SAND2022-12268C

Multi-Component DVFS

Dipayan Mukherjee

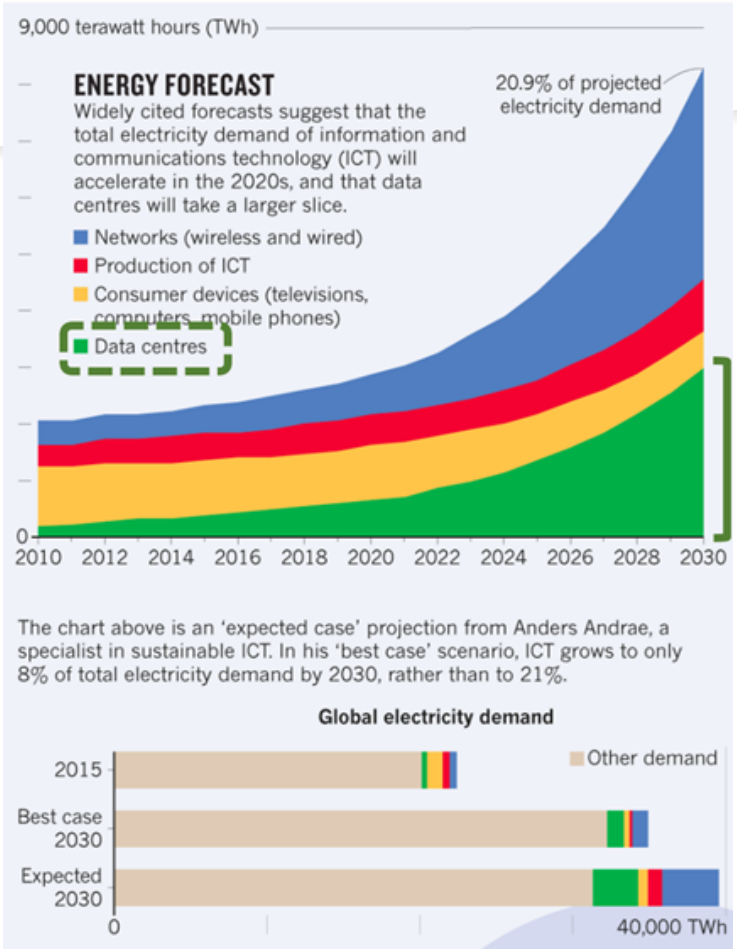
UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Sandia National Laboratories

Motivation

- The energy footprint of Information and Computation Technology is growing rapidly:
  - One estimate of Greenhouse Gasses (GHG) generated in 2020 by ICT puts it between 1.8% and 3.9% of global GHG emissions
- Need for energy-aware computing:
  - **High-Performance Computing**: Exascale computing through parallelism
  - **Mobile Devices (IoT, cell phones, satellites)**: Meeting the increasing demand for performance with a longer battery life



Ref : https://www.nature.com/articles/d41586-018-06610-y
Data Source : Andrae ASG, Edler T. On Global Electricity Usage of Communication Technology: Trends to 2030. *Challenges*. 2015; 6(1):117-157. https://doi.org/10.3390/challe6010117

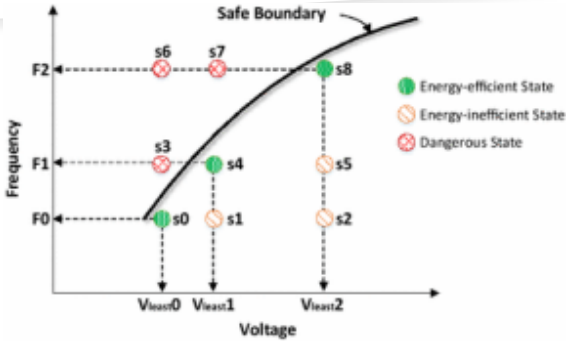
How to control energy consumption

Dynamic Voltage and Frequency Scheduling (DVFS) allows for lower power consumption by reducing the frequency of operation:

- Power and energy model:

$$P = C_L V^2 f + I_L V$$

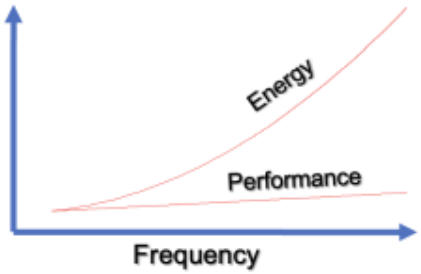
$$E_{total} = E_{dynamic} + E_{leakage}$$



where P is power consumed, C_L *and* I_L are constants, V is voltage, and f is the operating frequency

- Memory power correlates with access count and operating frequency

$$P_{MEM} = \frac{1}{2} C_L V^2 f \times access\_count + I_L V$$



- Lowers operating temperature of the system: longer life
- Provides a software method for controlling the system usage without massive hardware changes

Background

- **When to run DVFS**:
  - Compile Time : System makes the decision for allocation of resources and operating point based on estimated operating characteristics.(2)
  - Runtime Systems: Current state of the art considers all the state transition based on a single resource, like CPU or memory, but ignore any interaction among resources.(3) Runtime-DVFS for heterogeneous systems is a space to be explored.
- **Deciding operating point:**
  - Currently all operating point transitions are done instantaneously (race to idle), which can lead to loss of energy.
  - State transition latency can impact the performance of systems and can yield better results in place of aggressive DVFS.(4)

