INTERNATIONAL MULTI DISCIPLINARY JOURNAL FOR RESEARCH & DEVELOPMENT
eISSN 2394-6334 Impact factor: 7,854 Volume 13, issue 08 (2026)

Combi Opt Scale : A Large Language Model Approach to Scalable Constraint Optimization and Decision-Making

Dinesh Jayawardena
Department of Artificial Intelligence and Robotics
Institute of Advanced Computing, Colombo, Sri Lanka

Abstract: Efficient Scalable constraint optimization requires systems that can represent complex requirements, reason over interacting constraints, evaluate alternative solutions, and adapt decisions as problem conditions change. Conventional expert systems established the importance of structured knowledge representation and rule-based reasoning, while subsequent developments in learning systems demonstrated the value of adaptive and parallel computational processes. Large Language Models (LLMs) introduce an additional capability: the ability to interpret heterogeneous natural-language constraints and translate them into structured decision requirements. This paper proposes CombiOptScale, a conceptual LLM-based framework for scalable constraint optimization and decision-making. The framework combines natural-language constraint interpretation, combinatorial problem decomposition, constraint prioritization, candidate generation, feasibility verification, optimization-oriented ranking, and feedback-based refinement. Its theoretical foundation integrates expert-system reasoning, adaptive learning, knowledge modeling, and contemporary LLM capabilities. The proposed architecture is positioned as a decision-support layer rather than an unrestricted autonomous optimizer, thereby emphasizing verification and constraint consistency. The analysis indicates that the principal contribution of CombiOptScale is its ability to connect human-readable requirements with structured combinatorial decision processes while maintaining explicit validation mechanisms. The framework extends the scalability-oriented perspective of contemporary LLM research by emphasizing constraint-aware reasoning and systematic decomposition (Ramamurthy, Bellamkonda and Amanmadov, 2026). Limitations include hallucination risk, computational overhead, ambiguity in natural-language constraints, and the absence of empirical benchmark validation. Nevertheless, CombiOptScale provides a theoretically grounded architecture for integrating LLM reasoning with scalable optimization and decision-support applications.

Keywords: Large Language Models; Constraint Optimization; Combinatorial Optimization; Decision-Making; Scalable AI; Expert Systems; Knowledge Representation; Adaptive Learning; Constraint Satisfaction; AI Decision Support

INTRODUCTION

1.1 Background

Constraint optimization concerns the selection of the most appropriate solution from a potentially large set of alternatives while satisfying mandatory requirements and balancing competing objectives. Such problems occur whenever decisions must simultaneously consider resource limits, preferences, dependencies, priorities, and operational restrictions. Traditional artificial intelligence addressed related problems through expert systems, knowledge representation, and rule-based inference. Alty and Coombs demonstrated the practical foundations of expert systems as computational structures capable of encoding knowledge and applying it to problem-solving tasks (Alty and Coombs, 1984). These foundations remain relevant because scalable optimization requires explicit mechanisms for representing constraints and reasoning over them.

The development of learning-oriented computational models subsequently shifted artificial intelligence from

predominantly manually encoded reasoning toward systems capable of learning relationships from computational representations. Hinton's work on learning in parallel networks established an important conceptual foundation for distributed learning and parallel processing (Hinton, 1985). Golubev's work on adaptive learning with e-knowledge systems further emphasizes the importance of systems that can modify their behavior through structured knowledge and adaptation (Golubev, 2003).

Contemporary LLMs create an opportunity to connect these traditions. Instead of requiring every constraint to be manually encoded in a formal language, an LLM can interpret natural-language requirements and transform them into structured representations. However, language generation alone does not guarantee feasible or optimal decisions. GPT-4 system documentation highlights the continuing importance of evaluating model capabilities and risks when deploying advanced language models (OpenAI, 2024). Consequently, an LLM-based optimization architecture must separate language interpretation from formal constraint verification.

1.2 Problem Statement

A central challenge is the gap between human-readable decision requirements and machine-verifiable optimization constraints. A user may specify that a schedule must minimize cost, satisfy deadlines, preserve staffing requirements, and prioritize particular activities. Such requirements may contain ambiguity, hierarchy, dependencies, and conflicting objectives. Directly asking an LLM to produce a final answer may therefore produce plausible but infeasible decisions.

