

RESEARCH ARTICLE

Energy- and Latency-Aware Computation Offloading in Edge–Fog–Cloud Architectures for Industry 4.0

Mohammad Javad Esfahani^{1,*}¹ Faculty member, Academy of Social, Scientific and Applied Research (ASSAAR), London, UK*E-mail (corresponding author): mj.esfahani@assaar.co.uk**ARTICLE INFO****Article history**

Received 2025-07-08

Revised 2025-08-10

Accepted 2025-08-29

Keywords

Industry 4.0

Edge computing

Computation offloading

Metaheuristic optimization

Industrial Internet of Things

ABSTRACT

Industry 4.0 transforms manufacturing through cyber-physical systems, the Industrial Internet of Things, digital twins, and data-driven intelligence, yet the volume of latency-sensitive data these technologies generate strains conventional cloud-only computing. Edge and fog computing have emerged as the central enabling paradigm, distributing computation across a device–fog–cloud continuum. This paper surveys the enabling technologies of Industry 4.0 with an emphasis on edge–cloud computing and formulates the resulting computation-offloading problem as a constrained, load-dependent optimization model that jointly minimizes task latency and device energy under deadline and capacity constraints. A genetic algorithm and a particle swarm optimizer are developed and benchmarked against five baseline strategies on synthetic industrial workloads. Across one hundred tasks averaged over ten instances, the genetic algorithm reduced the weighted objective by 34.1 percent relative to a greedy heuristic and by 82.6 percent relative to cloud-only execution, while lowering deadline violations to 1.6 percent. The latency–energy trade-off, convergence, and scalability are analyzed, and managerial implications for industrial deployment are discussed.

1. Introduction

The fourth industrial revolution, commonly termed Industry 4.0, denotes the integration of physical production systems with digital networks, sensing, and computational intelligence to create autonomous, self-optimizing factories. Its diffusion is no longer speculative. The global Industry 4.0 market was valued at roughly 130 billion United States dollars in 2022 and is projected to expand at a compound annual growth rate exceeding 20 percent through the end of the decade, while the number of connected Industrial Internet of Things devices is expected to surpass 36 billion within the same horizon. Manufacturing already accounts for the largest share of global IoT spending, and surveys of industrial firms repeatedly find that more than two-thirds have launched or scaled digital-transformation initiatives. These figures signal a structural shift in how industrial value is created, monitored, and controlled.

This shift produces data at a scale and velocity that challenge the computational assumptions on which early cyber-physical systems were built. A single mid-sized smart factory can generate several terabytes of sensor, vision, and machine-log data per day, and a growing fraction of this data feeds applications with hard real-

time requirements such as closed-loop motion control, machine-vision quality inspection, and predictive maintenance. Transmitting all of this data to a distant cloud for processing incurs propagation and queueing delays that frequently exceed the tens-of-milliseconds budgets of such applications, consumes scarce wide-area bandwidth, and raises confidentiality concerns for proprietary process data. The mismatch between centralized cloud computing and decentralized industrial demand is the practical problem that motivates the present study.

Edge and fog computing address this mismatch by relocating computation closer to where data originate. Rather than routing every task to the cloud, an edge–fog–cloud architecture distributes workloads across a continuum: resource-constrained field devices, intermediate fog or edge servers situated within the plant, and elastic but distant cloud data centers. Each tier offers a different balance of computational capacity, communication latency, and energy cost, so the decision of where to execute each task becomes a consequential optimization problem. Poor placement either overwhelms low-power devices, saturates edge servers, or pushes latency-critical work to the cloud where deadlines are missed. The quality of the offloading policy therefore directly determines whether the enabling technologies of Industry 4.0 deliver their promised responsiveness and efficiency.

This paper makes three contributions. First, it provides a focused survey of the enabling technologies of Industry 4.0, organized around the edge–cloud computing continuum and its relationship to the Industrial Internet of Things, cyber-physical systems, digital twins, and data-driven intelligence. Second, it formulates energy- and latency-aware computation offloading as a constrained optimization problem with a realistic load-dependent execution model in which server performance degrades under contention. Third, it develops and evaluates two metaheuristic solvers, a genetic algorithm and a particle swarm optimizer, against a spectrum of baseline strategies, and analyzes the latency–energy trade-off, convergence behavior, and scalability of the proposed framework. The remainder of the paper reviews the related literature, details the system model and solution methods, presents and discusses numerical results, and concludes with limitations, managerial insights, and directions for future work.

2. Literature Review

Research on the enabling technologies of Industry 4.0 has matured from conceptual roadmaps into a substantial body of work that treats edge and cloud computing as foundational rather than peripheral. Early surveys positioned the Industrial Internet of Things as the connective tissue of the smart factory and identified the migration of computation toward the network edge as the decisive architectural response to latency, bandwidth, and privacy pressures (Qiu et al., 2020). Subsequent comprehensive reviews of enabling technologies broadened the frame, cataloguing edge and cloud computing alongside digital twins, the Internet of Everything, collaborative robotics, and next-generation networking as the pillars on which human-centric and resilient industrial systems are being constructed (Maddikunta et al., 2022; Karnik et al., 2022). A recurring theme across this literature is that no single tier suffices: the value lies in orchestrating a continuum, a position articulated most directly in studies of the convergence of edge, fog, and cloud resources within data-intensive industrial settings (Firouzi et al., 2022).

Within the computing layer itself, a distinct stream examines edge intelligence and the architectural patterns that support it. Investigations of deep learning executed at the industrial edge demonstrated that moving inference close to data sources can satisfy real-time constraints that cloud execution cannot, while also exposing the resource limits of edge hardware (Liang et al., 2020). Complementary work on fog computing contributed concrete platform designs and detailed analyses of latency, energy, and security at the industrial fog layer, establishing fog as an engineering reality rather than an abstraction (Pop et al., 2021; Caiza et al., 2020). Broader surveys of the edge-enabled smart city and of the IoT–edge–cloud continuum synthesized these threads into reference architectures and catalogued the open issues of orchestration, heterogeneity, and interoperability that continue to constrain deployment (Khan et al., 2020; Gkonis et al., 2023). The security dimension of this computing fabric has attracted particular attention, with reviews mapping the attack surface of the Industrial Internet of Things and the role of edge-resident, artificial-intelligence-based defenses (Alotaibi, 2023), and with healthcare-oriented analyses illustrating how the same IoT, big-data, and cloud stack underpins sector-specific variants of the paradigm (Aceto et al., 2020).

