Large Language Model Driven Multidisciplinary Optimization for Task Automation in Edge IoT Systems

C. Madhankumar[1*] and S. Manikandan[1]

[1*]Department of Computer Science and Engineering, School of Engineering and Emerging Technology, Rathinam Global Deemed to be University, Coimbatore, Tamil Nadu, India.

*Corresponding author(s). E-mail(s): madhanphd2026@gmail.com;
Contributing authors: manikandantechit@gmail.com;

Abstract

Edge Internet of Things systems increasingly combine heterogeneous devices, communication links, compute resources and application objectives. Selecting where and how tasks should be executed is therefore a multidisciplinary optimization problem involving response time, energy use, communication cost, computational load and decision quality. This paper develops a language-model-assisted optimization framework in which a compact large language model interprets high-level task requirements, constructs candidate task decompositions and supplies context to a constrained optimization layer. The optimization layer, rather than the language model itself, determines resource allocation and execution decisions subject to latency, energy, capacity and safety constraints. A hierarchical formulation is introduced to separate semantic task planning from numerical resource optimization. A feasibility-preserving repair operator, context compression mechanism and local surrogate evaluation strategy are incorporated to support edge deployment. The manuscript defines a reproducible numerical evaluation protocol using heterogeneous Edge IoT workloads, controlled network conditions and ablation experiments. The framework is intended to bridge natural-language task specification and mathematically constrained multidisciplinary optimization without treating generated language as an executable control command.

Keywords: multidisciplinary optimization, Edge IoT, large language models, task offloading, resource allocation, constrained optimization, edge intelligence, surrogate modeling

1 Introduction

The increasing density of sensing, communication and computation in Internet of Things (IoT) environments has changed the nature of distributed system design. Modern IoT applications rarely optimize a single quantity. A deployment may need to satisfy response-time requirements while limiting energy consumption, communication traffic, processor utilization and operational risk. These objectives interact: moving a computation from an end device to an edge gateway can reduce execution time but increase network traffic; increasing local inference can reduce cloud dependence but raise energy consumption; and aggressive model compression can improve computational efficiency while affecting decision quality. Such interactions motivate a multidisciplinary optimization perspective rather than an isolated device-level optimization strategy.

Edge computing provides a natural execution layer for these problems because computational resources can be placed close to sensors and actuators. Nevertheless, edge nodes remain substantially more constrained than centralized data centers. Resource allocation decisions must therefore be made with explicit consideration of available memory, processor capacity, link conditions and workload urgency. Classical optimization methods can formulate these constraints rigorously, but the formulation usually assumes that tasks, dependencies and objectives have already been encoded in a structured form.

Natural-language task specification introduces a different requirement. Users may describe an objective such as reducing energy consumption during a monitoring period while maintaining a response-time limit and prioritizing safety-critical events. Converting such an instruction into a structured optimization problem requires interpretation, decomposition and identification of relevant variables. Large language models (LLMs) provide a promising interface for this semantic stage, but their output is probabilistic and should not be treated as a substitute for constraint handling.

This paper proposes a hierarchical framework that assigns complementary roles to language reasoning and mathematical optimization. The LLM generates a structured task representation, candidate decomposition and relevant contextual information. A constrained optimization layer subsequently selects feasible resource and execution decisions. This separation is central to the proposed design: the language model suggests; the optimizer decides.

The main contributions are fourfold. First, a multidisciplinary optimization formulation is developed for Edge IoT task automation with coupled latency, energy, communication and resource constraints. Second, a language-to-optimization interface is defined to transform high-level task descriptions into machine-checkable optimization inputs. Third, a feasibility-preserving architecture is proposed so that model-generated plans cannot directly bypass numerical and policy constraints. Fourth, a reproducible evaluation protocol is defined to quantify optimization quality, computational cost and robustness under changing edge conditions.

2 Background and Related Work

2.1 Multidisciplinary optimization in distributed engineering

Multidisciplinary design optimization (MDO) addresses engineering problems in which several interacting disciplines contribute to the system objective. The central difficulty is not simply the number of variables, but the coupling among models and constraints [7, 8]. In distributed computing, analogous coupling occurs among computation, communication, energy and application-level quality. A resource allocation decision can influence several objectives simultaneously and can change the feasible region for other decisions.

