

Emulator Metrics

PERFORMANCE COMPARISON

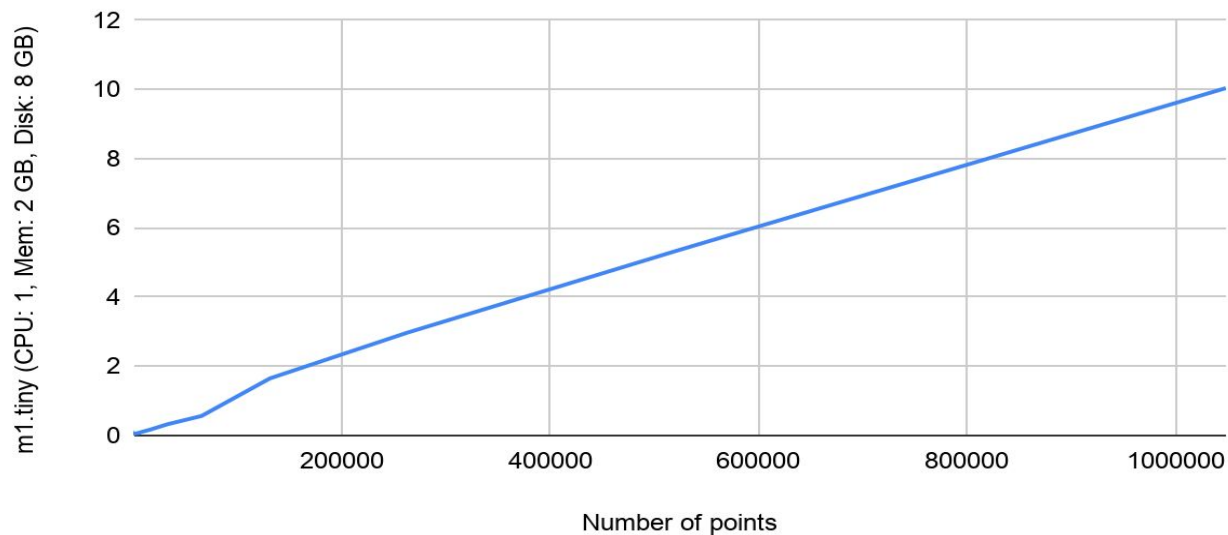
Resources

1. Resource 1: Edge
 - a. m1.tiny (CPU: 1, Mem: 2 GB, Disk: 8 GB)
2. Resource 2: Fog
 - a. m1.quad (CPU: 4, Mem: 10 GB, Disk: 20 GB)
3. Resource 3: Cloud
 - a. m1.xlarge (CPU: 44, Mem: 120 GB, Disk: 60 GB)

Running K_means: Clustering 4,8,16.....1048576 point into 3 clusters

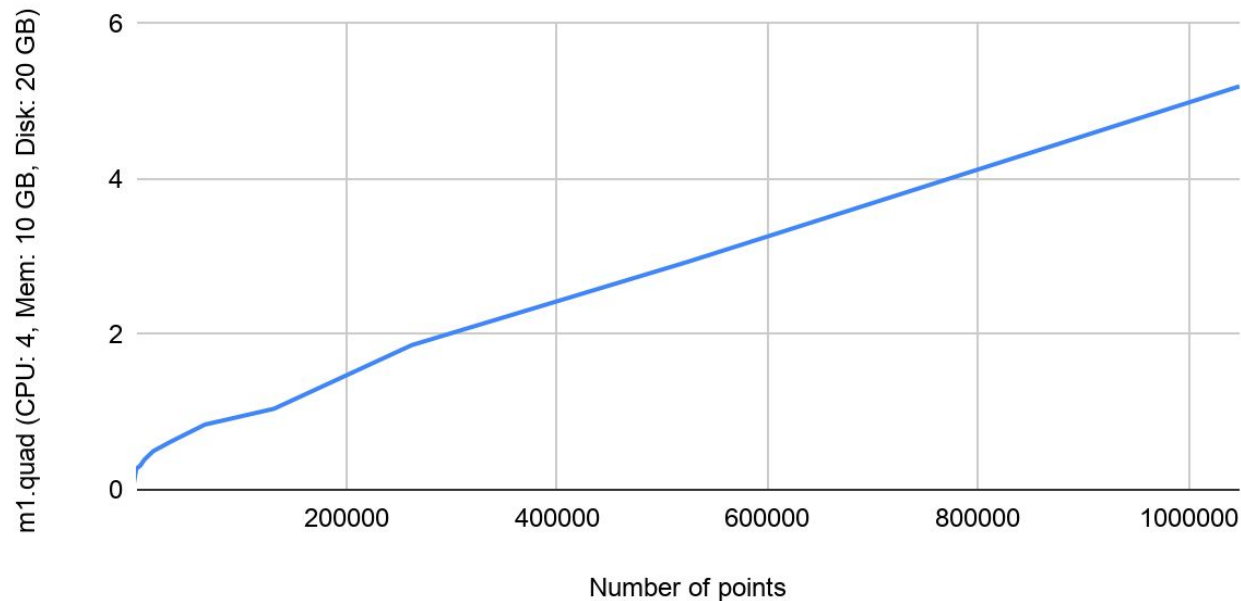
Individual performance: Edge resources

m1.tiny (CPU: 1, Mem: 2 GB, Disk: 8 GB)(execution time in seconds) vs. Number of points



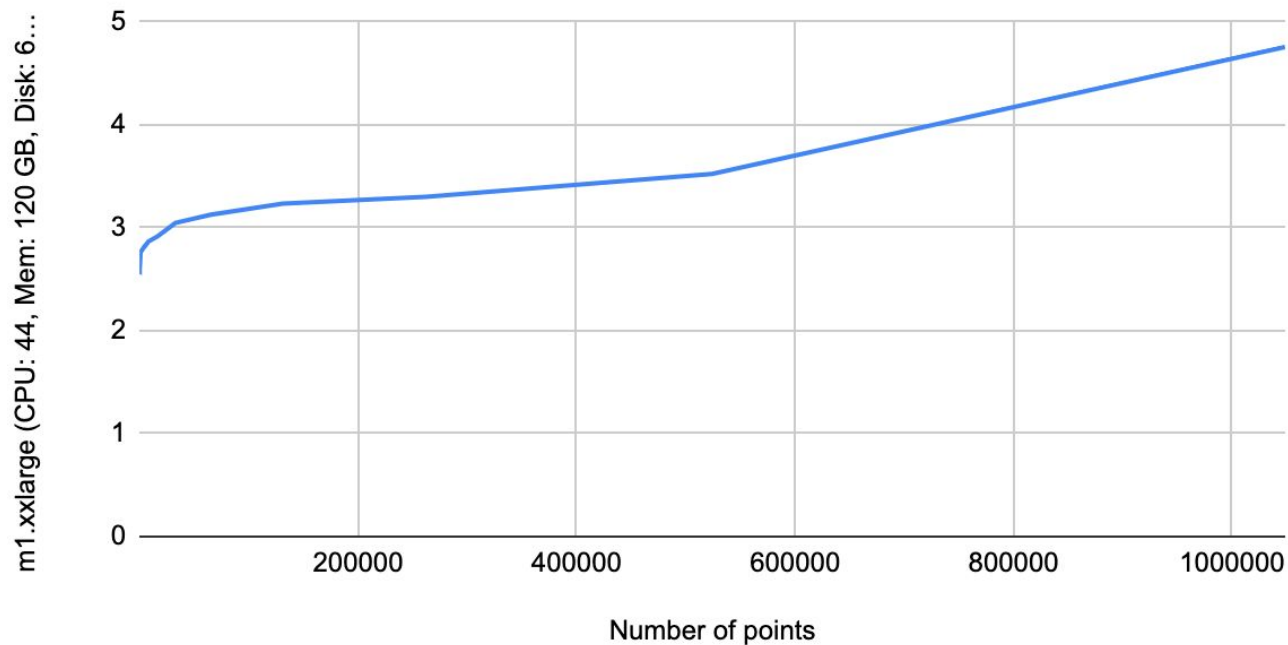
Individual performance: Fog resources

m1.quad (CPU: 4, Mem: 10 GB, Disk: 20 GB)(execution time in seconds) vs. Number of points



Individual performance: cloud resources

m1.xlarge (CPU: 44, Mem: 120 GB, Disk: 60 GB)(execution time in seconds) vs. Number of points



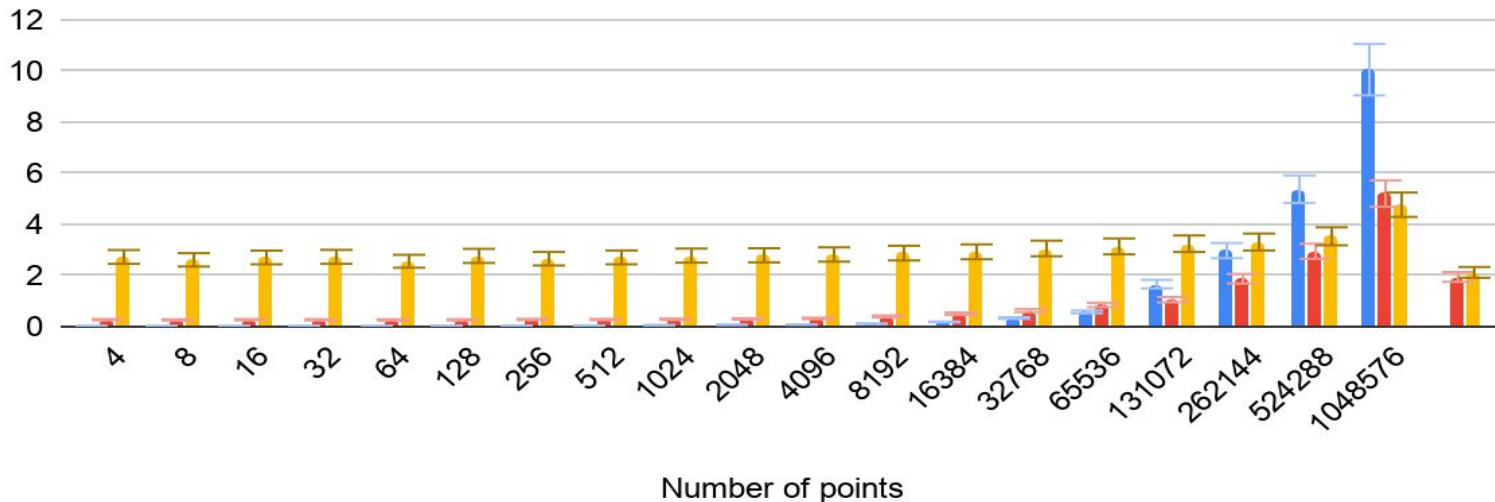
Comparison between resources

m1.tiny (CPU: 1, Mem: 2 GB, Disk: 8 GB)(execution time in seconds), m1.quad (CPU: 4, Mem: 10 GB, Disk: 20 GB)

■ m1.tiny (CPU: 1, Mem: 2 GB, Disk: 8 GB)(execution time in seconds)

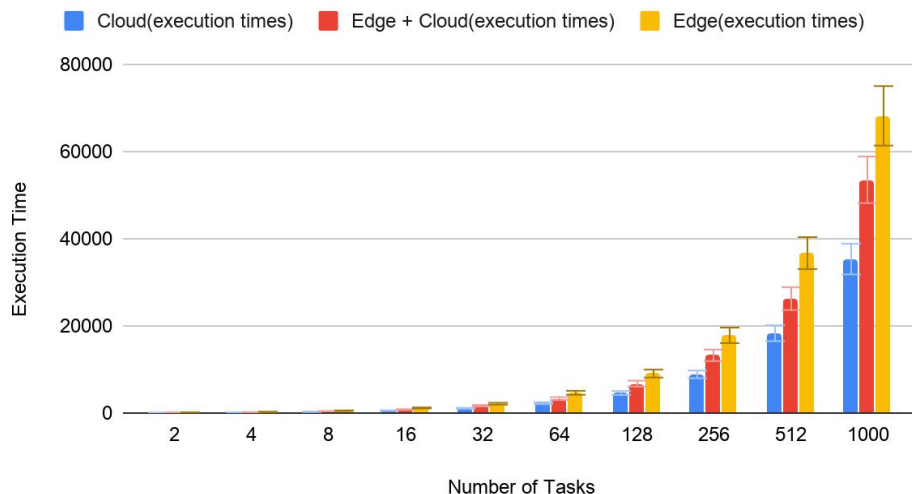
■ m1.quad (CPU: 4, Mem: 10 GB, Disk: 20 GB)(execution time in seconds)