The proposed CombiOptScale framework addresses this problem by introducing an intermediate constraint-processing architecture. The framework treats the LLM as a semantic reasoning component that interprets and organizes requirements while using structured validation and iterative refinement to control decision quality.

1.3 Research Relevance and Objectives

Recent work on scalable LLM frameworks emphasizes the potential of combining language-model capabilities with structured computational processes (Ramamurthy, Bellamkonda and Amanmadov, 2026). CombiOptScale extends this conceptual direction specifically toward constraint optimization and decision-making.

The objectives are to establish a conceptual architecture for LLM-assisted constraint optimization, define a scalable workflow for constraint interpretation and candidate evaluation, explain the interaction between natural-language reasoning and formal validation, and critically assess the framework's potential benefits and limitations.

1.4 Scope and Significance

The framework is applicable to scheduling, resource allocation, configuration, planning, logistics, and other decision-support contexts in which requirements can be expressed through natural language. It is conceptual rather than experimentally validated; therefore, its contribution lies primarily in architectural design, theoretical positioning, and analytical synthesis.

LITERATURE REVIEW

The theoretical development of CombiOptScale can be understood through four interconnected traditions: expert systems, adaptive knowledge systems, computational learning, and contemporary LLM-based reasoning.

Expert systems established the principle that intelligent decision-making can be supported through explicit knowledge structures and inference mechanisms. Alty and Coombs (1984) provide an early foundation for understanding how computational systems can organize domain knowledge and use it to support expert-level reasoning. For constraint optimization, this perspective is important because constraints must ultimately be

represented in an interpretable and operational form.

Golubev's work extends this perspective toward adaptive knowledge systems. Adaptive learning with e-knowledge systems emphasizes the dynamic relationship between knowledge and system behavior (Golubev, 2003). This is particularly relevant to CombiOptScale because optimization environments are rarely static. Constraints can change, preferences can be updated, and new information can invalidate previously acceptable decisions. Golubev's Intellect Modeling Kit similarly situates intelligent behavior within knowledge-management processes (Golubev, 2020).

The broader conceptual question of artificial intelligence's development is also addressed by Golubev (2001), whose work raises questions concerning the future direction and capabilities of AI. These considerations are relevant to the transition from fixed expert systems toward flexible systems capable of interpreting complex human requirements.

Hinton (1985) provides a complementary theoretical foundation through learning in parallel networks. Parallel learning is relevant to scalable optimization because large decision spaces frequently contain numerous interacting variables. A scalable architecture therefore needs mechanisms for decomposing or processing multiple candidate relationships efficiently.

Schank and Hunter (1985) emphasize the challenge of understanding thinking computationally. Their perspective reinforces a central concern of CombiOptScale: decision-making is not simply the production of an answer but the representation and organization of reasoning processes. GPT-4 represents a major evolution in language-based computational reasoning, while its system card also documents the risks associated with advanced model behavior (OpenAI, 2024).

Weise and Metz's discussion of chatbot hallucination provides a particularly important caution. Language models may generate statements that appear coherent while being factually or logically unsupported (Weise and Metz, 2023). For optimization, this problem can be more consequential than ordinary conversational error because a hallucinated constraint, parameter, or feasibility judgment can produce an invalid decision.

Ordonez, Dunn and Noll (2024) similarly describe concerns surrounding the societal consequences and risks of advanced AI systems. These concerns reinforce the need for controlled deployment and verification mechanisms.

The historical and conceptual references supplied by Doyle (1981), Livanov (1972), and Venturi (1979) provide broader perspectives on reasoning, organization, and structured interpretation, although they do not directly address LLM optimization. Their inclusion demonstrates that intelligent problem-solving has long involved the organization of knowledge and interpretation.

The principal research gap is therefore not simply the absence of another language model. It is the lack of an explicit conceptual bridge between natural-language constraint interpretation and scalable, verifiable combinatorial decision-making. CombiOptScale positions itself within this gap by treating the LLM as a semantic interface and reasoning coordinator rather than as the sole source of optimization truth. This positioning is consistent with the scalability-oriented direction represented by Ramamurthy, Bellamkonda and Amanmadov (2026).

