ΗΥ – 562 Προχωρημένα Θέματα Βάσεων Δεδομένων

1η Σειρά Ασκήσεων

Ημερομηνία Παράδοσης: 28/10/2018

Βασιλεία Φραγκιαδάκη

A.M. 1101

Exercise 1 Frequent Terms and Stop Words

Ζητείται να συλλέξουμε όλα τα stopwords στο ίδιο σύνολο κειμένων. Έχει χρησιμοποιηθεί ως αρχείο εισόδου "pg100.txt". Για αυτό το σκοπό πρέπει να υλοποιήσουμε τις απαραίτητες Map Reduce συναρτήσεις ώστε να επεκτείνουμε το παράδειγμα WordCount και να μπορέσουμε να ταυτοποιήσουμε τα stopwords, όλες δηλαδή τις λέξεις με συχνότητας εμφάνισης μεγαλύτερη από 4000. Ζητείται να μετρήσουμε μόνο τις έγκυρες λέξεις του αγγλικού αλφαβήτου, και να αφαιρέσουμε όλους τους μη αλφαριθμητικούς χαρακτήρες.

1. Στο πρώτο ερώτημα πρέπει να τυπώσουμε τις k πιο συχνές λέξεις μαζί με τη συχνότητά τους, υποθέτοντας ότι k=10

Αρχικά, σκεφτόμαστε ότι στον κώδικα του παραδείγματος WordCount, το Map Reduce job μετράει τις συχνότητες εμφάνισης των λέξεων στο αρχείο εισόδου. Για να τυπώσουμε τις k πιο συχνές λέξεις, αρκεί να προσθέσουμε άλλο ένα Map Reduce job, που θα παίρνει ως είσοδο το αρχείο που παράγεται από το 1ο job και θα τυπώνει (key,value) ζεύγη (pairs), όπου το key θα είναι η συχνότητα της κάθε λέξης και το value θα είναι το stopword. Χρειάζεται, ακόμα, να ορίσουμε ότι τα κλειδιά θα ταξινομούνται με φθίνουσα σειρά ώστε να μπορούμε να βλέπουμε πρώτα τις πιο συχνές λέξεις. Έπειτα, μπορούμε με κάποια εντολή σε τερματικό να τυπώσουμε τις 10 πιο συχνές λέξεις όπως ζητάει το ερώτημα.

Για να προσθέσουμε το 2ο job, ορίζουμε αρχικά άλλο έναν mapper και άλλο έναν reducer. Ο mapper, θα διαβάζει το αρχείο εξόδου του 1ου job, part-r-00000 από το φάκελο /output/temp, ανά γραμμή. Κάθε γραμμή χωρίζεται με τη χρήση της σταθεράς WORD BOUNDARY που είναι κάποια whitespaces και ο χαρακτήρας backspace που ορίζει ουσιαστική το τέλος κάθε λέξης. Κάθε λέξη που διαβάζεται, μετατρέπεται σε μικρά γράμματα ενώ τα μονά εισαγωγικά αφαιρούνται, ενώ κάθε μη αλφαριθμητικός χαρακτήρας αντικαθίσταται από το κενό, με τη χρήση κανονικής έκφρασης (regular expression). Επίσης, τα whitespaces στην αρχή και στο τέλος κάθε λέξης αφαιρούνται με τη χρήση της μεθόδου trim(). Οι εντολές αυτές είναι οι ακόλουθες:

```
word = word.toLowerCase();
word = word.replaceAll("'","");  //remove single quotes
word = word.replaceAll("[^a-zA-Z0-9]"," ");  //remove the rest with a space
word = word.trim(); //trim whitespaces
```

Έπειτα, κάθε λέξη χωρίζεται εκ νέου σε νέες υπολέξεις (subwords) αφού τα σημεία στίξης αντικαταστάθηκαν με το κενό (" "). Έπειτα, κάθε λέξη γράφεται ως ζεύγος με τιμή τη συχνότητα εμφάνισης 1.

Στη συνέχεια, ο 1ος reducer δέχεται ταξινομημένα τα κλειδιά-λέξεις όπου οι τιμές είναι γκρουπαρισμένες ανά κλειδί, όπως έχει γίνει από τον partitioner. Έτσι, κάθε κλειδί-λέξη δείχνει σε μία λίστα ακεραίων του Hadoop (IntWritable), μία μονάδα για κάθε φορά που εμφανίστηκε η λέξη. Ο reducer διατρέχει τη λίστα των ακεραίων και προσθέτει σε ένα ακέραιο sum, το άθροισμα, που αναπαριστά τη συχνότητα εμφάνισης της κάθε λέξης. Έπειτα, γράφει στο αρχείο εξόδου του 1ου job, τα ζεύγη, κάθε λέξη με την αντίστοιχη σειρά εμφάνισης σε κάθε γραμμή.

Ο 2ος mapper, διαβάζει το αρχείο εξόδου του 1ου job, ανά γραμμή. Θέλουμε να εντοπίσουμε, το κλειδί και την τιμή σε κάθε γραμμή, ώστε να μπορούμε να τα γράψουμε αντίστροφα με τη χρήση της μεθόδου context.write(). Δηλαδή, τώρα τα κλειδί είναι η συχνότητα εμφάνισης και η τιμή είναι η λέξη-stopword. Γι' αυτό το λόγο καθώς διατρέχει το πρόγραμμα τις λέξεις κάθε γραμμής, κάθε άδεια λέξη αγνοείται έως ότου διαβαστεί η 1η λέξη, οπότε και ενεργοποιείται η σημαία και πλέον το πρόγραμμα περιμένει να διαβάσει τη συχνότητα που είναι τύπου IntWritable.

