Intro to real-time streaming

STREAMING CONCEPTS

Mike Metzger
Data Engineer

What is 'real-time'?

- Definition varies depending on **context**
- Typically defines a **response timeframe**
- The response timeframe is defined as a sort of **guarantee**
- Could be:
  - 1 day
  - 1 hour
  - 1 minute

Real world example

Post office

- Different **classes** of service
- Delivery **timeframe varies based on** service **class**
- Only so much **capacity** for **faster** service
- Costs are **proportional** to service speed
- Service selection is up to the sender based on **options**

Relationship to streaming?

How does real-time relate to streaming data?

- Streaming processes are **limited** by available resources
  - How quickly can data be *transported*?
  - ... *processed*?
  - ... *delivered*?
  - How much does it *cost*?

Resources define implementation

- Helps **define** our **requirements** for streaming data processes
- **Speed** of transport
- Processing **latency**
- **Delivery**
- Data **storage**
- **Cost!**

Let's practice!

STREAMING CONCEPTS

Vertically scaling streaming systems

STREAMING CONCEPTS

Mike Metzger
Data Engineer

Why scale?

- Process the **same data in less time**
- Process **more data in the same time**
- Deliver data **more quickly** (reduce latency)
- Meet **guarantees** (SLAs)

Vertical scaling

- **Improve the capabilities** of a single system
- **Faster / better** components
  - CPU, RAM, Disk, Network
- All can affect streaming performance

Faster CPU / GPU performance

- **Faster** execution
- **Better** execution
  - New / improved instruction sets
- **GPU** processing
  - Machine learning
  - Deep learning
  - Image processing
  - Matrix operations

How does this affect streaming?

- Streaming processes **don't stop until complete**
- Different items can be in **different parts** of the pipeline, but total processing capacity is **limited by the system performance**
- Certain components have a **greater effect than others**, depending on workload
- **Benchmark / test!**

Let's practice!

STREAMING CONCEPTS

Horizontally scaling streaming systems

STREAMING CONCEPTS

Mike Metzger
Data Engineer

Horizontal scaling refresher

- Instead of scaling "up", **scale "out"**
- Typically means adding processing capability by **adding more**, rather than faster / better
- Works best with **embarrassingly parallel** situations
  - Tasks that can be split easily
  - E.g. processing a large group of non-interdependent images

Horizontal scaling with streaming

- Streaming data processing typically has **minimal delays**
- Can make transfer of data between workers **tricky**
- Best to process a full stream within a **single pipeline**
- Create **copies** of the pipelines

Pipeline copies

- As events occur, they initially **enter** a pipeline
- All tasks related to that process are **self-contained within the pipeline**, until completion
- Scale by **adding more pipelines**
- Can still **vertically scale** within a pipeline

Additional considerations

- **Other components** may be required
- Load **balancer** / director
  - Card dealer
  - Least busy node
- Eventually hit **bottlenecks**
  - Disk write performance
- Consider **shortening** streaming pipeline
  - Remove need to **immediately** process data

Let's practice!

STREAMING CONCEPTS

Streaming roadblocks

STREAMING CONCEPTS

Mike Metzger
Data Engineer

Scaling review

Vertical scaling - compute resources

- CPU
- RAM
- Disk (capacity and IO)
- Network

Horizontal scaling - more nodes

- Add machines as nodes / workers

Initial concerns

- **Compute resources**
  - Lack of adequate or **slow resources**
- **More nodes**
  - Requires more **connectivity**
  - Some form of **shared** resources
  - Added **complexity**
  - Usually some form of **cluster management**

Communication issues

Types of messaging problems:

- **Missing** messages
- **Delayed** messages
- **Out of order** messages
- **Repeat** messages

Missing messages

- Represent events that **never appear**
- Can be **difficult to detect**
- Sometimes handled with a **sequence identifier**
- Requesting the missing messages can **delay further responses**

Delayed messages

- Similar to missing messages
- May **cause issues** with the processing pipeline **due to delays**
- Often related to **system resource issues**

Out of order messages

- **Combination** of missing / delayed messages
- Results when an **older message appears after newer** ones
- Requires some measure of **sequence** or state to detect
- Handling these issues **depends on the type of data process** being run

Repeat messages

- Occurs when the **same message is sent multiple times** or resent due to systems issues
- Requires **sequence handling** to completely avoid, but might be safe to ignore
- **Sometimes is not an issue** (consider a temperature measurement)

Let's practice!

STREAMING CONCEPTS