Performance Engineering Track, Oct 7, Denver 2024

Paul Brebner and Roger Abelenda (chairs)

Welcome to Denver! The track train

Who am I?

- Why Performance Engineering & Open Source?
- Background (too many decades) in R&D in distributed systems and performance engineering
- Before joining Instaclustr CTO of a NICTA startup
  - Automated performance modelling from distributed traces
  - Lots of Australian government and enterprise customers
- 7 years+ as Instaclustr technology evangelist
  - Open Source Big Data technologies
  - Opportunity for regular performance & scalability experiments and analysis
  - Lots of blogs and conference talks, invited keynotes (e.g. International Conference on Performance Engineering cloud workshop), etc
  - First ApacheCon talks Las Vegas & Berlin 2019

Head of Kafka, Prague (Paul Brebner)

Motivation? First CFP!

- Why a Performance Engineering Track?
  - Because many Apache projects address domains with software performance and scalability challenges (E.g. Web, Cloud, Databases, Streaming Data, Big Data, Data Analytics, Search, Geospatial, etc) = Problems
  - While others provide performance engineering tools (E.g. benchmarking, testing, monitoring, etc) that are widely used = Solutions
  - The track will provide opportunities for cross-fertilization between projects of different software categories and maturity
    - Including incubator projects
- Open Source + Performance Innovation? (E.g. code analysis, simulation?)
  - Not yet, but one talk on byte code analysis for Camel was close, and LLMs have potential!
  - "Performance Prediction From Source Code Is Task and Domain Specific"
  - https://ieeexplore.ieee.org/document/10174021
- CFPs and track summaries are all in my LinkedIn profile

(1st train) "Bullet Trains" are Fast and Scalable! (Source: Adobe Stock)

Previous track events

Thanks to co-chairs (often same time-zone as event), reviewers, and volunteers and planners/conference PC

1. ApacheCon NA New Orleans 2022 – Sharan Foga
2. C/C Asia Beijing 2023 - Willem Jiang
3. C/C NA Halifax 2023 - Roger Abelenda
4. C/C EU Bratislava 2024 - Stefan Vodita
5. C/C Asia Hangzhou 2024 - Yu Xiao
6. C/C NA Denver 2024 - Roger Abelenda

Approx 25% acceptance rate, 34 talks, 600+ attendees

Talk acceptance algorithm = Performance Engineering + Apache Project (or open source value) + Interesting

Track train mascots

Talks on diverse performance engineering topics and these technologies

- Apache Kafka
- Apache JMeter & Selenium
- Kubernetes
- Apache Arrow
- Java Profiling
- Apache Flink
- Apache Spark/ML
- Apache Hadoop
- Apache Ozone
- Apache Cassandra
- Apache Camel
- Apache Lucene
- Apache Iceberg
- Apache Impala
- Oxia
- Apache Skywalking
- Apache Fury

Today’s talks

1. 10:50 am - Paul Brebner (co-chair), **Making Apache Kafka even faster and more scalable**
2. 11:45 am - Roger Abelenda (co-chair), **Skywalking Copilot: A performance analysis assistant**

Lunch 12:25 (95 min)

3. 2:00 pm - Ritesh Shukla, Tanvi Penumudy, **Overview of tools, techniques and tips - Scaling Ozone performance to max out CPU, Network and Disk**
4. 2:50 pm - Shawn McKinney, **Load testing with Apache JMeter**

Coffee Break 3:30 pm (30 min)

5. 4:00 pm - Chaokun Yang, **Introduction to Apache Fury Serialization**

   First Apache Incubator talk in the track!

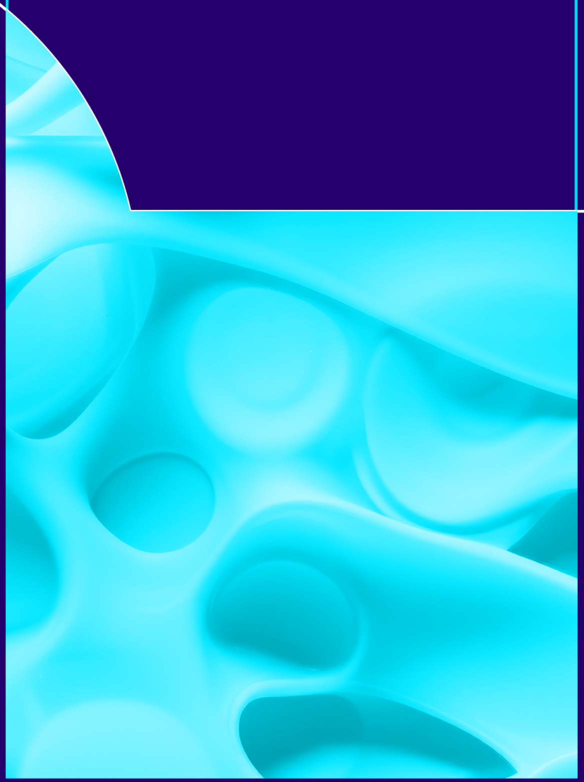
6. Your talk here next year ☺ (we lost a talk at the last minute due to visa issues)

Some other performance related topics…

- **Mon 4:50**
  - The Nuts and Bolts of Kafka Streams: An Architectural Deep Dive
- **Tue 2:50 pm**
  - Intelligent Utilization Aware Autoscaling for Impala Virtual Compute Clusters
  - Chasing for internode latency in C* 4.x
- **Wed 2:00 pm**
  - Scaling Solr: From Desktop to Cloud Scale
- **Wed 4:00 pm**
  - A Case Study in API Cost of Running Analytics in the Cloud at Scale with an Open-Source Data Stack
- **Wed 4:50 pm**
  - Unlocking sub second query performance on Lakehouse: Integrating Apache Druid with Apache Iceberg
- **Thu 11:45 am**
  - Optimizing Apache HoraeDB for High-Cardinality Metrics at AntGroup
  - Optimizing Analytic Workloads in Apple with Iceberg and Storage Partition Join

NetApp Instaclustr

MAKING APACHE KAFKA® EVEN FASTER AND MORE SCALABLE



Paul Brebner
Instaclustr Technology Evangelist

Community over Code Denver, October 7, 2024

© 2024 NetApp, Inc. All rights reserved.

This talk! Apache Kafka performance

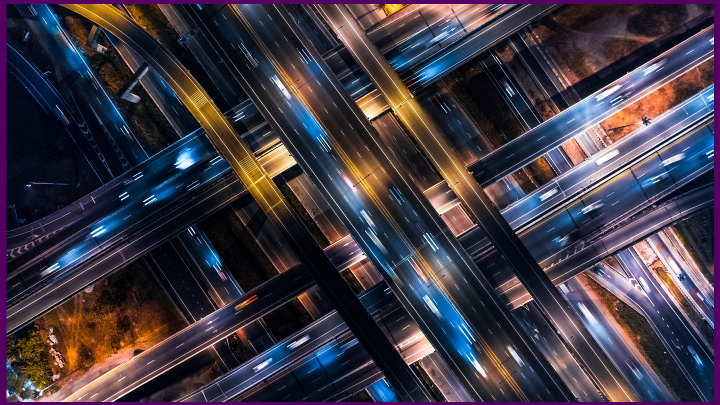
- What is the performance impact of two major architectural changes to Apache Kafka?
  - ZooKeeper → KRaft
  - Tiered storage
- Revisited max partitions experiments from New Orleans talk with current version of Kafka + KRaft
  - What breaks and when?
  - Is 1 Million partitions practical or not?
- And inspired by some of our internal Kafka benchmarking and data
  - Is workload latency impacted by KRaft?
    - Our internal testing suggested workload latency with KRaft was *faster* than ZooKeeper
    - But in theory it should be identical – what's going on?
  - Kafka tiering is here – *revenge of the n-tier architecture!?*
    - How does it work?
    - How can you test it?
    - What are the performance trade-offs?
  - Kafka clusters and Zipf's law – highlights from Bratislava talk
  - Observations
    - Tricky to reliably benchmark and understand results of Kafka at scale

Darth Sidious/Emperor Palpatine - Revenge of the Sith
(Source: Wikipedia/Star Wars, CC 2.0)

 © 2024 NetApp, Inc. All rights reserved.

PART 1

Kafka scalability and partitions

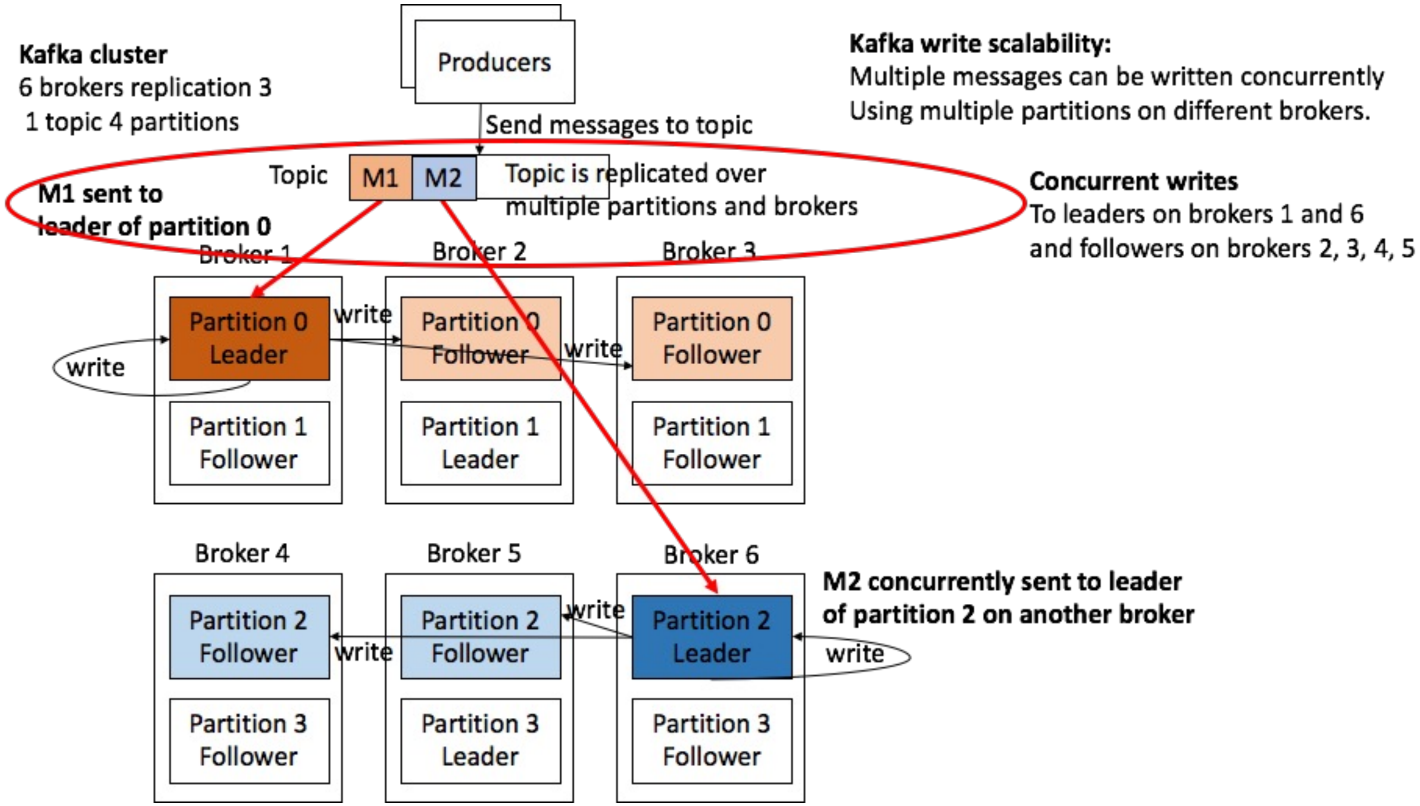


(Source: Shutterstock)

 © 2024 NetApp, Inc. All rights reserved.

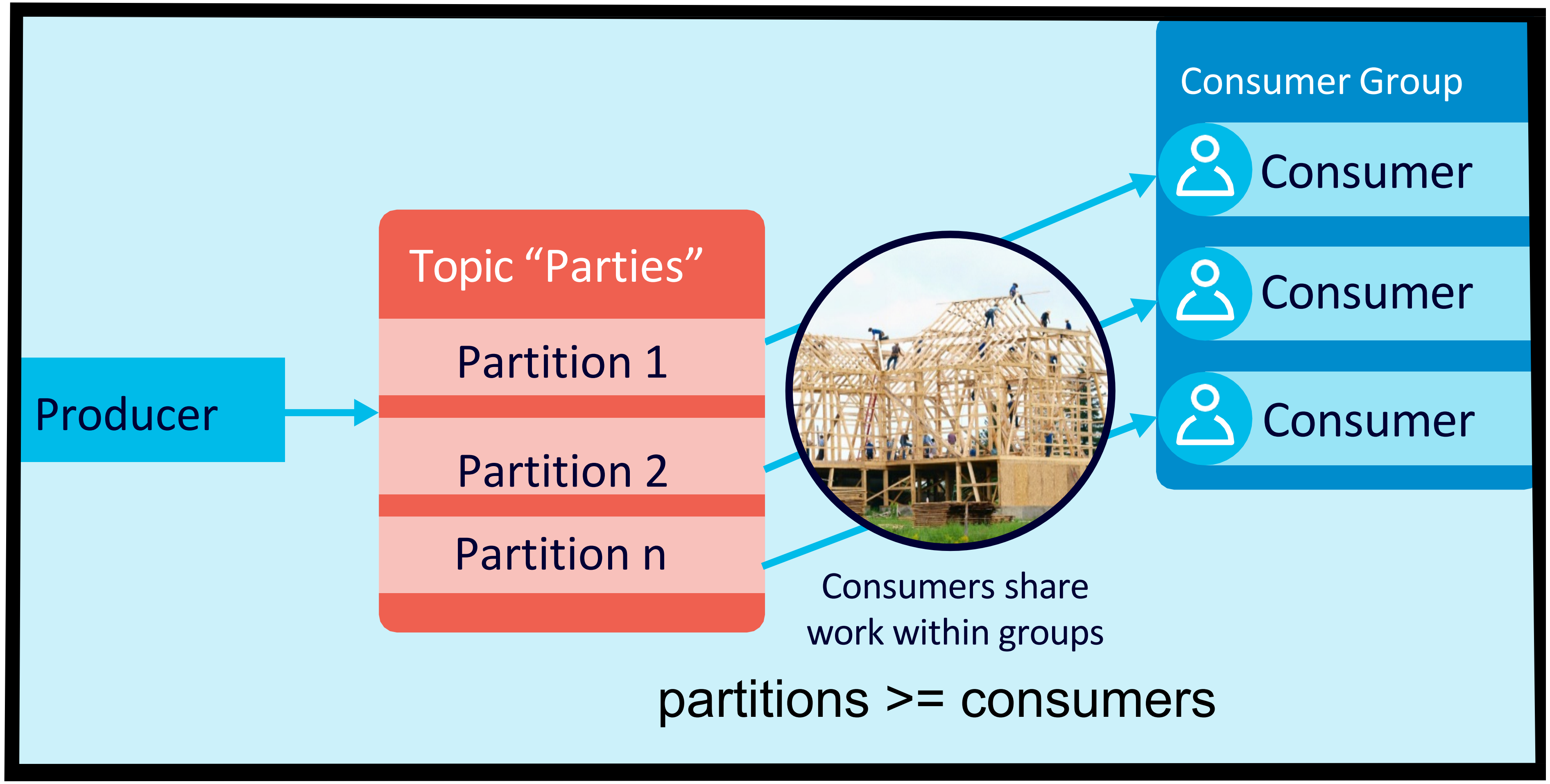
Why do we care about Kafka partitions?

