Fast, Highly Available, In-Memory Vector Database for AI at the Edge

Fast, Highly Available, In-Memory Vector Database for AI at the Edge

Our new AI overlords have arrived, and developers have rushed to integrate cloud-based AI services into their applications. While their adoption in the cloud is facilitated by fast (usually secure) collocated access, their adoption at the edge is beginning to receive more attention. Explore the use of AI at the edge, including the infrastructure, scalability, integration and cost challenges faced. As a part of the solution, we introduce Oracle Coherence's new vector features that enable us to deploy a lightweight, highly available, distributed in-memory vector database at the edge.

And of course, a demo of a fast, precise, non-GPU needed RAG import and chat demo.

Speaker

Adao Oliveira Junior

Solution Architect

Safe harbor statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, timing, and pricing of any features or functionality described for Oracle's products may change and remains at the sole discretion of Oracle Corporation.

Knowledge learning systems at the Edge

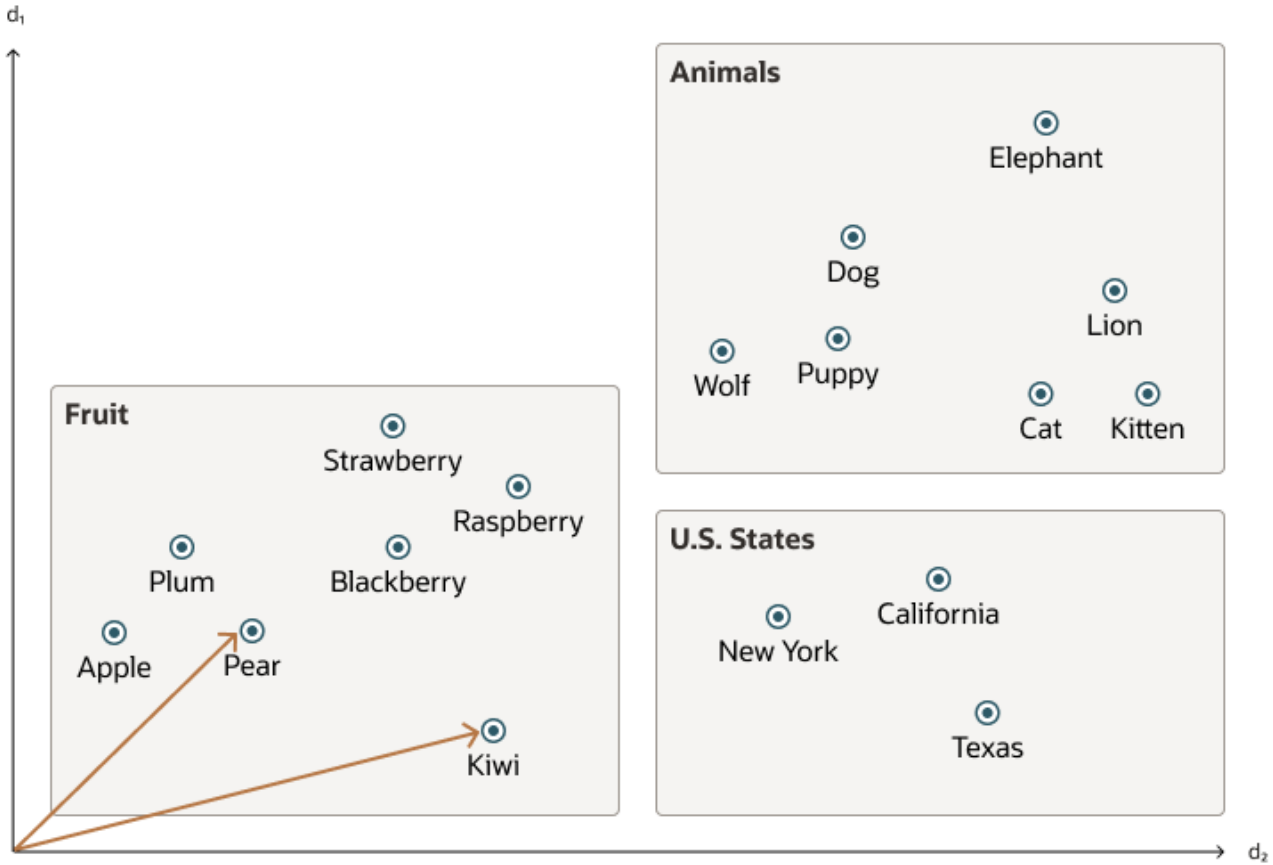
- Anomaly and fraud detection
  - Cell Tower equipment monitoring
  - Financial transactions monitoring, insurance fraud
  - Hospitals, diagnostics testing in radiology, pathology, cardiology and dermatology
- Language translation and NLP
  - Customer support at retail, healthcare, restaurants
- Similarity Matching in Recommendation system – audio/video streaming, retail, restaurant
- Secure personalized content
  - Medicine, ads, sales reports, response generation
- Image recognition
  - identify objects, classify images, and track motion
  - In-vehicle autonomous systems – aircraft, car, boat, rocket
- Offline “search” engine

