Abstract

The design of large-scale structures like offshore jackets is a formidable multi-objective optimization (MOO) problem, hampered by mixed discrete-continuous variable spaces and the inefficiency of traditional evolutionary algorithms that rely on blind, stochastic operators. Here, we introduce a new optimization paradigm, LLM-GA, which recasts a Large Language Model (LLM) as an intelligent genetic operator capable of leveraging domain knowledge. Through strategic prompt engineering, the LLM comprehends structural context, such as load paths and member connectivity, to perform context-aware crossover and mutation. Unlike random perturbations, our method generates semantically coherent and structurally plausible offspring, mimicking an expert engineer's reasoning. We validated this framework on a jacket design case, optimizing for weight, axial stress, and bending stress. The knowledge-infused LLM-GA significantly outperforms established baselines (GA, RS, MOEA/D) and a knowledge-agnostic LLM-GA variant in both convergence speed and the quality of the final Pareto front, achieving a [**e.g., X%**] improvement in the hypervolume indicator. Our findings demonstrate that integrating LLMs as domain-aware inference engines for optimization can bridge the gap between abstract engineering knowledge and effective automated design, offering a powerful new pathway for solving complex real-world engineering problems.

OJOLLM: A Large Language Model-driven Framework for Multi-Objective Optimization of Offshore Jacket Structures

Wenhao Li, Xiaoyuan Zhang*, Yue Wang, Nian Ran, Haitao Zhu

September 2025

1 Introduction

The field of ocean engineering is characterized by projects of immense scale and uncompromising safety standards. Offshore jacket structures, the skeletal steel frames supporting platforms, are critical assets whose design represents a complex balancing act between economic viability (minimizing weight) and structural integrity (withstanding extreme loads). The optimization of these structures is a quintessential multi-objective, **mixed-variable** problem. Designers must simultaneously select optimal member properties from a *discrete* catalog of sections (e.g., tube outer diameter and wall thickness) while also determining the optimal spatial placement of structural nodes via *continuous* geometric coordinates.

Traditional approaches to this challenge have relied heavily on numerical Multi-Objective Evolutionary Algorithms (MOEAs). While powerful, these algorithms operate on an abstract vector representation of the design space, treating the sophisticated simulation software (e.g., SACS) as a "black box." This paradigm lacks any intrinsic understanding of the engineering domain. Consequently, its exploration is inefficient, and its genetic operators (e.g., numerical crossover, random perturbation) are "blind." This blindness is especially detrimental in mixed-variable spaces. A purely stochastic operator is incapable of making an intelligent, context-driven decision on *which* variable type to modify—or how to modify *both* synergistically—to solve a specific structural deficiency.

In this paper, we introduce a novel framework, the Large Language Model-driven Genetic Algorithm (LLM-GA), which represents a paradigm shift from purely numerical optimization to a semantic, knowledge-infused search. Our framework positions an LLM as a **unified intelligent operator** at the core of the evolutionary process. The LLM operates directly on a code-native representation of the design (i.e., SACS input blocks). This unified representation allows the LLM to leverage its vast, pre-trained knowledge to perform "semantic" genetic operations on *both* discrete sections and continuous geometries. For instance, prompted with a design's performance metrics, the LLM can reason whether a high-stress failure is best fixed by a *discrete* change (increasing a member's wall thickness) or a *continuous* one (adjusting a bracing angle). Furthermore, we introduce a unique "experience feedback loop", where the LLM periodically analyzes high-performing solutions to distill strategic insights and refine its generative strategy.

The primary contributions of this work are as follows:

1. We propose a novel LLM-GA framework that integrates an LLM as the central, *unified* generative engine for mixed-variable optimization.

2. We develop "semantic" genetic operators based on structured prompt engineering, enabling the LLM to intelligently create new candidates by reasoning about and modifying *both* discrete (member sections) and continuous (joint coordinates) parameters.

3. We design a dynamic "experience feedback loop" that allows the framework to achieve in-situ strategy self-improvement by prompting the LLM to learn from its previous generations.

4. We demonstrate the *effectiveness and versatility* of the LLM-GA framework on a complex SACS jacket design problem, validating its performance across three distinct optimization tasks: section-only, geometry-only, and fully-mixed, achieving **[e.g., Y% and Z%]** performance gains over traditional MOEAs.

