# **LASSI-EE**: Leveraging LLMs to Automate Energy-Aware Refactoring of Parallel Scientific Codes

*Matthew T. Dearing, Yiheng Tao, Xingfu Wu, Zhiling Lan, Valerie Taylor*

UIC UNIVERSITY OF ILLINOIS CHICAGO College of Engineering

Argonne NATIONAL LABORATORY

## ABSTRACT

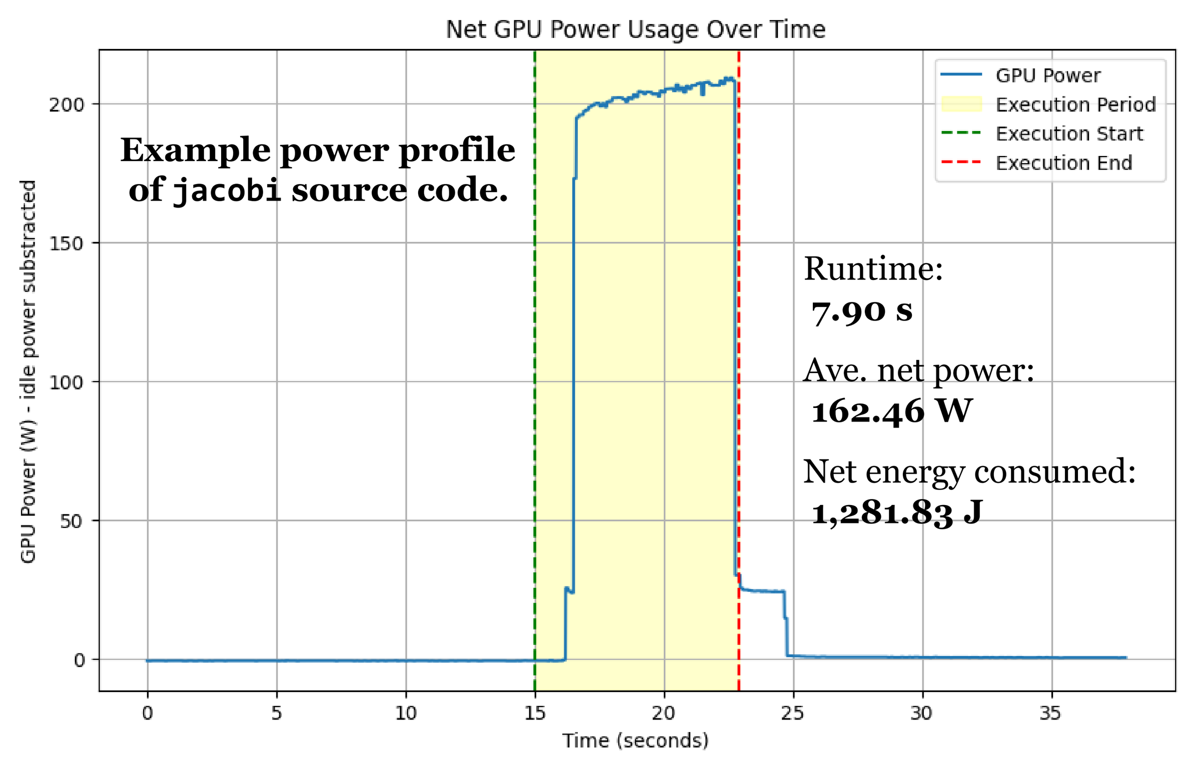
While large language models (LLMs) are increasingly used for generating parallel scientific code, most current efforts emphasize functional correctness, often overlooking performance and energy considerations. In this work, we propose LASSI-EE, an automated LLM-based refactoring framework that generates energy-efficient parallel code on a target parallel system for a given parallel code as input.

