Οικονομικό Πανεπιστήμιο Αθηνών
Τμήμα Πληροφορικής
Distributed Systems, Big Data and Cloud Computing
Φθινοπωρινό Εξάμηνο 2020-2021
Διδάσκων: Β. Καλογεράκη
Περιγραφή Εργασίας

Τα τελευταία χρόνια, λόγω της τεράστιας αύξησης της ποσότητας των πληροφορίας που έχουμε διαθέσιμης, μέσω των κινητών τηλεφώνων των χρηστών, υπάρχει δυνατότητα να γίνεται πολύ πιο εύκολα η ανάπτυξη εφαρμογών, κάτι το οποίο λίγο καιρό πριν θα ήταν αδύνατο χωρίς την αγορά εξειδικευμένου εξοπλισμού. Πλέον με το πλήθος των αισθητήρων που διαθέτει η κάθε συσκευή, υπάρχει ένα τεράστιο πλήθος πληροφορίας διαθέσιμο, και το πρόβλημα έχει μετατοπιστεί στο πως θα επεξεργαστεί και θα χρησιμοποιηθεί κατάλληλα αυτή η πληροφορία.

Για την αποθήκευση και επεξεργασία αυτού του μεγάλου όγκου πληροφορίας, τα τελευταία χρόνια, έχουν αναπτυχθεί πλήθος κατανεμημένων frameworks, τα οποία χρησιμοποιούν πολλούς “απλούς” υπολογιστές προκειμένου να αποθηκεύσουν και να επεξεργαστούν την πληροφορία, αντί να χρησιμοποιούν ένα “μεγάλο” και υψηλού κόστους server.

Στην παρούσα εργασία καλείστε να χρησιμοποιήσετε δύο τέτοια frameworks. Αρχικά το Hadoop, μέσω του οποίου μπορούμε να αποθηκεύσουμε μεγάλα αρχεία μοιράζοντας τα σε πολλούς υπολογιστές, το οποίο επίσης μας δίνει τη δυνατότητα να τρέξουμε διάφορα frameworks πάνω του. Στη συνέχεια θα γίνει χρήση του Apache Spark, το οποίο χρησιμοποιεί και επεκτείνει το Map-Reduce Programming paradigm, έτσι ώστε να προσφέρει στον χρήστη τη δυνατότητα για γράψει ένα πρόγραμμα το οποίο θα τρέξει σε όλους τους υπολογιστές του cluster, με τον πιο αποδοτικό τρόπο καθώς το Spark αναλαμβάνει τη οργάνωση και τον σωστό διαμοιρασμό της εργασίας στους διαθέσιμους υπολογιστές του cluster.

Το Spark, έχει εισάγει μια distributed δομή δεδομένων η οποία ονομάζεται RDD (Resilient Distributed Dataset). Όταν φορτώνουμε ένα αρχείο σε ένα RDD, αυτό κατανέμεται αυτόματα σε όλους workers που έχουμε διαθέσιμους. Στα RDDs εφαρμόζονται δύο ήδη πράξεων. Τα πρώτα λέγονται transformations και τα δεύτερα λέγονται actions. Τα transformations είναι πράξεις οι οποίες μας οδηγούν σε ένα νέο RDD ενώ τα actions κάνουν ένα υπολογισμό και μας γυρνάνε ένα αποτέλεσμα. Το Spark χρησιμοποιεί μια τεχνική που λέγεται lazy evaluation. Αυτό σημαίνει ότι τα transformations δε θα εκτελεστούν έως ότου να καλεστεί ένα action.

Το Spark υποστηρίζει 2 τύπους processing. Ο πρώτος λέγεται Batch και είναι ο υπολογισμός πάνω σε αρχεία που έχουμε αποθηκευμένα στο σύστημα μας (ή το cluster μας). Ο δεύτερος λέγεται Streaming και είναι δεδομένα που μας έρχονται από κάποια εξωτερική πηγή (Π.χ. ένα socket ή ένα πιο πολύπλοκο σύστημα π.χ. Kafka). Τέλος στις τελευταίες του εκδόσεις το Spark υποστηρίζει (και παροτρύνει το χρήστη να χρησιμοποιεί) μια δομή που λέγεται Dataframe. Τα Dataframes είναι και αυτά structured δομές δεδομένων, παρόμοιες με ένα table σχεσιακής βάσης. Τα Dataframes είναι υλοποιημένα με χρήση RDDs και οι προγραμματιστές του Spark παροτρύνουν τους χρήστες να χρησιμοποιούν αυτά αντί για τα RDDs καθώς με χρήση Dataframes, το Spark μπορεί και κάνει πολλά optimizations στην εκτέλεση χωρίς να χρειαστεί η παρέμβαση του χρήστη.

