ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΚΑΤΑΝΕΜΗΜΕΝΑ ΣΥΣΤΗΜΑΤΑ

Εαρινό Εξάμηνο 2016-2017

Υποχρεωτική εργασία

Τα τελευταία χρόνια, λόγω της τεράστιας αύξησης της ποσότητας της πληροφορίας που έχουμε διαθέσιμη, μέσω των πληροφορικών συστημάτων, η ιδέα του να επεξεργαζόμαστε τα δεδομένα σε μεγάλους και υψηλού κόστους servers έχει αρχίσει να εγκαταλείπεται και πλέον αντικαθίσταται από decentralized συστήματα. Το πλεονέκτημα αυτών των συστημάτων έγκειται στο υψηλό scalability που έχουν (ο διαχειριστής μπορεί εύκολα να προσθέσει/αφαιρέσει κόμβους από το σύστημα, ανάλογα με το πλήθος της πληροφορίας που θέλει να επεξεργαστεί), καθώς και στο πολύ μειωμένο κόστος σε σχέση με τη μίσθωση ή την αγορά ενός ακριβού dedicated server.

Στα συστήματα αυτά, κάθε κόμβος είναι υπεύθυνος να διαχειρίζεται ένα εύρος δεδομένων από το σύνολο δεδομένων του συστήματος. Έτσι, κάθε φορά που του ζητηθεί κάποιου είδους πληροφορία (όπως πχ μια διαδρομή μέσα σε μια πόλη), αυτός την επαναφέρει από την μνήμη του και την προωθεί κατάλληλα. Η διαδικασία αυτή όμως μπορεί να προσθέσει μεγάλο overhead, καθώς αυτή η διαθέσιμη πληροφορία μπορεί να ζητείται πολύ συχνότερα από τους χρήστες του συστήματος. Κατα συνέπεια, θα χρειάζεται να επανακτάται συνεχώς από το δίσκο του κάθε κόμβου, κάτι το οποίο δεν είναι επιθυμητό. Για να λύσουμε αυτό το πρόβλημα χρησιμοποιούμε memcached based συστήματα, τα οποία είναι υπεύθυνα να διατηρούν ένα υποσύνολο αυτής της πληροφορίας σε κάποιον κεντρικό κόμβο και να την επιστρέφουν κατάλληλα στους χρήστες όταν ζητηθεί.

Στην παρούσα εργασία καλείστε να λύσετε αυτό το πρόβλημα υλοποιώντας ένα memcached based σύστημα με χρήση του **Map Reduce Framework** [1], στο οποίο η πληροφορία που θέλουμε να αντλούμε είναι οι διαδρομές που ζητούν οι χρήστες μέσα σε μια πόλη. Ο λόγος που θέλουμε να υλοποιήσουμε ένα τέτοιο σύστημα είναι ο περιορισμός στον αριθμό των queries που μπορούμε να κάνουμε ανά ημέρα μέσω του Google Directions API. Οι διαδρομές που ζητώνται συχνά από τους χρήστες θα αποθηκεύονται σε ένα κομμάτι της μνήμης RAM του κεντρικού κόμβου, έτσι ώστε να είναι δυνατή η άμεση προσπέλαση αλλά και η επιστροφή στο κινητό του χρήστη που τις ζητά.

Η ανάπτυξη του συστήματος θα σας επιτρέψει να εξοικειωθείτε τόσο με την έννοια του MapReduce, όσο και με το να γνωρίσετε το περιβάλλον Android. Για την διευκόλυνσή σας, η ανάπτυξη αυτού του συστήματος θα γίνει σε δύο φάσεις: Η πρώτη φάση αφορά ***την υλοποίηση του MapReduce Framework που θα είναι υπεύθυνο να υλοποιήσει τον αλγόριθμο της επεξεργασίας των δεδομένων***. Η

δεύτερη φάση αφορά ***τη δημιουργία μιας εφαρμογής Android, η οποία μέσω των κατάλληλων διεπαφών που θα έχετε φροντίσει να υλοποιήσετε στην πρώτη φάση, θα μπορεί να καλεί το framework, έτσι ώστε να εκτελείται ο αλγόριθμος ανεύρεσης των διαδρομών, και στη συνέχεια αυτό το αποτέλεσμα να οπτικοποιείται από το κινητό***. Παρακάτω ακολουθούν λεπτομερείς οδηγίες για την κάθε φάση και τι καλείστε να υλοποιήσετε σε αυτές.

