Hadoop 101 for bioinformaticians

Attila Csordas

Anybody used Hadoop before?

Aim

1 hour:
theoretical introduction -> be able to assess whether your problem fits MR/Hadoop

practical session -> be able to start developing code

Hadoop & me

mostly non-coding bioinformatician at PRIDE

Cloudera Certified Hadoop Developer

~1300 hadoop jobs ~ 4000 MR jobs

3+ yrs of operator experience

Software

Highly accessed

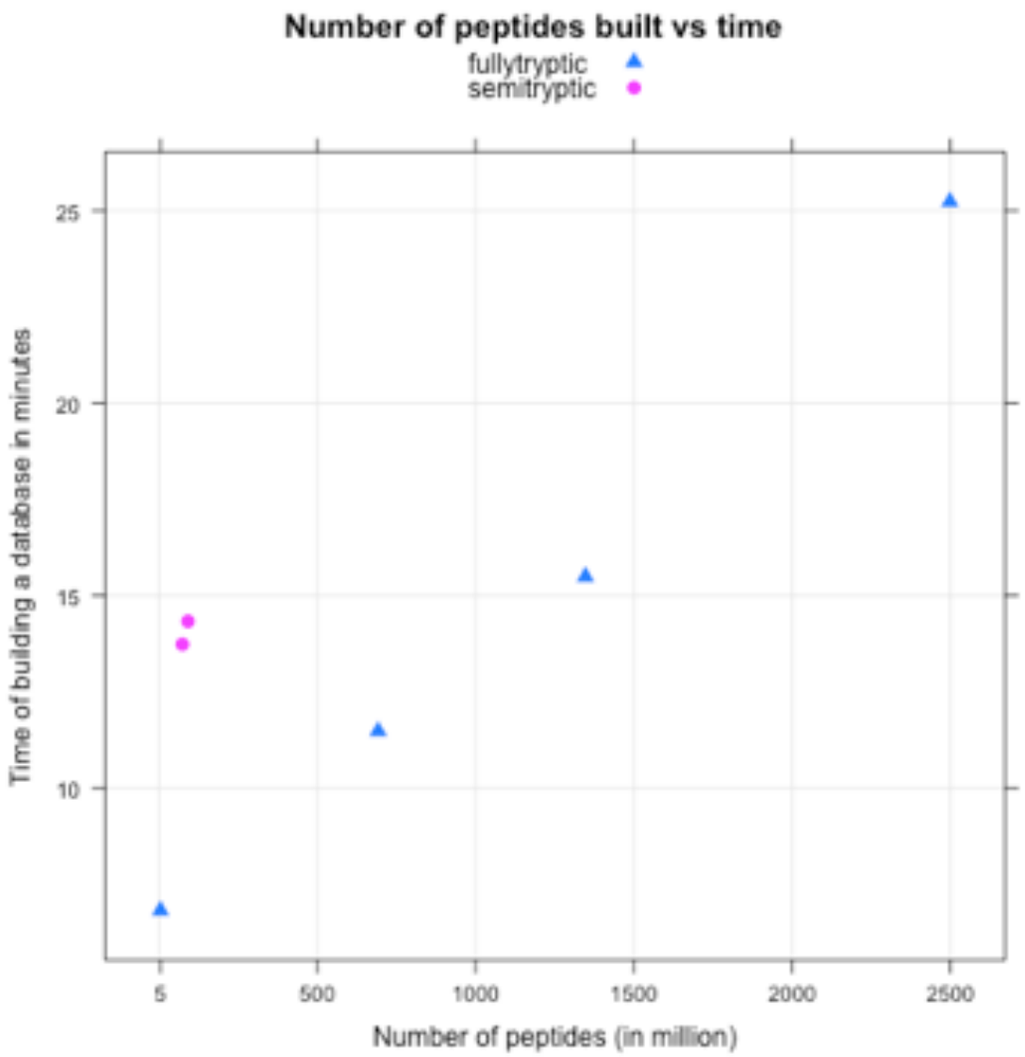
Open Access

Hydra: a scalable proteomic search engine which utilizes the Hadoop distributed computing framework

