Leveraging Large Language Model Based Agent for Automated Electricity Market Modelling and Simulation

Yuheng Cheng, Wenxuan Liu, *Member*, *IEEE*, Yusheng Xue, *Life Member*, *IEEE*, Jie Huang, *Member*, *IEEE*, Junhua Zhao, *Senior Member*, *IEEE*, and Fushuan Wen, *Fellow*, *IEEE*

***Abstract*—An electricity market is a complex, dynamically operated network encompassing multiple participants under defined rules, thereby ensuring real-time supply-demand balance and system reliability. However, the inherent complexity and dynamism of the electricity market pose significant challenges to conventional modelling approaches, which often rely on expert knowledge and manual processes informed by market regulations. This reliance frequently leads to inefficiencies and elevated risks of error. To address these limitations, this paper proposes a framework for automated electricity market modelling and simulation centered on a large language model based agent, termed the modelling and simulation system agent (MSS-Agent) framework. The proposed MSS-Agent framework employs the hierarchical chain-of-thought (HCoT) method to more accurately extract essential information from relevant documents, thereby enhancing modelling fidelity. Moreover, it integrates tool usage and reflexive debugging to optimize the code generation process, ensuring reliability in automated electricity market modelling and simulation. Experimental results demonstrate that the proposed MSS-Agent framework significantly improves both mathematical model extraction accuracy and code execution reliability. Consequently, the proposed MSS-Agent framework not only increases simulation efficiency but also provides more precise and dependable tools for informed decision-making in electricity markets.**

***Index Terms*—Electricity market, large language model, artificial intelligence (AI), chain-of-thought, modelling and simulation system, agent.**

I. Introduction

THE electricity market is a complex and dynamically operated network involving multiple participants governed by defined rules to ensure real-time supply-demand balance and system reliability [1]. The electricity market involves a diverse array of participants, including generation companies, grid operators, regulators, and consumers, which interact through competitive bidding, dispatch planning, and regulatory oversight [2], [3]. Given the intricate operation of the electricity market, a large amount of critical information, including evolving operational details, is recorded in unstructured textual formats such as market reports, policy documents, and research papers [4]. The reliance on unstructured textual information for essential information has posed challenges to the automated modelling and solving of the electricity market.

In this context, timely analysis and decision-making are of paramount importance for both regulators and market participants. The electricity market environment is characterized by rapid fluctuations and complex interactions, demanding real-time information processing for effective operation. However, traditional causal modelling methods, often relying on manual analysis and expert knowledge, face significant limitations.

First, these methods typically focus on representative scenarios due to the sheer complexity and dynamism of the electricity market, leading to potential systemic errors as critical, yet less frequent, events might be overlooked. For instance, a study that models an integrated electricity and carbon market using a weekly representative load profile [5] may capture the average weekly behavior but miss critical information such as negative prices resulting from excessive renewable energy generation. This oversight could lead to systemic errors in overall market regulation. From a trading perspective, the inability to perform automated, real-time, and precise electricity market modelling means that trading decisions are unlikely to be locally or globally optimal. Workarounds like rolling-horizon modelling [6] achieve optimality only within pre-defined, typical time periods, leaving the actual deviation from assumed conditions unassessed in real time.

Second, while machine learning (ML) methods such as

Manuscript received: July 11, 2025; revised: September 4, 2025; accepted: October 27, 2025. Date of CrossCheck: October 27, 2025. Date of online publication: November 25, 2025.

This work was supported by Basic Research Program of Jiangsu (No. BK20232026).

Y. Cheng, W. Liu, J. Zhao, and F. Wen are with the School of Electrical Engineering, Zhejiang University, Hangzhou 310027, China, and Y. Cheng, W. Liu, and J. Zhao are also with the School of Science and Engineering, The Chinese University of Hong Kong (Shenzhen), Shenzhen 518100, China (e-mail: yuhengcheng@link. cuhk. edu. cn; wenxuanliu@link. cuhk. edu. cn; zhaojunhua@cuhk.edu.cn; fushuan.wen@gmail.com).

Y. Xue (corresponding author) is with State Grid Electric Power Research Institute (NARI Group Corporation), Nanjing 211106, China (e-mail: xueyusheng@sgepri.sgcc.com.cn).

J. Huang is with NARI Technology Co., Ltd., Nanjing 211106, China (e-mail: huangjie1@sgepri.sgcc.com.cn).

DOI: 10.35833/MPCE.2025.000639

neural networks and support vector machines show promise in handling large datasets, they often suffer from a critical lack of interpretability, making it challenging to deploy them confidently in crucial power system planning and decision-making [7]. This limitation hinders their widespread adoption in tasks that directly rely on understanding the underlying reasoning from unstructured textual data.

In recent years, the rapid development of the large language model (LLM) [8] has brought new possibilities for addressing these issues. LLMs have demonstrated unprecedented capabilities in natural language processing (NLP), knowledge reasoning, and contextual understanding. Currently, advanced LLMs, represented by generative pre-trained transformer 4 (GPT-4) [9], have shown remarkable performance in various fields, including text generation, question-answering systems, and code writing. These models can understand and generate natural language, enabling them to directly interact with and comprehend research papers and generate corresponding mathematical models and code based on the content of papers. However, directly using LLMs for electricity market modelling still faces three main challenges. First, LLMs struggle with complex reasoning when dealing with long text inputs, often overlooking some critical information. Second, LLMs suffer from the "hallucination" problem [10], where they may generate seemingly plausible but inaccurate or entirely fabricated information. Although recent research such as constitutional artificial intelligence (AI) [11] and reinforcement learning from human feedback [12] has partially mitigated this issue, the unpredictability remains unacceptable in high-reliability environments such as power systems, where even a minor error can have catastrophic consequences. Additionally, LLMs can generate code, but they themselves cannot execute it. They interact with the feedback from code compilers to optimize the generated code continuously.

To overcome these challenges, LLM-based agents [13] offer a viable solution. LLM-based agents combine the language understanding and reasoning capabilities of LLMs with the interaction of external tools and environments, retaining the advantages of LLMs in processing natural language while enhancing the system reliability and practicality. Agents can optimize their behavior and output through a closed-loop process of planning, execution, observation, and adjustment. This approach can help LLMs better understand and extract key information from long texts and verify and improve the generated code through real-time interaction with the programming environment, significantly improving the system reliability and accuracy.

This paper proposes a framework for automated electricity market modelling and simulation centered on an LLM-based agent, termed the modelling and simulation system agent (MSS-Agent) framework. The proposed MSS-Agent framework employs a hierarchical chain-of-thought (HCoT) technique to decompose complex electricity market modelling tasks, gradually extracting and verifying key information from the text through multiple levels of reasoning steps. Simultaneously, it can utilize programming tools to execute the generated code and reflect and debug based on the execution results, ensuring the accuracy and reliability of the modelling results. The proposed MSS-Agent framework not only improves the accuracy of information extraction but also achieves an end-to-end automated process from text understanding to code implementation. By combining the language understanding capabilities of LLMs with practical tools, the proposed MSS-Agent framework provides a more intelligent and efficient solution for electricity market modelling and simulation.

The main contributions of this paper are as follows.

1) Proposing the MSS-Agent framework to automate the end-to-end process of electricity market modelling and simulation from unstructured text, overcoming the inefficiencies of manual and expert-driven approaches.

2) Employing an HCoT technique to guide LLMs for information extraction of electricity market modelling to improve accuracy and efficiency, which is crucial for explainable modelling.

3) Integrating reflection and debugging to optimize code generation and execution, which enhances simulation success rates and, importantly, provides transparency and traceability for improved interpretability of model-derived decisions.

The significance of this paper lies in improving the efficiency of electricity market information extraction and analysis, which provides more accurate and timely decision support for market participants. By automating electricity market simulations, we can mitigate human error and respond more quickly to market changes, thereby empowering decision-makers to make more informed choices in complex market environments.

The structure of this paper is as follows: Section II reviews the related studies. Section III describes the methodology. Section IV presents the experiments. Section V presents the discussions. Finally, Section VI concludes the paper and outlines future research directions.

