Building Effective Algorithms and Analytics for Hadoop and Other Systems

MapReduce Design Patterns

Donald Miner & Adam Shook

MapReduce Design Patterns

Donald Miner and Adam Shook

O'REILLY®
Beijing • Cambridge • Farnham • Köln • Sebastopol • Tokyo

MapReduce Design Patterns

by Do l M ▲ S oo

bl by ' lly M , , 1 5 v y N , S b o po l, C▲ 9547

' lly boo y b p o o l, b , o l po o o l l o
l o v l bl o o l *http://my.safaribooksonline.com* Fo o o o , o o o po /
o l l p 8 -998-9938o *corporate@oreilly.com*

Editors: ▲ y M o
Production Editor: C o p

Proofreader: D C ll
Cover Designer: y Co
Interior Designer: D v F o
Illustrator: b D

D b 1 F o

Revision History for the First Edition:

1 -11- F l

S *http://oreilly.com/catalog/errata.csp?isbn=9781449327170* o l l

N ll boo , N ll boo lo o , ' llylo o o ' lly M , *MapReduce Design Patterns*, o D v ' , l o ' lly M ,

M yo o by ll o p o l W o o pp boo , ' lly M , , o l , o v b p p o l p

W l v y p o b p p o o boo , p bl o o po b l y o o o o o ,o o l o o o o o

S N 978-1-449-3 717-

[S

For William

Table of Contents

Preface

W l o o *MapReduce Design Patterns*! T boo ll b q o y
l o F o o , boo o bvo ly bo p ,
pl o l o o lv p o bl W oo lo o
p boo v b p p o , p l ly *Design Patterns: Elements of Reusable Object-Oriented Software*, by l 1995 ,
o o ly o T o Fo boo Fo p , yo ’ll
pl o v o v lo ly b o o boo p ly
l pl ll lp yo o p o o yo T
ll b p lly l boo

T boo b o o p - boo oo boo o
o ’ llo p p o bl o v , l ly o oo boo , l o
boo o o o ’ll v o o b o py
p o o o o lv yo p o bl , b o p yo ll p o
yo l 9 %o y o bo llo yo ll

T boo o ly bo ly o oo po M p W o lly
y o o v o oo lo o oo po M p o o l oo
lo bo ▲ T o p v b bo q
, bo o l p , o o o o ly

p , ’ll l bo o o boo o b b
o boo yo ’v

Intended Audience

T o v o o o boo o ll p lo o
M p v lo p T y l o o y , o o o bl

M p , b l p o o o
o ll T o boo o p v yo o v o o o yo
o by yo o o p v o o o o lv p o bl
M p So , o y , boo b v o
v M p v lo p o , b ly b ll
o o

T boo l o o yo o l o bo M p
p T boo o ply o l o M p o
pl l pl o o o o M p y ,
ll lp o v lo p M p ly o v , q b o
p o v o o o p o o o
p , o l b o o o o o o y
l oo p o

To o o o boo , yo v o o l o oo p,
llo o pl o oo p yo p
oo p o ▲b ll b v p , lo o
o o o l l

Pattern Format

T p boo o llo l pl o o y o
o So p llo o o o y o ,
o o p

Intent

T o q p o o p o bl p o o lv

Motivation

T o pl y yo o l o o lv p o bl o o l
pp So yp lly b

Applicability

T o o o b o b bl o pply p
o p o bl So l o o p
o y lp yo p ll o yo o

Structure

T o pl l yo o M p o b l ’ll pl
p p o , p o , l o l yo o ’ll b
y o p o , o b ,o p o T o p
pl o o o lv p o bl

Consequences

T o p y o pl o p o p ll b
T o lo o p p p o

Resemblances

Fo v o p SQ o , o ll o l
o o o p o bl o l b o lv o l o y
v yo l o o po o
p o

So , SQ , ,o bo o o M p
ly q

Known Uses

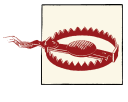
T o o l o o o o p

Performance Analysis

T o pl p o p o l o ly p o by p
po b v y M p ly o
b o p o p ly o p o W o o l
o o o yo l , o l b l o o

The Examples in This Book

▲llo pl boo o oo p v o 1 3 M p
p b o o p o o l y y ,
b o p o o o pl o y o llo , o p
oo p oo p lo l o ly y , b o p
o Mo o D ' M p o M p pl o ll b bl
o po l pl o p l y o o



l, y o mapreduce ▲ o llo o
pl , o p mapred ▲ J b l o
o boo o o , pl yo p o pl ll mapred
▲ o o p bl

pl llyo y o o o l , o ly o o o
l- o l b y , yo p yo o b lo
yo 'll o b p o v l o o yo ly

W o o po S v lo ' b
S v lo po p l b o v lo p o o

q o bo y o o p l o o p T o
b o bl , y o o b yo ’ o l o T
l o o - l l ll
l l

T o o pl boo , y o b o p lo o
o lp o o l ly o p o
o l p o p S XM p y pl , lly
wo ll-blo XM p p o o p o
o J v o

T o v bl ,o o ly o , po ,
▲llo ll-o XM , o o p l

W o llo S v lo bl boo

comments

```
<row Id="2579740" PostId="2573882" Text="Are you getting any results? What
are you specifying as the command text?" CreationDate="2010-04-04T08:48:51.347"
UserId="95437" />
```

Co o llo - p q o o o o l v o
po , q o o

posts

```
<row Id="6939296" PostTypeId="2" ParentId="6939137"
CreationDate="2011-08-04T09:50:25.043" Score="4" ViewCount=""
Body="&lt;p&gt;You should have imported Poll with &lt;code&gt;
from polls.models import Poll&lt;/code&gt;&lt;/p&gt;&#xA;"
OwnerUserId="634150" LastActivityDate="2011-08-04T09:50:25.043"
CommentCount="1" />

<row Id="6939304" PostTypeId="1" AcceptedAnswerId="6939433"
CreationDate="2011-08-04T09:50:58.910" Score="1" ViewCount="26"
Body="&lt;p&gt;Is it possible to gzip a single asp.net 3.5 page? my
site is hosted on IIS7 and for technical reasons I cannot enable gzip
compression site wide. does IIS7 have an option to gzip individual pages or
will I have to override OnPreRender and write some code to compress the
output?&lt;/p&gt;&#xA;" OwnerUserId="743184"
LastActivityDate="2011-08-04T10:19:04.107" Title="gzip a single asp.net page"
Tags="&lt;asp.net&gt;&lt;iis7&gt;&lt;gzip&gt;"
AnswerCount="2" />
```

o o q o o ▲ ll po q o ,
o o po o q o Q o
b pwo o wo p o yo po o v
o o o o lp o q o , o o q o
p y b o , y po bo pl bo v ,
po bo p , , p

o o bo yo po p TM T p
b o ll , b ’ o oo b ll oo l v l bl Mo o
q o yo o b p y b !

o b o ll b y o bo q o
Q o v PostTypeIdo 1, l v PostTypeId
o ▲ po o l q o v ParentId, l q o
o o v Q o , o v , v Title Tags

users

```
<row Id="352268" Reputation="3313" CreationDate="2010-05-27T18:34:45.817"
DisplayName="orangeoctopus" EmailHash="93fc5e3d9451bcd3fdb552423ceb52cd"
LastAccessDate="2011-09-01T13:55:02.013" Location="Maryland" Age="26"
Views="48" UpVotes="294" DownVotes="4" />
```

T bl o llo bo o o l o S v b
Mo o o o o p ’ p o l

o S v b v p o o , o p o pwo
q o o b o b

To l o bo , o o o l o b
README.txt

pl , p lp o o T o
l o S v b HashMap T HashMap o
l b l y l v l

```
package mrdp.utils;

import java.util.HashMap;
import java.util.Map;

public class MRDPUtils {

   // This helper function parses the stackoverflow into a Map for us.
   public static Map<String, String> transformXmlToMap(String xml) {
      Map<String, String> map = new HashMap<String, String>();
      try {
         // exploit the fact that splitting on double quote
         //  tokenizes the data nicely for us
         String[] tokens = xml.trim().substring(5, xml.trim().length() - 3)
            .split("\"");

         for (int i = 0; i < tokens.length - 1; i += 2) {
            String key = tokens[i].trim();
            String val = tokens[i + 1];

            map.put(key.substring(0, key.length() - 1), val);
         }
      } catch (StringIndexOutOfBoundsException e) {
```

```
            System.err.println(xml);
        }

        return map;
    }
}
```

Conventions Used in This Book

T o lb ypo p l o v o boo

Italic

, , l , l , l o

`Constant width`

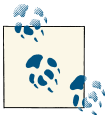
o po l , ll p p o o po l
v bl o o , b , yp , v o v bl ,
, y o

`Constant width bold`

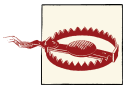
S o o o o o l b yp l lly by

`Constant width italic`

S o o l b pl - ppl v l o by v l
by o



T o p, o ,o l o



T o o o

Using Code Examples

T boo o lp yo yo o b o l, yo y o
boo yo p o o o o o o o o p
o l yo ’ p o po o o o Fo pl ,
p o v l o o o boo o o q p o
S ll o b CD- M o pl o ’ lly boo o q p
o ▲ q o by boo q o pl o o o
q p o o po o o pl o o boo
o yo p o ’ o o o q p o

Safari® Books Online

S oo l *www.safaribooksonline.com* o -
l l b y l v p o bo boo v o
o o o l ' l o ob y b

T ob y po o l , o v lo p , b , b v
po o l S oo l p y o o , po bl
o lv , l , o

S oo l o o po p po o o
o , o v , v l S b b v o o o
boo , v o , p p bl o p o lly bl b
o p bl l ' lly M , ll o o l, ▲ o W l y o
o l, M o o , S , Q , p , Fo l , C o , Jo
W l y & So , Sy , Mo , M boo , , ▲ o b , FT
, ▲p , M , N , M - ll, Jo & l , Co T o l
o y, o o Fo o o o bo S oo l , pl v
o l

How to Contact Us

l o q o o boo o p bl

' lly M ,
1 5 v y N o
S b o po l, C▲ 9547
8 -998-9938 S o C
7 7-8 9- 515 o lo b l
7 7-8 9- 1 4

W v b p o boo , l , pl , y o l
o o o p *http://oreil.ly/mapreduce-design-patterns*

To o o l q o bo boo , l o *bookques tions@oreilly.com*

Fo o o o bo o boo , o , o , , o b
http://www.oreilly.com

F o F boo *http://facebook.com/oreilly*

Fo llo o T *http://twitter.com/oreillymedia*

W o o T b *http://www.youtube.com/oreillymedia*

Acknowldgements

oo p bl by ’ lly l y o p o o o y T
ppo , p llyo o ▲ y , b ly lp l
o p o T y v o o o o v y l
ppo y y

▲ p l o o o o o boo p o v l o y
v To W l , ▲ l , To l , y T o
J o l o p o v o ly o o W pp
S ’ lp v l boo *Hadoop Operations*

T S v lo , o o boo , ly v l bl
C v Co o l ’ p o pl ll o p o
l o p o l o o W ly
o l o b o

Do o l l o ppo o o o o pl , o p o v
l y l o o o p o , o l ppo , l o
T o l o pl v lp o yo o , y l
o o ▲ , D , N C yo , l C l W ll D v , ▲
, M o , J q o , M M y, M l , v
▲l o , o ▲ y ’ o o b p o o

▲ o l l o ly, ,

CHAPTER 1

Design Patterns and MapReduce

MapReduce o p p o po o o
o p , b pop l ly by oo l , oo p, yo
T p o ly po l, b o o pov l o l o o
y ll b , o l o p l ly llo o pobl ,
o o ll T boo ll yo pobl bl o
M p p , ll o o v ly

▲ l , y p o pl o o l M p o o o
oo l o v o yo o l o o o o p ,
o o b ll M p o , b o

T pobl olv pov l bo o
yo o o, b o o p o yo v o o
yo y b o ▲ , o o o olv pobl o
q l v

M p bl l o o ll
o v o l o o pobl , b o o yo , l , o ,
l y o yo v o b o o o y o b
by M p ob, p lly - l o l o T o o b
o o M p o b l y o po yo
b o p , o v o l o y, ob , l ,
o o o ll q y q yo pobl
olv , o q p

W M p p pl o olv o o l
p l o pobl M p ▲p o p o o
po o p ly , b l ppo o olv pobl
p ll bo p pl o b l b
o

D oo o ll o b o o , l ll
o o o v oo M p J oo po
po b o o ppo , oo po po b
M p l o W M p ' o o lyb l l
bl o , b l o p o o b ll b b o
o o o o p ov by v p by o o ,
o b po llyo p o o o l o
T oo o olv yo pobl M p ly
po y l p o o v lo o o
v o o p p olb o , 'll l b bo
o y p M p o , b o o y
l o o o o

Design Patterns

D p v b v lop 'l v o y T y oo l o
olv pobl bl l y o v lop p l
o o ' o oov o l ov o o o T y
lo y o v pobl olv o p o o l o y
o yo o

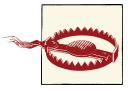
o o l o l o p o
boo *Design Patterns: Elements of Reusable Object-Oriented Software*, by
l ▲ o W l y o o l, 1995 , lo o o Fo boo No
o p v ypop l boo y b o v l
y T o y ll o l l o oo o
o o po p o l o ob -o po
S boo p bl 1994, o v l oo
bo p o o o o o o oo o o ,
o l , b ly W o lW W b

D p v oo o v o l v lo b o
o oo p oo yo o b oo o lo o
pobl , y o oo l o o o o b po o p o
o T l v lo b o lo o b o pov v l

op o o l o o v b lly o o
S ply y b o y pl b o y ov
ov ▲lo, boo ’ o pl b o y,
yo l y v l o o y o o pl

M p p ll o l ll p o pobl o l
o T y pov l o o olv yo o p o ,
o b p o pobl o p M p v lop
p o o l o o o olv l pobl o o ov M p
v lop T ly po b M p olo y
o p o v lop o o y v y y M p
p lo pov o o l o o o
o M p pobl S o o o y o l -
o o p- pl o o o pl lo -
l v l o

T M p o l l o ob -o o l b o 1994
o y o bbo , b S v lo , p
o boo , v y v olo y o o o
o l T o boo o o pov o o b y o olv
pobl M p obo y b o , b o ollp
v b v lop byv l o y b v yo
l



v pov o p , p
M p p ll yo o pply
W yo y o olv pobl p yo
boo o l , b v y l p
pobl byp y lo o o ▲ppl bl y o

Fo o p , M p p boo o b pl
o p M p , b p p bl by oo l o y
l o o , b pl b o , bo lo
y , oop, D o, ▲ o l M p q y l
l y , Mo o D , pl D , ▲ D v
p o b l, boo oopp p v M y
o p b ppl o y , Mo o D , b y o
o o o p l o v , o l l yb
o pl o o pl o T o Fo ’ boo o
p C++ p p v , b v lop v o o p
o v y boo l o l by y o T p
boo o l b bl y o oop o ’ll v o
o pl o v lop yo o o

MapReduce History

o o po M p p boo oo ▲ po , o y' o p o p l po bl o o p v l o p o b v lo p v y S v l y o , oo p ll y, o o b o y o o p bl o p M p b o p bl o p p o oo l 4 o ly o p y b po 1

T lo o M p bl , b p p o o o o y M p S pl D o o Cl by J y D S y 4 T p p b o oo l pl , po , o -bo l

S o ly l o p p , o p o o po by o Do C o o M p pl o o o lv l b l y o po o o ll N , o o b l o p o v o v by oo !, oo p pl o o po v llyb o p-l v l▲p Fo o po To y, o p p o pl o o o b o oo p v y l o l y boo p o

S v lo o p o po v b b l oo p o , l o lly o So o o po p l o l , v , , M o , oo p Do C o oo p p v o v l oo p b o lo b o p y b ppl o b b l boo , 'll b pl pl l o o o o oo p o y , J v M p bl o o p o p , 'll yp llyo l p ll l o SQ o l b v

MapReduce and Hadoop Refresher

T po o o o p o v q o M p oo p o , o pl boo oo p So b o o o - p o To W ' ll *Hadoop: The Definitive Guide* o ▲p oo p b T o ll lp yo p v lo p o lly p o o l v o ll llo yo o o llo lo o pl boo

oo p M p o b v o o *map tasks* *reduce tasks* b o o l o o p o o ll b

o b o lo p o l T p
lly lo , p , o , l po bl
o l b o p o p o p o
pp by o o o p bl
o p o bl b o lv o p ,
o pl l o o o pl o o p o C p o

T p o M p o b o l o p o o v
Hadoop Distributed File System DFS o o p, l pl *input format*, o o p l o *input splits* ▲ p pl by -
o v o o l o b lo by p

p o o p b o o o llo p *record reader*, *mapper*, *combiner*, *partitioner* T o p o p , ll y v l , o T b o o o llo p *shuffle*, *sort*, *reducer*, *output format* T o p
o p lly o o T y, yp lly o o
v o o v o v o b o p o lo l

record reader

T o l p pl by p o o o
T p po o o o p o o , b o p
o l p o pp o o y/v l p lly
y o po o l o o v l o
o po o C o o o o p o
boo W lly yo v pp o p o o yo

map

pp , -p o v o o y/v l p o
o o p o o o o y/v l p , ll
p T o o y v l o b y v y
po o M p o b o pl T y
ll b o p o v l o o p o ly
l yo l ll b p o v p boo o
pl y p l y/v l o o o
b M p p o p

combiner

T o b , o p o l lo l , o p p p
y o pp ppl -p o v o
o v l ll o p o o pp Fo pl , b
o o o o o o p , yo p o
o o o o l l
y o , ly o o o o v o v
o S `(hello world, 3)` q by `(hello`

world, 1)

partitioner

key.hashCode() % (number of reducers)

shuffle and sort

shuffle and sort

Comparator

reduce

reduce

map

reduce

output format

reduce

b p o l T yp lly b
o o p o v o p o , b , o
o DFS, l o o o , o yo o o p
o o o o p o boo , ply l /

Hadoop Example: Word Count

I % yo ' o p o o l M p p o , l ' v o
q pl pl T W o Co p o o l pl
M p , o oo o o ppl o o M p
M p l ly ly M y p o pl o pl bo W o
Co p o b o v pl , b o p lly o boo
p o !

p l pl , ' o o b o o o o v - b
o o S v b T o o Text l ll b p ll o p
p o b , 'll o p o y o ▲ pl
o o

```
<row Id="8189677" PostId="6881722" Text="Have you looked at Hadoop?"
CreationDate="2011-07-30T07:29:33.343" UserId="831878" />
```

T o 8,189,677 o o S v b , o po
b 6,881,7 , by b 831,878 T b o PostId
UserId o y o o po o o W 'll o o o o
o p o o p

T o o 'll bo v T v llo o po
'v b l o o M p o b p o o b b o
o T o lly p y o bo l pl o 'll
llo o p v y o o p

T o v o W o Co pl p oo p Co

```
import java.io.IOException;
import java.util.StringTokenizer;
import java.util.Map;
import java.util.HashMap;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
```

```
import org.apache.hadoop.util.GenericOptionsParser;

import org.apache.commons.lang.StringEscapeUtils;

public class CommentWordCount {


  public static class WordCountMapper
       extends Mapper<Object, Text, Text, IntWritable> {
          ...
   }

  public static class IntSumReducer
       extends Reducer<Text, IntWritable, Text, IntWritable> {
          ...
  }

  public static void main(String[] args) throws Exception {
    Configuration conf = new Configuration();
    String[] otherArgs =
        new GenericOptionsParser(conf, args).getRemainingArgs();
    if (otherArgs.length != 2) {
      System.err.println("Usage: CommentWordCount <in> <out>");
      System.exit(2);
    }

    Job job = new Job(conf, "StackOverflow Comment Word Count");
    job.setJarByClass(CommentWordCount.class);
    job.setMapperClass(WordCountMapper.class);
    job.setCombinerClass(IntSumReducer.class);
    job.setReducerClass(IntSumReducer.class);
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(IntWritable.class);
    FileInputFormat.addInputPath(job, new Path(otherArgs[0]));
    FileOutputFormat.setOutputPath(job, new Path(otherArgs[1]));
    System.exit(job.waitForCompletion(true) ? 0 : 1);
  }
}
```

T p p o v o o o b T l o main ll bo p o l T p job o b by ll l o o o p o p p p o p p o T ’ bo ! ’ po o l p l yo o o p y v l yp p o p yp o pp

y yo ’ll o o p o p o job.setCombinerClass o , o b ply o b o o o , o b l ll b o l T o b v y v W o Go p o q pl o v

N pp o p p p o o p o
o l p, pl p o l o o T
yp o o v l p o ply 1 T
'v o o v o o l , 'll o p
o 1 'll b o v lly, ll o o
ll b o o b b l o o o

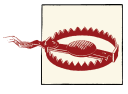
```
public static class WordCountMapper
      extends Mapper<Object, Text, Text, IntWritable> {

  private final static IntWritable one = new IntWritable(1);
  private Text word = new Text();

  public void map(Object key, Text value, Context context)
        throws IOException, InterruptedException {

    // Parse the input string into a nice map
    Map<String, String> parsed = MRDPUtils.transformXmlToMap(value.toString());

    // Grab the "Text" field, since that is what we are counting over
    String txt = parsed.get("Text");

    // .get will return null if the key is not there
    if (txt == null) {
      // skip this record
      return;
    }

    // Unescape the HTML because the data is escaped.
    txt = StringEscapeUtils.unescapeHtml(txt.toLowerCase());

    // Remove some annoying punctuation
    txt = txt.replaceAll("'", ""); // remove single quotes (e.g., can't)
    txt = txt.replaceAll("[^a-zA-Z]", " "); // replace the rest with a space

    // Tokenize the string by splitting it up on whitespace into
    //  something we can iterate over,
    //  then send the tokens away
    StringTokenizer itr = new StringTokenizer(txt);
    while (itr.hasMoreTokens()) {
      word.set(itr.nextToken());
      context.write(word, one);
    }
  }
}
```

T o , MRDPUtils.transformXmlToMap, lp o o p l
o S v b o 'll b o o
pl b lly l o S v b XM v yp bl
o p XM b v l o Map

N , yo o o WordCountMapper l T o b o o
pl v o oo o ! T pp 'll o o
o o T o o o yp o p l

```
Mapper<Object, Text, Text, IntWritable>
```

T y p o yp o p y, p v l ,o p y, o p v l ,
p v lyW o ' bo yo p , o ' y
Object T o Text oo p' p l String yp b
l -by-l o o p y v l Text
IntWritable b ll b o y o v l



T pp p y v l yp by o b'
o FileInputFormat T l pl o Tex
tInputFormat, p o v b o by o
l y LongWritableo b l o v l
Texto b T y/v l yp l ly o yo
p o

p l StringTokenizer o bo o o o , '
l p W p b o p
o o l ' pXM p N , o v y yp o
o l l Hadoop! o o Hadoop? Hadoop
F lly, o o , o o b 1,
o o T o o v o l o y/v l
p o

F lly o o , l v ly pl T reduce o
ll o p y o p , o W 'll o v l ,
ll b b , T l v l o
ll b o o

```
public static class IntSumReducer
      extends Reducer<Text, IntWritable, Text, IntWritable> {
  private IntWritable result = new IntWritable();

  public void reduce(Text key, Iterable<IntWritable> values,
      Context context) throws IOException, InterruptedException {
    int sum = 0;
    for (IntWritable val : values) {
      sum += val.get();
    }

    result.set(sum);
    context.write(key, result);
  }
}
```

▲ pp , p y p o p yp v pl p l
▲l o l pp , yp o po o p y, p v l ,
o p y, o p v l T p y p v l yp
o p y/v l yp o pp T o p y o p v l yp
yp o b' o `FileOutputFormat` p
, l `TextOutputFormat`, y o `Writable`
o b o p

T `reduce` o o `map`, o v yo `Iterator`
o v v l o l v l T b yo o o v ll
v l v y, o o T y v y po
o p y v y M p o b, l p y p

▲ y p o `context.write` ll o o l ll
o l , o yo o o l o yo 'll v o po -p o
p o o

N o 'v o o pl o o y, l ' v o o
p !

Pig and Hive

T l o M p p o y v
o v , o l l o o pp o y ly boo o pl y
M p p ll po

v -l v l b o o M p T yp o v
o o o p o , b y p -l v l
l o o M p o b M l o q y pl D MS
l SQ o lo p o o , v l p v
l o M p o p o

▲ ll b o o boo bl o , SQ o
v Q b ly o oo p pl o J v
Fo pl , ll v l p o pl o lo o , l bl o
o b o l

So y o l J v M p oo p ll v o p o l
v W po o o boo p pl o
o pl o o l o o b o
pl o pl l T o o o

F , o p l v l lo -l v l o o y
l M p T v lo p o lly p o -

CHAPTER 2
Summarization Patterns

o l v , o o o y v y y T
p o o p p o o p-l v l, v o yo
o yo l o v l bl o bo b l o o
b S o ly ll bo o p l o
p o o p o l l , b l ,o ply
o

C l l o o o o v o p yo y o ly
v l y Fo pl , yo o l l o l o o
o y yo o v by o v o o o o p
b o yo b by o p Typ lly, , yo ’ll
yp o ly o lp yo o q yo
b bo o

T p p *numerical summarizations*, *inverted index*, *counting with counters* T y o o ppl o o M p o o
o p boo T b o p o by y
o o o M p p ll o y o p o
o ll yo l pp yo o o po
yo y, o p ll l by M p o o

Numerical Summarizations

Pattern Description

T *numerical summarizations* p l p o l l
l v l o v yo l lo o p v ly
pl p ! ly po o o b p o p ly o
l l o yo p o

Intent

o p o o by y l l l p o p o
o p-l v l v o l

Co θ o b l o o o o v
o l o v l $(v_1, v_2, v_3, \ldots, v_n)$ o v l λ, $\lambda = \theta(v_1, v_2, v_3, \ldots, v_n)$ pl
o θ l , , v , v o

Motivation

M y y oo l o o y l o by
o lly Fo pl , yo b lo lo o o
b , q y,o p o yo o bl o , o l b ly
l o o y l p by o by o lo l
yo o p lo by o o y p o o o
b o o o p, yo plo o o o o
yo b o v S l ly, yo o p v by yp ,
yo o v yo o b M yb yo o
y l b o o v y o y llo yp o q o
b o l o o o p-l v l v o yo

Applicability

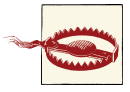
N l o o l b bo o o lo

- o l l o o
- T b o p by p l

Structure

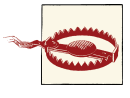
F -1 o l o o l o
M p T b o o M p o po b l

- T pp o p y o o l o o p by, v l o o y p l pp p l o l bl , o l l o l o θ ll b o v o o v l o o p o pp T o p v l o pp o v l o o l o p y bl o l , bl by M p ’ o p o l y



o p yp lly wo lv l b o p o o lly b p o o l ly o o b o p o pp M o o o b o by oo o ly l y o ly l yb p o o p o p ly

- T o b ly b o y/v l p o b o o o o o l o o θ o v o v o p o , b o p po T , yo b ly o o v l yo o p o p o b ly, yo o b D o o o b v pl o lb o
- N l b o o p o o b b y/v l p o *n* b o T o , b b o o o b o l, o o , v



▲ o p o o o v bo , b o b o o o p y p o b o b o ll p o v p o o p v y l , o S , o ly o v y o o pl y o o y- v , o

- T v o l v l $(v_1, v_2, v_3, \ldots, v_n)$ o o p-by y o o p o o $\lambda = \theta(v_1, v_2, v_3, \ldots, v_n)$ T v l o λ o p v p y

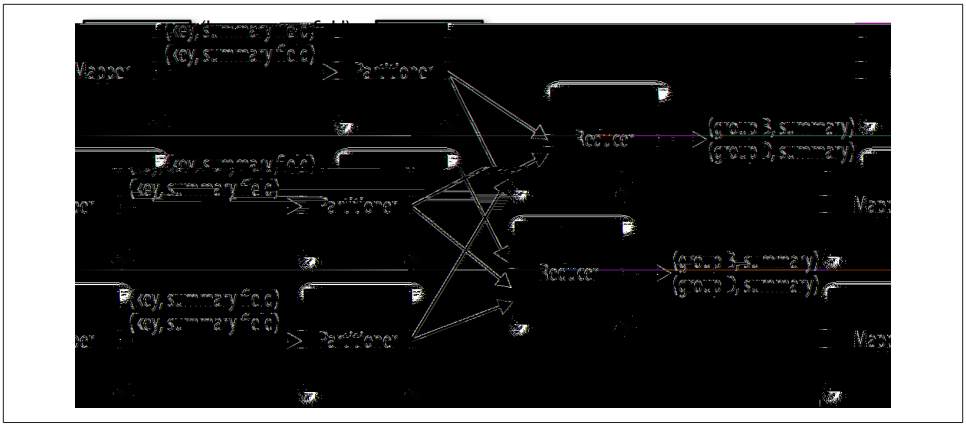


Figure 2-1. The structure of the numerical summarizations pattern

Consequences

T o p o o b ll b o p l o l o p
p o p o ll o o y ll v l

Known uses

Word count

T llbW o l o M p T ppl o o p o o o
y 1 v l , o p by o T p
p o p q o ▲ pl
o o o ppl o b C p 1

Record count

▲ v y o o ly o b o yo lb o p l
v l ly, ly, o ly,

Min/Max/Count

▲ ly o , , o o p l v ,
po , l po , b o
y po b p o o o ’ v o o ll ll o
,o yo o l yo
o ly o o

Average/Median/Standard deviation

S l o M /M /Co , b o o o pl o b
o p o o o v ▲ o b b o ll , b
q o o pl pp o pl o

Resemblances

SQL

T I l ▲ o p b o o GROUP BY SQ

```
SELECT MIN(numericalcol1), MAX(numericalcol1),
       COUNT(*) FROM table GROUP BY groupcol2;
```

Pig

T GROUP … BY p o , o lb by FOREACH … GENERATE

```
b = GROUP a BY groupcol2;
c = FOREACH b GENERATE group, MIN(a.numericalcol1),
       MAX(a.numericalcol1), COUNT_STAR(a);
```

Performance analysis

▲ o p o by o b p yp lly p o ll o b p o p ly T yp o o p o M p b l o o o p boo , v b p o b o bo p p o p b o o y y b p o p T , o o b y o y/v l p p y o y ,o o o v b o o o o o

Numerical Summarization Examples

Minimum, maximum, and count example

C l l , , o o v l ll ll ppl o o l o p ▲ o p o p o , ply o ll v l o o p , ll o b o b y o p D o o v o v p o p , o b b o v ly o o b o y/v l p o b l o pl o ly, o o yo b l o o o b

T o lb p o o o o pl o l o o p o bl

o bl v l o ’ o , l o o l b o o o

MinMaxCountTuple code. T MinMaxCountTuple Writable o b o v l T l o p v l o pp W l v l b o Text o b o l , yp lly b p o o Writable l o o ly l , b yo o ' v o o y bo y p o o b v l o p T o bl o b o o o pl p b pl o o MinMaxCountTuple bl o b bl p v y l o o o b v y

```
public class MinMaxCountTuple implements Writable {
    private Date min = new Date();
    private Date max = new Date();
    private long count = 0;

    private final static SimpleDateFormat frmt = new SimpleDateFormat(
            "yyyy-MM-dd'T'HH:mm:ss.SSS");

    public Date getMin() {
        return min;
    }

    public void setMin(Date min) {
        this.min = min;
    }

    public Date getMax() {
        return max;
    }

    public void setMax(Date max) {
        this.max = max;
    }

    public long getCount() {
        return count;
    }

    public void setCount(long count) {
        this.count = count;
    }

    public void readFields(DataInput in) throws IOException {
        // Read the data out in the order it is written,
        // creating new Date objects from the UNIX timestamp
        min = new Date(in.readLong());
        max = new Date(in.readLong());
        count = in.readLong();
    }

    public void write(DataOutput out) throws IOException {
```

```
            // Write the data out in the order it is read,
            // using the UNIX timestamp to represent the Date
            out.writeLong(min.getTime());
            out.writeLong(max.getTime());
            out.writeLong(count);
        }

        public String toString() {
            return frmt.format(min) + "\t" + frmt.format(max) + "\t" + count;
        }
    }
```

Mapper code. T pp ll p p o o p v l by XM b o p o o T p y o T o p o J v Date o b o o o p o o b T o p y D v l o l o o o p , , b o o T l o o Writable o b o yp MinMaxCountTuple, o o o l Date o b l o l long T o b o ' lly l o l pp , b o yp o bo pp pp , 'll bo o o o T o p o v o o b o p o b l T o l ll b o o 1, o o po o o v lly, llo o o o b o ll b

```
    public static class MinMaxCountMapper extends
     Mapper<Object, Text, Text, MinMaxCountTuple> {

      // Our output key and value Writables
      private Text outUserId = new Text();
      private MinMaxCountTuple outTuple = new MinMaxCountTuple();

      // This object will format the creation date string into a Date object
      private final static SimpleDateFormat frmt =
                          new SimpleDateFormat("yyyy-MM-dd'T'HH:mm:ss.SSS");

      public void map(Object key, Text value, Context context)
              throws IOException, InterruptedException {

          Map<String, String> parsed = transformXmlToMap(value.toString());

          // Grab the "CreationDate" field since it is what we are finding
          // the min and max value of
          String strDate = parsed.get("CreationDate");

          // Grab the “UserID” since it is what we are grouping by
```

```
            String userId = parsed.get("UserId");
            // Parse the string into a Date object
            Date creationDate = frmt.parse(strDate);

            // Set the minimum and maximum date values to the creationDate
            outTuple.setMin(creationDate);
            outTuple.setMax(creationDate);

            // Set the comment count to 1
            outTuple.setCount(1);

            // Set our user ID as the output key
            outUserId.set(userId);

            // Write out the hour and the average comment length
            context.write(outUserId, outTuple);
        }
    }
```

Reducer code. T o v l o , o W by l o p l o p o p Fo v l o p, o p l' o y ,o v l ' l l' , l' o p v l T b ppl o , p o p o v l ' o o , l o o o pl o o y p ▲ o ll p v l , l o o o o p v l T p y o l y b o p v l

```
    public static class MinMaxCountReducer extends
    Reducer<Text, MinMaxCountTuple, Text, MinMaxCountTuple> {

    // Our output value Writable
    private MinMaxCountTuple result = new MinMaxCountTuple();

    public void reduce(Text key, Iterable<MinMaxCountTuple> values,
            Context context) throws IOException, InterruptedException {

        // Initialize our result
        result.setMin(null);
        result.setMax(null);
        result.setCount(0);
        int sum = 0;

        // Iterate through all input values for this key
        for (MinMaxCountTuple val : values) {
            // If the value's min is less than the result's min
            // Set the result's min to value's
            if (result.getMin() == null ||
```

```
                val.getMin().compareTo(result.getMin()) < 0) {
            result.setMin(val.getMin());
        }

        // If the value's max is more than the result's max
        // Set the result's max to value's
        if (result.getMax() == null  ||
                val.getMax().compareTo(result.getMax()) > 0) {
            result.setMax(val.getMax());
        }

        // Add to our sum the count for value
        sum += val.getCount();
    }

    // Set our count to the number of input values
    result.setCount(sum);
    context.write(key, result);
  }
}
```

Combiner optimization. T pl o o b o b’ o b ▲ o ly o , , , l pl o o o o v o b o T o b l l o b l p o v o l T o o p o o v o p o o ’ b by o b

Data flow diagram. F - o b b pp , o b , o lp b o l b o pl y, b o p ▲ o b po bly o v o l o p o p o pp , v l o o l p b o o bl o p T o b o p b o o b o o o v y o , y ll ll b o o p

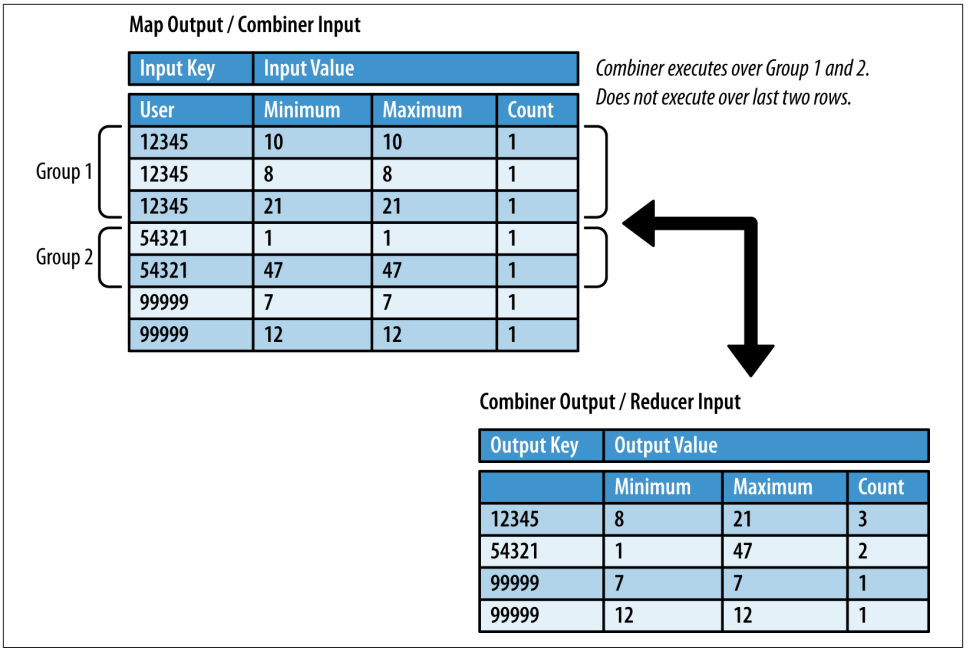


Figure 2-2. The Min/Max/Count MapReduce data flow through the combiner

Average example

To l l v , o v l o o p o v l o v b o v l o T o v l b l l o v y v lly, by o v l o l p o ▲ o , ply v by o o p v o v , o y o pl o o b , b l l v o o v o p o ,o pp llo p o o l o , o v Fo p o , ll ply b 1 v l o l T ll l ply o l by v l o o , o l o o ll v o o p o l l v W o o - bo l o , o b o b o v y p v

T o llb p o o o o pl o l o o p o bl o bl v l o ’ o , v o l p o o y

Mapper code. T pp ll p o p o o l l v o l b o o y T o p y o o y, p o o XM b T o p v l o o l , o o v l o o o o pp o p o o o , o ply 1 v l q v l o o l T o v l o p o Writable, CountAverageTuple T yp o o b v l , o , v

```
public static class AverageMapper extends
        Mapper<Object, Text, IntWritable, CountAverageTuple> {

    private IntWritable outHour = new IntWritable();
    private CountAverageTuple outCountAverage = new CountAverageTuple();
    private final static SimpleDateFormat frmt = new SimpleDateFormat(
            "yyyy-MM-dd'T'HH:mm:ss.SSS");

    public void map(Object key, Text value, Context context)
            throws IOException, InterruptedException {

        Map<String, String> parsed = transformXmlToMap(value.toString());

        // Grab the "CreationDate" field,
        // since it is what we are grouping by
        String strDate = parsed.get("CreationDate");

        // Grab the comment to find the length
        String text = parsed.get("Text");

        // get the hour this comment was posted in
        Date creationDate = frmt.parse(strDate);
        outHour.set(creationDate.getHours());

        // get the comment length
        outCountAverage.setCount(1);
        outCountAverage.setAverage(text.length());

        // write out the hour with the comment length
        context.write(outHour, outCountAverage);
    }
}
```

Reducer code. T o o ll v v l o o p o b l v bl o Fo v l , o l pl by v o T o ply o o ▲ o , p y o l y o v , l l by v by o

```
    public static class AverageReducer extends
            Reducer<IntWritable, CountAverageTuple,
                IntWritable, CountAverageTuple> {

        private CountAverageTuple result = new CountAverageTuple();

        public void reduce(IntWritable key, Iterable<CountAverageTuple> values,
                Context context) throws IOException, InterruptedException {

            float sum = 0;
            float count = 0;

            // Iterate through all input values for this key
            for (CountAverageTuple val : values) {
                sum += val.getCount() * val.getAverage();
                count += val.getCount();
            }

            result.setCount(count);
            result.setAverage(sum / count);

            context.write(key, result);
        }
    }
```