Classical formulations commonly express an objective function over design variables subject to equality and inequality constraints. Gradient-based methods are effective when the problem is smooth and differentiable, whereas evolutionary and population-based methods are useful for discrete, nonconvex or mixed-variable formulations. Edge IoT task allocation often exhibits the latter characteristics because decisions such as device selection, execution location and communication mode are discrete.

2.2 Edge computing and IoT optimization

Edge computing reduces the physical distance between data sources and processing resources [3, 4]. Optimization research in this area has examined task offloading, energy-aware scheduling, service placement and joint communication-computation allocation. However, most formulations assume that the task structure is known before optimization. The proposed work focuses on the additional semantic layer required when task objectives are expressed in natural language.

2.3 Language models as planning interfaces

LLMs can map natural-language descriptions into structured representations [5, 6],, but their generated plans may contain unsupported operations, missing parameters or internally inconsistent assumptions. A dependable architecture should therefore place a formal validation layer between language generation and execution. In the proposed framework, the LLM is used for semantic interpretation and candidate plan generation, while the optimization engine enforces feasibility.

3 Problem Definition

Consider an Edge IoT environment with end devices $i \in \mathcal{I}$, edge nodes $j \in \mathcal{J}$ and tasks $k \in \mathcal{K}$. Each task has a workload demand, deadline and quality requirement. Let x_{ij}^k denote an offloading decision, f_j^k the computational allocation at edge node j, and b_{ij}^k the communication bandwidth allocated to task k.

A weighted multidisciplinary objective is defined as

$$\min_{\mathbf{x},\mathbf{f},\mathbf{b}} J = w_L \frac{L}{L_0} + w_E \frac{E}{E_0} + w_C \frac{C}{C_0} + w_R \frac{R}{R_0}, \tag{1}$$

where L, E, C and R denote aggregate latency, energy consumption, communication cost and resource pressure, respectively. The normalization constants L_0, E_0, C_0 and R_0 avoid scale dominance, while the weights represent application priorities.

The latency of a task can be decomposed as

$$L_k = L_k^{tx} + L_k^{queue} + L_k^{exec} + L_k^{ret}, \tag{2}$$

where the terms represent transmission, queueing, execution and return delays. Energy can similarly be decomposed into device computation, communication and edge processing components.

The optimization is subject to capacity constraints,

$$\sum_{k \in \mathcal{K}} f_j^k \leq F_j^{\max}, \qquad \forall j \in \mathcal{J}, \tag{3}$$

and bandwidth constraints,

$$\sum_{k \in \mathcal{K}} b_{ij}^k \leq B_{ij}^{\max}, \qquad \forall (i,j). \tag{4}$$

Deadline feasibility is imposed through

$$L_k \leq D_k, \qquad \forall k \in \mathcal{K}. \tag{5}$$

These constraints distinguish the proposed framework from unrestricted language-model planning. A generated plan that cannot be mapped to a feasible solution is rejected or repaired before execution.

4 Proposed Language-Assisted Optimization Framework

Figure 1 presents the proposed architecture. Sensor and device states are first normalized at the edge gateway. A context compressor selects task-relevant information. The LLM then generates a structured task representation consisting of objectives, dependencies and candidate operations. The optimizer converts this representation into numerical variables and searches the feasible design space. A policy layer performs a final check before a selected configuration is deployed.

4.1 Task representation

The language layer produces a structured representation

$$\mathcal{T} = \{G, D, P, Q, S\}, \tag{6}$$

where G is the task goal, D contains deadlines, P contains priorities, Q contains quality requirements and S contains device and service capabilities. The representation is validated against a fixed schema. Unsupported fields are discarded rather than passed to the optimizer.