■ m1.xxlarge (CPU: 44, Mem: 120 GB, Disk: 60 GB)(execution time in seconds)

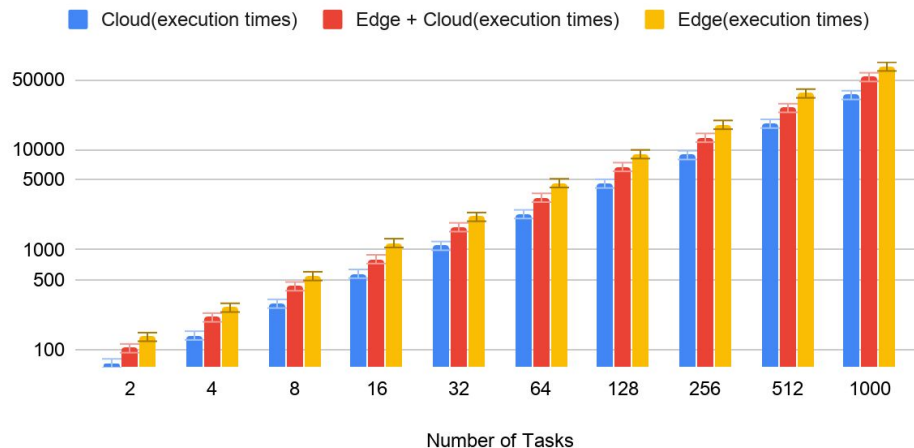


Vivek's Emulator comparison (Edge, Cloud and Edge-Cloud)

Execution time for a bag of tasks on the Emulator



Cloud(execution times), Edge + Cloud(execution times) and Edge(execution times)



The right graph has the vertical axis in log scale.

Total data transfer time and Data rate

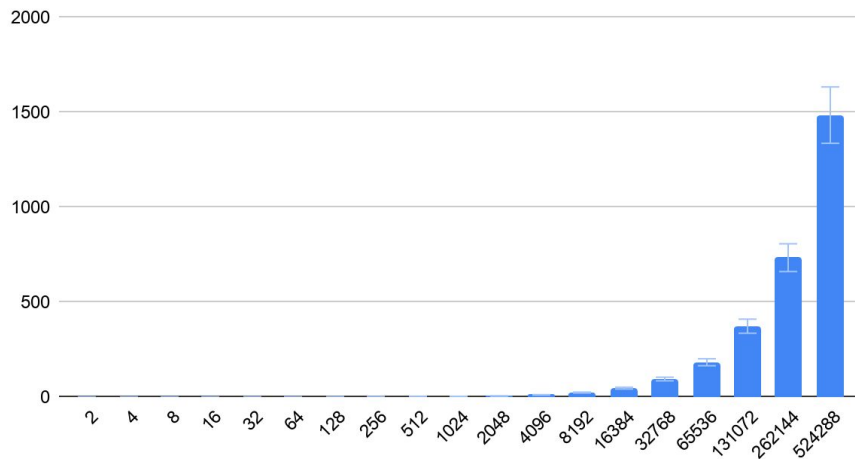
1. Resource 1: m1.small (CPU: 2, Mem: 4 GB, Disk: 20 GB)
 - a. PRODUCER
2. Resource 2: s1.large (CPU: 10, Mem: 30 GB, Disk: 120 GB, Disk: 120 GB root)
 - a. CONSUMER
3. KAFKA BROKER

Where Data rate = Number of Data_points / Total time in seconds to transfer the data

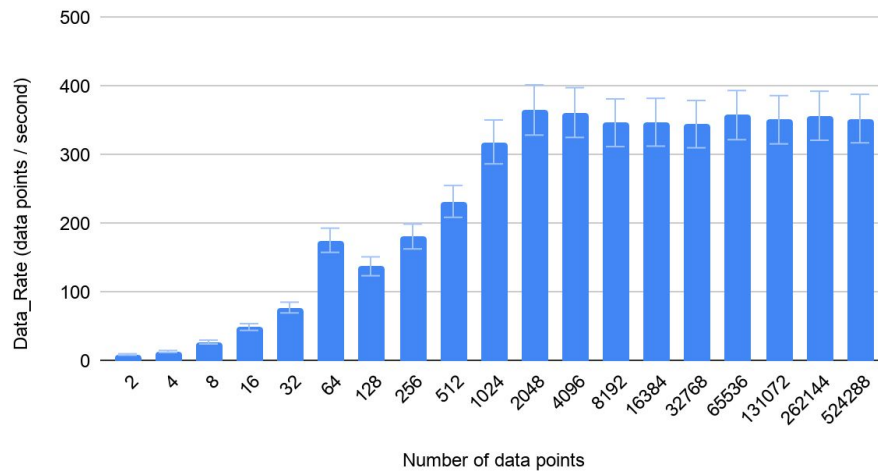
Using confluent_Kafka, I produced random points (2,4,8,16.....524288) and calculated the total time required **between generation of first point and receiving the last data point on consumer.**

Total data transfer time and Data rate

Number of data points and Total Transfer time (in seconds)



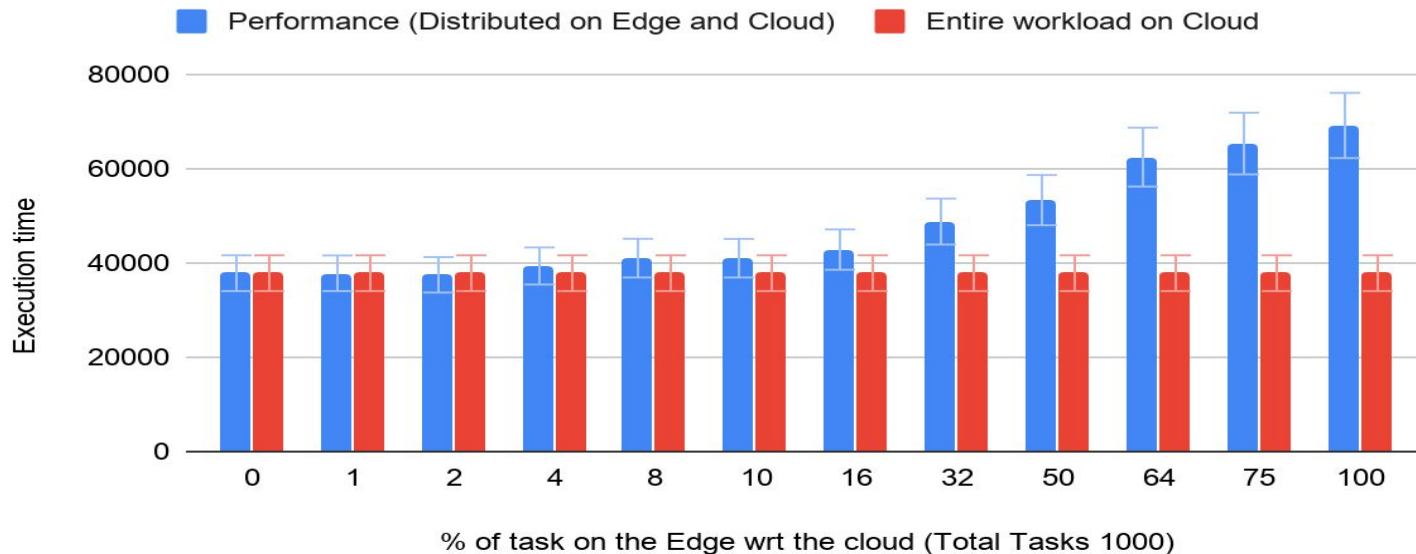
Data_Rate (data points / second) vs. Number of data points



Division of work on Edge and Cloud

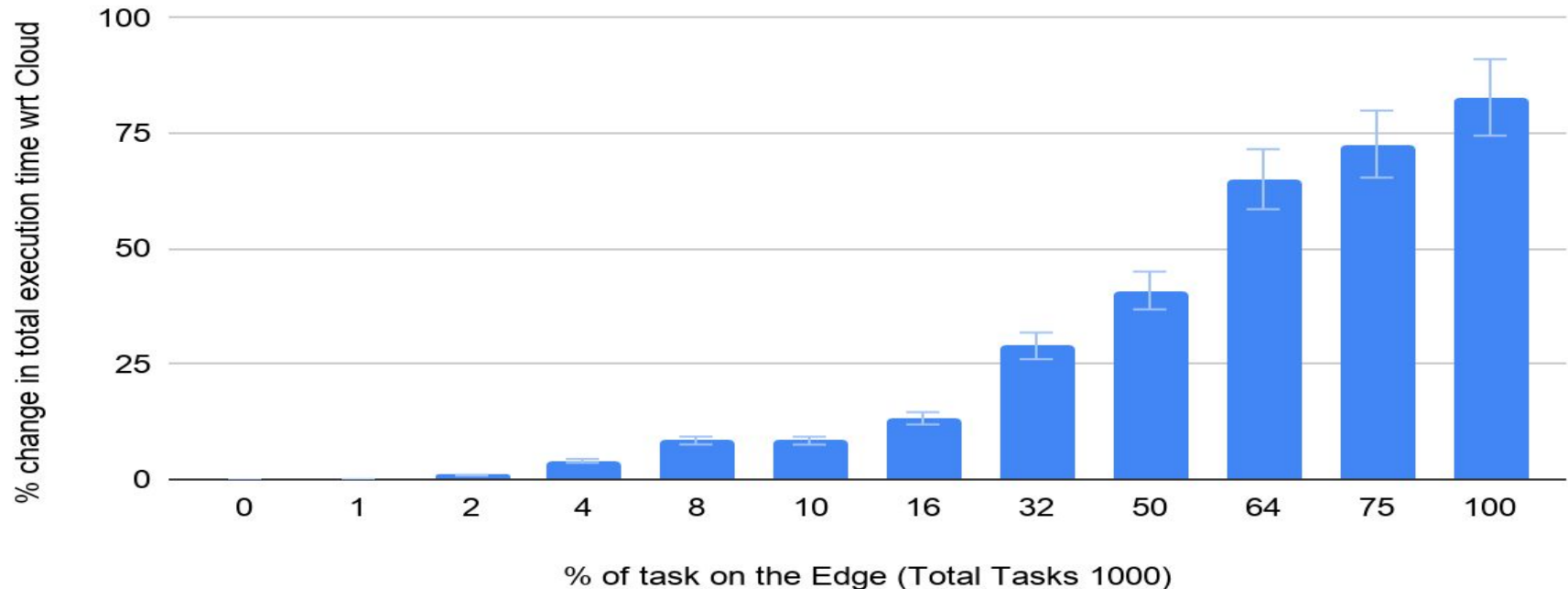
How much compute can I run on the edge without degrading my performance?