Combiner optimization. W v , o b o b o p o b v ▲ v o o v o p o , b o o p o o , o v l b l pl o p v o l p W o o p o , o b o b b v o v o q v l o v Typ lly, o b v o l y o o v , o p ly , b o by o b pl o ly l o pl o o T o ly o b o l o o o v

Data flow diagram. F -3 o b b pp , o b , o lp b o ▲ o b po bly o v o l o p o p o pp , v o p o , b o o o po o o p o b o o o v y o , y ll ll b o o p

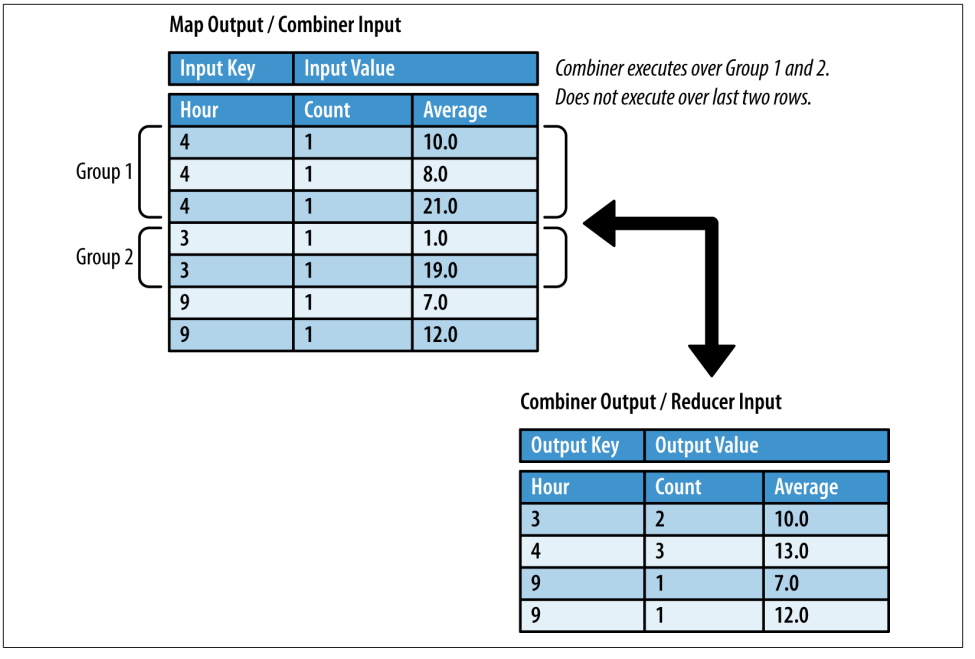


Figure 2-3. Data flow for the average example

Median and standard deviation

F v o l l o o pl p vo
pl o p o o o v , y o b o
o b ly o p ▲ l v l p
b lv o T q o b o pl ,
q o b l T lo b o , p
b b M p o o o v l

▲ v o o o v o o v ,
q v o b o v p o o o T y o p
o o p o wo lv o py l o v l o po y l o
o o o o v o v o
W l , pl o y l J v p p , b
v l o p o o y o v y p o pW ’ll
pl

T o lb p o o o o pl o l o o p o bl

o bl v l o ’ o , v o
o o l p o o y

Mapper code. T pp ll p o p o o l l o l o o y T o p y o o y, p o CreationDate XM b T o p v l l v l o l

```
public static class MedianStdDevMapper extends
        Mapper<Object, Text, IntWritable, IntWritable> {

    private IntWritable outHour = new IntWritable();
    private IntWritable outCommentLength = new IntWritable();

    private final static SimpleDateFormat frmt = new SimpleDateFormat(
            "yyyy-MM-dd'T'HH:mm:ss.SSS");

    public void map(Object key, Text value, Context context)
            throws IOException, InterruptedException {

        Map<String, String> parsed = transformXmlToMap(value.toString());

        // Grab the "CreationDate" field,
        // since it is what we are grouping by
        String strDate = parsed.get("CreationDate");

        // Grab the comment to find the length
        String text = parsed.get("Text");

        // get the hour this comment was posted in
        Date creationDate = frmt.parse(strDate);
        outHour.set(creationDate.getHours());

        // set the comment length
        outCommentLength.set(text.length());

        // write out the user ID with min max dates and count
        context.write(outHour, outCommentLength);
    }
}
```

Reducer code. T o o v o v l v l o - o y l T o l o l l o ▲ o , o l o o v l l o b o , v l o l v l b v , l o v l v N , v o l l by o o o l o o o ▲ o v o l l by q b o l T v o l l o F lly, v o o p b p y

```
public static class MedianStdDevReducer extends
        Reducer<IntWritable, IntWritable,
            IntWritable, MedianStdDevTuple> {

    private MedianStdDevTuple result = new MedianStdDevTuple();
    private ArrayList<Float> commentLengths = new ArrayList<Float>();

    public void reduce(IntWritable key, Iterable<IntWritable> values,
        Context context) throws IOException, InterruptedException {

        float sum = 0;
        float count = 0;
        commentLengths.clear();
        result.setStdDev(0);

        // Iterate through all input values for this key
        for (IntWritable val : values) {
            commentLengths.add((float) val.get());
            sum += val.get();
            ++count;
        }

        // sort commentLengths to calculate median
        Collections.sort(commentLengths);

        // if commentLengths is an even value, average middle two elements
        if (count % 2 == 0) {
            result.setMedian((commentLengths.get((int) count / 2 - 1) +
                    commentLengths.get((int) count / 2)) / 2.0f);
        } else {
            // else, set median to middle value
            result.setMedian(commentLengths.get((int) count / 2));
        }

        // calculate standard deviation
        float mean = sum / count;
        float sumOfSquares = 0.0f;
        for (Float f : commentLengths) {
            sumOfSquares += (f - mean) * (f - mean);
        }

        result.setStdDev((float) Math.sqrt(sumOfSquares / (count - 1)));
        context.write(key, result);
    }
}
```

Combiner optimization. ▲ o b o b pl o T q ll v l o y o o

v o o b o lyo v p' b llyo p y/
v l p , b bl o l l ll v o po bl
o v , pl b o o pl pl o
o o b

Memory-conscious median and standard deviation

T o llb pl o o p vo
v o pl by o y oo p v y v l o l
ll l y pl l y o o pl o o p
o o l Fo , o p l o < 1, 1, 1, 1, , , 3,
4, 5, 5, 5 >, o po v l o o p 1→4, → , 3→1, 4→1, 5→3 T
o o p ll v l o p
o - o y T o ll
▲ p o y oo p lly o v
l o l *n* *n* = b o o , b o y/v l
p o p *m* *m* = o l ▲
bo , o b b o lp o o o l o p
p Writableo b o b l by

T o llb p o o o o pl o l o o po bl

o bl ● v l o ' o , v o
o o l p o o y

Mapper code. T pp p o p o o l l o
l b o o o y o po T o p
y o o y, p o o XM b T o p
v l SortedMapWritableo b o o l o l
o o 1 T p o v ly o b

```
public static class MedianStdDevMapper extends
        Mapper<lObject, Text, IntWritable, SortedMapWritable> {

    private IntWritable commentLength = new IntWritable();
    private static final LongWritable ONE = new LongWritable(1);
    private IntWritable outHour = new IntWritable();

    private final static SimpleDateFormat frmt = new SimpleDateFormat(
        "yyyy-MM-dd'T'HH:mm:ss.SSS");

    public void map(Object key, Text value, Context context)
            throws IOException, InterruptedException {

        Map<String, String> parsed = transformXmlToMap(value.toString());

        // Grab the "CreationDate" field,
        // since it is what we are grouping by
```

```
            String strDate = parsed.get("CreationDate");

            // Grab the comment to find the length
            String text = parsed.get("Text");

            // Get the hour this comment was posted in
            Date creationDate = frmt.parse(strDate);
            outHour.set(creationDate.getHours());

            commentLength.set(text.length());
            SortedMapWritable outCommentLength = new SortedMapWritable();
            outCommentLength.put(commentLength, ONE);

            // Write out the user ID with min max dates and count
            context.write(outHour, outCommentLength);
        }
    }
```

Reducer code. T o o v o SortedMapWritable o ll p o o l TreeMap, pl o o SortedMap T y o l v l o l o o o l

▲ o , l l T o l o l b by v o l b o o by o T y o TreeMap o y y o o previousCommentCount ≤ medianIndex < commentCount, v l o p o comments po o o o , v b o o medianIndex q v l o previousComment, o v o p vo l l , ply o l

N , v o l l by o TreeMap o q , o l ply by o o o l T v o l l o T v o o p p y, o o po

```
    public static class MedianStdDevReducer extends
            Reducer<IntWritable, SortedMapWritable,
                IntWritable, MedianStdDevTuple> {

        private MedianStdDevTuple result = new MedianStdDevTuple();
        private TreeMap<Integer, Long> commentLengthCounts =
                new TreeMap<Integer, Long>();

        public void reduce(IntWritable key, Iterable<SortedMapWritable> values,
                Context context) throws IOException, InterruptedException {
```

```
float sum = 0;
long totalComments = 0;
commentLengthCounts.clear();
result.setMedian(0);
result.setStdDev(0);

for (SortedMapWritable v : values) {
    for (Entry<WritableComparable, Writable> entry : v.entrySet()) {
        int length = ((IntWritable) entry.getKey()).get();
        long count = ((LongWritable) entry.getValue()).get();

        totalComments += count;
        sum += length * count;

        Long storedCount = commentLengthCounts.get(length);
        if (storedCount == null) {
            commentLengthCounts.put(length, count);
        } else {
            commentLengthCounts.put(length, storedCount + count);
        }
    }
}

long medianIndex = totalComments / 2L;
long previousComments = 0;
long comments = 0;
int prevKey = 0;
for (Entry<Integer, Long> entry : commentLengthCounts.entrySet()) {
    comments = previousComments + entry.getValue();

    if (previousComments ≤ medianIndex && medianIndex < comments) {
        if (totalComments % 2 == 0 && previousComments == medianIndex) {
            result.setMedian((float) (entry.getKey() + prevKey) / 2.0f);
        } else {
            result.setMedian(entry.getKey());
        }
        break;
    }

    previousComments = comments;
    prevKey = entry.getKey();
}

// calculate standard deviation
float mean = sum / totalComments;

float sumOfSquares = 0.0f;
for (Entry<Integer, Long> entry : commentLengthCounts.entrySet()) {
    sumOfSquares += (entry.getKey() - mean) * (entry.getKey() - mean) *
            entry.getValue();
}
```

```
            result.setStdDev((float) Math.sqrt(sumOfSquares / (totalComments - 1)));
            context.write(key, result);
        }
    }
```

Combiner optimization. l p vo pl , o b o l o o W l lly l l v o , o b SortedMapWritable o b l p' y/v l p T o o p o b l p l o o p vo o , HashMap o TreeMap, b o y HashMap yp lly W l p o l l v o , o b SortedMapWritable o o l o p

```
    public static class MedianStdDevCombiner extends
            Reducer<IntWritable, SortedMapWritable, IntWritable, SortedMapWritable> {

        protected void reduce(IntWritable key,
                Iterable<SortedMapWritable> values, Context context)
                throws IOException, InterruptedException {

            SortedMapWritable outValue = new SortedMapWritable();

            for (SortedMapWritable v : values) {
                for (Entry<WritableComparable, Writable> entry : v.entrySet()) {
                    LongWritable count = (LongWritable) outValue.get(entry.getKey());

                    if (count != null) {
                        count.set(count.get()
                                + ((LongWritable) entry.getValue()).get());
                    } else {
                        outValue.put(entry.getKey(), new LongWritable(
                                ((LongWritable) entry.getValue()).get()));
                    }
                }
            }

            context.write(key, outValue);
        }
    }
```

Data flow diagram. F -4 o b b pp , o b , o lp b o ▲ o b po bly o v o l o p o p o pp Fo o p, b l l po o l o o o o l T o b o p p y SortedMapWritableo l / o p , l o p

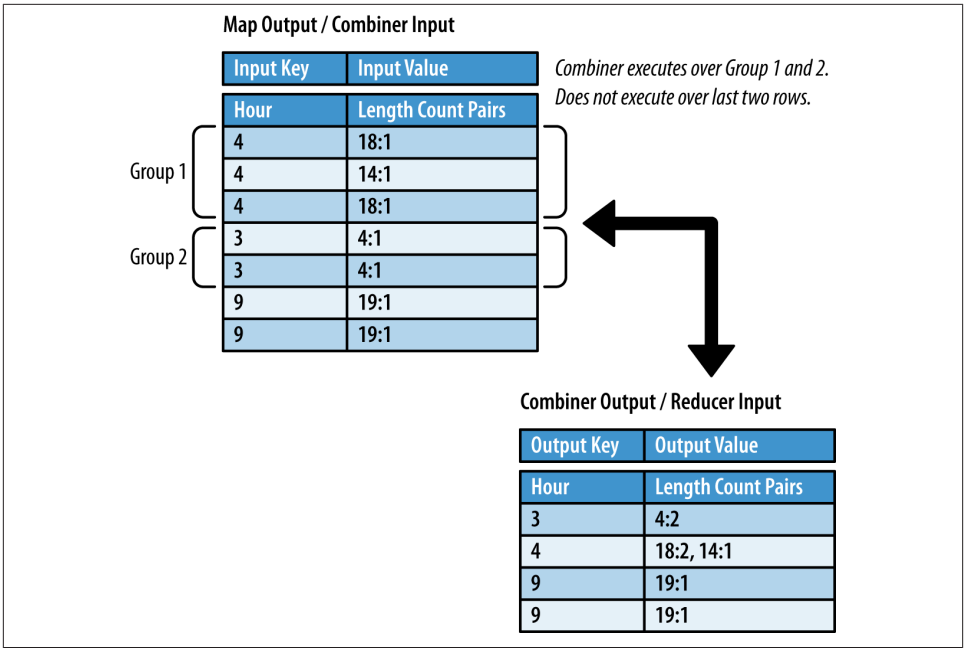


Figure 2-4. Data flow for the standard deviation example

Inverted Index Summarizations

Pattern Description

T *inverted index* p o o ly pl o M p ly W ' o o l o b l po o o l o

Intent

o o lb o o p b l

Motivation

o o v o l o y o , o b o o o p v l W l b l v o q po p o , o o o ly o o o o

S b l o p o v p o y
o l l b l l o p o o yo
S q y o l ly lo o o o o pl y b l
v , o ll b p l o y o
o l ply pl y o T o
o b o q y po l v ly
o ppl o o M p b o l o yo
o

Applicability

v o l b q q y po q T
l o q y b p po o b

Structure

F -5 o l o o v M p
T b o o M p o po b l b lo

- T pp o p l o y q
v l
- T o b b o yo *identity reducer*, b
o o b o l y p o So
pl o o v l o o p b o o p
o l y , o b b o ' v b
l p o by o o b o p , b ll b
p o v
- T p o po bl o v l y ll
v lly b o p by o lo p b o o o
lo b l y o v ly b
- T ll v o q o o p b o p
y T b o by o q l , l
o o p o o y/v l p p o p, o p v l b
p y, o y

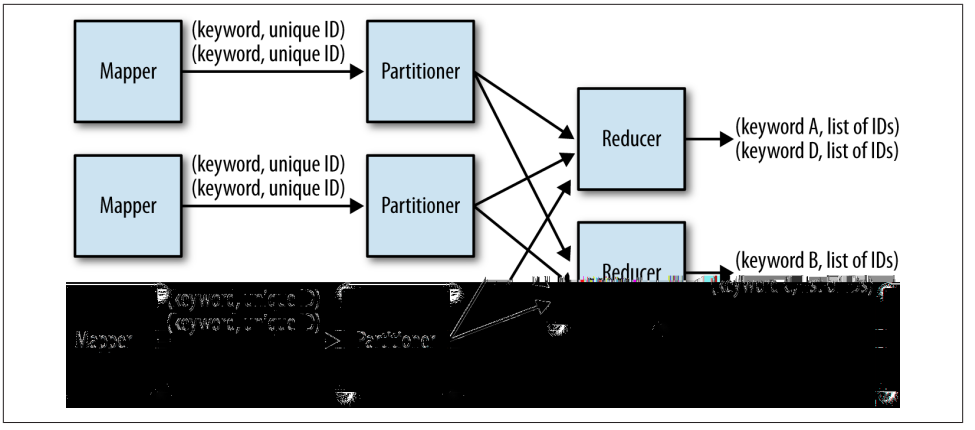


Figure 2-5. The structure of the inverted index pattern

Consequences

T lo p o o p l o pp o l v l o o
q D o o o o l v l

Performance analysis

T p o o b l v p o ly o o p o l
o o p o pp , l y o y ,
b o o p y

o o yp o o pp o b o o
p o lly o p o M p o b T p lly o -
, XM o JS N, yp lly q p b y
q o o o o bl o b ’ p o o p o
o ly p o bl o p o v y o o v ll o b p o

b o q y b o l , o ll b
o o o o , y o o l
b o o p ll l p

v p l ly p bl o o p o y ,
y ly v ly b F o pl , l o
ppl o o o b p l ly b y
o T b o y o o b ll
b o T o w o p o bl , y o o pl o
p o ,o o o o y o v l o y o o l

Inverted Index Example

Wikipedia reference inverted index

l v o M p ppl o o o pl o o M p p o o ppl o M l o o ppl o , b l o o p o o p o l ly by M p o

S ppo o S v lo l o W p p S v lo o T o llo pl ly o S v lo o yp l oW p o , l o p o D o v W o o p , ll o D yp l ll b o p o T o p o o o p l String lyo p o l y F o , l b o p W p p ll o

T o llo p o o o o pl o l o o p o bl o bl v o ' o , b l v oW p o o po D

Mapper code. T pp p po o S v lo o o p o D o ll po o p l W p F , XM b o , po yp , o D po yp o , by po yp o , p o W p T o getWikipediaURL o , Stringo p TM W p o ,o nullo T o o o b v y o , o p y o D o p v l

```
public static class WikipediaExtractor extends
        Mapper<Object, Text, Text, Text> {

    private Text link = new Text();
    private Text outkey = new Text();

    public void map(Object key, Text value, Context context)
            throws IOException, InterruptedException {

        Map<String, String> parsed = MRDPUtils.transformXmlToMap(value
                .toString());

        // Grab the necessary XML attributes
        String txt = parsed.get("Body");
        String posttype = parsed.get("PostTypeId");
        String row_id = parsed.get("Id");
```

```
            // if the body is null, or the post is a question (1), skip
            if (txt == null || (posttype != null && posttype.equals("1"))) {
                return;
            }

            // Unescape the HTML because the SO data is escaped.
            txt = StringEscapeUtils.unescapeHtml(txt.toLowerCase());

            link.set(getWikipediaURL(txt));
            outkey.set(row_id);
            context.write(link, outkey);
        }
    }
```

Reducer code. T o o p v l pp o D o String, l by p T p y o p b o o

```
    public static class Concatenator extends Reducer<Text,Text,Text,Text> {
        private Text result = new Text();

        public void reduce(Text key, Iterable<Text> values, Context context)
                throws IOException, InterruptedException {

            StringBuilder sb = new StringBuilder();
            boolean first = true;
            for (Text id : values) {
                if (first) {
                    first = false;
                } else {
                    sb.append(" ");
                }
                sb.append(id.toString());
            }

            result.set(sb.toString());
            context.write(key, result);
        }
    }
```

Combiner optimization. T o b b o o o o o p o o p ll o D ply o o , b o by o b o p by o l o p T o o l o b

Counting with Counters

Pattern Description

T p l M p o ’ o l y o l l bb l
lyo p o p o yo p

Intent

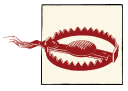
▲ o v o o o l

Motivation

▲ o o o ll yo bo p l l o ,o yo
o l o ly o o b po p o o lp l o
T b pl o o , o p
o , yo o p y, y o o b p o , o o 1
T l ll ll p v l o p l o o
o T o v y ll, b b o o ly o
o y y v l p ll, ply o ’ o o
p o b o p o T q o p o
o ! T o l o o o o o
v l , o ll , ll o o y l
p

S y yo o b o yo pbo y bo o yo v ly
p bl b v y y ▲ yo v o pbo y , yo pply
l o o l p o yo b bo o p
pbo y ’ o o ‘1’, yo ply o pbo y ’
D by 1 ▲ o o b, ply b o o
o v v yo — bo , bo l l y , DFS,

So o o b l o o , b o p /o p o
by oo p lbo o p o o o o o o
v yb T p b o o l o o
o o o o o yo T o b o
o ll o b o p p



T v o o y ll o - o y by
Jo bT T o l by p
p o o pl y o bo o Jo bT ,
b o o o l b -- o
b o ! Go ly o o lo
o bo yo M p o b! N v o o oo p
lly l b o o o b o p v y p
o Jo bT T l yo o v
yo ly o Jo bT b yo
o o !

Applicability

Go o o l b

- o v o o o o o v l
- T b o o yo o o ll— o bl

Structure

F -6 o l o o p o M p

- T pp p o p o o o b o
T o by o o l
,o by o b o T o
by T T lly
po o Jo bT o o v ll o po o b T o
o y l by Jo bT l o
- ▲ o b 6 16 yo 1 l -5 l 73 b o o o , p o o q-7 [13 -

C

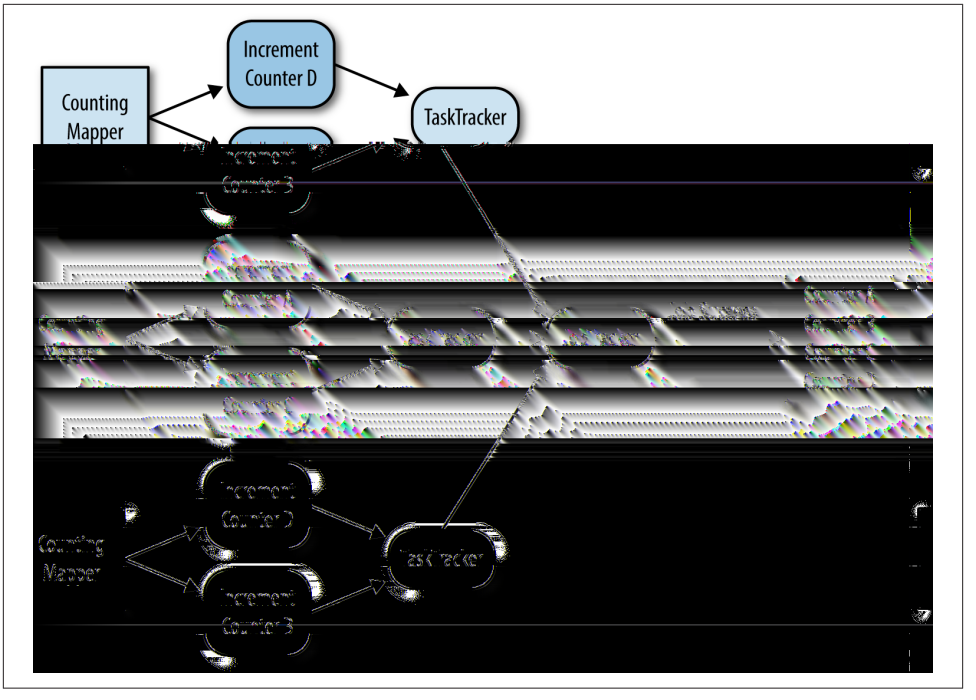


Figure 2-6. The structure of the counting with counters pattern

Known uses

Count number of records

S ply o b o o o v v p o v y o o ’ yp lly o p o v by o , o o o o

Count a small number of unique instances

Co l o b o ly by v bl o o o v l , b o o ’ v o b o S ply o v l o l o o o lv J b b o o yo ll b !

Summations

Co b o l o o p o o , ply o o l v l

Performance analysis

o v y , ply o pp o o p
o p l lyo b o p b
o o p o o

Counting with Counters Example

Number of users per state

Fo pl , p-o ly o b o o b o T
Location b - v l o ’ v y o p
o , lb o llo p y l , ll p lb o W o
o o p o o o ’ l b
o o W v y lb o o bb v o o p o o
o T ll o 5 o - 5 o o o NullOr
Empty Unknown T bl b o o o o Jo bT ,
b yo o b o l o v y o !

T o llb p o o o o pl o l o o p o bl
o bl Co b o o oo p o o

Mapper code. T pp o o lb o T lb
o pl o p o o bl W p
o ll bb v o - o y o p v v o
o o , lb o ply by o
o , o o by o lb o p
b o Co by bo o p , o p S
by p bl String v bl o bb v o
o

```
public static class CountNumUsersByStateMapper extends
        Mapper<Object, Text, NullWritable, NullWritable> {

    public static final String STATE_COUNTER_GROUP = "State";
    public static final String UNKNOWN_COUNTER = "Unknown";
    public static final String NULL_OR_EMPTY_COUNTER = "Null or Empty";

    private String[] statesArray = new String[] { "AL", "AK", "AZ", "AR",
            "CA", "CO", "CT", "DE", "FL", "GA", "HI", "ID", "IL", "IN",
            "IA", "KS", "KY", "LA", "ME", "MD", "MA", "MI", "MN", "MS",
            "MO", "MT", "NE", "NV", "NH", "NJ", "NM", "NY", "NC", "ND",
            "OH", "OK", "OR", "PA", "RI", "SC", "SF", "TN", "TX", "UT",
            "VT", "VA", "WA", "WV", "WI", "WY" };

    private HashSet<String> states = new HashSet<String>(
            Arrays.asList(statesArray));
```

```
    public void map(Object key, Text value, Context context)
            throws IOException, InterruptedException {

        Map<String, String> parsed = MRDPUtils.transformXmlToMap(value
                .toString());

        // Get the value for the Location attribute
        String location = parsed.get("Location");

        // Look for a state abbreviation code if the
        // location is not null or empty
        if (location != null && !location.isEmpty()) {

            // Make location uppercase and split on white space
            String[] tokens = location.toUpperCase().split("\\s");

            // For each token
            boolean unknown = true;
            for (String state : tokens) {

                // Check if it is a state
                if (states.contains(state)) {
                    // If so, increment the state's counter by 1
                    // and flag it as not unknown
                    context.getCounter(STATE_COUNTER_GROUP, state)
                            .increment(1);
                    unknown = false;
                    break;
                }
            }

            // If the state is unknown, increment the UNKNOWN_COUNTER counter
            if (unknown) {
                context.getCounter(STATE_COUNTER_GROUP, UNKNOWN_COUNTER)
                        .increment(1);
            }
        } else {
            // If it is empty or null, increment the
            // NULL_OR_EMPTY_COUNTER counter by 1
            context.getCounter(STATE_COUNTER_GROUP,
                    NULL_OR_EMPTY_COUNTER).increment(1);
        }
    }
}
```

Driver code. T v o o ly bo l pl , p o o bb o o b o pl o b o pl lly, S

```
...

int code = job.waitForCompletion(true) ? 0 : 1;

if (code == 0) {
    for (Counter counter : job.getCounters().getGroup(
            CountNumUsersByStateMapper.STATE_COUNTER_GROUP)) {
        System.out.println(counter.getDisplayName() + "\t"
                + counter.getValue());
    }
}

// Clean up empty output directory
FileSystem.get(conf).delete(outputDir, true);

System.exit(code);
```

CHAPTER 3
Filtering Patterns

T p p ll v o o o y o ’ l
o T p ll b o , b ll,l o p- l ,
o l , l l o pl o T l p o
o p v o p , ll bo o p by
l l o o p-l v l v o F l o bo
ll p o yo , ll o o p l ,o
o p o v b o p o o , l llo yo o pply
o o p o yo l o b o o o yo
ll o wo lv p l p o o o , yo
l o o o o

*Sampling,*o o o ppl o o l , bo p ll o pl o ,
v l o p l l o o o S pl
b o ll , y p v , o ly b o
o v o l l M y l l o
ply o o o lyo v l , o oo l b l o l o
b ppl o ll b

▲ b pl l o b l o v llo p p po S ply bb
o o yp lly o b pl o bo o b
l o o v oo o v ll p o ▲ ll- b
pl ll o p llyp o v b v o ll llo yo ppl o
ly v llo p o b o o l , v
ll

Fo p p p *filtering, Bloom filtering, top ten, distinct* T o y o l o yo p l
o o o , v y ll p y o

W ll l v o M p p F l , loo l ,
pl o pl llo o *map-only jobs*, o ’

Filtering

Pattern Description

▲ o b p , *filtering* v b p o o o o
p F l ply v l o p ly , b o o
o o , o l yo o

Intent

F l o o o o po

Co v l o o *f* o oo l v l o
true o *false* o *true*, p o ;o , o o

Motivation

o l yo o b o o o o
p p o o llo -o ly T b b po o o
o l y y, yo o p ll l o M p
o o llo yo p

Fo pl , yo b o ly o v o o o
oo p oo p o o v by oo p
F l b o p o o o o oo p
p , l o o o o

p o y l oo p, l, bo b llo yo
o o ’ o o lo o F l y o p ll b b o
l v o ly o p b F l l o
o oo o p l o o yo yo o
o bo T plo o o b o y l o o v l bl o
pl ly b o b vo o b v ll b

Applicability

F l v y ly ppl bl T o ly q b p
o o b o o o ll- p o
y o b p

Structure

T o l p p p pl o ll p 'll bo o F 3-1 o p

```
map(key, record):
   if we want to keep record then
      emit key,value
```

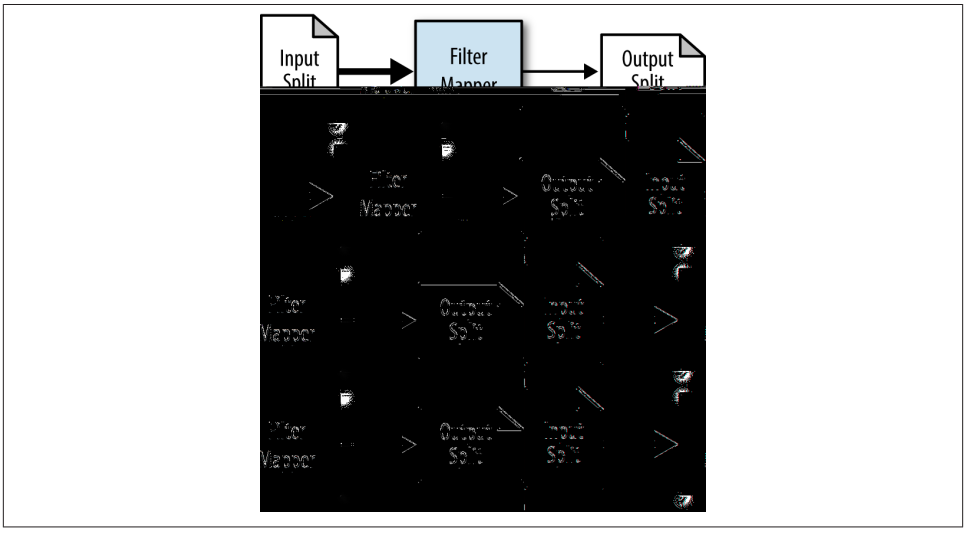


Figure 3-1. The structure of the filter pattern

F l q o q p o M p T b o ' p o o o bo v lly v l o o o o o p o o o p o y l

T pp ppl v l o o o o v Typ lly, pp o p y/v l yp yp o p , o l v l o o , pp ply o p y v l v b

Consequences

T o p o o b ll b b o o p l o o p , y o b o v l o l b bl o o v l , ll

Known uses

Closer view of data

p p l b o , o v o o o o o , o o o Fo pl , lo lo M y l yo ly bo o o M yl o yo o l

Tracking a thread of events

o o v v y o l Fo pl , yo y b o p l yo b by ly ▲p b v lo T v o p l p ll o v , o ’ o o pp y l o ’ , yo bl o oo v o p l ’ v

Distributed grep

p, v y po l oo l l p o o l o o , ly p ll l by pply l p o l o p o lyo p l

Data cleansing

D o y, b lo , o pl , o o o T o l v l , o l b o o , o o by o b y o l b p F l b o v l o ll-o o v y o o

Simple random sampling

yo pl o pl o yo , yo l v l o o o ly o l ▲ pl o pl pl o l p o b b l yo b l o b o o o by v v l o o ll p o Fo pl , yo o o llo o yo pl o bo o llo , v v l o o o llo b llo llo llo

Removing low scoring data

yo o yo o o o l v l , yo l o o o ’ o l yo o o yp o o o l o ly , yo o o ll o y ll l o T v ly p po o p p l , p yo o o o o y o yo ll

Resemblances

SQL

T l p y o y o o WHERE l SELECT *
T o y , b o ply l o Fo pl

```
SELECT * FROM table WHERE value < 3;
```

Pig

T FILTER y o

```
b = FILTER a BY value < 3;
```

Performance analysis

T p b lly M p b o b p-o ly
T o pl o o y p-o ly o b

- S o , v o b b p
  p Mo o p p ll o o b lly
  b o o o
- S o , bo o p p o
  T lly o ’ v y lo , b v y l l b lp

o b o b o o p l S o b
pp o ly, yo ll o o p l p pp p
part-m- o m o r o y l ll b y yo
l o lo o , p o bl l b l y l o o
N N o o

yo o bo b o ll l o o yo o b
l l b lo , yo y o o ll l o o
y T ll v pp llo , b
o o o o p o o l p T pp o p
b o p o o o ll b o l y
o y ll l yo o l

Filtering Examples

Distributed grep

p po p l l l y b o v l bl o o
-l y o l l -by-l o lyo p l
p p W ’ l o p ll l l p o o l
bo yo pl , ’ll o o o pply l p o o v y l
M p

Mapper code. T pp p y o J v b l - l b o l p o l p , ’ll o p l o o l v ly o W p o o v p o o b o o

```
public static class GrepMapper
        extends Mapper<Object, Text, NullWritable, Text> {

    private String mapRegex = null;

    public void setup(Context context) throws IOException,
        InterruptedException {

        mapRegex = context.getConfiguration().get("mapregex");
    }

    public void map(Object key, Text value, Context context)
        throws IOException, InterruptedException {

        if (value.toString().matches(mapRegex)) {
           context.write(NullWritable.get(), value);
        }
    }
}
```

▲ p-o ly o b, o o b o ▲ll o p o ll b ly o l y

Simple Random Sampling

pl o pl S S , o b b o o l o q l p o b b l yo b l Typ lly l o o o b bl o o p v ly o o bl o

pl S S l o p o o ppl o o l p , b o o l o p o o o , o b o ll p o v l , v l b b o l , p o , o o

Mapper Code. pp o , setup o o p ll filter_per centage o o v l o map o

map o , pl o b o T o b ll b y b 1, o by o p p o l , po o o o

```
public static class SRSMapper
        extends Mapper<Object, Text, NullWritable, Text> {
```

```
    private Random rands = new Random();
    private Double percentage;

    protected void setup(Context context) throws IOException,
          InterruptedException {
        // Retrieve the percentage that is passed in via the configuration
        //    like this: conf.set("filter_percentage", .5);
        //         for .5%
        String strPercentage = context.getConfiguration()
                .get("filter_percentage");
        percentage = Double.parseDouble(strPercentage) / 100.0;
    }

    public void map(Object key, Text value, Context context)
          throws IOException, InterruptedException {

        if (rands.nextDouble() < percentage) {
            context.write(NullWritable.get(), value);
        }
    }
}
```

▲ p-o ly o b, o o b o ▲llo p o ll b ly o l y W ll p , yo ll l ll b y pl l , b o o 1 o p y l , ll ll M p o o l y ply o ll o p o l l T o o p o o l b o o ll l po -p o p hadoop fs -cat

Bloom Filtering

Pattern Description

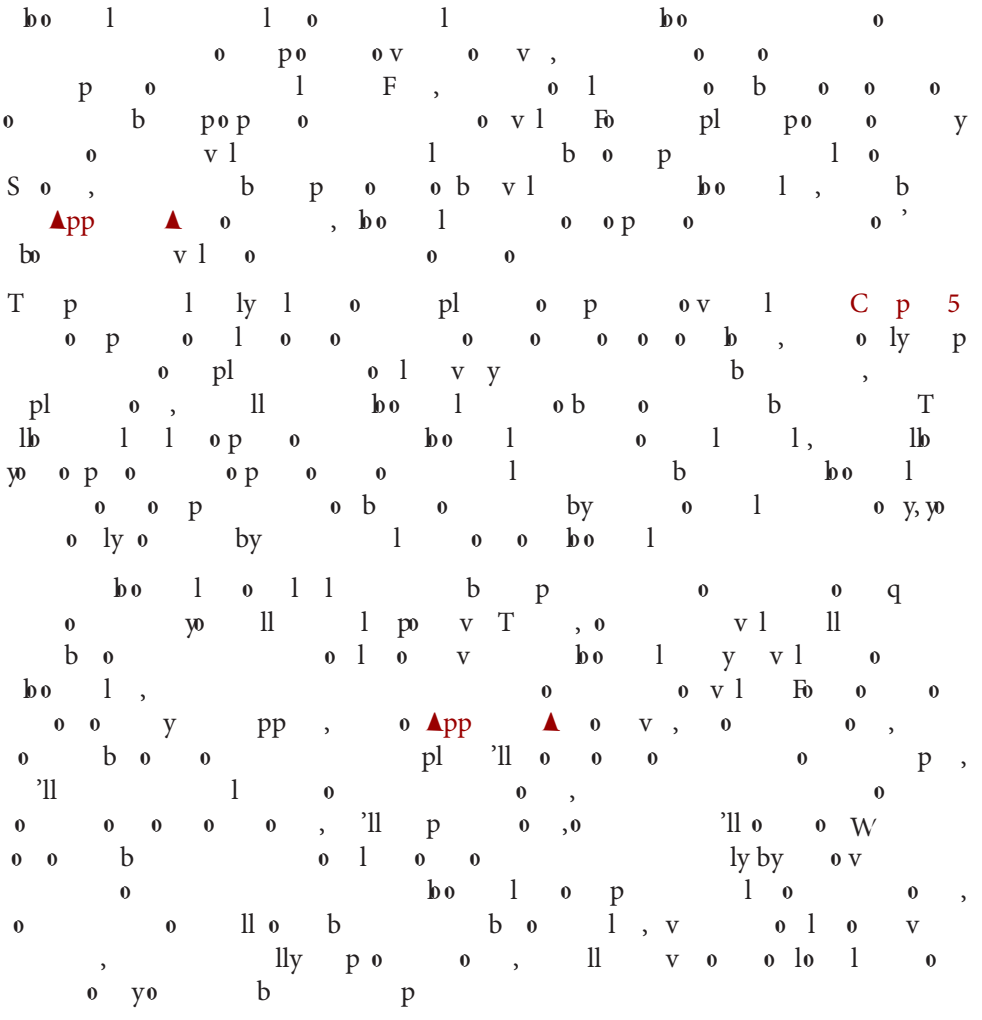
Bloom filtering o p vo p ,b q v l o o ppl o o

Intent

F l p o b o o p o v l o p o bl o p b , b pl o o T p l o v l ll b ll o hot values

Fo o , o o b o o v l p by bo l , p ;o o o o v

Motivation



Applicability

- D
- ▲
- T
- So

T o po p o o l l W p ,
b bo l o b T , p o ,
o o bo l o b p o v l
l o o o o b o bo l b p

T bo l o b - o ly T o , p
bo l l y o ,o ly p o b p
yp lly pp o p

Consequences

T o p o o b ll b b o o p bo l
b p o o l p o o y o lly b
o o v l , b bo l v o l po v

Known uses

Removing most of the nonwatched values

T o o l o v l ’ o Fo pl ,
yo y b o ly o o l o 1 , o
l oo p, p, p o , o l ,
bo l o , o o yo bo
l o y o o yo o , o o , o o ’ o
y T yo ’ll o l po v ’ b o l,
yo ll o o o o

Prefiltering a data set for an expensive set membership check

So , o v l b o o o b
p v Fo pl , yo v o b v o l b
o v l T o y b
b , b y o o p p l o o
o p l p o lly o yo l , yo v o
y p o bo l p yo v bo l
pl l o o o , yo o o p o o
o o o bl o v o bo l
bl o o v o v 95%o , yo ’ll l o o ly 5%
b o W pp o , yo ’ll v lly v 1 % y b
’ v o l o o q

, C p 5, ’ll p ll S Jo bo F l
bo l o o o o o y
o ll b l v o , o
ly

Resemblances

Performance analysis

Bloom Filtering Examples

Hot list

Bloom filter training.

```
public class BloomFilterDriver {
    public static void main(String[] args) throws Exception {
        // Parse command line arguments
        Path inputFile = new Path(args[0]);
        int numMembers = Integer.parseInt(args[1]);
        float falsePosRate = Float.parseFloat(args[2]);
        Path bfFile = new Path(args[3]);

        // Calculate our vector size and optimal K value based on approximations
        int vectorSize = getOptimalBloomFilterSize(numMembers, falsePosRate);
        int nbHash = getOptimalK(numMembers, vectorSize);

        // Create new Bloom filter
```

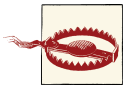
```
            BloomFilter filter = new BloomFilter(vectorSize, nbHash,
                    Hash.MURMUR_HASH);

            System.out.println("Training Bloom filter of size " + vectorSize
                    + " with " + nbHash + " hash functions, " + numMembers
                    + " approximate number of records, and " + falsePosRate
                    + " false positive rate");

            // Open file for read
            String line = null;
            int numElements = 0;
            FileSystem fs = FileSystem.get(new Configuration());

            for (FileStatus status : fs.listStatus(inputFile)) {
                BufferedReader rdr = new BufferedReader(new InputStreamReader(
                        new GZIPInputStream(fs.open(status.getPath()))));

                System.out.println("Reading " + status.getPath());
                while ((line = rdr.readLine()) != null) {
                    filter.add(new Key(line.getBytes()));
                    ++numElements;
                }

                rdr.close();
            }

            System.out.println("Trained Bloom filter with " + numElements
                + " entries.");

            System.out.println("Serializing Bloom filter to HDFS at " + bfFile);

            FSDataOutputStream strm = fs.create(bfFile);
            filter.write(strm);
            strm.flush();
            strm.close();

            System.exit(0);
        }
    }
```

▲ BloomFilter o b o o p l v o o p l b o o k b o p p l o listStatus l -by-l , l o bo l ▲ ll p l y, bo l l o l p o v o l BloomFilter l o Writable o b , l ly v l S ply FileSystem o b o FSDataOutputStream, p o l ’ write o , l b !