II. Related Studies

A. Electricity Market Modelling and Simulation

Electricity market modelling and simulation constitute a crucial area of research within modern energy economics. Its significance is ever-increasing with the proliferation of renewable energy sources and the growing complexity of electricity market. In recent years, numerous studies have focused on exploring various factors and their interactions within electricity market, including renewable energy integration, multi-period planning, competition mechanism, and market efficiency. For instance, advanced techniques such as game theory are employed to model the strategic interactions among market participants [14], while evolutionary game theory can be used to analyze long-term bidding strategies [15]. In [16], a model based on fuzzy Q-learning is proposed, which improves the efficiency of day-ahead electricity market modelling by reducing the number of iterations and increasing the probability of Nash equilibrium. This model not only effectively addresses the uncertainties introduced by renewable energy but also offers more optimal strategy selections in complex market environments. In [17],

a hybrid experimental learning approach is proposed, which combines ML and experimental economics to model trading behavior in the electricity market. Through this hybrid approach, the trader behavior can be effectively explained and predicted, thereby offering valuable decision support to electricity market participants.

Furthermore, specific issues in the electricity market, such as the effect of delay, impact of macro-climatic events, and role of energy storage systems, have also been investigated. For instance, [18] examines the impact of wind power forecast uncertainty on the electricity market, while [19] analyzes the role of energy storage systems in the electricity market. However, most existing methods rely on structured data and require experts to manually design models and adjust parameters. Consequently, they struggle to directly process large amounts of unstructured textual information in the electricity market and have difficulty in responding rapidly to changes in complex market rules. Although [20] utilizes LLMs and market sentiment agents to assist in day-ahead electricity price forecasting based on news text, such methods are often tailored to specific types of text and lack generality and automation capabilities. Therefore, achieving automation in electricity market modelling and simulation, as well as efficiently extracting modelling information from unstructured text, has become an urgent problem.

B. LLM-based Agent and Code Generation

In recent years, LLMs have made groundbreaking advancements in the field of NLP. Transformer [21], abandoning the traditional recurrent and convolutional neural network structures, employs a self-attention mechanism, which enables parallel computation and effectively captures long-range dependencies. Building upon this, the GPT series of models [22] have demonstrated powerful zero-shot and few-shot learning capabilities, with GPT-4 achieving near-human level performance on various NLP tasks. Furthermore, to enhance the reasoning capabilities of LLMs, chain-of-thought (CoT) prompting [23] significantly improves model performance on complex reasoning tasks by guiding the model to generate intermediate reasoning steps. For example, CoT can enhance the effectiveness of code generation tasks [24]. These studies have laid a solid foundation for building LLM-based agents.

Leveraging the robust language understanding and reasoning capabilities of LLMs, researchers have begun to explore the development of LLM-based agents, enabling them to interact with external tools and explore the environment to achieve specific goals. Among these, ReAct [25] combines the reasoning and action capabilities of LLMs, allowing the LLM to reason based on current goals and environmental information, generate a plan for the following action, and then use external tools to execute the action. The process iterates until the task is completed. MetaGPT [26] explores a multi-agent collaboration framework, automating complex software development tasks by simulating agents in different roles within a software company. However, while standard CoT improves reasoning, its single and linear thought process can be less effective for highly structured and multi-component tasks like mathematical model extraction from long texts, where maintaining context across different model elements (e. g., objective, constraints, variables) is a significant challenge. Also, although LLMs have shown some capability in code generation, the code generated directly often suffers from syntax errors, logical flaws, or deviations from requirements, necessitating further optimization and verification.

In code generation for specialized domains, relying solely on code generated by LLMs is insufficient to meet practical needs. More effective mechanisms are required to improve code quality. As an important optimization strategy, reflection mechanisms [27], where an agent critically evaluates its own output and iteratively refines it, have received increasing attention in recent years. The reflexion mechanism [28] improves code generation quality by having the LLM self-evaluate and reflect on the generated code and then iteratively optimize based on the evaluation results. Building on this, Self-Refine [29] utilizes the LLM to generate feedback and then iteratively improves the code based on the feedback, further enhancing code reliability.

In high-reliability domains such as electricity market, code must not only be correct in terms of syntax and logic but also ensure that the underlying mathematical model is accurate and can adapt to the complex operating rules and constraints of the electricity market. Exploration in this area is still in its early stages [30], [31]. A crucial direction for future research is to design effective reflection mechanisms that enable LLMs to understand the professional knowledge and operating mechanisms of the electricity market and to generate reliable code that meets practical needs.

III. Methodology

This section proposes the MSS-Agent framework. We aim to develop a framework capable of automatically extracting electricity market information and generating executable models. However, achieving this goal faces several major challenges. Firstly, electricity market description documents often contain a large number of specialized terms, complex mathematical expressions, and background information unrelated to modelling, requiring an accurate understanding and extraction of key information. Secondly, the process of converting unstructured text into structured mathematical models is complex, requiring the assurance of completeness and accuracy in information extraction. This is a task where standard and linear CoT prompting struggles, as it needs to track multiple distinct model components simultaneously. Finally, when translating mathematical models into executable code, it is necessary to handle various implementation details and potential errors, such as choosing appropriate data structures or correcting logical flaws in constraint formulations.

To address these challenges, the proposed MSS-Agent framework is designed with a hierarchical chain-of-thought information extraction (HCoT-IE) method, enhanced by Auto-Prompt optimization, to organize and extract electricity market information. Furthermore, the framework is combined with code templates and a reflection debugging mechanism, where the agent iteratively corrects its own code based on execution feedback, to achieve automatic model construction

and optimization. This section presents the overall structure of the proposed MSS-Agent framework and details its two key components: HCoT-IE for electricity market modelling and reflective coding for electricity market simulation.

A. Overall Structure of Proposed MSS-Agent Framework

As illustrated in Fig. 1, the proposed MSS-Agent framework consists of two key components. In the HCoT-IE for electricity market modelling, the proposed MSS-Agent framework first analyzes the electricity market description document and uses HCoT to reason about the key elements that need to be extracted from the document, such as the objective function, constraints, and market model parameters. The extracted information is transformed into structured data and used to construct a unified representation of the mathematical model. To achieve efficient information extraction, we designed an HCoT-IE method that enables the LLM to reason and organize the information needed for electricity market modelling in a hierarchical manner. Subsequently, the proposed MSS-Agent framework will sequentially extract the corresponding information from the long text, forming a complete mathematical model of the electricity market.

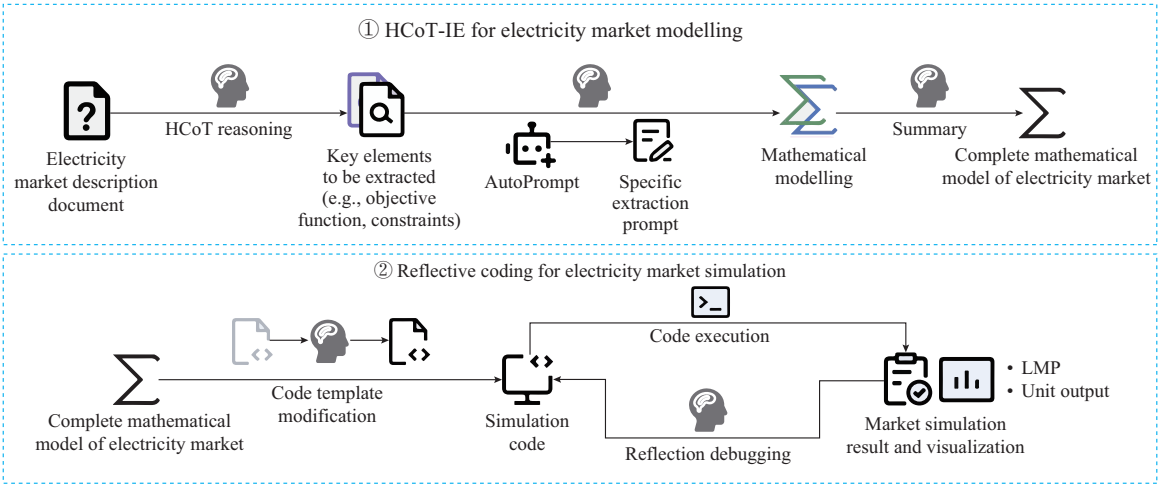

Fig. 1. Overall structure of proposed MSS-Agent framework that illustrates process from document analysis to simulation code validation.

