Adaptive Orchestration of Heterogeneous LLMs with Dynamic Escalation for Multi-Agent Collaboration

Bipin Bohara*, Hao Wang†, Praveenkumar Kondikoppa‡, Seung-Jong Park*
* *Department of Computer Science, Missouri University of Science and Technology*, Rolla, MO, USA
{bbdzv, seung-jong.park}@mst.edu
† *Department of Electrical and Computer Engineering, Stevens Institute of Technology*, Hoboken, NJ, USA
hwang9@stevens.edu
‡ *Google LLC*, Mountain View, CA, USA
pkondikoppa@google.com

***Abstract*—LLM based multi-agent systems offer strong reasoning capabilities but often incur substantial computational and operational costs. This work presents an adaptive two-orchestrator framework for coordinating heterogeneous small and large LLMs. Orchestrator-1 (O-1) determines the initial number of expert agents and model tier using task-level risk signals. The primary adaptive component, Orchestrator-2 (O-2) evaluates multiple round-level signals using a heuristic approach and selectively escalates only when the current model configuration is deemed insufficient. After evaluation across various benchmarks using models deployed with vLLM we found that compared with the Large-only configuration, the Small + Large configuration improved accuracy by 0.51–6.67 percentage points, reduced average cost per task by 15.99%–33.65%, and decreased execution time by 8.1%–18.06%. Only 0.10%–3.60% of tasks required escalation, demonstrating the effectiveness of O-2's signal-driven decision mechanism. These results validate that dynamic escalation can preserve or improve performance while reducing unnecessary use of Larger models.**

***Index Terms*—Adaptive orchestration, model escalation, multi-agent systems, SLM (Small Language Model)**

I. Introduction

LLM-based multi-agent systems improve problem solving through collaborative generation, review, and refinement, but using large models for every agent increases cost, latency, and hardware demand. A preliminary study with 200 samples each from MMLU and GSM-Hard motivated heterogeneous model use. As shown in Fig. 1, the Small + Large setting increased accuracy from 39.0% to 80.9% on MMLU and from 27.0% to 67.0% on GSM-Hard, while reducing average execution time by 75.59% and 78.47%, respectively.

Static heterogeneous assignment, however, cannot adapt when task difficulty changes during collaboration. We therefore propose the two-orchestrator framework in Fig. 2. O-1 selects the initial number of agents and model tier from task-level risk signals, while O-2 monitors round-level behavior and escalates agents when the current tier is insufficient. The objective is to reduce unnecessary large-model use while preserving accuracy and improving resource efficiency.

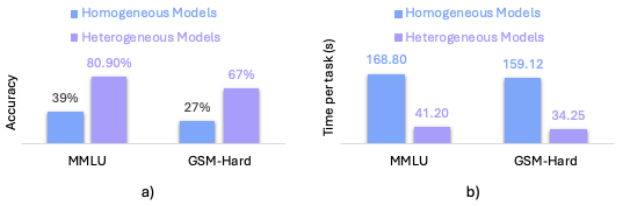


Fig. 1. Comparison of homogeneous and heterogeneous model-agent configurations: (a) accuracy and (b) average execution time per task.

II. Design

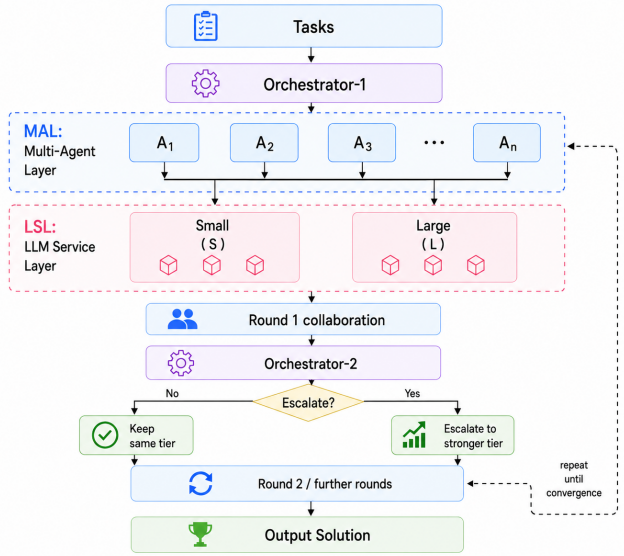


Fig. 2. System architecture of the proposed framework.

As shown in Fig. 2, the framework contains a multi-agent collaboration layer and an LLM service layer hosting small and large models. For each task, expert agents collaborate over multiple rounds by reviewing and refining candidate responses.

