SIKE: An Analysis of Serialization Interfaces for Kafka Experiments

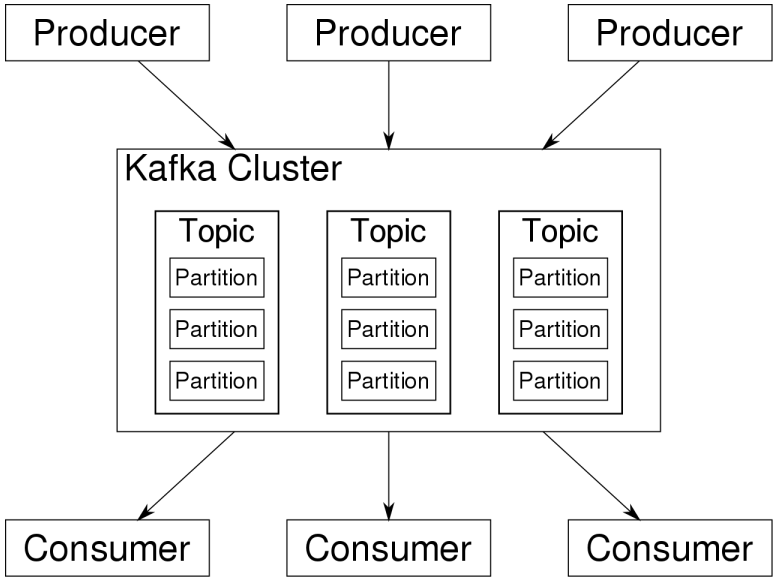
By: Sahil, Wenlong, Kaushik

Background and Motivation

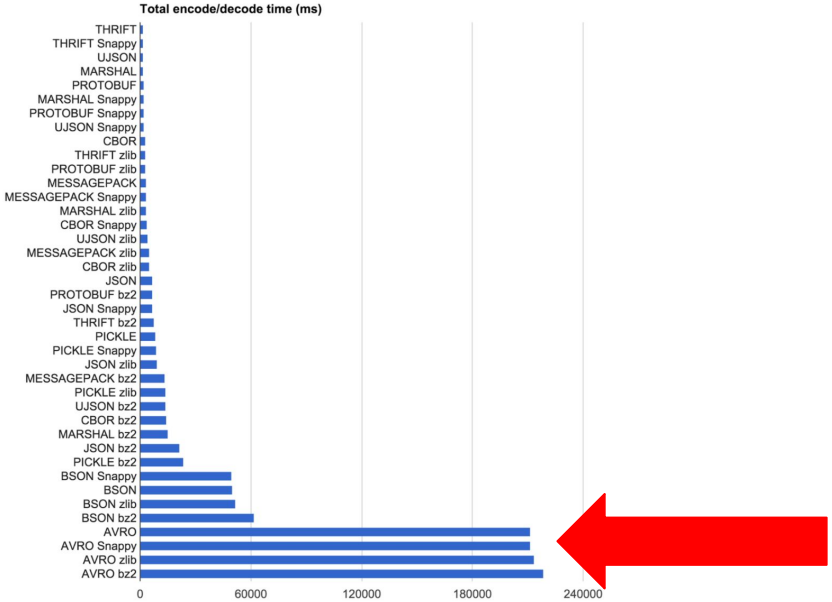
Apache Kafka

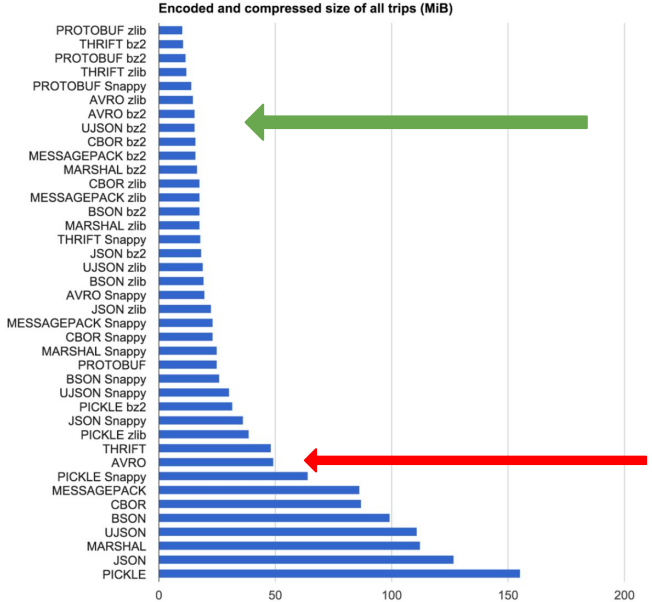
- Massively adopted in industry
  - Most of the Fortune 500 companies
- Event streams
- Producer/Consumer architecture
- Messages stored on distributed cluster
- Messages are byte arrays

