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

ScaleSolve: An Intelligent LLM-Based Combinatorial Framework for Scalability Constraint Management

Dr. Minh Quang Nguyen
Department of Artificial Intelligence Pacific Institute of Technology, Vietnam

Abstract: The increasing complexity of large-scale computational systems has created a need for intelligent approaches capable of managing heterogeneous constraints while preserving computational scalability. Conventional combinatorial methods provide rigorous optimization mechanisms, but their effectiveness may decline when problem structures become highly dynamic, multidimensional, or difficult to formulate explicitly. This paper proposes ScaleSolve, an intelligent Large Language Model (LLM)-based combinatorial framework for scalability constraint management. The framework conceptually integrates natural-language problem interpretation, constraint extraction, combinatorial problem formulation, candidate-generation mechanisms, adaptive search, feasibility verification, and scalability-aware solution selection. Its theoretical foundation is derived from established work on clustering, data mining, neural computational models, SDN controller placement, node scheduling, privacy-preserving computation, and imbalanced-data learning. The framework further builds on the emerging concept of combinatorial LLM frameworks for scalability constraints discussed by Ramamurthy, Bellamkonda, and Amanmadov (2026). The proposed methodology treats the LLM as an intelligent orchestration and reasoning layer rather than as an unrestricted optimization solver. This distinction enables deterministic validation and search procedures to complement language-based reasoning. The resulting architecture is designed to address constraint prioritization, search-space reduction, dynamic adaptation, and solution explainability. Analytical findings indicate that ScaleSolve offers a coherent architecture for connecting semantic problem understanding with combinatorial optimization while reducing dependence on manually specified solution strategies. The study identifies potential benefits for network management, data-intensive decision systems, clustering, scheduling, and other scalable optimization environments, while recognizing limitations related to model reliability, computational overhead, constraint-verification accuracy, and reproducibility.

Keywords: Large Language Models, Combinatorial Optimization, Scalability Constraints, Constraint Management, Intelligent Frameworks, Adaptive Search, Data Mining, Optimization, Constraint Reasoning

INTRODUCTION

1.1 Background

Large-scale computational environments increasingly require simultaneous management of multiple interdependent constraints. Scheduling, clustering, network configuration, resource allocation, controller placement, and data-intensive decision processes all involve combinatorial search spaces in which a relatively small increase in problem dimensions can produce substantial increases in computational complexity. Classical data-mining research demonstrates that discovering useful structures from large datasets requires systematic computational mechanisms rather than isolated decision rules (Fayyad, Djorgovski, & Weir, 1996). Similarly, clustering research has demonstrated the importance of efficient search strategies for navigating

alternative solution configurations (Al-Sultan, 1995; Bonner, 1964).

The emergence of Large Language Models introduces a different computational capability: semantic interpretation of complex human-defined requirements. An LLM can potentially translate natural-language descriptions into structured constraints, identify relationships among variables, generate alternative reasoning paths, and communicate optimization decisions in an interpretable form. However, language generation alone does not guarantee mathematical feasibility or optimization quality. Consequently, an effective architecture requires a controlled interaction between semantic reasoning and formal combinatorial mechanisms.

The proposed ScaleSolve framework addresses this requirement by positioning an LLM as a reasoning and orchestration layer surrounding formal constraint-processing components. This conceptual direction is consistent with the emerging application of LLMs to combinatorial scalability problems described by Ramamurthy, Bellamkonda, and Amanmadov (2026), while extending the idea toward a broader constraint-management architecture.

1.2 Problem Statement

Traditional combinatorial optimization approaches generally assume that the problem has already been translated into a formal representation. This assumption becomes problematic when requirements are expressed through ambiguous, heterogeneous, or evolving natural-language specifications. Conversely, an LLM can interpret natural-language requirements but may produce inconsistent or unverifiable solutions if it is allowed to perform unconstrained optimization.

The central research problem is therefore the design of an architecture that can combine semantic reasoning with deterministic constraint management. The challenge is not simply to make an LLM generate solutions, but to determine how it can identify constraints, classify their importance, reduce the search space, invoke appropriate combinatorial mechanisms, verify feasibility, and select scalable solutions.

1.3 Research Objectives

The primary objective is to develop a conceptual framework for intelligent scalability constraint management using an LLM-assisted combinatorial architecture. The study specifically aims to establish a structured mechanism for converting natural-language requirements into formal constraints, integrating adaptive search strategies, prioritizing hard and soft constraints, validating candidate solutions, and producing interpretable optimization outputs.

