Εξόρυξη, ομαδοποίηση και παρουσίαση δεδομένων από τον Παγκόσμιο Ιστό

Ονοματεπώνυμο: Μοσχόπουλος Νικόλαος
ΑΜ: 1054315
Έτος:4ο

Εργασία στα πλαίσια του μαθήματος:
Εξόρυξη Δεδομένων και Αλγόριθμοι Μάθησης

Περιεχόμενα

Εισαγωγή

Ο παγκόσμιος ιστός είναι μια τεράστια πηγή δεδομένων σχεδόν όλων των ειδών. Σε μία τέτοια πηγή δεδομένων μπορεί κανείς να βρει οποιαδήποτε πληροφορία επιθυμεί με ελάχιστο έως μηδαμινό κόστος σε χρόνο και χρήμα. Βέβαια, αυτές οι πληροφορίες δεν είναι πάντα εύκολο να αντληθούν επαρκώς, αφού είναι συχνά διάσπαρτες μεταξύ πολλών web servers ή έχουν αρκετές ποικίλες μορφές. Αν υπήρχε κάποιος τρόπος να συλλεχθεί και να ομαδοποιηθεί μεγάλος όγκος δεδομένων σε μικρό χρόνο και χώρο θα ήταν πολύ πιο εύκολο να αναζητηθεί κάποια πληροφορία, να παρθεί κάποια σημαντική απόφαση ή να αξιολογηθεί μια κατάσταση. Στο παρελθόν υπήρξαν πολλές τέτοιες προσπάθειες, αλλά δεν έπεισαν ιδιαίτερα αφού ήταν χρονοβόρες ή δεν κάλυπταν χιλιάδες παρά μόνο δεκάδες πηγές. Σήμερα έχει γίνει αρκετά μεγάλη βελτίωση όσον αφορά και την εξόρυξη τεράστιων όγκων δεδομένων από το διαδίκτυο, μα και όσον αφορά την καλύτερη ομαδοποίηση και παρουσίασή τους στο ευρύ κοινό. Για παράδειγμα ένας τρόπος με τον οποίο γίνεται το πρώτο είναι το DIPRE (Dual Iterative Pattern Relation Expansion) το οποίο βασίζεται στη δυαδικότητα μεταξύ προτύπων και σχέσεων, ενώ για το δεύτερο υπάρχουν τα ιστογράμματα τα οποία αναλαμβάνουν να αναλύσουν και να περιγράψουν πολλά στοιχεία βάσεων δεδομένων με σύντομο και σαφή τρόπο.

Η εξαγωγή πληροφοριών

Η εξαγωγή πληροφοριών μπορεί να χρονολογηθεί περίπου από το 1970. Από τότε είχαν αναπτυχθεί πολλές διαφορετικές προσεγγίσεις για διάφορες τέτοιες εργασίες ανάλογα με το στόχο της μελέτης και τη φύση της πηγής πληροφορίας. Τα περισσότερα συστήματα εξόρυξης δεδομένων σήμερα βασίζονται σε μεθόδους που χρησιμοποιούν κάποια μορφή μηχανικής μάθησης. Γενικά πάντως όλα τα συστήματα εξαγωγής πληροφοριών έχουν τρία κοινά γνωρίσματα:

- μια πηγή που περιλαμβάνει χρήσιμες οντότητες που είναι κρυμμένες σε ένα αχανές κείμενο,
- μια σημασιολογική ή γλωσσική πηγή στην οποία παρέχεται το πλαίσιο των σχέσεων μεταξύ οντοτήτων και
- αλγόριθμοι για την εκτέλεση των εργασιών επεξεργασίας.

Αυτά τα συστήματα βασίζονται συνήθως σε ένα σύνολο αρχικών προτύπων (Seed Patterns) που χρησιμοποιούν για να συγκρίνουν και να ανακτήσουν σχετικές πληροφορίες από άλλες βάσεις δεδομένων. Από αυτά δημιουργούνται άλλα πρότυπα τα οποία χρησιμοποιούνται επίσης για εξαγωγή νέων δεδομένων. Δεν είναι όμως κάθε πρότυπο σχετικό με την εκάστοτε έρευνα. Έγκυρα πρότυπα μπορούν να κάνουν ένα σύστημα ακριβέστερο για μια έρευνα, ενώ λανθασμένα πρότυπα το κάνουν πιο αναξιόπιστο αφού προσθέτουν άχρηστα δεδομένα. Ένα σημαντικό αρνητικό στοιχείο που έχουν αυτά τα συστήματα είναι ότι χρειάζεται πολλές φορές η χειρωνακτική παρέμβαση του ανθρώπου για τη βελτίωση της διαδικασίας, αφού τα δεδομένα συχνά χρειάζονται αλλαγές για να εξαχθεί η επιθυμητή σχέση-στόχος. Η μείωση του ανθρώπινου φόρτου στην ανάπτυξη κανόνων εξαγωγής αποτελεί τεράστια πρόκληση για όλους όσους ασχολούνται με την εξαγωγή πληροφοριών. Ανάλογα με τα επίπεδα της ανθρώπινης εμπλοκής τα περισσότερα συστήματα χρησιμοποιούν διαφορετικούς αλγορίθμους μηχανικής μάθησης.

Τα συστήματα αυτά μπορούν να ενταχθούν σε τρεις μεγάλες κατηγορίες: supervised, non-supervised και semi-supervised. Τα supervised συστήματα χρησιμοποιούν ένα σύνολο ενδεικτικών χειροκίνητων δεδομένων για να δημιουργήσουν κανόνες εξαγωγής και απαιτούν ανθρώπινη εποπτεία. Χρειάζονται κείμενα και συγκεκριμένες ονοματικές φράσεις για τη δημιουργία προτύπων. Η επίπονη δουλειά της χειροκίνητης ανθρώπινης παρέμβασης τα καθιστά συχνά δυσμενή σε πολλές πρακτικές εφαρμογές. Τα non-supervised συστήματα χρησιμοποιούν εξαιρετικά αυτοματοποιημένους αλγορίθμους και ως εκ τούτου δεν απαιτούν μεγάλη ανθρώπινη παρέμβαση. Στα μειονεκτήματά τους συγκαταλλέγεται το γεγονός ότι εμφανίζουν ανακρίβειες στα αποτελέσματα. Τα semi-supervised συστήματα βρίσκονται κάπου ανάμεσα στα supervised και τα non-supervised. Θεωρούνται πιο πρακτικά από τα δυο προαναφερθέντα, αφού οι αλγοριθμοί τους έχουν την ικανότητα να παράγουν νέους αλγορίθμους για την εξόρυξη των δεδομένων μέσα από τη διαδικασία της μηχανικής μάθησης, ενώ σταματούν μερικές φορές αυτόματα όταν η αξιοπιστία των αποτελεσμάτων πέσει κάτω από ένα όριο. Χρειάζονται μικρή ανθρώπινη εποπτεία.

