ΑΛΕΞΑΝΔΡΕΙΟ Τ.Ε.Ι. ΘΕΣΣΑΛΟΝΙΚΗΣ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Υλοποίηση αλγορίθμων ταξινόμησης μεγάλου όγκου δεδομένων με το μοντέλο MapReduce

Του φοιτητή

Γαβριηλίδη Αχιλλέα

Αρ. Μητρώου: 07/3257

Επιβλέπων καθηγητής

Κωνσταντίνος Διαμαντάρας

Θεσσαλονίκη 2013

ΠΕΡΙΛΗΨΗ

Η παρακάτω πτυχιακή εργασία ασχολείται με το project ανοιχτού κώδικα Apache Hadoop. Αναφέρεται το πώς ξεκίνησε, από ένα paper της Google μέχρι την σημερινή άνθηση σε όλους τους τομείς λόγω της ανάγκης για επεξεργασία του όγκου δεδομένων που μέχρι τώρα ήταν ανεκμετάλλευτος και φυσικά λόγω των επιδόσεων που προσφέρει στην επεξεργασία αυτών των δεδομένων ειδικά αν αναλογιστεί κανείς την αναλογία επιδόσεις/κόστος. Έτσι, πλέον υπάρχουν πολλές επιλογές και διανομές για το Hadoop και εκατοντάδες χρήσεις. Είναι σχεδόν σίγουρο ότι οποίας φύσης και να είναι τα δεδομένα, με την χρήση του Hadoop θα υπάρχει βελτιστοποίηση στην αποστολή μίας επιχείρησης. Αυτό το πετυχαίνει με την προσαρμοστικότητα, την εύκολη εγκατάσταση και διαχείριση ενός cluster (ειδικά με την χρήση κάποιας διανομής), αλλά και με την αρχιτεκτονική του. Επίσης, θα αναφέρεται στην μηχανική μάθηση, τους τύπους που υπάρχουν και συγκεκριμένα για τους αλγορίθμους ταξινόμησης. Τέλος, στα πλαίσια της πτυχιακής εφαρμόστηκε ο Naive Bayes ταξινομητής, παρεχόμενος από το Apache Mahout σε ένα εικονικό Hadoop cluster και αναλύθηκαν παράλληλα οι ευκολίες και οι δυσκολίες ανάπτυξης για το Hadoop.

ABSTRACT

The following dissertation is about the open-source project Apache Hadoop. It begins from the origin of Hadoop, from a Google paper and how it flourished in every field, due to the need for the processing of all the Big Data, that was left untapped until now and due to the performance it offers, especially if you factor in the performance/cost ratio. Thanks, to this growth there plenty of options and distributions for Hadoop now and hundreds of uses. It is almost certain that regardless of the nature of the data, with the use of Hadoop, there will be a significant improvement in a company's mission. Hadoop achieves this with its adjustability, its ease of deployment and management (especially with the use of a distribution) and its unique architecture. It is also about machine learning, its types and specifically about classification algorithms. Finally, in the context of this dissertation, the Naive Bayes classifier was applied in a virtual Hadoop cluster, which is provided from Apache Mahout and I analyzed the pros and cons of the development for Hadoop.

ΠΕΡΙΕΧΟΜΕΝΑ

Ευρετήριο σχημάτων

Ευρετήριο πινάκων

ΕΙΣΑΓΩΓΗ

Το Apache Hadoop γεννήθηκε από την ανάγκη μας να επεξεργαστούμε ένα τεράστιο όγκο δεδομένων . Ο ιστός παράγει όλο και περισσότερες πληροφορίες σε καθημερινή βάση, και είχε γίνει πολύ δύσκολο να δεικτοδοτηθούν πάνω από ένα δισεκατομμύριο σελίδες περιεχομένου . Για να αντιμετωπιστεί αυτό, η Google επινόησε ένα νέο στυλ της επεξεργασίας των δεδομένων το γνωστό ως MapReduce . Ένα χρόνο μετά η Google δημοσίευσε ένα paper που περιέγραφε το πλαίσιο MapReduce. Οι Doug Cutting και ο Mike Cafarella , εμπνευσμένοι από το paper, δημιούργησαν το Hadoop για να εφαρμόσουν αυτές τις έννοιες σε ένα πλαίσιο λογισμικού ανοικτού κώδικα για να υποστηρίξουν την διανομή για το project της μηχανής αναζήτησης Nutch . Με δεδομένο τον αρχικό σχεδιασμό, το Hadoop σχεδιάστηκε με μία απλή write-once υποδομή αποθήκευσης.

Το Hadoop έχει προχωρήσει πολύ πέρα από το ξεκίνημά του στην ευρετηρίαση του web και πλέον χρησιμοποιείται σε πολλές βιομηχανίες για μια τεράστια πληθώρα εργασιών που απαιτείται καθημερινά, και απαιτεί ποικιλία τύπων δεδομένων, τεράστιο όγκο και ταχύτητα (νέα δεδομένα πολύ συχνά) δεδομένων - τόσο δομημένων όσο και αδόμητων. Πλέον χρησιμοποιείται ευρέως σε διάφορους κλάδους, συμπεριλαμβανομένων των οικονομικών, των μέσων της ενημέρωσης και της ψυχαγωγίας, την κυβέρνηση, την υγειονομική περίθαλψη, τις υπηρεσίες πληροφοριών, το λιανικό εμπόριο και άλλες βιομηχανίες με απαιτήσεις μεγάλου όγκου δεδομένων, αλλά οι περιορισμοί της αρχικής υποδομής αποθήκευσης παραμένουν.

Το Hadoop είναι όλο και περισσότερο το framework επιλογής για εφαρμογές μεγάλης κλίμακας και υψηλών απαιτήσεων σε δεδομένα. Το Hadoop είναι χτισμένο για την επεξεργασία μεγάλου όγκου δεδομένων από Terabyte σε Petabyte έως και Exabyte. Με τόσα πολλά δεδομένα, είναι απίθανο ότι θα χωρούσαν σε ένα σκληρό δίσκο ενός και μόνο υπολογιστή, ή ακόμη χειρότερα στη μνήμη. Η ομορφιά του Hadoop είναι ότι έχει σχεδιαστεί για να επεξεργαστεί αποτελεσματικά τεράστιες ποσότητες δεδομένων από τη σύνδεση πολλών υπολογιστών χαμηλού κόστους μαζί για να εργάζονται παράλληλα. Χρησιμοποιώντας το μοντέλο MapReduce, το Hadoop μπορεί να δεχτεί προς εκτέλεση ένα ερώτημα σε ένα σύνολο δεδομένων, να το διαιρέσει, και να το τρέξει παράλληλα σε πολλαπλούς κόμβους. Η διανομή του υπολογισμού λύνει το πρόβλημα της ύπαρξης δεδομένων που είναι πάρα πολύ μεγάλα σε όγκο για να χωρέσουν σε ένα μόνο μηχάνημα .

Το Apache Hadoop περιλαμβάνει ένα κατανεμημένο σύστημα αρχείων το HDFS(Hadoop Distributed File System), που σπάει τα δεδομένα εισόδου και τα αποθηκεύει στους κόμβους υπολογισμού. Αυτό καθιστά δυνατό για τα δεδομένα να επεξεργαστούν παράλληλα χρησιμοποιώντας όλα τα μηχανήματα του cluster. Το Apache Hadoop Distributed File System είναι γραμμένο σε Java και μπορεί να τρέξει σε διαφορετικά λειτουργικά συστήματα.

Το Hadoop σχεδιάστηκε από την αρχή για να φιλοξενήσει πολλαπλές εφαρμογές συστημάτων αρχείων και υπάρχουν αρκετά διαθέσιμα. Το HDFS και το σύστημα αρχείων S3 που χρησιμοποιείται στην Cloud πλατφόρμα της Amazon είναι ίσως τα πιο ευρέως χρησιμοποιημένα, αλλά υπάρχουν και πολλά άλλα, όπως το maprfs της MapR που είναι μάλιστα από τα καλύτερα.

Οι άνθρωποι και οι ευφυείς οργανισμοί έχουν την ικανότητα να ταυτοποιούν πραγματικά δεδομένα χρησιμοποιώντας τις αισθήσεις τους και την αντιληπτική τους ικανότητα προκειμένου να λάβουν τις κατάλληλες αποφάσεις ώστε να επιβιώσουν στο περιβάλλον τους. Μία μηχανή, όπως ένας ηλεκτρονικός υπολογιστής, πρέπει να εκπαιδευθεί κατάλληλα ώστε να αναγνωρίζει πρότυπα (patterns) και να τα κατηγοριοποιεί αυτόματα σε κατηγορίες. Ανάλογα με την εφαρμογή, γίνεται κατάταξη των αντικειμένων σε κλάσεις με τη βοήθεια αλγορίθμων ταξινόμησης.

Το Apache Hadoop, είναι η πιο κατάλληλη πλατφόρμα για εκτέλεση αλγορίθμων ταξινόμησης όπως ο Naive Bayes, σε τεράστιους όγκους δεδομένων, χάρη στην ευελιξία, επεκτασιμότητα, διαθεσιμότητα και ευκολία στην διαχείριση που παρέχει, αλλά κυρίως χάρη στην επεξεργαστική ικανότητα του και την σχεδόν ίση κατανομή των πόρων μειώνοντας έτσι την πολυπλοκότητα των αλγορίθμων, αλλά και τον χρόνο εκτέλεσης τέτοιας φύσης.

Ένα χρήσιμο εργαλείο για την επίτευξη αυτού, εκτός από το Apache Hadoop είναι το Apache Mahout το οποίο προσφέρει επεκτάσιμους αλγορίθμους μηχανικής μάθησης οι οποίοι εκμεταλλεύονται το μοντέλο του MapReduce και είναι πολύ αποδοτικοί ακόμη και σε χαμηλού κόστους εξοπλισμό. Βέβαια, υπάρχουν πολλές δυσκολίες σε αυτό το «ταξίδι» του Apache Hadoop, κυρίως λόγω έλλειψης προσβάσιμης γνώσης, αλλά είναι σίγουρα κάτι που αξίζει να ασχοληθεί κανείς.

ΚΕΦΑΛΑΙΟ 1

Apache Hadoop.

ΕΙΣΑΓΩΓΗ

Το Apache Hadoop είναι μία πλατφόρμα λογισμικού ανοικτού κώδικα, για αποθήκευση και επεξεργασία μεγάλου όγκου δεδομένων σε ομάδα Η/Υ (cluster), με χαμηλό κόστος, συγκριτικά με άλλες πλατφόρμες που απαιτούν ακριβό εξοπλισμό. Επιπλέον, είναι πολύ εύκολα επεκτάσιμο προσθέτοντας στο cluster νέους κόμβους και «αυτόματα» παραλληλήσιμο. Το Hadoop είναι ένα από τα project του Apache Software Foundation με παγκόσμια κοινότητα και πλήθος χρηστών, και έχει την άδεια Apache License 2.0.

Το Apache Hadoop αποτελείται από τις ακόλουθες μονάδες:

- Hadoop Common – περιέχει τις βιβλιοθήκες και τα βοηθητικά προγράμματα που απαιτούνται από τις υπόλοιπες μονάδες του Hadoop.

- Hadoop Distributed File System (HDFS) – ένα κατανεμημένο σύστημα αρχείων που αποθηκεύει δεδομένα στους κόμβους του cluster, παρέχοντας πολύ υψηλό συνολικό εύρος ζώνης, στο σύνολο του cluster.
- Hadoop YARN – μία πλατφόρμα διαχείρισης των πόρων, υπεύθυνη για την διαχείριση των υπολογιστικών πόρων στο cluster και για την χρονοδρομολόγηση των εφαρμογών των χρηστών.
- Hadoop MapReduce – ένα προγραμματιστικό μοντέλο για επεξεργασία μεγάλου όγκου δεδομένων.

Όλες οι μονάδες του Hadoop έχουν σχεδιαστεί με μία θεμελιώδη υπόθεση, ότι οποιαδήποτε αστοχία υλικού είναι συνήθης φαινόμενο και γι' αυτό αντιμετωπίζεται αυτόματα από το λογισμικό του Hadoop.

ΥΠΟΚΕΦΑΛΑΙΟ 1.1

Αναδρομή και ξεκίνημα της τεχνολογίας

Οι μονάδες MapReduce και HDFS του Apache Hadoop, προέρχονται από τις δημοσιεύσεις της Google, MapReduce, Google File System (GFS) και BigTable. Το GFS, είναι ένα κατανεμημένο σύστημα αρχείων που εξελίχθηκε από την Google για προσωπική της χρήση στα συστήματα της, από το οποίο και προέρχεται το HDFS.

Ένα βασικό χαρακτηριστικό και των δύο συστημάτων αρχείων, είναι ο σχεδιασμός τους να παρέχουν αποτελεσματική και αξιόπιστη πρόσβαση στα δεδομένα, χρησιμοποιώντας μεγάλα cluster αποτελούμενα από χαμηλού κόστους υλικό. Το BigTable είναι ένα συμπιεσμένο, υψηλής απόδοσης σύστημα αποθήκευσης δεδομένων, ενσωματωμένο στο GFS και άλλες τεχνολογίες της Google και η ανάπτυξη του ξεκίνησε από το 2004, και χρησιμοποιείται μέχρι και σήμερα σε πολλά προϊόντα της Google. Μερικά από αυτά είναι το Google Maps, το Google Earth, το YouTube, το Gmail κ.α. Ο λόγος για ανάπτυξη του BigTable αντί για την

χρησιμοποίηση υπαρχόντων βάσεων δεδομένων είναι ότι το BigTable είναι βελτιστοποιημένο για πολύ μεγάλο όγκο δεδομένων και για «αυτόματη» επέκταση των κόμβων και εκμετάλλευση αυτών χωρίς την εκ νέου ρύθμιση από την αρχή και μπορεί να περιγραφεί ως ένας σποραδικός, κατανεμημένος και πολυδιάστατος ταξινομημένος χάρτης. Η αντίστοιχη μονάδα του BigTable στο οικοσύστημα του Apache Hadoop είναι το Apache HBase. Το MapReduce και αυτό αναπτύχθηκε αρχικά από την Google, για επεξεργασία μεγάλου όγκου δεδομένων, με ένα παράλληλο και κατανεμημένο αλγόριθμο πάνω σε ένα cluster. Από τότε όμως έχουν γραφτεί πολλές βιβλιοθήκες σε πολλές γλώσσες η κάθε μία με διαφορετικό επίπεδο βελτιστοποίησης. Μία από αυτές είναι και η ανοικτού κώδικα υλοποίηση MapReduce ως μέρος του Apache Hadoop.

ΥΠΟΚΕΦΑΛΑΙΟ 1.2

Τρόπος λειτουργίας MapReduce

Ένα πρόγραμμα MapReduce, σε όποια μορφή και να βρίσκεται αποτελείται από δύο μεθόδους, την μέθοδο Map() και την μέθοδο Reduce(). Η μέθοδος Map() είναι υπεύθυνη για το φιλτράρισμα και την ταξινόμηση των δεδομένων σε ξεχωριστές ουρές προς επεξεργασία και η μέθοδος Reduce() είναι η υπεύθυνη για την συλλογή των δεδομένων και την πραγματοποίηση μιας τελικής επεξεργασίας συνδυάζοντας τα ξεχωριστά αποτελέσματα από τις ουρές (π.χ. μία τελική πρόσθεση των δεδομένων). Γενικότερα, η πλατφόρμα (σύστημα) μιας υλοποίησης MapReduce λειτουργεί οργανώνοντας τους κατανεμημένους κόμβους, εκτελώντας τις διάφορες διεργασίες παράλληλα και διαχειρίζοντας όλες τις επικοινωνίες και την μεταφορά των δεδομένων ανάμεσα στα διάφορα μέρη του συστήματος, παρέχοντας παράλληλα πλεονασμό πόρων (redundancy) και ανοχή σφαλμάτων (fault tolerance).

Αυτό το μοντέλο προγραμματισμού είναι εμπνευσμένο από τις συναρτήσεις map και reduce που χρησιμοποιούνται στον συναρτησιακό προγραμματισμό αλλά έχουν διαφορετικό σκοπό στο MapReduce framework. Το MapReduce είναι μια πλατφόρμα ανάπτυξης (framework) για επεξεργασία παραλληλίσιμων προβλημάτων πάνω σε τεράστιο όγκο δεδομένων, χρησιμοποιώντας μεγάλο αριθμό υπολογιστικών κόμβων, είτε είναι σε μορφή cluster (αν βρίσκονται στο ίδιο δίκτυο και χρησιμοποιούν παρόμοιο υλικό) , είτε είναι σε μορφή grid (αν οι κόμβοι βρίσκονται μοιρασμένοι σε κατανεμημένα διαχειριστικά και χωροταξικά συστήματα και χρησιμοποιούν πιο ετερογενή υλικό). Η υπολογιστική επεξεργασία μπορεί να πραγματοποιηθεί είτε σε αδόμητα δεδομένα όπως σε ένα σύστημα αρχείων, είτε σε δομημένα δεδομένα όπως είναι σε μια βάση δεδομένων, αν και το μεγαλύτερο πλεονέκτημα του MapReduce είναι η ταχύτητα επεξεργασίας σε αδόμητα δεδομένα με χαμηλό κόστος, συγκριτικά με άλλες υλοποιήσεις. Το MapReduce, μπορεί να εκμεταλλευτεί την τοπικότητα των δεδομένων, επεξεργάζοντας δεδομένα που βρίσκονται κοντά στα αποθηκευτικά μέσα για μείωση της μετάδοσης των δεδομένων.

Το MapReduce αποτελείται βασικά από δύο φάσεις οι οποίες είναι και οι αντίστοιχες προγραμματιστικές μέθοδοι που αναφέρθηκαν και παραπάνω. Η μία φάση είναι η Map και η άλλη φάση είναι η Reduce.

- Στην Map φάση: Ο κύριος κόμβος (master node) , παίρνει την εισαγωγή (δεδομένα σε δομημένη ή αδόμητη μορφή), διαχωρίζει τα δεδομένα και το πρόβλημα προς επεξεργασία σε υπό-προβλήματα και τα κατανέμει στους εκτελεστικούς κόμβους (worker nodes). Ένας worker node μπορεί και αυτός με την σειρά του να διαχωρίσει το πρόβλημα που του ανατέθηκε σε μικρότερα δημιουργώντας μια πολύ-επίπεδη δενδρική δομή. Μετά κάθε worker node, επεξεργάζεται το μικρότερο πρόβλημα που του ανατέθηκε από το master node και του επιστρέφει την απάντηση.
- Στην Reduce φάση: Ο master node συλλέγει την απάντηση από κάθε υπό-πρόβλημα και τις συνδυάζει κάνοντας και αυτός μια τελική επεξεργασία για να παράγει την έξοδο – την απάντηση δηλαδή στο αρχικό πρόβλημα.

Το MapReduce βοηθάει και αναλαμβάνει την κατανεμημένη επεξεργασία των map και reduce διαδικασιών. Δεδομένου ότι κάθε mapping διαδικασία είναι ανεξάρτητη

από τις άλλες, όλες οι map διαδικασίες μπορούν να εκτελεστούν παράλληλα – αν και στην πράξη αυτό περιορίζεται από τον αριθμό τον ανεξάρτητων πηγών δεδομένων ή τον αριθμό των CPU κοντά σε κάθε πηγή δεδομένων. Παρόμοια, μία συλλογή από reducers μπορεί να εκτελεί παράλληλα την reduce φάση, δεδομένου ότι όλες οι έξοδοι (απαντήσεις) της map διαδικασίας που μοιράζονται το ίδιο κλειδί παρουσιάζονται στον ίδιο reducer την ίδια στιγμή. Αν και αυτή η διαδικασία φαίνεται ανεπαρκή σε σχέση με τους αλγορίθμους που είναι πιο σειριακοί, το MapReduce μπορεί να εφαρμοστεί σε σημαντικά μεγαλύτερο όγκο δεδομένων από τον όγκο δεδομένων που μπορούν τυπικά οι χαμηλού κόστους εξυπηρετητές (servers) να επεξεργαστούν. Για παράδειγμα, μία server farm είναι ικανή να ταξινομήσει όγκο δεδομένων της τάξης των petabyte μέσα σε λίγες μόνο ώρες (ανάλογα την περίπτωση και τα δεδομένα). Αυτή η παράλληλη επεξεργασία προσφέρει και μια πιθανότητα ανάκτησης από μερική βλάβη των server ή μέρους του αποθηκευτικού μέσου, ακόμη και κατά την διάρκεια της επεξεργασίας (αν ένας mapper ή ένας reducer αστοχήσει, η επεξεργασία την οποία έκαναν μπορεί να προγραμματιστεί από την αρχή – δεδομένου ότι τα αρχικά δεδομένα εισόδου είναι διαθέσιμα).

Ένας άλλος τρόπος να εξετάσουμε και να κατανοήσουμε το MapReduce, είναι σαν μια παράλληλη και κατανεμημένη επεξεργασία 5 βημάτων:

1. **Προετοιμασία της εισαγωγής για την μέθοδο Map()** – το γενικότερο σύστημα του MapReduce, ορίζει τους Map επεξεργαστές, εκχωρεί την K1 τιμή την οποία θα δουλέψει κάθε επεξεργαστής, και του παρέχει με όλα τα εισαγωγικά δεδομένα που σχετίζονται με αυτή την τιμή.
   Map (k1,v1) (1.1)

2. **Εκτέλεση του Map() κώδικα που έχει οριστεί από τον χρήστη** – η μέθοδος Map() εκτελείται ακριβώς μία φορά για κάθε K1 τιμή, δημιουργώντας έξοδο οργανωμένη ανά K2.
   Map (k1, v1) → list (k2, v2) (1.2)

3. **Ανάμιξη των εξόδων της Map() και κατανομή στους Reduce επεξεργαστές** – το MapReduce σύστημα, ορίζει τους Reduce

επεξεργαστές, εκχωρεί την K2 τιμή την οποία θα δουλέψει κάθε επεξεργαστής, και του παρέχει με όλα τα δεδομένα που σχετίζονται με αυτή την τιμή και ήταν έξοδο της αντίστοιχης Map() εκτέλεσης. Reduce (k2, list (v2)) (1.3)

4. **Εκτέλεση του Reduce() κώδικα που έχει οριστεί από τον χρήστη** – η μέθοδος Reduce() εκτελείται ακριβώς μία φορά για κάθε K2 τιμή που παράχθηκε από την αντίστοιχη Map() εκτέλεση. Reduce (k2, list (v2)) → list (v3) (1.4)

5. **Παραγωγή της τελικής εξόδου** – το MapReduce σύστημα συλλέγει όλα τις Reduce εξόδους και τις ταξινομεί κατά K2 για να παραχθεί το τελικό αποτέλεσμα.

Παρόλο που τα παραπάνω βήματα λογικά φαίνεται ότι εκτελούνται σειριακά – κάθε βήμα ξεκινά μόνο αν έχει ολοκληρωθεί το προηγούμενο – στην πράξη είναι συνυφασμένα εφόσον δεν επηρεάζουν το τελικό αποτέλεσμα. Σε πολλές περιπτώσεις τα εισαγωγικά δεδομένα είναι ήδη κατανεμημένα (διαχωρισμένα) μεταξύ πολλών server, οπότε το 1ο βήμα γίνετε σημαντικά πιο απλό, απλά αναθέτοντας τους Map server που θα επεξεργαστούν τα τοπικά για τον κάθε ένα δεδομένα. Παρόμοια, το 3ο βήμα μπορεί να επιταχυνθεί αναθέτοντας ως Reduce επεξεργαστές εκείνους που είναι όσο το δυνατόν πιο τοπικά στα Map-εξαγόμενα δεδομένα που πρέπει να επεξεργαστούν.

