Optimized Task Scheduling in Cloud Manufacturing with Multi-level Scheduling Model

Xiaoli ZHU
Wanbo Institute of Science & Technology, Huainan 232251, China

***Abstract*—Cloud Manufacturing (CMfg) utilizes the cloud computing paradigm to provide manufacturing services over the Internet flexibly and cost-effectively, where users only pay for what they use and may access services as needed. The scheduling method directly impacts the overall efficiency of CMfg systems. Manufacturing industries supply services aligned with customer-specific needs recorded in CMfg systems. CMfg managers develop manufacturing strategies based on real-time demand to establish service delivery timing. Many elements influence customer satisfaction, including dependability, timeliness, quality, and pricing. Therefore, CMfg depends on the use of multi-objective and real-time task scheduling. Multi-objective evolutionary algorithms have effectively examined many solutions, such as non-dominant, Pareto-efficient, and Pareto-optimal solutions, using both actual and synthetic workflows. This study introduces a new Multi-level Scheduling Model (MSM) and evaluates its effectiveness by comparing it with other multi-objective algorithms, including the weighted genetic algorithm, the non-dominated genetic sorting Algorithm II, and the starch Pareto evolution algorithm. The primary emphasis is on assessing the efficacy of algorithms and their suitability in commercial multi-cloud setups. The MSM's dynamic nature and adaptive features are emphasized, indicating its ability to effectively handle the complexity and demands of CMfg and resolve the scheduling issue within this environment. Experimental results suggest that MSM outperforms other algorithms by achieving a 20% improvement in makespan.**

Keywords*—*Cloud manufacturing; multi-level scheduling model; task scheduling; multi-objective optimization; resource allocation

I. Introduction

Cloud Manufacturing (CMfg) is a new manufacturing paradigm characterized by service-oriented, knowledge-based, and resource-sharing manufacturing. It can virtualize the manufacturing resources into services and realize the control and transmission of virtual manufacturing resources by extending the cloud, ensuring the networked, integrated, and adaptive collaboration of multi-user parties for the entire life cycle of the product [1]. With CMfg, physical resources are conveniently, efficiently shared, and allocated to produce customized products based on consumer demand [2]. Officially launched in 2010, CMfg is widely regarded as a promising direction for the future of manufacturing. Over the past decade, academics and industry have extensively studied and debated this issue [3]. The related topics of CMfg include architectural design, resource virtualization, service selection, service allocation, task scheduling, and service discovery [4]. Despite significant research efforts, the desired concept of CMfg has not yet been realized.

As a result of technological advances in virtualization and commercialization, cloud computing can schedule tasks efficiently on virtual machines [5]. Efficient distribution of resources across each task is a crucial aspect of distributed computing, and scheduling plays an essential role in this [6]. There are currently different scheduling techniques, including cloud service, workflow, static, and dynamic scheduling. Task scheduling is significantly impacted by challenges such as performance, reliability, scalability, load balancing, and dynamic resource reallocation across processing nodes [7]. A robust scheduling method is essential for coordinating work in cloud computing. The CMfg model consists of design, manufacturing, and logistics activities. These tasks are supported by the respective design, manufacturing, and logistics clouds. The suppliers can be individuals or companies, while the customers can be end-users or businesses. A central information store is a hub connecting operators, customers, and suppliers [8]. The sequence of interactions includes the following steps: providers and consumers interact; Consumers submit their requirements to operators; Operators assign tasks to providers based on consumer needs; Providers register their available resources; Operators deliver the resulting output to consumers [9].

In the CMfg operating paradigm, operators act as administrators responsible for monitoring and controlling a CMfg platform. Their debut enables consumers to receive affordable, reliable, and world-class manufacturing services whenever they need them, conveniently via the cloud platform [10]. In addition, the cloud platform offers tools that allow providers to distribute their resources and capabilities efficiently. Under the operator-led unified management, suppliers provide shared-purpose manufacturing resources to the CMfg platform and receive manufacturing tasks from the cloud platform. The customer base comprises corporate customers and individual buyers [11]. Under this centralized management structure, customers submit their requested tasks to the CMfg platform and subsequently receive the performance results of their orders. CMfg uses manufacturing paradigms to develop knowledge that includes rules, concepts, models, protocols, and algorithms [12]. This information is critical in all service lifecycle phases, including service creation, management, and implementation.

Task scheduling represents a significant problem. The efficient organization of work in distributed systems depends to a large extent on precise data about the availability of resources. Resource providers often deliver this information to a central database that planners can access. The exponential growth of cloud providers is obvious [13]. Within a commercial multi-cloud system, individual providers are primarily motivated to optimize their revenues and may put their interests ahead of the benefit to consumers and other providers. In a multi-cloud enterprise environment where multiple cloud providers are involved and have sensitive information about their resources, application planners must be careful and not rely on the information provided by the providers about the status of their resources. There is a constant risk that providers will misrepresent private data.

