2022 IEEE/ACM 15th International Conference on Utility and Cloud Computing (UCC) | 978-1-6654-6087-3/22/$31.00 ©2022 IEEE | DOI: 10.1109/UCC56403.2022.00039

Computation Offloading From Edge to Equipment for Smart Manufacturing

Hung H. Nguyen, Yi Zhou, Kushagra Kushagra, and Xiao Qin

***Abstract*—In smart manufacturing, data management systems are built with a multi-layer architecture, in which the most significant layers are the edge and the cloud. The edge layer renders support to data analysis that genuinely demands low latency. Cloud platforms store vast amounts of data while performing extensive computations such as machine learning and big data analysis. This type of data management system has a limitation rooted in the fact that all data needs to be transferred from the equipment layer to the edge layer in order to perform all data analyses. Even worse, data transferring adds delays to computation processes in smart manufacturing. We investigate an offloading strategy to shift a selection of computation tasks towards the equipment layer. Our computation offloading mechanism opts for smart manufacturing tasks that are not only light weight but also have no need to save data at the edge/cloud end. In our empirical study, we demonstrate that the edge layer can judiciously offload computing tasks to the equipment layer, which curtails computing latency and slashes the amount of transferred data during smart manufacturing process. Our experimental results confirm that our offloading strategy offers the capability for data analysis computing in real-time at the equipment level - an array of smart devices is slated to speed up the data analysis process in semiconductor manufacturing.**

***Index Terms*—edge computing, cloud computing, smart manufacturing, semiconductor, computation offloading, computing performance, algorithmic complexity, O(n), O(nlogn), O(n^2)**

I. Introduction

A. Background

With the fast development of Information Technology, cloud computing, artificial intelligence, and machine learning, modern production and manufacturing of the semiconductor industry has evolved by the virtue of the new technologies to build innovative smart manufacturing for higher productivity, efficiency in production control, and early detection of production issues [1]. In the realm of semiconductor manufacturing, monitoring and analyzing process data is critical and indispensable to bolster productivity. Data analysis techniques such as machine learning, deep learning, statistical process control (SPC), fault detection and classification (FDC), and advanced process control (APC) are deployed for the purpose of defect prediction, maintenance recommendations, resource allocation, to name just a few [2]–[4]. Amid a smart manufacturing process, production data from semiconductor equipment are collected and transferred to an edge layer and a cloud layer for further analysis and storage. Real-time computing in smart manufacturing is performed at the edge layer [5], [6].

Most data management systems for manufacturing are built with two leading platforms: the cloud and edge computing systems [7]–[9]. Cloud computing platforms, in a growing number of cases, are applied to store a large amount of data and to support complicated computing tasks that require high levels and abundant resources. These computing tasks include, but are not limited to, artificial intelligence algorithms, machine learning, and deep learning. The edge platform, which is closer to the devices and equipment, is used for computing tasks critical for the early detection of fault processes. An edge platform, designed and deployed to harvest data from devices and equipment, is slated to perform and orchestrate data analyses. A holistic combination of cloud and edge computing in data management systems helps to curb the latency in real-time data analysis computation. The cloud computing system with rich resources can be beneficial for big data tasks as well as long-term data storage, whereas the edge computing counterpart is responsible for detecting faults in semiconductor processing and swiftly resolving any issues. In a wide range of these systems, data analysis computing is pushed from the cloud level to the edge level with a hope to reduce the latency of data transferring and analysis computing [10]–[12].

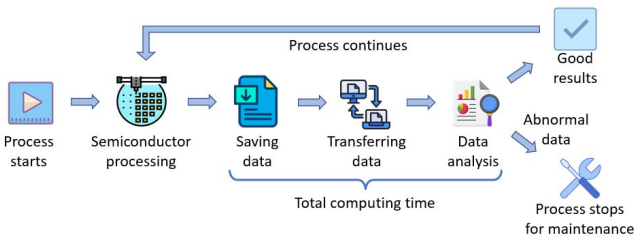


Fig. 1: An Example of Computing Time in Semiconductor Processing

B. Motivations

As an example, depicted in Figure 1, data analytic computing tasks are frequently and repeatedly performed during semiconductor manufacturing. The computing tasks may be simple process data monitoring, statistical process control, resources allocation, or more complex algorithms for early

