OPEN ACCESS

REPORT

The Innovation Energy

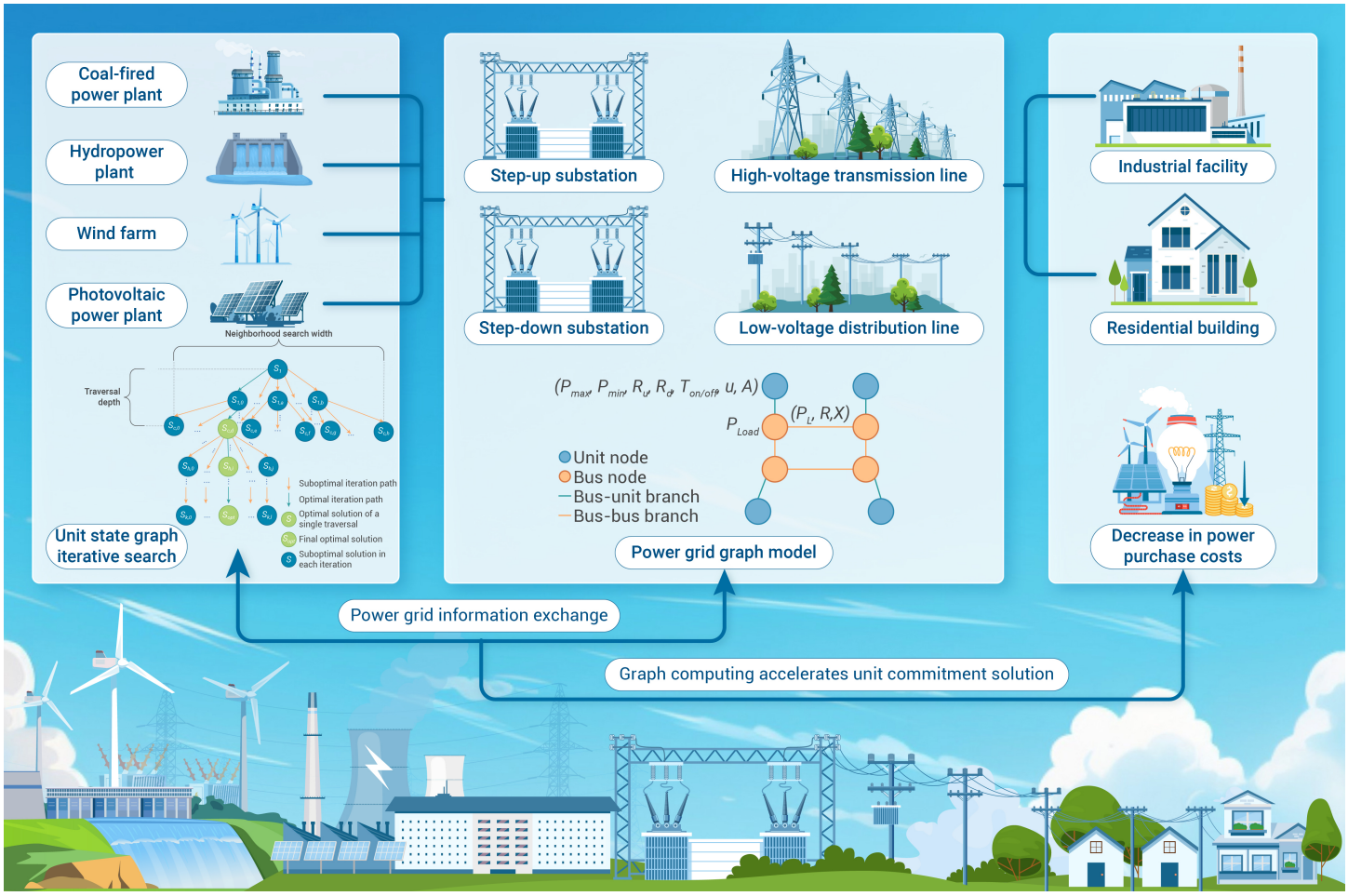
Graph computing technology for ultra-large-scale discrete optimization: A case study on security constrained unit commitment in energy system

Zhenkai Li,[1] Yiyang Cui,[2] Dounan Pan,[2] Jianzhe Liu,[1] and Canbing Li[1,*]

*Correspondence: licanbing@sjtu.edu.cn

Received: January 11, 2025; Accepted: July 8, 2025; Published Online: September 16, 2025; https://doi.org/10.59717/j.xinn-energy.2025.100108

GRAPHICAL ABSTRACT



PUBLIC SUMMARY

- Introduces a graph computing method for security constrained unit commitment (SCUC) in power systems.
- Reduces computation time by up to 92% compared to conventional solvers.
- Uses depth-first traversal and physics-informed edge weights to narrow the solution space.
- Demonstrates high efficiency and solution quality in large-scale energy systems.
- Provides an interpretable, scalable solution for complex mixed-integer programming problems.

