TWO SIGMA

www.twosigma.com

Designing Resilient Data Pipelines

Andrew Bolin

March 27, 2019

Disclaimer

This document is being distributed for informational and educational purposes only and is not an offer to sell or the solicitation of an offer to buy any securities or other instruments. The information contained herein is not intended to provide, and should not be relied upon for, investment advice. The views expressed herein are not necessarily the views of Two Sigma Investments, LP or any of its affiliates (collectively, "Two Sigma"). Such views reflect the assumptions of the author(s) of the document and are subject to change without notice. The document may employ data derived from third-party sources. No representation is made by Two Sigma as to the accuracy of such information and the use of such information in no way implies an endorsement of the source of such information or its validity.

The copyrights and/or trademarks in some of the images, logos or other material used herein may be owned by entities other than Two Sigma. If so, such copyrights and/or trademarks are most likely owned by the entity that created the material and are used purely for identification and comment as fair use under international copyright and/or trademark laws. Use of such image, copyright or trademark does not imply any association with such organization (or endorsement of such organization) by Two Sigma, nor vice versa.

About Two Sigma

What are we going to cover today?

I. Pipelines Crash Course

II. Observability

III. Structure

IV. Pipelines as Code

V. Data Validation

Data Pipelines Crash Course

What is a data pipeline?

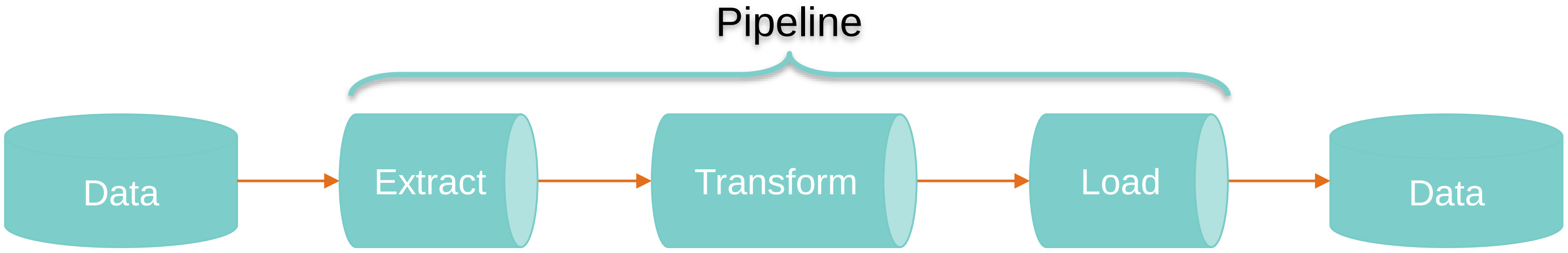
A collection of data processors with directional dependencies and no cycles

What does a data pipeline do?

Automates the process of bringing data from its origins to data consumers in a usable form

Data Pipelines Crash Course

Simple Pipeline Example:
ETL (Extract, Transform, Load)



[illegible]

[illegible]

Data Pipelines Crash Course

Real World Pipelines

Business Purpose	Frequency	Depends on Previous Runs?	Example SLO
Trading Day Analysis	Daily	Yes	Delivery X hours before market open
Research Environment Build	Weekly	No	New environment released biweekly
Trading Platform Integration Tests	On Demand	No	95% of pipelines complete within X hours

Common Concerns

How do I detect and understand failures in my pipelines?

How can I minimize the risk associated with changes to my pipelines?

How do I quickly recover from pipeline failures?

How do I know that the data I generate is good?

What are we going to cover today?

I. Pipelines Crash Course
II. Observability
III. Structure
IV. Pipelines as Code
V. Data Validation

Observability

Logging

Metrics Collection

Status Monitoring

Logging

What does logging give us?

- Logs provide a record of events
- Used to determine when and why something happened
  - Why did this pipeline task fail?
  - When was this pipeline scheduled?
  - Why was/wasn't this task launched?