- Simple Use Cases:
  - Distributed Messaging
  - Log aggregation
  - Ordered Event Stream
  - Commit Log



Problem/Motivation





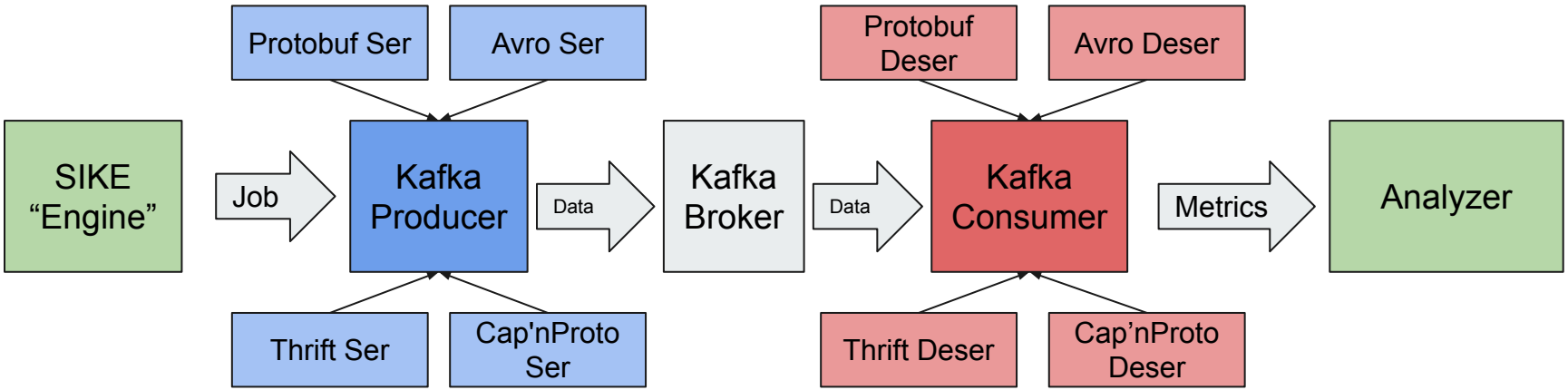
Longer bar is worse

Current Apache Kafka Serializers

- AVRO/JSON
  - AVRO has scheme registry
  - Forwards/Backwards compatibility
  - JSON is easy to use and understand
- Many kafka bindings
  - Some Community driven, some by Confluent
  - Python, C++, Java, Go, Rust, etc
- Some other serializers partially supported
  - Support is patchwork / incomplete
  - Schema Registry only works with AVRO