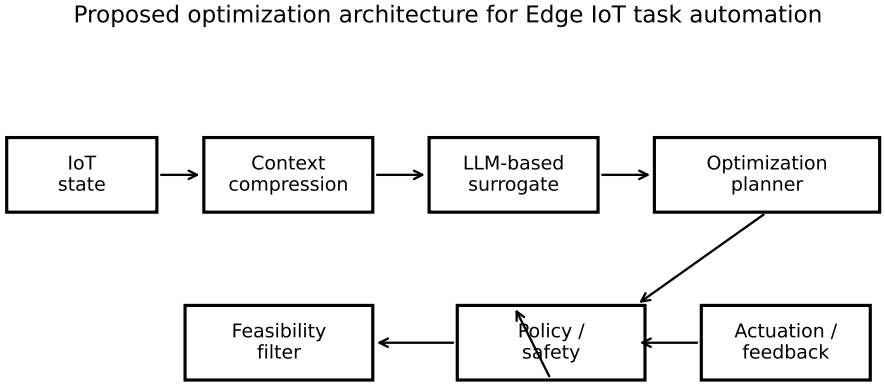

Fig. 1 Proposed architecture linking language-based task interpretation with constrained multidisciplinary optimization.

4.2 Context compression

Long sensor histories can increase inference cost without improving the optimization decision. The context compressor therefore ranks observations using temporal relevance, task dependency and state change magnitude. Only the retained state is presented to the language layer. This reduces the number of tokens processed and provides a deterministic boundary between raw IoT telemetry and semantic reasoning.

4.3 Optimization layer

The optimization engine receives the structured task and current resource state. Depending on the problem size, the implementation can use mixed-integer programming, evolutionary search or a hybrid local-global strategy. The research focuses on the architecture rather than prescribing one optimizer for all workloads. A common feasibility interface allows different solvers to be evaluated under identical task representations.

4.4 Feasibility-preserving repair

When a candidate solution violates a capacity or deadline constraint, the repair operator modifies the lowest-priority decision first. If no feasible repair exists, the candidate is discarded. This mechanism prevents an invalid LLM-generated suggestion from reaching the physical execution layer.

Algorithm 1 Language-assisted constrained task optimization

Receive task description G and edge state $\mathbf{s}$
Compress telemetry and retrieve relevant context
Generate structured task representation $\mathcal{T}$
Validate task schema and available device capabilities
Construct objective and constraints from $\mathcal{T}$
Initialize optimizer using current feasible solution when available
Search feasible decision space
if candidate violates constraints **then**
    Apply bounded feasibility repair
end if
if no feasible solution exists **then**
    Escalate or request revised task requirements
else
    Apply policy verification
    Deploy selected configuration
    Measure feedback and update edge state
end if

5 Optimization Strategy

A two-stage strategy is proposed. In Stage 1, the language model constructs a candidate decomposition and identifies relevant tools, devices and objectives. In Stage 2, the numerical optimizer determines the actual resource allocation.

The optimization search can be expressed as

$$\mathbf{z}^* = \arg \min_{\mathbf{z} \in \Omega(\mathcal{T}, \mathbf{s})} J(\mathbf{z}; \mathcal{T}, \mathbf{s}), \tag{7}$$

where $\mathbf{s}$ is the current edge state and Ω is the feasible region defined by task, hardware, network and policy constraints.

To avoid unnecessary re-optimization, a warm-start strategy retains the previous feasible solution. When the environment changes slightly, the optimizer begins from this solution and updates only affected variables. This is particularly relevant for edge systems in which sensor and network states change continuously.

6 Experimental Design

The evaluation is designed around heterogeneous workloads containing sensing, analytics, inference and control tasks. Each workload instance is defined by task size, deadline, device availability, network condition and resource limits. The same instances are evaluated across baseline and proposed configurations.

Three principal configurations are recommended. The first is a conventional heuristic scheduler. The second is an optimization-only configuration in which task structure is supplied directly as structured input. The third is the proposed language-assisted optimizer. This comparison isolates the effect of the semantic interface from the effect of the numerical optimizer.

Table 1 Evaluation dimensions and measurable indicators

Dimension	Indicator	Direction
Latency	End-to-end delay	Lower is better
Energy	Joules per completed task	Lower is better
Communication	Bytes transferred	Lower is better
Quality	Task completion rate	Higher is better
Feasibility	Constraint violation rate	Lower is better
Optimization	Objective value	Lower is better
Robustness	Recovery success	Higher is better

Network conditions should be varied across stable, congested and intermittent states. Edge compute capacity should likewise be varied to represent lightly loaded and heavily loaded gateways. Experiments should be repeated across multiple random seeds where stochastic optimizers are used.