1.4 Scope and Significance

ScaleSolve is applicable to problems involving multiple variables, competing objectives, and scalability requirements. Its conceptual application includes network controller placement, scheduling, clustering, resource allocation, and data-mining workflows. Existing studies on SDN controller placement demonstrate that network optimization can involve multiple interacting criteria and large-scale deployment considerations (Jalili, Keshtgari, Akbari, & Javidan, 2019). The significance of ScaleSolve lies in providing an architectural bridge between such formal optimization problems and natural-language-driven intelligent reasoning.

LITERATURE REVIEW

2.1 Foundations of Combinatorial Search

Clustering represents an important example of combinatorial decision-making because alternative assignments can generate substantially different structural outcomes. Al-Sultan (1995) demonstrated the use of Tabu Search for clustering, illustrating how intelligent local-search mechanisms can navigate complex solution spaces. Bonner (1964) similarly examined clustering techniques, establishing an early computational basis for organizing objects according to similarity.

These studies provide an important theoretical foundation for ScaleSolve: an intelligent framework should not depend exclusively on a single deterministic search procedure. Instead, it should identify the structural characteristics of the problem and select or coordinate an appropriate search mechanism.

2.2 Data Mining and Intelligent Knowledge Discovery

Fayyad, Djorgovski, and Weir (1996) positioned knowledge discovery and data mining as systematic processes for extracting useful patterns from data. Dunham (2002) further established data mining as a structured computational discipline involving multiple analytical tasks. Chakrabarti et al. (2014) emphasized curriculum-level foundations for data mining, reflecting the breadth of techniques required for intelligent data analysis.

Castelo-Branco et al. (2020) demonstrated the relationship between business intelligence, data mining, and retail sales support. This application perspective is relevant because scalable constraint management is rarely isolated from operational decision-making. A framework must ultimately translate computational analysis into decisions that satisfy operational requirements.

2.3 Learning-Based Computational Models

Dave and Dutta (2014) reviewed neural-network-based models for software effort estimation, illustrating the application of learning-based models to complex prediction problems. He and Garcia (2009) investigated learning from imbalanced data, highlighting the difficulties created when data distributions are uneven.

These studies indicate that intelligent systems must account for the characteristics and limitations of their input information. For ScaleSolve, this principle translates into an input-analysis stage that assesses ambiguity, incompleteness, constraint conflicts, and data imbalance before generating candidate solutions.

2.4 Network Scheduling and Optimization

Ashourian, Gheisari, and Talkhoncheh (2015) investigated node scheduling for resilient packet-ring networks. Their work demonstrates the relationship between scheduling decisions and system resilience. Jalili (2019) addressed an SDN-based framework for wireless local-area networks, while Jalili, Keshtgari, and Akbari (2018) examined a set-covering controller-placement problem for large-scale SDNs.

Jalili et al. (2019) extended this research through multicriteria analysis of controller placement. Collectively, these studies demonstrate that network-scale optimization can require simultaneous consideration of coverage, placement, connectivity, and competing criteria. ScaleSolve incorporates this principle by treating scalability as a constraint dimension rather than merely as an after-the-fact performance metric.

2.5 Privacy and Intelligent Constraint Processing

Gheisari, Wang, Bhuiyan, and Zhang (2017) proposed MAPP, a modular arithmetic algorithm for privacy-preserving IoT computation. Their work is relevant to ScaleSolve because scalable intelligent systems may

operate in environments where computational decisions must coexist with privacy constraints.

Gheisari and Esnaashari (2017), through their survey of face-recognition algorithms, also illustrate how algorithm selection requires comparison of advantages and disadvantages rather than assuming universal superiority of one technique. ScaleSolve adopts a similar principle by enabling the reasoning layer to consider alternative optimization mechanisms according to problem characteristics.

2.6 Research Gap and Theoretical Positioning

The reviewed literature demonstrates strong individual capabilities in search, clustering, scheduling, data mining, learning, network optimization, and privacy-preserving computation. However, the studies largely address specific algorithmic problems rather than a unified natural-language-driven architecture for scalable constraint management.

The emerging work of Ramamurthy, Bellamkonda, and Amanmadov (2026) provides a direct conceptual basis for considering an LLM within a combinatorial scalability framework. ScaleSolve extends this direction by defining distinct stages for semantic interpretation, constraint formalization, search orchestration, feasibility verification, scalability assessment, and explanation. Thus, its theoretical position is not that an LLM replaces combinatorial optimization, but that an LLM can coordinate formal computational mechanisms through semantic reasoning.

METHODOLOGY

3.1 Research Design

This study adopts a conceptual framework-development methodology. The framework is synthesized from the functional requirements identified across the provided literature. Rather than claiming experimental superiority without benchmark measurements, the methodology evaluates how the proposed architecture can logically integrate semantic reasoning with established combinatorial principles.

