დიდი მოცულობის მონაცემების დამუშავება Apache Spark-ის გამოყენებით

17 ივნისი 2016

თბილისი, საქართველო

გიორგი ჯვარიძე
@0xh3x

ჩემს შესახებ

- Software Engineer - Zalando-ს Fashion Insights Center-ში. Data Engineering / Data Science მიმართულებებით

- მანამდე TripAdvisor, aMind, MIA და სხვები

GERMANY

Berlin
Technology Headquarters

Dortmund
Fashion Platform Hub

Erfurt
High-Tech Logistic Centre

Hamburg
Advertisement Technology

Mönchengladbach
High-Tech Logistic Centre

IRELAND

Dublin
Fashion Insights Centre

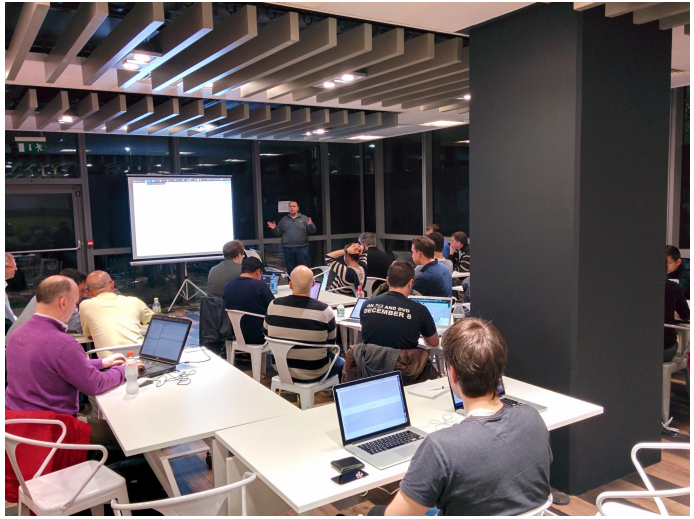
FINLAND

Helsinki
Tech Hub

Dublin R

დუბლინის R პროგრამირების ენის მომხმარებელთა ჯგუფის თანაორგანიზატორი
http://www.meetup.com/DublinR/