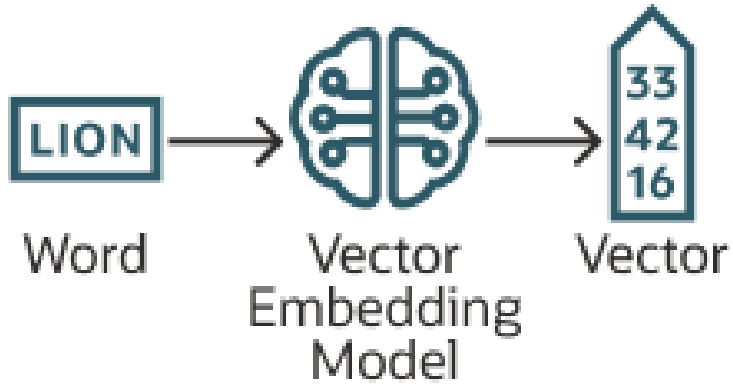
Considerations about Edge GenAI

- Can coordinate w/ larger central Data Center GenAI LLM hybrid architectures
  - Need zero touch deployment, fleet & observability tools
- A LLM in every pocket - iPhone, Android
  - Mobile RAG for personal content creation but not the focus for this session
- Not addressed by pure cloud providers
  - RAG, LLM inference running locally in k8s for latency, data privacy, disconnected reasons
- A cluster for every store, restaurant, hospital, vehicle, bank branch, factory floor, cell tower, oil/gas facility, battle platoon, home?

What are Vector Embeddings?*

- **Measures the relatedness of tokens (text strings)**
- **Embeddings are commonly used for:**
  - Search
    (where results are ranked by relevance to a query string)
  - Clustering
    (where text strings are grouped by similarity)
  - Recommendations
    (where items with related text strings are recommended)
  - Anomaly detection
    (where outliers with little relatedness are identified)
  - Diversity measurement
    (where similarity distributions are analyzed)
  - Classification
    (where text strings are classified by their most similar label)
- **= A vector of floating point numbers**

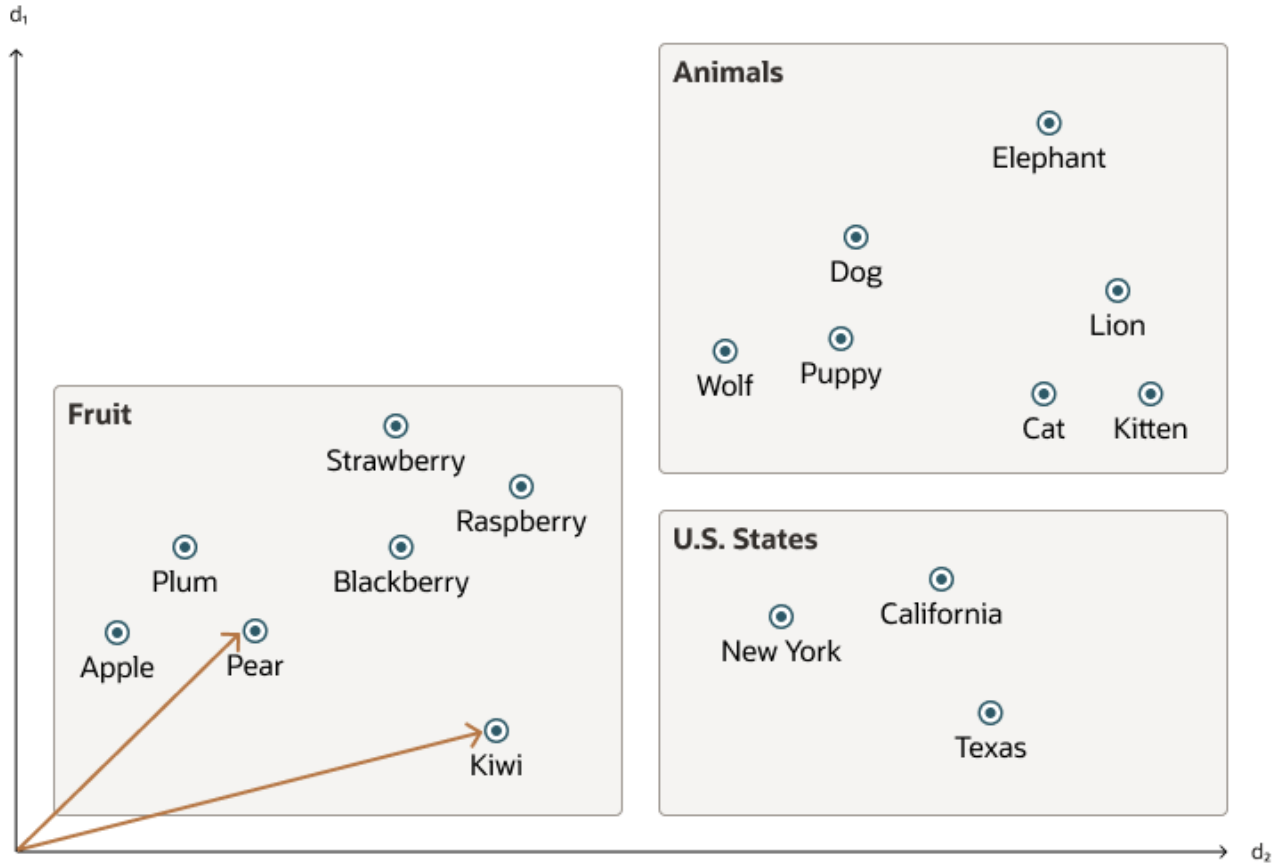




*OpenAI recommends cosine distance for its embeddings https://platform.openai.com/docs/guides/embeddings

