KAFKA + NETAPP ^AI

Building a Connector Better than Market

🎙️ Guang Zhao

DATA

🍺 NetApp Ontap/StorageGrid

🐿️ Kafka pipeline

INTERFACE

🍺 Simple (object) storage service (S3)

🐿️ Kafka Connect

🦀 Aiven/Confluent/... Connectors

END OF STORY?

Everything seems standard...

PROBLEM

✅ Parquet, CSV, JSON

❓ 1GB ~ 1TB per object

LIMITATION LEFT & RIGHT

ONTAP/STORAGEGRID

Unstable single read >10GB object

MARKET

Confluent	Aiven	Lenses
<2GB[a]	<min(Instance,10GB)[a,c,d]	<10GB[d]
Closed	-	-
-	Utf-8 only[b]	-
-	Inefficient conversion	-

- a. Parquet
- b. Text
- c. "Download approach"
- d. "Single stream approach"

PROBLEM IS SCALE

- Simple I/O abstraction for all formats and patterns
- Scalable I/O implementation for object storage

LET'S SOLVE IT

BACKGROUND

TEXT: SEQUENTIAL ACCESS

```
1,first,row
2,second,row
...
N,nth,row
```

```
{ object: 1 }
{ object: 2 }
...
{ object: N }
```

Scan the file from the first to the last line.

PARQUET: SEMI RANDOM ACCESS

```
+----------------+
| Chunk 1, Col 1 <-------+-------+
| Chunk 1, Col 2 |       | Row 1 |
| C...           |       | Row 2 |
| Chunk 1, Col N |       | ...   |
+----------------+       | Row M |
| Chunk 2, Col 1 |       +-------+
| C...           |
| Chunk 2, Col N |
+----------------+
...
| Chunk N        |
+----------------+
| File Metadata  |
+ ---------------+
```

1. jump to the end to read Metadata
2. jump back to chunk, read row in multiple offsets
   - Col_1[i], Col_2[i], ..., Col_N[i]

KAFKA CONNECT

- Poll external records by batches
- Convert types
- Publishes records to Kafka topic

External system as what?

#0: ALL ARE ... BYTE STREAMS

```
abstract class java.io.InputStream {
    ...
    // Only forward
    long skip(long n) throws IOException;

    // Let's not get into mark() and reset()...
}
```

Can't support Parquet

Can't handle large S3 object

BREAK THE ABSTRACTION

#1: WITH RANDOM ACCESS

```
class RandomAccessInputStream extends FilterInputStream {
    ...
    // Forward and backward
    void seek(long offset) throws IOException;
}
```

~~Can't support Parquet~~

Can't handle large S3 object

#2: BROKEN INTO EXTENTS

```
class ExtentInputStream extends RandomAccessInputStream {
    long extentSize;
    long extentOffset;
    ...
}
```

Extent: (offset, size)

DIAGRAM

```
     +-----------------+-----------------+-----+
     | Parquet Decoder | Unicode Decoder | ... |
     +-----------------+-----------------+-----+
                       |
                       |
+----------------------+-------------------------------+
| byte[0], byte[1], ...                          byte[N] |
+================+================+=====+================+
| ext[0]         | ext[1]         | ... | ext[M]         |
+----------------+----------------+-----+----------------+
| S3.GetRange[0] | S3.GetRange[1] | ... | S3.GetRange[M] |
+----------------+----------------+-----+----------------+
```

POWER OF ABSTRACTION

```
                 +-------------+
                 |  Connector  |
                 +-------+-----+
                         |
     +-----------------+----------------+-----+
     | Parquet Decoder | Unicode Decoder | ... |
     +-----------------+----------------+-----+
                         |
+----------------------+-------------------------------+
| byte[0], byte[1], ...                        byte[N]  |
+=================+=================+=====+================+
| ext[0]          | ext[1]          | ... | ext[M]         |
+-----------------+-----+----------+-----+----------------+
                         |
        +---------------+-------------+--------+
        | Object Store  | File System |        |
```

- Extents 🔄 external systems
- Bytes 🔄 data formats

EXTENSIBLE

```
        Data Formats
            ^
            |
    Parquet |
            |
    Text    |
            |   🍺 Ontap  StorageGrid  😄 Aws
            +-----------------------------------> Systems
    Extent-Stream
```

(AUTO-)TUNE OUR CONNECTOR

EXTERNAL I/O CHARACTERISTICS

- External systems
- Workload
- Data formats
- Size

? Find the best Connecor performance

UNKNOWNS

- External system designs & changes
- Documentation inaccuracies
- Human intuition unreliable

MACHINE LEARNING PROBLEM

$$\arg\max_{param} P(connector|param, sys, work)$$

- parameters: extent size
- P, model: throughput
- System: S3, Ontap, StorageGrid, Local
- Representative workload?