ΥΠΟΚΕΦΑΛΑΙΟ 1.3

Ροή δεδομένων στο MapReduce

Το MapReduce framework ως αναφορά την ροή των δεδομένων αποτελείται από τα εξής μέρη:

- **Τον input reader** – Ο input reader διαχωρίζει τα δεδομένα σε κατάλληλου μεγέθους ‘κομμάτια’ (στην πράξη κυμαίνονται από 16MB έως 128MB – όχι απαραίτητα) και το framework αναθέτει ένα κομμάτι σε κάθε Map μέθοδο. Ο input reader διαβάζει δεδομένα από σταθερού τύπου αποθήκευση (συνήθως ένα κατανεμημένο σύστημα αρχείων όπως το GFS και το HDFS – τα παροδικά όμως δεδομένα βρίσκονται σε τοπικούς δίσκους, στο τοπικό σύστημα αρχείων και αντλούνται από τους reducer) και παράγει ζεύγη κλειδιού/τιμής. Σε ένα κοινό παράδειγμα θα διαβάσει ένα φάκελο γεμάτο με αρχεία κειμένου και θα επιστρέψει κάθε γραμμή ως εγγραφή.
- **Την μέθοδο Map** – Η Map μέθοδος παίρνει μία σειρά/λίστα από ζεύγη κλειδιού/τιμής, επεξεργάζεται το κάθε ένα από αυτά και εξάγει μηδέν ή περισσότερα ζεύγη κλειδιού/τιμής. Ο τύπος των ζευγών της μεθόδου Map στην είσοδο μπορεί να διαφέρει από τον τύπο των ζευγών στην έξοδο (και συνήθως έτσι συμβαίνει). Αν για παράδειγμα η εφαρμογή κάνει καταμέτρηση των λέξεων ενός αρχείου, η μέθοδος Map θα χωρίζει την κάθε σειρά σε λέξεις και θα εξάγει ένα ζεύγος κλειδιού/τιμής για κάθε λέξη. Κάθε εξαγόμενο ζεύγος θα περιέχει την λέξη ως το κλειδί και τον αριθμό τον στιγμιότυπων της κάθε λέξης στην σειρά ως την τιμή.
- **Την μέθοδο Partition (διαχωρισμού)** – Κάθε έξοδος της μεθόδου Map κατανέμεται σε ένα συγκεκριμένο reducer από την partition μέθοδο της εφαρμογής για λόγους διαχωρισμού. Στην μέθοδο partition δίνετε το κλειδί και ο αριθμός των reducer και επιστρέφει τον δείκτη του επιθυμητού reducer. Το σύνηθες προκαθορισμένο είναι να γίνει hash το κλειδί και να χρησιμοποιηθεί το ακέραιο υπόλοιπο της hashed τιμής με τον αριθμό των reducer. Είναι σημαντικό να επιλεγεί μία partition μέθοδος η οποία δίνει μία περίπου ομοιόμορφη κατανομή των δεδομένων ανά κομμάτι (shard) για λόγους εξισορρόπησης φόρτου (load-balancing), αλλιώς όλη η λειτουργία του MapReduce μπορεί να καθυστερήσει περιμένοντας αργούς reducer να τελειώσουν (δηλαδή reducer που τους έχουν ανατεθεί περισσότερα δεδομένα από αυτά που τους αναλογούν). Μεταξύ των map και reduce φάσεων, τα δεδομένα αναμιγνύονται (γίνετε παράλληλη ταξινόμηση / ανταλλαγή μεταξύ των κόμβων) για να μετακινηθούν τα δεδομένα από τον Map κόμβο που τα παρήγαγε, στο shard στο οποίο θα γίνουν reduce. Αυτή η μετακίνηση μερικές φορές, μπορεί να πάρει περισσότερο χρόνο από τον

υπολογιστικό χρόνο και εξαρτάται από το εύρος ζώνης του δικτύου, την ταχύτητα των CPU, τον όγκο των δεδομένων που παρήχθησαν και από τον χρόνο υπολογισμού των map και reduce φάσεων.

- **Την μέθοδο Comparison (σύγκρισης)** – Η εισαγωγή για κάθε Reduce αντλείτε από τον κόμβο όπου εκτελέστηκε η μέθοδος Map και ταξινομείτε χρησιμοποιώντας την μέθοδο σύγκρισης.
- **Την μέθοδο Reduce** – Το MapReduce framework «καλεί» την μέθοδο Reduce μία φορά για κάθε μοναδικό κλειδί σε ταξινομημένη σειρά. Η μέθοδος Reduce επαναλαμβάνεται μέχρι να περάσει από όλες τις τιμές που σχετίζονται με το συγκεκριμένο κλειδί και παράγει μηδέν ή παραπάνω εξόδους. Στο παράδειγμα της καταμέτρησης των λέξεων, η μέθοδος Reduce παίρνει τις τιμές εισαγωγής, τις συνοψίζει και παράγει μία έξοδο για κάθε λέξη και το τελικό αποτέλεσμα.
- **Τον output writer** – Ο Output Writer γράφει την έξοδο του reduce στο σύστημα αποθήκευσης, συνήθως ένα κατανεμημένο σύστημα αρχείων (GFS, HDFS).

ΥΠΟΚΕΦΑΛΑΙΟ 1.4

Χρήσεις του MapReduce

Το MapReduce είναι πολύ χρήσιμο σε ένα ευρύ φάσμα εφαρμογών, όπως, κατανεμημένη αναζήτηση βασισμένη σε μοτίβα (pattern), κατανεμημένη ταξινόμηση (κάτι που γίνετε και μέσα στο ίδιο το framework του MapReduce), κατασκευή ανεστραμμένων δεικτών (inverted index - για δυνατότητα πλήρης αναζήτησης σε μορφή κειμένου κάτι που συνήθως απαιτεί μεγάλη επεξεργαστική ισχύ), clustering εγγράφων, μηχανική μάθηση, στατιστική μετάφραση κ.α. Επιπλέον, το μοντέλο του MapReduce έχει προσαρμοστεί σε πολλά υπολογιστικά περιβάλλοντα όπως πολυπύρηνα συστήματα, desktop grid, υπολογιστικά περιβάλλοντα υποστηριζόμενα από εθελοντές χρήστες (για την εξέλιξη ερευνών σε θέματα υγείας κ.α.), δυναμικά cloud περιβάλλοντα και κινητά περιβάλλοντα.

Το μοντέλο του MapReduce, έχει εφαρμοστεί από πληθώρα εταιριών για διάφορους λόγους και με διαφορετικές υλοποιήσεις αλλά κατά βάση οι αρχές είναι οι ίδιες. Μερικές από αυτές είναι η Yahoo, η Amazon, το Facebook και πολλές άλλες. Η πιο βασική από την οποία και ξεκίνησε αυτό το μοντέλο είναι η Google, η οποία χρησιμοποίησε το MapReduce για να αναπαράγει το index της για το Διαδίκτυο (web), και ήταν αυτή η ανάγκη που την οδήγησε σε αυτό το κατόρθωμα το οποίο σήμερα εκμεταλλεύεται από εκατοντάδες εταιρίες.

ΥΠΟΚΕΦΑΛΑΙΟ 1.5

Apache Hadoop

Μία από τις υλοποιήσεις του μοντέλου του MapReduce (που ξεκίνησε η Google) είναι το Hadoop του οποίου το ξεκίνημα έγινε το 2005 από τους Mike Cafarella και Doug Cutting. Ο Cutting τότε δούλευε στην Yahoo! και το ανέπτυξε για να υποστηρίξει την διανομή για το project αναζήτησης Nutch και έδωσε το όνομα Hadoop από ένα παιχνίδι ελέφαντα του γιου του. Αργότερα έγινε top-level project του Apache Software Foundation και εξελίσσεται συνεχώς με την βοήθεια της τεράστιας κοινότητας αλλά και των εταιριών που κάνουν δικές τους διανομές αλλά συνεισφέρουν και στον βασικό κώδικα του Hadoop.

Το Hadoop framework είναι γραμμένο κυρίως σε Java, μαζί με κάποιο κώδικα σε C και shell-scripts για τις εφαρμογές που εκτελούνται μέσω της γραμμής εντολών. Για την ανάπτυξη στο Hadoop χρησιμοποιείται συνήθως η Java η οποία είναι και επίσημα υποστηριζόμενη μαζί με την Python, μέσω των Java και Python API που παρέχονται από την Apache, για διεπαφή με το Hadoop. Το Hadoop αποτελείται από διάφορα πακέτα/μέρη τα οποία όλα μαζί συμπληρώνουν το οικοσύστημα του Hadoop και παίζει διαφορετικό ρόλο το κάθε ένα στην λειτουργία και την αρχιτεκτονική αυτού του συστήματος. Το πακέτο Hadoop Common, το οποίο παρέχει αφαιρέσεις σε επίπεδο λειτουργικού συστήματος και σε επίπεδο συστήματος αρχείων, την μηχανή MapReduce (ανάλογα την έκδοση MapReduce/MR1 ή YARN/MR2) και το Hadoop Distributed File System (HDFS). Αυτά είναι τα βασικά και κύρια πακέτα για την λειτουργία του Apache Hadoop. Το

Hadoop Common πακέτο περιέχει τα απαραίτητα Java αρχεία (JAR) και scripts που χρειάζονται για το ξεκίνημα του Hadoop. Το πακέτο επίσης, παρέχει πηγαίο κώδικα, τεκμηρίωση του Hadoop και ένα μέρος για συμβολή στο project.

Για την αποτελεσματική χρονοδρομολόγηση της εργασίας που απαιτείται να πραγματοποιηθεί, κάθε σύστημα αρχείων συμβατό με το Hadoop πρέπει να παρέχει ενημέρωση για την τοποθεσία των κόμβων, το όνομα του rack (συγκεκριμένα του switch του δικτύου) όπου βρίσκεται ένα κόμβος επεξεργασίας. Έτσι, οι εφαρμογές που τρέχουν στο Hadoop μπορούν να χρησιμοποιήσουν αυτή την πληροφορία για να εκτελέσουν την εργασία στον κόμβο που βρίσκονται τα δεδομένα και σε περίπτωση αστοχίας, σε κάποιο κόμβο στο ίδιο rack/switch μειώνοντας έτσι την κίνηση στο κεντρικό δίκτυο (backbone traffic). Το HDFS χρησιμοποιεί αυτή την μέθοδο όταν αναπαράγει δεδομένα, προσπαθώντας να έχει αντίγραφα των δεδομένων σε διαφορετικά rack/switch για λόγους ασφάλειας σε περίπτωση αστοχίας. Ο στόχος είναι να μειωθούν οι επιπτώσεις σε περίπτωση διακοπής ρεύματος σε ένα rack ή σε περίπτωση αστοχίας κάποιου switch, ώστε ακόμη και αν συμβεί κάτι τέτοιο να υπάρχει αντίγραφο των δεδομένων κάπου αλλού και τα δεδομένα να είναι ακόμη αναγνώσιμα.

Ένα μικρό Hadoop cluster αποτελείται από ένα master κόμβο και πολλούς worker κόμβους. Ο master κόμβος αποτελείται από ένα JobTracker και ένα NameNode, αλλά αν είναι μικρό το cluster μπορεί να έχει και ρόλο worker node ταυτόχρονα, δηλαδή να έχει και ρόλο TaskTracker και DataNode. Κάθε slave/worker κόμβος συνήθως είναι παράλληλα DataNode και TaskTracker, αν και υπάρχει η δυνατότητα συνήθως όταν είναι πολύ μεγάλο το cluster και σε ειδικές εφαρμογές, να έχει κόμβους μόνο για τα δεδομένα (DataNode) και κόμβους μόνο για την επεξεργασία (TaskTracker). Μία από τις βασικές απαιτήσεις του Hadoop είναι, κάθε κόμβος να έχει την ίδια έκδοση του Java Runtime Environment (JRE – 1.6 και πάνω). Συνήθως χρησιμοποιείται η 1.6 γιατί υποστηρίζεται καλύτερα από κάποιες διανομές αλλά τελευταία υπάρχει και υποστήριξη για την 1.7. Επιπλέον, πρέπει να έχει γίνει εγκατάσταση και παραμετροποίηση του Secure Shell (ssh), σε κάθε κόμβο για να υπάρχει επικοινωνία μεταξύ τους και για την λειτουργία των script για το ξεκίνημα και τον τερματισμό του cluster.

Σε ένα μεγάλο cluster, το HDFS διαχειρίζεται μέσω ενός NameNode server αφιερωμένο μόνο σε αυτό, για να φιλοξενεί τον κατάλογο (index) του συστήματος αρχείων, και από έναν δευτερεύον NameNode server για να παράγει στιγμιότυπα της δομής της μνήμης του NameNode. Με αυτό τον τρόπο προστατεύεται το σύστημα αρχείων από αλλοίωση και μειώνεται η απώλεια δεδομένων. Παρόμοια σε ένα μεγάλο cluster, μπορεί να υπάρχει ένας κόμβος με μόνη ιδιότητα αυτή του JobTracker server για να διαχειρίζεται τον προγραμματισμό των εργασιών. Υπάρχουν όμως και περιπτώσεις όπου η MapReduce μηχανή του Hadoop αναπτύσσεται σε διαφορετικό σύστημα αρχείων από το HDFS. Σε αυτή την περίπτωση ο NameNode, ο δευτερεύον NameNode και οι αρχιτεκτονική των DataNode (όλα μέρη του HDFS), αντικαθίστανται από τα αντίστοιχα του συστήματος αρχείων που χρησιμοποιείται.

ΥΠΟΚΕΦΑΛΑΙΟ 1.6

Hadoop Distributed File System (HDFS)

Το Hadoop Distributed File System (HDFS) είναι ένα κατανεμημένο, επεκτάσιμο και φορητό σύστημα αρχείων γραμμένο σε Java για το Hadoop framework. Κάθε κόμβος σε ένα Hadoop cluster, συνήθως έχει ένα NameNode και ένα cluster από DataNode κόμβους, οι οποίοι και αποτελούν το HDFS cluster. Κάθε κόμβος DataNode προσφέρει block δεδομένων μέσω δικτύου στους άλλους κόμβους, χρησιμοποιώντας ένα πρωτόκολλο (τα δεδομένα στέλνονται σε block) αποκλειστικό στο HDFS. Το σύστημα αρχείων χρησιμοποιεί τα TCP/IP επίπεδα για την επικοινωνία και οι clients χρησιμοποιούν την Remote procedure call (RPC) για να επικοινωνούν μεταξύ τους.

Το HDFS αποθηκεύει τα δεδομένα σε πολλαπλούς κόμβους και το μέγεθος συνήθως είναι από gigabyte μέχρι και petabyte σε κάποιες εφαρμογές. Την αξιοπιστία του την επιτυγχάνει αντιγράφοντας τα δεδομένα σε πολλαπλούς κόμβους και έτσι δεν χρειάζεται αποθήκευση RAID στους κόμβους, κάτι που θα κόστιζε πολύ περισσότερο. Με την προκαθορισμένη τιμή αναπαραγωγής των δεδομένων, στο 3, τα δεδομένα είναι ανά πάσα στιγμή σε τρείς κόμβους: σε δύο

στο ίδιο rack και σε ένα σε διαφορετικό rack. Οι DataNode κόμβοι μπορούν να επικοινωνούν μεταξύ τους για να εξισορροπήσουν τα δεδομένα, να μεταφέρουν αντίγραφα των δεδομένων σε άλλους κόμβους και να κρατήσουν την αντιγραφή και αναπαραγωγή των δεδομένων υψηλά. Το HDFS δεν είναι πλήρως συμβατό με το POSIX (Portable Operating System Interface) standard για την συμβατότητα των συστημάτων αρχείων, γιατί οι απαιτήσεις για ένα POSIX σύστημα αρχείων είναι διαφορετικές από τους στόχους μια εφαρμογής Hadoop. Το πλεονέκτημα είναι ότι το HDFS έχει αυξημένη απόδοση για δεδομένα και υποστήριξη για μη-POSIX διεργασίες. Επίσης, από την έκδοση 2.0 από τον Μάιο του 2012, προστέθηκαν δυνατότητες υψηλής διαθεσιμότητας (high-availability), επιτρέποντας στον κεντρικό metadata server (τον NameNode), να τεθεί σε κατάσταση backup (όχι αυτόματα) σε περίπτωση αστοχίας στο σύστημα, αλλά έχει ήδη ξεκινήσει και η ανάπτυξη για αυτόματο fail-over. Όπως, ήδη αναφέραμε σε ένα HDFS σύστημα αρχείων, εκτός από τον NameNode υπάρχει και ο δευτερεύον NameNode, αλλά παρά την ονομασία του, η ρόλος του δεν είναι να αναλαμβάνει την θέση του NameNode όταν αυτός είναι offline. Αυτό που κάνει ο δευτερεύον NameNode είναι να συνδέεται τακτικά με τον NameNode και να φτιάχνει στιγμιότυπα του καταλόγου πληροφοριών του, τα οποία μετά αποθηκεύονται από το σύστημα σε τοπικούς ή απομακρυσμένους καταλόγους. Αυτές οι εικόνες καταλόγου μπορούν να χρησιμοποιηθούν αργότερα για να επανακτηθεί ο NameNode μετά από σφάλμα, χωρίς να χρειαστεί να γίνουν από την αρχή όλες οι ενέργειες εκκίνησης του συστήματος αρχείων και να επεξεργαστεί το αρχείο καταγραφών (log) για να δημιουργηθεί μια ενημερωμένη δομή καταλόγου. Επιπλέον, ο NameNode είναι ένα από τα πιο σημαντικά σημεία του HDFS αλλά και του Hadoop γενικότερα, γιατί είναι το μοναδικό σημείο για την αποθήκευση και την διαχείριση των μετά-δεδομένων (metadata), και έτσι μπορεί να γίνει σημείο συμφόρησης (bottleneck) όταν απαιτείται να υποστηρίξει μεγάλο αριθμό αρχείων, ιδίως όταν πρόκειται για μεγάλο αριθμό μικρών αρχείων. Έτσι, μία νέα πρόσθεση στο HDFS, το HDFS Federation, προσπαθεί να αντιμετωπίσει αυτό το πρόβλημα επιτρέποντας διαφορετικά namespaces να εξυπηρετούνται από διαφορετικά NameNode.

Ένα πλεονέκτημα της χρήσης του HDFS η επίγνωση για τα δεδομένα μεταξύ του JobTracker και του TaskTracker. Ο JobTracker προγραμματίζει map και reduce διεργασίες στους TaskTracker έχοντας επίγνωση της τοποθεσίας των δεδομένων.

Για παράδειγμα: αν ο κόμβος Α έχει τα δεδομένα (x, y, z) και ο κόμβος Β τα δεδομένα (a, b, c), ο JobTracker θα προγραμματίσει τον κόμβο Β να εφαρμόσει τις map και reduce διεργασίες στα δεδομένα (a, b, c) και τον κόμβο Α να τις εφαρμόσει στα δεδομένα (x, y, z). Αυτό μειώνει τον όγκο της κίνησης στο δίκτυο και αποτρέπει την μη αναγκαία μεταφορά δεδομένων η οποία θα φόρτωνε περιττά το δίκτυο. Αυτό ισχύει όταν χρησιμοποιείται το Hadoop με το HDFS. Όταν χρησιμοποιείται με άλλα συστήματα αρχείων, αυτή η δυνατότητα δεν είναι πάντα διαθέσιμη. Τα πλεονεκτήματα βέβαια είναι εμφανή στους χρόνους ολοκλήρωσης των διεργασιών, ιδίως σε διεργασίες απαιτητικές σε δεδομένα. Το HDFS όμως σχεδιάστηκε για αμετάβλητα αρχεία και δεν είναι το πιο κατάλληλο για συστήματα που απαιτούν ταυτόχρονες διεργασίες εγγραφής. Ένας ακόμη περιορισμός του HDFS είναι ότι δεν μπορεί να φορτωθεί (mount) άμεσα από ένα υπάρχον λειτουργικό σύστημα, δηλαδή δεν μπορεί ο χρήστης να έχει άμεση πρόσβαση σε αυτό από το λειτουργικό σύστημα όπως συμβαίνει με άλλα συστήματα αρχείων. Γι' αυτό το να εισάγεις και να εξάγεις δεδομένα από το HDFS, κάτι που συνήθως γίνετε πριν και μετά την εκτέλεση μίας διεργασίας, μπορεί να γίνει χρονοβόρο και άβολο. Αυτός είναι και ο λόγος που αναπτύχθηκε ένα εικονικό FUSE (Filesystem in Userspace) σύστημα αρχείων, αλλά μόνο για Linux και κάποια άλλα τύπου Unix λειτουργικά συστήματα.

Για απόκτηση πρόσβασης στα αρχεία του HDFS μέσω μιας εφαρμογής, χρησιμοποιείται το native Java API και το Thrift API της Apache (αρχικά αναπτύχθηκε από το Facebook) για την παραγωγή του κώδικα του client στην γλώσσα επιλογής του χρήστη (C++, Java, Python, PHP, Ruby, Erlang, Perl, Haskell, C#, Cocoa, Smalltalk και OCaml). Επιπλέον, υπάρχει και η επιλογή μίας διεπαφής (interface) μέσω της γραμμής εντολών και η περιήγηση στο HDFS από την web διεπαφή του HDFS (HDFS-UI) μέσω HTTP.

ΥΠΟΚΕΦΑΛΑΙΟ 1.6.1
Άλλα συστήματα αρχείων

Το Apache Hadoop υποστηρίζει επίσημα αρκετά άλλα συστήματα αρχείων τα οποία είναι τα εξής:

- **Το Amazon S3 σύστημα αρχείων** – Αυτό είναι στοχευόμενο και σχεδιασμένο για τα cluster που αναπτύσσονται στην server on-demand υποδομή του Amazon Elastic Compute Cloud. Στο συγκεκριμένο σύστημα αρχείων επειδή τα αρχεία βρίσκονται απομακρυσμένα, το Hadoop χάνει τα πλεονεκτήματα που προσφέρουν στην απόδοση του, η τοπικότητα των δεδομένων και η επίγνωση της τοποθεσίας των δεδομένων. Εδώ δεν υπάρχει επίγνωση της τοποθεσίας (rack) των δεδομένων και υπάρχει καθυστέρηση μέχρι να είναι σίγουρο ότι μεταφέρθηκαν τα δεδομένα χωρίς σφάλμα.
- **Το CloudStore (Kosmos Distributed File System)** – το οποίο προσφέρει επίγνωση της τοποθεσίας (rack) και ήταν μια C++ υλοποίηση του Google File System. Το CloudStore προσφέρει επεκτασιμότητα, αναπαραγωγή και αντιγραφή των δεδομένων σε πολλαπλούς κόμβους για ασφάλεια, έλεγχο για την ακεραιότητα των δεδομένων, fail-over από την πλευρά του client και πρόσβαση από Java, Python αλλά και C++. Τέλος, προσφέρει και την δυνατότητα φόρτωσης (mount) του σε λειτουργικό σύστημα Linux μέσω μίας μονάδας (module) FUSE.
- **Το FTP σύστημα αρχείων** – Σε αυτή την περίπτωση τα αρχεία απλά βρίσκονται σε απομακρυσμένους FTP server και είναι προσβάσιμα μέσω αυτών.
- **Το HDFS (το οποίο αναλύθηκε εκτενώς παραπάνω)** – Το HDFS είναι το προκαθορισμένο κατανεμημένο σύστημα αρχείων του Hadoop και έχει όπως αναφέραμε ήδη επίγνωση της τοποθεσίας των δεδομένων και

σχεδιάστηκε να επεκτείνεται σε δεκάδες petabyte δεδομένων και εκτελείται πάνω από το σύστημα αρχείων του λειτουργικού συστήματος που χρησιμοποιείται.

- **Το maprfs σύστημα αρχείων της MapR (εταιρία που κάνει δική της διανομή του Hadoop)** – Το maprfs σχεδιάστηκε από την αρχή για να παρέχει υψηλή διαθεσιμότητα, σωστά στιγμιότυπα και αντίγραφα, παρέχοντας παράλληλα υψηλότερη επεκτασιμότητα από το HDFS και υψηλότερη απόδοση. Το maprfs είναι μέρος της διανομής της MapR και ως το προκαθορισμένο σύστημα αρχείων στο Elastic Map Reduce της Amazon και στην Google Compute Engine.
- **Τα μόνο για ανάγνωση HTTP και HTTPS συστήματα αρχείων**.

Γενικά, το Hadoop μπορεί να λειτουργήσει με οποιοδήποτε κατανεμημένο σύστημα αρχείων το οποίο μπορεί να φορτωθεί (mount) από το λειτουργικό σύστημα χρησιμοποιώντας ένα URL του τύπου file://. Αυτό όμως έχει ένα μειονέκτημα γιατί έτσι θα χαθεί η τοπικότητα που προσφέρει το Hadoop. Αυτό συμβαίνει γιατί το Hadoop χρειάζεται να γνωρίζει ποιοι κόμβοι είναι πιο κοντά στα δεδομένα, και αυτή την πληροφορία μπορούν να την παρέχουν μόνο «γέφυρες» συστημάτων αρχείων που είναι σχεδιασμένες για το Hadoop. Χωρίς καμία τροποποίηση ή «γέφυρα» προσφέρουν αυτή τη δυνατότητα το S3 της Amazon και το CloudStore, μέσω των URL s3:// και kfs:// αντίστοιχα.