The ScaleSolve architecture consists of six principal layers: input interpretation, constraint extraction, combinatorial formulation, adaptive search, validation, and scalability-aware decision output.

3.2 Input Interpretation Layer

The first layer receives a natural-language problem specification. For example, a user may specify that a network must place controllers while minimizing communication delay, maintaining coverage, satisfying capacity limits, and remaining scalable when additional nodes are introduced.

The LLM interprets the semantic content and identifies entities, variables, objectives, constraints, and contextual assumptions. This mechanism is particularly useful when requirements are not initially expressed as mathematical equations.

However, semantic interpretation must not be treated as equivalent to formal verification. The output of the LLM therefore becomes an intermediate representation rather than the final solution.

3.3 Constraint Extraction and Classification

The second layer converts interpreted requirements into structured constraints. Constraints are categorized into:

- hard constraints that must always be satisfied;
- soft constraints that may be traded against competing objectives;
- scalability constraints that control system behavior as problem size increases;
- resource constraints involving computational, memory, bandwidth, or capacity limits;
- relational constraints defining dependencies among variables.

For example, a controller-placement problem can define controller capacity and node coverage as hard constraints while treating latency minimization as an optimization objective. The multicriteria nature of controller placement identified by Jalili et al. (2019) provides a practical foundation for this distinction.

3.4 Combinatorial Problem Formulation

Once constraints are extracted, ScaleSolve converts them into a formal optimization representation:

$$
\min_{x \in X} F(x)
$$

subject to:

$$
g_i(x) \leq b_i,\quad i=1,\ldots,m
$$

$$
h_j(x)=c_j,\quad j=1,\ldots,k
$$

where (x) represents candidate decisions, (F(x)) represents the objective function, and (g_i) and (h_j) represent inequality and equality constraints.

The LLM does not independently determine whether a candidate satisfies these equations. Instead, it generates or selects a structured formulation that can subsequently be evaluated by a deterministic constraint engine.

This separation is central to the ScaleSolve design. The framework proposed by Ramamurthy, Bellamkonda, and Amanmadov (2026) motivates the integration of LLM reasoning with combinatorial scalability, while ScaleSolve emphasizes an explicit verification boundary between language reasoning and formal optimization.

3.5 Adaptive Search Layer

The adaptive search layer selects an appropriate combinatorial strategy according to the problem structure.

For clustering-oriented problems, Tabu Search principles are relevant because they support systematic exploration of alternative configurations (Al-Sultan, 1995). For network scheduling and placement problems, structured search can incorporate coverage, resilience, and multicriteria requirements (Ashourian, Gheisari, & Talkhoncheh, 2015; Jalili et al., 2019).

The LLM can assist by identifying candidate strategies, generating initial solution structures, explaining why a strategy is appropriate, and adapting search priorities when constraint conflicts are detected. The actual candidate evaluation remains computationally verifiable.

3.6 Constraint Verification

Every generated candidate is passed through a validation mechanism. A candidate solution is accepted only if all mandatory constraints are satisfied. Soft constraints can be represented through weighted penalties:

[

Score(x)=F(x)+\sum_{i=1}^{s}w_iP_i(x)

]

where (P_i(x)) represents the penalty associated with violating or weakening a soft requirement.

This mechanism prevents semantically plausible but mathematically invalid LLM outputs from being treated as valid solutions.

3.7 Scalability Assessment

Scalability is evaluated by considering how solution quality and computational requirements change as problem size increases. A simplified scalability ratio can be represented as:

[

S(n)=\frac{T(n)}{T(n_0)}

]

where (T(n)) represents computational time for a problem of size (n), while (T(n_0)) is the baseline execution time.

A solution that produces excellent results at small scale but becomes computationally impractical at larger scale should not be classified as an optimal scalable solution. This distinction is particularly important for large-scale SDN placement and scheduling problems, where system dimensions directly affect optimization complexity (Jalili, Keshtgari, & Akbari, 2018; Jalili et al., 2019).

3.8 Explanation and Feedback Layer

The final layer converts validated computational results into human-readable explanations. The LLM explains which constraints were satisfied, which objectives dominated the decision, why alternative candidates were rejected, and how scalability affected the final choice.

This function addresses a major practical limitation of purely algorithmic optimization: technically valid

solutions can be difficult for decision-makers to interpret. ScaleSolve therefore treats explanation as an output function rather than as evidence of correctness.

3.9 Hypothetical Operational Example

Consider a large SDN containing hundreds of network nodes. The administrator specifies that controller placement must maintain coverage, respect controller capacity, reduce communication delay, and remain manageable if the network expands.

