GPU Performance Tuning and Power Efficiency on the DGX A100 Cluster

Khanin Udomchoksakul
Electrical and Computer Engineering
Carnegie Mellon University
Pittsburgh, USA
kudomcho@andrew.cmu.edu

Orathai Sangpetch
Carnegie Mellon-KMITL Thailand Program
CMKL University
Bangkok, Thailand
orathai@cmkl.ac.th

Akkarit Sangpetch
Carnegie Mellon-KMITL Thailand Program
CMKL University
Bangkok, Thailand
akkarit@cmkl.ac.th

***Abstract*—The complexity of current Deep learning has been growing rapidly nowadays. Such advancement allows various organizations such as private sectors and government to leverage intelligent systems on their use cases. High Performance Computing (HPC) infrastructure nowadays has pivoted to GPU-oriented systems, enabling developers and researchers to train complex models with large datasets unlike conventional clusters equipped only with CPU cores. However, focus on power efficiency on the HPC system has not been prevalent especially on the new system such as DGX A100 that does not have datapoints on how GPUs consumed power. Even though such HPC cluster can be powerful, always allowing it to run at the maximum capacity results to financial cost to the HPC provider at the end. Therefore, for any organization providing the system, it is crucial for them to balance the cluster capabilities while maintaining overall power consumption which can potentially be costly in the long term. This paper reveals A100 GPU metrics that are relevant to Power usage and explains GPU profiling applied to Deep learning workload on the cluster, saving up to 32% of the power usage while compromising only 11.5% of training time compared to a default profile. Then, the paper investigates literature review that could be learned further adopted to the current system at CMKL university as the next milestone.**

***Index Terms*— HPC, Deep Learning, Power Efficiency, GPU clock speed profiling, Data Analysis and interpretation, Power Management, Power consumption.**

I. Introduction

The emergence of the intelligent systems attracts organizations seeking smart solutions to solve complex problems. Particularly when businesses or organizations heavily rely on data, there is no denying that these sectors pivot their interest in advancements such as Machine learning or Deep learning to resolve complex use cases. One use case collaborated with academia and private sector is using Deep learning and computer vision that aims to detect clean, intact beer bottles for recycle in the factory [1]. Learning technologies enables business or organizations to unlock their problem solving unlike never before. However, to tackle such problems requires a powerful computing station that contains resources such as GPUs and memory necessary to train and develop those system and infrastructure like HPC is desired the need.

Apex AI infrastructure maintained by CMKL University equips with 6 DGX A100 systems to serve artificial intelligence workloads, enabling research and development across industries in Thailand [2]. Nevertheless, with enormous compute comes with great power budgets. The default behavior of the A100 GPU is that its clock maintains at 1410 MHz as a maximum capacity once a workload is executed. This indicated that no dynamic adjustment exists on the GPU profiling, running statically at full maximum capacity throughout all durations which results to drawing high amount of power consumption. One perspective is that Apex approximately costs 4,100 USD for monthly operational cost. With GPU being the most power-consuming hardware on the cluster, tuning GPU power usage optimally means we are targeting the right component to save cost without losing significant performance [3]. Thus, it is clear that having a specific clock frequency appropriate to the running workloads on GPU is important to save the power consumption. We begin from an exploratory analysis is necessary to understand how default GPU profiles operate so that we can evaluate methodologies accordingly; then, system tuning with different profiles allows multiple perspectives of model performance workloads and power consumption; after that, future work that helps optimize resource scheduling or hardware optimization will be explored. Doing so helps the HPC provider to achieve up to 32% power efficiency compared to original state with the sample Deep learning workload while allowing minimal downtime during training, as well as understand how current GPU profiles behave that nowhere publishes.

II. Related Work

A. Hardware Evaluation

There are a few research findings that discovered DGX A100's behaviors in Power and Thermal metrics. In [4], Matej et al., showed performance, power consumption, and thermal behavior analysis of the Nvidia DGX-A100 system that was compared with its predecessor, Nvidia DGX-2, based on Tesla V100 GPUs, using synthetic benchmark to obtain the performance of floating-point computing units with Tensor Cores [6]. By performing Dynamic Frequency and Voltage Scaling (DVFS), they discovered that the most optimal GPU profile is 1035-1020 MHz and derived that the A100 GPU performed 51 GFLOPS/W for double-precision workload and 91 GFLOPS/W for tensor core double precision workload, concluding DGX A100 as the most power efficient server compared to its previous version. Our research work presents a

different perspective in benchmarking with a production-ready Deep learning workload that provides overall power draw in second and epoch units for each GPU profile to visualize the actual time taken and power amount required to achieve.