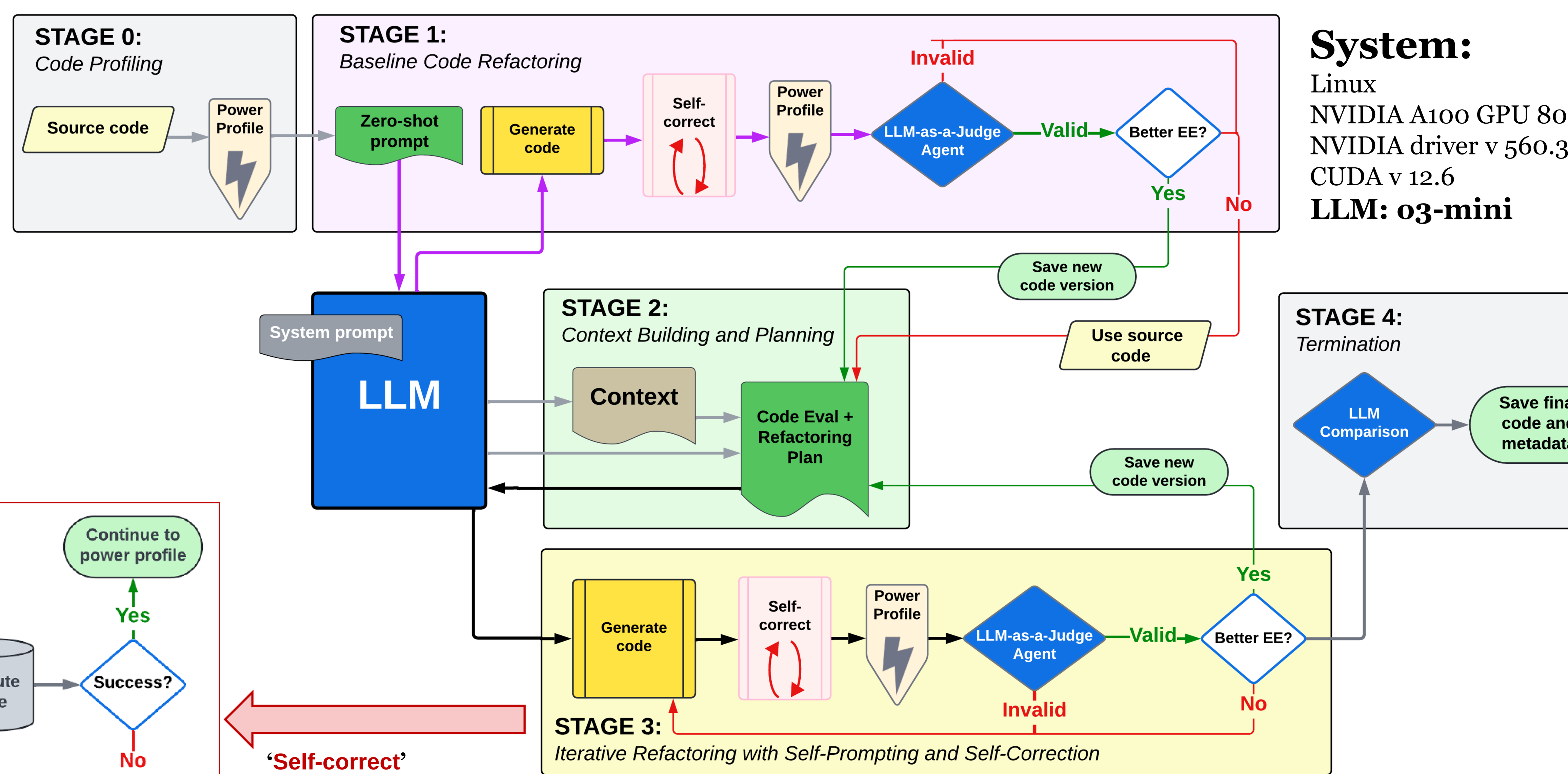
Through a multi-stage, iterative pipeline process, **LASSI-EE achieved an average energy reduction of 47% across 85% of the 20 HeCBench benchmarks** tested on NVIDIA A100 GPUs. Our findings demonstrate the broader potential of LLMs, not only for generating correct code but also for enabling energy-aware programming. We also address key insights and limitations within the framework, offering valuable guidance for future improvements.