REPORT

Graph computing technology for ultra-large-scale discrete optimization: A case study on security constrained unit commitment in energy system

Zhenkai Li,[1] Yiyang Cui,[2] Dounan Pan,[2] Jianzhe Liu,[1] and Canbing Li[1,*]

[1]School of Electrical Engineering, Shanghai Jiao Tong University, Shanghai 200030, China

[2]College of Smart Energy, Shanghai Jiao Tong University, Shanghai 200030, China

*Correspondence: licanbing@sjtu.edu.cn

Received: January 11, 2025; Accepted: July 8, 2025; Published Online: September 16, 2025; https://doi.org/10.59717/j.xinn-energy.2025.100108

Citation: Li Z., Cui Y., Pan D., et al. (2025). Graph computing technology for ultra-large-scale discrete optimization: A case study on security constrained unit commitment in energy system. *The Innovation Energy* **2**:100108.

This paper proposes a graph computing-based method to address the ultra-large-scale discrete optimization problem of security constrained unit commitment in power systems. Feasible unit commitment states are modeled as nodes in a graph, with edge weights defined based on operational priorities and physical information. The search is guided via depth-first traversal and cumulative weight maximization, effectively reducing the number of state solutions to be computed. Meanwhile, graph parallel computing techniques are employed on the grid graph model to accelerate the solution process at each node. Case studies demonstrate that the proposed method significantly reduces computation time by up to 92% compared to conventional solvers, while ensuring solution quality, particularly in large-scale systems. This approach offers an interpretable, efficient, and scalable paradigm for addressing complex mixed integer programming problems in energy systems.

INTRODUCTION

Modern energy power systems face escalating challenges in balancing economic efficiency, operational security, and system reliability. Central to this challenge is the security constrained unit commitment (SCUC) problem, a critical optimization task in energy scheduling that determines the optimal on/off states and power outputs of generating units over a scheduling horizon, subject to stringent constraints (e.g., transmission limits, ramping rates). SCUC directly impacts energy costs, system efficiency, and reliability, yet its computational complexity—classified as an ultra-large-scale nonlinear discrete optimization (ULSNDO) problem—poses significant barriers to real-time implementation in energy systems. For instance, optimizing the Shanghai grid over 96 time intervals involves 526,080 variables (263,040 discrete) and 2,484,132 constraints (60% nonlinear), demanding efficient and robust solution methods tailored to energy system operations.

Existing approaches exhibit notable limitations in addressing ULSNDO problems in energy systems. Classical branch-and-bound (B&B) methods struggle with the "curse of dimensionality", rendering them intractable for large-scale instances. While heuristic and AI-driven techniques (e.g., LSTM,[1] red fox-inspired FOX-RES,[2] E-Seq2Seq[3]) offer improved computational efficiency, they often sacrifice solution quality through local convergence, instability, or poor generalization. Hybrid algorithms combine foundational approaches with domain-specific enhancements. For example, neural network-assisted B&B methods, as proposed in Korekane's work for dynamic berth allocation problems,[4] leverage neural networks to prioritize node exploration in the search tree. Similarly, Hamid Hosseini Dolatabadi's work introduces a heuristic algorithm for optimal PMU placement in power systems,[5] integrating domain knowledge to refine B&B efficiency. Commercial solvers like Gurobi further advance hybrid approaches by combining B&B with presolve routines, cutting planes, heuristics, and parallel computing to accelerate mixed integer programming (MIP) solutions.[6] Gurobi also supports external heuristics or AI-driven initialization to set initial variable values, enhancing convergence speed. While hybrid methods effectively generate high-quality initial solutions and optimize search directions to prune the B&B tree, they remain limited by the exponential growth of search nodes in large-scale problems. GPU-accelerated B&B, though promising for specific combinatorial problems such as differential cluster search in block ciphers[7] and large-scale 0-1 knapsack problems,[8] faces scalability challenges in SCUC due to irregular tree structures, complex MIP models, and memory bottlenecks. These shortcomings hinder the ability of current methods to handle large-scale energy system scenarios efficiently.

Graph computing models systems as nodes (entities) and edges (relationships), excelling in interconnected problems like biological networks.[9~10] However, its application to power system optimization remains unexplored—particularly for SCUC, which lacks inherent graph structures.

To address the instability in solution outcomes and computational inefficiency of existing methods in large-scale energy systems, this paper constructs a graph by connecting the possible unit states of SCUC, and iteratively searches the graph to find the child node with the maximum accumulated weight, using it as the root for further iterations, ultimately obtaining the optimal unit state. The results demonstrate that the proposed method significantly reduces solution time while ensuring accuracy, offering a new approach for solving large-scale 0-1 MIP (MIP where integer variables are restricted to 0 or 1) problems and enabling efficient solutions for SCUC, which directly address critical bottlenecks in energy power system operations and enhance the reliability of large-scale energy management.