T bo l l b l o DFS ly
J o p p l FileSystem o b p o BloomFilter.read
Fields D l o o bo l o setup o o
o lb M pp o

Mapper code. T setup o ll o o pp by oo p
o p o o y ll o map , bo l l o
DistributedCache b o b p o T DistributedCache
oo p l y l DFS p o b l l y o
q l T bo l p vo ly o l o
o

p o , o o p o T o
o o o , o l o y o T l
o bo l o b , o
o p o l y



▲ bo l o by o o T po
o o T y bo , b
by l v y yo l o ,
b o llb
l

```
public static class BloomFilteringMapper extends
    Mapper<Object, Text, Text, NullWritable> {

  private BloomFilter filter = new BloomFilter();

  protected void setup(Context context) throws IOException,
      InterruptedException {
    // Get file from the DistributedCache
    URI[] files = DistributedCache.getCacheFiles(context
        .getConfiguration());
    System.out.println("Reading Bloom filter from: "
        + files[0].getPath());

    // Open local file for read.
    DataInputStream strm = new DataInputStream(new FileInputStream(
        files[0].getPath()));

    // Read into our Bloom filter.
    filter.readFields(strm);
    strm.close();
  }

  public void map(Object key, Text value, Context context)
      throws IOException, InterruptedException {
```

```
        Map<String, String> parsed = transformXmlToMap(value.toString());

        // Get the value for the comment
        String comment = parsed.get("Text");
        StringTokenizer tokenizer = new StringTokenizer(comment);
        // For each word in the comment
        while (tokenizer.hasMoreTokens()) {
          // If the word is in the filter, output the record and break
          String word = tokenizer.nextToken();
          if (filter.membershipTest(new Key(word.getBytes()))) {
            context.write(value, NullWritable.get());
            break;
          }
        }
      }
    }
```

p-o ly o b, o o b o ▲llo p o ll b ly o l y

HBase Query using a Bloom filter

lo o l p v o p o by l yo Fo o llb pl , lo l p vo ly D o ll v p o o l 1,5 W lo l o o l b o q y o v o o o bo y l y q , p p p o

T o llb p o o o o pl o l o o p o bl

o bl v l o ’ o , l o o o p o o l 1,5

Mapper Code. T `setup` o ll o o pp by oo p o p o o y ll o `map` o J l p vo pl , lo l l o `DistributedCache` b o b `map` o T lo l ll D v p o o l 1,5 T l l o v 1 5%o ll , o llb l o lo o y q o o lo l , o o o bl o b `setup`

p o , ’ D o o lo l po v , q D o o , ll y po b l o o p l po v by v y ’ l p o l 1,5 , o o p o l y

```
public static class BloomFilteringMapper extends
    Mapper<Object, Text, Text, NullWritable> {
```

```
private BloomFilter filter = new BloomFilter();
private HTable table = null;

protected void setup(Context context) throws IOException,
    InterruptedException {

  // Get file from the Distributed Cache
  URI[] files = DistributedCache.getCacheFiles(context
        .getConfiguration());
  System.out.println("Reading Bloom filter from: "
      + files[0].getPath());

  // Open local file for read.
  DataInputStream strm = new DataInputStream(new FileInputStream(
      files[0].getPath()));

  // Read into our Bloom filter.
  filter.readFields(strm);
  strm.close();

  // Get HBase table of user info
  Configuration hconf = HBaseConfiguration.create();
  table = new HTable(hconf, "user_table");
}

public void map(Object key, Text value, Context context)
    throws IOException, Interrupted           IOExce   h                    
```

Query Buffer Optimization

T p v o pl ly v yo q y
o o o o o bo p , b b o p
ppo b q , o o l b l o b ll
q o p o o p T
o p o o y o yo o o bly o
o y b o q y T l q o
p o p o l p
v o p o b b , o o o o J
b o l b pp o ’ cleanup
o T Contexto b b o o p l
mapo reduce o

Top Ten

Pattern Description

T *top ten* p b p v o o yo o o y
o yo o , o p l ,
o v , o o o p p o

Intent

v l v ly ll b o o p *K* o , o o
yo , o o l

Motivation

F o l po p o ly b o yp lly
o q p o T po o p o
b o o p o o yo bo
p p o o b o p l yo o
o o p o o b o o o
o , yo pply p o M p o o
v l o yo

T o y p p l ly p o o p o
o yo pl o p p o o M p o SQ ,
yo b l o o yo by v l , o p *K*
o o M p , ’ll o p , o lo
ly wo lv o o yo l T p ll
o bo l b o -v l o o v o o

l , o p o o l y W o p o S v b W o o l b o y o v W l l o o y o b W p o o o o b o

Applicability

- T p q o p o o b l y b o o T , b bl o o p o o o o o l
- T b o o p o o l b ly b o p o b p o o o o lo o

Structure

T p l bo pp T pp ll b l o p *K*, llo v l o p *K* ll o p o l o p *K* S b o o o o o pp o *K* *K* l v ly ll, ’llo ly o o o p F 3-3

```
class mapper:
   setup():
      initialize top ten sorted list

   map(key, record):
      insert record into top ten sorted list
      if length of array is greater-than 10 then
         truncate list to a length of 10

   cleanup():
      for record in top sorted ten list:
         emit null,record

class reducer:
   setup():
      initialize top ten sorted list

   reduce(key, records):
      sort records
      truncate records to top 10
      for record in records:
         emit record
```

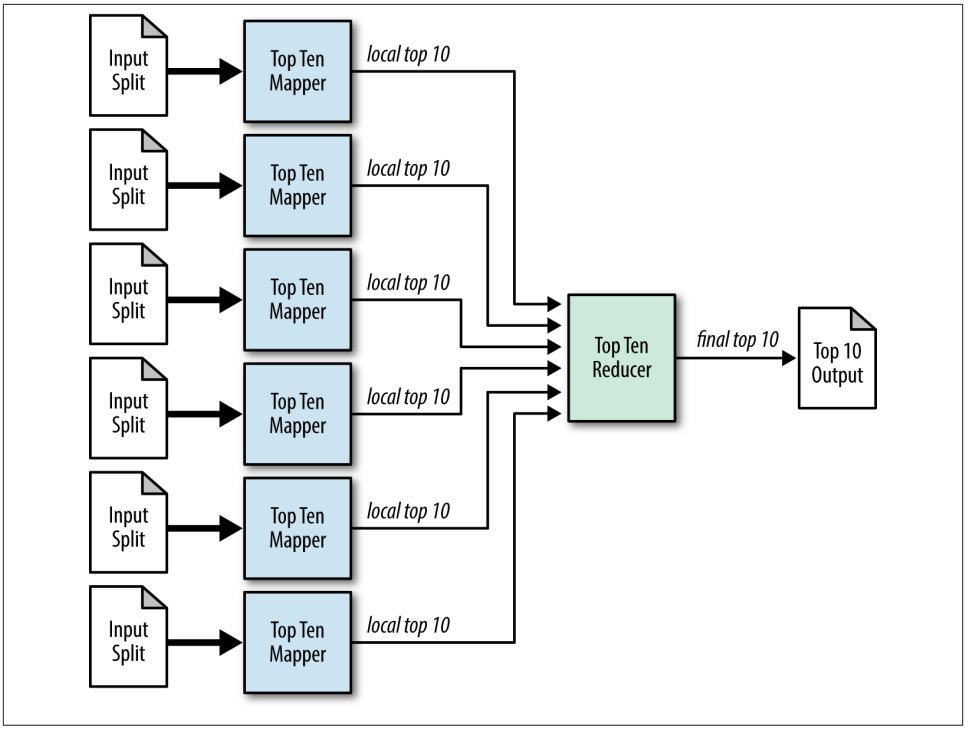


Figure 3-3. The structure of the top ten pattern

T pp o p yo b o K o ll l K v l l p p o pp , b o , ’ll lly K o o y v l , ll y T b K o p l p

W o l p K * M o o o o y, ll, M b o p o , ’ll o pp p yo K v l o p K o o v l o ll ll y

T o o l o p K o v y pp b o v bl ll o o p o o o l pl o o b o o

Consequences

T o p K o

Known uses

Outlier analysis

l lly T y y b v l y
yo y , o po o yo b l , l l
o p , y v yo o p p v o yo

Select interesting data

yo bl o o yo o by o o o v l o , yo p ll
o v l bl T p l ly l yo pl o b o o lo -
o p o , b ll oo lo SQ b , o
l l o yo o l V l o b o pl yo
by pply v l o , o b o o
l o p ll o yo o v o o

Catchy dashboards

T ’ p y o lo y boo o , o o o y o p l o
o , b y T p o l b o p bl o o p
bo yo b yo ll o o o
bo yo o v o ll o o p o

Resemblances

SQL

o l ll SQ b ,o y o b b l ,
yo o l v o by o o yo o p ,
l o o l o lo pp o M p , b yo
ll o l p , o p v o p o

```
SELECT * FROM table ORDER BY col4 DESC LIMIT 10;
```

Pig

ll v p o q y y o o o p l y T o
o p o o SQ q y, b o p v
o o T o yo ’ll o
J v M p o

```
B = ORDER A BY col4 DESC;
C = LIMIT B 10;
```

Performance analysis

T p o o o p p yp lly v y oo , b b
o po l o o o o Mo o l o p o
o l , l o b o o l

T b o p y o o p o y o p o o o p K o , o b ll o o M p , o o o v o o o K*M o T b b

▲ l b o b o o

- T o b o p v o p o oo y o o o o o o b l , o o y
- T o ll v b o o v o , y o o o p o l o
- N lly, o ll ll b y o o b o o
- ▲ y o o o y o p o b l y o bb o J v v l ’ o y Fo pl , yo o ll llo v l o ▲ y o p o , ▲ y v y b T ll o b p l p o bl yo ’ lly b o o o p , b yo o v y l b yo y o o y l
- W o o p l o p ll l W o b lly b o o o p v o p o p l b o S o ly o , o v o p ll l wo lv o v l o , o v v l o o ▲ , o o o p , b b o o v y l

▲ K l , p b o l Co K v llo , llo o F v llo b o o y v l p pl , o v y pp ll llo o o T l ll v ly v o l llo o o ly p ll l b

▲ o p o yo o l yo v l K l b o p pl o p l o o , b yo o o p l ’ yo v l o ly , o b p yo o o p o , yo p vo M p o b, o v l o 5 ,485, yo o yo l o ll o v v l o l 5 ,485 T o y o v l l 5 ,845 o p p vo o p ll

Fo ll o , p o ly o p y ll v l o *K*,
o o , o yo l ly p b T y
l o o lo o l ly o v

Top Ten Examples

Top ten users by reputation

D o p o o o M p
pp o p o o p pl o p o
p T pp lly l p pl o o p o ,
po bl o l J b o o yo o b o o ly
o ! M l pl o l o l l l pl
o p l

T o lb p o o o o pl o l o o p o bl

o bl v l o o o ,o p o o o o p
b o p o

Mapper code. T pp p o ll p o o TreeMap ▲
TreeMap b l o Map o o y T l o o Integers
T , o o o TreeMap, l
b v l b o v ▲ ll o v b p o , o p
o TreeMap o p o cleanup o T o
ll o ll y/v l p v b o map, l o setup
ll o b o y ll o map

```
public static class TopTenMapper extends
    Mapper<Object, Text, NullWritable, Text> {

  // Stores a map of user reputation to the record
  private TreeMap<Integer, Text> repToRecordMap = new TreeMap<Integer, Text>();

  public void map(Object key, Text value, Context context)
      throws IOException, InterruptedException {
    Map<String, String> parsed = transformXmlToMap(value.toString());

    String userId = parsed.get("Id");
    String reputation = parsed.get("Reputation");

    // Add this record to our map with the reputation as the key
    repToRecordMap.put(Integer.parseInt(reputation), new Text(value));

    // If we have more than ten records, remove the one with the lowest rep
    // As this tree map is sorted in descending order, the user with
    // the lowest reputation is the last key.
    if (repToRecordMap.size() > 10) {
```

```
        repToRecordMap.remove(repToRecordMap.firstKey());
      }
    }

    protected void cleanup(Context context) throws IOException,
        InterruptedException {
      // Output our ten records to the reducers with a null key
      for (Text t : repToRecordMap.values()) {
        context.write(NullWritable.get(), t);
      }
    }
  }
```

Reducer code. v ll, o p o y ' v y l o pp o o o b o v o job.setNumReduceTasks(1) NullWritable o y, ll b o p o p o o ll p o l o p o T o ll o o TreeMap TreeMap' bo v , l b v l o v o p ▲ ll v l v b o v , v l o TreeMap l o l y o T o v by p o TreeMap p o o o p v l T b o ly reduce o , b ll b o ly o p o p, b o cleanup o o l l o o

```
  public static class TopTenReducer extends
      Reducer<NullWritable, Text, NullWritable, Text> {

    // Stores a map of user reputation to the record
    // Overloads the comparator to order the reputations in descending order
    private TreeMap<Integer, Text> repToRecordMap = new TreeMap<Integer, Text>();

    public void reduce(NullWritable key, Iterable<Text> values,
        Context context) throws IOException, InterruptedException {
      for (Text value : values) {
        Map<String, String> parsed = transformXmlToMap(value.toString());

        repToRecordMap.put(Integer.parseInt(parsed.get("Reputation")),
            new Text(value));

        // If we have more than ten records, remove the one with the lowest rep
        // As this tree map is sorted in descending order, the user with
        // the lowest reputation is the last key.
        if (repToRecordMap.size() > 10) {
          repToRecordMap.remove(repToRecordMap.firstKey());
        }
      }

      for (Text t : repToRecordMap.descendingMap().values()) {
        // Output our ten records to the file system with a null key
```

```
        context.write(NullWritable.get(), t);
      }
    }
  }
```

T o o o b o b, l o o
o l lly b T o b o l plyo p
o y p o ▲l o , o b
o o o p o , b o l ly b o o
o p *K* o v bl p `setup` o
J b o p l o o
▲ ly o *K*

Distinct

Pattern Description

T p l o l , b ’ o ll b yo o l o
o bo l o o T lo p o l p
pl o o q o

Intent

o v o l o yo o q o v l

Motivation

o q o v l v l p l
p pl o o l , pl o ly
l o b o p o bl T pl o p
o o p o o p-l v l ly l Fo pl , v y
o o v yo b , yo o ll b b o v y
o ly v yo b o o , yo ’ll b
o o o o yo o o ly o l l p o yo
p b b o , b o v yo
b ll l T o , yo o l pl o
yo v o lyo o bo v v

o o ’ ly o b ly o T y
o b bl o b l o o y ll b ly Fo pl ,
o b b o ly o o TT v bo , o ly ,
v , b o W o ’ bo p,
o y ,o TT v o

Applicability

T o ly o q yo v pl v l yo T
o q , b o l b lly o p o !

Structure

T p p y l o M p pb M p ’ b l y o
o p y o o o v pl T p pp o o
o ’ o T o b l y b l
p lp o bly l b o pl D pl
o o b b o o , o o b ll pl
p p

```
map(key, record):
   emit record,null

reduce(key, records):
   emit key
```

T pp o l o q
v l o TT b pl , , b b o ,
v v l T pp o p o y, ll v l

T o p ll o by y, o ’ll v o ll p yW
plyo p y, o ’ o y ll v y
o p o , o p o b q

o p b o o ’
o l l o l S b o l v ly , pp
ll o l o ll o

T o o o yo l yo yo o p
l o b l , b o yo
ll , b o T l ll o o o b
bo o o p o

Consequences

T o p o o b q , b yo o b p
v o o p o o o

Known uses

Deduplicate data

yo v y b o o ll o o o l
v , yo o v pl p

Getting distinct values

T l yo o y o b pl , b
o o pl o o

Protecting from an inner join explosion

yo bo o o o b o yo o y
o q , yo v b o o Fo pl , yo v
3, o y o , , o y o
, yo ’ll p 6, , o , ll o o ! y
p , yo p o yo v l o y q
p o bl

Resemblances

SQL

SELECT DISTINCT p o o p o o SQ

```
SELECT DISTINCT * FROM table;
```

Pig

T DISTINCTo p o

```
b = DISTINCT a;
```

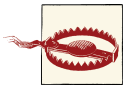
Performance analysis

p ’ p o p o l po o v T
o o o o p M p o b b o
yo yo ll T b o ly p o
o l b o o by o o o pp , p o
o o b bl o l lly, pl v y
p pl o b l o o , p y llo
o o b o p

o b o o p by o by lo o Jo bT
o o bo pl T b o o p by v by b
o yo bo T bo o y by
ll , o o o T b l v o
plo y o plo y , b lly yo o l ’ p o
by o l o o ’ o p oo o , b yo o p l
ll b y ll b yo v p p ▲
o o v o blo o o , yo blo
64M , v l 64M o

S o o o o b o , yo ll
l v ly l b o o o b ▲ y o o p

pp , o o p o pp , ll o b o S
o l b o o p o , b o o l o
po l, p , yo yo o l
, o bl b o J b lo l oo ll



o o o o y lo yo l l
b o o yo o b ▲ oo o p
o l b lo o b o lo o o bly
o b o lo o v y l

Distinct Examples

Distinct user IDs

F o v l pl o M p ’ po
p q y o v l o y,
o o p o v l , ply o o y o v y
o o

T o llo p o o o o pl o l o o p o bl
o bl v l o ’ o , o D

Mapper code. T M pp ll D o p o T D ll
b o p y ll v l

```
public static class DistinctUserMapper extends
      Mapper<Object, Text, Text, NullWritable> {

  private Text outUserId = new Text();

  public void map(Object key, Text value, Context context)
      throws IOException, InterruptedException {

    Map<String, String> parsed = transformXmlToMap(value.toString());

    // Get the value for the UserId attribute
    String userId = parsed.get("UserId");

    // Set our output key to the user's id
    outUserId.set(userId);

    // Write the user's id with a null value
    context.write(outUserId, NullWritable.get());
  }
}
```

Reducer code. T o o b l o D l by
M p o v q y o ll v l
T v l o p y o l y ll v l

```
public static class DistinctUserReducer extends
      Reducer<Text, NullWritable, Text, NullWritable> {

  public void reduce(Text key, Iterable<NullWritable> values,
      Context context) throws IOException, InterruptedException {

    // Write the user's id with a null value
    context.write(key, NullWritable.get());
  }
}
```

Combiner optimization. ▲ o b o l b p D
pl y ll b o v o b l p' o p , o o
o / q T o o b o b

CHAPTER 4
Data Organization Patterns

o o p v o p o l , p ll b o o
T v l o v l o o l pl by y y p o ,
, o o T p lly b y , p o ,
, o b pb o p o

y o o , o o p o M p o l o o ly p
l ly pl o D ll yp lly v o b o o o
ly o y , v o b o
o o l o o ly M p

T p o v l p b o y o ll p
p o

- T *structured to hierarchical* p
- T *partitioning* *binning* p
- T *total order sorting* *shuffling* p
- T *generating data* p

T p p o o o o lv o o p o bl
F o pl , y o y o y o o b l, b ,
v b b o S J o b C p 139 C p 6 o o l
o o o l p o bl o o b p o o o lv o o pl
p o bl

Structured to Hierarchical

Pattern Description

T *structured to hierarchical* p o o
v y o po , p y y
lo p

Intent

T o yo o -b o l o , JS No XM

Motivation

W o D MS o oo p y ,o o yo
o l o o o yo o o o v S
oo p o ’ o yo , yo o l v o
l o wo o o

Fo pl ,o S v lo o bl bo o , bl bo
po , p yo bvo o o l SQ b
W yo v po o S v lo , ll p o b o l
o o v T v o o pl yo y o o ly
l v lo v l po y o o l l o po
l o o T q yo o o o , p v o p o ,
llo yo o o yo l o yo o p
by po o o o lo po v o ,
o l bl , yp o ly ll b o v
p o l o v l l p po

o ly, o ’ l y o o p o W o o po
o S v lo q o , oo p ’ o o y
ly T o , o l o o M p o b
o b o p o lly

▲ o y o l y o p bl o o
- l o ll Mo o D o l l o b oo
o o yp o

Applicability

T o llo o l b o p o b pp o p

- o v o l by o o o y
- o o -b

Structure

F 4-1 o o p T p o o o po o llo

- yo o o b l pl o o l , oo p l ll `MultipleInputs` o `org.apache.hadoop.mapreduce.lib.input` ly v l bl `MultipleInputs` llo yo o p y p p pp l o p T o o o v yo lo o o lyo o p , yo o ' p
- T pp lo p o oo o v o o yo o T o p y o l l o yo o y oo o l o Fo pl , o S v lo pl , oo o l b po D o lo o v p o o o bo o o o y o p o po o o To o , yo ply o o o o l b l o o p v l
- l, o b ' o o lp yo oo o o l ypo lly o p y o v o , b o o o p o ll yo o l b o o , o o l o l b p
- T v o ll o y by y ▲llo o p l o p o o b po v o yo o o , o ll l o yo o o b l l o l o W XM o JS N, yo 'll b l l o b o o p T pl o o XM , po v v l o v o o o yo o o o , o o , yo 'll o po p o b b l l o o

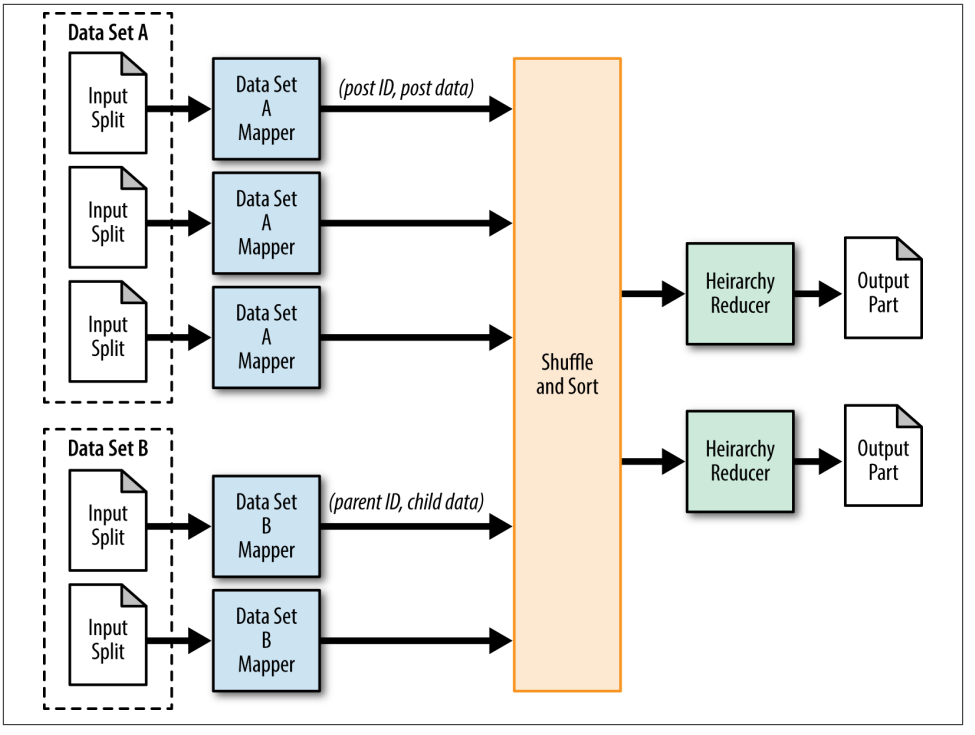


Figure 4-1. The structure of the structured to hierarchical pattern

Consequences

T o p ll b l o , o p by y yo p

o v , b l y o XM JS N v o o o o p-
l v l oo l o p llo o yo lly o
o b ll- o o p- o -bo o , ’ lly o oo
o po -p o p

Known uses

Pre-joining data

D v o , o ly l p po o l
b o b o o o o pl o b y o , yo
p yo o v o No SQ o lo ly

Preparing data for HBase or MongoDB

l y o o , o yo o o b
o p p o o lo o o Mo o D C
bl b l po v M p p l ly v T
l v o o v l o o ,

▲ o p o bl o p o l
T o o b l p o bl bo y M p o b y
b o , b lly yo o p o o
pl p o v ly

Structured to Hierarchical Examples

Post/comment building on StackOverflow

pl , ll po o o S v b
o p o ▲ y ll bo o l

```
Posts
    Post
      Comment
      Comment
    Post
      Comment
      Comment
      Comment
```

T o lb p o o o o pl o l o o p o bl
o bl v l o po o , XM y o
o l po

Driver code. W o ' lly b o o v , b
o o o MultipleInputs ▲ll o ly
MultipleInputs o b o p po p
p v pp T p o po o p o v v
o l , p o v o args y

```
public static void main(String[] args) throws Exception {
    Configuration conf = new Configuration();
    Job job = new Job(conf, "PostCommentHierarchy");
    job.setJarByClass(PostCommentBuildingDriver.class);

    MultipleInputs.addInputPath(job, new Path(args[0]),
            TextInputFormat.class, PostMapper.class);

    MultipleInputs.addInputPath(job, new Path(args[1]),
            TextInputFormat.class, CommentMapper.class);

    job.setReducerClass(UserJoinReducer.class);

    job.setOutputFormatClass(TextOutputFormat.class);
    TextOutputFormat.setOutputPath(job, new Path(args[2]));

    job.setOutputKeyClass(Text.class);
```

```
        job.setOutputValueClass(Text.class);

        System.exit(job.waitForCompletion(true) ? 0 : 2);
    }
```

Mapper code. , o pp l ,o o o o o
po bo , po D o o p yW o p p
v l p p o po o C o o o o
o o p

```
    public static class PostMapper extends Mapper<Object, Text, Text, Text> {

        private Text outkey = new Text();
        private Text outvalue = new Text();

        public void map(Object key, Text value, Context context)
                throws IOException, InterruptedException {

            Map<String, String> parsed = MRDPUtils.transformXmlToMap(value
                    .toString());

            // The foreign join key is the post ID
            outkey.set(parsed.get("Id"));

            // Flag this record for the reducer and then output
            outvalue.set("P" + value.toString());
            context.write(outkey, outvalue);
        }
    }

    public static class CommentMapper extends Mapper<Object, Text, Text, Text> {
        private Text outkey = new Text();
        private Text outvalue = new Text();

        public void map(Object key, Text value, Context context)
                throws IOException, InterruptedException {

            Map<String, String> parsed = MRDPUtils.transformXmlToMap(value
                    .toString());

            // The foreign join key is the post ID
            outkey.set(parsed.get("PostId"));

            // Flag this record for the reducer and then output
            outvalue.set("C" + value.toString());
            context.write(outkey, outvalue);
        }
    }
```

Reducer code. T b l l XM o b ▲ll v l
o p o o ll l o o W o o
by l o v l T l o v post o
l T , p o ll, XM o o p o
p o l

T pl o o nestElements o l b W o o XM l b y o
b l l o , b pl l o v y o y

```
public static class PostCommentHierarchyReducer extends
        Reducer<Text, Text, Text, NullWritable> {

    private ArrayList<String> comments = new ArrayList<String>();
    private DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();
    private String post = null;

    public void reduce(Text key, Iterable<Text> values, Context context)
            throws IOException, InterruptedException {
        // Reset variables
        post = null;
        comments.clear();

        // For each input value
        for (Text t : values) {
            // If this is the post record, store it, minus the flag
            if (t.charAt(0) == 'P') {
                post = t.toString().substring(1, t.toString().length())
                        .trim();
            } else {
                // Else, it is a comment record. Add it to the list, minus
                // the flag
                comments.add(t.toString()
                        .substring(1, t.toString().length()).trim());
            }
        }
        // If there are no comments, the comments list will simply be empty.

        // If post is not null, combine post with its comments.
        if (post != null) {
            // nest the comments underneath the post element
            String postWithCommentChildren = nestElements(post, comments);

            // write out the XML
            context.write(new Text(postWithCommentChildren),
                    NullWritable.get());
        }
    }
    ...
```

T nestElements o po l o o o
o XM o o p DocumentBuilder o o l lp o o
o py Element o b o o , o o b T o py
o o l o row o post o comment T l
Document o o XM

```
    private String nestElements(String post, List<String> comments) {
        // Create the new document to build the XML
        DocumentBuilder bldr = dbf.newDocumentBuilder();
        Document doc = bldr.newDocument();

        // Copy parent node to document
        Element postEl = getXmlElementFromString(post);
        Element toAddPostEl = doc.createElement("post");

        // Copy the attributes of the original post element to the new one
        copyAttributesToElement(postEl.getAttributes(), toAddPostEl);

        // For each comment, copy it to the "post" node
        for (String commentXml : comments) {
            Element commentEl = getXmlElementFromString(commentXml);
            Element toAddCommentEl = doc.createElement("comments");

            // Copy the attributes of the original comment element to
            // the new one
            copyAttributesToElement(commentEl.getAttributes(),
                    toAddCommentEl);

            // Add the copied comment to the post element
            toAddPostEl.appendChild(toAddCommentEl);
        }

        // Add the post element to the document
        doc.appendChild(toAddPostEl);

        // Transform the document into a String of XML and return
        return transformDocumentToString(doc);
    }

    private Element getXmlElementFromString(String xml) {
        // Create a new document builder
        DocumentBuilder bldr = dbf.newDocumentBuilder();

        return bldr.parse(new InputSource(new StringReader(xml)))
                .getDocumentElement();
    }

    private void copyAttributesToElement(NamedNodeMap attributes,
            Element element) {

        // For each attribute, copy it to the element
        for (int i = 0; i < attributes.getLength(); ++i) {
```

```
            Attr toCopy = (Attr) attributes.item(i);
            element.setAttribute(toCopy.getName(), toCopy.getValue());
        }
    }

    private String transformDocumentToString(Document doc) {

        TransformerFactory tf = TransformerFactory.newInstance();
        Transformer transformer = tf.newTransformer();
        transformer.setOutputProperty(OutputKeys.OMIT_XML_DECLARATION,
                "yes");
        StringWriter writer = new StringWriter();
        transformer.transform(new DOMSource(doc), new StreamResult(
                writer));
        // Replace all new line characters with an empty string to have
        // one record per line.
        return writer.getBuffer().toString().replaceAll("\n|\r", "");
    }
}
```

Question/answer building on StackOverflow

T o o o p vo pl ll p vo ly ’o p p o ly lVo v o o po , o o o po po q o T o b o b po o o bo q o o ly by PostTypeIdW ’ll o p o by Id q o ParentId

T b o ppl o o p o l o ly o v ly, l - o o o l o o

T o lb p o o o o pl o l o o p o bl o bl v o p o p vo pl , p o l - o o p o o q o , , o y

Mapper code. T pp o o o q o o , b b vo o ll b Fo q o , ll Id y l b l q o Fo , ll ParentId y l b l

```
public class QuestionAnswerBuildingDriver {

    public static class PostCommentMapper extends
            Mapper<Object, Text, Text, Text> {

        private DocumentBuilderFactory dbf = DocumentBuilderFactory
                .newInstance();
        private Text outkey = new Text();
```

```
    private Text outvalue = new Text();

    public void map(Object key, Text value, Context context)
            throws IOException, InterruptedException {

        // Parse the post/comment XML hierarchy into an Element
        Element post = getXmlElementFromString(value.toString());

        int postType = Integer.parseInt(post.getAttribute("PostTypeId"));

        // If postType is 1, it is a question
        if (postType == 1) {
            outkey.set(post.getAttribute("Id"));
            outvalue.set("Q" + value.toString());
        } else {
            // Else, it is an answer
            outkey.set(post.getAttribute("ParentId"));
            outvalue.set("A" + value.toString());
        }

        context.write(outkey, outvalue);
    }

    private Element getXmlElementFromString(String xml) {
        // same as previous example, “Mapper code” p   8
    }
}
```

Reducer code. T o v y l o p v o pl o p v l b q o , b o o v l q o o p v o pl T q o o p o T lp o o o b v y T y b v p v o pl

```
public static class QuestionAnswerReducer extends
        Reducer<Text, Text, Text, NullWritable> {

    private ArrayList<String> answers = new ArrayList<String>();
    private DocumentBuilderFactory dbf = DocumentBuilderFactory
            .newInstance();
    private String question = null;

    public void reduce(Text key, Iterable<Text> values, Context context)
            throws IOException, InterruptedException {
        // Reset variables
        question = null;
        answers.clear();

        // For each input value
        for (Text t : values) {
```

```
            // If this is the post record, store it, minus the flag
            if (t.charAt(0) == 'Q') {
                question = t.toString().substring(1, t.toString().length())
                        .trim();
            } else {
                // Else, it is a comment record. Add it to the list, minus
                // the flag
                answers.add(t.toString()
                        .substring(1, t.toString().length()).trim());
            }
        }

        // If post is not null
        if (question != null) {
            // nest the comments underneath the post element
            String postWithCommentChildren = nestElements(question, answers);

            // write out the XML
            context.write(new Text(postWithCommentChildren),
                    NullWritable.get());
        }
    }

    ... // ommitted helper functions
}
```

Partitioning

Pattern Description

T *partitioning* p o v o o o , , p o , o b b o ’ lly bo o o o

Intent

T o l o p o o , ll

Motivation

yo o bo p l o — po o p l — o lly p o o So bo o o b q o llo o b l o o ll b , b o by o o l v o yo ly To p o v p o , yo o b b p o o o p l T , p l b o o b ly , o b o ly o bo

o by o o o o T lp o
ly p o , b l y o p by o Fo
, ppo yo v v p y yo oo p l ,
b o v o o oo ll by yo o ly bo
o J y 7 o F b y 3 o y , yo ll o
o v o l b y yo v
p o o o , yo v l J y , l F b y ,
, yo o l o ly o yo M p o bo v J y F b y
p o o l b v b y p o by y!

o l o lp o yo v v l yp o o
, ly o o N o SQ Fo pl , TT v
b , yo ’ll v `GET` `POST` q , l y , o
▲ ly y bo o ly o o y o , o p o o
o ll lp o o o b o v b o v

D MS, yp l o o p o yo o lly l by
`WHERE` l So , o pl , yo yp lly l o o by o y,
p p yo o l p o by o y T ppl M p ll yo
yo l l o b o o pp o o v
o v , yo o l o p o yo

T o o o p o o v o b l p o ▲M p
o b ll o v ll p o o y

Applicability

T o o q o pply p o o y p o yo
o o v o Fo pl , yo o yo o o p o by
yo , yo o yo ll v v p o

o o q by ly b
o p o Fo pl , yo v b o p , b yo o ’ o
o b p , o b o o yo

Structure

T p pb p o p o
! T o l p o b ; ll yo v o o
o p o o o o o o o p o
F 4- o o p

- o , y pp b
- T o p o o p T o p o ll
o o o ; o po o p
l p o

- o , y b p o o l p o D ll o o o p o , o b pl , ,o , p p o

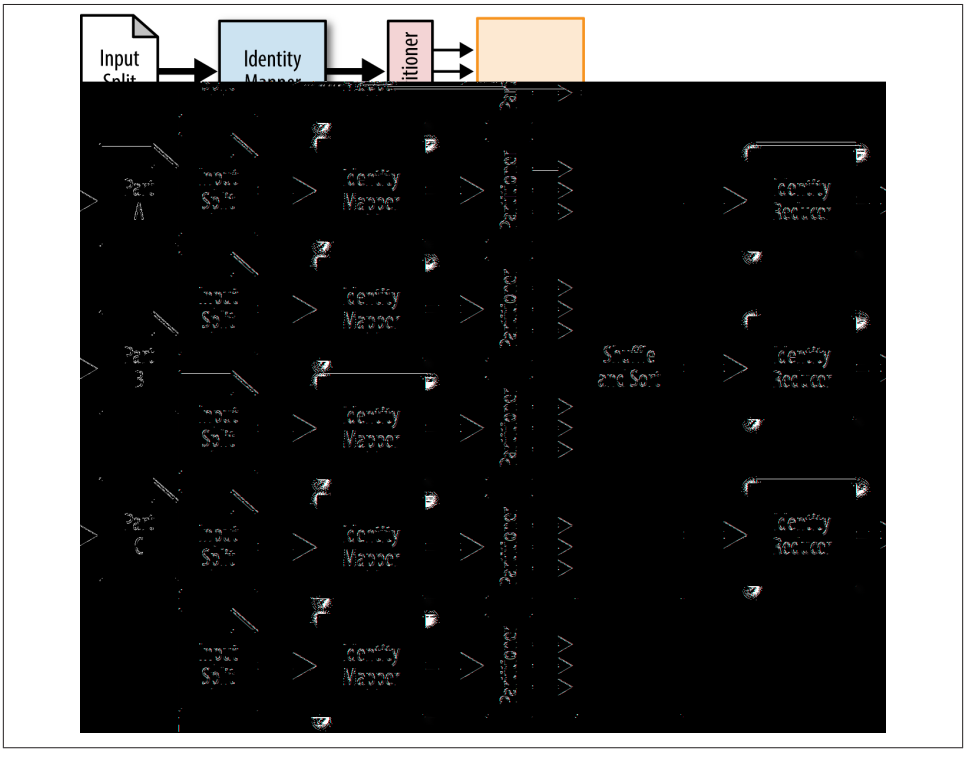


Figure 4-2. The structure of the partitioning pattern

Consequences

T o p o l o o b ll v o p l o p o

S o y ll b o o o l l , pl o o bb - o p SequenceFiles, bly o y- o - o oo p

Known uses

Partition pruning by continuous value

o v o o o o v bl , o l v l ,
yo yo bo o ly b o o
o b ll llo yo o b o lo o ly p

Partition pruning by category

o v o o o o v bl , o o o o v l
l ly o , o y, p o o ,o l

Sharding

▲ y yo v o o — —
yo o p o o

Resemblances

SQL

So SQ b llo o o lly p o bl T llo p
o p llo b o l l po o o l v
b o SQ

Other patterns

T p l o b p p o , b
p o p o b vo p

Performance analysis

T p o o p l p o ll
l ly o v l b o o p o p o o o o1 5 %
o o v y l pl v ly, llo ll
o o ll lo o p o ly