## LASSI FRAMEWORK

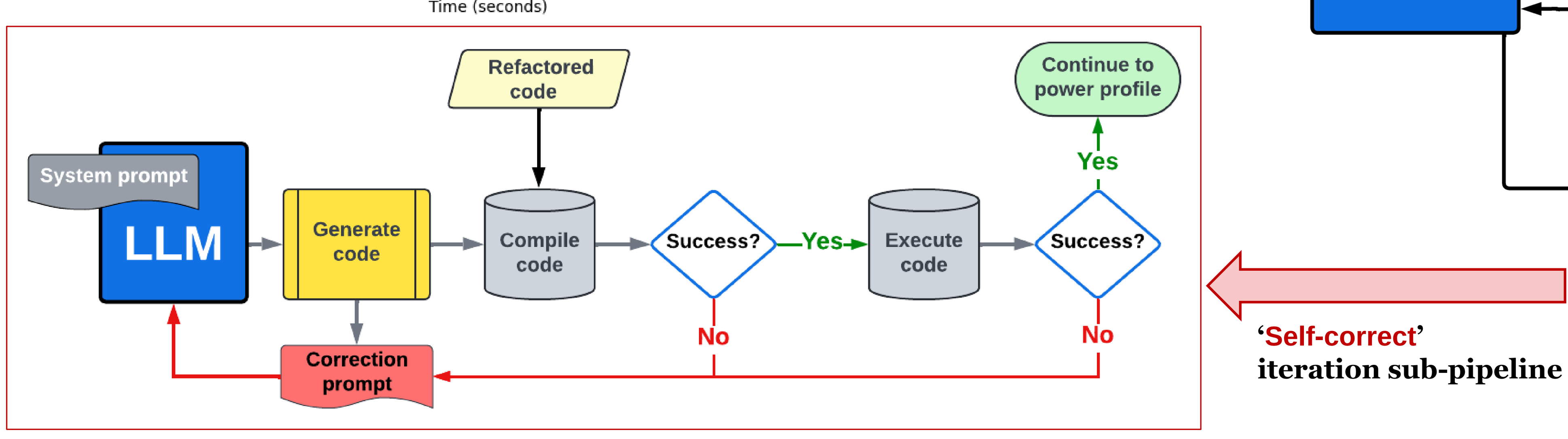
*A novel code generation framework:*

- **L**LM-based **A**utomated **S**elf-correcting pipeline for generating parallel **S**c**I**entific codes
- Augmented context through **self-prompting**.
- Feedback from code compilation and execution for **self-correction**.
- Previous version for code translation [1].
- Code runtime power profiling, LLM-as-a-Judge Agent.
- Strategy toward consistent energy-efficient parallel code refactoring: *combine* the base LLM capability (Stage 1) with the context-infused, iterative, and self-prompted LLM results (Stage 3).




**System:**
Linux
NVIDIA A100 GPU 80 GB
NVIDIA driver v 560.35.05
CUDA v 12.6
**LLM: o3-mini**




**'Self-correct' iteration sub-pipeline**

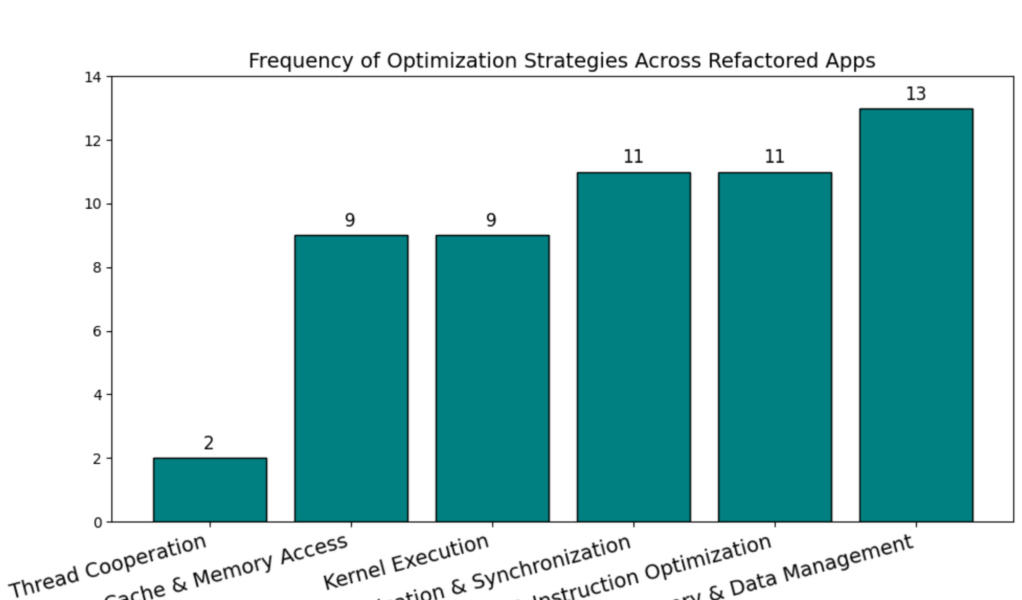
**HeCBench codes** [2]