B. *Software Optimization*

One of the prevalent methods to optimize training time for Deep learning is to utilize special hardware such as Tensor core to accelerate training process. Nvidia, a GPU manufacturer and innovation company, proposed an alternative to allow scripts to use Tensor core using DLProf profiling tool. Their sample shows that they saved average iteration time from 588 ms to 188.4 ms after using TF32 (Tensor core). Despite that this method works for training the model, our limitation is that the HPC provider does not currently have control over the user's code when submitting jobs on their Apex infrastructure unless the provider provides templates for using DLprof that users follow. This method will be investigated further in the future work whether such procedure works on the HPC cluster.

III. PROPOSED SOLUTION

We define optimal performance & power efficiency as yielding the training results comparable to the default setting while saving power usage cost throughout the process. Budgets allocating to each user at Apex infrastructure generally include power consumption, processing time and hardware provision. Apex infrastructure usually serves researchers and students to train their models, inference prediction with large dataset, and any other intensive computation. However, Deep learning training is one of the most prevalent use cases at CMKL and this workload represents a general usage in terms of compute resources and allocated time. Therefore, any profile that reduces costs substantially and yields acceptable training outcomes is our candidate to be applied to the Apex cluster.

The methodologies follow 4 main components,

1) Exploratory Data Analysis
2) Relevant Metrics solidification
3) Different GPU Clock Speed Profiling
4) Applying Optimal power efficient

A. *Exploratory Data Analysis*

To observe behavior of the GPUs on the cluster, we used a real-world sample workload to benchmark a baseline state. Even though CMKL has various use cases such as a single GPU, multi-GPU and Multi-node usage, our focus is on the single GPU as a primary use case at CMKL university. Our workload is a production-ready CNN-based classifier workload that classify 3 states of the beer bottle as one of the CMKL projects developed by both Carnegie Mellon University and CMKL university. The number of images trained is 53,705. This workload utilized a single GPU to train and test. In our experiment, we observed training loss value and time to evaluate GPU metrics that are related to Power draw. NVIDIA-SMI [5] is a system management interface built by NVIDIA for monitoring and management GPU capabilities that allow us to measure values of each metric when running the workload as well as modify GPU clock speed and power limit.

Relevant metrics that were used are the following:

1. GPU utilization [%]
2. Memory utilization [%]
3. Current clock speed [MHz]
4. Current memory speed [MHz]
5. GPU temperature [C]
6. Memory temperature [C]
7. Memory used [MB]
8. Power draw [Watt]
9. Training seconds
10. Training Loss

When running the benchmark, we made sure that no other processes were utilizing the GPU resource we reserved. We ran the same benchmark 6 times in order to test behavior consistency. We discover that the A100 GPU equips with a default kernel profile that uses 210 MHz when idle and increases clock speed to 1095 MHz for several seconds and reach to 1410 MHz when running the workload. It was discovered that the error was less than 7%. Running benchmark with the default profile without adjusting any parameters gives these characteristics:

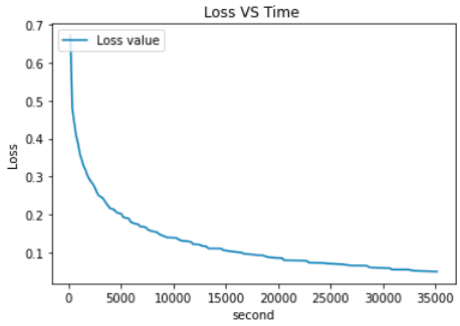

Fig 1. Cross-entropy loss value VS time taken for training the model plot

We used loss at 0.2 and 0.1 as checkpoints for power measurement because 0.2 is the most significant reduction while 0.1 shows convergence point. Note that these losses are only approximate points because some benchmarking exceed or get below to the references by 0.02.

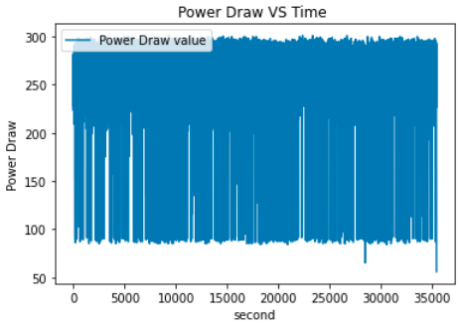

Fig 2. Power draw VS second plot

From fig.2, the power leveling can be observed that the max power draw is approximately 300 Watts. The average power draw for the default profile is 257.81 Watts throughout training time while the mode values are 0 and 264 Watts. The plot also suggests that there is steady power consumption.

