5G EDGE CLOUD APPLICATIONS

By: Rahul Rajkumar and Andrew Michael

THE TEAM

Rahul Rajkumar

Andrew Michael

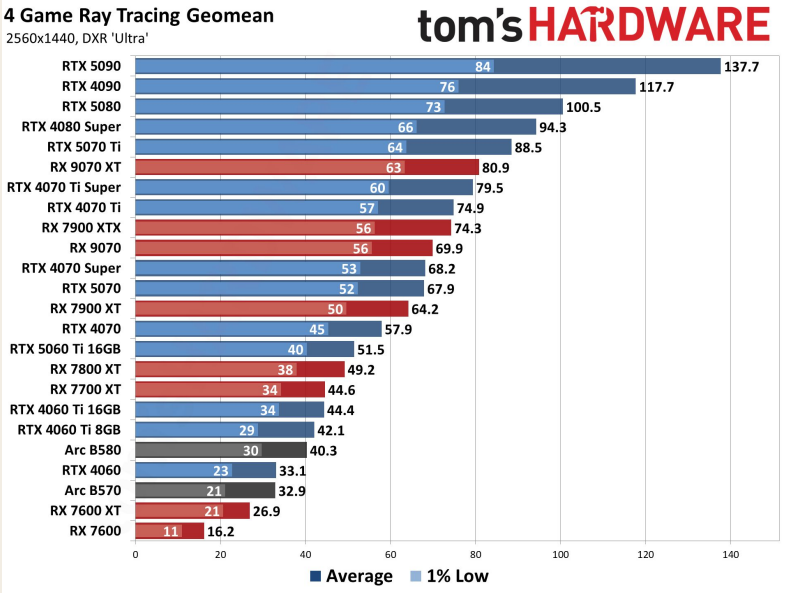
Special thanks to our advisors: Ivan Seskar & Xiaotian (Jack) Zhou

Edge Computing

- Distributed computing model that brings compute closer to the sources of data.
- Lower latency, higher bandwidth, and addresses privacy and security concerns with cloud computing
- Needs improvement in running AI applications

Project Overview

- Improve scheduling issue by creating benchmarks
- Focusing on AI applications, specifically running LLMs on Nvidia GPUs
- Prioritizing latency of LLM response



CONDUCTING THE EXPERIMENT

- Send prompts to LLM (llama 3.1:8b) being run on Nvidia GPUs (Tesla V100 & A100) via a Python script 
- Collect GPU metrics & LLM performance metrics
- Store and understand the collected data

TOOLS USED

EXPORTERS

Exports GPU live metric information to port

PROMETHEUS

Listens to the exporters and collects that data

GRAFANA

Takes GPU metrics & displays it on dashboard

OLLAMA

Receives prompts and sends them to local LLM

mongoDB

MONGODB

Collects the metrics sent from Prometheus & stored in database

SET UP VISUALS

```python
def fetch_intial_metrics():
    intial_doc = {}
    print("Initializing Stationary Metrics...\n")
    for metric_name, metric_query in INITIAL_METRICS.items():
        if metric_name == 'GPU Info':
            response = (requests.get(url=PROMETHEUS_URL, params={"query":metric_query})).json()
            response = response["data"]["result"][0]['metric']['name']
            intial_doc[metric_name] = response
        else:
            response = (requests.get(url=PROMETHEUS_URL, params={"query":metric_query})).json()
            response = float(response["data"]["result"][0]["value"][1])
            intial_doc[metric_name] = response

    intial_doc["Start Time"] = INITIAL_TIME_STRING
    intial_doc["Category"] = "Initial"

    mongo_col.insert_one(intial_doc)
    print('Finished Collecting Initial Metrics...\n')
    return intial_doc

async def fetch_verbose(prompt):
    print(f"Sending a prompt to Ollama...\n")
    def block_call():
        return ollama_client.chat(model='llama3.1:8b', messages=[
            {
                'role': 'user',
                'content': prompt,
            },
```

ollama

Benchmark Testing > dataset > ollama

Documents 2.4K | Aggregations | Schema | Indexes 1

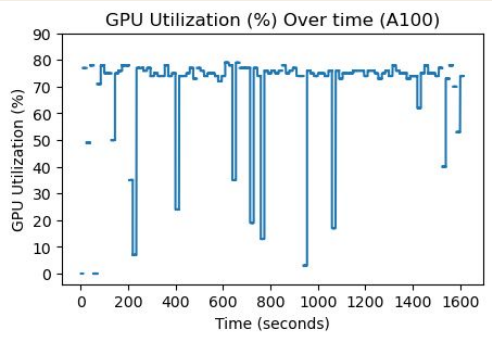
Type a query: { field: 'value' } or Generate query