- Open-source heterogeneous apps in multiple languages.
- Runtime args selected for measurable power draw.
- Codes compiled with:

```
nvcc -std=c++14 -Xcompiler -Wall -arch=sm_80 -O3
```

| *Category* | *Application* | **Lines of Code** | **Runtime args** |
|---|---|---|---|
| Bandwidth | randomAccess | 158 | [10] |
| Bioinformatics | all-pairs-distance | 328 | [10000] |
| Computer vision and image processing | colorwheel<br>marchingCubes* | 154<br>574 | [10000, 8, 1]<br>[100] |
| Cryptography | chacha20* | 130 | [300000] |
| Data compression and reduction | segment-reduce | 95 | [16384, 100] |
| Data encoding, decoding, or verification | entropy*<br>murmurhash3 | 171<br>245 | [10000, 1024, 1]<br>[750000, 500] |
| Graph and Tree | floydwarshall | 295 | [2048, 100, 256] |
| Language and kernel features | layout<br>threadfence | 197<br>142 | [2500]<br>[100, 250000000] |
| Machine learning | dense-embedding | 193 | [10000, 8, 1] |
| Math | jacobi†<br>jaccard<br>matrix-rotate‡ | 235<br>417<br>67 | None<br>[1024, 512, 1000]<br>[30000, 100] |
| Search | bsearch<br>keogh* | 279<br>143 | [10000, 1]<br>[256, 22500000, 100] |
| Signal processing | extrema | 349 | [750] |
| Simulation | lid-driven-cavity<br>pathfinder | 1079<br>286 | None<br>[10000, 1000, 1000] |

* Code includes a dependency that was not considered during refactoring.
†Because this app did accept runtime arguments, we modified the thread block size multiplier from 2048 to 16384 in the source code.
‡ App contains parallel and serial components, so the serial portion was removed.




- Strategy categories of optimization techniques implemented by the 17 refactored codes with energy savings, as identified by the LLM-as-a-Judge agent.
- Over half of the apps implement four or more optimization strategies.