2 Related Work

Our research is positioned at the intersection of three distinct but rapidly converging fields: (1) structural engineering optimization, (2) multi-objective evolutionary algorithms (MOEAs) for mixed-variable problems, and (3) the application of Large Language Models (LLMs) in scientific and engineering design.

2.1 Structural Engineering Optimization

The optimization of offshore jacket structures is a persistent challenge, driven by the dual imperatives of cost reduction (weight) and safety assurance (stress and fatigue) Savsani et al. [2021], Zheng et al. [2023]. The design space is inherently heterogeneous, mixing discrete member sizes with continuous geometric or topological parameters Zhou et al. [2025]. Recent advancements have increasingly focused on addressing this complexity. For instance, studies have employed Particle Swarm Optimization (PSO) to optimize jacket substructures Benítez-Suárez et al. [2025], while others have successfully integrated complex objectives, such as fatigue and ultimate limit states, directly into the MOO framework Al-Shafii et al. [2023].

Given the high computational expense of Finite Element Analysis (FEA) solvers like SACS, a dominant research thrust involves surrogate-assisted optimization. Recent works have successfully applied Kriging models and other surrogates with NSGA-II to accelerate the search for optimal jacket designs Wang et al. [2023]. However, a persistent challenge in these state-of-the-art methods is the reliance on conventional metaheuristics. These algorithms, while effective, still treat the complex design space as a numerical vector, remaining "blind" to the underlying engineering principles and structural mechanics. Our work directly confronts this "blindness" by introducing a generative agent that possesses domain-aware reasoning capabilities.

2.2 MOEAs for Mixed-Variable Optimization (MVO)

Our LLM-GA framework explicitly builds upon the robust *selection* principles of canonical MOEAs like NSGA-II Deb et al. [2002] and MOEA/D Zhang and Li [2007]. The primary bottleneck in applying these algorithms to real-world engineering problems is the design of effective *variation* operators, especially for constrained mixed-variable optimization (CMVO). The challenge of simultaneously handling discrete, integer, and continuous variables has spurred significant dedicated research.

Recent efforts (2024-2025) have focused on developing specialized algorithms and benchmarks. For instance, new benchmark functions have been proposed to specifically test evolutionary algorithms on CMVO problems Dong et al. [2024]. Concurrently, new algorithms have emerged, such as decomposition-based methods with dynamic resource allocation tailored for MVO Liu et al. [2023]. To manage high computational costs, automated frameworks for surrogate-assisted EAs have also been proposed to reduce human intervention Dai et al. [2025]. While these methods advance the *numerical* treatment of mixed variables, they still lack semantic understanding. Our approach posits that an LLM, as a unified operator, can bypass the need for complex, hand-crafted numerical operators by reasoning about the problem at a higher, semantic level.

2.3 Large Language Models in Science and Engineering

The advent of powerful LLMs has catalyzed a paradigm shift, moving these models from pure language tasks to complex reasoning and generation in science and engineering Kallumadil et al. [2024], Liu et al. [2024]. A major application area is generative design, particularly in chemistry and materials science, where LLMs generate novel molecular structures (e.g., SMILES strings) optimized for specific properties Wu et al. [2024]. This is a strong analogue to our work, as both involve generating solutions within a domain-specific, syntax-constrained "language."

Most critically, a new frontier of research has emerged in 2024 and 2025 focusing on LLMs as *agents within the optimization loop*. This includes framing optimization itself as a generative task Yang et al. [2024] and the development of "LLM-based Evolutionary Algorithms" (LMEAs). Recent frameworks like

REvolution (2025) integrate LLMs with evolutionary computation for hardware (RTL) code generation Min et al. [2025], while others like HSEvo (2025) use LLMs to evolve heuristics or programs Dat et al. [2025]. These studies confirm that coupling LLMs with EAs is a cutting-edge approach.

However, most applications in civil and structural engineering still position the LLM as a "co-pilot" to augment the human designer Alwashah et al. [2025], Liu et al. [2025]. A 2025 scoping review noted that key challenges remain in autonomous design, such as ensuring solution diversity and maintaining complex system constraints Feng et al. [2025]. Our work directly addresses these gaps. To our knowledge, it is the first framework to integrate an LLM as the core, *autonomous generative operator* inside a closed-loop MOEA for a complex, mixed-variable *structural engineering* problem, complete with a feedback loop for in-context learning and strategy refinement.