MapReduce

Το MapReduce framework είναι ένα προγραμματιστικό μοντέλο που επιτρέπει την παράλληλη επεξεργασία μεγάλων όγκων δεδομένων. Το MapReduce στηρίζεται στη χρήση δύο συναρτήσεων:

map(key,value) -> [(key2, value2)]
reduce(key2,[value2]) -> [final_value]

- "Map" συνάρτηση: επεξεργάζεται ένα ζεύγος key/value και παράγει ένα ενδιάμεσο ζεύγος key/value. Η είσοδος στη συνάρτηση map μπορεί να είναι γραμμές ενός αρχείου κλπ., και έχουν τη μορφή (κλειδί, τιμή). Η συνάρτηση map μετατρέπει κάθε τέτοιο ζευγάρι σε ένα άλλο ζευγάρι (κλειδί2, τιμή2). Η map συνάρτηση μπορεί να εκτελείται παράλληλα, πάνω σε διαφορετική είσοδο δεδομένων και σε διαφορετικούς κόμβους. Ο βαθμός παραλληλίας εξαρτάται από την εφαρμογή και μπορεί να την ορίσει ο χρήστης.
- "Reduce" συνάρτηση: συγχωνεύει όλα τα ενδιάμεσα values που σχετίζονται με το ίδιο κλειδί και παράγει τα τελικά αποτελέσματα. Για κάθε ξεχωριστό κλειδί δημιουργείται μια λίστα από τις τιμές που αντιστοιχούν σε αυτό το κλειδί. Η συνάρτηση αυτή υπολογίζει μια τελική τιμή για το κλειδί, επεξεργάζοντας τη λίστα των τιμών που αντιστοιχούν σε αυτό το κλειδί. Η επεξεργασία της συνάρτησης reduce γίνεται αφού έχει τελειώσει η επεξεργασία όλων των map συναρτήσεων.

Το πλεονέκτημα του MapReduce συστήματος είναι ότι επιτρέπει την κατανεμημένη επεξεργασία των εργασιών μέσω των συναρτήσεων “map” και “reduce”. Υπό την προϋπόθεση ότι κάθε “map” συνάρτηση είναι ανεξάρτητη από την άλλη, όλες οι “map” συναρτήσεις μπορούν να εκτελούνται παράλληλα, αν και στην πράξη περιορίζεται από την πηγή των δεδομένων ή τον αριθμό των επεξεργαστών για αυτά τα δεδομένα. Με τον ίδιο τρόπο, θα μπορούσαμε να είχαμε και πολλές “reduce” συναρτήσεις, το μόνο που απαιτείται είναι ότι όλα τα αποτελέσματα των “map” συναρτήσεων που μοιράζονται το ίδιο κλειδί, να πηγαίνουν στην ίδια “reduce” συνάρτηση. Οι χρήστες αυτών των συστημάτων μπορούν να εκμεταλλευθούν κατανεμημένους υπολογιστικούς πόρους για την γρήγορη επεξεργασία μεγάλου όγκου δεδομένων. Επίσης, ο παραλληλισμός προσφέρει κάποια δυνατότητα ανάκαμψης από μερική αποτυχία των κόμβων κατά τη διάρκεια της λειτουργίας: εάν ένας map ή reduce κόμβος πέσει/αποτύχει και αν τα δεδομένα είναι ακόμα διαθέσιμα, η εργασία μπορεί να επαναπρογραμματιστεί σε κάποιο άλλο διαθέσιμο κόμβο.