’ p y yo o , o Spl v y l p o o v l ll
p o , v o ly ▲ l pl o o p o
o ly o o o p o b b

Fo pl , o l l o S v lo
p o o p o p y q llyo v o , o o ll v y l ly
b l yo o To p v , y o p o
o o o y ,o p p o ly

T o o ’ p o o v p o , yo o o
l p o l p o J l llo p

Partitioning Examples

Partitioning users by last access date

S v b , o o y , o o o p o b o y o l T o by o p o o o o p l p o b o

T o lb p o o o o pl o l o o p o bl o bl v o o o , p o o b o y o l ,o p o p y

Driver code. T v l l o T o b o b o o o b l p o , p o o b o T l y o b o , 8 T o o pl p o o o Alo , b o p o ll o p o o o v o pl 1 , l y 11, p 4 y o 8 o 11 ll o ll o b , o b o o v ly 4

```
...
// Set custom partitioner and min last access date
job.setPartitionerClass(LastAccessDatePartitioner.class);
LastAccessDatePartitioner.setMinLastAccessDate(job, 2008);

// Last access dates span between 2008-2011, or 4 years
job.setNumReduceTasks(4);
...
```

Mapper code. T pp p ll l o o p o T o p y, ll p o o p v l T o p o o o o p o o pp o p p o T y l o o p o p

```
public static class LastAccessDateMapper extends
        Mapper<Object, Text, IntWritable, Text> {

    // This object will format the creation date string into a Date object
    private final static SimpleDateFormat frmt = new SimpleDateFormat(
            "yyyy-MM-dd'T'HH:mm:ss.SSS");

    private IntWritable outkey = new IntWritable();

    protected void map(Object key, Text value, Context context)
            throws IOException, InterruptedException {
```

```
            Map<String, String> parsed = MRDPUtils.transformXmlToMap(value
                    .toString());

            // Grab the last access date
            String strDate = parsed.get("LastAccessDate");

            // Parse the string into a Calendar object
            Calendar cal = Calendar.getInstance();
            cal.setTime(frmt.parse(strDate));
            outkey.set(cal.get(Calendar.YEAR));

            // Write out the year with the input value
            context.write(outkey, value);
        }
    }
```

Partitioner code. T p o y/v l p o p by pp o p o y/v l p ll b b p o ll b o p by o p T p o pl Configurable T setConf o ll o o o o p o , v l o l p ll o o o T v po bl o ll LastAccessDatePartitioner.setMinLastAccessDate o b o o T o b o y l o p o o o T l 8, o ll o l b o S v b 8 ll b o p o o

```
    public static class LastAccessDatePartitioner extends
            Partitioner<IntWritable, Text> implements Configurable {

        private static final String MIN_LAST_ACCESS_DATE_YEAR =
                "min.last.access.date.year";

        private Configuration conf = null;
        private int minLastAccessDateYear = 0;

        public int getPartition(IntWritable key, Text value, int numPartitions) {
            return key.get() - minLastAccessDateYear;
        }

        public Configuration getConf() {
            return conf;
        }

        public void setConf(Configuration conf) {
            this.conf = conf;
            minLastAccessDateYear = conf.getInt(MIN_LAST_ACCESS_DATE_YEAR, 0);
        }
```

```
        public static void setMinLastAccessDate(Job job,
                int minLastAccessDateYear) {
            job.getConfiguration().setInt(MIN_LAST_ACCESS_DATE_YEAR,
                    minLastAccessDateYear);
        }
    }
```

Reducer code. T o v y pl ply o o p v l T o o p o b o po

```
    public static class ValueReducer extends
            Reducer<IntWritable, Text, Text, NullWritable> {

        protected void reduce(IntWritable key, Iterable<Text> values,
                Context context) throws IOException, InterruptedException {
            for (Text t : values) {
                context.write(t, NullWritable.get());
            }
        }
    }
```

Binning

Pattern Description

T *binning* p , l p vo p , o v o o o p v o o o o

Intent

Fo o , l o o o o o o

Motivation

v y l o p o o b o o lv p o bl T o o b o p o b l M p o o o ,o o l o o b o

pl p p p o p o T o v o l o p , lly l o o o lb o T o pp ll o v o l p po bl o p b T , yo v o b o pp , yo o o o p o lo o llo l T b o M M o l b l y o lb o ly T p o p ll v o o p l p o y o o v p o bl

Structure

- T p ’ v q MultipleOutputs l , p o b’ o p o l pl l
- T pp bo l , o l o o b o , o b S F 4-3
- N o o b , p o ,o p

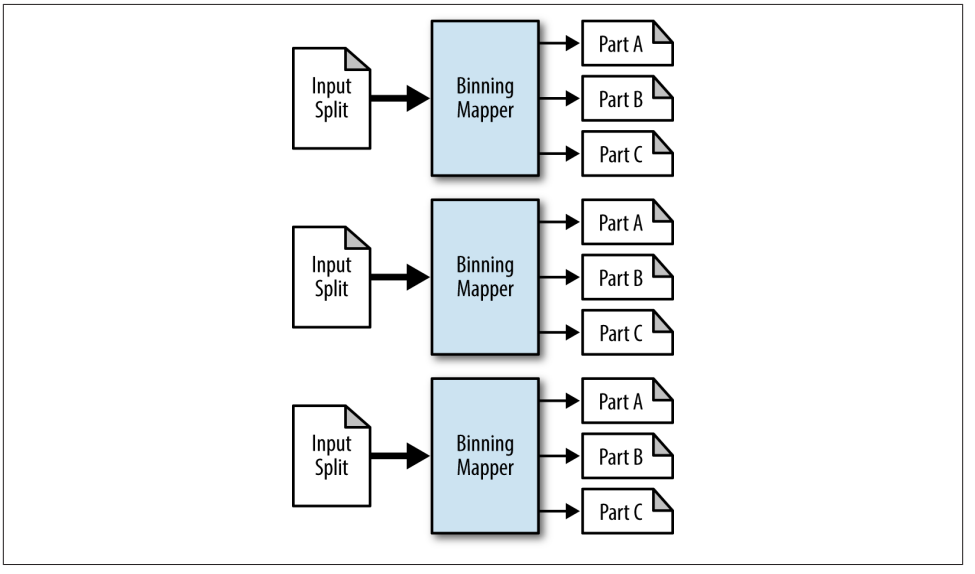
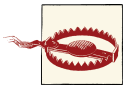


Figure 4-3. The structure of the binning pattern

Consequences

pp o p o ll l p b



D o l o b l b o y l ▲ o p o , y o o l o p o p o o ll o p o l l

Resemblances

Pig

T SPLIT o p o pl p

```
SPLIT data INTO
    eights IF col1 == 8,
    bigs IF col1 > 8,
    smalls IF (col1 < 8 AND col1 > 0);
```

Performance analysis

T p l b l y p o p o p-o ly
o b N o , l , o o b p o , o o p o
o o b o o b l

Binning Examples

Binning by Hadoop-related tags

W o l by o b o o lb -o ly
o v o o v llo W o ly bo oo p- l ,
p lly oo p, p , v , b ▲l o , po o oo p y
o l , ’ll p o o b

T o lb p o o o o pl o l o o p o bl

o bl v o S v b po , b po o o b b o
oo p, p , v , b ▲l o , p b o po o
oo p o l

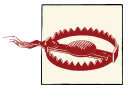
Driver code. T v p y bo l pl o , p
`MultipleOutputs` o b `MultipleOutputs` , `bins`,
pp o o p T lly o p
o yo o b p o bl by l , o b o o o
yo o ’ p l b o o p W l o b o
o o , p-o ly o b

```
...
// Configure the MultipleOutputs by adding an output called "bins"
// With the proper output format and mapper key/value pairs
MultipleOutputs.addNamedOutput(job, "bins", TextOutputFormat.class,
        Text.class, NullWritable.class);

// Enable the counters for the job
// If there are a significant number of different named outputs, this
// should be disabled
MultipleOutputs.setCountersEnabled(job, true);

// Map-only job
job.setNumReduceTasks(0);
...
```

Mapper code. T pp o `MultipleOutputs` o
T pp o o v l - l o o o po

o o o o po o ,
o b o l pl ll lly b pl y
o pp o p b F lly, bo yo po o
o o o p o , o p o p b

o b MultipleOutputs l p! , yo y o v
o p ll



T yp l l , part-mnnnnn, ll b lo p
o y T l ll b p y l Contexto b o
y/v l p , l ll b *bin_name*-mnnnnn
o lb pl , *bin_name* ll b , hadoop-tag, pig-tag, hive-tag,
hbase-tag,o hadoop-post

N o o p o o o b o NullOutputFormat
ll o v p yo p l mapred p
▲ , o p l o o o _tem
porary o y o o o p o y DFS T
y b v o o o p

```
public static class BinningMapper extends
    Mapper<Object, Text, Text, NullWritable> {

    private MultipleOutputs<Text, NullWritable> mos = null;

    protected void setup(Context context) {
        // Create a new MultipleOutputs using the context object
        mos = new MultipleOutputs(context);
    }

    protected void map(Object key, Text value, Context context)
            throws IOException, InterruptedException {

        Map<String, String> parsed = MRDPUtils.transformXmlToMap(value
                .toString());

        String rawtags = parsed.get("Tags");

        // Tags are delimited by ><. i.e. <tag1><tag2><tag3>
        String[] tagTokens = StringEscapeUtils.unescapeHtml(rawtags).split(
                "><");

        // For each tag
        for (String tag : tagTokens) {
            // Remove any > or < from the token
            String groomed = tag.replaceAll(">|<", "").toLowerCase();

            // If this tag is one of the following, write to the named bin
            if (groomed.equalsIgnoreCase("hadoop")) {
```

```
                mos.write("bins", value, NullWritable.get(), "hadoop-tag");
            }
            if (groomed.equalsIgnoreCase("pig")) {
                mos.write("bins", value, NullWritable.get(), "pig-tag");
            }
            if (groomed.equalsIgnoreCase("hive")) {
                mos.write("bins", value, NullWritable.get(), "hive-tag");
            }
            if (groomed.equalsIgnoreCase("hbase")) {
                mos.write("bins", value, NullWritable.get(), "hbase-tag");
            }
        }

        // Get the body of the post
        String post = parsed.get("Body");

        // If the post contains the word "hadoop", write it to its own bin
        if (post.toLowerCase().contains("hadoop")) {
            mos.write("bins", value, NullWritable.get(), "hadoop-post");
        }
    }

    protected void cleanup(Context context) throws IOException,
            InterruptedException {
        // Close multiple outputs!
        mos.close();
    }
}
```

Total Order Sorting

Pattern Description

T *total order sorting* p o o o o o o o

Intent

o o o yo p ll lo o y

Motivation

So y q l p o So M p ,o o lly p ll l, o y T b yp l v o q pp o b o pply

v l ll o by y, b o ly, o o
bb l o ll W o o o l o o , yo
o o p l , o o o o o p o
pl M p o b, o ll b o , b o l ll o
b

So b o l p o p So by , p o v l
v o F o b o b y
o l o M p , o pp bo
bo o l by bo o l o , p v ly T b
l o o , ll, o o p y o
So b b l bo o o p y yo
o l

T o l o o o v o o ppl o po
o o llo -o y po o v , v o o M p
v l l p po , o o p lly p v o p o o ly o b o p ly

Applicability

T q p yo bv o yo o y o b o p bl o
b o

Structure

To lo o y b o o o o pl p yo 'll T o
yo v o o p o v by o v l
ll p o q l- b o T ll
ll o o T o l o p o p
o p o o p o by o y T bo o
o o , o o o , o o o o

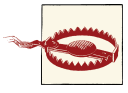
T p o p ly p , o
p lly o T ly p o p o l o y o
o o ly o b o o yo o o q ly o v ,
b v l p o ll o o p o ll ▲l o , o ,
yo y b bl o p o yo l , p lly v ly
b Fo pl , yo o o by D, yo v ll o
, yo o , o o v
o o T b o by D o l b p o v ly
yo o b o o l , yo v b by b
o yo o

T ly p o pl o T p o b o
o pl T p pl p o v ly pl o pl
o l v ly pl l ll T o ly p o llo

- T pp o pl o pl W v o , o p o y o p y o ll o p o W o ’ ll bo l o , o ’ll ll v l o v o p
- ▲ o , b o o o l o p o o yo ’ll o ly o o bl pl Fo pl , yo pl o o o , pl bo o o o l v , v p o ▲ yo v b ll o o , v 1 , by 1, , , T v 1, 1% o o o l b o ly p
- lyo ll b T ll o ll o y o o o l y o o , o ll b y T , llo v b o ll , l o y ll b l o bo

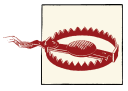
T o p l v ly o ppl o o M p o p o T o o p o llb

- T pp o y y ly p o v , o l o v l o b o
- ▲ o p o lb p p o l oo p, yo `TotalOrderPartitioner`, b l p lly o p po o p o l po p vo p o o
- T ’ o b pl T l o o vy l T o ply v l v o o p T b o o b q l o b o p o o `TotalOrderPartitioner` o o po p ly



No b o p o o b q l o b o o p yo o b o yo ’v b l , yo ’ll o b l

yo o v p y o y o y o y, o y , l by o Fo pl , yo o o by l , y o , y lbo l `Smith^Baltimore`



T o ly v y oo p v y l ’
o o l o o
l , o ! T "10000" l "9" y
o p , o p
b o o l yp

Consequences

T o p l ll o o , o p l ll b o
o l o oo p, yo ’ll b bl o `hadoop fs -cat output/part-r-*` v o

Resemblances

SQL

SQ p y y!

```
SELECT * FROM data ORDER BY col1;
```

Pig

y lly p y y, b ’ v y p v o p o
, ll l - M p o b o p o ,
p o l o

```
c = ORDER b BY col1;
```

Performance analysis

T o p o p v b yo v ly v o b p
o b l p o , o lly o

T o b b l p o o o lyo
l o o o v o T o p l ll,
o o v l ▲l o , yo yo ly v o o , ll
o o o b l o v

T o po o b p o l o o o
o p , b o o v llo o v o
llo b o T o , yo o l l v lyl b o

Total Order Sorting Examples

Sort users by last visit

T o S v b o o o ’ o
, ’ l o v o by l y v v

Fo pl , v p l v bo ly o p
▲l o , o o M p o b ,o o ly o o o

Driver code. ’ b v o o o o b l p o l v
pl , p o o

T o p p o l p o p
v bl o p l o p o l o y
T p o l by TotalOrderPartitioner o y/v l
p o p o p ly T o y o o o p
b o o b T o oo p l o b o o T
o o o b p-o ly o ly o b SequenceFi
leOutputFormat

```
    public static void main(String[] args) throws Exception {
        Configuration conf = new Configuration();
        Path inputPath = new Path(args[0]);
        Path partitionFile = new Path(args[1] + "_partitions.lst");
        Path outputStage = new Path(args[1] + "_staging");
        Path outputOrder = new Path(args[1]);

        // Configure job to prepare for sampling
        Job sampleJob = new Job(conf, "TotalOrderSortingStage");
        sampleJob.setJarByClass(TotalOrderSorting.class);

        // Use the mapper implementation with zero reduce tasks
        sampleJob.setMapperClass(LastAccessDateMapper.class);
        sampleJob.setNumReduceTasks(0);

        sampleJob.setOutputKeyClass(Text.class);
        sampleJob.setOutputValueClass(Text.class);

        TextInputFormat.setInputPaths(sampleJob, inputPath);

        // Set the output format to a sequence file
        sampleJob.setOutputFormatClass(SequenceFileOutputFormat.class);
        SequenceFileOutputFormat.setOutputPath(sampleJob, outputStage);

        // Submit the job and get completion code.
        int code = sampleJob.waitForCompletion(true) ? 0 : 1;

        ...
```

T o o b y pp o pl o T p
o p o o b, o ’ll y pp o o p y/v l
p y o o o p T o b o o 1 , b y
o bl b b N , p o l o , v o
v o y

T p o l InputSampler l y T pl p o
l by o o p o yo o b RandomSam
pler, o bl b o pl o p vo o b' o p T
b p v o p o , o p o o ▲ o
o o o RandomSampler lb yo o b o p pl ll b
pl T ll o , b yo o oo b o
▲ p o l , o b T p o l
o y l , y o b o pl

yo b o l ly o , o l b o l
o p p o l o b o v o v
o o b v o y

```
...

if (code == 0) {
    Job orderJob = new Job(conf, "TotalOrderSortingStage");
    orderJob.setJarByClass(TotalOrderSorting.class);

    // Here, use the identity mapper to output the key/value pairs in
    // the SequenceFile
    orderJob.setMapperClass(Mapper.class);
    orderJob.setReducerClass(ValueReducer.class);

    // Set the number of reduce tasks to an appropriate number for the
    // amount of data being sorted
    orderJob.setNumReduceTasks(10);

    // Use Hadoop's TotalOrderPartitioner class
    orderJob.setPartitionerClass(TotalOrderPartitioner.class);

    // Set the partition file
    TotalOrderPartitioner.setPartitionFile(orderJob.getConfiguration(),
            partitionFile);

    orderJob.setOutputKeyClass(Text.class);
    orderJob.setOutputValueClass(Text.class);

    // Set the input to the previous job's output
    orderJob.setInputFormatClass(SequenceFileInputFormat.class);
    SequenceFileInputFormat.setInputPaths(orderJob, outputStage);

    // Set the output path to the command line parameter
    TextOutputFormat.setOutputPath(orderJob, outputOrder);

    // Set the separator to an empty string
    orderJob.getConfiguration().set(
```

```
                "mapred.textoutputformat.separator", "");

            // Use the InputSampler to go through the output of the previous
            // job, sample it, and create the partition file
            InputSampler.writePartitionFile(orderJob,
                    new InputSampler.RandomSampler(.001, 10000));

            // Submit the job
            code = orderJob.waitForCompletion(true) ? 0 : 2;
        }

        // Clean up the partition file and the staging directory
        FileSystem.get(new Configuration()).delete(partitionFile, false);
        FileSystem.get(new Configuration()).delete(outputStage, true);

        System.exit(code);
    }
```

Analyze mapper code. T pp ply p ll l o
o y o o T p v l o p b T y/v l
p , p o o b o o , o SequenceFile o
p o l o TotalOrderPartitioner T o o o b

```
    public static class LastAccessDateMapper extends
            Mapper<Object, Text, Text, Text> {

        private Text outkey = new Text();

        public void map(Object key, Text value, Context context)
                throws IOException, InterruptedException {

            Map<String, String> parsed = MRDPUtils.transformXmlToMap(value
                    .toString());

            outkey.set(parsed.get("LastAccessDate"));
            context.write(outkey, value);
        }
    }
```

Order mapper code. T o b ply y pp o p y/v l
p o p l o o o pl o

Order reducer code. TotalOrderPartitioner o o o ll o , ll
o o o p v l NullWritableo b T ll p o
p l o o by l T p o
o o o ll p l o p o o llyo

```
    public static class ValueReducer extends
            Reducer<Text, Text, Text, NullWritable> {
```

```
        public void reduce(Text key, Iterable<Text> values, Context context)
                throws IOException, InterruptedException {
            for (Text t : values) {
                context.write(t, NullWritable.get());
            }
        }
    }
```

Shuffling

Pattern Description

T o lo o *shuffling* p o ppo o , b l l o o o o o

Intent

o v o o yo o o pl ly o

Motivation

T o l p b bo pply o o o o o yo p o p bo o pl ly o y o

T o o ly b , b o o l o p po o o y ▲ o o o p bl o pl

▲ o y ly b o po o o o o ’p v y, b ll ly T o o p o v o o o l o yo y l , o o p o o y

▲ o o o l o b bl o p o o o o p bl o pl Fo pl , o ll b pl o pl v y p ll o , ’ll pl T lo ly o v o pl o v p bl l ▲l o , p o b o ’ v o b o p o pl o pl v y yo pl

Structure

- ▲ll pp o o p o v l lo o y
- T o o y , o

o o , o o o T , o o
o y o , p o o o

T pp l p b ly o y T o l
b oo o o y by o
o o o y o

Consequences

o p l o o o

Resemblances

SQL

T SQ q v l o o o by o v l , o
o o l bl T o o o p o b o
o o b , ll p o o o W o ’ v o o
ll y o o lo M p , p vo p T
b o o

```
SELECT * FROM data ORDER BY RAND()
```

Pig

S l b o SQ p o ORDER BY o
o o l , o o lo y ,
GROUP BY o y, FLATTEN o p T v ly pl
l p p o po b

```
c = GROUP b BY RANDOM();
d = FOREACH c GENERATE FLATTEN(b);
```

Performance analysis

T l o v y p o p o p S o
o o o pl ly o , b o o ll b o pl ly
b l W o , ll b o p o T o l ll
l o b v y p bl o v by b o
T y o p l o p

, yp l p o p o p o o p p
pply T p l llo o v o llo b
o DFS, o l v ly b o o l b

Shuffle Examples

Anonymizing StackOverflow comments

To o y S v b o , pl p o D o D, o T l

T o lb p o o o o pl o l o o p o bl

o bl v l o S v b o , o y o by o v D , o v o o , o ly l o

Mapper code. T pp o o l y o p XM b bo , o b o b o l o XM Do o D, o o , o lb ‘T’ o v o o , o XM b v l ▲ o y o p b ly o o

```
public static class AnonymizeMapper extends
    Mapper<Object, Text, IntWritable, Text> {

private IntWritable outkey = new IntWritable();
private Random rndm = new Random();
private Text outvalue = new Text();

public void map(Object key, Text value, Context context)
        throws IOException, InterruptedException {

    Map<String, String> parsed = MRDPUtils.transformXmlToMap(value
            .toString());

    if (parsed.size() > 0) {
        StringBuilder bldr = new StringBuilder();
        // Create the start of the record
        bldr.append("<row ");

        // For each XML attribute
        for (Entry<String, String> entry : parsed.entrySet()) {

            // If it is a user ID or row ID, ignore it
            if (entry.getKey().equals("UserId")
                    || entry.getKey().equals("Id")) {
            } else if (entry.getKey().equals("CreationDate")) {

                // If it is a CreationDate, remove the time from the date
                // i.e., anything after the 'T' in the value
                bldr.append(entry.getKey()
                        + "=\""
```

```
                        + entry.getValue().substring(0,
                                entry.getValue().indexOf('T')) + "\" ");
                } else {
                    // Otherwise, output the attribute and value as is
                    bldr.append(entry.getKey() + "=\"" + entry.getValue()
                            + "\" ");
                }

            }
            // Add the /> to finish the record
            bldr.append("/>");

            // Set the sort key to a random value and output
            outkey.set(rndm.nextInt());
            outvalue.set(bldr.toString());
            context.write(outkey, outvalue);
        }
    }
}
```

Reducer code. T l o p v l o o po o y

```
public static class ValueReducer extends
        Reducer<IntWritable, Text, Text, NullWritable> {

        protected void reduce(IntWritable key, Iterable<Text> values,
            Context context) throws IOException, InterruptedException {

        for (Text t : values) {
            context.write(t, NullWritable.get());
        }
    }
}
```

CHAPTER 5
Join Patterns

v ll yo o y Fo pl , p yo v
o o o SQ b b p q ly M l ,
b lo v o p ly o DFS ▲l o , ly
ly o lo o o o DFS l
o o yp po o y T l o o

D llo v pl , l ’ v y v l bl o o , o v
l o p ly o T o p
o o pl y Jo b o ll o
y b o l o o l o o yp o p 11 T
o o o ll

SQ , o o pl pl o , b
l llo o S ly o , o M p o ly pl
M p o p o l y/v l p , yp lly o p
W o o l o p o b blyo ,
o o o o o o o p o o ly
Typ lly, o l o p o o o o p o , o o o o p o ll
q v y l by o p o b o p , v y o
yo o Fo pl , o by o o o o by
o l q l o by o o b — ’ b o y l
o lo b o

o po llo o pl y o ,o o b yo o b
o y o o pl o bo o
o pl p , lo o -o o o o lo o
o p ▲ yo l po b l , q o o o

p ▲ y M p o p o , o b v y
po o o v y o bo ▲ y o o
o o o l , o o p o
p

o p o p b o p o o o l o yp
o o o ▲ p o oo , p l lyo o l
, o yo o , yp o o yo o ,
C p o o pl ly , b o b

T p p , *reduce side join*, o b , bo
o v o bo l ▲ , o p
p o o o p o o p- b o
oo p M p ▲ F lly, bo o o
yo p o C p o

C oo yp o o o yo o b ll M o p y
l o o ▲ppl b l y o o o p
p o

A Refresher on Joins

yo o o o SQ b o , yo p o b bly p o , b o
o o oo p, o y b b o o o p

Jo po blyo o o o pl o p o o M p
y , M p v y oo p o l by bo v y
o o o p o l o , o o o v y l o o o o
p lly o v o p lv , l ’ o o v
y *join* yp o o

▲*join* o p o o b o o oo o b o l
o o l , o *foreign key* T o y l l o l bl
o l o o bl , o o -
b bl pl pl y o o bo pl o , o l ’ v
To pl y pl o o o yp , o ll b , *A* *B*,
o y *f* ▲ yp o o b , p
o bl *A* T bl 5-1 *B* T bl 5- , y ll b p o
p o

Table 5-1. Table A

User ID	Reputation	Location
3	3738	New York, NY
4	12946	New York, NY
5	17556	San Diego, CA
9	3443	Oakland, CA

Table 5-2. Table B

User ID	Post ID	Text
3	35314	Not sure why this is getting downvoted.
3	48002	Hehe, of course, it's all true!
5	44921	Please see my post below.
5	44920	Thank you very much for your reply.
8	48675	HTML is not a subset of XML!

INNER JOIN

W p o pl o ' p y yp o o y y o , lly y l bo *inner join*. W yp o o , o o bo *A* *B* o l v l o v o y *f* b o o , ll o l o bo *A* *B* o bl o o v l o *f* o *A* b o *B*, v v , o p l bl o o o p o

T bl 5-3 o l o o o p o b *A* *B* D *f*

Table 5-3. Inner Join of A + B on User ID

A.User ID	A.Reputation	A.Location	B.User ID	B.Post ID	B.Text
3	3738	New York, NY	3	35314	Not sure why this is getting downvoted.
3	3738	New York, NY	3	48002	Hehe, of course, it's all true!
5	17556	San Diego, CA	5	44921	Please see my post below.
5	17556	San Diego, CA	5	44920	Thank you very much for your reply.

o D o 3 o 5 p bo bl , o y ll b l bl 4 9 bl *A* 8 bl *B* o p o bl , o o ll b o o v , o ll b p yp o o o , b o o yp o o !

OUTER JOIN

▲ o o l o o , b o o y o p
bo bl ll b l bl T yp o o o
yp ll ly o ll b l bl

left outer join, o l bl ll b l bl ,
ll v l o l o bl o o o
y o p bl ll b ▲ *right outer*
join l o , b bl o p
l bl v l ll pp o p ▲ *full outer join* ll o ll
o o bo bl , o o l o b o o bo l
o o

T bl 5-4 o l o l o o o p o b *A* *B* o
D

Table 5-4. Left Outer Join of A + B on User ID

A.User ID	A.Reputation	A.Location	B.User ID	B.Post ID	B.Text
3	3738	New York, NY	3	35314	Not sure why this is getting downvoted.
3	3738	New York, NY	3	48002	Hehe, of course, it's all true!
4	12946	New York, NY	null	null	null
5	17556	San Diego, CA	5	44921	Please see my post below.
5	17556	San Diego, CA	5	44920	Thank you very much for your reply.
9	3443	Oakland, CA	null	null	null

o D o 3 o 5 p bo bl , o y ll b
l bl 4 9 bl *A* o o v o po v l bl
B, b l o o *A* o l , ll b p b
o ll v l o l p o ly bl *B* 8 *B* o o v
A, o o

T bl 5-5 o l o o o p o b *A* *B* o
D

Table 5-5. Right Outer Join of A + B on User ID

A.User ID	A.Reputation	A.Location	B.User ID	B.Post ID	B.Text
3	3738	New York, NY	3	35314	Not sure why this is getting downvoted.
3	3738	New York, NY	3	48002	Hehe, of course, it's all true!
5	17556	San Diego, CA	5	44921	Please see my post below.
5	17556	San Diego, CA	5	44920	Thank you very much for your reply.
null	null	null	8	48675	HTML is not a subset of XML!

▲ , o Do 3o 5 p bo bl , o y ll b
l bl 8 *B* o o v *A*, b p b *B*
bl 4 9 o y o ’ v bl *B*

T bl 5-6 o l o llo o o p o b *A* *B*o
D

Table 5-6. Full Outer Join of A + B on User ID

A.User ID	A.Reputation	A.Location	B.User ID	B.Post ID	B.Text
3	3738	New York, NY	3	35314	Not sure why this is getting downvoted.
3	3738	New York, NY	3	48002	Hehe, of course, it’s all true!
4	12946	New York, NY	null	null	null
5	17556	San Diego, CA	5	44921	Please see my post below.
5	17556	San Diego, CA	5	44920	Thank you very much for your reply.
null	null	null	8	48675	HTML is not a subset of XML!
9	3443	Oakland, CA	null	null	null

, o Do 3o 5 p bo bl , o y ll
b l bl 4, 8, 9 p l bl v o
y o o o p v o ppo bl

ANTIJOIN

▲ *antijoin* llo o o T , l bl
o o ly o o o o *f*

T bl 5-7 o l o o o p o b *A* *B*o D

Table 5-7. Antijoin of A + B on User ID

A.User ID	A.Reputation	A.Location	B.User ID	B.Post ID	B.Text
4	12946	New York, NY	null	null	null
null	null	null	8	48675	HTML is not a subset of XML!
9	3443	Oakland, CA	null	null	null

4, 8, 9 o o o v l o *f* bo bl , o y l
bl o o 3 5 o p , y bo bl

CARTESIAN PRODUCT

▲ *Cartesian product*o *cross product* o o bl
p v y o o o bl bl *X* o *n* o bl *Y*
o *m* o , o po o *X* *Y*, o *X* × *Y*, o *n* × *m*

o l o o o p o , C p o o o o
o y ▲ ll p o p , o p o ly
p v o p o o yo pl , M p o
p o

T bl 5-8 o l o C p o b *A* *B*

Table 5-8. Cartesian Product, A × B

A.User ID	A.Reputation	A.Location	B.User ID	B.Post ID	B.Text
3	3738	New York, NY	3	35314	Not sure why this is getting downvoted.
3	3738	New York, NY	3	48002	Hehe, of course, it's all true!
3	3738	New York, NY	5	44921	Please see my post below.
3	3738	New York, NY	5	44920	Thank you very much for your reply.
3	3738	New York, NY	8	48675	HTML is not a subset of XML!
4	12946	New York, NY	3	35314	Not sure why this is getting downvoted.
4	12946	New York, NY	3	48002	Hehe, of course, it's all true!
4	12946	New York, NY	5	44921	Please see my post below.
4	12946	New York, NY	5	44920	Thank you very much for your reply.
4	12946	New York, NY	8	48675	HTML is not a subset of XML!
5	17556	San Diego, CA	3	35314	Not sure why this is getting downvoted.
5	17556	San Diego, CA	3	48002	Hehe, of course, it's all true!
5	17556	San Diego, CA	5	44921	Please see my post below.
5	17556	San Diego, CA	5	44920	Thank you very much for your reply.
5	17556	San Diego, CA	8	48675	HTML is not a subset of XML!
9	3443	Oakland, CA	3	35314	Not sure why this is getting downvoted.
9	3443	Oakland, CA	3	48002	Hehe, of course, it's all true!
9	3443	Oakland, CA	5	44921	Please see my post below.
9	3443	Oakland, CA	5	44920	Thank you very much for your reply.
9	3443	Oakland, CA	8	48675	HTML is not a subset of XML!

Reduce Side Join

Pattern Description

T *reduce side join* p b o o p o o
o p , b pl o pl pp o ll o o p o
p v o o

Intent

Jo l l pl o by o o y

Motivation

▲ o blyo o pl o o o M p ,
o v y v o b o yo yp o
o b bo v l v o l o o o yo
▲l o , o y o o yo ▲ll ,
o ll l ly q l o o o b b
b l o o p T o , b yo v
o v l bl ’ o bo o , by ll !
o ly, llo l , yp o o y b yo o ly o

Applicability

▲ o o l b

- M l pl *large* b o by o y ll b o o
b o o y, y pl o
- o l b l yo b bl o y o o p o

Structure

- T pp p p o o p o by p o o o
o y o o T o y
o p y, p o o p v l T o p
v l l by o q o , *Ao B* o
S F 5-1
- ▲ p o b ,o o p o b o
b y/v l p o v ly o
- T p o o o p o by o ll v l o
p o p o po y l Fo pl , ll o l *A* o
‘▲’ l ll o l *B* o ‘ ’ l T l
o v o o bo o o Fo
o , o o o p ll l o p y Fo o o l ,
,o ll , p y l ll o o p y l T o o
by lyo l p y T o o o - p y l
p y bl

```
-- Inner Join
A = JOIN comments BY userID, users BY userID;

-- Outer Join
A = JOIN comments BY userID [LEFT|RIGHT|FULL] OUTER, users BY userID;
```

Performance analysis

▲pl o p lo o o l ’ o o
yo p o o p lo o o
b l o , p y llo ll b o l o
p Fo o , o ll yp lly l l v ly o
yo yp l ly

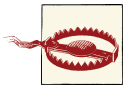
yo o p b p b o C
p o , o yo o o So b o p
o lyo

Reduce Side Join Example

User and comment join

pl , ’ll b o bl o S v lo
So , o p v
o y T o l lo p o o
l o v , v o po p o bl o o o
o o o T o o o , o
b o D o y pl ,
’ll p o ,o , o T o o o o
o b o o

oo p ppo b l y o l pl p yp o , llo yo o
pp l p o o p pl o o
T ly lp l, b yo o ’ v o o lo o o
p p pl o o llo pl , o pp l
o o o o o pp l o p
D o y, o v l lo l
o l o o T o p ll v l
o o p o y, p o o o T
o o o o o p



v o p y v l yp o b l o
llo pp l

T o lb p o o o o pl o l o o p o bl
o bl v o o o l o ’ o ,
o o o bo o o

Driver code. T o b o o l ly o o o
o o l pl p l yW l o o yp o b o
o o args[2] o b T l v p o v o
o MultipleInput o lb

```
...
// Use MultipleInputs to set which input uses what mapper
// This will keep parsing of each data set separate from a logical standpoint
// The first two elements of the args array are the two inputs
MultipleInputs.addInputPath(job, new Path(args[0]), TextInputFormat.class,
        UserJoinMapper.class);
MultipleInputs.addInputPath(job, new Path(args[1]), TextInputFormat.class,
        CommentJoinMapper.class);

job.getConfiguration()..set("join.type", args[2]);
...
```

User mapper code. T pp p p l o XM b
D o o o p lb p v l
p p l A o o v l T lb o o
v l o

```
public static class UserJoinMapper extends Mapper<Object, Text, Text, Text> {

    private Text outkey = new Text();
    private Text outvalue = new Text();

    public void map(Object key, Text value, Context context)
            throws IOException, InterruptedException {

        // Parse the input string into a nice map
        Map<String, String> parsed =
                MRDPUtils.transformXmlToMap(value.toString());

        String userId = parsed.get("Id");

        // The foreign join key is the user ID
        outkey.set(userId);

        // Flag this record for the reducer and then output
        outvalue.set("A" + value.toString());
        context.write(outkey, outvalue);
    }
}
```

W y o p v l o p , o o ' v o b T o ppo y o o p o by p o ly l o yo o o o q o p o o p , b o b ▲l o , o y po p y, yo o ' o p v l ,

Comment mapper code. T pp p p l o o XM V y l o UserJoinMapper, oo b D o o o p b p v l T o ly XM b UserId p po o o , Id D , pp p p l B o o v l

```
public static class CommentJoinMapper extends
        Mapperlt;Object, Text, Text, Text> {

    private Text outkey = new Text();
    private Text outvalue = new Text();

    public void map(Object key, Text value, Context context)
            throws IOException, InterruptedException {

        Map<String, String> parsed = transformXmlToMap(value.toString());

        // The foreign join key is the user ID
        outkey.set( parsed.get("UserId"));

        // Flag this record for the reducer and then output
        outvalue.set("B" + value.toString());
        context.write(outkey, outvalue);
    }
}
```

Reducer code. T o o ll v l o o p b o o p o o o o l ▲ ll v l b l , l o b o l T o b l ly b o yp o o , b l y wo lv o bo l o Context o b T yp o o p ll o o b o o p o ' bo o b o bo o b

```
public static class UserJoinReducer extends Reducer<Text, Text, Text, Text> {

  private static final Text EMPTY_TEXT = Text("");
    private Text tmp = new Text();
    private ArrayList<Text> listA = new ArrayList<Text>();
    private ArrayList<Text> listB = new ArrayList<Text>();
```

```
    private String joinType = null;

    public void setup(Context context) {
        // Get the type of join from our configuration
        joinType = context.getConfiguration().get("join.type");
    }

    public void reduce(Text key, Iterable<Text> values, Context context)
            throws IOException, InterruptedException {

        // Clear our lists
        listA.clear();
        listB.clear();

        // iterate through all our values, binning each record based on what
        // it was tagged with.  Make sure to remove the tag!
        while (values.hasNext()) {
            tmp = values.next();
            if (tmp.charAt(0) == 'A') {
                listA.add(new Text(tmp.toString().substring(1)));
            } else if (tmp.charAt('0') == 'B') {
                listB.add(new Text(tmp.toString().substring(1)));
            }
        }

        // Execute our join logic now that the lists are filled
        executeJoinLogic(context);
    }

    private void executeJoinLogic(Context context)
            throws IOException, InterruptedException {
        ...
    }
```

T p yp o o Texto b T p y o o y, pl ’ D T p v l o y o o o o ‘ ’, ll ll o p o ‘ ’▲ y yp o o yo o l o p o o l b o p o o o p Fo pl y, XM v l o l o p y XM v l o o o p v l

N ,l ’ bo o o yp F p o bo l o p y, ply p o o o bo p o o v l o

```
    if (joinType.equalsIgnoreCase("inner")) {
        // If both lists are not empty, join A with B
        if (!listA.isEmpty() && !listB.isEmpty()) {
            for (Text A : listA) {
                for (Text B : listB) {
                    context.write(A, B);
```

```
            }
        }
    }
} ...
```

Fo l o o , l o p y, o A B l p y, o p o o A p y

```
... else if (joinType.equalsIgnoreCase("leftouter")) {
    // For each entry in A,
    for (Text A : listA) {
        // If list B is not empty, join A and B
        if (!listB.isEmpty()) {
            for (Text B : listB) {
                context.write(A, B);
            }
        } else {
            // Else, output A by itself
            context.write(A, EMPTY_TEXT);
        }
    }
} ...
```

▲ o o v y l , p o o p y l o B o A l l p y, o o B p yo p y

```
... else if (joinType.equalsIgnoreCase("rightouter")) {
    // For each entry in B,
    for (Text B : listB) {
       // If list A is not empty, join A and B
        if (!listA.isEmpty()) {
            for (Text A : listA) {
                context.write(A, B);
            }
        } else {
            // Else, output B by itself
            context.write(EMPTY_TEXT, B);
        }
    }
} ...
```