Για την υλοποίηση της εργασίας θα πρέπει να στήσετε τα frameworks σε ένα VM σε μια Linux based διανομή (Π.χ. Lubuntu) είτε σε Docker, πάνω στο οποίο ζητάμε να στήσετε πρώτα τα frameworks και στη συνέχεια θα τρέξετε την εφαρμογή σας πάνω του.

Σε αυτή την εργασία, πρέπει να υλοποιήσετε ένα σύστημα το οποίο χρησιμοποιώντας δεδομένα ταξί από την πόλη την Νέας Υόρκης, να μπορεί να απαντήσει σε διάφορα ερωτήματα με χρήση του Spark. Για την εργασία αυτή, σας δίνεται ένα dataset αποθηκευμένο σε μορφή csv και η ακριβής περιγραφή του βρίσκεται μετά την εκφώνηση της εργασίας.
Για την εργασία να κάνετε τις κατάλληλες ρυθμίσεις ώστε να δημιουργήσετε ένα Hadoop και Spark cluster.

Για την εργασία πρέπει να απαντηθούν τα παρακάτω ερωτήματα:

Ερωτήματα:

1) Έστω ότι χωρίζουμε την πόλη σε τεταρτημόρια. Να υπολογιστεί ο αριθμός των ταξί που ξεκίνησαν τις διαδρομές τους σε διαστήματα μιας μέρας σε κάθε τεταρτημόριο.

Παράδειγμα output:
{Day : 1, Area:1 }->300
...
{Day : 31, Area:2 }->500

2) Να βρεθεί το τεταρτημόριο από το οποίο ξεκίνησαν οι μεγαλύτερες σε διάρκεια διαδρομές. Στη συνέχεια να βρεθεί το τεταρτημόριο στο οποίο ξεκίνησαν οι μεγαλύτερες σε απόσταση διαδρομές. (Μέσοι όροι)

3) Να τυπωθούν οι διαδρομές οι οποίες ήταν μεγαλύτερες του ενός χιλιομέτρου και διάρκειας μεγαλύτερης των 10 λεπτών και με περισσότερους από δύο πελάτες (Batch και Steam)

4) Σε κάθε χρονικό διάστημα μιας ώρας, να βρεθεί πόσες κλήσεις ταξί έγιναν (Batch και Stream)

5) Με input από τον χρήστη, όπου θα δίνονται 2 ζεύγη συντεταγμένων και μια συγκεκριμένη ώρα, να υπολογίζεται πόσα ταξί βρίσκονται (ξεκίνησαν ή τελείωσαν τη διαδρομή τους) την τελευταία ώρα σε αυτή την περιοχή.

6)Για κάθε μέρα, βρείτε την ώρα που ο κάθε vendor είχε τις περισσότερες κούρσες. (π.χ. 13:00-14:00 -> 1000) (Batch και Stream)

7) Για τα σαββατοκύριακα τυπώστε ανά ώρα, πόσα ταξί καλέστηκαν. Τι παρατηρείτε;

8)Αφού κάνετε τα παραπάνω, μπορείτε να σκεφτείτε και να προσθέσετε ότι άλλο ενδιαφέρον στατιστικό νομίζετε εσείς.

Παραδοτέα:

Καλείστε να τρέξετε τα ερωτήματα σε distributed περιβάλλον (cluster). Τα 4 πρώτα ερωτήματα είναι σε batch μορφή. Για τα streaming ερωτήματα, μπορείτε για παράδειγμα να φτιάξετε ένα πρόγραμμα το οποίο θα διαβάζει γραμμή γραμμή το αρχείο και θα στέλνει τις γραμμές σε ένα socket.

Csv Column Description:

Data fields

- **id** - a unique identifier for each trip
- **vendor_id** - a code indicating the provider associated with the trip record
- **pickup_datetime** - date and time when the meter was engaged
- **dropoff_datetime** - date and time when the meter was disengaged
- **passenger_count** - the number of passengers in the vehicle (driver entered value)
- **pickup_longitude** - the longitude where the meter was engaged
- **pickup_latitude** - the latitude where the meter was engaged
- **dropoff_longitude** - the longitude where the meter was disengaged
- **dropoff_latitude** - the latitude where the meter was disengaged
- **store_and_fwd_flag** - This flag indicates whether the trip record was held in vehicle memory before sending to the vendor because the vehicle did not have a connection to the server - Y=store and forward; N=not a store and forward trip
- **trip_duration** - duration of the trip in seconds