In the reflective coding for electricity market simulation, the proposed MSS-Agent framework utilizes the code generation and modification capabilities of LLMs to modify code templates into corresponding executable simulation code based on the code templates and the extracted mathematical modelling information of the electricity market. Next, the proposed MSS-Agent framework will execute the simulation code in a compilation environment and generate corresponding electricity market simulation results such as locational marginal price (LMP) and unit output. To improve the effectiveness of the modelling and simulation, the proposed MSS-Agent framework incorporates a reflection mechanism. This process is iterative: if the code execution fails, error feedback triggers a reflection debugging step. The MSS-Agent analyzes the error, compares the flawed code against the original mathematical model, and produces a revised version for the next attempt. This cyclical process of execution, reflection, and debugging continues until the code runs successfully, ensuring a robust and reliable final simulation program.

The core idea of the entire framework is to leverage the reasoning and code generation capabilities of LLMs to achieve automation and intelligence in market model construction, significantly reducing the cost and difficulty of building electricity market models.

B. HCoT-IE for Electricity Market Modelling

Extracting accurate electricity market information from documents is a critical challenge in the process of electricity market modelling. This challenge manifests in three main aspects. First, electricity market description documents are often lengthy and contain a large amount of non-critical information, requiring accurate identification and extraction of core elements necessary for modelling. Second, electricity market information is often scattered throughout the document in various forms, necessitating effective integration of these fragmented pieces of information. Finally, the extracted information needs to be transformed into a standard mathematical model format, which demands a deep understanding of electricity market mechanisms. To address these challenges, the HCoT-IE method is proposed, as shown in Fig. 2. HCoT-IE decomposes the information extraction process into multiple elements to be extracted and iteratively extracts them from the original document.

First, the electricity market description document is taken as the input, serving as the initial information source $\mathcal{D}$ for HCoT-IE. Generally, the description document can be a research paper or a policy document on the electricity market. These documents are first preprocessed using PDF parsing tools like MinerU to convert the PDF document $D \in \mathcal{D}$ into an editable Markdown format along with LaTeX formulas. It is important to note that the effectiveness of this initial step, and consequently the entire HCoT-IE process, is contingent on the quality of the source document and the ability of parsing tools to handle complex layouts and formulas. Next, HCoT-IE method decomposes the information extraction process of electricity market modelling into several phases.

1) Reasoning Phase

Initially, based on the parsed document content $D_{markdown}$, HCoT calls an LLM to perform reasoning. This process can be represented as:

$$E = LLM_{reason}(D_{markdown}, P_{reason}) \tag{1}$$

where $E = \{e_1, e_2, ..., e_n\}$ is the set of elements to be extracted; $LLM_{reason}(\cdot)$ denotes the LLM used in reasoning phase; and P_{reason} denotes a predefined reasoning prompt that guides the LLM to perform CoT reasoning. Through this process, the LLM can consider the document content and determine which elements need to be extracted for modelling, such as the objective function, constraint, market model parameter, and price limits. In this phase, we use a carefully designed P_{reason} to guide the LLM through a step-by-step reasoning process, resulting in the set E.

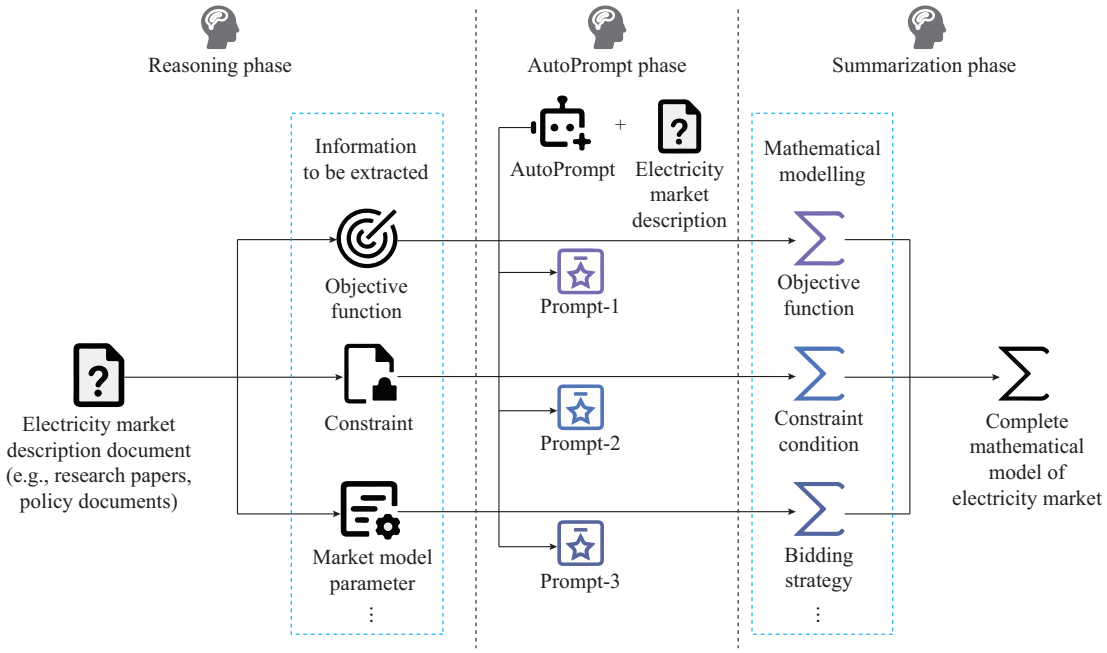

Fig. 2. Process of HCoT-IE for electricity market modelling.

2) AutoPrompt Phase

Considering that each element to be extracted $e_i \in E$ may differ significantly from others, their locations in the document and the way they are described may vary, and the background knowledge required to construct these elements may also differ. HCoT-IE method introduces an AutoPrompt mechanism, where each element e_i from E generated in the reasoning phase is used as input to call the LLM for AutoPrompt mechanism, generating a more precise prompt $P_{auto}^{e_i}$ (represented as Prompt-1, Prompt-2, and Prompt-3 in Fig. 2) to obtain more accurate information extraction results. This process can be represented as:

$$P_{auto}^{e_i} = LLM_{auto}(e_i, P_{base}) \tag{2}$$

where $LLM_{auto}(\cdot)$ denotes the LLM used for AutoPrompt phase; and P_{base} is a set of base prompts used to guide the LLM on generating more specific prompts for a particular element. For instance, if the element e_i is an "objective function", the AutoPrompt mechanism takes this high-level concept and, using a base prompt, instructs the LLM to generate a much more specific query such as "Extract the mathematical formula for the objective function, including all variables and their definitions, and state whether it is a minimization or maximization problem." This targeted approach significantly improves extraction accuracy. Concrete examples of the base prompts used in experiments can be found in Section IV. HCoT then uses the generated prompt $P_{auto}^{e_i}$ to extract relevant information I_{e_i} from the electricity market description, i.e., the document $D_{markdown}$ in Fig. 2, forming the various components of the mathematical model.

$$I_{e_i} = LLM_{extract}(D_{markdown}, P_{auto}^{e_i}) \tag{3}$$

where $LLM_{extract}(\cdot)$ denotes the LLM used for mathematical formula extraction.

3) Summarization Phase

Finally, HCoT consolidates all extracted information $I = \{I_{e_1}, I_{e_2}, ..., I_{e_n}\}$, calling the LLM to perform deduplication, combination, and organization, ultimately forming the complete mathematical model $\mathcal{M}$. This phase can be represented as:

$$\mathcal{M} = LLM_{summarize}(I, P_{summarize}) \tag{4}$$

where $LLM_{summarize}(\cdot)$ denotes the LLM used for summarization phase; and $P_{summarize}$ is a summarization prompt that guides the LLM on integrating the extracted information into the final mathematical model.

Through HCoT-IE, the complex task of information extraction of electricity market modelling can be broken down into multiple sub-steps. A more accurate and comprehensive mathematical model can be obtained by iteratively extracting each element in the electricity market modelling process, though care must be taken to ensure that the increased model complexity is translated into code with high fidelity.

C. Reflective Coding for Electricity Market Simulation

To transform an abstract market model into an executable program and simulate market operation using a computer, it is necessary to convert the mathematical model into code. However, directly converting the complete mathematical model $\mathcal{M}$ into code $\mathcal{C}$ faces the following challenges.