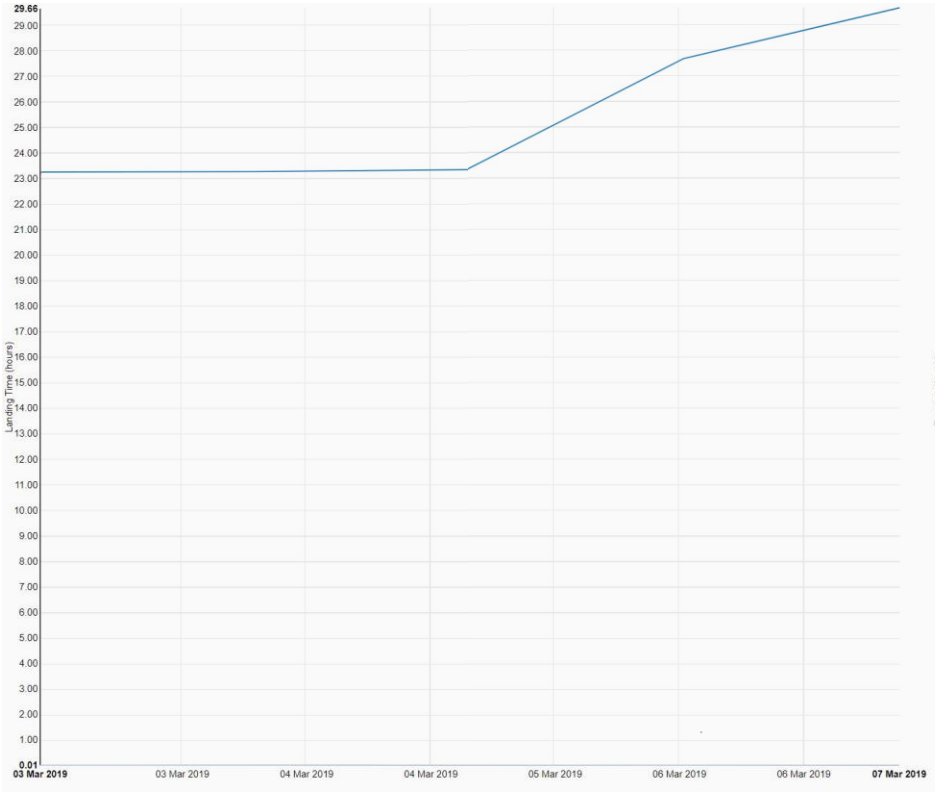
Pipeline Logging Checklist

- All of the processors in the data pipeline produce logs
- The pipeline scheduler or workflow manager produces logs
- Logs are available after batch pipelines have completed their execution