complexity




Anomaly Creation

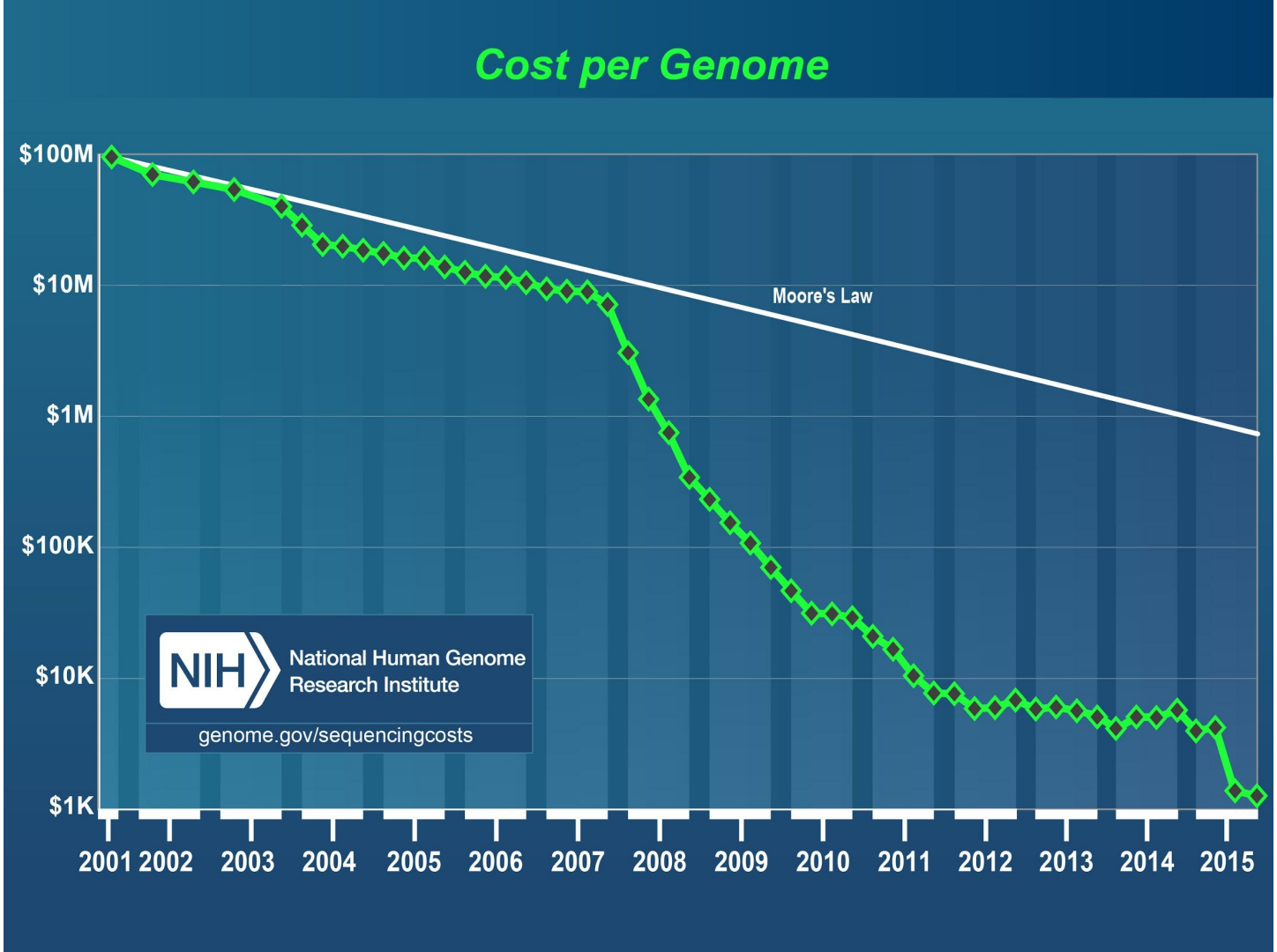
Big Data

3 **V**s of Big Data

Volume - მონაცემთა მოცულობა

Velocity - დამუშავების სიჩქარე

Variety - სტრუქტურირები, არასტრუქტურირებული და ორივე ერთად


Cost per Genome
$100M
$10M
$1M
$100K
$10K
$1K
Moore's Law
NIH
National Human Genome Research Institute
genome.gov/sequencingcosts
2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015

მაშტაბირებადობა

ვერტიკალური vs ჰორიზონტალური

ისტორია

2003 - Google - The Google File System
2004 - Google - MapReduce: Simplified Data Processing on Large Clusters.
2006 - Google - Bigtable: A Distributed Storage System for Structured Data.
2007 - Amazon - Dynamo: Amazon’s Highly Available Key-value Store.