Hung H. Nguyen, Kushagra Kushagra, and Xiao Qin are with the Department of Computer Science and Software Engineering, Auburn University, Auburn, AL 36849, USA
E-mail: hhn0005@auburn.edu, kzk0104@auburn.edu, xqin@auburn.edu

Yi Zhou is with TSYS School of Computer Science, Columbus State University, Columbus, GA 31907, USA.
E-mail: zhou_yi@columbusstate.edu

The corresponding author of this paper is Yi Zhou

978-1-6654-6087-3/22/$31.00 ©2022 IEEE
DOI 10.1109/UCC56403.2022.00039

Authorized licensed use limited to: Auburn University. Downloaded on November 17,2026 at 23:19:40 UTC from IEEE Xplore. Restrictions apply.

fails prediction and other purposes [13]. Indeed, reducing computing and processing latency is vital and desirable for increasing production throughput and achieving high product quality.

When chip size is reduced to a single-digit nanometer, the number of chips per wafer increases - the amount of data per wafer significantly goes up [14]. When it comes to an edge platform that performs data analysis, a vast amount of data needs to be transferred from equipment to the edge platform. Therefore, the latency of data analysis and computing process is enlarged. Network bandwidth, of course, is likely to be overloaded and saturated by delivering too much data from thousands of equipment in the factory to the edge layer. Furthermore, handling all sort of computing tasks at the edge layer inevitably faces the grand challenge of the system complexity: it is obliged to perform data management and analysis on all different data types from heterogeneous equipment types in semiconductor processing. These problems motivate us to propose an offloading method, for smart manufacturing in the semiconductor industry, to judiciously shift a selection of computing tasks from the edge layer to the equipment layer.

C. Benefits and Contributions

In this study, we propose a judicious method of selecting computing tasks that can be offloaded to the equipment level to achieve the lowest computing latency for smart manufacturing. With this approach in place, some computing tasks are performed at the edge level while the other selected tasks are running at the equipment layer where the data is originated.

Our empirical study in the arena of smart manufacturing in the semiconductor industry demonstrates three-fold strengths of our computation offloading mechanism. An immediate benefit of our new design is reducing the overhead of handling data for computing tasks at the edge level, including storing data into temporary storage at the edge for computation. Another advantage is curbing computing latency for smart manufacturing tasks: our offloading mechanism enables smart manufacturing systems to detect defected materials early amid a manufacturing process. Third, our offloading mechanism plays a vital role in smart manufacturing because each piece of equipment produces a different type of data, including the format of data and the meaning of data. Our offloading method reduces the development time to build a data management system that has to handle a wide variety of data types and all computing tasks at the edge layer.

We systematically conduct a performance comparison on edge-layer-based computing against equipment-layer-based counterpart. The comparison study weighs in multiple factors, including algorithmic complexities, data size, CPU utilization, and computer type - edge servers or equipment nodes. After completing the empirical study, we propose a way of offloading computing tasks to the equipment layer to immensely shorten processing latency. The contributions of this research are summarized as follows.

- *Contribution 1.* We conduct a performance comparison between edge-based computing and equipment-based computing.
- *Contribution 2.* We propose a computation offloading mechanism to cut back the latency of smart manufacturing tasks.

D. A Roadmap

Section II reviews related works about edge computing for smart manufacturing environment. Section III explains the model we use for calculating total computation time at the edge layer and the equipment layer. Section IV describes the settings of the system that we use for experiments and the factors that can affect the experimental results. In section V, the results of the experiments are reviewed and discussed. Based on the results, we propose a selection method for offloading computing tasks to the equipment level. Section VI gives conclusions of our studies and discusses about possible additional research for offloading computation from the edge layer to the equipment layer.

II. Related Works

Edge computing has evolved as an important technique for enterprise data management systems. Data generated in manufacturing has increased in both volume and complexity [15]. Building an effective system architecture for data management and analysis is vital for smart manufacturing, especially in semiconductor processing.

Umpteen attempts have been made to design system architectures to slash the computing time for real-time applications. Proposed system architectures tried to reduce latency in data analysis computing by creating multi-layer systems which collaborate between the cloud and edge layers. The computing performance can be achieved by offloading computing tasks from the cloud to the edge where it is closer to the data [16].

In [10], the authors proposed a hybrid computing framework to support various real-time requirements in smart manufacturing. This hybrid architecture system includes the following elements: cloud server, device computing layer, software defined network layer, and edge computing layer. In this architecture, all real-time computing tasks are handled by the edge computing layer.