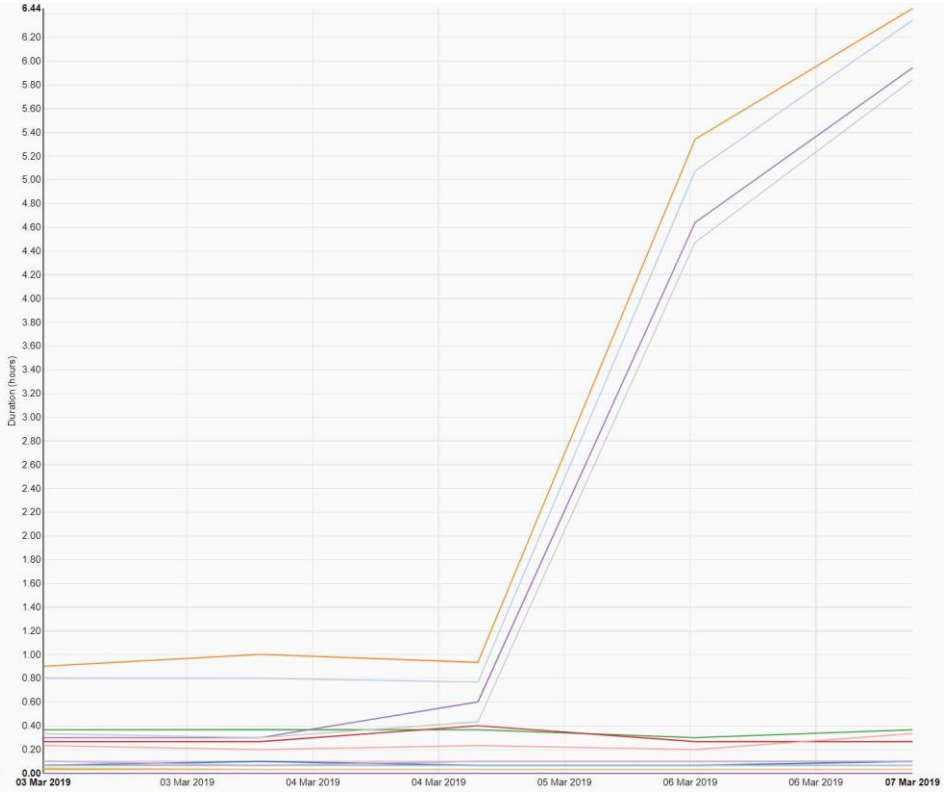
Metrics Collection

- What type of metrics to collect?
  - Latency
  - Queued time
  - Resource usage
- …and at what level to collect them?
  - Individual task level
  - Pipeline level

Pipeline Latency



Task Latency



Status Monitoring

- Monitor statuses at both the task and pipeline level
- Monitor status over time and across pipeline instances
- Make use of states
  - Running
  - Failed
  - Queued
  - Blocked on Concurrency
  - Canceled

What are we going to cover today?

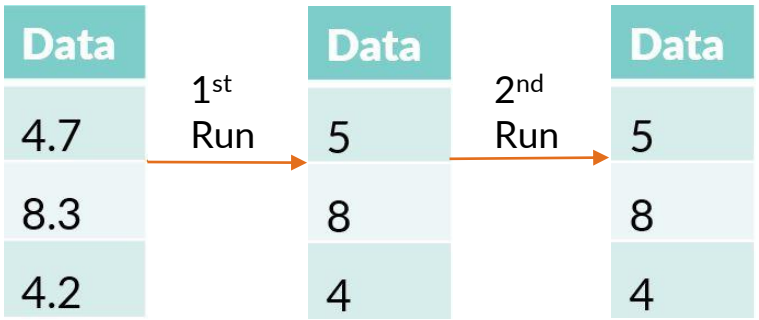
I. Pipelines Crash Course
II. Observability
III. Structure
IV. Pipelines as Code
V. Data Validation

Idempotent Tasks

- A task is idempotent if, given the same inputs, it can be rerun without producing different results

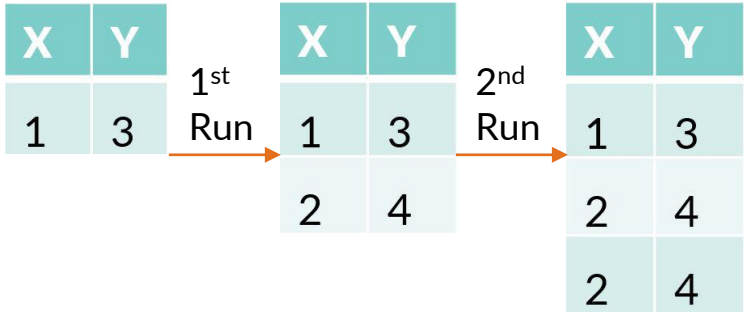
Idempotent

Rounding db entries to nearest integer



Not Idempotent

Appending rows to a csv



Immutable Inputs

What is immutable data?

- Data is immutable if it cannot be modified after it is created

Why use immutable inputs?

- Ensures reproducibility when combined with idempotent tasks

Ways to ensure immutability?

- Version your data
- Use immutable data stores

Benefits of Idempotency and Immutability

- Simplifies testing of tasks and pipelines
- Allows tasks to be retried without unintended effects
  - Reduces instances of operator error
  - Facilitates automatic retries
  - Facilitates checkpointing

What are we going to cover today?