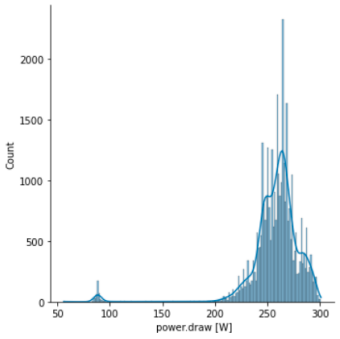

Fig 3. Power draw distribution plot

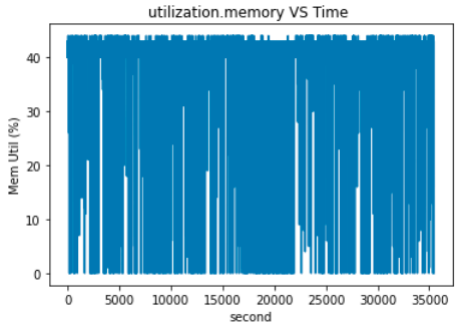

Fig 4. Memory utilization VS time plot

From fig.4, the Memory utilization on average is 41.5%, max value is 44% while the mode values are 0% and 42%.

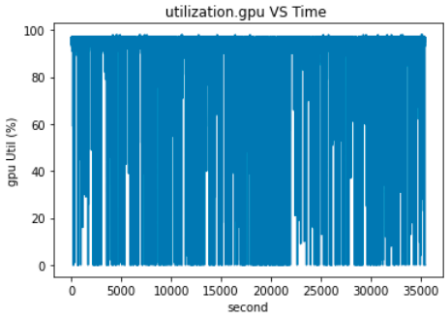

Fig 5. GPU utilization VS time plot

From fig.5, the GPU utilization on average is 93.87%, max value is 98% while the mode values are 0% and 96%.

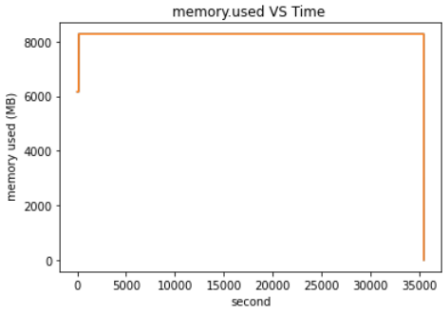

Fig 6. Memory usage VS time plot

From fig.6, we can see that this workload consumes 8272 MB when training the model. Every iteration always consumed 8 GB of the GPU memory. Figure 7 shows that GPU clock speed starts from 210 MHz as an idle state, to 1095 MHz in some seconds and eventually maintains at 1410 MHz across the training time. The memory clock speed cannot be changed for the A100 GPU, so it stays at 1215 MHz always.

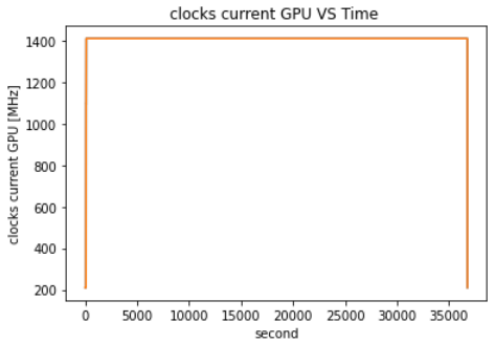

Fig 7. Current Clock speed VS time plot

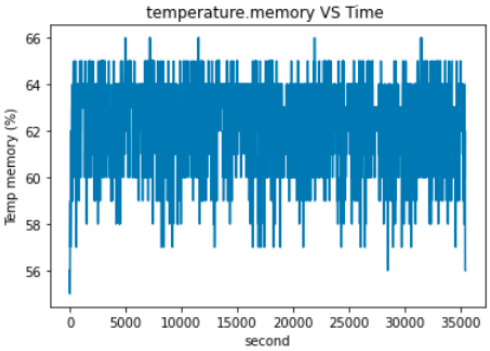

Fig 8. Memory temperature VS time plot

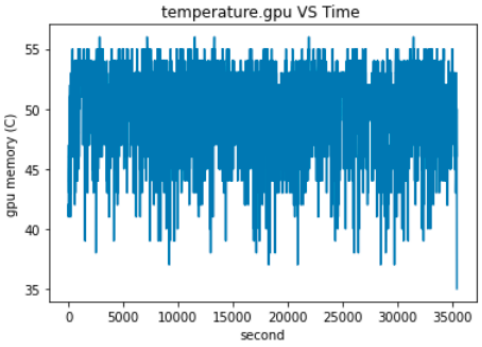

Fig 9. GPU temperature VS time plot

Temperature of CPU and memory are also captured. The average of memory temperature, max memory temperature and its mode are 62.8 C, 66 C, and 0 C and 64% respectively while the average of GPU temperature, max GPU temperature and its mode are 51.4 C, 56 C, and 0 C and 53% respectively.

