EE 451
Course Project Proposal
Due: October 20th, 2024
Kae Sawada
SID: 6598172388

Parallelization of Scientific Realtime Generative AI Response Verification System for Continuous Quality Assurance in Safety-Critical Software

Introduction

The integration of Artificial Intelligence (AI) into safety-critical systems requires rigorous, continuous quality assurance to mitigate risks associated with autonomous decision-making errors. Traditional serial and manual approaches to AI output verification are not sufficient due to the real-time nature and complexity of such systems.
For autonomous systems, a separate AI system could be used to supervise another system. Once the primary AI produces its response, the secondary AI would analyze the response related to the features of a correct statement. This proposal introduces a parallelization of such verification system that utilizes multiple Large Language Models (LLMs) designed to enhance throughput, and scalability of the real-time verification pipelines in high-reliability applications.

By parallelizing the generative AI output verification system, while maintaining or enhancing decision-making accuracy, compared to a serial implementation, this approach will enable continuous quality assurance with improved decision-making accuracy, significantly boosting system efficiency and capacity compared to serial implementations. Leveraging task-level and data-level parallelism, the system can achieve superior efficiency through optimized memory hierarchy usage, data locality, and reduced communication overhead between cores/processors. Theoretical analysis of upper and lower bounds on performance gains, guided by Amdahl's and Gustafson's Laws, will demonstrate the cost-optimal parallel algorithm, as measured by token processing rate, speed-up ratios, resource utilization, and execution time. As time permits, architectural considerations, such as multicore or GPU efficiency, will be addressed to ensure scalability and high throughput.

Description of Implementation

Architecture Overview

The system architecture will leverage multi-agent LLMs operating together to evaluate and verify AI-generated outputs. Using task-level and data-level parallelism, the system effectively utilizes multiple cores, optimizing memory hierarchy and data locality to minimize communication overhead and latency.