I. Pipelines Crash Course
II. Observability
III. Structure
IV. Pipelines as Code
V. Data Validation

Pipelines as Code

What does Pipelines as Code mean?

- Defining pipelines in configuration or code

How is that accomplished?

- Pipelines can be expressed as Directed Acyclic Graphs (DAGs)
- DAGs can be encoded in declarative configuration
- DAGs can be generated using a general purpose programming language

Declarative Pipeline Example

```
# General
# General
name: 'Extract'
id: 1
description: 'Extract data from source db'

# Execution
hostGroup: batch
runAs: 'abolin'
maxRun: 1h
command: 'extract.py'
maxAutoRetry: 3
autoRetryOn: [error]
```

```
# Tasks
    # Dependencies
    prerequisites: []

# Alerting
    email: [abolin@twosigma.com]
    warn: 30m
    alertOn: [error, failure]

    # Resources
    cpus: 1
    memory: 1024

warn: 4h
```

Benefits of Pipelines as Code

- Storing pipelines in source control
  - Code review of pipeline updates
  - History of changes
  - Possible to keep the pipeline alongside the code it executes
- Static analysis
- Supports automation of pipeline deployments
- Regenerating historical pipeline outputs is possible with versioned pipelines, idempotent tasks, and immutable inputs

What are we going to cover today?

I. Pipelines Crash Course
II. Observability
III. Structure
IV. Pipelines as Code
V. Data Validation

Usable Data

What is usable data?

- Data that can be used for its intended business purpose

What causes data to be unusable?

- Does not conform to the format required by consumers
- Is wrong or impossible
- Is incomplete

Price	Currency
One Hundred Forty	$
-140.01	$
140.01	-

Data Validation

What is data validation?

- The process of ensuring data is usable through the use of data validators

What are data validators?

- A data processor which verifies or tests the properties of its input data
  - Data type validation
  - Constraint validation
  - Quantitative validation

Time	Price
9:30	97.01
9:31	97.05
9:32	97.03

Time	Price
9:30	97.01
9:31	45.03
9:32	180.45

Pros and Cons of Data Validation

Pros

- Provides guarantees on the usability of the data
- Can detect soft failures in data pipelines
- Can prevent execution of downstream tasks using invalid data