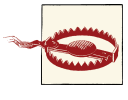
▲ llo o o o pl , o p ll o , o o pp o p l ▲ o p y, o v y l A, o B B l o p y,o o p A by l A p y, o p B

```
... else if (joinType.equalsIgnoreCase("fullouter")) {
    // If list A is not empty
    if (!listA.isEmpty()) {
        // For each entry in A
        for (Text A : listA) {
            // If list B is not empty, join A with B
            if (!listB.isEmpty()) {
```

```
                for (Text B : listB) {
                    context.write(A, B);
                }
            } else {
                // Else, output A by itself
                context.write(A, EMPTY_TEXT);
            }
        }
    } else {
        // If list A is empty, just output B
        for (Text B : listB) {
            context.write(EMPTY_TEXT, B);
        }
    }
} ...
```

Fo o , l o o l p y, o p o o o - p y l p y T o b

```
... else if (joinType.equalsIgnoreCase("anti")) {
    // If list A is empty and B is empty or vice versa
    if (listA.isEmpty() ^ listB.isEmpty()) {

        // Iterate both A and B with null values
        // The previous XOR check will make sure exactly one of
        // these lists is empty and therefore the list will be skipped
        for (Text A : listA) {
            context.write(A, EMPTY_TEXT);
        }

        for (Text B : listB) {
            context.write(EMPTY_TEXT, B);
        }
    }
} ...
```



o o o lb o p o p o p l l p p y o b lly ▲ o o p o p b ll l o l b o o p p y o b T ll p o p p o o lb -o ly

Combiner optimization. o lb p o o , o b ll o p o v o p o pl

Reduce Side Join with Bloom Filter

Reputable user and comment join

T pl v y l o p vo o , b o p o o bo l o l o o o pp o p T ll lp o o b o o o ly S y o ly o p bl , , 1,5 p o ▲ o o l b , o o o v y , p o 1,5 p o o o o o b T q ll o b p o o p o o o l o po p o pp o o o o b o , lly o / bo l p l ly l o o p o , y o b l ll llo o o p o o o T l o o p o q ll o o b o , o bo l o v l

F l o o o p o q pl o o UserJoinMapper l , b p o o v , b o o p o o v l bl o o T o o bo l , ll o o o y b o p o ▲p p o o bo l ll v l 1,5 p o

o llb pl , bo pp l ly o p vo T UserJoinMapper p o o y/v l p o o o l 1,5 p o T CommentJoin Mapper l bo l o DistributedCache p o o yo p T p vo o pl T v o l ly DistributedCache o o bo l T o o llb o , o o o o o o bo l DistributedCache b o ▲pp ▲

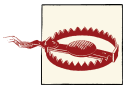
User mapper code. T D p ll o XM o b p o p o 1,5 , o o p b o y D

```
public static class UserJoinMapper extends Mapper<Object, Text, Text, Text> {

    private Text outkey = new Text();
    private Text outvalue = new Text();

    public void map(Object key, Text value, Context context)
        throws IOException, InterruptedException {

        Map<String, String> parsed = transformXmlToMap(value.toString());
```

```
            // If the reputation is greater than 1,500,
            // output the user ID with the value
            if (Integer.parseInt(parsed.get("Reputation")) > 1500) {
                outkey.set(parsed.get("Id"));
                outvalue.set("A" + value.toString());
                context.write(outkey, outvalue);
            }
        }
    }
```

Comment mapper code. T bo l lly l o DistributedCache p o o y ll o p o ▲ l o , D p ll o XM o o b p o bo l p , o o p b o y D

```
    public static class CommentJoinMapperWithBloom extends
            Mapper<Object, Text, Text, Text> {

        private BloomFilter bfilter = new BloomFilter();
        private Text outkey = new Text();
        private Text outvalue = new Text();

        public void setup(Context context) {
            Path[] files =
                    DistributedCache.getLocalCacheFiles(context.getConfiguration());
            DataInputStream strm = new DataInputStream(
                new FileInputStream(new File(files[0].toString())));
            bfilter.readFields(strm);
        }

        public void map(Object key, Text value, Context context) {
                throws IOException, InterruptedException {

            Map>String, String> parsed = transformXmlToMap(value.toString());

            String userId = parsed.get("UserId");

            if (bfilter.membershipTest(new Key(userId.getBytes()))) {
                outkey.set(userId);
                outvalue.set("B" + value.toString());
                context.write(outkey, outvalue);
            }
        }
    }
```



l o , o ' o v y ' p o
p o o o l y W l l p o v o
o p o CommentJoinMapperWithBloom, y o ' b
o p o ll b o o
o T 1 % o byo lyo p D
p o 1,5 T v o o
bo l v ly b o o o p
o pp p o o o bo l l p o v
o y ll yo o o p o

Replicated Join

Pattern Description

▲ *replicated join* p l yp o o o p o b o l y ll b p o o p-

Intent

T p o pl ly l o l y o p

Motivation

▲ pl o ly l, b l o ll b o o o b o ▲ll p v y l o lly o o y p p o p , l by JVM p yo l v l o , yo b b o p ll, o o l o o T o o ly p p , v y l b p o M p o b

T o l o pl o lly lo ly o o l o o l l T o o yp q p o o p yo l ▲l o y o b o o o y o v p , o l b o p pl o , ll p o l o o

Applicability

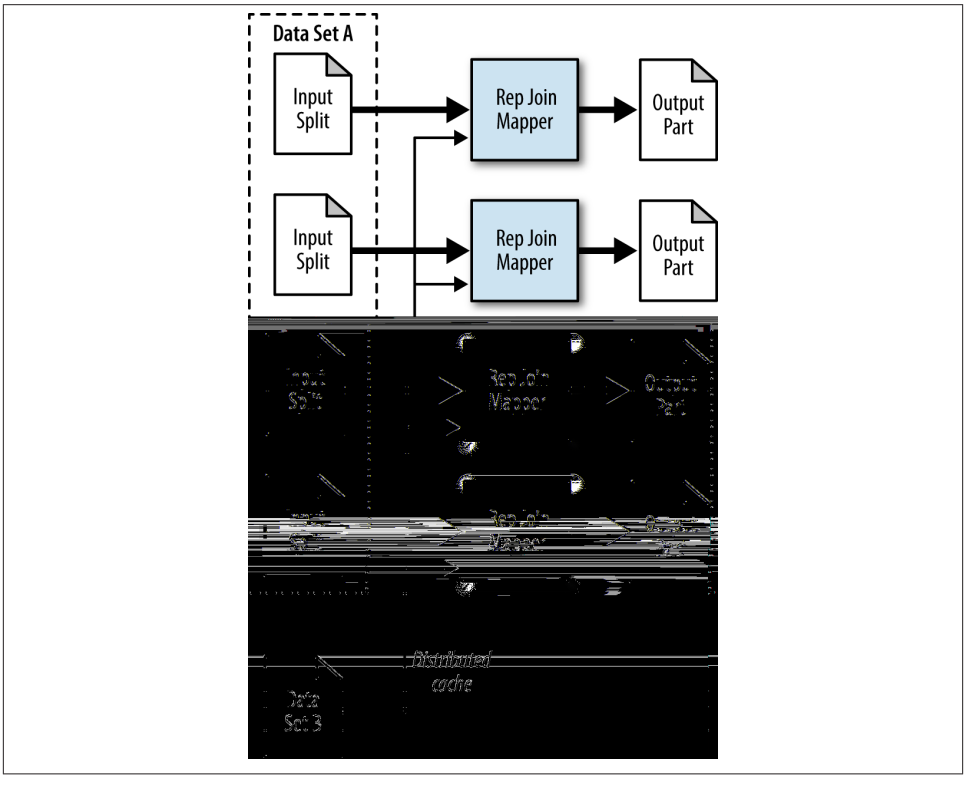
▲ pl o o l b

- T yp o o o o o l o o , l p b l p o o p o

- ▲llo , p o l o , b o o yo p

Structure

- T pp po bl o ll l o b p p o o - o y bo p bl ▲ p p o pl , pp p o o o ll o - o y o y o o - o y , o o o o p , b o o yp S F 5-
- N o o b , p o ,o o p p-o ly



Consequences

T o p b o p l q v l o b o p T p l o ll o o o l o o , p o M p ly ll b o p ll, po bl ll v l

Resemblances

Pig

v ppo o pl o o pl o o o o o p o y ly l o o ppo o pl o , o o o l ’ o bo v T o o l o o b ll b l o - o y

```
huge = LOAD 'huge_data' AS (h1,h2);
smallest = LOAD 'smallest_data' AS (ss1,ss2);
small = LOAD 'small_data' AS (s1,s2);
A = JOIN huge BY h1, small BY s1, smallest BY ss1 USING 'replicated';
```

Performance analysis

▲ pl o b yp o o b o q , b o o T l o o o o b o ly JVM, l ly p o o o y yo ll o v o p p o yo o o yo o o y p o o lly pl p ▲l o , b o y oo p o yo o - o y o ly b o by o o o T ll b l o J v o b o v T lly, yo o y yo o yo ll o

Replicated Join Examples

Replicated user comment example

T pl lo ly l o p vo pl o lo o l pl T `DistributedCache` l o p l o o ll p , b o lo o l p o o , l o o y o l o ll v b o o , o p p

T o llo p o o o o pl o l o o p o bl

o bl v ll o o o l o o , o o o

Mapper code. D p p o pp , o DistributedCache o o y o p D p ll o o o T , D o o HashMap o v l map o T o o o y o o l o , o o l o , o lo v o l o , yo ll v o JVM o pl o

▲ p, o v ll o map o p o Fo p o , D p ll o o T D o v v l o HashMap b l p p o p v l o , p v l o p b v v l v l o o , b l o o b , p v l o p p y Text o b T ' ll o ! T p o o y

```
public static class ReplicatedJoinMapper extends
        Mapper<Object, Text, Text, Text> {

  private static final Text EMPTY_TEXT = new Text("");
    private HashMap<String, String> userIdToInfo = new HashMap<String, String>();

    private Text outvalue = new Text();
    private String joinType = null;

    public void setup(Context context) throws IOException,
            InterruptedException {
        Path[] files =
                DistributedCache.getLocalCacheFiles(context.getConfiguration());
        // Read all files in the DistributedCache
        for (Path p : files) {
            BufferedReader rdr = new BufferedReader(
                    new InputStreamReader(
                            new GZIPInputStream(new FileInputStream(
                                    new File(p.toString())))));

            String line = null;
            // For each record in the user file
            while ((line = rdr.readLine()) != null) {

            // Get the user ID for this record
                Map<String, String> parsed = transformXmlToMap(line);
                String userId = parsed.get("Id");

                // Map the user ID to the record
                userIdToInfo.put(userId, line);
            }
        }

        // Get the join type from the configuration
        joinType = context.getConfiguration().get("join.type");
    }
```

```
        public void map(Object key, Text value, Context context)
                throws IOException, InterruptedException {

            Map<String, String> parsed = transformXmlToMap(value.toString());

            String userId = parsed.get("UserId");
            String userInformation = userIdToInfo.get(userId);

            // If the user information is not null, then output
            if (userInformation != null) {
                outvalue.set(userInformation);
                context.write(value, outvalue);
            } else if (joinType.equalsIgnoreCase("leftouter")) {
                // If we are doing a left outer join,
                // output the record with an empty value
                context.write(value, EMPTY_TEXT);
            }
        }
    }
```

Composite Join

Pattern Description

▲ *composite join* p l yp o o o p o b p o o p- y v y l o p

Intent

p o pl ly l o l o ll o p o v , q o b l yo o p p v y p y

Motivation

Co po o p l ly l yo o o v y l o o v , b o by o y, p o by o y, v y p l o o yp o o W , yo b yo yo p p yo , o po o l - po v o yp

o o p b l ppo o o po o `CompositeInputFormat` T o l y o o ly llo o T p o pp b p o o p y, p b v o b o p o o o , ll o o p l o y b p o lly, o o ly o p o v l o b b o o y, o p l ’

pl bl , , o b DFS bb o pp y , o o
o p p p o ppl bl yo yo l v
o o p o o o po o , yo p o b bly b o
o l o p by y ly

Applicability

▲ o po o o l b

- ▲ o llo o
- ▲ll ly l
- ▲ll b o y p y o pp
- ▲ll v b o p o
- p o o by o y, ll o y o
  p o o T , p o *X* o *A* *B* o
  o y o y p o ly p o *X* Ho
  v l o o p o o y, o F 5-3
- T o o o y v o b p p

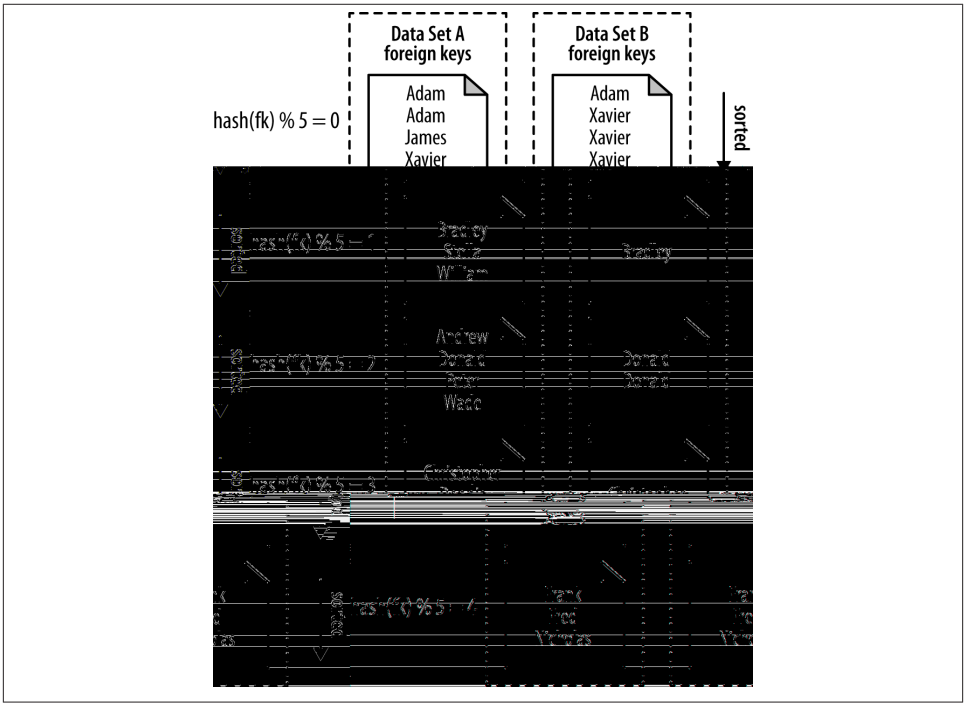


Figure 5-3. Data sets that are sorted and partitioned on the same key

Structure

- T v o l o o o o b o o p yp o p o o p , ll o yp o T o l l o S F 5-4
- T pp v y v l T o v l v o p pl ply o p o l y
- N o o b , p o , o o p p-o ly

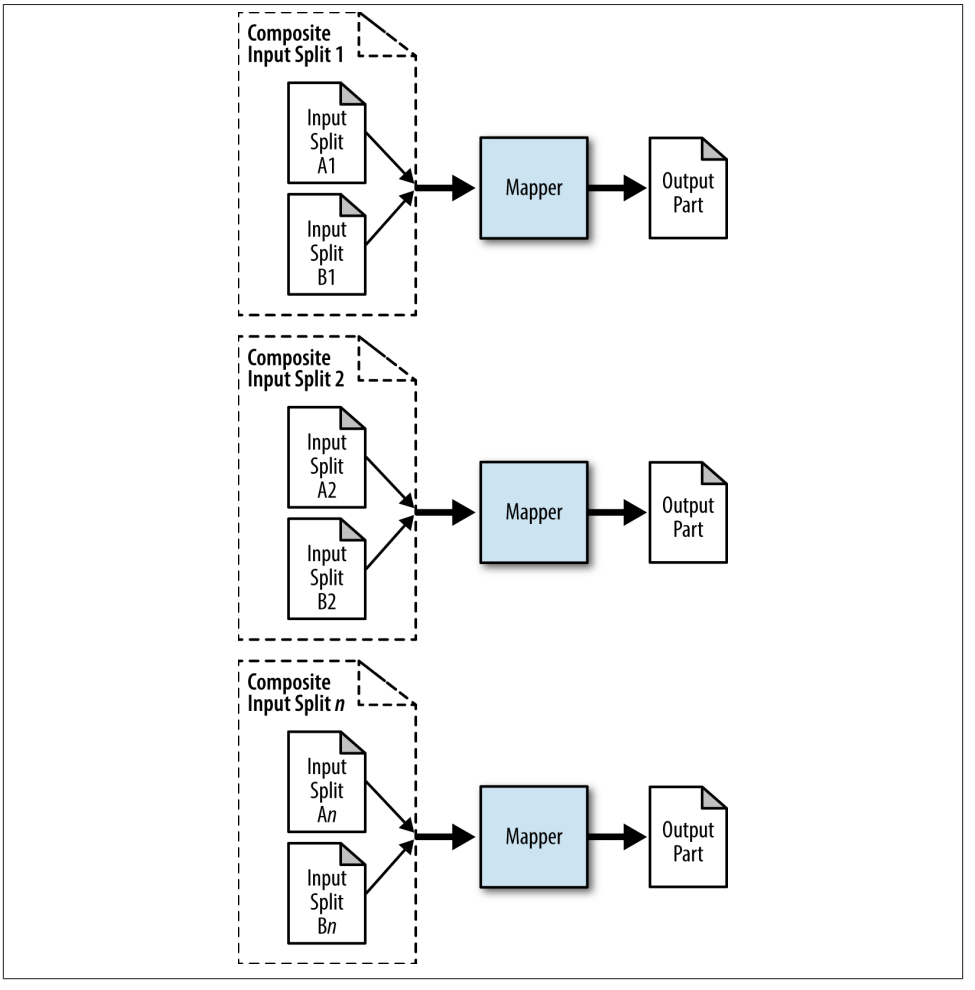


Figure 5-4. The structure of the composite join pattern

Consequences

T o p b o p l q v l o b o p T p l o ll o o o o o o o , y b ll v l

Performance analysis

▲ o p o b l v ly q ly o v l o v , M p o o ly p o b o o o o lo l T p v l p o by y o b o b o o

▲ y o o p p o o o o p o o ly T p p o o b yp lly M p o b, b ly , o p o b o v o v T , o o p o p p v o o v llo

Composite Join Examples

Composite user comment join

To p o o o o p o , bo o v b p p o by M p o p `TextOutputFormat` T y o D, v l XM o o XM , b o o o p `KeyValueTextOutputFormat` p o ly q T y ll b o p y o o o o b D v l ll b o p v l o o

o by o y, v b y o `Text` o b `LongWritable` o b T , 1 345 o b o T b `CompositeInputFormat` `Text` o b y o o p o o o pp o p v o b pl T v o o o o o M p o l o , l pp o v l

T o llo p o o o o pl o l o o p o bl

o bl v o l o o o o o ,
o o o

Driver code. T v p p o o b p o , p o o , ly o p o y, yp o o o o T `CompositeInputFormat` l o l `mapred` ▲ , b o o l o `mapreduce` ▲ T o p o o o p o o o p o o p o

T p o lp o o o p o l
o yp o o , p o l o p ll ,
y Patho Stringo b , p o
o o

T ’ ll o ! ▲ q p , o b
l o pl o p o

Fo o , o o o bo l o
o p o b o CompositeInputFormat
o o

```
public static void main(String[] args) throws Exception {

    Path userPath = new Path(args[0]);
    Path commentPath = new Path(args[1]);
    Path outputDir = new Path(args[2]);
    String joinType = args[3];

    JobConf conf = new JobConf("CompositeJoin");
    conf.setJarByClass(CompositeJoinDriver.class);
    conf.setMapperClass(CompositeMapper.class);
    conf.setNumReduceTasks(0);

    // Set the input format class to a CompositeInputFormat class.
    // The CompositeInputFormat will parse all of our input files and output
    // records to our mapper.
    conf.setInputFormat(CompositeInputFormat.class);

    // The composite input format join expression will set how the records
    // are going to be read in, and in what input format.
    conf.set("mapred.join.expr", CompositeInputFormat.compose(joinType,
            KeyValueTextInputFormat.class, userPath, commentPath));

    TextOutputFormat.setOutputPath(conf, outputDir);

    conf.setOutputKeyClass(Text.class);
    conf.setOutputValueClass(Text.class);

    RunningJob job = JobClient.runJob(conf);
    while (!job.isComplete()) {
        Thread.sleep(1000);
    }

    System.exit(job.isSuccessful() ? 0 : 1);
}
```

Mapper code. T p o pp o y TupleWritable T pl o b o Texto b q v l o b o ▲ po o o , o o Texto b p ly o o o T p p o , o p p , oo T pp ply b o b o pl o p T o ly o o b o pl , o y o p y v l o , o l b o o p o o b o p

```
public static class CompositeMapper extends MapReduceBase implements
        Mapper<Text, TupleWritable, Text, Text> {

    public void map(Text key, TupleWritable value,
            OutputCollector<Text, Text> output,
            Reporter reporter) throws IOException {

        // Get the first two elements in the tuple and output them
        output.collect((Text) value.get(0), (Text) value.get(1));
    }
}
```

Reducer and combiner. T p o o o b pl o b po ly

Cartesian Product

Pattern Description

T *Cartesian product* p v y o p v y o o l pl p v yo o T o l y o o o , o b p ly b o o pl

Intent

p o p v y l o v yo o

Motivation

▲C po lb l o p b v yp o o po bl b o o o o b ly p o by o y, C po ply p v y o o v y o o ll o

W , C p o o o o M p p v y
ll b o p o o v ly pl bl , o b p ll l v y ll,
q b o o p o b o o ▲ y p p o
o b o o p o v o by
o o l b o o p o v

v y yo o l o o C p o , b o
ply o o y o o o o p o oo o pl o o p by
o Mo o C p o o o o l y ly
o o o

Applicability

C p o

- o o ly l o p b ll p o v l o
- o ’v llo o o lv p o bl
- o v o o o o

Structure

- T o p o o p pl o b p o
o ▲ l l , o p o bl o
o p o o bo o pl v T o v p
o o o pp l , ply bo o o l y
S F 5-5
- M o , o b , o p o T p-o ly o b

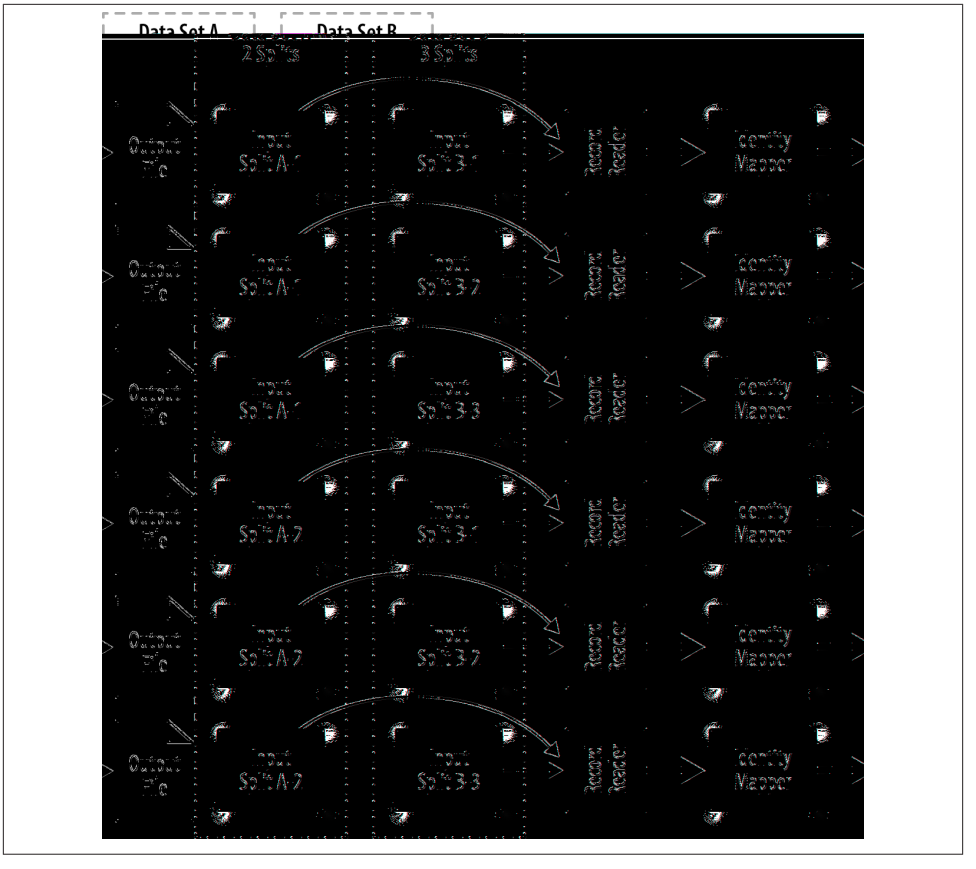


Figure 5-5. The structure of the Cartesian product pattern

Consequences

T l po pl q v l o b o p v y
po bl pl o b o o p o p lo p

Resemblances

SQL

▲l o v y ly , C p o y lly pl o ll
o SQ J l o l pl bl o where l

```
SELECT * FROM tablea, tableb;
```

Pig

p o C p o C SS l o o
b p v o p o o l b p ly

```
A = LOAD 'data1' AS (a1, a2, a3);
DUMP A;
(1,2,3)
(4,5,6)

B = LOAD 'data2' AS (b1, b2);
DUMP B;
(1,2)
(3,4)
(5,6)

C = CROSS A, B;

DUMP C;
(1,2,3,1,2)
(1,2,3,3,4)
(1,2,3,5,6)
(4,5,6,1,2)
(4,5,6,3,4)
(4,5,6,5,6)
```

Performance Analysis

T C p o p o v pb o , v l - o o ly llo o p o llo o o l b v y p ly b ll p y p b o v yb T ll lly o o ly , y p b by C p o bl byo o b l o pl o b o o q l o o l b o p b l , llo o o ’ o o q o

p pl p p v yo p pl — v ly o $O(n^2)$, n b b o by ▲ l o o l p pl , p pl b o o o o l p pl l p pl o o o , p pl o b o b o T v o o p o ! l l o o o , o l o b ! o y C p o bl , bl o o M p

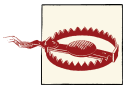
Cartesian Product Examples

Comment Comparison

T pl o o o p o l - o S v b o T l - o p p o o o l y o o o b o o o o b o y l o , p o p o l y Co o o o v o o b o p p o

T pl llo pl boo , p y p l o o o , o p o o C p o o p pl o Jo b o b p o o 11 p pl , o b o l o 1 1 p pl , b 1 1 p o o p o T o o p p o l C p o p p o pp o p o o pl by l o o l , p ll o o T o o l , o , o p p T p o o l o o o l

T o llb p o o o o pl o l o o p o bl o bl v oo o S v b o , p o o l b o b o l o b p



T o o oo p pl o o p o bl o v o 1 3 o o b o o p o b M p F v o o oo p M p ll v o l y p b o !

Input format code. T `CartesianImportFormat` p yb o l p o o `CompositeInputFormat` p v o pl o o p o o pl o ppo C p o o o o o o p p o o p o o pl ▲ l b bo l , o o pl D o b p, `getInputSplits` o p o o p pl o bo o l o `CompositeInputSplit` T o by ly p o o o pl , l l o p o T p pl o p o l o p o

```
public static class CartesianInputFormat extends FileInputFormat {

    public static final String LEFT_INPUT_FORMAT = "cart.left.inputformat";
    public static final String LEFT_INPUT_PATH = "cart.left.path";
```

```
public static final String RIGHT_INPUT_FORMAT = "cart.right.inputformat";
public static final String RIGHT_INPUT_PATH = "cart.right.path";

public static void setLeftInputInfo(JobConf job,
        Class<? extends FileInputFormat> inputFormat, String inputPath) {
    job.set(LEFT_INPUT_FORMAT, inputFormat.getCanonicalName());
    job.set(LEFT_INPUT_PATH, inputPath);
}

public static void setRightInputInfo(JobConf job,
        Class<? extends FileInputFormat> inputFormat, String inputPath) {
    job.set(RIGHT_INPUT_FORMAT, inputFormat.getCanonicalName());
    job.set(RIGHT_INPUT_PATH, inputPath);
}

public InputSplit[] getSplits(JobConf conf, int numSplits)
        throws IOException {
    // Get the input splits from both the left and right data sets
    InputSplit[] leftSplits = getInputSplits(conf,
            conf.get(LEFT_INPUT_FORMAT), conf.get(LEFT_INPUT_PATH),
                numSplits);
    InputSplit[] rightSplits = getInputSplits(conf,
            conf.get(RIGHT_INPUT_FORMAT), conf.get(RIGHT_INPUT_PATH),
                numSplits);

    // Create our CompositeInputSplits, size equal to
    // left.length * right.length
    CompositeInputSplit[] returnSplits =
            new CompositeInputSplit[leftSplits.length *
                rightSplits.length];

    int i = 0;
    // For each of the left input splits
    for (InputSplit left : leftSplits) {
            // For each of the right input splits
        for (InputSplit right : rightSplits) {
            // Create a new composite input split composing of the two
            returnSplits[i] = new CompositeInputSplit(2);
            returnSplits[i].add(left);
            returnSplits[i].add(right);
            ++i;
        }
    }

    // Return the composite splits
    LOG.info("Total splits to process: " + returnSplits.length);
    return returnSplits;
  }

public RecordReader getRecordReader(InputSplit split, JobConf conf,
        Reporter reporter) throws IOException {
        // Create a new instance of the Cartesian record reader
```

```
            return new CartesianRecordReader((CompositeInputSplit) split,
                    conf, reporter);
        }

        private InputSplit[] getInputSplits(JobConf conf,
                String inputFormatClass, String inputPath, int numSplits)
                throws ClassNotFoundException, IOException {
            // Create a new instance of the input format
            FileInputFormat inputFormat = (FileInputFormat) ReflectionUtils
                    .newInstance(Class.forName(inputFormatClass), conf);

            // Set the input path for the left data set
            inputFormat.setInputPaths(conf, inputPath);

            // Get the left input splits
            return inputFormat.getSplits(conf, numSplits);
        }
    }
```

Driver code. T v y p o CartesianInputFormat T p p bo o p o , p o o p o b p o o

```
    public static void main(String[] args) throws IOException,
            InterruptedException, ClassNotFoundException {

        // Configure the join type
        JobConf conf = new JobConf("Cartesian Product");
        conf.setJarByClass(CartesianProduct.class);

        conf.setMapperClass(CartesianMapper.class);
        conf.setNumReduceTasks(0);

        conf.setInputFormat(CartesianInputFormat.class);

        // Configure the input format
        CartesianInputFormat.setLeftInputInfo(conf, TextInputFormat.class, args[0]);
        CartesianInputFormat.setRightInputInfo(conf, TextInputFormat.class, args[0]);

        TextOutputFormat.setOutputPath(conf, new Path(args[1]));

        conf.setOutputKeyClass(Text.class);
        conf.setOutputValueClass(Text.class);

        RunningJob job = JobClient.runJob(conf);
        while (!job.isComplete()) {
            Thread.sleep(1000);
        }

        System.exit(job.isSuccessful() ? 0 : 1);
    }
```

Record reader code. T o pp o p o o p o D p, getRecordReader ll by o o CartesianRecordReader T o o o l o p o o b ,o o pl

T ll o next o o l o pp p y, o o pp p v l T y/ v l p v o pp o p o by o

S b q ll o next o o ll o o o , llb pp o p o , l y o o , l o - l llbo p b , o o o l T o , p o o

T p o o pl l l o l , o o y/v l p ▲ po , o v C p o o bo p pl o p

So o o pl o o o o o b v y, close() getPos() T lo o o p o o ppo o l b pl , o o o l o yo o o o o b l pl , l o o o ly o o bo o p o o v o 3 o o o , o ’ o p pl b o o p o o b

```
public static class CartesianRecordReader<K1, V1, K2, V2> implements
        RecordReader<Text, Text> {

    // Record readers to get key value pairs
    private RecordReader leftRR = null, rightRR = null;

    // Store configuration to re-create the right record reader
    private FileInputFormat rightFIF;
    private JobConf rightConf;
    private InputSplit rightIS;
    private Reporter rightReporter;

    // Helper variables
    private K1 lkey;
    private V1 lvalue;
    private K2 rkey;
    private V2 rvalue;
    private boolean goToNextLeft = true, alldone = false;
```

```
public CartesianRecordReader(CompositeInputSplit split, JobConf conf,
        Reporter reporter) throws IOException {
    this.rightConf = conf;
    this.rightIS = split.get(1);
    this.rightReporter = reporter;

    // Create left record reader
    FileInputFormat leftFIF = (FileInputFormat) ReflectionUtils
            .newInstance(Class.forName(conf
                    .get(CartesianInputFormat.LEFT_INPUT_FORMAT)), conf);

    leftRR = leftFIF.getRecordReader(split.get(0), conf, reporter);

    // Create right record reader
    rightFIF = (FileInputFormat) ReflectionUtils.newInstance(Class
            .forName(conf
                    .get(CartesianInputFormat.RIGHT_INPUT_FORMAT)), conf);

    rightRR = rightFIF.getRecordReader(rightIS, rightConf, rightReporter);

    // Create key value pairs for parsing
    lkey = (K1) this.leftRR.createKey();
    lvalue = (V1) this.leftRR.createValue();

    rkey = (K2) this.rightRR.createKey();
    rvalue = (V2) this.rightRR.createValue();
}

public boolean next(Text key, Text value) throws IOException {
    do {
        // If we are to go to the next left key/value pair
        if (goToNextLeft) {
            // Read the next key value pair, false means no more pairs
            if (!leftRR.next(lkey, lvalue)) {
                // If no more, then this task is nearly finished
                alldone = true;
                break;
            } else {
                // If we aren't done, set the value to the key and set
                // our flags
                key.set(lvalue.toString());
                goToNextLeft = alldone = false;

                // Reset the right record reader
                this.rightRR = this.rightFIF.getRecordReader(
                        this.rightIS, this.rightConf,
                        this.rightReporter);
            }
        }

        // Read the next key value pair from the right data set
        if (rightRR.next(rkey, rvalue)) {
```

```
                // If success, set the value
                value.set(rvalue.toString());
            } else {
                // Otherwise, this right data set is complete
                // and we should go to the next left pair
                goToNextLeft = true;
            }

            // This loop will continue if we finished reading key/value
            // pairs from the right data set
        } while (goToNextLeft);

        // Return true if a key/value pair was read, false otherwise
        return !alldone;
    }
}
```

Mapper code. T pp p o p o p Fo Texto b , o o o T o o y o o o b o o o , p o p o l y

```
public static class CartesianMapper extends MapReduceBase implements
        Mapper<Text, Text, Text, Text> {

    private Text outkey = new Text();

    public void map(Text key, Text value,
            OutputCollector<Text, Text> output, Reporter reporter)
            throws IOException {

        // If the two comments are not equal
        if (!key.toString().equals(value.toString())) {
            String[] leftTokens = key.toString().split("\\s");
            String[] rightTokens = value.toString().split("\\s");

            HashSet<String> leftSet = new HashSet<String>(
                    Arrays.asList(leftTokens));
            HashSet<String> rightSet = new HashSet<String>(
                    Arrays.asList(rightTokens));

            int sameWordCount = 0;
            StringBuilder words = new StringBuilder();
            for (String s : leftSet) {
                if (rightSet.contains(s)) {
                    words.append(s + ",");
                    ++sameWordCount;
                }
            }

            // If there are at least three words, output
```

```
                if (sameWordCount > 2) {
                    outkey.set(words + "\t" + key);
                    output.collect(outkey, value);
                }
            }
        }
    }
```

CHAPTER 6
Metapatterns

T p o o o ' o p o o lv p l p o bl , b p l p T *metapatterns* ly l o p bo p T o ll b *job chaining*, p o v l p o o lv o pl , l p o bl T o o *job merging*, o p o o p o v l ly M p o b, v ly ll l pl b o o

Job Chaining

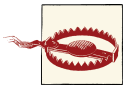
Jo b ly po o v o p o l pl o yo v o M y p o pl y ' o lv p o bl l M p o b So o b ll p ll l, o ll v o p o o o b , o o yo o o o o lv p o bl o M p o b , yo 'll b bl o l o l l o ll

Jo b o o o o pl p o o l b ' o o o bo o M p o Sy l oo p o l o M p o b v y ll, b l l o b lo o l o T o p o l o o o l l o o b l p o p o , pp o o o b ll b So ll o pp l o o yo p l v o , o p o o

▲ o pl o o oo l v o ll yo o lo o o b lo yo p y o pl , yo o l o o o T

pp o b o o l o b pl
o o b-by-o b b o , o p o ▲p p o , o l y o
b l o lo o o o b l o b o ly o o
y l o o p o lly o o p M p

p l o o p ll o M p o o ll o
b o b o y yo o o b o
o , bo o o p o o o
o p , by ll o M p o b M y , o v ,
o p o o b ll b p o q v ly o l o T
o y o o o lo o l y ▲ v
o b o pl ,o o po o o o b p pp



▲ o p o bl M p o po y
l o , y y b y, ll
o o o v p y o o y p o lo

o o b, b o yp lly p o
o o o y v o o yo ’
l o o p W , o o p l l ly o
po , v ll b lo T y o oo o
o p l bo o o blo o b l y
J pl y o b o p
o p o oo v l

T o o p o o o ly CombineFileInputFormat o
o b lo o p CombineFileInputFormat
ll blo l p o o l p pl
b o b p o by pp

With the Driver

o b bly pl o o p o o b o v v
ply o l pl o b- p v T ’ o p l bo M p
v oo p; ’ p y J v o ’ v o o o o
p l l o y

T v o M p o b ll q y o l
o ’ll v o p lly b o p p o o b p p
o o o b o by o po y p v bl

p o o o , po y o o l b l p o y o ’
l p o pl o o o b o pl p ly ll p yo
l q ly ▲l o , b lo o po y yo lly
b yo ’ll o yo l y o o

o p y ly po l pp o o lb
o o b J b o p o llo po y p o p o lly
l p o b y o o b

o l o o l pl o b p ll l by `Job.submit()` o `Job.wait ForCompletion()` T b o ly o
o b b o T llb yo o v l o b o `Job.is Complete()`, o blo o b o pl o , o o ly po ll o ll
o o b o pl

T o o p y o o o b ’ o oo o o o
o b o pl o l o o o o p y
o b l , yo o l b o o o y o l o

’ p yo bvo p o o o b l o
o o p o p v o b o o pl T
o l `JobControlo` o o

Job Chaining Examples

Basic job chaining

T o lo pl o o p l o lb o pl p o o
o p o o y po T o l b o
l M p o b, b l o o p o o bo v -
v b o po o b lb - v b W o o b o
p o o o o p o o b b o b
o po Fo p pl l o ,
o , b , pl o T l o p o o D,
b o y po , p o

T v b o po p l l b o o b
o ’ o T p `DistributedCache` o
o b o o p ’ p o T o
o o o pl o , l l v p
o o o b

T o lb p o o o o pl o l o o p o bl

o bl v o S v lb po , b b o y b lb
o bo v b o v po p ▲l o o o
p o o p o p

Job one mapper. o loo v , l ’ o pp
o bo o b T pp o D o o by

v l o OwnerUserId b o p y o o b, o o o v l l o o o byo T v l l v o l l v b o po p T AVERAGE_CALC_GROUP public static v l v l

```
    public static class UserIdCountMapper extends
            Mapper<Object, Text, Text, LongWritable> {

        public static final String RECORDS_COUNTER_NAME = "Records";

        private static final LongWritable ONE = new LongWritable(1);
        private Text outkey = new Text();

        public void map(Object key, Text value, Context context)
                throws IOException, InterruptedException {

            Map<String, String> parsed = MRDPUtils.transformXmlToMap(value
                    .toString());

            String userId = parsed.get("OwnerUserId");

            if (userId != null) {
                outkey.set(userId);
                context.write(outkey, ONE);
                context.getCounter(AVERAGE_CALC_GROUP,
                        RECORDS_COUNTER_NAME).increment(1);
            }
        }
    }
```