When to use each distance metric for accuracy

- **L2 Squared (Euclidean)**
  - Anomaly and Fraud Detection
  - Clustering Analysis
- **Inner Dot Product**
  - Image Retrieval and Matching
  - Neural Networks and Deep Learning
  - Music Recommendation
- **Cosine Similarity – most popular RAG metric**
  - Document Similarity
  - Topic Modeling
  - Collaborative Filtering



https://medium.com/kx-systems/how-vector-databases-search-by-similarity-a-comprehensive-primer-c4b80d13ce63

Finding a Solution

Open-source and scalable to enterprise

Coherence: A jewel in Oracle's middleware

- In-memory data grid with persistence
- Out-of-the-box integration with Oracle Database, MySQL, PostgreSQL
- More than just another cache: distributed processing, eventing, continuous querying
- Fast, scalable, reliable, multi-site capable
- Successful adoption by large customers
- Used in mission critical systems, analytics
- {Kubernetes, cloud, polyglot}-ready
    - Open-source Kubernetes operator
    - Native Java, C++, .Net, Go, Python, JavaScript libraries + REST, gRPC, GraphQL, OpenTelemetry

Coherence AI: A new jewel facet

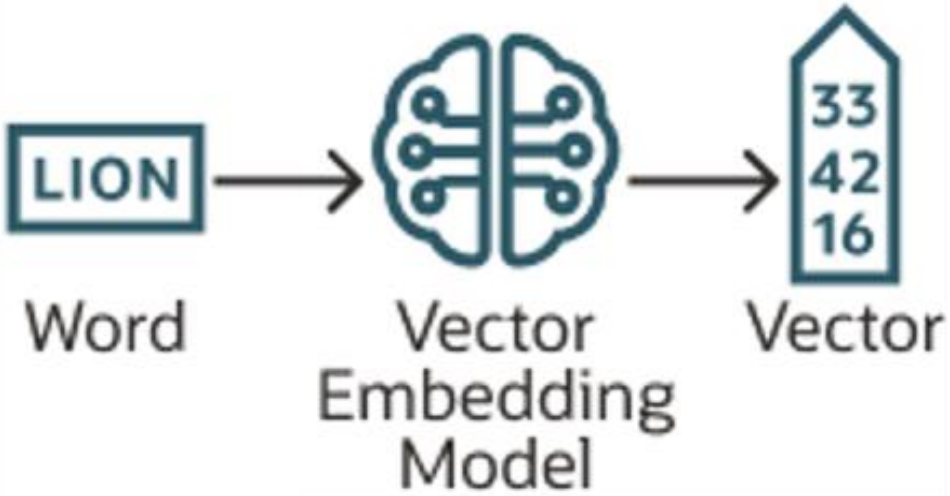
Coherence CE

Two major things added to become a Vector Store/Database

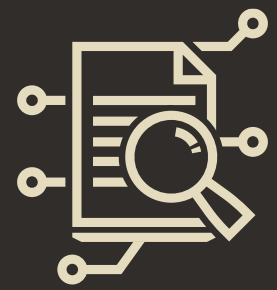
1. Calculating and Storing Vector Embeddings
2. In-place Parallel Semantic Similarity Search