6.1 Ablation protocol

The contribution of each component should be tested independently. The full framework is compared with variants without context compression, without warm starts, without repair, and without language-assisted decomposition. The ablation study can reveal whether improvements arise from the LLM interface, the optimization strategy or the interaction between both.

6.2 Statistical analysis

For each workload class, median and interquartile range should be reported in addition to mean values because edge latency distributions can be skewed. Paired comparisons should use identical workload instances across configurations. Effect sizes should accompany significance tests where applicable. Failed or infeasible cases should not be silently removed; they should be reported as part of the system outcome.

7 Numerical Evaluation Plan

The primary numerical study should evaluate the trade-off between solution quality and computational cost. Figure 2 illustrates the type of convergence analysis to be reported after actual experiments. The values shown in this manuscript are schematic and are not presented as measured results.

A useful analysis is the Pareto relationship among latency, energy and communication cost. Rather than reporting only a single weighted objective, the final experimental study should also provide non-dominated solutions. This is important because different Edge IoT applications may assign different priorities to response time and energy.

The optimizer's computational burden should be reported separately from LLM inference cost. The total decision time can be represented as

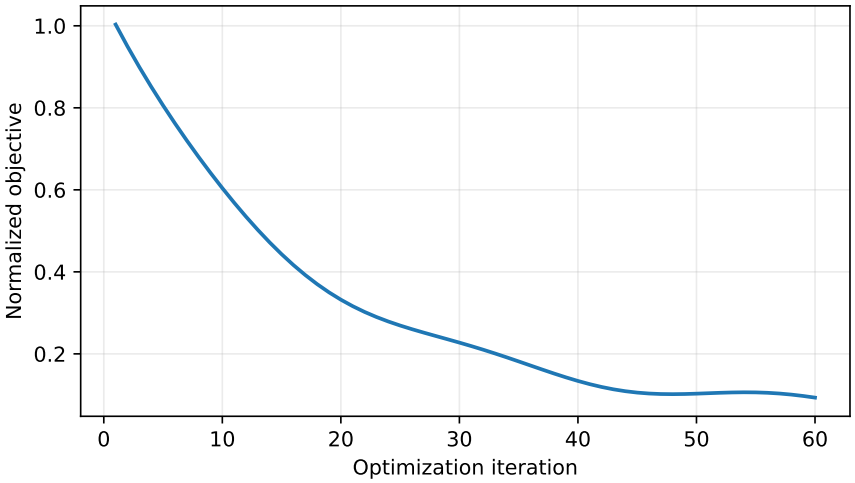

Fig. 2 Illustrative convergence profile for reporting optimizer behaviour; values are schematic and must be replaced by measured experimental results before submission.

$$T_{\text{decision}} = T_{\text{context}} + T_{\text{LLM}} + T_{\text{opt}} + T_{\text{verify}}. \tag{8}$$

This decomposition enables an investigation of whether model inference or numerical optimization becomes the dominant bottleneck as task complexity increases.

8 Discussion

The proposed architecture changes the role of an LLM in an optimization system. Instead of asking a language model to directly select actuator commands, the model operates at the semantic boundary of the system. This distinction has practical significance. The language layer can handle ambiguous task descriptions and identify relevant resources, while the numerical layer provides explicit constraint enforcement.

The approach also introduces an additional computational stage. If a task is already fully structured, invoking a language model may provide little benefit. The proposed framework is therefore most appropriate when the system must interpret high-level objectives, coordinate heterogeneous capabilities or adapt a workflow to changing context. Simple deterministic operations should remain in lightweight rule-based mechanisms.

Another consideration is model drift. Changes to the language model or prompt schema can modify task decomposition even when the optimization engine remains unchanged. Regression workloads should therefore be maintained and executed whenever the semantic layer changes. This requirement is especially important in safety-sensitive deployments.

