeeq, M. M., Kak, S. F., Yahia, H. S., Mahmood, M. R., & Ibrahim, I. M. (2021). Comprehensive survey of big data mining approaches in cloud systems. Qubahan Academic Journal, 1(2), 29-38.

[16] Chen, W., Qiu, X., Cai, T., Dai, H. N., Zheng, Z., & Zhang, Y. (2021). Deep reinforcement learning for Internet of Things: A comprehensive survey. IEEE Communications Surveys & Tutorials, 23(3), 1659-1692.

[17] Sarker, I. H., Khan, A. I., Abushark, Y. B., & Alsolami, F. (2023). Internet of things (iot) security intelligence: a comprehensive overview, machine learning solutions and research directions. Mobile Networks and Applications, 28(1), 296-312.

Edited by: Zhigao Zheng
Special issue on: Graph Powered Big Aerospace Data Processing
Received: Oct 28, 2023
Accepted: Nov 15, 2023