- Topics are divided into partitions which enable concurrency/redundancy
  - Broker and Consumer side



 © 2024 NetApp, Inc. All rights reserved.

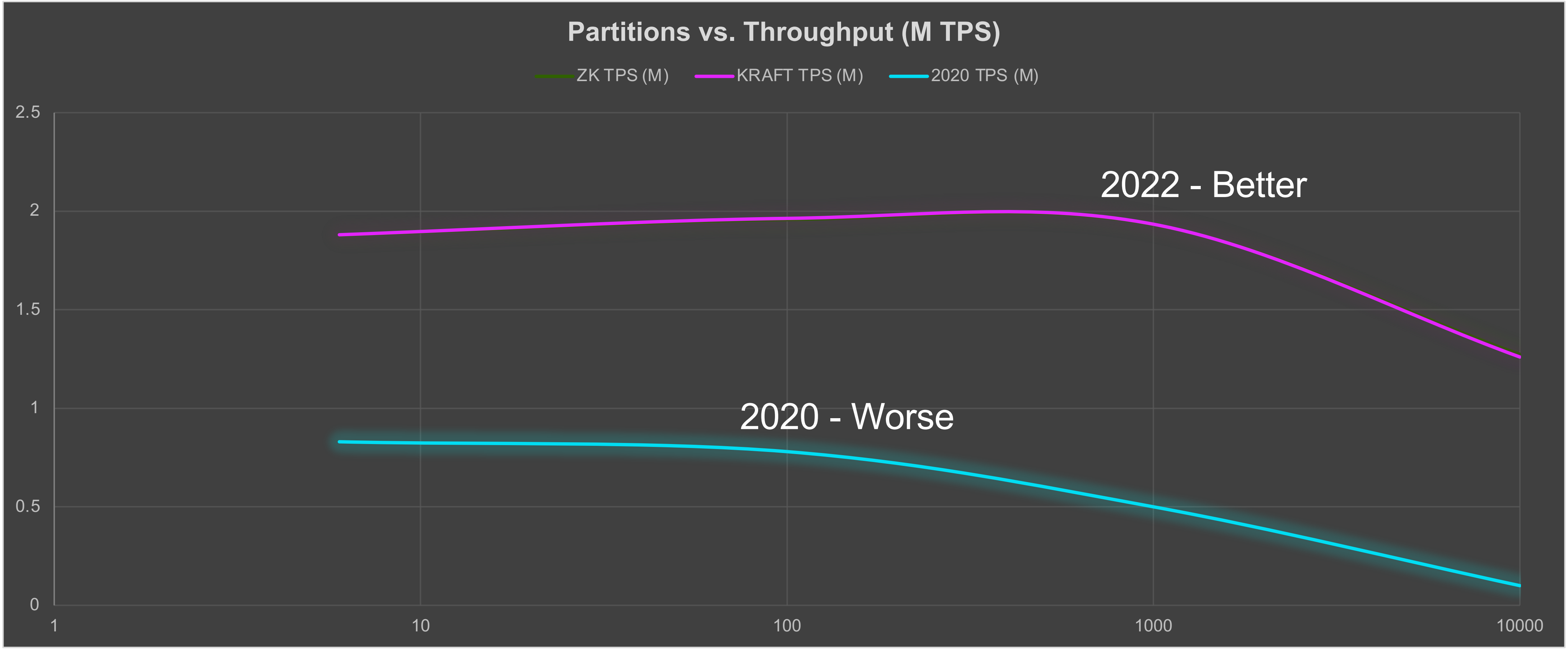
Partitions enable consumers to share work in a consumer group



 © 2024 NetApp, Inc. All rights reserved.

Partitions – concurrency mechanism – more is better – until it's not

2022 results better due to improvements to Kafka and h/w



 © 2024 NetApp, Inc. All rights reserved.

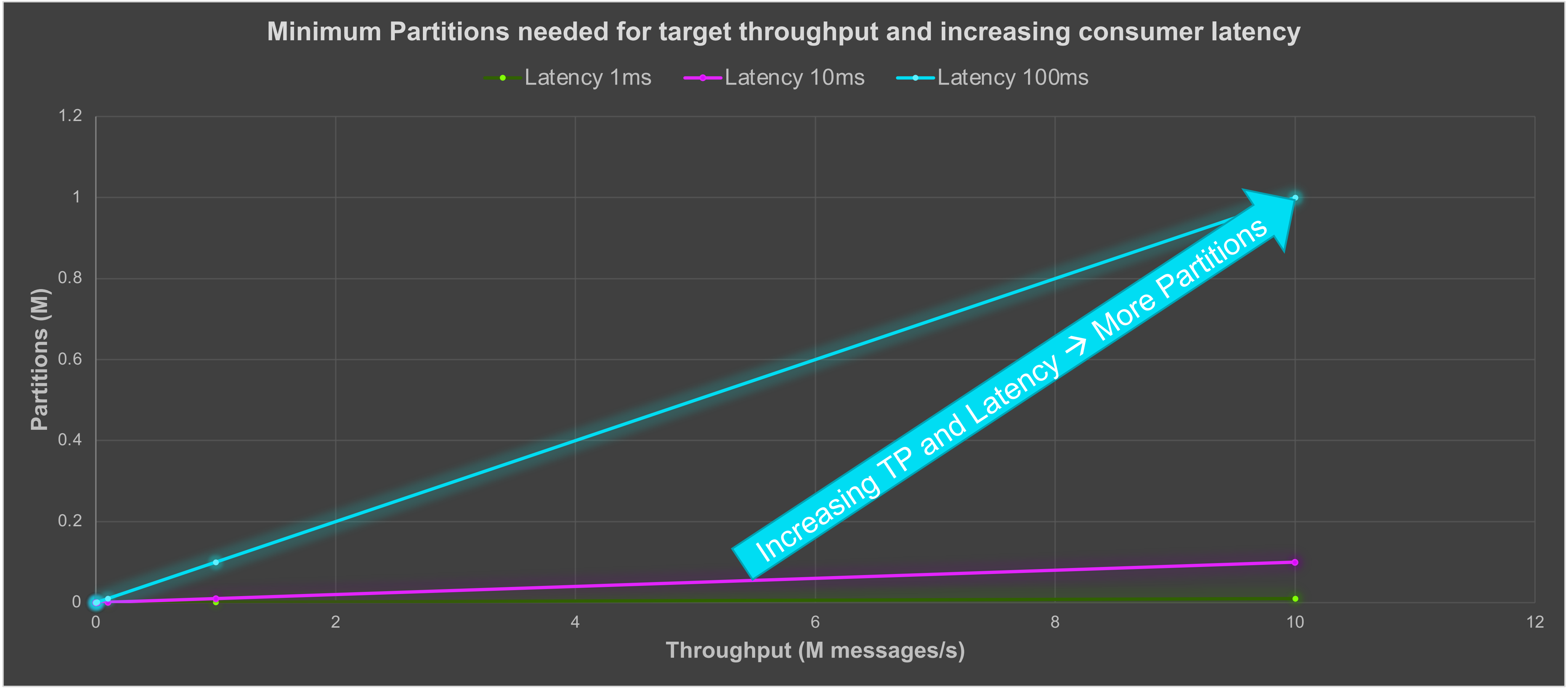
Why do we care about Kafka partitions?

- Little's Law
  - Concurrency = Throughput x Time, rearranged Throughput = Concurrency/Time
  - Default Kafka consumers are single-threaded and require >= 1 partitions
    - Max consumers <= partitions
  - For higher throughput need to maximize concurrency/consumers and/or reduce time
    - Slow-consumer problem → more consumers/partitions

Slow consumers are a problem
(Source: Getty Images)

 © 2024 NetApp, Inc. All rights reserved.

Little's Law: Partitions = TP x RT | RT is Kafka consumer latency



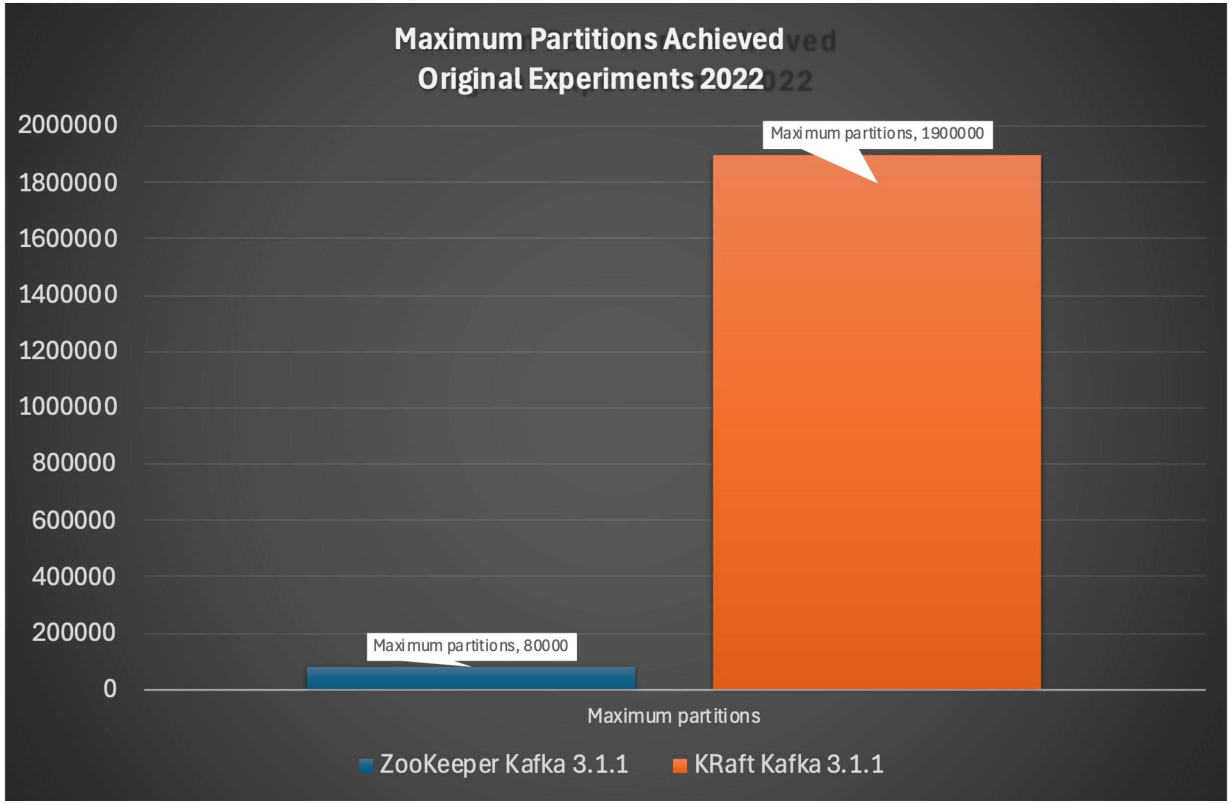
 © 2024 NetApp, Inc. All rights reserved.

Why do we care about Kafka partitions?

- Producer writes messages to topic partition either
  - Manually specified partition or automatically determined partition (based on hash function of Key)
- Order is only guaranteed within partitions
- You may need lots of topics (e.g. for fine-grained message delivery)
  - >= 1 partition per topics, which also means lots of partitions
- So, for some use cases you may need lots of partitions
- Creating lots of partitions is feasible with the replacement of ZooKeeper by KRaft
  - KRaft handles meta-data operations including partitions creation
  - Now very easy (too easy?!) to create lots of partitions very fast
- Increasing partitions is a de-facto load test for Kafka
  - Without any producers/consumers/messages required
  - Partition replication consumes Kafka cluster CPU resources

 © 2024 NetApp, Inc. All rights reserved.

Previous results (2022)
ZooKeeper 80,000 partitions
KRaft 1.9M partitions



 © 2024 NetApp, Inc. All rights reserved.

Maximum partitions experiment - 1

(Source: Adobe Stock)

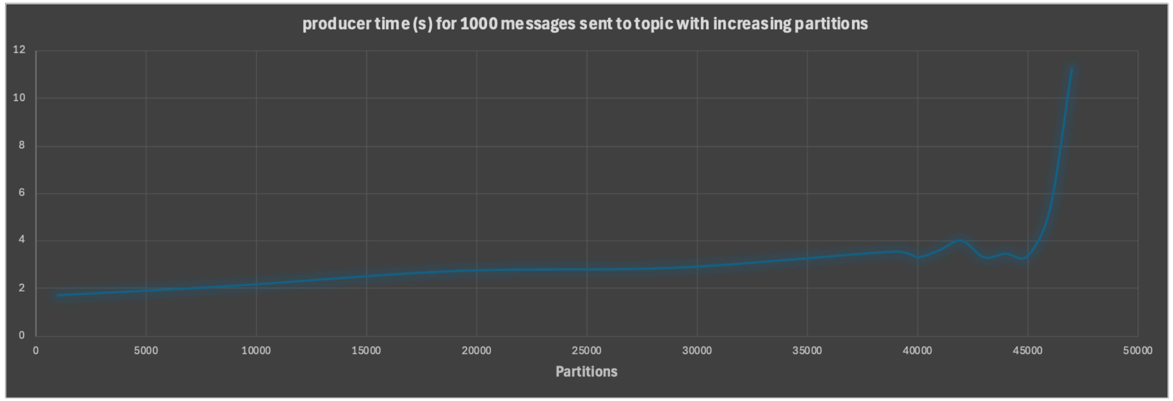
- Original results
  - Kafka 3.1.1/3.2.1
  - 1.9M partitions
  - But tricky to do
- Second attempt
  - Kafka 3.6.1 (KRaft GA, batch size bug fixed, KIP-868) with 12 brokers (4 cores each) + 3 controllers = 48+
  - Increased mmap and max files to lots (millions)
  - Methodology – bash script to incrementally increase partitions on single topic
  - Monitor CPU and partitions etc
  - Total partitions = partitions x RF (3)
  - Achieved 1M partitions BUT
    - Our Kafka console metrics failed so can't see what's going on
    - Kafka CLI topic describe failed (out of memory) so can't confirm total partitions
    - Kept going until **2.25M** partitions, CPU 90%, Controllers CPU spiked
    - But does it work? No. Send test message, Kafka CLI producer failed (timeout).
    - What's wrong? Kafka cluster brokers and controllers saturated, and controller logs have "error" messages

 © 2024 NetApp, Inc. All rights reserved.