Για τις ανάγκες του map reduce χρειάζεται να κατασκευάσουμε τρία components, *τον master node, τους worker nodes* και *τον reducer node*.

1. **Master node:** Είναι υπεύθυνος να γνωρίζει ποιοι είναι οι υπάρχοντες κόμβοι στο δίκτυο. Ο κόμβος αυτός κρατάει στη μνήμη του τις τελευταίες διαδρομές

που έχουν ζητηθεί (π.χ. τις 100 τελευταίες διαδρομές). Όταν αυτή η μνήμη γεμίσει, τότε αφαιρείται από τη μνήμη η πιο παλιά εγγραφή. Όταν ζητηθεί μια διαδρομή από τον κόμβο αυτό (και αυτή δεν υπάρχει στην μνήμη cache), τότε ο master node ζητάει από τους workers να ελέγξουν για το αν υπάρχει κάποια σχετική διαδρομή στη βάση του συστήματος. Κάθε ένας από αυτούς ενημερώνει ξεχωριστά τον master όταν ολοκληρώσει την εργασία του. Μόλις ολοκληρώσουν όλοι οι workers, δίνει εντολή στον reducer να ξεκινήσει την διαδικασία του reduce και περιμένει να του επιστρέψει όσες πιθανές διαδρομές βρέθηκαν. Όταν πάρει τη λίστα από αυτές τις διαδρομές, ελέγχει ποια από αυτές βρίσκεται πιο κοντά στα σημεία αφετηρίας **και** προορισμού που έχει ζητήσει ο χρήστης (με τη χρήση της Ευκλείδιας απόστασης[2] της τοποθεσίας του χρήστη και των σημείων αφετηρίας και προορισμού αντίστοιχα) και του την προωθεί ανάλογα. Αν ο reducer node δεν επιστρέψει αποτελέσματα, τότε ο master node ρωτάει στο Google Directions API έτσι ώστε να μάθει τη διαδρομή. Έπειτα, αποθηκεύει τη διαδρομή στην cache και την προωθεί στον worker που είναι υπεύθυνος για την περιοχή αυτή έτσι ώστε αυτός να την περάσει στην βάση δεδομένων.

2. **Worker Nodes:** Οι κόμβοι αυτοί είναι υπεύθυνοι για ένα εύρος από τις αποθηκευμένες διαδρομές. Προκειμένου κάθε κόμβος να αναλαμβάνει περίπου ίσο όγκο δεδομένων για να αναζητήσει διαδρομές, κάθε διαδρομή που υπάρχει μέσα στη βάση, είναι αποθηκευμένη στη μορφή<(Source LatLng, Destination LatLng),DirectionsObject>, όπου τα source και destination LatLng θα είναι ακρίβειας 2 δεδαδικών ψηφίων. (Αυτό αντιστοιχεί σε ακρίβεια 1.11Km). Στη συνέχεια, χρησιμοποιώντας μια hash function(π.χ. SHA1 ή MD5) παίρνουμε το Hash του String SourceLatLng+DestinationLatLng και τα hash των IP+Port του κάθε Mapper. Έτσι ο κάθε Mapper θα είναι υπεύθυνος για όσα hashes εγγραφών είναι μικρότερα από το hash του IP+Port του. (Προσοχή για το ποια hashes αντιστοιχούν στον Mapper με το μικρότερο hash, θα χρειαστεί η χρήση mod). Αυτοί οι κόμβοι διαβάζουν από την βάση τις αποθηκευμένες διαδρομές, φιλτράρουν τις διαδρομές για τις οποίες είναι υπεύθυνοι και στη συνέχεια ομαδοποιούν τις διαδρομές που ξεκινάνε και τελειώνουν σε κοντινά σημεία με βάση την τοποθεσία που έχει δώσει ο χρήστης. Ουσιαστικά, οι worker nodes εκτελώντας τη διαδικασία του map παράγουν tuples που είναι της μορφής {(Source LatLng, Destination LatLng),DirectionsObject}. Καθένας προωθεί τα αποτελέσματα του στον Reducer Node και ενημερώνει τον master ότι ολοκλήρωσε την εργασία του.
3. **Reducer Node:** Ο Reducer Node δέχεται από τους worker nodes tuples της μορφής: <(Source LatLng, Destination LatLng),DirectionsObject>. Όταν λάβει το κατάλληλο σήμα από τον master ξεκινά τη διαδικασία του reduce. Ο ρόλος του είναι να ομαδοποιήσει όσες διαδρομές έχουν τα ίδια (Source LatLng, Destination LatLng) και έτσι να φτιάξει ένα αντικείμενο της μορφής <(Source LatLng, Destination LatLng),ListOfDirectionObjects>. Αυτό θα επιστραφεί στον master node κι εκείνος θα επιστρέψει την κατάλληλη διαδρομή.