🤖 AI ATTEMPT #1

CODE-ONLY AI

👦: Hey 🤖, given my code, what's the optimal parameter?

$$\arg\max_{params} P(connector|param)$$

AI CORRECT?

🤖: "Disk block sizes commonly are 512B or 4KB. Set 1KB."

🤖: "S3 recommends PutObject limit 5GB, max 5TB, Set 5GB."

BENCHMARK

TPC-H DATASET

	Parquet Small	Parquet Large	CSV Small
Table	Customer	Lineitem	Customer
Scale	10	30	10
Rows	1.5M	37.23B	1.5M
Compress	1:2	1:2	1:1
Size(B)	118M	6.3G[a,b]	237M

- a. Beyond Confluent limit
- b. Aiven min instance disk space

WORKLOAD

```
num_polls = 10
batch_size = 128
for i in range(num_polls):
    poll(batch_size)
```

🍺 STORAGEGRID TIME (MS)

Extent(B)	Csv Small	Parquet Small	Parquet Large
1K	480,501	>1min	>1min
4K	470,438	>1min	>1min
1M	4,695	17,134	12,292
4M	2,967	8,630	9,304
1G	2,937	6,782	6,377
4G	2,669	6,242	6,196

- std ~ 5% mean

🍺 STORAGEGRID VS 😀 AWS

Extent(B)	StorageGrid[a]	Aws[a]
1K	480,501	573,496
1M	4,695	4,813
1G	2,937	2,895

- a. CSV Small • std ~ 5% mean

TAKEAWAY

- Scales better than Confluent
- StorageGrid vs Aws comparable
- Optimal extent size
  - depends on data format
  - requires internal knowledge
  - is not what naive AI suggests

🤖 AI ATTEMPT #2

STOCHASTIC GRADIENT DESCENT

```
num_polls = 10
batch_size = 128
x = extent_size
for i in range(num_polls):
    time = model(poll(batch_size), x)
    grad = gradient(model, time, x)
    x = step(grad, x)
```