Maximum partitions experiment - 2

(Source: Adobe Stock)

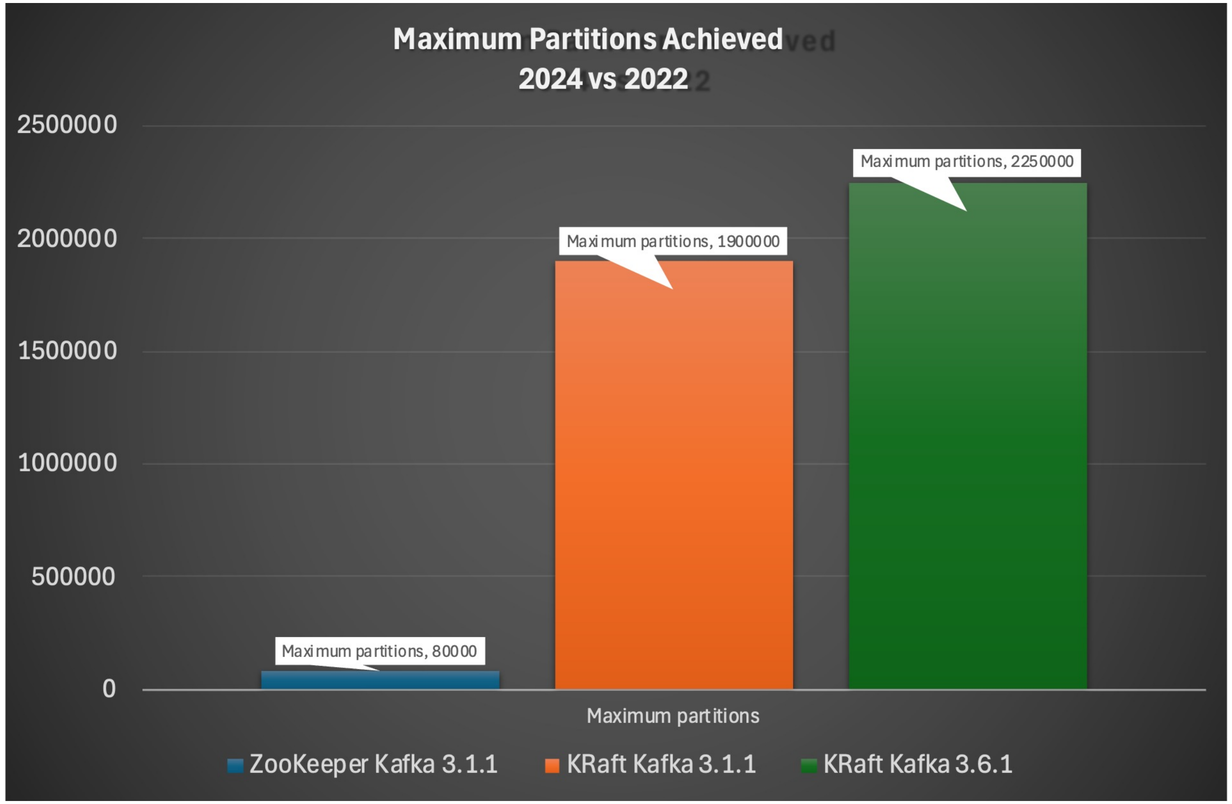
- Tried again this time to determine when producer fails
  - Send (and time) message to two topics, baseline topic with 3 partitions only, and “lots” topic with increasing partitions
- CLI producer fails around 48,000 partitions
  - Due to a timeout error getting topic meta-data



- But still works with Java producer!
  - Unsure what the difference is and didn’t push it any higher
  - Also, slower than expected
- Conclusions?
  - Ensure that cluster max files and mmap are >> partitions x 2
  - With RF=3 CPU increases with increasing partitions (this was expected as replication uses resources)
  - Tricky to benchmark large partitions when producer (CLI) and metrics fail as well
  - To produce/consume messages to clusters with large numbers of partitions you need a larger cluster again
    - E.g. one of our largest Kafka clusters has 500k partitions and 100 brokers (8 cores) = 800 cores (i.e. 16 x bigger!)

 © 2024 NetApp, Inc. All rights reserved.

Summary graph of results



 © 2024 NetApp, Inc. All rights reserved.

Hypotheses from original talk

What	ZooKeeper	KRaft	Results
Workload data operations	FAST	FAST	Identical Confirmed Really???
Meta-data changes	SLOW	FAST	Confirmed
Maximum Partitions	LESS	MORE	Confirmed
Robustness	YES	WATCH OUT	OS settings and producer!

 © 2024 NetApp, Inc. All rights reserved.

Conclusions

- 2M+ partitions (including RF) is possible
  - 46,000 partitions per core
- But to create/use lots of partitions you need
  - Bigger clusters
    - At least x2 to be useable (guess, 50% CPU headroom for workload)
    - i.e. 23,000 partitions per core
  - Increased configuration settings
  - Keep track of cluster health with metrics (assuming they don't fail) and controller logs
  - Java producer was more reliable but may need configuration tuning

(Source: Adobe Stock)

 © 2024 NetApp, Inc. All rights reserved.

PART 2

Kafka KRaft workload latency

Paddle faster!
(Source: Shutterstock)

© 2024 NetApp, Inc. All rights reserved.

Next, workload latency hypothesis revisited

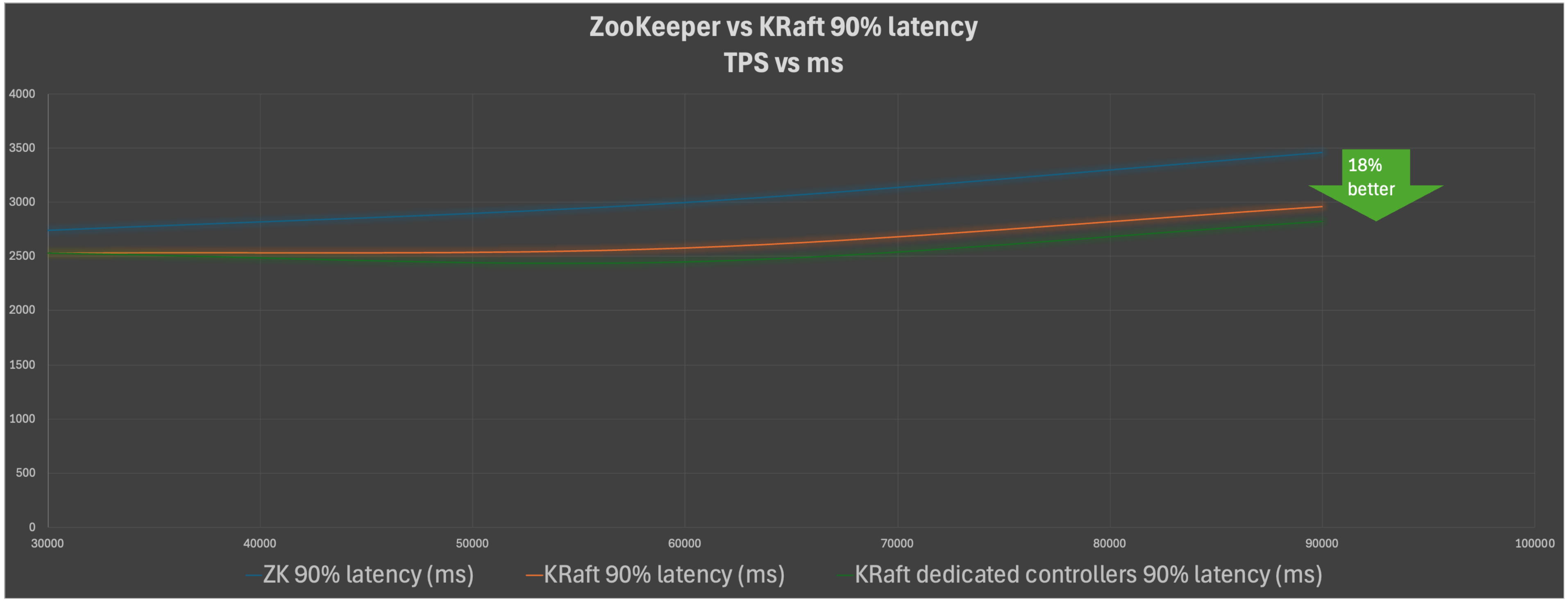
- My original New Orleans 2022 tests indicated there was no difference in throughput for ZK vs. KRaft
- But recent internal tests showed KRaft workload latency maybe slightly faster than ZooKeeper
- In theory there should be no difference – as KRaft/meta-data isn't involved in the workload write/read paths
- What's going on?
- Redid our internal tests…

 © 2024 NetApp, Inc. All rights reserved.

Internal results show 90th percentile latency is 18% faster with KRaft

A few “odd” things! 95% CPU so clusters overloaded, latency is high (2.5s+)!

But what about lower loads or 50% latencies?



 © 2024 NetApp, Inc. All rights reserved.

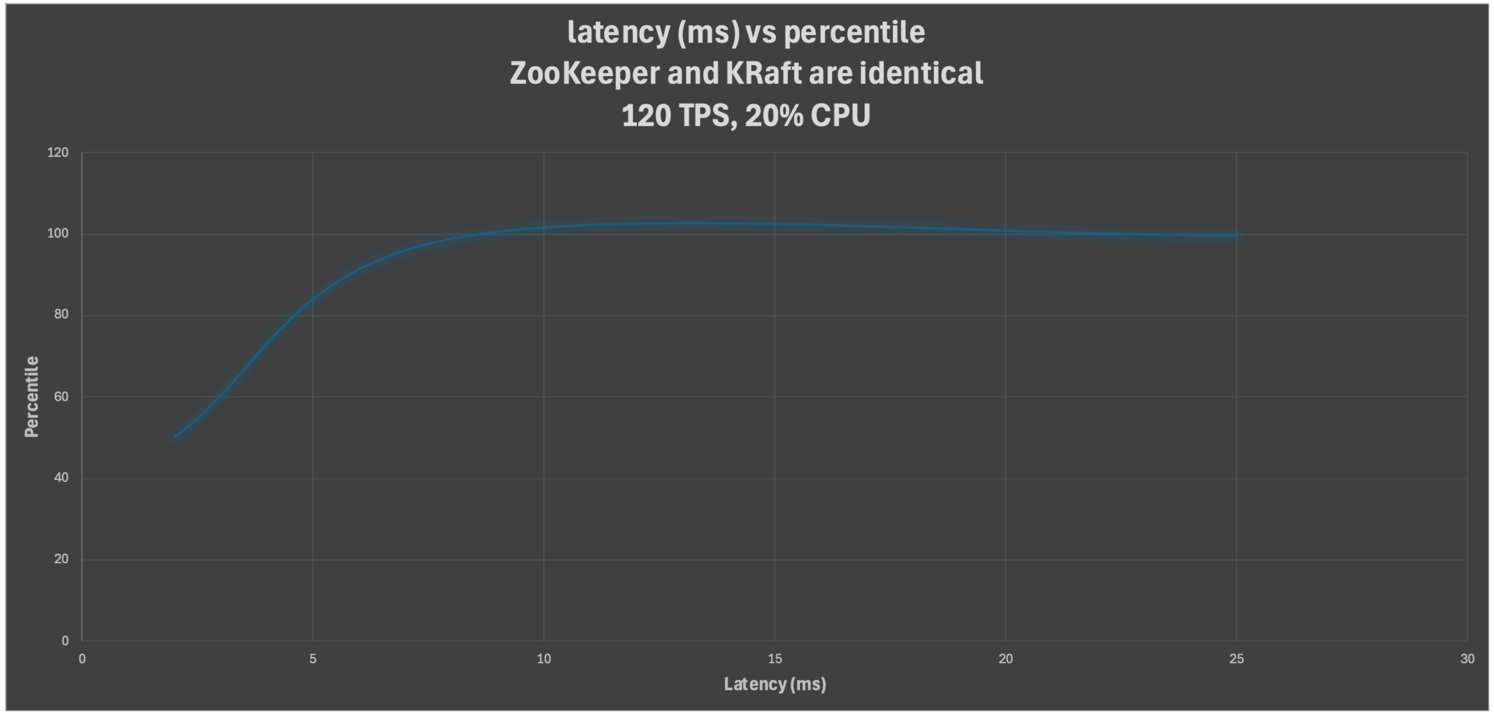
My results - 1

50 to 99.9 percentiles, constant load 120TPS 20% CPU

ZooKeeper and KRaft end-to-end latencies are identical and fast (50% < 2ms, 99% < 8ms, 99.9% < 25ms)

More in line with what I expect from lightly-loaded clusters

Faster than internal benchmark latencies (designed for measuring maximum capacity)



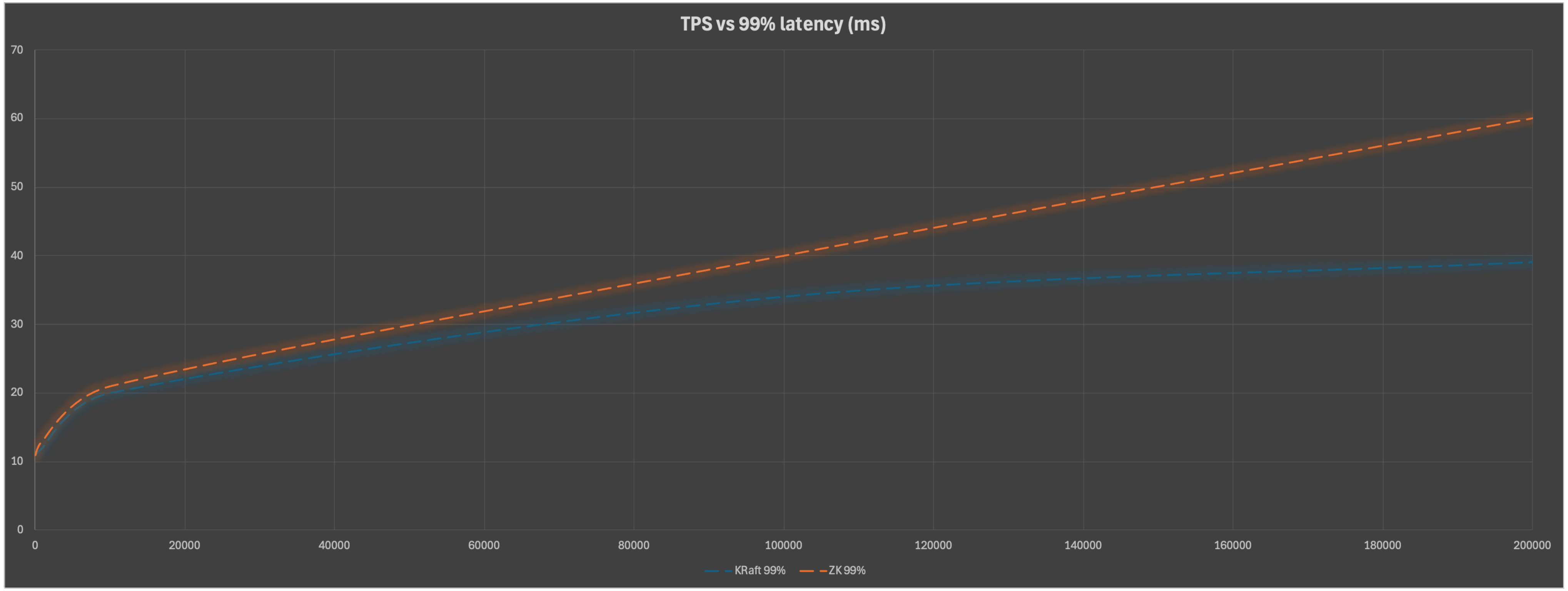
 © 2024 NetApp, Inc. All rights reserved.