ScaleSolve first extracts these requirements, categorizes capacity and coverage as hard constraints, and represents latency as an optimization criterion. It then constructs candidate placements and evaluates them using deterministic feasibility checks. A search mechanism explores alternative placements while the scalability layer estimates how the solution behaves when additional nodes are introduced.

The final output may identify a solution that is marginally inferior on one local objective but substantially more stable under network growth. This illustrates the framework's principal concept: scalability is incorporated into the optimization process rather than evaluated only after optimization.

RESULTS

The conceptual analysis produces four principal findings. First, semantic interpretation and formal optimization are complementary rather than interchangeable functions. The literature demonstrates that clustering, scheduling, controller placement, and data mining require structured computational mechanisms (Al-Sultan, 1995; Ashourian, Gheisari, & Talkhoncheh, 2015; Fayyad, Djorgovski, & Weir, 1996). ScaleSolve therefore assigns interpretation to the LLM while preserving formal feasibility checking for the computational layer.

Second, constraint classification substantially improves the organization of complex optimization problems. Distinguishing hard, soft, resource, relational, and scalability constraints allows competing objectives to be handled explicitly. This is particularly relevant to multicriteria network optimization, where alternative controller placements can satisfy different combinations of operational requirements (Jalili et al., 2019).

Third, adaptive search provides a more appropriate architecture than dependence on a single optimization mechanism. The reviewed research includes clustering, scheduling, learning, and network-specific algorithms, indicating that computational requirements vary by problem class (Al-Sultan, 1995; Dave & Dutta, 2014; Jalili, 2019). ScaleSolve consequently uses the LLM to assist with strategy selection while delegating candidate evaluation to formal computational procedures.

Fourth, scalability must be treated as an explicit decision criterion. Existing large-scale SDN research shows that controller placement and coverage problems become increasingly complex as system size grows (Jalili, Keshtgari, & Akbari, 2018). The ScaleSolve architecture therefore incorporates scalability evaluation before final solution selection. This principle aligns directly with the emerging LLM-combinatorial scalability perspective presented by Ramamurthy, Bellamkonda, and Amanmadov (2026).

Overall, the findings indicate that ScaleSolve provides a logically coherent framework for combining semantic reasoning, combinatorial search, constraint verification, and scalability analysis. Nevertheless, these findings are architectural rather than empirical; controlled benchmark experiments would be required to establish quantitative improvements in runtime, solution quality, or scalability.

DISCUSSION

ScaleSolve contributes theoretically by repositioning the LLM from an unrestricted solution generator to an intelligent coordination mechanism. This distinction is important because combinatorial optimization requires mathematical feasibility, while natural-language reasoning primarily provides semantic flexibility. The framework consequently creates a layered architecture in which each computational component performs the function for which it is best suited.

The proposed architecture also provides a unifying interpretation of several strands of the reviewed literature. Tabu-based clustering demonstrates the value of adaptive search (Al-Sultan, 1995), scheduling research demonstrates constraint-dependent decision-making (Ashourian, Gheisari, & Talkhoncheh, 2015), and SDN controller-placement research demonstrates multicriteria optimization at scale (Jalili, Keshtgari, Akbari, & Javidan, 2019). Data-mining research further establishes the importance of systematic knowledge extraction from complex information environments (Dunham, 2002; Fayyad, Djorgovski, & Weir, 1996).

From a practical perspective, ScaleSolve could reduce the gap between human problem descriptions and formal optimization systems. An operations manager does not necessarily need to formulate every constraint mathematically before interacting with the framework. The LLM can transform the initial specification into a structured representation, after which deterministic mechanisms provide computational assurance.

However, this benefit introduces several trade-offs. LLM interpretation can be sensitive to ambiguous language, omitted assumptions, and inconsistent requirements. If an important constraint is incorrectly extracted, subsequent optimization may be mathematically correct but operationally inappropriate. Consequently, the framework requires constraint confirmation, automated validation, and potentially human review for high-impact decisions.

Another limitation concerns computational overhead. Introducing an LLM into an optimization pipeline may increase latency and resource consumption. The architecture therefore should avoid invoking the LLM repeatedly for deterministic calculations that can be handled more efficiently by conventional algorithms. The LLM should primarily operate where semantic reasoning or strategy adaptation adds value.

Data quality represents another limitation. Research on imbalanced data demonstrates that learning systems can encounter substantial difficulties when input distributions are uneven (He & Garcia, 2009). ScaleSolve similarly requires mechanisms for identifying incomplete or biased inputs before treating them as reliable optimization requirements.