This paper follows the following structure. CMfg-related work is reviewed in Section II focusing on scheduling models and optimization techniques. Section III describes our proposed method, emphasizing its adaptive features and effectiveness in addressing CMfg complexities. Section IV provides detailed empirical evaluation results. MSM is compared with existing algorithms, and its practical applications are discussed in section V. Finally, Section VI concludes the paper.

II. Related Work

Scheduling methodologies are one of the most elaborate research areas in CMfg, as the distribution of resources in CMfg poses a series of challenging and intelligent research problems. Several scheduling techniques have been designed for task allocation, resource synchronization, and system optimization. The methods adopted in these papers include multi-objective evolutionary algorithms, Chaos Optimization Algorithm (COA), and creative work that integrates Deep Reinforcement Learning (DRL) with attention mechanisms. Each research corresponds to a different problem, which includes integrated planning, production resource planning, end-to-end solutions, collaborative task planning, logistics integration, and setup time/cost. This section provides a thorough examination and comparative assessment of these studies, revealing their approaches, objectives, and notable results while emphasizing their contribution to addressing complex planning problems in CMfg. Table I provides a comparative analysis of the works, highlighting the diversity of approaches used and their respective contributions to addressing planning challenges in CMfg.

The main goal of the CMfg paradigm is to centralize distributed manufacturing capabilities and businesses, thus enabling enhanced personalized production. Production orders consist of multiple items jointly fulfilled by distributed providers at lower costs. The CMfg platform sets meaningful priorities, identifies acceptable suppliers and production processes for numerous orders, and plans hybrid activities resulting from different orders across manufacturing resources. The goal is to increase production efficiency by managing the trade-offs between orders. Laili, et al. [14] examined the multi-phase integrated scheduling of hybrid jobs in a CMfg context. This included assigning order priorities, selecting suppliers and production processes, and planning production lines. This technique considers five main objectives to evaluate the interrelationships between diverse resources and manufacturing operations. Six exemplary multi-objective evolutionary algorithms were used to address the integrated planning problem. The experiments conducted under six different production conditions indicate that the integrated scheduling method outperforms standard sequential decision-making, resulting in lower production costs and times. In addition, a comprehensive study was carried out to determine the most suitable solution to the integrated planning problem in different situations by comparing the six methods.

Hu, et al. [15] studied the scheduling problem for manufacturers in a CMfg environment. They analyze five factors that impact manufacturer resource planning: task load, task reliability, manufacturing efficiency, availability of manufacturing resources, and Internet of Things (IoT) compatibility. The research creates planning indices and a model with an objective function. The objective function is efficiently resolved using the COA to arrange production orders across several domains. The simulation results validate the comprehensiveness and effectiveness of the proposed planning index. This study integrates all pertinent factors that impact manufacturer planning in the CMfg environment by developing a mathematical model. The scheduling problem is simplified into an arithmetic problem using a linear programming approach. The manufacturer's scheduling algorithm, rooted in chaos theory, effectively addresses the issue of inference and delivers high-quality service in conditional manufacturing to consumers via the CMfg platform.

DRL is increasingly recognized as a viable approach to solving scheduling challenges in CMfg and demonstrates impressive performance in dynamic and unpredictable cloud environments. Nevertheless, the industry needs improved planning algorithms and readily available modeling methods to support the actual use of these advances. Wang, et al. [16] proposed a unique end-to-end scheduling solution to solve job scheduling challenges in CMfg precisely. Their technique utilizes the multi-head attention process to uncover connections between companies and activities. The model is trained using DRL. What is noteworthy is that this concept has a significant reduction in response times compared to heuristic algorithms and allows planning solutions to be created in a matter of seconds. In contrast to previous DRL algorithms, the approach has higher planning performance and uses a more easily understandable modeling method. Significantly, the proposed model only depends on the objective function to ensure continuous training, so there is no need for a reward function based on steps. Combining multi-head attention with DRL for planning problems is a novel approach with promising results. The experimental results of a case study on processing vehicle structural components in CMfg show that the proposed scheduling approach outperforms priority distribution rules, heuristic algorithms, and DRL algorithms in both performance and efficiency.

