CS406-CS531: Performance Engineering with AI

In this project, you will use ChatGPT, which is a state-of-the-art large language model, (or another AI model/tool) to analyze and evaluate its capabilities in detecting and classifying performance-related bugs in a sequential/parallel C++ code and in fixing parallel, performance inefficient OpenMP and CUDA codes. As mentioned above, you can try other models/tools; e.g., OpenAI Codex, Microsoft Copilot, etc. The purpose of this project is to understand and evaluate the state of AI-based performance engineering - the more you prove the usefulness of the tool(s) you choose the better. This being said, ChatGPT (maybe also with some plugins) currently seems to be the best candidate. Here are some rules:

1) Each team can have 2 or 3 students.
2) CS406 students can team up with CS531 students.
3) There is a progress report and a final report - reports must be in LaTeX.

Project Overview

There are two questions we are interested in:

- **Part 1: Performance Bug Detection and Classification with ChatGPT**

  In this part of the project, you will use ChatGPT to detect and classify performance-related bugs in a given C++ code - which can be sequential, OpenMP-based (CPU parallel) or CUDA-based (GPU Parallel). A good set of (sub)classes you can use for classification are given in the accompanying research paper which will be published soon (check RQ1 in the paper).

- **Part 2: Fixing the Performance of a Parallel OpenMP/CUDA Code**

  In this part of the project, you will evaluate the capabilities of AI on fixing performance bugs. There will be two sources for these performance bugs; the first source is the codes that you will/can write. These can be based on and/or similar to the example codes we have seen in the lectures (which you can find at SuCourse). The second source is the dataset of the accompanying paper. You can find the dataset as a zip file accompanying this document (check RQ2 and RQ3 in the paper).

In this project, sky's the limit; the more effort you put into it, the better the results. There will be many bonus points available and if the results are interesting, it may also yield a research paper in a workshop/conference. So if you want to work in large language models (or using them for high-performance computing), do your best. **At the end of the course, your project report will be evaluated and graded by comparing it with the quality of other projects' reports.**

There are multiple ways/levels to interact with the ChatGPT model (and there are multiple ChatGPT models). You can use only one, start from the first and gradually increase the level of interaction.

- You can use the web interface, prompts and extract all the information you can.
  - Use **perf** or any other performance data extractor tool to get performance related data for the code and feed it to ChatGPT.
  - Also describe the architecture to ChatGPT.
  - Measure the difference in quality after every step.
- You can use the ChatGPT APIs provided by OpenAI and automate the process: (https://openai.com/blog/introducing-chatgpt-and-whisper-apis). You may need funding for this one and we will compensate for testing purposes.
- You can train your own model with a large corpus of performance bugs given in the paper accompanying the project document. The dataset generated may need to be further curated, and then used. This fine tuned model can be used to detect performance-related bugs in the code and suggest possible fixes.

Deadlines:

Progress report: April 30, 2023

Final report: May 31, 2023