The authors in [17] proposed an edge-supported cloud computing platform specifically for smart manufacturing. This platform includes three modules: the edge production module, the edge metrology module, and the cloud module. The edge layer handles production quality and process control.

Similarly, multiple cloud-edge systems have been proposed in [8], [18], [19] for supporting smart manufacturing.

There are four common points in the proposed cloud-edge architecture.

1) Moving time-sensitive computing tasks from the cloud layer to the edge layer in order to achieve low latency and early detection of defects.

2) The physical device layer produces and sends the data to the edge layer for computing. This layer includes machines,

Authorized licensed use limited to: Auburn University. Downloaded on November 17,2026 at 23:19:40 UTC from IEEE Xplore. Restrictions apply.

sensors, cameras, motors, industrial PCs, and other mechanical and electrical devices.
3) An algorithm exists to coordinate and schedule computing tasks in the edge layer.
4) The systems were built to support smart manufacturing.

In the above architecture, latency in computing at the edge layer increases when the amount of data increases. Furthermore, the local bandwidth from the device layer to the edge layer will become a bottleneck with the increase in the amount of data that needs to be transferred from equipment to the edge layer.

III. Modeling Computation Time at the Edge and Equipment Layers

We develop a simple yet effective computation time model with respect to two cases, which facilities making decisions of offloading computation from the edge layer to the equipment layer.

A. Computing Time on the Edge Layer

In this scenario, data analytic is carried out at the edge layer. Equipment data and process data must be transferred to the edge layer for computation. One common method in the semiconductor industry for transferring data from the equipment layer to the edge layer is file transfer. More specifically, the control system running on the equipment computer first writes processed data files during material processing. Next, after the material processing is finished, the control system sends the data files to the edge. Then, the edge performs analytical computations to check the system status and determine if the product quality is normal for the process to continue. The computing time on the edge layer can be expressed as:

$$T_{Ed_total} = T_{Eq_write} + T_{transfer} + T_{Ed_read} + T_{Ed_comp}, \tag{1}$$

where T_{Ed_total} denotes the total time to perform computations on the edge server; T_{Eq_write} denotes the total time to write all process data or test data for material to data files; $T_{transfer}$ denotes the total time to transfer data files for material from the equipment layer to the edge layer; T_{Ed_read} represents the time for the edge server to read material data files; and T_{Ed_comp} is the time for the edge server to perform computing tasks on the data set.

Assuming there are *N* process steps in total, and process step i^{th} has M_i data files, the total time for an equipment computer to write all process data or test data to data files is computed as follows:

$$T_{Eq_write} = \sum_{i=1}^{N} \sum_{j=1}^{M_i} t_{Eq_write,i,j}, \tag{2}$$

where $t_{Eq_write,i,j}$ denotes the time to write data to output data file j^{th} at process step i^{th}.

The total time to transfer data files from the equipment layer to the edge layer is calculated as:

$$T_{transfer} = \sum_{k=1}^{K} t_{transfer,k}, \tag{3}$$

where *K* represents the number of files to be transferred, and $t_{transfer,k}$ denotes the time for transferring file k^{th}.

And K can be computed as:

$$K = \sum_{i=1}^{N} M_i, \tag{4}$$

where M_i is the number of data files that are produced at process step i^{th}.

B. Computing Time on the Equipment layer

In the scenario of performing analytical computing at the equipment layer, there is no need to transfer data files from the equipment layer to the edge layer. In particular, data produced at the equipment layer is saved to local data files. After the material processing is completed, the control system running on the equipment computer reads the data file to perform analytical computations. Accordingly, the computing time on the equipment layer can be formed as:

$$T_{Eq_total} = T_{Eq_write} + T_{Eq_read} + T_{Eq_comp}, \tag{5}$$

where T_{Eq_total} denotes the total time to perform computations on the equipment layer; T_{Eq_write} denotes the total time for an equipment computer to write all process data or test data for material processing to data files, which is calculated according to Equation (2); T_{Eq_read} is the time for the equipment computer to read the local data files; and T_{Eq_comp} represents the time for the equipment computer to perform computing tasks on the data set.