Chen, et al. [17] studied cloud-edge collaboration manufacturing task scheduling (CETS) to improve customer satisfaction and optimize production balance. CETS aims to increase the efficiency of cloud-based manufacturing services, especially at individual process levels. It coordinates the scheduling of tasks by leveraging current production data at the edge and manufacturing service information in the cloud environment. The study introduces an attention-based DRL algorithm designed for CETS requirements, which are highly dynamic and state-intensive. The DRL method is built on the mathematical model of CETS as a partially observable Markov decision process. It then creates the AV-MPO framework that uses a Gated Transformer-XL (GTrXL) within an On-Policy Maximum Posterior Policy Optimization (V-MPO) framework. The efficacy, learning progress, generalization, large scale, and robustness of AV-MPO are investigated extensively via experiments. Moreover, AV-MPO is compared with rule-based algorithms and other state-of-the-art DRL methods, such as Proximal Policy Optimization (PPO), Soft Actor-Critic (SAC), and Dueling Deep Q Network (Dueling DQN). The experimental results confirm that AV-MPO successfully handles the inherent difficulties of the CETS problem. Compared to other algorithms, it shows higher efficiency in handling this work.

The CMfg system has undergone significant changes with the advancement of new technologies. Customers can request a wide range of services through a customer-centric framework, gaining access to distributed production resources. Salmasnia and Kiapasha [18] criticize the unreasonable assumption made in previous research that all tasks are immediately available when planning begins. To ensure the model's accuracy in representing reality, two crucial factors are taken into account: (1) the time and cost associated with transferring subtasks between various services offered by companies located in different geographical regions, and (2) the time and cost involved in establishing a service capable of performing multiple subtasks. A thorough model integrates three essential components: the impact of cost on the CMfg system, the duration required to fulfill an order, and the level of service quality. The results underscore the significance of considering the arrival time of tasks and the temporal and financial implications of logistics and setup to achieve more accurate outcomes. The GAMS program solves small and medium-scale problems, while a genetic algorithm is developed to address larger issues. Moreover, a sensitivity analysis is performed to understand better how variables such as time, cost, and user requirements impact the final solution. This comprehensive approach facilitates a more profound comprehension of the intricate interplay of diverse elements within CMfg systems.

Zhang, et al. [19] propose a new approach called Individualized Requirement-Driven CMfg Multi-Task Scheduling (IRCMMS) to address customers' specific needs in demand-driven cloud manufacturing. This strategy attempts to address the particular needs of individual consumers while benefiting the overall system. This technique is primarily designed to obtain an approximation to an optimal Pareto solution set, thereby providing a wider range of possibilities for the CMfg system. The validation process confirmed the applicability and effectiveness of the IRCMMS model, which involved several simulation instances and experimental data. Moreover, these findings emphasize the algorithm's efficacy in effectively addressing the challenges posed by the IRCMMS model.

Wang, et al. [20] presented a novel offline scheduling method for DRL, which aims to overcome the challenges of online trial-and-error methods while maintaining the intrinsic advantages of DRL. A sequential Markov approach was suggested to represent decision-making processes, where each task was defined as an individual agent. Subsequently, a Decision Transformer (DT) framework was presented to convert the decision problem in online planning into a categorization problem in an offline setting. A reference model based on attention was developed and trained offline through the DT architecture to serve as an agent's guide. The experimental results showed that the proposed approach can support online DRL algorithms such as Deep Double Q-Network (DDQN), Deep Recurrent Q-Network (DRQN), PPO, and the offline learning algorithm Behavior Cloning (BC) consistently outperformed, both in terms of planning performance and model generalization. The results highlight the effectiveness of the proposed offline DRL scheduling algorithm in providing excellent scheduling performance while avoiding the difficulties associated with online trial-and-error methods.

TABLE I. SCHEDULING APPROACHES IN CMFG

Reference	Objective	Methodology	Findings
Laili, et al. [14]	Integrated scheduling in CMfg	Multi-objective evolutionary algorithms	More effective and reduced costs/time
Hu, et al. [15]	Manufacturer scheduling factors in CMfg	Chaos Optimization Algorithm	COA efficiently handled scheduling jobs in various conditions
Wang, et al. [16]	End-to-end scheduling solution in CMfg	Multi-head attention with deep reinforcement learning	Outperformed various DRL and heuristic algorithms
Chen, et al. [17]	Cloud-edge collaboration manufacturing task scheduling	AV-MPO, attention-based DRL method	AV-MPO is efficient in handling CETS and outperformed other algorithms
Salmasnia and Kiapasha [18]	Task transfer and setup time consideration in CMfg	Developed a model considering cost implications, job completion time, and service quality	Consideration of logistics setup time is crucial for accurate CMfg solutions
Zhang, et al. [19]	Individualized requirement-driven CMfg multi-task scheduling	Multifactorial evolutionary algorithm	IRCMMS efficiently addressed individual customer requirements
Wang, et al. [20]	Offline DRL scheduling in CMfg	Offline Markov decision process modeling, Decision Transformer framework	Outperformed online DRL algorithms in scheduling performance