Steven Lewis[1]*, Attila Csordas[2], Sarah Killcoyne[3], Henning Hermjakob[2], Michael R Hoopmann[1], Robert L Moritz[1], Eric W Deutsch[1] and **John Boyle[1]**

Linear scalability

Db build time = f(# of peptides)



Search time = f(job complexity)

# of proteins: 4 mill (quarter nr) 8 mill (half nr) 16 mill (nr)

Theoretical introduction

- Big Data
- Data Operating System
- Hadoop 1.0
- MapReduce
- Hadoop 2.0

Big Data

difficult to process on a single machine

Volume: low TB - low PB

Velocity: generation rate

Variety: tab separated, machine data, documents

biological repositories fit the bell: e.g.. PRIDE

Data Operating System

features	components
distributed	storage
scalable	resource management
fault-tolerant	processing engine 1
redundant	processing engine 2

Hadoop 1.0

storage	Hadoop Distributed File System aka HDFS	redundant, reliable, distributed
processing engine/ cluster resource management	MapReduce	distributed, fault-tolerant resource management, scheduling & processing engine

single use data platform

MapReduce

programming model for large scale, distributed, fault tolerant, batch data processing

execution framework, the “runtime”

software implementation

Stateless algorithms

output only depends on the current input but not on previous inputs

dependent: Fibonacci series: $F(k+2)=F(k+1)+F(k)$

Parallelizable problems

easy to separate to parallel tasks

might need a to maintain a global, shared state

Embarassingly/pleasingly parallel problems

easy to separate to parallel tasks

no dependency or communication between those tasks

Pleasingly Parallel MCMC: cracked wide open for MapReduce and Hadoop

Edit

Posted on March 9, 2014 by attilacsordas

MCMC methods guarantee an accurate enough result (say parameter estimation for a phylogenetic tree). But they give it to you usually in the long-run and many burn-in steps might be necessary before performing ok. And if the data size grows larger, the number of operations to draw a sample grows larger too (N -> O(N) for most MCMC methods.

Asymptotically Exact, Embarrassingly Parallel MCMC

Willie Neiswanger*, Chong Wang†, and Eric Xing‡

Machine Learning Department, Carnegie Mellon University

Functional programming roots

higher-order functions that accept other functions as arguments

map: applies its argument *f() to* all elements of a list

fold: takes *g()* + initial value -> final value

sum of squares: map(x^2), fold(+) + 0 as initial value

MapReduce

map(key, value) : [(key, value)]

shuffle&sort: values grouped by key

reduce(key, iterator<value>) : [(key, value)]