Η τεχνική DIPRE

Σκοπός των σύγχρονων μεθόδων είναι να δώσουν όσο το δυνατόν καλύτερα αποτελέσματα στην αναζήτηση δεδομένων, περιορίζοντας παράλληλα όσο περισσότερο

γίνεται την παρέμβαση του ανθρώπινου παράγοντα. Μια τέτοια μέθοδος είναι το DIPRE που πρωτοδιατυπώθηκε από τον Sergey Brin, το οποίο εκμεταλλευόμενο τη δυαδικότητα μεταξύ συνόλων προτύπων και σχέσεων αναπτύσσει μια σχέση ξεκινώντας από ένα μικρό δείγμα. Πιο συγκεκριμένα αν θεωρηθεί ότι υπάρχει μικρός αριθμός βιβλίων ψάχνοντας περισσότερα και ότι η αρχική σχέση είναι συγγραφέας-τίτλος, μπορεί να βρεθεί ένα σύνολο βιβλίων από το διαδίκτυο με αυτά τα χαρακτηριστικά. Από αυτά τα βιβλία αντλούνται νέα πρότυπα από τις αναφορές τους και χρησιμοποιούνται για να βρεθούν νέα βιβλία. Έτσι αποκτιούνται όλο και περισσότερα πρότυπα και παράλληλα βιβλία φτάνοντας τελικά να υπάρχει μια μεγάλη λίστα βιβλίων αλλά και προτύπων που βοήθησαν στον εντοπισμό τους.

Αν έπρεπε να προσδιοριστεί το πρόβλημα σε μια πιο επιστημονική βάση, θα μπορούσε να θεωρηθεί μια βάση δεδομένων D με μη αρχειοθετημένες πληροφορίες όπως για παράδειγμα το Ίντερνετ. Το R είναι η σχέση στην οποία πρέπει να καταλήξει ο αλγόριθμος όταν τερματίσει η οποία αποτελείται από τα στοιχεία r1,…,r_n (δηλαδή R=r1,…,r_n). Κάθε πλειάδα t του R υπάρχει μία ή παραπάνω φορές στο D και κάθε τέτοια εμφάνιση αποτελείται από όλα τα κομμάτια του t, που παριστάνονται σαν strings σε μικρή απόσταση μεταξύ τους στο D. Στο παράδειγμα των βιβλίων που αναφέρθηκε και προηγουμένως μπορεί να θεωρηθεί ότι η σχέση στόχος R που αναζητείται είναι το σύνολο των βιβλίων που εμφανίζονται στον Παγκόσμιο Ιστό τα οποία προσπαθούν να αντληθούν με βάση το πρότυπο ζεύγους συγγραφέας-τίτλος.

Δυο σημαντικές έννοιες που θα απασχολήσουν στη συνέχεια και πρέπει να αναλυθούν είναι το coverage και το error rate. Το coverage είναι ένα κλάσμα εγγραφών που βοηθά να κριθεί και να αξιολογηθεί η ποιότητα ενός κανόνα ταξινόμησης, ενώ το error rate εκφράζει το ποσοστό των προτύπων που έχουν ταξινομηθεί εσφαλμένα από ένα μοντέλο απόφασης. Στόχος στο συγκεκριμένο παράδειγμα είναι να μεγιστοποιηθεί το coverage και παράλληλα να ελαχιστοποιηθεί το error rate, δίνοντας έμφαση πρώτα στην ελαχιστοποίηση του error rate. Αν το error rate ξεφύγει πάνω από το 10% τότε ίσως αυτό να είναι καταστροφικό. Στην πραγματικότητα δεν μπορεί να υπολογιστεί ακριβώς το R συνεπώς δεν μπορεί να υπολογστεί ακριβώς και το coverage και το error rate. Μπορεί ωστόσο με τη βοήθεια διάφορων προσπαθειών να προσεγγιστεί το error rate (και από αυτό ίσως και το coverage) για να αποκτηθεί έτσι μια καλύτερη εικόνα για την ποιότητα της μεθόδου.

Ένα άλλο θέμα που πρέπει να αναλυθεί είναι οι έννοιες των πλειάδων (tuples) και των προτύπων (patterns). Σε μια βάση δεδομένων μια πλειάδα είναι μια εγγραφή (μια σειρά) και αντιπροσωπεύει όλες τις πληροφορίες από κάθε πεδίο που σχετίζονται με αυτή την εγγραφή. Βρίσκοντας πρότυπα ανάμεσα στα δεδομένα, τα τελευταία γίνονται σχετικά προβλέψιμα και το επόμενο βήμα είναι η κατανόηση ποια από αυτά εμφανίζονται συχνά, ποια σχετίζονται μεταξύ τους ή ποια σχετίζονται με άλλα που ενδεχομένως δεν έχουν ακόμα ανακαλυφθεί. Τα πρότυπα πρέπει να είναι πολύ προσεκτικά ορισμένα ούτως ώστε να μην ταιριάζουν με άλλες πλειάδες που δεν χρειάζονται. Αυτό βέβαια στην πράξη δεν είναι πάντα τόσο εύκολο. Στο έργο αυτό χρησιμοποιήθηκε μια πολύ περιορισμένη κατηγορία κανονικών εκφράσεων. Πρέπει να ειπωθεί ότι για ένα σύνολο προτύπων P, ένα υψηλό coverage και ταυτόχρονα ένα χαμηλό error rate μπορεί να βοηθήσει να γίνει μια πολύ καλή προσέγγιση στο R βρίσκοντας απλά όλα τα ταιριάσματα όλων των προτύπων. Έτσι, αν υπάρχει ένα καλό σύνολο προτύπων μπορεί εύκολα να βρεθεί και ένα πολύ καλό σύνολο πλειάδων και η τεχνική να χαρακτηριστεί αρκετά πετυχημένη. Βέβαια είναι σημαντικό να μπορεί να εκτελεστεί και η αντίστροφη διαδικασία δηλαδή, δεδομένου ενός ολοκληρωμένου συνόλου πλειάδων να μπορούν να “αλιευθούν” σαφή σύνολα προτύπων. Κάτι τέτοιο θα μπορούσε να γίνει κοιτώντας όλες τις πλειάδες και μετά βρίσκοντας όλες τις

ομοιότητες μεταξύ των εμφανίσεων τους. Ο συνδυασμός των ανωτέρω είναι η βάση της τεχνικής DIPRE και μπορεί να της δώσει πολύ μεγάλη αξία.