A parallel literature treats the cyber-physical foundations and digital representations that the computing continuum serves. Reviews of cyber-physical systems architectures for industrial applications established the layered models that connect physical processes to computational decision-making and remain widely cited as organizing frameworks (Pivoto et al., 2021), and large-scale categorizations have since structured the sprawling body of cyber-physical-systems research into navigable taxonomies (Oks et al., 2024). The digital twin, understood as a synchronized virtual counterpart of a physical asset, has been examined both in terms of its enabling tools and reference models and in terms of its concrete manufacturing applications, with successive reviews tracing its evolution from concept to deployed system and its dependence on edge and cloud infrastructure for real-time fidelity (Qi et al., 2021; Lu et al., 2020; Leng et al., 2021; Javaid et al., 2023). Recent surveys explicitly couple the digital twin to the Industrial Internet of Things, underscoring that twin synchronization is itself a demanding distributed-computing workload (Xu et al., 2023).

The intelligence that operates on industrial data constitutes a further pillar. Reviews of the interplay between artificial intelligence and big data in Industry 4.0 surveyed the techniques, applications, and open challenges of data-driven manufacturing (Jagatheesaperumal et al., 2022), while analyses of machine learning in manufacturing emphasized the path from isolated models toward integrated, knowledge-rich systems (Chen et al., 2023). The intersection of artificial intelligence with industrial cyber-security has been examined as a domain of both heightened threat and new defensive opportunity (Bécue et al., 2021). Distributed and privacy-preserving learning has emerged as a natural complement to edge architectures, with surveys of federated learning in the Industrial Internet of Things showing how model training can be pushed to the edge without centralizing sensitive data (Boobalan et al., 2022). Blockchain has been reviewed as an enabler of trust and traceability across industrial processes (Javaid et al., 2021), and dedicated platforms for big-data analytics in advanced manufacturing have been proposed and evaluated (Bonnard et al., 2021). Studies of implementation more broadly have weighed the managerial and technological barriers to adoption, cautioning that organizational readiness often lags technical capability (Bajić et al., 2021; Rikalovic et al., 2022), and work on manufacturing sustainability has linked these technologies to environmental as well as

economic objectives (Jamwal et al., 2021). Finally, the networking substrate that ties the continuum together has been surveyed through the lenses of time-sensitive networking and fifth-generation connectivity and of intelligent network slicing for industrial traffic (John et al., 2024; Wu et al., 2022).

Taken together, this body of work establishes the enabling technologies of Industry 4.0 and identifies edge–cloud computing as their connective and performance-critical layer. What remains comparatively underdeveloped is a tractable, reproducible optimization treatment of where computation should execute across the continuum when servers experience realistic contention and tasks carry hard deadlines. Much of the surveyed literature characterizes the problem qualitatively or evaluates placement heuristics under load-independent assumptions. The present study addresses this gap by formulating computation offloading as a constrained, load-dependent optimization problem and by quantifying the benefit that population-based metaheuristics deliver over conventional heuristics and naive placement strategies.

3. Methodology

3.1 System model

The system under study is a three-tier edge–fog–cloud architecture serving a set of computational tasks generated by Industrial Internet of Things devices on a factory floor. The three tiers are the device tier, comprising resource-constrained field devices such as programmable controllers, vision cameras, and sensor gateways; the fog tier, comprising edge servers located within the plant; and the cloud tier, comprising elastic but distant data-center resources. Figure 1 depicts the architecture and the offloading decision associated with each task.

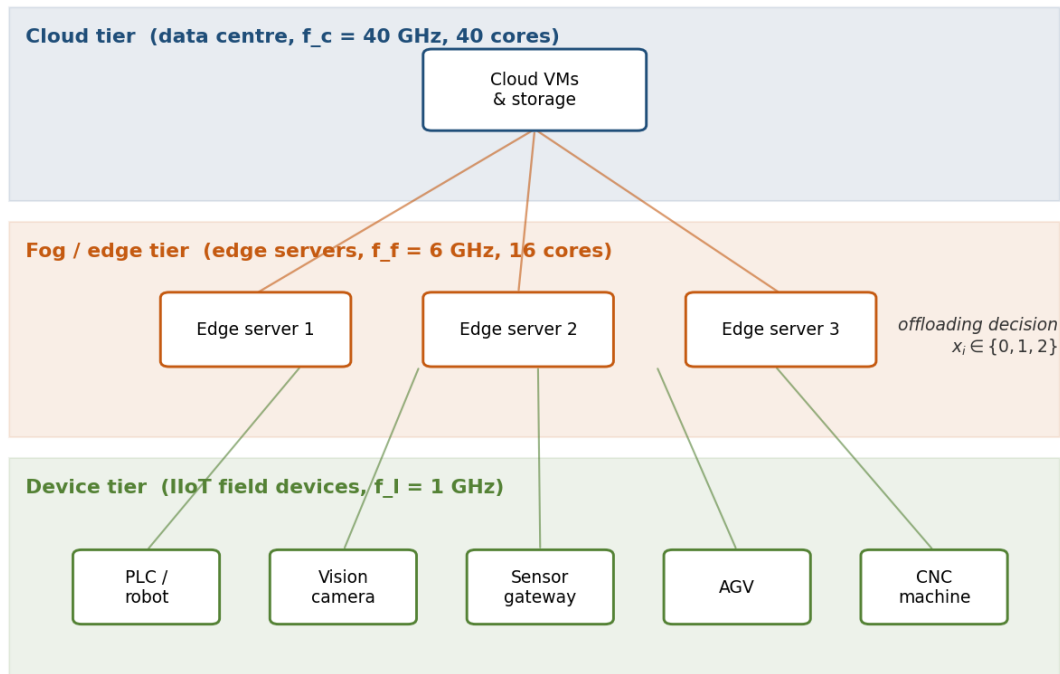


Figure 1. Three-tier edge–fog–cloud architecture for Industry 4.0, showing the device, fog, and cloud tiers and the per-task offloading decision.

Let $\mathcal{N} = \{1, 2, \dots, N\}$ denote the set of tasks. Each task i is characterized by an input data size D_i in bits and a computational workload C_i in CPU cycles, and is subject to a soft real-time deadline $T_i^{\max}$ in seconds. The

offloading decision for task i is encoded by the variable $x_i \in \{0,1,2\}$, where $x_i = 0$ denotes local execution on the originating device, $x_i = 1$ denotes execution on the fog tier, and $x_i = 2$ denotes execution on the cloud tier. The complete solution is the assignment vector $\mathbf{x} = (x_1, x_2, \dots, x_N)$.

