HY562
Assignment 1
Μουστάκας Σεραφείμ
3334

Άσκηση 1:

Η άσκηση 1 έγινε με το τρόπο που ζητήθηκε στην άσκηση και για ευκολία ένωσα τα 2 υποερωτήματα σε ένα project οπότε συνολικά έγιναν 3 jobs.

Στο πρώτο job,αρχικά,μετράω τις λέξεις που περιέχονται με το κλασσικό WordCount. Δηλαδή,στο map βρίσκω όλες τις λέξεις και προσθέτω δίπλα το νούμερο 1. Έπειτα στο reduce phase προσθέτω τις ίδιες λέξεις μεταξύ τους και παράγω ένα αρχείο με όλες τι λέξεις και πόσες φορές εμφανίζονται

Στο δεύτερο job, στο map επιλέγω όλες τις λέξεις που έχουν συχνότητα εμφάνισης >4000 και στο reduce με τη βοήθεια ενός compare class κάνω Sort κατά φθίνουσα σειρά τα αποτελέσματα δίνοντας έμφαση στη συχνότητα εμφάνισης τους,δηλαδή στο value του ζευγαριού.

Το τρίτο job , συμπληρώθηκε στο κώδικα με σκοπό να λύσω και το πρώτο ερώτημα της άσκησης δηλαδή να εκτυπώνω τις 10 μεγαλύτερες σε συχνότητα λέξεις. Στο map βάζω σε ένα hashmap τις λέξεις και στο reduce phase γράφω σε ένα αρχείο τις 10 μεγαλύτερες αφού πρώτα κάνω sort το hashmap και πάρω τα 10 πρώτα στοιχεία του.

Άσκηση 2:

Χρόνοι εκτέλεσης

1)

10 reducers 0 combiner			
Job1		Job2	
Map	**Reduce**	**Map**	**Reduce**
174772	**244890**	**192082**	**211113**

2)

10 reducers 1 combiner			
Job1		Job2	
Map	**Reduce**	**Map**	**Reduce**

186322	**239625**	**204487**	**218262**

Συγκρίνοντας τους χρόνους παρατηρούμε ότι στο Job1 που είχα ενεργοποιημένο το combiner παρατηρώ μια μικρή αύξηση στο Map που είναι λογικό καθώς σε αυτό το χρόνο υπολογίζεται και η εκτέλεση του combiner αλλά στο reduce έχουμε μια μικρή μείωση που είναι λογική καθώς ο combiner έδωσε στο reducer πιο οργανωμένα τα ενδιάμεσα αποτελέσματα του map και έτσι τα επεξεργάστηκε πιο εύκολα.

3)

10 reducers 1 combiner and Compress			
Job1		Job2	
Map	**Reduce**	**Map**	**Reduce**
186392	**232886**	**206853**	**223413**

Σε αυτό το πίνακα, σε σύγκριση με τους προηγούμενους παρατηρούμε ότι η συμπίεση βοήθησε το πρώτο reducer αρκετά καθώς βλέπουμε μια μείωση στο χρόνο. Όμως και εδώ παρατηρώ ότι οι Mappers πήρανε παραπάνω χρόνο κάτι που είναι λογικό καθώς παίρνει επιπρόσθετο χρόνο το compress. Επίσης,ενδιαφέρον είναι να παρατηρήσουμε ότι ανέβηκε αρκετά ο χρόνος του reduce στο δεύτερο job που πιστεύω οφείλεται ότι έπρεπε να κάνει αποσυμπίεση σε ένα μικρό μέγεθος αρχείου οπότε υπήρξε κάποιο overhead. Οπότε, το συμπέρασμα μου σε αυτό το ερώτημα είναι ότι η συμπίεση των ενδιάμεσων δεδομένων βοηθάει μόνο αν έχουμε μεγάλο αριθμό δεδομένων.

4)

50 reducers			
Job1		Job2	
Map	**Reduce**	**Map**	**Reduce**
177479	**1088733**	**743143**	**1391505**

Έδω παρατήρησα ένα πολύ ενδιαφέρον αποτέλεσμα. Ενώ με απλή λογική θα περίμενε κανείς ότι περισσότεροι reducers θα κάνανε γρηγορότερα τη δουλειά που τους ανατέθηκε παρατηρώ ακριβώς το αντίθετο. Ο χρόνος εκτέλεσης εκτοξεύτηκε κυριολεκτικά σε ένα μέσο χρόνο εκτέλεσης τα 4 λεπτά. Έπειτα από σκέψη και έρευνα συνειδητοποίησα ότι είναι λογικό καθώς έχω ένα μικρό σε μέγεθος όγκο δεδομένων οπότε προσθέτοντας περισσότερους από 10 περίπου reducers δεν θα βελτιωθεί ο χρόνος. Ο λόγος είναι ότι πρέπει να γίνουν 50 initialise-setup των reducers και να γίνουν 50 προσπελάσεις σε αρχεία που από μόνα τους αυτά επιφέρουν ένα μεγάλο κάτω φράγμα εκτέλεσης του job. Συμπερασματικά, είναι χρήσιμο

να παρατηρούμε τον όγκο των δεδομένων μας και να υπολογίζουμε από πριν πόσους περίπου reducers θα χρειαστούμε.

Άσκηση 3

Σε αυτή την άσκηση δεν αντιμετώπισα κάποιο πρόβλημα. Την υλοποίησα με ένα job και αρχικά ξεκινώ και σε κάθε λέξη βάζω δίπλα της το αρχείο που τη βρήκα.

πχ. the file1
the file2
the file1
a file3
a file3

Έπειτα στο reducer η διαδικασία που ακολούθησα είναι ότι αφού έχω σαν κλειδί τη λέξη μπορώ να πάρω μια λίστα από αρχεία στα οποία υπάρχει σαν values οπότε για κάθε λέξη δημιούργησα ένα string που έχει τον δικό μου counter που μετράει τις λέξεις , τη λέξη, και δίπλα όλα τα αρχεία στα οποία αναφέρεται αλλά από μια φορά το καθένα καθώς μπορεί μια λέξη να υπάρχει πολλές φορές σε ένα αρχείο.

Όσο αναφορά το δεύτερο ερώτημα, ο δικός μου counter τον γράφω σε ένα text file στο τέλος της εκτέλεσης και μας δίνει τον αριθμό των λέξεων. Όμως τον ίδιο αριθμό μπορούμε να τον κάνουμε extract και από το ίδιο το hadoop και είναι ο REDUCER_INPUT_RECORDS.

Άσκηση 4

Παίρνοντας σαν βάση τη προηγούμενη άσκηση, προσέθεσα ένα combiner όπως μας ζητήθηκε ο οποίος μετράει πόσες φορές αναφέρθηκε από το map για μια λέξη ένα αρχείο.

πχ.

the file1 file1 file2 -> combiner -> the file1 #2 file2

Τέλος, ο reducer προσθέτει στην αρχή τον ID counter της κάθε λέξης.