Migration from Redshift to Spark

Sky Yin
Data scientist @ Stitch Fix

SPARK SUMMIT EAST 2017

What's Stitch Fix?

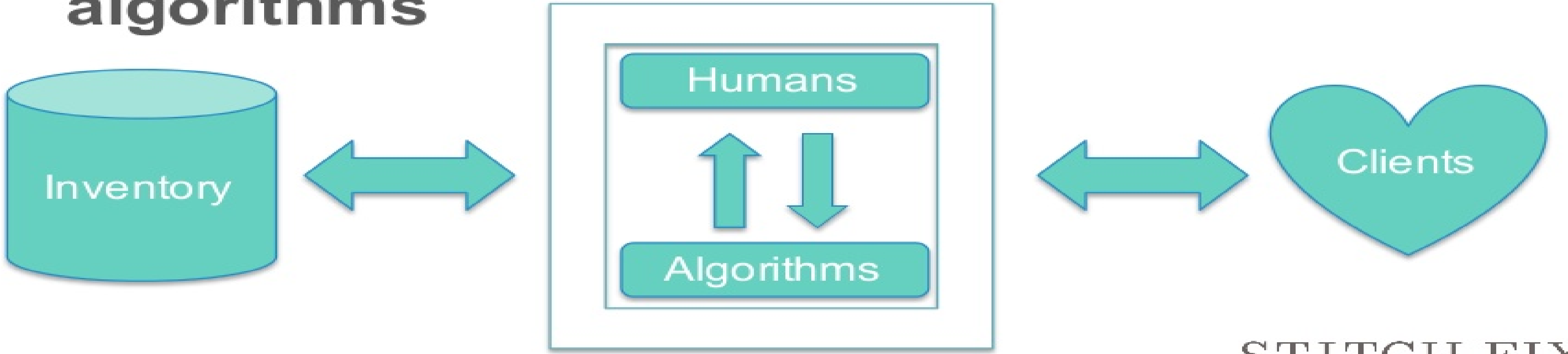
- An online personal clothes styling service since 2011

SPARK SUMMIT EAST 2017

STITCH FIX

A loop of recommendation

- We recommend clothes through a combination of human stylists and algorithms



STITCH FIX

What do I do?

- As part of a ~80 people data team, I work on inventory data infrastructure and analysis
- The biggest table I manage adds ~500-800M rows per day

SPARK SUMMIT EAST 2017

STITCH FIX

Data infrastructure: computation

- Data science jobs are mostly done in Python or R, and later deployed through Docker

Data infrastructure: computation

- Data science jobs are mostly done in Python or R, and later deployed through Docker
  - Gap between data scientist daily work flow and production deployment
  - Not every job requires big data

Data infrastructure: computation

- Data science jobs are mostly done in Python or R, and later deployed through Docker
- As business grows, Redshift was brought in to help extract relevant data

Redshift: the good parts

- Can be very fast
- Familiar interface: SQL
- Managed by AWS
- Can scale up and down on demand
- Cost effective*

Redshift: the troubles

- Congestion: too many people running queries in the morning (~80 people data team)

Redshift: the troubles

- Congestion: too many people running queries in the morning
  - Scale up and down on demand?

Redshift: the troubles

- Congestion: too many people running queries in the morning
  - Scale up and down on demand?
  - Scalable =/= easy to scale

Redshift: the troubles

- Congestion: too many people running queries in the morning
- **Production pipelines and exploratory queries share the same cluster**

Redshift: the troubles

- Congestion: too many people running queries in the morning
- Production pipelines and ad-hoc queries share the same cluster
  - Yet another cluster to isolate dev from prod?

Redshift: the troubles

- Congestion: too many people running queries in the morning
- Production pipelines and dev queries share the same cluster
  - Yet another cluster to isolate dev from prod?
    - Sync between two clusters
    - Will hit scalability problem again (even on prod)

Redshift: the troubles

- Congestion: too many people running queries in the morning
- Production pipelines and ad-hoc queries share the same cluster
- There's no isolation between computation and storage
  - Can't scale computation without paying for storage
  - Can't scale storage without paying for unneeded CPUs

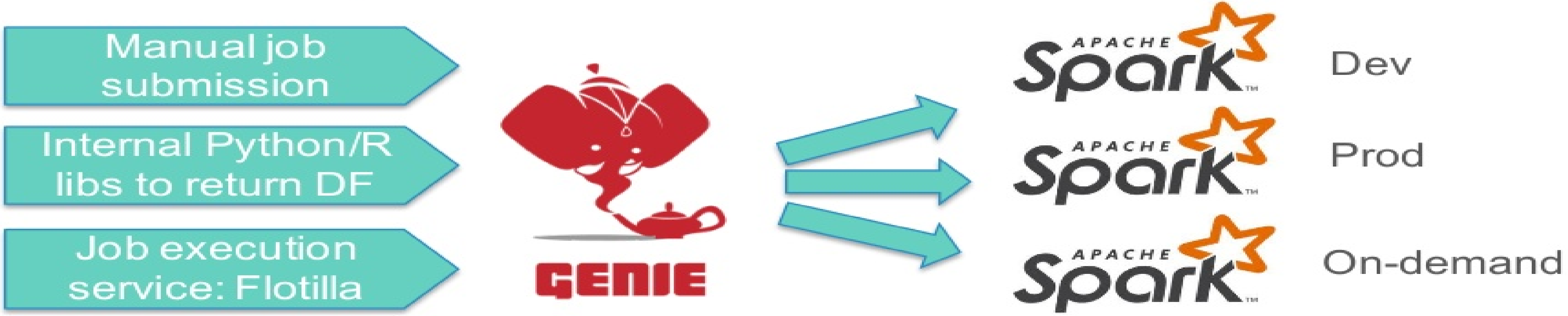
Data Infrastructure: storage

- Single source of truth: S3, not Redshift
- Compatible interface: Hive metastore
- With the foundation in storage, the transition from Redshift to Spark is much easier

