A Mathematical Framework for Semantic-Aware Computational Orchestration in Cyber-Physical Manufacturing Systems

Bhupender Sharma[1], Dr. Sandeep Kumar Yadav[2]

1*Research Scholar, Department of Mechanical Engineering, School of Engineering & Technology,

Vikrant University, Gwalior, (M.P.), India Email Id – sharma05031989@gmail.com

2*Ph.D. Guide, Department of Mechanical Engineering, School of Engineering & Technology,

Vikrant University, Gwalior, (M.P.), India Email Id - sandy3061987@gmail.com

Abstract

The rapid development of Industry 4.0 has increased the need for intelligent computational frameworks capable of coordinating physical manufacturing processes, digital information, and distributed computing resources. Cyber-physical manufacturing systems integrate industrial equipment, sensors, control systems, communication networks, and computational platforms to support real-time monitoring and decision-making. However, conventional computational orchestration approaches often focus primarily on resource availability, processing latency, and workload distribution, with limited consideration of the semantic relevance of tasks, physical process constraints, and manufacturing operational priorities.

This paper proposes a mathematical framework for semantic-aware computational orchestration in cyber-physical manufacturing systems. The proposed framework integrates semantic task representation, physical process constraints, computational resource allocation, and latency-aware scheduling within a unified optimisation model. Manufacturing tasks are characterised by their computational requirements, semantic importance, deadlines, data dependencies, and operational criticality. An orchestration mechanism assigns these tasks to edge, fog, or cloud computing resources while minimising a weighted objective function comprising execution latency, communication overhead, energy consumption, resource utilisation imbalance, and semantic mismatch penalties.

The framework incorporates feasibility constraints related to computational capacity, task precedence, real-time deadlines, and physical process requirements. A multi-agent coordination mechanism is proposed to support distributed decision-making across manufacturing equipment, edge nodes, and cloud infrastructure. The mathematical formulation provides a foundation for developing heuristic, mixed-integer optimisation, or reinforcement-learning-based scheduling algorithms. The framework is intended to improve computational efficiency, responsiveness, and operational consistency in intelligent manufacturing environments. Its performance must be established through simulation and comparison with conventional scheduling approaches.

Keywords: Cyber-physical manufacturing systems; semantic-aware computing; computational orchestration; mathematical optimisation; edge–cloud computing; multi-agent systems; Industry 4.0; real-time scheduling.

Introduction

Background

Modern manufacturing facilities increasingly depend on interconnected physical and digital systems to achieve flexible production, predictive maintenance, quality assurance, energy efficiency, and autonomous operational control. Industrial sensors generate continuous data streams, while programmable logic controllers, industrial gateways, edge servers, and cloud platforms process these data to support manufacturing decisions.

Cyber-physical manufacturing systems (CPMS) combine physical production processes with computational intelligence and networked communication. Within such environments, computing tasks may include detecting abnormal machine vibration, identifying product defects, predicting equipment failures, monitoring temperature and pressure, and optimising production schedules.

The distribution of these tasks across edge and cloud infrastructure introduces a complex orchestration problem. Edge computing offers lower communication latency and supports time-critical decisions, whereas cloud computing provides greater computational capacity for resource-intensive analytics. Selecting an appropriate execution location requires more than evaluating processor availability or network latency. The orchestration mechanism must also understand what each task means in the context of manufacturing operations.

For example, a machine-overheating alert associated with a critical production line may require immediate processing at the edge. In contrast, historical energy-consumption analysis may tolerate longer execution times and be processed in the cloud. Although these tasks may have similar computational requirements, their semantic significance and operational consequences differ considerably.

Semantic-aware computational orchestration addresses this challenge by incorporating task meaning, operational context, dependencies, and criticality into resource-allocation and scheduling decisions.