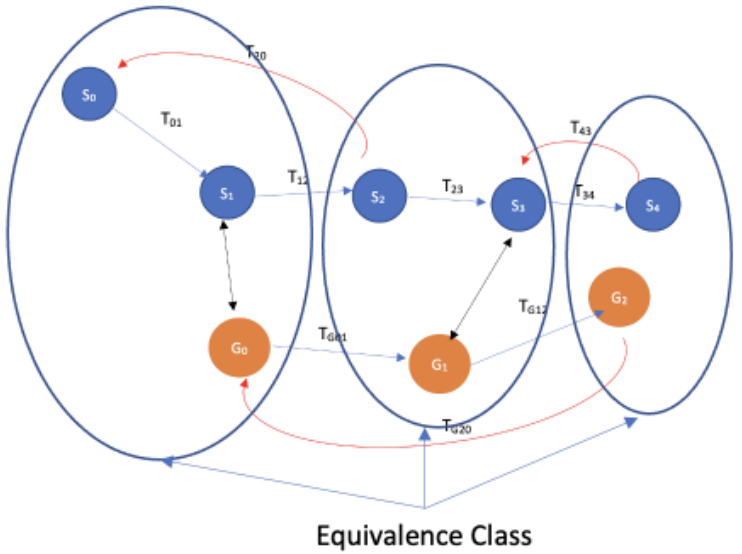
(1) B. Acun et. al . Fine-Grained Energy Efficiency Using Per-Core DVFS with an Adaptive Runtime System . 2019 IGSC . doi : 10.1109/IGSC48788.2019.8957174
(2) Sheng Yang *et al*., "Adaptive energy minimization of embedded heterogeneous systems using regression-based learning," *2015 (PATMOS)*, doi: 10.1109/PATMOS.2015.7347594
(3) Q. Fettes, *et al* ."Dynamic Voltage and Frequency Scaling in NoCs with Supervised and Reinforcement Learning Techniques," in *IEEE Transactions on Computers*, 1 March 2019, doi: 10.1109/TC.2018.2875476
(4) D. H. K. Kim et al, "Racing and Pacing to Idle: Theoretical and Empirical Analysis of Energy Optimization Heuristics," *2015 ICCPS*, doi: 10.1109/CPSNA.2015.23

Background

- GPU DVFS:
  - CPU DVFS schemes split up applications into serialized compute and memory phases: GPUs break this assumption
  - **CRISP:** Split GPU applications into the **Load-Critical Path** and **Compute/Store Path**
- DRAM DVFS:
  - **MemScale:** Proposes DVFS for memory controllers and DFS for DRAM chips
  - **Voltron:** Proposes voltage scaling for DRAM chips while maintaining correctness
- Full System DVFS:
  - Having ignorant DVFS schemes for separate components can lead to performance pathologies.
  - **CoScale:** Combined Memory and CPU DVFS scheme
  - Multiple components multiplicatively grows search space

Problem Statement

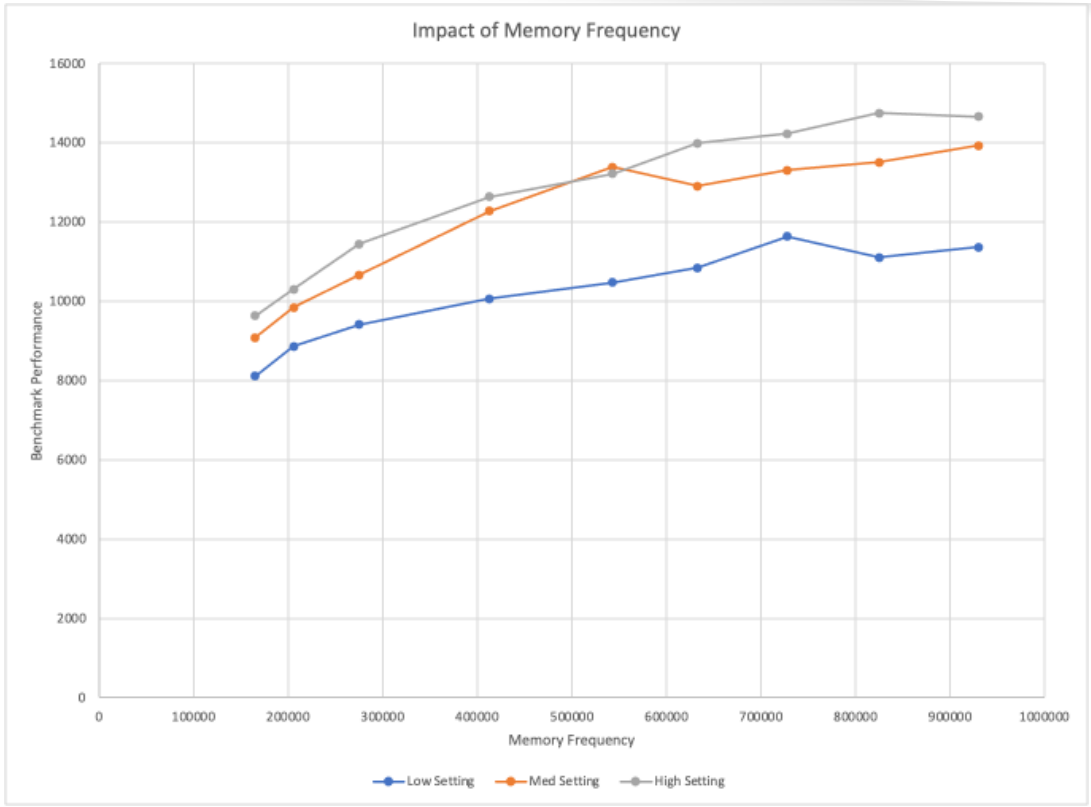
- How do different resources impact the power-performance of the system in a multi-component system?
- Are there substantial energy-performance inefficiency zones with respect to voltage and frequency for different components?
- Can varying demand across multiple resources be served to achieve optimal energy efficiency?