S3 + HIVE

Journey to Spark

- Netflix Genie as a "job server" on top of a group of Spark clusters in EMR



Journey to Spark: SQL

- Heavy on SparkSQL with a mix of PySpark
- Lower migration cost and gentle learning curve
- Data storage layout is transparent to users
  - Every write we create a new folder with timestamp as batch_id, to avoid consistency problem in S3

Journey to Spark: SQL

- Difference in functions and syntax
  - Redshift

```sql
SELECT number :: float, BTRIM(string)
FROM foo
JOIN bar USING (buzz)
```

  - SparkSQL

```sql
SELECT CAST(number AS double), TRIM(string)
FROM foo
JOIN bar ON foo.buzz = bar.buzz
```

Journey to Spark: SQL

- Difference in functions and syntax
- Partition
  - In Redshift, it's defined by `dist_key` in schema
  - In Spark, the term is a bit confusing
    - There's another kind of partition on storage level
    - In our case, it's defined in Hive metastore
      - `S3://bucket/table/country=US/year=2012/`

Journey to Spark: SQL

- Functions and syntax
- Partition
- SparkSQL

```
SET spark.sql.shuffle.partitions = 2
SELECT * FROM dataframe DISTRIBUTE BY key
```

  – DataFrame API

```
df.repartition(10, "column")
```

Journey to Spark: SQL

- Functions and syntax
- Partition
- **Sorting**
  - Redshift defines `sort_key` in schema
  - Spark can specify that in runtime

```
SELECT * FROM dataframe SORT BY key
```

OR

```
df.sortWithinPartitions(”column”)
```

Journey to Spark: Performance

- There's a price in performance if we naively treat Spark as yet another SQL data warehouse

SPARK SUMMIT EAST 2017

STITCH FIX

Journey to Spark: Performance

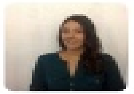
- Key difference: Spark is an in-memory computing platform while Redshift is not

Journey to Spark: Performance

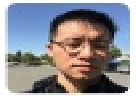
- Key difference: Spark is an in-memory computing platform while Redshift is not
- FAQ #2: “Why is my Spark job so slow?”

Journey to Spark: Performance

- Key difference: Spark is an in-memory computing platform while Redshift is not
- FAQ #2: “Why is my Spark job so slow?”
  - Aggressive caching



sonyaberg 10:36 PM
@sky i got a query down from about 2 hours to 4 minutes by simply caching the temp tables. thanks!
2 ! 2 1



sky 10:42 PM
awesome 30x speed boost!

Journey to Spark: Performance

- Key difference: Spark is an in-memory computing platform while Redshift is not
- FAQ #2: "Why is my Spark job so slow?"
  - Aggressive cache
  - Use partition and filtering
  - Detect data skew
  - Small file problem in S3

Journey to Spark: Performance

- FAQ #1: "Why did my job fail? Hmm, the log says some executors were killed by YARN???"

Journey to Spark: Performance

- FAQ #1: “Why did my job fail?”
  - Executor memory and GC tuning
    - Adding more memory doesn’t always work

Journey to Spark: Performance

- FAQ #1: “Why did my job fail? “
  - Executor memory and GC tuning
  - Data skew in shuffling
    - Repartition and salting

Journey to Spark: Performance

- FAQ #1: "Why did my job fail? "
  - Executor memory and GC tuning
  - Data skew in shuffling
    - Repartition and salting
    - `spark.sql.shuffle.partitions`
      - On big data, 200 is too small: 2G limit on shuffle partition
      - On small data, 200 is too big: small file problem in S3

Journey to Spark: Performance

- FAQ #1: "Why did my job fail? "
  - Executor memory and GC tuning
  - Data skew in shuffling
  - Broadcasting a large table unfortunately
    - Missing table statistics in Hive metastore

Journey to Spark: Productivity

- SQL, as a language, doesn't scale well
  - Nested structure in sub-queries
  - Lack of tools to manage complexity

Journey to Spark: Productivity

- SQL, as a language, doesn't scale well
- **CTE: one trick pony**

```sql
WITH
    A AS (...)
    B AS (...)
SELECT *
FROM A JOIN B ON ...
```

Journey to Spark: Productivity

- SQL, as a language, doesn't scale well
- CTE: one trick pony

```
WITH
    A AS (...)
    B AS (...)
SELECT *
FROM A JOIN B ON ...
```

=> Spark temp tables

Journey to Spark: Productivity

- Inspiration from data flow languages
  - Spark DataFrames API
  - dplyr in R
  - Pandas pipe
- Need for a higher level abstraction

Journey to Spark: Productivity

- `Dataset.transform()` to chain functions

```
def withActiveClient(df: DataFrame): DataFrame = {
  // complicated logic to define active client
}

def withLTV(df: DataFrame): DataFrame = {
  // complicated logic to calculate life time value
}

val activeClientLTV = client
    .transform(withActiveClient)
    .transform(withLTV)
```

Thank You.

Contact me @piggybox or syin@stitchfix.com

SPARK SUMMIT EAST 2017