Problem statement

Existing resource-orchestration approaches can face several limitations in complex manufacturing environments:

1. Limited contextual understanding: Tasks may be scheduled according to resource availability without adequately considering their manufacturing significance.
2. Latency and deadline conflicts: Computationally demanding tasks may compete with safety-critical or time-sensitive operations.
3. Distributed resource constraints: Edge devices and cloud servers have different processing capacities, energy characteristics, and communication delays.
4. Interdependent manufacturing tasks: Certain computational operations must be completed before subsequent monitoring, control, or decision-making activities can proceed.
5. Physical process limitations: Scheduling decisions must respect process operating limits and the timing requirements of the equipment being monitored.
6. Dynamic operating conditions: Machine failures, changing workloads, communication congestion, and production variations can alter the optimal allocation of computational resources.

Consequently, a unified mathematical model is required to coordinate computational tasks by considering semantic relevance, physical process requirements, resource limitations, and real-time performance.

Research aim

The primary aim of this study is to formulate a mathematical framework for semantic-aware computational orchestration in cyber-physical manufacturing systems.

The specific objectives are to:

- Develop a formal representation of manufacturing tasks using semantic and computational attributes.
- Formulate a constrained optimisation model for assigning tasks to heterogeneous edge–cloud resources.
- Integrate latency, energy consumption, communication overhead, and semantic mismatch into a unified objective function.
- Establish a multi-agent orchestration architecture for distributed manufacturing environments.

- Define a simulation-based evaluation methodology for assessing scheduling performance, resource efficiency, and deadline compliance.

Literature Review

Cyber-physical systems and edge–cloud computing

Cyber-physical systems integrate physical processes with computational intelligence, communication networks, and feedback mechanisms. Their manufacturing applications require reliable information exchange between industrial equipment and computational services. Edge and edge–cloud computing have been investigated as approaches to address the latency, energy, security, privacy, and reliability challenges associated with processing industrial data. Cao et al. provide a systematic overview of these challenges and the corresponding quality-of-service optimisation approaches.

Edge computing allows data to be processed near its source, while cloud computing provides access to larger computational resources. A hybrid architecture can therefore distribute workloads according to their time sensitivity and computational requirements. However, effective workload distribution remains challenging because the resources are heterogeneous and communication conditions can change dynamically.

Computational task scheduling and resource optimisation

Task scheduling in edge–cloud environments involves assigning computational workloads to processing nodes while satisfying capacity, timing, and service-quality requirements. Common objectives include reducing execution latency, limiting energy consumption, improving resource utilisation, and ensuring deadline compliance.

Recent research has examined heuristic, metaheuristic, and artificial-intelligence-based scheduling approaches. These methods provide different trade-offs between solution quality, computational overhead, scalability, and adaptability. Task dependencies create an additional challenge because the execution of one task may depend on the completion of another.

Semantic-aware orchestration

Semantic-aware orchestration extends conventional scheduling by incorporating the meaning and operational context of computational tasks. In a manufacturing environment, semantic information may identify the equipment involved, the process stage, the type of anomaly, the required response, and the operational consequence of delayed processing.

For example, a defect-detection task may be associated with a specific product type, inspection stage, and quality threshold. A predictive-maintenance task may be associated with a machine's operating condition and maintenance priority. This contextual information can help determine which task should receive computational resources first and where it should execute.

Semantic awareness in this paper is treated as a formal task-representation and decision-making mechanism, rather than as a claim that the proposed framework already understands industrial meaning automatically. The semantic attributes must be defined through manufacturing ontologies, process metadata, engineering rules, or validated classification systems.

Proposed System Architecture

The proposed architecture consists of five interacting layers.