IV. Experimental Setup

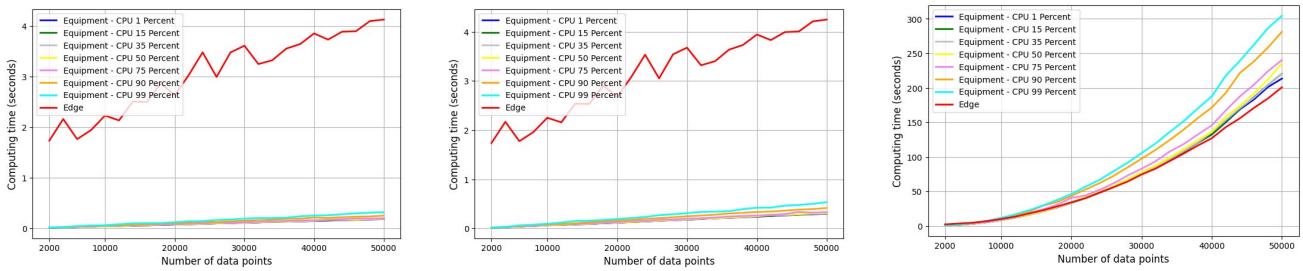
We implemented the proposed method of offloading computation from the edge layer to the equipment layer. We conducted a series of experiments with different factors that can have bearing on the performance of computation at the equipment layer. Then we compared the performance at the equipment layer with that at the edge layer.

To carry out experiments, we set up a system that includes a server for edge computing and a computer for equipment computing. The edge computer has an AMD Threadripper Pro 3975WX 32 cores processor, 512 GB memory, 2TB SSD NVMe for the operating system, and 2TB SSD SATA for data. The operating system on the edge server is Ubuntu 20.04. The equipment computer is a regular computer with Windows 10 Professional, Intel i7 8 cores CPU, 16GB memory, and 1TB hard drive.

The experiments aim to determine the factors or combination of thereof that can deliver better performance at the equipment layer.

- Data size: the number of data points which are produced for each material (for example, wafer) during processing. The data size we used for experiments range from 2000 data points to 50000 data points.
- CPU utilization: the current utilization of the CPU during analytic computation (excluding the utilization that is used for performing analytic computation).

Authorized licensed use limited to: Auburn University. Downloaded on November 17,2026 at 23:19:40 UTC from IEEE Xplore. Restrictions apply.



(a) O(n) (b) O(nlogn) (c) O(n^2)

Fig. 2: Performance on Different Complexities

- Algorithm complexity: We conducted the experiments with different algorithmic complexities: O(n), O(nlogn), and O(n^2)
- Edge or Equipment: We performed experiments on the edge server and equipment computer and compared the results.

For each algorithmic complexity, we measured computing time with different data sizes, from 2000 data points to 50000 data points. The experiments were conducted on the edge server and on the equipment computer. For the equipment computer, we measured computing time with different percentages of CPU utilization. Total computing time is measured for each test case.

V. Experimental Results

We conduct extensive experiments to evaluate the performance and effectiveness of our offloading strategy. In this part of the study, we shed light on the impacts of various factors on computing performance. In particular, we first investigate the impacts of the algorithmic complexity of computing tasks. Then, we examine the influence of the CPU utilization of equipment computers, followed by the examination of the performance factor of data size. Finally, we present a computing offload algorithm that takes into factors algorithmic complexity, CPU utilization, memory usage to make offloading decisions, thereby minimizing computing time.

A. Algorithmic Complexity of Computing Tasks

In the first group of experiments, we investigate the impacts of the time complexity (i.e., Big O) of computing tasks on computing time. We study three computational tasks with different time complexities, namely, O(n), O(nlogn), and O(n^2). We compare the computing time of the three computational tasks running on the edge layer against that on the equipment layer. To thoroughly study the equipment performance, we vary the CPU utilization of the equipment computer when performing the computational tasks.

1) Computational Tasks with O(n) and O(nlogn): Fig. 2a and 2b show the computing time for O(n) and O(nlogn) tasks running on the device computer and the edge server with respect to different numbers of data points. It indicates that performing O(n) and O(nlogn) computing tasks on the equipment computer saves more time than performing the tasks on the edge server. These results are expected. The reason is that O(n) and O(nlogn) computing tasks can be completed in a very short time. In contrast, data needs to be transferred from the equipment layer to the edge layer first in order to perform computing tasks on the edge layer. In these cases, data transfer contributes most of the latency in edge computing due to the short computation time of O(n) and O(nlogn) tasks.

