Agentic Frameworks for Physics-Informed Neural Networks: Evaluating Planner Architectures

Byeongha Jo [a,b], Jaejun Lee [a,b], Yonggyun Yu [a,b], Hogeon Seo [a,b*]
[a] *Korea Atomic Energy Research Institute, 111, Daedeok-daero 989 beon-gil, Yuseong-gu, Daejeon, 34057, Korea*
[b] *University of Science & Technology, 217, Gajeong-ro, Yuseong-gu, Daejeon, 34113, Korea*
[*] *Corresponding author: hogeony@hogeony.com*

Keywords : agentic framework, task decomposition, small language models, physics-informed neural networks

1. Introduction

Ensuring the safety of nuclear power plants involves predicting thermal-hydraulic phenomena, such as heat distribution within fuel rods. Traditional analyses utilize high-fidelity numerical codes, which require domain expertise to define geometries, select physical models, and manage mesh generation. Physics-Informed Neural Networks (PINNs) have been introduced as a mesh-free approach by embedding physical laws directly into the loss function [1]. However, configuring these models requires translating physical phenomena into specific mathematical formulations, a process that generally necessitates deep learning expertise that may fall outside the primary focus of domain engineers.

Agentic artificial intelligence, utilizing large language models, provides a method to support domain engineers by translating natural language into executable scientific workflows [2]. In domains with specific data security considerations, such as nuclear engineering, computational workflows are frequently deployed in on-premise environments. While large-scale local infrastructure is available in certain facilities, the operational utilization of these automated tools on standard engineering workstations suggests potential advantages in exploring relatively compact language models [3]. These scaled-down models, however, can exhibit performance variations in complex zero-shot reasoning tasks. When tasked with extracting an entire physical schema simultaneously during the problem definition phase, they may occasionally misinterpret specific boundary conditions or parameters, potentially introducing logical inconsistencies into the downstream simulation pipeline.

This study investigates an agentic framework designed to assist engineering workflows by restructuring the problem definition phase. Decomposing tasks into ordered steps influences the structural consistency of model outputs, with variations dependent on the specific problem configurations [4]. Applying this concept to a one-dimensional steady-state heat conduction problem, we compare a baseline single-step extraction method against parallel and sequential information processing architectures. The objective is to systematically determine how the structure of information extraction influences the operational reliability and convergence success rate of local agentic systems in physical simulations.

2. Methods and Results

2.1 Physical Model and PINN Formulation

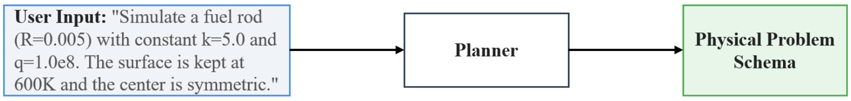
The test environment is based on a one-dimensional steady-state heat conduction problem within a cylindrical fuel rod. The spatial domain is defined along the radial coordinate $r \in [0, R_{fuel}]$, where R_{fuel} is the outer radius of the fuel pellet. Assuming azimuthal symmetry, zero axial heat conduction, and a constant thermal conductivity k, the energy conservation equation with a constant volumetric heat generation rate q''' is expressed as:

$$\frac{1}{r}\frac{d}{dr}\left(rk\frac{dT}{dr}\right) + q''' = 0$$

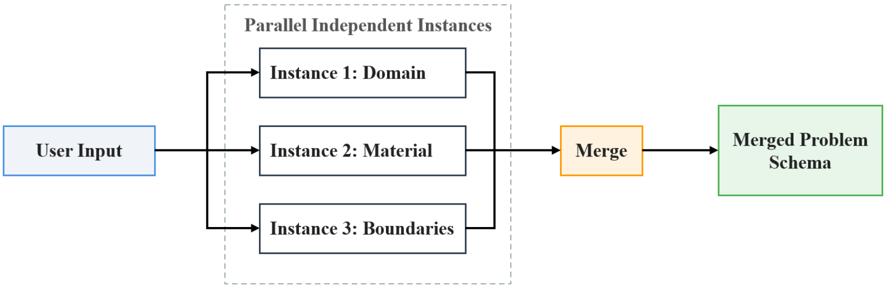
To form a well-posed problem, two boundary conditions are applied: a symmetry condition at the center ($\left.\frac{dT}{dr}\right|_{r=0}$) and a Dirichlet condition fixing the temperature at the fuel surface ($T(R_{fuel}) = T_{surf}$). The framework utilizes a PINN to approximate the temperature distribution $T(r)$, using automatic differentiation to compute spatial derivatives. The network is trained by minimizing a composite loss function that calculates the mean squared residual of the governing partial differential equation at interior collocation points, alongside the deviations from the prescribed boundary constraints.

2.2 Planner Architectures

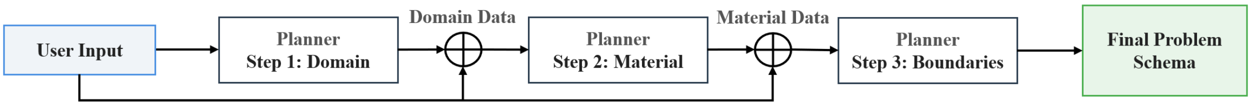
The framework centers on an agentic system that utilizes a planner module to convert unstructured user input into a physical problem schema. To evaluate the impact of information processing strategies on small language models, this study compares three planner architectures. The baseline planner executes a single-step extraction (Fig. 1(a)), attempting to identify the domain geometry, material properties, and boundary conditions simultaneously in one inference pass. The parallel planner isolates these tasks (Fig. 1(b)), deploying independent agent instances to extract specific parameter categories simultaneously before merging the results. The sequential planner processes information in a chained structure (Fig. 1(c)) where distinct categories of physical parameters are extracted in a predefined order. Each step in the sequential method integrates the original user input with the validated output from previous steps to provide expanded context for subsequent extractions, ensuring that later parameter definitions remain physically consistent with the established variables.