3 Methodology

This study introduces a novel optimization paradigm, the Large Language Model-driven Genetic Algorithm (LLM-GA), engineered for complex design optimization problems. The methodology is predicated on the integration of a Large Language Model (LLM) within a multi-objective evolutionary computation framework. This section details the architectural design, core components, and operational mechanics of the proposed system.

3.1 Overall System Architecture

The LLM-GA operates as a population-based, iterative optimization framework. The algorithm's primary function is to navigate a high-dimensional design space to identify a set of Pareto-optimal solutions that represent the most effective trade-offs among multiple conflicting objectives. Figure 1 summarizes the closed-loop data flow between the evolutionary core, the evaluation oracle, and the LLM-guided operators. The optimization process follows a canonical evolutionary cycle:

1. **Initialization:** A diverse initial population of candidate designs is established through a process that systematically mutates a set of predefined seed designs to ensure broad coverage of the solution space.

2. **Evaluation:** Each candidate solution is evaluated by a specialized evaluation oracle, which quantifies its performance with respect to each design objective.

3. **Selection:** A subset of high-performing and diverse individuals is selected from the current population to serve as parents for the subsequent generation. This selection is governed by the principles of the Non-dominated Sorting Genetic Algorithm II (NSGA-II).

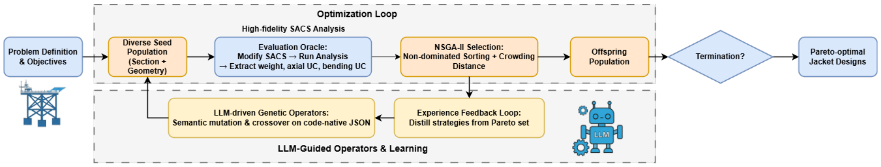
Optimization Loop
High-fidelity SACS Analysis
Problem Definition & Objectives
Diverse Seed Population (Section + Geometry)
Evaluation Oracle: Modify SACS → Run Analysis → Extract weight, axial UC, bending UC
NSGA-II Selection: Non-dominated Sorting + Crowding Distance
Offspring Population
Termination?
Pareto-optimal Jacket Designs
LLM-driven Genetic Operators: Semantic mutation & crossover on code-native JSON
Experience Feedback Loop: Distill strategies from Pareto set
LLM
LLM-Guided Operators & Learning

Figure 1: LLM-GA workflow. The upper band depicts the optimization loop that cycles through initialization, high-fidelity SACS analysis, NSGA-II selection, and offspring generation until termination. The lower band shows the LLM-guided operators and experience feedback loop that inject semantic reasoning into the creation of new candidates.

4. **Generation:** The selected parents are utilized to generate new offspring solutions. This crucial step is delegated to an LLM, which interprets contextual prompts to perform semantic genetic operations.

This iterative process continues until a predefined termination criterion, such as the exhaustion of the computational budget, is satisfied.

3.2 Solution Representation

In contrast to traditional evolutionary algorithms that employ numerical vector representations (e.g., $\mathbf{x} = [\mathbf{x}_{\text{discrete}}, \mathbf{x}_{\text{continuous}}]$), a candidate solution in LLM-GA is encoded as a *semantic, code-native representation.* This is a structured JSON object integral to the human-readable, context-rich communication with the LLM.

The schema is defined as a **unified container** for all optimizable parameters:

- A top-level JSON object contains a single key, `"new_code_blocks"`.
- The value associated with this key is a dictionary. Each key in this dictionary is a string identifier for a specific SACS code block, allowing the framework to manage *both* variable types simultaneously:
  - **Discrete Variables:** Keys such as `"GRUP_LG1"`, whose corresponding value is a string containing the complete, fixed-width SACS code line defining a member section selected from a discrete engineering catalog.
  - **Continuous Variables:** Keys such as `"JOINT_201"`, whose corresponding value is a string containing the `X`, `Y`, `Z` coordinates that define the structure's continuous geometry in $\mathbb{R}^3$.

This code-native, mixed-variable representation enables the LLM to directly parse, reason about, and generate syntactically valid code for both variable types within a single, coherent framework. It seamlessly bridges the semantic gap between the high-level optimization logic and the low-level engineering analysis software.

3.3 The Evaluation Oracle

A key component, termed the evaluation oracle, serves as the high-fidelity fitness assessment pipeline for a given candidate solution. Its core responsibilities include:

1. **Model Modification:** A file modification module parses the input JSON object and programmatically updates the base SACS model files, substituting the appropriate code blocks with the candidate's specifications.