Η τεχνική του DIPRE χρησιμοποιείται για την εξαγωγή σχέσεων και λειτουργεί με αλγόριθμο που έχει τα εξής πέντε βήματα:

1. Ξεκινάμε με ένα πολύ μικρό βήμα R', της σχέσης R στην οποία θέλουμε να καταλήξουμε.
2. Καταγράφουμε όλες τις εμφανίσεις πλειάδων του R' στο D. Μαζί με την κάθε πλειάδα διατηρούμε και το url και το surrounding text της.
3. (Ρουτίνα GenPatterns(O)) Δημιουργούμε πρότυπα για εμφανίσεις οι οποίες είναι σχετικά παρόμοιες. Χρειάζεται τα πρότυπα αυτά να έχουν υψηλό coverage και χαμηλό error rate, άρα πρέπει να είναι αρκετά συγκεκριμένα.
4. Αναζητούμε στη βάση δεδομένων πλειάδες που μοιάζουν με τα πρότυπα που έχουμε δημιουργήσει.
5. Εκτελούμε τα τρία προηγούμενα βήματα μέχρι το R' να προσεγγίσει ικανοποιητικά το R.

Αυτή η διαδικασία δεν είναι σίγουρο ότι δίνει πάντα τα επιθυμητά αποτελέσματα. Συγκεκριμένα, πολλά αποτελέσματα μπορεί να ξεφύγουν από αυτά που πρέπει να ληφθούν επειδή το πρότυπο δεν ήταν αρκετά συγκεκριμένο. Αυτό με τη σειρά του μπορεί να οδηγήσει και σε νέα ψεύτικα πρότυπα και αυτά να εμφανίσουν συνολικά ένα πλήθος από πλειάδες που δεν ταυτίζονται με το αρχικό λήμμα της αναζήτησης. Γι' αυτό το λόγο θα πρέπει τα πρότυπα να είναι προσεκτικά ορισμένα για να ελαχιστοποιούν τις πλειάδες που δεν χρειάζεται να εμφανίζονται. Ένα άλλο μέτρο ασφάλειας θα μπορούσε να είναι η αυστηροποίηση του κανόνα 4 στον παραπάνω αλγόριθμο. Δηλαδή για την εμφάνιση μιας πλειάδας ως σωστό αποτέλεσμα να απαιτείται η ομοιότητά της με παραπάνω από ένα πρότυπα.

Για το εν λόγω πείραμα, επιλέχθηκε να αναζητηθούν βιβλία με βάση το πρότυπο ζεύγος συγγραφέας-τίτλος μέσα από το διαδίκτυο. Πολλές ιστοσελίδες ταξινομούν τα βιβλία τους με βάση αυτό το πρότυπο, οπότε αναμένεται η αναζήτηση με βάση το DIPRE να έχει θετικά αποτελέσματα. Όμως, πριν ξεκινήσει να τρέχει ο αλγόριθμος του DIPRE είναι απαραίτητο να καθοριστούν τα πρότυπα στα οποία θα στηριχτεί για την αναζήτηση αφού τα πρότυπα παίζουν κομβικό ρόλο στην όλη διαδικασία. Στο πείραμα χρησιμοποιήθηκε ένα πολύ απλό πρότυπο που κρίθηκε ικανοποιητικό, χωρίς ωστόσο να είναι σίγουρο αν είναι ή όχι το κορυφαίο. Το πρότυπο αυτό αποτελείται από πέντε χαρακτηριστικά (order, urlprefix, prefix, middle, suffix) όπου το order είναι boolean τιμή, ενώ τα υπόλοιπα είναι strings. Για να ταιριάξει ένα αποτέλεσμα στο πρότυπο αυτό θα πρέπει:

- η τιμή order να είναι αληθινή,
- το url του εγγράφου να ταιριάζει με το urlprefix και
- το url του εγγράφου να περιέχει κείμενο που να ταιριάζει με την κανονική έκφραση *prefix, author, middle, title, suffix*.

Από τα παραπάνω καθίσταται σαφές ότι τελικά χρειάζονται συνολικά επτά χαρακτηριστικά για να αναπαρασταθεί το πρότυπο "συγγραφέας, τίτλος, order, url, prefix, middle, suffix". Το order έχει να κάνει με τη σειρά που εμφανίζονται στο κείμενο ο συγγραφέας και ο τίτλος. Το url είναι το url του εγγράφου που εμφανίστηκαν ο συγγραφέας και ο τίτλος. Το prefix περιλαμβάνει το κείμενο πριν τον συγγραφέα, το middle το κείμενο ανάμεσα στο συγγραφέα και τον τίτλο και το suffix το κείμενο μετά τον τίτλο (ομοίως και αν ο τίτλος με το συγγραφέα είχαν αντίθετες θέσεις).

Επίσης, ένα σημαντικό στοιχείο της τεχνικής DIPRE είναι η εξαγωγή νέων λιστών προτύπων δεδομένων ορισμένων λιστών βιβλίων. Η ακρίβεια των αποτελεσμάτων και της απόδοσης μπορεί να επηρεαστεί πολύ, ανάλογα με τον τρόπο που θα υλοποιηθεί αυτό το κομμάτι. Γι' αυτό είναι καλό να χρησιμοποιηθεί ένα μικρό σύνολο ευρετικών για τη δημιουργία κατάλληλων προτύπων. Ένας καλός αλγόριθμος (ρουτίνα GenOnePattern(O)) για τη κατασκευή τους θα μπορούσε να είναι ο ακόλουθος:

1. Επαληθεύουμε ότι το order και το middle όλων των εμφανίσεων είναι τα ίδια. Αν όχι, δεν είναι δυνατόν να δημιουργήσουμε ένα πρότυπο κατάλληλο για όλα αυτά. Θέτουμε outpattern.order το order και outpattern.middle το middle.
2. Βρίσκουμε το μεγαλύτερο prefix που ταιριάζει σε όλα τα urls. Θέτουμε outpattern.urlprefix σε αυτό το prefix.
3. Θέτουμε outpattern.prefix στο μεγαλύτερο suffix που ταιριάζει στις εμφανίσεις του prefix.
4. Θέτουμε outpattern.suffix στο μεγαλύτερο prefix που ταιριάζει στις εμφανίσεις του suffix.

Αρνητικό αυτού του αλγορίθμου είναι ότι ενδεχομένως τα πρότυπα που δημιουργούνται να είναι πολύ γενικά και έτσι ίσως δώσει λανθασμένα αποτελέσματα (στο πείραμά αυτό μη βιβλία).