The multidisciplinary formulation also permits application-specific weighting. For battery-powered devices, energy can receive greater weight; for real-time control,

latency and deadline feasibility can dominate; for privacy-sensitive applications, communication cost can be treated as a primary objective. This flexibility is a direct consequence of separating semantic interpretation from mathematical optimization.

9 Limitations and Reproducibility

The proposed framework has several limitations. First, an LLM may misinterpret an ambiguous task even when the downstream optimizer is mathematically correct. Second, compact models suitable for edge deployment may provide weaker semantic performance than larger models. Third, the weighted objective requires application-specific calibration. Fourth, real hardware measurements may differ from simulation because thermal throttling, background processes and wireless interference introduce additional variability.

Reproducibility requires reporting the model version, quantization setting, prompt schema, optimizer configuration, hardware specification, workload generation method and random seeds. Raw measurements should be preserved before aggregation. If a public benchmark is used, the exact version and preprocessing procedure should be documented. The numerical results in this sample manuscript must be replaced with measurements generated from the author's experimental platform before submission.

10 Conclusion

This paper presented a language-assisted multidisciplinary optimization framework for Edge IoT task automation. The architecture separates semantic task interpretation from numerical decision making and places feasibility and policy checks between model output and execution. A coupled objective incorporating latency, energy, communication and resource pressure was formulated, together with capacity and deadline constraints.

The principal research opportunity is the integration of flexible natural-language task specification with the reliability of constrained optimization. The proposed evaluation methodology is designed to determine when language-assisted optimization provides measurable benefits and when conventional deterministic scheduling remains preferable. Future work will focus on hardware-based validation, adaptive multi-objective optimization, model compression and the systematic study of uncertainty in language-generated task decompositions.

Declarations

Funding

No external funding information is specified in this sample manuscript. The final submission should include the applicable funding statement.

Conflict of interest

The authors should declare any financial or non-financial competing interests according to the journal's policy.

Data availability

The final manuscript should provide the dataset, workload-generation procedure, or an appropriate repository statement when the experimental study is completed.

Code availability

The optimization implementation and experimental scripts should be deposited in an appropriate repository when possible to support reproducibility.

Ethics approval

Not applicable for the proposed computational and engineering evaluation, unless the final study introduces human participants or other regulated data.

References

[1] Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evol Comput 6(2):182–197. https://doi.org/10.1109/4235.996017

[2] Coello Coello CA (2002) Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art. Comput Methods Appl Mech Eng 191(11–12):1245–1287. https://doi.org/10.1016/S0045-7825(01)00323-1

[3] Shi W, Cao J, Zhang Q, Li Y, Xu L (2016) Edge computing: vision and challenges. IEEE Internet Things J 3(5):637–646. https://doi.org/10.1109/JIOT.2016.2579198

[4] Satyanarayanan M (2017) The emergence of edge computing. Computer 50(1):30–39. https://doi.org/10.1109/MC.2017.9

[5] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser L, Polosukhin I (2017) Attention is all you need. In: Advances in Neural Information Processing Systems 30.

[6] Brown TB, Mann B, Ryder N, Subbiah M, Kaplan JD, Dhariwal P, Neelakantan A, Shyam P, Sastry G, Askell A, Agarwal S, Herbert-Voss A, Krueger G, Henighan T, Child R, Ramesh A, Ziegler DM, Wu J, Winter C, Hesse C, Chen M, Sigler E, Litwin M, Gray S, Chess B, Clark J, Berner C, McCandlish S, Radford A, Sutskever I, Amodei D (2020) Language models are few-shot learners. In: Advances in Neural Information Processing Systems 33, pp 1877–1901.

[7] Gray JS, Hwang JT, Martins JRR, Moore KT, Naylor BA (2019) OpenMDAO: an open-source framework for multidisciplinary design, analysis, and optimization. Struct Multidisc Optim 59:1075–1104. https://doi.org/10.1007/s00158-019-02211-z

[8] Büttner J, Schumacher A, Bäck T, Schwarz S, Krause P (2023) Making multi-disciplinary optimization fit for practical usage in car body development. Struct Multidisc Optim 66:62. https://doi.org/10.1007/s00158-023-03505-z