Όμως έχουν αναπτυχθεί κάποιες τέτοιου είδους «γέφυρες» συστημάτων αρχείων από κάποιες εταιρίες και είναι οι εξής:

- Το IBM General Parallel File System, του οποίου ο κώδικας έγινε διαθέσιμος το 2009.
- Η Parascale, δημοσίευσε των κώδικα για την λειτουργία του Hadoop μαζί με το Parascale σύστημα αρχείων τον Απρίλιο του 2010.
- Η Appistry, δημοσίευσε έναν οδηγό (driver) για το σύστημα αρχείων του Hadoop για χρήση με το CloudIQ Storage τον Απρίλιο του 2010.
- Η HP (Hewlett-Packard), σχεδίαζε την ανάπτυξη ενός οδηγού (driver) για το σύστημα αρχείων IBRIX Fusion τον Ιούνιο του 2010.

- Η MapR Technologies, Inc., ανακοίνωσε τον Μάιο του 2011 ένα εναλλακτικό σύστημα αρχείων για το Hadoop , το οποίο θα αντικαθιστούσε το HDFS.

ΥΠΟΚΕΦΑΛΑΙΟ 1.7

Apache Hadoop MapReduce

Πάνω από το σύστημα αρχείων, λειτουργεί η μηχανή MapReduce του Hadoop, που αποτελείται από τον JobTracker, στον οποίο καταθέτουν MapReduce εργασίες προς εκτέλεση οι client εφαρμογές. Ο JobTracker, προωθεί την εργασία στους διαθέσιμους TaskTracker κόμβους στο cluster, προσπαθώντας να κρατήσει την επεξεργασία όσο είναι δυνατό πιο κοντά στα δεδομένα. Με ένα σύστημα αρχείων που παρέχει την τοποθεσία (rack) που βρίσκονται τα δεδομένα, όπως είναι το HDFS, ο JobTracker γνωρίζει σε ποιον κόμβο είναι τα δεδομένα και ποιοι είναι πιο κοντά. Προτεραιότητα φυσικά έχει ο ίδιος ο κόμβος που έχει τα δεδομένα, αλλά αν για κάποιο λόγο δεν μπορεί να εκτελέσει την εργασία, προτεραιότητα παίρνουν οι κόμβοι που βρίσκονται στο ίδιο rack. Αυτό μειώνει την κίνηση στον κορμό του δικτύου. Αν ένας TaskTracker κόμβος έχει κάποιο σφάλμα ή αργεί να επικοινωνήσει, το μέρος της εργασίας που εκτελούσε ανά-προγραμματίζετε. Ο TaskTracker σε κάθε κόμβο, δημιουργεί ένα ξεχωριστό Java Virtual Machine (εκτός από αυτό που εκτελεί την εργασία), για να αποτρέψει να αστοχήσει ο TaskTracker αν η τρέχουσα εργασία «κολλήσει» τον JVM. Ο TaskTracker στέλνει ένα σήμα (heartbeat) στον JobTracker κάθε λίγα λεπτά για να ελέγξει εκείνος την κατάσταση του και για να ξέρει αν αποκρίνεται ή όχι ο TaskTracker, ώστε να ανά-προγραμματίσει την δουλειά που εκτελούσε σε περίπτωση που δεν λάβει κάποιο heartbeat. Οι πληροφορίες και η κατάσταση του JobTracker και TaskTracker, γίνονται διαθέσιμες μέσω του Java web εξυπηρετητή Jetty και έτσι μπορούν να αναγνωστούν μέσω ενός προγράμματος περιήγησης (web browser). Μέχρι την έκδοση 0.20 του Hadoop, αν ο JobTracker είχε κάποια αστοχία όλες οι τρέχουσες εργασίες χανόντουσαν. Από την έκδοση 0.21 και μετά προστέθηκαν κάποια σημεία ελέγχου σε αυτή την διαδικασία και ο JobTracker καταγράφει τι γίνετε και τι

διεργασίες εκτελεί στο σύστημα αρχείων. Έτσι, όταν ξεκινά ο JobTracker, ελέγχει για τέτοια δεδομένα και μπορεί να εκκινήσει την εργασία από εκεί που διακόπηκε.

Υπάρχουν όμως και περιορισμοί σε αυτή την προσέγγιση οι οποίοι είναι:

- Η κατανομή των εργασιών στους TaskTracker είναι πολύ απλή. Κάθε TaskTracker έχει ένα αριθμό διαθέσιμων θέσεων (για παράδειγμα τέσσερις θέσεις). Κάθε ενεργή map ή reduce διεργασία καταλαμβάνει μία θέση. Ο JobTracker κατανέμει την εργασία στον πιο κοντινό TaskTracker στα δεδομένα, με διαθέσιμη θέση. Δεν λαμβάνει όμως υπόψη τον τρέχον φόρτο του κόμβου που πρόκειται να αναθέσει μια εργασία, δηλαδή την πραγματική διαθεσιμότητα του TaskTracker.
- Αν ένας TaskTracker είναι πολύ αργός, μπορεί να καθυστερήσει όλη την εκτέλεση μιας MapReduce εργασίας, ειδικά προς το τέλος μια εργασίας, που μπορεί τα πάντα να περιμένουν την πιο αργό κόμβο TaskTracker να τελειώσει. Παρόλα αυτά υπάρχει μια επιλογή, η speculative εκτέλεση η οποία αν είναι ενεργοποιημένη, επιτρέπει μία εργασία να εκτελείται σε πολλαπλούς θυγατρικούς κόμβους.

ΥΠΟΚΕΦΑΛΑΙΟ 1.7.1
Χρονοδρομολόγηση MapReduce εργασιών

Το προκαθορισμένο μοντέλο χρονοδρομολόγησης που χρησιμοποιεί το Hadoop, είναι το FIFO (First-In-First-Out), δηλαδή ο πρώτη εργασία που θα εισαχθεί στην ουρά, θα είναι και η πρώτη που θα προγραμματιστεί για εκτέλεση. Προαιρετικά, προσφέρει και την δυνατότητα πέντε προτεραιοτήτων χρονοδρομολόγησης, για την χρονοδρομολόγηση εργασιών που βρίσκονται σε ουρά. Από την έκδοση 0.19 και μετά ο χρονοδρομολογητής εργασιών, παραμετροποιήθηκε από την αρχή και δεν είναι πλέον άμεσα συνδεδεμένος με τον JobTracker και αυτό προσέφερε την δυνατότητα να χρησιμοποιηθεί και κάποιος διαφορετικός χρονοδρομολογητής όπως ο Fair χρονοδρομολογητής ή ο Capacity χρονοδρομολογητής, οι ιδιότητες των οποίων είναι παρακάτω.

Fair Scheduler

Ο fair scheduler αναπτύχθηκε από το Facebook και ο στόχος του είναι να παρέχει γρήγορους χρόνους απόκρισης για μικρές εργασίες και QoS (Quality of Service – Ποιότητα Υπηρεσίας) για τις σημαντικές εργασίες. Ο fair scheduler βασίζεται σε τρεις βασικές γενικές αρχές:

- Οι εργασίες ομαδοποιούνται ανά κατηγορία.
- Σε κάθε ομάδα εργασιών, δίνετε ένα ελάχιστο και εγγυημένο μερίδιο, ως αναφορά τον προγραμματισμό τους.
- Η πλεονάζουσα χωρητικότητα (το μερίδιο χρόνου που έχει απομείνει) μοιράζετε ισάξια ανάμεσα στις εργασίες.

Οι εργασίες, που δεν έχουν κατηγοριοποιηθεί τοποθετούνται σε μια προκαθορισμένη ομάδα. Κάθε ομάδα πρέπει να έχει ορισμένο ένα ελάχιστο αριθμό διαθέσιμων θέσεων για εργασίες map, ένα για εργασίες reduce και ένα άνω όριο στις εκτελούμενες εργασίες.

Capacity Scheduler

Ο capacity scheduler αναπτύχθηκε από την Yahoo!, και έχει πολλές ομοιότητες με τον fair scheduler ως προς τις ιδιότητες τους. Οι πιο βασικές του ιδιότητες είναι οι εξής:

- Οι εργασίες οργανώνονται σε ουρές.
- Σε κάθε ουρά δίνετε μόνο ένα μέρος από το συνολική χωρητικότητα των πόρων.
- Οι ελεύθεροι πόροι, που δεν έχουν δοθεί σε κάποια ουρά, κατανέμονται σε ουρές που έχουν ξεπεράσει την συνολική τους χωρητικότητα.
- Μέσα σε μία ουρά, μία εργασία με υψηλή προτεραιότητα έχει πρόσβαση στους πόρους της ουράς.
- Μόλις μια εργασία αρχίσει να εκτελείται δεν μπορεί να διακοπεί.

ΥΠΟΚΕΦΑΛΑΙΟ 1.8

Άλλες εφαρμογές των στοιχείων του Apache Hadoop

Το HDFS σύστημα αρχείων δεν χρησιμοποιείται αποκλειστικά για εκτέλεση εργασιών MapReduce. Έχει και άλλες εφαρμογές, πολλές από τις οποίες είναι και αυτές project της Apache και ανήκουν στο γενικότερο οικοσύστημα του Hadoop και λειτουργούν πάνω από αυτό προσφέροντας επιπλέον δυνατότητες, στα οποία θα αναφερθούμε παρακάτω. Μερικά από αυτά που συνεργάζονται με το HDFS είναι, η βάση δεδομένων HBase, το Apache Mahout σύστημα μηχανικής μάθησης και το Apache Hive σύστημα αποθήκης δεδομένων. Επίσης, το Hadoop μπορεί να χρησιμοποιηθεί για οποιαδήποτε εργασία που έχει υψηλές απαιτήσεις σε δεδομένα, χρειάζεται πολύ επεξεργασία (αλλά όχι σε πραγματικό χρόνο) και που μπορεί να ολοκληρωθεί χωρίζοντας και επεξεργάζοντας τα κομμάτια των δεδομένων παράλληλα. Πολλές τέτοιας φύσης εφαρμογές στις οποίες έχει χρησιμοποιηθεί το Hadoop είναι:

- Ανάλυση αρχείων καταγραφής.
- Ανάλυση για λόγους marketing, για την βοήθεια στην λήψη αποφάσεων, για έρευνα των τάσεων της αγοράς σε κάποιο τομέα κλπ.
- Προχωρημένη εξόρυξη δεδομένων και μηχανική μάθηση.
- Επεξεργασία εικόνας.
- Επεξεργασία μηνυμάτων XML.
- Επεξεργασία ελεύθερου κειμένου (π.χ. Twitter) και περιήγηση του web (web crawling).
- Γενική αρχειοθέτηση ακόμη και σχεσιακών ή σε μορφή πινάκων δεδομένων για λόγους συμβατότητας.

ΥΠΟΚΕΦΑΛΑΙΟ 1.9

Εταιρίες που χρησιμοποιούν το Apache Hadoop

Yahoo!

Η Yahoo! Inc. , το Φεβρουάριο του 2008, παρουσίασε μία από τις μεγαλύτερες εφαρμογές του Hadoop μέχρι τότε, το Yahoo! Search Webmap, το οποίο είναι μία εφαρμογή που «τρέχει» σε ένα cluster με πάνω από δέκα χιλιάδες επεξεργαστικούς πυρήνες πάνω σε λειτουργικό σύστημα Linux και παράγει δεδομένα που βοηθούν και χρησιμοποιούνται σε κάθε ερώτημα αναζήτησης προς την Yahoo!. Υπάρχουν πολλά Hadoop cluster στην Yahoo! και κανένα HDFS σύστημα αρχείων ή MapReduce εργασία, χωρίζεται σε πολλαπλά datacenter. Κάθε κόμβος ενός Hadoop cluster, όταν ξεκινά, φορτώνει την εικόνα του Linux συμπεριλαμβανομένου και της διανομής του Hadoop. Μία από τις εργασίες που εκτελούν τα cluster τις Yahoo! είναι ο υπολογισμός των δεικτών για την μηχανή αναζήτησης της. Τον Ιούνιο του 2009, η Yahoo! έκανε διαθέσιμο τον πηγαίο κώδικα, της έκδοσης του Hadoop που λειτουργεί στα πραγματικά της cluster, και συνεχίζει να συνεισφέρει στο Hadoop και την κοινότητα του με ότι αλλαγή κάνει. Επιπλέον, οι προγραμματιστές της εταιρίας διορθώνουν σφάλματα (bugs) που έχει το project, παρέχουν βελτιώσεις στην σταθερότητα του Hadoop και παρέχουν αυτό τον κώδικα ώστε και άλλοι χρήστες να επωφεληθούν από την εργασία που κάνει αρχικά για δικό της όφελος.

Facebook

Το 2010, το Facebook ισχυριζόταν ότι έχει το μεγαλύτερο Hadoop cluster στον κόσμο με είκοσι ένα Petabyte δεδομένων και συνεχίζει να αυξάνεται σε μέγεθος μέχρι και σήμερα. Τον Ιούνιο του 2012 τα δεδομένα είχαν φτάσει τα εκατό Petabyte και τον Νοέμβριο του 2012 ανακοίνωσαν ότι η αποθήκη των δεδομένων τους αυξάνεται κάθε μέρα κατά μισό Petabyte περίπου.

Άλλοι χρήστες/εταιρίες

Το Hadoop χρησιμοποιείται, πλέον από πάρα πολλές εταιρείες οι μισές των οποίων είναι στην λίστα Fortune 50 (λίστα με τις πιο κερδοφόρες και ισχυρές εταιρίες). Βέβαια σε κάθε εταιρία χρησιμοποιείται για διαφορετικό σκοπό και σε διαφορετική έκταση με cluster μερικών δεκάδων κόμβων, μέχρι cluster με χιλιάδες κόμβους και με Petabyte δεδομένων προς επεξεργασία. Μερικές από αυτές είναι η A9.com (Amazon), η Adobe, η AOL, η Cloudspace, το Cornell University Web Lab,

το eBay, το Greece.com , η Hulu, η IBM, η ImageShack, το Last.fm, το LinkedIn, το Mail.gr, η NAVTEQ Media Solutions, η The New York Times, η Rackspace, το Spotify, το StumbleUpon, η Telefonica Research, το Twitter, το University of Glasgow, το University of Maryland, το Veoh και πολλές άλλες.

ΥΠΟΚΕΦΑΛΑΙΟ 1.10

Project του οικοσυστήματος του Apache Hadoop

Πέρα από το HDFS σύστημα αρχείων, το MapReduce (MR1), το YARN (MR2), ολόκληρη η πλατφόρμα/οικοσύστημα του Apache Hadoop, πλέον θεωρείται ότι αποτελείται και από άλλα project που λειτουργούν πάνω στο «βασικό» Hadoop και είναι και αυτά project της Apache. Αυτά είναι:

- **Το Apache Ambari** – Το Apache Ambari project έχει ως στόχο να κάνει την διαχείριση του Hadoop πολύ πιο απλή αναπτύσσοντας λογισμικό για την ρύθμιση, και παραμετροποίηση του Hadoop, για την διαχείριση του Hadoop και για την επίβλεψη και παρακολούθηση των cluster του Hadoop. Το Ambari παρέχει, μία διαισθητική και εύκολη στην χρήση web διεπαφή, υποστηριζόμενη από τα RESTful API του και έχει συμβατότητα με τα: HDFS, MapReduce, Hive, HCatalog, HBase, Zookeeper, Oozie, Pig, και Sqoop. Το Ambari, βοηθά τους διαχειριστές του συστήματος βήμα προς βήμα να εγκαταστήσουν και να ρυθμίσουν νέες υπηρεσίες στο cluster σε οποιοδήποτε αριθμό κόμβων. Παρέχει κεντρική διαχείριση για ξεκίνημα, τερματισμό και παραμετροποίηση των υπηρεσιών για όλο το cluster. Έχει ένα κεντρικό πίνακα για παρακολούθηση της υγείας και της κατάστασης όλου του cluster. Χρησιμοποιεί το Ganglia σύστημα παρακολούθησης για συλλογή μετρικών του συστήματος και το Nagios για προειδοποιήσεις για κρίσιμες καταστάσεις για το σύστημα (αστοχία κόμβου, χαμηλό υπόλοιπο του δίσκου κ.α.). Επίσης, βοηθά τους προγραμματιστές εφαρμογών να χρησιμοποιήσουν τα REST API του Ambari για να ενσωματώσουν όλα τα πλεονεκτήματα του στην εφαρμογή τους.

- **Το Apache Avro** – Το Apache Avro, είναι ένα σύστημα σειριοποίησης των δεδομένων, παρέχοντας πλούσιες δομές δεδομένων, μία συμπαγή, γρήγορη και δυαδική μορφή δεδομένων και απομακρυσμένη κλήση συναρτήσεων (RPC). Επιπλέον, ενσωματώνεται εύκολα με δυναμικές γλώσσες. Η παραγωγή κώδικα δεν είναι απαραίτητη για την ανάγνωση ή εγγραφή αρχείων δεδομένων, ούτε για την χρήση ή υλοποίηση των πρωτοκόλλων RPC. Η παραγωγή κώδικα είναι μόνο προαιρετική βελτιστοποίηση, που αξίζει μόνο όταν χρησιμοποιούνται στατικές γλώσσες.
- **Το Apache Cassandra** – Το Apache Cassandra είναι μία βάση δεδομένων με πολλαπλούς κύριους κόμβους, χωρίς ένα μοναδικό σημείο αστοχίας. Παρέχει επεκτασιμότητα και υψηλή διαθεσιμότητα χωρίς να θυσιάζει τις επιδόσεις. Είναι η καταλληλότερη επιλογή, για δεδομένα κρίσιμα στον στόχο μίας εταιρίας, χάρη στην γραμμική επεκτασιμότητα και την αποδεδειγμένη χαμηλή ανοχή σε σφάλματα που παρέχει σε cloud υποδομές. Έχει την καλύτερη υποστήριξη για την αναπαραγωγή και την αντιγραφή δεδομένων σε πολλά datacenter, παρέχοντας χαμηλή καθυστέρηση για τους χρήστες και την ησυχία για τους διαχειριστές, ότι ακόμη και σε περίπτωση διακοπής του ρεύματος σε μια περιοχή θα υπάρχει υποστήριξη. Το μοντέλο δεδομένων του Cassandra, προσφέρει την ευκολία των δεικτών (column indexes) με τις επιδόσεις των ενημερώσεων σε μορφή log (log-structured updates). Τέλος, παρέχει υποστήριξη για αποκανονικοποίηση, materialized views της βάσης και ισχυρές δυνατότητες caching.
- **Το Apache Chukwa** – Το Apache Chukwa, είναι ένα σύστημα συλλογής δεδομένων για την διαχείριση μεγάλων κατανεμημένων συστημάτων. Το Chukwa βασίζεται στο HDFS και το Map/Reduce framework και υιοθετεί την επεκτασιμότητα και την ευρωστία του Hadoop. Επίσης, περιλαμβάνει ένα ευέλικτο και ισχυρό toolkit για την παρουσίαση, την παρακολούθηση και την ανάλυση των αποτελεσμάτων, για την καλύτερη χρήση των συλλεγμένων δεδομένων.
- **Το Apache HBase** – Το Apache HBase, είναι ένα μία επεκτάσιμη και κατανεμημένη βάση δεδομένων που υποστηρίζει δομημένη αποθήκευση δεδομένων για μεγάλους πίνακες και παρέχει τυχαία και σε πραγματικό χρόνο πρόσβαση ανάγνωσης/εγγραφής στα δεδομένα. Ο στόχος του

project είναι η αποθήκευση πολύ μεγάλων πινάκων – με δισεκατομμύρια γραμμές και εκατομμύρια στήλες – σε cluster με χαμηλό κόστος. Το Apache HBase, είναι ένα μοντέλο αποθήκευσης, κατανεμημένο, με βάση τις στήλες (column-oriented), βασισμένο στο BigTable της Google, που είναι ένα κατανεμημένο σύστημα αποθήκευσης για δομημένα δεδομένα. Το BigTable εκμεταλλεύεται την κατανεμημένη αποθήκευση των δεδομένων που παρέχεται από το Google File System και το HBase παρέχει παρόμοιες δυνατότητες με το BigTable και λειτουργεί πάνω στο Hadoop και το HDFS. Μερικά από τα χαρακτηριστικά του HBase είναι:

- Γραμμική και προσαρμόσιμη επεκτασιμότητα.
- Αυστηρά σταθερές αναγνώσεις και εγγραφές.
- Αυτόματο και παραμετροποιήσιμο διαχωρισμό των πινάκων.
- Αυτόματη υποστήριξη failover μεταξύ των RegionServers.
- Βασικές και εύκολες στην χρήση κλάσεις για την υποστήριξη των Hadoop MapReduce εργασιών με HBase πίνακες.
- Εύκολο στην χρήση Java API για την πρόσβαση των client.
- Block cache και Bloom φίλτρα για ερωτήματα σε πραγματικό χρόνο προς την βάση.
- Προβλέψιμο push down των ερωτημάτων μέσω φίλτρων που εκτελούνται στον server.
- Πύλη Thrift και μία RESTful web υπηρεσία που υποστηρίζει XML, Protobuf και επιλογές για κωδικοποίηση δυαδικών δεδομένων.
- Επεκτάσιμο shell βασισμένο σε jruby (JIRB).
- Υποστήριξη για εξαγωγή μετρικών μέσω του μετρικού υποσυστήματος του Hadoop ή μέσω του JMX, σε αρχεία ή στο Ganglia.

- **Το Apache Hive** – Το Apache Hive είναι μία υποδομή αποθήκευσης δεδομένων που αρχικά αναπτύχθηκε από το Facebook και παρέχει σύνοψη δεδομένων και ερωτήματα τύπου ad hoc. Διευκολύνει την διαχείριση και τα ερωτήματα προς μεγάλα dataset που βρίσκονται σε κατανεμημένη αποθήκευση. Το Hive προσφέρει ένα μηχανισμό να ορίσεις μια δομή σε αυτά τα δεδομένα και να εφαρμόσεις ερωτήματα προς αυτά σε μια γλώσσα τύπου SQL που λέγετε HiveQL. Ταυτόχρονα, παρέχει την δυνατότητα στους προγραμματιστές να ενσωματώσουν τους παραμετροποιημένους

mapper και reducer τους, όταν είναι δύσκολο ή ανεπαρκή να εφαρμόσουν την λογική τους στην HiveQL. Τα metadata αποθηκεύονται σε μία Apache Derby ενσωματωμένη βάση δεδομένων, αλλά μπορούν να χρησιμοποιηθούν και άλλες τύπου client/server βάσεις δεδομένων όπως η MySQL. Οι μορφές αρχείων που υποστηρίζει είναι TEXTFILE (αρχείο κειμένου), SEQUENCEFILE (σειριακό αρχείο), ORC και RCFILE (αρχεία σε μορφή στηλών, βελτιστοποιημένα για το Hadoop, για γρήγορη φόρτωση, γρήγορη επεξεργασία, αποτελεσματική χρήση του χώρου, και προσαρμοστικότητα σε δυναμικά μοτίβα πρόσβασης στα δεδομένα). Μερικά από τα χαρακτηριστικά του Hive είναι:

- Δεικτοδότηση για επιτάχυνση (compaction και Bitmap index) και σχεδιάζονται και άλλες τεχνικές δεικτοδότησης.
- Διαφορετικοί τύποι αποθήκευσης δεδομένων όπως: απλό κείμενο, RCFile, HBase, ORC κ.α.
- Αποθήκευση των metadata σε ένα RDBMS (σύστημα διαχείρισης σχεσιακής βάσης δεδομένων), μειώνοντας τον χρόνο για τους σημασιολογικούς ελέγχους κατά την εκτέλεση των ερωτημάτων.
- Μπορεί να λειτουργήσει σε συμπιεσμένα δεδομένα που είναι αποθηκευμένα στο οικοσύστημα του Hadoop με τους αλγορίθμους gzip, bzip2, snappy κ.α.
- Ενσωματωμένες συναρτήσεις για διαχείριση ημερομηνιών, αλφαριθμητικών και άλλα εργαλεία εξόρυξης δεδομένων αλλά και επέκταση αυτών για ειδικές περιπτώσεις.
- Ερωτήματα τύπου SQL που μετατρέπονται έμμεσα σε εργασίες MapReduce.

Παρόλα αυτά δεν υποστηρίζει, materialized views των πινάκων, έχει περιορισμένη υποστήριξη υπό-ερωτημάτων και άλλους περιορισμούς που όμως σχεδιάζετε να προστεθούν.