III. PROPOSED METHOD

The suggested method is divided into three steps. The cloud manager maintains a global queue to handle inbound service requests from clients at the first stage, known as matching, where each request represents one or more tasks. The cloud manager selects the i^{th} task in the global queue and calculates the completion time that this task will use in multiple Virtual Machines under the control of the Content Security Policy (CSP). The entire time includes both scheduling and execution. The manager decides the scheduling process of the k^{th} cloud at the allocation stage to provision service requests to minimize the makespan. Within the cloud, allocation follows a First in First out (FIFO) principle. Additional tasks queued while the manager is allocating are scheduled in FIFO order. The manager selects an alternative VM for the first task that provides the fastest completion time.

When implementing the Continuous Linked Settlement (CLS) strategy, the first task is specifically assigned to VM-2, given that the CLS strategy does not comply with the FIFO order. This stage enables adjustments to tasks depending on the scheduling system. Subsequent scheduling is carried out to complete the calculations. Crucially, every cloud has the potential to carry out several tasks concurrently. The suggested method consolidates service needs worldwide, allowing the manager to choose each job from the queue progressively. The manager evaluates the appropriateness of tasks for all VMs to choose the most suitable VM. Afterward, the manager selects the c^{th} cloud with the most appropriate virtual machine and the necessary state and index. To ascertain the optimal state of the VM, the first step involves invoking the procedure, which examines the scheduling approach used by the cloud c. This entails determining whether the cloud uses Round-Robin (RR) or cloud list scheduling.

When RR scheduling is used, the method determines the number of VMs (*VMcount*) in cloud c. At first, the value of *VMcount* is set to 1. After assigning the task to the main VM and updating the VM count, the algorithm proceeds with the RR scheduling approach, where the main VM is scheduled first, followed by the other VMs. If the number of VMs equals the index, the task is given to the VM with that index. Otherwise, the task fails. If the CLS technique is selected and neither RR scheduling nor CLS in cloud c can handle the task, the algorithm will search for the most suitable VM across all available clouds. Fig. 1 depicts the multi-cloud architecture. The following steps delineate the procedure for scheduling and rescheduling the queues and index. The scheduling process involves initializing cloud settings and implementing the RR scheduling algorithm. It includes identifying VMs, allocating the task to the principal VM, updating the *VMcount*, managing errors, and iterating through the process. Rescheduling entails evaluating unlimited cloud tasks, exploring different values for the index, and adjusting c from 1 to M (the total number of clouds). This process calls for scheduling that includes the parameters of (i, cloud, and index).

Table II depicts the cloud control matrix structure. Two separate clouds are present in this situation, each consisting of two VMs that use different scheduling techniques: RR and cloud list. Given the assumption that tasks are received in a sequential numeric sequence by the cloud manager when the first task (T_0) arrives, the manager determines the VM that can complete the task quickly. As a result, the matching process is ignored, and the cloud control matrix is modified accordingly, indicating that *CCM* (T_0, *VM2*) has an infinite value (∞). Later, the manager identifies another VM (VM3) with the quickest completion time for task T_0.

Consequently, *VM1* is allocated task T_1 since it has a shorter completion time. Afterward, the RR scheduling algorithm is used to schedule T_1 on *VM1*. Subsequently, with the arrival of task T_3, the manager does an assessment and allocates it to *VM4* after scrutinizing the completion durations of T_3 on all accessible VMs. The completion timings are as follows: 6 plus 2, 3 plus 2, 8 plus 3, and 5 plus 0, respectively. Out of these options, the sum of 5 and 0 reflects the shortest time needed to complete the task, resulting in the assignment of T_3 to *VM4*.

The study introduces a multi-level scheduling strategy for dynamic workflow scheduling, abbreviated as *MSM/M2S*. The suggested technique is a list scheduling heuristic that prioritizes activities according to their performance and then organizes them in the setup order. Workflow assignments are prioritized based on their bottom level, described in graph theory as the longest route from each task to the exit task. The ranking is decided by each task by the use of the following recursive function:

$$rank\ (i) = \begin{cases} load\ (i) + \max\{comm\ (i,j) + rank\ (j)\} & if\ i \neq exit \\ load\ (i) & if\ i = exit \end{cases} \quad (1)$$