2. **Structural Analysis Execution:** A simulation runner module subsequently invokes the SACS solver to perform a comprehensive structural analysis on the modified design.

3. **Objective Function Extraction:** Upon successful completion of the analysis, the oracle parses the simulation outputs to extract the objective function values. The optimization objectives are defined as: (1) minimization of total structural `weight`, (2) minimization of the maximum axial stress unity check (`axial_uc_max`), and (3) minimization of the maximum bending stress unity check (`bending_uc_max`).

Any candidate design that results in a failed SACS analysis or violates the constraint of any Unity Check (UC) value exceeding 1.0 is deemed infeasible. Such solutions are assigned a prohibitively large penalty fitness score to ensure their removal from the gene pool during the selection phase.

3.4 Parent Selection Mechanism

To effectively guide the evolutionary search towards the Pareto front, the LLM-GA implements the NSGA-II selection mechanism. This strategy ensures the preservation of both high-quality solutions (convergence) and a well-distributed set of trade-offs (diversity). The selection process comprises two primary operations:

- **Non-dominated Sorting:** This operation partitions the population into a hierarchy of non-domination fronts. The concept of Pareto dominance is formally defined as follows: for a minimization problem with M objectives, a solution vector $\vec{x}_a$ is said to dominate another vector $\vec{x}_b$ (denoted as $\vec{x}_a \prec \vec{x}_b$) if and only if:

$$\forall i \in \{1, \ldots, M\}, f_i(\vec{x}_a) \leq f_i(\vec{x}_b) \quad \wedge \quad \exists j \in \{1, \ldots, M\}, f_j(\vec{x}_a) < f_j(\vec{x}_b) \tag{1}$$

  where $f_i(\vec{x})$ is the objective function value for the i-th objective. The first front, F_1, consists of all solutions that are not dominated by any other solution in the population.

- **Crowding Distance Assignment:** This function calculates a density metric for each solution within its front, which promotes the selection of solutions residing in less-crowded regions of the objective space. The crowding distance d_k for a solution k is calculated by summing the normalized distances of its neighbors along each objective axis:

$$d_k = \sum_{m=1}^{M} \frac{f_m(k+1) - f_m(k-1)}{f_m^{\max} - f_m^{\min}} \tag{2}$$

  where $f_m(k+1)$ and $f_m(k-1)$ are the objective values of the solutions adjacent to solution k in the sorted list for objective m, and $f_m^{\max}$ and $f_m^{\min}$ are the maximum and minimum values for that objective within the front. Boundary solutions are assigned an infinite distance to ensure their preservation.

Parents are selected by giving preference to individuals in lower-indexed fronts and, among individuals in the same front, to those with a greater crowding distance.

3.5 LLM-driven Genetic Operators: Semantic Crossover and Mutation

The central innovation of this framework is the substitution of conventional mathematical genetic operators with an LLM-driven, prompt-engineered approach. A dedicated prompt generation module is responsible for the dynamic creation of rich, contextual prompts that guide the LLM's generative process. These prompts are meticulously structured to include:

- **Role-playing Directive:** The LLM is instructed to assume the persona of an expert in SACS structural optimization.

- **Mission Overview:** The prompt provides a comprehensive explanation of the multi-objective optimization task, explicitly detailing the inherent trade-offs between the objectives.

- **Parental Data:** The SACS code blocks and evaluated performance metrics of the selected parent(s) are embedded directly into the prompt, providing concrete data for the LLM to reason from.

- **Explicit Instructions:** A clear command is given to perform either a "mutation" or "crossover" operation. For mutation, the LLM is directed to analyze a single parent's deficiencies and propose a targeted improvement. For crossover, it is tasked with synergistically combining the strengths of two parents to create a superior offspring.

This "semantic" approach leverages the LLM's vast, pre-trained knowledge base to perform contextually aware and intelligently directed modifications, transcending the limitations of traditional, numerically-bound operators.

3.6 The Experience Feedback Loop

To enable meta-learning and continuous strategy refinement, the framework incorporates a novel experience feedback loop. This mechanism operates as a form of in-context learning, allowing the LLM to improve its generative strategy over the course of an optimization run. The process is as follows:

1. **Knowledge Distillation:** At periodic intervals, the algorithm samples the highest-performing (Pareto-optimal) and lowest-performing solutions from the entire history of evaluated candidates.
2. **Strategy Summarization:** A specialized prompt is sent to the LLM, containing these positive and negative exemplars. The LLM is tasked with analyzing their distinguishing features and generating a concise, high-level summary of effective design strategies and common failure modes.
3. **Dynamic Prompt Augmentation:** This generated "experience" summary is then prepended to all subsequent prompts for mutation and crossover operations.