SIKE

System Overview



Serialization Data

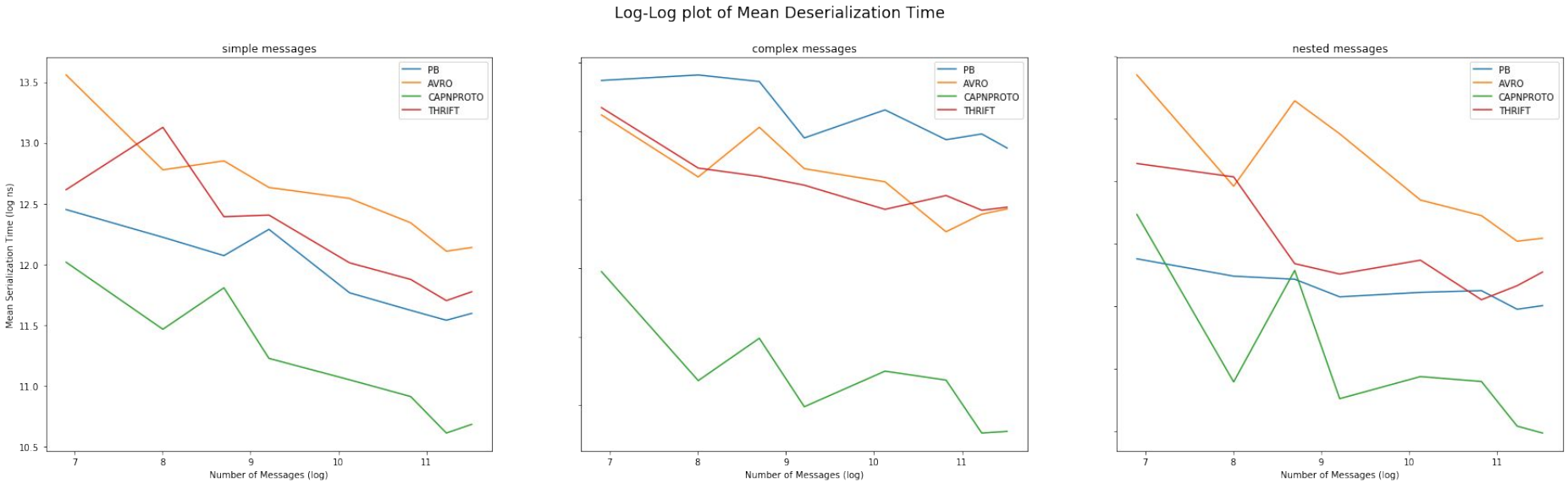
- 3 different formats
- 100 Kb Query string
- 100 element array
- 100 element map

```
message SimpleMessage {
  int64 timestamp = 1;
  string query = 2;
  int32 page_number = 3;
  int32 result_per_page = 4;
}

message ComplexMessage{
  int64 timestamp = 1;
  map<string, int32> storage = 2;
  repeated int32 arr = 3;
}

message NestedMessage{
  int64 timestamp = 1;
  int32 id = 2;
  SimpleMessage simpleMsg = 3;
}
```

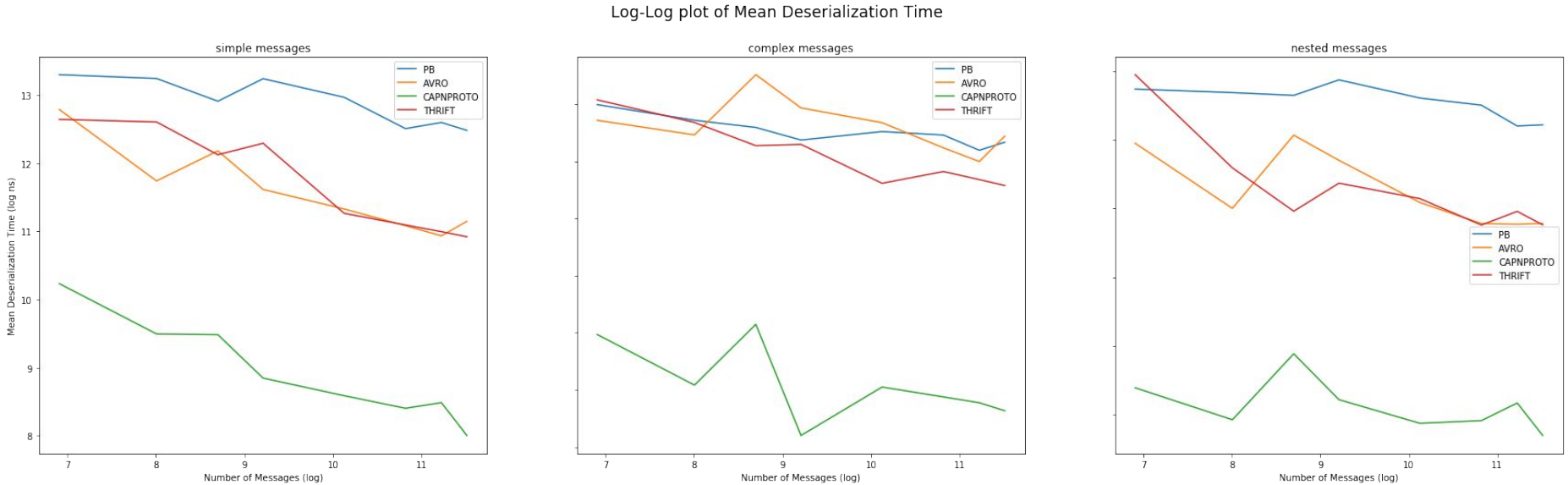
- Simple:
  - Avro: 100018 bytes
  - Protobuf: 100017 bytes
  - Capnproto: 100064
  - Thrift: 100019
- Complex:
  - Avro: 878 bytes
  - Protobuf: 1200 bytes
  - Capnproto: 2848 bytes
  - Thrift: 877 bytes
- Nested:
  - Avro: 100022 bytes
  - Protobuf: 100023 bytes
  - Capnproto: 100088 bytes
  - Thrift: 100031 bytes