B. Relevant Metrics Solidification

To visualize correlation between GPU metrics to capture what relationship we will focus on further, we generate correlation heatmap plot [5] for all relevant metrics shown in the figure 10. It can be observed that GPU and memory utilization obtain 0.61 positive correlation to the power draw followed by GPU temperature at 0.35 and memory temperature at 0.23.

Fig 10. Default profile metrics heatmap plot

We used scatter plot to see what correlation they lie into. Figure 11 shows that once the GPU and memory utilization reach to top percentage, i.e., 96-98% for GPU and 40-44% for Memory, the power instantly gets consumed to a certain level with slight fluctuation for first 1000 seconds. Figure 12 even represents the similar trend, concluding that power draw is high when GPU and memory utilization are high. Small distribution occurs when slight fluctuation of utilization drops and instantly come up.

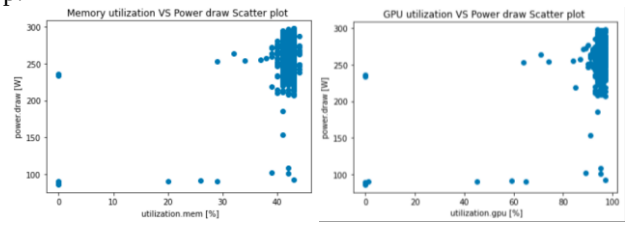

Fig 11. 1000 seconds Power Draw and Memory utilization (left) and Power Draw and GPU utilization (right) scatter plot

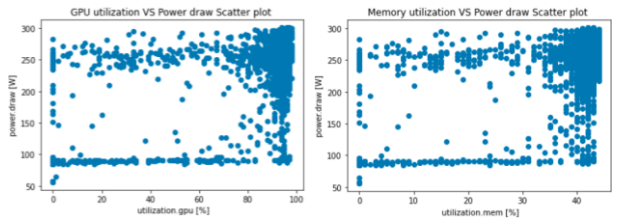

Fig 12. Total seconds Power Draw and GPU utilization (left) and Power Draw and Memory utilization (right) scatter plot

We observed that temperature can range widely when it is at the center of distribution while the edges, i.e. 66 C for GPU and 57 C for memory, have small range.

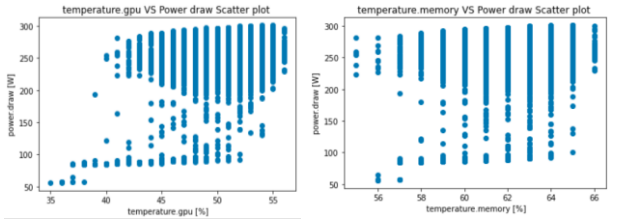

Fig 13. Total seconds Power Draw and Memory utilization (left) and Power Draw and GPU utilization (right) scatter plot

C. Different GPU Clock Speed Profiling

Speculating knobs that the GPU can be tuned, we discovered that the DGX A100 system allows GPU clock modification. This enables us to analyze tradeoffs between how fast processing the GPU can perform and how power efficient we wish to save in a static manner. Therefore, we selected GPU system clocks ranging from 795 MHz, 885 MHz, 990 MHz, 1095 MHz, 1185 MHz, 1290 MHz to 1410 MHz. We set approximately 200 MHz difference for each level because we hoped to observe wide amount of consumption difference. To see comparison between each profile, the same workload must be run over. Running the benchmark provides the following results:

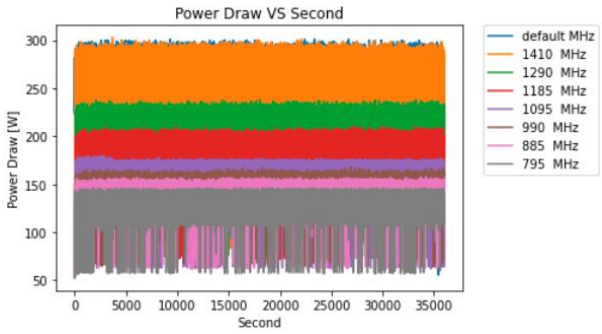

Fig 14. All clock profiles Power draw VS second plot.

Training process consumes steady power draw across all profiles, only different power magnitude. Despite clear power draw leveling on the figure 14, the values of each profile are the following:

795 MHz:
Average Power draw: 125.7 W
Max Power draw: 147 W
Mode of Power draw: 0 W, 130 W

885 MHz:
Average Power draw: 136.5 W
Max Power draw: 158 W
Mode of Power draw: 0 W, 141 W

990 MHz:
Average Power draw: 145.26 W
Max Power draw: 168 W
Mode of Power draw: 0 W, 151 W

1095 MHz:
Average Power draw: 155.9 W
Max Power draw: 181 W
Mode of Power draw: 0 W, 160 W

1185 MHz:
Average Power draw: 182.5 W
Max Power draw: 211 W
Mode of Power draw: 0 W, 187 W

1290 MHz:
Average Power draw: 205.4 W
Max Power draw: 238 W
Mode of Power draw: 0 W, 212 W

1410 MHz:
Average Power draw: 256.5 W
Max Power draw: 303 W
Mode of Power draw: 0 W, 261 W

One observation is that the default profile's power draw slightly exceeds that of 1410 MHz profile, despite having the same clock speed. It can also be noted that difference between 1410 MHz or default profile and 1290 MHz is enormous compared to other levels, consuming 20% power draw less on average. 1185 MHz draws power less than 1290 MHz by 12% on average while 1095 MHz goes down by 14.57% from 1185 MHz. However, we see that the average power consumption difference from 1095 MHz to 990 MHz is less than 7% and the rest profile below 990 MHz have power usage difference less than 9%, which is not substantial compared to higher clock

profiles. Note that all GPU and memory clock speeds stayed constant throughout the training time and used the same amount of memory on the GPU. This shows that the Apex has been using the maximum speed regardless of any workloads submitted to the cluster, and each profile yields distinct power draw leveling. In order to determine the most static optimal profile, we investigate other metric values.

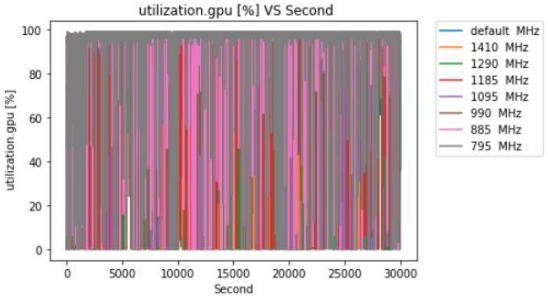

Fig 15. All clock profiles GPU Utilization VS second plot.

795 MHz:
Average GPU Utilization: 96.9 %
Max GPU Utilization: 96.9%
Mode GPU Utilization: 0%, 98 %

885 MHz:
Average GPU Utilization: 96.7%
Max GPU Utilization: 98%
Mode GPU Utilization: 0%, 97%

990 MHz:
Average GPU Utilization: 96.3%
Max GPU Utilization: 98%
Mode GPU Utilization: 0%, 97%

1095 MHz:
Average GPU Utilization: 95.7%
Max GPU Utilization: 98%
Mode GPU Utilization: 0%, 97%

1185 MHz:
Average GPU Utilization: 95.7%
Max GPU Utilization: 98%
Mode GPU Utilization: 0%, 96%

1290 MHz:
Average GPU Utilization: 95.6%
Max GPU Utilization: 98%
Mode GPU Utilization: 0%, 96%

1410 MHz:
Average GPU Utilization: 95.39%
Max GPU Utilization: 100%
Mode GPU Utilization: 0%, 96%

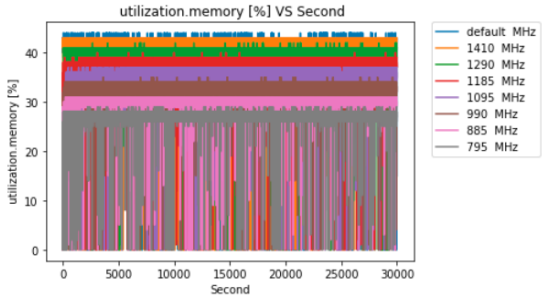

Fig 16. All clock profiles Memory Utilization VS second plot.

795 MHz:
Average Memory Utilization: 27 %
Max Memory Utilization: 29%
Mode Memory Utilization: 0%, 28 %

885 MHz:
Average Memory Utilization: 29.9 %
Max Memory Utilization: 31 %
Mode Memory Utilization: 0%, 30%

990 MHz:
Average Memory Utilization: 32.4%
Max Memory Utilization: 35%
Mode Memory Utilization: 0%, 32%

1095 MHz:
Average Memory Utilization: 35%
Max Memory Utilization: 37%
Mode Memory Utilization: 0%, 35%

1185 MHz:
Average Memory Utilization: 37.2 %
Max Memory Utilization: 40%
Mode Memory Utilization: 0%, 37 %

1290 MHz:
Average Memory Utilization: 39.6 %
Max Memory Utilization: 42 %
Mode Memory Utilization: 0%, 40%

1410 MHz:
Average Memory Utilization: 41.5 %
Max Memory Utilization: 45%
Mode Memory Utilization: 0%, 42 %

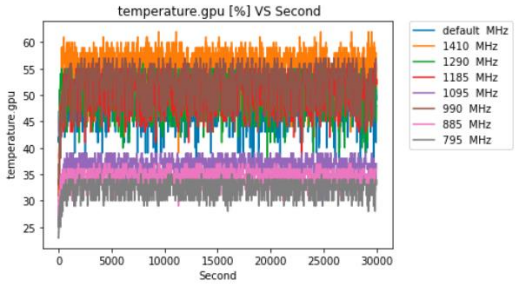

Fig 17. All clock profiles GPU temperature VS second plot.