- **Το Apache HCatalog** – Το Apache HCatalog είναι ένα σύστημα για την διαχείριση των πινάκων και της αποθήκευσης για το Hadoop, το οποίο προσφέρει στους χρήστες με διαφορετικά δεδομένα – Apache Pig, Apache MapReduce και Apache Hive – την δυνατότητα να διαβάσουν και να γράψουν εύκολα δεδομένα στο grid. Η αφηρημένη έννοια των πινάκων στο

HCatalog παρουσιάζει στους χρήστες μια σχεσιακή όψη των δεδομένων στο HDFS και εγγυάται ότι δεν χρειάζεται οι χρήστες να ανησυχούν για την τοποθεσία και τον τύπο (format) της αποθήκευσης των δεδομένων. Το HCatalog, παρουσιάζει τα δεδομένα από αρχεία τύπου RCFile, από αρχεία κειμένου και από σειριακά αρχεία σε πινακοειδή όψη. Επιπλέον, παρέχει REST API, για να μπορούν εξωτερικά συστήματα να έχουν πρόσβαση στα metadata των πινάκων. Το Apache HCatalog παρέχει τα εξής πλεονεκτήματα στους διαχειριστές των grid:

- Ελευθερώνει τον χρήστη, από το να ξέρει που είναι αποθηκευμένα τα δεδομένα χάρη στην αφηρημένη έννοια που έχει για τους πίνακες.
- Προσφέρει την δυνατότητα ειδοποιήσεων σχετικά με την διαθεσιμότητα των δεδομένων.
- Προσφέρει ορατότητα για τον «καθαρισμό» και την αρχειοθέτηση των δεδομένων.

Το HCatalog, υποστηρίζει την ανάγνωση και την εγγραφή σε οποιουδήποτε τύπου αρχείο στο οποίο μπορεί να εγγραφεί ένα Hive SerDe (serializer – deserializer). Προσφέρει, υποστήριξη για αρχεία RCFile, CSV, JSON και σειριακά αρχεία. Για την χρήση ενός ειδικού format ο χρήστης πρέπει να παρέχει το InputFormat, το OutputFormat και το SerDe. Το HCatalog, βασίζεται και λειτουργεί πάνω στο σύστημα αποθήκευσης metadata του Hive και χρησιμοποιεί στοιχεία της Hive DDL (Data Definition Language). Παρέχει διεπαφές (interfaces) για ανάγνωση και εγγραφή για το Pig και το MapReduce και χρησιμοποιεί την διεπαφή γραμμής εντολών του Hive για έκδοση εντολών ορισμού δεδομένων και εξερεύνηση metadata. Επίσης, παρουσιάζει μια REST διεπαφή για να επιτρέπει σε εξωτερικά εργαλεία πρόσβαση σε Hive DDL εργασίες, όπως το “create table” (δημιουργία πίνακα) και “describe table” (περιγραφή πίνακα). Στο HCatalog τα δεδομένα αποθηκεύονται σε πίνακες οι οποίοι μπορούν να τοποθετηθούν σε βάσεις δεδομένων και μπορούν να διαχωριστούν σε ένα ή και παραπάνω κλειδιά. Για μία δεδομένη τιμή ενός κλειδιού (ή ένα σετ κλειδιών), θα υπάρχει ένα διαχώρισμα (partition), που περιέχει όλες τις γραμμές με αυτή την τιμή (ή σετ τιμών).

- **Το Apache Pig** – Το Apache Pig, σου επιτρέπει να γράψεις πολύπλοκους MapReduce μετασχηματισμούς, χρησιμοποιώντας μία απλή scripting γλώσσα. Η Pig Latin (το όνομα της γλώσσας) ορίζει ένα σετ μετασχηματισμών σε δεδομένα όπως aggregate, join και sort. Το Pig μεταφράζει τα Pig Latin scripts σε MapReduce εργασίες, για να μπορούν να εκτελεστούν στο Hadoop. Η Pig Latin μπορεί να επεκταθεί χρησιμοποιώντας συναρτήσεις ορισμένες από τον χρήστη (UDF – User Defined Functions), τις οποίες μπορεί να γράψει ο χρήστης σε Java ή σε μία άλλη scripting γλώσσα και να τις καλέσει απευθείας μέσα από την Pig Latin. Το Pig σχεδιάστηκε για να εκτελεί πολλές διαδοχικές εργασίες σε δεδομένα, κάνοντας το, ιδανικό για τρείς κατηγορίες εργασιών με μεγάλες απαιτήσεις σε δεδομένα (Big Data): επεξεργασία σε κανονικούς και μεγάλους αγωγούς δεδομένων (data pipelines) τύπου εξαγωγή-μετασχηματισμός-φόρτωση (ETL, extract-transform-load), έρευνα σε ακατέργαστα δεδομένα και επαναληπτική επεξεργασία δεδομένων. Όποια και να είναι η χρήση, το Pig προσφέρει:
    - **Επεκτασιμότητα.** Οι χρήστες μπορούν να δημιουργήσουν επιπλέον συναρτήσεις για ειδικές εφαρμογές ανάλογα τις ανάγκες τους.
    - **Ευκολία στον προγραμματισμό.** Σύνθετες εργασίες που περιέχουν αλληλένδετους μετασχηματισμούς δεδομένων μπορούν να απλοποιηθούν και να κωδικοποιηθούν ως ακολουθίες ροής δεδομένων. Τα προγράμματα Pig, πραγματοποιούν μεγάλες και σημαντικές εργασίες, αλλά είναι εύκολα στην εγγραφή και την συντήρηση.
    - **Αυτόματη βελτιστοποίηση.** Το σύστημα βελτιστοποιεί αυτόματα την εκτέλεση των Pig εργασιών, αφήνοντας τους χρήστες να επικεντρωθούν στην σημασιολογία.

  Το Pig εκτελείται στο Hadoop και χρησιμοποιεί το MapReduce και το HDFS. Η γλώσσα ανάπτυξης λέγεται Pig Latin, η οποία αποτελεί μια αφαίρεση τους ιδιώματος της Java στο MapReduce και η μορφή της είναι παρόμοια με SQL. Αν και η SQL είναι μία δηλωτική γλώσσα ενώ η Pig Latin είναι μία γλώσσα ροής. Η SQL είναι πολύ καλή για ερωτήματα προς τα δεδομένα, ενώ η Pig Latin σου επιτρέπει να γράψεις μία ροή δεδομένων που

περιγράφει πως θα μετασχηματιστούν τα δεδομένα σου. Επίσης, τα script σε Pig Latin μπορούν να είναι γραφήματα (δηλαδή δεν απαιτούν μόνο μία έξοδο), επιτρέποντας να φτιαχτούν πολύπλοκες ροές δεδομένων με πολλαπλές εισόδους, μετασχηματισμούς και εξόδους. Οι χρήστες μπορούν να γράψουν δικές τους συναρτήσεις για επέκταση της Pig Latin, χρησιμοποιώντας Java, Python, Ruby ή άλλες γλώσσες scripting. Οι χρήστες μπορούν να εκτελέσουν τα Pig scripts με δύο τρόπους:

  - **Τοπική εκτέλεση.** Με πρόσβαση σε ένα κόμβο, όλα τα αρχεία εγκαθίστανται και εκτελούνται χρησιμοποιώντας τον τοπικό κόμβο και σύστημα αρχείων.
  - **MapReduce εκτέλεση.** Αυτός είναι ο προκαθορισμένος τρόπος εκτέλεσης, για τον οποίο απαιτείται πρόσβαση σε ένα Hadoop cluster.

- **Το Apache Oozie** – Το Apache Oozie, είναι μία εφαρμογή web γραμμένη σε Java, η οποία χρησιμοποιείται για τον προγραμματισμό εργασιών για το Apache Hadoop. Το Oozie συνδυάζει πολλές εργασίες διαδοχικά σε μία λογική μονάδα εργασιών. Είναι ενσωματωμένο με το Hadoop stack και υποστηρίζει Hadoop εργασίες για το Apache MapReduce, το Apache Pig, το Apache Hive και το Apache Sqoop. Επιπλέον, μπορεί να χρησιμοποιηθεί για να προγραμματίσει εργασίες ιδιαίτερες για κάποιο σύστημα, όπως Java προγράμματα ή shell scripts. Υπάρχουν, δύο βασικοί τύποι εργασιών Oozie:
  - **Οι Oozie Workflow** εργασίες είναι κατευθυνόμενοι ακυκλικοί γράφοι (DAGs – Directed Acyclical Graphs), οι οποίοι περιγράφουν μια ακολουθία ενεργειών για εκτέλεση, αλλά η Workflow εργασία πρέπει να περιμένει.
  - **Οι Oozie Coordinator** εργασίες είναι επαναλαμβανόμενες Oozie Workflow εργασίες οι οποίες ενεργοποιούνται από την διαθεσιμότητα του χρόνου και των δεδομένων.
  - **Τέλος, το Oozie Bundle** το οποίο παρέχει ένα τρόπο για ομαδοποίηση πολλαπλών Coordinator και Workflow εργασιών και ένα τρόπο για την διαχείριση του κύκλου ζωής αυτών τον εργασιών.

Το Apache Oozie επιτρέπει στους διαχειριστές του Hadoop να δημιουργούν πολύπλοκους μετασχηματισμούς δεδομένων από πολλαπλές συνθετικές εργασίες. Αυτό επιτρέπει μεγαλύτερο έλεγχο στις πολύπλοκες εργασίες και κάνει πιο εύκολη την επανάληψη αυτών των εργασιών σε συγκεκριμένα χρονικά διαστήματα. Επίσης, βοηθά τους διαχειριστές να αντλούν μεγαλύτερη αξία από την επένδυση τους στο Hadoop. Μία Oozie Workflow εργασία, όπως αναφέρθηκε είναι μία συλλογή από εργασίες διατεταγμένες σε κατευθυνόμενο ακυκλικό γράφο (DAG – Directed Acyclic Graph). Οι κόμβοι ελέγχου (control) ορίζουν την χρονολογική σειρά των εργασιών, θέτοντας κανόνες για την εκκίνηση και τον τερματισμό ενός workflow, οι οποίοι ελέγχουν την πορεία εκτέλεσης του workflow με κόμβους απόφασης, διακλάδωσης και ένωσης (decision, fork και join κόμβοι). Οι κόμβοι ενέργειας (action) ενεργοποιούν την εκτέλεση των εργασιών. Αλλά ενώ το Oozie ενεργοποιεί τις workflow εργασίες το Hadoop MapReduce τις εκτελεί. Αυτό επιτρέπει το Oozie να εκμεταλλευτεί άλλες δυνατότητες μέσα στο Hadoop stack για να εξισορροπήσει τον φόρτο των εργασιών και να χειριστεί τις βλάβες κατά την εκτέλεση. Το Oozie ανιχνεύει την ολοκλήρωση των εργασιών μέσω callback και polling. Όταν το Oozie ξεκινά μία εργασία, παρέχει ένα μοναδικό callback HTTP URL στην εργασία και ενημερώνει αυτό το URL όταν ολοκληρωθεί. Αν η εργασία αποτύχει να καλέσει το callback URL, το Oozie μπορεί να «ρωτήσει» (poll) την εργασία για να ενημερωθεί αν ολοκληρώθηκε. Πολλές φορές είναι απαραίτητο να εκτελεστούν Oozie workflows σε συγκεκριμένα και τακτικά χρονικά διαστήματα, αλλά σε συντονισμό με τα απρόβλεπτα επίπεδα της διαθεσιμότητας των δεδομένων ή των γεγονότων. Σε αυτές τις περιπτώσεις, ο Oozie Coordinator επιτρέπει την μοντελοποίηση των μηχανισμών ενεργοποίησης (triggers) της εκτέλεσης του workflow, ως προς τα δεδομένα και τον χρόνο ή τα γεγονότα που απαιτούνται. Αφού ολοκληρωθούν και ικανοποιηθούν αυτές οι απαιτήσεις, τότε ξεκινά η workflow εργασία. Ο Oozie Coordinator μπορεί επίσης να διαχειριστεί πολλαπλά workflow που είναι εξαρτημένα από το αποτέλεσμα των workflow που θα ακολουθήσουν. Οι έξοδοι αυτών των μεταγενέστερων workflow, γίνονται η είσοδος στο επόμενο workflow και αυτή η αλληλουχία ονομάζεται «αγωγός εφαρμογής δεδομένων» (data application pipeline).

- **Το Apache Zookeeper** – Το Apache Zookeeper παρέχει λειτουργικές υπηρεσίες για ένα Hadoop cluster, όπως μία κατανεμημένη υπηρεσία ρύθμισης του cluster, μια υπηρεσία συγχρονισμού και ένα μητρώο ονομάτων (naming registry) για κατανεμημένα συστήματα. Οι κατανεμημένες εφαρμογές το χρησιμοποιούν για να αποθηκεύσουν και να μεσιτεύσουν τις ενημερώσεις για σημαντικές πληροφορίες ρυθμίσεων. Το Zookeeper παρέχει μία πολύ απλή διεπαφή, υπηρεσίες για την λειτουργία ενός cluster και έχει κάποια βασικά πλεονεκτήματα, τα οποία είναι τα εξής:
    - **Ταχύτητα.** Το Zookeeper είναι ιδιαίτερα γρήγορο, όπου το φόρτο της εργασίας αποτελείται κυρίως από ανάγνωση σε δεδομένα παρά από εγγραφή σε δεδομένα. Η ιδανική αναλογία ανάγνωσης/εγγραφής είναι περίπου 10:1.
    - **Αξιοπιστία.** Το Zookeeper, αναπαράγεται σε ένα σετ κόμβων (που λέγεται ανσάμπλ – ensemble) και οι server γνωρίζουν ο ένας για τον άλλο. Όσο μία βασική ομάδα των server είναι διαθέσιμη, η υπηρεσία του Zookeeper θα είναι επίσης διαθέσιμη και δεν υπάρχει μοναδικό σημείο αστοχίας.
    - **Απλότητα.** Το Zookeeper κρατά ένα ιεραρχικό ονοματοχώρο παρόμοιο με αυτόν που έχουν τα αρχεία και οι κατάλογοι.
    - **Διάταξη.** Το Zookeeper διατηρεί αρχείο όλων των συναλλαγών, το οποίο μπορεί να χρησιμοποιηθεί για αφαιρέσεις υψηλότερου επιπέδου όπως τα αρχέτυπα συγχρονισμού (synchronization primitives).

  Το Zookeeper επιτρέπει τις κατανεμημένες διεργασίες να συντονιστούν με τις άλλες μέσω ενός κοινόχρηστου ιεραρχικού ονοματοχώρου μητρώων δεδομένων, γνωστών ως znodes. Κάθε znode προσδιορίζεται από ένα μονοπάτι (path) με στοιχεία (path elements) τα οποία διαχωρίζονται από μία κάθετο (“/”). Εκτός από την ρίζα (root), κάθε znode έχει ένα γονιό και κανένας znode δεν μπορεί να διαγραφεί αν έχει παιδιά. Αυτό έχει πολλά κοινά με ένα κανονικό σύστημα αρχείων αλλά το Zookeeper προσφέρει αξιοπιστία μέσω πλεονάζον (redundant) υπηρεσιών. Κάθε υπηρεσία αναπαράγεται σε πολλούς κόμβους και κάθε κόμβος διατηρεί μια εικόνα του δέντρου των δεδομένων και των καταγραφών των συναλλαγών στην

μνήμη. Οι client συνδέονται σε ένα μοναδικό Zookeeper server και διατηρούν μία σύνδεση TCP μέσω της οποίας στέλνουν και λαμβάνουν αιτήσεις και απαντήσεις αντίστοιχα. Αυτή η αρχιτεκτονική επιτρέπει στο Zookeeper να παρέχει υψηλή ωφέλιμη μεταφορά δεδομένων, υψηλή διαθεσιμότητα με χαμηλή καθυστέρηση, αλλά το μέγεθος της βάσης δεδομένων που μπορεί να διαχειριστεί περιορίζεται από την μνήμη.

- **Το Apache Sqoop** – Το Apache Sqoop, είναι ένα εργαλείο σχεδιασμένο για την αποδοτική μεταφορά όγκου δεδομένων μεταξύ του Apache Hadoop και αποθηκών δομημένων δεδομένων (structured datastores) όπως είναι οι σχεσιακές βάσεις δεδομένων. Το Sqoop εισάγει δεδομένα από εξωτερικά δομημένα datastore στο HDFS ή συσχετιζόμενα συστήματα όπως το Hive και το HBase. Το Sqoop μπορεί επίσης να χρησιμοποιηθεί για να εξάγει δεδομένα από το Hadoop σε εξωτερικά δομημένα datastore, όπως είναι οι σχεσιακές βάσεις δεδομένων και οι επιχειρησιακές αποθήκες δεδομένων (enterprise data warehouses). Το Sqoop μπορεί να δουλέψει με σχεσιακές βάσεις όπως: την Teradata, την Netezza, την Oracle, την MySQL, την Postgres και την HSQLDB. Τα χαρακτηριστικά του Sqoop είναι τα εξής:
    - **Επιτρέπει εισαγωγή δεδομένων** από εξωτερικά datastore και επιχειρησιακές αποθήκες δεδομένων στο Hadoop.
    - **Παραλληλοποιεί την μεταφορά των δεδομένων** για γρήγορες επιδόσεις και βέλτιστη αξιοποίηση των πόρων του συστήματος.
    - **Γρήγορη αντιγραφή δεδομένων** από εξωτερικά συστήματα στο Hadoop.
    - **Κάνει την ανάλυση των δεδομένων πιο αποδοτική.**
    - **Μετριάζει τον υπερβολικό φόρτο** σε εξωτερικά συστήματα.

  Το Sqoop παρέχει ένα εύκολο στην προσαρμογή (pluggable) μηχανισμό σύνδεσης για την βέλτιστη διασύνδεση με εξωτερικά συστήματα. Το API επέκτασης του Sqoop παρέχει ένα βολικό framework για την ανάπτυξη νέων μηχανισμών σύνδεσης, οι οποίοι μπορούν να ενσωματωθούν σε διάφορες εγκαταστάσεις του Sqoop για να παρέχουν συνδεσιμότητα με διάφορα συστήματα. Το Sqoop από μόνο έχει διάφορους μηχανισμούς σύνδεσης οι οποίοι μπορούν να χρησιμοποιηθούν για δημοφιλείς βάσεις δεδομένων και συστήματα αποθήκευσης δεδομένων.

- **Το Apache Mahout** – Το Apache Mahout είναι μία βιβλιοθήκη από επεκτάσιμους αλγορίθμους μηχανικής μάθησης, οι οποίοι έχουν εφαρμοστεί στο Apache Hadoop χρησιμοποιώντας το μοντέλο του MapReduce. Η μηχανική μάθηση είναι ένας κλάδος της τεχνητής νοημοσύνης, ο οποίος είναι επικεντρωμένος στην ενεργοποίηση της ικανότητας μάθησης των μηχανών χωρίς να έχουν προγραμματιστεί ρητά για το τι θα κάνουν και χρησιμοποιείται πολύ συχνά για την βελτίωση της μελλοντικής επίδοσης βασισμένη σε προηγούμενα αποτελέσματα και στοιχεία, με πολλές εφαρμογές σε όλους σχεδόν τους κλάδους των επιστημών. Στο Mahout, αφού ο μεγάλος όγκος των δεδομένων (Big Data) αποθηκευτεί στο HDFS, εκείνο παρέχει τα απαραίτητα επιστημονικά εργαλεία δεδομένων, για την αυτόματη εύρεση προτύπων/μοτίβων με σημασία μέσα σε αυτά. Το Mahout έχει ως στόχο να κάνει πιο γρήγορη και πιο εύκολη την διαδικασία μετατροπής του μεγάλου όγκου δεδομένων (Big Data) σε πληροφορία με μεγάλη αξία. Το Mahout υποστηρίζει τέσσερις βασικές περιπτώσεις χρήσης που σχετίζονται με τα δεδομένα:
    - **Συνεργατικό φιλτράρισμα (Collaborative filtering)** – όπου γίνετε εξόρυξη της συμπεριφοράς του χρήστη και γίνονται προτάσεις προϊόντων (π.χ. οι προτάσεις κατά την αγορά στο Amazon).
    - **Ομαδοποίηση (Clustering)** – κατά την οποία το σύστημα παίρνει αντικείμενα που ανήκουν σε μια συγκεκριμένη κλάση (για παράδειγμα ιστοσελίδες ή άρθρα εφημερίδων) και τα οργανώνει σε φυσικά επερχόμενες (γίνονται από το σύστημα) ομάδες, ώστε τα αντικείμενα που ανήκουν στην ίδια ομάδα να έχουν παρόμοια χαρακτηριστικά μεταξύ τους.
    - **Ταξινόμηση (Classification)** - κατά την οποία το σύστημα μαθαίνει από τις υπάρχουσες κατηγοριοποιήσεις και μετά ταξινομεί αυτόματα τα αντικείμενα χωρίς κατηγορία στην καλύτερη και πιο ταιριαστή κατηγορία
    - **Συχνή εξόρυξη στοιχειοσυνόλου (Frequent itemset mining)** – κατά την οποία το σύστημα αναλύει αντικείμενα που ανήκουν σε μία ομάδα (π.χ. αντικείμενα σε ένα καλάθι αγορών ή όρους σε μία συνεδρία ερωτημάτων) και μετά αναγνωρίζει ποια αντικείμενα συνήθως παρουσιάζονται μαζί.

Το Mahout όπως αναφέρθηκε παρέχει υλοποιήσεις διαφόρων αλγορίθμων μηχανικής μάθησης, μερικοί από τους οποίους είναι σε τοπική μορφή και κάποιοι σε κατανεμημένη μορφή (για χρήση με το Hadoop). Κάθε αλγόριθμος στην βιβλιοθήκη του Mahout μπορεί να κληθεί χρησιμοποιώντας την γραμμή εντολών του Mahout. Παρακάτω ακολουθεί μία λίστα αλγορίθμων οι οποίοι είναι σε κατανεμημένη μορφή για χρήση με το Hadoop και είναι ταξινομημένοι με βάση τις τέσσερις κατηγορίες που αναφέρθηκαν παραπάνω:

Πίνακας 1 "Αλγόριθμοι Mahout για χρήση με το Hadoop"

Αλγόριθμος	Κατηγορία	Περιγραφή
Distributed Item-based Collaborative Filtering	**Συνεργατικό φιλτράρισμα (Collaborative filtering)**	Υπολογίζει την προτίμηση ενός χρήστη για ένα αντικείμενο κοιτάζοντας τις προτιμήσεις του/της για παρόμοια αντικείμενα.
Collaborative Filtering Using a Parallel Matrix Factorization	**Συνεργατικό φιλτράρισμα (Collaborative filtering)**	Μεταξύ ενός πίνακα (matrix) από αντικείμενα που δεν έχει δει ο χρήστης, προβλέπει ποια αντικείμενα μπορεί να προτιμήσει.
Canopy Clustering	**Ομαδοποίηση (Clustering)**	Για προ-επεξεργασία των δεδομένων πριν την χρήση του K-means ή του Hierarchical clustering algorithm.
Dirichlet Process Clustering	**Ομαδοποίηση (Clustering)**	Εκτελεί ομαδοποίηση Bayesian mixture.
Fuzzy K-Means	**Ομαδοποίηση (Clustering)**	Ανακαλύπτει soft cluster (ομάδες) όπου ένα συγκεκριμένο σημείο μπορεί να ανήκει σε παραπάνω από ένα cluster.
Hierarchical Clustering	**Ομαδοποίηση**	Φτιάχνει μία ιεραρχία από

	(Clustering)	cluster (ομάδες) χρησιμοποιώντας προσέγγιση «από κάτω προς τα πάνω» (agglomerative “bottom up”) ή «από πάνω προς τα κάτω» (divisive “top down”).
K-Means Clustering	**Ομαδοποίηση (Clustering)**	Στοχεύει να διαχωρίσει n παρατηρήσεις σε k clusters (ομάδες), όπου κάθε παρατήρηση ανήκει στο cluster με το πιο κοντινό μέσο.
Latent Dirichlet Allocation	**Ομαδοποίηση (Clustering)**	Αυτόματα και από κοινού ομαδοποιεί λέξεις σε «θέματα» και έγγραφα σε μίξη από «θέματα».
Mean Shift Clustering	**Ομαδοποίηση (Clustering)**	Για την εύρεση τρόπων ή ομάδων στο δισδιάστατο χώρο, όπου ο αριθμός των ομάδων είναι άγνωστος.
Αλγόριθμος	**Κατηγορία**	**Περιγραφή**
Minhash Clustering	**Ομαδοποίηση (Clustering)**	Για γρήγορη εκτίμηση ομοιοτήτων μεταξύ δύο dataset.
Spectral Clustering	**Ομαδοποίηση (Clustering)**	Εύρεση σημείων ομαδοποίησης (cluster) με τη χρήση ιδιοδιανυσμάτων πινάκων που προκύπτουν από τα δεδομένα.
Bayesian	**Ταξινόμηση (Classification)**	Χρησιμοποιείται για την ομαδοποίηση αντικειμένων σε δυαδικές κατηγορίες.
Random Forests	**Ταξινόμηση (Classification)**	Ένα σύνολο μεθόδων μάθησης για ταξινόμηση

		(και παλινδρόμηση) που λειτουργεί κατασκευάζοντας ένα πλήθος δέντρων αποφάσεων.
Parallel FP Growth Algorithm	**Συχνή εξόρυξη στοιχειοσυνόλου (Frequent itemset mining)**	Αναλύει αντικείμενα σε μία ομάδα και μετά αναγνωρίζει ποια από αυτά συνήθως παρουσιάζονται μαζί.