1) Model complexity. Electricity market models often contain complex objective functions, constraints, and bidding strategies. Directly converting these models into code requires significant effort and time.

2) Code implementation details. Even the model is converted into code, it is still necessary to consider code implementation details such as the choice of data structures for electricity market parameters, the specific application programming interface (API) syntax of the chosen optimization solver, and the correct indexing of variables across loops and constraints. These details must be carefully handled to ensure the correctness and efficiency of the code.

3) Debugging difficulties. Due to the complexity of the model and code, debugging is also very difficult. Potential errors can range from simple syntax mistakes to subtle logical flaws, such as using an incorrect inequality sign in a generator limit or encountering a solver error that declares the model "infeasible" from an incorrectly formulated constraint. Even if an error is found, it is difficult to locate the root cause.

To address these challenges, this paper proposes the reflective coding for electricity market simulation, which uses LLM-based reflection debugging mechanism to improve the code generation. As shown in Fig. 3, the specific process is as follows.

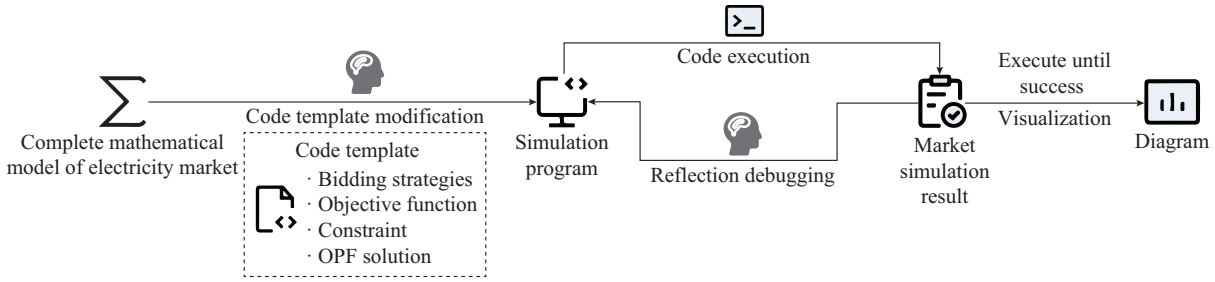

Fig. 3. Process of reflective coding for electricity market simulation.

1) Code Template Modification Phase

The complete mathematical model $\mathcal{M}$ of electricity market obtained in HCoT is input to the LLM, along with a predefined code template $\mathcal{T}$. This code template is designed as a flexible scaffold rather than a rigid program. It contains the boilerplate structure required for a market simulation, such as data handling, solver initialization, and result visualization, but leaves the core logic sections of electricity market as well-defined placeholders. It also predefines the main modules of the simulation program, such as the market clearing module and the unit output module. It uses placeholders to indicate where code needs to be filled in according to the mathematical model. Then, the code generation capability of the LLM is used to modify the preset code template according to the mathematical model, generating the initial simulation code $\mathcal{C}_0$. This phase can be represented as:

$$\mathcal{C}_0 = LLM_{generate}(\mathcal{M}, \mathcal{T}, P_{generate}) \tag{5}$$

where $LLM_{generate}(\cdot)$ denotes the LLM used for simulation generation; and $P_{generate}$ is a prompt that guides the LLM on modifying the code template according to the mathematical model. To enable the LLM to modify the code template more accurately, the code template uses comments to thoroughly explain the functional role of each module, as well as the parts that can be modified and those that should not be modified. For example, a placeholder in the market clearing module might be preceded by comments like "TODO: define the objective function based on cost minimization as specified in model $\mathcal{M}$", guiding the LLM to insert the specific objective function it has already understood. This modular design allows the proposed MSS-Agent framework to adapt to different market structures by changing the logic within the placeholders without altering the overall template architecture.

2) Code Execution and Reflection Debugging Phase

The generated simulation code $\mathcal{C}_i$ is run to perform electricity market simulation, and the simulation results, denoted as $\mathcal{R}_i$ (e.g., LMP and unit output data), are recorded. Subsequently, the reflective coding for electricity market simulation performs different processing based on the code execution status.

1) If $\mathcal{C}_i$ executes successfully, the reflective coding for electricity market simulation visualizes the simulation results $\mathcal{R}_i$ using charts and other forms.

2) If the execution of $\mathcal{C}_i$ fails, the reflective coding for electricity market simulation feeds the error information $\mathcal{E}_i$ back to the LLM and uses the understanding and reasoning capabilities of LLM to analyze the root cause of the error and perform code debugging. This phase can be represented as:

$$\mathcal{C}_{i+1} = LLM_{debug}(\mathcal{C}_i, \mathcal{E}_i, \mathcal{M}, P_{debug}) \tag{6}$$

where $LLM_{debug}(\cdot)$ is the LLM used for code debugging; and P_{debug} is a set of prompts used to guide the LLM in code debugging. The LLM will combine the error information $\mathcal{E}_i$, the current simulation code $\mathcal{C}_i$, and the original mathematical model $\mathcal{M}$ to perform a comprehensive analysis and identify the logical differences between the code implementation and the mathematical model. The code execution and reflection debugging process is iterated, ultimately generating more accurate and efficient simulation code $\mathcal{C}_f$.

The core of the reflection debugging mechanism lies in an-

alyzing the logs $\mathcal{L}_i$ generated during code execution (also generated during successful code execution) and the error information $\mathcal{E}_i$, comparing the mathematical logic of $\mathcal{M}$ with the current simulation code $\mathcal{C}_i$ to identify the differences Δ (as shown in (7)), and adjusting the code based on these differences.

$$\Delta = LLM_{compare}(\mathcal{M}, \mathcal{C}_i, \mathcal{L}_i, \mathcal{E}_i) \tag{7}$$

where $LLM_{compare}(\cdot)$ is the LLM used for comparing mathematical logic, code, logs, and error information.

The LLM implicitly distinguishes different types of errors according to their source. For instance, syntactic or runtime errors (e.g., type error) are identified directly from the traceback of interpreter. The LLM can typically resolve these by correcting specific lines of code. More complex logical errors often manifest as solver failures (e.g., model infeasibility). In such cases, the error information $\mathcal{E}_i$ includes the solver's log, which may indicate conflicting constraints. The LLM is then guided to perform a more rigorous comparison, cross-referencing the code implementation of the identified constraints with their original mathematical formulation in $\mathcal{M}$ to find and fix the logical discrepancy. For example, if the comparison finds that a certain constraint in the mathematical model is not reflected in the code, or if a type mismatch error occurs during code execution, the LLM will modify the code accordingly to implement the constraint or resolve the error.

By using code generation and reflection debugging mechanisms, the reflective coding for electricity market simulation can significantly improve the code generation efficiency and effectively solve the problems of model complexity, code implementation details, and debugging difficulties.

IV. Experiments

This section validates the effectiveness of the proposed MSS-Agent framework through a series of experiments. The experimental design is described in detail, including dataset construction, evaluation metric design, hyperparameter settings, and experimental procedures. The primary validation is performed on a curated dataset of academic papers, chosen for their explicit and rigorous model formulations, which provide a reliable ground truth for assessing performance. Through these experiments, the performance of the proposed MSS-Agent framework is evaluated in electricity market modelling, information extraction, and code generation. Comparisons with baseline methods demonstrate the advantages of the proposed MSS-Agent framework.

A. Experimental Design

1) Test Dataset Construction

We constructed a dataset containing 60 documents covering different types of electricity market mechanisms and scenarios. These documents are mainly from high-quality academic papers published in recent years. This choice was strategic, as academic papers provide well-defined mathematical models that serve as a clear ground truth for validating the accuracy of our information extraction and code generation processes. To ensure the representativeness of the dataset, we selected academic papers covering a wide range of electricity market conditions and structures, including different electricity market models (e.g., day-ahead energy, ancillary service, capacity markets), various energy entities (e.g., renewable generation, energy storage system, demand response), and diverse market mechanisms. This variety presents a robust test for the ability of the MSS-Agent to handle the linguistic and structural variability inherent in technical documents, which is a challenge LLMs are fundamentally designed to address.

2) Evaluation Metric Design

To quantitatively evaluate the performance of the proposed MSS-Agent framework, the following three evaluation metrics are designed.