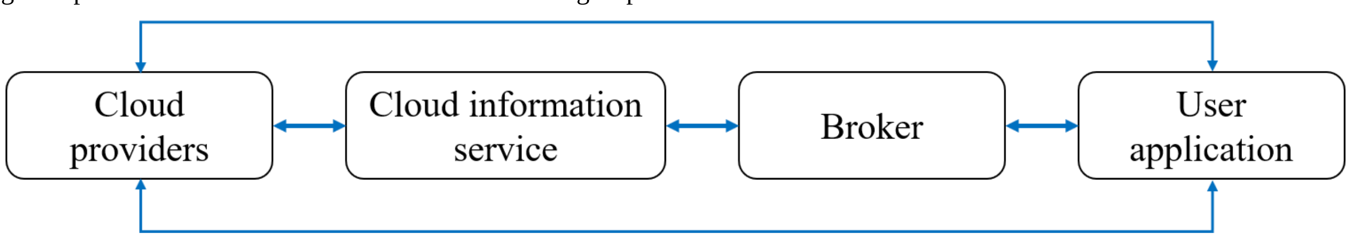

Fig. 1. Multi-cloud architecture.

TABLE II. CLOUD CONTROL MATRIX STRUCTURE

Clouds	VM	T_0	T_1	T_2	T_3	T_4
First cloud (RR scheduling)	VM1	8	1	6	5	3
	VM2	1	4	3	3	6
Second cloud (cloud list scheduling)	VM3	2	9	7	7	4
	VM4	7	5	4	2	2

Eq. (1) defines *load(i)* as the remaining burden of task *i*, and *comm(i,j)* as the communication output between tasks *i* and *j*. The reason for using makespan values to prioritize assignments is based on the structural hierarchical aim that preserves priority linkages in the structured list and substantially influences cost since longer activities take more resources. To organize the task list, we assume a recursive MSM, where n tasks are scheduled one after another using the multi-level scheduling strategy.

The proposed approach for dynamic workflow scheduling consists of a series of steps. These steps include initiating the algorithm and assigning tasks to a list, initializing an empty rank list, starting with the exit task, determining ranks based on a recursive function that takes into account workload and communication output, sorting tasks in descending order of ranks, conducting an auction among tasks using MSM, assigning the winning task to allocated resources, removing completed tasks, and concluding the scheduling strategy by paying the final cost to the task winner. This technique employs a prioritization strategy that considers the performance and execution time of tasks. It aims to optimize the usage of resources in situations that require dynamic workflow scheduling.

While attaining balance in a game is important, the system's usefulness becomes unnecessary if it cannot reach this equilibrium within an acceptable timescale. The technique, which is based on bargaining, also deals with inherent complications in communication. The complexities of the MSM system's algorithms and communication methods are carefully analyzed, calculated, and explained. A comprehensive guide is provided to improve understanding of the suggested method. The presented workflow illustrates that the load of each step indicates its exceptional use for rank estimation. Fig. 2 and Table III depict time and cost as the two essential resources for task execution. The cost considerations are clearly outlined since the suggested solution focuses on a multi-cloud environment. The expense of doing a job on a virtual machine is determined by the execution duration and the cost per unit of time, expressed using a multiplication function, as shown in Eq. (2).

$$cost = \sum_{i=1}^{m}(c_t(t_i) - s_t(t_i)) \times p_j \tag{2}$$

In Eq. (2), s_t represents the beginning time, c_t stands for the completion time, and p_j signifies the price factor. Fig. 2 illustrates the process, accompanied by two matrices that provide specific information on the costs and time required for the resources. The placements of the workflow assignments are calculated based on the first stages. The given tasks are planned by organizing them in decreasing order depending on their rankings. This ordered list, indicated as $A = (T_1; T_2; T_3; T_4)$, is used for scheduling. The first auction begins at time zero, which is assumed to be the start time of task T_1.

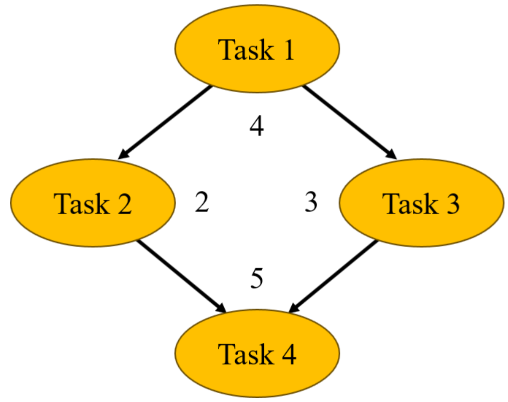

Fig. 2. Task execution process.