- **Το Apache Flume** – Το Apache Flume, είναι μία κατανεμημένη, αξιόπιστη και διαθέσιμη υπηρεσία για αποδοτική συλλογή, άθροιση και μετακίνηση μεγάλου όγκου συνεχής ροής δεδομένων (streaming data), στο HDFS. Έχει μία ευέλικτη αρχιτεκτονική βασισμένη σε συνεχές ροές δεδομένων ροής (streaming data flows) και είναι ισχυρή και ανεκτική σε σφάλματα, με ρυθμιζόμενους μηχανισμούς αξιοπιστίας για failover και ανάκτηση από σφάλματα. Το Flume επιτρέπει στο Hadoop να αξιοποιεί στο έπακρο τα πολύτιμα δεδομένα καταγραφής. Συγκεκριμένα το Flume επιτρέπει στους χρήστες να:
    - **Εισάγουν ροές δεδομένων (stream data) από πολλαπλές πηγές** στο Hadoop για ανάλυση.
    - **Συλλέξουν καταγραφές (logs) web υψηλού όγκου** σε πραγματικό χρόνο.
    - **Απομονώσουν** τον εαυτό τους από παροδικές ανόδους, όταν ο ρυθμός των εισερχόμενων δεδομένων ξεπερνά το ρυθμό με τον οποίο τα δεδομένα μπορούν να γραφτούν στον προορισμό.
    - **Εγγυηθούν την παράδοση των δεδομένων.**
    - **Επεκταθούν οριζόντια** για να χειριστούν επιπλέον όγκο δεδομένων.

  Η αρχιτεκτονική υψηλού επιπέδου του Flume, επικεντρώνεται στην παροχή μίας βελτιωμένης βάσης κώδικα (codebase) η οποία είναι εύκολη στην χρήση και εύκολα επεκτάσιμη. Η ομάδα του project το έχει σχεδιάσει με τα ακόλουθα μέρη (component):

- **Γεγονός (Event)** – μία μοναδική μονάδα δεδομένων η οποία μεταφέρεται από το Flume (συνήθως μία καταχώρηση του αρχείου καταγραφής).
- **Πηγή (Source)** – η οντότητα μέσω της οποία τα δεδομένα εισέρχονται στο Flume. Οι πηγές είτε ενεργά δημοσκοπούν/ρωτούν (poll) για τα δεδομένα ή παθητικά περιμένουν τα δεδομένα να παραδοθούν σε αυτές. Μία ποικιλία από πηγές επιτρέπει τα δεδομένα τους να συλλεχθούν, όπως οι log4j καταγραφές και τα syslogs.
- **Νεροχύτης (Sink)** – η οντότητα που μεταφέρει τα δεδομένα στον προορισμό. Μία ποικιλία από sinks επιτρέπουν τα δεδομένα να μεταδοθούν σε μία σειρά από προορισμούς. Ένα παράδειγμα είναι ο HDFS sink που γράφει γεγονότα στο HDFS.
- **Κανάλι (Channel)** – ο αγωγός μεταξύ της πηγής (source) και του νεροχύτη (sink). Οι πηγές καταπίνουν τα γεγονότα μέσα στο κανάλι και οι νεροχύτες αποστραγγίζουν το κανάλι.
- **Πράκτορας (Agent)** – οποιαδήποτε φυσική Java εικονική μηχανή που τρέχει το Flume. Είναι μία συλλογή από πηγές (sources), νεροχύτες (sinks) και κανάλια (channels).
- **Πελάτης (Client)** – παράγει και μεταδίδει το γεγονός (event) στην πηγή (source) λειτουργώντας εντός του πράκτορα (agent).

Μία ροή στο Flume ξεκινά από τον client. Ο client μεταδίδει το γεγονός σε μία πηγή λειτουργώντας εντός του πράκτορα. Η πηγή που λαμβάνει το γεγονός, το παραδίδει σε ένα ή περισσότερα κανάλια. Αυτά τα κανάλια αποστραγγίζονται από ένα ή περισσότερους νεροχύτες λειτουργώντας εντός του ίδιου πράκτορα. Τα κανάλια επιτρέπουν την αποσύνδεση του ρυθμού πρόσληψης από τον ρυθμό αποστράγγισης χρησιμοποιώντας το γνωστό μοντέλο παραγωγού-καταναλωτή (producer-consumer) της ανταλλαγής δεδομένων. Όταν ξαφνικές άνοδοι (spikes) στην δραστηριότητα από την πλευρά του client, προκαλούν την πιο γρήγορη παραγωγή των δεδομένων από ότι μπορεί να χειριστεί η τροφοδοτούμενη χωρητικότητα του προορισμού, το μέγεθος του καναλιού αυξάνεται. Αυτό επιτρέπει τις πηγές να συνεχίζουν την κανονική λειτουργία τους κατά την διάρκεια της

ανόδου της δραστηριότητας. Οι Flume πράκτορες μπορούν να συνδεθούν μεταξύ τους, συνδέοντας τον νεροχύτη ενός πράκτορα με την πηγή ενός άλλου πράκτορα. Αυτό επιτρέπει την δημιουργία πολύπλοκων τοπολογιών ροής δεδομένων. Το Flume είναι σχεδιασμένο για να είναι πολύ αξιόπιστο, οπότε καθόλου δεδομένα δεν χάνονται κατά την κανονική λειτουργία. Επίσης, υποστηρίζει δυναμική αναδιαμόρφωση χωρίς την ανάγκη επανεκκίνησης, το οποίο επιτρέπει τη μείωση του χρόνου διακοπής για τους Flume πράκτορες. Το Flume είναι σχεδιασμένο για να είναι πλήρως κατανεμημένο χωρίς κανένα κεντρικό σημείο συντονισμού. Κάθε πράκτορας τρέχει ανεξάρτητα από τους άλλους χωρίς κανένα εγγενή μοναδικό σημείο αποτυχίας. Επιπλέον, διαθέτει ενσωματωμένη υποστήριξη για failover και εξισορρόπηση του φόρτου και η πλήρως αποκεντρωμένη αρχιτεκτονική του παίζει καθοριστικό ρόλο στην επεκτασιμότητα του. Έτσι, δεδομένου ότι κάθε πράκτορας λειτουργεί ανεξάρτητα, το Flume μπορεί να επεκταθεί οριζόντια με ευκολία.

- **Το Apache Hue** – Το Apache Hue, είναι μία web διεπαφή που υποστηρίζει το Apache Hadoop και το οικοσύστημα του. Το Hue συγκεντρώνει, τα πιο συνήθη μέρη του Apache Hadoop σε μία μοναδική διεπαφή και έχει ως στόχο την καλύτερη εμπειρία του χρήστη. Ο βασικός στόχος του είναι οι χρήστες να «χρησιμοποιούν απλά» το Hadoop, χωρίς να ανησυχούν για την υποκείμενη πολυπλοκότητα του συστήματος ή να χρησιμοποιούν την γραμμή εντολών. Τα βασικά μέρη του Hue είναι τα εξής:
    - Ένα πρόγραμμα περιήγησης αρχείων για το HDFS.
    - Ένα πρόγραμμα περιήγησης των εργασιών για το MapReduce και το YARN.
    - Ένα πρόγραμμα περιήγησης για το Apache HBase.
    - Συντάκτες ερωτημάτων για το Apache Hive, το Apache Pig και το Cloudera Impala.
    - Συντάκτης ερωτημάτων για το Apache Sqoop2.
    - Συντάκτης ερωτημάτων και πίνακα ελέγχου για το Apache Oozie.
    - Ένα πρόγραμμα περιήγησης για το Apache Zookeeper.

  Επίσης, το Hadoop είναι μέρος μερικών σημαντικών διανομών πιο βασική των οποίων είναι η CDH (Cloudera Distribution Hadoop).

Τα παραπάνω project της Apache είναι ίσως τα πιο σημαντικά για την επέκταση, διαχείριση, χρήση κλπ. του Apache Hadoop όμως υπάρχουν και κάποια ακόμη και αυτά σχεδιασμένα για να λειτουργούν πάνω στο Hadoop, τα οποία αναφέρονται παρακάτω ενδεικτικά:

- **Το Apache Accumulo** – Το Apache Accumulo είναι ένα σύστημα αποθήκευσης και ανάκτησης δεδομένων, υψηλής απόδοσης με έλεγχο πρόσβασης σε επίπεδο κελιού. Είναι μια επεκτάσιμη υλοποίηση του Big Table της Google σχεδιασμένο να δουλεύει πάνω στο Apache Hadoop και το Apache Zookeeper.
- **Το Apache Storm** – Το Apache Storm είναι ένα κατανεμημένο σύστημα υπολογισμού πραγματικού χρόνου, για γρήγορη επεξεργασία μεγάλων ροών δεδομένων. Το Storm προσθέτει αξιόπιστες δυνατότητες πραγματικού χρόνου στο Apache Hadoop από την έκδοση 2.0 και μετά και το βοηθά να συλλάβει νέες επιχειρηματικές ευκαιρίες με πίνακες ελέγχου χαμηλής καθυστέρησης, ειδοποιήσεις ασφάλειας και λειτουργικές βελτιώσεις ενσωματωμένες με άλλες εφαρμογές που τρέχουν σε ένα Hadoop cluster.
- **Το Apache Falcon** – Το Apache Falcon, είναι ένα framework για την απλοποίηση της διαχείρισης των δεδομένων και της επεξεργασίας των μέσων πληροφορίας (pipelines) στο Apache Hadoop. Δίνει την δυνατότητα στους χρήστες να αυτοματοποιήσουν την μετακίνηση και την επεξεργασία των δεδομένων (dataset) για περιπτώσεις χρήσης όπως: λήψη δεδομένων, αγωγών δεδομένων, αποκατάσταση καταστροφών μετά από κάποιο σφάλμα και διατήρηση δεδομένων. Πλέον, οι χρήστες αντί να γράφουν κώδικα «με το χέρι» για την λογική επεξεργασίας πολύπλοκων dataset και ροών δεδομένων, μπορούν να βασιστούν στο Falcon για αυτές τις λειτουργίες, μεγιστοποιώντας έτσι την επαναχρησιμοποίηση και την συνέπεια μεταξύ των εφαρμογών του Hadoop.
- **Το Apache Knox Gateway** – Το Apache Knox Gateway (“Knox”) είναι ένα σύστημα που παρέχει ένα μοναδικό σημείο αυθεντικοποίησης και πρόσβασης για τις υπηρεσίες του Apache Hadoop που υπάρχουν σε ένα cluster. Ο στόχος του project είναι να απλοποιήσει την ασφάλεια του Hadoop, για χρήστες που έχουν πρόσβαση στα δεδομένα του cluster και

εκτελούν εργασίες πάνω σε αυτά, καθώς και για τους διαχειριστές που ελέγχουν την πρόσβαση και την διαχείριση του cluster. Το Knox τρέχει σαν ένας server (ή σαν ένα cluster από server) που εξυπηρετούν ένα ή περισσότερα Hadoop cluster.

- **Το Apache Bigtop** – Το Apache Bigtop, είναι ένα project για την ανάπτυξη πακέτων και δοκιμών για το οικοσύστημα του Hadoop. Ο βασικός στόχος του Bigtop είναι να χτίσει μια κοινότητα γύρω από το πακετάρισμα και την διαλειτουργικότητα των δοκιμών, των project που σχετίζονται με το Hadoop. Αυτό περιλαμβάνει δοκιμές σε διάφορα επίπεδα (πακετάρισμα, πλατφόρμα, κατά την εκτέλεση – runtime, αναβάθμιση, κλπ.), που έχουν αναπτυχθεί από μία κοινότητα με εστίαση στο σύστημα του Hadoop ως ένα σύνολο, και όχι ως μεμονωμένα project.

ΥΠΟΚΕΦΑΛΑΙΟ 1.11

Διανομές του Apache Hadoop

Υπάρχει ένα μεγάλος αριθμός εταιριών που προσφέρουν εμπορικές εφαρμογές και διανομές και/ή παρέχουν υποστήριξη για το Hadoop. Κάποιες από αυτές ακολουθούν παρακάτω.

- **CDH (Cloudera's Distribution Including Apache Hadoop) της Cloudera** – Η CDH είναι η πιο ολοκληρωμένη, δοκιμασμένη και ευρέως διαδεδομένη διανομή του Apache Hadoop. Η CDH είναι 100% ανοιχτού κώδικα και είναι η μόνη λύση Hadoop που προσφέρει μαζική επεξεργασία, διαδραστική SQL διαδραστική αναζήτηση, καθώς και επιχειρησιακού (enterprise) επιπέδου συνεχή διαθεσιμότητα. Πιο πολλές επιχειρήσεις έχουν κατεβάσει την CDH από ότι όλες τις άλλες διανομές μαζί. Αυτό το ισχυρό εργαλείο αυτοματοποίησης προσφέρει τον πιο γρήγορο και τον πιο εύκολο τρόπο για να στήσεις το cluster σου και να εξερευνήσεις τις πρώτες σου περιπτώσεις χρήσης. Η CDH προσφέρει τα βασικά στοιχεία το Hadoop – επεκτάσιμη αποθήκευση και κατανεμημένη επεξεργασία – αλλά και τις απαραίτητες επιχειρησιακές δυνατότητες, όπως ασφάλεια, υψηλή διαθεσιμότητα και

ενσωμάτωση με ένα ευρύ φάσμα λύσεων υλικού και λογισμικού. Όλη αυτή η εργασία ενσωμάτωσης γίνετε αυτόματα για τον χρήστη και όλη η λύση της Cloudera είναι διεξοδικά δοκιμασμένη και πλήρως τεκμηριωμένη, βοηθώντας στην λύση πραγματικών επιχειρησιακών προβλημάτων. Κάποια χαρακτηριστικά της CDH είναι:

- **Ευελιξία.** Δυνατότητα αποθήκευσης οποιουδήποτε είδους δεδομένων και επεξεργασία αυτών με μια σειρά από διαφορετικά framework υπολογισμού, συμπεριλαμβανομένου της μαζικής επεξεργασίας, της διαδραστικής SQL, της αναζήτησης ελεύθερου κειμένου, της μηχανικής μάθησης και του στατιστικού υπολογισμού.
- **Ενσωμάτωση.** Εύκολη και γρήγορη εγκατάσταση και ρύθμιση ενός cluster και γρήγορη χρήση μίας πλήρης, πακεταρισμένης πλατφόρμας Apache Hadoop.
- **Ασφάλεια.** Επεξεργασία και έλεγχος ευαίσθητων δεδομένων και διευκόλυνση της πολύ-μίσθωσης.
- **Επεκτασιμότητα.** Δυνατότητα ενεργοποίησης μίας πληθώρας εφαρμογών και εύκολη επέκταση τους μαζί με την επιχείρηση.
- **Υψηλή Διαθεσιμότητα.** Εκτέλεση εργασιών σημαντικών για την αποστολή της επιχείρησης με σιγουριά και ασφάλεια.
- **Ανοιχτός κώδικας.** Πλεονέκτημα για τους χρήστες της CDH χάρη στην ταχεία καινοτομία, χωρίς το κλείδωμα των ενημερώσεων από την εταιρία.
- **Συμβατότητα.** Δυνατότητα αξιοποίησης και επέκτασης των υπαρχουσών πόρων και επενδύσεων πληροφορικής (IT).

Η CDH διανομή της Cloudera, αποτελεί μέρος διαφορετικών εκδόσεων της λύσης της Cloudera, οι οποίες είναι: η Cloudera Standard, η Cloudera Enterprise, το Cloudera Manager το οποίο είναι μέρος και των δύο απλά χωρίς όλες τις δυνατότητες διαθέσιμες στην Cloudera Standard και ο Cloudera Navigator που είναι ένα αυτοματοποιημένο σύστημα διαχείρισης των δεδομένων.

- **Η MapR διανομή του Apache Hadoop της MapR Technologies Inc.** – Η διανομή MapR της εταιρίας ισχυρίζεται ότι προσφέρει πλήρη προστασία των δεδομένων, κανένα μοναδικό σημείο αστοχίας, βελτιωμένη απόδοση

και δραματικά πλεονεκτήματα στην ευκολία της χρήσης. Η MapR συνεισφέρει στο Apache Hadoop και στα project που σχετίζονται με αυτό όπως το Apache HBase, το Apache Pig, το Apache Hive και το Apache Zookeeper. Η MapR, επίσης έκανε μία συμφωνία με την EMC Corporation τον Μάιο του 2011 και υποστηρίζει μία διανομή του Apache Hadoop που είναι συγκεκριμένη για τα συστήματα της EMC. Επιπλέον, επιλέχτηκε από την Amazon για να προσφέρει μια αναβαθμισμένη και πιο αξιόπιστη έκδοση της υπηρεσίας Amazon Elastic Map Reduce (EMR) που είναι μέρος των υπηρεσιών Amazon Web Services (AWS) και προσφέρει τις M3 και M5 εκδόσεις της MapR ως premium επιλογές. Γενικά, έχει επιλεγεί από πολλές εταιρείες λόγο της αποδεδειγμένης υψηλής απόδοσης της σε συγκριτικές δοκιμές με άλλες κατανεμημένες/παράλληλες υλοποιήσεις επεξεργασίας δεδομένων αλλά και με το ίδιο το Apache Hadoop και με άλλες διανομές αυτού. Άλλη μία εταιρεία που την επέλεξε ως τεχνολογικό συνεταίρο είναι η Google, η οποία έχει υλοποίηση της MapR στην πλατφόρμα επεξεργασίας που προσφέρει (Google Compute Engine) και ήταν σε αυτή την πλατφόρμα που έσπασε το ρεκόρ ταξινόμησης δεδομένων σε λιγότερο από ένα λεπτό. Άλλοι συνέταιροι είναι η Cisco Systems, η Infomatica Corporation, η Impetus Technologies, η Talend και η Teradata Corporation. Η MapR προσφέρει τρεις εκδόσεις της διανομής τους γνωστές ως M3, M5 και M7. Η M3 είναι μία ελεύθερη έκδοση της M5 αλλά με μειωμένες δυνατότητες και λειτουργίες αλλά πάλι πολύ ισχυρές και ικανές για το ξεκίνημα και την ενασχόληση με το MapReduce και φυσικά με την ταχύτητα της MapR. Η M7 είναι σαν την M5 αλλά προσθέτει μία ειδικά κατασκευασμένη υλοποίηση του HBase, που υλοποιεί το HBase API απευθείας στο επίπεδο του συστήματος αρχείων. Επιπλέον, η διανομή της MapR έχει δική της υλοποίηση κατανεμημένου συστήματος αρχείων το maprfs και πολλές άλλες διαφορές στην αρχιτεκτονική με το βασικό Apache Hadoop και έτσι καταφέρνει την υψηλή διαθεσιμότητα, την μη ύπαρξη μοναδικού σημείου αστοχίας, την υψηλή απόδοση, την χαμηλή καθυστέρηση και την ευκολία προς το χρήστη σε όλες τις φάσεις του cluster όπως εγκατάσταση, ρύθμιση, συντήρηση, έλεγχος κ.α. Γενικότερα, το maprfs είναι ίσως η καλύτερη και η πιο πλήρης αντικατάσταση για το HDFS. Εξαλείφει το μειονέκτημα του HDFS ότι έχει ένα σημείο αστοχίας τον NameNode, μειώνει τον φόρτο στο

NameNode και έτσι επιτρέπει την γρήγορη εκκίνηση του NameNode και του cluster. Μπορεί να επεκταθεί σε ένα τρισεκατομμύρια αρχεία, σε ένα με δέκα Exabyte δεδομένων και σε πάνω από δέκα χιλιάδες κόμβους, σε αντίθεση με το HDFS που είναι σχεδιασμένο για περίπου εκατόν πενήντα εκατομμύρια αρχεία, δέκα με πενήντα Petabyte δεδομένων και 2 χιλιάδες κόμβους. Προσφέρει πλήρη τυχαία ανάγνωση/εγγραφή στο σύστημα αρχείων, υψηλή αξιοπιστία και γρήγορη επανεκκίνηση. Εκμεταλλεύεται καλύτερα νέες τεχνολογίες όπως είναι οι SSD και οι συνδέσεις δικτύου 10GE και φτάνει ταχύτητες πιο κοντά στις ταχύτητες του υλικού και παρόλα αυτά ακόμη και για cluster των Exabyte οι απαιτήσεις δεν είναι πάρα πολύ υψηλές. Παρέχει καλύτερη διαχείριση ενσωματωμένη στην διανομή, ειδοποιήσεις, παρακολούθηση του cluster, αυτοματοποίηση λειτουργιών κ.α.

- **Η HDP (Hortonworks Data Platform) της Hortonworks Inc.** – Η διανομή της Hortonworks, Apache Hadoop Data Platform είναι 100% ανοιχτού κώδικα, πλήρως δοκιμασμένη, πιστοποιημένη και έχει σχεδιαστεί με στόχο την ενσωμάτωση με τις υπάρχουσες data center τεχνολογίες και προσφέρει την επαναχρησιμοποίηση των υπαρχόντων πόρων και ικανοτήτων. Γι’ αυτό και προσφέρει συμβατότητα με πολλά συστήματα. Επίσης, η εταιρία είναι από τους μεγαλύτερους συνεισφέροντες στο Apache Hadoop και σε όλα τα project που σχετίζονται. Η έκδοση 2 της HDP (η πιο πρόσφατη) προσφέρει επεξεργασία και αποθήκευση με μεγάλη επεκτασιμότητα και πολλαπλές εργασίες (workloads), εφαρμογές και επεξεργαστικές μηχανές με μεγαλύτερη απόδοση με την βοήθεια της δεύτερης έκδοσης του MapReduce (MR2) το YARN (η μόνη διανομή προς το παρόν που συμπεριλαμβάνει όλες τις βελτιώσεις του YARN), του HDFS2 και της υψηλής διαθεσιμότητας και ασφάλειας που παρέχει. Επίσης, προσφέρει την διανομή της για Windows και είναι η μόνη που προσφέρει κάτι τέτοιο. Ενσωματώνεται εύκολα με την επιχειρησιακή λογική και το οικοσύστημα της Microsoft και είναι διαθέσιμη για Windows Server είτε στο Windows Azure Cloud. Επιπλέον, η λύση της Hortonworks ενσωματώνει σχεδόν όλα τα συναφή project του Hadoop, για να προσφέρει μια πλατφόρμα με όλα τα βασικά στοιχεία απαραίτητα για τις λειτουργικές υπηρεσίες των δεδομένων που απαιτούνται από μία επιχείρηση και προσφέρει πραγματική πολλαπλή

χρήση των δεδομένων, χάρη στην δυνατότητα που έχει να αλληλεπιδρά ταυτόχρονα με πολλούς τρόπους με τα δεδομένα.

- **Το HDInsight της Microsoft που προσφέρεται στο Windows Azure** – Το HDInsight είναι μία υπηρεσία του Windows Azure που προσφέρει το Apache Hadoop στο Cloud, παρέχοντας την δυνατότητα προβολής και διορατικότητας πάνω στα δεδομένα μέσω του Excel, εύκολη διαχείριση και μία διανομή του Hadoop έτοιμη για λειτουργία για κάθε επιχείρηση. Επίσης, προσφέρει πλούσιες δυνατότητες για την ανάπτυξη εφαρμογών για την χρήση της μέσω του γνωστού .NET framework, της JAVA και άλλες γλώσσες προγραμματισμού. Βοηθά τους χρήστες και τις επιχειρήσεις να ξεπεράσουν πολλά από τα προβλήματα που αντιμετωπίζουν όπως:
    - **Τα δεδομένα πραγματικού χρόνου** τα οποία δημιουργούνται κάθε μέρα σε τεράστιους όγκους και πάνω σε αυτά πρέπει να απαντηθούν κρίσιμα για μία επιχείρηση ερωτήματα
    - **Τα αδόμητα δεδομένα** τα οποία αποτελούν το 85% των νέων δεδομένων που δημιουργούνται καθημερινά, όπως το ελεύθερο κείμενο (Twitter κλπ.) και τα αρχεία κειμένου, οι εικόνες, τα βίντεο, τα δεδομένα από διαφόρων ειδών αισθητήρες, τα αρχείων καταγραφών ενός web server κ.α., τα οποία πρέπει να διαχειριστούν μαζί με τα δεδομένα από τις βάσεις δεδομένων και τις αποθήκες δεδομένων.
    - **Η απειρία σχετικά με το Hadoop.**
    - **Η δύσκολη ενσωμάτωση άλλων συστημάτων με εργαλεία επιχειρησιακής λογικής (Business Intelligence).**
    - **Η ασφάλεια και η διαχείριση των δεδομένων.** Η Microsoft δίνει την δυνατότητα στους διαχειριστές, για την αποτελεσματική διαχείριση και παρακολούθηση του συστήματος και των δεδομένων, ώστε να παρέχεται η απαραίτητη πρόσβαση προς τους χρήστες, ενώ παράλληλα εξασφαλίζεται η συμμόρφωση με τους κανόνες ασφαλείας.