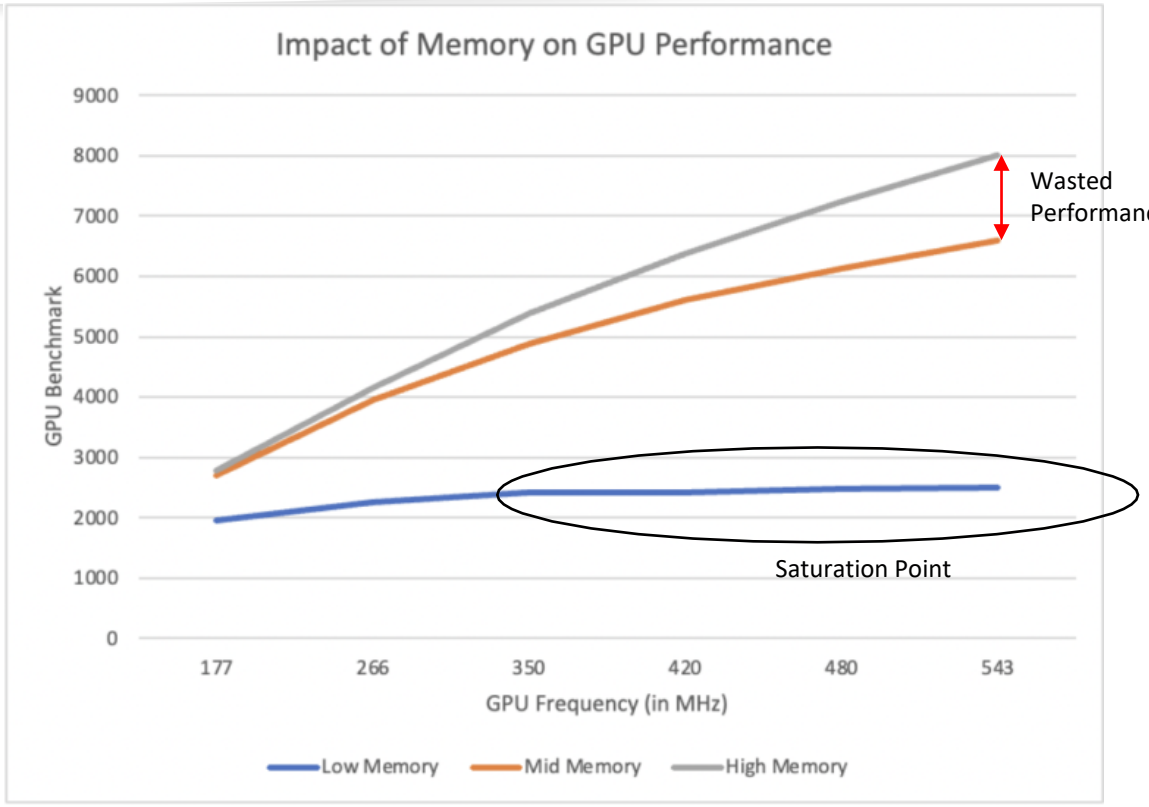


Methodology

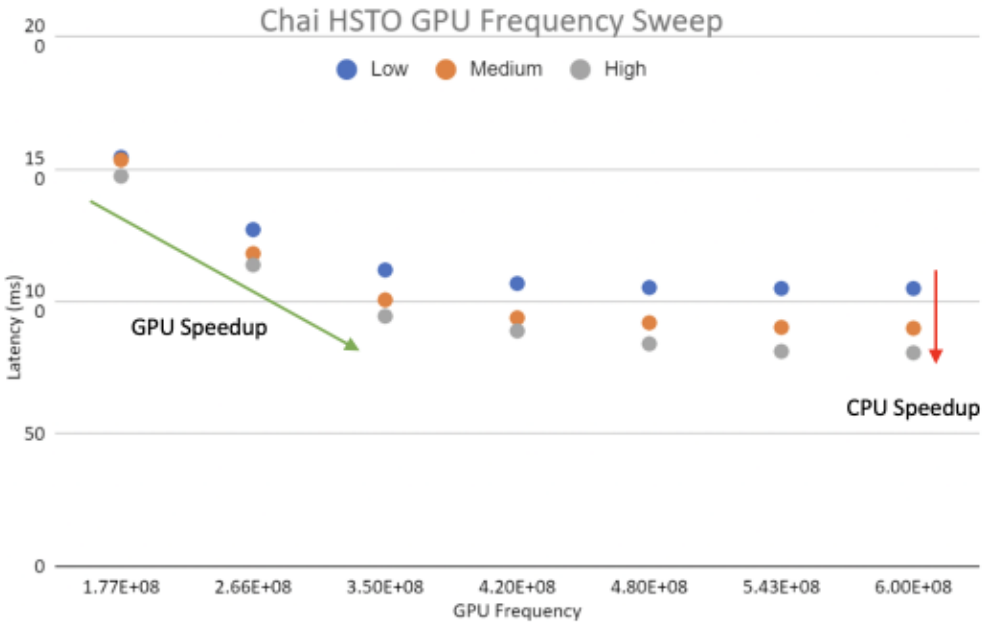
- Analyze the energy and performance of each component across various benchmarks
- Quantify the frequency dependence among various components
- Identify the clusters of operating points based on energy-performance requirement