The three tiers differ in processing capacity, communication characteristics, and energy cost. The device, fog, and cloud tiers possess CPU clock frequencies f_ℓ , f_f , and f_c respectively, with $f_\ell < f_f < f_c$. Offloading to the fog tier uses a low-power short-range wireless link of rate R_f and device transmit power P_f , whereas offloading to the cloud uses a higher-power wide-area link of rate R_c and transmit power P_c , with $P_f < P_c$. This distinction creates a genuine trade-off: the cloud offers the fastest computation but the costliest communication, while the fog offers cheaper communication at lower computational speed, and local execution avoids communication entirely at the price of slow, energy-intensive on-device processing.

3.2 Latency and energy models

The latency experienced by a task comprises a transmission component and a computation component. For local execution there is no transmission, so the latency is determined solely by the device clock.

$$T_i^\ell = \frac{C_i}{f_\ell} \quad (1)$$

For execution on an offloaded tier $m \in \{f, c\}$, the task data must first be transmitted over the corresponding link, after which the task is computed on a server whose effective speed depends on contention. The transmission time is

$$T_{i,m}^{\text{tx}} = \frac{D_i}{R_m} \quad (2)$$

A realistic feature of shared industrial servers is that their capacity is divided among the tasks that occupy them. Each offloaded tier is modeled as a pool of parallel processing cores, K_f cores at the fog tier and K_c cores at the cloud tier. When the number of tasks assigned to a tier exceeds its core count, the per-task computational throughput degrades proportionally. Defining the load-dependent sharing factor for tier m under assignment $\mathbf{x}$ as

$$s_m(\mathbf{x}) = \max\left(1, \frac{n_m(\mathbf{x})}{K_m}\right), \quad n_m(\mathbf{x}) = \sum_{i=1}^N \mathbb{1}\{x_i = m\} \quad (3)$$

the total latency of a task offloaded to tier m is the sum of its transmission time, its contention-adjusted computation time, and a fixed propagation and queueing delay δ_m ,

$$T_{i,m} = \frac{D_i}{R_m} + \frac{C_i s_m(\mathbf{x})}{f_m} + \delta_m \quad (4)$$

This coupling, whereby the latency of one task depends on how many other tasks share its tier, is what renders the assignment problem combinatorially hard and motivates the use of metaheuristic search.

The device energy associated with a task depends on its placement. Local execution consumes dynamic CPU energy governed by the effective switched-capacitance coefficient κ ,

$$E_i^\ell = \kappa C_i f_\ell^2 \quad (5)$$

Offloading consumes radio energy proportional to the transmission time and the tier-specific transmit power,

$$E_{i,m} = P_m \frac{D_i}{R_m}, \quad m \in \{f, c\} \quad (6)$$

Cloud execution incurs no device computation energy beyond transmission, which is the basis of the energy advantage of offloading for compute-heavy tasks.

3.3 Optimization problem

The objective is to choose the assignment vector $\mathbf{x}$ that jointly minimizes total task latency and total device energy, subject to deadlines and tier capacities. Because latency and energy are measured in different units, each is normalized by a reference value: T_{ref} , the latency incurred if all tasks executed locally, and E_{ref} , the corresponding all-local energy. A weight $w \in [0,1]$ expresses the relative priority of latency over energy, allowing a decision-maker to tune the policy toward responsiveness or efficiency. The weighted objective is

$$Z(\mathbf{x}) = w \frac{\sum_{i=1}^N T_{i,x_i}}{T_{\text{ref}}} + (1-w) \frac{\sum_{i=1}^N E_{i,x_i}}{E_{\text{ref}}} + \frac{\rho}{N} (V(\mathbf{x}) + O(\mathbf{x})) \quad (7)$$

where $V(\mathbf{x})$ counts deadline violations, $O(\mathbf{x})$ counts the number of task assignments exceeding tier capacities, and ρ is a penalty coefficient. The deadline-violation count is

$$V(\mathbf{x}) = \sum_{i=1}^N \mathbb{1}\{T_{i,x_i} > T_i^{\max}\} \quad (8)$$

and the capacity-overflow term penalizes assignments that exceed the admissible task counts $\bar{n}_f$ and $\bar{n}_c$ on the fog and cloud tiers,

$$O(\mathbf{x}) = \max(0, n_f(\mathbf{x}) - \bar{n}_f) + \max(0, n_c(\mathbf{x}) - \bar{n}_c) \quad (9)$$

The optimization problem is therefore to minimize $Z(\mathbf{x})$ over all assignment vectors $\mathbf{x} \in \{0,1,2\}^N$. The search space contains 3^N candidate solutions, and the load-dependent coupling prevents the problem from decomposing into independent per-task decisions, so exhaustive search is infeasible for realistic task counts.

3.4 Solution methods

Two population-based metaheuristics are developed to solve the offloading problem, together with five baseline strategies for comparison. The genetic algorithm represents each candidate solution as an integer-coded chromosome of length N whose genes take values in $\{0,1,2\}$. The population is initialized at random, with one individual seeded from the greedy heuristic to accelerate early progress. Each generation applies elitism to preserve the best individuals, tournament selection to choose parents, uniform crossover with probability p_c to recombine assignments, and gene-wise mutation with probability p_m to maintain diversity. The procedure iterates for a fixed number of generations and returns the best assignment found. Figure 2 presents the flowchart of the algorithm, and the pseudocode is given below.