Word Count

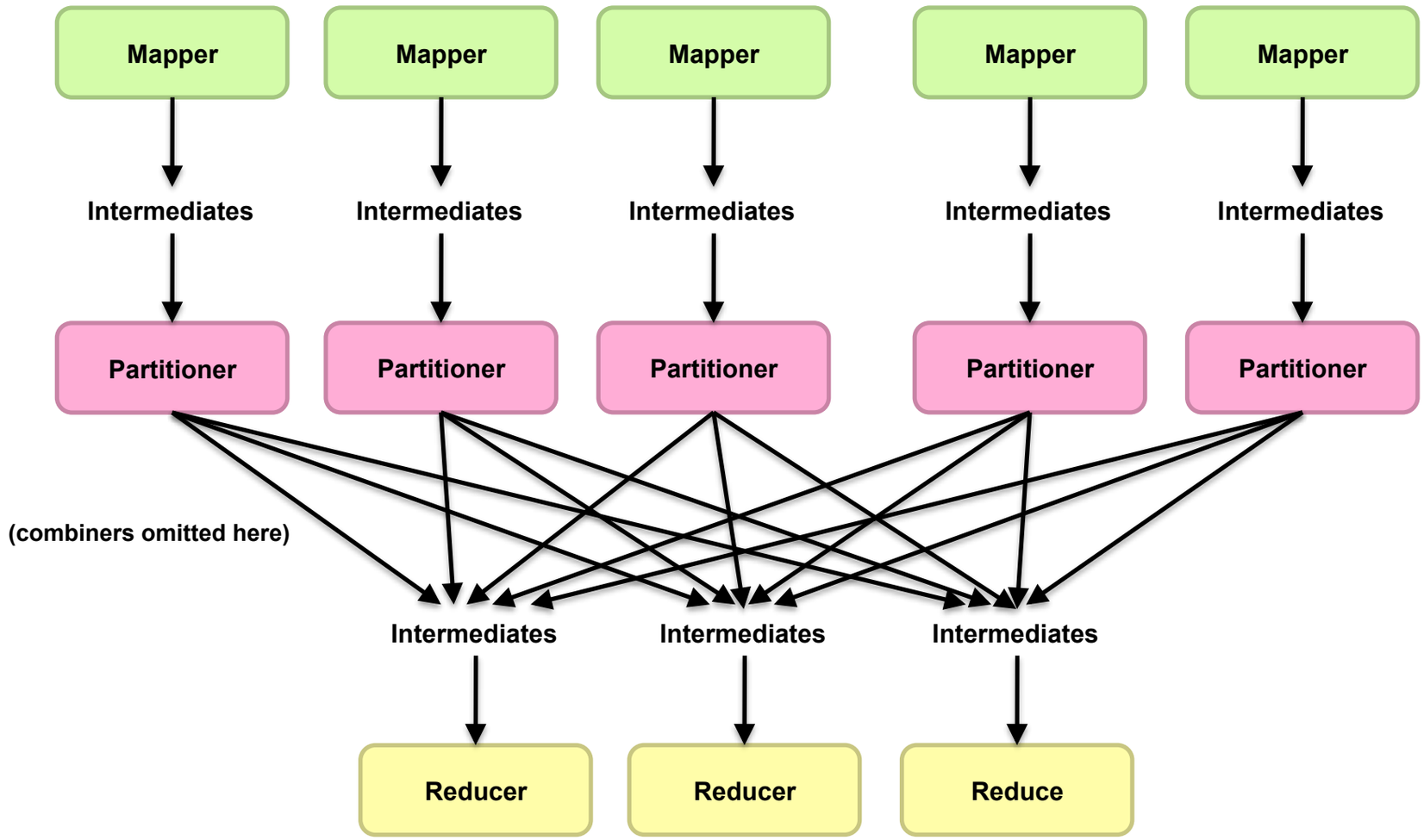
```
Map(String docid, String text):
    for each word w in text:
        Emit(w, 1);

Reduce(String term, Iterator<Int> values):
    int sum = 0;
    for each v in values:
        sum += v;
        Emit(term, value);
```