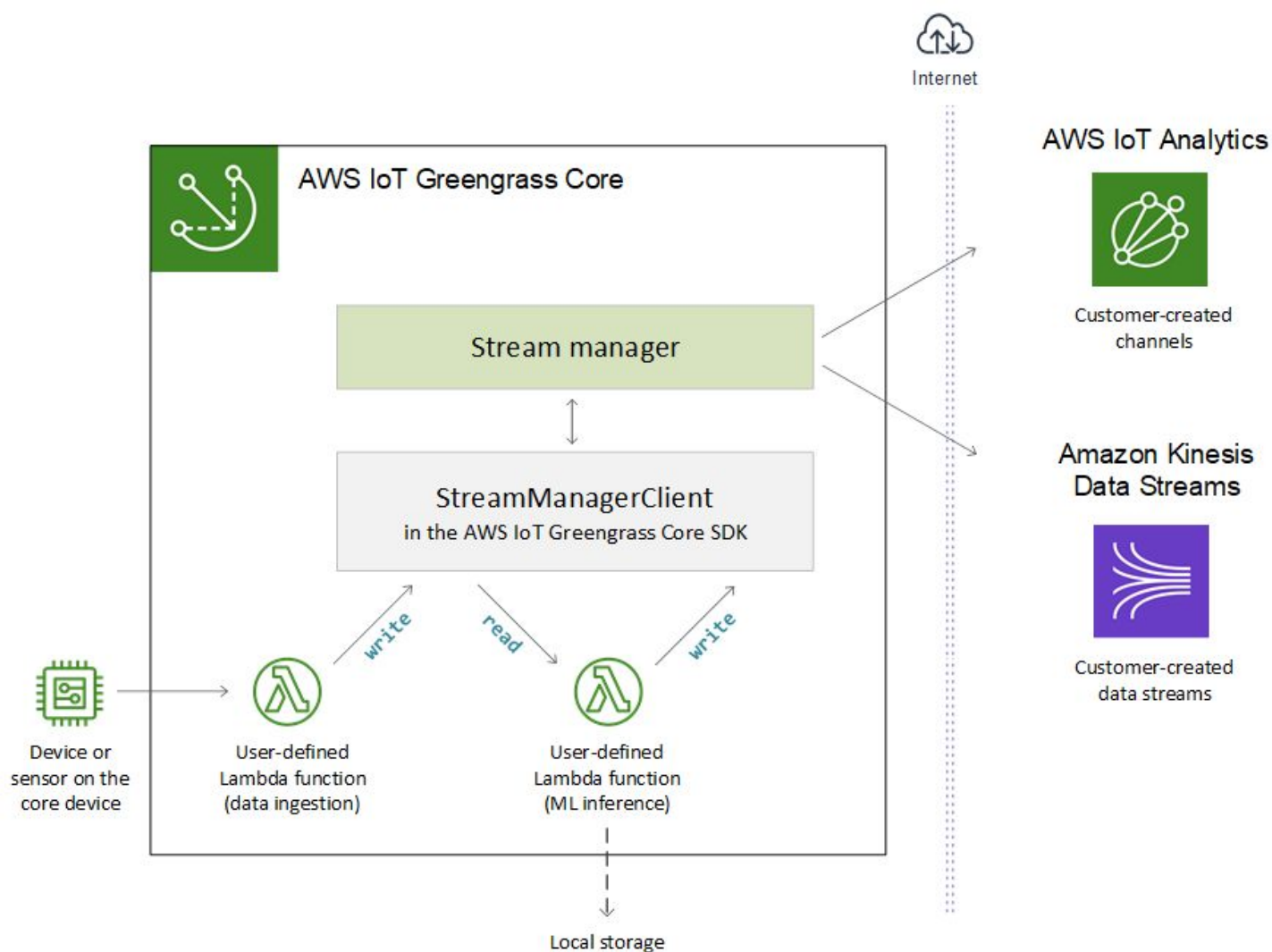
Performance Evaluation of varying workload distributions on the Edge



Division of work on Edge and Cloud

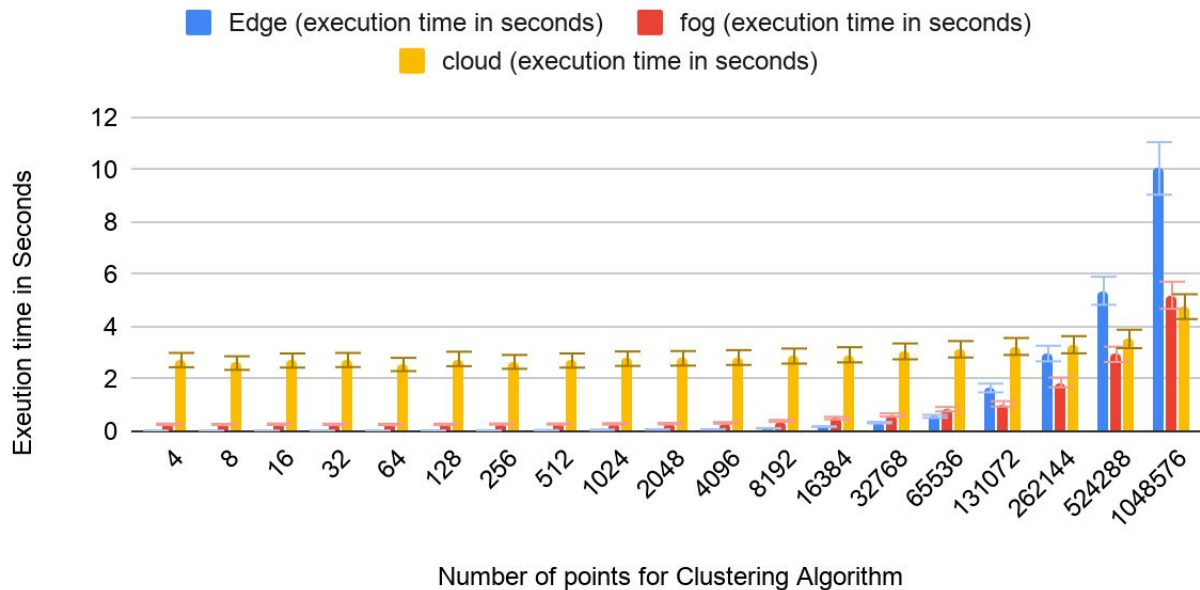
% change in total execution time with respect to total execution of workload on the Cloud





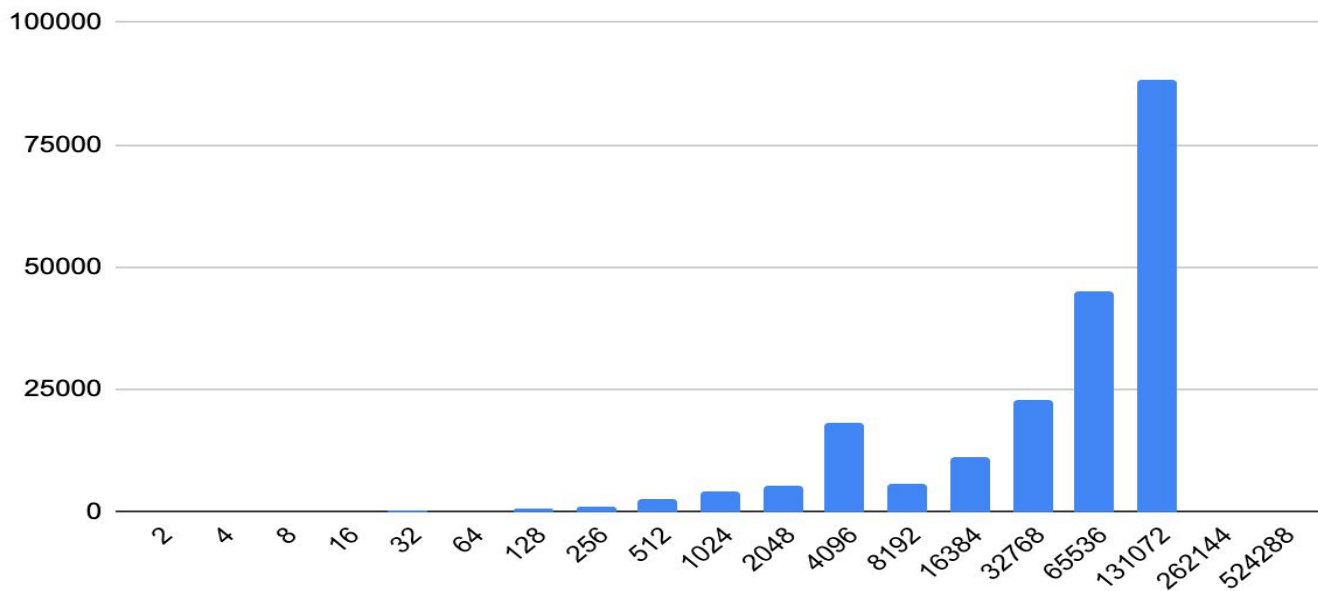
Using K_means provided with Andre

Execution of Clustering Algorithm to study Computation performance metrics



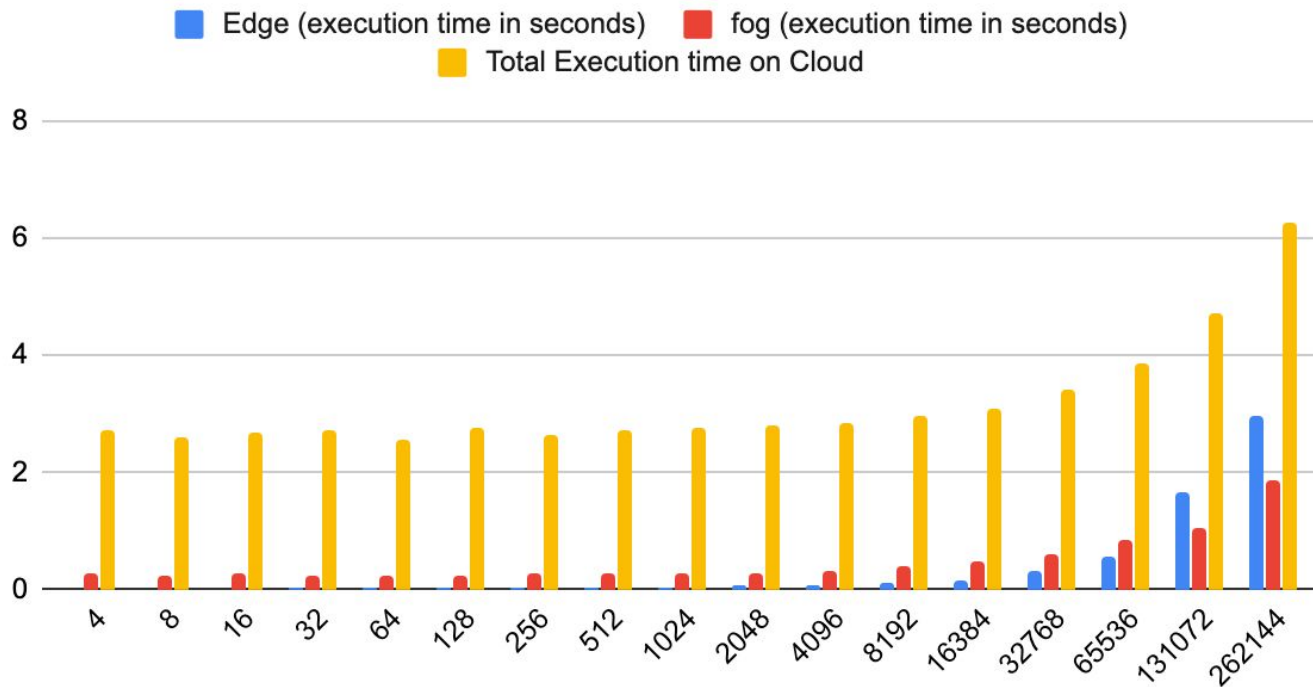
Transfer of data rate as a List: Calculate the data_rate

Number of data points and Data_Rate (data points / second)



