CS 378 – Big Data Programming

Lecture 12

User Sessions from Logs

Review

- Assignment 5 – Avro Objects

- We'll look at implementation details of:
  - Mapper
  - Combiner
    - Should we use one? Can we use one?
  - Reducer
  - Avro generated Java code

Other Issues

- File upload issues with IE?
- Running MRUnit tests with Avro objects
  - Codehaus jackson version consistency
    - jackson-mapper-asl
    - jackson-core-asl
  - Avro serialization
- “shaded” JAR file – all dependencies included
  - Except Hadoop JAR – Why?

Review - Design Pattern

- Structured to hierarchical design pattern

- Data sources linked by some foreign key
- Data is structured and row based
  - For example, from databases
- Data is semi-structured and event based
  - Web logs

Sessionizing Web Logs

- Create user sessions from individual web log entries

- Represents all the actions by a user
- Allows later analysis to “replay” the user actions

- Collect measures and metrics about user behavior
  - Pages viewed, time on page, clicks
  - Path through the site, entry to the site (from a search engine?)

Sessionizing Web Logs

- To start (this or any “big data” application)
- We need to understand the data
  - Fields, values
  - Data size

- We need to define our goal
  - What do we want to end up with

Web Logs

- Let's look at some data
- These web logs have been processed from raw Apache logs
  - So they already have some structure
  - Parameter/value pairs
  - Easily parsed (lots of work has been done for us)

Web Logs

- Our goal is to aggregate user actions into sessions, so we can better understand
  - User behavior
  - The impact changes have on user behavior

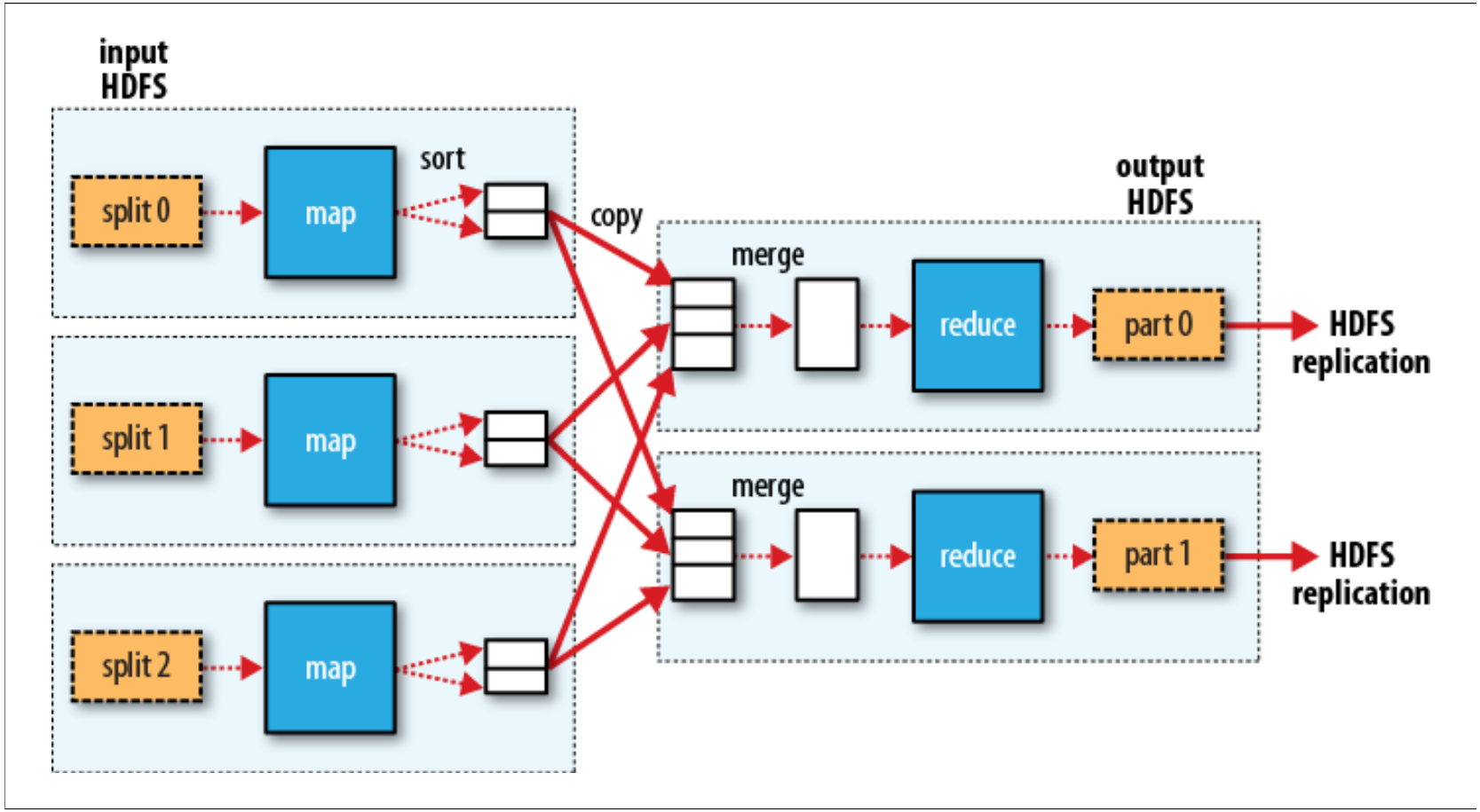
- So what should a session look like?

User Session

- Data about the session as a whole
- List of pages viewed, actions taken
  - Ordered in time
- In our logs, what data is session-wide
- What data is impression/action specific

MapReduce in Hadoop

Figure 2.4, Hadoop - The Definitive Guide



Assignment 6

- Define an Avro object for user session
  - One user session for each unique userID/apikey
  - Session will include an array of impressions
  - Impressions ordered by timestamp
  - Each impression will contain an array of IDs (0 or more)

- Identify data associated with the session as a whole
- Identify data associated with individual impressions
- Include all the fields listed in the assignment
- Create enums where requested

Recommendations

- Run WordCount on dataSet6.txt – see what's in it
- Build your log entry parser and test it
  - Return a Map indexed by parameter name, with value being the parameter value
  - You can use this parser on other log types in the future
- Get you app working with just a few fields populated
  - Session with no impressions
  - Add impressions, but just the fields in the provided schema
  - Extend the schema and compile it
  - Then populate the new field(s) in your mapReduce code
- Write some unit tests as you go