Layer 1 — Physical Manufacturing Environment

Sensors, actuators, PLCs, robots, CNC machines, production lines

Layer 2 — Semantic Context and Task Representation

Equipment identity, task type, process state, priority, dependencies, deadlines

Layer 3 — Mathematical Orchestration Engine

Constraint evaluation, objective optimisation, task placement, scheduling

Layer 4A — Edge/Fog

Fast response, local inference, immediate monitoring

Layer 4B — Cloud

Large-scale analytics, historical data, model training

Layer 5 — Monitoring and Feedback

Execution monitoring, deadline verification, state updates, rescheduling

Physical manufacturing layer

This layer includes production machinery, industrial sensors, actuators, programmable logic controllers, and inspection systems. It generates operational data such as temperature, vibration, pressure, motor current, production rate, and product-quality measurements.

Semantic context layer

Each computational task is represented using a structured semantic descriptor containing:

- Equipment and process identifiers.
- Task category and intended operational outcome.
- Computational workload and data volume.
- Priority and deadline.
- Physical-process constraints.
- Dependencies on other tasks.
- Data sensitivity and permitted execution locations.

The descriptors may be represented using an ontology, knowledge graph, or structured metadata schema.

Mathematical orchestration layer

The orchestration engine evaluates available computational resources and solves the task-placement problem. Its decisions are constrained by processor capacity, memory availability, network conditions, deadlines, dependencies, and task compatibility.

Mathematical Formulation

This section defines the central research contribution: a constrained task-placement and scheduling model.

Sets and indices

Let:

- \(I=\{1,2,\ldots,N\}\) denote the set of computational tasks.
- \(J=\{1,2,\ldots,M\}\) denote the set of edge, fog, and cloud nodes.
- \(K\) denote the set of task types.
- \(E\) denote the set of directed task-dependency relationships.

Each task \(i\in I\) is assigned to one eligible computational node.

Task parameters

Each task is described by the following parameters.

Symbol	Description
\(c_i\)	Required computational workload, in CPU cycles
\(d_i\)	Input data volume
\(D_i\)	Task deadline
\(p_i\)	Semantic and operational importance score
\(s_i\)	Semantic compatibility requirement
\(r_i\)	Minimum resource requirement
\(a_i\)	Task arrival time
\(e_{ij}\)	Estimated energy consumption on node \(j\)
\(L_{ij}\)	Estimated end-to-end latency on node \(j\)
\(m_{ij}\)	Semantic mismatch penalty for assigning task \(i\) to node \(j\)
\(b_{ij}\)	Binary eligibility indicator for task–node assignment

The importance score \(p_i\) may be normalised between zero and one, with higher values indicating greater operational importance. It is an input derived from engineering rules or validated assessments, not an automatically known quantity.

Computational resource parameters

Let \(f_j\) represent the effective processing rate of node \(j\), measured in CPU cycles per second. Let \(C_j\) represent its available processing capacity over the relevant scheduling interval.

The estimated processing time for task \(i\) on node \(j\) is:

\[T_{ij}^{\mathrm{proc}}=\frac{c_i}{f_j} \]

A simplified end-to-end latency model is:

\[L_{ij}= T_{ij}^{\mathrm{proc}} +T_{ij}^{\mathrm{comm}} +T_{ij}^{\mathrm{queue}} +T_{ij}^{\mathrm{setup}} \]

where the terms represent processing, communication, queueing, and setup delays, respectively.

For cloud execution, communication latency may include transmitting data to the cloud and receiving the required result. For edge execution, the corresponding transmission distance may be shorter, although edge congestion can still cause delays.

4.4 Decision variables

Define the binary assignment variable:

$$x_{ij} = \begin{cases} 1, & \text{if task } i \text{ is assigned to node } j, \\ 0, & \text{otherwise.} \end{cases}$$

Let C_i denote the predicted completion time of task i, and let z_i denote its deadline-violation indicator.

The task-placement decision is represented by x_{ij}. The completion-time and scheduling variables are used to enforce temporal feasibility.

Research Methodology

This study is designed as a mathematical modelling and simulation-based investigation. The framework should be implemented and tested before any quantitative performance claims are made.

Research design

The research will proceed through five stages.