Performance evaluation with data_rate = 88183.81658

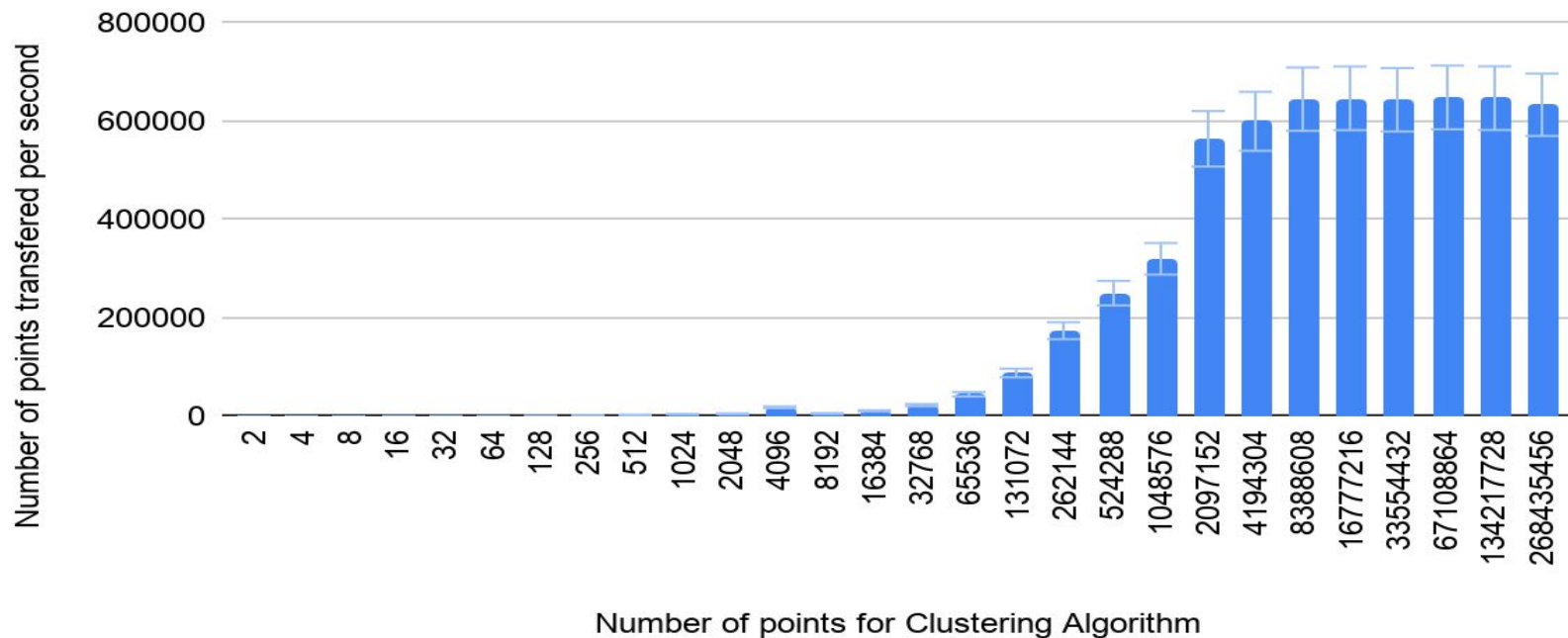
Number of points, Edge (execution time in seconds), fog (execution time in seconds) and Total Execution time on Cloud



Transfer of data rate as a List: Calculate the data_rate

'stable data rate for points greater than 131072 to steady the data rate'

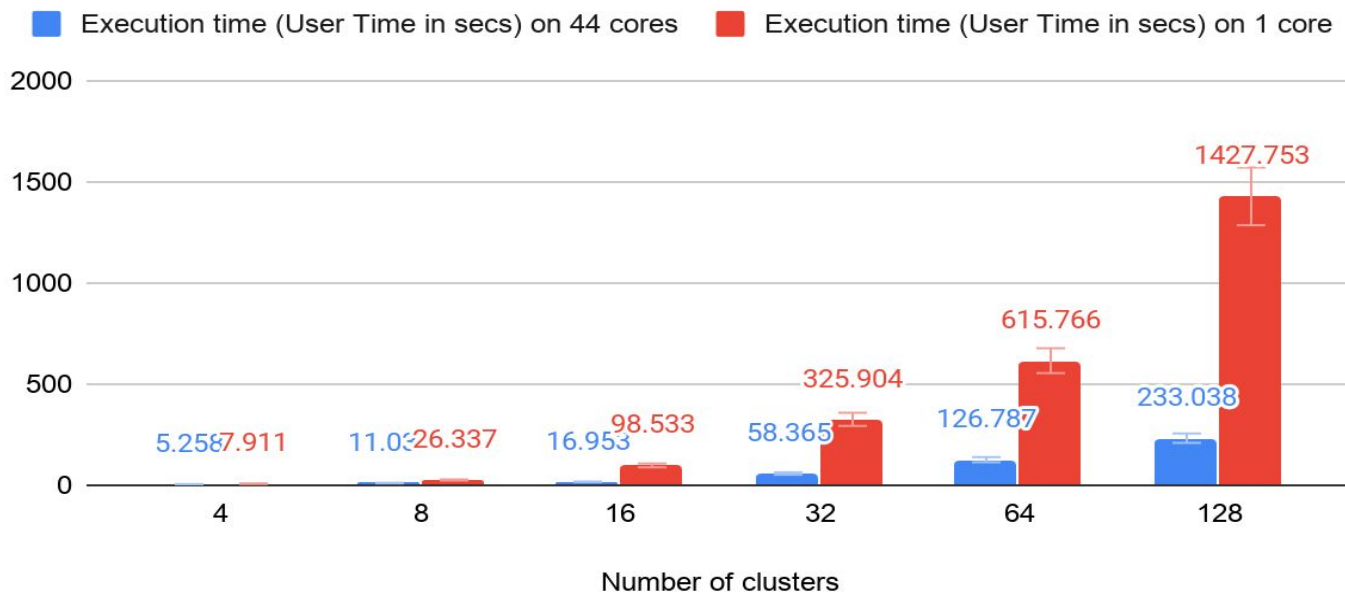
Execution of Clustering Algorithm on XSEDE resources to
Study latency Metrics



Performance comparison as complexity increases

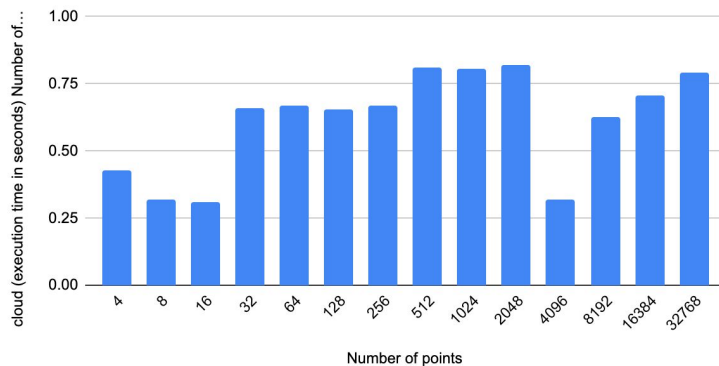
‘Trade-off between working on Edge and Cloud’

Execution time (User Time in secs) on 44 cores and Execution time (User Time in secs) on 1 core

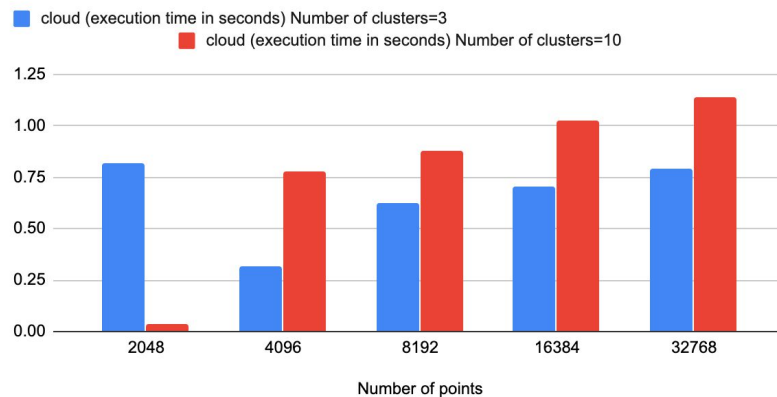


End to end experiment for transfer of data and running K-Means clustering

cloud (execution time in seconds) Number of clusters=3 vs.
Number of points

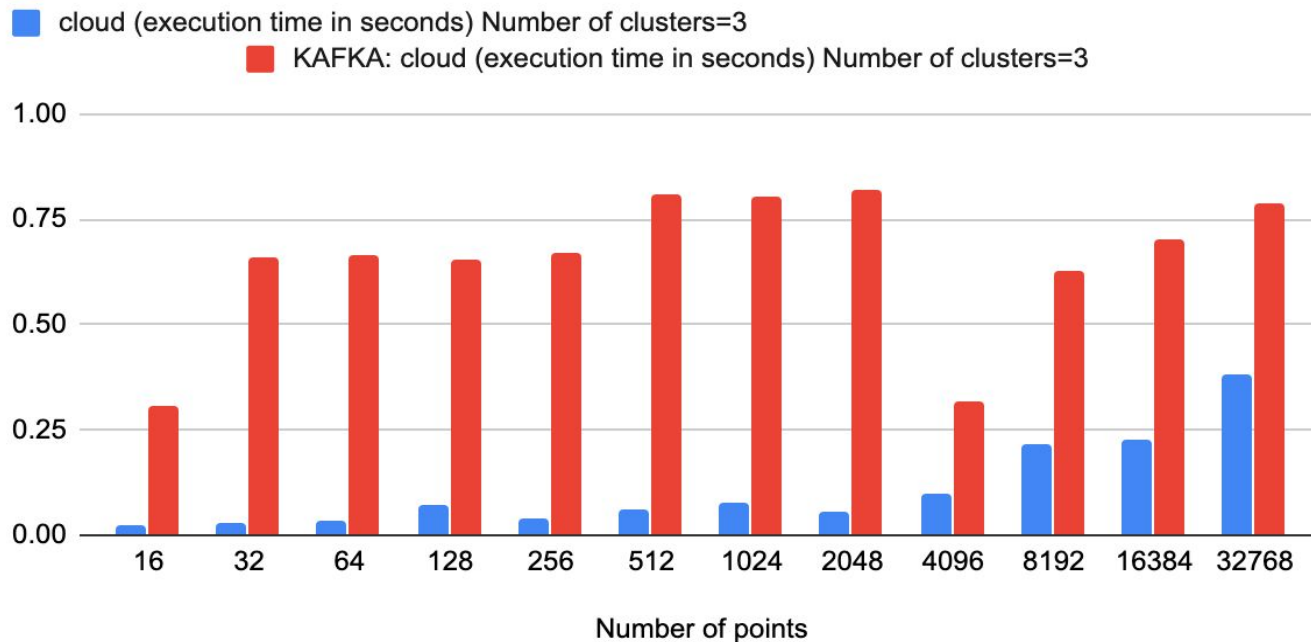


cloud (execution time in seconds) Number of clusters=3 and
cloud (execution time in seconds) Number of clusters=10



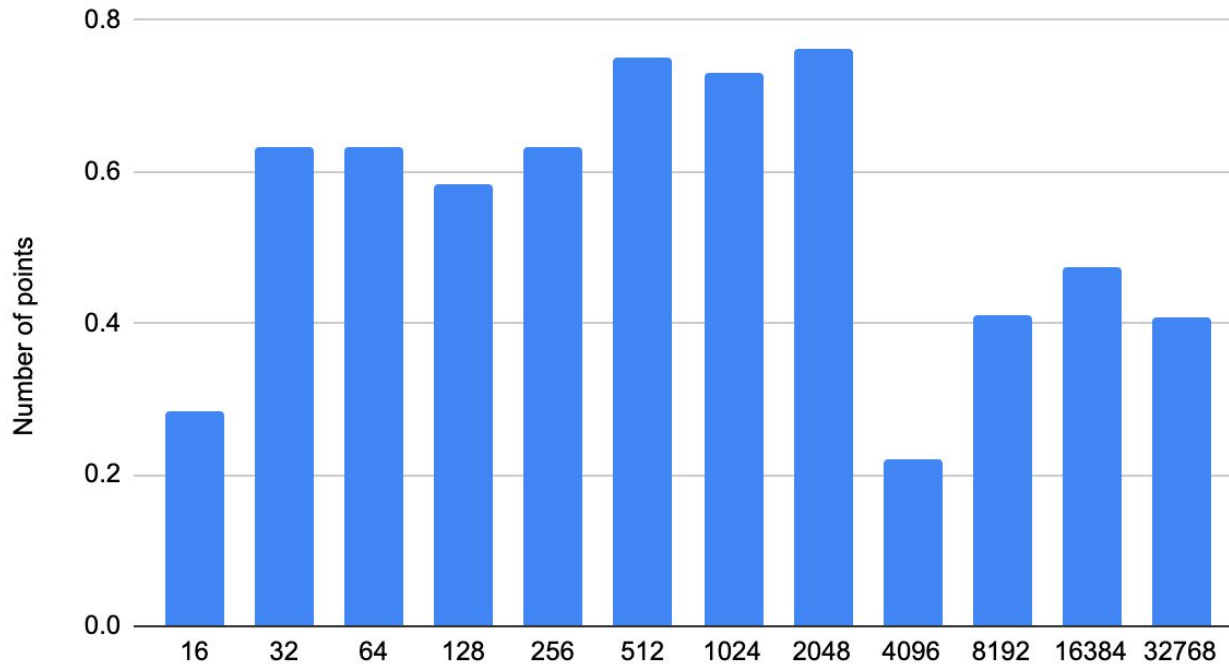
Benchmarking Andre K_Means on large (num cores=10)

cloud (execution time in seconds) Number of clusters=3 and
KAFKA: cloud (execution time in seconds) Number of cluster...