Doug Cutting

APACHE HBASE

HUE

Pig

Zookeeper

Impala

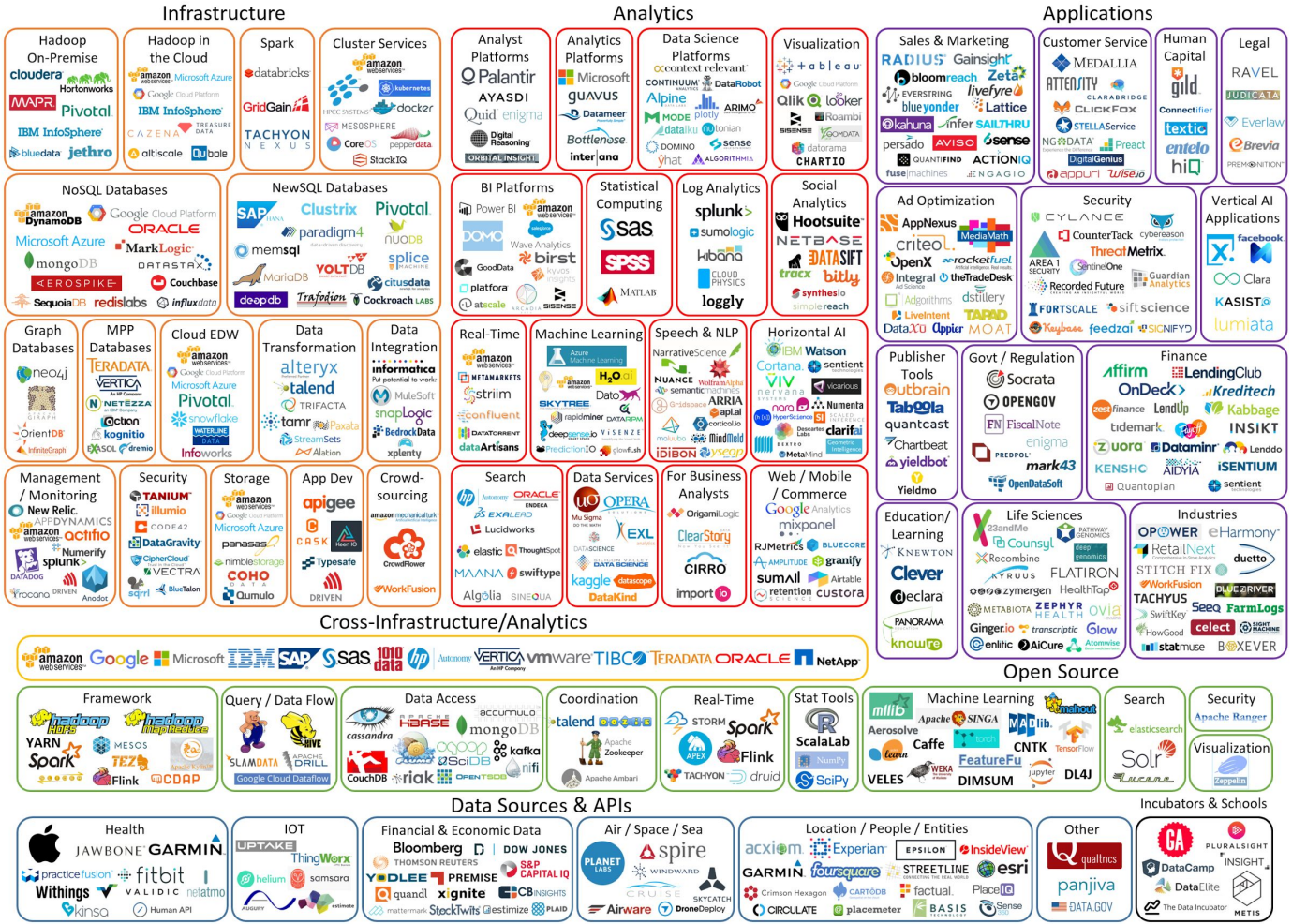
Apache Solr

ისტორია

Matei Zaharia

http://blog.madhukaraphatak.com/history-of-spark/

Big Data Landscape 2016 (Version 3.0)



Last Updated 3/23/2016

FIRSTMARK

წყარო http://mattturck.com/2016/02/01/big-data-landscape/

SMACK Stack

PANCAKE STACK

Presto 

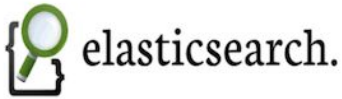
Arrow 

NiFi 

Cassandra 

AirFlow 

Kafka 

ElasticSearch 

Spark 

TensorFlow 

Algebird 

CoreNLP 

Kibana 

advancedspark.com.

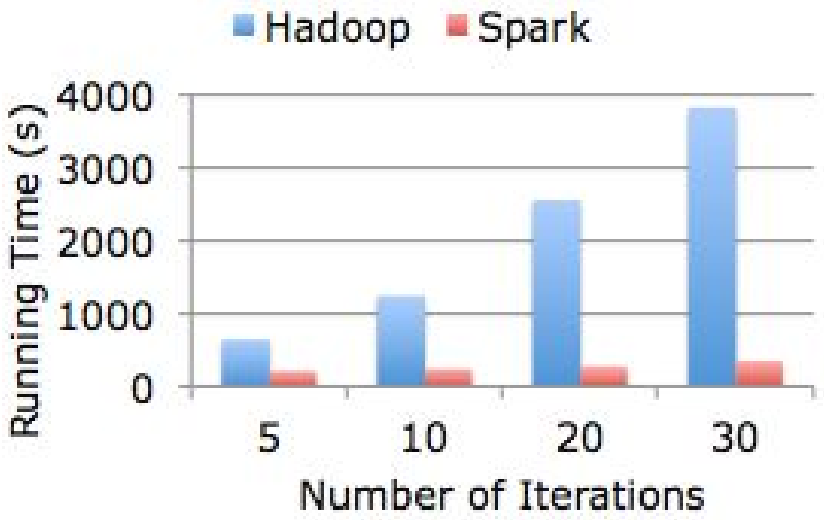
რა არის Apache Spark?

spark.apache.org

Apache Spark is a fast and general engine for large-scale data processing

Spark vs Hadoop MR

სიჩქარე



მონაცემების “ქეშირების” საშუალება. იტერატიული გამოთვლებისთვის

Spark vs Hadoop MR

პრობლემის უკეთ გამოხატვის საშუალება

- ნაკლები კოდი იგივე პრობლემის გადასაჭრელად
- map() და reduce() vs Collection API -ს მსგავსი აბსტრაქცია
- Scala, Java, Python და R ენების პირდაპირი (native) მხარდაჭერა