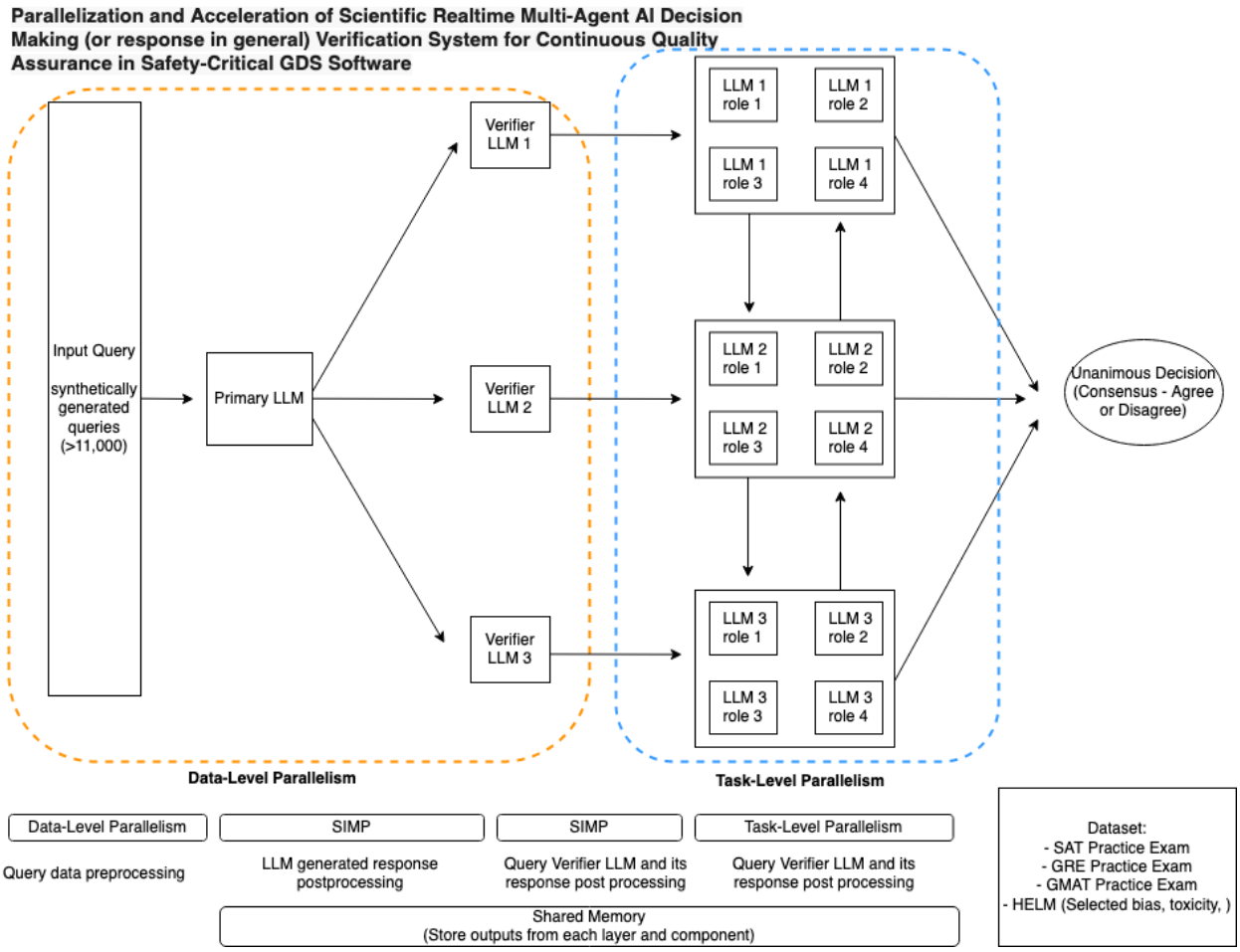

Figure 1: The diagram shown summarizes the system architecture of the verification process simulation pipelines. The system consists of 3 stage simulation, input layer that comprises of data ingestion and the output of the LLM under test. The second layer consumes the primary LLM's output, independently decide whether it agrees with the primary LLM's outputs or not. In the third layer, the three LLM will play different roles to mimic a panel of experts and non-experts. The output of this layer will be

Parallelization Strategy

Task-Level Parallelism: Tasks such as syntax verification, semantic coherence assessment, and fact-checking are assigned to distinct processors. This allows reduction on the overall verification time. Let T_{serial} be the time it takes for one processor to perform all tasks serially. If $T_{task\ i}$ represents the time for task i, then under parallel execution with n processors, the expected time $T_{parallel}$ can be approximated by:

$$T_{\text{parallel}} \approx \max(T_{\text{task 1}}, T_{\text{task 2}}, \ldots, T_{\text{task n}})$$

Where each $T_{task\ i}$ is the maximum time, any processor spends on a single task.
Data-Level Parallelism: To maximizing the usage of computational resources and minimizing idle times, input data will be partitioned and distributed across processors. Each processor will work independently on its partitioned data.
Technological Stack: Programming Languages used is Python for LLM integration and management, with critical performance segments implemented in C++ for efficiency.

Parallel Computing Tools: MPI for Python (mpi4py) for managing data distribution and task coordination, and OpenMP for handling intra-node parallelism effectively.

Evaluation

Metrics for Evaluation:
Throughput will be measured in number of tokens processed per second to assess how effectively the system handles large volumes of data.
Speedup will be calculated by comparing the execution times of parallel versus serial implementations, capturing the efficiency gains from parallelization and scalability of it. As mentioned below, we will use Amdahl's Law and Gustafson's Law to evaluate the obtained speedup and its theoretical backing.
Resource Utilization will be monitored to ensure all available CPU/GPU resources are being utilized effectively and minimizing waste.
Execution Time: Total time taken for processing the set amount of input (set number of tokens in the input). This system is meant for a real-time system monitoring, hence, delay tolerance is minimal.
Theoretical Performance Bounds: Using Amdahl's and Gustafson's Laws, we will establish theoretical upper bounds on performance gains.
Efficiency (E) will be measured to optimize the number of cores used for the proposed pipeline.

$$E = \frac{Speedup}{Par\#\ of\ processors\ used}$$

Amdahl's Law will be used to compute the theoretical upper bound expected speedup for a given fixed workload. This provides a theoretical benchmark for assessing the scalability limits imposed by the serial portions of the task. Mathematically, Amdahl's Law is expressed as

$$S_{speedup} = \frac{Serial\ Time}{Parallel\ Time} = \frac{S+P}{S+\frac{P}{f}} = \frac{1}{S+\frac{(1-S)}{f}} \leq \frac{1}{S}$$

where S id the serial portion of the program, P is the parallelizable portion of the program, f is the number of processor (or speedup factor), and S + P = 1.
Gustafson's Law will be applied to predict scalability when the workload increases with the number of processors, which is more reflective of the expanding data environments in real-time systems.

$$Scaled\ Speedup = \frac{Serial\ Time}{Parallel\ Time} = \frac{S'+P'p}{S'+\mathrm{P}'} = \frac{S'+(1-S')p}{S'+\mathrm{P}'} = \frac{S'+(1-S')p}{1}$$
$$= S'+(1-S')p \approx\ (1-S')p$$

Experimental Setup

A series of experiments will be designed to simulate safety-critical scenarios where AI decision-making is crucial. The system will be evaluated under controlled conditions with predefined datasets like SAT and GRE practice exams, alongside the HELM dataset for measuring societal acceptance and the correctness of AI decisions.

