Putting A Spark in Web Apps

Apache Big Data, Seville ES, 2016

David Fallside

Intro

- Web app development has moved from Java etc to Node.js and JavaScript
- JS environment relatively simple and very rich, “npm install” provides access to world’s largest OSS repo
- Node.js environment scales very well “horizontally” but large compute requirements offloaded to back-end engines
- Apache Spark is an ideal engine:
  - Optimized for multiple types of compute tasks: coding, SQL, streaming, ML, graph manipulation
  - All available in single environment, reduces TCO
  - Fast: parallelised operators and in-memory policies, scalable from laptop to large distributed clusters
- Native APIs for Scala, Java, Python, (R) but not JavaScript …. and so EclairJS

Web App Requirements

- Simplicity
  - “npm install eclairjs” and “require(‘eclairjs’);”
  - Uniform coding style
  - Separation of servers
- Enable Spark functionality: SQL, streaming, ML, graph
- Performance
  - In vs out of JVM
  - Support interactive applications

Getting Started Demo

Code Semantics

```
var rdd = sparkSession.read().textFile(file).rdd();

var rdd2 = rdd.flatMap(function(sentence) {
  return sentence.split(" ");
});

var rdd3 = rdd2.filter(function(word) {
  return word.trim().length > 0;
});

var rdd4 = rdd3.mapToPair(function(word, Tuple2) {
  return new Tuple2(word.toLowerCase(), 1);
}, [spark.Tuple2]);

var rdd5 = rdd4.reduceByKey(function(value1, value2) {
  return value1 + value2;
});

var rdd6 = rdd5.mapToPair(function(tuple, Tuple2) {
  return new Tuple2(tuple._2() + 0.0, tuple._1());
}, [spark.Tuple2]);

var rdd7 = rdd6.sortByKey(false);

rdd7.take(10).then(function(val) {
  console.log("Success:", val);
  stop();
}).catch(stop);
```

immutable

Spark operator

lambda function

action vs transformation

Spark class, passed arg

Word Count As A Web App

EclairJS & Web App

Browser Script

```
function clickme() {
  if (ws) {
    // First clear old results then start count
    ws.send(JSON.stringify({startCount: true}));
    …

window.onload = function() {
  ws = new WebSocket("ws://"+URIPort);

  // On msg from Node, display the results
  ws.onmessage = function(e) {
    if (e.data) {
      var data = JSON.parse(e.data);
      data.forEach(function(item){
        // display data on screen …
```

0

1

```
var wss = new WebSocketServer({
  server: server
});

wss.on('connection', function(ws) {
  ws.on('message', function(message) {
    var msg = JSON.parse(message);
    if (msg && msg.startCount) {
      startCount();
    } …

var wcount = require('./wcount.js');

function startCount() {
  var file = './data/dream.txt';
  wcount.start(file, function(rawdata){
    var results = [];
    rawdata.forEach(function(result) {
      results.push({count:result[0],
        word:result[1]})
    });
    wss.clients.forEach(function(client) {
      try {
        // Send results to browser
        client.send(JSON.stringify(results));
      } …
```

2

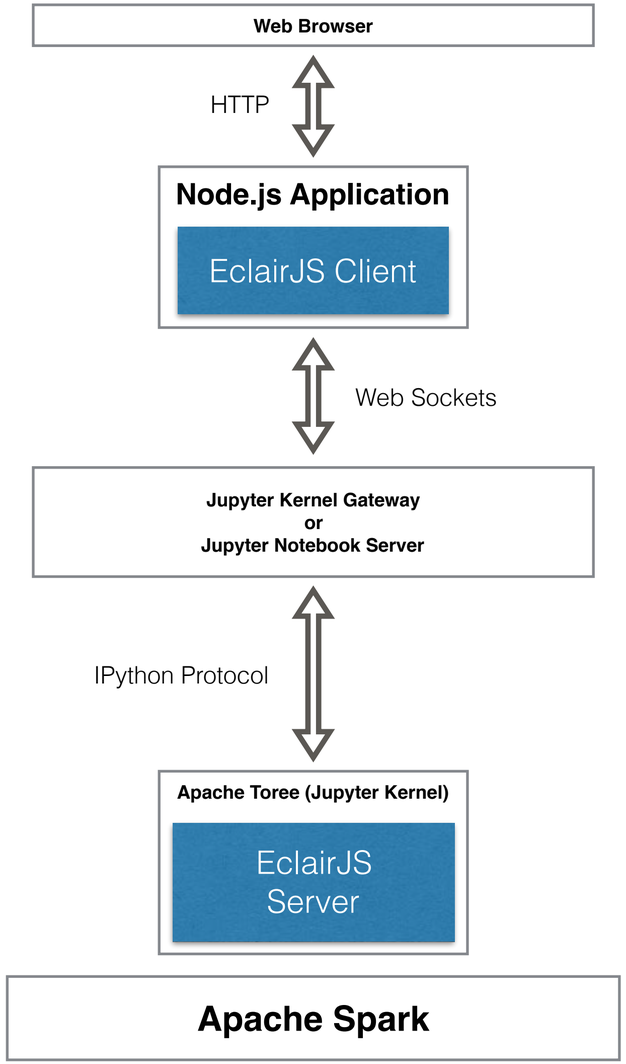
```
// Wrapper class
function WCount() {
}

WCount.prototype.start = function(file, callback) {
  startWCount(file, callback);
}

function startWCount(file, callback) {
  var rdd = sparkSession.read().textFile(file).rdd();

  var rdd2 = rdd.flatMap(function(sentence) {
    return sentence.split(" ");
  });
  // etc …

  rdd7.take(10).then(function(val) {
    callback(val);
  }) …
```