Job one reducer. T l o ly v l ply o p v l llo o 1 p , o p b p y ▲ o l o byo o o p, o o l l v

```
    public static class UserIdSumReducer extends
            Reducer<Text, LongWritable, Text, LongWritable> {

        public static final String USERS_COUNTER_NAME = "Users";
        private LongWritable outvalue = new LongWritable();

        public void reduce(Text key, Iterable<LongWritable> values,
                Context context) throws IOException, InterruptedException {

            // Increment user counter, as each reduce group represents one user
            context.getCounter(AVERAGE_CALC_GROUP, USERS_COUNTER_NAME).increment(1);

            int sum = 0;
            for (LongWritable value : values) {
                sum += value.get();
            }
```

```
            outvalue.set(sum);
            context.write(key, outvalue);
        }
    }
```

Job two mapper. T pp o o pl p vo o b o o o p T p p o pl T v b o po p p ll o Contexto b o b o o T MultipleOutputs l y l ll T o o p o b F lly, p o DistributedCache o b l po D o p o T p o o p

Co p o p, p o T p v l p o D b o po T o by ply pl o b o l o T pp o p y o D o p v l o b o po lo ’ p o , l by b T ’ b o po o p o v , b pp o p ly

▲ o p o l o p o MultipleOutputs.write pl o p l ▲ o o p y o y o b o y b lo o bo v v b o po T l o l o /part T b o b o l , o o ll pp -m-*nnnnn*, *nnnnn* D b W , o l ll b o bo b o l ll o b o part l T o o p /o p o pl o p ll l o b

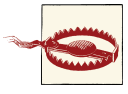
F lly, MultipleOutputs lo l p

```
public static class UserIdBinningMapper extends
        Mapper<Object, Text, Text, Text> {

    public static final String AVERAGE_POSTS_PER_USER = "avg.posts.per.user";

    public static void setAveragePostsPerUser(Job job, double avg) {
        job.getConfiguration().set(AVERAGE_POSTS_PER_USER,
                Double.toString(avg));
    }

    public static double getAveragePostsPerUser(Configuration conf) {
        return Double.parseDouble(conf.get(AVERAGE_POSTS_PER_USER));
    }

    private double average = 0.0;
    private MultipleOutputs<Text, Text> mos = null;
```

```
private Text outkey = new Text(), outvalue = new Text();
private HashMap<String, String> userIdToReputation =
        new HashMap<String, String>();

protected void setup(Context context) throws IOException,
        InterruptedException {
    average = getAveragePostsPerUser(context.getConfiguration());

    mos = new MultipleOutputs<Text, Text>(context);

    Path[] files = DistributedCache.getLocalCacheFiles(context
            .getConfiguration());

    // Read all files in the DistributedCache
    for (Path p : files) {
        BufferedReader rdr = new BufferedReader(
                new InputStreamReader(
                        new GZIPInputStream(new FileInputStream(
                                new File(p.toString())))));

        String line;
        // For each record in the user file
        while ((line = rdr.readLine()) != null) {
            // Get the user ID and reputation
            Map<String, String> parsed = MRDPUtils
                    .transformXmlToMap(line);
            // Map the user ID to the reputation
            userIdToReputation.put(parsed.get("Id"),
                    parsed.get("Reputation"));
        }
    }
}

public void map(Object key, Text value, Context context)
        throws IOException, InterruptedException {

    String[] tokens = value.toString().split("\t");

    String userId = tokens[0];
    int posts = Integer.parseInt(tokens[1]);

    outkey.set(userId);
    outvalue.set((long) posts + "\t" + userIdToReputation.get(userId));

    if ((double) posts < average) {
        mos.write(MULTIPLE_OUTPUTS_BELOW_NAME, outkey, outvalue,
                MULTIPLE_OUTPUTS_BELOW_NAME + "/part");
    } else {
        mos.write(MULTIPLE_OUTPUTS_ABOVE_NAME, outkey, outvalue,
                MULTIPLE_OUTPUTS_ABOVE_NAME + "/part");
    }
}
```

```
        protected void cleanup(Context context) throws IOException,
                InterruptedException {
            mos.close();
        }
    }
```

Driver code. N o l ' b o o o pl v b o o o o o o o o b o o b T o b by p o -l o p o p p o p o y ll b l by v o o b



▲ o o o o p o y o o p o y T o o p , b y b o o o o p o v o o y o o w o l o p o y l y o b b o , o b ll v

```
    public static void main(String[] args) throws Exception {

        Configuration conf = new Configuration();
        Path postInput = new Path(args[0]);
        Path userInput = new Path(args[1]);
        Path outputDirIntermediate = new Path(args[2] + "_int");
        Path outputDir = new Path(args[2]);

        // Setup first job to counter user posts
        Job countingJob = new Job(, "JobChaining-Counting");
        countingJob.setJarByClass(JobChainingDriver.class);

        // Set our mapper and reducer, we can use the API's long sum reducer for
        // a combiner!
        countingJob.setMapperClass(UserIdCountMapper.class);
        countingJob.setCombinerClass(LongSumReducer.class);
        countingJob.setReducerClass(UserIdSumReducer.class);

        countingJob.setOutputKeyClass(Text.class);
        countingJob.setOutputValueClass(LongWritable.class);

        countingJob.setInputFormatClass(TextInputFormat.class);

        TextInputFormat.addInputPath(countingJob, postInput);

        countingJob.setOutputFormatClass(TextOutputFormat.class);
        TextOutputFormat.setOutputPath(countingJob, outputDirIntermediate);
```

```
        // Execute job and grab exit code
        int code = countingJob.waitForCompletion(true) ? 0 : 1;

        ...
```

T o b o b o o o b T pl
o , b o o pl o b l l o y o o
o b o , b o v l o o b o v po
p T v l o o b o o W o pp o
bl p , p-o ly o b T o y p o p y o
o o o o MultipleOutputs DistributedCache T o b
o o v o

ly o po ly, o l , o p o y
l p T po o o v bo p v y

```
            MultipleOutputs.addNamedOutput(binningJob,
                    MULTIPLE_OUTPUTS_ABOVE_NAME, TextOutputFormat.class,
                    Text.class, Text.class);

            MultipleOutputs.setCountersEnabled(binningJob, true);

            TextOutputFormat.setOutputPath(binningJob, outputDir);

            // Add the user files to the DistributedCache
            FileStatus[] userFiles = FileSystem.get(conf).listStatus(userInput);
            for (FileStatus status : userFiles) {
                DistributedCache.addCacheFile(status.getPath().toUri(),
                        binningJob.getConfiguration());
            }

            // Execute job and grab exit code
            code = binningJob.waitForCompletion(true) ? 0 : 1;
        }

        // Clean up the intermediate output
        FileSystem.get(conf).delete(outputDirIntermediate, true);

        System.exit(code);
    }
```

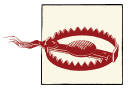
Parallel job chaining

T v p ll l o b l o p vo pl T o ly b o b b p ll l o o l o pl o T o o b pl p o v , y q p vo pl o v o pl lly T b o l l o b o v l o ly

T o lb p o o o o pl o l o o p o bl

o bl v p vo pl ' o p o b , p ll l o b o v bo b o l l v p o o

Mapper code. T pp pl p v l o y T o l o p o o p l T p o o p q y T y o ll p o o o p ll p o o o v l l o NullWritable b o o p ll o o , b y o v l v l



T b p v o v y l , o p
bl o ll y/v l p o v o
T b o v lly o o o
p pl p ll l o
bl b o o pp ' o p

```
public static class AverageReputationMapper extends
        Mapper<LongWritable, Text, Text, DoubleWritable> {

    private static final Text GROUP_ALL_KEY = new Text("Average Reputation:");
    private DoubleWritable outvalue = new DoubleWritable();

    protected void map(LongWritable key, Text value, Context context)
            throws IOException, InterruptedException {
        // Split the line into tokens
        String[] tokens = value.toString().split("\t");

        // Get the reputation from the third column
        double reputation = Double.parseDouble(tokens[2]);

        // Set the output value and write to context
        outvalue.set(reputation);
        context.write(GROUP_ALL_KEY, outvalue);
    }
}
```

Reducer code. T ply o p o v l ,
b p o T v l l o p p
y

```
public static class AverageReputationReducer extends
        Reducer<Text, DoubleWritable, Text, DoubleWritable> {

    private DoubleWritable outvalue = new DoubleWritable();

    protected void reduce(Text key, Iterable<DoubleWritable> values,
            Context context) throws IOException, InterruptedException {

        double sum = 0.0;
        double count = 0;
        for (DoubleWritable dw : values) {
            sum += dw.get();
            ++count;
        }

        outvalue.set(sum / count);
        context.write(key, outvalue);
    }
}
```

Driver code. T v o p o -l o p o p o o bo o b ▲ lp o ll o b o b o o , ll lbo T Jobo b o bo o o o b o pl o So lb o b ll , v o b o l p o v o bo o b o pl , y o o l pp o p lb p ▲ o b o o b

```
public static void main(String[] args) throws Exception {

    Configuration conf = new Configuration();

    Path belowAvgInputDir = new Path(args[0]);
    Path aboveAvgInputDir = new Path(args[1]);
    Path belowAvgOutputDir = new Path(args[2]);
    Path aboveAvgOutputDir = new Path(args[3]);

    Job belowAvgJob = submitJob(conf, belowAvgInputDir, belowAvgOutputDir);
    Job aboveAvgJob = submitJob(conf, aboveAvgInputDir, aboveAvgOutputDir);

    // While both jobs are not finished, sleep

    while (!belowAvgJob.isComplete() || !aboveAvgJob.isComplete()) {
        Thread.sleep(5000);
    }

    if (belowAvgJob.isSuccessful()) {
        System.out.println("Below average job completed successfully!");
    } else {
        System.out.println("Below average job failed!");
    }

    if (aboveAvgJob.isSuccessful()) {
        System.out.println("Above average job completed successfully!");
    } else {
        System.out.println("Above average job failed!");
    }

    System.exit(belowAvgJob.isSuccessful() &&
            aboveAvgJob.isSuccessful() ? 0 : 1);
}
```

T lp o o o b lbo v y o yo o o , p Job.submit Job.waitForCompletion T ll b o b ly , lb ppl o o o ▲ , Job o o main o l o pl o

```
private static Job submitJob(Configuration conf, Path inputDir,
        Path outputDir) throws Exception {
```

```
        Job job = new Job(conf, "ParallelJobs");
        job.setJarByClass(ParallelJobs.class);

        job.setMapperClass(AverageReputationMapper.class);
        job.setReducerClass(AverageReputationReducer.class);

        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(DoubleWritable.class);

        job.setInputFormatClass(TextInputFormat.class);
        TextInputFormat.addInputPath(job, inputDir);

        job.setOutputFormatClass(TextOutputFormat.class);
        TextOutputFormat.setOutputPath(job, outputDir);

        // Submit job and immediately return, rather than waiting for completion
        job.submit();
        return job;
    }
```

With Shell Scripting

T o o o b v y l o p v o pp o o pl
o pl o b b v o v l o b v , p
o ll p o b o p ly y yo o l
o o l o o ll p

T o b o pl o o b
o o b b b o v o o p l o b
v p l o J v T po o b p o o
l yo o ly b bl o lly o p l o b ▲l o , yo ’ll
b bl o o b v l y b p o o l o o o o -
l , b o p o b ll p
v , y , oo l o J v Fo pl , l
p ’ll po -p o o o p , y b v y l o o
sedo awk, b l l o o J v

o o o pp o y b o pl o o pl
o b b o b p ll l o o b b o
o , b y o b l J v

W pp y oo p M p o b p , b l
J v M p o b, o b, o v , b o b
T l po -p o , b , p p o , o l
b , o

l, ll p l o o b o b q ly
Fo o o b ppl o , y o o b l v -b
b oo p

Bash example

pl , ll o o b o b p ll l
o b pl T p b o o o p v bl o lly
o b ,

Bash script. p o p o v bl o b o bl
o T o o o bo o b , cat o p o ,
l p ll ly o p

```
#!/bin/bash

JAR_FILE="mrdp.jar"
JOB_CHAIN_CLASS="mrdp.ch6.JobChainingDriver"
PARALLEL_JOB_CLASS="mrdp.ch6.ParallelJobs"
HADOOP="$( which hadoop )"

POST_INPUT="posts"
USER_INPUT="users"
JOBCHAIN_OUTDIR="jobchainout"

BELOW_AVG_INPUT="${JOBCHAIN_OUTDIR}/belowavg"
ABOVE_AVG_INPUT="${JOBCHAIN_OUTDIR}/aboveavg"

BELOW_AVG_REP_OUTPUT="belowavgrep"
ABOVE_AVG_REP_OUTPUT="aboveavgrep"

JOB_1_CMD="${HADOOP} jar ${JAR_FILE} ${JOB_CHAIN_CLASS} ${POST_INPUT} \
    ${USER_INPUT} ${JOBCHAIN_OUTDIR}"
JOB_2_CMD="${HADOOP} jar ${JAR_FILE} ${PARALLEL_JOB_CLASS} ${BELOW_AVG_INPUT} \
    ${ABOVE_AVG_INPUT} ${BELOW_AVG_REP_OUTPUT} ${ABOVE_AVG_REP_OUTPUT}"

CAT_BELOW_OUTPUT_CMD="${HADOOP} fs -cat ${BELOW_AVG_REP_OUTPUT}/part-*"
CAT_ABOVE_OUTPUT_CMD="${HADOOP} fs -cat ${ABOVE_AVG_REP_OUTPUT}/part-*"

RMR_CMD="${HADOOP} fs -rmr ${JOBCHAIN_OUTDIR} ${BELOW_AVG_REP_OUTPUT} \
    ${ABOVE_AVG_REP_OUTPUT}"

LOG_FILE="avgrep_`date +%s`.txt"
```

T p o p echo o p o o
o b, o o l ,o p l
p pp , o o b o o

o o b o pl lly, o p o o b o b l ll o p l ▲ll o p o q , lo p o l o o lyo l , o b l o l , o p DFS

```
{
  echo ${JOB_1_CMD}
  ${JOB_1_CMD}

  if [ $? -ne 0 ]
  then
    echo "First job failed!"
    echo ${RMR_CMD}
    ${RMR_CMD}
    exit $?
  fi

  echo ${JOB_2_CMD}
  ${JOB_2_CMD}

  if [ $? -ne 0 ]
  then
    echo "Second job failed!"
    echo ${RMR_CMD}
    ${RMR_CMD}
    exit $?
  fi

  echo ${CAT_BELOW_OUTPUT_CMD}
  ${CAT_BELOW_OUTPUT_CMD}

  echo ${CAT_ABOVE_OUTPUT_CMD}
  ${CAT_ABOVE_OUTPUT_CMD}

  echo ${RMR_CMD}
  ${RMR_CMD}

  exit 0

} &> ${LOG_FILE}
```

Sample run. ▲ pl o p o lb T M p ly o p o o b v y

```
/home/mrdp/hadoop/bin/hadoop jar mrdp.jar mrdp.ch6.JobChainingDriver posts \
        users jobchainout
12/06/10 15:57:43 INFO input.FileInputFormat: Total input paths to process : 5
12/06/10 15:57:43 INFO util.NativeCodeLoader: Loaded the native-hadoop library
12/06/10 15:57:43 WARN snappy.LoadSnappy: Snappy native library not loaded
12/06/10 15:57:44 INFO mapred.JobClient: Running job: job_201206031928_0065
...
```

```
12/06/10 15:59:14 INFO mapred.JobClient: Job complete: job_201206031928_0065
...
12/06/10 15:59:15 INFO mapred.JobClient: Running job: job_201206031928_0066
...
12/06/10 16:02:02 INFO mapred.JobClient: Job complete: job_201206031928_0066

/home/mrdp/hadoop/bin/hadoop jar mrdp.jar mrdp.ch6.ParallelJobs \
        jobchainout/belowavg jobchainout/aboveavg belowavgrep aboveavgrep
12/06/10 16:02:08 INFO input.FileInputFormat: Total input paths to process : 1
12/06/10 16:02:08 INFO util.NativeCodeLoader: Loaded the native-hadoop library
12/06/10 16:02:08 WARN snappy.LoadSnappy: Snappy native library not loaded
12/06/10 16:02:12 INFO input.FileInputFormat: Total input paths to process : 1
Below average job completed successfully!
Above average job completed successfully!

/home/mrdp/hadoop/bin/hadoop fs -cat belowavgrep/part-*
Average Reputation: 275.36385831014724

/home/mrdp/hadoop/bin/hadoop fs -cat aboveavgrep/part-*
Average Reputation: 2375.301960784314

/home/mrdp/hadoop/bin/hadoop fs -rmr jobchainout belowavgrep aboveavgrep
Deleted hdfs://localhost:9000/user/mrdp/jobchainout
Deleted hdfs://localhost:9000/user/mrdp/belowavgrep
Deleted hdfs://localhost:9000/user/mrdp/aboveavgrep
```

With JobControl

T JobControl ControlledJob l p y o M p o b o l b bl o o o o b o lly y' y by l p JobCon trol yo o o b , b o b oo vy o pl ppl o

To JobControl, by pp yo o b ControlledJob Do l v ly pl yo yo o b l yo lly o l , p yo l o ControlledJob yo Jobo Configuration p , lb l o p o ControlledJobs T , yo o -by-o o JobControlo b , l

o ll v o p o po y l p o v o l

o yo o ’v o o v
o b o bo v o v Typ lly, o
p vo o ’ p T o o p o l
o v o o o o p o o po , -
l M p T l o o o p y
p l o M p

Job control example

Fo pl o v JobControl, l ’ o b p vo o pl
o b o b p ll l o b W l y l pp
o , o o o o v T v o p
o o b o o b o b o l o b,
JobControl o o b o p ll l o b
T l o b o v JobControl b yo o p o o l
o -b po o o o b o lp o o
o o o b ▲ll o b b o pl ly o b o
o b , b l

Main method. ’ bo o , p o l
ll p ll o ll o o b o W
p l v bl o o o b T o b
o v lp o po o pl o o o b,
w Configuration o lp o o ControlledJob
o b Configuration o pp , , o o
o b

T binningControlledJob o p ,o v y p vo
o b o pl lly T o o b p o b
ControlledJob T o o b ll o b by JobControl l b
o b o pl lly o ’ o pl lly, o o b o ’
b ll

▲ll ControlledJobs o JobControlo b , T
ll o JobControl.run ll bb l o po o b o pl W
l o b l o y o b l o o o ly
o p l p p o o

```
    public static final String AVERAGE_CALC_GROUP = "AverageCalculation";
    public static final String MULTIPLE_OUTPUTS_ABOVE_NAME = "aboveavg";
    public static final String MULTIPLE_OUTPUTS_BELOW_NAME = "belowavg";

    public static void main(String[] args) throws Exception {
        Configuration conf = new Configuration();
```

```
        Path postInput = new Path(args[0]);
        Path userInput = new Path(args[1]);
        Path countingOutput = new Path(args[3] + "_count");
        Path binningOutputRoot = new Path(args[3] + "_bins");
        Path binningOutputBelow = new Path(binningOutputRoot + "/"
                + JobChainingDriver.MULTIPLE_OUTPUTS_BELOW_NAME);
        Path binningOutputAbove = new Path(binningOutputRoot + "/"
                + JobChainingDriver.MULTIPLE_OUTPUTS_ABOVE_NAME);

        Path belowAverageRepOutput = new Path(args[2]);
        Path aboveAverageRepOutput = new Path(args[3]);

        Job countingJob = getCountingJob(conf, postInput, countingOutput);

        int code = 1;
        if (countingJob.waitForCompletion(true)) {
            ControlledJob binningControlledJob = new ControlledJob(
                    getBinningJobConf(countingJob, conf, countingOutput,
                            userInput, binningOutputRoot));

            ControlledJob belowAvgControlledJob = new ControlledJob(
                    getAverageJobConf(conf, binningOutputBelow,
                            belowAverageRepOutput));
            belowAvgControlledJob.addDependingJob(binningControlledJob);

            ControlledJob aboveAvgControlledJob = new ControlledJob(
                    getAverageJobConf(conf, binningOutputAbove,
                            aboveAverageRepOutput));
            aboveAvgControlledJob.addDependingJob(binningControlledJob);

            JobControl jc = new JobControl("AverageReputation");
            jc.addJob(binningControlledJob);
            jc.addJob(belowAvgControlledJob);
            jc.addJob(aboveAvgControlledJob);

            jc.run();
            code = jc.getFailedJobList().size() == 0 ? 0 : 1;
        }

        FileSystem fs = FileSystem.get(conf);
        fs.delete(countingOutput, true);
        fs.delete(binningOutputRoot, true);

        System.exit(code);
    }
```

Helper methods. Fo llb ll lp o o l Jobo Configurationo b ▲ ControlledJob b o l T p o , l o b o lp ll l o b T p o -

```
public static Job getCountingJob(Configuration conf, Path postInput,
        Path outputDirIntermediate) throws IOException {
    // Setup first job to counter user posts
    Job countingJob = new Job(conf, "JobChaining-Counting");
    countingJob.setJarByClass(JobChainingDriver.class);

    // Set our mapper and reducer, we can use the API's long sum reducer for
    // a combiner!
    countingJob.setMapperClass(UserIdCountMapper.class);
    countingJob.setCombinerClass(LongSumReducer.class);
    countingJob.setReducerClass(UserIdSumReducer.class);

    countingJob.setOutputKeyClass(Text.class);
    countingJob.setOutputValueClass(LongWritable.class);

    countingJob.setInputFormatClass(TextInputFormat.class);

    TextInputFormat.addInputPath(countingJob, postInput);

    countingJob.setOutputFormatClass(TextOutputFormat.class);
    TextOutputFormat.setOutputPath(countingJob, outputDirIntermediate);

    return countingJob;
}

public static Configuration getBinningJobConf(Job countingJob,
        Configuration conf, Path jobchainOutdir, Path userInput,
        Path binningOutput) throws IOException {
    // Calculate the average posts per user by getting counter values
    double numRecords = (double) countingJob
            .getCounters()
            .findCounter(JobChainingDriver.AVERAGE_CALC_GROUP,
                    UserIdCountMapper.RECORDS_COUNTER_NAME).getValue();
    double numUsers = (double) countingJob
            .getCounters()
            .findCounter(JobChainingDriver.AVERAGE_CALC_GROUP,
                    UserIdSumReducer.USERS_COUNTER_NAME).getValue();

    double averagePostsPerUser = numRecords / numUsers;

    // Setup binning job
    Job binningJob = new Job(conf, "JobChaining-Binning");
    binningJob.setJarByClass(JobChainingDriver.class);

    // Set mapper and the average posts per user
    binningJob.setMapperClass(UserIdBinningMapper.class);
    UserIdBinningMapper.setAveragePostsPerUser(binningJob,
            averagePostsPerUser);

    binningJob.setNumReduceTasks(0);

    binningJob.setInputFormatClass(TextInputFormat.class);
```

```
        TextInputFormat.addInputPath(binningJob, jobchainOutdir);

        // Add two named outputs for below/above average
        MultipleOutputs.addNamedOutput(binningJob,
                JobChainingDriver.MULTIPLE_OUTPUTS_BELOW_NAME,
                TextOutputFormat.class, Text.class, Text.class);

        MultipleOutputs.addNamedOutput(binningJob,
                JobChainingDriver.MULTIPLE_OUTPUTS_ABOVE_NAME,
                TextOutputFormat.class, Text.class, Text.class);
        MultipleOutputs.setCountersEnabled(binningJob, true);

        // Configure multiple outputs
        conf.setOutputFormat(NullOutputFormat.class);
        FileOutputFormat.setOutputPath(conf, outputDir);
        MultipleOutputs.addNamedOutput(conf, MULTIPLE_OUTPUTS_ABOVE_5000,
                TextOutputFormat.class, Text.class, LongWritable.class);
        MultipleOutputs.addNamedOutput(conf, MULTIPLE_OUTPUTS_BELOW_5000,
                TextOutputFormat.class, Text.class, LongWritable.class);

        // Add the user files to the DistributedCache
        FileStatus[] userFiles = FileSystem.get(conf).listStatus(userInput);
        for (FileStatus status : userFiles) {
            DistributedCache.addCacheFile(status.getPath().toUri(),
                    binningJob.getConfiguration());
        }

        // Execute job and grab exit code
        return binningJob.getConfiguration();
    }

    public static Configuration getAverageJobConf(Configuration conf,
            Path averageOutputDir, Path outputDir) throws IOException {

        Job averageJob = new Job(conf, "ParallelJobs");
        averageJob.setJarByClass(ParallelJobs.class);

        averageJob.setMapperClass(AverageReputationMapper.class);
        averageJob.setReducerClass(AverageReputationReducer.class);

        averageJob.setOutputKeyClass(Text.class);
        averageJob.setOutputValueClass(DoubleWritable.class);

        averageJob.setInputFormatClass(TextInputFormat.class);

        TextInputFormat.addInputPath(averageJob, averageOutputDir);

        averageJob.setOutputFormatClass(TextOutputFormat.class);
        TextOutputFormat.setOutputPath(averageJob, outputDir);
```

```
        // Execute job and grab exit code
        return averageJob.getConfiguration();
    }
```

Chain Folding

C o l o p o ppl o M p o b lly, l o b y o b b o l pl pp ,o o pp S o b p o o l v lo o l T o o b o bl b p p o pl ly - o lo o lo , o o ' lly o o o o o p o o W b l l M p , o l o o b p p ll v o p o b

T b o o l o o o v M p p p l , b / o lo o o ,o l o v o M p o b , po y o DFS, o b o , ' o l /

T b o p o lo o o o o o l

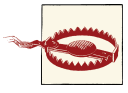
1 T lo p p l pl p p , o o p T o l b yo p-o ly o b pl o , o llo by l o p, o o ' Co o - o b o b p-o ly o b, o llo by o l M p o b p p p W o o p o , p-o ly o b ll o p o o b l y , ll b lo by o o b

, p p o p-o ly o b o l o b, po y v , / ly ▲l o , , o v o C y p o v o o p o , lly ' y o o o o v o po bly l l y o

o b p p o b o o , p p o b o T p l p o b p vo p o v / o po y o p-o ly o bo l o - po v

3 N o p p o o b o o p
o ▲ po bl , pl p p p o b o o
b o p o *decrease* o o , l o p o
increase o o , o , o po bl
b yo y o o o o l
, bo p p o l p l v ly o
o o p o o o
o p vo , l p po o o
y

T p b o o pl o b l T
yo p p-p po o p vo , yo
ll o o o po y o , ll o
o bo o o p o T b p y
o o l o



l p q bo o o y Fo
pl , v pl o o o b oo
b ll o l o y v l bl o
, b b o l v p

l o o b o o , y o l
ly po bl T o p v p o M p o b
yp lly p o p p l bo ,
l / o , o Fo pl , yo o ly bo
o 1 , l o p p , o
o p o

’ o o pl o pl o lp pl y o l

To pl y po , o F 6-1 T o l o o p
o p o pl o o l o pp o o M p
o b bo o

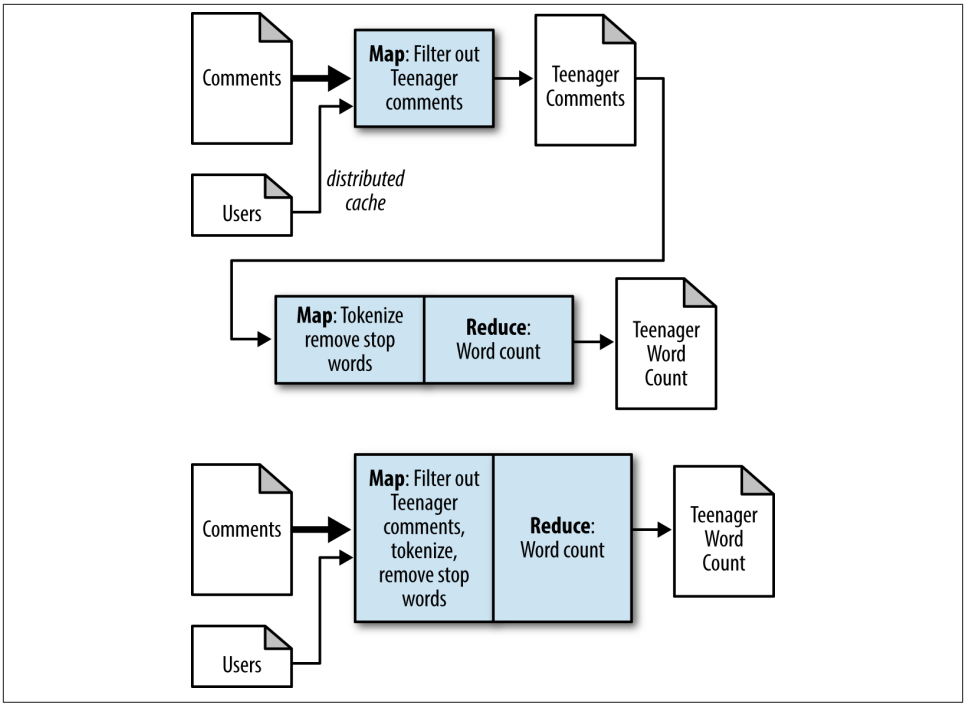


Figure 6-1. Original chain and optimizing mappers

T o b p o o o o o o W o o o o p o o yo T o ’ o , y o o o , p-o ly pl o b o p p o o o o b

To pl y p o , o o lb F 6- T o l o p o p o pl o o l o o o M p o b bo o

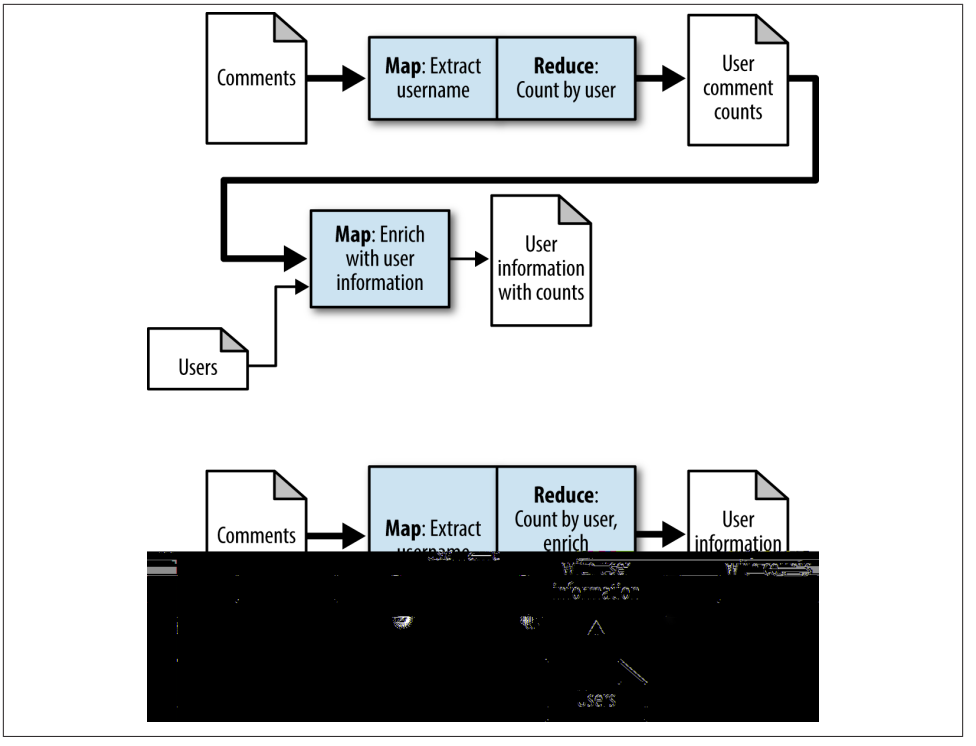
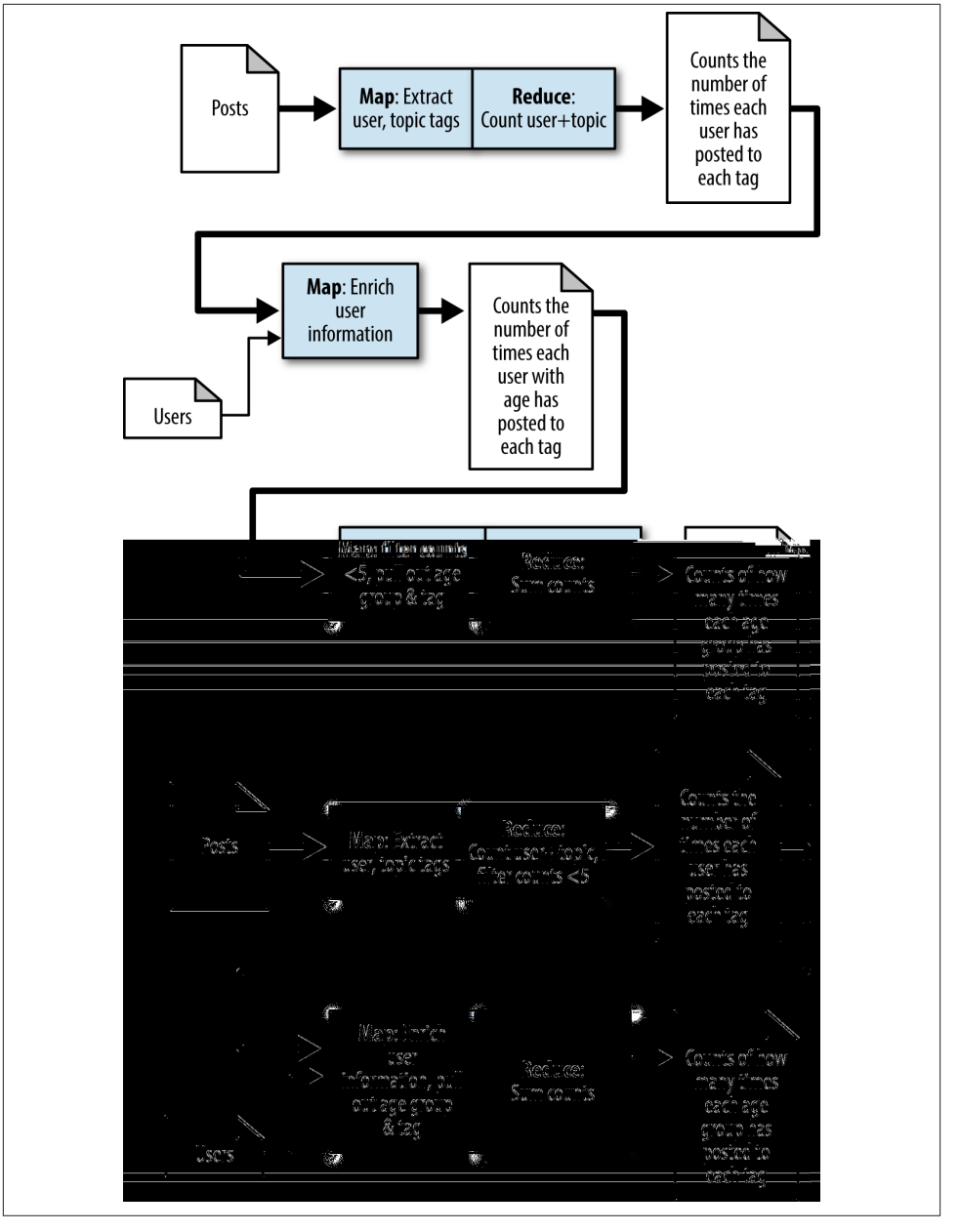


Figure 6-2. Original chain and optimizing a reducer with a mapper

T o b ' o o b o o po o M p o b, pl o o o o o o , p-o ly pl o b o

To pl y p , o o lb F 6-3 T o l o p o p o pl o o l o o o M p o b bo o



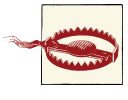
l 5, lly o p by o p p o l o
W bo p l , pl o
, l l o v Fo lb p , o o o v
l o M p o b, o v pl o o p
p o o M p o b T v b
bo o o F 6-3 o M p o b ll o ly l
b o o lb o M p o b b l

T o p y o o pl o l lly
p o o , o l pp o p l l ll
`ChainMapper` `ChainReducer` o - o b b lly l pl p
p , pl o o l pp o v lo p
p o , yo o l `ChainMapper`
`ChainReducer` pp o o o lb oo o p

The ChainMapper and ChainReducer Approach

`ChainMapper` `ChainReducer` p l pp l lb yo
o l pl p p pp l pl p p
o v ly p o l p p o v l p
p , o lb by p , o lb by v l p p o v ,o lyo
p p o p v wo

p p o p p l T o p o
p o by o , p o by , o o T p
p o b o o p o o o l
o p o T l o po -p o o p o o o l l



p yp o p yp b
p p o p `<LongWritable, Text>`, b
o p p `<LongWritable, Text>`

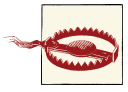
Chain Folding Example

Bin users by reputation

T pl l o o o o b pl , o
pp pl o o l p p T o p XM
o o D o o o T o pp
D o p o , p p v
`DistributedCache`

T o v l pp l o o l
T b LongSumReducer ply o ll v l
b T o p b p y

F lly, pp ll ll b o b o p
o b b o bo v 5, T b o M p o b
ChainMapper ChainReducer



T pl p mapred ▲ , b ChainMapper
ChainReducer o v l bl mapreduce p
pl

T o lb p o o o o pl o l o o p o bl
o bl v o po o o , b b o
p o b b o bo v 5,

Parsing mapper code. T pp pl o D o p po
o o p o o 1

```
public static class UserIdCountMapper extends MapReduceBase implements
        Mapper<Object, Text, Text, LongWritable> {

    public static final String RECORDS_COUNTER_NAME = "Records";
    private static final LongWritable ONE = new LongWritable(1);
    private Text outkey = new Text();

    public void map(Object key, Text value,
            OutputCollector<Text, LongWritable> output, Reporter reporter)
            throws IOException {

        Map<String, String> parsed = MRDPUtils.transformXmlToMap(value
                .toString());

        // Get the value for the OwnerUserId attribute
        outkey.set(parsed.get("OwnerUserId"));
        output.collect(outkey, ONE);
    }
}
```

Replicated join mapper code. T pp pl o o p o
p vo pp p p o po
D o p o T p ll o map o o p v l
’ p o T y o p b p v l

```
public static class UserIdReputationEnrichmentMapper extends MapReduceBase
        implements Mapper<Text, LongWritable, Text, LongWritable> {
```

```
        private Text outkey = new Text();
        private HashMap<String, String> userIdToReputation =
                new HashMap<String, String>();

        public void configure(JobConf job) {

            Path[] files = DistributedCache.getLocalCacheFiles(job);

            // Read all files in the DistributedCache
            for (Path p : files) {
                BufferedReader rdr = new BufferedReader(
                        new InputStreamReader(
                                new GZIPInputStream(new FileInputStream(
                                        new File(p.toString())))));

                String line;
                // For each record in the user file
                while ((line = rdr.readLine()) != null) {
                    // Get the user ID and reputation
                    Map<String, String> parsed = MRDPUtils
                            .transformXmlToMap(line);

                    // Map the user ID to the reputation
                    userIdToReputation.put(parsed.get("Id",
                            parsed.get("Reputation"));
                }
            }
        }

        public void map(Text key, LongWritable value,
                OutputCollector<Text, LongWritable> output, Reporter reporter)
                throws IOException {

            String reputation = userIdToReputation.get(key.toString());
            if (reputation != null) {
                outkey.set(value.get() + "\t" + reputation);
                output.collect(outkey, value);
            }
        }
    }
```