Transfer time (Computation E2E - benchmarking (just K-Means)) ?

Number of points Vs Transfer time in seconds



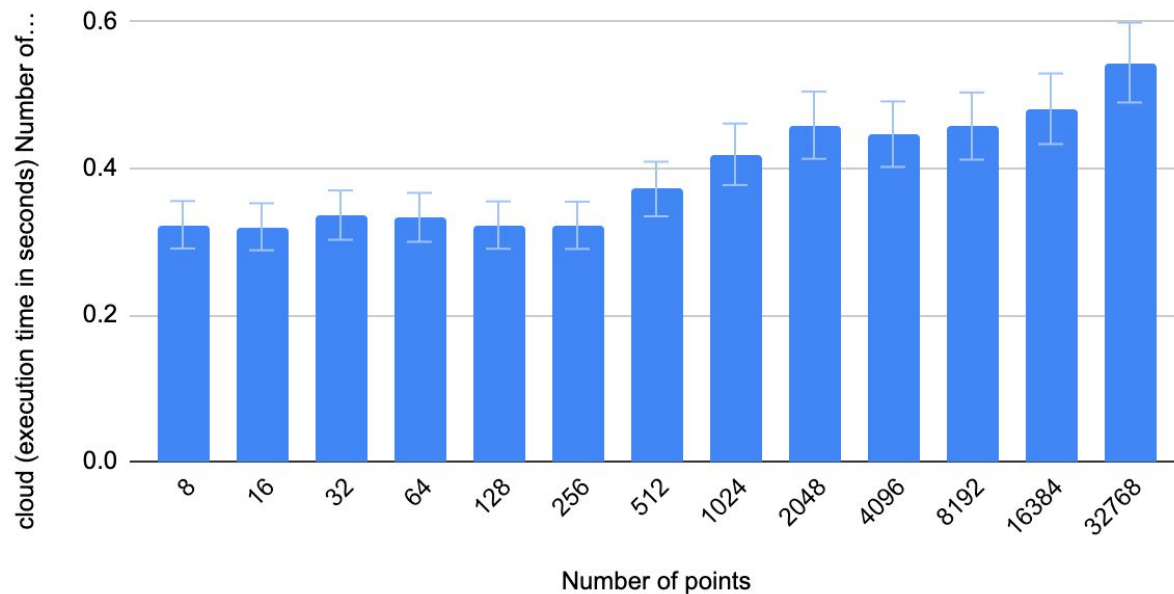
NTP-date: synchronize two machine upto milliseconds

```
(conkafka) krb09@js-171-25:~/kafka/scripts$ date  
Thu May 28 14:22:43 EDT 2020
```

```
(conkafka) krb09@js-168-81:~/kafka/scripts$ date  
Thu May 28 14:22:43 EDT 2020
```

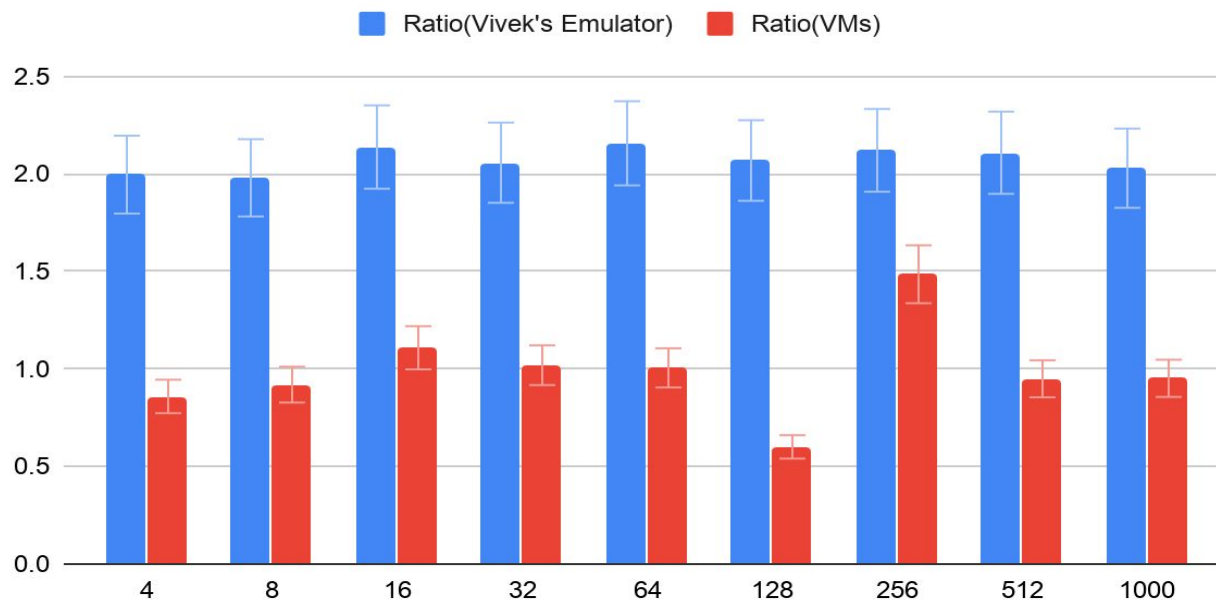
End to end experiment for transfer of data and running K-Means clustering

cloud (execution time in seconds) Number of clusters=5 vs.
Number of points



Vivek's Emulator Reproduction of results

Number of Tasks, Ratio(Vivek's Emulator) and Ratio(VMs)



Questions

1. How do I bring the systems (VMs and Vivek's Emulator) to the same scale?
 - a. Computation in emulator is: Ops/execution power (this is not in seconds?)
 - b. Computation time in VMs is seconds
2. There is also some constant time for cloud resources (starting the cores?). How do I include that in the Vivek's emulator?

Difference between the two systems

- Vivek's Emulator:
 - 1 Task = x operations (x is defined wrt a distribution)
 - Show data of 50 tasks (define a uniform distribution with mean = 1000), each task consist of 1000 operations
 - Execution time = Ops/execution power + data_transfer time (Ops / data_rate)
 - Time for distribution of workload on cores is not benchmarked, significant part of execution in
- VMs
 - Execution time = Transfer time via Kafka brokers + Computation time on VMs is seconds
 - Also, includes time for distribution of workload on cores: Overheads

Other way round: Verify the Emulator on VMs

- Resource 1: Edge
 - m1.tiny (CPU: 1, Mem: 2 GB, Disk: 8 GB)
- Resource 2: Cloud
 - m1.xlarge (CPU: 44, Mem: 120 GB, Disk: 60 GB)
- Should I try to split the K-means to run x% points on Resource 1 and then the total -x % points on Resource 2 to see if similar numbers are generated?
 - This would be a good approach? Openstack API?

Next Steps


- Breakdown execution for edge devices
 - Maximum data set that can give us good execution times
 - What is good?
- Continue Experimenting with diff Clustering sizes
 - Why are we doing this?
- Applications other than K-Means
- Increase the Parameter Space
 - Diff data rates
 - Increase the number of VMs in the experiment: Openstack API

Model: Emulator works

- Working

- Extensions done : https://docs.google.com/presentation/d/18sciSqvggFaK2OLIJhqkJJz99NW8MwlwCDZvodwRgvl/edit#slide=id.g882a99e560_0_135

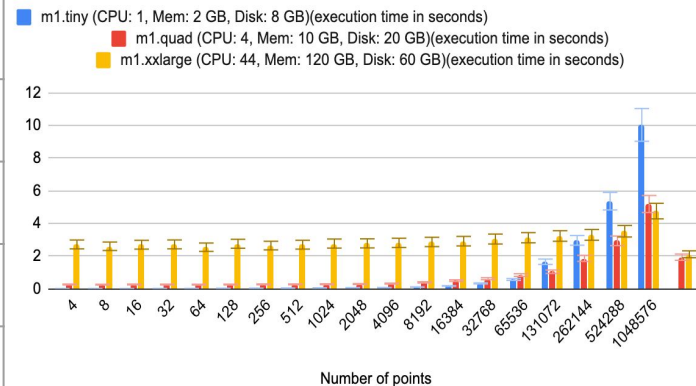
Task management overhead?

1.  Compute resources have already been acquired and are ready to be used for task execution.
2. The size of the acquired resources remains constant. We consider dynamism only in the performance of the acquired compute resources.
3. Entities providing information regarding the performance of the compute resources are external to the system and are assumed to be accurate.
4. We assume that the performance of the cores do not vary during execution but vary only between different task submissions.
5. We assume that there is no cost to make scheduling decisions, i.e., time to make scheduling decisions is 0, and does not influence the scheduling decision.

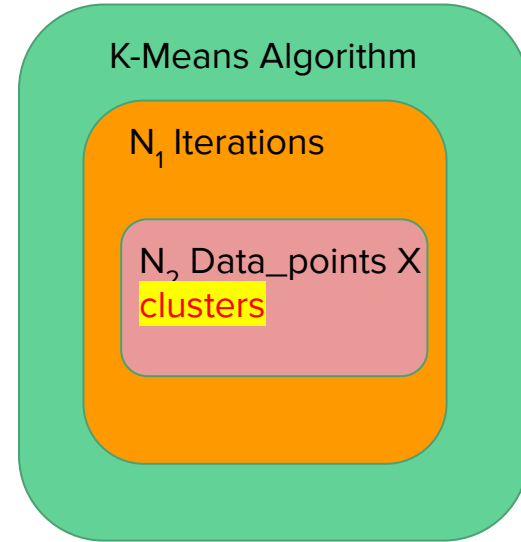
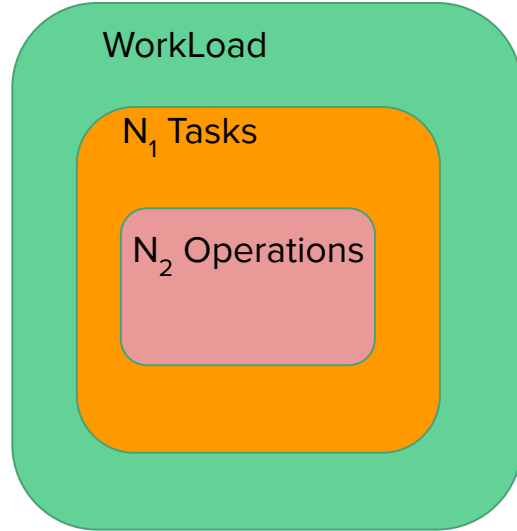
Overhead for Xsede Machines

Number of cores used	Start-time(secs)	End-time(secs)	Total execution time=overhead(secs)
1	1592836927	1592836927	0
2	1592836914	1592836914	0
4	1592836899	1592836899	0
8	1592836716	1592836716	0
16	1592836690	1592836690	0.009999990463
32	1592836660	1592836660	0.01999998093
44	1592836290	1592836290	0.02999997139

m1.tiny (CPU: 1, Mem: 2 GB, Disk: 8 GB)(execution time in seconds), m1.quad (CPU: 4, Mem: 10 GB, Disk: 20 GB)(exe...