Evaluation

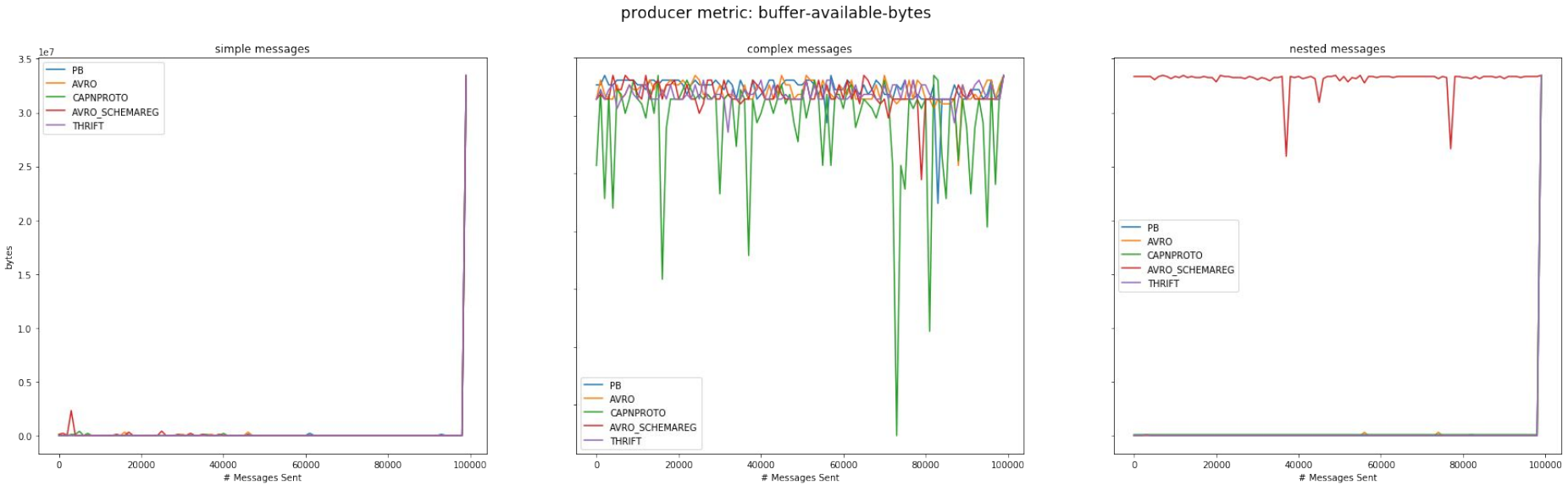
Raw Serialization Comparisons



Raw Deserialization Comparisons

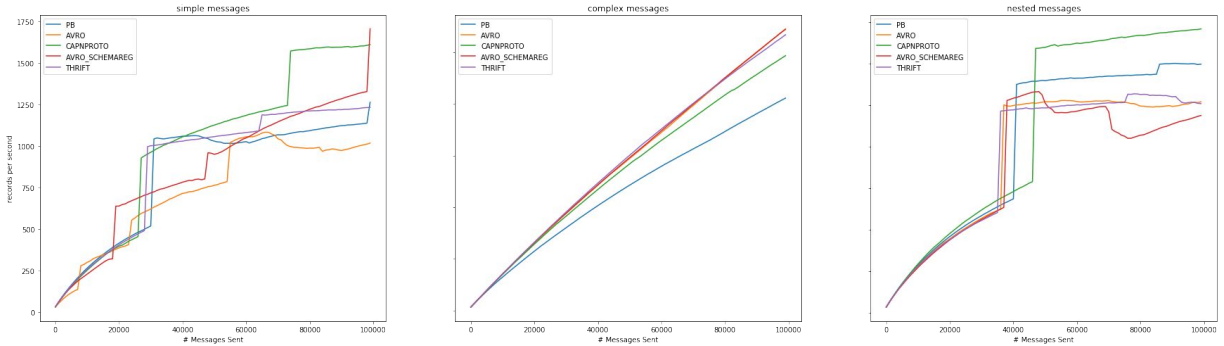


Kafka Producer Performance

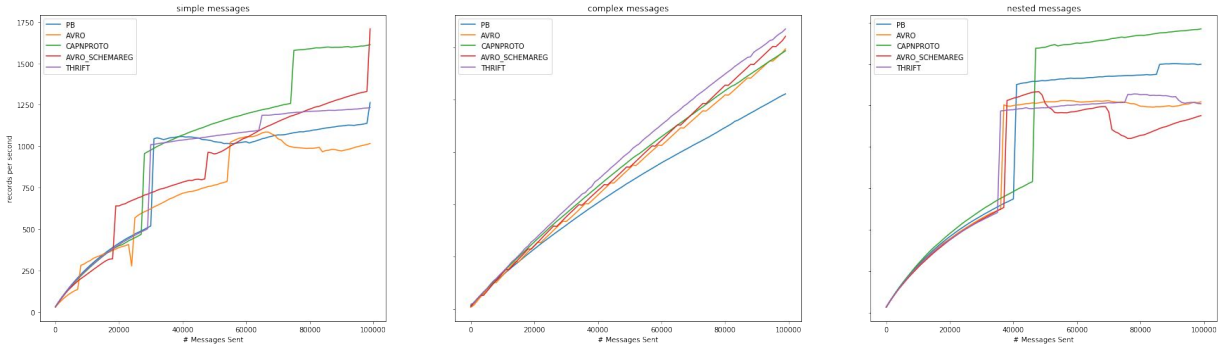


Producer Send Rate + Consumer Receive Rate

producer metric: record-send-rate