Reference

Prasanna, Victor 2024. "EE/CSCI 451: Parallel and Distributed Computation Lecture #13." Department of Computer Science, University of Southern California, October 15. https://piazza.com/class_profile/get_resource/lzucri269tu70c/m2f67i9s3zs2ec.
Prasanna, Victor 2024. "EE/CSCI 451: Parallel and Distributed Computation Lecture #14." Department of Computer Science, University of Southern California, October 17. https://piazza.com/class_profile/get_resource/lzucri269tu70c/m2f67i9s3zs2ec.

Wickramasinghe, Sachini 2024. "EE/CSCI 451: Introduction to Parallel and Distributed Computation Discussion #6." University of Southern California, October 8. https://piazza.com/class_profile/get_resource/lzucri269tu70c/m2f67i9s3zs2ec.

Liang, Percy, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, et al. “Holistic Evaluation of Language Models.” arXiv, October 1, 2023. https://doi.org/10.48550/arXiv.2211.09110.

Wu, Qingyun, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, et al. “AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation.” arXiv, October 3, 2023. http://arxiv.org/abs/2308.08155.

David Culler, Richard IQ, Abhijit Sahay, Klaus Erik Schauser, Eunice Santos, Ramesh Subramonian, and Thorsten von Eicken
“LogP: Towards a Realistic Model of Parallel Computation,” n.d., Computer Science Division, University of California Berkeley

Barbic, Jernej. “Multi-Core Architectures,” 15-213, 2007, May 3rd https://www.cs.cmu.edu/~fp/courses/15213-s07/lectures/27-multicore.pdf

“PJM - System Operations.” Accessed October 9, 2024. https://www.pjm.com/markets-and-operations/ops-analysis.

Barbic, Jernej. “Multi-Core Architectures,” n.d.

“Stanford Large Network Dataset Collection.” Accessed October 9, 2024. http://snap.stanford.edu/data/index.html.

Weng, Yixuan, Minjun Zhu, Fei Xia, Bin Li, Shizhu He, Shengping Liu, Bin Sun, Kang Liu, and Jun Zhao. “Large Language Models Are Better Reasoners with Self-Verification.” arXiv, October 19, 2023. http://arxiv.org/abs/2212.09561.

Shen, Shannon Zejiang, Hunter Lang, Bailin Wang, Yoon Kim, and David Sontag. “Learning to Decode Collaboratively with Multiple Language Models.” arXiv, August 27, 2024. http://arxiv.org/abs/2403.03870.

Tihanyi, Norbert, Mohamed Amine Ferrag, Ridhi Jain, and Merouane Debbah. “CyberMetric: A Benchmark Dataset for Evaluating Large Language Models Knowledge in Cybersecurity.” arXiv, February 12, 2024. http://arxiv.org/abs/2402.07688.

Huang, Jun, Jiawei Zhang, Qi Wang, Weihong Han, and Yanchun Zhang. “Exploring Advanced Methodologies in Security Evaluation for Large Language Models,” n.d.

Smith, Eric Michael, Melissa Hall, Melanie Kambadur, Eleonora Presani, and Adina Williams. “‘I’m Sorry to Hear That’: Finding New Biases in Language Models with a Holistic Descriptor Dataset.” arXiv.org, May 18, 2022. https://arxiv.org/abs/2205.09209v2.

Solat, Siamak, and Farid Naït-Abdesselam. “Parallel Committees: High-Performance, Scalable, Secure and Fault-Tolerant Data Replication Using a Novel Sharding Technique.” In *2023 Fifth International Conference on Blockchain Computing and Applications (BCCA)*, 546–53. Kuwait, Kuwait: IEEE, 2023. https://doi.org/10.1109/BCCA58897.2023.10338937.

Zheng, Xi, Aloysius K. Mok, Ruzica Piskac, Yong Jae Lee, Bhaskar Krishnamachari, Dakai Zhu, Oleg Sokolsky, and Insup Lee. “Testing Learning-Enabled Cyber-Physical Systems with Large-Language Models: A Formal Approach.” arXiv, January 15, 2024. https://doi.org/10.48550/arXiv.2311.07377.

“Universal and Transferable Attacks on Aligned Language Models.” Accessed April 10, 2024. http://llm-attacks.org/.

UpTrain AI. “Decoding Perplexity and Its Significance in LLMs,” December 4, 2023. https://blog.uptrain.ai/decoding-perplexity-and-its-significance-in-llms/.

Plimpton, Steve. “Fast Parallel Algorithms for Short-Range Molecular Dynamics.” *Journal of Computational Physics* 117, no. 1 (March 1, 1995): 1–19. https://doi.org/10.1006/jcph.1995.1039.