Για την αντιμετώπιση αυτού του προβλήματος ορίστηκε η έννοια του specificity για ένα πρότυπο. Ο τύπος για τον υπολογισμό του για ένα οποιοδήποτε πρότυπο θα μπορούσε να είναι:

specificity(p) = |p.middle| |p.urlprefix| |p.prefix| |p.suffix|

Το specificity ελέγχει αν ένα πρότυπο ταιριάζει με υπερβολικά πολλά αποτελέσματα, κι αν ναι το απορρίπτει (υψηλό specificity). Επίσης, απορρίπτει ένα πρότυπο αν ταιριάζει μόνο με το αποτέλεσμα από το οποίο προήλθε (χαμηλό specificity). Πιο συγκεκριμένα, απαιτείται: specificity(p)n > t και n > 1, όπου n είναι ο αριθμός των βιβλίων με εμφανίσεις λόγω του προτύπου p και t είναι ένα ακέραιο νούμερο που ορίζεται σαν κατώτερο όριο (κατώφλι). Όλο αυτό πάντως προδίδει και έναν κίνδυνο αυτών των προσεγγίσεων που είναι η ανησυχία μήπως κινηθούν αρκετά μακριά από το R, αν δημιουργήσουν πρότυπα που αναγκάζουν σωστά αλλά και λάθος δεδομένα να συνυπάρχουν στην ίδια βάση δεδομένων.

Τώρα θα παρουσιαστεί ο αλγόριθμος GenPatterns(O) ο οποίος στηρίζεται στον αλγόριθμο GenOnePattern(O) και έχει δυο απλά βήματα:

1. Ομαδοποιούμε όλα τα αποτελέσματα ο σε O με βάση το order και το middle. Τα σύνολα που προκύπτουν θα είναι $O_1,\ldots,O_k$.
2. Για κάθε σύνολο O_i, p <- GenOnePattern(O_i). Αν το p πληροί τις προϋποθέσεις που έχουμε θέσει στο specificity, η έξοδος είναι p. Αλλιώς:
    a. Αν όλα τα ο στο O_i έχουν το ίδιο url τότε απορρίπτουμε το σύνολο O_i.
    b. Αλλιώς, χωρίζουμε τις εμφανίσεις ο στο O_i σε υποσύνολα με βάση τους χαρακτήρες στα url τους που είναι ένα παλιό p.urlprefix και επαναλαμβάνουμε τη διαδικασία του βήματος 2 για όλα αυτά τα υποσύνολα.

Αν και αυτός ο αλγόριθμος δεν θεωρείται ιδιαίτερα πολύπλοκος ή μοντέρνος, εντούτοις βάση των αποτελεσμάτων φαίνεται να λειτουργεί ικανοποιητικά.

Στην τεχνική DIPRE υπάρχουν δυο ακόμα σημαντικές λειτουργίες που πρέπει να αναφερθούν. Το πρώτο είναι η εύρεση εμφανίσεων βιβλίων από μια μεγάλη λίστα βιβλίων και το δεύτερο η εύρεση προτύπων από μια επίσης μεγάλη λίστα προτύπων που να ταιριάζουν στα βιβλία αυτά. Για την πρώτη λειτουργία, την εύρεση εμφανίσεων βιβλίων,

θα χρειαστεί να περαστούν όλα τα δεδομένα μέσα από δυο fgrep φίλτρα. Το πρώτο περνάει μόνο από γραμμές που έχουν έγκυρο συγγραφέα, ενώ το δεύτερο περνάει μόνο από γραμμές που έχουν έγκυρο τίτλο. Μόλις ολοκληρωθεί αυτή η διαδικασία ένα πρόγραμμα Python ελέγχει αν υπάρχουν αντίστοιχοι συγγραφείς και τίτλοι στη γραμμή και αν υπάρχουν, τους ταυτοποιεί και παράγει κατάλληλες εμφανίσεις ως έξοδο. Όσον αφορά τη δεύτερη λειτουργία, ένα πρόγραμμα σε Python αναλαμβάνει να μεταφράσει κάθε πρότυπο σε ένα ζευγάρι κανονικών εκφράσεων, ένα για το url και ένα για την πραγματική εμφάνιση. Αφού δοκιμαστεί κάθε url για να φανεί ποια πρότυπα εφαρμόζονται σ' αυτό, το πρόγραμμα Python δοκιμάζει κάθε γραμμή για τις σχετικές κανονικές εκφράσεις. Αυτός ο τρόπος επίλυσης της δεύτερης λειτουργίας δεν φαίνεται ο ιδανικότερος, αφού είναι πολύ αργός και ίσως στο μέλλον υπάρξει σημαντική βελτίωση αν χρησιμοποιηθούν κάποια εργαλεία όπως το Flex ή η rex C βιβλιοθήκη. Πάντως, σε αυτή τη φάση η δημιουργία προτύπων από εμφανίσεις δεν θέτει μεγάλο ζήτημα στην απόδοση αφού υπάρχουν μόνο μερικές χιλιάδες εμφανίσεις. Μελλοντικά ίσως το πρόβλημα να οξυνθεί αν βρεθούν περισσότερα πρότυπα και άρα περισσότερα αποτελέσματα.

Γενικά μέχρι αυτή τη στιγμή έχουν πραγματοποιηθεί πάρα πολλά πειράματα με βάση τη συγκεκριμένη τεχνική και αν και σε πολύ περιορισμένο χρόνο, τα περισσότερα έχουν δώσει πολύ θετικά αποτελέσματα. Σε ένα πείραμα που έγινε υπό την επίβλεψη του Sergey Brin αποφασίστηκε να χρησιμοποιηθεί ένα αποθετήριο 24 εκατομμυρίων ιστοσελίδων που αντιστοιχούν σε περίπου 147 gigabytes. Αυτές οι ιστοσελίδες αποτελούν κομμάτι της μηχανής αναζήτησης της Google καθώς και άλλων οργανισμών. Το αποθετήριο έχει τόσο πολλά δεδομένα που ακόμα και μια απλή αναζήτηση σε όλα αυτά μπορεί να κοστίσει πολύ χρόνο ακόμα κι αν δε γίνει σημαντική επεξεργασία. Έτσι, η λύση είναι να γίνουν επαναληπτικά περάσματα σε ορισμένα υποσύνολα του αποθετηρίου.