## LASSI-EE Pipeline Results for Generating Energy-aware Codes

| *App name* | Source code Energy (J) | Ave Power (W) | Runtime (s) | Best Refactored Code Energy (J) | Ave Power (W) | Runtime (s) | % diff from Source Code Energy (J) | Ave Power (W) | Runtime (s) | Code Version | Corrections Count | Refactor Count |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| randomAccess | 500.01 | 37.26 | 13.43 | 212.89 | 36.20 | 5.89 | -57.42% | -2.84% | -56.14% | 8 | 5 | 11 |
| all-pairs-distance | 899.31 | 94.07 | 9.56 | 771.57 | 175.75 | 4.39 | -14.20% | 86.83% | -54.08% | 6 | 2 | 10 |
| colorwheel | 2,527.68 | 124.33 | 20.34 | 157.43 | 60.78 | 2.59 | -93.77% | -51.11% | -87.27% | 4 | 1 | 5 |
| marchingCubes† | 386.91 | 37.27 | 10.39 | N/A | N/A | N/A | N/A | N/A | N/A | N/A | 10 | 6 |
| chacha20 | 377.35 | 35.07 | 10.76 | 93.91 | 28.28 | 3.33 | -75.11% | -19.36% | -69.05% | 4 | 4 | 8 |
| segment-reduce | 539.94 | 50.23 | 10.76 | 325.36 | 93.49 | 3.49 | -39.74% | 86.12% | -67.57 | 6 | 0 | 10 |
| entropy | 1,625.87 | 93.39 | 17.41 | 231.14 | 23.30 | 10.04 | -85.78% | -75.05% | -42.33% | 10 | 1 | 11 |
| murmurhash3 | 1,380.36 | 96.43 | 14.31 | 1,252.43 | 81.91 | 15.29 | -9.27% | -15.08% | 6.85 | 3 | 1 | 7 |
| floydwarshall | 622.39 | 65.58 | 9.5 | 607.77 | 65.92 | 9.22 | -2.35% | 0.52 | -2.95% | 1 | 10 | 6 |
| layout | 1,150.25 | 104.47 | 11.02 | 1,187.05 | 109.61 | 10.83 | 3.20% | 4.92% | -1.72% | N/A | 0 | 6 |
| threadfence | 605.58 | 37.71 | 16.07 | 55.82 | 41.34 | 1.35 | -90.78% | 9.63% | -91.60% | 4 | 0 | 8 |
| dense-embedding | 996.27 | 28.61 | 34.82 | 1,215.41 | 27.60 | 44.04 | 22.00% | -3.53% | 26.48 | N/A | 0 | 6 |
| jacobi | 1,281.83 | 162.46 | 7.90 | 656.61 | 99.79 | 6.59 | -48.78% | -38.58% | -16.58% | 6 | 2 | 10 |
| jaccard | 2,034.60 | 91.36 | 22.27 | 1174.46 | 84.74 | 13.86 | -42.28% | -7.25% | -37.76% | 4 | 0 | 8 |
| matrix-rotate | 623.14 | 66.08 | 9.43 | 115.74 | 26.73 | 4.33 | -81.43% | -59.55% | -54.08% | 5 | 0 | 9 |
| bsearch | 2,526.86 | 235.05 | 10.76 | 11.12 | 16.35 | 0.68 | -99.56% | -93.04% | -93.68% | 3 | 0 | 7 |
| keogh | 1,159.41 | 28.05 | 41.34 | 949.94 | 27.50 | 34.55 | -18.07% | -1.96% | -16.42% | 4 | 1 | 6 |
| extrema | 850.71 | 92.97 | 9.15 | 403.28 | 66.33 | 6.09 | -52.59% | -28.65% | -33.44% | 0 | 0 | 6 |
| lid-driven-cavity | 860.71 | 62.78 | 13.72 | 531.22 | 75.89 | 7.01 | -38.28% | 20.88 | -48.91% | 8 | 5 | 11 |
| pathfinder | 2,062.25 | 89.47 | 23.06 | 1,460.15 | 74.69 | 19.55 | -29.20% | -16.52% | -15.22% | 6 | 2 | 10 |

(1) One exemplar result of three trials per code is reported in the table.
(2) N/A represents that the pipeline did not generate a more energy efficient refactored code compared to the source code.
†The pipeline could not generate any codes that were functional equivalent to the source code before reaching the maximum number of tries configured for our experimental setup.