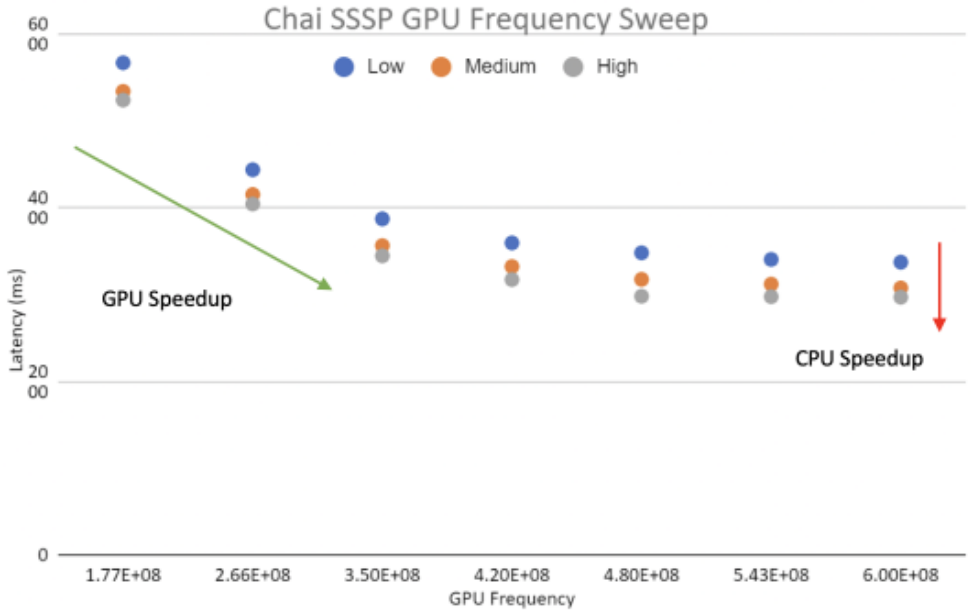
Dependency among resources



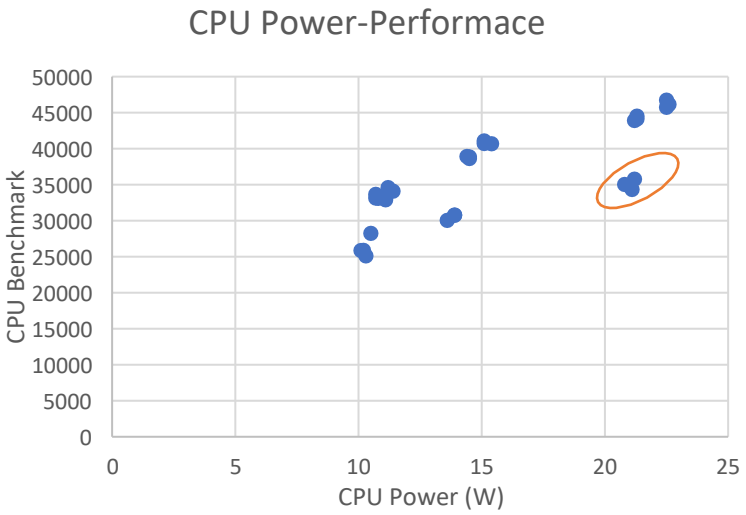


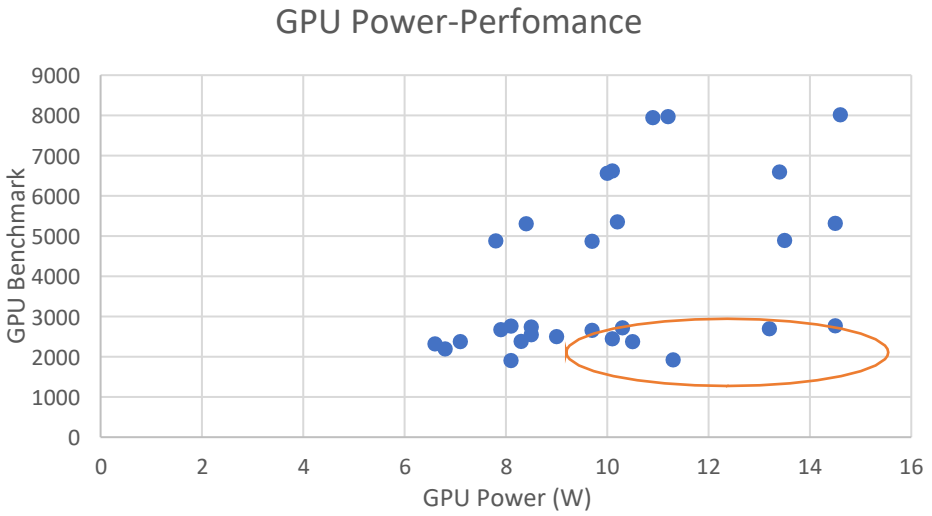
Dependency among resources

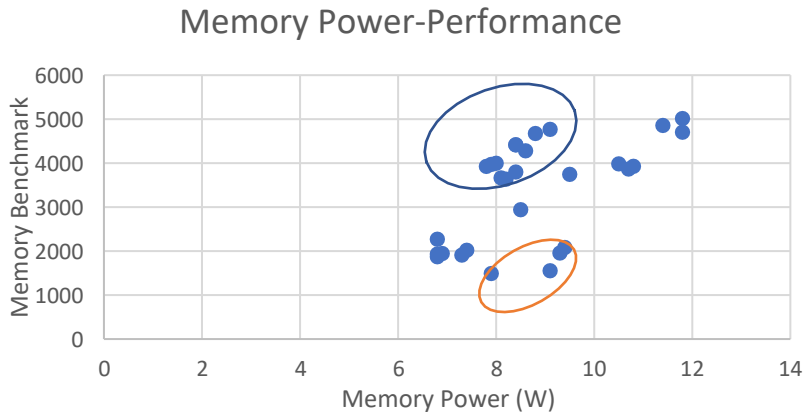




Dependency among resources



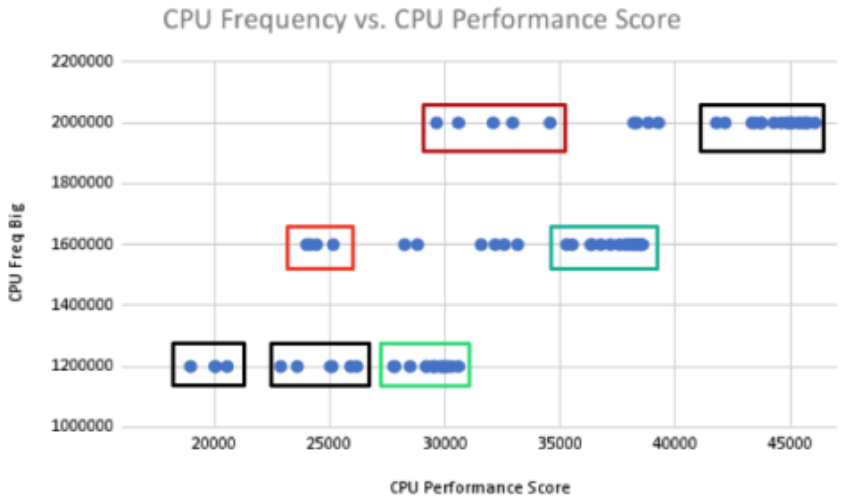


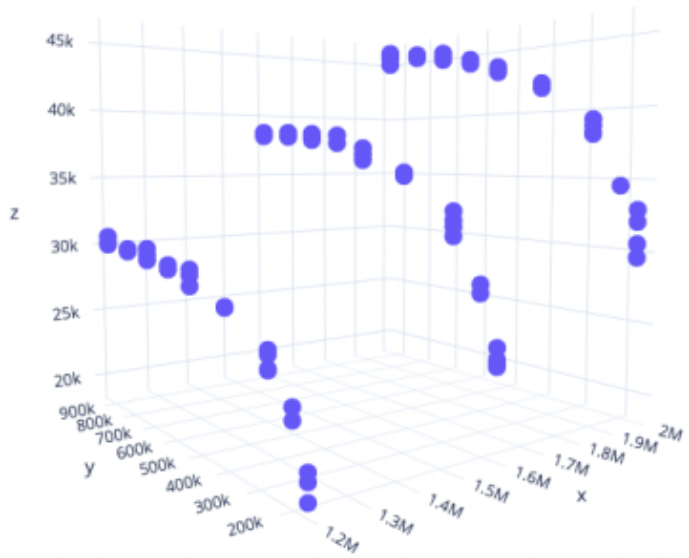


UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

Sandia National Laboratories

Dependency among resources





Placeholder for the video link

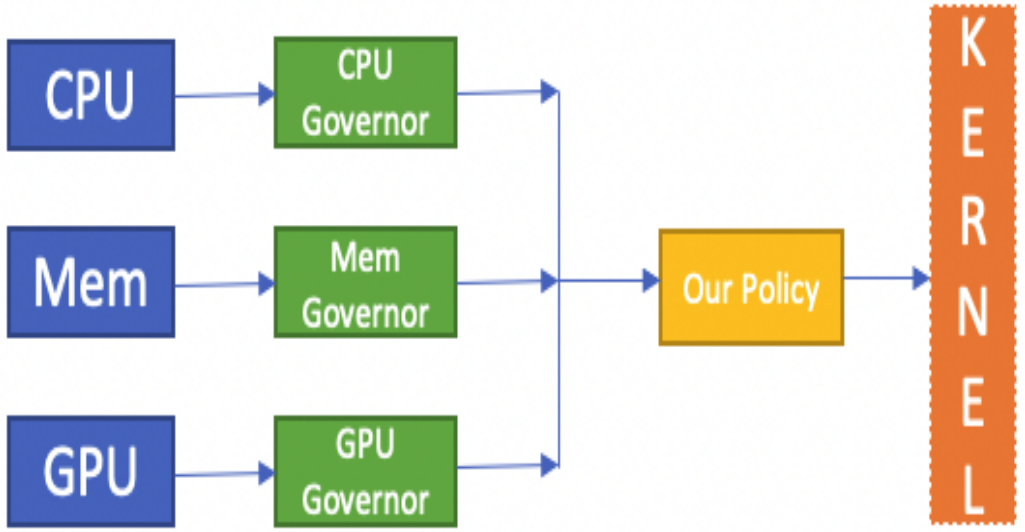
Dependency among resources

	MemPerf	CPUPerf	GPUPerf
MemFreq	0.921	0.632	0.776
CPUFreq	0.152	0.737	0.070
GPUFreq	-0.130	0.041	0.538

	GPU	CPU	Mem
Total	-0.36	-0.58	-0.2
KMeans	-0.70	-0.65	-0.055
NW	-0.33	-0.83	-0.37
Pathfinder	-0.83	-0.45	-0.15
StreamCluster	-0.18	-0.80	-0.53
HotSpot3D	-0.23	-0.91	-0.17
BackProp	-0.26	-0.85	-0.41

Approach

- We tried multiple clusters based on various methods and parameters:
  - Power-based: We clustered operating points based on power consumption