TABLE III. TASK DESCRIPTION

	Route 1		Route 2	
Task id	Cost	Time	Cost	Time
1	2	4	4	2
2	6	6	7	4
3	6	6	7	4
4	2	2	3	1

IV. RESULTS

When evaluating balance failure, the main criterion is the cost of disorder, which represents the difference between the highest possible job value of a balance in the game and the desired output. Researchers are focused on multi-level optimization to obtain a collection of better non-dominated solutions known as the Pareto set rather than a single perfect solution. The theoretical evaluation of the cost associated with insurrection is impractical and ignored. Here, some of the findings are shown, while the rest of the results are related to randomly created processes.

The proposed model (MSM) is being evaluated in comparison to numerous other approaches, including Weighted Genetic Algorithm (WGA) [21], Bi-Objective Scheduling Algorithm (BOSA) [22], Non-dominated Sorting Genetic Algorithm-II (NSGA-II) [23], and Strength Pareto Evolutionary Algorithm (SPEA2) [24]. Since MSM deals with multi-objective requirements, other comparable models were considered for comparison purposes. SPEA2 improves SPEA by preserving boundary solutions while offering Pareto-optimum outcomes. NSGA-II offers enhanced capabilities for handling three or more objective functions. This is achieved through reference points to ensure variety in Pareto points. The Bi-objective scheduling technique uses an approximation method to determine the best solutions on the Pareto curve. All strategies use Pareto functions to resolve problems with multiple objectives by assigning weights to objective functions.

V. Discussion

The Pareto set is generated a posteriori, and then an arrangement suitable to the client's needs is selected. The task count and execution sequences of each arrangement are defined by two fitness functions, one for time and one for cost. The algorithms utilized include SPEA2, NSGA-II, BOSS, and WGA. MSM produces non-dominated arrangements compared to Pareto, based on comparative analysis and graphical representation. Table IV summarizes the simulation parameters for the proposed model. A comparison of MSS and Pareto Front arrangements evaluated by multiple evolutionary algorithms is illustrated in Fig. 3 and Fig. 4. Accordingly, Fig. 5 and Fig. 6 present the objective space of randomly generated workflows, providing a comprehensive view of the results of various algorithms.

The study's limitations include the complexity associated with integrating real-time data and fluctuating demand patterns within CMfg systems, which may influence scheduling accuracy and responsiveness. Additionally, while MSM shows promise in improving scheduling efficiency, its applicability to large-scale CMfg operations and the generalizability of findings across different industrial contexts require further validation and refinement. Future research could focus on developing hybrid optimization approaches that integrate machine learning techniques to adaptively optimize scheduling strategies in response to evolving production environments. Moreover, exploring the integration of IoT-enabled sensors for real-time data acquisition and predictive analytics could enhance MSM's performance in anticipating and mitigating disruptions within CMfg workflows. Addressing these limitations and pursuing these avenues of research will advance our understanding and practical application of scheduling models in contemporary manufacturing settings.

TABLE IV. Simulation Settings

Parameters	Value
Number of tasks	1000
Energy	200 J
CPU time	160 ms
Number of clouds	5
Number of VMs	20-100
Number of users	100

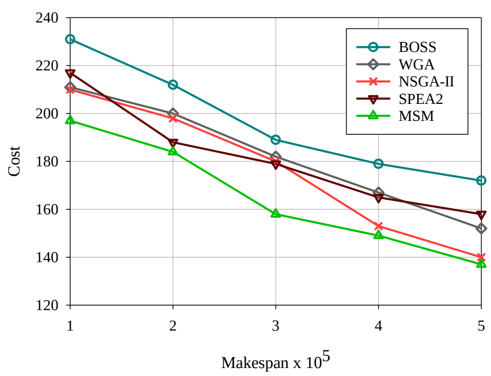

Fig. 3. Cost comparison for ten resources and 100 tasks.

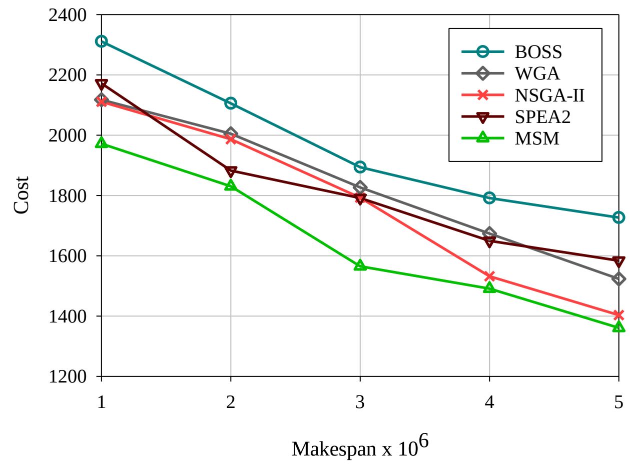

Fig. 4. Cost comparison for 20 resources and 1000 tasks.