Relate between the Emulator and the VMs experiment



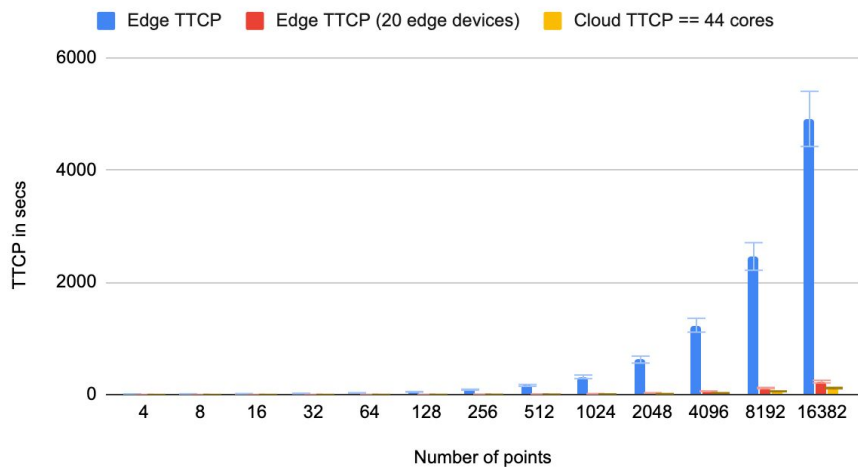
Mathematical deduction?

K- means Clustering:

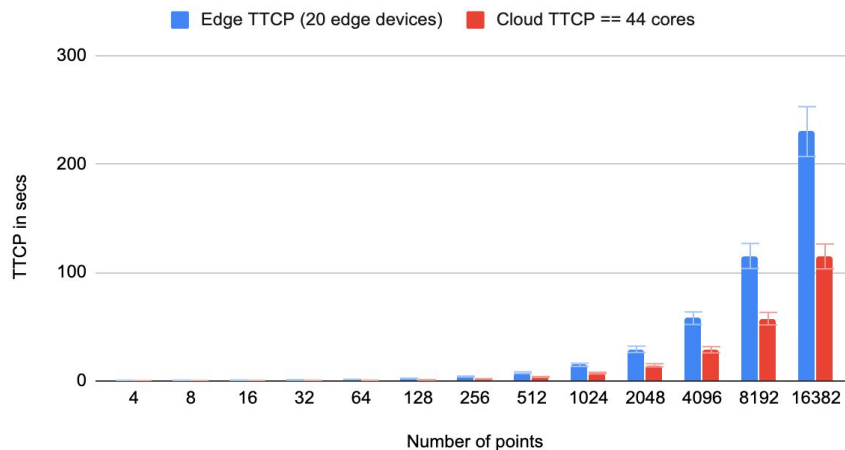
- Each iteration has number of points * **number of clusters** operations which do not influence the total complexity.
 - For instance 512 points * number of cluster,
Each iterations = $512 \times i \sim 512 * \text{number of clusters operations}$ ————— (1)
- In the scikit K-means script, we have 300 iterations completed = 10.04 seconds (for an edge device)
- 1 iteration = $10.04 / 300 \text{ seconds} \Rightarrow D$ —————(2)
 $\Rightarrow 1 \text{ second} = 512 \times 1/D \text{ operations ??}$ —————from (1) and (2)

K_means Experiment on Emulator

TTCP vs Data_points

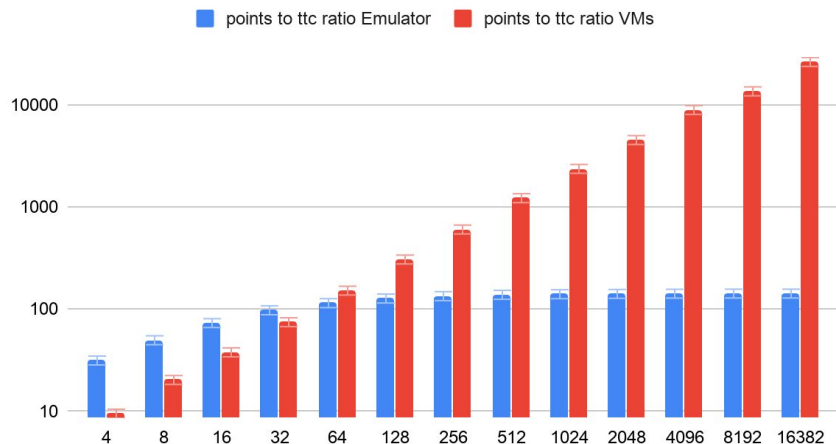


TTCP vs data points

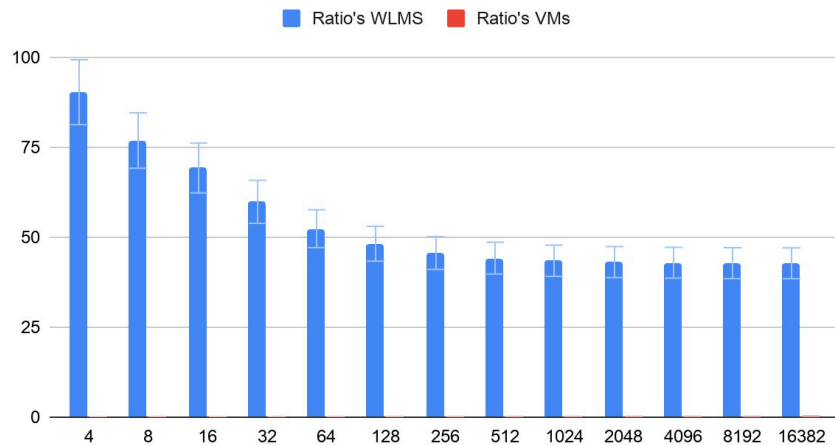


Ratios between data_points, and devices computation times

points to ttc ratio Emulator (LOG SCALE)

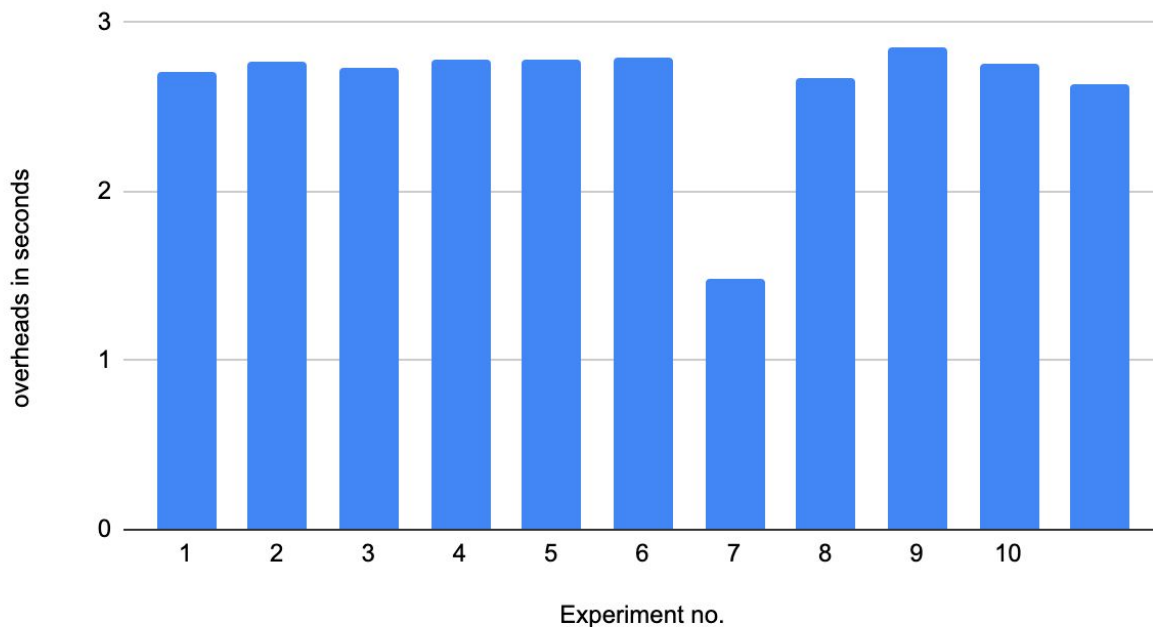


performance on Edge / performance on clouds



Overheads: one point one cluster K-Means on VMs

1 point 1 cluster experiment: Overheads?

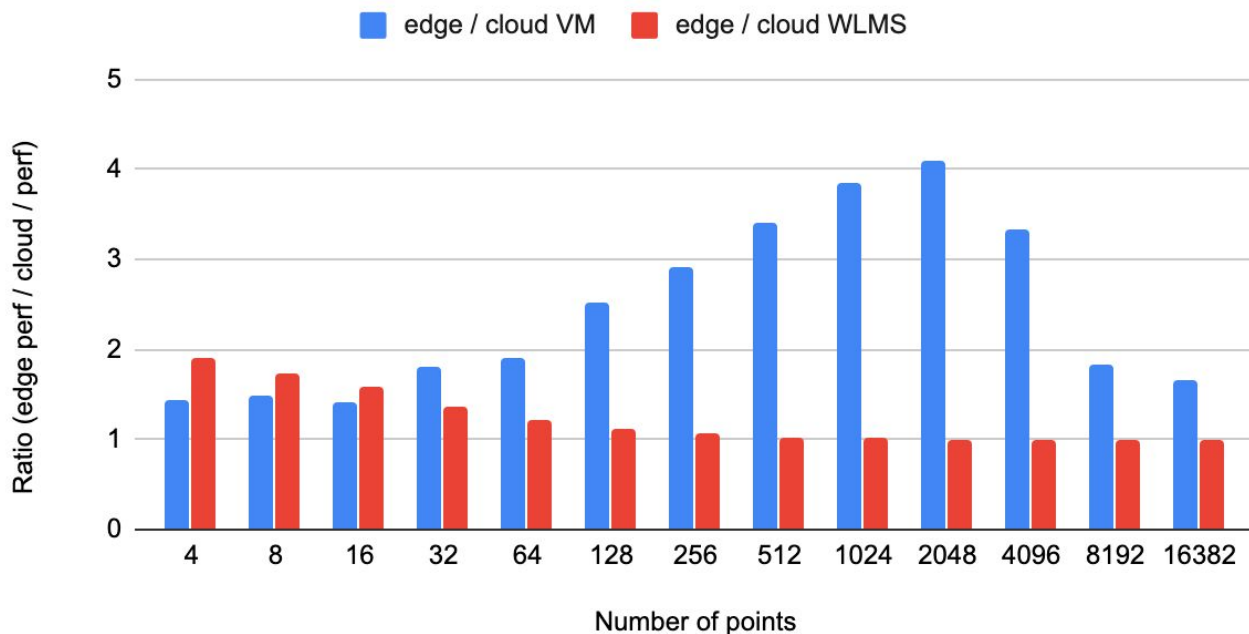


Ratios between data_points, and devices computation times

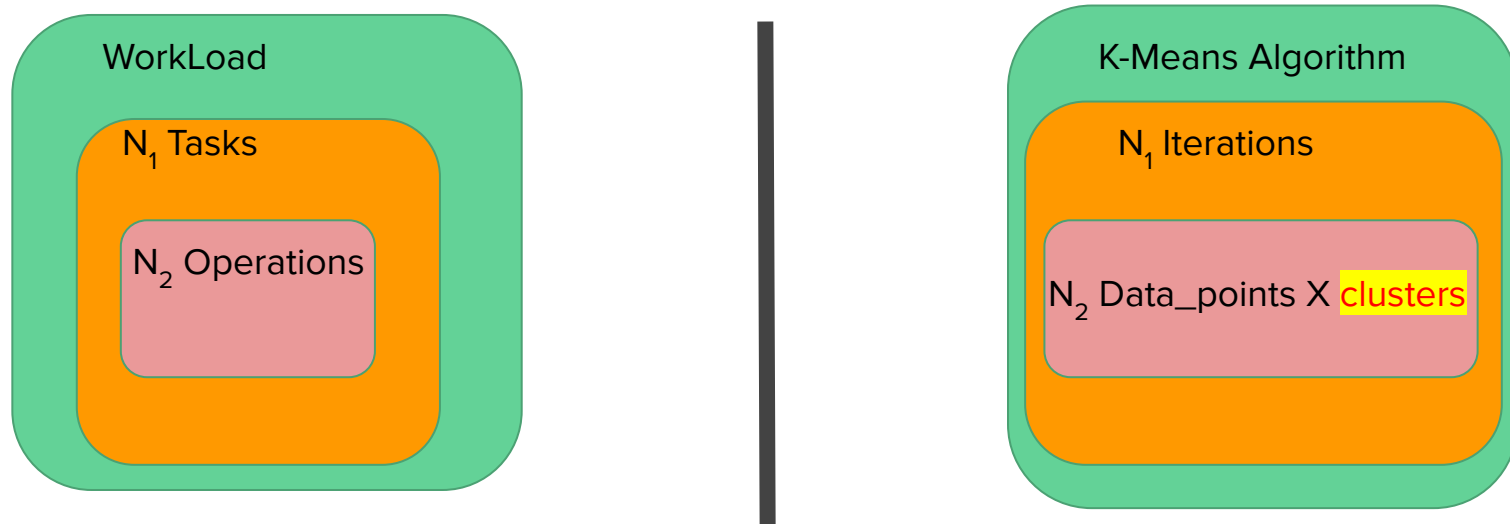
- Apples and oranges
- I repeated the experiments on the VMS and the emulator for one core (cloud devices).