My results - 2

Latency under increasing load (max 50% CPU)

No difference for 50% but some improvement for 99%

This confirms the internal benchmark results

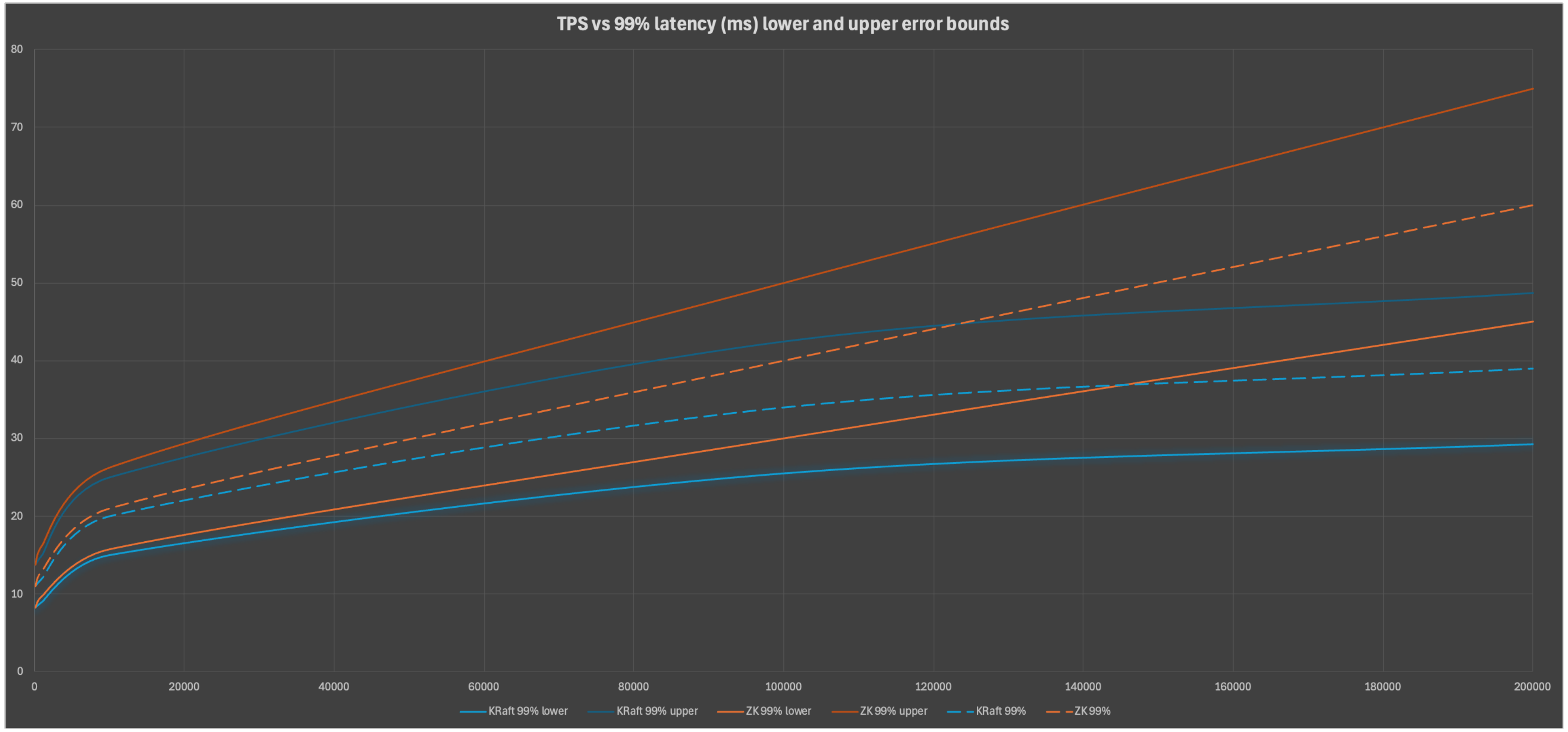


 © 2024 NetApp, Inc. All rights reserved.

My results - 2

But note that higher percentiles like 99th percentile aren't statistically robust

Graph with lower and upper error bounds - overlap

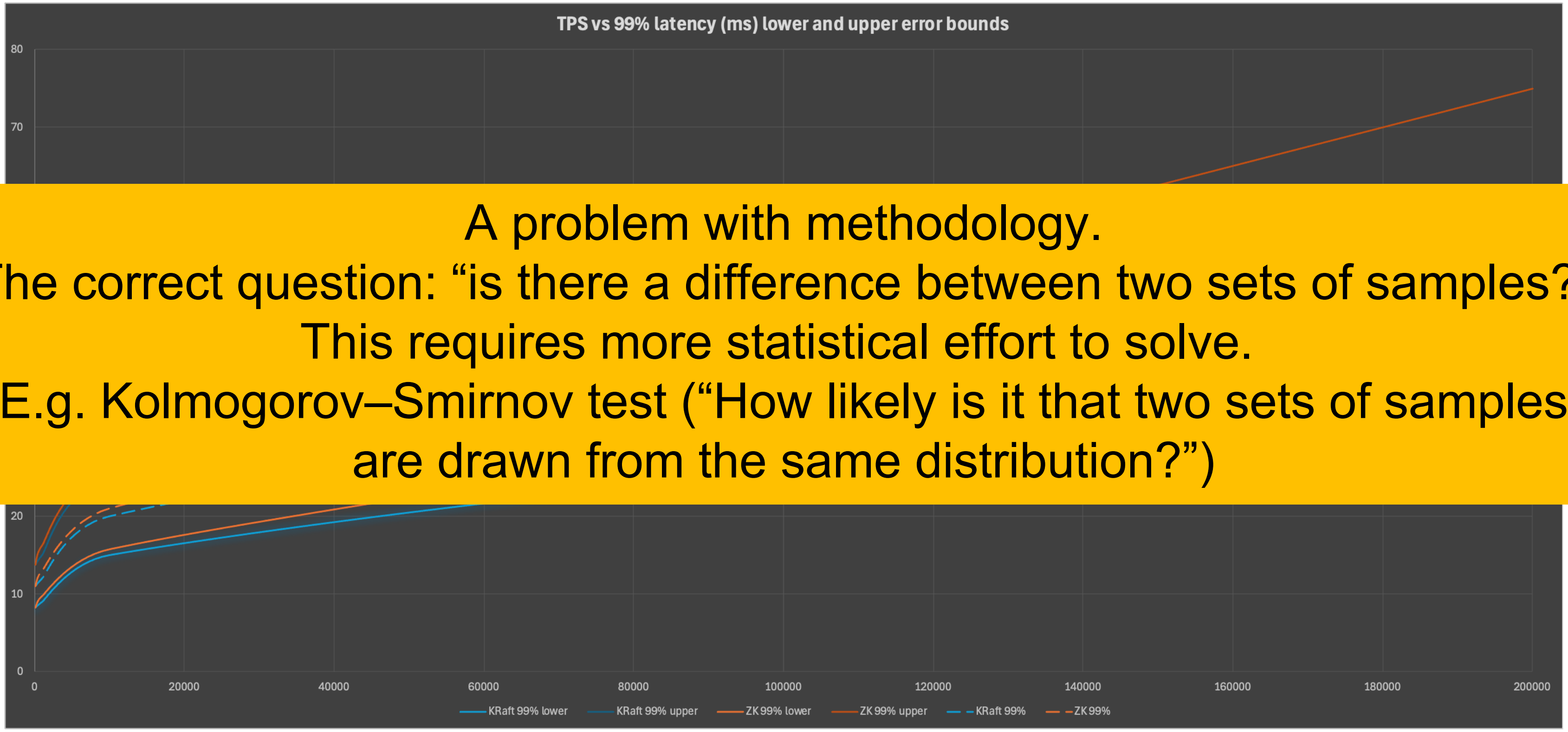


 © 2024 NetApp, Inc. All rights reserved.

My results - 2

But note that higher percentiles like 99th percentile aren’t statistically robust

Graph with lower and upper error bounds - overlap



A problem with methodology.
The correct question: “is there a difference between two sets of samples?”
This requires more statistical effort to solve.
E.g. Kolmogorov–Smirnov test (“How likely is it that two sets of samples are drawn from the same distribution?”)

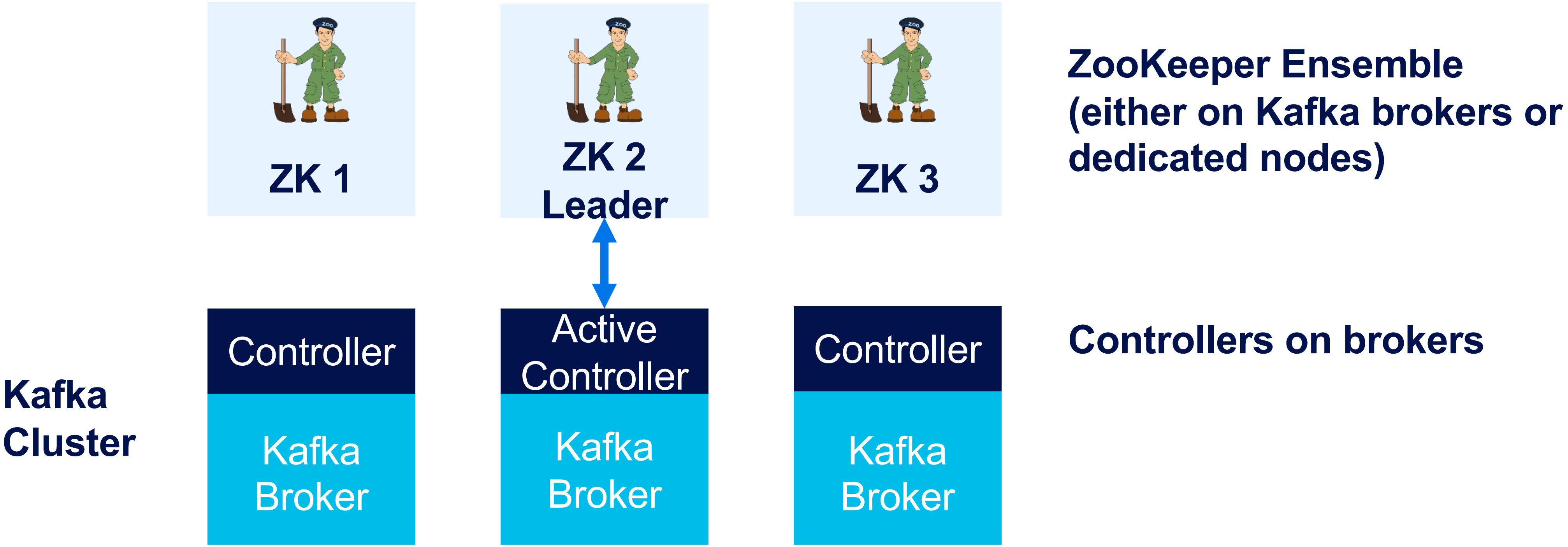
 © 2024 NetApp, Inc. All rights reserved.

Theory – more resources are available for the data workloads

- If there is a difference, why?
- Previously ZooKeeper and KRaft could be run on dedicated nodes
  - But not the Kafka controllers which shared the Kafka brokers
- But now the recommended way to run Kafka KRaft in production is with dedicated Controller nodes
  - Which allows a slightly higher headroom for Kafka on the brokers
  - And may explain the reduced higher percentile latencies under load?

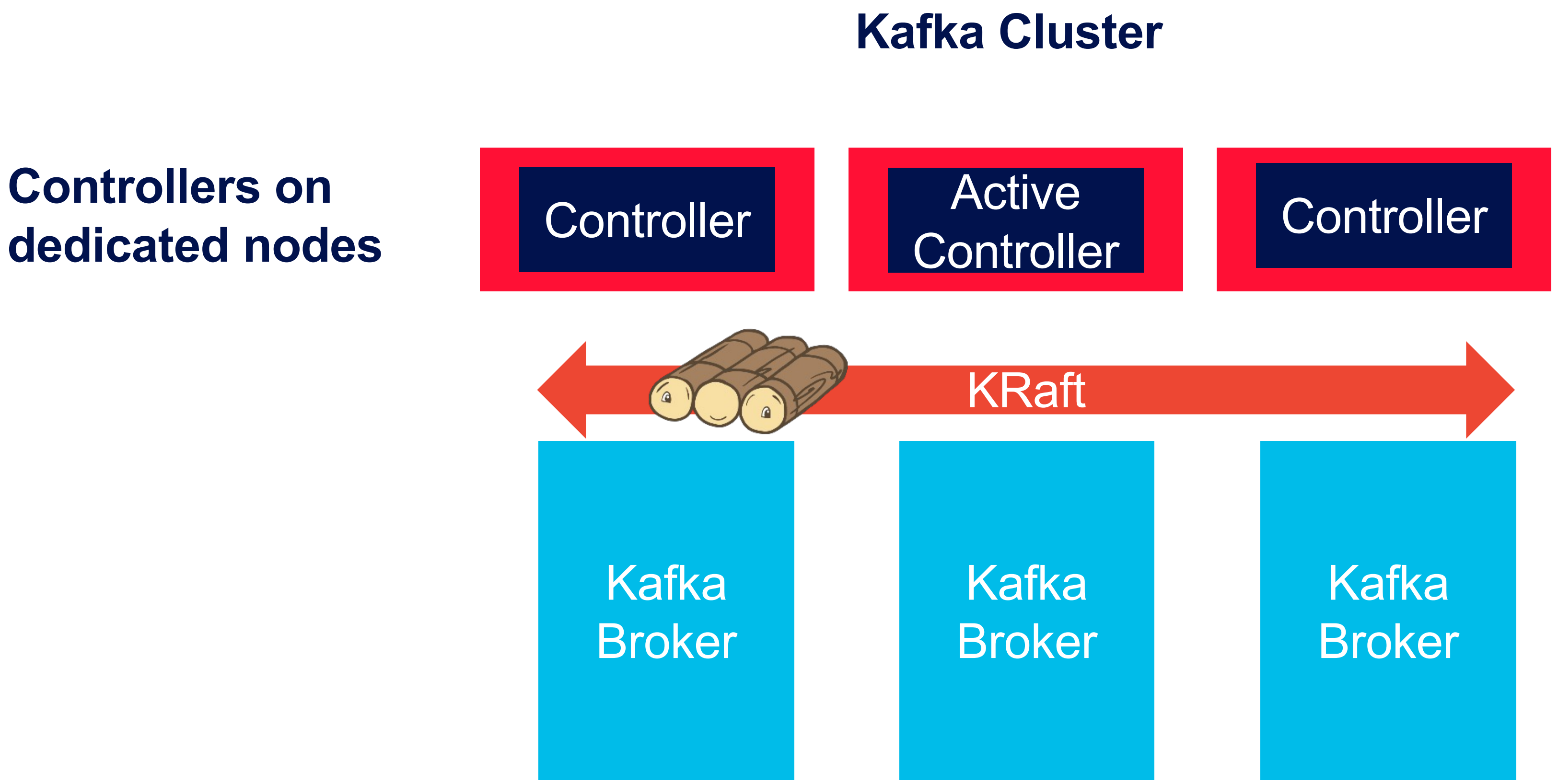
 © 2024 NetApp, Inc. All rights reserved.

Original Kafka + ZooKeeper



 © 2024 NetApp, Inc. All rights reserved.

Kafka KRaft cluster with dedicated Controller nodes



 © 2024 NetApp, Inc. All rights reserved.

PART 3

Is Kafka tiered-storage more like a fountain or a dam?

Latona Fountain
(Source: Wikimedia)

Hoover Dam
(Source: Paul Brebner)

 © 2024 NetApp, Inc. All rights reserved.