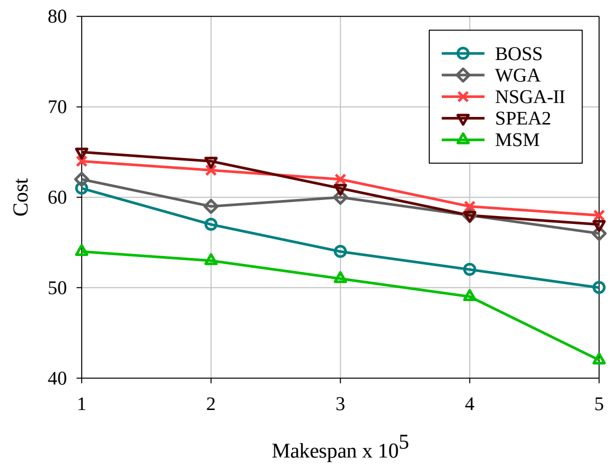

Fig. 5. Cost comparison for randomly generated workflow with 50 resources and 100 tasks.

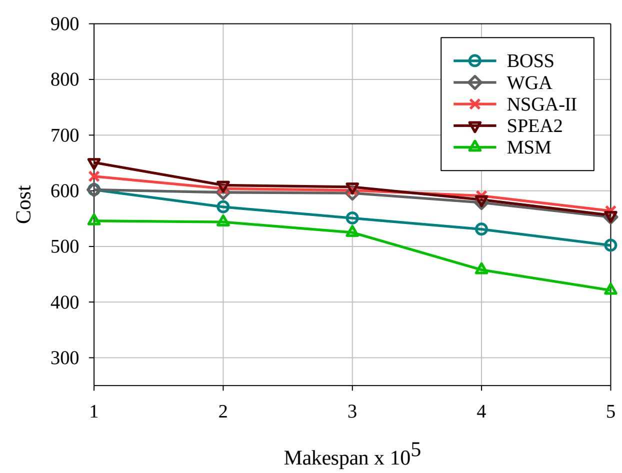

Fig. 6. Cost comparison for randomly generated workflow with 100 resources and 1000 tasks.

VI. Conclusion

This study examined the critical area of efficient scheduling within the expansive landscape of cloud computing and CMfg. Global research attention has been captured by the emergence of CMfg, which offers on-demand manufacturing services through the Internet. With real-world and synthetic workflow applications, this research faced numerous challenges associated with effective scheduling within CMfg. The multi-objective evolutionary algorithms evaluated solutions via non-dominant, optimal, efficient, or non-inferior evaluations. The MSS algorithm was compared with popular algorithms like SPEA2, NSGA-II, BOSS, and WSGA. These algorithms are tested against efficiency and applicability in commercial multi-cloud environments. This research demonstrated that MSS can be dynamic and adaptive, navigating the intricacies and demands of manufacturing and scheduling processes. Comparative analysis highlighted MSS's distinct advantages and effectiveness in optimizing scheduling mechanisms, particularly in complex multi-cloud environments. The results of this study offer valuable information on how to improve scheduling efficiency in cloud-based manufacturing paradigms in the future.

References

[1] V. Hayyolalam, B. Pourghebleh, A. A. P. Kazem, and A. Ghaffari, "Exploring the state-of-the-art service composition approaches in cloud manufacturing systems to enhance upcoming techniques," The International Journal of Advanced Manufacturing Technology, vol. 105, no. 1-4, pp. 471-498, 2019.

[2] V. Hayyolalam, B. Pourghebleh, M. R. Chehrehzad, and A. A. Pourhaji Kazem, "Single-objective service composition methods in cloud manufacturing systems: Recent techniques, classification, and future trends," Concurrency and Computation: Practice and Experience, vol. 34, no. 5, p. e6698, 2022.

[3] C. Wan, H. Zheng, L. Guo, X. Xu, R. Y. Zhong, and F. Yan, "Cloud manufacturing in China: A review," International Journal of Computer Integrated Manufacturing, vol. 33, no. 3, pp. 229-251, 2020.

[4] Y. Ping, Y. Liu, L. Zhang, L. Wang, and X. Xu, "Sequence generation for multi-task scheduling in cloud manufacturing with deep reinforcement learning," Journal of manufacturing systems, vol. 67, pp. 315-337, 2023.

[5] B. Pourghebleh, V. Hayyolalam, and A. A. Anvigh, "Service discovery in the Internet of Things: review of current trends and research challenges," Wireless Networks, vol. 26, no. 7, pp. 5371-5391, 2020.