consumer metric: records-consumed-rate



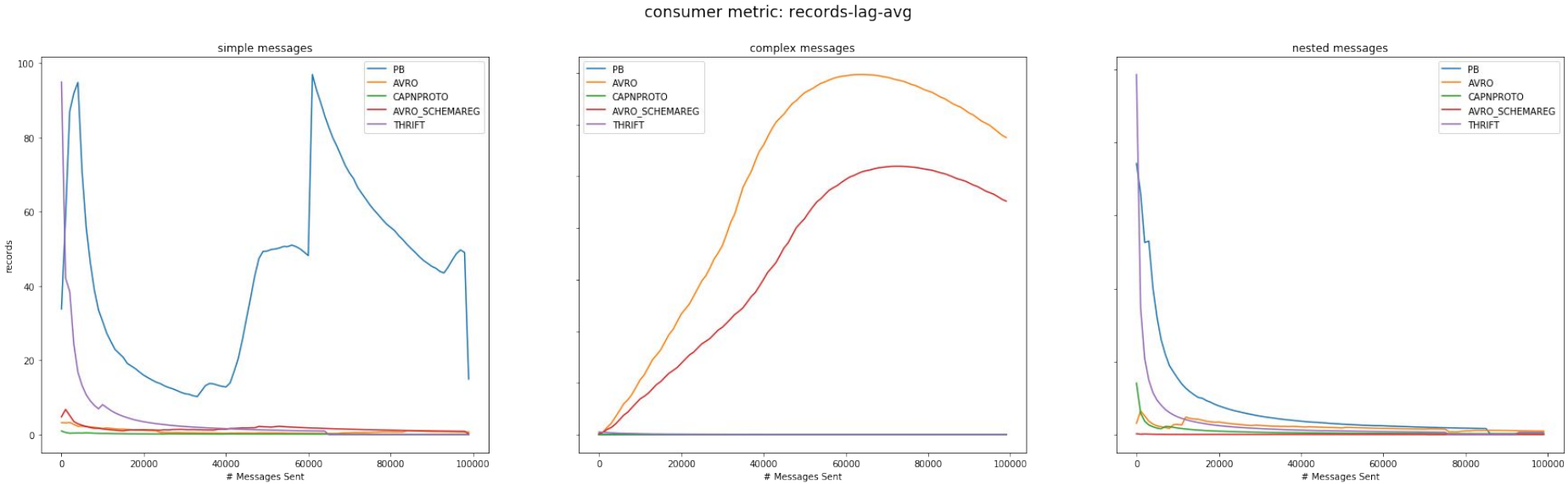
Future Work

- Profile more serializers
  - Kryo, Coffer, MessagePack, etc
- Profile Kafka Streams
- Profile Connect/KSQL
- Dynamically picking serializer
  - Base off data types/format

Questions?