Ratios between data_points, and devices computation times (1 CORE)

Ratio of performance on edge & performance of cloud VS points



Relate between the Emulator and the VMs experiment



VMs have 1 core per resource, therefore no concurrent execution of K-means. That is each iteration happens on 1 core and then we move on to the next iteration sequentially.

For the emulator, the same thing happens. Each task (== each iteration) happens sequentially.

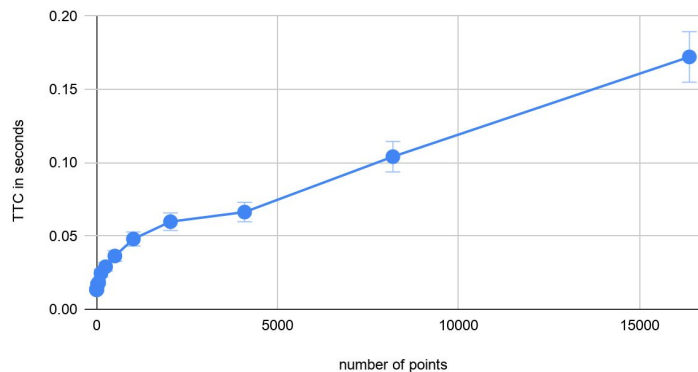
Ops per second to calculate Total time to execution

In the emulator: total time to execution = total ops in a task / performance of core

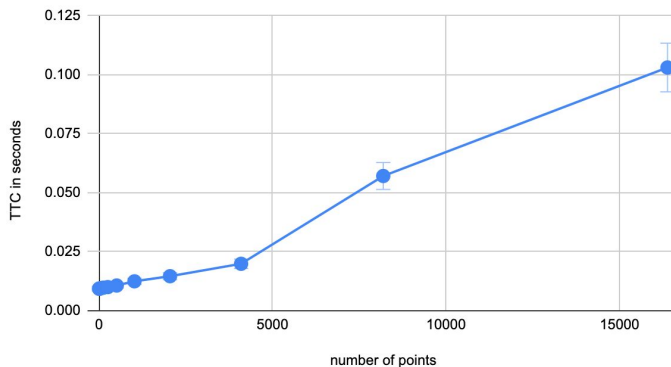
*performance of core = ops per second.

Tried to mimic the performance of these resources from the VM on jetstream

Performance of K-means on edge



Performance of K-means on Cloud



Ops per second to calculate Total time to execution

K-means clustering for D points in 3 cluster (300 iterations) = T secs

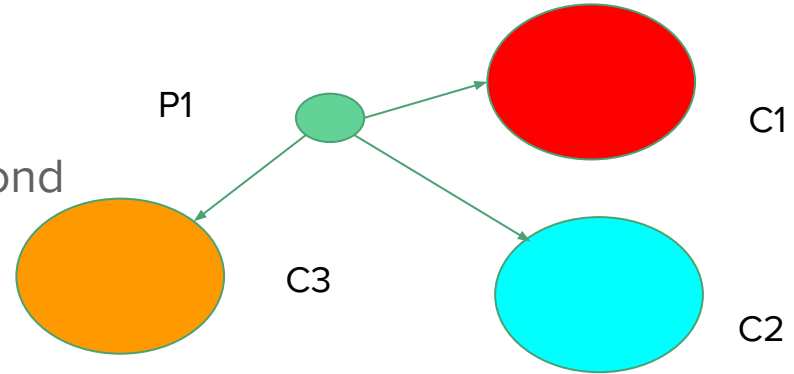
$$\Rightarrow T \text{ secs} = D \times 3 \times 300 \text{ ops}$$

$$\Rightarrow 1 \text{ sec} = (D \times 3 \times 300) / T \Rightarrow \text{ops per second}$$

Feeding this into the Emulator:

$$\Rightarrow \text{TTC} = \text{number of ops } (D \times 3) / [(D \times 3 \times 300) / T] \Rightarrow T / 300 \Rightarrow \text{constant for the size of data points.}$$

T differs for each of the data-size. So do I need to enter each time. But the number of ops per sec on a resource should be constant?



How do I calculate the ops per second for Emulator

1. I can figure out the ratio of performance between the two resources.

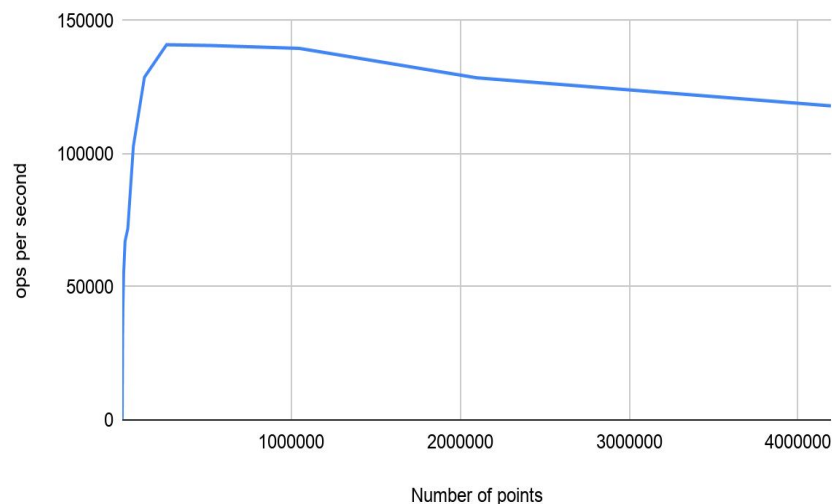
BUT

How do I calculate the number of ops done per second by the resource?

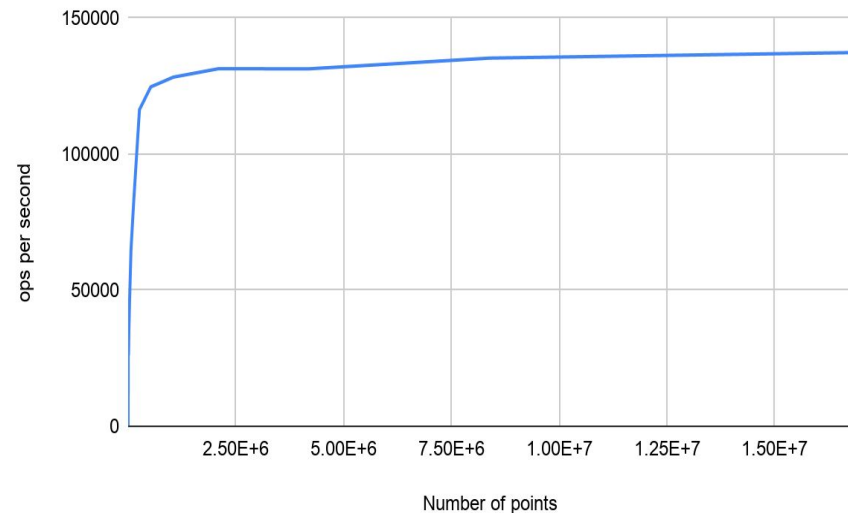
2. Till now I have been calculating the number of ops done to complete the K-means algorithm. How do I calculate the general performance?

July,16th 2020:Performance: Operations per second (steady value)

Number of ops per second for K-means on m1.tiny



Number of ops per second for K-means on m1.xlarge

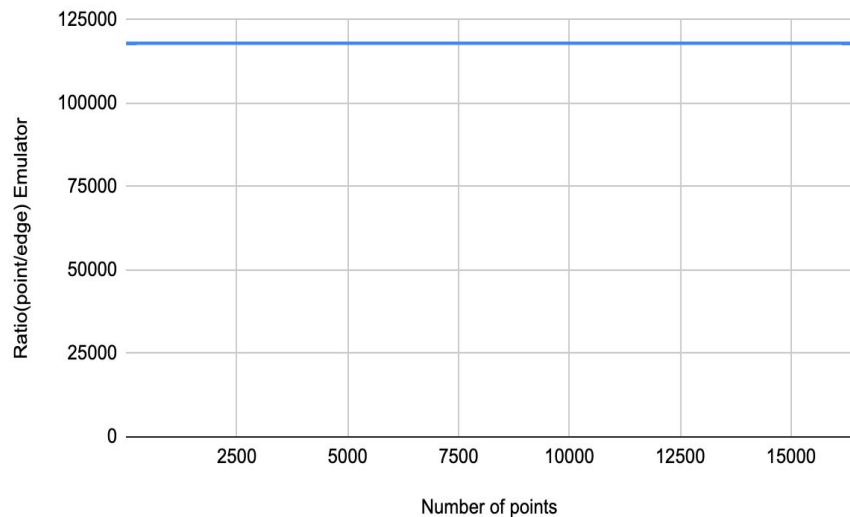


Calculating 4 ratios

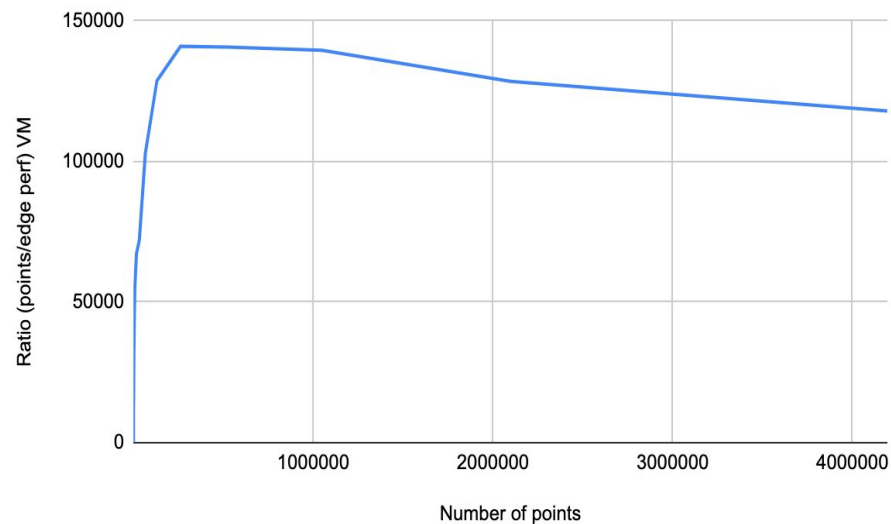
1. Ratio (points / perf.edge)
2. Ratio (points / perf.cloud)
3. Ratio(edge.perf / cloud.perf): Emulator Vs VM
4. Ratio(edge.emulator / edge.vm)

Ratio (Data_points / perf.edge)