Spark vs Hadoop MR

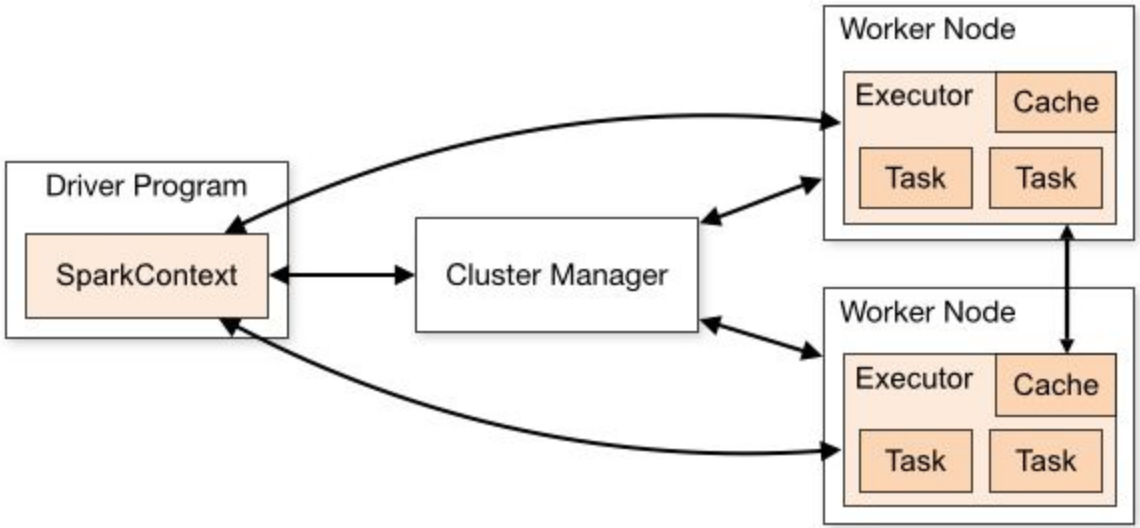
პრობლემის უკეთ გამოხატვის საშუალება

- ნაკლები კოდი იგივე პრობლემის გადასაჭრელად
- map() და reduce() vs Collection API -ს მსგავსი აბსტრაქცია
- Scala, Java, Python და R ენების პირდაპირი (native) მხარდაჭერა

```
text_file = spark.textFile("hdfs://...")

text_file.flatMap(lambda line: line.split())
    .map(lambda word: (word, 1))
    .reduceByKey(lambda a, b: a+b)
```

არქიტექტურა



RDD

Resilient Distributed Dataset

- ობიექტების დანაწევრებული კოლექცია, Partitioning
- არამოდიფიცირებადი, Read-Only/Immutable
- თვითაღდგენის საშუალებით, Resilient, Lineage
- გვიანი შესრულება, Lazy evaluation

RDD

Resilient Distributed Dataset

- ობიექტების დანაწევრებული კოლექცია, Partitioning
- არამოდიფიცირებადი, Read-Only/Immutable
- თვითაღდგენის საშუალებით, Resilient, Lineage
- გვიანი შესრულება, Lazy evaluation

```
text_file = spark.textFile("hdfs://...")

rdd1 = text_file.flatMap(lambda line: line.split())
rdd2 = rdd1.map(lambda word: (word, 1))
rdd3 = rdd2.reduceByKey(lambda a, b: a+b)
```

Transformations & Actions

- ტრანსფორმაციების შესრულება შესაძლებელია გადავადდეს
- ტრანსფორმაციის შედეგი არის ახალი RDD
- Action ფუნქციები იწვევს მთლიანი გამოთვლითი გრაფის შესრულების დაწყებას

Transformations & Actions

- ტრანსფორმაციების შესრულება შესაძლებელია გადავადდეს
- ტრანსფორმაციის შედეგი არის ახალი RDD
- Action ფუნქციები იწვევს მთლიანი გამოთვლითი გრაფის შესრულების დაწყებას

```
text_file = spark.textFile("hdfs://...")

rdd1 = text_file.flatMap(lambda line: line.split())
rdd2 = rdd1.map(lambda word: (word, 1))
rdd3 = rdd2.reduceByKey(lambda a, b: a+b)
print(rdd3.collect())
```

Transformations

```
map(f: T => U)
flatmap(f: T => TraversableOnce[U])
filter(f: T => Boolean)
mapPartitions(Iterator[T] => Iterator[U])
sample(withReplacement, fraction, seed)
union(otherRdd[T])
intersection(otherRdd[T])
distinct()
groupByKey()
reduceByKey(f: (V, V) => V)
sortByKey([ascending])
join(other: RDD[(K, W))
cogroup(other: RDD[(K, W))
cartesian(other: RDD[U])
```