The framework supports memory, relay, report, and debate interaction paradigms and multiple decision mechanisms; our experiments use unanimity consensus, requiring agreement from all agents.

Resource allocation occurs in two stages. O-1 uses the reasoning SLM listed in Table II and a weighted task-level heuristic to select the number of agents and initial model tier. Lower-risk tasks begin on small models, while higher-risk tasks are assigned directly to large models. During execution, O-2 evaluates round-level signals, including divergence, lack of convergence, and insufficient capability. It retains the current tier when adequate and escalates agents when additional capability is required.

III. Evaluation and Results

Configuration	**Kubernetes Cluster** deployed on 1 Head node, 9 worker nodes as below
Worker Node	CPU: 2x Intel Xeon Gold 6548Y+, 32 cores/64 threads each Mem: 16x 32 GB DDR5-5600 RDIMMs; 512 GB each GPUs: 4x NVIDIA L40S PCIe GPUs
Head Node	CPU: 2 × Intel Xeon Gold 6428N, 32 cores/64 threads each Mem: 6 × 64 GB DDR5-5600 RDIMMs; 1,024 GB total GPUs: 4x NVIDIA L40S PCIe GPUs

TABLE I
HARDWARE CONFIGURATION OF THE KUBERNETES CLUSTER USED FOR EVALUATION.

Models	Deployment Configurations
openai/gpt-oss-20b nvidia/NVIDIA-Nemotron-3-Nano-30B-A3B-FP8 Qwen/Qwen3.5-35B-A3B-FP8	CPU: 1x Intel Xeon Gold 6548Y+, 32 cores/64 threads Mem: 256 GB GPU: 2x NVIDIA L40S PCIe GPUs
openai/gpt-oss-120b nvidia/NVIDIA-Nemotron-3-Super-120B-A12B-FP8 Qwen/Qwen3.5-122B-A10B-FP8	CPU: 1 entire compute node Mem: 1 entire compute node GPU: 4x NVIDIA L40S PCIe GPUs
mistralai/Ministral-3-3B-Reasoning-2512	CPU: 1x Intel Xeon Gold 6548Y+, 16 cores/42 threads Mem: 128 GB GPU: 1x NVIDIA L40S PCIe GPUs

TABLE II
MODEL DEPLOYMENT CONFIGURATIONS USED IN THE EVALUATION.

The framework was evaluated on the Kubernetes cluster in Table I, with models served through vLLM [3]. Models from OpenAI [5], NVIDIA [6], [7], and Qwen [8] were deployed as summarized in Table II. Experiments used randomly sampled datasets [4] of MMLU, GSM-Hard, MMLU-Pro, Finance Bench, and PubMedQA, with a Large-only configuration as the baseline.

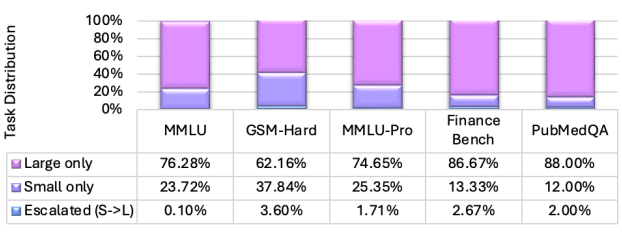


Fig. 3. Task allocation across model tiers, including tasks escalated by O-2.

Fig. 3 shows that O-1 assigned 12.00%–37.84% of tasks to small models, while O-2 escalated only 0.10%–3.60% of all tasks when additional capability was required. The Small + Large configuration improved accuracy on every benchmark (Fig. 4), with gains ranging from 0.51 to 6.67 percentage points; the largest gain occurred on Finance Bench, from 72.00% to 78.67%.

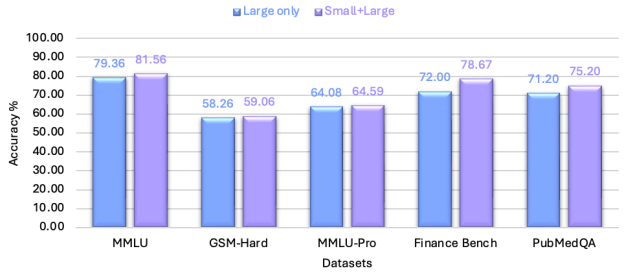


Fig. 4. Accuracy comparison between the Large-only and Small + Large configurations.