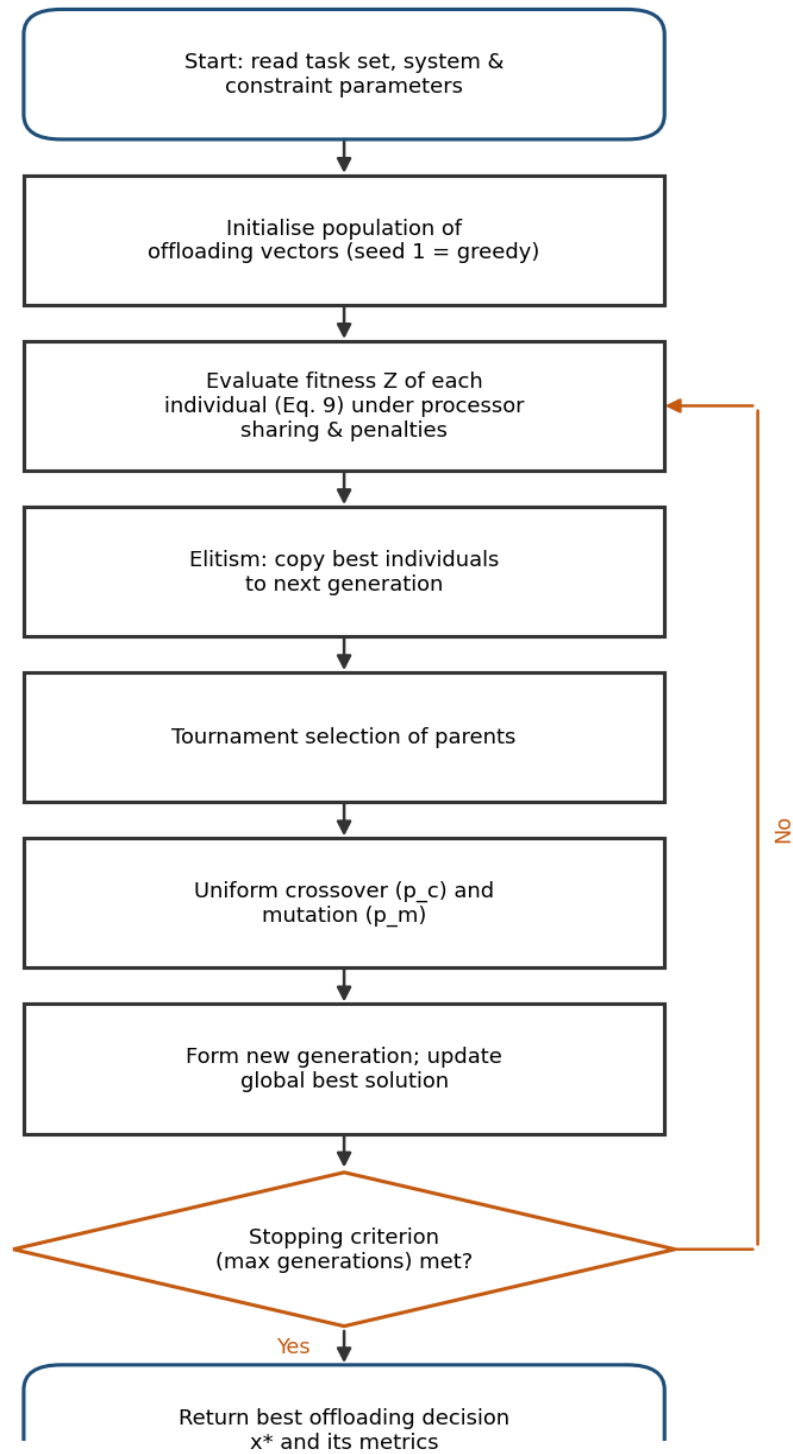


Figure 2. Flowchart of the genetic-algorithm offloading optimizer.

Algorithm 1: Genetic algorithm for computation offloading

Input: task set, system parameters, weight w , capacities

Output: best offloading vector x^* and its objective value

1: initialize population P of integer vectors in $\{0,1,2\}^N$

2: seed $P[0]$ with the greedy heuristic solution

3: evaluate Z for every individual in P using Eq. (9)

4: record the best individual x^*

5: for generation = 1 to G do

```

6:   copy the elite individuals into the new population P'
7:   while |P'| < population size do
8:       select parents p1, p2 by binary tournament
9:       apply uniform crossover with probability p_c
10:      apply gene-wise mutation with probability p_m
11:      add offspring to P'
12:  P <- P'
13:  evaluate Z for every individual in P
14:  update x* if a better solution is found
15: return x*

```

The particle swarm optimizer represents each candidate as a continuous position vector in $[0,3)^N$ that is decoded to a discrete assignment by flooring and clipping each coordinate. Each particle maintains a velocity updated from its own best-known position and the global best, with inertia weight w_v and acceleration coefficients c_1 and c_2 . Velocities are bounded to prevent divergence, and positions are clipped to the feasible range. The swarm iterates for the same number of evaluations as the genetic algorithm to permit a fair comparison.

Five baselines contextualize the metaheuristic results. The all-local, all-fog, and all-cloud strategies assign every task to a single tier and reveal the cost of ignoring heterogeneity. The random strategy assigns tasks uniformly at random. The greedy heuristic assigns each task to the tier of lowest individual cost under a myopic, single-task latency estimate that ignores contention, then repairs capacity overflows by relocating the least-penalized tasks; because it neglects the load-dependent coupling, it serves as a strong but contention-blind benchmark.

3.5 Experimental design

Tasks were generated synthetically to reflect industrial workloads. Input data sizes were drawn uniformly between 0.2 and 4.0 megabytes, computational intensity between 100 and 500 cycles per bit, and deadlines uniformly between 1.0 and 5.0 seconds. The device, fog, and cloud clock frequencies were set to 1, 6, and 40 gigahertz respectively; the fog and cloud links to 50 megabits per second; the fog and cloud transmit powers to 0.2 and 0.7 watts; the fog and cloud core counts to 16 and 40; and the propagation delays to 2 and 20 milliseconds. Tier capacities were set to 55 percent and 50 percent of the task count for fog and cloud respectively. Table 1 summarizes the descriptive statistics of a representative one-hundred-task instance. Four experiments were conducted: a main comparison of all methods at one hundred tasks averaged over ten random instances; an analysis of convergence; a sweep of the latency–energy weight from zero to one; and a scalability study across task counts from twenty to four hundred. Each metaheuristic used a population or swarm of sixty over one hundred twenty generations.

Table 1. Descriptive statistics of a representative one-hundred-task workload instance.

Task attribute	Minimum	Mean	Maximum
Input data size (MB)	0.26	2.29	3.99
Computational workload (giga-cycles)	0.29	5.86	14.58
Deadline (s)	1.04	3.00	4.88

4. Numerical Results

4.1 Comparison of offloading strategies

The main experiment compared all seven strategies on one-hundred-task instances, averaging each metric over ten independent random instances. Table 2 reports the mean objective value, total latency, total device energy, deadline violations, and runtime for each method, and Figure 4 visualizes the objective values and deadline violations.

Table 2. Performance of offloading strategies at one hundred tasks with equal latency–energy weighting, averaged over ten instances. Lower objective values are better.

Method	Objective Z	Total latency (s)	Total energy (J)	Deadline violations	Runtime (s)
All-Local	2.352 ± 0.093	501.2	501.20	67.6	0.000
All-Fog	2.920 ± 0.072	556.1	6.77	72.9	0.000
All-Cloud	1.115 ± 0.025	67.2	23.69	1.2	0.000
Random	1.022 ± 0.141	245.1	166.79	30.5	0.000
Greedy	0.294 ± 0.047	111.0	17.74	8.3	0.000
PSO	0.347 ± 0.039	140.1	80.90	6.2	0.820
GA	0.194 ± 0.046	119.5	41.00	1.6	0.886

The genetic algorithm achieved the lowest objective value of 0.194, outperforming the particle swarm optimizer at 0.347 and the greedy heuristic at 0.294. Relative to the greedy heuristic, the genetic algorithm reduced the objective by 34.1 percent, and relative to cloud-only execution it reduced the objective by 82.6 percent. The single-tier strategies performed poorly: all-local and all-fog placement each violated more than two-thirds of deadlines because contention and slow local processing overwhelmed the timing budgets, while all-cloud placement kept deadlines but at substantial communication energy. The genetic algorithm also achieved the lowest deadline-violation count among the contention-aware methods, missing only 1.6 of one hundred deadlines on average, compared with 8.3 for the greedy heuristic. The runtime of both metaheuristics remained below one second, which is acceptable for the planning horizons over which offloading policies are typically recomputed.