  - Performance: Operating points were clustered based on the performance score
  - Power-perf: We used multi-dimensional clusters, such as DBSCAN.
- EDP(ν) : We use energy-delay product with ν determining the balance of power and performance : power $* perf^{\nu}$
- Designed state transition policy based on the system-wide resource utilization and efficiency requirement ν
- Implement our policy using scripts which allows all resources to work in coherence



Results: Operating point clusters

[177, 1.4, 165]	0
[350, 1.4, 165]	0
[543, 1.4, 165]	0
[177, 1.7, 165]	1
[350, 1.7, 165]	1
[543, 1.7, 165]	1
[177, 1.4, 543]	2
[177, 1.4, 933]	2
[350, 1.4, 543]	2
[350, 1.4, 933]	2
[543, 1.4, 543]	2
[543, 1.4, 933]	2
[177, 2.0, 165]	2
[350, 2.0, 165]	2
[543, 2.0, 165]	2
[177, 1.7, 543]	3
[177, 1.7, 933]	3
[350, 1.7, 543]	3
[350, 1.7, 933]	3
[543, 1.7, 543]	3
[543, 1.7, 933]	3
[177, 2.0, 543]	4
[177, 2.0, 933]	4
[350, 2.0, 543]	4
[350, 2.0, 933]	4
[543, 2.0, 543]	4
[543, 2.0, 933]	4