MATERIALS AND METHODS

Initial solution search

The SCUC objective function is to minimize the energy purchase cost, so the average quotation ranking list of the units is constructed to determine the startup-shutdown order of the units. The line constraints, minimum startup-shutdown time constraints and startup-shutdown times constraints are ignored, and all units are considered to be on-state, and the lower limit of the units output is relaxed, so that the initial output of the units is optimized. The average quotation of the unit is as follows:

$$A_{i,t} = \frac{\sum_{j=1}^{n_i} p_{i,j,t} q_{i,j}}{\sum_{j=1}^{n_i} p_{i,j,t}} \tag{1}$$

where $A_{i,t}$ is the average quotation of unit i in time interval t, $p_{i,j,t}$ is the power output of unit i at quotation segment j during time interval t, $q_{i,j}$ is the quotation of unit i for quotation segment j, n_i is the number of quotation segments for unit i.

The unit ranking index is defined as follows:

$$Yi,t = c1\frac{Ai,t}{Abase} - c2\frac{\max(p_t - p_{t-1}, 0)}{p_{base}}\frac{Ri,u}{Rbase} - c3\frac{\max(p_{t-1} - p_t, 0)}{p_{base}}\frac{Ri,d}{Rbase} \tag{2}$$

where Yi,t is the ranking index of unit i at time interval t, Ri,u is the ramp-up rate of unit i, Ri,d is the ramp-down rate of unit i, p_t is the total load at time interval t, A_{base} is the base value for quotation, p_{base} is the base value for total load change, R_{base} is the base value for ramp-up/down rates. c_1, c_2 and c_3 are positive weights corresponding to each factor, c_1 governs the influence of the unit's economic cost, while c_2 and c_3 jointly determine the impact of its physi-

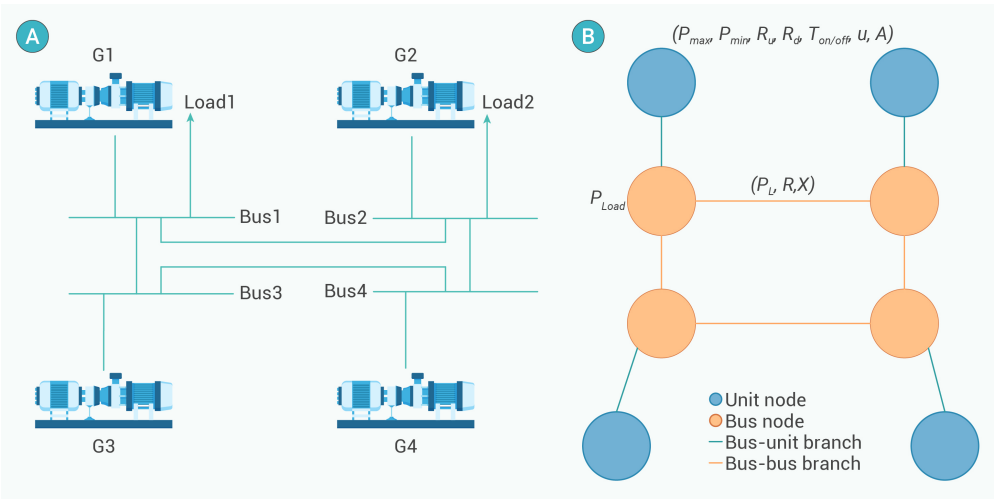


Figure 1. A simple power grid and its graphical model

cal ramping capability relative to the system load level.

For the units whose output is less than the lower limit, the average quotation is regarded as very large. The dynamic priority list of units for each time interval is generated by sorting the unit ranking index Yi,t in ascending order. This index incorporates physics-grounded weighting coefficients (c_1, c_2, c_3) that mitigate operational bias through dynamic cost-flexibility trade-off balancing.

After the initial output distribution, units with output below the lower limit are set to shut down, while the remaining units are set to operate. The predicted value of the unit state does not necessarily meet the constraints, and it needs to be adjusted.

(1) Adjust the unit state based on the must-on and must-off units

In the dynamic priority list of units, a part of the units in the front of the list are selected as must-on units, and a part of the units in the back of the list are selected as must-off units.

(2) Adjust the unit state based on the spinning reserve constraint (Equation 3)

$$\sum_{i=1}^{N_g}(P_{\max,i}u_{i,t}) \geqslant (1+\alpha)P_{L,t} \tag{3}$$

where $P_{max,i}$ is the upper limit of output of unit i, α is the spinning reserve coefficient, $P_{L,t}$ is the total load of time interval t, $u_{i,t}$ is the on/off status of unit i at time interval t (0: off, 1: on), N_g is the number of units. For time intervals that do not satisfy Equation 3, the units are sequentially started according to the dynamic priority list of the units, until all time intervals meet Equation 3.

(3) Adjust the unit state based on the start-up and shut-down count constraint

The unit state can change at most once within a scheduling cycle. Merge the unit state time intervals with shorter duration into the adjacent state time intervals with longer durations. If the unit state durations are the same, the entire time interval's unit state should be modified to the one with the higher proportion of duration within the scheduling cycle.

(4) Adjust the unit state based on the minimum startup-shutdown time constraints

Consider the unit state from the previous scheduling cycle to ensure that the duration of the unit's on and off state in the current scheduling cycle is not less than the minimum on and off time constraint.