Υλοποίηση κατανεμημένης επεξεργασίας χρησιμοποιώντας το MapReduce framework [1] πάνω από Java 8 [3]. Ο τρόπος λειτουργίας του MapReduce θα γίνεται ως εξής:

1. Σε κάθε worker κόμβο θα υπάρχει ένα instance της εφαρμογής ανεύρεσης των διαδρομών.
2. Κάθε instance (master node, worker nodes, reducer node) θα ακούει σε κάποιο προκαθορισμένο port για συνδέσεις.
3. Όταν λάβει ο master node ένα query από τον χρήστη, αναζητά την διαδρομή που είναι πιο κοντά στα σημεία εκκίνησης **και** τέλους του query του χρήστη. Αν την έχει ήδη στην μνήμη την επιστρέφει αμέσως. Αν δεν την έχει στην μνήμη, δίνει εντολή στους workers να ξεκινήσουν την διαδικασία του map. Αναμένει να ολοκληρώσουν όλοι την διαδικασία του map και στην συνέχεια ενημερώνει τον reducer να ξεκινήσει την διαδικασία του reduce. Όταν ολοκληρωθεί η διαδικασία, αν υπάρχουν αποτελέσματα τότε διαλέγει το πιο κοντινό στον χρήστη και του το επιστρέφει (ώστε να το οπτικοποιήσει στο κινητό του). Αν δεν έχει, επιστρέφει κατάλληλο μήνυμα με το οποίο ενημερώνει τον χρήστη αν επιθυμεί να γίνει κλήση στο Google Directions API για να πάρει την διαδρομή και να την αποθηκεύσει στη RAM του. Αν γίνει κλήση στο Google Directions API, τότε ο master ενημερώνει κατάλληλα τον αντίστοιχο worker να την περάσει στη βάση δεδομένων.
4. Κάθε worker, θα εξάγει τα δεδομένα για το συγκεκριμένο εύρος δεδομένων για το οποίο είναι υπεύθυνος.
5. Θα χρησιμοποιήσετε το MapReduce framework της Java 8 ώστε να επεξεργαστείτε αυτά τα δεδομένα παράλληλα [3]. Κάθε map εργασία αναζητά στις διαδρομές για τις οποίες είναι υπεύθυνη και θα τις επεξεργάζεται κατάλληλα ώστε να εξάγει τα ενδιάμεσα ζεύγη key/value pairs. Στη δική μας περίπτωση κάθε κλειδί αναφέρεται σε ένα ζεύγος Latitude και Longitude και το value είναι το DirectionsObject.
6. Στη συνέχεια τα ενδιάμεσα ζεύγη key/value pairs επεξεργάζονται από τη reduce συνάρτηση η οποία θα αναλάβει να κάνει το grouping με βάση το latitude και longitude του χρήστη και θα επιστρέφει την λίστα με τις διαδρομές στον master node.
7. Όταν ο master node παραλάβει την λίστα επιλέγει την διαδρομή η οποία είναι πιο κοντά στον χρήστη και την επιστρέφει, ειδάλλως ακολουθείται η διαδικασία που περιγράψαμε στο βήμα 3.