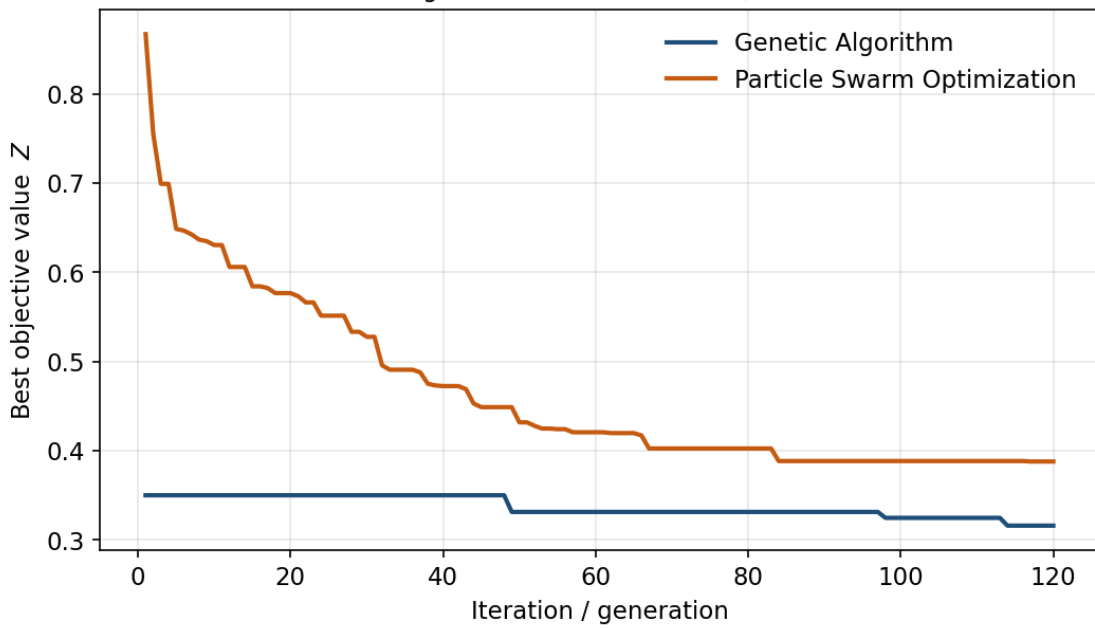


Figure 3. Convergence of the genetic algorithm and the particle swarm optimizer on a representative instance.

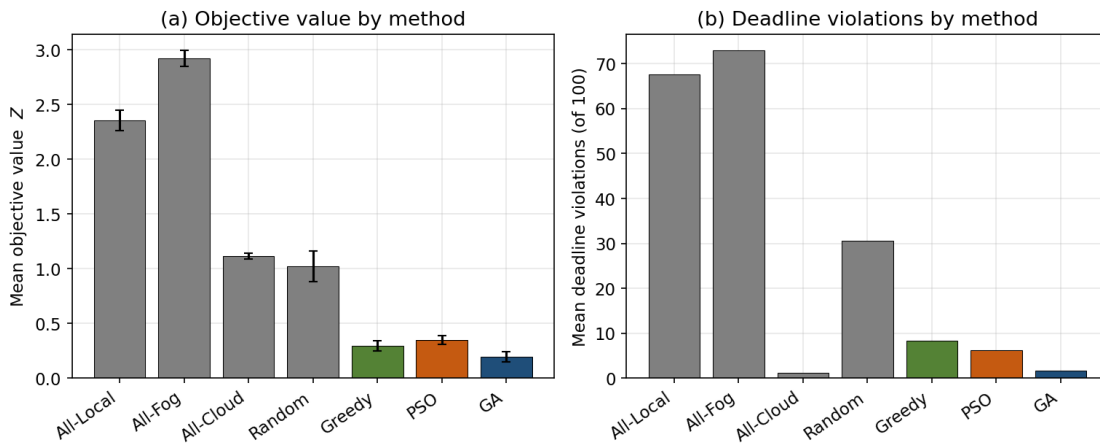


Figure 4. Comparison of offloading strategies by (a) objective value and (b) deadline violations.

4.2 Convergence behavior

Figure 3 traces the best objective value found by each metaheuristic over successive iterations on a representative instance. The genetic algorithm began near 0.350 and converged to approximately 0.316, with most of the improvement realized within the first forty generations before settling into refinement. The particle swarm optimizer began from a substantially worse position near 0.867 and improved steadily to approximately 0.388, reflecting its broader initial exploration but slower convergence to the high-quality region of the search space. The warm start of the genetic algorithm from the greedy solution explains its favorable initial position, while its elitism ensures monotonic improvement of the incumbent solution.

4.3 Latency–energy trade-off

The weight w governs the priority assigned to latency relative to energy and traces out the achievable trade-off frontier. Table 3 and Figure 5 report total latency and total device energy as the weight varies from zero, pure energy minimization, to one, pure latency minimization. As the weight increased, total latency fell from 116.1 seconds to 103.5 seconds, a reduction of 10.8 percent, while total device energy rose from 34.4 joules

to 42.8 joules, an increase of 24.5 percent. The monotonic direction of this exchange confirms that the optimizer responds correctly to the decision-maker's stated priority, shifting tasks from the energy-cheap but slower fog tier toward the faster but communication-costly cloud tier as latency is weighted more heavily. Deadline violations remained low across the entire sweep, indicating that the frontier is composed of feasible operating points rather than achieved by sacrificing timeliness.

Table 3. Latency–energy trade-off achieved by the genetic algorithm as the latency weight varies, at one hundred tasks.