(5) Re-verify the unit state against the spinning reserve constraint

If the unit state does not satisfy Equation 3, proceed to step (2) and repeat the process until the verification is successfully passed.

The optimal initial solution is obtained after the unit state prediction value is adjusted by the above five steps.

Graph path search iteration

The power system topology naturally forms a graph structure, with units and buses as nodes. The edges are divided into bus-unit branches and bus-bus branches. The attributes of unit nodes include output limits (P_{max}, P_{min}), ramp-up rate (R_u), ramp-down rate (R_d), minimum startup-shutdown time (T_{on}, T_{off}), and startup/shutdown state (must-on, must-off, or pending) ($\boldsymbol{u}$), average quotation for each time interval ($\boldsymbol{A}$). The attributes of bus nodes represent the bus load for each time interval ($\boldsymbol{P_{Load}}$). Bus-unit branches have no attributes, while bus-bus branches have attributes including maximum allowable flow (P_L) and branch impedance (R, X).

A simple power grid and its graphical model are shown in Figure 1.

After representing the power system using a graph model, each node not only stores physical information but can also act as an independent computational node. After adopting the DC power flow model, the power flow in each branch is linearly related to the unit output and load at each node, as shown in Equation 4. Therefore, the SCUC problem is transformed into a large mixed integer linear programming problem, which further simplifies to a large linear programming (LP) problem when the unit states are known. Using the interior point method as an example, for the LP SCUC problem, each element of the coefficient matrix can be computed in a single step based on the attributes of each node, its neighboring nodes, and branches. The iterative matrix involves the rapid solution of linear equations, with a graph-scheduling-based parallel LU decomposition algorithm performing LU decomposition concurrently at different nodes, thereby accelerating the matrix solution process. This graph-based parallel approach reduces the LP problem-solving time. Subsequent LP problems, given known unit states, are solved using this method.

$$P_{line,i} = \sum_j PTDF_{ij} \cdot P_j \tag{4}$$

where $P_{line,i}$ is the power flow on branch i, $PTDF_{ij}$ is the power transfer distribution factor of node j for branch i, and P_j is the net injection power at bus j.

Based on the unit's operational states in the previous scheduling cycle, along with the minimum startup-shutdown time constraint and the startup-shutdown count constraint, the majority of infeasible solutions can be eliminated. For instance, if a unit has been continuously operating for 1 time interval prior to the current scheduling cycle, with a scheduling cycle consisting of six time intervals and a minimum startup-shutdown time of three time intervals, then the unit's state in the current cycle can only correspond to one of the following five scenarios: 110000, 111000, 111100, 111110, 111111. List all possible unit state solutions, and filter them based on the adjustment criteria from steps (1) to (4) mentioned above, which eliminates most infeasible unit state solutions, resulting in N_s feasible unit state solutions. The solution space of the unit state is shown in Figure 2. In general, unit and system constraints, along with must-on and must-off units, can exponentially eliminate most infeasible states. The former eliminates the most unit states due to its broader condition coverage, followed by the latter, while the unit states excluded based on the previous scheduling cycle are the fewest. Let the unit state solutions be represented as matrices S_1, S_2, ..., S_{Ns}, which can be viewed as individual nodes in a graph model. The distance between them is defined as the L_1 norm, as follows:

$$dist(S_i, S_j) = ||S_i - S_j||_1 \tag{5}$$

where $dist(S_i, S_j)$ is the distance between S_i and S_j. The L1 norm of a matrix is defined as the sum of the absolute values of all its elements.

The optimal initial solution is the root node, with each node representing a unit state solution. Nodes are connected to the nearest 50% of nodes, and within each layer, nodes with higher weight relative to the previous layer should prioritize selecting child nodes, forming the unit state graph.

For example, in the first iteration, 8 unit state solutions closest to the initial solution A are selected from the feasible domain, along with the initial solution, making a total of 9 unit state solutions. These are:

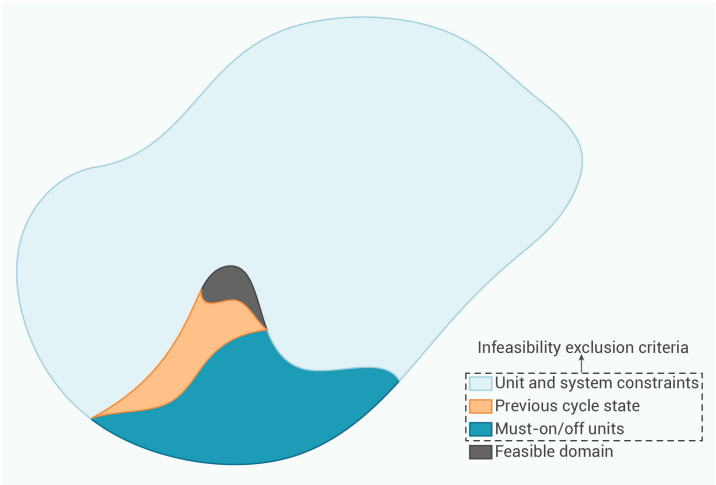


Figure 2. Solution space of the unit

$$A=\begin{bmatrix}0&0&1\\1&1&0\end{bmatrix}\text{ (root node)},\ B=\begin{bmatrix}0&0&1\\1&0&0\end{bmatrix},$$
$$C=\begin{bmatrix}0&0&1\\0&0&0\end{bmatrix},\ D=\begin{bmatrix}1&1&1\\0&0&0\end{bmatrix},\ E=\begin{bmatrix}1&1&1\\0&0&1\end{bmatrix},$$
$$F=\begin{bmatrix}1&1&1\\1&0&0\end{bmatrix}\ G=\begin{bmatrix}1&1&0\\0&0&1\end{bmatrix}\ H=\begin{bmatrix}1&1&0\\0&0&0\end{bmatrix}\ I=\begin{bmatrix}1&1&0\\1&0&0\end{bmatrix}.$$

After the first round of connections, B is the most optimal, followed by C, and then F, their connection is shown in Figure 3.

In the graph path search iteration, the optimal initial solution (root node) is used as the starting point. The edge weights between adjacent nodes are assigned based on the changes in the unit states. If the change involves only shutting down units, the edge weight is increased. If only starting up units is involved, the weight increases according to the priority of the units being started. If both starting and shutting down units are involved, the edge weight is set according to the priority of the corresponding units.

When the unit state changes from S_i to S_j, the edge weight is defined as follows:

$$W_{i,j}=\frac{e^{\alpha\cdot\left(\frac{Y_{new,d}-1}{N_g-1}\right)}}{e^{\beta\cdot\left(\frac{Y_{new,u}-1}{N_g-1}\right)}+e^{\alpha\cdot\left(\frac{Y_{new,d}-1}{N_g-1}\right)}} \tag{6}$$

where W_{ij} is the weight of the edge from S_i to S_j, $Y_{new,u}$ is the priority ranking of the newly started units, $Y_{new,d}$ is the priority ranking of the newly shut down units. Each W_{ij} is not arbitrary heuristics but is explicitly defined based on economically and physically motivated operational priorities. Therefore, the weight W_{ij} quantifies the operational preference (cost and flexibility impact) of transitioning between two physically feasible commitment states. A higher weight signifies a transition deemed more desirable based on these explicitly defined physical and economic factors. This transparent linkage between the search direction decision (edge weight) and the underlying system physics/economics (ramp rates, quotation, load) is a key aspect of the method's interpretability, contrasting with opaque weightings learned by neural networks, while enabling direct future extensions for multi-objective optimization (e.g., emission penalties via augmented priority definitions).

Using W_{ij} to represent the weight restricts it to the range (0,1), making it a dimensionless quantity. This ensures that the increase in W_{ij} when shutting down lower-priority units is faster compared to the increase when starting up higher-priority units.

Set the traversal depth and begin extracting subgraphs from the root node. Perform depth-first search (DFS) on the subgraph, accumulating edge weights along the path. Upon reaching the endpoint, save the path and its total edge weight. Backtrack, remove the current path, and continue searching other paths. Calculate the total edge weight of each child node relative to the root node, which serves as the child node's priority. Compare the highest-priority child node's solution with the root node's solution. If the former is better, set it as the new root and repeat the process; otherwise, increase the search depth until the new highest-priority child node's solution surpasses the root node's, then set it as the new root and continue. This process continues until all nodes are traversed. The iteration stops when either of the following two conditions is met: 1) The difference in the objective function values between the root and node the optimal child node (the new root node) in an iteration is less than a given threshold, and the node with the smaller value is selected as the optimal solution; or 2) All subgraphs have been traversed using DFS, and the node with the smallest objective function value is chosen as the optimal solution. The above process is shown in Figure 4.