2) Computational Tasks with O(n^2): When it comes to the computational tasks with the time complexity of O(n^2), the edge server outperforms the equipment computer in terms of computing time (see Fig. 2c). There exists two main reasons behind this observation. First, it takes much longer time for O(n^2) computational tasks to be completed. Second, when performing O(n^2) tasks, the number of data points plays an important role in computing time. Specifically, the computing time increases exponentially as data size increases. Unlike the O(n) and O(nlogn) cases above, data transfer from the equipment layer to the edge layer in this case becomes an insignificant factor in computing time.

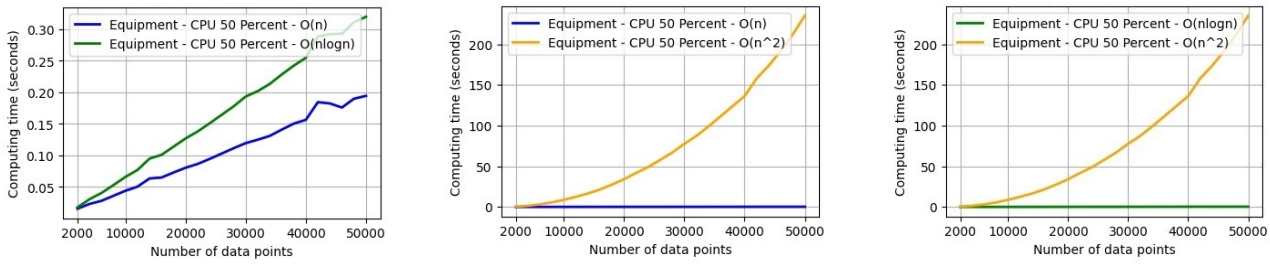
3) Performance Comparison Between Tasks of O(n), O(nlogn), and O(n^2): W compare the computing performance between tasks of O(n), O(nlogn), and O(n^2) on the equipment computer. The computing time of both O(n) and O(nlogn) tasks goes up linearly as the data size increases. On the other hand, the computing time of O(n^2) tasks grows exponentially as the data size increases. We obtains similar results with different CPU utilization. Fig. 3 shows the result when CPU utilization is 50%.

B. CPU Utilization

In this set of experiments, we evaluate the impacts of CPU utilization on computing time. We measure the computing time of O(n), O(nlogn) and O(n^2) tasks under different percentages of CPU utilization and different sizes of data.

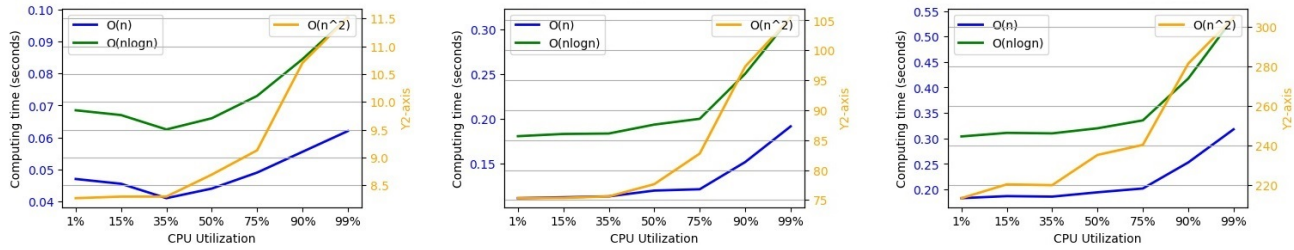
As shown in Fig. 4, for the cases of 10000, 30000, and 50000 data points, when the CPU is busy for other tasks, it takes a longer time for the computational tasks to complete. The computing time increases exponentially after CPU utilization becomes higher than 50%. The results show that CPU utilization is a significant factor to determine whether to

Authorized licensed use limited to: Auburn University. Downloaded on November 17,2026 at 23:19:40 UTC from IEEE Xplore. Restrictions apply.



(a) Performance O(n) vs O(nlogn) (b) Performance O(n) vs O(n^2) (c) Performance O(nlogn) vs O(n^2)

Fig. 3: The impact of algorithmic complexity on performance of equipment computer with 50% CPU utilization



(a) Performance - 10000 Data Points (b) Performance - 30000 Data Points (c) Performance - 50000 Data Points

Fig. 4: The impact of CPU utilization on performance of equipment computer

offload computing tasks from the edge layer to the equipment layer.