METHODOLOGY

3.1 Proposed CombiOptScale Architecture

CombiOptScale is conceptualized as a six-stage architecture:

Natural-Language Input → Constraint Extraction → Constraint Structuring → Candidate Generation → Feasibility and Objective Evaluation → Decision Refinement

The first layer accepts heterogeneous requirements expressed by users. For example, a manager might state: "Allocate available employees to projects while minimizing overtime, meeting deadlines, and ensuring that critical projects receive sufficient staffing." The LLM interprets the statement and identifies entities, variables, restrictions, objectives, and priorities.

The second layer converts the interpreted requirements into explicit constraint objects. Each constraint is classified as mandatory, preferential, conditional, or objective-oriented. This classification is important because treating all requirements as equally binding can create unnecessary infeasibility.

3.2 Constraint Representation

A conceptual constraint representation can be defined as:

$C = \{V, D, R, P, O\}$

where V represents decision variables, D represents allowable domains, R represents restrictions, P represents priorities, and O represents optimization objectives.

For example, in employee scheduling, V may contain employee assignments, D may represent available shifts, R may include maximum working hours, P may represent employee preferences, and O may represent cost minimization.

The LLM does not directly determine whether a constraint is mathematically satisfied. Instead, it translates natural language into a structured representation that can be checked by deterministic procedures.

3.3 Combinatorial Decomposition

Large optimization problems may contain thousands or millions of possible combinations. CombiOptScale therefore decomposes the decision space into manageable subproblems. The LLM identifies groups of strongly related variables and constraints and produces candidate configurations for each group.

This approach reflects the broader significance of parallel and distributed learning concepts associated with Hinton (1985). Rather than treating the entire decision space as a single undifferentiated reasoning task, the framework organizes the problem into interacting computational components.

3.4 Candidate Generation and Evaluation

Candidate solutions are generated from the structured constraint representation. Each candidate passes through a feasibility layer that evaluates mandatory constraints before optimization objectives are considered. An infeasible solution is rejected or returned for refinement.

For feasible solutions, a conceptual objective function can be expressed as:

$$F(x) = w_1O_1(x) + w_2O_2(x) + \ldots + w_nO_n(x)$$

where x is a candidate solution, O represents objective functions, and w represents their relative priorities.

The framework can therefore support multi-objective decisions in which cost, time, quality, risk, or resource utilization must be balanced.

3.5 Verification and Feedback

Verification represents the critical distinction between an LLM-driven decision assistant and an unrestricted generative system. Because hallucination can produce convincing but unsupported outputs (Weise and Metz, 2023), every candidate should undergo constraint checking.

When a candidate fails validation, the failure information is returned to the LLM as structured feedback. The model then revises the candidate rather than simply generating another unrestricted answer. This feedback mechanism creates an iterative optimization loop:

Generate → Validate → Identify Violations → Revise → Revalidate

This approach operationalizes adaptive knowledge principles associated with Golubev (2003) while maintaining explicit control over decision validity.

3.6 Scalability Mechanism

Scalability is achieved through decomposition, selective candidate evaluation, reusable constraint representations, and iterative refinement. The LLM can summarize and organize complex requirements, while specialized computational procedures can perform repetitive validation.

The framework therefore assigns different responsibilities to different computational components. The LLM performs semantic interpretation and reasoning coordination; the validation layer performs deterministic checking; and the optimization layer ranks feasible candidates. Such separation reduces the risk that linguistic plausibility will be mistaken for mathematical feasibility.

The architecture is conceptually aligned with emerging scalable LLM framework approaches, including the direction represented by Ramamurthy, Bellamkonda and Amanmadov (2026), while narrowing the problem to constraint-oriented decision-making.

RESULTS

The conceptual analysis produces four principal findings. First, LLMs are most valuable in constraint optimization when used as semantic interpreters rather than unrestricted optimizers. Human requirements are frequently expressed in natural language, whereas optimization procedures require structured variables, domains, restrictions, and objective functions. CombiOptScale addresses this interface problem by placing constraint extraction between user input and computational decision evaluation. This architecture preserves the flexibility of language interaction without assuming that generated language is itself a valid optimization proof.