Latency weight w	Total latency (s)	Total energy (J)	Deadline violations
0.0	116.1	34.37	1.0
0.2	110.5	31.73	0.8
0.4	109.4	37.54	0.6
0.5	104.2	34.45	0.6
0.6	104.5	37.00	0.4
0.8	105.1	41.65	0.4
1.0	103.5	42.79	0.6

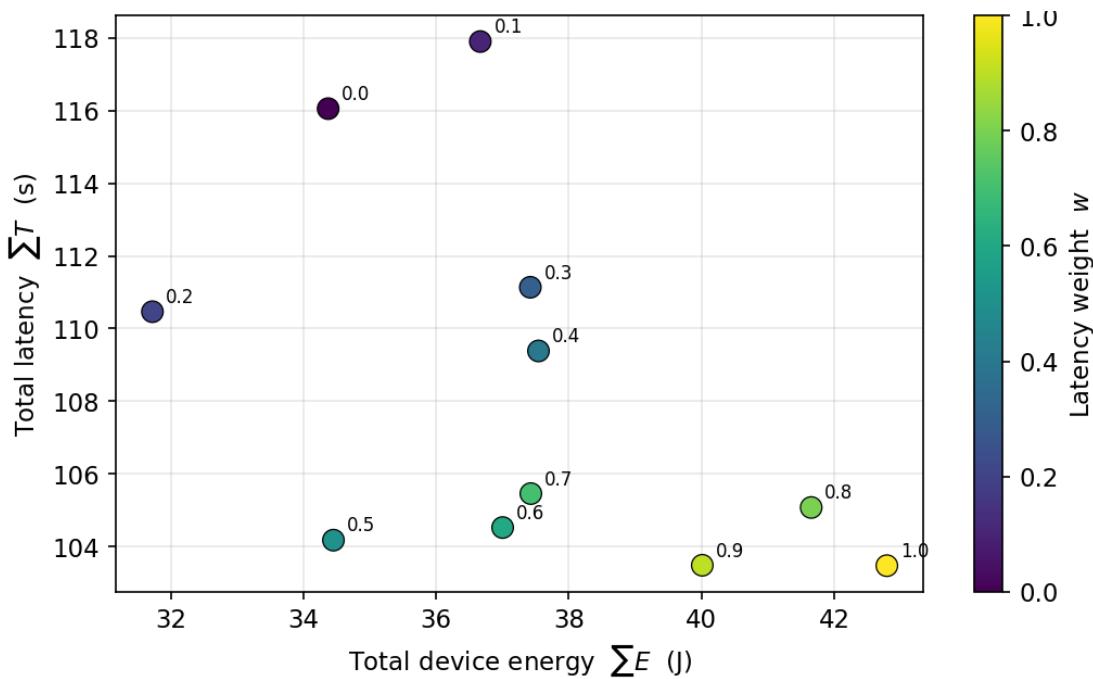


Figure 5. Latency–energy trade-off frontier obtained by varying the latency weight from zero to one.

4.4 Scalability

Table 4 and Figure 6 examine how the genetic algorithm scales as the number of tasks grows from twenty to four hundred, reporting its objective value, the greedy objective for comparison, the percentage improvement of the genetic algorithm over the greedy heuristic, and the genetic-algorithm runtime. At twenty tasks the two methods were indistinguishable, because the problem is small enough that the greedy heuristic finds a near-optimal placement despite ignoring contention. As the task count rose, however, contention intensified and the advantage of contention-aware search widened, reaching 38.3 percent at one hundred tasks and remaining near 28 percent at four hundred tasks. The runtime grew gracefully from 0.40 seconds at twenty

tasks to 2.66 seconds at four hundred tasks, a sub-quadratic increase that suggests the approach remains practical at the scale of a moderately sized production cell.

Table 4. Scalability of the genetic algorithm relative to the greedy heuristic across task counts.

Tasks	GA objective	Greedy objective	Improvement (%)	GA runtime (s)
20	0.081	0.081	0.0	0.397
50	0.093	0.108	13.7	0.579
100	0.169	0.274	38.3	0.870
200	0.493	0.702	29.8	1.482
400	0.888	1.230	27.8	2.657

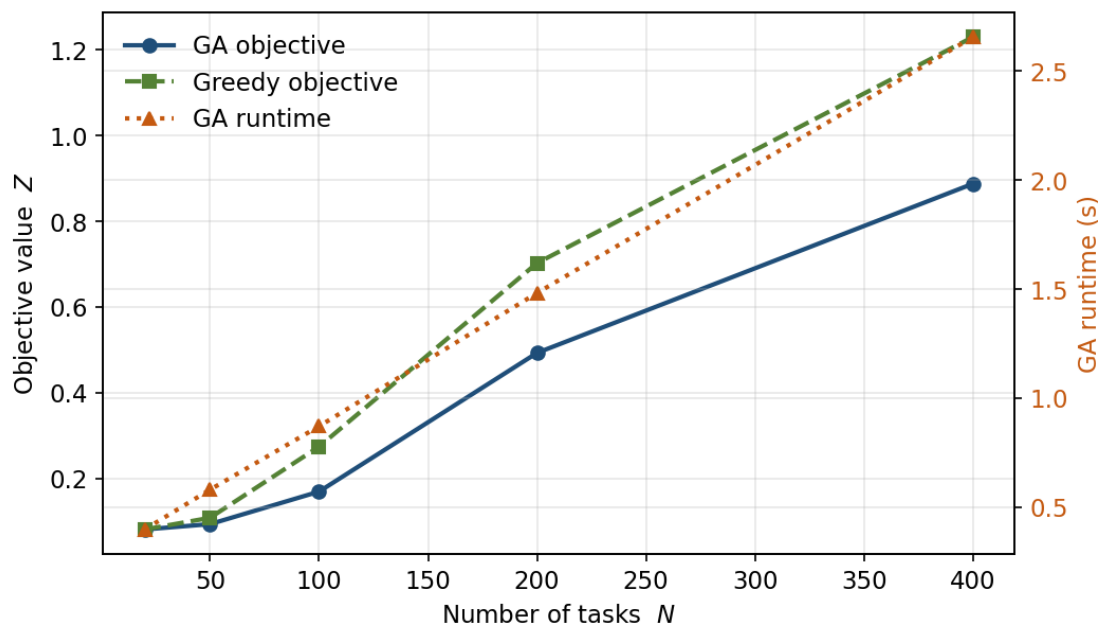


Figure 6. Scalability of the genetic-algorithm offloading optimizer in objective value and runtime.

5. Discussion

The results substantiate the central proposition of the survey, namely that edge–cloud computing functions as the performance-critical layer of the Industry 4.0 technology stack and that the manner in which computation is distributed across the continuum materially determines system performance. The naive single-tier strategies, which correspond to the cloud-only and device-only extremes that motivated the rise of fog computing, performed worst, confirming that the value of the architecture is realized only through deliberate orchestration. Cloud-only execution, although it met nearly all deadlines because of the speed of cloud processors, incurred communication energy and would, in a wider-area deployment with greater propagation delay, fail the latency-critical applications that dominate the industrial edge. Device-only and fog-only execution failed on timeliness under contention. The intermediate strategies that reason about heterogeneity therefore dominated, and among them the contention-aware metaheuristic dominated the contention-blind greedy heuristic by a margin that grew with problem size.