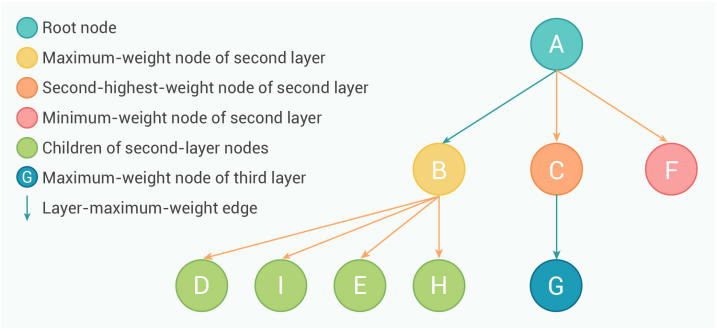


Figure 3. Unit state diagram

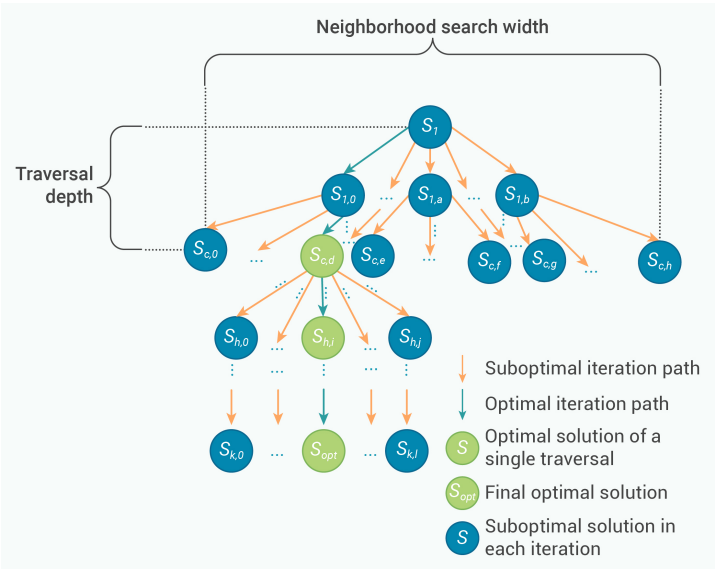


Figure 4. Graph search iterative process

To account for renewable energy impacts, scenario analysis can be employed to characterize renewable generation uncertainty. The initial scenario set S_i^t is constructed based on renewable power forecasts $\hat{y}_t$ obtained from numerical weather prediction and machine learning:

$$S_i^t=\hat{y}_t-\varepsilon_t^{(i)},\quad \varepsilon_t^{(i)}\sim\mathcal{N}(0,\sigma_t^2)\quad (i=1,\ldots,N) \tag{7}$$

where $\hat{Y}_t$ is the forecasted renewable power at time interval t, $\varepsilon_t^{(i)}\sim\mathcal{N}(0,\sigma_t^2)$ is the stochastic forecast error at time interval t of scenario i, σ_t^2 is the forecast error variance at time interval t, N is the total number of scenarios, and S_i^t is the renewable power at time interval t of scenario i.

To reduce computational complexity, K-means clustering compresses scenarios into K representative scenarios R_k (centroid scenario of cluster k), with probabilities assigned proportional to cluster sizes:

$$p_k=\frac{|\Gamma_k|}{N} \tag{8}$$

where $|\Gamma_k|$ is the cardinality of scenario cluster k. The reduced scenario set is denoted by $\Omega=\{R_k\mid k=1,\ldots,K\}$.

Compared to conventional SCUC, the objective function incorporating renewable energy includes a penalty term for renewable curtailment,

Table 1. Comparison of optimization results

number of units	method	cost/$	time/s
250	Gurobi	4.559400877×10^7	230.91
		4.571556729×10^7	30.18
	graph computing	4.571235441×10^7	28.72
		4.571446887×10^7	29.98
	LSTM + Gurobi	4.559784334×10^7	274.61
	Red Fox + Gurobi	4.559695513×10^7	281.59
500	Gurobi	9.399298762×10^7	428.05
		9.458836865×10^7	120.78
	graph computing	9.458507750×10^7	119.56
		9.458719632×10^7	132.86
	LSTM + Gurobi	9.399298762×10^7	587.53
	Red Fox + Gurobi	9.398801165×10^7	452.90
1000	Gurobi	1.9022586280×10^8	3090.31
		1.8997674307×10^8	240.76
	graph computing	1.8996687480×10^8	242.47
		1.8996916092×10^8	251.83
	LSTM + Gurobi	1.9022520202×10^8	3010.35
	Red Fox + Gurobi	1.9022521670×10^8	2530.52

expressed as:

$$\sum_{t=1}^{T}\sum_{k=1}^{N}(p_k\cdot\phi_{kt}) \tag{9}$$

where ϕ_{kt} is the penalty cost for scenario R_k at time interval t.

To ensure multi-scenario security with renewable integration, two-way reserve constraints must satisfy:

$$\begin{aligned}\sum_g r^u_{g,t} &\geqslant \max_i \varepsilon^{(i)}_t\\ \sum_g r^d_{g,t} &\geqslant -\min_i \varepsilon^{(i)}_t\end{aligned} \tag{10}$$

where $r^u_{g,t}$ and $r^d_{g,t}$ are respectively the upward and downward reserve capacities of conventional unit g at time interval t. With revised objectives and constraints, conventional units operate online (lower bounds relaxed) for initial dispatch. Unit priority, edge weight, and graph iteration follow the established procedure. Note that Equation (2) must use net load values. Figure 1 should be revised to include renewable generation units in both the physical grid and graph model, with dedicated graph nodes containing scenario-based output attributes.