Count amino acids in peptides

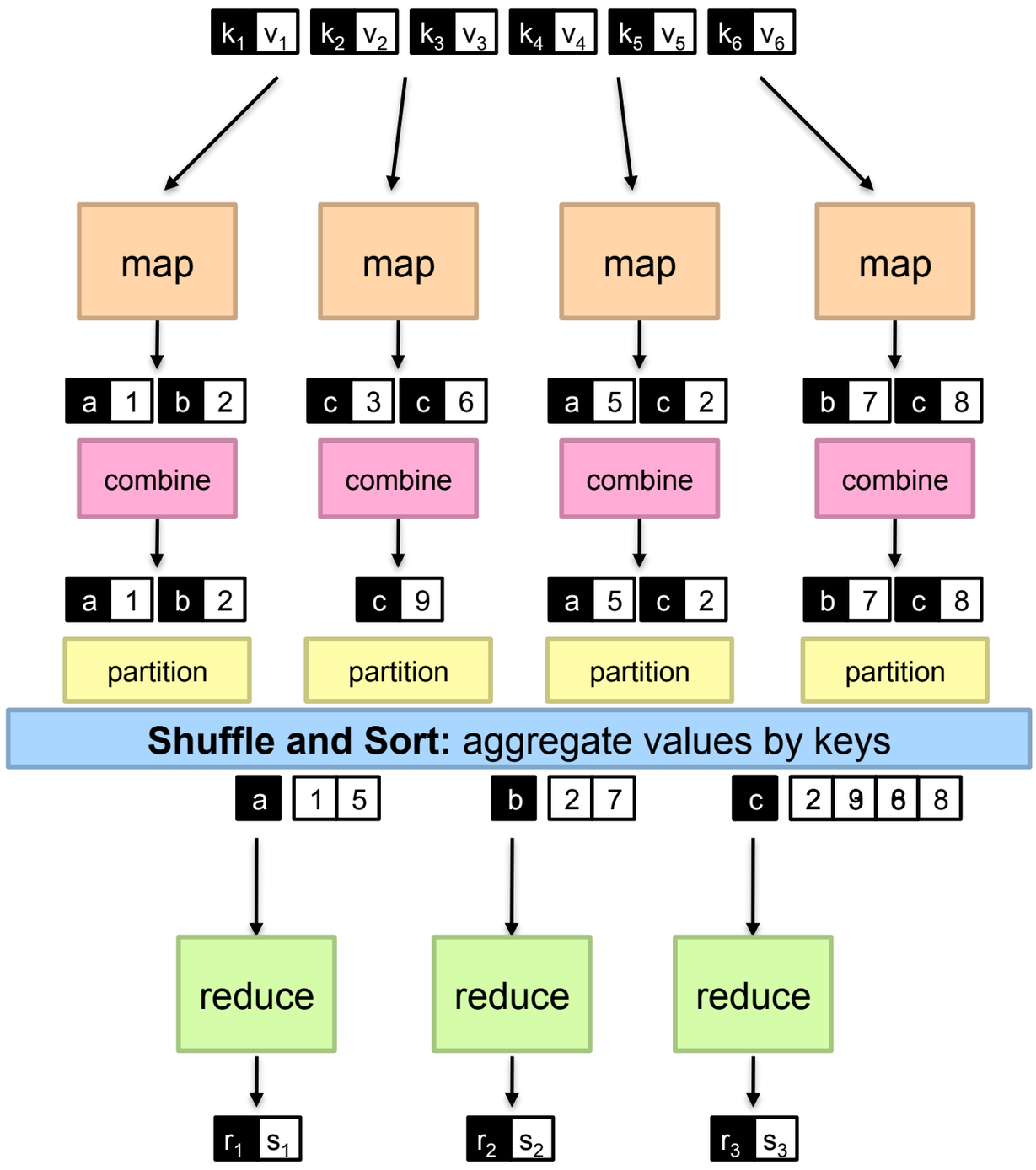
```
Map(~~byte offset of the line~~, peptide sequence):
    for each amino acid in peptide:
        Emit(amino acid, 1);

Reduce(amino acid, Iterator<Int> values):
    int sum = 0;
    for each v in values:
        sum += v;
    Emit(amino acid, value);
```