Τα πλεονεκτήματα του HDInsight και της Microsoft είναι τα εξής:

- **Η ανοιχτή λογική και η ευελιξία του.** Δίνει την δυνατότητα της χρήσης του Apache Hadoop (100% συμβατό) στο Windows Azure

και μαζί με την Hortonworks Data Platform (HDP) μπορεί να λειτουργήσει και σε Windows Server και σε Linux.

- **Διορατικότητα στα δεδομένα μέσω του Excel.** Παρέχει την δυνατότητα ανάλυσης αδόμητων δεδομένων από το HDInsight στο Excel και τον συνδυασμό των δεδομένων από πολλές πηγές για δυναμική ανάλυση των δεδομένων χρησιμοποιώντας τα χαρακτηριστικά του Excel: Power Query, PowerPivot, Power View και Power Map.
- **Ταχύτητα.** Δραματική επιτάχυνση της ανάπτυξης ενός Hadoop cluster, το οποίο μπορεί να είναι έτοιμο για χρήση μέσα σε λεπτά. Μετά μπορούν να χρησιμοποιηθούν τα web εργαλεία και τα API για την αλληλεπίδραση και την παρακολούθηση του cluster.
- **Hadoop έτοιμο για την επιχείρηση.** Προσφέρει υψηλή ασφάλεια μέσω ενός Secure κόμβου που βοηθά για την ασφάλιση του cluster και εύκολη διαχείριση μέσω PowerShell scripts (εκτός από τις δυνατότητες που υπάρχουν από τα web εργαλεία).
- **Πλούσια εμπειρία ανάπτυξης.** Προσφέρει την δυνατότητα επιλογής της γλώσσας προγραμματισμού, όπως η Java, η χρήση του .NET framework κ.α. Επίσης, μπορούν να χρησιμοποιηθούν LINQ ερωτήματα προς το Hive μέσω της .NET.

- **Η Intel Distribution for Apache Hadoop της Intel.** Η διανομή της Intel είναι μια πλατφόρμα που παρέχει κατανεμημένη επεξεργασία και διαχείριση δεδομένων για τις επιχειρησιακές εφαρμογές που αναλύουν τεράστιες ποσότητες δεδομένων διαφορετικού τύπου. Είναι λογισμικό ανοιχτού κώδικα που περιλαμβάνει το Apache Hadoop και άλλα συστατικά στοιχεία του οικοσυστήματος του Hadoop, μαζί με βελτιώσεις και διορθώσεις από την Intel. Αποδεδειγμένη στην παραγωγή σε μερικές από τις πιο απαιτητικές επιχειρήσεις στον κόσμο, η διανομή της Intel υποστηρίζεται από μια παγκόσμια ομάδα μηχανικών που έχουν πρόσβαση και την εμπειρογνωμοσύνη για ολόκληρη τη στοίβα λογισμικού που αποτελείται και από τον υποκείμενο επεξεργαστή, την μονάδα αποθήκευσης και τις συσκευές δικτύωσης (αν είναι προϊόντα της Intel φυσικά). Τα βασικά χαρακτηριστικά της είναι:

- Έως και τριάντα φορές (30x) μεγαλύτερη απόδοση στο Hadoop, χάρη στις βελτιστοποιήσεις για τους Intel Xeon επεξεργαστές, τους Intel δίσκους SSD και τις συσκευές δικτύου 10 GbE της Intel. Έτσι, με την διανομή της Intel και αν χρησιμοποιηθούν τα τελευταία προϊόντα και τεχνολογίες που παρέχει η Intel, εξασφαλίζεται ότι το σύστημα θα λειτουργεί στην βέλτιστη απόδοση του σε ένα ευρύ φάσμα εφαρμογών.
- Ασφάλεια των δεδομένων χωρίς επιπτώσεις στην απόδοση, με βελτιώσεις Intel AES-NI (για τους επεξεργαστές που το υποστηρίζουν) στην κωδικοποίηση και την αποκωδικοποίηση στο HDFS. Επίσης παρέχει έλεγχο πρόσβασης βασισμένο σε ρόλους χρηστών με διαβάθμιση στο HBase σε επίπεδο κελιού.
- Εύκολη επεκτασιμότητα σε διαφορετικές τοποθεσίες και προσαρμοστικότητα στην αντιγραφή/αναπαραγωγή των δεδομένων στο HBase και το HDFS.
- Έως και τρεισήμισι (3.5x) μεγαλύτερη απόδοση στα ερωτήματα Hive.
- Υποστήριξη για στατιστική ανάλυση στα δεδομένα με τον R connector.
- Παρέχει την δυνατότητα αναλύσεων γραφημάτων με το Intel Graph Builder.
- Επιχειρησιακού επιπέδου υποστήριξη και υπηρεσίες από την Intel.

Η Intel, μαζί με την διανομή της παρέχει και τον Intel Manager ο οποίος παρέχει μία κονσόλα διαχείρισης για την διανομή για να απλοποιήσει την ανάπτυξη, την ρύθμιση, την παραμετροποίηση και την ασφάλεια του Hadoop cluster. Τα βασικά χαρακτηριστικά του είναι:

- Απλοποιεί την ανάπτυξη/εγκατάσταση του Hadoop.
- Αυτοματοποιημένη ρύθμιση μέσω του Intel Active Tuner.
- Διαχείριση του cluster με εύκολα στην χρήση εργαλεία (wizards).
- Περιεκτική παρακολούθηση και καταγραφή των γεγονότων του συστήματος.
- Προληπτικοί έλεγχοι της υγείας όλου του cluster.
- Παραμετροποίηση και βελτιστοποίηση της απόδοσης.
- Έλεγχος της αυθεντικοποίησης και της αδειοδότησης.

Γενικά, ο Intel Manager βρίσκει μόνος του τους server κόμβους, εγκαθιστά όλα τα απαραίτητα πακέτα, δίνει ρόλους στους κόμβους, ελέγχει την χρησιμοποίηση των πόρων και παρέχει μία web διεπαφή για την παρακολούθηση του cluster. Επίσης, αυτοματοποιεί την λογική της δοκιμής-σφάλματος ως προς της ρυθμίσεις με την εφαρμογή ενός αλγορίθμου τεχνητής νοημοσύνης που βρίσκει την βέλτιστη ρύθμιση για κάθε εφαρμογή και έτσι αξιοποιεί έως και το 95% της διαθέσιμης απόδοσης και μειώνει της καθυστέρηση κατά 30%.

Τέλος, η Intel αυτή την στιγμή προσφέρει και μία HPC διανομή του Apache Hadoop, την Intel HPC Distribution for Apache Hadoop η οποία είναι ακόμη σε BETA έκδοση και συνδυάζει την διανομή της Intel με την Intel Enterprise έκδοση του Lustre. Το Lustre είναι ένα παράλληλο σύστημα αρχείων το οποίο χρησιμοποιείται από πολλούς από τους πιο γρήγορους supercomputer.

- **Η Google** προσφέρει την δυνατότητα εκτέλεσης Hadoop 0.20 προγραμμάτων στην Google App Engine με την βοήθεια του AppEngine-MapReduce. Επίσης, αν και δεν προσφέρει δική της διανομή, δίνει την δυνατότητα ανάπτυξης του Hadoop στα εικονικά μηχανήματα που προσφέρει με το Google Compute Engine και τα πλεονεκτήματα είναι ότι προσφέρει: γρήγορους χρόνους εκκίνησης, υψηλή απόδοση εισαγωγής/εξαγωγής δεδομένων (I/O), εύκολη επεκτασιμότητα.
- **Η Splunk** προσφέρει ενσωμάτωση με το Hadoop με ένα προϊόν που διαθέτει, το Hadoop Connect, το οποίο είναι πιστοποιημένο από την MapR, την Cloudera, την Hortonworks και το Apache Hadoop. Αυτή η ενσωμάτωση επιτρέπει τους χρήστες να αναζητήσουν τα δεδομένα του Hadoop από το Splunk και να εισάγουν δεδομένα από το Hadoop στο Splunk και αντίστροφα. Επίσης, είναι από τις καλύτερες υπηρεσίες για ανάλυση και οπτικοποίηση των δεδομένων με γραφήματα, πίνακες κλπ.
- **Η Oracle** ανακοίνωσε την διανομή Big Data Appliance, η οποία ενσωματώνει την διανομή της Cloudera (CDH), το Linux της Oracle, την γλώσσα προγραμματισμού R και μία NoSQL βάση δεδομένων με το υλικό Exadata.

- **Η IBM** προσφέρει το WebSphere eXtreme Scale (προηγουμένως ObjectGrid), το οποίο περιλαμβάνει δύο στυλ του Hadoop μοτίβου MapReduce στα DataGrid API της (γνωστά και ως πράκτορες – agents). Μαζί με την επεκτάσιμη κατανεμημένη δυνατότητα caching των δεδομένων, προσφέρει ταυτόχρονα την δυνατότητα του MapReduce για παραλληλοποίηση των εργασιών και την δυνατότητα να αποθηκεύει πολλά δεδομένα στην μνήμη για γρήγορη πρόσβαση για χρήση από την εργασία. Επίσης, έχει υψηλή διαθεσιμότητα. Η IBM, προσφέρει και το InfoSphere BigInsights το οποίο και στην βασική και στην επιχειρηματική έκδοση βασίζεται στο Hadoop.
- **Η Pentaho** η οποία προσφέρει λογισμικό επιχειρησιακής λογικής, ανακοίνωσε την υποστήριξη για το Apache Hadoop, επιτρέποντας τις εταιρείες που χρησιμοποιούν το λογισμικό της να έχουν πρόσβαση στις δυνατότητες ενσωμάτωσης και ανάλυσης δεδομένων του λογισμικού της απευθείας από το Apache Hadoop ή διανομές βασισμένες σε αυτό. Τον Ιανουάριο του 2012, έκανε διαθέσιμες δημόσια τις δυνατότητες του λογισμικού της και μετακίνησε ολόκληρη την μηχανή ενσωμάτωσης δεδομένων (Pentaho Kettle) από την LGPL License στην Apache License.
- **Η Dell** πρόσθεσε την εφαρμογή της Pentaho, Pentaho Business Analytics στην Apache Hadoop λύση της για ανάλυση μεγάλου όγκου δεδομένων. Αυτή αποτελείται από Dell servers, Dell συσκευές δικτύου, το ανοιχτού κώδικα λογισμικό της Dell το Crowbar cloud deployment framework και την διανομή της Cloudera του Hadoop.
- **Η EMC** κυκλοφόρησε το EMC Greenplum Community Edition και το EMC Greenplum HD Enterprise Edition τον Μάιο του 2011. Η community έκδοση, με προαιρετική επί πληρωμής τεχνική υποστήριξη, αποτελείται από το Hadoop, το HDFS, το HBase, το Hive και το Zookeeper. Η enterprise έκδοση είναι βασισμένη στην MapR και προσφέρει εμπορικά χαρακτηριστικά όπως στιγμιότυπα και αναπαραγωγή των δεδομένων ευρείας περιοχής.
- **Η EMC Isilon** ανακοίνωσε υποστήριξη για το HDFS στο OneFS clustered σύστημα αρχείων.

- **Η BMC Software** παρέχει το BMC Control-M για το Hadoop, το οποίο προσθέτει δυνατότητες παρακολούθησης και διαχείρισης Hadoop ροών εργασίας με αυτόματες ειδοποιήσεις και προβλεπτικές αναλύσεις.
- **Ο JobServer της Grand Logic** επιτρέπει στους προγραμματιστές και τους διαχειριστές να αναπτύσσουν, να διαχειρίζονται και να παρακολουθούν την Hadoop υποδομή τους, με υποστήριξη για επεξεργασία των Hadoop εργασιών και διαχείριση των HDFS αρχείων.
- **Η MetaScale** προσφέρει ουδέτερη και ανεξάρτητη από πλατφόρμες υπηρεσία για υποστήριξη και υλοποίηση του Hadoop.
- **Το OceanSync Hadoop Management and Visualization Software της Dovestech**, το οποίο επιτρέπει στου χρήστες να ελέγχουν, να παρακολουθούν και να οπτικοποιούν όλα τα μέρη ενός Hadoop cluster συμπεριλαμβανομένου διαχείριση των ροών εργασιών δεδομένων και οπτικοποίηση των δεδομένων εξαγωγής από την επεξεργασία. Το πακέτο προσφέρεται σε τρεις εκδόσεις, την OceanSync Free Desktop Edition (ελεύθερη έκδοση), την OceanSync Enterprise Edition with Visualization (επιχειρησιακή έκδοση με οπτικοποίηση) και την OceanSync Mobile for iPhone/Android συσκευές.
- **Η Pivotal** προσφέρει το Pivotal HD, μία διανομή του Hadoop που περιλαμβάνει το HAWQ, το οποίο παρέχει 100% SQL ANSI συμβατότητα.
- **Η Platform Computing** ανακοίνωσε υποστήριξη για το Hadoop MapReduce API στο Symphony λογισμικό της.
- **Η Silicon Graphics International** προσφέρει λύσεις βελτιστοποιημένες για το Hadoop βασισμένες στις σειρές server, SGI Rackable και CloudRack με υπηρεσίες υλοποίησης.
- **Η sqrrl** προσφέρει το sqrrl enterprise, το οποίο επεκτείνει το Hadoop με το Apache Accumulo και συνδυάζει τα χαρακτηριστικά πολλών datastore.
- **Η Syncsort** προσφέρει μία ETL (extract-transform-load – εξαγωγή-μετασχηματισμός-φόρτωση) λύση, η οποία επεκτείνει τις δυνατότητες του Hadoop, μετατρέποντας το σε ένα υψηλά επεκτάσιμο, προσιτό (οικονομικά) και εύκολο στην χρήση περιβάλλον ενσωμάτωσης δεδομένων.

- **Η Talend** προσφέρει το Talend Open Studio for Big Data, το οποίο παρέχετε υπό την Apache Software License και αυτό περιλαμβάνει την εγγενή υποστήριξη για το Apache Hadoop.
- **Η Teradata** προσφέρει το Teradata Appliance for Hadoop, καθώς και υποστήριξη για εγκαταστάσεις του Hadoop σε επίπεδο λογισμικό.
- **Η TIBCO** υποστηρίζει το Hadoop με το Businessworks Hadoop plugin που προσφέρει και με τον TIBCO Spotfire Client.
- **Η WANdisco** προσφέρει το WDD, το οποίο είναι η διανομή της WANdisco που περιλαμβάνει και το Apache Hadoop. Επίσης, συνεισφέρουν στο Hadoop και σε όλα ανοιχτού κώδικα project.
- **Η Zettaset** προσφέρει μία νέα έκδοση του Big Data Mgt Platform βασισμένη στο Hadoop, η οποία προσφέρει υψηλή διαθεσιμότητα μέσω του NameNode Failover, μία εξορθολογισμένη διεπαφή, ένα πρωτόκολλο για την χρόνο στο δίκτυο και ενσωματωμένη ασφάλεια με αυθεντικοποίηση Kerberos.

ΥΠΟΚΕΦΑΛΑΙΟ 1.12

Εξέλιξη/Διαφορετικότητα του Apache Hadoop συγκριτικά με τα υπάρχουσα συστήματα

Το Hadoop διαφέρει σε αρκετούς τομείς σε σχέση με τους μέχρι τώρα τρόπους ανάπτυξης εφαρμογών με μεγάλη απαίτηση σε δεδομένα. Παρακάτω είναι οι λόγοι για τους οποίες διαφέρει και είναι πολύ πιο εύκολο στην χρήση:

- **Το Hadoop μπορεί να χειριστεί τα δεδομένα με ένα πολύ ρευστό τρόπο.** Το Hadoop είναι κάτι περισσότερο από απλώς μία φτηνότερη λύση από μία βάση δεδομένων και ένα εργαλείο ανάλυσης. Σε αντίθεση με τις βάσεις δεδομένων, στο Hadoop δεν είναι απαραίτητο τα δεδομένα να είναι δομημένα. Τα δεδομένα μπορεί να είναι αδόμητα και χωρίς να ακολουθούν κάποιο σχήμα. Οι χρήστες μπορούν να τοποθετούν τα δεδομένα τους στο Hadoop χωρίς να χρειάζεται να τα διαμορφώσου . Αντιθέτως, οι σχεσιακές

βάσεις δεδομένων απαιτούν τα δεδομένα να είναι δομημένα και να υπάρχει ορισμένο σχήμα πριν την αποθήκευση τους.

- **Το Hadoop έχει ένα απλουστευμένο μοντέλο προγραμματισμού.** Το απλοποιημένο μοντέλο προγραμματισμού του Hadoop επιτρέπει στους χρήστες να γράψουν γρήγορα εφαρμογές και να δοκιμάσουν τα κατανεμημένα συστήματα. Η εκτέλεση υπολογισμών σε μεγάλους όγκους δεδομένων έχει εφαρμοστεί και στο παρελθόν, συνήθως σε ένα κατανεμημένο περιβάλλον, αλλά η ανάπτυξη κατανεμημένων συστημάτων είναι εξαιρετικά δύσκολη. Θυσιάζοντας κάποια ευελιξία προγραμματισμού, το Hadoop καθιστά πολύ πιο εύκολο το να αναπτύξει κάποιος κατανεμημένες εφαρμογές.
- **Το Hadoop είναι εύκολο στην διαχείριση.** Εναλλακτικά υπολογιστικά συστήματα υψηλής απόδοσης (HPC), επιτρέπουν να εκτελούνται προγράμματα σε μεγάλες συλλογές υπολογιστών, αλλά απαιτούν συνήθως άκαμπτη ρύθμιση του προγράμματος και γενικά απαιτούν τα δεδομένα να αποθηκεύονται σε ένα ξεχωριστό σύστημα αποθήκευσης (SAN – Storage Area Network). Οι χρονοδρομολογητές σε cluster HPC απαιτούν προσεκτική διαχείριση και δεδομένου ότι η εκτέλεση κάποιου προγράμματος είναι ευαίσθητη στην αποτυχία ενός κόμβου, η διαχείριση ενός cluster Hadoop είναι πολύ πιο εύκολη. Το Hadoop χειρίζεται αόρατα (χωρίς παρέμβαση) θέματα ελέγχου των εργασιών, όπως η αποτυχία κάποιου κόμβου. Εάν ένας κόμβος αποτύχει, το Hadoop εγγυάται ότι οι υπολογισμοί θα εκτελεστούν σε άλλους κόμβους και ότι τα δεδομένα που είναι αποθηκευμένα σε αυτόν τον κόμβο θα ανακτηθούν από άλλους κόμβους για την συνέχιση της εργασίας.
- **Το Hadoop είναι ευέλικτο.** Οι σχεσιακές βάσεις δεδομένων είναι καλές στο να αποθηκεύουν και να επεξεργάζονται σύνολα δεδομένων με προκαθορισμένα και σταθερά μοντέλα δεδομένων. Για αδόμητα δεδομένα, οι σχεσιακές βάσεις δεδομένων δεν έχουν την ευελιξία και την επεκτασιμότητα που απαιτείται. Το Apache Hadoop καθιστά δυνατή την φτηνή επεξεργασία και την ανάλυση τεράστιων όγκων, δομημένων και αδόμητων δεδομένων μαζί, και την επεξεργασία των δεδομένων χωρίς τον ορισμό όλης της δομής τους από την αρχή.

ΥΠΟΚΕΦΑΛΑΙΟ 1.13

Λόγοι χρήσης του Apache Hadoop

Υπάρχουν αρκετοί λόγοι για να επιλέξει κάποιος το Apache Hadoop για την επεξεργασία και την ανάλυση των δεδομένων. Παρακάτω είναι μερικοί από αυτούς:

- **Είναι οικονομικά αποδοτικό.** Το Apache Hadoop ελέγχει και μειώνει το κόστος αποθηκεύοντας τα δεδομένα πιο οικονομικά ανά Terabyte σε σχέση με άλλες πλατφόρμες. Αντί για χιλιάδες έως δεκάδες χιλιάδες δολάρια ανά Terabyte, το Hadoop παρέχει υπολογιστικές και αποθηκευτικές ικανότητες για εκατοντάδες δολάρια ανά Terabyte.
- **Είναι ανεκτικό σε σφάλματα.** Η ανοχή σε σφάλματα είναι ένα από τα πιο σημαντικά πλεονεκτήματα της χρήσης του Hadoop. Ακόμη και αν μεμονωμένοι κόμβοι έχουν υψηλά ποσοστά αποτυχίας κατά την εκτέλεση εργασιών σε ένα μεγάλο cluster, τα δεδομένα αναπαράγονται σε όλο το cluster, έτσι ώστε να μπορούν εύκολα να ανακτηθούν σε περίπτωση σφάλματος του δίσκου, του κόμβου ή του rack.
- **Είναι ευέλικτο.** Ο ευέλικτος τρόπος που τα δεδομένα είναι αποθηκευμένα στο Apache Hadoop είναι ένα από τα μεγαλύτερα πλεονεκτήματα του - επιτρέποντας στις επιχειρήσεις να δημιουργήσουν αξία και κέρδος από τα δεδομένα που μέχρι τώρα θεωρούνταν πάρα πολύ ακριβά για να αποθηκεύονται και να επεξεργάζονται σε παραδοσιακές βάσεις δεδομένων. Με το Hadoop, μπορούν να χρησιμοποιηθούν όλα τα είδη των δεδομένων, τόσο δομημένων όσο και αδόμητων, για την εξαγωγή πιο σημαντικών επιχειρηματικές ιδεών όσο πιο πολλά είναι τα δεδομένα.
- **Είναι επεκτάσιμο.** Το Hadoop είναι μια εξαιρετικά επεκτάσιμη πλατφόρμα αποθήκευσης, επειδή μπορεί να αποθηκεύσει και να διανείμει πολύ μεγάλα σύνολα δεδομένων σε cluster αποτελούμενα από εκατοντάδες φθηνούς server που λειτουργούν παράλληλα. Το πρόβλημα με τα παραδοσιακά

συστήματα διαχείρισης σχεσιακών βάσεων δεδομένων (RDBMS), είναι ότι δεν μπορούν να κλιμακωθούν για την επεξεργασία τεράστιων όγκων δεδομένων.

ΕΠΙΛΟΓΟΣ

Όπως είδαμε σε αυτό το κεφάλαιο το Apache Hadoop, από τα πρώτα του βήματα μέχρι σήμερα, έχει εξελιχθεί σε όλους τους τομείς και έχει γίνει ακόμη πιο προσιτό και εύκολο στην χρήση. Αυτό έγινε χάρη στην ευρεία υιοθέτηση από πολλές εταιρείες αλλά και από την ανάγκη εξαγωγής χρησιμοποιήσιμης πληροφορίας από τους τεράστιους όγκους δεδομένων που παράγονται κάθε μέρα και μέχρι τώρα έμεναν ανεκμετάλλευτοι. Έτσι, υπάρχουν πλέον πολλές διανομές και εφαρμογές του Hadoop, που το βελτιώνουν σε όλους τους τομείς αλλά και το κάνουν πιο εύκολο στην προσαρμογή σε οποιοδήποτε τύπο δεδομένων. Φυσικά, τίποτα δεν θα ήταν εφικτό χωρίς την δύναμη του Apache Hadoop με την αρχιτεκτονική του, το HDFS, την προσαρμοστικότητα του, το μοντέλο MapReduce που τα ξεκίνησε όλα και όλα τα συναφή project που το βελτιώνουν. Ο καλύτερος τρόπος για την εκμετάλλευση αυτής της δύναμης είναι η χρησιμοποίηση της στην μηχανική μάθηση, ένας τομέας με όλο και μεγαλύτερη ανάγκη για εφαρμογή, από αναγνώριση εικόνων, προτύπων, ταξινόμηση και πολλά άλλα με εφαρμογές στην ασφάλεια, στις web αγορές, στην πρόβλεψη κ.α.