C. Data Size

Now we are in the position of investigating the impacts of data size on the computing time. Again, we measure the computing times of O(n), O(nlogn) and O(n^2) tasks with respect to the number of data points. We carry out the experiments under different CPU utilization conditions of the equipment computer ranging from 1% to 99%. The experimental results are plotted in Fig. 3 for CPU utilization of 50%.

We can observe from the experimental data, as an example in Fig. 3, the computing time grows as the data size increases. And among these three computational tasks of different complexities, the curves of O(n) and O(nlogn) tasks grow relatively linearly. In contrast, the computing time of O(n^2) task soars exponentially. In a nutshell, the number of data points serves as a non-negligible factor in computing time. The larger the number of data points, the longer the computing time.

D. Algorithm Design

In this part of the study, we propose our computing offload algorithm that facilitates making decisions on when to offload heavy tasks from the edge layer to the equipment layer. In order for the equipment computer to perform its functionalities, such as controlling the equipment and producing data, it is vital to preserve certain computing resources (e.g., CPU time and memory space) and avoid pushing the equipment computer to its maximum limits. Our observations and analysis above reveal that the most significant factors that affect computing time are the time complexity of computational tasks and the CPU utilization of the equipment computer. For example, our experimental results in Section V-A, Section V-B, and Section V-C indicate that O(n^2) tasks perform better at the edge layer, whereas O(n) and O(nlogn) tasks perform better at the equipment layer.

Taking into account the three significant factors: CPU utilization, memory usage, and time complexity, we propose an algorithm to determine whether to offload a specific computation task to the equipment layer or perform the task on the edge layer, aiming to minimize computing time.

The computing offload algorithm outlined in Algorithm 1 first checks the current CPU utilization of the equipment computer; if it is greater than a pre-defined CPU threshold, the computational task will be performed on the edge layer (see Lines 1-3 in Algorithm 1). Next, the computing offload algorithm examines the current memory usage; if it is greater than a pre-defined memory threshold, the computational task will be performed on the edge layer (see Lines 6-9 in Algorithm 1). Lastly, the computing offload algorithm looks into the time complexity of the computational task; if the time complexity is bigger than $O(n^2)$, the computational task will be executed on the edge layer, otherwise, the computational task will be offloaded and performed on the equipment computer (see Lines 11-15 in Algorithm 1).

Authorized licensed use limited to: Auburn University. Downloaded on November 17,2026 at 23:19:40 UTC from IEEE Xplore. Restrictions apply.

Algorithm 1 Computing Offload Algorithm

Input:
CPU_Threshold
RAM_Threshold
Current_CPU_Utilization
Current_RAM_Utilization

if $Current_CPU_Utilization > CPU_Threshold$ **then**
    Computing on edge layer
    Return
end if

if $Current_RAM_Utilization > RAM_Threshold$ **then**
    Computing on edge layer
    Return
end if

if $Complexity >= O(n^2)$ **then**
    Computing on edge layer
else
    Computing on equipment layer
end if

VI. Conclusions and Future Works

Our experiments showed that computing tasks could be offloaded from the edge layer to the equipment layer in some instances resulting in an improvement in performance and a reduction in the data transfer requirement between layers in a manufacturing environment. On these bases, we propose an algorithm to determine which layer to perform a computational task based on the factors of the existing conditions of the system.

In this study, we used data file copying as the method of transferring data between layers. In future research, other methods of transferring data in semiconductor manufacturing can be considered, such as SECS/GEM or KLARF. We will conduct experiments to choose the best data transfer method for each specific scenario.

Another dimension for the desired improvement we will pursue is developing a communication model to allow the system to send requests to the equipment layer to perform a predefined algorithm. The system can also send a new algorithm to the equipment layer when needed. We also intend to add a new feature to the system that will allow switching computing tasks between layers based on the existing conditions and availability of resources. We firmly believe that such a system will help in improving manufacturing throughput and product quality.

Acknowledgment

This work is supported in part by the U.S. National Science Foundation under Grants IIS-1618669, OAC-1642133, and CCF-0845257.

References

[1] M. Khakifirooz, M. Fathi, and K. Wu, "Development of smart semiconductor manufacturing: Operations research and data science perspectives," *IEEE Access*, vol. 7, pp. 108 419–108 430, 2019.

[2] S.-J. Chen, H.-W. Liu, and W.-J. Wang, "A fault tolerance mechanism for semiconductor equipment monitoring," in *2017 IEEE 7th International Symposium on Cloud and Service Computing (SC2)*, 2017, pp. 171–176.