1) Mathematical modelling coverage

This metric is used to measure the completeness and accuracy of the mathematical model information extracted by the proposed MSS-Agent framework. We compare the extracted mathematical model with the mathematical model in the original document to evaluate whether the extracted information covers all the key elements in the original document, including the objective function, constraints, variable definitions, etc. The score range of this metric is 0-100, with higher scores indicating more complete and accurate extracted information. Specifically, a score of 90-100 indicates a complete and accurate extraction result; a score of 70-89 indicates a relatively complete extraction result with minor omissions or inaccuracies; a score of 50-69 indicates that some key elements are covered, but with numerous omissions or some errors; a score of 30-49 indicates that only a few key elements are covered and there are significant errors; and a score of 0-29 indicates a near-total lack of relevant coverage or a fundamentally flawed extraction. The LLM is used for preliminary evaluation of this metric, and then two experts in the field of electricity markets manually verify and correct it.

2) Code generation compliance

This metric is used to measure whether the execution results of the code generated by the proposed MSS-Agent framework are as expected. We compare the execution results of the code with the expected market behavior (e.g., price, output, etc.) to evaluate whether the code correctly simulates the market operation mechanism. This metric evaluates the compliance of the code with expected outcomes on a scale of 0 to 100, where higher scores reflect greater consistency. On this scale, a score of 90-100 signifies that the simulation results fully align with the expected behavior of the model. A score of 70-89 denotes that the results are generally correct but may exhibit minor deviations in detail. As scores decrease into the 50-69 and 30-49 ranges, they reflect increasingly significant discrepancies between the simulation output and the expected results. A score below 30 indicates a critical failure, where the generated code either fails to execute or produces results that are fundamentally incorrect. The LLM is used for the preliminary evaluation of this metric, and then two experts in the field of electricity markets manually verify and correct it.

3) Code generation *Pass@N*

This metric measures the ability of an agent to generate

correct code within a limited number of attempts. Specifically, *Pass@N* measures the proportion of successful code generations within N debugging cycles. In the code execution phase, we allow the proposed MSS-Agent framework to perform code debugging iteratively based on error information. If the generated code can be successfully executed and passes all test cases within N attempts, it is considered a success. The general formula for *Pass@N* is defined as [32]:

$$Pass@N = \frac{N_{pass}}{N_{total}} \tag{8}$$

where N_{pass} is the number of instances (documents, in our case) for which a correct solution is generated within N attempts; and N_{total} is the total number of instances. In our evaluation, we employ *Pass*@3, which allows for up to three debugging cycles. Physically, *Pass@N* reflects the efficiency and robustness of the code generation process. A high *Pass@N*, especially when N is low, indicates that the agent can quickly find a correct solution with minimal feedback. This demonstrates its ability to effectively understand and rectify errors. This metric can be verified automatically by the compiler without manual intervention.

3) Experimental Setup and Hyperparameters

The experimental setup and hyperparameters used in our experiments are shown in Table I. In the following subsections, we provide further details on our experimental setup, including the specific prompts used to guide the LLM and the underlying assumptions of our market model.

TABLE I
EXPERIMENTAL SETUP AND HYPERPARAMETERS

Item	Setting
LLM (reasoning) used for information extraction	Gemini 2 Flash
LLM (coding) used for code generation and debugging	Gemini 2 Flash
LLM (evaluation) used for coverage and compliance evaluation	GPT-4
IEEE case size used for liveness experiments	3-bus case
Randomness of LLM output	0.7
The maximum length of LLM output	2048
Diversity of LLM output	0.95

For the reasoning phase, an example prompt of P_{reason} is: "Given the following electricity market description, identify the key elements required for mathematical modelling. Consider elements such as the objective function (e.g., cost minimization, profit maximization), constraints (e.g., power balance, generator limits), decision variables, and market parameters."

For the AutoPrompt phase, an example prompt of P_{base} is: "You are an expert in electricity market modelling. Given the element 'element', generate a specific and detailed prompt that would guide an LLM to extract all relevant information related to this element from a technical document. The prompt should be clear, concise, and unambiguous." The 'element' placeholder would be replaced with the specific element (e.g., "objective function") identified in the reasoning phase.

For the summarization phase, an example prompt of $P_{summarize}$ is: "Synthesize the following extracted information into a complete and coherent mathematical model of an electricity market. Ensure that all variables are clearly defined, the objective function is explicitly stated, and all constraints are properly formulated. Use standard mathematical notation."

For the template modification phase, a prompt example of $P_{generate}$ is: "Given the following mathematical model of an electricity market and a code template, modify the template to generate executable code that accurately simulates the market. Pay close attention to the comments in the template, which indicate where specific model components should be implemented." For code execution and reflection debugging phase, an example prompt of P_{debug} is: "Given the following error, source code, and the original mathematical model, please modify the code to resolve the error, considering consistency with the model."

Our baseline experiments consider a simplified day-ahead electricity market with a single period. Generators submit linear supply offers, and demand is assumed to be inelastic. The market clearing process aims to minimize the total generation cost while satisfying power balance and generator capacity constraints. Network constraints are initially ignored for simplicity, but are incorporated in later experiments using the IEEE systems.

B. Experimental Results

Based on the above dataset, evaluation metrics, and hyperparameter settings, the following experiments are conducted to verify the effectiveness of the HCoT method.

1) Effectiveness of HCoT Method

This part mainly verifies the performance of different components in the HCoT method. The performance of the following methods is compared in the information extraction task.

1) Method 1: directly using an LLM by inputting the original document and prompting it to extract all relevant information.

2) Method 2: using the standard CoT method by prompting the LLM to think step by step and extract information.

3) Method 3: using the HCOT method proposed in this paper, breaking down the information extraction task into multiple steps, and extracting information step by step.

4) Method 4: using the HCoT method, adding the AutoPrompt mechanism to automatically generate more accurate prompts for extracting each element.

The performance comparison of different information extraction methods is shown in Table II.

TABLE II
PERFORMANCE COMPARISON OF DIFFERENT INFORMATION EXTRACTION METHODS

Method	Coverage rate (%)	Compliance rate (%)	*Pass*@3 (%)
1	71.12	67.54	91.66
2	76.95	70.75	93.33
3	78.55	70.42	93.33
4	81.91	72.03	98.33

As can be observed from Table II, the performance of Method 1 is the worst, which shows that complex tasks of electricity market information extraction are difficult to complete with simple prompts. Method 2 shows improvement in all evaluated metrics compared with Method 1, proving the effectiveness of Method 2. Method 3 improves the coverage rate by about 1.6 percentage points compared with that of Method 2, which shows that decomposing the information extraction task into multiple steps can extract information more comprehensively. Interestingly, Method 3 shows slightly lower compliance rate than Method 2. This counter-intuitive result suggests a trade-off. On one hand, the more thorough process of Method 3 uncovers complex and nuanced constraints that Method 2 might miss, which explains its higher coverage rate. On the other hand, these same complex constraints are more difficult to implement correctly in code. This difficulty increases the risk of minor implementation errors, which in turn slightly lowers the compliance rate. This challenge is precisely what Method 4 is designed to address. Method 4 achieves the best performance in all metrics, with the coverage rate reaching 81.91%, the compliance rate reaching 72.03%, and the *Pass*@3 reaching 98.33%. Compared with Method 3, after adding the AutoPrompt mechanism, the coverage rate is improved by about 3.4 percentage points, the compliance rate is improved by about 1.6 percentage points, and the *Pass*@3 is improved by about 5 percentage points. This fully demonstrates that the AutoPrompt mechanism can generate more accurate prompts for different elements to be extracted, thereby improving the accuracy and completeness of information extraction.

2) Effectiveness of Reflection Debugging Mechanism

This part mainly verifies the effectiveness of the reflection debugging mechanism in the reflective coding for electricity market simulation. The performance of the following two methods are compared in the code generation task.

1) Method 5: using the reflective coding for electricity market simulation to generate code but without the reflection debugging mechanism.

2) Method 6: using the reflective coding for electricity market simulation to generate code with the reflection debugging mechanism.

The performance comparison of different code generation methods is shown in Table III.

