FC with RG – Novel HPC Gaudi 3

Joseph Miao, Shang En Sim, Erin Tan, Jeremy Wang

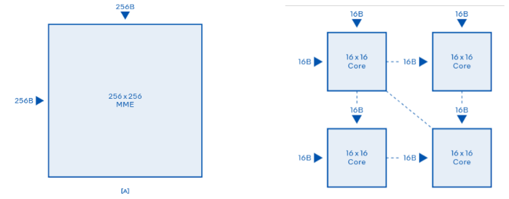
Background

Intel Gaudi 3: Intel's latest AI Accelerator (launched Sep 2024), presented as a cost-efficient alternative to GPUs

- 1.1-2.5x cost per dollar vs NVIDIA H100
- Academic literature is sparse

Architecture: not a GPU! Large Matrix Multiplication Engines reduce the input bandwidth for the same number of operations

Software suite: Optimum Habana



Goals/Planning/Milestones

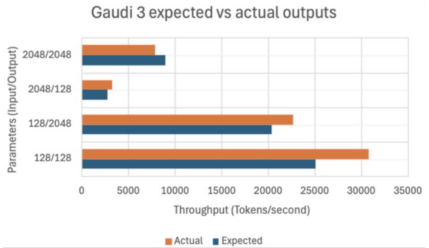
Goal: Evaluate the performance and memory usage of Intel Gaudi 3 for LLM inference and compare against Intel Xeon Max.

Other Targets: Tutorials for setup and Llama authentication

Setup Challenges

- No access to machine until week 10
- Setup/permissions issues resolved week 13
- We had issues with `HABANA_LOGS` pointing to locations where the user did not have read/write permissions. This caused `SEGFAULT`
- We could not modify the habana-torch environment without cloning it.

Initial Results



Gaudi 3 Benchmarks (instructions from Intel's model performance data)

- All lengths except 2048/2048 outperformed Intel's results (firmware optimizations)

vLLM: Replication difficulties (8B - 5379.02 tks/s)

- More work to be done

Power Analysis: Not possible unless given root access

- Approximate with hl-smi

Discussion

Preliminary Results:

Improvements may be due to an improved CPU (Xeon 6972P vs. Xeon 8480+) or software/firmware/optimization updates in the interim. Unsure what causes lower values for longer inputs/outputs

Literature Overview:

Could not find journal/conference papers on the Gaudi 3. Gaudi 2 has competitive TF/W vs. H100 and is faster in decode of LLM but lacks software support.

Lessons Learned and Next Steps

What Could Be Improved

- Semester planning/organization
- Communication with instructors
- Working around delays

Next Semester Goals

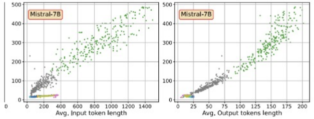
- Extend benchmarking to additional LLM models
- Deepen analysis of vLLM performance
- Further research power efficiency on Gaudi 3
- Maintain biweekly technical syncs with Intel team

FC with RG – Novel HPC Power Analysis

Ibhan Aggarwal, Bryson Bennett, Lucas Goldfarb, Charles McHenry, Srinidhi S P

Background

- **Intel Xeon Max Series SPR** - for HPC (64 GB of HBM, 48 cores, 96 threads, and 105 MB of cache)
- **Memory modes**: HBM-only, Flat, Cache. Clustering: Quad and SNC4. (4 dash nodes available)
- **Intel AMX** allow CPUs to accelerate matrix multiplications, significantly improving LLM inference speed—**up to 6.3× throughput** gain vs. older CPUs.
- **Motivation**: Training → resource-intensive, inference → 90% of compute load
- CPUs use less power with larger and expandable memory to handle large models and KV caches without offloading
- Analyzed trends in inference energy usage from CPU/GPU systems to compare to our CPU only system

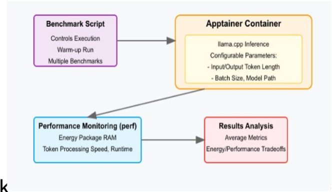


Goals/Planning/Milestones

- Evaluate the **power/energy consumption** of the Intel Xeon Max SPR across memory modes and inference params (batch size, input/output token length etc.)
- Develop Apptainer scripts to **easily port testing** across different configurations
- Compare findings to existing results and analyze trends

Workflow

- Apptainer image executes inference on **llama.cpp** with desired parameters
- Bash script performs multiple runs of this Apptainer image and uses **perf** to benchmark power consumption


Benchmark Script
Controls Execution
Warm-up Run
Multiple Benchmarks

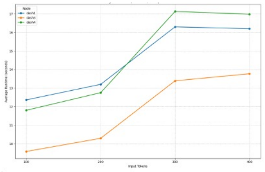
Apptainer Container
llama.cpp Inference
Configurable Parameters:
- Input/Output Token Length
- Batch Size, Model Path

Performance Monitoring (perf)
Energy Package RAM
Token Processing Speed, Runtime

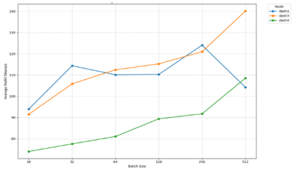
Results Analysis
Average Metrics
Energy/Performance Tradeoffs

Initial Results

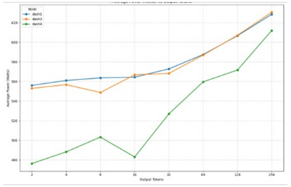
Average Runtime vs Input Tokens



Average Prefill tokens/s vs Batch Size



Average Power vs Output Tokens



Discussion

Input tokens:

- Average Energy Ram, Energy Package, and Runtime increase initially, but appear to taper off as input token length increases

Output tokens:

- Average Energy Ram, Energy Package, and Runtime exponentially increase as output token length increases

Batch Size:

- Average Energy Ram, Energy Package, and Runtime decrease as batch size increases
- Average Power peaks around batch size of 64-128, showing optimal hardware utilization

Lessons Learned and Next Steps

- Developed a benchmarking workflow in order to get power statistics from LLM inference
- Analyzed trends in CPU only inference as they relate to power consumption

Next Steps:

- Explore the impact (power and performance) of using numactl to memory bind to certain numa domains
- Look into utilizing OpenMPI to accelerate inference on the Xeon Max SPR
- Compare power results across generations of Intel CPUs