Ο 2ος reducer δέχεται ταξινομημένα τα κλειδιά, κατά φθίνουσα σειρά, και τις τιμές – λέξεις γκρουπαρισμένες ανά κλειδί. Διατρέχοντας όλα τα κλειδιά, καταγράφονται μόνο τα ζεύξη που έχουν κλειδί μεγαλύτερο από 4000.

Το Job Chaining γίνεται μέσα στη μέθοδο run(). Μετά την 1η εργασία, προστίθεται μία δεύτερη, που παίρνει ως είσοδο το αρχείο εξόδου της 1ης εργασίας που βρίσκεται σε ένα φάκελο /output/temp και γράφει σε ένα αρχείο εξόδου στο φάκελο /output/final. Έπειτα ορίζουμε ότι ο τύπος δεδομένων του κλειδιού είναι IntWritable και της τιμής είναι Text.

Τέλος, για την ταξινόμηση των κλειδιών εισάγεται στη μέθοδο run() η ακόλουθη εντολή.

```
job2.setSortComparatorClass(DescendingComparator.class);
```

Η κλάση DescendingComparator κάνει extend την κλάση WritableComparable, που επιτρέπει τη σύγκριση των τύπων δεδομένων Writable του Hadoop. Αρκεί να κάνουμε override τη μέθοδο compare, ώστε να επιτρέψουμε τη σύγκριση του τύπου δεδομένων IntWritable, που είναι η συχνότητες και να αλλάξουμε το πρόσημο ώστε να αντιστρέψουμε την κλασική αύξουσα σειρά σε φθίνουσα.

Τώρα για να τυπώσουμε τις 10 πιο συχνές λέξεις μαζί με τη συχνότητά τους, ανοίγουμε ένα τερματικό και κατευθυνόμαστε στο φάκελο output/final. Με την εντολή:

```
cat part-r-00000 | head –n 10
```

τυπώνονται οι 10 πρώτες γραμμές.

```
[cloudera@quickstart final]$ cat part-r-00000 | head
30047   the
28429   and
23848   i
21411   to
18837   of
16167   a
14680   you
13205   my
12256   that
12207   in
```

Εναλλακτικά, μπορούμε να εκτελέσουμε το jar αρχείο στο hadoop και να διαβάσουμε το αρχείο από το HDFS. Έτσι, κάνουμε export το project WordCountFrequency σε jar αρχείο με όνομα WordcountFrequency.jar και το αποθηκεύουμε στο φάκελο /home/cloudera/. Έπειτα, μέσω τερματικού από το φάκελο αυτό εισάγουμε τις εντολές:

```
hadoop fs –put workspace/WordCountFrequency/pg100.txt
hadoop jar WordcountFrequency.jar exercise1.WordCount pg100.txt output
```

Η διεύθυνση του φακέλου ouput/ στο σύστημα αρχείων HDFS του Hadoop είναι:

```
hdfs://quickstart.cloudera:8020/user/cloudera/output/
```

Έτσι, μπορούμε να εκτελέσουμε την ακόλουθη εντολή στο τερματικό, και να τυπωθούν οι 10 πρώτες λέξεις με τη συχνότητα εμφάνισής τους, όπως φαίνεται στην ακόλουθη εικόνα:

```
[cloudera@quickstart ~]$ hadoop fs -cat output/final/part-\* | head -n 10
30047   the
28429   and
23848   i
21411   to
18837   of
16167   a
14680   you
13205   my
12256   that
12207   in
```

2. Στο 2^{ο} ερώτημα, ζητείται να αποθηκεύσουμε όλες τις λέξεις σε ένα csv αρχείο στο σύστημα αρχείων HDFS του Hadoop.

Τοπικά, εκτελούμε την ακόλουθη εντολή, που συγχωνεύει το αρχείο εξόδου του προγράμματός μας σε ένα csv αρχείο με όνομα “frequency-words.csv”.

```
[cloudera@quickstart WordCountFrequency]$ hadoop fs -getmerge output/final/* fre
quency-words.csv
```

Έχει απομείνει να δώσουμε στο csv αρχείο την κατάλληλη μορφοποίηση, δηλαδή θέλουμε να περιέχει όλες τις λέξεις χωρίς τις συχνότητές τους. Γι’ αυτό το λόγο, εκτελούμε την ακόλουθη εντολή, που μεταφέρει τη 2η στήλη με τις λέξεις, σε ένα νέο αρχείο csv, που ονομάζεται “stopwords.csv”.

```
[cloudera@quickstart WordCountFrequency]$ cat frequency-words.csv | cut -f2 -s >
 stopwords.csv
```

Τέλος, για τη μεταφορά του αρχείου “stopwords.csv” στο HDFS εκτελούμε την ακόλουθη εντολή.

```
[cloudera@quickstart WordCountFrequency]$ hadoop fs -put ./stopwords.csv  output
/
```

Μπορούμε να επιβεβαιώσουμε ότι το αρχείο υπάρχει στο HDFS, όταν το εκτυπώσουμε με την ακόλουθη εντολή.

```
[cloudera@quickstart WordCountFrequency]$ hadoop fs -cat output/stopwords.csv
the
and
i
to
of
a
you
my
that
in
is
d
not
with
s
for
me
it
his
be
he
this
your
but
have
as
thou
him
so
will
what
her
thy
all
by
no
do
```