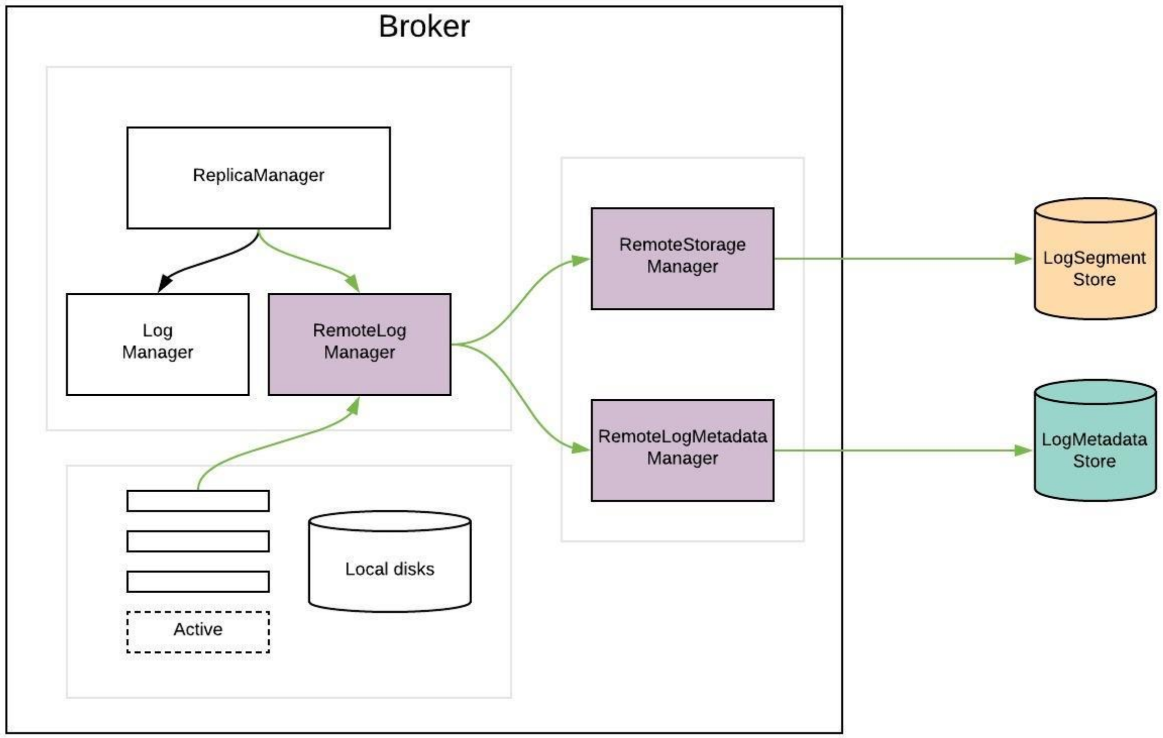
Kafka tiered storage: Hot & cold data

- KIP-405
  - 2020 – 2024+
  - Early access of tiered storage 3.6.1 (2023) - limitations
  - 3.8.0 (2024) Kafka Tiered Storage V1 (15420) – some fixes but still early access
- Motivations
  - Quantity of data and cost – can store more data (essentially unbounded) and for longer for less $
  - Operational performance
    - broker maintenance operations are faster if local data is reduced
    - e.g. broker replacement in minutes not hours (130x faster)
    - reduces the time that workload latencies can be elevated due to cluster maintenance events
  - Scalability/elasticity – can scale compute resources (brokers) and storage independently
- Major architectural change
  - From homogeneous cluster architecture with data only stored on local disk (x RF)
  - To heterogeneous cluster
    - 2-tiers – compute+storage and storage only
    - Kafka brokers still have local storage but
    - Data optionally also stored on alternative remote object storage – can be read from remote storage if no longer local
    - No changes to client code

(Source: Adobe Stock)

 © 2024 NetApp, Inc. All rights reserved.

High level design (from KIP-405)



- Requires a plugin for each remote storage type (RemoteStorageManager)
- Changes so Kafka writes data to remote storage, maintains remote metadata, changes to follower replication, reads from local or remote, handles remote deletion, and has new configurations and metrics, etc.
- And handle errors slow/unavailable remote storage (e.g. dedicated thread pools)

 © 2024 NetApp, Inc. All rights reserved.

What's "interesting"?

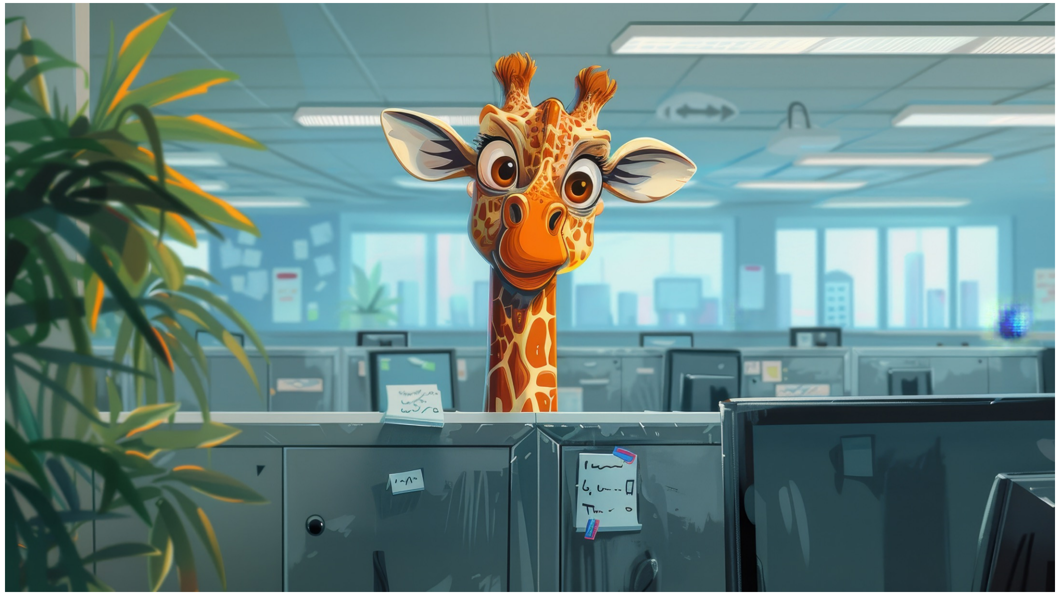
- Turns out I didn't know much about Kafka storage
  - Behind the partitions are segments
- Is tiered-storage more like a tiered fountain or a dam?
  - Many online documents are ambiguous/wrong about how/when data is written to remote storage
    - i.e. does data eventually cascade from one tier to the next or is it just a dam/write-through cache (more or less?)
- Performance testing was tricky
  - Why? Because I didn't understand how tiering worked, the impact of various settings, etc.
    - But basically – how do you know if the data is being written to remote storage? Read locally or from remote storage?
  - Is there any impact on write or read performance? Does it matter?

A 2-tiered fountain (Ganymede's fountain)
in Bratislava, Slovakia
(Source: Paul Brebner)

 © 2024 NetApp, Inc. All rights reserved.

What’s behind the partitions?

- Append-only file system logs
  - Fast and supports replaying
- Records are written to disk segments
  - Only one active/current segment (per topic/partition)
  - When full (time or space) it is closed/rolled
  - New segment started
  - Segments eventually deleted

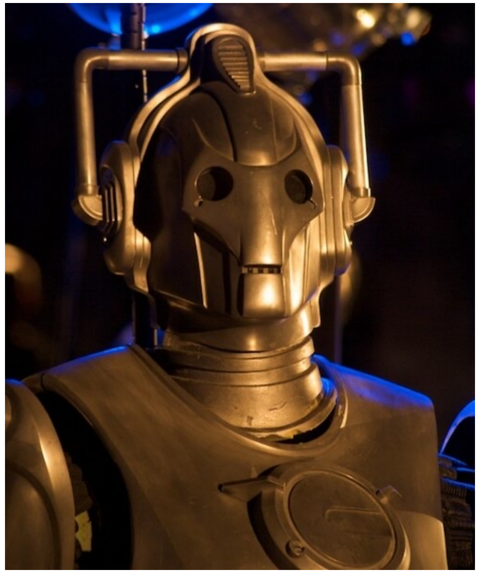


(Source: Adobe Stock)

 © 2024 NetApp, Inc. All rights reserved.

Delete delete delete delete!

- Do records last forever?
  - No – deleted after retention period (7 days default)
  - Or space
- Until deletion, records are available for reading by consumers
  - Once deleted there's an exception and consumers can't read them

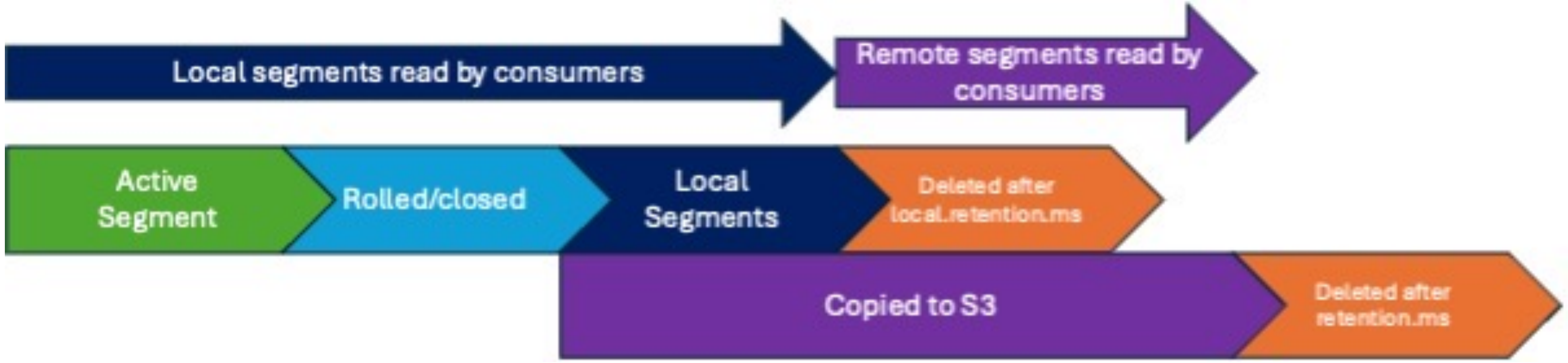


Dr Who Cybermen
(Source: Wikimedia)

 © 2024 NetApp, Inc. All rights reserved.

Tiered-storage

- Once enabled for a cluster tiered-storage can be enabled per topic
  - For Instaclustr Kafka, must provide an AWS S3 bucket which is used as the remote storage
- New configurations for local storage
  - Local.retention.bytes and local.retention.ms
  - Original configurations are now for remote
  - Want local time/space << remote

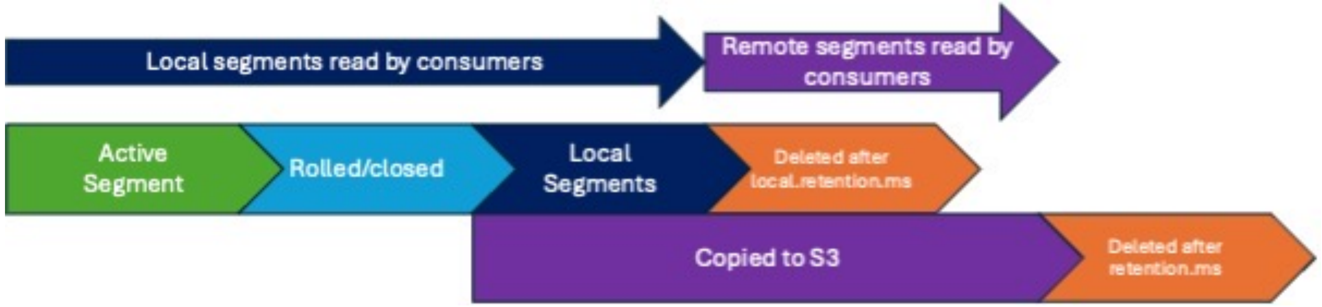


 © 2024 NetApp, Inc. All rights reserved.

When are remote segments written?

A 2-tiered fountain (Ganymede's fountain) in Bratislava, Slovakia
(Source: Paul Brebner)

- It took ages for the 1st remote file to appear in S3 – why?
- I incorrectly assumed (lots of docs imply this) that
  - Remote segments are only written once the local retention is reached (i.e. tiered fountain model)
  - This is incorrect
  - Local segments are eligible for copying to remote storage *once they are closed* but
    - It can take some time for them to be copied as asynchronous, and
    - Kafka remote storage component both uses and has limited (tunable) resources, and
    - Defaults for segment closing are 1GB (segment.bytes) and 7 days (log.roll.hours) so can take some time (particularly with low TP and many partitions)
- And remote segment deletion is potentially slow
  - Lots of settings and uses Kafka resources
  - once eligible for deletion they are
    - eventually removed



 © 2024 NetApp, Inc. All rights reserved.

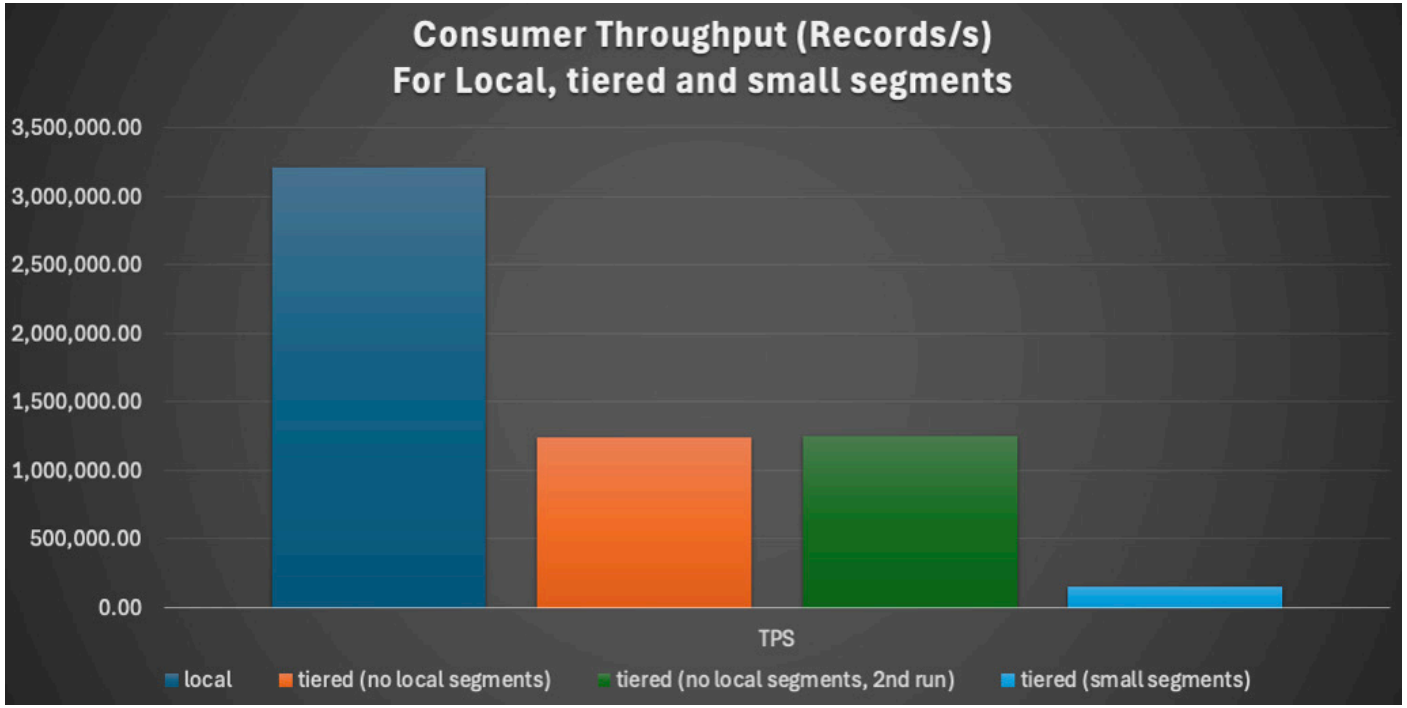
Performance impact?

- Kafka homogeneous architecture with fast (e.g. SSDs) local storage is known for
  - low latency and high throughput performance
- Does tiered storage have any performance impacts?
- And how to test?
  - Tricky to ensure that the consumer is really reading from remote storage
  - Method – create local only and remote mostly topics (this is a bit artificial c.f. production)
    - For remote
      - Create topic with tiering enabled, 6 partitions, and very short local.retention.ms time limit
        - Also tried smaller segment size
      - Write lots of data until at least 10GB of segments in S3
      - Wait for 30m
      - Run the consumer from offset 0 and record the performance
    - Our internal tests used a EBS-backed Kafka cluster, mine used SSD local storage (faster)

 © 2024 NetApp, Inc. All rights reserved.