795 MHz:
Average GPU temperature: 32.7 C
Max GPU temperature: 35 C
Mode GPU temperature: 0 C, 33 C

885 MHz:
Average GPU temperature: 34.8 C
Max GPU temperature: 37 C
Mode GPU temperature: 0 C, 35 C

990 MHz:
Average GPU temperature: 52.3 C
Max GPU temperature: 57 C
Mode GPU temperature: 0 C, 54 C

1095 MHz:
Average GPU temperature: 36.8 C
Max GPU temperature: 39 C
Mode GPU temperature: 0 C, 38 C

1185 MHz:
Average GPU temperature: 51.5 C
Max GPU temperature: 56 C
Mode GPU temperature: 0 C, 54 C

1290 MHz:
Average GPU temperature: 51.3 C
Max GPU temperature: 56 C
Mode GPU temperature: 0 C, 52 C

1410 MHz:
Average GPU temperature: 56.2 C
Max GPU temperature: 62 C
Mode GPU temperature: 0 C, 57 C

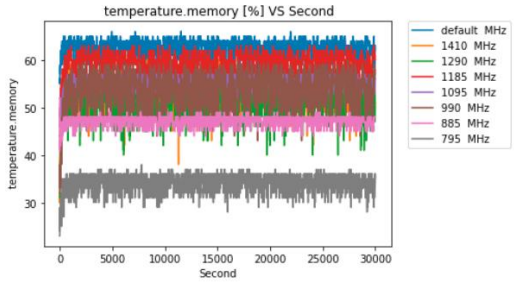

Fig 18. All clock profiles Memory temperature VS second plot.

795 MHz:
Average Memory temperature: 34.24 C
Max Memory temperature: 38 C
Mode Memory temperature: 0 C, 35 C

885 MHz:
Average Memory temperature: 47 C
Max Memory temperature: 52 C
Mode Memory temperature: 0 C, 47 C

990 MHz:
Average Memory temperature: 53.8 C
Max Memory temperature: 60 C
Mode Memory temperature: 0 C, 56 C

1095 MHz:
Average Memory temperature: 55.5 C
Max Memory temperature: 57 C
Mode Memory temperature: 0 C, 56 C

1185 MHz:
Average Memory temperature: 59.4 C
Max Memory temperature: 63 C
Mode Memory temperature: 0 C, 61 C

1290 MHz:
Average Memory temperature: 52.2 C
Max Memory temperature: 57 C
Mode Memory temperature: 0 C, 54 C

1410 MHz:
Average Memory temperature: 54.2 C
Max Memory temperature: 60 C
Mode Memory temperature: 0 C, 55 C

Figure 16 shows that each profile reads and writes memory differently based on the clock speed, contributing to distinct power consumption throughout the period. The next consideration is estimating the amount of power usage throughout benchmarking to decide which profile provides the most optimal balance between time taken to train the model, model accuracy after training, and the total power draw consumed.

Figure 19 represents that all profiles stay close to each other when the loss is from 0.66 to 0.4; after that, they gradually spread out, indicating an effect of different clock speeds. Looking at the loss 0.2, we see that the default, 1410 MHz, 1290 MHz, 1185 MHz, and 1095 MHz stay between 5037 and 5567 seconds while the rest below spread out, particularly 885 MHz and 795 MHz. When the loss reaches to 0.1, each profile starts to stay on significant seconds away from each other, with default profile reaching to 0.1 first, closely followed by 1410 MHz, while slower clock speeds such as 795 MHz, 885 MHz, and 990 MHz take more seconds to reach to the comparable point.

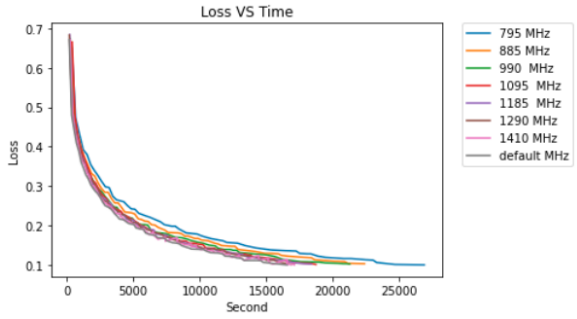

Fig 19. Loss value VS Time in second All Clock profiles at loss 0.1

	795 MHz	885 MHz	990 MHz	1095 MHz	1185 MHz	1290 MHz	1410 MHz	Default
Loss ≈ 0.2	7,332	6,622	6,083	5,567	5,318	5,506	5,014	5,171
Loss ≈ 0.1	26,885	22,894	21280	18731	18,391	18,017	16,276	16,562