Performance based cluster

[177, 1.4, 165]	0
[350, 1.4, 165]	0
[543, 1.4, 165]	0
[177, 1.4, 543]	0
[177, 1.4, 933]	0
[350, 1.4, 543]	0
[350, 1.4, 933]	0
[543, 1.4, 543]	0
[543, 1.4, 933]	0
[177, 1.7, 165]	1
[350, 1.7, 165]	1
[177, 1.7, 543]	1
[543, 1.7, 543]	1
[177, 1.7, 933]	2
[543, 1.7, 933]	2
[177, 2.0, 165]	3
[350, 2.0, 165]	3
[543, 2.0, 165]	3
[177, 2.0, 543]	3
[177, 2.0, 933]	4
[543, 2.0, 933]	4

Power based cluster

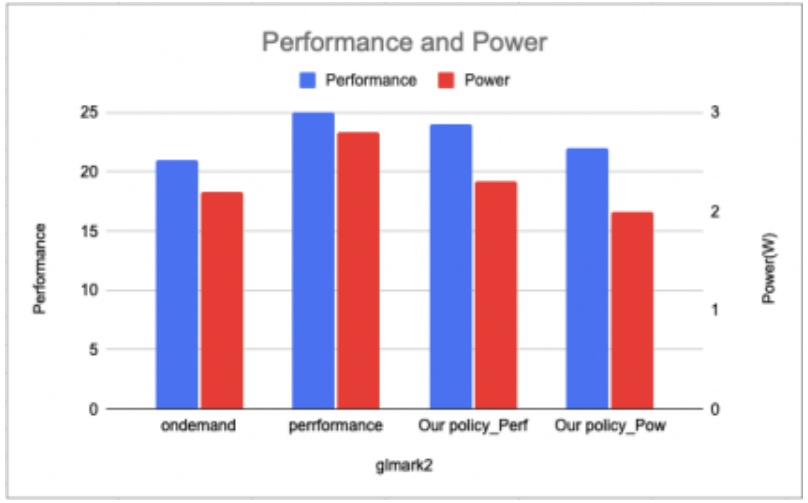
Results: Operating point clusters

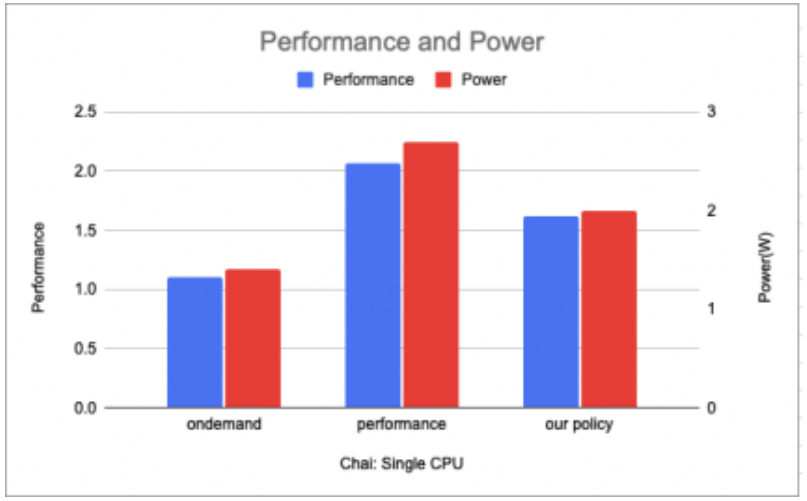
MemFreq	CPUBig	CPUSmall	GPUFreq	All_ EDP(2)	cluster
933	2	1.4	350	2.986725	2
933	2	1.4	543	2.924607	2
933	2	1.4	177	2.860782	2
543	2	1.4	177	2.564976	2
543	2	1.4	350	2.520117	2
543	2	1.4	543	2.484012	2
165	2	1.4	177	1.648041	0
165	2	1.4	543	1.551258	0
933	1.7	1.2	543	1.548798	0
933	1.7	1.2	177	1.548111	0
933	1.7	1.2	350	1.521018	0
165	2	1.4	350	1.511463	0
543	1.7	1.2	350	1.330173	0
543	1.7	1.2	543	1.325286	0
543	1.7	1.2	177	1.315152	0
933	1.4	1	177	0.814584	1
933	1.4	1	543	0.805383	1
933	1.4	1	350	0.800757	1
165	1.7	1.2	350	0.800127	1
165	1.7	1.2	543	0.799554	1
165	1.7	1.2	177	0.745962	1
633	1.4	1	177	0.736635	1
543	1.4	1	543	0.73005	1
543	1.4	1	350	0.719199	1
543	1.4	1	177	0.714264	1
206	1.4	1	177	0.508911	3
165	1.4	1	350	0.415197	3
165	1.4	1	177	0.410364	3
165	1.4	1	543	0.394689	3

Cluster	CPUBig	CPUSmall	Mem	GPU	All_EDP(1)
2	2	1.4	933	543	2.85466406
2	2	1.4	933	350	2.43247119
0	2	1.4	543	543	1.98811663
0	2	1.4	933	177	1.97867863
0	2	1.4	543	350	1.74607915
0	1.7	1.2	933	543	1.72374422
0	1.4	1	933	543	1.53291816
0	2	1.4	543	177	1.51879952
0	1.7	1.2	933	350	1.51600265
1	1.7	1.2	933	177	1.24527708
1	1.7	1.2	543	543	1.16138224
1	1.4	1	543	543	1.11076558
1	1.7	1.2	543	350	1.09394551
1	1.4	1	933	350	1.07125339
1	1.7	1.2	543	177	0.93127056
1	1.4	1	933	177	0.8917054
1	1.4	1	543	350	0.80321873
1	1.4	1	633	177	0.74225907
1	2	1.4	165	543	0.71886749
1	2	1.4	165	350	0.68690973
1	2	1.4	165	177	0.66783666
1	1.4	1	543	177	0.66482986
3	1.7	1.2	165	350	0.40655554
3	1.7	1.2	165	543	0.38561406
3	1.7	1.2	165	177	0.38167938
3	1.4	1	206	177	0.32265259
3	1.4	1	165	543	0.27591557
3	1.4	1	165	177	0.26262173
3	1.4	1	165	350	0.26137725