Throughput

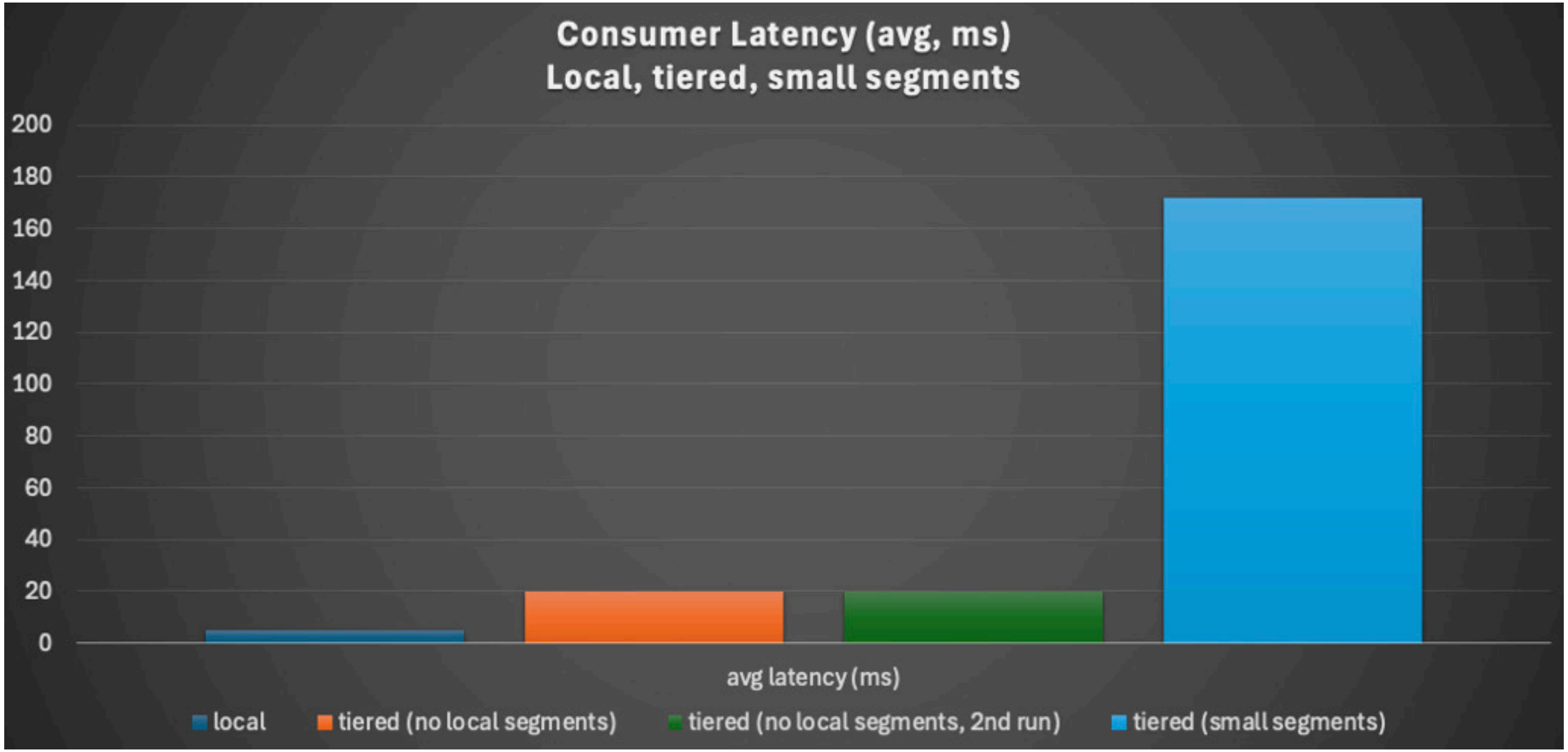
- Local is 2.5 times higher TP than remote
- Small segments perform worse
- No difference between
  - 1^{st} and 2^{nd} reads
  - Broker caching not used?
  - Maybe room for improvement



 © 2024 NetApp, Inc. All rights reserved.

Latency

- Same story



 © 2024 NetApp, Inc. All rights reserved.

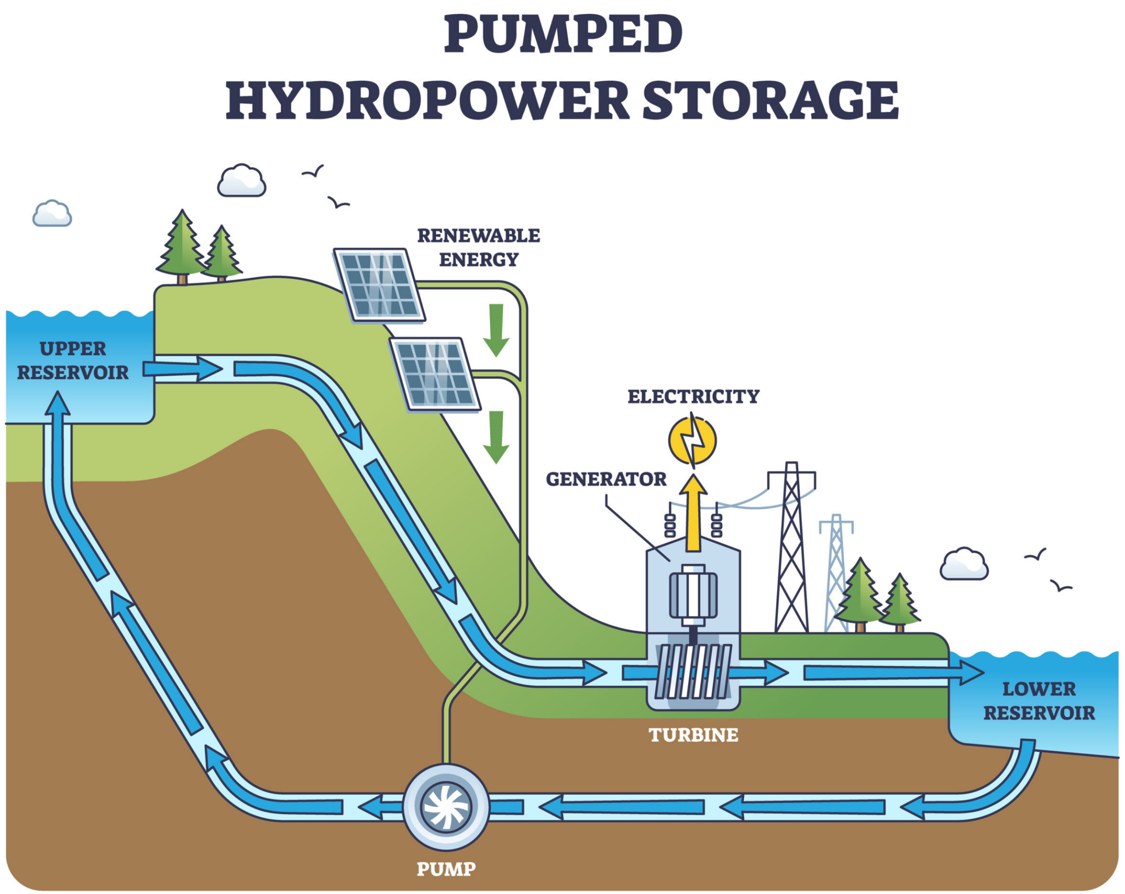
Producer/write impact?

- No measurable impact on latency or throughput as copying to remote storage is asynchronous
- BUT copying uses more broker resources, 10%+
  - reads and deletes also use Kafka resources
  - may need a bigger cluster
- Some optimizations possible, for example
  - Ratio of local vs. remote data
  - Kafka remote storage configurations/resources are tunable – I just used defaults
  - Segment size (but not too small)
  - Prefetching and cache settings? Only noticed these after experiments!
  - Speed/throughput of cloud storage
  - Due to reduced TP & increased latency from remote storage may need to increase the number of consumers and partitions

 © 2024 NetApp, Inc. All rights reserved.

Better model? Pumped hydro dam!

- Data copied as soon as possible from primary (upper reservoir) to secondary storage (lower reservoir)
- Data is available locally until it is deleted (dam spillway)
- If data isn't available locally anymore then read back from secondary storage (pump)
- Kafka tiered storage is basically a write-through cache



(Source: Adobe stock)

 © 2024 NetApp, Inc. All rights reserved.

Observations

- Due to copying strategy (asap) most of the records have
  - >> RF replication
  - i.e. local (RF) + remote storage (multiple copies)
- Does Kafka need local storage still?
  - Yes – that's how it works
  - But potentially you can reduce the size of local storage substantially
  - Depending on workload
    - If most consumers are keeping up with and reading from the last few segments locally then you don't need much local storage
    - But if some consumers can get behind you may need more local storage to ensure responsive processing in semi-real time
    - Many replaying use cases are not so time critical and should work reading from remote storage only
- Doesn't Kafka now just have the same architecture as Pulsar?
  - Independent compute and storage nodes!
  - Revenge of n-tiered architectures!

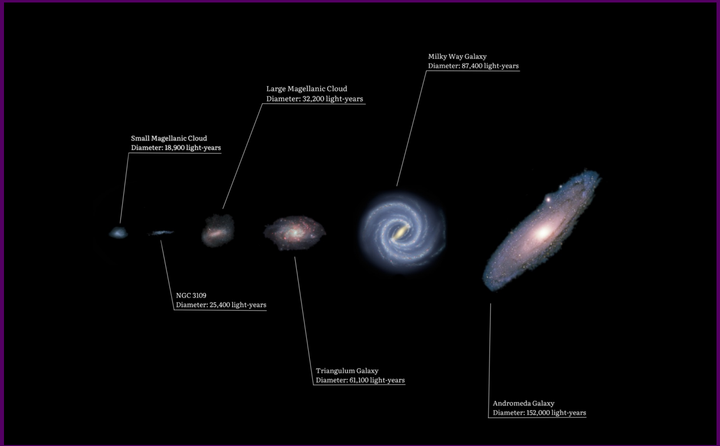
Darth Sidious/Emperor Palpatine - Revenge of the Sith
(Source: Wikipedia/Star Wars, CC 2.0)

 © 2024 NetApp, Inc. All rights reserved.

PART 4

Kafka Clusters and Zipf’s Law: size distribution

Extract of talk from C/C Bratislava 2024
"Why Apache Kafka Clusters Are Like Galaxies
(And Other Cosmic Kafka Quandaries Explored)”



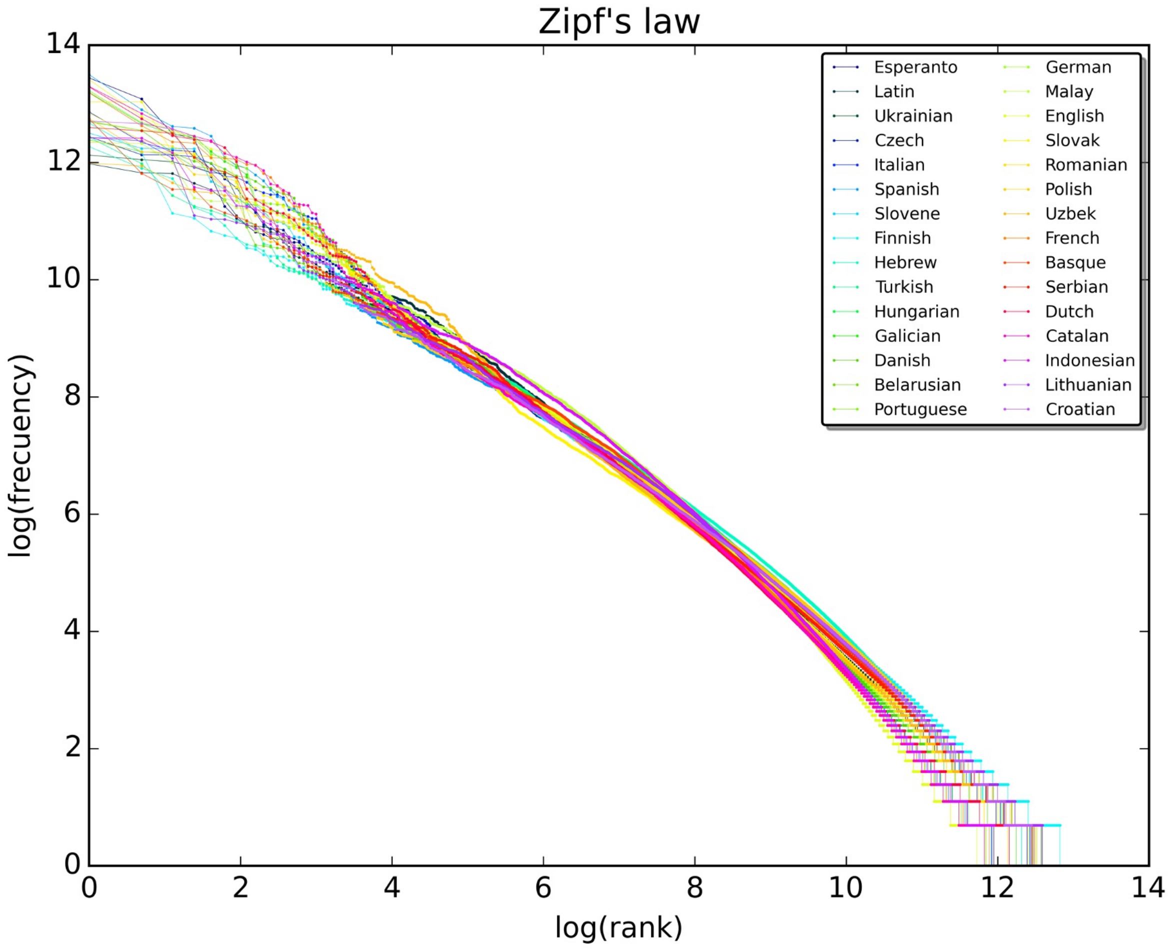
Visual size comparison of the six largest Local Group galaxies, with details (Source: Wikipedia)

 © 2024 NetApp, Inc. All rights reserved.

Zipf's Law

Scaling/power law

- Distribution function
  - Most frequent observation is twice as common
    - as next and so on (i.e. 1/rank)
    - Long-tailed distribution
    - 80/20 rule (20% of people own 80% of $)
    - C.f. Pareto (discrete vs. continuous)
  - Log-log rank vs frequency/size gives approx. straight line
  - Common examples
    - Frequency of words
    - Wealth distribution
    - Animal species size
    - Earthquakes
    - City sizes
    - Computer systems (e.g. workload modelling, subsystem capacity)
    - Galaxy sizes



 © 2024 NetApp, Inc. All rights reserved.

Apache Kafka + Galaxies?

Size and scale predictions

- Question: How large are the largest structures in the universe?
- Answer: Bigger!
- Zipf's law predicted that
  - bigger galaxies would be detected in older parts of the universe
  - beyond the reach of the Hubble at the time
  - confirmed with the James Webb telescope observations
- But what's this got to do with Kafka?