Calculating and Storing Vector Embeddings

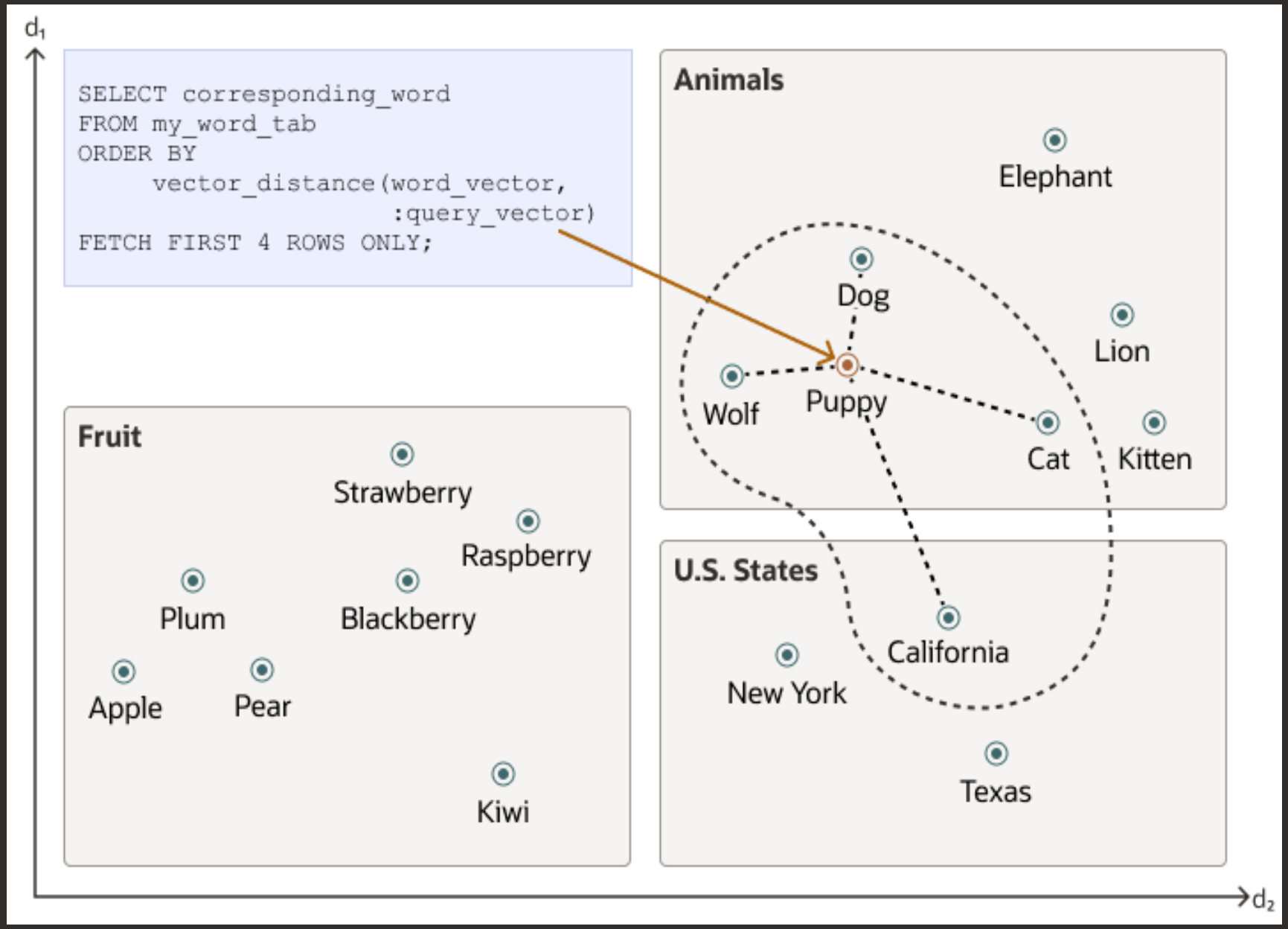
- **Distance calculation for semantic similarity search**
  - CosineDistance - default
  - L2SquaredDistance
  - InnerProductDistance
  - Custom
- **Specialized Vector types**
  - BitVector, Int8Vector, Float32Vector
  - Based on BitSet, byte[], float[] respectively
  - Custom
- **Any ONNX format Embedding Model**
  - i.e. all-MiniLM-L6-v2 sentence transformer model