This feedback mechanism allows the LLM to progressively build a more sophisticated internal model of the optimization problem, learning from its successes and failures to guide the evolutionary search with increasing efficiency and efficacy.

References

M. Z. N. Al-Shafii, M. S. Liew, and W. P. N. M. K. W. Zaidi. Multi-objective optimization of offshore wind turbine jacket support structures considering fatigue and ultimate limit states. *Ocean Engineering*, 284:115264, 2023.

Z. Alwashah, H. Liu, B. Xiao, S. Mueller, and X. Shao. Generative AI in construction: Emerging trends and use cases. In *Proceedings of the CIB World Building Congress*, Purdue University, IN, USA, jun 2025.

B. Benítez-Suárez, R. Quevedo-Reina, G. M. Álamo, and L. A. Padrón. PSO-based design and optimization of jacket substructures for offshore wind turbines. *Marine Structures*, 101:103759, 2025.

R. Dai, J. Jie, Z. Wang, H. Zheng, and W. Wang. Automated surrogate-assisted particle swarm optimizer with an adaptive parental guidance strategy for expensive engineering optimization problems. *Journal of Computational Design and Engineering*, 12(3):145–183, mar 2025.

P. V. T. Dat, L. Doan, and H. T. T. Binh. HSEvo: Elevating automatic heuristic design with diversity-driven harmony search and genetic algorithm using LLMs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 26931–26938, 2025.

Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, apr 2002.

H. Dong, L. Gao, Q. Lin, and M. Li. Benchmarking evolutionary algorithms on constrained mixed-variable multi-objective optimization problems. *Swarm and Evolutionary Computation*, 86:101511, 2024.

W. B. Feng, Z. Szajnfarber, and J. H. Panchal. Guiding large language models (LLMs) through the engineering design process. In *Proceedings of the AI4SE & SE4AI Workshop*, Washington, DC, USA, sep 2025.

T. D. Kallumadil et al. Large language models for scientific discovery and engineering design. *Nature Reviews Materials*, 9:600–615, 2024.

J. Liu, S. M. H. Adriaenssens, and J. K. G. Generative AI for engineering design: A review. *Computer-Aided Design*, 173:103770, 2024.

J. Liu et al. Generative AI, large language models, and ChatGPT in construction education, training, and practice. *Journal of Construction Engineering and Management*, 151(3):04025001, 2025.

Q. Liu, R. Wang, K. Zhang, and T. Zhang. MOEA/D with dynamic resource allocation for mixed-variable multiobjective optimization problems. *IEEE Transactions on Evolutionary Computation*, 27(5):1436–1450, oct 2023.

K. Min, K. Cho, J. Jang, and S. Kang. REvolution: An evolutionary framework for RTL generation driven by large language models. *arXiv preprint arXiv:2510.21407*, oct 2025. Accepted to ASP-DAC 2026.

V. Savsani, P. Dave, B. D. Raja, and V. Patel. Topology optimization of an offshore jacket structure considering aerodynamic, hydrodynamic and structural forces. *Engineering with Computers*, 37:2911–2930, 2021.

Y. Wang, Z. Wang, Z. Sun, and W. Shi. Multi-objective optimization of offshore wind turbine jacket structures based on surrogate model and NSGA-II. *Renewable Energy*, 216:119028, 2023.

C. Wu, J. Yuan, Z. Lin, C. Li, H. Wu, and X. He. Prompt-MolOpt: Prompt-based multi-objective molecular optimization. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2024.

C. Yang, X. Wang, Y. Lu, H. Liu, Q. V. Le, D. Zhou, and X. Chen. Large language models as optimizers. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2024.

Qingfu Zhang and Hui Li. MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6):712–731, dec 2007.

S. Zheng, C. Li, and Y. Xiao. Efficient optimization design method of jacket structures for offshore wind turbines. *Marine Structures*, 89:103372, 2023.

J. Zhou, Q. Liu, T. Fang, and G. Li. A topology optimization method for metallic flexible disk coupling based on fatigue constraints. *Structural and Multidisciplinary Optimization*, 68(4):112, 2025.