3

4

Architectural Detour

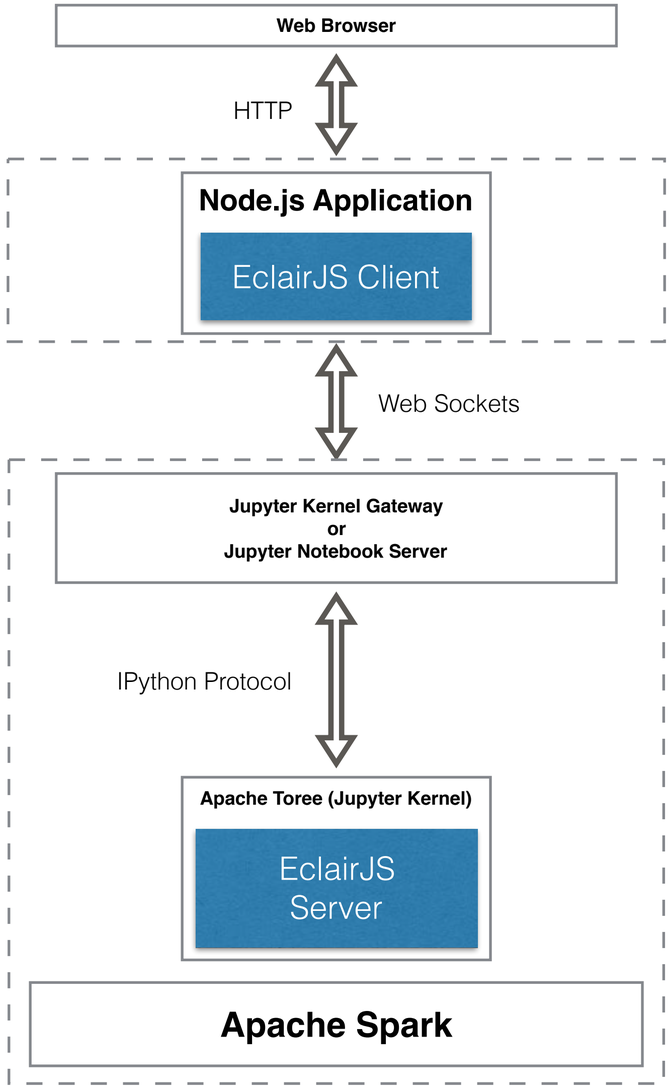
- EclairJS components
  - Client: JS API, “npm install” target
  - Server: JS - Java mappings, JVM Nashorn (v8)
- Jupyter infrastructure
  - Transport for interactive applications
  - Multi-channel IPython Protocol includes TCP- streaming support
  - JKG multiplexes WS - IPyP
- Jupyter for Notebooks + EclairJS = JS Notebooks



Notebook

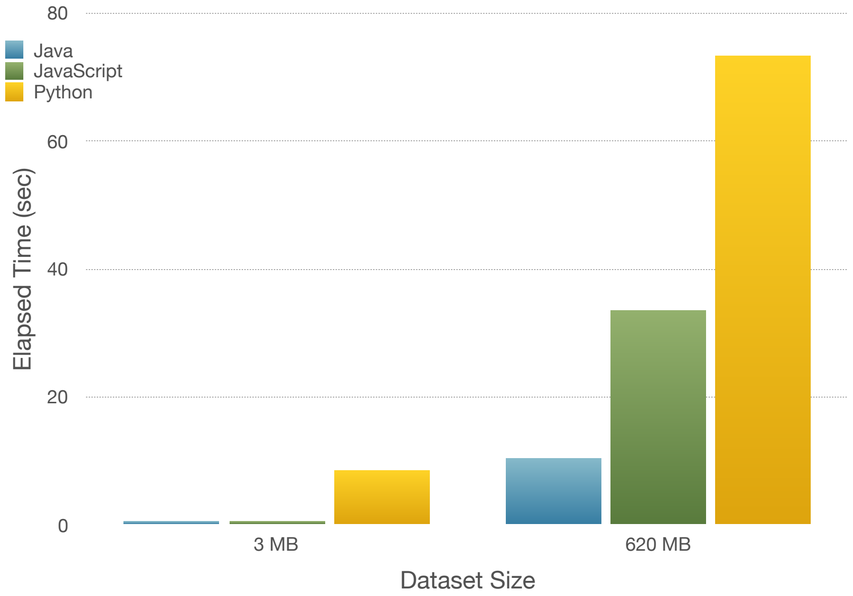
Deployment

- Server separation advantages
- Node.js and Spark scaling characteristics
- Server separation well-suited to cloud
  - E.g. in IBM cloud
  - Bluemix, Node.js SDK
  - IBM Analytics for Apache Spark or Spark deployed on SoftLayer



Performance

- Simple test based on MovieLens datasets
  - read, filter, map, count
  - 3MB/100k and 620MB/24m
  - Spark running local mode
- Java < JavaScript << Python
  - Difference maintained across test set sizes
  - Java and PySpark API's built-in



In Closing

- EclairJS: Simplicity, Performance & Spark functionality
- Open source on GitHub, see eclairjs.org
- Project is active and looking for collaborators
- Gracias, merci, thank-you!