The widening of the genetic algorithm’s advantage as the task count increased is the most consequential finding for practice. It indicates that the benefit of sophisticated offloading optimization is not uniform but

concentrated in precisely the regimes that matter most, namely densely loaded edge environments where many tasks compete for finite server capacity. In lightly loaded settings a simple heuristic suffices, and the additional computational cost of metaheuristic search is not warranted. As load rises and contention couples the placement decisions, the myopic heuristic increasingly misjudges the consequences of its choices, and the population-based search that internalizes the load-dependent coupling recovers performance that the heuristic leaves on the table. This pattern aligns with the broader literature's emphasis on orchestration and resource management as the open problems of the continuum, and it offers a concrete, quantified basis for deciding when to invest in advanced optimization.

The latency–energy trade-off analysis demonstrates that the framework provides decision-makers with a tunable control rather than a single fixed policy. By adjusting a single weight, an operator can move along the frontier between an energy-frugal posture, appropriate for battery-constrained or sustainability-focused deployments, and a latency-aggressive posture, appropriate for time-critical control loops. That deadline violations remained low across the sweep is important: the trade-off operates within the feasible region, so tuning toward energy savings does not silently sacrifice the timeliness guarantees on which industrial applications depend. This controllability connects the technical contribution to the managerial reality that different production contexts weigh responsiveness and operating cost differently.

Several aspects of the comparison between the two metaheuristics merit comment. The genetic algorithm consistently outperformed the particle swarm optimizer in both final solution quality and deadline satisfaction. The discrete, combinatorial structure of the offloading problem favors the integer-coded genetic representation and its crossover operator, which recombines whole placement decisions, over the continuous-position encoding of the swarm, which must be decoded to discrete assignments and thereby loses information near decision boundaries. The swarm's broader initial exploration did not compensate for this representational mismatch within the allotted iterations. This observation is consistent with the general finding that solution encoding is often more decisive than the choice of search metaphor for combinatorial problems, and it suggests that practitioners adapting these methods to richer offloading models should attend first to representation.

6. Conclusion

This paper surveyed the enabling technologies of Industry 4.0 with an emphasis on the edge–cloud computing continuum and developed a constrained, load-dependent optimization framework for computation offloading across the device, fog, and cloud tiers. A genetic algorithm and a particle swarm optimizer were evaluated against five baselines on synthetic industrial workloads, and the genetic algorithm reduced the weighted latency–energy objective by 34.1 percent relative to a greedy heuristic and by 82.6 percent relative to cloud-only execution while holding deadline violations to 1.6 percent, with its advantage widening under heavier load and a controllable latency–energy trade-off available through a single weight.

The study has limitations that bound the interpretation of these results. The workloads were synthetic and the system model, although calibrated to realistic orders of magnitude, abstracts away device mobility, time-varying channel conditions, task dependencies, and server failures; the offloading policy was computed offline over a static task set rather than online under arriving tasks; and the evaluation considered a single

fog cluster rather than a geographically distributed federation. For managers, the principal insight is that the return on investment in advanced offloading optimization is concentrated in densely loaded edge deployments, so the additional engineering effort is justified for high-utilization production cells but not for lightly loaded ones, and that the tunable weight allows a single framework to serve both latency-critical and sustainability-focused objectives without redesign. Future research should extend the framework to online and dynamic settings with arriving tasks and varying channels, incorporate task dependencies and multi-objective Pareto formulations, validate the model on physical edge–fog–cloud testbeds with real industrial traces, and integrate the offloading optimizer with the digital-twin and federated-learning workloads that increasingly define the industrial edge.

References

- Aceto, G., Persico, V., & Pescapé, A. (2020). Industry 4.0 and health: Internet of Things, big data, and cloud computing for Healthcare 4.0. *Journal of Industrial Information Integration*, 18, 100129. <https://doi.org/10.1016/j.jii.2020.100129>
- Alotaibi, B. (2023). A survey on Industrial Internet of Things security: Requirements, attacks, AI-based solutions, and edge computing opportunities. *Sensors*, 23(17), 7470. <https://doi.org/10.3390/s23177470>
- Bajić, B., Rikalovic, A., Suzic, N., & Piuri, V. (2021). Industry 4.0 implementation challenges and opportunities: A managerial perspective. *IEEE Systems Journal*, 15(1), 546–559. <https://doi.org/10.1109/JSYST.2020.3023041>
- Bécue, A., Praça, I., & Gama, J. (2021). Artificial intelligence, cyber-threats and Industry 4.0: Challenges and opportunities. *Artificial Intelligence Review*, 54(5), 3849–3886. <https://doi.org/10.1007/s10462-020-09942-2>
- Bonnard, R., Arantes, M. D. S., Lorbieski, R., Vieira, K. M. M., & Nunes, M. C. (2021). Big data/analytics platform for Industry 4.0 implementation in advanced manufacturing context. *The International Journal of Advanced Manufacturing Technology*, 117(5–6), 1959–1973. <https://doi.org/10.1007/s00170-021-07834-5>
- Boobalan, P., Ramu, S. P., Pham, Q.-V., Dev, K., Pandya, S., Maddikunta, P. K. R., Gadekallu, T. R., & Huynh-The, T. (2022). Fusion of federated learning and industrial Internet of Things: A survey. *Computer Networks*, 212, 109048. <https://doi.org/10.1016/j.comnet.2022.109048>
- Caiza, G., Saeteros, M., Oñate, W., & Garcia, M. V. (2020). Fog computing at industrial level, architecture, latency, energy, and security: A review. *Heliyon*, 6(4), e03706. <https://doi.org/10.1016/j.heliyon.2020.e03706>
- Chen, T., Sampath, V., May, M. C., Shan, S., Jorg, O. J., Aguilar Martín, J. J., Stamer, F., Fantoni, G., Tosello, G., & Calaon, M. (2023). Machine learning in manufacturing towards Industry 4.0: From “For Now” to “Four-Know”. *Applied Sciences*, 13(3), 1903. <https://doi.org/10.3390/app13031903>
- Firouzi, F., Farahani, B., & Marinšek, A. (2022). The convergence and interplay of edge, fog, and cloud in the AI-driven Internet of Things (IoT). *Information Systems*, 107, 101840. <https://doi.org/10.1016/j.is.2021.101840>
- Gkonis, P., Giannopoulos, A., Trakadas, P., Masip-Bruin, X., & D’Andria, F. (2023). A survey on IoT-Edge-Cloud continuum systems: Status, challenges, use cases, and open issues. *Future Internet*, 15(12), 383. <https://doi.org/10.3390/fi15120383>
- Jagatheesaperumal, S. K., Rahouti, M., Ahmad, K., Al-Fuqaha, A., & Guizani, M. (2022). The duo of artificial intelligence and big data for Industry 4.0: Applications, techniques, challenges, and future research directions. *IEEE Internet of Things Journal*, 9(15), 12861–12885. <https://doi.org/10.1109/JIOT.2021.3139827>
- Jamwal, A., Agrawal, R., Sharma, M., & Giallanza, A. (2021). Industry 4.0 technologies for manufacturing sustainability: A systematic review and future research directions. *Applied Sciences*, 11(12), 5725. <https://doi.org/10.3390/app11125725>
- Javaid, M., Haleem, A., Singh, R. P., Khan, S., & Suman, R. (2021). Blockchain technology applications for Industry 4.0: A literature-based review. *Blockchain: Research and Applications*, 2(4), 100027. <https://doi.org/10.1016/j.bcr.2021.100027>
- Javaid, M., Haleem, A., & Suman, R. (2023). Digital twin applications toward Industry 4.0: A review. *Cognitive Robotics*, 3, 71–92. <https://doi.org/10.1016/j.cogr.2023.04.003>