Table 1. Training time in second at approximate loss 0.2 and 0.1

	795 MHz	885 MHz	990 MHz	1095 MHz	1185 MHz	1290 MHz	1410 MHz	Default
Loss ≈ 0.2	910591	895652	875686	852732	955256	1122528	1283723	1336256
Loss ≈ 0.1	3375313	3121479	3083790	2906176	3339079	3694927	4178652	4277915

Table 2. Cumulative Power draw in Watts at approximate loss 0.2 and 0.1

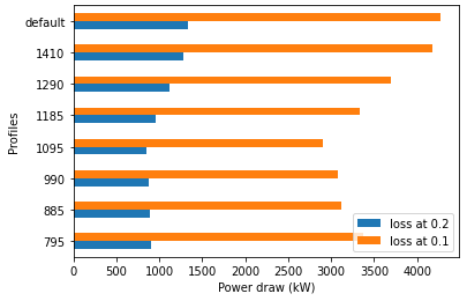

Fig 20. Cumulative Power draw VS profiles plot at 2 losses

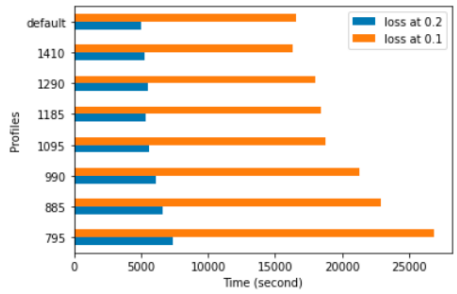

Fig 21. Cumulative training time VS profiles plot at 2 losses

D. Applying Optimal power efficient profile

The first important factor we prioritize when optimizing is training time. At the loss approximately 0.2 in the table 1, the training time range is from 5,171 seconds to 7,332 seconds where the median is 5,442 seconds. Table 1 shows that higher clock speeds yield less training time across all profiles, but the difference is not substantial compared to at the approximate loss 0.1. For a reference point at loss 0.1, training time gives more substantial difference among all profiles; the training time range is from 16,562 seconds to 26,885 seconds where the median is 18561 seconds. We discover that at loss 0.1, 1410 MHz and default profile hold similar training time between 16,276 and 16,562 seconds; 1290MHz, 1185 MHz, and 1095 MHz have training time proximity between 18,017 and 18,731 seconds; 990 MHz, and 885 MHz also have proximity between 21,280 and 22,894 seconds while only 795 MHz uses 26,885 seconds, the slowest training period of all profiles.

Second factor is total amount of power used to train for each profile. In table 2, for the default profile, the training time at approximate loss 0.1 takes 16,562 seconds (270 minutes) and consumes 4,277,915 Watts in total. One observation from figure 20 and 21 is that 1410 MHz and default profile yield similar training time and cumulative power draw, with only 2.4% difference and 1.8% in training time; it can be concluded that the default profile consumes the most power for this benchmark. Another observation is that lower clock speed profiles use more power than some of the higher clock speeds to reach to the same loss due to low frequency which increases training time. For instance, 795 MHz, 885 MHz , and 990 MHz take 3,375,313, 3,121,479 and 3,083,790 Watts loss 0.1, while 1095 MHz spends 2,906,176 Watts, using 13.8%, 6.89%, 5.75% less training time than 795 MHz, 885 MHz, and 990 MHz respectively.

From table 1 and table 2, we can conclude that at loss = 0.2 the default frequency has the highest cumulative power draw (1336 KW) and least training time while we see 1095 MHz profile that consumes the least power (852 KW) while only 7% slower than the default profile. At loss = 0.1, default frequency (1410) has the highest cumulative power draw (4277 KW) and least training time while we see 1095 MHz profile that consumes the least power (2906 KW) while only 11.5% slower than the 1410 profile.

We also use power draw per second ratio to help determine the most optimal profile based on table 1 and table 2. We divide the cumulative power draw at each loss reference and average them to have the final ratio. Because power draw is consumed at a steady clock speed, we derive the ratio as the following:

795 MHz	885 MHz	990 MHz	1095 MHz	1185 MHz	1290 MHz	1410 MHz	Default
124.85 W/S	135.75 W/S	144.5 W/S	154.16 W/S	180.55 W/S	204 W/S	256 W/S	258 W/S

Table 3. Estimated Power draw: Second ratio for each profile

Table 3 represents that 1095 MHz shows a promising candidate to be optimal because it is only 34 W/S different from 795 MHz with 300 MHz faster frequency while 26.39 W/S different from 1185 MHz with only 90 MHz slower and even 103.84 W/S different from the default with 315 MHz slower frequency. Also, from table 1 and table 2, 1095 MHz only spends 2169 seconds (36 minutes) more than the default profile, but consumes 1,371,739 Watts less than the default setting, yielding 32% power usage saving with only 11.5% more training time.