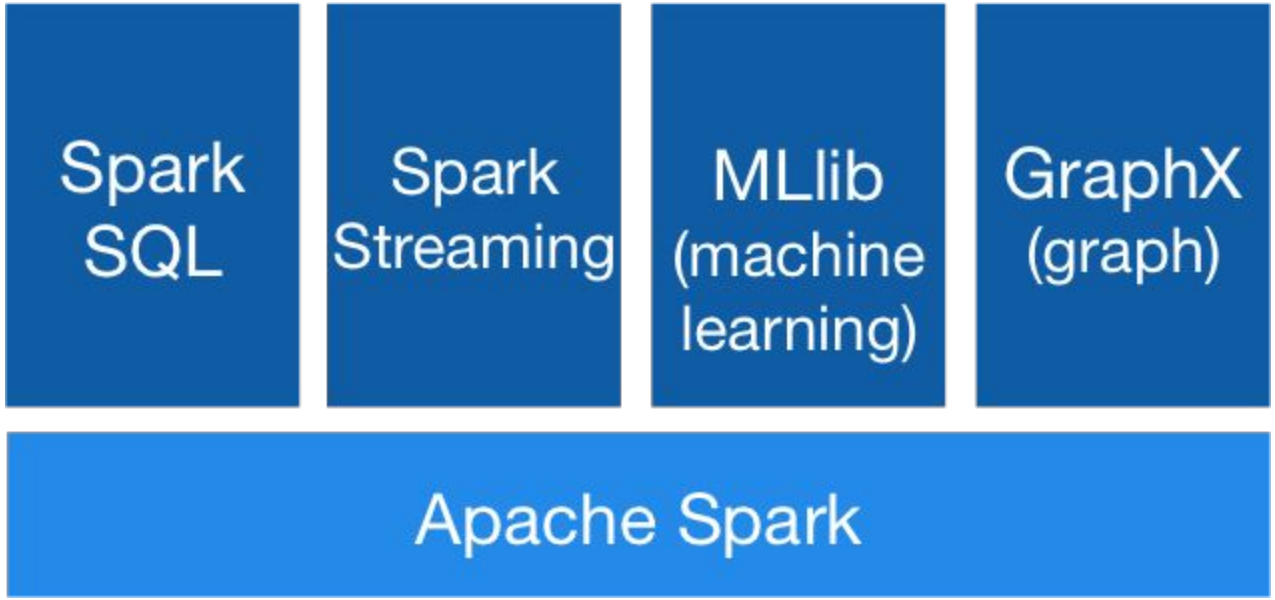
და სხვა მრავალი

Actions

```
reduce(f: (T, T) => T)
collect()
count()
first()
take(num)
takeSample(withReplacement, fraction, seed)
takeOrdered(num)(order)
saveAsTextFile(fileName)
saveAsSequenceFile(fileName)
saveAsObjectFile(fileName)
countByValue()
countByKey()
foreach(f: T=>Unit)
```