TABLE III
PERFORMANCE COMPARISON OF DIFFERENT CODE GENERATION METHODS

Method	*Pass*@3 (%)
5	85.00
6	98.33

As can be observed from Table III, the *Pass*@3 of Method 6 is significantly higher than that of Method 5. This shows that the reflection debugging mechanism can effectively identify and fix errors in the code, thereby improving the quality of the generated code. By analyzing the logs and error information in the code execution process, and comparing the mathematical logic and code implementation of the model, the reflection debugging mechanism can help the LLM better understand the root cause of code errors and generate more accurate code. A deeper analysis of the debugging cycles reveals that the reflection debugging mechanism adeptly handled several common error categories. Syntactic errors such as incorrect indentation or variable name typos were almost always corrected in the first attempt. More challenging were logical errors related to the optimization model, such as incorrect indexing of variables in constraints or improper formulation of objective function terms. The reflection debugging mechanism proved effective at resolving these by using the solver's feedback (e.g., infeasibility reports on specific constraints) to pinpoint the logical flaw, demonstrating its ability to bridge the gap between the abstract mathematical model and its concrete implementation.

3) Comparison of Different LLMs

This part compares the performance of different LLMs under the HCoT method. We selected two representative LLMs: DeepSeek V3 [33] and Gemini 2 Flash [34]. These models were chosen to represent different design philosophies. DeepSeek V3, developed by DeepSeek AI, is specifically trained on a large corpus of code and technical documents, making it a strong candidate for code generation and specialized domain understanding. In contrast, Gemini 2 Flash is part of Google's latest generation of models, designed to offer a balance of high performance and efficiency with strengths in multimodal reasoning and broad natural language understanding. This comparison allows us to evaluate whether a specialized model or a powerful generalist model is more effective for the end-to-end task of the proposed MSS-Agent framework. The performance comparison of different LLMs is shown in Table IV.

TABLE IV
PERFORMANCE COMPARISON OF DIFFERENT LLMS

LLM	Coverage rate (%)	Compliance rate (%)	*Pass*@3 (%)
DeepSeek V3	71.25	68.87	96.67
Gemini 2 Flash	81.91	72.03	98.33

As Table IV shows, the Gemini 2 Flash outperforms the DeepSeek V3 in all metrics. This result suggests that for the end-to-end task of the proposed MSS-Agent framework, strong general reasoning and natural language understanding capabilities, which are critical for the HCoT-IE phase, are more impactful than specialized coding proficiency alone. While DeepSeek V3 is highly competent, the well-rounded performance of Gemini 2 Flash appears better suited for the initial and complex task of interpreting and structuring the model from unstructured text. This shows that choosing a more powerful LLM can significantly improve the performance of the HCoT method.

4) Impact of IEEE System Bus Sizes

This part analyzes the impact of different IEEE system bus sizes on the quality of code generation under the reflective coding for electricity market simulation. We selected three standard IEEE systems of different sizes from the widely used Power System Test Case Archive [35]: the IEEE 3-

bus system, the IEEE 30-bus system, and the IEEE 118-bus system, representing small, medium, and large power systems, respectively. The performance comparison of different IEEE system bus sizes is shown in Table V.

TABLE V
PERFORMANCE COMPARISON OF DIFFERENT IEEE SYSTEM BUS SIZES

Case	Coverage rate (%)	Compliance rate(%)	*Pass*@3 (%)
IEEE 3-bus	81.91	72.03	98.33
IEEE 30-bus	76.78	74.24	91.07
IEEE 118-bus	75.41	74.00	96.67

The results show that as IEEE system bus sizes increases, that is, as the scale of the power system expands, the coverage rate shows a slight downward trend. The compliance rate is the highest for the IEEE 30-bus system, while *Pass*@3 is the highest for the IEEE 3-bus system. This decline in coverage rate for larger power systems is attributable to the increased complexity and length of the corresponding documents, which can challenge the ability of LLM to maintain context over longer inputs. The non-monotonic trend of the compliance rate and *Pass*@3 reveals a more nuanced interaction with complexity. The lowest *Pass*@3 of IEEE 30-bus system suggests that it represents a "medium-complexity trap". Its model is intricate enough to introduce significant coding challenges such as complex indexing, which are less prevalent in the simpler IEEE 3-bus system. However, unlike the larger IEEE 118-bus system, it lacks a highly regular, pattern-based structure, which prevents the LLM from generalizing its coding approach and leads to more errors. Conversely, the highest compliance rate of IEEE 30-bus system indicates that it may be a "model fidelity sweet spot", which is a standard, well-defined problem in literature that, when correctly coded, produces outcomes closely aligned with expectations. Simpler systems may lack this fidelity, while larger ones may introduce minor numerical deviations due to scale.

Nevertheless, the *Pass*@3 for all test systems remains above 90%, demonstrating the overall effectiveness of the reflective coding for electricity market simulation, even as system complexity increases. It is important to distinguish the scalability of the MSS-Agent from the computational scalability of the simulation itself. While the MSS-Agent successfully formulates larger models, the primary bottleneck for very large-scale models (e.g., regional or global electricity markets) becomes the computational time required by the optimization solver to find a solution, rather than the model generation process itself.

C. Case Study

To further validate the practical effectiveness of the proposed MSS-Agent framework, a case study was conducted utilizing the IEEE 3-bus system, a simplified yet representative model for power system analysis. The input for this case study is the research paper [36], as presented in the Fig. 4. The proposed MSS-Agent framework processed this document, extracting the relevant market model information, including the objective function (maximizing total expected revenue), decision variables (capacity allocated to each market), and various constraints (capacity, cost, individual limits, and derivative market constraints). Subsequently, the MSS-Agent generated Python code based on a predefined template, incorporating the extracted information and utilizing the reflective coding for electricity market simulation.

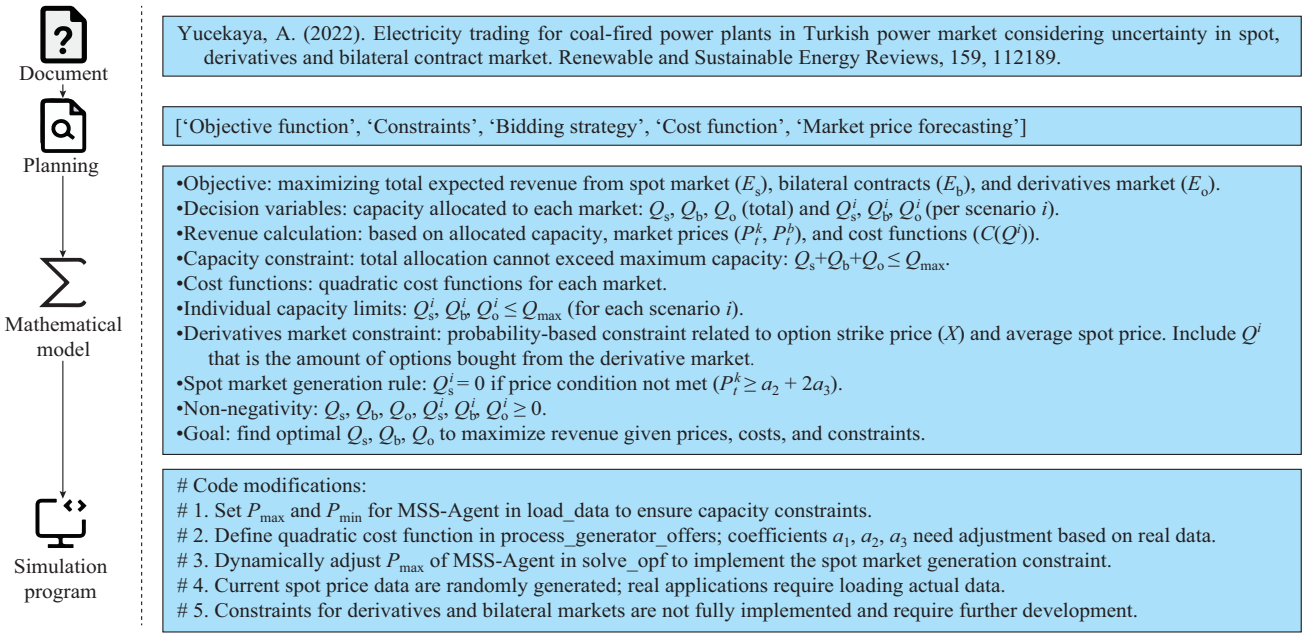

Fig. 4. Information extraction and model formulation.