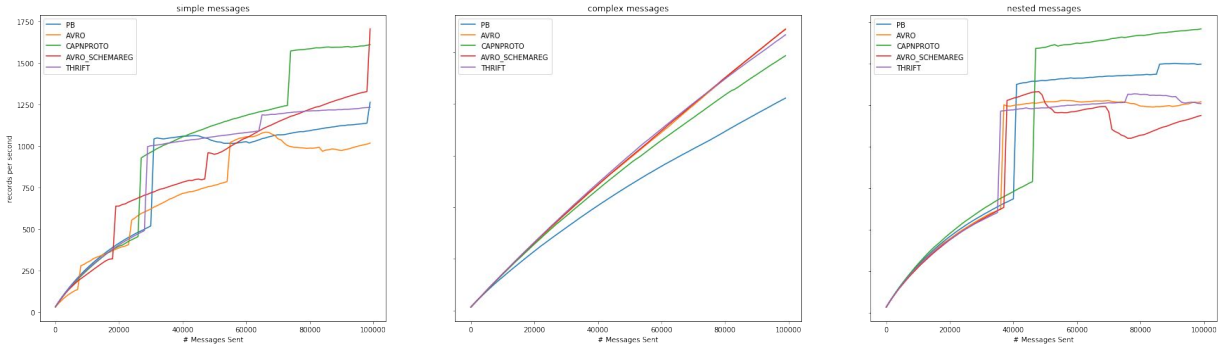
Lots of Graphs

Kafka Consumer Performance

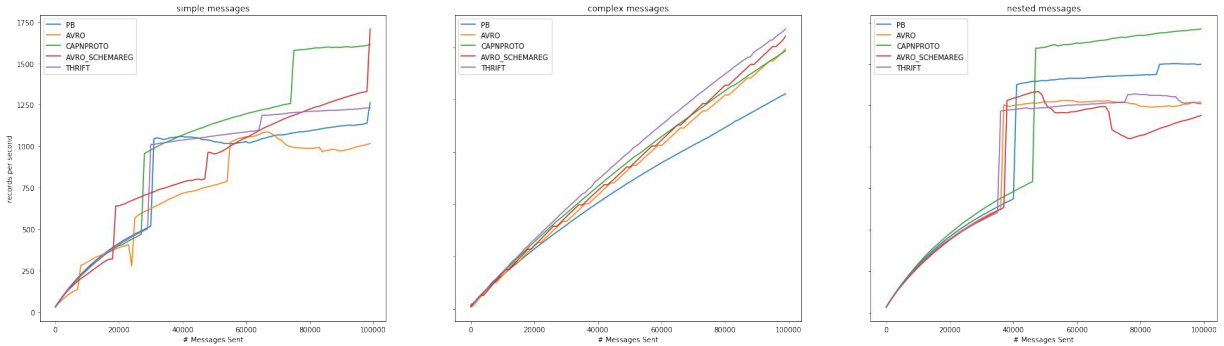


Producer Send Rate + Consumer Receive Rate

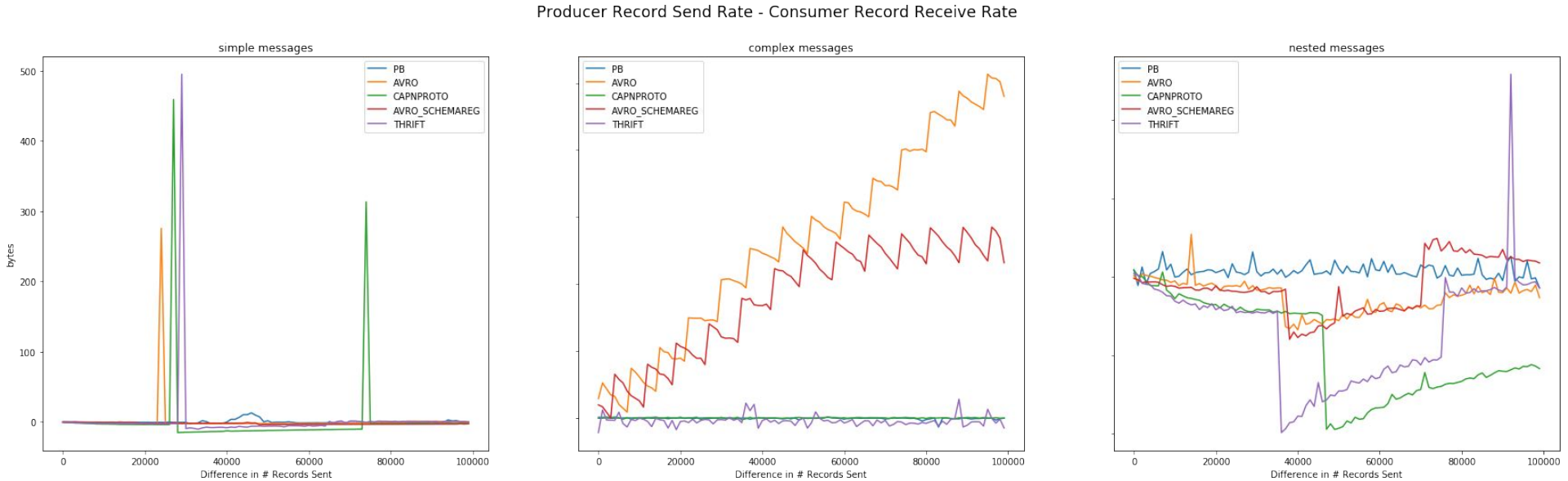