Στην αρχή το πείραμα ξεκίνησε με μόνο πέντε βιβλία τα οποία έδωσαν μέσω της τεχνικής DIPRE 199 εμφανίσεις άλλων βιβλίων και δημιούργησαν 3 πρότυπα. Όσο προχωρούσε εμφανίζονταν νέα βιβλία και νέα πρότυπα. Παρατηρήθηκε πάντως ότι τα περισσότερα βιβλία που εμφανίστηκαν ήταν επιστημονικής φαντασίας. Για τα αποτελέσματα του πειράματος πρέπει να ειπωθεί ότι από 3972 εμφανίσεις βιβλίων παράχθηκαν 105 πρότυπα, 24 από τα οποία είχαν url prefixes που δεν ήταν ολοκληρωμένα url, ενώ ένα πέρασμα από δυο εκατομμύρια url παρήγαγε 9369 μοναδικά ζεύγη συγγραφέας-τίτλος. Βέβαια το αρνητικό ήταν ότι μέσα στα αποτελέσματα υπήρχαν και κάποια λίγα που δεν ήταν βιβλία. Η απομάκρυνσή τους ήταν η μόνη χειροκίνητη παρέμβαση στην όλη διαδικασία. Για την τελική επανάληψη επιλέχθηκε ένα υποσύνολο του αποθετηρίου που περιείχε τα βιβλία εργασίας (περίπου 156.000 έγγραφα). Από αυτά παράχθηκαν 15.257 βιβλία με πολύ λίγα λανθασμένα δεδομένα. Τελικά μετά από πολλές επαναλήψεις και λίγη χειροκίνητη παρέμβαση για να αφαιρεθούν ορισμένα δεδομένα που δεν θα έπρεπε να ανακτηθούν δημιουργήθηκε ένα αποθετήριο με πάνω από 15.000 διαφορετικά βιβλία.

Σχετικά με την ποιότητα των αποτελεσμάτων, φάνηκε ότι ήταν πάρα πολύ καλή αφού από τα 20 τυχαία αποτελέσματα που πάρθηκαν σαν δείγμα τα 19 ήταν όντως βιβλία, ενώ το αποτέλεσμα που δεν ήταν βιβλίο ήταν άρθρο. Μεγάλη έκπληξη ήταν το γεγονός ότι αρκετά βιβλία δεν αναφέρονταν σε όλες τις πηγές από τις οποίες έγινε η άντληση των δεδομένων όπως για παράδειγμα η Amazon που ενώ ισχυρίζεται ότι έχει μεγάλους καταλόγους με εκατομμύρια βιβλία εντούτοις δεν είχε πέντε από τα είκοσι βιβλία του παραπάνω δείγματος. Ορισμένα μικροπροβλήματα στα δεδομένα επειδή ο συγγραφέας ή ο τίτλος του ίδιου βιβλίου είχαν καταχωρηθεί παραπάνω από μια φορές με διαφορετικό τρόπο ξεπεράστηκαν χωρίς να χαθεί πολύς χρόνος.

Γενικά, μπορεί να ειπωθεί ότι η μέθοδος DIPRE αποδείχθηκε πολύ αξιόλογο εργαλείο στην εξαγωγή δομημένων δεδομένων από τον παγκόσμιο ιστό. Αυτή η μέθοδος μπορεί να ακολουθηθεί παρόμοια και για άλλους τομείς όπως εστιατόρια, ταινίες, μουσική κτλ ενώ δεδομένου ότι το πείραμα βρίσκεται ακόμα σε πρώιμο στάδιο και είναι σε εξέλιξη, στο μέλλον ίσως φανεί χρήσιμο σε ακόμα περισσότερους τομείς και ίσως και με ακόμη ακριβέστερα αποτελέσματα. Υπάχουν όμως πολλές προκλήσεις για την αναβάθμιση της μεθόδου. Μια από αυτές είναι η σάρωση μεγάλου αριθμού προτύπων και πλειάδων σε τεράστια αποθετήρια. Ίσως κάποιες βελτιώσεις στους αλγορίθμους που υλοποιούν τις συγκεκριμένες διαδικασίες να λύσουν αργότερα αυτό το πρόβλημα. Το πιο σημαντικό πρόβλημα όμως είναι να αποτραπεί η οποιαδήποτε παρέκκλιση της μεθόδου από τον αρχικό στόχο ειδικά όσο προχωράει η εξόρυξη των δεδομένων. Γενικά, η διατήρηση των αποτελεσμάτων υπό έλεγχο καθώς η διαδικασία επεκτείνεται είναι σημαντική και γι' αυτό υπάρχουν αρκετές δυνατότητες. Μια από αυτές είναι να επαναπροσδιοριστεί το $M_D(P)$ ώστε να απαιτεί πολλαπλά πρότυπα να ταιριάζουν με μια πλειάδα. Μια πιο ακραία ίσως λύση είναι να ανατεθεί ένα βάρος σε κάθε πλειάδα και πρότυπο. Πιο συγκεκριμένα, σε μια πλειάδα ανατίθεται ένα βάρος με βάση τα βάρη των προτύπων που ταιριάζει και σε ένα δημιουργημένο πρότυπο ανατίθεται ένα βάρος με βάση τα βάρη των πλειάδων που ταιριάζουν σε αυτό. Μέσω υπολογισμών με τα βάρη των πλειάδων και των προτύπων ίσως προκύψουν καλύτερα αποτελέσματα.

Στο τέλος ο Brin υποστήριξε ότι ένα από τα πιο εκπληκτικά αποτελέσματα του εγχειρήματος αυτού ήταν η εύρεση βιβλίων που δεν περιλαμβάνονταν σε μεγάλες διαδικτυακές πηγές βιβλίων. Αν η λίστα βιβλίων μπορούσε να επεκταθεί και αν σχεδόν όλα τα βιβλία μπορούσαν να εκμαιευθούν από μια αποιαδήποτε βάση δεδομένων, τότε θα μπορούσε να δημιουργηθεί μια νέα λίστα βιβλίων μεγαλύτερη από οποιαδήποτε άλλη στο διαδίκτυο. Μια τέτοια αλλαγή στη ροή των πληροφοριών θα μπορούσε να έχει σημαντικά κοινωνικά οφέλη.

Το πείραμα του Praveena Mettu

Εκτός βέβαια από το πείραμα του Brin που αναφέρθηκε παραπάνω, υπήρξαν και άλλα πειράματα πάνω στην τεχνική DIPRE με σκοπό την περαιτέρω εξέλιξή της. Ένα από αυτά είναι αυτό που πραγματοποίησε ο Praveena Mettu ο οποίος θέλησε να επεκτείνει το σύνολο δεδομένων στο οποίο μπορεί να εκτελεστεί το DIPRE και να αξιολογήσει την αποτελεσματικότητά του σε ένα πιο ελεύθερο κείμενο φυσικής γλώσσας στο διαδίκτυο. Οι πρώτες δυο βασικές επιλογές που έπρεπε να κάνει ήταν η μηχανή αναζήτησης από την οποία θα έπαιρνε πληροφορίες και η γλώσσα προγραμματισμού που θα χρησιμοποιούσε για να φτιάξει μια εφαρμογή σχετική με το πείραμα. Επέλεξε μηχανή αναζήτησης την Google AJAX Search API επειδή είναι πολύ δημοφιλής στην αγορά και έχει μελετηθεί εκτενώς και γλώσσα προγραμματισμού την Java αφού είναι συγκριτικά πιο γρήγορη από τις C# και Perl που ήταν οι άλλες υποψήφιες.