და სხვა მრავალი

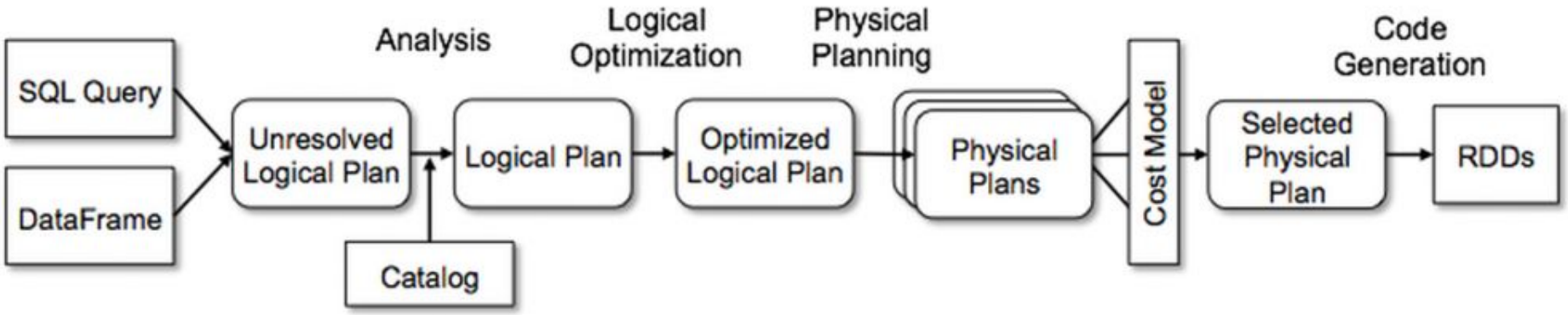
კომპონენტები



Spark SQL და DataFrame API

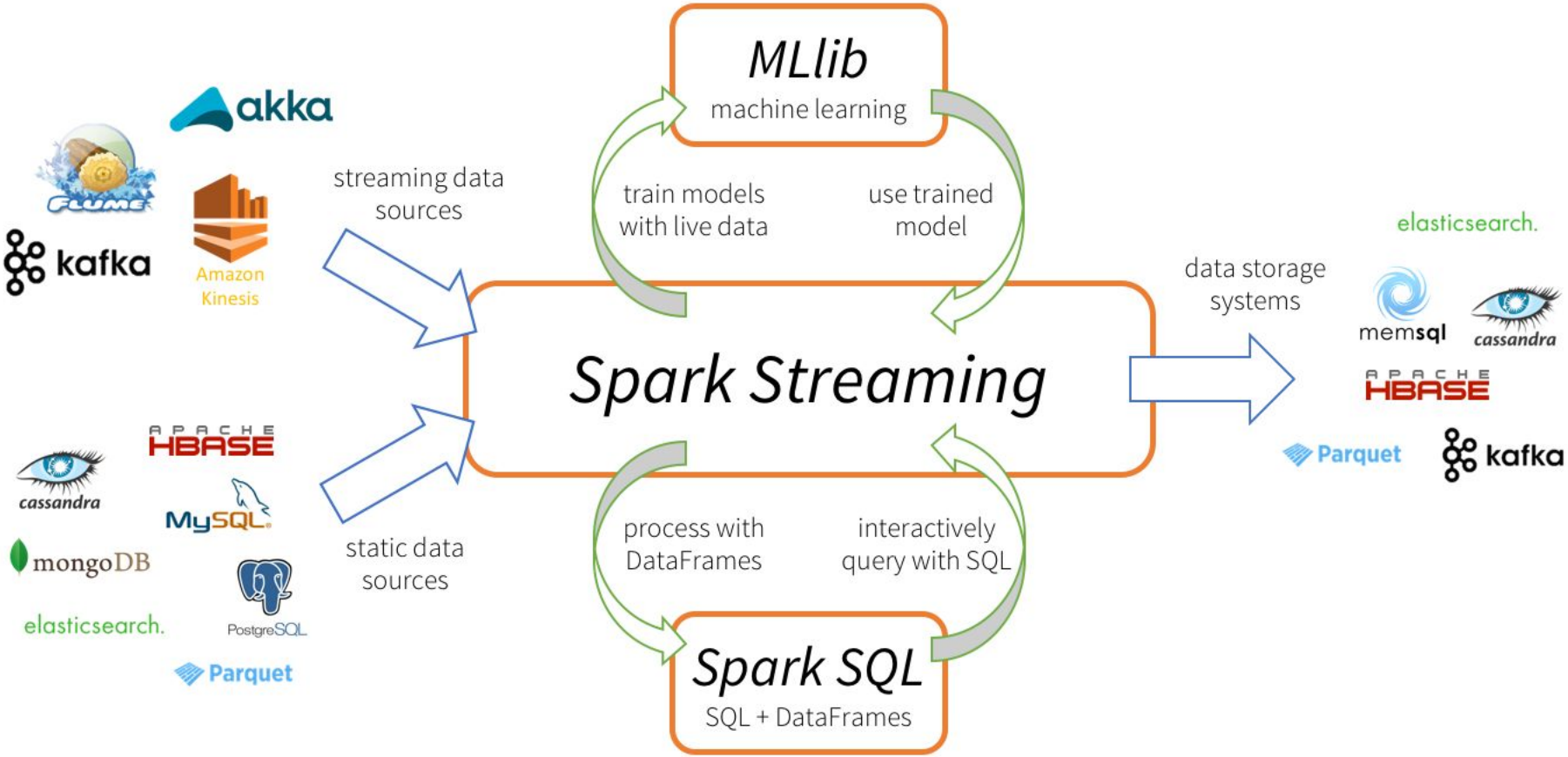
- SQL შესრულების ძრავი
- თუმცა განსხვავდება ტრადიციული რელაციული მონაცემთა ბაზებისგან
- დაშენებულია RDD API-ზე, RDD + სქემა
- ოპტიმიზატორი
- DataFrame API ინსპირირებულია R-ის data.frame-დან
- DataSet API ტიპიზირებული DataFrame API