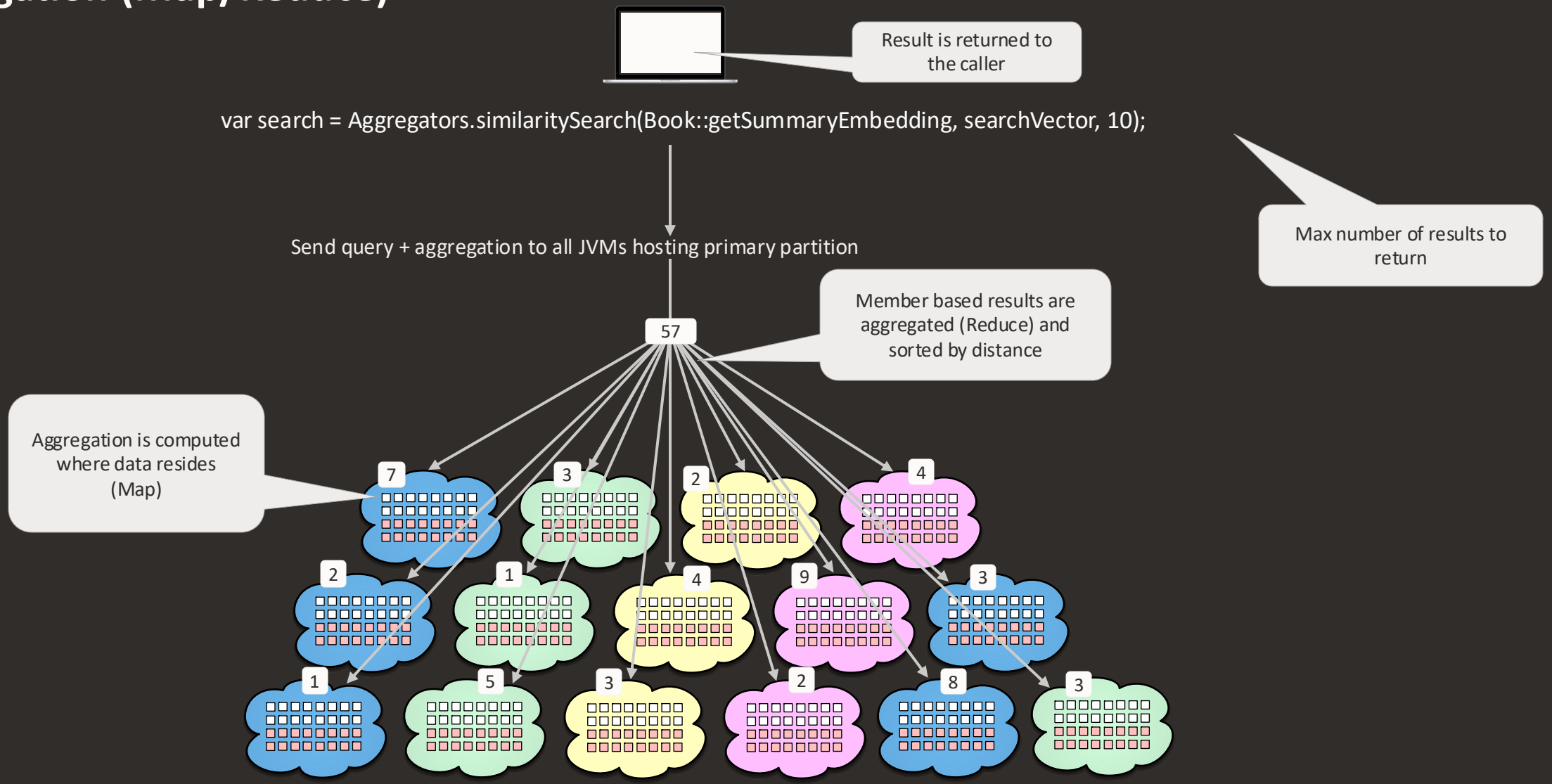
Semantic Similarity Search



- SimilaritySearch – Coherence Entry Aggregator
- Based on relatedness of tokens/text strings/mixed media
- Parallel search where data resides for high performance!
- Like a map reduce operation



Aggregation (Map/Reduce)



Indexing of Semantic Similarity Search

- HNSW – Hierarchical Navigable Small World
  - Default Cosine, but can configure for L2, IP

```
NamedMap<String, Book> books = session.getMap("books");
books.addIndex(new HnswIndex<>(Book::getSummaryEmbedding, 768));
```

- Binary Quantization
  - Converts float32 to 1 bit values – 32x smaller, very fast!
  - Hamming Distance used to retrieve efficiently
  - Oversampling can be configured and re-scoring auto performed

```
NamedMap<String, Book> books = session.getMap("books");
books.addIndex(new BinaryQuantIndex<>(Book::getSummaryEmbedding).oversamplingFactor(5));
```

Metadata Filtering of Semantic Similarity Search

- RAG metadata can be pre-filtered in conjunction w/ Similarity Search

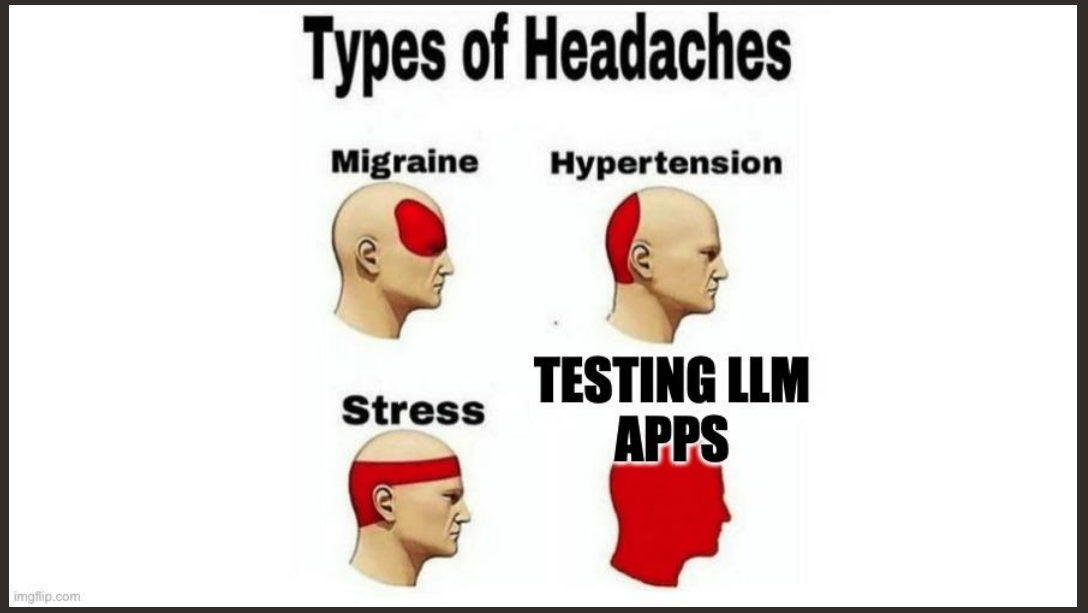
```
var search  = Aggregators.similaritySearch(Book::getSummaryEmbedding, searchVector, 3)
                         .filter(Filters.equal(Book::getAuthor, "Jules Verne"));
var results = books.aggregate(search);
```

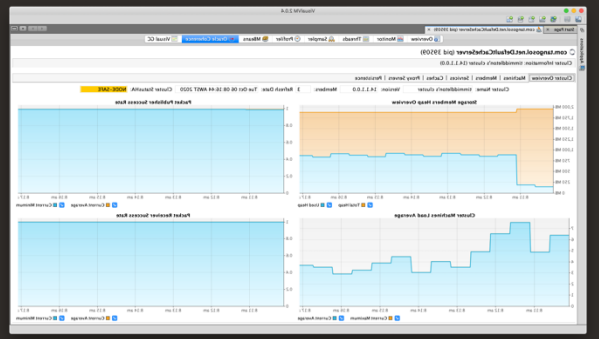
- Works w/ brute force or index-based search
- Very powerful queries combining exact metadata filtering w/ Similarity Search
- Can combine w/ standard Coherence Filters or custom
- This is traditionally very difficult to scale for Vector-only Databases

