SILICON VALLEY
DATA SCIENCE

WHAT'S NEW IN SPARK 2.0:
STRUCTURED STREAMING AND DATASETS

Andrew Ray

StampedeCon 2016

Silicon Valley Data Science is a boutique consulting firm focused on transforming your business through data science and engineering.

ANDREW RAY

@collegeisfun

- Contributor to Apache Spark
- Hadoop ecosystem guru
- Senior Data Engineer @ SVDS
- Prev. Data Sci. @ Walmart
- PhD Mathematics @ UNL
- From St. Louis

AGENDA

- Spark Refresher
  - RDD
  - DataFrame
  - Streaming
- What's New in 2.0
  - Datasets
  - Structured Streaming


Apache Spark
REFRESHER
@SVDataScience

RDD

- Redundant Distibuted Dataset
- Collection of typed objects
- Low level API
  - map
  - flatMap
  - reduceByKey
  - Etc …
- Lazy evaluation

WORD COUNT: RDD

```
val lines = sc.textFile("/a/dir/of/files/")

val counts = lines.flatMap(_.split(" "))
                  .map(x => (x, 1))
                  .reduceByKey(_ + _)

counts.take(10)
```


DATAFRAME

- Collection of tabular data
- Named columns with specified data types
- Higher level API
- Mix with SQL
- Operations not checked until analysis.

WORD COUNT: DATAFRAME

```
val lines = sqlCtx.read.text("/a/dir/of/files/”)

val counts = lines.select(
                  explode(split($"value"," ")).as("word")
                )
                .groupBy("word")
                .count()

counts.show()
```


SPARK STREAMING

- Micro batch
- RDD for values in each iteration

WORD COUNT: STREAMING

```
val ssc = new StreamingContext(sc, Seconds(5))
val lines = ssc.textFileStream("/a/dir/of/files/")

val counts = lines.flatMap(_.split(" "))
                  .map(x => (x, 1))
                  .reduceByKey(_ + _)

counts.print()
ssc.start()
```


WORD COUNT: STREAMING (FIXED)

```
val ssc = new StreamingContext(sc, Seconds(5))
ssc.checkpoint("/somewhere/durable/")
val lines = ssc.textFileStream("/a/dir/of/files/")
val counts = lines.flatMap(_.split(" "))
                  .map(x => (x, 1))
                  .updateStateByKey {
                  (values: Seq[Int], state: Option[Int]) =>
                    Some(values.sum + state.getOrElse(0))
                  }
counts.print()
ssc.start()
```



Spark 2.0
WHAT'S NEW
@SVDataScience

NEW IN 2.0

- Stable Dataset API
- Alpha of Structured Streaming
- Some other stuff
- 1000's of commits

SPARK 2.0 MIGRATION

- New entry point – SparkSession
  - Replaces `SparkContext` and `SQLContext`
  - In shell: `spark`
- `type DataFrame = Dataset[Row]`
  - Java code change: `DataFrame` => `Dataset<Row>`
- Default build is with Scala 2.11


PART I
Dataset

DATASET

- Collection of typed objects
  - Primitives
  - Scala case class
  - Java bean class
- Use Spark Encoders
  - Operate on without deserializing to objects
- Compile time correctness checks
- Optimized by Catalyst

WORD COUNT: DATASET

```
val lines = spark.read.textFile("/a/dir/of/files/")

val counts = lines.flatMap(_.split(" "))
                  .groupByKey(identity)
                  .count()

counts.show()
```


DATASET NOTES

- Two sets of methods
  - Typed (return Dataset)
    - groupByKey
  - Untyped (return DataFrame)
    - groupBy
- Easy to convert DataFrame to Dataset of your object
  - df.as[Person]
  - df.as(Encoders.bean(Person.class))
- Python and R only have DataFrame

DATASET SECOND EXAMPLE

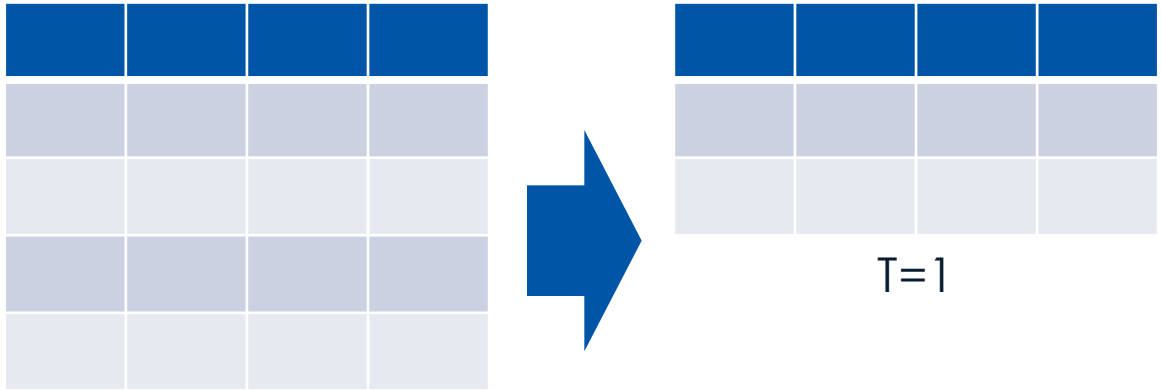
```
case class Person(name: String, age: Option[Long])

val path = "examples/src/main/resources/people.json"
val people = spark.read.json(path).as[Person]

def toId(p: Person): String = p.name + p.age.getOrElse(99)

val ids = people.map(toId)

ids.show()
```