ADD DATA | EXPORT DATA | UPDATE | DELETE

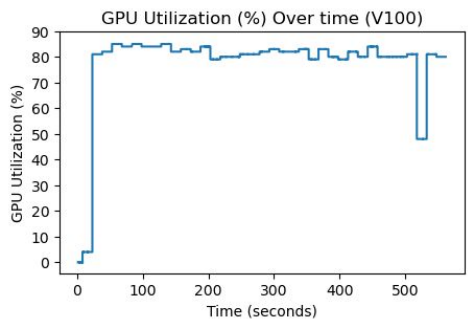
```
_id: ObjectId('68755bb2c9327eb904ad10ef')
Power Limit (Watts) : 250
Memory Clock Limit (MHz) : 877000000
Total Memory Allocation (MB) : 17179869184
GPU Clock Limit (MHz) : 1380000000
GPU Info : "Tesla V100-PCIE-16GB"
Start Time : "15:34:09"
Category : "Initial"
Collect : "True"

_id: ObjectId('68755bb2c9327eb904ad10f0')
GPU Utilization (%) : 0
Power Draw (Watts) : 23.74
GPU Temp (°C) : 22
GPU Current Clock (MHz) : 135000000
Memory Current Clock (MHz) : 877000000
Memory Allocation Used (MB) : 9437184
Memory Utilization (%) : 0
Current Time : "15:34:10"
Time Delta : 1
Category : "Prompt 0"
Iteration : 1
Collect : "True"

_id: ObjectId('68755bb2c9327eb904ad10f1')
```

DATA VISUALIZED

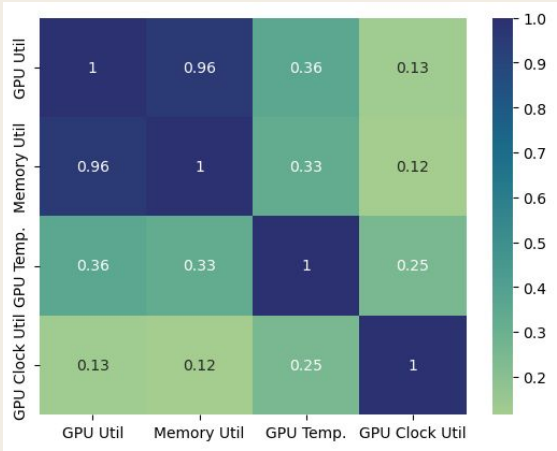
A100



V100



GPU Util



Heatmap

*Test run on V100 only consisted of 70 request vs A100 was 700 requests.

CREATING BENCHMARKS

- Created a relative weighted-sum score (compare between 2 different GPUs)
- Use the three most important metrics which are, GPU utilization, memory utilization, and response time
- Created 3 weights based on these metrics and multiply them against their average
- Our tests result in the scores:

Tesla A100: 0.103

Tesla V100: -0.233

CONCLUSIONS

We concluded a scoring method to compare performance of GPUs in running LLMs

Restricted to only this application & Nvidia GPUs

However, there are many other factors to consider...

FUTURE WORK

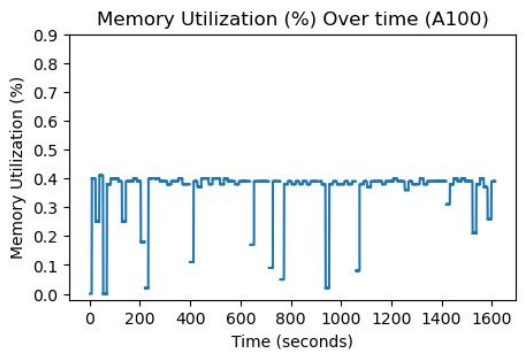
- Expanding other GPUs, networking, other hardware
- GPU is not always idle, impacts to benchmark score if other processes are running
- Creating a more rigorous scoring method

Thank you!

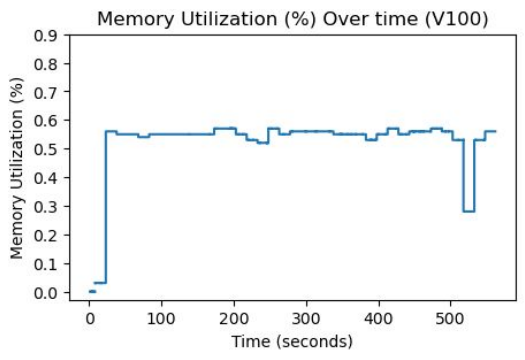
By: Rahul Rajkumar and Andrew Michael

DATA VISUALIZED

A100



V100



Memory Util

- Memory utilization for A100 and V100 over time
- Memory utilization for V100 remains stable but A100 fluctuates

WEIGHTS

$$w_1 = 1/\sigma_{\text{GPU Util}}$$

$$w_2 = 1/\sigma_{\text{Mem Util}}$$

$$w_3 = -e^{\text{rt}/(25/\ln 2)}$$

- Reciprocal of the standard deviation of the gpu utilization over the entire test
- Reciprocal of the standard deviation of the memory utilization over the entire test
- Negative of the exponential of the average response time divided by 25 over the natural log of 2