(a) Baseline Planner



(b) Parallel Planner



(c) Sequential Planner

Fig. 1. Architectural comparison of planners. (a) Baseline Planner performs single-step extraction. (b) Parallel Planner executes independent concurrent extraction followed by a merge process. (c) Sequential Planner relies on chained context extraction, where ⊕ indicates the integration of original input with preceding extracted data.

2.3 Implementation Details

Once the problem schema is defined, the system generates an executable PyTorch script. The experiments evaluate SLMs ranging from 0.6B to 4B parameters on local A100 GPUs. For each model, ten independent trials are conducted. While the initial natural language user query remains strictly identical across all trials to ensure a consistent problem baseline, the underlying system orchestration and agent-specific instruction prompts are structurally tailored to correspond with the respective planner architecture.

To isolate the performance impact of the planner architectures, all neural network hyperparameters are kept constant across trials. This includes the depth of hidden layers, the number of neurons per layer, the density of collocation points, and the loss weighting factors. The training process uniformly applies a two-phase optimization strategy, initiating with the Adam optimizer and transitioning to L-BFGS to refine the optimization. The performance metric for comparison is the success rate, defined strictly as the percentage of trials that successfully satisfy the physical constraints with an L2 error below 0.001.

2.4 Results

The experimental outcomes indicate that the effectiveness of the planner architecture relates to the parameter scale of the language model. For the 4B model the choice of information extraction structure did not produce observable differences in the outcome with all trials satisfying the specified error threshold across the tested configurations as detailed in Table 1. Conversely the 1.2B model and the STAR model did not successfully formulate the physical schema in any of the applied planner architectures.

Performance variations corresponding to the planner architectures were observed in the 1.7B and 0.6B models. For these models the sequential extraction method resulted in a greater number of successful problem formulations compared to the baseline and parallel approaches. The application of the parallel architecture corresponded to an increased success rate for the 1.7B model compared to its baseline but yielded zero successful trials for the 0.6B model. This observation indicates that extracting interdependent physical

Model	Success Rate (%)		
	Baseline	Parallel	Sequential
Qwen3-4B	100	100	100
Qwen3-1.7B	20	50	70
LFM2.5-1.2B-Thinking	0	0	0
STAR-0b6	0	0	0
Qwen3-0.6B	10	0	30

Table I: Success rates of the evaluated small language models across baseline and parallel and sequential planner architectures based on ten independent trials.

parameters simultaneously without a sequential context may introduce formulation errors in certain models. The recorded data suggests that providing cumulative historical context through a sequential pipeline can serve as a structural approach to support the success rate of constrained models in the evaluated physical simulations.

A qualitative inspection of unsuccessful trials suggests that the dominant failure mode in capacity-constrained models emerged during the initial physical schema extraction stage, where limited domain knowledge and incomplete adherence to the instruction frequently led to omitted or inconsistent definitions of physical variables and boundary conditions. Nevertheless, in a subset of retried cases, retaining the prior interaction history occasionally guided the model toward a representation more closely aligned with the stated problem specification, implying that cumulative contextual feedback can partially compensate for these limitations.

3. Conclusions

This study investigated an agentic framework designed to support domain engineers in formulating physics-informed neural networks for thermal analysis using small language models. To evaluate the impact of information processing structures on operational reliability, we compared baseline, parallel, and sequential planner architectures within a one-dimensional steady-state heat conduction problem. The experimental data suggests that decomposing the problem definition phase into a sequential pipeline with cumulative historical context can mitigate formulation errors and support the overall success rate of capacity-constrained models. While the current evaluation establishes a verifiable baseline, practical engineering applications frequently involve coupled multi-physics and higher-dimensional transient phenomena. Future research will investigate the scalability of the sequential architecture in these advanced domains, specifically evaluating how hierarchical state-passing manages an increased number of physical constraints without exceeding the context window limitations inherent to on-premise language models.

ACKNOWLEDGMENTS

This work was supported in part by Korea Atomic Energy Research Institute R&D Program under Grant KAERI-526140-26, and in part by the National Research Council of Science & Technology (NST) grant by the Korea government (MSIT) (No. GTL24031-000).

REFERENCES

[1] M. Raissi, P. Perdikaris, G. E. Karniadakis, “ Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations”, Journal of Computational physics, Vol. 378, pp. 686-707, 2019.

[2] D. A. Boiko, R. MacKnight, B. Kline, G. Gomes, “Autonomous chemical research with large language models”, Nature, Vol. 624, No. 7992, pp. 570-578, 2023.

[3] O. Friha, M. A. Ferrag, B. Kantarci, B. Cakmak, A. Ozgun, N. Ghoualmi-Zine, “Llm-based edge intelligence: A comprehensive survey on architectures, applications, security and trustworthiness”, IEEE Open Journal of the Communications Society, Vol. 5, pp. 5799-5856, 2024.

[4] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou, “Chain-of-thought prompting elicits reasoning in large language models”, Advances in neural information processing systems, Vol. 35, pp. 24824-24837, 2022.