Cluster	CPUBig	CPUSmall	Mem	GPU	All_EDP(0.5)
1	2	1.4	933	543	2.96214768
1	2	1.4	933	350	2.80440265
1	2	1.4	543	543	2.53789254
1	2	1.4	933	177	2.5195131
1	2	1.4	543	350	2.44816372
3	2	1.4	543	177	2.23724676
3	1.7	1.2	933	543	2.09644149
3	1.7	1.2	933	350	1.94657308
3	1.4	1	933	543	1.84746116
0	1.7	1.2	543	543	1.76175922
0	1.7	1.2	933	177	1.75729127
0	2	1.4	165	543	1.69230069
0	1.7	1.2	543	350	1.68778228
0	2	1.4	165	350	1.68394641
0	1.4	1	543	543	1.65132531
0	2	1.4	165	177	1.62880446
0	1.4	1	933	350	1.56357551
0	1.7	1.2	543	177	1.56158096
0	1.4	1	933	177	1.43477354
0	1.4	1	543	350	1.41081572
2	1.4	1	633	177	1.33277859
2	1.4	1	543	177	1.29797278
2	1.7	1.2	165	543	1.20825011
2	1.7	1.2	165	350	1.19067886
2	1.7	1.2	165	177	1.15089445
2	1.4	1	165	543	1.01997308
2	1.4	1	206	177	0.99258021
2	1.4	1	165	350	0.95265348
2	1.4	1	165	177	0.93976235

Result : Impact of DVFS Policy





UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

Sandia National Laboratories

Conclusion

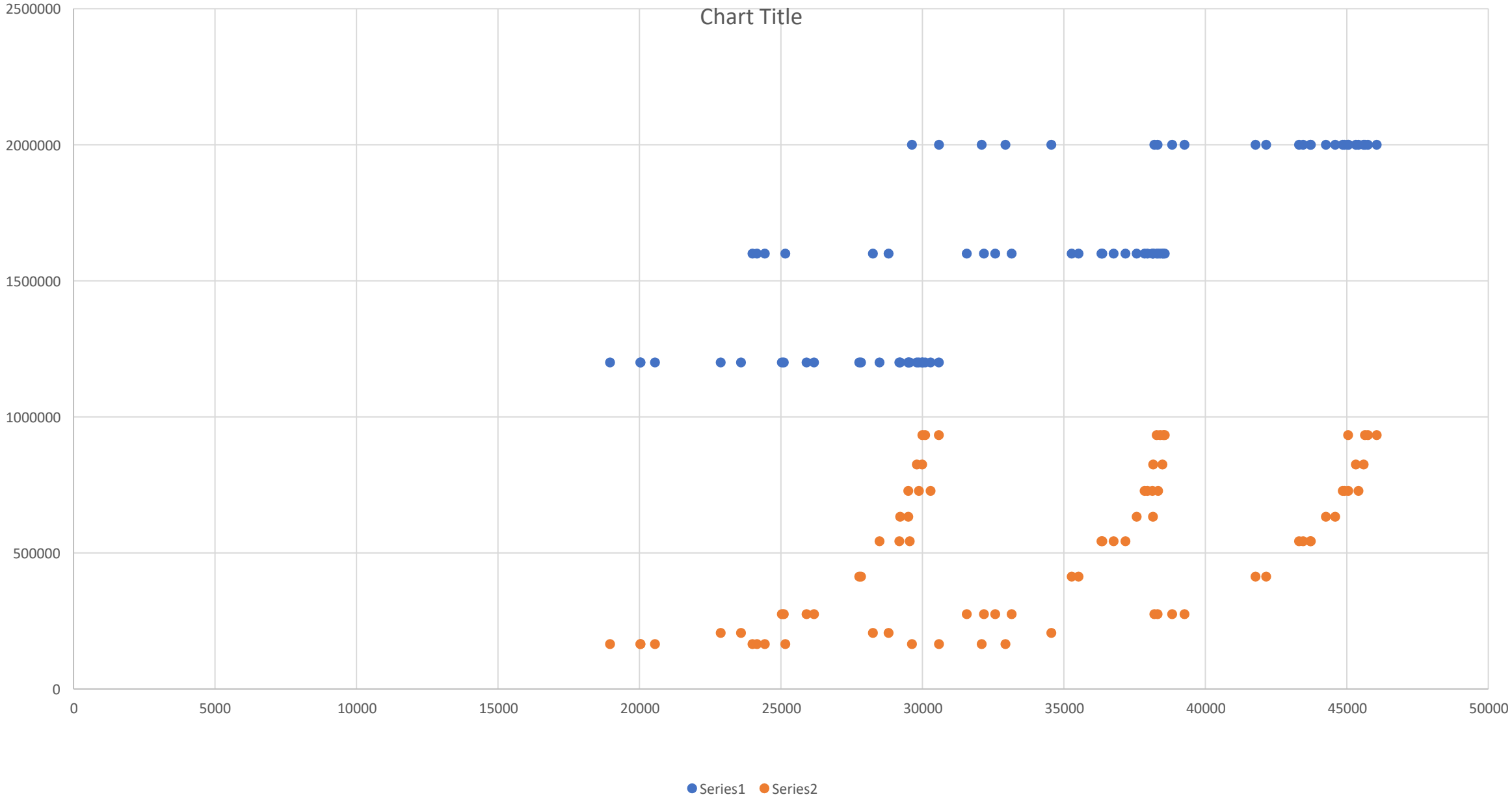
- Memory has a significant impact on the performance of other resources in a heterogeneous SoC
- Finer grained control over DVFS improves energy efficiency of the system
- Saving **45%** energy while incurring minimal performance degradation