producer metric: record-send-rate



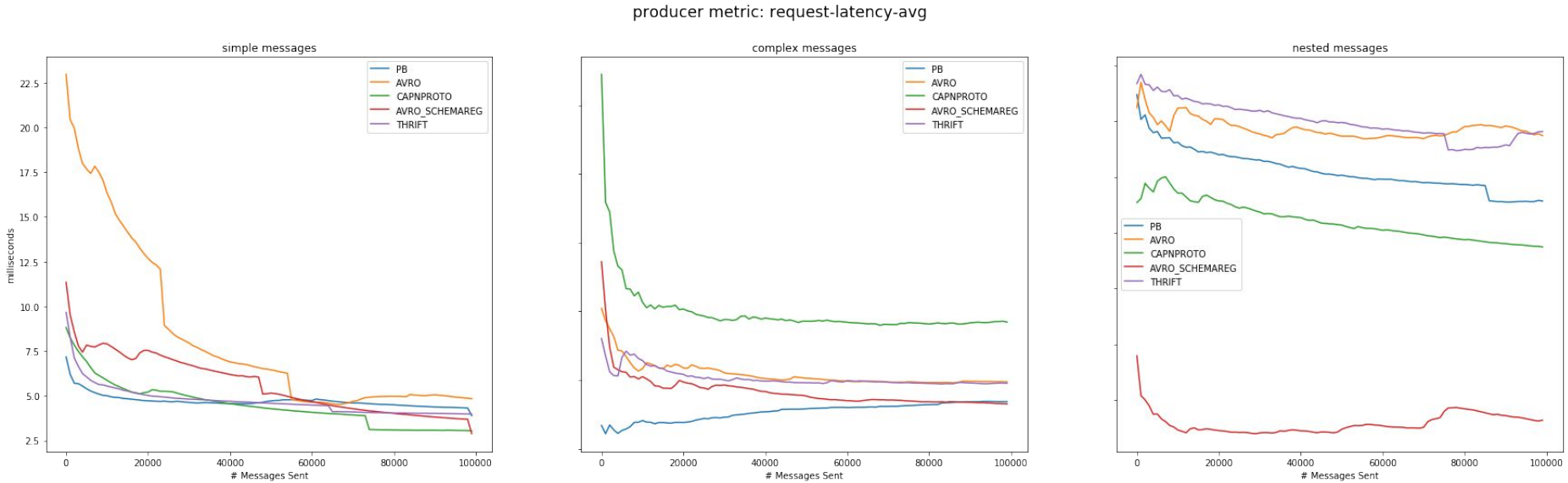
consumer metric: records-consumed-rate



Producer Send Rate + Consumer Receive Rate



Producer Request Latency



Consumer Request Rate