The simulation produced results for generation output over time and LMPs across three buses over time, as illustrated in Figs. 5 and 6. Notably, the "Generation MSS-Agent" output in Fig. 5 represents the optimized generation

schedule determined by the MSS-Agent, while the other "Generation" outputs represent predefined baseline generation profiles categorized by their operational roles: base-load coal units (CG1 and CG2), flexible gas turbines (GSTONE1 and GSTONE2), and strategic reserve units (STAN2). Crucially, the simulation results shown in Figs. 5 and 6 aligned with the expected behavior of the electricity market model described in the input document. The proposed MSS-Agent framework successfully captured the key dynamics of the electricity market, including the interplay between spot and derivative markets, and produced a feasible and economically rational generation dispatch. This case study demonstrates the capability of proposed MSS-Agent framework to automate the process of the electricity market modelling and simulation, starting from a textual description and culminating in executable code that yields meaningful results.

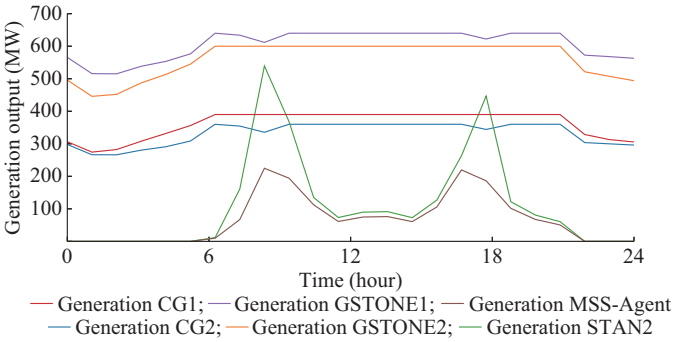

Fig. 5. Generation output over time.

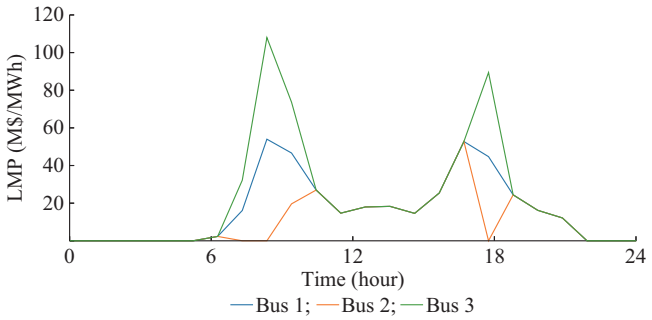

Fig. 6. LMPs across three buses over time.

V. Discussion

The proposed MSS-Agent framework, while demonstrated in an academic context, holds significant potential as a decision-support and rapid-prototyping tool for market regulators, analysts, and participants. For instance, regulators could use it to quickly translate a new policy document into a simulation model, enabling a preliminary impact assessment that accelerates the policy analysis cycle. However, transitioning it to real-world implementation faces hurdles. Stakeholder adoption would require a "human-in-the-loop" approach for expert verification, and practical deployment would necessitate the integration with complex data systems and the use of secure and on-premises LLMs to protect sensitive market data.

The strength of the proposed MSS-Agent framework lies in its adaptability. To simulate a carbon tax, for example, the HCoT-IE would extract the tax rules, and the reflective coding for electricity market simulation would modify the objective function of code template to include a new emission-based cost term. Similarly, renewable energy incentives can be implemented as adjustments to revenue functions or as new system constraints. Despite this flexibility, the framework has limitations. Its performance is contingent on the underlying capabilities of LLMs and the quality of input documents (poorly structured or ambiguous texts can degrade extraction accuracy). Furthermore, the reflective coding while effective for optimization problems, would require significant modification for fundamentally different paradigms such as agent-based equilibrium models.

Future work will focus on extending the proposed MSS-Agent framework to handle a wider variety of documents such as official market rulebooks and high-level policy papers. This will require enhancing the ability of the MSS-Agent to interpret legalistic prose and translate qualitative goals into quantitative model constraints, likely reinforcing the need for expert validation. It is also important to clarify that the proposed MSS-Agent framework is an offline model construction tool, not a real-time operational system. It handles dynamic market conditions by enabling the rapid creation of high-fidelity models for scenario analysis, where users can then test the impact of volatile inputs like fluctuating renewable generation or demand shifts.

VI. Conclusion

This paper addresses the core challenges of automated electricity market analysis by proposing the MSS-Agent framework, which integrates the HCoT-IE method and the reflective coding for electricity market simulation. The HCoT-IE method ensures accurate model extraction from complex documents, while the integrated reflection debugging mechanism enables the agent to autonomously generate and validate reliable simulation codes. Experimental results and a case study demonstrate that the proposed MSS-Agent framework significantly improves the efficiency and accuracy of the end-to-end modelling workflow. Future research will focus on enhancing the versatility of the framework to handle a broader range of market structures such as multi-period and stochastic models, thereby expanding its applicability as a powerful tool for policy analysis and market design in increasingly complex energy systems.

References

[1] X. Yu and Y. Xue, "Smart grids: a cyber–physical systems perspective," *Proceedings of the IEEE*, vol. 104, no. 5, pp. 1058-1070, May 2016.

[2] Y. Xue and X. Yu, "Beyond smart grid: cyber–physical–social system in energy future (point of view)," *Proceedings of the IEEE*, vol. 105, no. 12, pp. 2290-2292, Dec. 2017.

[3] J. Yang, Z. Y. Dong, F. Wen *et al*., "Spot electricity market design for a power system characterized by high penetration of renewable energy generation," *Energy Conversion and Economics*, vol. 2, no. 2, pp. 67-78, May 2021.

[4] C. Bu, K. Zhang, D. Shi *et al*., "Does environmental information disclosure improve energy efficiency?" *Energy Policy*, vol. 164, p. 112919, May 2022.

[5] W. Liu, B. He, Y. Xue *et al*., "A comprehensive modeling framework for coupled electricity and carbon markets," *Energy Conversion and*