Android Application

Την υλοποίηση μίας εφαρμογής που θα εκτελείται σε συσκευές με λειτουργικό Android και θα εμφανίζει την διαδρομή που ζητά ο χρήστης. Η εφαρμογή θα υλοποιηθεί στην πλατφόρμα Android και θα επωφελείται από το Map Reduce framework το οποίο θα τρέχει ανεξάρτητα. Η εφαρμογή android θα εκτελείται ώς εξής:

1. Η βασική οθόνη της εφαρμογής αυτής θα περιλαμβάνει μια οθόνη Google Maps. Πάνω σε αυτή την οθόνη ο χρήστης θα μπορεί να επιλέξει το σημείο προορισμού του και να ζητήσει οδηγίες για εκεί.
2. Με το που γίνει αυτό, η εφαρμογή θα απευθύνεται στον master node ζητώντας τη διαδρομή και δίνοντας σαν παράμετρο την γεωγραφική θέση του χρήστη.

3. Ακολουθείται η διαδικασία που περιγράψαμε στο κομμάτι της υλοποίησης της κατανεμημένης επεξεργασίας. Αν τυχόν, η διαδρομή δεν υπάρχει, τότε επιστρέφεται κατάλληλο μήνυμα («ότι δεν βρέθηκε») στο κινητό και παράλληλα ζητά την συνδρομή του χρήστη στο αν επιθυμεί να γίνει η ανάκτηση μέσω του Google Directions API.
4. Με το που λάβει τη διαδρομή από τον master πρέπει να "ζωγραφίσει τη διαδρομή πάνω στο χάρτη.

Παραδοτέα εργασίας:

Το project θα παραδοθεί σε δύο φάσεις:

Παραδοτέο Α: (Ημερομηνία παράδοσης: 23/4/2017)

Στο παραδοτέο αυτό, θα πρέπει να έχετε ολοκληρώσει εντελώς το memcached based σύστημα με τη χρήση του map-reduce framework, όπως ακριβώς σας έχει ζητηθεί, έτσι ώστε να μπορεί να χρησιμοποιηθεί στην επόμενη φάση της εργασίας του μαθήματος.

Παραδοτέο Β: (Ημερομηνία παράδοσης: 2/6/2017)

Το παραδοτέο αυτό αποτελεί το android application, που περιγράφηκε πριν. Στη φάση αυτή το σύστημα θα πρέπει να είναι πλήρως λειτουργικό, με όλα τα components του να λειτουργούν σωστά.

Ομάδες: Όλοι οι φοιτητές θα πρέπει να σχηματίσουν ομάδες των τριών (3) ή τεσσάρων (4) ατόμων προκειμένου να εκπονήσουν την προγραμματιστική τους εργασία. Γλώσσα προγραμματισμού θα είναι η Java, στην οποία και θα παρέχεται υποστήριξη από τους βοηθούς του μαθήματος.

Αναφορές – Χρήσιμοι Σύνδεσμοι:

[1] MapReduce : Simplified Data Processing on Large Clusters, OSDI 2004, San Fransisco, CA http://labs.google.com/papers/mapreduce-osdi04.pdf

[2] Ευκλείδια απόσταση μεταξύ δύο σημείων στον χώρο: https://en.wikipedia.org/wiki/Euclidean_distance#Two_dimensions

[3] https://docs.oracle.com/javase/8/docs/api/java/util/stream/package-summary.html

[4] Android. URL : http://code.google.com/android/http://code.google.com/android/

[5] Android SDK: http://developer.android.com/sdk/index.html

[6] Android Studio http://developer.android.com/sdk/index.html