- John, J., Noor-A-Rahim, Md., Vijayan, A., Poor, H. V., & Pesch, D. (2024). Industry 4.0 and beyond: The role of 5G, WiFi 7, and Time-Sensitive Networking (TSN) in enabling smart manufacturing. *Future Internet*, 16(9), 345. <https://doi.org/10.3390/fi16090345>
- Karnik, N., Bora, U., Bhadri, K., Kadambi, P., & Dhatrak, P. (2022). A comprehensive study on current and future trends towards the characteristics and enablers of Industry 4.0. *Journal of Industrial Information Integration*, 27, 100294. <https://doi.org/10.1016/j.jii.2021.100294>
- Khan, L. U., Yaqoob, I., Tran, N. H., Kazmi, S. M. A., Dang, T. N., & Hong, C. S. (2020). Edge-computing-enabled smart cities: A comprehensive survey. *IEEE Internet of Things Journal*, 7(10), 10200–10232. <https://doi.org/10.1109/JIOT.2020.2987070>
- Leng, J., Wang, D., Shen, W., Li, X., Liu, Q., & Chen, X. (2021). Digital twins-based smart manufacturing system design in Industry 4.0: A review. *Journal of Manufacturing Systems*, 60, 119–137. <https://doi.org/10.1016/j.jmsy.2021.05.011>
- Liang, F., Yu, W., Liu, X., Griffith, D., & Golmie, N. (2020). Toward edge-based deep learning in industrial Internet of Things. *IEEE Internet of Things Journal*, 7(5), 4329–4341. <https://doi.org/10.1109/JIOT.2019.2963635>
- Lu, Y., Liu, C., Wang, K. I.-K., Huang, H., & Xu, X. (2020). Digital twin-driven smart manufacturing: Connotation, reference model, applications and research issues. *Robotics and Computer-Integrated Manufacturing*, 61, 101837. <https://doi.org/10.1016/j.rcim.2019.101837>
- Maddikunta, P. K. R., Pham, Q.-V., Prabadevi, B., Deepa, N., Dev, K., Gadekallu, T. R., Ruby, R., & Liyanage, M. (2022). Industry 5.0: A survey on enabling technologies and potential applications. *Journal of Industrial Information Integration*, 26, 100257. <https://doi.org/10.1016/j.jii.2021.100257>
- Oks, S. J., Jalowski, M., Lechner, M., Mirschberger, S., Merklein, M., Vogel-Heuser, B., & Möslin, K. M. (2024). Cyber-physical systems in the context of Industry 4.0: A review, categorization and outlook. *Information Systems Frontiers*, 26(5), 1731–1772. <https://doi.org/10.1007/s10796-022-10252-x>
- Pivoto, D. G. S., de Almeida, L. F. F., da Rosa Righi, R., Rodrigues, J. J. P. C., Lugli, A. B., & Alberti, A. M. (2021). Cyber-physical systems architectures for industrial Internet of Things applications in Industry 4.0: A literature review. *Journal of Manufacturing Systems*, 58, 176–192. <https://doi.org/10.1016/j.jmsy.2020.11.017>
- Pop, P., Zarrin, B., Barzegaran, M., Schulte, S., Punnekkat, S., Ruh, J., & Steiner, W. (2021). The FORA fog computing platform for industrial IoT. *Information Systems*, 98, 101727. <https://doi.org/10.1016/j.is.2021.101727>
- Qi, Q., Tao, F., Hu, T., Anwer, N., Liu, A., Wei, Y., Wang, L., & Nee, A. Y. C. (2021). Enabling technologies and tools for digital twin. *Journal of Manufacturing Systems*, 58, 3–21. <https://doi.org/10.1016/j.jmsy.2019.10.001>
- Qiu, T., Chi, J., Zhou, X., Ning, Z., Atiquzzaman, M., & Wu, D. O. (2020). Edge computing in industrial Internet of Things: Architecture, advances and challenges. *IEEE Communications Surveys & Tutorials*, 22(4), 2462–2488. <https://doi.org/10.1109/COMST.2020.3009103>
- Raja Sree, T. (2022). Role of fog-assisted Industrial Internet of Things: A systematic review. *Transactions on Emerging Telecommunications Technologies*, 33(11), e4611. <https://doi.org/10.1002/ett.4611>
- Rikalovic, A., Suzic, N., Bajic, B., & Piuri, V. (2022). Industry 4.0 implementation challenges and opportunities: A technological perspective. *IEEE Systems Journal*, 16(2), 2797–2810. <https://doi.org/10.1109/JSYST.2021.3101673>
- Wu, Y., Dai, H.-N., Wang, H., Xiong, Z., & Guo, S. (2022). A survey of intelligent network slicing management for industrial IoT: Integrated approaches for smart transportation, smart energy, and smart factory. *IEEE Communications Surveys & Tutorials*, 24(2), 1175–1211. <https://doi.org/10.1109/COMST.2022.3158270>
- Xu, H., Wu, J., Pan, Q., Guan, X., & Guizani, M. (2023). A survey on digital twin for industrial Internet of Things: Applications, technologies and tools. *IEEE Communications Surveys & Tutorials*, 25(4), 2569–2598. <https://doi.org/10.1109/COMST.2023.3297395>