Catalyst Optimizer



Streaming

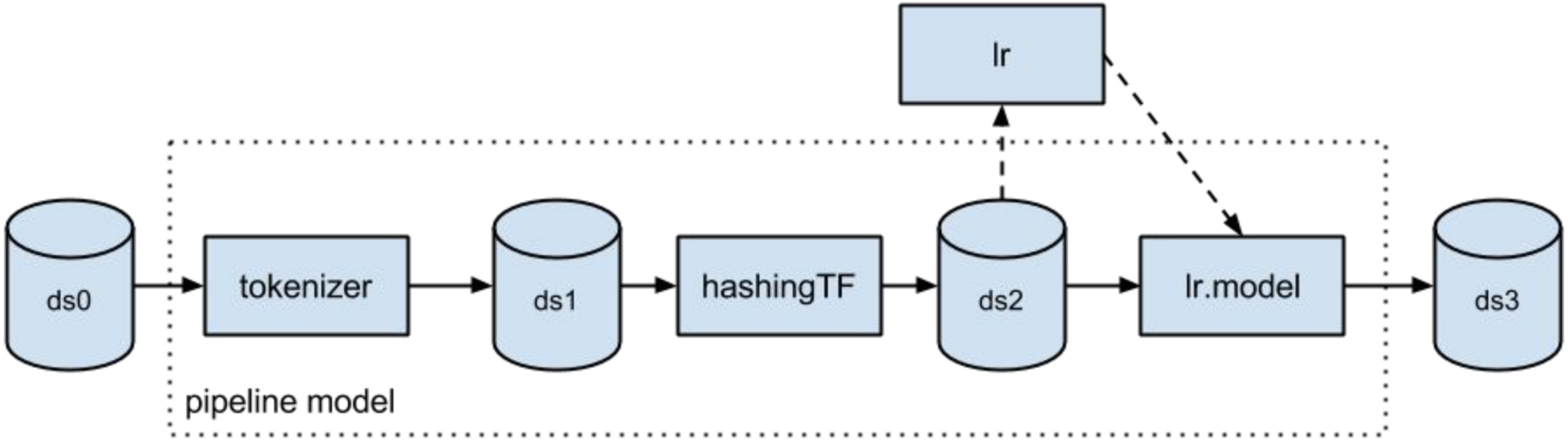
- როდესაც მონაცემების დამუშავება რეალურ დროში გჭირდება
- ნაცნობი RDD და DataFrame API ფუნქციები ნაკადების დამუშავებისას
- არსებული კოდის ხელახლა გამოყენება
- განსაკუთრებით მნიშვნელოვანია ე.წ. IoT აპლიკაციებისთვის
- ალტერნატივა: Apache Flink და სხვები


akka
FLUME
kafka
Amazon Kinesis
streaming data sources
MLlib
machine learning
train models with live data
use trained model
Spark Streaming
data storage systems
elasticsearch.
memsql
cassandra
APACHE HBASE
Parquet
kafka
APACHE HBASE
cassandra
MySQL
mongoDB
elasticsearch.
PostgreSQL
Parquet
static data sources
process with DataFrames
interactively query with SQL
Spark SQL
SQL + DataFrames

MlLib და GraphX

- Machine Learning ალგორითმების და გრაფებთან სამუშაო ბიბლიოთეკები
- რეგრესიის, კლასიფიკაციის, რეკომენდაციის სისტემების ცნობილი ალგორითმების იმპლემენტაციები
- ყველა ალგორითმი ადვილად არ პარალელიზდება