Observability for LLM Retrieval (RAG) based applications

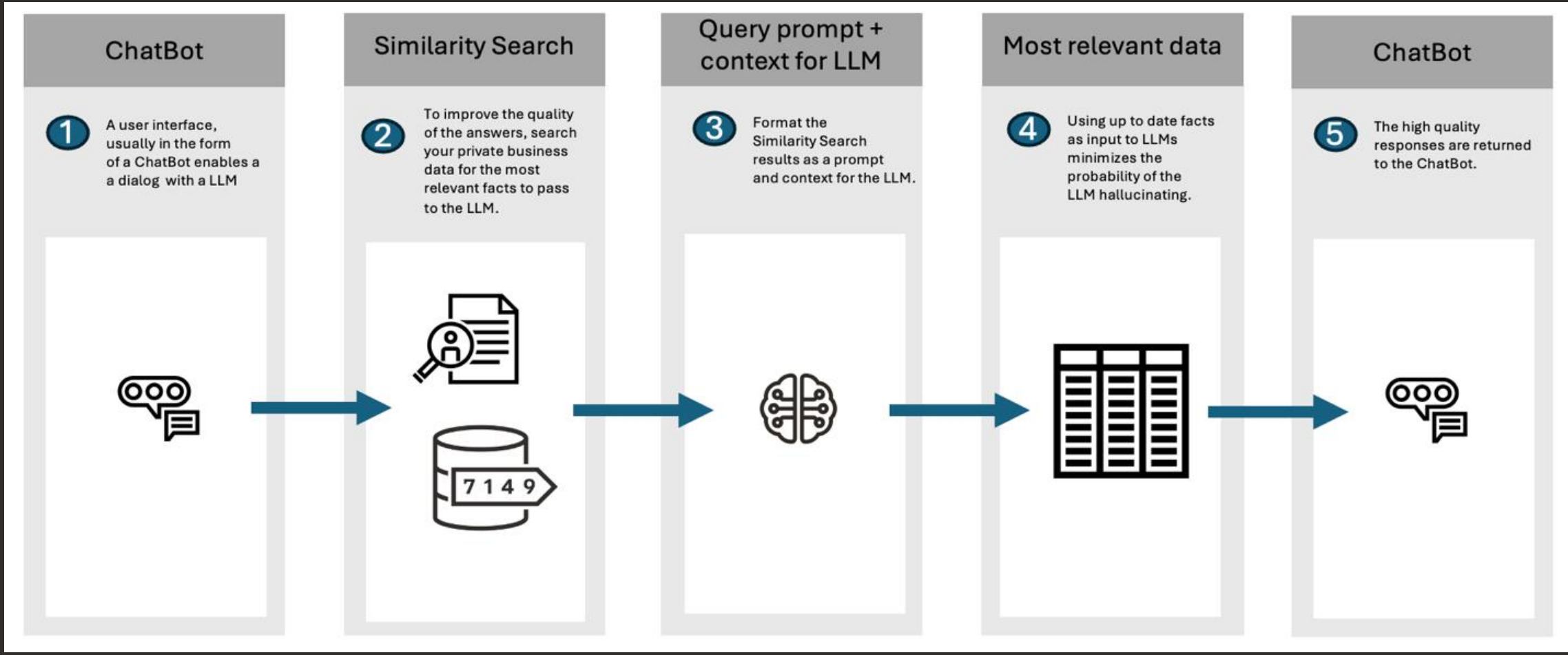
- ✓ Request/resp freq, times, rates
- ✓ Track token usage, costs
- ✓ Doc, Chunk, embedding stats
- ✓ Cost v Power consumption
- ✓ Reduce trial & error prod testing

- JMX Mbeans
  - no equivalent in Python world
  - Coherence VisualVM plugin
    - https://github.com/oracle/coherence-visualvm
- OpenTelemetry integration
- Prometheus, Jaeger, Zipkin, Grafana dashboards





Example RAG workflow



Demo

Using Coherence for Retrieval Augmented Generation (RAG)

Augment LLMs w/ supplemental content via computed relevant vector embeddings

Developed innovative RAG system on Coherence:

- Highly scalable data ingestion (millions of docs)
- Outperform for accuracy, reduce hallucinations
- Integration with popular AI libraries, observability
- Optionally integrates w/ any persistent vector DB
  - OpenSearch, Oracle DB 23ai
  - Persistent storage and search of docs, chunks and vectors

Internal Use Case validation: Findings

Internal data ingestion workload: the public Oracle Docs dataset

- All Applications, and Technology documentation
- 1,837,944 documents, split into 8,125,179 chunks
- Embeddings created using all-mpnet-base-v2 model

Factor	Before Coherence AI	With Coherence AI
Embedding Machine implementation	Custom Python program on GPU writing to MySQL, OpenSearch	36x 10 OCPU E4.Flex VMs scaling out Coherence + Helidon, writing to DB 23ai
Overall execution time	40 hours, 8 minutes	1 hour, 30 minutes
Cost per ingestion run	$160	$15