## EXAMPLE CODE CHANGE

`threadfence`

**(a) Source Code**

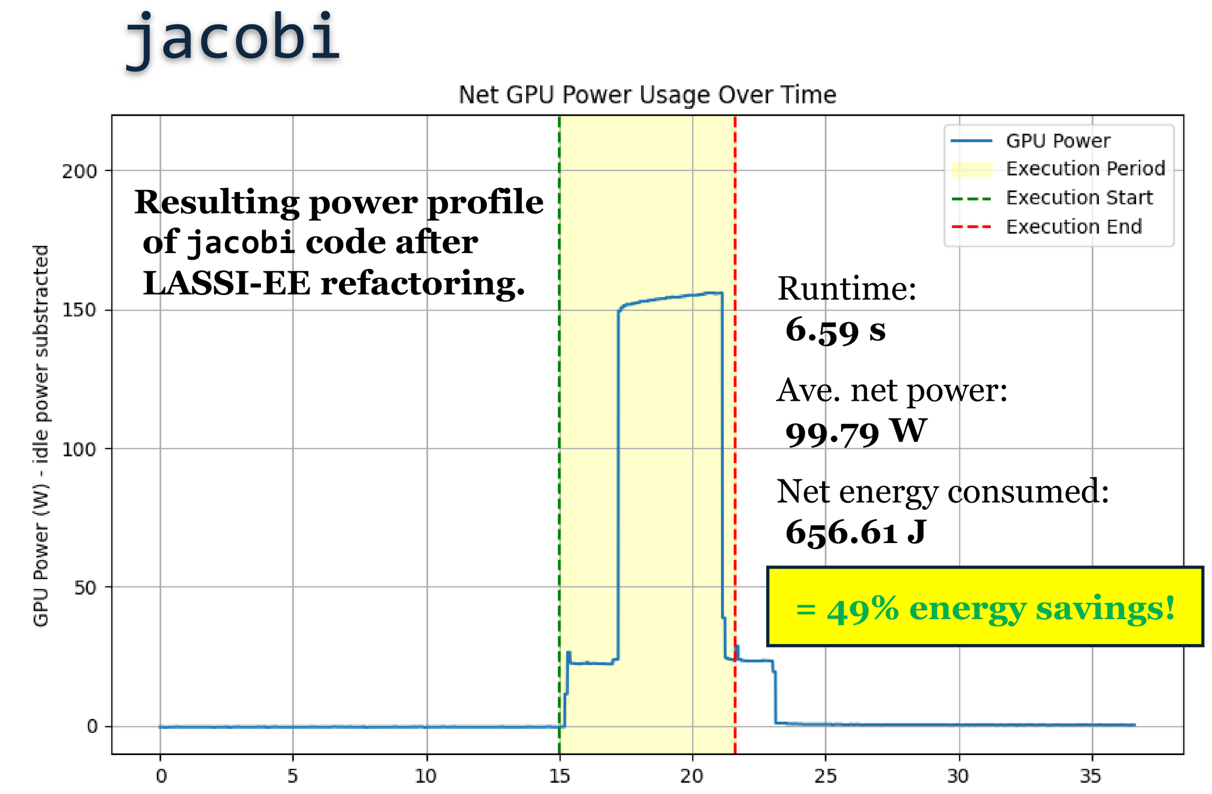
```
for (int n = 0; n < repeat; n++) {
    cudaMemcpy(d_array, h_array, N * sizeof(float),
          cudaMemcpyHostToDevice);
    cudaDeviceSynchronize();
    auto start = std::chrono::steady\_clock::now();

    <snip>
}
```

**(b) Refactored Code**

```
// Copy the constant input array from host to device only
   once.
cudaMemcpy(d_array, h_array, N * sizeof(float),
      cudaMemcpyHostToDevice);

// Create CUDA events for timing.
cudaEvent_t start, stop;
cudaEventCreate(&start);
cudaEventCreate(&stop);

// Run the kernel "repeat" times.
for (int n = 0; n < repeat; n++) {
    // Record the start event.
    cudaEventRecord(start, 0);

    <snip>
}
```

## POWER PROFILING

`jacobi`




This work is supported in part by US National Science Foundation grant CCF-2413597.
This material is based upon work supported by the U.S. Department of Energy, Office of Science, under contract number DE-AC02-06CH11357, at Argonne National Laboratory.

## REFERENCES

[1] Dearing, M. T., et al. "LASSI: An LLM-Based Automated Self-Correcting Pipeline for Translating Parallel Scientific Codes," in *2024 IEEE International Conference on Cluster Computing Workshops* (CLUSTER Workshops), Sep. 2024, pp. 136–143.

[2] Z. Jin et al., (2023) "A Benchmark Suite for Improving Performance Portability of the SYCL Programming Model." *IEEE IS-PASS*, pp. 325-327.

https://github.com/SPEAR-UIC/LASSI