ML Pipelines



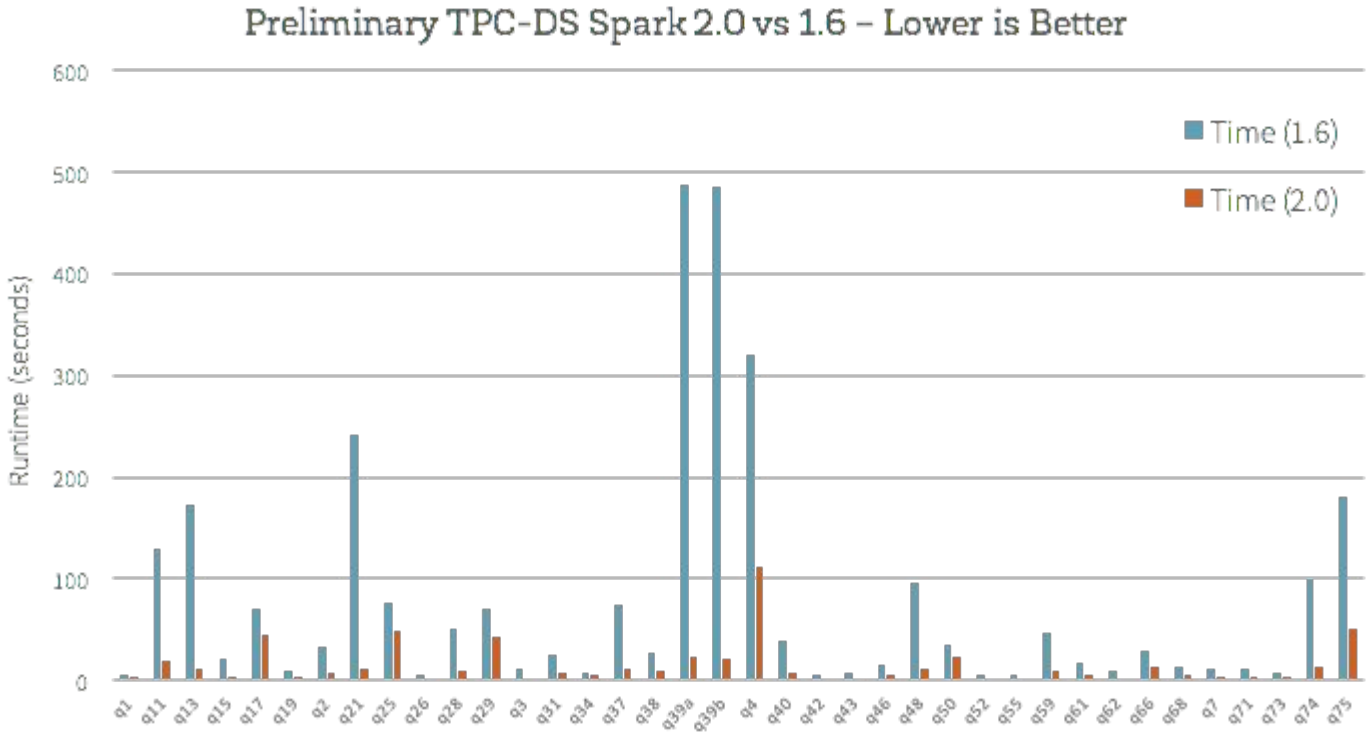
Spark 2.0

- 99 TPC-DS სტანდარტის მთლიანი შესრულების შესაძლებლობა
- უნიფიცირებული DataFrames და Datasets API Scala/Java-ში
- Machine learning pipeline-ების შენახვის საშუალება
- ახალი დისტრიბუციული ალგორითმების მხარდაჭერა R-დან Generalized Linear Models (GLM), Naive Bayes, Survival Regression, K-Means…
- Structured Streaming

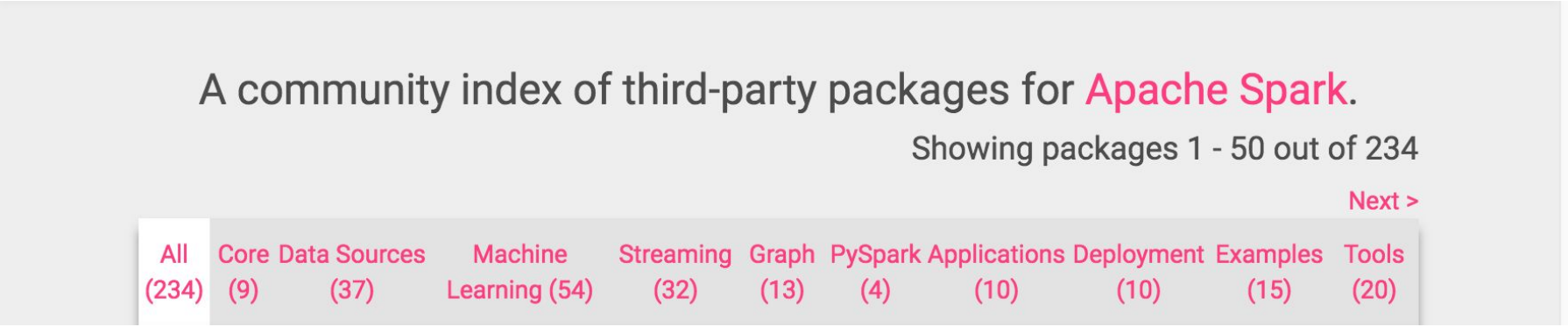
https://databricks.com/blog/2016/05/11/apache-spark-2-0-technical-preview-easier-faster-and-smarter.html

Spark 2.0

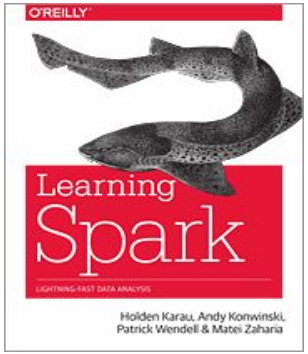
სისწრაფე - Spark როგორც კომპილატორი

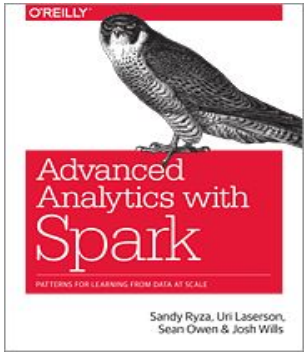


spark-packages.org



რესურსები





http://spark.apache.org/
https://databricks.com/
https://github.com/onurakpolat/awesome-bigdata

https://www.edx.org/course/big-data-analysis-apache-spark-uc-berkeleyx-cs110x
https://www.edx.org/course/introduction-apache-spark-uc-berkeleyx-cs105x

https://fb.com/groups/DataScienceGeorgia

მადლობა!