Στο αρχικό στάδιο του πειράματος χρειάζονται δυο παράμετροι που δίνονται από το χρήστη. Θα πρέπει βέβαια να καθοριστεί και σε ποιον ιστότοπο θα περιοριστεί η έρευνα. Ο Praveena Mettu αποφάσισε ότι για τα πειράματά του αυτός ο ιστότοπος θα είναι η www.wikipedia.org, αλλά ο κάθε χρήστης θα μπορεί μελλοντικά να επιλέξει όποιο site εκείνος επιθυμεί. Είναι σημαντικό πάντως, ειδικά το site, να καθοριστεί από τον κάθε μελλοντικό χρήστη γιατί αλλιώς η αναζήτηση θα είναι παρόμοια με μια συνηθισμένη καθημερινή αναζήτηση στο Google.

Το επόμενο βήμα είναι η δημιουργία του string που θα βοηθήσει το Google AJAX Search API να δώσει αποτελέσματα. Αυτό γίνεται με τον παρακάτω αλγόριθμο:

1. Αντικατάσταση όλων των διαστημάτων στις παραμέτρους με +.
2. Αν το SearchSite είναι άδειο, αντικατάσταση του με www.google.com.
3. SearchString = “%22 “ + parameter1 + “%22” + “+” + “%22 ” + parameter2 + “%22” + “ site:” + searchsite

Έτσι δημιουργείται από το searchstring το url και πηγαίνει στην αναζήτηση API. Τα αποτελέσματα API έχουν τη μορφή JSON αντικειμένου και μέσα εκεί υπάρχουν οκτώ εμφανίσεις πλειάδων αποτελεσμάτων. Κάθε αποτέλεσμα string χωρίζεται σε “prefix”, “middle term” και “suffix” ανάλογα με τις δυο παραμέτρους. Η ροή εισόδου διαβάζεται γραμμή προς γραμμή και προσαρτάται στο String Builder. Στη συνέχεια μετατρέπεται σε ένα string “response” και τα δεδομένα τοποθετούνται σε ένα αντικείμενο JSON. Μετά από κατάλληλη επεξεργασία στην κάθε οκτάδα αποτελεσμάτων και αποθήκευση λεπτομερών πληροφοριών για το κάθε αποτέλεσμα ξεχωριστά η επεξεργασία περνά στην επόμενη οκτάδα αποτελεσμάτων με το κατάλληλο string. Υπάρχουν συνολικά 64 πλειάδες αποτελεσμάτων.

Όμως, τα περισσότερα αποτελέσματα δεν είναι πάντα συμβατά με την αναζήτηση που γίνεται και γι’ αυτό το πείραμα κινδυνεύει να αποτύχει. Έτσι, η εστίαση γίνεται στα έγκυρα αποτελέσματα τα οποία αποθηκεύονται και αναλύονται με βάση το prefix, το middle term και το suffix. Το νέο string αναζήτησης θα είναι:

Search query = Longest_prefix + "+*" + most_repeated_middle_term + "*+" + Longest_suffix + " site:” + searchsite

Μεταβιβάζοντας αυτό το string στο API αναζήτησης, τα αποτελέσματα νέων πλειάδων μπορούν ξανά να φτάσουν τα 64. Και πάλι, ωστόσο, πολύ λίγες από αυτές τις πλειάδες είναι έγκυρες και η όλη διαδικασία θα πρέπει να ακολουθηθεί ξανά από την αρχή. Τέτοιες επαναλήψεις μπορούν να συνεχιστούν μέχρι να σταματήσουν να επιστρέφονται έγκυρα αποτελέσματα. Η απουσία ενός αλγορίθμου φιλτραρίσματος για τα έγκυρα δεδομένα καθώς και η αδυναμία χειροκίνητης επεξεργασίας αυτών είναι δυο προβλήματα που ταλανίζουν ακόμα αρκετούς αλγορίθμους semi-supervised. Τα αποτελέσματα εμφανίζονται πολύ εύκολα στην οθόνη αφού για να γίνει αυτό απαιτείται μονάχα μια απλή εντολή System.out.println.

Μερικές βασικές διαπιστώσεις που έγιναν μετά από αρκετές εφαρμογές του πειράματος του Praveena Mettu σχετικά με τα αποτελέσματα του πειράματος αυτού είναι ότι συνήθως μετά την τέταρτη επανάληψη τα αποτελέσματα έτειναν να ξεφεύγουν αρκετά από το αρχικό ερώτημα της αναζήτησης. Προσαρμογές για τεχνικές φιλτραρίσματος ή χειρωνακτικές παρεμβάσεις για τη σχετικότητα των αποτελεσμάτων με το ερώτημα της αναζήτησης κρίνονται απαραίτητες και υπάρχουν πολλοί ερευνητές που έχουν προτείνει παρόμοιες βελτιώσεις όπως και ο Brin στην τεχνική DIPRE.

Η κύρια διαφορά ανάμεσα σε αυτό το πείραμα και σε εκείνο που πραγματοποίησε ο Brin είναι ότι το πείραμα του Brin εστίασε κυρίως στην εξαγωγή συγγραφέων και βιβλίων από το διαδίκτυο, ενώ αυτό το πείραμα προσπάθησε να εφαρμόσει τις ίδιες αρχές σε ένα ευρύτερο σύνολο παίρνοντας αποτελέσματα για οποιεσδήποτε δυο παραμέτρους. Πάντως παρότι τα πειράματα του Praveena Meetu έδειξαν πολύ θετικά στοιχεία εντούτοις υπάρχει ακόμα αρκετή δυνατότητα για εξέλιξη. Μερικές προτάσεις βελτίωσης έχουν να κάνουν με την εφαρμογή παρόμοιων με το DIPRE τεχνικών στα πειράματα αυτά αλλά και με την κατάργηση κάποιων περιορισμών των μηχανών αναζήτησης που χρησιμοποιούνται για αυτά.