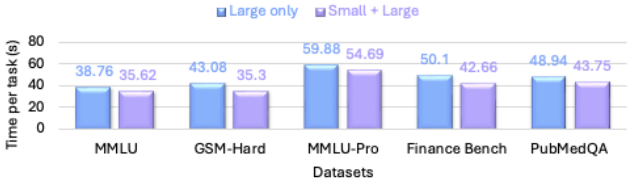


Fig. 5. Execution-time comparison between the Large-only and Small + Large configurations.

Datasets	Large only	Small + Large	Cost reduction %
MMLU	0.3585	0.2902	19.06%
GSM-Hard	0.3985	0.2644	33.65%
MMLU-Pro	0.5539	0.4413	20.32%
Finance Bench	0.4634	0.3682	20.55%
PubMedQA	0.4526	0.3802	15.99%

TABLE III
COST COMPARISON AND REDUCTION BETWEEN THE LARGE-ONLY AND SMALL + LARGE CONFIGURATIONS.

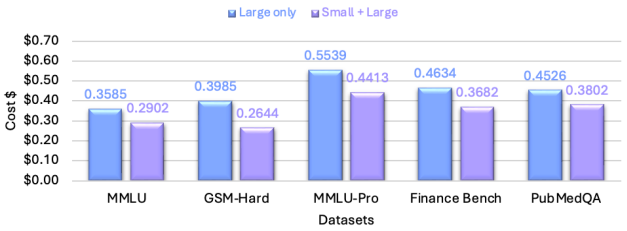


Fig. 6. Per-task cost comparison between the Large-only and Small + Large configurations.

The adaptive configuration reduced execution time by 8.1%–18.06% (Fig. 5) and per-task cost by 15.99%–33.65% (Table III and Fig. 6). Overall, the results show that task-aware initial allocation and selective runtime escalation reduce

large-model utilization and resource cost while maintaining or improving accuracy.

IV. Conclusion

This study developed a two-orchestrator framework for efficient collaboration among heterogeneous LLM agents. O-1 determines the initial agent configuration from task-level risk signals, while O-2 serves as the adaptive control mechanism during multi-round execution. By combining multiple round-level indicators through heuristic evaluation, O-2 promotes agents to a larger model only when the current tier appears inadequate, rather than escalating based on a fixed round. Evaluation across five benchmarks, the Small + Large configuration achieved accuracy gains of 0.51–6.67 percentage points, cost reductions of 15.99%–33.65%, and execution-time reductions of 8.1%–18.06% compared with the Large-only baseline. The limited escalation rate of 0.10%–3.60% shows that O-2 can identify when additional model capacity is necessary while avoiding unnecessary upgrades. These findings demonstrate that adaptive, signal-based orchestration can improve resource efficiency without sacrificing performance.

Acknowledgment

This material is based upon work partially supported by the National Science Foundation under Grant No. #2403248 and #2403399.

References

[1] Jonas Becker, Lars Benedikt Kaesberg, Niklas Bauer, Jan Philip Wahle, Terry Ruas, and Bela Gipp. 2025. MALLM: Multi-Agent Large Language Models Framework. In Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pages 418–439, Suzhou, China. Association for Computational Linguistics.

[2] Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. 2024. Improving factuality and reasoning in language models through multiagent debate. In Proceedings of the 41st International Conference on Machine Learning (ICML'24), Vol. 235. JMLR.org, Article 467, 11733–11763.

[3] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient Memory Management for Large Language Model Serving with PagedAttention. In Proceedings of the 29th Symposium on Operating Systems Principles (SOSP '23). Association for Computing Machinery, New York, NY, USA, 611–626. https://doi.org/10.1145/3600006.3613165

[4] Ye, Rui, Xiangrui Liu, Qimin Wu, Xianghe Pang, Zhenfei Yin, Lei Bai, and Siheng Chen. "X-mas: Towards building multi-agent systems with heterogeneous llms." arXiv preprint arXiv:2505.16997

[5] OpenAI, "gpt-oss-120b & gpt-oss-20b Model Card," *arXiv e-prints*, Art. no. arXiv:2508.10925, 2025, doi: 10.48550/arXiv.2508.10925.

[6] NVIDIA, "NVIDIA Nemotron 3: Efficient and Open Intelligence," White Paper, 2025. [Online]. Available: https://arxiv.org/abs/2512.20856

[7] NVIDIA, "Nemotron 3 Nano: Open, Efficient Mixture-of-Experts Hybrid Mamba-Transformer Model for Agentic Reasoning," Technical Report, 2025. [Online]. Available: https://arxiv.org/abs/2512.20848

[8] Qwen Team, "Qwen3.5: Towards Native Multimodal Agents," Feb. 2026. [Online]. Available: https://qwen.ai/blog?id=qwen3.5