Docs data ingestion is 26x faster and 11x cheaper with Coherence AI

90%+

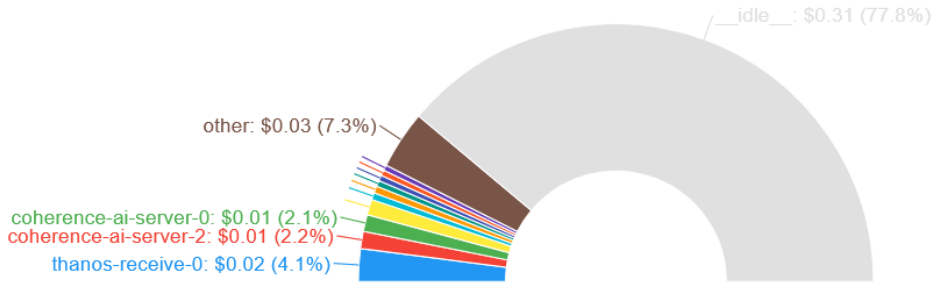
Cost reduction using Coherence

Highly efficient, scalable

Today by pod

25 May 2024 by Pod

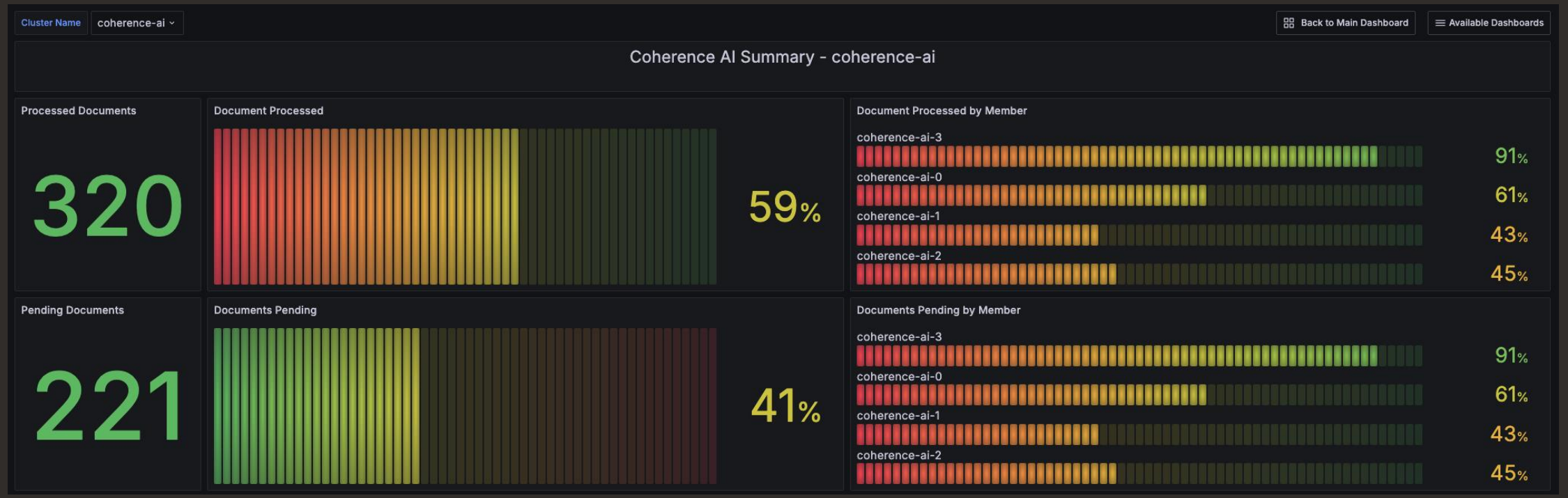
Date Range: Today | Breakdown: Pod | Resolution: Entire window | Currency: USD