Reducer code. T pl o v l o o p o p y D p o

```
    public static class LongSumReducer extends MapReduceBase implements
            Reducer<Text, LongWritable, Text, LongWritable> {

        private LongWritable outvalue = new LongWritable();

            public void reduce(Text key, Iterator<LongWritable> values,
                    OutputCollector<Text, LongWritable> output, Reporter reporter)
                    throws IOException {
```

```
            int sum = 0;
            while (values.hasNext()) {
                  sum += values.next().get();
            }
            outvalue.set(sum);
            output.collect(key, outvalue);
        }
    }
```

Binning mapper code. T pp MultipleOutputs o b o o T p y p o p llo p o T p o v l o p o v l 5, o b pp o p ly

```
public static class UserIdBinningMapper extends MapReduceBase implements
        Mapper<Text, LongWritable, Text, LongWritable> {

    private MultipleOutputs mos = null;

    public void configure(JobConf conf) {
        mos = new MultipleOutputs(conf);
    }

    public void map(Text key, LongWritable value,
            OutputCollector<Text, LongWritable> output, Reporter reporter)
            throws IOException {

        if (Integer.parseInt(key.toString().split("\t")[1]) < 5000) {
            mos.getCollector(MULTIPLE_OUTPUTS_BELOW_5000, reporter)
                    .collect(key, value);
        } else {
            mos.getCollector(MULTIPLE_OUTPUTS_ABOVE_5000, reporter)
                    .collect(key, value);
        }
    }

    public void close() {
        mos.close();
    }
}
```

Driver code. T v l o o o ChainMapper ChainReducer T o p pp T o y o o pp pl o ChainMapper o o p pl o ll b ll b o b b o y o l o T , ChainReducer o o pl o , lly pp o N o yo o ’ ChainMapper o pp ChainReducer

T o o JobConfo pp / l , p
o p y v l p yp , o JobConf o pp / l T
b pp o o v l pp o o p
N o p l o o q , o ply p p y JobConfo b T
v p l o p v l by
o by v l T o p o yo o ll o o o o y
y o v l pp o , p o ,
o p o b by byValue = false

o o o pp , l o
o DistributedCache o o o pp p o W
l o o MultipleOutputs NullOutputFormat
yp l TextOutputFormat o o p o ll p v o o
l p y p l

```
public static void main(String[] args) throws Exception {
    JobConf conf = new JobConf("ChainMapperReducer");
    conf.setJarByClass(ChainMapperDriver.class);

    Path postInput = new Path(args[0]);
    Path userInput = new Path(args[1]);
    Path outputDir = new Path(args[2]);

    ChainMapper.addMapper(conf, UserIdCountMapper.class,
            LongWritable.class, Text.class, Text.class, LongWritable.class,
            false, new JobConf(false));

    ChainMapper.addMapper(conf, UserIdReputationEnrichmentMapper.class,
            Text.class, LongWritable.class, Text.class, LongWritable.class,
            false, new JobConf(false));

    ChainReducer.setReducer(conf, LongSumReducer.class, Text.class,
            LongWritable.class, Text.class, LongWritable.class, false,
            new JobConf(false));

    ChainReducer.addMapper(conf, UserIdBinningMapper.class, Text.class,
            LongWritable.class, Text.class, LongWritable.class, false,
            new JobConf(false));

    conf.setCombinerClass(LongSumReducer.class);

    conf.setInputFormat(TextInputFormat.class);
    TextInputFormat.setInputPaths(conf, postInput);

    // Configure multiple outputs
    conf.setOutputFormat(NullOutputFormat.class);
    FileOutputFormat.setOutputPath(conf, outputDir);
    MultipleOutputs.addNamedOutput(conf, MULTIPLE_OUTPUTS_ABOVE_5000,
            TextOutputFormat.class, Text.class, LongWritable.class);
    MultipleOutputs.addNamedOutput(conf, MULTIPLE_OUTPUTS_BELOW_5000,
```

```
        conf.setOutputKeyClass(Text.class);
        conf.setOutputValueClass(LongWritable.class);

        // Add the user files to the DistributedCache
        FileStatus[] userFiles = FileSystem.get(conf).listStatus(userInput);
        for (FileStatus status : userFiles) {
            DistributedCache.addCacheFile(status.getPath().toUri(), conf);
        }

        RunningJob job = JobClient.runJob(conf);

        while (!job.isComplete()) {
            Thread.sleep(5000);
        }

        System.exit(job.isSuccessful() ? 0 : 1);
    }
```

Job Merging

o b o l , *job merging* o o p o o o o / o M p p p l Jo b p o lb o l o b lb o M p p p l T b o o b lb p o ly o Fo o l - l o b , b o p v p o o l o p o o o o o -o -lb o o l o v o p o v o v , lly p p o p o pl , XM

▲ v o o b o o v v o o T o o b bo lb p , p o o p o W o b , ’ll v o M p o b lb lly p o o o b o o ppl o F 6-4 T o l o p o p o o pp o , o o bo o

N o pp yo o pply o b o o o o b o T o ! T o yo o o l b o o b , o o p o yo ’ll v v l bl yo l

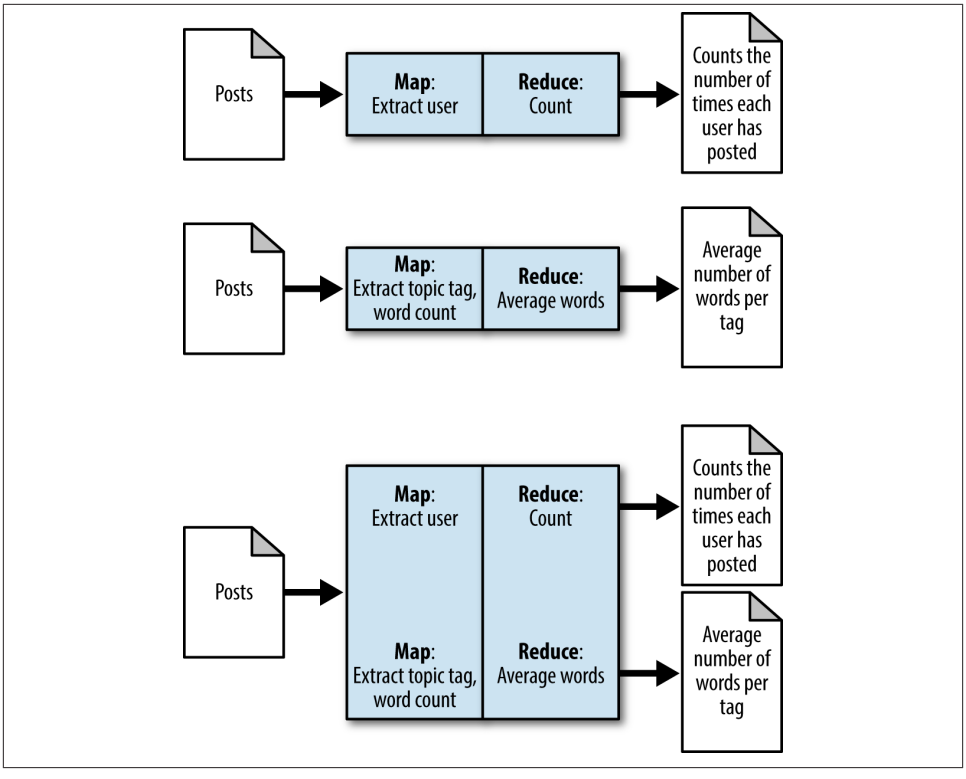


Figure 6-4. Original jobs and merged jobs

ly, p o ll b l v o ly o p o l y o b p o o l D v lo p o p o o o l o ly ll o l l o W o b , y'll v o o o o ll v o b p o T l ly o o o o b o o l v ly o v o

o ly, yo y b o p q b o pply p T o o bvo o bo o b o v y o p o , b y'll b p p l o yp S l o o po ly o p b ly p o bl , b b o o pl y

Jo b y p o So ll v o b o o o o , b ly o o b p o o l o o b l Fo o p p v , o pl o o o , b

l o b o o ▲ l v l, p o ll
o b p o o l o o l p o , l
o ll p o o o o o b o o y ll
o T p o o o b o lb

1 o o o pp o

T o pl o y o o Co py p o o , b y
o pl p o o o oo - o o lp
yo o p o T o o o p o o o lp
p o p o p o l o

pp , o y v l o y p
o

T y o p o l o o
p o ’ p T y o o p
o o l yp , yo ply
, o o yo o l p o ▲p o o
p, p o o o p

T l y o o o o po pl -l y o
p ly o o l T ly l yo o
, b b o o

3 , p o - o
o lly

▲ pp , yo o py p o o -
o v - llo o lp o T - o o l
p o o b o

4 MultipleOutputs o p o p o o b

MultipleOutputs p lo p o lp l lb yo o o
o l o o p o , o l o l M
o o p l y o o o l o MultipleOutputs, l
o p o o o l

Job Merging Examples

Anonymous comments and distinct users

T pl o b ▲ o y S v b o p 1 1 D
D p 68 o pl o p o v ,

o p v y o , l o
o y v o o o T o pp o o S
v lb l v , o o b o ll ly
o p o o T y, o b o lyo

T o lb p o o o o pl o l o o p o bl

o bl v o o , o y v o o
o D

TaggedText WritableComparable. ▲ o `WritableComparable`o b o
`Text` T l yo pl lb b o o b ,
v o p

T o b o p v b v bl o v bl
o l `String` pp o `Text` v l l o l by
o b T o o lb o T
`compareTo` o o b l o o p bl lb o
y M p o T o o q l y
y q l, o b o p v l ly
y o q l, v l o o p o
o by , by v l T yp o o p o ll p o
o p

```
A:100004122
A:120019879
D:10
D:22
D:23
```

```
public static class TaggedText implements WritableComparable<TaggedText> {

    private String tag = "";
    private Text text = new Text();

    public TaggedText() { }

    public void setTag(String tag) {
        this.tag = tag;
    }

    public String getTag() {
        return tag;
    }

    public void setText(Text text) {
        this.text.set(text);
    }

    public void setText(String text) {
```

```
            this.text.set(text);
        }

        public Text getText() {
            return text;
        }

        public void readFields(DataInput in) throws IOException {
            tag = in.readUTF();
            text.readFields(in);
        }

        public void write(DataOutput out) throws IOException {
            out.writeUTF(tag);
            text.write(out);
        }

        public int compareTo(TaggedText obj) {
            int compare = tag.compareTo(obj.getTag());
            if (compare == 0) {
                return text.compareTo(obj.getText());
            } else {
                return compare;
            }
        }

        public String toString() {
            return tag.toString() + ":" + text.toString();
        }
    }
```

Merged mapper code. T p o ply p p o o lp o , o p o p b v l o o p o context T p o l ly o o l p v pl o o bo o p Texto b y v l T y o v yp o y/v l p p p b T anonymizeMap o o y o o o p v l , distinctMap o b D o o o p y/v l p o o lp p o ▲ o o y o D o

lp o p p o , b p o l b o l map o , T l Map<String,String> b p o bo lp o ▲ y l l o p o l b v y b l b o l b pl !

```
public static class AnonymizeDistinctMergedMapper extends
              Mapper<Object, Text, TaggedText, Text> {

        private static final Text DISTINCT_OUT_VALUE = new Text();

        private Random rndm = new Random();
        private TaggedText anonymizeOutkey = new TaggedText(),
                distinctOutkey = new TaggedText();
        private Text anonymizeOutvalue = new Text();

        public void map(Object key, Text value, Context context)
                    throws IOException, InterruptedException {
              anonymizeMap(key, value, context);
              distinctMap(key, value, context);
        }

        private void anonymizeMap(Object key, Text value, Context context)
                    throws IOException, InterruptedException {

              Map<String, String> parsed = MRDPUtils.transformXmlToMap(value
                    .toString());

              if (parsed.size() > 0) {
                    StringBuilder bldr = new StringBuilder();
                    bldr.append("<row ");
                    for (Entry<String, String> entry : parsed.entrySet()) {

                          if (entry.getKey().equals("UserId")
                                  || entry.getKey().equals("Id")) {
                                // ignore these fields
                          } else if (entry.getKey().equals("CreationDate")) {
                              // Strip out the time, anything after the 'T'
                              // in the value
                                  bldr.append(entry.getKey()
                                          + "=\""
                                          + entry.getValue().substring(0,
                                                  entry.getValue().indexOf('T'))
                                                  + "\" ");
                          } else {
                                // Otherwise, output this.
                                bldr.append(entry.getKey() + "=\"" + entry.
                                                getValue() + "\" ");
                          }
                    }

                    bldr.append(">");
                    anonymizeOutkey.setTag("A");
                    anonymizeOutkey.setText(Integer.toString(rndm.nextInt()));
                    anonymizeOutvalue.set(bldr.toString());
                    context.write(anonymizeOutkey, anonymizeOutvalue);
              }
        }
```

```
        private void distinctMap(Object key, Text value, Context context)
                    throws IOException, InterruptedException {

                Map<String, String> parsed = MRDPUtils.transformXmlToMap(value
                        .toString());

                // Otherwise, set our output key to the user's id,
                // tagged with a "D"
                distinctOutkey.setTag("D");
                distinctOutkey.setText(parsed.get("UserId"));

                // Write the user's id with a null value
                context.write(distinctOutkey, DISTINCT_OUT_VALUE);
        }
}
```

Merged reducer code. T ’ ll o setup cleanup l o b o MultipleOutputs l y T reduce o o o 5 T7 o 8 yo

```
            if (key.getTag().equals("A")) {
                anonymizeReduce(key.getText(), values, context);
            } else {
                distinctReduce(key.getText(), values, context);
            }
        }

        private void anonymizeReduce(Text key, Iterable<Text> values,
                Context context) throws IOException, InterruptedException {

            for (Text value : values) {
                mos.write(MULTIPLE_OUTPUTS_ANONYMIZE, value,
                        NullWritable.get(), MULTIPLE_OUTPUTS_ANONYMIZE + "/part");
            }
        }

        private void distinctReduce(Text key, Iterable<Text> values,
                Context context) throws IOException, InterruptedException {
            mos.write(MULTIPLE_OUTPUTS_DISTINCT, key, NullWritable.get(),
                    MULTIPLE_OUTPUTS_DISTINCT + "/part");
        }

        protected void cleanup(Context context) throws IOException,
                InterruptedException {
            mos.close();
        }
    }
```

Driver code. T v o b o l y o v MultipleOutputs ▲ll b o o b o pp pl o

```
    public static void main(String[] args) throws Exception {

        // Configure the merged job
        Job job = new Job(new Configuration(), "MergedJob");
        job.setJarByClass(MergedJobDriver.class);

        job.setMapperClass(AnonymizeDistinctMergedMapper.class);
        job.setReducerClass(AnonymizeDistinctMergedReducer.class);
        job.setNumReduceTasks(10);

        TextInputFormat.setInputPaths(job, new Path(args[0]));
        TextOutputFormat.setOutputPath(job, new Path(args[1]));

        MultipleOutputs.addNamedOutput(job, MULTIPLE_OUTPUTS_ANONYMIZE,
                TextOutputFormat.class, Text.class, NullWritable.class);
        MultipleOutputs.addNamedOutput(job, MULTIPLE_OUTPUTS_DISTINCT,
                TextOutputFormat.class, Text.class, NullWritable.class);

        job.setOutputKeyClass(TaggedText.class);
```

```
        job.setOutputValueClass(Text.class);

        System.exit(job.waitForCompletion(true) ? 0 : 1);
    }
```

CHAPTER 7

Input and Output Patterns

p , ’ll b o o p o b bly o o o v lbo y o p o v v l o M p o p o p o ll o l y o lb o o y oo p M p o o o bo So yo p - o po o DFS p o o o l o ,o ly o o p o M p So b oo p p o l blb p pl o ’ o yo , o o InputFormato OutputFormat o o pl y

T p p l p *generating data*, *external source input*, *partition pruning* ▲ll p p p o p y p p o pl ly y o o b o p p C o p o y o b y l o o yo o lb

l p , oo p ll o l y o y yo o T o p p , *external source output*, o y o o oo p DFS J l o p o , o o p o p po p o l y o o o o

Customizing Input and Output in Hadoop

oo p lb yo o o y y lb o o o y o o o o o p o blb DFS o yb o o o , o o o pp p p T o l yo ’ll b pl y o o RecordReader InputFormat T o oo p M p o v y l y o o pp pl

o o p l o llb yo o o y y o b o y
OutputFormat RecordWriter

InputFormat

o o p l o p o o o b o o

1 V l p o o o o b ,
Spl p bb l o b l o yp InputSplit, o
o p o p o

3 C RecordReader pl o o b o y/v l p o
InputSplit T p o byo o pp

T o o o p o b l o FileInputFormat, o o p
l b TextInputFormat T p o v l p o o b
by llo p p T b lly pl p l b
o o l o l by , bb pp bo Fo pl ,
16 by l DFS ll p pl b by
0MB-64MB, 64MB-128MB 128MB-160MB p ll b lyo o
p pl , RecordReader pl o po bl o
y/v l p o o ll by b

Typ lly, RecordReader o l po b l yo bo , b
p pl bo y b y p o b bly ll o llo o bo
y Fo pl , TextInputFormat l LineRecordReader o
y/v l p o p o l o , p by l
T y b o by l o v l
o p o l v y l ly o
by o p pl ll b l p l , LineRecordRead
er ll p v o o o pl l T b
o o o bb o o o o o ,
o o D No o bb T ll l by
o FSDataInputStream l , lly o ’ v o l
y o l o bb

Do ’ b o o p pl bo yo o o , b o
o o ly o yo ’ pl o y !

C o p o o l o l -b p ▲ b yo
p p InputSplito b y/v l p , o
o o ,yo y o pp o M p
o b p ll l J b o p p pl p
y o v o b l y

T `InputFormat` b l o o b o

getSplits

T pl o o `getSplits` yp lly v `JobContext` o b o v o p `List` o `InputSplit` o b T p pl v o o yo o lb o o l , v l o o o T T o l p o p T o l o oo pl o v y o o o y y p o , b o o o - b o o b b o Jo bT

createRecordReader

T o o b - o pl o o `Record Reader`, 'll o l o ly Typ lly, ly , b o `initialize` o ll by o

RecordReader

T `RecordReader` b l y/v l p o v `InputSplit` W l `InputSplit` p by -o v o pl , `RecordReader` o o o p o by pp T y oo p M p o *schema on read* `RecordReader` , b o l lyo o pl o , b o p p o o b y o p o o `Writable` `Comparable` y `Writable` v l C o yp v y o o o p o , y o b -o y o p o o o pp

▲ `RecordReader` bo by p pl o y/v l p o o l -b p , by po o l `RecordReader` o l y/v l p T o l o p o T o bo ▲ o — o o pp v lb p o l o p W l l o v , o o bo o y o o pl o

Go o XM W l `TextInputFormat` o b l o , XM l yp lly o o l llb pl by yp lM p p y p p pl bo y, yo o pl o ▲ bo o o o , yo o o b o XM l ▲ o o p pl ,

o l b o o XM T ll llb
M p o o o v o o XM l , l o
pl y XM o ▲ y XM pp by o o o
XM l ll b by p p

T `RecordReader` b l b o o b o v

initialize
T o p ’ `InputSplit` `TaskAt`
`temptContext`, p p o Fo l -b p o ,
oo pl o o by po o l o b

getCurrentKey and getCurrentValue
T o by o o v y/v l p o
pl o o `Mapper` o o b by o
ll po bl !

nextKeyValue
o po o o `InputFormat` l , l y/
v l p `true` l o

getProgress
o po o o `InputFormat` l , o p o l o
by o o

close
T o by o o l p o o y/v l
p o p o

OutputFormat

S l ly o p o , oo p l o o p o o o b o o

1 V l o p o o o o b
C `RecordWriter` pl o ll o p o o b

l p o `FileInputFormat`, `FileOutputFormat` o o l -
b o p o o p o M p o b o DFS, y
l -b o p o o ▲ ll o lv o o yo T
l by oo p `TextOutputFormat`, o y/v l p o DFS
o o p o y b l
v l p l o o o p o y T `TextOutputFormat` lo v l
o p o y o o p o o M p o b

T TextOutputFormat LineRecordWriter o y/v l p o p o , p o p o o T l toString o o l y/v l p o p l DFS, l by b T b l l b v o b o o

▲ , l InputFormat, yo o o o o DFS ▲ b yo y/v l p o o o o J v , JD C b o o , yo M p o o p ll l b l J v yo o l l b o o o o y

T OutputFormat b l o b o o pl o

checkOutputSpecs

T o o v l o p p o o o b, o y o o l y p o o b b , o p o l b o v

getRecordWriter

T o RecordWriter pl o l y/v l p o o p , yp lly FileSystemo b

getOutputCommiter

T o p o o o b p l o , o po l o pl o , l p — lo o Fo l -b o p , FileOutputCommitter b o l ll vy l ll po yo p o o p o v lo p o o o p o y y

RecordWriter

T RecordWriter b l y/v l p o l y ,o o o p l RecordReader o p , o o o l p o v , o o l y b o p o o v ▲ yp b p o o ,b o v OutputFormat.getRecordWriter

T RecordWriter b l pl , o o ly o o

write

T o ll by o o y/v l p o b T pl o o o p v y o yo T pl ’ll o ll y/v l p o l - o y y/v l o l y

close

T o by o o y/v l p o
o T b o l y l l , o y o o
o o v ,o yo l p

Generating Data

Pattern Description

T *generating data* p b o b o
o o o , o ly p ll l

Intent

o o b o o

Motivation

T p o llo o boo o ’ b
W p , yo o b b l y

’ o o Typ lly yo ’ll b o o
o v o v o v , yo o o , M p
ll y o o

T o o o o p o l o
o o p v o l b l o l l o ll
ll oo ll l o b l o b l o y o o
p oo o o p o ly l

o l o o p o b ,
o o ly T /T So DFS

o ly, pl o o p ’ o oo p b
o o o o l p o o o p o
p pl o p o ll o o o , o
p pl o o , o v o oo l o o

Structure

To pl p oo p, pl o `InputFormat` l
`RecordReader` o T p o o pl lyo bl vo o

o o , o b b l o ly o b b o o o l DFS Fo o p , y pp , b yo o o o pp -p o p , o v ly y S F 7-1

T p p-o ly

- T `InputFormat` pl o o T b o pl o l b o bl
- T `RecordReader` pl o o o o , yo o o o p pl o ll o o Fo pl , o o / , v p pl o o o
- o , `IdentityMapper` o o o

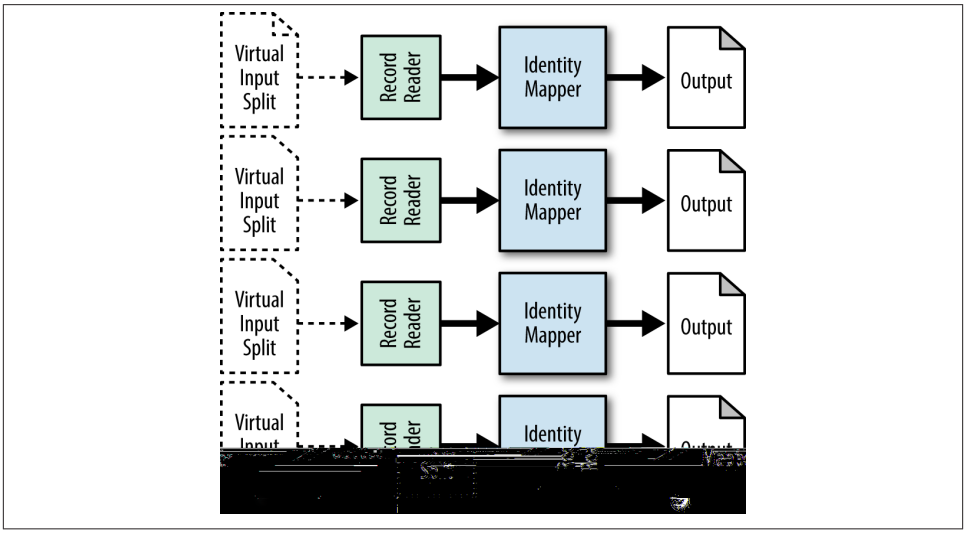


Figure 7-1. The structure of the generating data pattern

T l y yo o pl p o o b y p l o l bo o T , yo `InputFormat` `RecordReader` p o T p y p l l o ppl o

Consequences

pp o p l o o

Resemblances

T b o y o o SQ , b o
b q o

Performance analysis

T o o o o p o o y o p
o l, o p yo v , yo
yo b l p ll l o l o v ,
l l o p o p yo v p b y ll
o

Generating Data Examples

Generating random StackOverflow comments

To o S v b , ’ll l o 1, o
o bl b W l o v o o o , o o D
o l ly o ’ b q , o D, o o

T o llb p o o o o pl o l o o p o bl

Driver code. T v p o o l o o o b
o o p o ll o o o ▲ll
o p o v o p o y T y pp o
o b, p bl by b o o o

```
public static void main(String[] args) throws Exception {
    Configuration conf = new Configuration();

    int numMapTasks = Integer.parseInt(args[0]);
    int numRecordsPerTask = Integer.parseInt(args[1]);
    Path wordList = new Path(args[2]);
    Path outputDir = new Path(args[3]);

    Job job = new Job(conf, "RandomDataGenerationDriver");
    job.setJarByClass(RandomDataGenerationDriver.class);

    job.setNumReduceTasks(0);

    job.setInputFormatClass(RandomStackOverflowInputFormat.class);

    RandomStackOverflowInputFormat.setNumMapTasks(job, numMapTasks);
    RandomStackOverflowInputFormat.setNumRecordPerTask(job,
```

```
                numRecordsPerTask);
        RandomStackOverflowInputFormat.setRandomWordList(job, wordList);

        TextOutputFormat.setOutputPath(job, outputDir);

        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(NullWritable.class);

        System.exit(job.waitForCompletion(true) ? 0 : 2);
    }
```

InputSplit code. T FakeInputSplit l ply InputSplit pl Writable T o pl o o yo o v o ,o o o q v l b v l T p pl o o o o o o

```
    public static class FakeInputSplit extends InputSplit implements
            Writable {

        public void readFields(DataInput arg0) throws IOException {
        }

        public void write(DataOutput arg0) throws IOException {
        }

        public long getLength() throws IOException, InterruptedException {
            return 0;
        }

        public String[] getLocations() throws IOException,
                InterruptedException {
            return new String[0];
        }
    }
```

InputFormat code. T p o o p po l o p pl o o o p o , RandomStackOverflowRecordReader o p W o v getSplits o o o b o FakeInputSplit pl T b p ll o o o W o ll createRecordReader, RandomStackOverflowRecordReader , l ,

```
    public static class RandomStackOverflowInputFormat extends
            InputFormat<Text, NullWritable> {

        public static final String NUM_MAP_TASKS = "random.generator.map.tasks";
        public static final String NUM_RECORDS_PER_TASK =
                "random.generator.num.records.per.map.task";
        public static final String RANDOM_WORD_LIST =
                "random.generator.random.word.file";
```

```
    public List<InputSplit> getSplits(JobContext job) throws IOException {

        // Get the number of map tasks configured for
        int numSplits = job.getConfiguration().getInt(NUM_MAP_TASKS, -1);

        // Create a number of input splits equivalent to the number of tasks
        ArrayList<InputSplit> splits = new ArrayList<InputSplit>();
        for (int i = 0; i < numSplits; ++i) {
            splits.add(new FakeInputSplit());
        }

        return splits;
    }

    public RecordReader<Text, NullWritable> createRecordReader(
            InputSplit split, TaskAttemptContext context)
            throws IOException, InterruptedException {
        // Create a new RandomStackOverflowRecordReader and initialize it
        RandomStackOverflowRecordReader rr =
                new RandomStackOverflowRecordReader();
        rr.initialize(split, context);
        return rr;
    }

    public static void setNumMapTasks(Job job, int i) {
        job.getConfiguration().setInt(NUM_MAP_TASKS, i);
    }

    public static void setNumRecordPerTask(Job job, int i) {
        job.getConfiguration().setInt(NUM_RECORDS_PER_TASK, i);
    }

    public static void setRandomWordList(Job job, Path file) {
        DistributedCache.addCacheFile(file.toUri(), job.getConfiguration());
    }
}
```

RecordReader code. T o lly v o FakeInputSplit l o , b ply o T b o o o p ll o o b o o , l o o o o DistributedCache Fo ll o nextKeyValue, o o pl o b o T bo yo o by lp o o ly l o o l , b o y o l o o T o o p o o y o v b ll o , o false, l o o o p o pp

```
    public static class RandomStackOverflowRecordReader extends
            RecordReader<Text, NullWritable> {
```

```
private int numRecordsToCreate = 0;
private int createdRecords = 0;
private Text key = new Text();
private NullWritable value = NullWritable.get();
private Random rndm = new Random();
private ArrayList<String> randomWords = new ArrayList<String>();

// This object will format the creation date string into a Date
// object
private SimpleDateFormat frmt = new SimpleDateFormat(
        "yyyy-MM-dd'T'HH:mm:ss.SSS");

public void initialize(InputSplit split, TaskAttemptContext context)
        throws IOException, InterruptedException {

    // Get the number of records to create from the configuration
    this.numRecordsToCreate = context.getConfiguration().getInt(
            NUM_RECORDS_PER_TASK, -1);

    // Get the list of random words from the DistributedCache
    URI[] files = DistributedCache.getCacheFiles(context
            .getConfiguration());

    // Read the list of random words into a list
    BufferedReader rdr = new BufferedReader(new FileReader(
            files[0].toString()));

    String line;
    while ((line = rdr.readLine()) != null) {
        randomWords.add(line);
    }
    rdr.close();
}

public boolean nextKeyValue() throws IOException,
        InterruptedException {
    // If we still have records to create
    if (createdRecords < numRecordsToCreate) {
        // Generate random data
        int score = Math.abs(rndm.nextInt()) % 15000;
        int rowId = Math.abs(rndm.nextInt()) % 1000000000;
        int postId = Math.abs(rndm.nextInt()) % 100000000;
        int userId = Math.abs(rndm.nextInt()) % 1000000;
        String creationDate = frmt
                .format(Math.abs(rndm.nextLong()));

        // Create a string of text from the random words
        String text = getRandomText();

        String randomRecord = "<row Id=\"" + rowId + "\" PostId=\""
                + postId + "\" Score=\"" + score + "\" Text=\""
```

```
                    + text + "\" CreationDate=\"" + creationDate
                    + "\" UserId\"=" + userId + "\" />";

            key.set(randomRecord);
            ++createdRecords;
            return true;
        } else {
            // We are done creating records
            return false;
        }
    }

    private String getRandomText() {
        StringBuilder bldr = new StringBuilder();
        int numWords = Math.abs(rndm.nextInt()) % 30 + 1;

        for (int i = 0; i < numWords; ++i) {
            bldr.append(randomWords.get(Math.abs(rndm.nextInt())
                    % randomWords.size())
                    + " ");
        }
        return bldr.toString();
    }

    public Text getCurrentKey() throws IOException,
            InterruptedException {
        return key;
    }

    public NullWritable getCurrentValue() throws IOException,
            InterruptedException {
        return value;
    }

    public float getProgress() throws IOException, InterruptedException {
        return (float) createdRecords / (float) numRecordsToCreate;
    }

    public void close() throws IOException {
        // nothing to do here...
    }
}
```

External Source Output

Pattern Description

▲ l p , l o o p p o y
o o oo p DFS

Intent

o o M p o p o o v lb o

Motivation

W p , bl oo p o M p o ly
o l o T ly l o lb o y
o o b l v o l o T p p o
l y ly o p y/v l p ly y b lb
M p ly v o ppl o - , o M p o b l
lb o l o p ll l

M p pp o , o p ll l ▲ l
o o p , yo o b o y l p ll l
bo o ll o p o o

Structure

F 7- o l o o p , pl b lb

- T `OutputFormat` v o p p o o o b o o p o
o o b b o T pl o l o lly
o l, o ’ b oo o p o ll o ly o o l
o bl o o T o l o
po bl o l `RecordWriter` pl o
- T `RecordWriter` ll y/v l p o l o M l
`RecordReader`, pl o v p o l o
b o D o o o o b , bl y o
o l o ’ ▲ T o o o o
ll o po

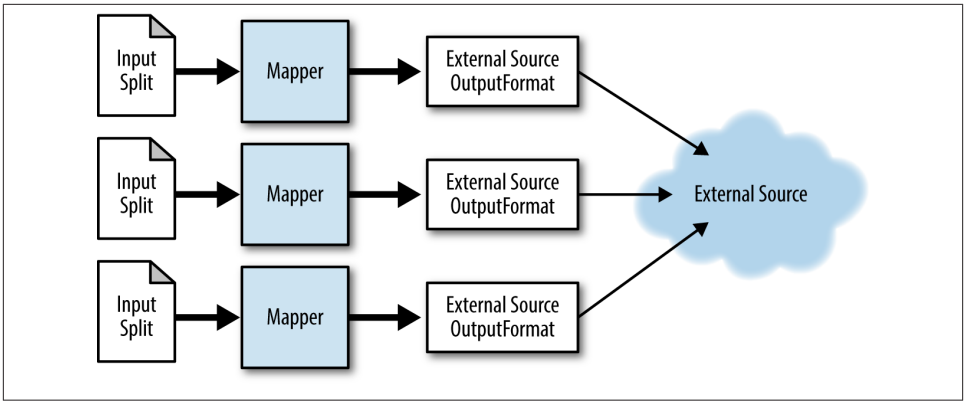
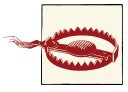


Figure 7-2. The structure of the external source output pattern

Consequences

T o p b o l o l o lly



N o l bo o pp , y o , y y/ v l p write o ' b v yp l M p o b, po yo p o l y v o l , o p ply W o l o ly, ll v l , l o o ' o lly o bo ll v o p bl , o o OutputCommitter o po yo p o l y T po y o p b , l v o l o , l po ,o l o l y o v o l

Performance analysis

F o M p p p v , ' o o y bo p o v , yo o v o b v y l v o l p ll l o o v o o l SQ b o o o o ll To wo , yo y v o v l b o yo yp lly o l o b o p ll l o T o ly p o bl o o p ll l ppo p ll l o Fo pl , o o SQ b , yo o l v o p b

External Source Output Example

Writing to Redis instances

T pl b o o b o p ll l o
M p o p - o , - o y, y-v l o o o
v , y o , , l , , o
▲N S C o o S X y , ,
o y l p

o o o oo p o , J o o
J o p - o bl ly ll v l ▲ l o l
o o l v l bl o b

l o pl boo , o l ly pl b
o pl p o o o o o
DFS o l o o `FileOutputFormat`
pl , S v b o o bl b o
, p lly - o - p o pp T pp
o ly b v ly o l

▲ p b l v l , l o J v `Hash`
`Map` v y o y v y o o o
b llo l -v l p

T o b b o po o pl o llb p o bl

o bl v o o o , o ly b - o - p o
pp o o bl b o p ll l

OutputFormat code. T `RedisHashOutputFormat` po bl o bl
v y o b o o p o o b b o Jo bT o b
b b , l o `RecordWriter` o l ll o p y/v l
p Typ lly, l DFS o v , o bo o DFS,
ll `RecordWriter` l o

T o p o o o o v bl b by v o
ll o o q o o o b , v o pl `public`
`static` o o o o o o o v b p o
T o p o l o o CSV
y o ll o p o `checkOutputSpecs` o ,
bo o p b o v bo l o b, ll ly
l o T yo ’ll o v y yo o o !