PART II
Structured
Streaming

STRUCTURED STREAMING

- Extension of DataFrame/Dataset to streaming
- Input is unbounded append only table



WORD COUNT: STRUCTURED STREAMING

```
val df = spark.readStream.text("/a/dir/of/files/")

val counts = df.as[String]
               .flatMap(_.split(" "))
               .groupBy("value")
               .count()

val query = counts.writeStream
                  .outputMode("complete")
                  .format("console")
                  .start()
```

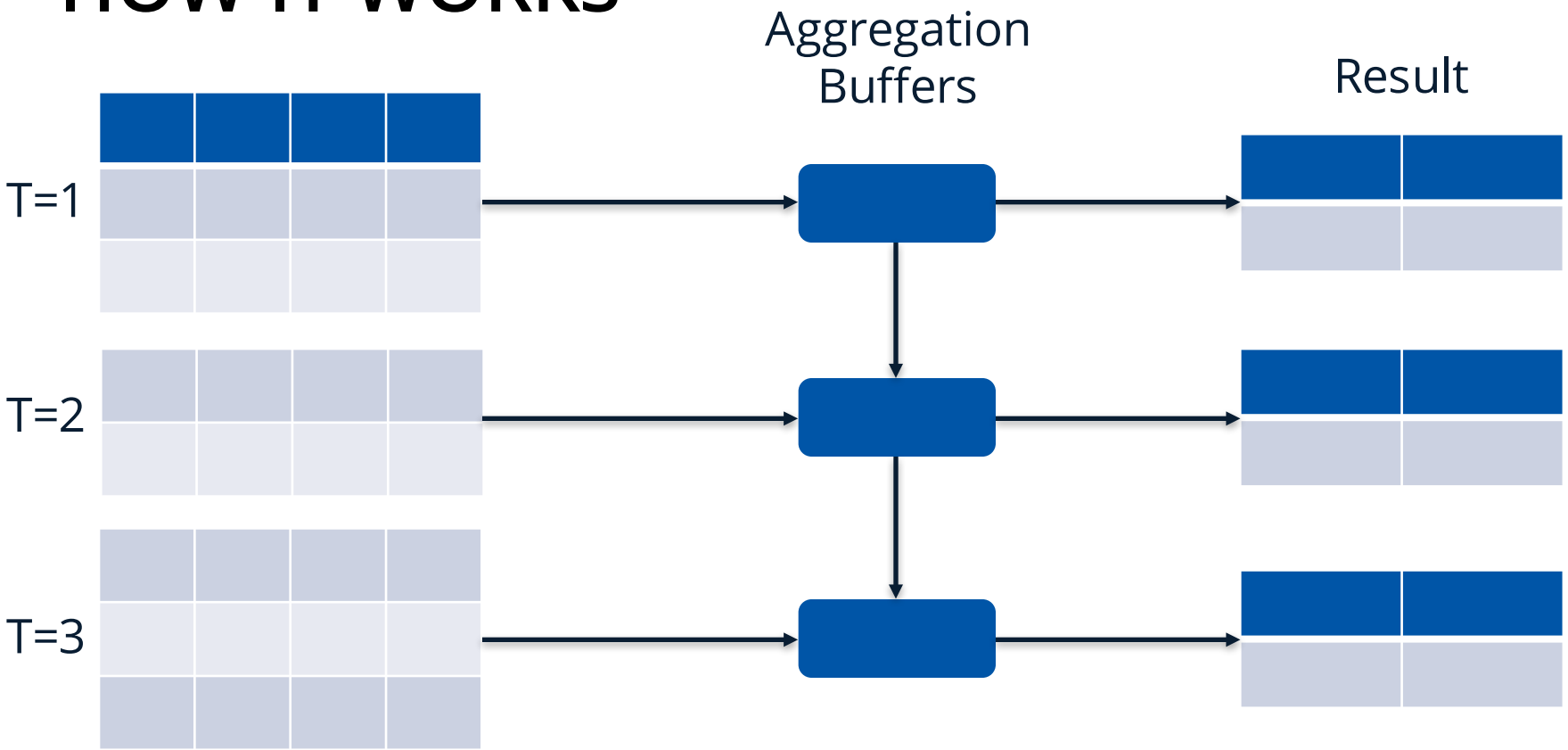

WORD COUNT: BATCH

```
val df = spark.readStream.text("/a/dir/of/files/")

val counts = df.as[String]
               .flatMap(_.split(" "))
               .groupBy("value")
               .count()

val query = counts.writeStream
                  .outputMode("complete")
                  .format("console")
                  .start()

counts.show()
```