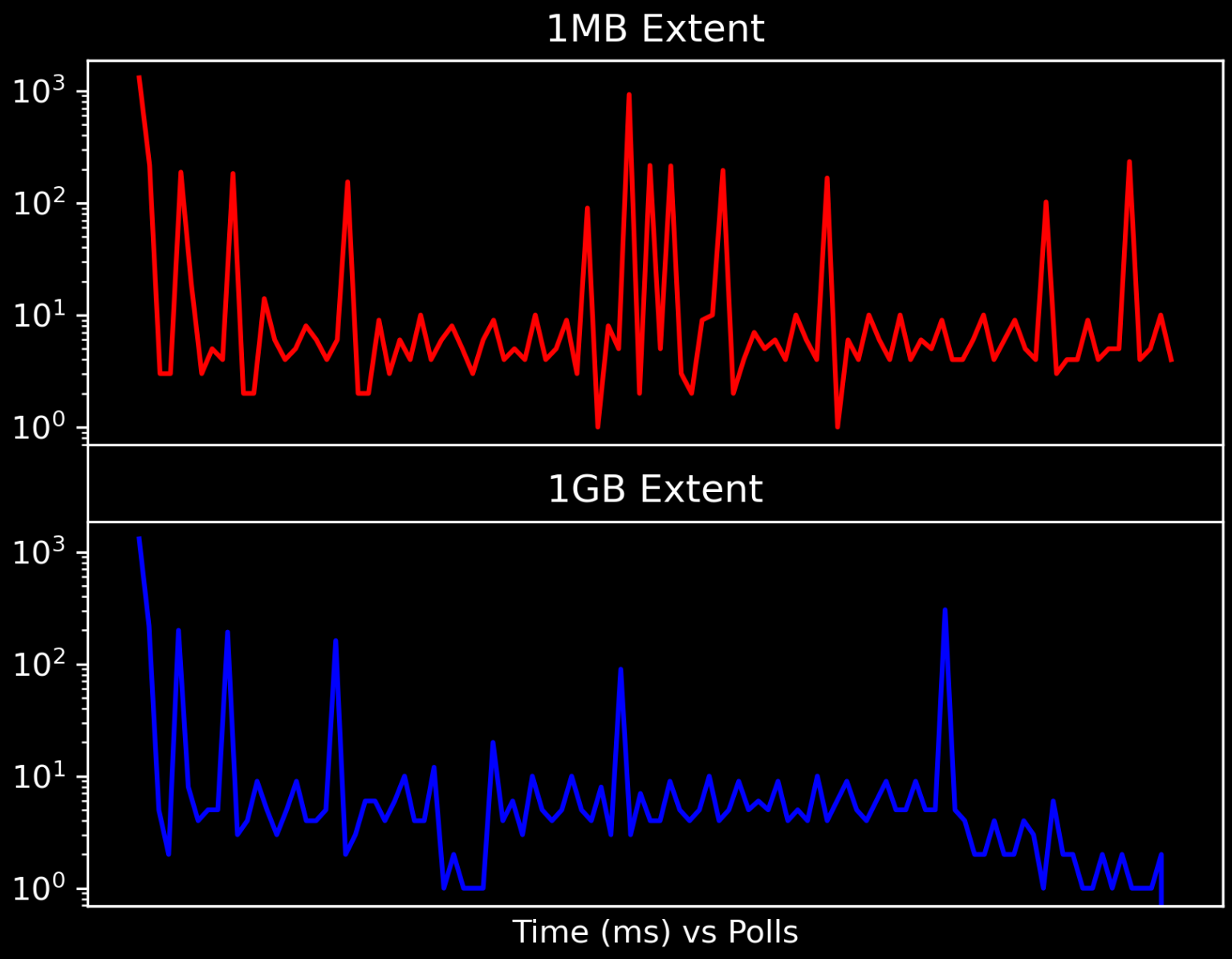
Convergence: found an extent size with fastest poll

PROBLEMS

Model is discrete, non-differentiable.

It assumes each poll time is stable given extent.

But there is no bound.


1MB Extent
10^3
10^2
10^1
10^0
1GB Extent
10^3
10^2
10^1
10^0
Time (ms) vs Polls

🤷 NO GUARANTEE

Based on local info, cannot converge to optimal extent

🤖 AI ATTEMPT #3

Go above abstraction

Combine LLM (#0) and ML (#1)

AGENT FOR BENCHMARK + TUNE

```
while True:
    extent_sizes = Llm.generate_response(
        'Guess optimal extent size',
        context)

    times = []
    for x in extent_sizes:
        times += repeat(
            10,
            model(poll(batch_size), x))

    better_extent_size = find_min(times, extent_sizes)

    context.add(better_extent_size)
```

RESULT

Experimental... ⏳

FEATURES

- Unlimited size
- Extensible formats
- Extensible endpoints
- (Autotune agent ⏳)
- Unicode
- Aync object discovery
- Nested prefixes
- 1-1 Type conversion

👍 Better than Market {🦀,...}

HEADS UP

- 👁️ Cassandra Parquet/Avro Transformer by Stefan
- Operational to analytical:

👁️ Cassandra - 🐿️ Kafka - X?

THANKS

- Anup: Testing
- Amanda: Organizing
- Carlos: Organizing
- Justin: Organizing
- Liam: Discussion
- Nilkua: NetApp configs
- Varun: Organizing
- Win: Ontap Details
- Team Kafka: Review PR
- Team Open Source: Discussion
- Team PoC: Customer & Product

🤷 All technical errors are mine.

THOUGHTS?

💻 https://github.com/instaclustr/kafka-connect-connectors

▶ Contribution welcome!

📧 Guang.Zhao@netapp.com

Slack #opensauce

🏢 Melbourne