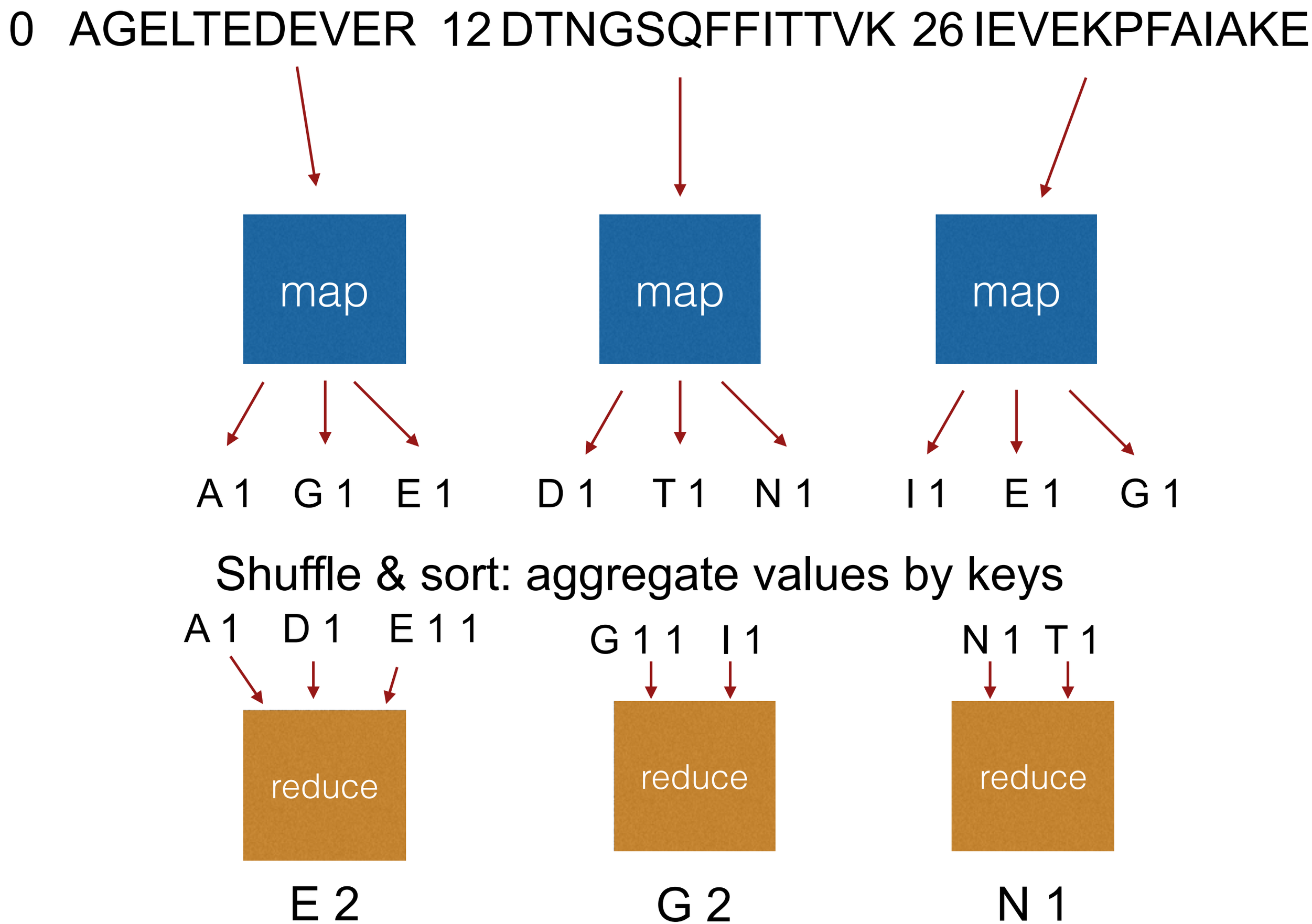


Source: redrawn from a slide by Cloduera, cc-licensed

source: Jimmy Lin



source: Jimmy Lin


0 AGELTEDEVER
12 DTNGSQFFITTVK
26 IEVEKPFAIAKE
map
map
map
A 1 G 1 E 1
D 1 T 1 N 1
I 1 E 1 G 1
Shuffle & sort: aggregate values by keys
A 1 D 1 E 1 1
G 1 1 I 1
N 1 T 1
reduce
reduce
reduce
E 2
G 2
N 1

Hadoop 1.0: single use

Pig: scripting 4 Hadoop, Yahoo!, 2006

```
input_lines = LOAD '/tmp/my-copy-of-all-pages-on-internet' AS (line:chararray);

-- Extract words from each line and put them into a pig bag
-- datatype, then flatten the bag to get one word on each row
words = FOREACH input_lines GENERATE FLATTEN(TOKENIZE(line)) AS word;

-- filter out any words that are just white spaces
filtered_words = FILTER words BY word MATCHES '\\w+';

-- create a group for each word
word_groups = GROUP filtered_words BY word;

-- count the entries in each group
word_count = FOREACH word_groups GENERATE COUNT(filtered_words) AS count, group AS word;

-- order the records by count
ordered_word_count = ORDER word_count BY count DESC;
STORE ordered_word_count INTO '/tmp/number-of-words-on-internet';
```

Hive: SQL queries 4 Hadoop Facebook


```
hive> CREATE TABLE invites (foo INT, bar STRING) PARTITIONED BY (ds STRING);
```

```
hive> SELECT a.foo FROM invites a WHERE a.ds='2008-08-15';
```