RESULTS

Case study

The cases in this paper are adapted from the MATPOWER case_ACTIVSg10k, which is a synthetic network based on the Western Electricity System in the United States. The original model represents a power system with 10000 buses, 1474 conventional units, 1011 renewable units, and 12706 branches. By consolidating conventional and renewable units separately at the same nodes and removing small units, while preserving the network topology, test cases are created with 10000 buses, 12706 branches, and three unit scales (250, 500, 1000) to simulate future Chinese provincial market expansion and benchmark ultra-large-scale performance (currently, approximately 200 units participate in market clearing in a typical Chinese provincial grid). The unit and load parameters are locally-calibrated using Shanghai grid data to align with domestic market units. Each case is solved

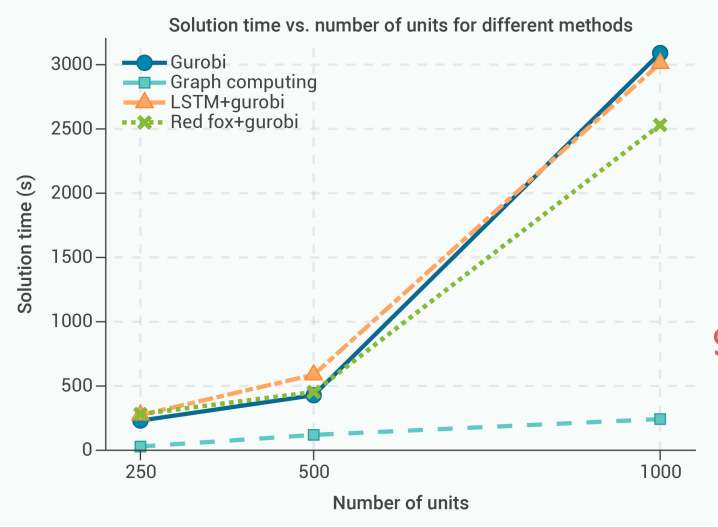


Figure 5. Relationship between solution time and number of units

using graph computing(①c_1=0.6, c_2=0.8, c_3=0.8 ②c_1=1, c_2=0.45, c_3=0.45 ③c_1=1, c_2=0.6, c_3=0.4), Gurobi, LSTM + Gurobi and Red Fox + Gurobi to obtain the solution and computation time for each method. For each case, the results of the three graph computing implementations (①,②,③) are presented sequentially from top to bottom in the Table 1. Gurobi first uses presolve techniques to simplify the problem. Then, it constructs a search tree through B&B. During the branching process, cutting planes are added to eliminate non-integer solutions, and heuristics are employed to quickly find feasible solutions. LSTM excels at handling sequential data and capturing long-term dependencies, making it suitable for predicting current unit state solution from load vectors by leveraging patterns in historical data. The prediction can serve as initial solution for Gurobi. The red fox algorithm is a novel optimization algorithm that mimics red fox behavior, combines global and local search, and uses a unique population control to handle complex constraints. It generates initial SCUC solutions for Gurobi.

The results calculated by different methods are as follows:

The results show that for systems with 250, 500, and 1000 units, the minimum operational costs were achieved by Gurobi, Red Fox + Gurobi, and graph computing, respectively. Table 1 demonstrates that for each case, the optimal solutions obtained via graph computing under the three parameter configurations (①, ②, ③) exhibit minimal variation. Computational times remain comparable, with maximum-cost solutions exceeding minimum-cost solutions by merely 0.007%, 0.0035%, and 0.0052% respectively. This indicates low parameter sensitivity and high solution stability. While the graph computing method yielded slightly higher costs than the optimal solutions in smaller-scale systems (0.26% and 0.64% above the minimum for 250 and 500 units), it outperformed all other methods in the 1000-unit system, reducing costs by approximately 0.14%. Notably, graph computing consistently exhibited the shortest computation time across all scales, with reductions of 87.56%, 89.54%, and 89.80% (250 units); 72.07%, 79.65%, and 73.60% (500 units); and 92.15%, 91.95%, and 90.42% (1000 units) compared to Gurobi, LSTM + Gurobi, and Red Fox + Gurobi, respectively. Its computational time scales polynomially with unit count, rising polynomially in contrast to Gurobi's exponential-like escalation. Figures 5 illustrates the computation time of all methods across varying unit scales.

As shown in Figure 5, the computation time of the proposed graph computing method scales approximately linearly with the increasing number of units, whereas Gurobi, LSTM + Gurobi, and Red Fox + Gurobi exhibit exponential time growth. Notably, the computation time of LSTM + Gurobi and Red Fox + Gurobi may intermittently exceed or fall below that of Gurobi across different scales. In contrast, the graph computing method consistently achieves the lowest computation time, demonstrating robust stability.

The higher costs for smaller systems (0.26% for 250 units, 0.64% for 500 units) represent a deliberate efficiency-cost trade-off, operationally justified by:

(1) During corrective intraday scheduling of day-ahead schedules, rapid SCUC resolution facilitates multiple hourly reschedules, curtailing renewable curtailment/reserve costs from net-load fluctuations while mitigating security risks and penalties of delayed scheduling, yielding net system cost reductions.