1. System modelling: Define manufacturing tasks, physical-process context, edge and cloud nodes, and task dependencies.
2. Mathematical formulation: Implement the objective function, task-placement variables, capacity constraints, deadline constraints, and semantic eligibility rules.
3. Algorithm development: Develop a MILP-based reference solver and a scalable orchestration heuristic.
4. Simulation: Generate representative manufacturing workloads and evaluate the framework under varying network conditions, task priorities, and computing capacities.
5. Comparative analysis: Compare the proposed method with conventional schedulers using repeated experiments and statistical analysis.

Simulation environment

A prototype can be developed using Python, with an optimisation library such as Pyomo or PuLP and an appropriate MILP solver. Discrete-event simulation can be used to model task arrivals, queues, network delays, and execution completion.

The simulation should represent a manufacturing environment with multiple machines, sensors, edge nodes, and a cloud server.

Example workload categories include:

- Machine-condition monitoring.
- Defect detection and quality inspection.
- Predictive-maintenance analytics.
- Production scheduling and optimisation.
- Energy-consumption analysis.
- Process anomaly detection.

These workloads should be assigned realistic task sizes, deadlines, semantic priorities, and execution eligibility based on measured industrial data where available. Synthetic data may be used during initial development, provided its assumptions are clearly documented.

Experimental scenarios

Scenario	Experimental condition	Research purpose
S1	Low workload and stable network	Evaluate baseline performance
S2	High task arrival rate	Assess congestion and scalability
S3	Variable network latency	Test sensitivity to communication changes
S4	Mixed critical and non-critical tasks	Evaluate semantic prioritisation
S5	Reduced edge capacity	Assess cloud offloading and resource allocation
S6	Node failure or temporary disconnection	Evaluate rescheduling and recovery
S7	Tight task deadlines	Measure deadline compliance and infeasibility

Each scenario should be repeated with controlled random seeds where stochastic arrivals or network conditions are used. The simulation configuration, task distributions, solver settings, and hardware assumptions should be documented for reproducibility.

Performance Evaluation Metrics

The framework should be evaluated using measurable indicators rather than relying on qualitative claims.

Average task latency

For \(N\) completed tasks, average latency is:

\[\overline{L}=\frac{1}{N}\sum_{i=1}^{N}L_i \]

Lower values indicate faster end-to-end task execution.

Deadline-miss ratio

Let \(\mathcal{I}\) denote the set of tasks included in the evaluation. The deadline-miss ratio is:

\[R_{\mathrm{miss}}= \frac{\sum_{i\in\mathcal I} \mathbf{1}(C_i>D_i)} {|\mathcal I|} \times100\% \]

This metric should also be reported separately for critical and non-critical tasks.

Total energy consumption

\[E_{\mathrm{total}}= \sum_{i\in I}\sum_{j\in J}e_{ij}x_{ij} \]

The energy model must clarify whether it includes processing, networking, idle power, and cloud infrastructure overhead.

Results and Discussion: Proposed Evaluation Framework

Important: No simulation data have been supplied or generated for this manuscript. Consequently, this section defines how results should be reported; it does not present fabricated performance figures.

Expected analytical behaviour

The mathematical formulation supports several testable hypotheses.

- H1: Incorporating semantic priorities will reduce the relative frequency of deadline misses among high-priority tasks when the system has sufficient feasible capacity.
- H2: Edge–cloud task placement will reduce latency for some time-sensitive tasks compared with a cloud-only execution policy, provided edge capacity and communication conditions are favourable.
- H3: A multi-objective scheduler will achieve different latency–energy trade-offs from a scheduler that optimises latency alone.
- H4: Explicit task-dependency and compatibility constraints will reduce infeasible assignments compared with unconstrained resource allocation.
- H5: The computational cost of solving the optimisation problem will increase as task count and scheduling complexity grow, motivating scalable heuristics for larger systems.

These hypotheses are not guaranteed outcomes. For example, semantic prioritisation may improve critical-task performance while increasing the waiting time of lower-priority tasks. Similarly, cloud execution may outperform edge execution when the edge node is heavily congested.