[3] C.-P. Tsai, H.-C. Hsiao, Y.-C. Chao, M. Hsu, and A. R. Chang, "Bridging the gap between big data system software stack and applications: The case of semiconductor wafer fabrication foundries," in *2018 IEEE International Conference on Big Data (Big Data)*, 2018, pp. 1865–1874.

[4] J. Moyne, J. Samantaray, and M. Armacost, "Big data emergence in semiconductor manufacturing advanced process control," in *2015 26th Annual SEMI Advanced Semiconductor Manufacturing Conference (ASMC)*, 2015, pp. 130–135.

[5] P. K. Illa and N. Padhi, "Practical guide to smart factory transition using iot, big data and edge analytics," *IEEE Access*, vol. 6, pp. 55 162–55 170, 2018.

[6] S. Trinks and C. Felden, "Edge computing architecture to support real time analytic applications : A state-of-the-art within the application area of smart factory and industry 4.0," in *2018 IEEE International Conference on Big Data (Big Data)*, 2018, pp. 2930–2939.

[7] J. Vater, L. Harscheidt, and A. Knoll, "A reference architecture based on edge and cloud computing for smart manufacturing," in *2019 28th International Conference on Computer Communication and Networks (ICCCN)*, 2019, pp. 1–7.

[8] J. Moon and J. Jeong, "Smart manufacturing scheduling system: Dqn based on cooperative edge computing," in *2021 15th International Conference on Ubiquitous Information Management and Communication (IMCOM)*, 2021, pp. 1–8.

[9] C.-C. Lin, D.-J. Deng, Y.-L. Chih, and H.-T. Chiu, "Smart manufacturing scheduling with edge computing using multiclass deep q network," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 7, pp. 4276–4284, 2019.

[10] X. Li, J. Wan, H.-N. Dai, M. Imran, M. Xia, and A. Celesti, "A hybrid computing solution and resource scheduling strategy for edge computing in smart manufacturing," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 7, pp. 4225–4234, 2019.

[11] C. Yang, S. Lan, L. Wang, W. Shen, and G. G. Q. Huang, "Big data driven edge-cloud collaboration architecture for cloud manufacturing: A software defined perspective," *IEEE Access*, vol. 8, pp. 45 938–45 950, 2020.

[12] C. Jiang, J. Wan, and H. Abbas, "An edge computing node deployment method based on improved k-means clustering algorithm for smart manufacturing," *IEEE Systems Journal*, vol. 15, no. 2, pp. 2230–2240, 2021.

[13] M. Ghahramani, Y. Qiao, M. C. Zhou, A. O'Hagan, and J. Sweeney, "Ai-based modeling and data-driven evaluation for smart manufacturing processes," *IEEE/CAA Journal of Automatica Sinica*, vol. 7, no. 4, pp. 1026–1037, 2020.

[14] R. G. Goss and K. Veeramuthu, "Heading towards big data building a better data warehouse for more data, more speed, and more users," in *ASMC 2013 SEMI Advanced Semiconductor Manufacturing Conference*, 2013, pp. 220–225.

[15] Q. Qi and F. Tao, "A smart manufacturing service system based on edge computing, fog computing, and cloud computing," *IEEE Access*, vol. 7, pp. 86 769–86 777, 2019.

[16] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE internet of things journal*, vol. 3, no. 5, pp. 637–646, 2016.

[17] J. Ying, J. Hsieh, D. Hou, J. Hou, T. Liu, X. Zhang, Y. Wang, and Y.-T. Pan, "Edge-enabled cloud computing management platform for smart manufacturing," in *2021 IEEE International Workshop on Metrology for Industry 4.0 IoT (MetroInd4.0 IoT)*, 2021, pp. 682–686.

[18] C. Zhang and W. Ji, "Edge computing enabled production anomalies detection and energy-efficient production decision approach for discrete manufacturing workshops," *IEEE Access*, vol. 8, pp. 158 197–158 207, 2020.

[19] C. K. M. Lee, Y. Z. Huo, S. Z. Zhang, and K. K. H. Ng, "Design of a smart manufacturing system with the application of multi-access edge computing and blockchain technology," *IEEE Access*, vol. 8, pp. 28 659–28 667, 2020.

Authorized licensed use limited to: Auburn University. Downloaded on November 17,2026 at 23:19:40 UTC from IEEE Xplore. Restrictions apply.