ΚΕΦΑΛΑΙΟ 2

Μηχανική Μάθηση - Ταξινόμηση (Classification) Naive Bayes

ΕΙΣΑΓΩΓΗ

Η μάθηση σε ένα γνωστικό σύστημα, όπως γίνεται αντιληπτή στην καθημερινή ζωή μπορεί να συνδεθεί με δύο βασικές ιδιότητες:

- Την ικανότητα στην πρόσκτηση γνώσης κατά την αλληλεπίδραση του με το περιβάλλον.
- Την ικανότητα να βελτιώνει με την επανάληψη τον τρόπο μίας ενέργειας.

Έχουν προταθεί διάφοροι ορισμοί για την μάθηση:

- Simon(’83), «η μάθηση σηματοδοτεί προσαρμοστικές αλλαγές σε ένα σύστημα με την έννοια ότι αυτές του επιτρέπουν να κάνει την ίδια εργασία, ή εργασίες της ίδιας κατηγορίας, πιο αποδοτικά και αποτελεσματικά την επόμενη φορά».
- Minsky(’85), «…είναι να κάνουμε χρήσιμες αλλαγές στο μυαλό μας».
- Michalski(’86), «…είναι η δημιουργία ή η αλλαγή της αναπαράστασης των εμπειριών».

Για τα συστήματα που ανήκουν στην συμβολική Τεχνητή Νοημοσύνη (ΤΝ), η μάθηση προσδιορίζεται ως πρόσκτηση επιπλέον γνώσης, που επιφέρει μεταβολές στην υπάρχουσα γνώση. Τα τεχνητά νευρωνικά δίκτυα (που ανήκουν στην μη συμβολική ΤΝ) έχουν δυνατότητα μάθησης μετασχηματίζοντας την εσωτερική τους δομή, παρά καταχωρώντας κατάλληλα αναπαριστάμενη γνώση.

Ο άνθρωπος προσπαθεί να κατανοήσει το περιβάλλον του παρατηρώντας το και δημιουργώντας μια απλοποιημένη (αφαιρετική) εκδοχή του που ονομάζεται μοντέλο (model). Η δημιουργία ενός τέτοιου μοντέλου, ονομάζεται επαγωγική μάθηση (inductive learning), ενώ η διαδικασία γενικότερα ονομάζεται επαγωγή (induction). Επιπλέον ο άνθρωπος έχει τη δυνατότητα να οργανώνει και να συσχετίζει τις εμπειρίες και τις παραστάσεις του δημιουργώντας νέες δομές που ονομάζονται πρότυπα (patterns). Η δημιουργία μοντέλων ή προτύπων από ένα σύνολο δεδομένων, από ένα υπολογιστικό σύστημα, ονομάζεται μηχανική μάθηση (machine learning). Διάφοροι ορισμοί:

- Carbonell (1987), «... η μελέτη υπολογιστικών μεθόδων για την απόκτηση νέας γνώσης, νέων δεξιοτήτων και νέων τρόπων οργάνωσης της υπάρχουσας γνώσης».
- Mitchell (1997), «Ένα πρόγραμμα υπολογιστή θεωρείται ότι μαθαίνει από την εμπειρία Ε σε σχέση με μια κατηγορία εργασιών Τ και μια μετρική

απόδοσης P, αν η απόδοση του σε εργασίες της T, όπως μετριούνται από την P, βελτιώνονται με την εμπειρία E».

- Witten & Frank (2000), «Κάτι μαθαίνει όταν αλλάζει τη συμπεριφορά του κατά τέτοιο τρόπο ώστε να αποδίδει καλύτερα στο μέλλον».

ΥΠΟΚΕΦΑΛΑΙΟ 2.1

Είδη μηχανικής μάθησης

Η μηχανική μάθηση αναφέρεται στον σχεδιασμό αλγορίθμων για τη δημιουργία ενός αυτόματου συστήματος που θα αποκτά γνώση βασιζόμενο σε εμπειρικά δεδομένα. Η έννοια της επιβλεπόμενης μάθησης έχει ιδιαίτερη σημασία. Δεδομένης μίας υπάρχουσας συλλογής αντικειμένων για τα οποία είναι γνωστή κλάση (ή κατηγορία) στόχος είναι να βρεθεί μία συνάρτηση μεταβλητών που θα περιγράφει το μοντέλο της κλάσης. Η επιτυχία πρόβλεψης αξιολογείται με ένα νέο σύνολο δεδομένων.

Έχουν αναπτυχθεί πολλές τεχνικές μηχανικής μάθησης που χρησιμοποιούνται ανάλογα με τη φύση του προβλήματος και εμπίπτουν σε ένα από τα παρακάτω δυο είδη:

- μάθηση με επίβλεψη (supervised learning) ή μάθηση με παραδείγματα (learning from examples),
- μάθηση χωρίς επίβλεψη (unsupervised learning) ή μάθηση από παρατήρηση (learning from observation).

Στη μάθηση με επίβλεψη το σύστημα καλείται να "μάθει" μια έννοια ή συνάρτηση από ένα σύνολο δεδομένων, η οποία αποτελεί περιγραφή ενός μοντέλου. Στη μάθηση χωρίς επίβλεψη το σύστημα πρέπει μόνο του να ανακαλύψει συσχετίσεις ή ομάδες σε ένα σύνολο δεδομένων, δημιουργώντας πρότυπα, χωρίς να είναι γνωστό αν υπάρχουν, πόσα και ποια είναι.

Στη μάθηση με επίβλεψη το σύστημα πρέπει να "μάθει" επαγωγικά μια συνάρτηση που ονομάζεται συνάρτηση στόχος (target function) και αποτελεί έκφραση του μοντέλου που περιγράφει τα δεδομένα. Η συνάρτηση στόχος χρησιμοποιείται για

την πρόβλεψη της τιμής μιας μεταβλητής, που ονομάζεται εξαρτημένη μεταβλητή ή μεταβλητή εξόδου, βάσει των τιμών ενός συνόλου μεταβλητών, που ονομάζονται ανεξάρτητες μεταβλητές ή μεταβλητές εισόδου ή χαρακτηριστικά. Η επαγωγική μάθηση στηρίζεται στην "υπόθεση επαγωγικής μάθησης" (inductive learning hypothesis), σύμφωνα με την οποία: κάθε υπόθεση h που προσεγγίζει καλά τη συνάρτηση στόχο για ένα αρκετά μεγάλο σύνολο παραδειγμάτων, θα προσεγγίζει το ίδιο καλά τη συνάρτηση στόχο και για περιπτώσεις που δεν έχει εξετάσει. Στην μάθηση με επίβλεψη διακρίνονται δυο είδη προβλημάτων (learning tasks), τα προβλήματα ταξινόμησης και τα προβλήματα παρεμβολής.

- Η ταξινόμηση (classification) αφορά στη δημιουργία μοντέλων πρόβλεψης διακριτών τάξεων (κλάσεων/κατηγοριών) (π.χ. ομάδα αίματος).
- Η παρεμβολή (regression) αφορά στη δημιουργία μοντέλων πρόβλεψης αριθμητικών τιμών (π.χ. πρόβλεψη ισοτιμίας νομισμάτων ή τιμής μετοχής).

ΥΠΟΚΕΦΑΛΑΙΟ 2.2

Ταξινόμηση/Κατηγοριοποίηση (Classification)

Η κατηγοριοποίηση (classification) είναι μία τεχνική της εξόρυξης δεδομένων, κατά την οποία ένα στοιχείο ανατίθεται σε ένα προκαθορισμένο σύνολο κατηγοριών. Ο όρος κατηγοριοποίηση συναντάται στην βιβλιογραφία και ως ταξινόμηση. Γενικότερα, ο στόχος της διαδικασίας αυτής είναι η ανάπτυξη ενός μοντέλου, το οποίο αργότερα θα μπορεί να χρησιμοποιηθεί για την κατηγοριοποίηση μελλοντικών δεδομένων. Τέτοια παραδείγματα είναι ο διαχωρισμός των emails με βάση την επικεφαλίδα τους ή το περιεχόμενό τους, η πρόβλεψη καρκινικών κυττάρων χαρακτηρίζοντας τα ως καλοήθη ή κακοήθη, η κατηγοριοποίηση πελατών μιας τράπεζας ανάλογα με την πιστωτική τους ικανότητα κ.α.

Η κατηγοριοποίηση μπορεί να περιγραφεί ως μία διαδικασία δύο βημάτων:

- **Εκμάθηση (Learning):** Στο πρώτο βήμα της διαδικασίας δημιουργείται/προσδιορίζεται το μοντέλο με βάση ένα σύνολο προκατηγοριοποιημένων παραδειγμάτων, που ονομάζεται δεδομένα

εκπαίδευσης (training data). Τα δεδομένα εκπαίδευσης αναλύονται από ένα αλγόριθμο κατηγοριοποίησης, προκειμένου να σχηματιστεί το μοντέλο. Λόγω του ότι τα δεδομένα εκπαίδευσης ανήκουν σε μία προκαθορισμένη κατηγορία, η οποία είναι γνωστή, η κατηγοριοποίηση αποτελεί μέθοδος μάθησης με επίβλεψη (supervised learning). Το μοντέλο, που λέγεται και αλλιώς κατηγοριοποιητής (classifier), αναπαρίσταται με τη μορφή κανόνων κατηγοριοποίησης (classification rules), δέντρων απόφασης (decision trees) ή μαθηματικών τύπων.

- **Κατηγοριοποίηση (Classification):** Μετά την δημιουργία του μοντέλου, το επόμενο βήμα είναι η αξιολόγησή του. Για να επιτευχθεί αυτό, χρησιμοποιούμε τα δοκιμαστικά δεδομένα (test data) για να υπολογίσουν την ακρίβεια του μοντέλου. Το μοντέλο κατηγοριοποιεί τα δοκιμαστικά δεδομένα. Έπειτα, η κατηγορία που σχηματίστηκε με βάση τα δοκιμαστικά δεδομένα συγκρίνεται με την πρόβλεψη που έγινε για τα δεδομένα εκπαίδευσης, τα οποία είναι ανεξάρτητα από αυτά της δοκιμής. Η ακρίβεια του μοντέλου υπολογίζεται από το ποσοστό των δειγμάτων δοκιμής που κατηγοριοποιήθηκαν σωστά σε σχέση με το υπό εκπαίδευση μοντέλο.

Στην περίπτωση που το μοντέλο κριθεί αποδεκτό, τότε μπορεί να χρησιμοποιηθεί για την κατηγοριοποίηση μελλοντικών δειγμάτων δεδομένων, των οποίων η κατηγοριοποίηση είναι άγνωστη.

ΥΠΟΚΕΦΑΛΑΙΟ 2.2.1
Κατηγορίες μεθόδων Ταξινόμησης/Κατηγοριοποίησης (Classification)

Bayesian

Η Bayesian κατηγοριοποίηση αποτελεί μία κατηγορία μεθόδων της κατηγοριοποίησης και βασίζεται στη στατιστική θεωρία κατηγοριοποίησης του Bayes. Αυτό σημαίνει ότι πραγματοποιείται μια πιθανοτική πρόβλεψη, δηλαδή προβλέπει την πιθανότητα ένα δείγμα X να ανήκει σε κάποια κατηγορία. Ο απλούστερος Bayesian κατηγοριοποιητής είναι ο Naïve Bayesian. Αυτός υποθέτει

ότι η επίδραση ενός γνωρίσματος σε μία κατηγορία είναι ανεξάρτητη από τις τιμές των υπόλοιπων γνωρισμάτων. Ο λόγος που γίνεται αυτό είναι για να αποφεύγονται οι πολύπλοκοι υπολογισμοί κατά τη συνθήκη ανεξαρτησίας της κατηγορίας. Στη μάθηση κατά Bayes (Bayesian learning) κάθε παράδειγμα εκπαίδευσης μπορεί σταδιακά να μειώσει ή να αυξήσει την πιθανότητα να είναι σωστή μια υπόθεση. Μία πρακτική δυσκολία στην εφαρμογή της μάθησης κατά Bayes είναι η απαίτηση για τη γνώση πολλών τιμών πιθανοτήτων. Όταν αυτές οι τιμές δεν είναι δυνατό να υπολογιστούν επακριβώς, υπολογίζονται κατ' εκτίμηση από παλαιότερες υποθέσεις, εμπειρική γνώση, κτλ. Η παραπάνω δυσκολία εφαρμογής έχει δώσει μεγάλη πρακτική αξία σε μια απλουστευμένη εκδοχή της μάθησης κατά Bayes, τον απλό ταξινομητή Bayes, στον οποίο γίνεται η παραδοχή ότι τα χαρακτηριστικά είναι ανεξάρτητα μεταξύ τους.

Περιγραφή Naïve Bayesian

Ο απλός ταξινομητής Bayes (simple/naive Bayes classifier) είναι μια πρακτική μέθοδος μάθησης που στηρίζεται σε στατιστικά στοιχεία (κατανομές πιθανότητας).

Υποθέτουμε ότι έχουμε ένα σύνολο δεδομένων S και έστω ότι κάθε δείγμα δεδομένων X=(x1,x2,...,xn) με m κατηγορίες C1,C2,..,Cm. Δεδομένου ενός αγνώστου δείγματος δεδομένων X, ο κατηγοριοποιητής θα προβλέψει ότι το X ανήκει στην κατηγορία C που έχει την μέγιστη εκ των υστέρων (posterior) πιθανότητα με βάση το X. Αυτό σημαίνει ότι το X κατηγοριοποιείται στην Ci αν και μόνο αν:

$$p(Ci \mid x) > p(Cj \mid x) \text{ για κάθε } 1 \le j \le m \text{ και } j \ne 1 \qquad (2.1)$$

Ο στόχος, λοιπόν, είναι να βρούμε την μέγιστη posterior πιθανότητα, δηλαδή το μέγιστο p(Ci|X) για κάθε κλάση, με αποτέλεσμα ο Naïve Bayesian κατηγοριοποιητής να έχει υψηλή απόδοση. Η απόδοση του συγκρίνεται με αυτή των δέντρων απόφασης και κάποιους κατηγοριοποιητές που στηρίζονται σε νευρωνικά δίκτυα σε ορισμένες εφαρμογές.

Δέντρα απόφασης

Τα δέντρα απόφασης χρησιμοποιούνται ευρέως για την κατηγοριοποίηση και πρόβλεψη δεδομένων. Ένα δέντρο απόφασης κατασκευάζεται σύμφωνα με ένα σύνολο εκπαίδευσης προ-κατηγοριοποιημένων δεδομένων. Κάθε εσωτερικός κόμβος προσδιορίζει τον έλεγχο των γνωρισμάτων και κάθε κλαδί που συνδέει τους εσωτερικούς με τους απόγονους αντιστοιχεί σε μία πιθανή τιμή για το γνώρισμα, όπως εμφανίζεται και στην διπλανή εικόνα.

Νευρωνικά δίκτυα

Μια άλλη τεχνική κατηγοριοποίησης που χρησιμοποιείται σε εφαρμογές εξόρυξης γνώσης για πρόβλεψη και κατηγοριοποίηση στηρίζεται στα νευρωνικά δίκτυα. Τα βήματα αυτής της διαδικασίας χονδρικά είναι:

- Η αναγνώριση των χαρακτηριστικών εισόδου και εξόδου
- Δημιουργία ενός δικτύου με την κατάλληλη τοπολογία
- Επιλογή του συνόλου εκπαίδευσης (train data)
- Εφαρμογή του δικτύου με ένα αντιπροσωπευτικό σύνολο δεδομένων, ώστε να μεγιστοποιείται η δυνατότητα του δικτύου να αναγνωρίζει τα πρότυπα
- Επαλήθευση-αξιολόγηση του δικτύου με την χρήση ενός σύνολο ελέγχου (test data).

Παραγωγή κανόνων κατηγοριοποίησης

Η γνώση που αποκτούμε κατά την διαδικασία της κατηγοριοποίησης μπορεί να αναπαρασταθεί και με τη χρήση κανόνων. Οι κανόνες κατηγοριοποίησης, σε σχέση με τα δέντρα απόφασης, γίνονται ευκολότερα κατανοητοί όταν το δέντρο που παράχθηκε είναι μεγάλο. Έτσι μπορούμε να μετατρέψουμε ένα δέντρο απόφασης σε ένα σύνολο κανόνων κατηγοριοποίησης. Αυτό μπορεί να επιτευχθεί εάν θεωρήσουμε ότι κάθε κανόνας αντιστοιχεί σε ένα μονοπάτι του δέντρου από τη ρίζα μέχρι ένα κόμβο φύλλο. Άρα κάθε φύλλο παράγει ένα κανόνα. Οι συνθήκες που θα μας οδηγήσουν στο φύλλο (υπόθεση) αποτελούν το αριστερό μέρος του κανόνα, ενώ το φύλλο (αποτέλεσμα) αντιστοιχεί στο δεξιό μέρος του κανόνα. Για παράδειγμα:

IF Color=”Green” AND Size=”Small” THEN Fruit=”Grapes”

ΕΠΙΛΟΓΟΣ

Η τεχνητή νοημοσύνη και πιο συγκεκριμένα η μηχανική μάθηση είναι ένας τομέας που ανθίζει όλο και περισσότερο με εκατοντάδες εφαρμογές. Όμως μέχρι τώρα ακριβώς εκεί που υπήρχε ανάγκη για επεξεργασία μεγάλου όγκου δεδομένων, δεν ήταν δυνατό ή συνήθως ασύμφορο για μικρότερες επιχειρήσεις για την εφαρμογή της. Πλέον όμως χάρη στο Apache Hadoop, αυτό μπορεί να γίνει με πολύ μικρότερο κόστος και με μεγαλύτερη ευκολία από συστήματα σχεσιακών βάσεων δεδομένων και HPC cluster και με μεγαλύτερη επιτυχία και ταχύτητα αναλογικά με το κόστος. Αναγνωρίζοντας αυτή την ανάγκη, η κοινότητα της Apache δημιούργησε το Apache Mahout, μία βιβλιοθήκη από αλγορίθμους μηχανικής μάθησης, «έτοιμοι» και ελεύθεροι για υλοποίηση με μεγάλη επεκτασιμότητα.

ΚΕΦΑΛΑΙΟ 3

Υλοποίηση Naïve Bayes με χρήση Apache Mahout στο Apache Hadoop

ΕΙΣΑΓΩΓΗ

Για την υλοποίηση ενός αλγορίθμου ταξινόμησης (classification), στην πλατφόρμα του Apache Hadoop, χρησιμοποιώντας το μοντέλο του MapReduce επιλέχτηκε ο αλγόριθμος Naïve Bayes. Το Hadoop όπως έχει αναφερθεί είναι πολύ κατάλληλο για τέτοιου είδους εφαρμογές που επιδρούν σε πολλά δεδομένα χάρη στην κατανεμημένη αποθήκευση και επεξεργασία του. Επίσης, το Hadoop επιλέχτηκε γιατί η μηχανική μάθηση είναι υπολογιστικά πολύπλοκη. Για παράδειγμα:

πολυπλοκότητα Naïve Bayes → O(n*m) (3.1)

πολυπλοκότητα C4.5 (Δέντρο Απόφασης) → $O(m*n^2)$ (3.2)

Οπότε το Mahout είναι πολύ κατάλληλο χάρη στην επεκτασιμότητα των αλγορίθμων και η παράλληλη λειτουργία του πάνω στο Hadoop μειώνει την πολυπλοκότητα ενός αλγορίθμου. Συγκεκριμένα επιλέχτηκε η υλοποίηση του Naïve Bayes που είναι από τους διαθέσιμους αλγορίθμους στο Apache Mahout και για την δοκιμή του εκτελέστηκε το Twenty Newsgroups Classification παράδειγμα.

Το Twenty Newsgroups dataset, είναι μια συλλογή από περίπου 20.000 έγγραφα newsgroup, διαχωρισμένη (σχεδόν) ομοιόμορφα σε 20 διαφορετικές ομάδες. Η συλλογή αυτών των είκοσι διαφορετικών ομάδων έχει γίνει ένα δημοφιλές σύνολο δεδομένων για πειράματα σε εφαρμογές κειμένου, σε τεχνικές μηχανικής μάθησης, όπως η ταξινόμηση και η ομαδοποίηση σε κείμενο. Θα χρησιμοποιήσουμε τον

Mahout Bayes ταξινομητή για την δημιουργία ενός μοντέλου που θα μπορούσε να ταξινομήσει ένα νέο έγγραφο σε μία από τις είκοσι ομάδες συζητήσεων.

ΥΠΟΚΕΦΑΛΑΙΟ 3.1

Προαπαιτήσεις για την υλοποίηση

Η υλοποίηση έγινε χρησιμοποιώντας:

- ένα εικονικό μηχάνημα μέσω του Oracle VM Virtualbox 4.3.4 r91027,
- με εγκατεστημένο το λειτουργικό CentOS 6.2 x64 bit,
- τον ssh-server για επικοινωνία με το λειτουργικό σύστημα,
- την διανομή της Cloudera CDH 4.4 σε pseudo-distributed mode (σε ψευδό-κατανεμημένη λειτουργία) μέσω του Cloudera Manager,
- με εγκατεστημένα τα:
- το Apache Maven 3.0.4 για την εγκατάσταση του Apache Mahout
- το Apache Mahout 0.8.

Για την εγκατάσταση του CentOS μπορείτε να εγκατασταθεί και το γραφικό περιβάλλον αλλά δεν είναι απαραίτητο ειδικά για τους slave κόμβους (αν γίνετε κανονική εγκατάσταση όχι ψευδό-κατανεμημένη). Και αν χρειαστεί αργότερα μπορεί να γίνει με την εντολή:
$ yum – y groupinstall “X Window System” “Desktop”.
Για να δουλέψει σωστά ο ssh-server αλλά και η επικοινωνία των nodes μεταξύ τους και με τον Cloudera Manager στο αρχείο */etc/hosts* πρέπει να υπάρχει κάθε κόμβος στην μορφή:
192.168.1.101 n1.example.com n1
192.168.1.102 n2.example.com n2
…
Για την εγκατάσταση και την λειτουργία του Cloudera Manager πρέπει να απενεργοποιηθεί το SELINUX (=disabled).
Ακολουθούν οι εντολές για την εγκατάσταση του Apache Maven:

$ wget http://mirror.cc.columbia.edu/pub/software/apache/maven/maven-3/3.0.5/binaries/apache-maven-3.0.5-bin.tar.gz (όπου 3 και 3.0.5 μπαίνει η αντίστοιχη έκδοση)

$ sudo tar xzf apache-maven-3.0.5-bin.tar.gz -C /usr/local
$ cd /usr/local
$ sudo ln -s apache-maven-3.0.5 maven
$ sudo vi /etc/profile.d/maven.sh

Και μέσα στο παραπάνω αρχείο βάλετε τα παρακάτω:
export M2_HOME=/usr/local/maven
export PATH=${M2_HOME}/bin:${PATH}
Τέλος,
$ mvn –version
για τον έλεγχο της έκδοσης.

Για την εγκατάσταση του Mahout:
$ svn co http://svn.apache.org/repos/asf/mahout/trunk
Μετά πηγαίνετε στον κατάλογο που έγινε check out το Mahout και εκτελέσετε τα παρακάτω:
$ mvn –DskipTests install
$ mvn test
Μετά πηγαίνετε στον core κατάλογο που βρίσκεται στον κατάλογο που ήσασταν και εκτελέσετε τα παρακάτω:
$ mvn –DskipTests install
Μετά πηγαίνετε στον κατάλογο examples και εκτελέσετε τα παρακάτω:
$ mvn compile
Μετά σιγουρευτείτε ότι η $HADOOP_HOME μεταβλητή έχει την τοποθεσία του Hadoop (π.χ. /usr/lib/Hadoop).
Το ίδιο και για την $JAVA_HOME.
Το ίδιο και για την $MAHOUT_HOME (π.χ. /usr/lib/mahout ή εκεί που μόλις το κάνατε εγκατάσταση).
Τέλος, αν θέλετε να εκτελέσετε το Mahout τοπικά θέσετε την $MAHOUT_LOCAL=TRUE.

Για όλες τις παραπάνω περιπτώσεις οι εντολές είναι ίδιες οπότε παρακάτω υπάρχουν μόνο για την $HADOOP_HOME:

$ export HADOOP_HOME=/usr/lib/hadoop
$ export PATH=$PATH:$HADOOP_HOME/bin

Για την εκτέλεση του Twenty Newsgroups Classification Example εκτελέσετε τα παρακάτω:
Αν δεν είναι σε λειτουργία το Hadoop
$ cd $HADOOP_HOME/bin
$ ./start-all.sh # αυτό δεν ισχύει για την διανομή της Cloudera του Hadoop.