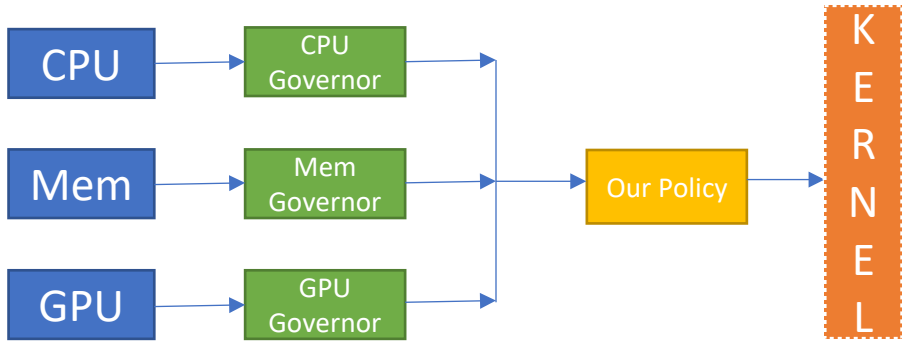
Future Work

- Analyze the latency transition characteristics of GPU and Memory
- **Finer-grained DVFS policy.**
- Dynamic Performance Demand Prediction
- **Temporal analysis of application resource requirements for efficient scheduling**

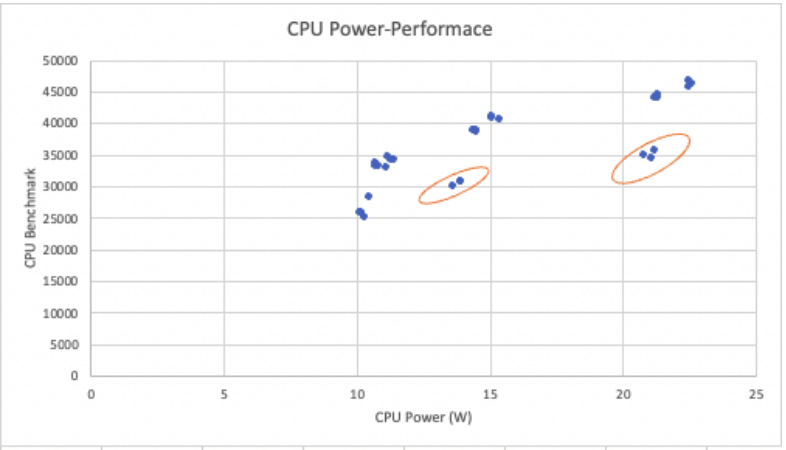
Work Slides

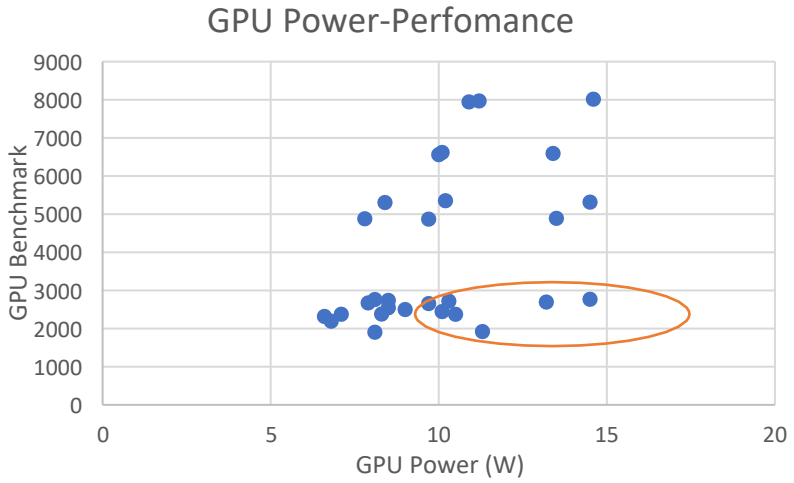
3D representation of data

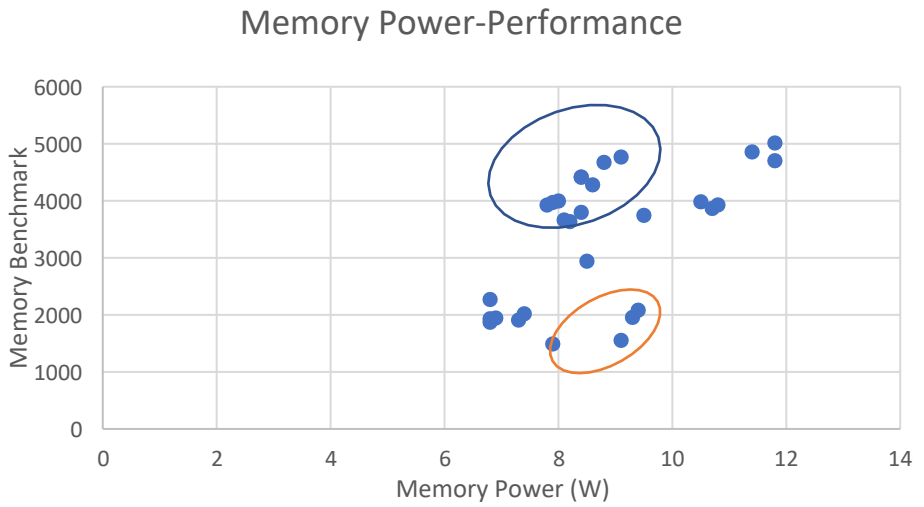



CPU
Mem
GPU
CPU Governor
Mem Governor
GPU Governor
Our Policy
KERNEL

Results


CPU Power-Performace
CPU Benchmark
CPU Power (W)
50000
45000
40000
35000
30000
25000
20000
15000
10000
5000
0
0
5
10
15
20
25


GPU Power-Perfomance
GPU Benchmark
GPU Power (W)
9000
8000
7000
6000
5000
4000
3000
2000
1000
0
0
5
10
15
20


Memory Power-Performance
Memory Benchmark
Memory Power (W)
6000
5000
4000
3000
2000
1000
0
0
2
4
6
8
10
12
14