Ablation study

An ablation study can establish whether each component contributes to the overall performance. Run the proposed method under the following configurations:

- Full semantic-aware model.
- Semantic mismatch term removed.
- Deadline-priority term removed.
- Energy term removed.
- Task-dependency constraints removed.
- Dynamic rescheduling disabled.

The difference between each configuration and the complete model can indicate the contribution of the corresponding component. Removing a hard safety or physical feasibility constraint should not be treated as an acceptable operational alternative; such experiments, if conducted at all, should remain confined to offline simulation.

Limitations and Future Research

The proposed framework has several limitations.

First, semantic importance and compatibility scores may be difficult to define consistently across different manufacturing environments. Domain-specific ontologies and expert-validated rules are required.

Second, the simplified mathematical model assumes that task costs and node capacities can be estimated. In practice, workload execution time, network latency, and available resources are uncertain and may change during execution.

Third, a weighted objective function requires careful calibration. Poorly chosen weights can produce undesirable trade-offs, such as excessive energy savings at the expense of responsiveness. Hard operational requirements should therefore remain explicit constraints.

Fourth, large-scale task placement is computationally challenging. Future research can investigate decomposition, rolling-horizon optimisation, multi-agent negotiation, reinforcement learning, and robust or stochastic optimisation.

Fifth, the framework does not by itself establish closed-loop physical safety. Integrating physical process dynamics and validated control models would be necessary for stronger guarantees concerning real-time process behaviour.

Future work should focus on a reproducible simulation prototype, calibration with industrial data, formal analysis of deadline feasibility, and comparative experiments with existing schedulers.

Conclusion

This paper proposes a mathematical framework for semantic-aware computational orchestration in cyber-physical manufacturing systems. The framework integrates manufacturing task semantics, heterogeneous computing resources, latency, energy consumption, deadline requirements, and task dependencies into a constrained optimisation model.

The central contribution is the explicit inclusion of semantic importance and compatibility in task-placement decisions alongside conventional computational objectives. A layered architecture connects physical manufacturing equipment, semantic task representation, a mathematical orchestration engine, edge–cloud execution, and feedback monitoring.

A multi-agent implementation is outlined to support distributed resource coordination, while a simulation-based methodology is proposed to compare the framework with conventional scheduling methods. The framework provides a basis for investigating how contextual information can improve the allocation of computational resources in intelligent manufacturing.

Its practical effectiveness remains to be demonstrated through implementation, experimental evaluation, and comparison against established scheduling approaches. Such validation would help determine whether semantic-aware orchestration provides measurable improvements in critical-task responsiveness, energy efficiency, and manufacturing system performance.

References

1. 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. https://doi.org/10.1109/TII.2021.3073066

2. Wang, L., Wu, C., & Fan, W. (2021). A survey of edge computing resource allocation and task scheduling optimization. Journal of System Simulation, 33(3), 509–520. Publisher article. china-simulation.com

3. Sahni, Y., Cao, J., Yang, L., & Wang, S. (2022). Distributed resource scheduling in edge computing: Problems, solutions, and opportunities. Computer Networks, 219, Article 109430. https://doi.org/10.1016/j.comnet.2022.109430

4. A state-of-the-art review of task scheduling for edge computing: A delay-sensitive application perspective. (2023). Electronics, 12(12), 2599. Publisher article.

5. Singh, R. K. B., & Reddy, K. H. K. (2025). Edge-AI empowered cyber-physical systems: A comprehensive review on performance analysis. Computer Science Review, 58, Article 100769. https://doi.org/10.1016/j.cosrev.2025.100769

6. Nagabushnam, G., Choi, Y., & Kim, K. H. (2025). Energizing the fog: A systematic survey on task scheduling strategies for energy optimization. Cluster Computing, 28(6), Article 375. https://doi.org/10.1007/s10586-025-05200-8