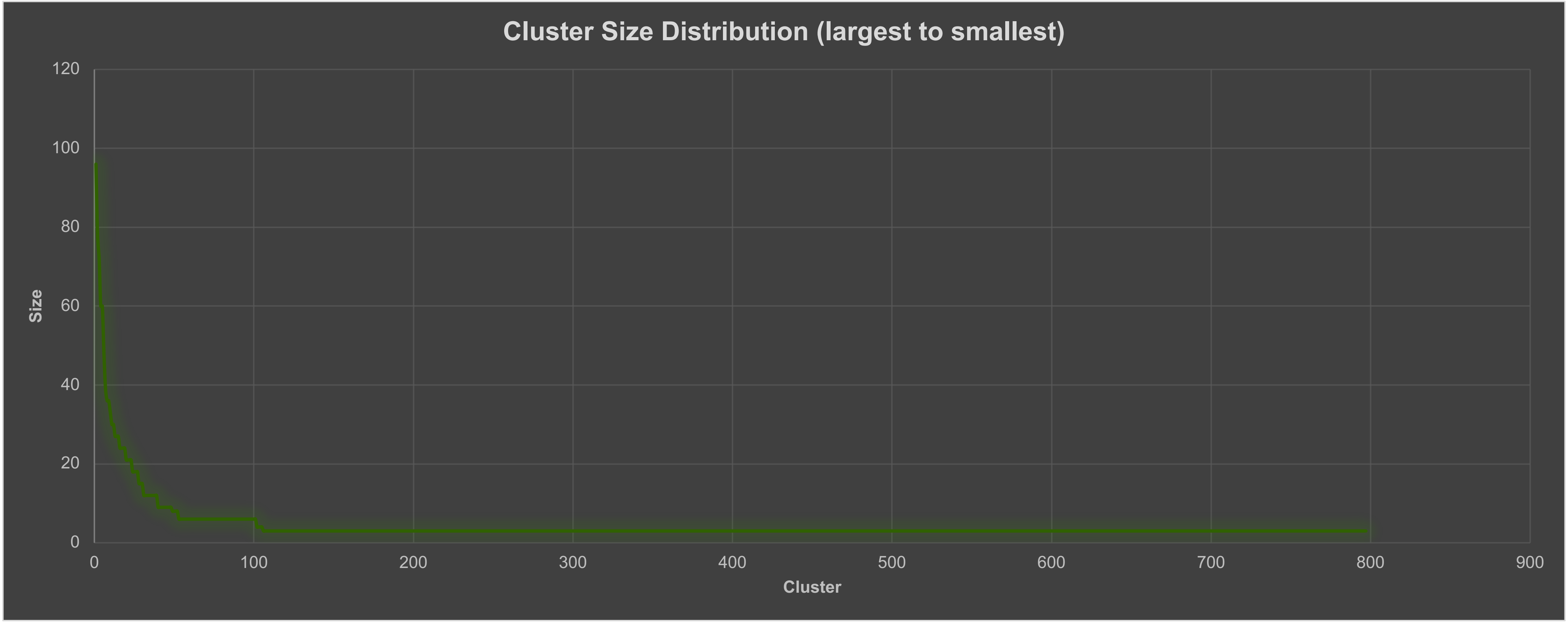
Image from NASA's James Webb Space Telescope
showing older and bigger galaxy clusters

 © 2024 NetApp, Inc. All rights reserved.

Kafka Clusters and Zipf's Law

All our managed Kafka clusters, size = number of brokers per cluster

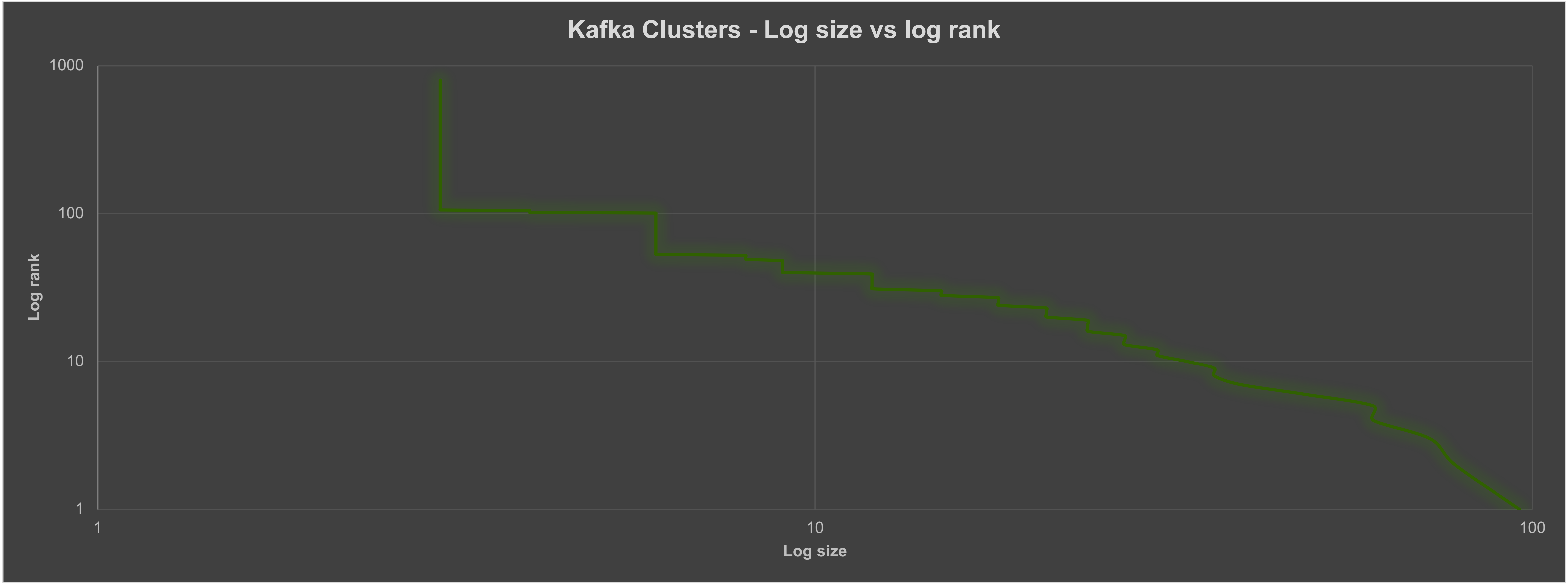
What is the distribution? Definitely a long-tailed power law



 © 2024 NetApp, Inc. All rights reserved.

Kafka Clusters – log size vs. log rank

Approximately Zipfian



 © 2024 NetApp, Inc. All rights reserved.

So What? Kafka and Zipf's Law (1)

Can expect larger clusters (animals, galaxies etc.)

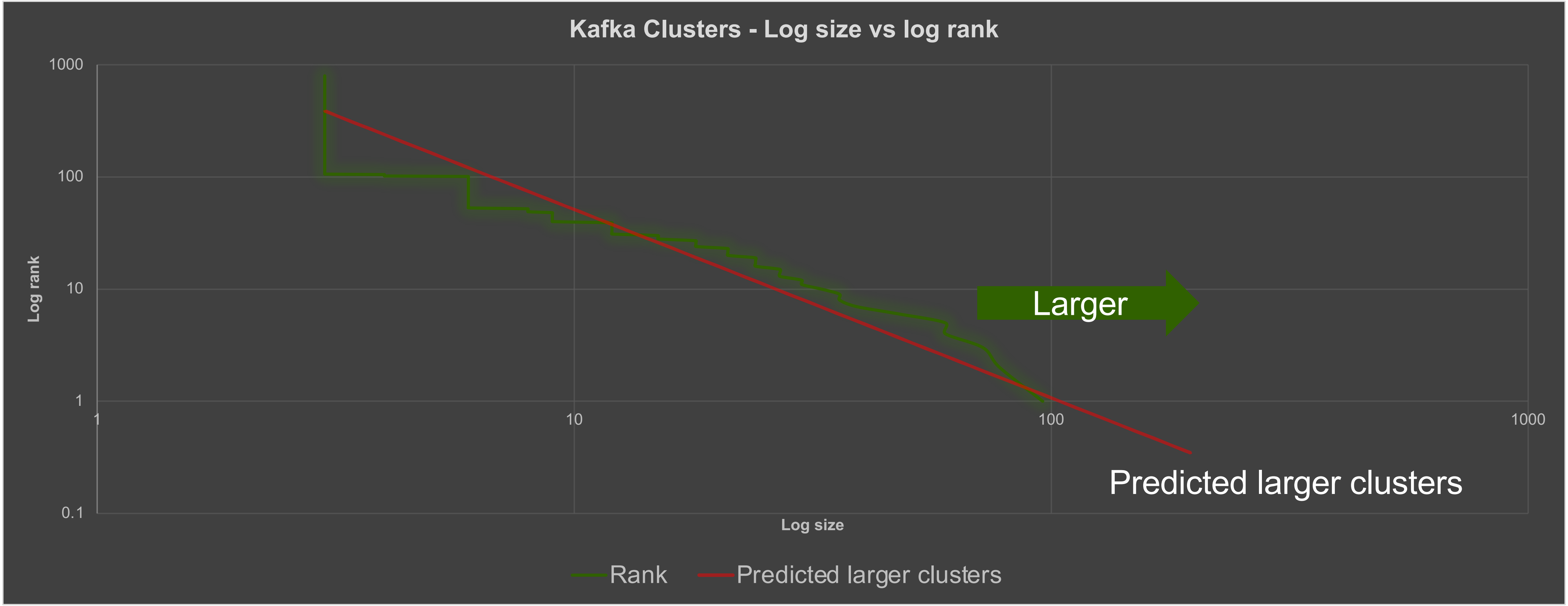
African Elephant, 7 t

Maraapunisaurus, extinct dinosaur, 150 t

 © 2024 NetApp, Inc. All rights reserved.

Predicted larger clusters

Extrapolation of size from Zipf's law + largest observed cluster

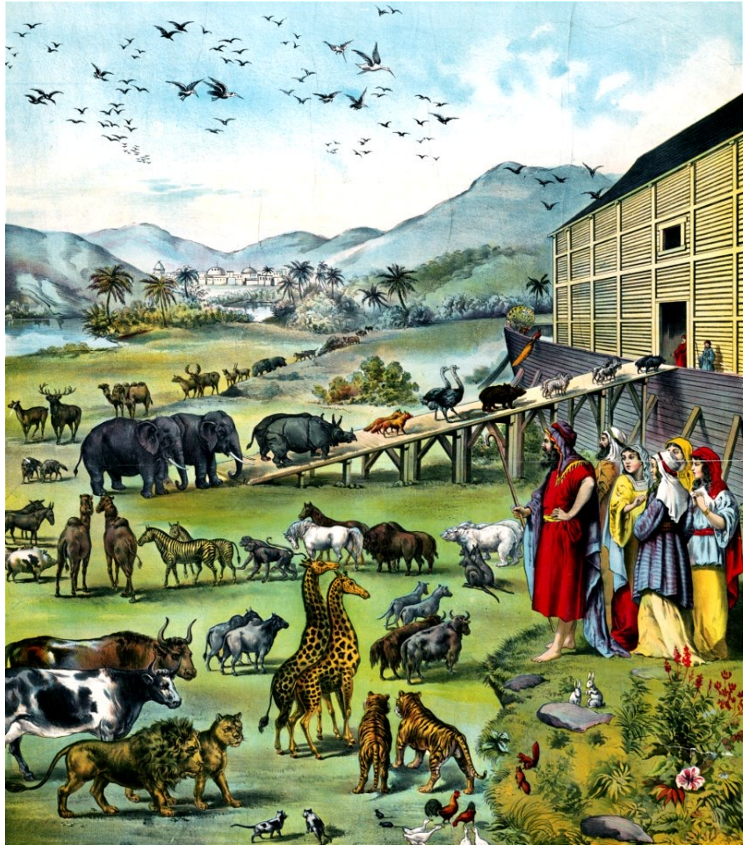


 © 2024 NetApp, Inc. All rights reserved.

So What? Kafka and Zipf's Law (2)

Estimate total nodes for more clusters

Animal transportation problem

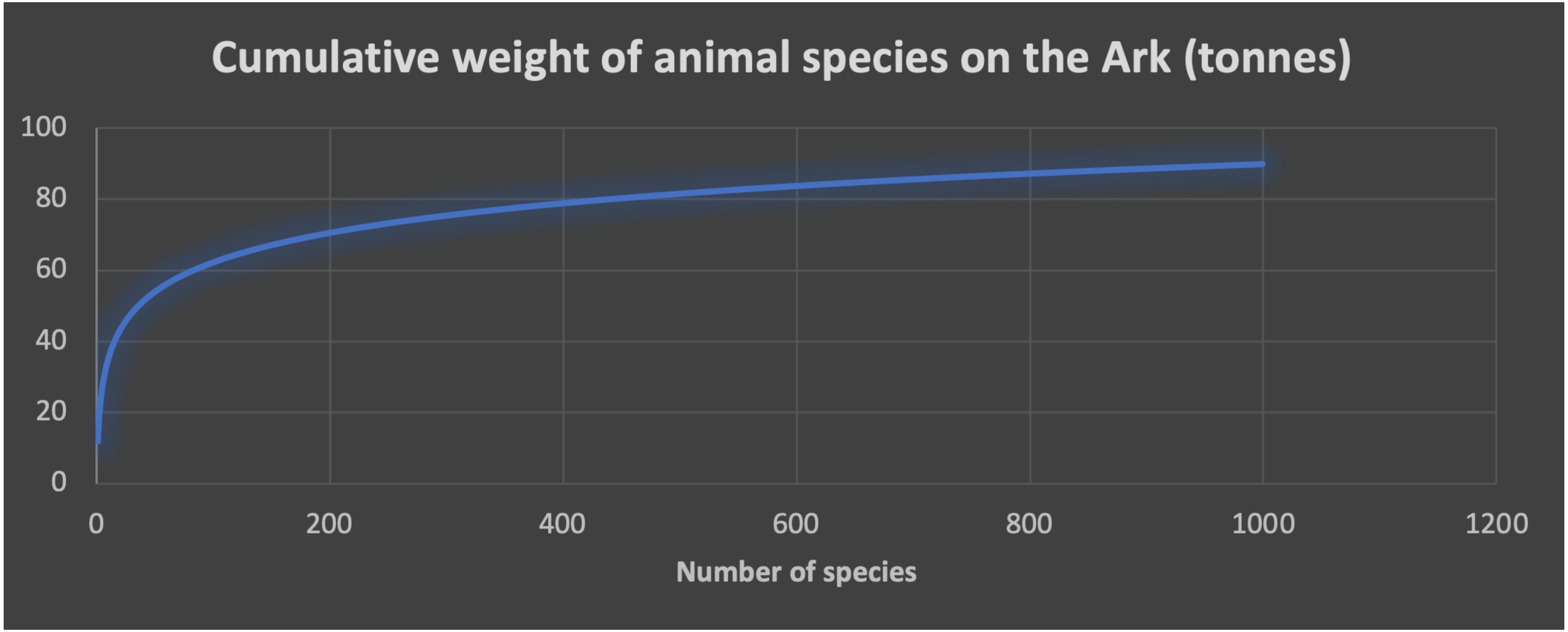


How many animals can fit in a boat?
(Source: Public Domain)

 © 2024 NetApp, Inc. All rights reserved.

If you know the size of the biggest thing you can predict the total size

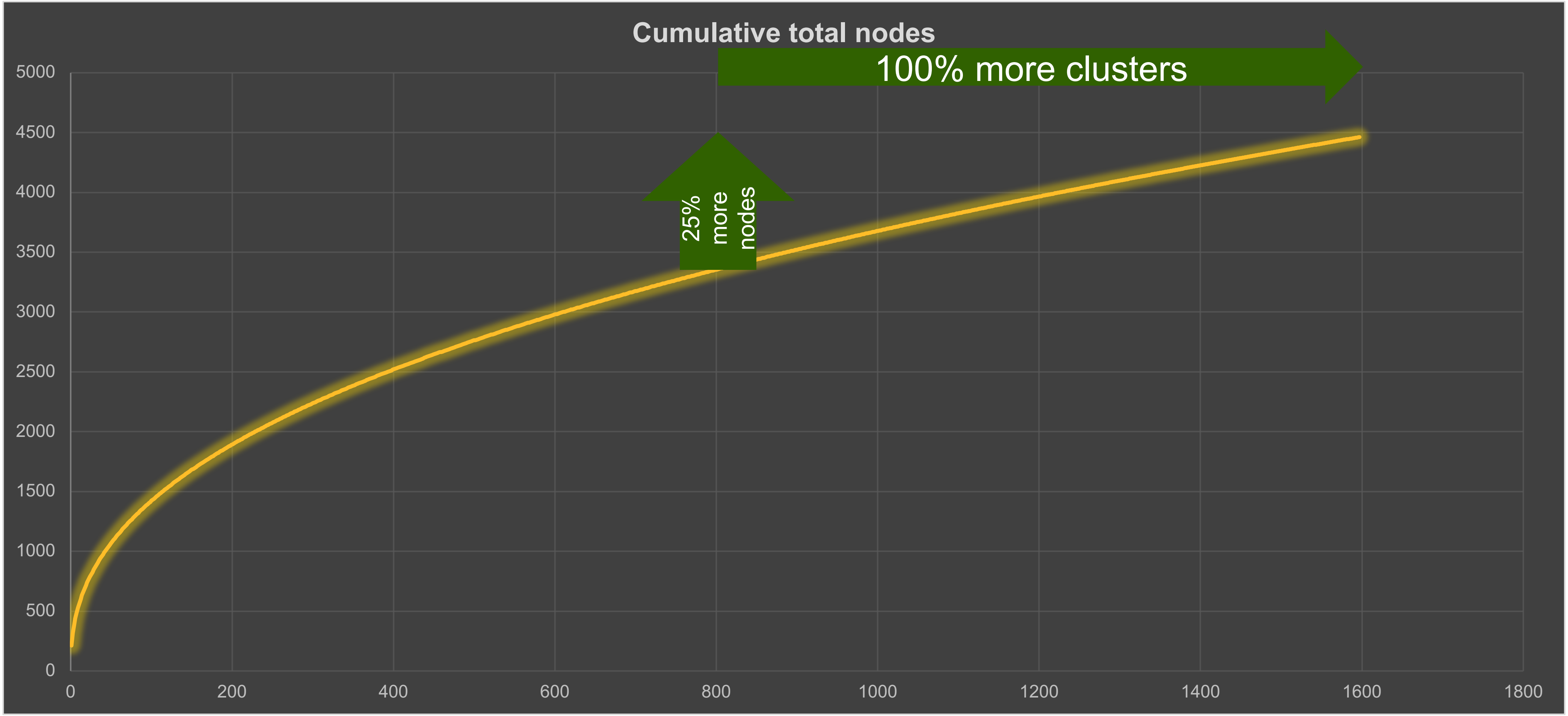
Total weight of animals on Ark (assuming elephant is the largest) tends to 90 tonnes



 © 2024 NetApp, Inc. All rights reserved.