Hadoop 2.0

processing engines:	MapReduce, Tez, HBase, Storm, Giraph, Spark…	batch, interactive, online, streaming, graph …
cluster resource management	YARN	distributed, fault-tolerant resource management, scheduling &
storage	HDFS2	redundant, reliable, distributed

multi use data platform

Practical session

Objective: up & running w/ Hadoop in 30 mins

requirements

- java
- intellij idea
- maven

Outline

- check out https://github.com/attilacsordas/hadoop_introduction
- elements of a MapReduce app, basic Hadoop API & data types
- bioinformatics toy example: counting amino acids in sequences
- executing a hadoop job locally
- 2 more examples building on top of AminoAcidCounter

https://github.com/attilacsordas/hadoop_for_bioinformatics

GitHub, Inc. [US] https://github.com/attilacsordas/hadoop_for_bioinformatics

PUBLIC attilacsordas / hadoop_for_bioinformatics

Unwatch 2 | Star 0 | Fork 0

hadoop tutorial for bioinformaticians — Edit

37 commits | 1 branch | 0 releases | 1 contributor

branch: master | hadoop_for_bioinformatics /

change groupId and artifactId

attilacsordas authored 2 days ago — latest commit 76fa58e67f

fastainput	small yeast fasta input file for fasta example	a month ago
input	Initial commit of the bioinformatics hadoop intro package working on ...	a month ago
output	Initial commit of the bioinformatics hadoop intro package working on ...	a month ago
src	minor changes so project can be built without errors with Eclipse usi...	2 days ago
README.md	Update README.md	19 days ago
pom.xml	change groupId and artifactId	2 days ago

README.md

hadoop_for_bioinformatics

hadoop 101 tutorial for bioinformaticians

Code

Issues 0

Pull Requests 0

Wiki

Pulse

Graphs

Network

Settings

HTTPS clone URL

https://github.

You can clone with HTTPS, SSH, or Subversion.

Clone in Desktop

Download ZIP

MapReduce app components

- driver
- mapper
- reducer

repo: bioinformatics.hadoop.AminoAcidCounter.java

Inputformats

specifies data passed to Mappers

specified in driver

determines input splits and RecordReaders extracting key-value pairs

default formats: TextInputFormat, KeyValueTextInputFormat, SequenceFileInputFormat ...

Data types

everything implements Writable, de/serialization

all keys are WritableComparable: sorting keys

box classes for primitive data types: Text, IntWritable, LongWritable ...