Economics, vol. 5, no. 1, pp. 1-14, Feb. 2024.
[6] S. Shi, T. Yu, J. Huang *et al*., "Decision optimization of generation companies in electricity and carbon spot markets considering cash flow and carbon compliance period," *Automation of Electric Power Systems*, vol. 48, no. 19, pp. 40-50, Oct. 2024.
[7] A. Stratigakos, S. Pineda, J. M. Morales *et al*., "Interpretable machine learning for DC optimal power flow with feasibility guarantees," *IEEE Transactions on Power Systems*, vol. 39, no. 3, pp. 5126-5137, May 2024.
[8] Y. Chang, X. Wang, J. Wang *et al*., "A survey on evaluation of large language models," *ACM Transactions on Intelligent Systems and Technology*, vol. 15, no. 3, pp. 1-45, Jun. 2024.
[9] J. Achiam, S. Adler, S. Agarwal *et al*. (2023, Mar.). GPT-4 technical report. [Online]. Available: https://arxiv.org/abs/2303.08774
[10] L. Huang, W. Yu, W. Ma *et al*., "A survey on hallucination in large language models: principles, taxonomy, challenges, and open questions," *ACM Transactions on Information Systems*, vol. 43, no. 2, pp. 1-55, Mar. 2025.
[11] Y. Bai, S. Kadavath, S. Kundu *et al*. (2022, Dec.). Constitutional AI: harmlessness from AI feedback. [Online]. Available: https://arxiv. org/abs/2212.08073
[12] J. Lin, Z. Ma, R. Gomez *et al*., "A review on interactive reinforcement learning from human social feedback," *IEEE Access*, vol. 8, pp. 120757-120765, Jul. 2020.
[13] L. Wang, C. Ma, X. Feng *et al*., "A survey on large language model based autonomous agents," *Frontiers of Computer Science*, vol. 18, no. 6, p. 186345, Dec. 2024.
[14] C. Lou, C. Li, L. Zhang *et al*., "Two-stage bidding strategy with dispatch potential of electric vehicle aggregators for mitigating three-phase imbalance," *Journal of Modern Power Systems and Clean Energy*, vol. 13, no. 5, pp. 1823-1835, Sept. 2025.
[15] L. Cheng, P. Huang, T. Zou *et al*., "Evolutionary game-theoretical approaches for long-term strategic bidding among diverse stakeholders in large-scale and local power markets: basic concept, modelling review, and future vision," *International Journal of Electrical Power & Energy Systems*, vol. 166, p. 110589, May 2025.
[16] M. R. Salehizadeh and S. Soltaniyan, "Application of fuzzy *Q*-learning for electricity market modeling by considering renewable power penetration," *Renewable and Sustainable Energy Reviews*, vol. 56, pp. 1172-1181, Apr. 2016.
[17] W. Liu, J. Zhao, J. Qiu *et al*., "Interpretable hybrid experimental learning for trading behavior modeling in electricity market," *IEEE Transactions on Power Systems*, vol. 38, no. 2, pp. 1022-1032, Mar. 2023.
[18] Z. Chen, Z. Li, D. Lin *et al*., "Multi-time-scale optimal scheduling of integrated energy system with electric-thermal-hydrogen hybrid energy storage under wind and solar uncertainties," *Journal of Modern Power Systems and Clean Energy*, vol. 13, no. 3, pp. 904-914, May 2025.
[19] W. Cai, R. Mohammaditab, G. Fathi *et al*., "Optimal bidding and offering strategies of compressed air energy storage: a hybrid robust-stochastic approach," *Renewable Energy*, vol. 143, pp. 1-8, Dec. 2019.
[20] X. Lu, J. Qiu, Y. Yang *et al*., "Large language model-based bidding behavior agent and market sentiment agent-assisted electricity price prediction," *IEEE Transactions on Energy Markets*, *Policy and Regulation*, vol. 3, no. 2, pp. 223-235, Jun. 2025.
[21] A. Vaswani, "Attention is all you need," in *Proceedings of Advances in Neural Information Processing Systems*, Long Beach, USA, Dec. 2017, pp. 1-7.
[22] M. Zhang and J. Li, "A commentary of GPT-3 in MIT technology review 2021," *Fundamental Research*, vol. 1, no. 6, pp. 831-833, Nov. 2021.
[23] J. Wei, X. Wang, D. Schuurmans *et al*., "Chain-of-thought prompting elicits reasoning in large language models," *Advances in Neural Information Processing Systems*, vol. 35, pp. 24824-24837, Nov. 2022.
[24] G. Yang, Y. Zhou, X. Chen *et al*., "Chain-of-thought in neural code generation: from and for lightweight language models," *IEEE Transactions on Software Engineering*, vol. 50, no. 9, pp. 2437-2457, Sept. 2024.
[25] S. Yao, J. Zhao, D. Yu *et al*. (2022, Oct.). ReAct: synergizing reasoning and acting in language models. [Online]. Available: https://arxiv.org/abs/2210.03629
[26] S. Hong, M. Zhuge, J. Chen *et al*. "MetaGPT: Meta programming for a multi-agent collaborative framework," in *Proceedings of the Twelfth International Conference on Learning Representations*, Vienna, Austria, May 2024, pp. 1-6.
[27] Z. Ji, T. Yu, Y. Xu *et al*., "Towards mitigating LLM hallucination via self reflection," in *Findings of the Association for Computational Linguistics*: *EMNLP*. Singapore: Association for Computational Linguistics, 2023, pp. 1827-1843.
[28] N. Shinn, F. Cassano, A. Gopinath *et al*., "Reflexion: language agents with verbal reinforcement learning," in *Proceedings of Advances in Neural Information Processing Systems*, New Orleans, USA, Dec. 2023, pp. 1-7.
[29] A. Madaan, N. Tandon, P. Gupta *et al*., "Self-refine: iterative refinement with self-feedback," in *Proceedings of Advances in Neural Information Processing Systems*, New Orleans, USA, Dec. 2023, pp. 1-6.
[30] Y. Cheng, X. Zhou, H. Zhao *et al*., "Large language model for low-carbon energy transition: roles and challenges," in *Proceedings of 2024 4th Power System and Green Energy Conference*, Shanghai, China, Aug. 2024, pp. 810-816.
[31] Y. Cheng, H. Zhao, X. Zhou *et al*. (2024, Aug.). GAIA: a large language model for advanced power dispatch. [Online]. Available: https://arxiv.org/abs/2408.03847
[32] M. Chen, J. Tworek, H. Jun *et al*. (2021, Jul.). Evaluating large language models trained on code. [Online]. Available: https://arxiv.org/abs/2107.03374
[33] A. Liu, B. Feng, B. Xue *et al*. (2024, Dec.). DeepSeek V3 technical report. [Online]. Available: https://arxiv.org/abs/2412.19437
[34] G. Team, R. Anil, S. Borgeaud *et al*. (2023, Dec.). Gemini: a family of highly capable multimodal models. [Online]. Available: https://arxiv.org/abs/2312.11805
[35] R. D. Zimmerman and C. E. Murillo-Sanchez. (2011, Dec.). Matpower 4.1 user's manual. [Online]. Available: https://matpower. org/docs/MATPOWER-manual-4.1.pdf
[36] A. Yucekaya, "Electricity trading for coal-fired power plants in Turkish power market considering uncertainty in spot, derivatives and bilateral contract market," *Renewable and Sustainable Energy Reviews*, vol. 159, p. 112189, May 2022.

Yuheng Cheng received the B.S. degree in electronic engineering from the University of Electronic Science and Technology of China, Chengdu, China, in 2017, and the M.S. degree in computer science from the Hong Kong University of Science and Technology, Hong Kong, China, in 2018. He is currently pursuing the Ph.D. degree in computer science with the Chinese University of Hong Kong (Shenzhen), Shenzhen, China. His research interests include natural language processing, large language model, and energy management.

Wenxuan Liu received the B.S. degree in finance and the B.E. degree in electronic engineering from the University of Electronic Science and Technology of China, Chengdu, China, in 2013, the M.Phil. degree in applied economics from the same university in 2016, and the Ph.D. degree in computer and information engineering from The Chinese University of Hong Kong (Shenzhen), Shenzhen, China, in 2021. He is currently a Postdoctoral Researcher with the School of Science and Engineering, The Chinese University of Hong Kong (Shenzhen). He is also a Research Fellow with the Shenzhen Institute of Artificial Intelligence and Robotics for Society, Shenzhen, China. His research interests include electricity market, carbon market, machine learning, and experimental economics.

Yusheng Xue received the Ph.D. degree in electrical engineering from the University of Liege, Liege, Belgium, in 1987. He became a Member of the Chinese Academy of Engineering in 1995. He is currently the Honorary President of State Grid Electric Power Research Institute (NARI Group Corporation), Nanjing, China. His research interests include nonlinear stability, control, and power system automation.

Jie Huang received the Ph.D. degree from the Nanjing University of Science and Technology, Nanjing, China, in 2011. He is currently a Professorate Senior Engineer with NARI Technology Co., Ltd., Nanjing, China. His research interests include carbon market and carbon emission risk management strategy.

Junhua Zhao received the Ph.D. degree in electrical engineering from the University of Queensland, Brisbane, Australia, in 2007. He was a Senior Lecturer with the University of Newcastle, Newcastle, Australia, and also with the Center for Intelligent Electricity Networks, University of Newcastle. He is currently the Assistant Dean and an Associate Professor with the School of Science and Engineering, The Chinese University of Hong Kong (Shenzhen), Shenzhen, China. He is also the Director of the Energy Market and Finance Lab, Shenzhen Finance Institute, The Chinese University of Hong Kong, Hong Kong, China. His research interests include power system analysis and computation, smart grid, electricity market, data mining,

and artificial intelligence.

Fushuan Wen received the B.E. and M.E. degrees in electrical engineering from Tianjin University, Tianjin, China, in 1985 and 1988, respectively, and the Ph.D. degree in electrical engineering from Zhejiang University, Hangzhou, China, in 1991. In 1991, he joined the School of Electrical Engineering, Zhejiang University, as a Faculty Member, where he has been a Full Professor, since 1997. Since 2022, he has been a Full Professor with the Hainan Institute, Zhejiang University. He is currently a part-time Distinguished Professor with Hangzhou Dianzi University, Hangzhou, China, and a Visiting Principal Research Scientist with the Shenzhen Institute of Artificial Intelligence and Robotics for Society, Shenzhen, China. His research interests include power industry restructuring, power system alarm processing, fault diagnosis and restoration strategy, smart grid, electricity market, and artificial intelligence application in power and integrated energy systems.