Εκτέλεση του classify-20newsgroups.sh το οποίο βρίσκεται στο παράρτημα και αυτοματοποιεί την εκτέλεση του παραδείγματος. Τα βήματα που εκτελεί είναι:

- Κατεβάζει το 20news-bydate.tar.gz από το Twenty Newsgroups dataset.
- Εξάγει το dataset.
- Παράγει το dataset για εισαγωγή στον ταξινομητή εκπαίδευσης (training classifier).
- Παράγει το dataset για εισαγωγή στον ταξινομητή δοκιμής (testing classifier).
- Εκπαιδεύει τον ταξινομητή.
- Δοκιμάζει τον ταξινομητή.

Το script δίνει επιλογή για καθαρισμό της προσωρινής τοποθεσίας των δεδομένων αν εκτελέστηκε ξανά το script και επιλεγεί ανάμεσα σε τρεις αλγορίθμους: τον Complementary Naive Bayes, τον Naive Bayes και τον SGD (Stochastic Gradient Descent). Για την εκτέλεση του πληκτρολογήσετε τα παρακάτω:
$ cd $MAHOUT_HOME
$ ./ examples/bin/classify-20newsgroups.sh
Για εκτέλεση των βημάτων χειροκίνητα, απαιτείται η εκτέλεση των παρακάτω εντολών αφού κατεβάσετε το αρχείο από το παρακάτω link http://people.csail.mit.edu/jrennie/20Newsgroups/20news-bydate.tar.gz .

1. Εξαγωγή των αρχείων:

   $ mkdir 20news-bydate
   $ tar –zxvf 20news-bydate.tar.gz –C 20news-bydate

2. Δημιουργία νέου καταλόγου και αντιγραφή όλου του dataset εκεί:

   $ mkdir 20news-all
   $ cp -R 20news-bydate//* 20news-all*

3. Δημιουργία σειριακών αρχείων από το dataset, το οποίο είναι ο ενδιάμεσος τύπος αρχείων που θα χρησιμοποιηθεί από το Mahout:

   $ mahout seqdirectory \
   $ -i 20news-all \
   $ -o 20news-seq

   -i κατάλογος εισαγωγής
   -o κατάλογος εξαγωγής

4. Μετατροπή των σειριακών αρχείων εξαγωγής σε διανύσματα, ο τύπος αρχείων που χρησιμοποιεί το Mahout:

   $ mahout seq2sparse \
   $ -i 20news-seq \
   $ -o 20news-vectors -lnorm -nv -wt tfidf

   -i κατάλογος εισαγωγής
   -o κατάλογος εξαγωγής
   -lnorm αν τα διανύσματα εξόδου πρέπει να είναι logNormalize
   -nv αν τα διανύσματα εξόδου πρέπει να είναι NamedVectors
   -wt το είδος του βάρους που χρησιμοποιείτε (π.χ. TFIDF)

5. Διαχωρισμός των παραγόμενων διανυσματικών δεδομένων για την δημιουργία δύο σετ: το Training set (τα δεδομένα που θα χρησιμοποιηθούν για την εκπαίδευση του αλγορίθμου ταξινόμησης για την παραγωγή του μοντέλου) και το Holdout set (τα δεδομένα που θα χρησιμοποιηθούν για την δοκιμή του αλγορίθμου ταξινόμησης). Στην συγκεκριμένη περίπτωση παίρνουμε το 20% των αρχείων από κάθε κατάλογο και το γράφουμε στον κατάλογο δοκιμής. Το υπόλοιπο των αρχείων γράφετε στον κατάλογο με τα δεδομένα εκπαίδευσης.

   $ mahout split \
   $ -i 20news-vectors/tfidf-vectors \
   $ --trainingOutput 20news-train-vectors \
   $ --testOutput 20news-test-vectors \
   $ --randomSelectionPct 20 --overwrite --sequenceFiles -xm sequential

   -i κατάλογος εισαγωγής
   --sequenceFiles δείχνει ότι τα αρχεία εισόδου είναι σειριακά
   -xm μέθοδος που θα χρησιμοποιηθεί (sequencial ή mapreduce)
   --randomSelectionPct δείχνει το ποσοστό των δεδομένων που θα χρησιμοποιηθεί για testing
   -ow αντικατάσταση του καταλόγου εξόδου

6. Εκπαίδευση του Naive Bayes Ταξινομητή. Η διαδικασία εκπαίδευσης θα αναλύσει τις λέξεις που εμφανίζονται στα αρχεία και ανήκουν σε μια συγκεκριμένη κατηγορία και θα παράγει το μοντέλο που θα χρησιμοποιηθεί για να καθοριστεί η πλέον πιθανή κατηγορία για ένα έγγραφο με βάση τις λέξεις που έχει. Κατά την εκπαίδευση, ο αλγόριθμος καταμετρά τον αριθμό του πόσες φορές εμφανίζεται κάθε λέξη σε ένα έγγραφο σε μια κλάση και τον διαιρεί με τον αριθμό των λέξεων που εμφανίζονται συνολικά σε αυτή την κλάση. Αυτό είναι η δεσμευμένη πιθανότητα, η πιθανότητα του να εμφανιστεί μία λέξη σε μια συγκεκριμένη κατηγορία.

   $mahout trainnb \
   $ -i 20news-train-vectors -el \

$ -o model \
$ -li labelindex \
$ -ow

-I κατάλογος εισόδου – το σετ των δεδομένων εκπαίδευσης
-o μοντέλο εξόδου
-li διαδρομή στο χώρο που θα αποθηκευτεί το label index
ow αντικατάσταση του καταλόγου εξόδου

7. Προβολή του label index (καταλόγος δεικτών με τις κατηγορίες)

   $ mahout seqdumper -i labelindex

8. Δοκιμή του αλγορίθμου χρησιμοποιώντας το σετ δεδομένων εκπαίδευσης. Εφόσον εδώ τα ίδια δεδομένα χρησιμοποιούνται και για την εκπαίδευση και για την δοκιμή του αλγορίθμου, περιμένουμε τα αποτελέσματα να είναι 100% ακριβή.

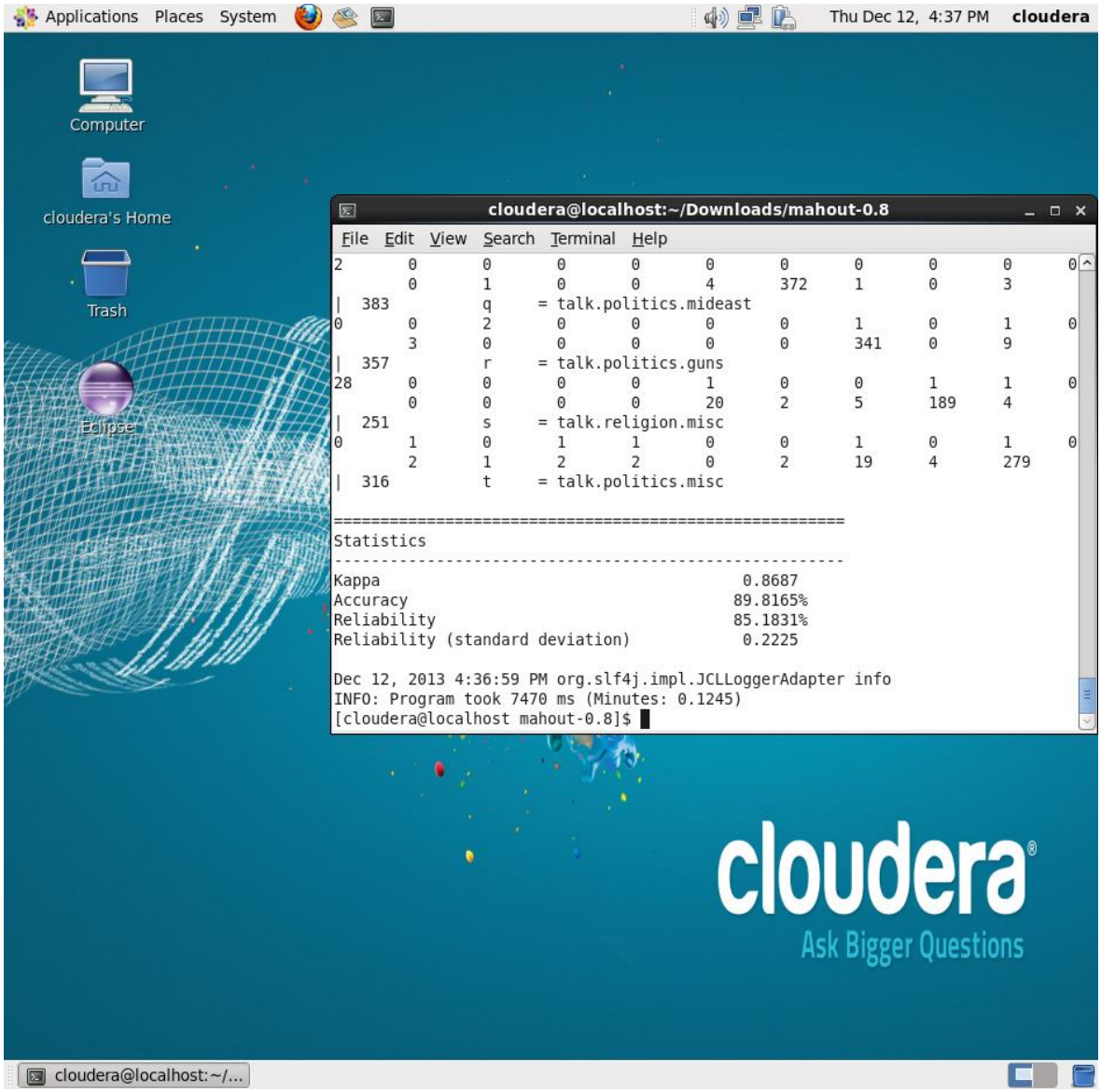
   $ mahout testnb \
   $ -i 20news-train-vectors\
   $ -m model \
   $ -l labelindex \
   $ -ow -o 20news-testing

   -I κατάλογος εισόδου με τα δεδομένα προς ταξινόμηση
   -m το μοντέλο του ταξινομητή που θα χρησιμοποιηθεί (αυτό που δημιουργήθηκε προηγουμένως)
   -l διαδρομή του label index
   ow αντικατάσταση του καταλόγου εξόδου
   -o κατάλογος εξόδου

9. Δοκιμή του ταξινομητή χρησιμοποιώντας τα πραγματικά δεδομένα δοκιμής. Τώρα τα αποτελέσματα δεν θα είναι όπως είναι αναμενόμενο 100% ακριβή, εφόσον αυτά τα δεδομένα δεν χρησιμοποιήθηκαν κατά την εκπαίδευση.

   *$ mahout testnb *
   $ -i 20news-test-vectors
   *$ -m model *
   *$ -l labelindex *
   $ -ow -o 20news-testing2

Το αποτέλεσμα στο τέλος είναι κάπως έτσι:



Σχήμα 1 "Αποτελέσματα εκτέλεσης Naive Bayes στο Apache Mahout"

Όπως φαίνεται από την εικόνα η ακρίβεια των αποτελεσμάτων είναι στο 89.8 (το έχω δει μέχρι και 95% σε κάποιες εκτελεσεις). Σε γενικές γραμμές είναι σχετικά καλή ταξινόμηση, για παράδειγμα όπως φαίνεται από την εικόνα, στην κατηγορία talk.politics.guns ταξινόμησε σωστά τα 341 από τα 357. Δηλαδή ταξινόμησε λάθος μόνο 16.

ΕΠΙΛΟΓΟΣ

Αυτό έγινε εφικτό χάρη στις δυνατότητες του Apache Mahout και την απόδοση του Apache Hadoop. Θα μπορούσε να γίνει το ίδιο σε κάποια άλλη γλώσσα και να τρέξουμε κανονικά σε οποιοδήποτε σύστημα ανάλογα την γλώσσα (Java,.Net,php κλπ.). Μάλιστα υπάρχουν έτοιμες υλοποιήσεις για το Naive Bayes ταξινομητή και άλλους αλγορίθμους μηχανικής μάθησης οι οποίοι είναι πολύ πιο εύκολοι στην χρήση από το Mahout, αλλά όσο αυξάνουν τα δεδομένα αυξάνεται και ο χρόνος εκτέλεσης ταχύτατα. Για δεδομένα σε μέγεθος των Gigabyte ή έως και Exabyte που είναι ικανό το Apache Hadoop να διαχειριστεί, κάποιος «κλασσικός» αλγόριθμος θα έκανε υπερβολικά πολύ χρόνο να τελειώσει, εκτός και αν ήταν υλοποιημένος σε MPI και έτρεχε παράλληλα σε ένα HPC cluster. Παρόλα αυτά και εκεί έχει δυσκολίες η υλοποίηση γιατί συνήθως είναι πολύ πιο χαμηλού επιπέδου (πιο πολύ λεπτομέρεια – ανταλλαγή μηνυμάτων MPI κλπ.) αλλά και πιο δύσκολο στην διαχείριση κάποιες φορές το HPC cluster. Ενώ με το Apache Hadoop και ειδικά με την βοήθεια κάποιων διανομών η διαχείριση είναι πολύ απλή και η υλοποίηση πιο εύκολη από την χρήση του MPI (αλλά όχι από την απλή χρήση μιας βιβλιοθήκης). Αυτό είναι και το αρνητικό, του Apache Mahout, αλλά και του οικοσυστήματος του Apache Hadoop γενικότερα, ότι δεν υπάρχει ακόμη πάρα

πολύ γνώση και θέλει υπομονή, χρόνο και πολύ ερεύνα. Υπάρχουν βέβαια μαθήματα από εταιρείες όπως η Cloudera, η Hortonworks κ.α. αλλά πολλά από αυτά είναι με πληρωμή και όσα είναι δωρεάν δεν είναι πολύ προχωρημένα. Φυσικά υπάρχει και υποστήριξη από την κοινότητα αλλά δεν χρησιμοποιείται από «απλού» χρήστες σε τέτοιο επίπεδο ώστε να είναι εύκολη η εύρεση της πληροφορίας, όπως για άλλα project και τεχνολογίες. Για παράδειγμα, για να εφαρμόσεις τον Naive Bayes σε δικό σου dataset πρέπει να γράψεις δικό σου κώδικα για μετατροπή στον κατάλληλο τύπο δεδομένων και ίσως και δικό σου κώδικα ή μετατροπή κάποιου άλλου για την εκπαίδευση, το μοντέλο κλπ. Θέλει λοιπόν, να το αγαπήσεις το Hadoop και μετά θα σε ανταμείψει με τις επιδόσεις του και τις δυνατότητες που προσφέρει, γιατί σίγουρα είναι η καλύτερη λύση όταν πρόκειται για πολύ μεγάλο όγκο δεδομένων, χάρη στην δύναμη του MapReduce και την οικονομική του προσέγγιση.

ΒΙΒΛΙΟΓΡΑΦΙΑ

1. http://hadoop.apache.org/
2. http://ambari.apache.org/
3. http://avro.apache.org/
4. http://cassandra.apache.org/
5. http://chukwa.apache.org/
6. http://hbase.apache.org/
7. http://hive.apache.org/
8. http://mahout.apache.org/
9. http://pig.apache.org/
10. http://zookeeper.apache.org/
11. http://ganglia.sourceforge.net/
12. http://www.nagios.org/
13. http://hive.apache.org/docs/hcat_r0.5.0/
14. http://oozie.apache.org/
15. http://sqoop.apache.org/
16. http://el.wikipedia.org/wiki/Αναγνώριση_προτύπων
17. http://el.wikipedia.org/wiki/Κατηγοριοποίηση
18. http://aibook.csd.auth.gr/include/slides/Chap24.pdf
19. http://www.mapr.com/products/apache-hadoop
20. http://www.mapr.com/products
21. http://www.mapr.com/products/google-cloud-platform
22. http://en.wikipedia.org/wiki/MapReduce
23. http://en.wikipedia.org/wiki/Hadoop
24. http://hortonworks.com/products/hdp-2/
25. http://hortonworks.com/why-hortonworks-for-hadoop/
26. http://docs.hortonworks.com/HDPDocuments/HDP1/HDP-Win-1.3.0/bk_installing_hdp_for_windows/content/win-getting-ready.html
27. http://www.youtube.com/watch?v=7FcMhTTG1Cs
28. http://www.cloudera.com/content/support/en/documentation/manager/cloudera-manager-v4-latest.html
29. http://www.cloudera.com/content/cloudera/en/products-and-services.html
30. http://www.youtube.com/watch?v=A02SRdyoshM

31. http://www.microsoft.com/en-us/sqlserver/solutions-technologies/business-intelligence/big-data.aspx
32. http://www.windowsazure.com/en-us/services/hdinsight/
33. http://hadoop.intel.com/
34. https://cwiki.apache.org/confluence/display/MAHOUT/Quickstart#Footnote1
35. https://cwiki.apache.org/confluence/display/MAHOUT/Twenty+Newsgroups
36. https://cwiki.apache.org/confluence/display/MAHOUT/Wikipedia+Bayes+Example
37. https://cwiki.apache.org/confluence/display/MAHOUT/BuildingMahout
38. http://preilly.me/2013/05/10/how-to-install-maven-on-centos/
39. http://phillyjug.files.wordpress.com/2013/03/hadoopmahout.pdf
40. http://www.blogdugas.net/?p=113
41. http://chimpler.wordpress.com/2013/06/24/using-the-mahout-naive-bayes-classifier-to-automatically-classify-twitter-messages-part-2-distribute-classification-with-hadoop/
42. http://skife.org/mahout/2013/02/14/first_steps_with_mahout.html
43. http://hanishblogger.blogspot.gr/2013/03/machine-learning-categorization-with.html
44. http://sujitpal.blogspot.gr/2012/09/learning-mahout-classification.html
45. https://cwiki.apache.org/confluence/display/MAHOUT/Creating+Vectors
46. http://stackoverflow.com/questions/14998250/simple-mahout-classification-example
47. http://whichlight.com/blog/news-article-recommendations-using-bayesian-classification-with-apache-mahout/
48. https://github.com/nelken/Mahout-example
49. http://acadmedia.wku.edu/Pavan/Mahout_in_Action.pdf
50. https://bitbucket.org/jaganadhg/blog/src/7b5dac6e138d013016d49edfc4944ec98c36cf2d/bck9/java/src/org/bc/kl/ClassifierDemo.java?at=default
51. https://github.com/fredang/mahout-naive-bayes-example/tree/master/src/main/java/com/chimpler/example/bayes
52. https://github.com/fredang/mahout-naive-bayes-example2/blob/master/src/main/java/com/chimpler/example/bayes2/Classifier.java
53. https://cwiki.apache.org/confluence/display/MAHOUT/Creating+Vectors+from+Text

54. https://github.com/tdunning/Chapter-16
55. https://github.com/fredang/mahout-naive-bayes-example/tree/master/src/main/java/com/chimpler/example/bayes
56. http://www.cs.ucy.ac.cy/courses/EPL660/labs/Stalo/Lab8.pdf
57. http://aws.amazon.com/free/
58. https://cloud.google.com/products/compute-engine/
59. https://github.com/itsamiths/hadoop-eclipse-plugins
60. http://developer.yahoo.com/hadoop/tutorial/module3.html
61. http://wiki.apache.org/hadoop/EclipseEnvironment
62. http://blog.cloudera.com/blog/2009/04/configuring-eclipse-for-hadoop-development-a-screencast/

ΠΑΡΑΡΤΗΜΑΤΑ

Κώδικας Script classify-20newsgroups.sh

```
#!/bin/bash
#
# Licensed to the Apache Software Foundation (ASF) under one or more
# contributor license agreements.  See the NOTICE file distributed with
# this work for additional information regarding copyright ownership.
# The ASF licenses this file to You under the Apache License, Version 2.0
# (the "License"); you may not use this file except in compliance with
# the License.  You may obtain a copy of the License at
#
#     http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.
#


#
# Downloads the 20newsgroups dataset, trains and tests a classifier.
#
# To run:  change into the mahout directory and type:
# examples/bin/classify-20newsgroups.sh

if [ "$1" = "--help" ] || [ "$1" = "--?" ]; then
  echo "This script runs SGD and Bayes classifiers over the classic 20 News Groups."
  exit
fi

SCRIPT_PATH=${0%/*}
```

```
if [ "$0" != "$SCRIPT_PATH" ] && [ "$SCRIPT_PATH" != "" ]; then
 cd $SCRIPT_PATH
fi
START_PATH=`pwd`

WORK_DIR=/tmp/mahout-work-${USER}
algorithm=( cnaivebayes naivebayes sgd clean)
if [ -n "$1" ]; then
 choice=$1
else
 echo "Please select a number to choose the corresponding task to run"
 echo "1. ${algorithm[0]}"
 echo "2. ${algorithm[1]}"
 echo "3. ${algorithm[2]}"
 echo "4. ${algorithm[3]} -- cleans up the work area in $WORK_DIR"
 read -p "Enter your choice : " choice
fi

echo "ok. You chose $choice and we'll use ${algorithm[$choice-1]}"
alg=${algorithm[$choice-1]}

echo "creating work directory at ${WORK_DIR}"

mkdir -p ${WORK_DIR}
if [ ! -e ${WORK_DIR}/20news-bayesinput ]; then
 if [ ! -e ${WORK_DIR}/20news-bydate ]; then
  if [ ! -f ${WORK_DIR}/20news-bydate.tar.gz ]; then
   echo "Downloading 20news-bydate"
   curl http://people.csail.mit.edu/jrennie/20Newsgroups/20news-bydate.tar.gz -o
${WORK_DIR}/20news-bydate.tar.gz
  fi
  mkdir -p ${WORK_DIR}/20news-bydate
  echo "Extracting..."
  cd ${WORK_DIR}/20news-bydate && tar xzf ../20news-bydate.tar.gz && cd .. && cd ..
 fi
```

```
fi
#echo $START_PATH
cd $START_PATH
cd ../..

set -e

if [ "x$alg" == "xnaivebayes"  -o  "x$alg" == "xcnaivebayes" ]; then
 c=""

 if [ "x$alg" == "xcnaivebayes" ]; then
   c=" -c"
 fi

 set -x
 echo "Preparing 20newsgroups data"
 rm -rf ${WORK_DIR}/20news-all
 mkdir ${WORK_DIR}/20news-all
 cp -R ${WORK_DIR}/20news-bydate/*/* ${WORK_DIR}/20news-all

 echo "Creating sequence files from 20newsgroups data"
 ./bin/mahout seqdirectory \
   -i ${WORK_DIR}/20news-all \
   -o ${WORK_DIR}/20news-seq -ow

 echo "Converting sequence files to vectors"
 ./bin/mahout seq2sparse \
   -i ${WORK_DIR}/20news-seq \
   -o ${WORK_DIR}/20news-vectors  -lnorm -nv  -wt tfidf

 echo "Creating training and holdout set with a random 80-20 split of the generated vector
dataset"
 ./bin/mahout split \
   -i ${WORK_DIR}/20news-vectors/tfidf-vectors \
   --trainingOutput ${WORK_DIR}/20news-train-vectors \
```

```
  --testOutput ${WORK_DIR}/20news-test-vectors  \
  --randomSelectionPct 40 --overwrite --sequenceFiles -xm sequential

 echo "Training Naive Bayes model"
 ./bin/mahout trainnb \
  -i ${WORK_DIR}/20news-train-vectors -el \
  -o ${WORK_DIR}/model \
  -li ${WORK_DIR}/labelindex \
  -ow $c

 echo "Self testing on training set"

 ./bin/mahout testnb \
  -i ${WORK_DIR}/20news-train-vectors\
  -m ${WORK_DIR}/model \
  -l ${WORK_DIR}/labelindex \
  -ow -o ${WORK_DIR}/20news-testing $c

 echo "Testing on holdout set"

 ./bin/mahout testnb \
  -i ${WORK_DIR}/20news-test-vectors\
  -m ${WORK_DIR}/model \
  -l ${WORK_DIR}/labelindex \
  -ow -o ${WORK_DIR}/20news-testing $c

elif [ "x$alg" == "xsgd" ]; then
 if [ ! -e "/tmp/news-group.model" ]; then
  echo "Training on ${WORK_DIR}/20news-bydate/20news-bydate-train/"
  ./bin/mahout org.apache.mahout.classifier.sgd.TrainNewsGroups ${WORK_DIR}/20news-
bydate/20news-bydate-train/
 fi
 echo "Testing on ${WORK_DIR}/20news-bydate/20news-bydate-test/ with model: /tmp/news-
group.model"
```

```
 ./bin/mahout org.apache.mahout.classifier.sgd.TestNewsGroups --input ${WORK_DIR}/20news-
bydate/20news-bydate-test/ --model /tmp/news-group.model
elif [ "x$alg" == "xclean" ]; then
 rm -rf ${WORK_DIR}
 rm -rf /tmp/news-group.model
fi
# Remove the work directory
#
```