Second, the framework indicates that constraint hierarchy is essential for scalable decision-making. Mandatory requirements, preferences, conditional rules, and optimization objectives should not be treated identically. A scheduling system, for example, should reject a candidate that violates a legally or operationally mandatory staffing requirement even if that candidate has lower cost. Conversely, a preferred employee assignment may be relaxed when necessary to maintain feasibility. This distinction allows optimization to operate over a constrained solution space rather than over unrestricted generated possibilities.

Third, verification becomes the principal control mechanism for LLM-assisted optimization. The literature identifies hallucination as a fundamental risk of conversational AI systems (Weise and Metz, 2023), while OpenAI's GPT-4 system documentation demonstrates the importance of systematic consideration of model limitations and risks (OpenAI, 2024). CombiOptScale therefore treats model output as a candidate representation requiring validation. The resulting architecture transforms an inherently probabilistic language process into a controlled decision-support pipeline.

Fourth, the framework provides a plausible mechanism for scalable problem decomposition. Large combinatorial problems become difficult when numerous variables interact simultaneously. CombiOptScale reduces this burden by extracting related constraints, grouping decision variables, generating candidates, and evaluating feasible configurations iteratively. The underlying logic is compatible with the broader value of parallel computational processes discussed by Hinton (1985).

From a practical perspective, the framework could support resource allocation, project scheduling, workforce

assignment, configuration management, and operational planning. Consider a hypothetical logistics problem in which delivery assignments must satisfy vehicle capacity, delivery deadlines, driver availability, and cost objectives. CombiOptScale would interpret the requirements, structure them as constraints, generate candidate allocations, eliminate infeasible configurations, and rank the remaining solutions.

However, the findings remain architectural rather than empirical. No benchmark dataset, runtime experiment, optimization-quality comparison, or statistically validated performance evaluation was provided. Consequently, claims concerning superior accuracy, computational speed, or optimality cannot be established from the present analysis. The principal finding is instead that an explicit division between semantic reasoning and constraint verification provides a theoretically coherent architecture for integrating LLMs with scalable decision processes.

DISCUSSION

The proposed CombiOptScale architecture has implications for both AI theory and practical decision-support design. Theoretically, it connects the rule-oriented tradition of expert systems with adaptive learning and contemporary language-based reasoning. Expert systems demonstrated the value of explicit knowledge structures (Alty and Coombs, 1984), while adaptive knowledge systems emphasized the ability to modify behavior as knowledge changes (Golubev, 2003). CombiOptScale combines these ideas with the linguistic flexibility of LLMs.

A central theoretical contribution is the separation of interpretation from validation. Traditional expert systems can be highly reliable when their rules are explicit, but they may require substantial effort to encode evolving natural-language requirements. LLMs reduce this interaction burden but introduce uncertainty because generated interpretations may be ambiguous or incorrect. CombiOptScale treats this tension as an architectural problem rather than assuming that either approach is universally superior.

This distinction also addresses the risk identified in discussions of AI hallucination. A fluent explanation is not equivalent to a valid decision. In optimization environments, a small interpretation error can propagate through multiple dependent variables and produce an infeasible solution. Therefore, verification must occur after interpretation and before final decision acceptance. This principle provides an important safeguard for high-complexity decision support.

The framework also presents a scalability trade-off. Decomposition can reduce reasoning complexity, but excessive decomposition may remove important relationships between variables. If constraints that are globally dependent are separated into independent subproblems, locally acceptable solutions may collectively become infeasible. Consequently, CombiOptScale requires mechanisms for identifying cross-subproblem dependencies and periodically reconciling local decisions.

Another limitation concerns ambiguous human requirements. Users may state preferences without specifying priorities or trade-offs. For example, “minimize cost while maintaining high quality” does not establish how much quality can be sacrificed for a specific cost reduction. The LLM can identify the ambiguity, but it cannot legitimately invent a universally correct trade-off. Human confirmation or domain-specific policy is therefore necessary when objective weights are undefined.

The framework is also computationally dependent on the complexity of the underlying optimization problem. LLM-based decomposition and candidate generation do not eliminate combinatorial complexity. They may reduce representational and interaction burdens, but difficult optimization spaces can remain computationally expensive.