[6] B. Pourghebleh, A. A. Anvigh, A. R. Ramtin, and B. Mohammadi, "The importance of nature-inspired meta-heuristic algorithms for solving virtual machine consolidation problem in cloud environments," Cluster Computing, pp. 1-24, 2021.

[7] Q. Hu, X. Wu, and S. Dong, "A Two-Stage Multi-Objective Task Scheduling Framework Based on Invasive Tumor Growth Optimization Algorithm for Cloud Computing," Journal of Grid Computing, vol. 21, no. 2, p. 31, 2023.

[8] H. Cao and Z. Hou, "Krill Herd Algorithm for Live Virtual Machines Migration in Cloud Environments," International Journal of Advanced Computer Science and Applications, vol. 14, no. 5, 2023.

[9] W. Wang and Z. Liu, "Cloud Service Composition using Firefly Optimization Algorithm and Fuzzy Logic," International Journal of Advanced Computer Science and Applications, vol. 14, no. 3, 2023.

[10] S. Chiappa, E. Videla, V. Viana-Céspedes, P. Piñeyro, and D. A. Rossit, "Cloud manufacturing architectures: State-of-art, research challenges and platforms description," Journal of Industrial Information Integration, p. 100472, 2023.

[11] J. Delaram, M. Houshamand, F. Ashtiani, and O. Fatahi Valilai, "Development of public cloud manufacturing markets: A mechanism design approach," International Journal of Systems Science: Operations & Logistics, vol. 10, no. 1, p. 2079751, 2023.

[12] M. S. Kavre, V. K. Sunnapwar, and B. B. Gardas, "Cloud manufacturing adoption: a comprehensive review," Information Systems and e-Business Management, pp. 1-71, 2023.

[13] S. K. Srichandan, S. K. Majhi, S. Jena, K. Mishra, and R. Bhat, "A Secure and Distributed Placement for Quality of Service-Aware IoT Requests in Fog-Cloud of Things: A Novel Joint Algorithmic Approach," IEEE Access, 2024.

[14] Y. Laili, S. Lin, and D. Tang, "Multi-phase integrated scheduling of hybrid tasks in cloud manufacturing environment," Robotics and Computer-Integrated Manufacturing, vol. 61, p. 101850, 2020.

[15] Y. Hu, F. Zhu, L. Zhang, Y. Lui, and Z. Wang, "Scheduling of manufacturers based on chaos optimization algorithm in cloud manufacturing," Robotics and Computer-Integrated Manufacturing, vol. 58, pp. 13-20, 2019.

[16] X. Wang, L. Zhang, Y. Liu, C. Zhao, and K. Wang, "Solving task scheduling problems in cloud manufacturing via attention mechanism and deep reinforcement learning," Journal of Manufacturing Systems, vol. 65, pp. 452-468, 2022.

[17] Z. Chen, L. Zhang, X. Wang, and K. Wang, "Cloud–edge collaboration task scheduling in cloud manufacturing: An attention-based deep reinforcement learning approach," Computers & Industrial Engineering, vol. 177, p. 109053, 2023.

[18] A. Salmasnia and Z. Kiapasha, "Integration of sub-task scheduling and logistics in cloud manufacturing systems under setup time and different task arrival times," International Journal of Computer Integrated Manufacturing, pp. 1-24, 2023.

[19] W. Zhang, J. Xiao, W. Liu, Y. Sui, Y. Li, and S. Zhang, "Individualized requirement-driven multi-task scheduling in cloud manufacturing using an extended multifactorial evolutionary algorithm," Computers & Industrial Engineering, vol. 179, p. 109178, 2023.

[20] X. Wang, L. Zhang, Y. Liu, and C. Zhao, "Logistics-involved task scheduling in cloud manufacturing with offline deep reinforcement learning," Journal of Industrial Information Integration, vol. 34, p. 100471, 2023.

[21] R. Li, "Use linear weighted genetic algorithm to optimize the scheduling of fog computing resources," Complexity, vol. 2021, pp. 1-12, 2021.

[22] M. Hosseini Shirvani and R. Noorian Talouki, "Bi-objective scheduling algorithm for scientific workflows on cloud computing platform with makespan and monetary cost minimization approach," Complex & Intelligent Systems, vol. 8, no. 2, pp. 1085-1114, 2022.

[23] A. J. Miriam, R. Saminathan, and S. Chakaravarthi, "Non-dominated Sorting Genetic Algorithm (NSGA-III) for effective resource allocation in cloud," Evolutionary Intelligence, vol. 14, pp. 759-765, 2021.

[24] A. Khalili and S. M. Babamir, "A Pareto-based optimizer for workflow scheduling in cloud computing environment," International Journal of Information and Communication Technology Research, vol. 8, no. 1, pp. 51-59, 2016.