T getRecordWriter o o b o o Re
cordWriter o po , o o v bl q
by RedisHashRecordWriter o T o
l o RedisHashOutputFormat, o q b o o
o v o T l o l o lb o

T l o o o p o getOutputCommitter T o p o
by o o y po yo p b o o
l o b Fo pl o , o ’ yp lly
l o b - ▲ b o b
o y ▲ o p o q by o , b NullOutputFor
mat o o p o pl o o ’ o y

```
public static class RedisHashOutputFormat extends OutputFormat<Text, Text> {

    public static final String REDIS_HOSTS_CONF =
            "mapred.redishashoutputformat.hosts";
    public static final String REDIS_HASH_KEY_CONF =
            "mapred.redishashinputformat.key";

    public static void setRedisHosts(Job job, String hosts) {
        job.getConfiguration().set(REDIS_HOSTS_CONF, hosts);
    }

    public static void setRedisHashKey(Job job, String hashKey) {
        job.getConfiguration().set(REDIS_HASH_KEY_CONF, hashKey);
    }

    public RecordWriter<Text, Text> getRecordWriter(TaskAttemptContext job)
            throws IOException, InterruptedException {
        return new RedisHashRecordWriter(job.getConfiguration().get(
                REDIS_HASH_KEY_CONF), job.getConfiguration().get(
                REDIS_HOSTS_CONF));
    }

    public void checkOutputSpecs(JobContext job) throws IOException {
        String hosts = job.getConfiguration().get(REDIS_HOSTS_CONF);
        if (hosts == null || hosts.isEmpty()) {
            throw new IOException(REDIS_HOSTS_CONF
                    + " is not set in configuration.");
        }

        String hashKey = job.getConfiguration().get(
                REDIS_HASH_KEY_CONF);
        if (hashKey == null || hashKey.isEmpty()) {
            throw new IOException(REDIS_HASH_KEY_CONF
                    + " is not set in configuration.");
        }
    }

    public OutputCommitter getOutputCommitter(TaskAttemptContext context)
```

```
            throws IOException, InterruptedException {
        return (new NullOutputFormat<Text, Text>()).getOutputCommitter(context);
    }

    public static class RedisHashRecordWriter extends RecordWriter<Text, Text> {
        // code in next section
    }
}
```

RecordReader code. T RedisHashRecordWriter l o o v J l o y/v l p o ly o , p o v v b o o ll o ll T o o o y o o J

T o o o J p o T p write o o J T o y o b b o o T y/v l p o J o o F lly, ll J o close o

```
public static class RedisHashRecordWriter extends RecordWriter<Text, Text> {

    private HashMap<Integer, Jedis> jedisMap = new HashMap<Integer, Jedis>();
    private String hashKey = null;

    public RedisHashRecordWriter(String hashKey, String hosts) {
        this.hashKey = hashKey;

        // Create a connection to Redis for each host
        // Map an integer 0-(numRedisInstances - 1) to the instance
        int i = 0;
        for (String host : hosts.split(",")) {
            Jedis jedis = new Jedis(host);
            jedis.connect();
            jedisMap.put(i, jedis);
            ++i;
        }
    }

    public void write(Text key, Text value) throws IOException,
            InterruptedException {
        // Get the Jedis instance that this key/value pair will be
        // written to
        Jedis j = jedisMap.get(Math.abs(key.hashCode()) % jedisMap.size());

        // Write the key/value pair
        j.hset(hashKey, key.toString(), value.toString());
    }

    public void close(TaskAttemptContext context) throws IOException,
            InterruptedException {
```

```
            // For each jedis instance, disconnect it
            for (Jedis jedis : jedisMap.values()) {
                jedis.disconnect();
            }
        }
    }
```

Mapper Code. T M pp v y o bo l yo pp T D p o v o o p T o p o o ll vy l o , lb o b l pl o v o

```
    public static class RedisOutputMapper extends
            Mapper<Object, Text, Text, Text> {

        private Text outkey = new Text();
        private Text outvalue = new Text();

        public void map(Object key, Text value, Context context)
                throws IOException, InterruptedException {

            Map<String, String> parsed = MRDPUtils.transformXmlToMap(value
                    .toString());

            String userId = parsed.get("Id");
            String reputation = parsed.get("Reputation");

            // Set our output key and values
            outkey.set(userId);
            outvalue.set(reputation);

            context.write(outkey, outvalue);
        }
    }
```

Driver Code. T v o p o l ll o public static o o p o T o b b l yo

```
    public static void main(String[] args) throws Exception {
        Configuration conf = new Configuration();

        Path inputPath = new Path(args[0]);
        String hosts = args[1];
        String hashName = args[2];

        Job job = new Job(conf, "Redis Output");
        job.setJarByClass(RedisOutputDriver.class);

        job.setMapperClass(RedisOutputMapper.class);
        job.setNumReduceTasks(0);
```

```
        job.setInputFormatClass(TextInputFormat.class);
        TextInputFormat.setInputPaths(job, inputPath);

        job.setOutputFormatClass(RedisHashOutputFormat.class);
        RedisHashOutputFormat.setRedisHosts(job, hosts);
        RedisHashOutputFormat.setRedisHashKey(job, hashName);

        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(Text.class);

        int code = job.waitForCompletion(true) ? 0 : 2;

        System.exit(code);
    }
```

External Source Input

Pattern Description

T *external source input* p o ’ b o DFS, b o o y o o o o p, SQ b o b v

Intent

o o b p ll l o o o p o yo M p o

Motivation

T yp l o l o M p o ly yo o o o yo o pl o , DFS , ly W p , yo oo p M p o o l o , b o b v , p ll ly o pp

T o y yo o ly ly o o o yb o b o o o o oo p o v o o l Fo pl , p b o l y l ly o b p v o p o , o b ly M p o b o p- o - v l bl ▲ b pp o b y l , p b p o o ly l o l, ll p p l

M p pp o , b p ll l l o T v o o o v ll- bo o p ll l o o l Fo pl , o b , p b o b o bl , lb o v y q p ll l b o o q b

Structure

F 7-3 o l o p

- T InputFormat ll InputSplit o b , y b b o o o b ▲ p pl o b l p , l ly p o o ll b p , p o o l -b p b l o T p o l b o o SQ bl o b o b v p o l ▲ b p b p ll l, oo o M p
- T InputSplit o ll o l o o o o o o b T o b o o o o lp o p ▲ o InputSplit l o pl Writable , b o o o o p pl o o o T T T b o p b o T T q v l o b o p pl by p o T InputSplit o l RecordReader o p o
- T RecordReader o b o o p o v InputSplit o o o y/v l p T pl o o l p o o b p y o o q o o l o , JD C o b o b o ST ll o ST l v

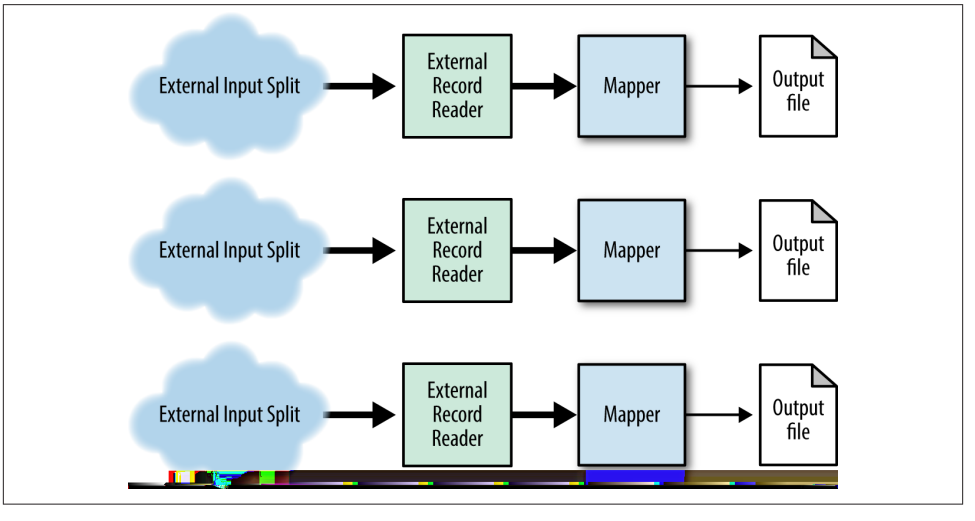


Figure 7-3. The structure of the external source input pattern

Consequences

D b o l o o M p o b p p o ’ o o o

Performance analysis

T bo l o M p o b pl p o o b o o o T o y o l ll l pl o o , l - SQ b ’ o o l 1, pp ll bb o ▲ o p o bl y b o v o p o b bly o M p l ’ o b pl , o o y b o o l o o o b p bl o T o l o b p o bl o l

External Source Input Example

Reading from Redis Instances

T pl o o o o ▲ , CSV l o o o o o o ll o S b o b o , b p ll l ▲ll o o p o , o o , y/v l p o o ll v T pl y pp o plyo p y/v l p v o

T o b b o po o pl o lb p o bl o bl v l o CSV o , ll o o p ll l

InputSplit code. T `RedisInputSplit` p o b p o by v l M pp pl , o o b o o p pl , ll y T p pl pl `Writable` , o l bl by o , l l o o o o o o v l o W b o v `getLocations` o , o p b bT ll p o T T o

```
public static class RedisHashInputSplit extends InputSplit implements Writable {

    private String location = null;
    private String hashKey = null;

    public RedisHashInputSplit() {
        // Default constructor for reflection
    }
```

```
        public RedisHashInputSplit(String redisHost, String hash) {
            this.location = redisHost;
            this.hashKey = hash;
        }

        public String getHashKey() {
            return this.hashKey;
        }

        public void readFields(DataInput in) throws IOException {
            this.location = in.readUTF();
            this.hashKey = in.readUTF();
        }

        public void write(DataOutput out) throws IOException {
            out.writeUTF(location);
            out.writeUTF(hashKey);
        }

        public long getLength() throws IOException, InterruptedException {
            return 0;
        }

        public String[] getLocations() throws IOException, InterruptedException {
            return new String[] { location };
        }
    }
```

InputFormat code. T RedisHashInputFormat o o RedisHashOutput Format y y o o o v bl o o o o o o getSplits o , o o v b o RedisHashInputSplits b o b o o T ll o p o o T o y o p pl o o b v l by RedisHashRecordReader T createRecordReader o ll by o o o o T o ’ initialize o ll by o , o ▲ by o v o , l o o l o o p pl pl o

```
    public static class RedisHashInputFormat extends InputFormat<Text, Text> {

        public static final String REDIS_HOSTS_CONF =
                "mapred.redishashinputformat.hosts";
        public static final String REDIS_HASH_KEY_CONF =
                "mapred.redishashinputformat.key";
        private static final Logger LOG = Logger
                .getLogger(RedisHashInputFormat.class);
```

```
    public static void setRedisHosts(Job job, String hosts) {
        job.getConfiguration().set(REDIS_HOSTS_CONF, hosts);
    }

    public static void setRedisHashKey(Job job, String hashKey) {
        job.getConfiguration().set(REDIS_HASH_KEY_CONF, hashKey);
    }

    public List<InputSplit> getSplits(JobContext job) throws IOException {
        String hosts = job.getConfiguration().get(REDIS_HOSTS_CONF);

        if (hosts == null || hosts.isEmpty()) {
            throw new IOException(REDIS_HOSTS_CONF
                    + " is not set in configuration.");
        }

        String hashKey = job.getConfiguration().get(REDIS_HASH_KEY_CONF);
        if (hashKey == null || hashKey.isEmpty()) {
            throw new IOException(REDIS_HASH_KEY_CONF
                    + " is not set in configuration.");
        }

        // Create an input split for each host
        List<InputSplit> splits = new ArrayList<InputSplit>();
        for (String host : hosts.split(",")) {
            splits.add(new RedisHashInputSplit(host, hashKey));
        }

        LOG.info("Input splits to process: " + splits.size());
        return splits;
    }

    public RecordReader<Text, Text> createRecordReader(InputSplit split,
            TaskAttemptContext context) throws IOException,
            InterruptedException {
        return new RedisHashRecordReader();
    }

    public static class RedisHashRecordReader extends RecordReader<Text, Text> {
        // code in next section
    }

    public static class RedisHashInputSplit extends
            InputSplit implements Writable {
        // code in next section
    }
}
```

RecordReader code. T RedisHashRecordReader o o o o T initialize o ll by o p o v p pl p o , o o o W o o b o y/v l p ll b o T o ' v o o o b l , o ply p ll v y o v o o W o o o v b o lp l b y

nextKeyValue, o po o o ' bl o b o y v l ▲ v l o true o o y/v l p o p o v ll y/v l p , false is returned o p o pl

T o o o o by o o y v l o pp o p o o l o o b v po bl T getProgress o l o po l o Jo b T o l l o b po bl F lly, close o o l p o S p ll ll o o o o initialize o , o o o

```
public static class RedisHashRecordReader extends RecordReader<Text, Text> {

    private static final Logger LOG =
            Logger.getLogger(RedisHashRecordReader.class);
    private Iterator<Entry<String, String>> keyValueMapIter = null;
    private Text key = new Text(), value = new Text();
    private float processedKVs = 0, totalKVs = 0;
    private Entry<String, String> currentEntry = null;

    public void initialize(InputSplit split, TaskAttemptContext context)
            throws IOException, InterruptedException {
        // Get the host location from the InputSplit
        String host = split.getLocations()[0];
        String hashKey = ((RedisHashInputSplit) split).getHashKey();

        LOG.info("Connecting to " + host + " and reading from "
                + hashKey);

        Jedis jedis = new Jedis(host);
        jedis.connect();
        jedis.getClient().setTimeoutInfinite();

        // Get all the key/value pairs from the Redis instance and store
        // them in memory
        totalKVs = jedis.hlen(hashKey);
        keyValueMapIter = jedis.hgetAll(hashKey).entrySet().iterator();
        LOG.info("Got " + totalKVs + " from " + hashKey);
        jedis.disconnect();
    }
```

```
        public boolean nextKeyValue() throws IOException,
                InterruptedException {

            // If the key/value map still has values
            if (keyValueMapIter.hasNext()) {
                // Get the current entry and set the Text objects to the entry
                currentEntry = keyValueMapIter.next();
                key.set(currentEntry.getKey());
                value.set(currentEntry.getValue());
                return true;
            } else {
                // No more values? return false.
                return false;
            }
        }

        public Text getCurrentKey() throws IOException,
                InterruptedException {
            return key;
        }

        public Text getCurrentValue() throws IOException,
                InterruptedException {
            return value;
        }

        public float getProgress() throws IOException, InterruptedException {
            return processedKVs / totalKVs;
        }

        public void close() throws IOException {
            // nothing to do here
        }
    }
```

Driver code.

```
        // Use the identity mapper
        job.setNumReduceTasks(0);

        job.setInputFormatClass(RedisHashInputFormat.class);
        RedisHashInputFormat.setRedisHosts(job, hosts);
        RedisHashInputFormat.setRedisHashKey(job, hashKey);

        job.setOutputFormatClass(TextOutputFormat.class);
        TextOutputFormat.setOutputPath(job, outputDir);

        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(Text.class);

        System.exit(job.waitForCompletion(true) ? 0 : 3);
    }
```

Partition Pruning

Pattern Description

Partition pruning o y o p p pl o p l o b lo o M p b o o l

Intent

o v o p o by p v l , yo o y lly lo b o q by ppl o

Motivation

Typ lly, ll lo o M p o b o p p ll l l o o b o o b o q y, lo llo l l o p o yp o by o o v l , yo wo o o p o by lo o o ly o l Fo pl , yo o o ly ly b o , p o yo by ll o yo o ly o lo o

T v o p o l b l p ly, o yo M p o bo v o v , b o v T o by ply yo q y o , pl o o o b ▲ y o o o l b o p y o o o l y p p o T p o o o lo , llo b o p o b o q y

T p o lly l o wo l l l ly o
yo v o o ly o yp o p o
p o , yo p o pl o
ply o p l ll ly p o o S
ll yo ly p o q y l , yo o ’
o - pl o o ly T
v v o o v lo p , yo lo o lly
oo o yo bo !

Structure

F 7-4 o o p o p , pl b lo

- T `InputFormat` p o o l T `getSplits` o
p y p l o , b p pl ll b
, b o p W l o o yp lly
o l , o o o o o q y o l p Fo
, o o l y by , `InputFormat` p
p , o l o p ll o M p o b
l v by , y 1 o o , o ly o
o b by M p o b lo o o M T
y p o o o b o
q y, p o l
- T `RecordReader` pl o p o o b o
l -b p , o l `LineRecordReader` b o y/
v l p o l l o , yo ’ll v o o o
o o yo

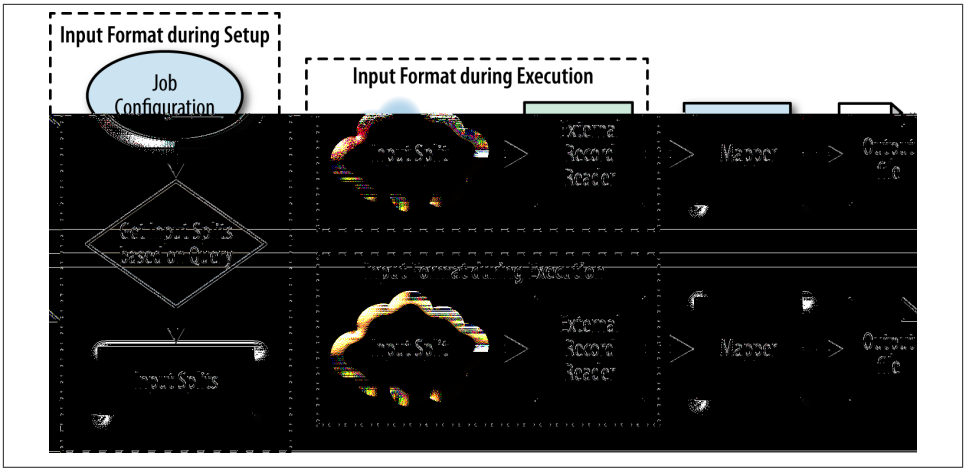


Figure 7-4. The structure of the partition pruning pattern

Consequences

o p o ly o o by M p o b, o v lo o o ly T o o p o p o o v ll p o o T o by o p ll o p o yo p b o v o p

Resemblances

SQL

M y o l o l b l p o p p lyW yo bl , yo p y o b o l p o b ll l o v l o pp o p o

```
CREATE TABLE parted_data
(foo_date    DATE)
PARTITION BY RANGE(foo_date)
(
PARTITION foo_2012 VALUES LESS THAN(TO_DATE('01/01/2013','DD/MM/YYYY')),
PARTITION foo_2011 VALUES LESS THAN(TO_DATE('01/01/2012','DD/MM/YYYY')),
PARTITION foo_2010 VALUES LESS THAN(TO_DATE('01/01/2011','DD/MM/YYYY')),
);
```

T , yo q y p v l WHERE l , b ll o lly o ly l v p o

```
SELECT * FROM parted_data WHERE foo_date=TO_DATE('01/31/2012');
```

Performance analysis

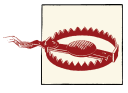
T p b o p yo p
ly b o b o q y l p
p o v v by b o o b o l
o v o p y y o / , p o p o
o p b ppl p p o o b

Partition Pruning Examples

Partitioning by last access date to Redis instances

T pl o y o o
o ly b -o- p o pp , p o o
p l T -o- p o pp p o b o l
o T o o o o p
o T , J y F b y o
o , M ▲p lo 1, o o

y b , o ll ly b o
q W p vo pl oo l o y
v o l , p o ll b o o o o
o p o , ll p o T o pl ly p y
o l o pp o o o ,
v v o v b p o p o p o



y o b b o lly o o o o
J v o l , b v ly- o o l
b o by yo o T y, ll b
y p v o p l v o v bl o
ly,o b p v o l

T o b b o po o pl o lb p o bl
o bl v o , p o -o- p o pp by l
o

Custom WritableComparable code. To lp b o o o , o `Writable Comparable` pl o o lb pp o o o by
o T l o o o l o b o
, ll l o T l o p o-b
v l o o , b l o p o o
q y pl T o pl `compareTo`, `toString`, `hashCode` o l v y oo J v v b p !

```
public static class RedisKey implements WritableComparable<RedisKey> {

    private int lastAccessMonth = 0;
    private Text field = new Text();

    public int getLastAccessMonth() {
        return this.lastAccessMonth;
    }

    public void setLastAccessMonth(int lastAccessMonth) {
        this.lastAccessMonth = lastAccessMonth;
    }

    public Text getField() {
        return this.field;
    }

    public void setField(String field) {
        this.field.set(field);
    }

    public void readFields(DataInput in) throws IOException {
        lastAccessMonth = in.readInt();
        this.field.readFields(in);
    }

    public void write(DataOutput out) throws IOException {
        out.writeInt(lastAccessMonth);
        this.field.write(out);
    }

    public int compareTo(RedisKey rhs) {
        if (this.lastAccessMonth == rhs.getLastAccessMonth()) {
            return this.field.compareTo(rhs.getField());
        } else {
            return this.lastAccessMonth < rhs.getLastAccessMonth() ? -1 : 1;
        }
    }

    public String toString() {
        return this.lastAccessMonth + "\t" + this.field.toString();
    }

    public int hashCode() {
        return toString().hashCode();
    }
}
```

OutputFormat code. T o p o ly b , ll o l o T o o o pl InputFormat l T o p o p o o l o p y Text o b o p v l ▲ y o l ll o y o yo p

S o o pl o o o p o o p , o o v y yo p p o o o b ▲ o p o ll q by o , o NullOutputFormat' o p o

```
public static class RedisLastAccessOutputFormat
        extends OutputFormat<RedisKey, Text> {

    public RecordWriter<RedisKey, Text> getRecordWriter(
            TaskAttemptContext job) throws IOException, InterruptedException {
        return new RedisLastAccessRecordWriter();
    }

    public void checkOutputSpecs(JobContext context) throws IOException,
            InterruptedException {
    }

    public OutputCommitter getOutputCommitter(TaskAttemptContext context)
            throws IOException, InterruptedException {
        return (new NullOutputFormat<Text, Text>()).getOutputCommitter(context);
    }

    public static class RedisLastAccessRecordWriter
            extends RecordWriter<RedisKey, Text> {
        // Code in next section
    }
}
```

RecordWriter code. T RedisLastAccessRecordWriter pl o p l o p o T o o o l o o ll p p T p o o - o - - p p write o o v p o p T write o p o o o o o o l o T p o o b v y, b b o o l →J▲N, 1→F , , 11→D C T ll b o - o o T close o o ll

```
public static class RedisLastAccessRecordWriter
        extends RecordWriter<RedisKey, Text> {

    private HashMap<Integer, Jedis> jedisMap = new HashMap<Integer, Jedis>();

    public RedisLastAccessRecordWriter() {
        // Create a connection to Redis for each host
```

```
            int i = 0;
            for (String host : MRDPUtils.REDIS_INSTANCES) {
                Jedis jedis = new Jedis(host);
                jedis.connect();
                jedisMap.put(i, jedis);
                jedisMap.put(i + 1, jedis);
                i += 2;
            }
        }

        public void write(RedisKey key, Text value) throws IOException,
                InterruptedException {
            // Get the Jedis instance that this key/value pair will be
            // written to -- (0,1)->0, (2-3)->1, ... , (10-11)->5
            Jedis j = jedisMap.get(key.getLastAccessMonth());

            // Write the key/value pair
            j.hset(MONTH_FROM_INT.get(key.getLastAccessMonth()), key
                    .getField().toString(), value.toString());
        }

        public void close(TaskAttemptContext context) throws IOException,
                InterruptedException {
            // For each jedis instance, disconnect it
            for (Jedis jedis : jedisMap.values()) {
                jedis.disconnect();
            }
        }
    }
```

Mapper code. T pp o p p o v l o o p RedisKey o p v l T o o l p v Calendar SimpleDateFormat l

```
    public static class RedisLastAccessOutputMapper extends
            Mapper<Object, Text, RedisKey, Text> {

        // This object will format the creation date string into a Date object
        private final static SimpleDateFormat frmt = new SimpleDateFormat(
                "yyyy-MM-dd'T'HH:mm:ss.SSS");

        private RedisKey outkey = new RedisKey();
        private Text outvalue = new Text();

        public void map(Object key, Text value, Context context)
                throws IOException, InterruptedException {

            Map<String, String> parsed = MRDPUtils.transformXmlToMap(value
                    .toString());

            String userId = parsed.get("Id");
            String reputation = parsed.get("Reputation");
```

```
            // Grab the last access date
            String strDate = parsed.get("LastAccessDate");

            // Parse the string into a Calendar object
            Calendar cal = Calendar.getInstance();
            cal.setTime(frmt.parse(strDate));

            // Set our output key and values
            outkey.setLastAccessMonth(cal.get(Calendar.MONTH));
            outkey.setField(userId);
            outvalue.set(reputation);

            context.write(outkey, outvalue);
        }
    }
```

Driver code. T v bo v y l o o b ob o o ▲ll p l o o ly l by o p o l o

```
    public static void main(String[] args) throws Exception {
        Configuration conf = new Configuration();

        Path inputPath = new Path(args[0]);

        Job job = new Job(conf, "Redis Last Access Output");
        job.setJarByClass(PartitionPruningOutputDriver.class);

        job.setMapperClass(RedisLastAccessOutputMapper.class);
        job.setNumReduceTasks(0);

        job.setInputFormatClass(TextInputFormat.class);
        TextInputFormat.setInputPaths(job, inputPath);

        job.setOutputFormatClass(RedisHashSetOutputFormat.class);

        job.setOutputKeyClass(RedisKey.class);
        job.setOutputValueClass(Text.class);

        int code = job.waitForCompletion(true) ? 0 : 2;

        System.exit(code);
    }
```

Querying for user reputation by last access date

T pl o o o q y o o o o l o pl , yo p o v o p o l DFS, p o o F o o ly l ll ly by p o

T o p o p o wo yo o ' v o y
o -o - p o pp o v , o
o o ly o o q o ' v b ,
o o ly o o l p o Fo ,
p JA N,F ,MA ,N V o o l ll p pl ,
o o o , 1, 5 All o
ll b , b o ly o o 1 5 ll b p ll T
b v o o o ll ll ,
o ly o o o o y!

T o b lo o po o pl o llo p o bl

o bl v q y o o p o pp by o , o ly
q o y q y p ll l

InputSplit code. T p pl o v y l o p pl l
So p pl p 197 o o o y, o o o
l pl y T b p o b o o , o
ll b o ly b o

```
public static class RedisLastAccessInputSplit
        extends InputSplit implements Writable {

    private String location = null;
    private List<String> hashKeys = new ArrayList<String>();

    public RedisLastAccessInputSplit() {
        // Default constructor for reflection
    }

    public RedisLastAccessInputSplit(String redisHost) {
        this.location = redisHost;
    }

    public void addHashKey(String key) {
        hashKeys.add(key);
    }

    public void removeHashKey(String key) {
        hashKeys.remove(key);
    }

    public List<String> getHashKeys() {
        return hashKeys;
    }

    public void readFields(DataInput in) throws IOException {
        location = in.readUTF();
        int numKeys = in.readInt();
```

```
            hashKeys.clear();
            for (int i = 0; i < numKeys; ++i) {
                hashKeys.add(in.readUTF());
            }
        }

        public void write(DataOutput out) throws IOException {
            out.writeUTF(location);
            out.writeInt(hashKeys.size());
            for (String key : hashKeys) {
                out.writeUTF(key);
            }
        }

        public long getLength() throws IOException, InterruptedException {
            return 0;
        }

        public String[] getLocations() throws IOException, InterruptedException {
            return new String[] { location };
        }
    }
```

InputFormat code. T p o l ll ly RedisLastAccessInputS plito b o l o o M l o p o o l p Fo o p 7 , o p o RedisKeyo b , p o o b pl o o o pp pl o lly po o -o - p pl o o y o p pl , bo o o o pl pl o b o p l o , o o y , y o pl l y b ▲List o o v l o p T ll b o p pl q v l o b o q o y q y

T b o lp l p o lp o v o o , ll o o o o T l o o p o o static bb o b v y

```
    public static class RedisLastAccessInputFormat
            extends InputFormat<RedisKey, Text> {

        public static final String REDIS_SELECTED_MONTHS_CONF =
                "mapred.redilastaccessinputformat.months";
        private static final HashMap<String, Integer> MONTH_FROM_STRING =
                new HashMap<String, Integer>();
        private static final HashMap<String, String> MONTH_TO_INST_MAP =
                new HashMap<String, String>();
        private static final Logger LOG = Logger
```

```
        .getLogger(RedisLastAccessInputFormat.class);

static {
    // Initialize month to Redis instance map
    // Initialize month 3 character code to integer
}

public static void setRedisLastAccessMonths(Job job, String months) {
    job.getConfiguration().set(REDIS_SELECTED_MONTHS_CONF, months);
}

public List<InputSplit> getSplits(JobContext job) throws IOException {

    String months = job.getConfiguration().get(
            REDIS_SELECTED_MONTHS_CONF);

    if (months == null || months.isEmpty()) {
        throw new IOException(REDIS_SELECTED_MONTHS_CONF
                + " is null or empty.");
    }

    // Create input splits from the input months
    HashMap<String, RedisLastAccessInputSplit> instanceToSplitMap =
                new HashMap<String, RedisLastAccessInputSplit>();

    for (String month : months.split(",")) {
        String host = MONTH_TO_INST_MAP.get(month);
        RedisLastAccessInputSplit split = instanceToSplitMap.get(host);
        if (split == null) {
            split = new RedisLastAccessInputSplit(host);
            split.addHashKey(month);
            instanceToSplitMap.put(host, split);
        } else {
            split.addHashKey(month);
        }
    }

    LOG.info("Input splits to process: " +
            instanceToSplitMap.values().size());
    return new ArrayList<InputSplit>(instanceToSplitMap.values());
}

public RecordReader<RedisKey, Text> createRecordReader(
        InputSplit split, TaskAttemptContext context)
        throws IOException, InterruptedException {
    return new RedisLastAccessRecordReader();
}

public static class RedisLastAccessRecordReader
```

```
            extends RecordReader<RedisKey, Text> {
            // Code in next section
    }
}
```

RecordReader code. T RedisLastAccessRecordReader b o o pl o o v o o l pl , v y o initialize o , o o ply o

nextKeyValue, o o o o o o ll,o v o ll o o o v o v l , ly false, o o o p , o o p ll ll o p T o o ll l v l p o ▲ o - l bo p o o o o pl , ll bo p b o o o o o o o b

T pl o o o l o o RedisHash RecordReader o

```
public static class RedisLastAccessRecordReader
        extends RecordReader<RedisKey, Text> {

    private static final Logger LOG = Logger
            .getLogger(RedisLastAccessRecordReader.class);
    private Entry<String, String> currentEntry = null;
    private float processedKVs = 0, totalKVs = 0;
    private int currentHashMonth = 0;
    private Iterator<Entry<String, String>> hashIterator = null;
    private Iterator<String> hashKeys = null;
    private RedisKey key = new RedisKey();
    private String host = null;
    private Text value = new Text();

    public void initialize(InputSplit split, TaskAttemptContext context)
            throws IOException, InterruptedException {

        // Get the host location from the InputSplit
        host = split.getLocations()[0];

        // Get an iterator of all the hash keys we want to read
        hashKeys = ((RedisLastAccessInputSplit) split)
                .getHashKeys().iterator();

        LOG.info("Connecting to " + host);
    }
```

```
    public boolean nextKeyValue() throws IOException,
            InterruptedException {

        boolean nextHashKey = false;
        do {
            // if this is the first call or the iterator does not have a
            // next
            if (hashIterator == null || !hashIterator.hasNext()) {
                // if we have reached the end of our hash keys, return
                // false
                if (!hashKeys.hasNext()) {
                    // ultimate end condition, return false
                    return false;
                } else {
                    // Otherwise, connect to Redis and get all
                    // the name/value pairs for this hash key
                    Jedis jedis = new Jedis(host);
                    jedis.connect();
                    String strKey = hashKeys.next();
                    currentHashMonth = MONTH_FROM_STRING.get(strKey);
                    hashIterator = jedis.hgetAll(strKey).entrySet()
                            .iterator();
                    jedis.disconnect();
                }
            }

            // If the key/value map still has values
            if (hashIterator.hasNext()) {
                // Get the current entry and set
                // the Text objects to the entry
                currentEntry = hashIterator.next();
                key.setLastAccessMonth(currentHashMonth);
                key.setField(currentEntry.getKey());
                value.set(currentEntry.getValue());
            } else {
                nextHashKey = true;
            }
        } while (nextHashKey);

        return true;
    }

    ...
}
```

Driver code. T v o o o ly p v o l T o o p by p o o o o , o v y l o o p o y o o b ▲ , y pp p o y ly o v

```
public static void main(String[] args) throws Exception {
    Configuration conf = new Configuration();

    String lastAccessMonths = args[0];
    Path outputDir = new Path(args[1]);

    Job job = new Job(conf, "Redis Input");
    job.setJarByClass(PartitionPruningInputDriver.class);

    // Use the identity mapper
    job.setNumReduceTasks(0);

    job.setInputFormatClass(RedisLastAccessInputFormat.class);
    RedisLastAccessInputFormat.setRedisLastAccessMonths(job,
            lastAccessMonths);

    job.setOutputFormatClass(TextOutputFormat.class);
    TextOutputFormat.setOutputPath(job, outputDir);

    job.setOutputKeyClass(RedisKey.class);
    job.setOutputValueClass(Text.class);

    System.exit(job.waitForCompletion(true) ? 0 : 2);
}
```

CHAPTER 8

Final Thoughts and the Future of Design Patterns

▲ o bo o ’ , M p o v q ly l¶
y p o pp p v y y o o v M o p o ly
o b o *MapReduce Design Patterns*, o b o b l o b o p T p o ll v o y’
o o o p o o ly by o , b l o by
l y o

p , ’ll p l o l o M p
p W ll y o o W y ll b o p
o ll o y’ p o l o yW ll
p o o y

Trends in the Nature of Data

M p y o o p ’ b o ly y o
b o pl o y M p o b ly o
o o b oo o p l¶ p ly o o l
o o o l o o p l o y o ly
p

Images, Audio, and Video

o o o bv o o o , o ,
v o ly T o o oo o b y
M p b l yp lly v y l l o ly
y v o o o b M l ly b o

o o l o l o o p o ly, p o
pl o , o M p yp o ly
ll S M p boo , ’ll o l ly
yp o lly , v o l o o , b ll
o o o o l

pl y p l l o l
V o v o lb p l o v , l o o o - o l
To o p o , y l o y v o M p o llb v y o
,o - o l p p T o o o o b
o ly T o , ’ ll o boo 1 -p l by 1 -p l by
5- o o o v o o o ▲ l o l
po p l y, ’ll o p o o o lb lly pl o o
p pl p o p ly , po bl y ll ll Fo
pl , S D , o p - o ly l b , p lly b l o l
l - o l

Streaming Data

M p o lly b ly y , b ly l l
l p o o y p o o M p y , o ly
p o b o v l Fo pl , o
b v lb , b M p o b o ly v y o

T o v o o F , p o o ’ o o o
o ’ o lly, p o v ll
p o o p o l o o l b S o , M p y
yp lly p o l v ly l blb o o v o
b o p o W , o o by o T
l p o l M p

▲ p v o o bo l l , p l ly o b ll by
o b o o o p y So o p o l p
o o o o p p p o bl o
o ly p o o N p o o -l ly
o o b M p ll N o v l y l
oo p v o pp p, o o bly o l p o S
o p - o So pl o , ly l by T

T o lly o o p o b p
o boo , b o o y o
o v o o b o lly o

T o RecordReader T p p
o RecordReader o b l y
o l T o p o l o
l o pl

T o pl p o b o v lo - p o b
o v y o o T o p p o
o *k* b o v y o , p-o ly o b
k pp p pl y o l o

The Effects of YARN

▲ N ▲ o o N o o -v b l y v o oo p
M p ly v o ll v lly o
bl l M y oo p o y o o o , ll
b o p ▲ l v l, ▲ N pl po b l o Jo bT
T T o l o M ,o No M p o , o ▲p
pl o M p ppl o o o b T o M No M b
y o p o l o o p- - b p
lb b y o p o ▲ppl o M l o - p
o lo o p o b o o b o o lb o q ,
l by o M No M

W o p o p o o o o
o l, M p o o o ’ boo y o p l
o o M , , o lp o ,o o o

M p p ll o o lv , b M p
ll ll o v , o b l o b ppl o
o o o o ▲ N, o o o o l o
o p o bl y b o l o o lv o o W ’ll o
p ll ll b ’ v y y o , l
o l o l o b o W ll l ly v lly ▲ppl
o M p o b l o pl ly o o o lv yp o
p o bl

Patterns as a Library or Component

v , p o o , o o y o o po
p b l - l y l l b y T yp o p o o
o l p , ll, l b y p p
yo , o pl p T v l
o o oo p M p p o oo p l b ,
`TotalOrderPartitioner`, `ChainReducer`, `MultipleOutputs`

T v y l o po o o T p boo
p o lp yo o lv p o bl o y l y o
o , o l p o b o yo o v l p po o
o o b lp l lb

How You Can Help

yo yo 'v v lo p o v l M p p yo v ' b o
yo l o , yo o l ly o o o o o o
o l

T b o q o yo o l y o T o o
q o o oo p o boo

Is the problem you are trying to solve similar to another pattern's target problem?
y po o p v y o o o o C p 5,
p l , q o o ly

What is at the root of this pattern?
o p o b bly v lo p p o o lv v y p p o bl v
o o p o o D v lo p ll b o o lb
p o o p o bl o p o o lv o o pl p o b
l T o o o ly v p l

What is the performance profile?
o o p ll po o
o y ll b l o p v o p o ll
b Fo pl , o p o pl y b p o o v o
b y

How might have you solved this problem otherwise?
F o pl o o M p o SQ
l p o lp o p lly b o M p -
p o l o

APPENDIX A

Bloom Filters

Overview

Go v by o o bo 197 , bo l p o b b l
o b l o bo l v
o p v o v o J v Set, l
o o p , o l Fo pl , o 3
p o o o y bo l o 1 4
, lly o bo l
o p o p bo F l p 49

W l l v o y v , o l y 1 %
W l l po v po bl , l v o T l
o v o o yb o ll v v y
W o l bo l , l b o , b o o v T
b o bo l pl o l o ,
Go bo F l , b y yp lly q o o y ▲ o l
o , p o b b l y o l po v bo l o b
l o y v b , y o
b v - o v o l o ll
p o

T o llb v bl o l pl o o bo l b b

m
T b o b l

n
T b o b

p
T l p v

k
T b o o o p o l o o o *m*
b o o b o

▲ bo l p by o o o *m* b l o o F
l *n*, *k* o v l o b *m* o v o
o o *m* - *1* T b o bo l l o o T
o p o o ll *training* bo l ▲ l o bo
l , o b y l y b o o o p v o l W
b l o , o o
b o y l b o ll o o, o
ll b o , yb b o
l , *k* o l v ll b o o

T l o o b v y b b y v b
o by o b o o o l yb b o l v b
o, o *false positive* T lly, l po v b o o ll
n o o ,o l pp o o o *n*

T o lb o b b o o o o bo l ,
l o o bo l o yo bo l o b
l po v ▲ o pl o oo p bo l b
o bo l p 53

Use Cases

T o l b o o o o bo l y ppl o
b o oo l p o o o o o p v o p o , bo
l o l ly b l o l b o o p o

Representing a Data Set

o o b o bo l o p v y l ppl
o ▲ llo o l p by o o y, ll
p v / q ply o p ll o ▲ bo l
lly b o by q o p , lb o
o y o o q o T o bv o o
o p l bo l l po v W o o
b l v o o o o , b y o 1 %
v l o o ▲ po -p o o o p o o l b
,o q y l b l o oo o p o

Reduce Queries to External Database

v y o o o lbo l o b o q o b bo o y p yo v l y o l lbo l , ppl o o y l b o v l b o v q y b l y o o o , po v lbo l b o o po y b l , b b o o p o b l q y b T ll lb o y p o bl T o p o lly l l b o q bo o v l o l po v , lbo l y b o p o o y

Google BigTable

oo l ' T bl lbo l o o l o o - y p lbo l o bb o y, v o l o o l o l v v l , v ly o v l v o p l o v l o o y l o v q , p o o b lly

Downsides

T l po v l o o lbo l v lbo l l o o v 1% l po v , yo v llo o l l v l , bo o o o o o po v l W o o l p l lyo

T o lly, yo o o v l o lbo l l o v l o l q b o b o o , b ly l ly o o l o p l b S o o o l o y y o o l y o l o ll *Counting Bloom Filter*, p o W lbo l , o b o o , byo W l o v , byo T q o o y o b , l o l l o v o v lb o l T , o o lb ll l v v l o o , p o bl o v l o v l

W lbo l b ppl o l M p , l o v ly lbo l o b ▲ lbo l

l o DFS, ly b byo ppl o o v , o bo l o l q p v / o p o , b o v yo p o bo l o pl o o o b ▲ po , l b ll b

Tweaking Your Bloom Filter

o bo l l o , b v yb l o o pp o o o b o l yo o o , bo l b pp o p ly o v -p l po v T b l po v , o b q o bo l ' y F ▲-1 o o o l l o bo l o p l-*k*

$$m = \frac{-n \cdot ln(p)}{ln(2)^2}$$

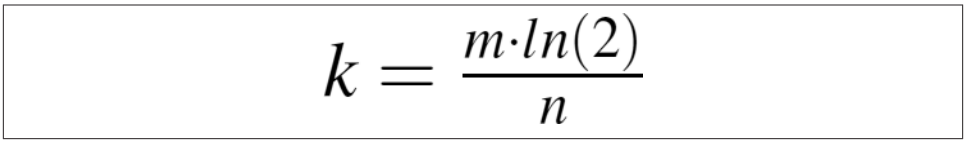
Figure A-1. Optimal size of a Bloom filter with an optimal-k

T o lb J v lp o l l o p l *m*

```
/**
 * Gets the optimal Bloom filter sized based on the input parameters and the
 * optimal number of hash functions.
 *
 * @param numElements
 *            The number of elements used to train the set.
 * @param falsePosRate
 *            The desired false positive rate.
 * @return The optimal Bloom filter size.
 */
public static int getOptimalBloomFilterSize(int numElements,
        float falsePosRate) {
    return (int) (-numElements * (float) Math.log(falsePosRate)
            / Math.pow(Math.log(2), 2));
}
```

T o p l-*k* b o o o l b o bo l W oo p bo l pl o , o bo l b o o o v o b o p vo o l o pp o p o bo l o p l-*k*

F ▲- o o o p l-*k* b o l lyo o bo l b o l o l



$$k = \frac{m \cdot ln(2)}{n}$$

Figure A-2. Optimal-k of a Bloom filter

T o lb lp o l l o p l-*k*

```
/**
 * Gets the optimal-k value based on the input parameters.
 *
 * @param numElements
 *            The number of elements used to train the set.
 * @param vectorSize
 *            The size of the Bloom filter.
 * @return The optimal-k value, rounded to the closest integer.
 */
public static int getOptimalK(float numElements, float vectorSize) {
    return (int) Math.round(vectorSize * Math.log(2) / numElements);
}
```

Index

A

B

C

We'd like to hear your suggestions for improving our indexes. Send email to index@oreilly.com.

D

E

F

G

H

I

J

K

R

S

T

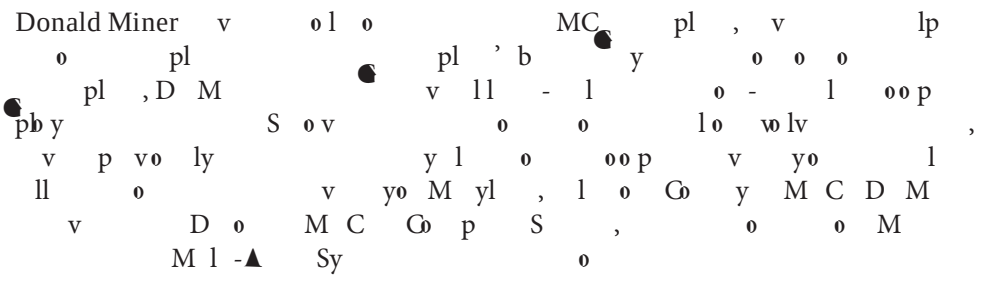
U

V

W

Y

About the Authors



Colophon