Η τεχνική Snowball

Μια ακόμη τεχνική που αναπτύσσει βασικά στοιχεία της μεθόδου DIPRE, είναι η τεχνική Snowball. Πιο συγκεκριμένα το snowball είναι μια προσπάθεια δημιουργίας προτύπων και εξαγωγής πλειάδων από έγγραφα κειμένου. Επίσης, εισάγει σε μια στρατηγική αξιολόγησης της ποιότητας των προτύπων και των πλειάδων που δημιουργούνται σε κάθε επανάληψη της διαδικασίας εξαγωγής. Στο κομμάτι της εξαγωγής προτύπων και πλειάδων το DIPRE και η snowball φαίνεται να μοιάζουν αρκετά. Η διαφορά που προσφέρει η snowball ως βελτίωση του DIPRE είναι ότι τα πρότυπα της περιλαμβάνουν ετικέτες ονομαστικών οντοτήτων. Αυτό εν ολίγοις σημαίνει ότι λέξεις που συχνά εμφανίζονται κοντά μεταξύ τους μέσα σε ένα κείμενο και έχουν νοηματική σχέση μπορούν να ενωθούν και να φτιάξουν ένα πρότυπο. Έτσι, παράγονται πρότυπα με ευέλικτο τρόπο τα οποία παρουσιάζουν και υψηλό coverage.

Γενικά, ένα πρότυπο snowball είναι μια πλειάδα πέντε στοιχείων: <left, tag1, middle, tag2, right>, όπου το tag1 και το tag2 είναι ετικέτες ονομαστικών οντοτήτων και το left, middle, right είναι διανύσματα που συσχετίζουν όρους με βάρη. Για να ταιριάξει ένα τμήμα κειμένου με την αναπαράσταση της πλειάδας πέντε στοιχείων θα χρειαστεί και η παρουσία μιας ακόμα αντίστοιχης ισοδύναμης πλειάδας πέντε στοιχείων. Μετά από κατάλληλη επεξεργασία ο αλγόριθμος snowball παίρνει όλες τις απαραίτητες πληροφορίες (κυρίως από τα διανύσματα) για τη δημιουργία ενός προτύπου. Πάντως, από μέχρι τώρα έρευνες φαίνεται ότι το στοιχείο middle είναι το πιο ενδεικτικό της σχέσης μεταξύ των στοιχείων της πλειάδας. Για να δημιουργηθούν πρότυπα το snowball χρησιμοποιεί μερικούς ακόμα αλγορίθμους ούτως ώστε να ομαδοποιηθούν οι εμφανίσεις των γνωστών πλειάδων σε έγγραφα, εφόσον τα συμφραζόμενά τους είναι παρόμοια.

Μετά τη δημιουργία προτύπων το Snowball εστιάζει στην ανακάλυψη νέων πλειάδων. Από τις ετικέτες ονομαστικών οντοτήτων και τις πλειάδες πέντε στοιχείων επιλέγονται ορισμένες νέες πλειάδες με βάση τον αριθμό των αρχικών πλειάδων που χρησιμοποιήθηκαν για τη δημιουργία τους. Η δημιουργία καλών προτύπων είναι απαιτητική διαδικασία αλλά απαραίτητη μιας και είναι βασικό μέρος του συστήματος και υπεύθυνη για την βελτίωση της τεχνικής DIPRE (που είναι και ο αρχικός στόχος του snowball). Γι’ αυτό και η τεχνική snowball χρησιμοποιεί μια πληθώρα αλγορίθμων για να ελέγχει την αποδοτικότητα των προτύπων που χρησιμοποιεί.

Ορισμένα πειράματα που έγιναν πάνω στην τεχνική snowball είχαν αρκετά ενθαρρυντικά αποτελέσματα. Παρατηρήθηκε ότι η ακρίβειά του δε χανόταν ακόμα και σε αρκετά μεγάλο αριθμό επαναλήψεων, ενώ σε σχέση με το DIPRE παρατηρήθηκε ότι η ακρίβεια του τελευταίου είναι μεγαλύτερη μόνο στις πρώτες επαναλήψεις, αφού δεν είχε τρόπο να αποτρέψει τις αναξιόπιστες πλειάδες που χρησιμοποιούσε για την επόμενη επανάληψη του. Πάντως και το snowball εμφάνισε κάποια προβλήματα αφού σε κάποια πειράματα δεν κατάφερε να προχωρήσει πέρα από τη δεύτερη επανάληψη. Συμπεραίνοντας, το snowball έχει υψηλότερο recall (εκφράζεται με ένα κλάσμα με αριθμητή τον αριθμό των σωστών αποτελεσμάτων και παρονομαστή τον αριθμό των αποτελεσμάτων που θα έπρεπε να έχουν επιστραφεί) από το DIPRE, αλλά η ακρίβειά τους είναι σχετικά συγκρίσιμη.

Γενικά για ιστογράμματα

Αφού αναλύθηκαν στα προηγούμενα διάφορες καινοτόμες τεχνικές και πειράματα για την εξόρυξη δεδομένων από το διαδίκτυο κρίνεται σκόπιμο να αναφερθούν και ορισμένοι τρόποι αναπαράστασης αυτών των δεδομένων για την καλύτερη κατανόηση και ομαδοποίησή τους. Υπάρχουν πολλοί τρόποι για να γίνει κάτι τέτοιο. Ένας από αυτούς είναι και τα ιστογράμματα τα οποία είναι γραφική απεικόνιση στατιστικών συχνοτήτων περιοχών τιμών ενός μεγέθους και χρησιμοποιούνται, όπως αναφέρθηκε και στην εισαγωγή, για να περιγράψουν εκτενή δεδομένα με περιεκτικό και κατανοητό τρόπο. Η χρήση τους είναι ευρεία αφού μπορούν να χρησιμοποιηθούν από την αναπαράσταση των δεδομένων μιας εταιρείας μέχρι τη δημιουργία μιας εικόνας. Στις βάσεις δεδομένων τα ιστογράμματα χρησιμοποιούνται για να συνοψίσουν δεδομένα και να παρέχουν εκτιμήσεις. Το κυριότερο πλεονέκτημά τους σε σχέση με άλλους τρόπους αναπαράστασης στις βάσεις δεδομένων είναι η αρκετά μεγαλύτερή τους ακρίβεια. Δεν παρουσιάζονται στους χρήστες ή δεν εμφανίζονται οπτικά, συνεπώς υπάρχει ένα ευρύ φάσμα επιλογών για την κατασκευή τους. Υπάρχουν προγραμματιστικοί αλγόριθμοι οι οποίοι δημιουργούν ιστογράμματα, αλλά δουλεύουν μόνο σε τετραγωνικό χρόνο και γραμμικό χώρο. Εδώ θα μελετηθεί η γραμμικού χρόνου κατασκευή της 1+ε προσέγγισης optimal ιστογραμμάτων για πολυλογαριθμικό χώρο. Τα αποτελέσματα εκτείνονται γενικότερα σε ροές δεδομένων και γενικεύουν την 1+ε προσέγγιση για αρκετά προβλήματα που απαιτούν διαχωρισμό του ευρετηρίου σε διαστήματα, με ελάχιστες παραδοχές.