bioinformatics toy example

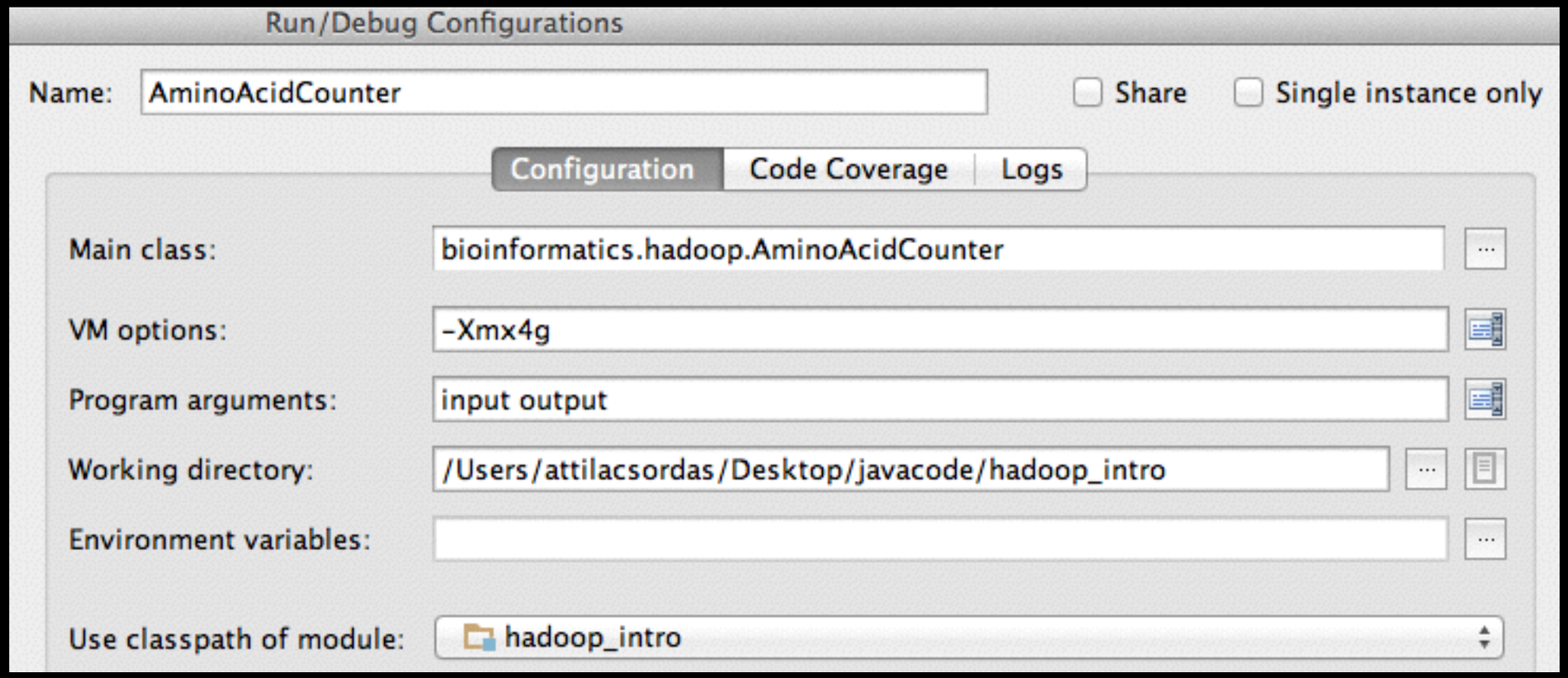
Our task is to prepare data to form a statistical model on what physico-chemico properties of the constituent amino acid residues are affecting the visibility of the high-flyer peptides for a mass spectrometer.

AGELTEDEVER
QSVHLVENEIQASIDQIFSHLER
DTNGSQFFITTVK
IEVEKPFAIAKE

repo: headpeptides.txt in input folder

run the code!

local
pseudodistributed
cluster



Debugging

do it locally if possible
print statements

write to map output

mapred.map.child.log.level=DEBUG
mapred.reduce.child.log.level=DEBUG
-> syslog task logfile

running debuggers remotely is hard

JVM debugging options

task profilers

Counters

```
14/02/18 12:47:41 INFO mapred.JobClient: Counters: 17
14/02/18 12:47:41 INFO mapred.JobClient:   File Output Format Counters
14/02/18 12:47:41 INFO mapred.JobClient:     Bytes Written=94
14/02/18 12:47:41 INFO mapred.JobClient:   FileSystemCounters
14/02/18 12:47:41 INFO mapred.JobClient:     FILE_BYTES_READ=1888
14/02/18 12:47:41 INFO mapred.JobClient:     FILE_BYTES_WRITTEN=67446
14/02/18 12:47:41 INFO mapred.JobClient:   File Input Format Counters
14/02/18 12:47:41 INFO mapred.JobClient:     Bytes Read=154
14/02/18 12:47:41 INFO mapred.JobClient:   Map-Reduce Framework
14/02/18 12:47:41 INFO mapred.JobClient:     Reduce input groups=19
14/02/18 12:47:41 INFO mapred.JobClient:     Map output materialized bytes=1158
```

track records for statistics, malformed records

AminoAcidCounterwithMalformedCounter

- Task 1: modify mapper to count positions too: R_2 -> AminoAcidPositionCounter

- Task 2: count positions & normalise to peptide length -> AminoAcidPositionCounterNormalizedToPeptideLength

Run jar on the cluster

set up an account on a hadoop cluster

move input data into HDFS

set mainClass in pom.xml

mvn clean install

cp jar from target to servers

ssh into hadoop

run hadoop command line

homework, next steps

count all the 3 outputs for 1 input k,v pair

count all amino acid pairs

run FastaAminoAcidCounter & check FastaInputFormat

run the jar on a cluster

https://github.com/lintool/MapReduce-course-2013s/tree/master/slides

n * (trial, error) -> success