Ratio(point/edge) Emulator

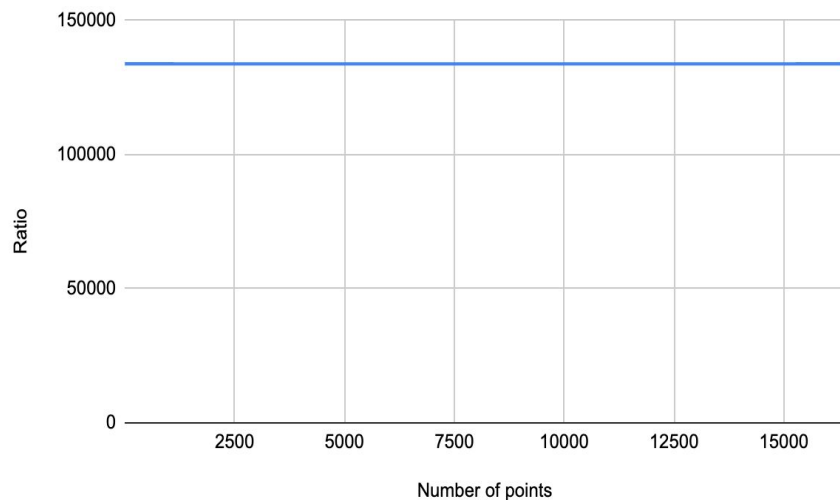


Ratio (points/edge perf) VM

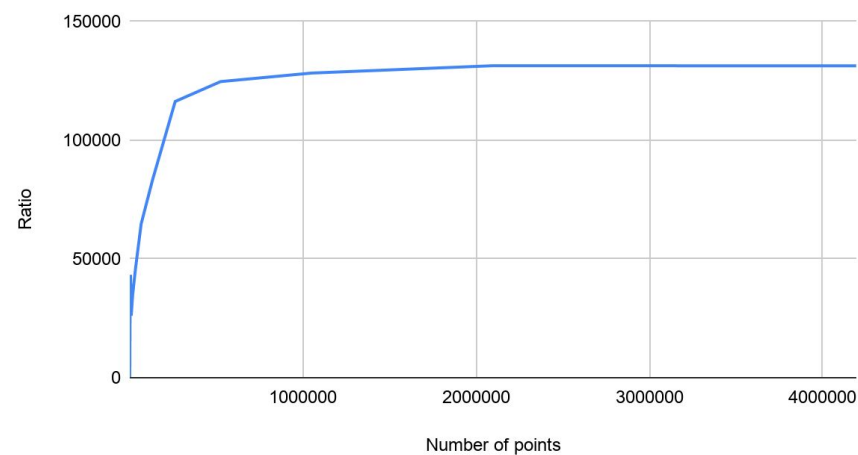


Ratio (points / perf.cloud)

Ratio(point/cloud) Emulator

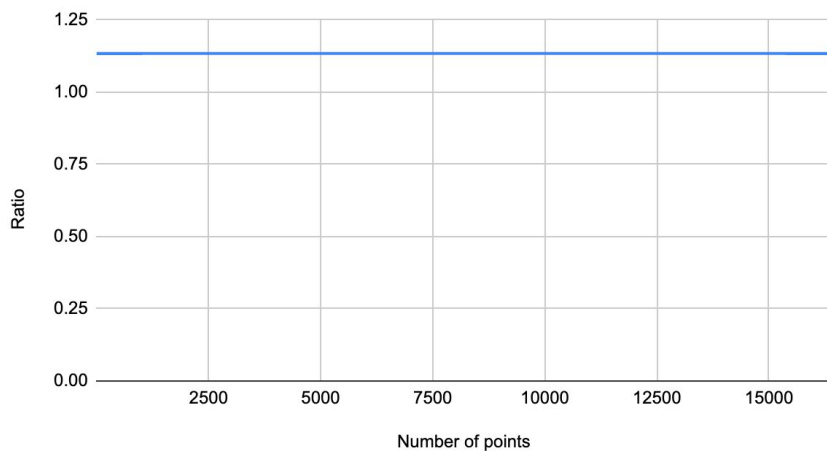


Ratio (points / cloud.perf) VM

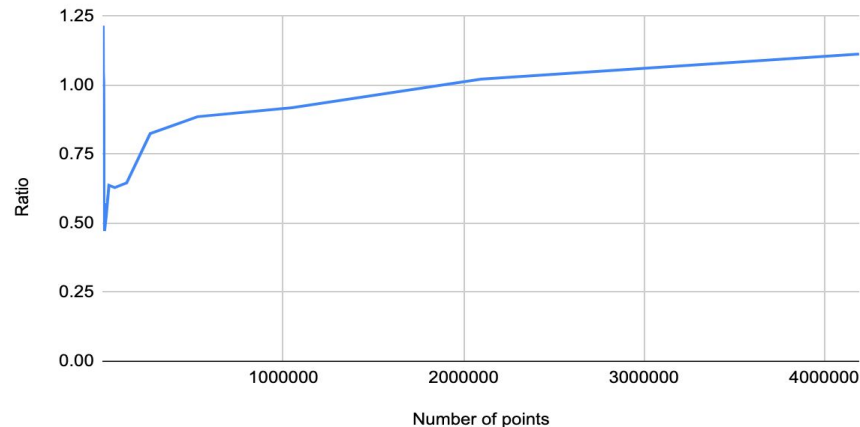


Ratio(edge.perf / cloud.perf): Emulator Vs VM

Ratio(edge/cloud) emulator

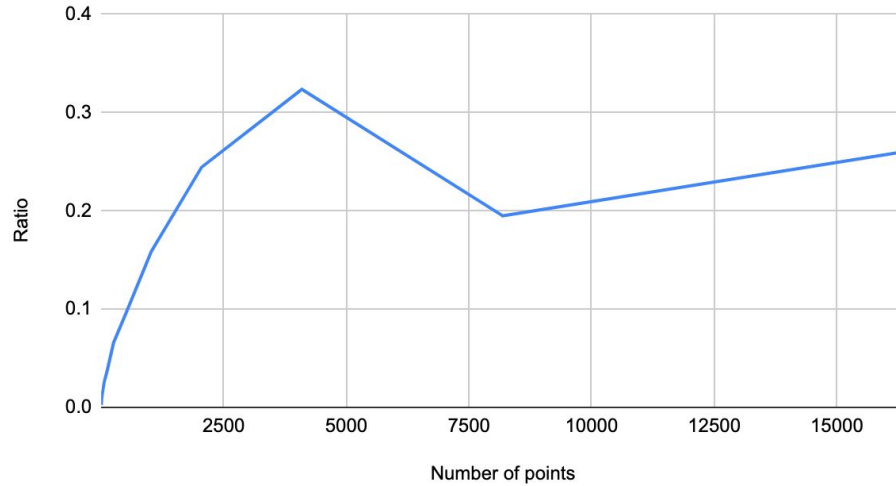


Ratio (edge / cloud) VM

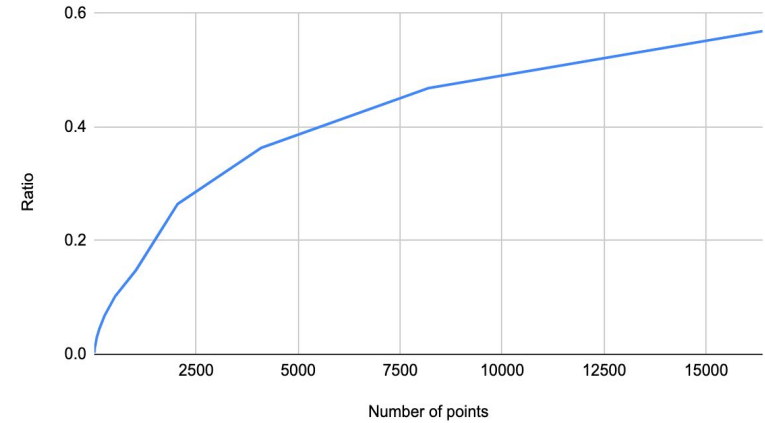


Ratio: (edge.emulator/edge.vm) & (cloud.emulator/cloud.vm)

Ratio(cloud.emulator/cloud.vm)



Ratio(edge.emulator/edge.vm)

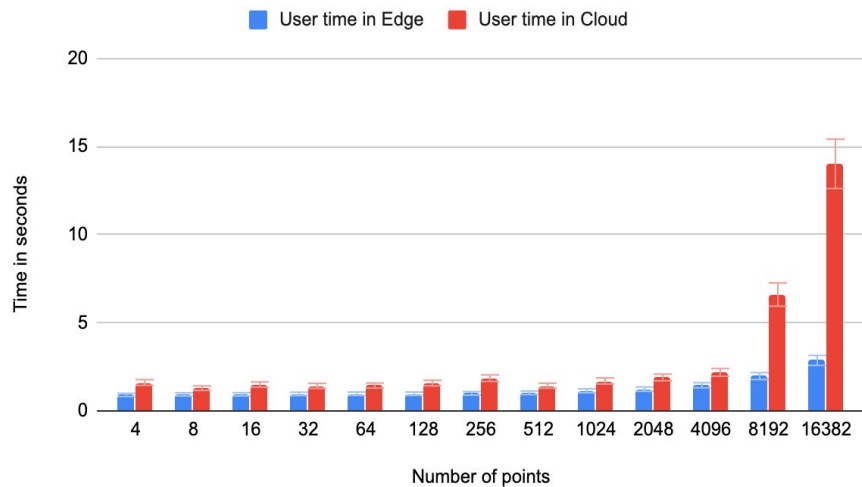


July20th, 2020: Real, User and Sys time

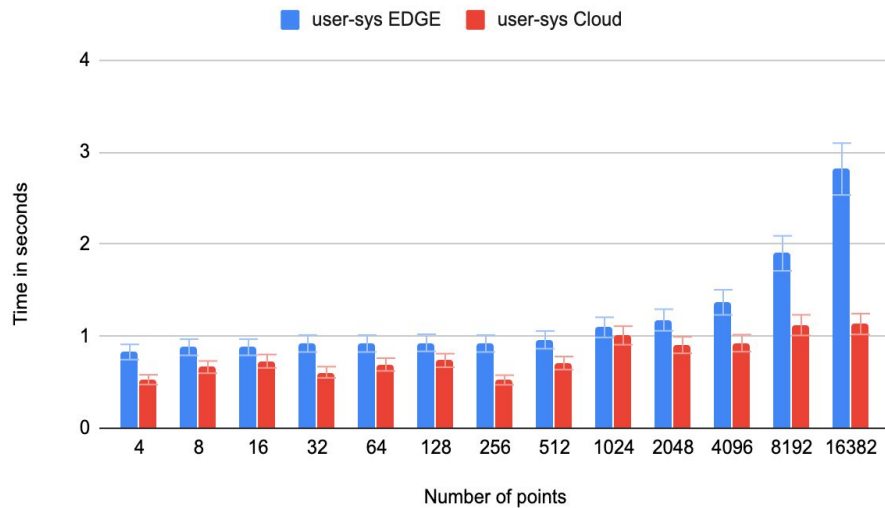
- Real is the actual amount of time it took to run the command (as if you had timed it with a stopwatch)
- User and sys are how much 'work' the CPU had to do to execute the command. This 'work' is expressed in units of time.
 - User is how much work the CPU did to run to run the command's code
 - Sys is how much work the CPU had to do to handle 'system overhead' type tasks (such as allocating memory, file I/O, ect.) in order to support the running command
- Real experimental time = User - sys?

(User time) Vs (User time Vs Sys Time)

User Time: Edge Vs Cloud

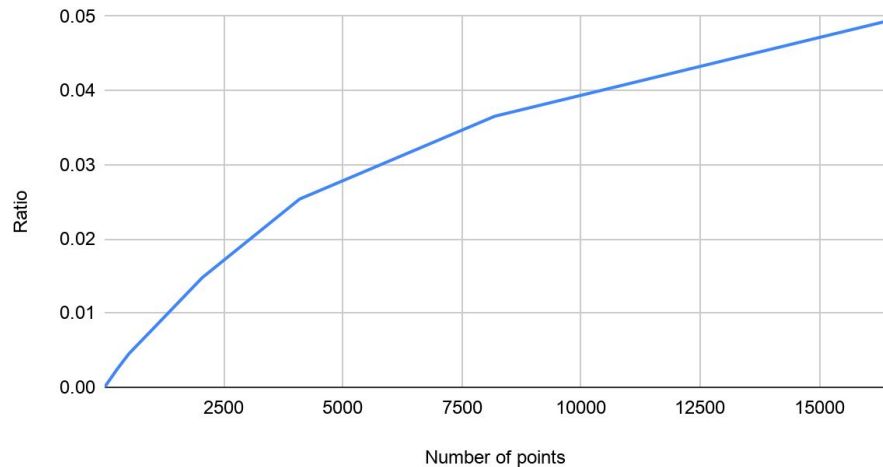


(user-sys) Edge Vs Cloud



Ratio: (edge.emulator/edge.vm) & (cloud.emulator/cloud.vm)

Ratio (Edge.perf on Emulator/Edge.perf on VM)



Ratio (cloud.perf on emulator / cloud.perf on VM)