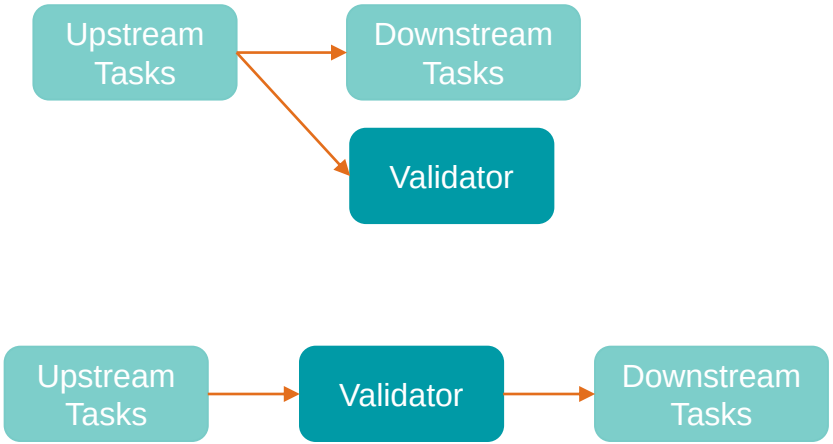
Cons

- Can increase pipeline latency
- Flaky validators introduce instability

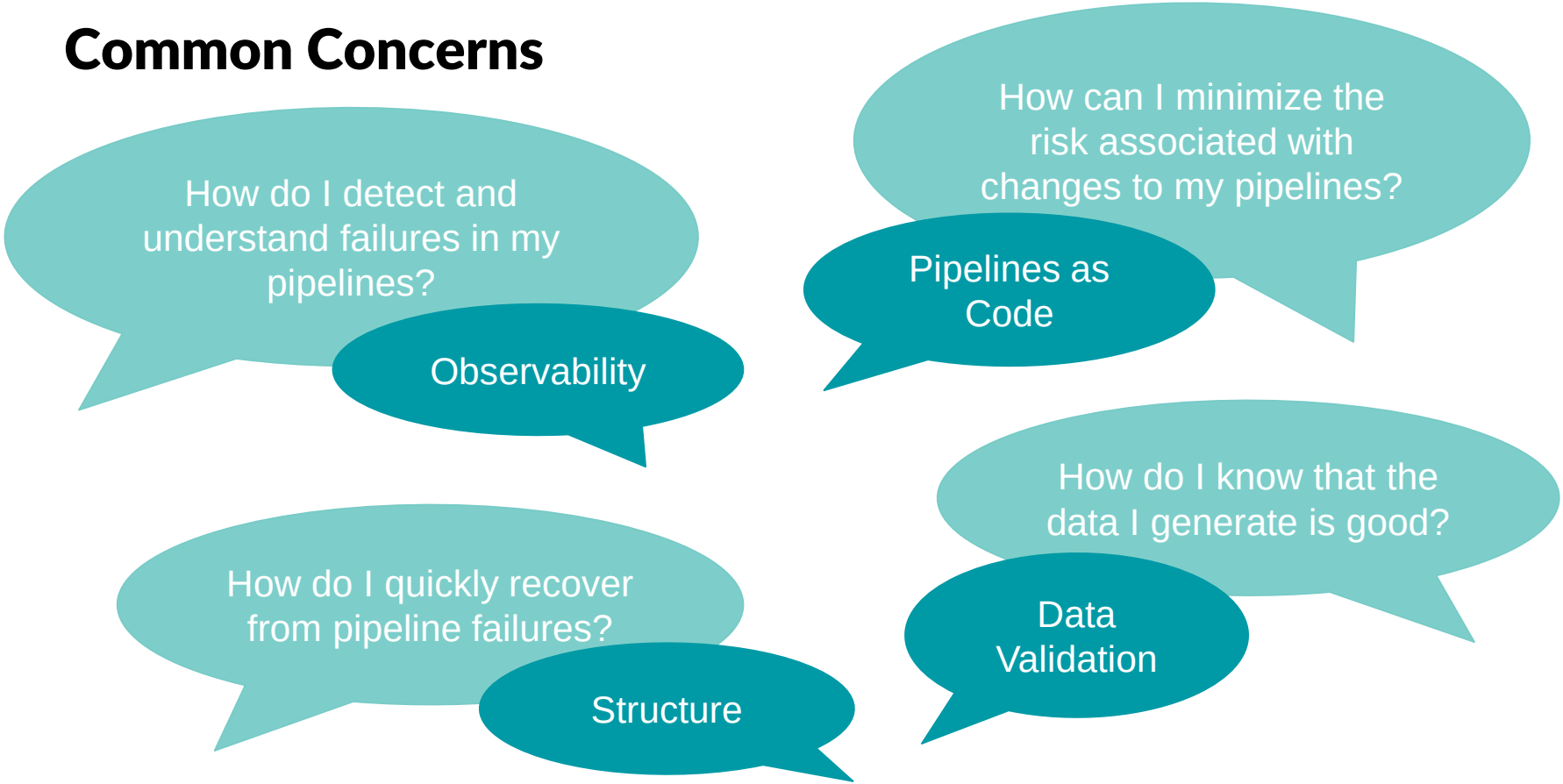
Adding Validators to Data Pipelines

Quarantine Method

- New validators are always added as leaf nodes to the data pipeline
- After observing the validators over several execution cycles:
  - Continue running the validator outside of the critical path
  - Move the validator into the critical path of the pipeline
  - Remove the validator



Common Concerns



Thanks for Attending!

If you have any questions

- I'll be available right outside the hall right after this talk
- Later you can find me at the Two Sigma booth