Finally, reproducibility remains an important concern. Conventional optimization algorithms can generally be specified through explicit procedures, whereas LLM outputs can vary according to prompts and model behavior. ScaleSolve must therefore preserve structured intermediate representations, validation logs, constraint definitions, and search parameters. The emerging work of Ramamurthy, Bellamkonda, and Amanmadov (2026) reinforces the relevance of this integration between LLM-based reasoning and combinatorial scalability, while the present framework emphasizes stronger separation between semantic generation and formal verification.

CONCLUSION

This paper proposed ScaleSolve, an intelligent LLM-based combinatorial framework for scalability constraint management. The framework integrates semantic problem interpretation, constraint extraction, formal problem formulation, adaptive search, deterministic feasibility validation, scalability assessment, and

explanatory output.

The central contribution is architectural: the LLM is not treated as a replacement for established optimization algorithms but as an intelligent layer capable of translating human requirements into computationally manageable structures and coordinating alternative search mechanisms. This design is supported by principles identified across research on clustering, data mining, scheduling, neural models, privacy-preserving computation, and SDN optimization.

ScaleSolve has potential relevance to large-scale network management, scheduling, resource allocation, clustering, and other multidimensional optimization environments. Its principal limitations involve LLM reliability, computational overhead, data quality, and reproducibility. Future research should therefore implement the framework in benchmark environments and compare LLM-assisted optimization against conventional combinatorial approaches using solution quality, execution time, constraint-violation rate, scalability, and explanation quality as evaluation measures.

REFERENCES

1. Al-Sultan, K. S. (1995). A Tabu search approach to the clustering problem. Pattern Recognition,28, 1443–1451.

2. Ashourian, M., Gheisari, M., & Talk honcheh, A. H. (2015). Animproved node scheduling scheme for resilient packet ring network. Majlesi Journal of Electrical Engineering,9,43–50.

3. Azim, M. A., Abdelhamid, A. A., Badr, N., & Tolba, M. (2019). Largevocabulary Arabic continuous speech recognition using tied states acoustic models . Asian Journal of InformationTechnology,18,49.

4. Bonner, R. (1964). On some clustering techniques.IBM Journal ofResearch and Development,8,22–32.

5. Castelo-Branco, F.,et al.(2020). Business intelligence and datamining to support sales in retail. In Marketing and smart technologies (pp. 406–419). Springer.

6. Chakrabarti, S., Ester, M., Fayyad, U., Gehrke, J., Han, J., Morishita, S., Piatetsky-Shapiro, G., & Wang, W. (2014) Data Mining Curriculum: A Proposal (Version 1.0). ACM SIGKDD. 30April 2006. Retrieved 27 January 2014.

7. Dave, V. S., & Dutta, K. (2014). Neural network-based models for software effort estimation: A review. Artificial IntelligenceReview,42, 295–307.

8. Dunham, M. H. (2002).Data mining introductory and advanced topics, Pearson Education.

9. Fayyad, U., Djorgovski, S. G., & Weir, N. (1996).Advances in knowledge discovery and data mining. MIT Press (pp. 471–494).

10. Gheisari, M., & Esnaashari, M. (2017). A survey to face recognitional gorithms: advantageous and disadvantageous. Journal of Modern Technology and Engineering,2,57–65.

11. Gheisari, M., Wang, G., Bhuiyan, M. Z. A., & Zhang, W. (2017). Mapp:A modular arithmetic algorithm for privacy preserving in IoT. In2017 IEEE international symposium on parallel and distributed processing with applications and 2017 IEEE international conference on ubiquitous

computing and communications (ISPA/IUCC) (pp. 897–903). IEEE.

12. He, H., & Garcia, E. A. (2009). Learning from imbalanced data.IEEE Transactions on Knowledge and Data Engineering,21, 1263–1284.

13. Jalili, A. (2019). A new SDN-based framework for wireless localarea networks. International Journal of Nonlinear Anal ysisand Applications,10, 177–183.

14. Jalili, A., Keshtgari, M., & Akbari, R. (2018). A new set covering controller placement problem model for large scale SDNs.In formation Systems & Telecommunication,25.

15. Jalili, A., Keshtgari, M., Akbari, R., & Javidan, R. (2019). Multicriteria analysis of controller placement problem in software defined networks. Computer Communications,133, 115–128.

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

17. Geo Philip, Paulson, Artificial Intelligence and Machine Learning Applications in Project Schedule Forecasting: A Predictive Framework for Time-Control in Complex Building Projects (April 16, 2026). Available at SSRN: https://ssrn.com/abstract=6588119 or http://dx.doi.org/10.2139/ssrn.6588119