The last factor considered is model performance after training. We found that once the model converges at loss 0.1 the model yields 87.77% accuracy on average for every profile. Therefore, we have discovered that 1095 MHz profile passes our requirements and therefore is the most optimal balance between performance and training time. O

IV. Literature Review

What we have achieved is a significant power efficient and performance balance in the static setting for the HPC. An option to dynamically allocate resource or modify profiling based on various usages is also desired. Tanash et al., create a supervised machine learning model which is embedded into the Slurm resource manager to predict the memory usage and required amount of time to compute the workload based on historical HPC log files showing actual amounts used in the past [7]. Their model helped save the allocated time from five days to ten hours for big workloads. Apex Infrastructure also uses Slurm [8] as the jobs scheduling software for students, so there could be an opportunity to use similar techniques onto it unless the Apex does not contain historical logs from requested jobs or actual usage. Another approach by Qiqi et al., use RLSchert, a job scheduler implemented on deep reinforcement learning, that estimates the state of the system applying a dynamic job remaining runtime predictor, providing an accurate spatiotemporal view of the cluster status, as well as learning the optimal policy to wither select or kill jobs based on the status by imitation learning and the proximal policy [9]. One advantage for this approach is that no labels are required for the model to learn which provides flexibility and enables active learning. However, a primary challenge is designing an objective function for the model to learn.

V. Conclusion and Future Work

This research presents a static optimization for the Apex HPC infrastructure that helps reduce 32% of the power consumption on common workloads on the Apex while compromising 11.5% more training time when compared to the default setting. We will apply this methodology on the 1st node from 6 nodes as an initial trial. After successfully deploying static optimization to Apex nodes, the next step is to investigate a method to derive an adaptive tuning algorithm. We will investigate on using learning technologies to dynamically adjust compute characteristics or resource allocation based on user's usage. In case we may require labels or dataset for supervised learning, we could enable job submission from the users to obtain historical logs or benchmark over with more complex and longer training time workload to obtain more spectrum of workload characteristics. Then, we may scale up our methodologies to different use cases such as multiple-GPU or multi-node usage that are used from time to time.

VI. Acknowledgement

This research has been supported by Program Management Unit for National Competitiveness Enhancement (PMU-C); Office of National Higher Education Science Research and Innovation Council for system experimentation.

VII. References

[1] "ThaiBev: Automation of Bottle Screening for Recycling", https://www.cmkl.ac.th/project/thaibev-automation-of-bottle-screening-for-recycling (accessed Dec, 2021).

[2] "The Next Generation of Cloud HPC+AI Infrastructure ", https://www.cmkl.ac.th/research/apex (accessed Sep, 2021).

[3] Matej Špeťko, Ondřej Vysocký, Branislav Jansík, Lubomír Říha, "DGX-A100 Face to Face DGX-2—Performance, Power and Thermal Behavior Evaluation", 2021.

[4] NVIDIA Corp., NVIDIA DGX A100 The universal system for AI infrastructure, 2020. https://images.nvidia.com/aem-dam/Solutions/Data-Center/nvidia-dgx-a100-infographic.pdf

[5] NVIDIA Corp. Nvidia-Smi—NVIDIA System Management Interface Program. 2016. Available online: http://developer.download.nvidia.com/compute/DCGM/docs/nvidia-smi-367.38.pdf (accessed on 4 September 2021).

[6] Boston University, “Correlation Analysis”, 2021. https://sphweb.bumc.bu.edu/otlt/mph-modules/bs/bs704_correlation-regression/bs704_correlation-regression2.html

[7] NVIDIA Corp., Tensor Cores. https://www.nvidia.com/en-us/data-center/tensor-cores/

[8] Mohammed Tanash, Brandon Dunn, Daniel Andresen, William Hsu, Huichen Yang, Adedolapo Okanlawon, “Improving HPC System Performance by Predicting Job Resources via Supervised Machine Learning”, 2019. https://dl.acm.org/doi/pdf/10.1145/3332186.3333041?fbclid=IwAR1zgXKO_cS4P9GUm5ob__8vbZD8einMbnuj1CvnmARqIM7OmwoUPLEpUiI.

[9] Slurm Workload Manager, 2022. https://slurm.schedmd.com/documentation.html

[10] Qiqi Wang, Hongjie Zhang, Cheng Qu, Yu Shen, Xiaohui Liu, Jing Li , “RLSchert: An HPC Job Scheduler Using Deep Reinforcement Learning and Remaining Time Prediction”, 2021. https://www.mdpi.com/1996-1073/14/2/376/html