(2) The ≤0.64% cost deviation falls within the ≤1% tolerance threshold defined by the U.S. Department of Energy Advanced Research Projects Agency-Energy funded Grid Optimization Competition Challenge 3.[11]

CONCLUSION

Ultra-large-scale, nonlinear, and discrete optimization problem is a key challenge of modern energy engineering. The SCUC in electric power systems constitutes a quintessential instance of such problems, serving as a critical energy scheduling task for grid operations. In this paper, unit state diagram DFS and graph parallel algorithm for unit output optimization—both grounded in a physically interpretable graph computing framework—are proposed. By eliminating infeasible unit states and narrowing the feasible domain, the depth-first search method identifies the optimal iteration direction. The graph parallel algorithm solves each unit state scenario during the iteration process. Compared to traditional B&B methods, the proposed method introduces a physics-aware edge weight-guided search direction during node exploration, where edge weights are parametrically derived from unit operational constraints and system states, demonstrating characteristics of heuristic-informed search with explicit physical grounding. When solving power allocation for each node, it employs graph node parallel computing technology that fully exploits physical relationships in power grid topology to accelerate computation, embodying AI's capability of mining global physical information patterns. The proposed method not only reduces both the solution time and the rate at which it increases with problem size, but also yields better solutions for large-scale problems compared to conventional methods while maintaining interpretability through direct mapping of physical constraints. This provides a new solution paradigm for large-scale 0-1 MIP problems, particularly in energy and power systems like SCUC, by effectively integrating the strong generalization capabilities of AI with the computational efficiency of heuristic algorithms. This is achieved through the synergistic implementation of physics-guided search strategies and topology-aware parallel processing mechanisms, offering enhanced computational feasibility for these demanding large-scale energy system applications.

However, practical deployment requires addressing grid-to-graph mapping, real-time data synchronization; future work will develop adaptive grid-graph libraries with native support for multi-objective edge weights, including carbon-emission-aware optimization.

REFERENCES

1. Ramesh A. V. and Li X. (2024). Spatio-Temporal Deep Learning-Assisted Reduced Security-Constrained Unit Commitment. *IEEE Trans. Power Sys.* **2**:4735–4746. DOI:10.1109/TPWRS.2023.3313430
2. Kamboj V. K. and Malik O. P. (2023). Optimal Unit Commitment of Integrated Power System with Demand of Medical Oxygen and Renewable Energy Sources. *IEEE SMART* **12**:529–535. DOI:10.1109/SMART59791.2023.10428678
3. Yang N., Yang C., Wu L., et al. (2022). Intelligent Data-Driven Decision-Making Method for Dynamic Multisequence: An E-Seq2Seq-Based SCUC Expert System. *IEEE Trans. Ind. Inform.* **5**:3126–3137. DOI:10.1109/TII.2021.3107406
4. Korekane S. and Nishi T. (2021). Neural Network Assisted Branch-and-Bound Method for Dynamic Berth Allocation Problems. *IEEE SMC* :208-213. DOI: 10.1109/SMC52423.2021.9658903.
5. Hamid Hosseini Dolatabadi S., Bhuiyan T. H. and Golshan M. E. H. (2024) . A Heuristic Solution Algorithm for a Comprehensive Optimal Phasor Measurement Unit Placement Considering Zero-Injection Buses and Practical Constraints in Power System State Observability Problem. *IEEE Access* :79919-79936. DOI: 10.1109/ACCESS.2024.3409651.
6. Gurobi Optimization. (2025). Mixed-Integer Programming (MIP) – A Primer on the Basics. https://www.gurobi.com/resources/mixed-integer-programming-mip-a-primer-on-the-basics/.
7. Yeoh W.Z., Teh J. S. and Chen J. (2020). Automated Search for Block Cipher Differentials: A GPU-Accelerated Branch-and-Bound Algorithm. *Information Security and Privacy. ACISP* **2020**:7453–7456. DOI:10.1007/978-3-030-55304-3_9
8. Shen J., Shigeoka K., Ino F., et al. (2017). An Out-of-Core Branch and Bound Method for Solving the 0-1 Knapsack Problem on a GPU. Algorithms and Architectures for Parallel Processing. *ICA3PP* :254-267. DOI: 10.1007/978-3-319-65482-9_17.
9. Han J., Li T., He Y., et al. (2024). Predicting the effect of chemicals on fruit using graph neural networks. *Sci. Rep.* **14**:8203. DOI:10.1038/s41598-024-58991-y
10. Schattgen S.A., Guion K., Crawford J.C., et al. (2022). Integrating T cell receptor sequences and transcriptional profiles by clonotype neighbor graph analysis (CoNGA). *Nat. Biotechnol.* **40**:54–63. DOI:10.1038/s41587-021-00989-2
11. HHolzer J.T., Coffrin C.J., DeMarco C., et al. (2024). Grid Optimization Competition Challenge 3 Problem Formulation. *Richland, WA: Pacific Northwest National Laboratory.* https://www.pnnl.gov/publications/grid-optimization-competition-challenge-3-problem-formulation

FUNDING AND ACKNOWLEDGMENTS

This work was funded by the National Key R&D Program of China (No. 2022YFB2404200). The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

AUTHOR CONTRIBUTIONS

All authors contributed to the manuscript and approved the final version.

DECLARATION OF INTERESTS

The authors declare no competing interests.

DATA AND MATERIALS AVAILABILITY

Data are available from the corresponding author upon reasonable request.