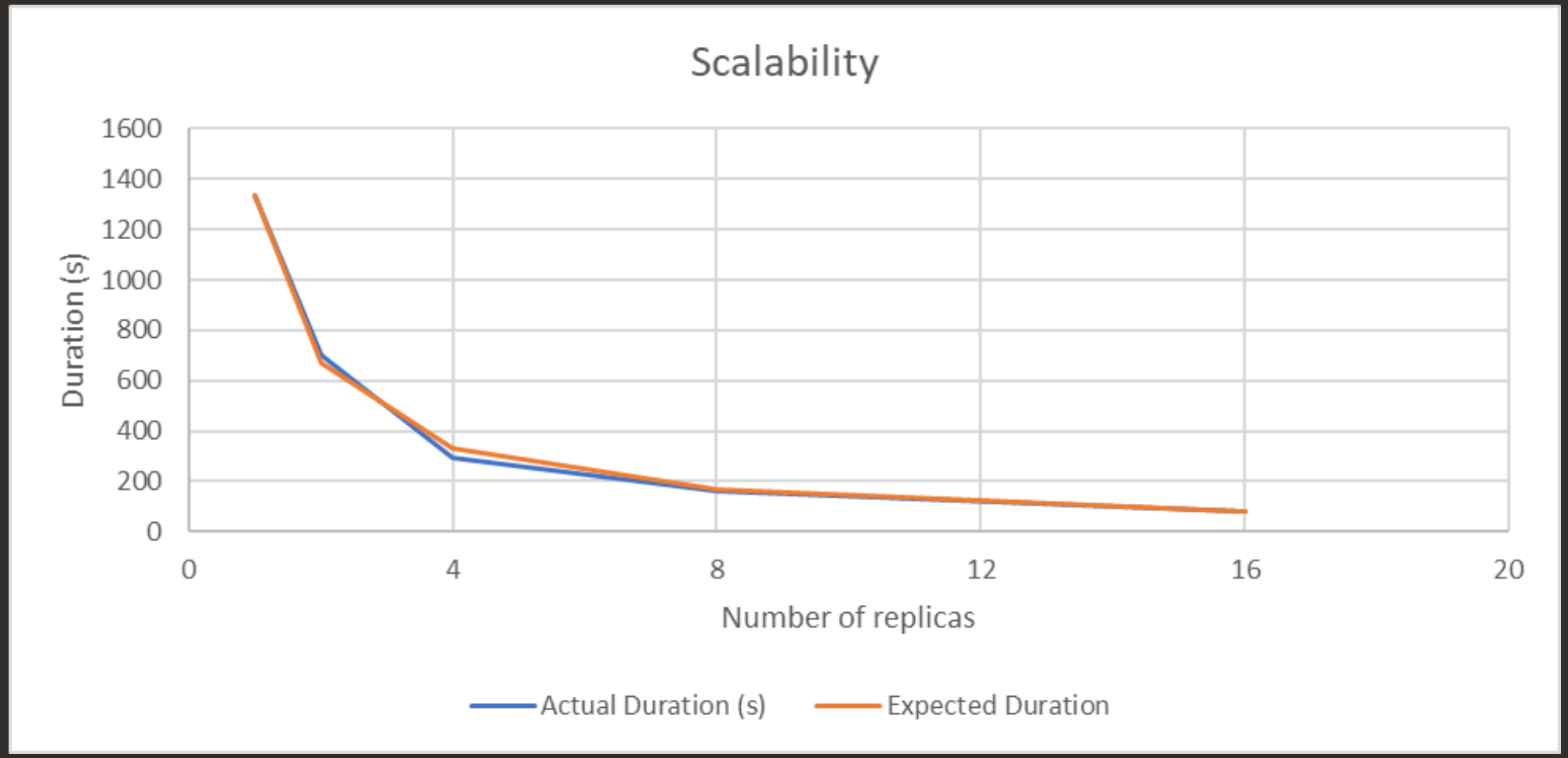
Name	CPU	RAM	PV	Efficiency	↓ Total cost
Totals	$0.21	$0.18	$0.00	17.9%	$0.39
__idle__	$0.14	$0.17	$0.00	—	$0.31
thanos-receive-0	$0.01	$0.00	$0.00	4.4%	$0.02
coherence-ai-server-2	$0.01	$0.00	$0.00	Inf%	$0.01
coherence-ai-server-0	$0.01	$0.00	$0.00	Inf%	$0.01
coherence-ai-server-1	$0.01	$0.00	$0.00	Inf%	$0.01

Reduce carbon footprint and energy consumption = lower cost

Grafana dashboards



Linear Scalability



Unique to Coherence AI – Elegant Simplicity at Edge

- ✓ Multipurpose In-Memory Data Grid w/ Vector Store/DB capabilities
- ✓ Scale from 1 JVM on Raspberry Pi to thousands in Data Center
- ✓ Unlimited data types
- ✓ Distributed Cache w/ auto rebalancing, fault tolerance
- ✓ In-memory vector storage and similarity search + metadata filtering
- ✓ Continuous Query cache & pub/sub events
- ✓ Polyglot, GraphQL, CohQL
- ✓ Observability using Mbeans, OpenTelemetry
- ✓ Optional Integration w/ any document storage system
  - ✓ Oracle DB 23ai

Coherence CE is Open-source

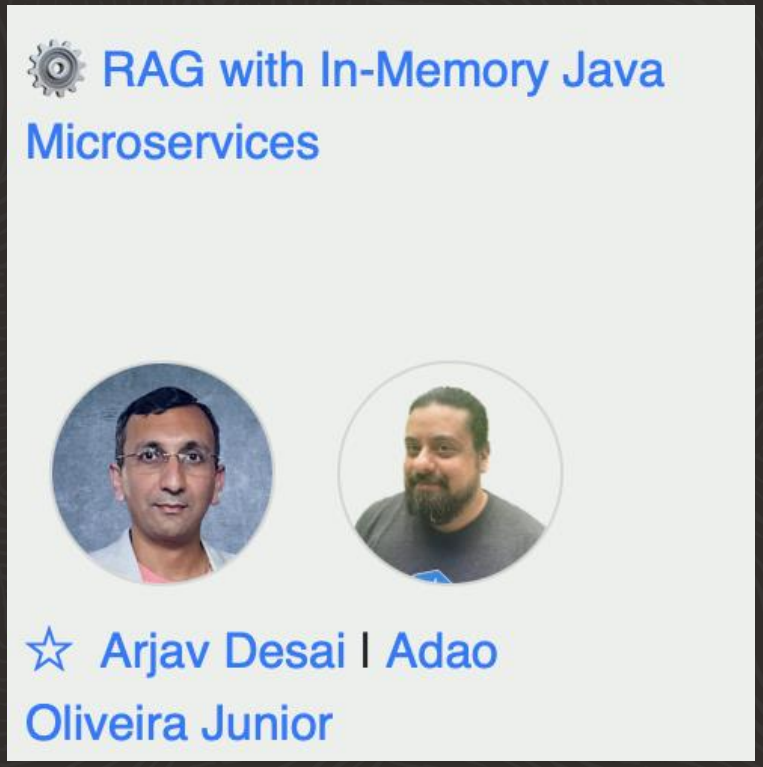
https://coherence.community/
https://github.com/oracle/coherence

Other sessions

RAG with In-Memory Java Microservices

La Fronteira I

Thursday - 4:00pm



Thank you


ORACLE