Doubling number of Kafka clusters

Only increases total nodes by 25%

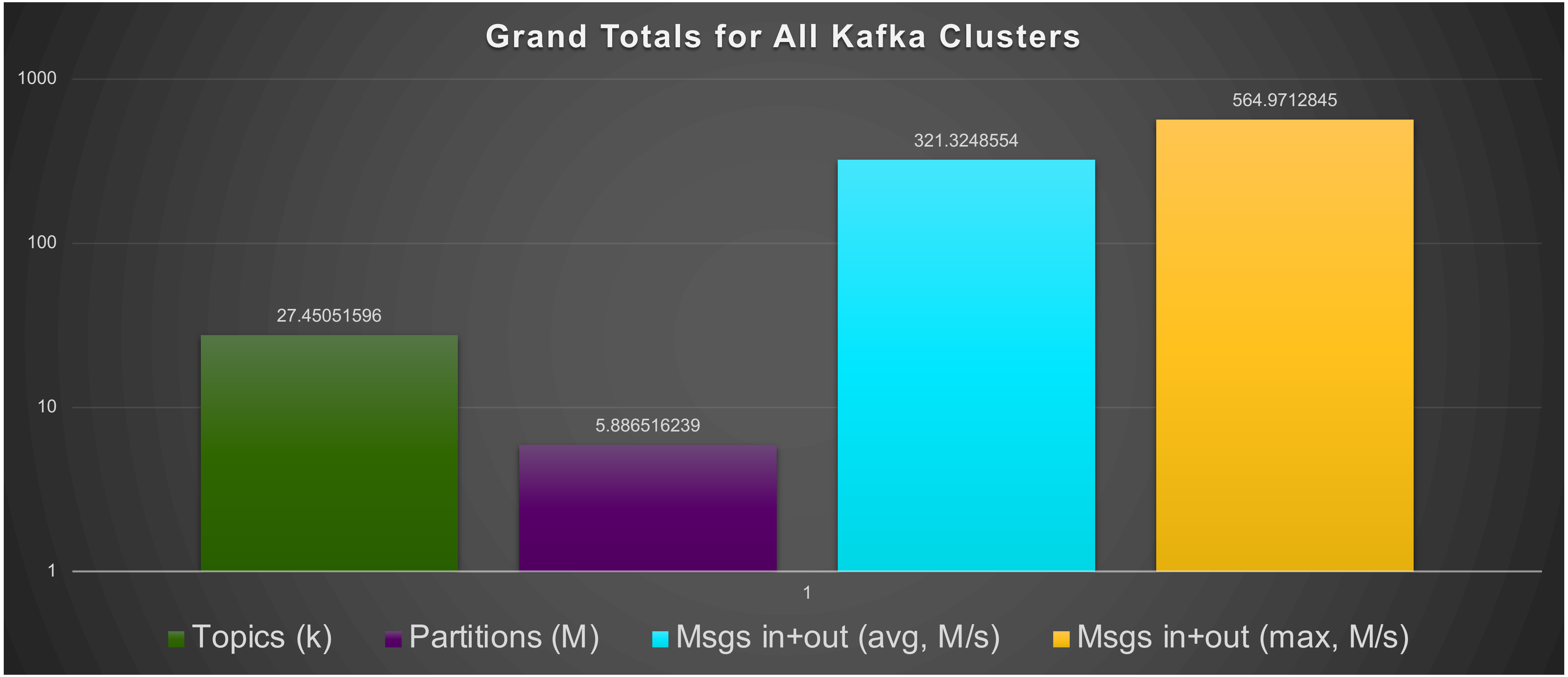


 © 2024 NetApp, Inc. All rights reserved.

Assuming Zipf distribution…

Knowing metrics for our biggest cluster we can estimate total values for ALL CLUSTERS

27K topics (probably underestimate), 5.8 M partitions; 321-564 million messages/s

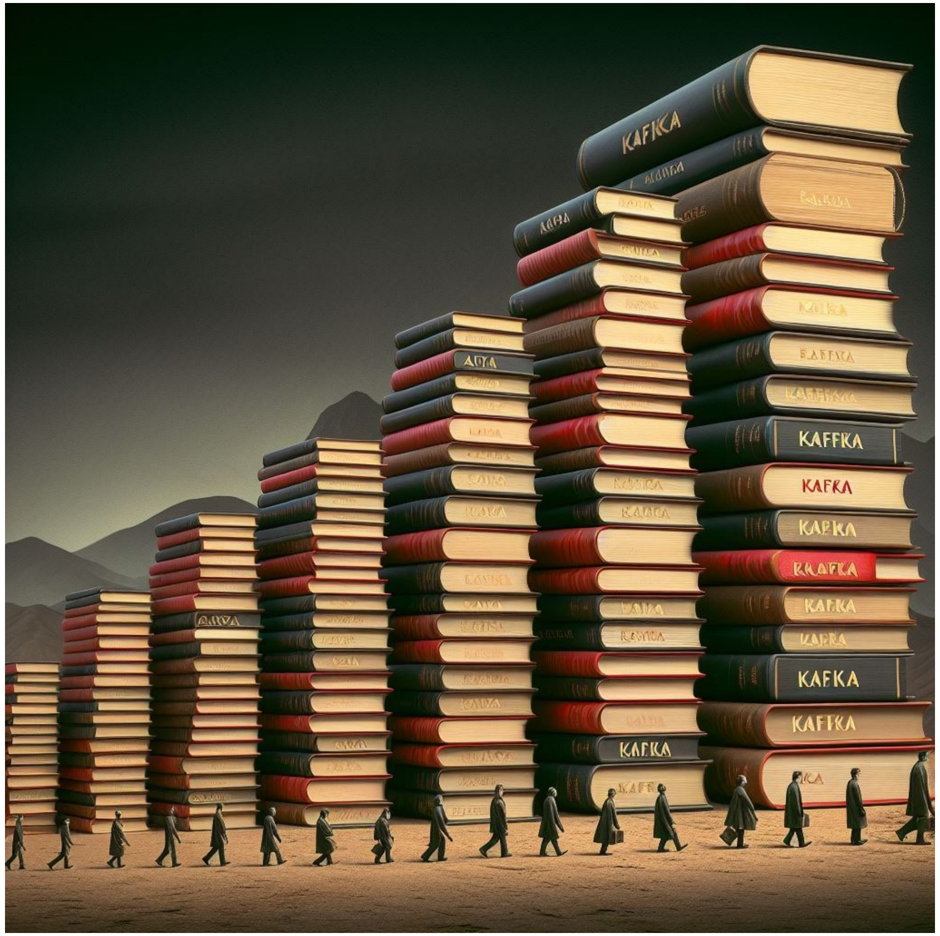


 © 2024 NetApp, Inc. All rights reserved.

Conclusions

Kafka cluster size distribution is Zipfian

- Lots of small clusters
- Few big clusters
- Can make predictions from top cluster/s
  - Bigger clusters possible
  - Totals – e.g. topics, partitions, brokers, throughput
- A wide distribution of sizes is observed
  - Kafka is horizontally scalable
  - Fits many different customer workloads
  - Some clusters split/multiply over time
  - Some clusters grow in size over time



(Source: DALL·E 3)

 © 2024 NetApp, Inc. All rights reserved.

Summary: Four unsurprising Kafka performance results

1. KRaft supports lots of partitions and
2. (maybe) slightly faster data workload latencies
3. Tiered-storage uses extra broker resources and is slightly slower for reads (from remote storage)
4. Kafka clusters size distribution follow Zipf's Law

Unsurprised piglets
(Source: Adobe Stock)

 © 2024 NetApp, Inc. All rights reserved.

But some interesting performance engineering things

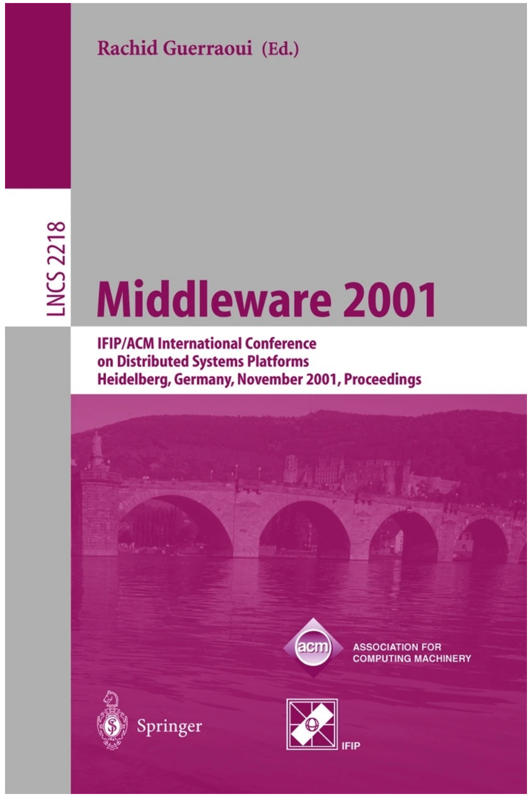
- Hard to benchmark Kafka still
  - Better, more sophisticated (e.g. end-to-end latency, high loads, metrics capture and analysis), easier to use tools (e.g. “Kafka benchmarking as a service”)
    - Tried OpenMessaging but couldn’t get it to work
- More science is better – e.g. results comparison

Curious pig
(Source: Adobe Stock)

 © 2024 NetApp, Inc. All rights reserved.

But some interesting performance engineering things

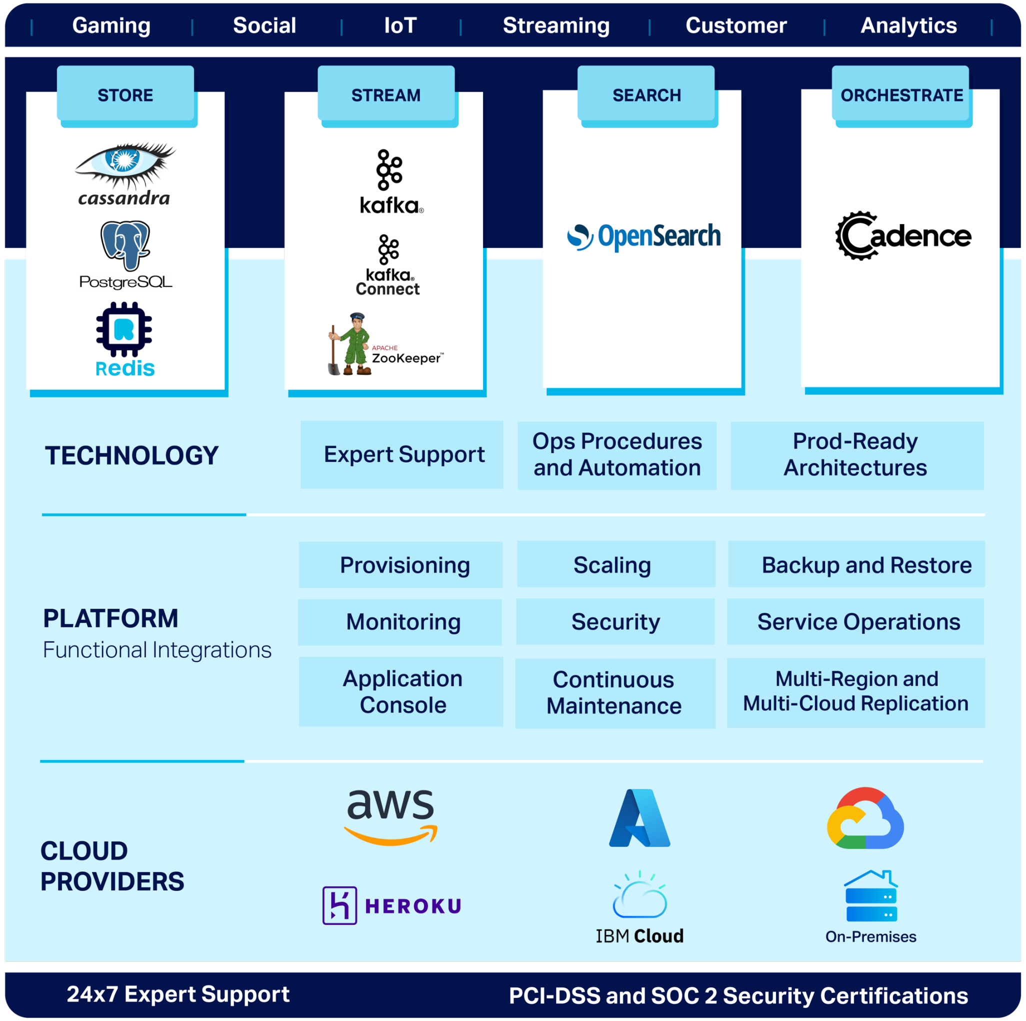
- Performance benchmarking of cached (e.g. tiered storage) systems is (still) tricky
  - One of my earlier papers/talks was on Enterprise Java caching
    - **Entity Bean A, B, C's: Enterprise Java Beans Commit Options and Caching, Middleware 2001**
    - Results were variable depending on cache settings, workloads and cache hit ratio etc
    - Caching benefit not linear/predictable
    - Hard to get repeatable/understandable results
  - Not much has changed
    - For tiered storage, the baseline is local (cache) however
    - So remote storage performance is harder to understand
    - And there may still be surprises – in general we recommend benchmarking your workloads
- Zip's law is widely applicable
  - And fits our Kafka cluster distribution
  - With some useful predictions



 © 2024 NetApp, Inc. All rights reserved.

Instaclustr Managed Platform

- Try us out!
  - Apache Kafka and more
  - Free 30-day trial
  - Developer size clusters
  - www.instaclustr.com
- All my blogs (100+):
  - https://instaclustr.com/paul-brebner



 © 2024 NetApp, Inc. All rights reserved.

Paul Brebner

Open Source Technology Evangelist

NetApp Instaclustr

 © 2024 NetApp, Inc. All rights reserved.

NetApp

THANK YOU

© 2024 NetApp, Inc. All rights reserved.