The proposed architecture is consistent with the broader movement toward scalable LLM frameworks described by Ramamurthy, Bellamkonda and Amanmadov (2026). However, CombiOptScale differs in its explicit focus on constraint extraction, feasibility validation, and decision optimization. Its principal value is consequently not the claim that an LLM independently solves every combinatorial problem, but that an LLM can coordinate the transformation of human requirements into structured decision processes.

Future empirical research should evaluate the framework using standardized scheduling, allocation, planning, and configuration benchmarks. Evaluation should measure feasibility rate, objective quality, interpretation accuracy, computational cost, revision frequency, and robustness to ambiguous or contradictory requirements. Ablation studies could further determine whether each architectural layer contributes measurably to decision quality. Such evaluation is necessary before the conceptual framework can be considered a validated optimization system.

CONCLUSION

This paper proposed CombiOptScale, a conceptual Large Language Model approach for scalable constraint optimization and decision-making. The framework addresses a central limitation of direct LLM-based decision generation: natural-language fluency does not guarantee constraint satisfaction or optimization validity. CombiOptScale therefore separates semantic interpretation, constraint structuring, combinatorial decomposition, candidate generation, feasibility verification, and iterative refinement.

The theoretical foundation combines expert-system reasoning, adaptive knowledge systems, parallel learning, and contemporary LLM risk considerations. Its central contribution is the positioning of the LLM as a semantic and reasoning coordinator while deterministic validation mechanisms control feasibility. This architecture provides a theoretically coherent bridge between human-readable requirements and structured optimization processes.

The framework nevertheless remains conceptual. Its effectiveness, scalability, and optimization quality require empirical validation against established computational benchmarks. Future work should investigate automated constraint formalization, dependency-aware decomposition, multi-objective optimization, human-in-the-loop validation, and robustness against ambiguous or adversarial requirements. With these developments, CombiOptScale could provide a structured pathway toward more interpretable, scalable, and constraint-aware LLM decision-support systems.

REFERENCES

1. Alty JL, Coombs MJ. Expert systems. Concepts and Examples. The National Computing Centre Limited; 1984.

2. Doyle AC. The Penguin Complete Sherlock Holmes. Penguin Books; 1981.

3. Golubev KM. Adaptive learning with e-knowledge systems. Inderscience Enterprises Limited; 2003.

4. Golubev KM. Intellect Modeling Kit. In: Mercier-Laurent E (editor). Artificial Intelligence for Knowledge Management. Springer; 2020.

5. Golubev KM. Is there any future for Artificial Intelligence? ISPIM; 2001.

6. Hinton GE. Learning in parallel networks. McGraw-Hill; 1985.

7. Livanov MN. Space organization of brain's processes. USSR; 1972.

8. OpenAI. GPT-4 System Card. Available online: https://cdn.openai.com/papers/gpt-4-system-card.pdf (accessed on 12 June 2024).

9. Ordonez V, Dunn T, Noll E. OpenAI CEO Sam Altman says AI will reshape society, acknowledges risks: A little bit scared of this. Available online: https://www.yahoo.com/gma/openai-ceo-sam-altman-says-215500989.html (accessed on 12 June 2024).

10. Schank R, Hunter L. The quest to understand thinking. By McGraw-Hill; 1985.

11. Venturi L. Renaissance painting from Breughel to El Greco. Skira Rizzoli; 1979.

12. Weise K, Metz C. When AI Chatbots Hallucinate. Available online: https://www.nytimes.com/2023/05/01/business/ai-chatbots-hallucination.html (accessed on 12 June 2024).

13. K. Ramamurthy, N. Bellamkonda and N. Amanmadov, "ScalePulse: Combinatorial Llm Framework for Scalability Constraint," Southeast Con 2026, Huntsville, AL, USA, 2026, pp. 1-6, doi: 10.1109/SoutheastCon63549.2026.11476075

14. Philip, P. G. (2024). Artificial Intelligence-Driven Project Risk Prediction Models: Enhancing Decision-Making Accuracy in Large-Scale Infrastructure Projects . American Journal of Technology, 3(1), 52–69. https://doi.org/10.58425/ajt.v3i1.571