Όνοματεπώνυμο: Μουλαδένιος Κωνσταντίνος
ΑΕΜ: 2379

ΕΡΓΑΣΙΑ ΣΤΑ ΚΑΤΑΝΕΜΗΜΕΝΑ ΣΥΣΤΗΜΑΤΑ

Στόχος μας είναι να υλοποιήσουμε ένα πρόγραμμα χρησιμοποιώντας το apache hadoop framework, το οποίο θα προτείνει στους χρήστες ενός κοινωνικού δικτύου φίλους, ανάλογα με το πόσους κοινούς φίλους έχουν μεταξύ τους. Για την υλοποίηση χρειαζόμαστε 3 κλάσεις, την MapFunction, την ReduceFunction και την AggragateJob, οι οποίες επεκτείνουν και χρησιμοποιούν έτοιμες κλάσεις του hadoop framework.

Η MapFunction διαβάζει από ένα αρχείο txt που δημιουργήσαμε το οποίο σε κάθε γραμμή του περιέχει το όνομα ενός χρήστη και τους φίλους αυτού του χρήστη, χωριζόμενα μεταξύ τους με κενό. Για παράδειγμα μία γραμμή έχει τη μορφή: Alice John,Ben,Sarah,Jessica
Η Alice είναι ο χρήστης και οι John,Ben,Sarah,Jessica είναι οι φίλοι της. Η συνάρτηση map δέχεται 3 ορίσματα . Πρώτα δέχεται ένα κλειδί (key) τύπου LongWritable (στην ουσία επέκταση του Long) το οποίο είναι η θέση της γραμμής του αρχείου, δηλαδή σε κάθε κλήση της δηλώνει σε ποιά γραμμή του αρχείου βρίσκεται και πρόκειται να διαβάσει.Το δεύτερο όρισμα είναι τύπου Text map και είναι το περιεχόμενο της γραμμής που διαβάζει, δηλαδή ο χρήστης με τους φίλους του. Το τρίτο και τελευταίο όρισμα είναι τύπου Context και

εκεί είναι που θα αποθηκεύσουμε το αποτέλεσμα του map. Η map θέλει ως αποτέλεσμα ένα key – value που ως key θα περιέχει ένα ζεύγος χρηστών και ως value θα περιέχει τους φίλους του χρήστη που διαβάστηκε στην κλήση της. Με λίγα λόγια η map παίρνει τον χρήστη κάθε γραμμής και τον κάνει συνδυασμό με τους φίλους του για να δημιουργήσει τα ζευγάρια. Μόλις τελειώσουν οι κλήσεις της map έχουμε διάφορα ζευγάρια με τη μορφή : (Alice-John, Alice's Friends) , (John-Alice,John's Friends) κ.ο.κ. Μέσα στην ύλοποίηση της όμως έχουμε βάλει έναν έλεγχο που ταξινομεί το κάθε ζεύγος ανάλογα με την αλφαβητική σειρά πχ αν ένα ζεύγος ήταν (Tim-Alex)
το μετατρέπουμε σε (Alex -Tim). Με αυτό το τρόπο όταν τελειώσουν οι κλήσεις της συνάρτησης θα έχουμε (Alex-Tim,Alex's Friends) και (Alex-Tim,Tim's Friends) . Αυτό μας συμφέρει γιατί το hadoop έχει έναν μηχανισμό (shuffle and sort) ο οποίος παίρνει αυτά τα στοιχεία που έχουν το ίδιο key και πριν τα στείλει στη ReduceFunction ,τα ενώνει μετατρέποντας τα σε (Alex-Tim, [Alex's Friends,Tim's Friends]).

Οπότε η ReduceFunction δέχεται ένα string με το ζευγάρι ως KEY και ως value όλους τους φίλους του ζευγαριού. Στόχος της ReduceFunction είναι να εντοπίσει τους κοινούς φίλους του ζευγαριού και εφόσον αυτοί ξεπερναν έναν αριθμό, στην συγκεκριμένη περίπτωση 5, να προτείνει σε κάθε μέλος του ζευγαριού έναν φίλο από τους μη κοινούς. Χωρίζει το KEY και απομονώνει τους δύο χρήστες. Στη συνέχεια καθορίζει ποιός από τους φίλους στη λίστα [φίλοι,φίλοι] είναι του πρώτου χρήστη και ποιός είναι του δεύτερου. Σε αυτό μας βοηθάνε τα ονόματα, για παράδειγμα εάν στους φίλους

περιέχεται το όνομα του πρώτου χρήστη τότε πρόκειται για τους φίλους του δεύτερου χρήστη αφού δεν είναι δυνατό να είναι κανείς φίλος με τον εαυτό του. Έτσι ξεχωρίζουμε και βάζουμε σε δύο πίνακες τους φίλους του πρώτου και του δεύτερου χρήστη. Έπειτα ελέγχουμε τα στοιχεία αυτών των πινάκων μεταξύ τους για να βρούμε τον αριθμό των κοινών φίλων και επαναλαμβάνουμε την διαδικασία για να βρούμε ποιοί ακριβώς είναι και να τους αποθηκεύσουμε σε έναν άλλο πίνακα. Έτσι αν βρούμε ότι οι κοινοί φίλοι είναι παραπάνω από 5 τότε για τον πρώτο χρήστη διασχίζουμε τον πίνακα με τους φίλους του δεύτερου χρήστη και ελέγχουμε κάθε φορά αν αυτός ο φίλος ανήκει στους κοινούς και αν το όνομα του φίλου είναι ίδιο με το όνομα του πρώτου χρήστη. Αν δν ισχύουν αυτά τότε προτείνουμε αυτό το φίλο στον πρώτο χρήστη. Την ίδια διαδικασία ακολουθούμε και για το δεύτερο χρήστη.Έτσι ως έξοδο η reduce έχει τον χρήστη και τον προτεινόμενο φίλο.

Η κλάση AggregateJob είναι η main class και αποτελεί τον συνδετικό κρίκο ανάμεσα στις άλλες δύο κλάσεις .Στην ουσία είναι η κλάση που “τρέχει” το πρόγραμμα. Δέχεται δύο arguments, ένα για το path του input και ένα για το path του output. Έτσι όταν τρέχουμε το πρόγραμμα στο command line πληκτρολογούμε :
$HADOOP_HOME/bin/hadoop jar <path of jar file> <path of input folder> <path of output folder>