NOTES

- Streams are DataFrames/Datasets
- Enables code reuse between batch and streaming
- No schema inferance
- Limitations enforced at Analysis
  - Aggregation chains
  - Distinct operations
  - Some outer joins to static datasets
  - Joins of streams
- Batch duration optional

HOW IT WORKS



WINDOWS

- Don’t need to be a multiple of the batch duration
- Not just *processing time*
- Possible to do *event time* windows
- Just another column

EVENT TIME WINDOW

```
import org.apache.spark.sql.types.StructType
val schema = new StructType().add("url", "string")
                             .add("event_time", "timestamp")
val events = spark.readStream.schema(schema).json("events/")

val counts = events
    .groupBy($"url", window($"event_time", "1 hour").as("w"))
    .count()
    .orderBy($"w", $"count".desc)

val query = counts.writeStream.outputMode("complete")
    .format("console").start()
```

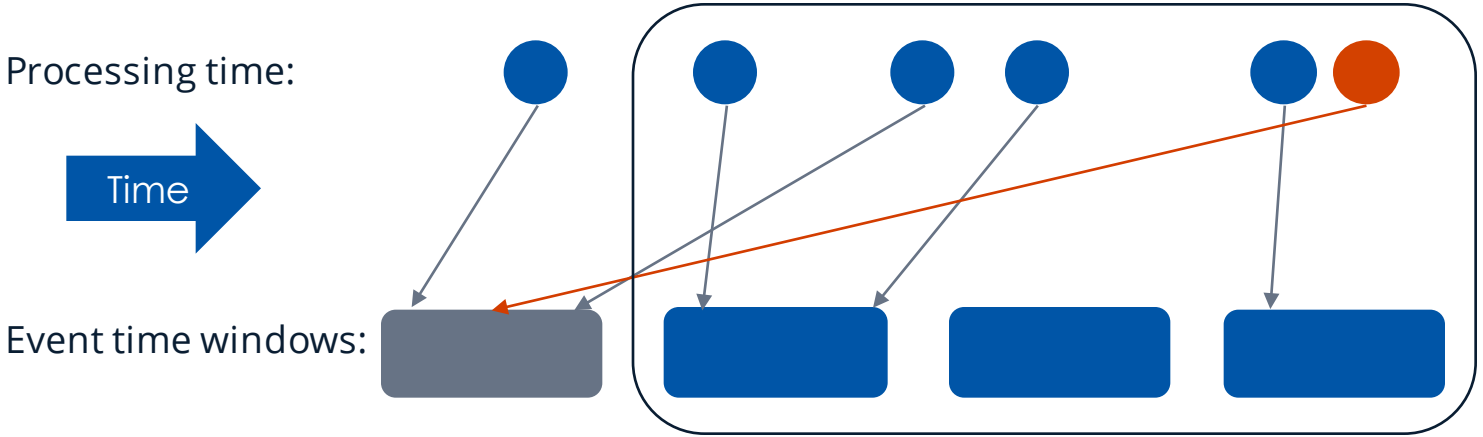

INPUT

```
{"url":"google.com", "event_time":"2016-07-21 23:38:04"}
{"url":"google.com", "event_time":"2016-07-21 23:44:04"}
{"url":"google.com", "event_time":"2016-07-21 22:27:04"}
{"url":"yahoo.com", "event_time":"2016-07-21 23:10:04"}
...
```


DISCARD DELAY

(not implemented yet)

- Discard highly delayed events
- Will help limit active state to bounded size



SOURCE & SINK OPTIONS

Currently very limited

- Source
  - Files
  - Socket
- Sink
  - Parquet – append output mode only
  - For each – custom code
  - Console
  - Memory

Don’t use structured streaming in production

RESOURCES

Spark Docs

- spark.apache.org/docs/latest/

Spark Examples

- github.com/apache/spark/tree/master/examples

Structured Streaming Umbrella JIRA

- issues.apache.org/jira/browse/SPARK-8360

SILICON VALLEY
DATA SCIENCE

Yes, we're hiring!
info@svds.com

THANK YOU

Andrew Ray
@collegeisfun
andrew@svds.com