Αν και η κατασκευή ιστογραμμάτων σε μεγάλες βάσεις δεδομένων θεωρείται μια κοστοβόρα διαδικασία, εντούτοις πολλά συστήματα σχεσιακών βάσεων χρησιμοποιούν κάποια μορφή ιστογραμμάτων. Τα ιστογράμματα παρέχουν τη δυνατότητα για λήψη πληροφοριών πάνω σε γενικά στατιστικά για ένα ικανό κομμάτι του πληθυσμού. Συνήθως δουλεύουν σε αριθμητικούς τομείς και απαντούν σε ερωτήματα κόστους, προσεγγιστικών απαντήσεων κτλ. Τυχόν σφάλματα στις προσεγγίσεις τους επηρεάζουν άμεσα ή μεταβατικά πολλές εκτιμήσεις των βάσεων δεδομένων. Αποδεικνύεται ότι τα σειριακά ιστογράμματα είναι τα ιδανικότερα για την ελαχιστοποίηση τέτοιου είδους λαθών. Γενικότερα όμως, επειδή το αναμενόμενο σφάλμα είναι σχεδόν πάντα μηδενικό, η εστίαση γίνεται στις ακραίες περιπτώσεις σφαλμάτων, αφού η μείωση του αναμενόμενου σφάλματος δεν έχει νόημα.

Έτσι, η βελτιστοποίηση ερωτημάτων είναι ένα βασικό πρόβλημα και για τα συστήματα βάσεων δεδομένων. Ένα τέτοιο ερώτημα μεταφράζεται σε ένα δέντρο operators βάσεων δεδομένων οι οποίοι πρέπει να τρέξουν και να παράξουν την απάντηση. Η εκτέλεση κάθε operator συνεπάγεται κόστος και η βελτιστοποίηση του ερωτήματος έχει να κάνει με το σχηματισμό κατάλληλου σχεδίου εκτέλεσης με το λιγότερο δυνατό κόστος. Ο ρόλος των ιστογραμμάτων έχει να κάνει με την εκτίμηση του κόστους των operators βάσεων δεδομένων για ένα ερώτημα.

Τρεις βασικοί operators είναι το select, το join και το group by. Το select συμβολίζεται με το σύμβολο σ και χρησιμοποιείται για την επιλογή ενός υποσυνόλου πλειάδων μιας βάσης δεδομένων σύμφωνα με μια δεδομένη συνθήκη επιλογής. Αντιστοιχεί συνήθως σε μια ισότητα ή σε ένα εύρος τιμών που πρέπει να εκτελεστεί στη βάση. Υπάρχουν διάφορες προτάσεις σχετικές με την κατασκευή καλών ιστογραμμάτων για τέτοιες λειτουργίες. Το αποτέλεσμα μιας τέτοιας εκτίμησης μέσω ενός ιστογράμματος αντιπροσωπεύει τον κατά προσέγγιση αριθμό των πλειάδων της βάσης δεδομένων που ικανοποιούν την αναζήτηση (selectivity estimate). Το join είναι μια δυαδική λειτουργία που επιτρέπει το συνδυασμό

πολλών δεδομένων από μια βάση δεδομένων. Έχει διάφορες υποκατηγορίες όπως Inner Join, Theta Join, Left Join κτλ που η κάθε μια εξειδικεύεται σε κάποιο διαφορετικό συνδυασμό δεδομένων. Οι join operators είναι πολύ σημαντικοί για μια βάση δεδομένων αν και είναι δαπανηρές λειτουργίες. Και εδώ υπάρχουν αξιόλογες προτάσεις για την κατασκευή καλών ιστογραμμάτων που εκτιμούν το κόστος των join λειτουργιών. Το group by υπολογίζει ταυτόχρονα πολλαπλά συγκεντρωτικά αποτελέσματα. Στα θετικά του συγκαταλέγεται το γεγονός ότι περιορίζει τον αριθμό των ταυτόχρονων προσβάσεων στο σύστημα βάσης δεδομένων αλλά και το ότι περιορίζει τον αριθμό των διαφορετικών υπολογισμών που μπορούν να εκτελεστούν ταυτόχρονα με μια σάρωση πάνω στα δεδομένα.

Το select, το join και το group by θεωρούνται γενικά πολύ αξιόπιστοι operators που προσφέρουν αρκετά μεγάλη ακρίβεια στα αποτελέσματα τους. Όμως κάτι ανάλογο δεν ισχύει για όλους τους operators. Επειδή τα αποτελέσματα των ερωτημάτων υπολογίζονται συνήθως χρησιμοποιώντας μια ποικιλία στατιστικών της βάσης, πολλές φορές αυτά τα στοιχεία απλώς προσεγγίζουν την κατανομή των τιμών των δεδομένων δείχνοντας μια ανακριβή εικόνα του πραγματικού τους περιεχομένου. Αυτό το αποτέλεσμα σε συνδυασμό με την ολοένα αυξανόμενη πολυπλοκότητα των ερωτημάτων, καταδεικνύει την κρίσιμη σημασία των έγκυρων operators.

Τα ιστογράμματα χρησιμοποιούνται σε συστήματα όπου χρειάζεται προσεγγιστική απάντηση ερωτημάτων, όπου δηλαδή ο κύριος στόχος είναι η παροχή γρήγορης αλλά προσεγγιστικής απάντησης. Η βασική αρχή πίσω από το σχεδιασμό τέτοιων συστημάτων είναι ότι για πολύ μεγάλα σύνολα δεδομένων στα οποία η εκτέλεση πολύπλοκων ερωτημάτων είναι χρονοβόρα, είναι προτιμότερη μια γρήγορη προσεγγιστική απάντηση. Αυτό είναι πολύ βοηθητικό για γρήγορες και προσεγγιστικές αναλύσεις μεγάλων όγκων δεδομένων. Μια επιπλέον εφαρμογή των ιστογραμμάτων είναι η εξόρυξη δεδομένων χρόνου, αφού αποτελούν έναν εναλλακτικό τρόπο συμπίεσης πληροφοριών χρονοσειρών. Η προσέγγιση των ιστογραμμάτων είναι επίσης χρήσιμη στην απλοποίηση καμπυλών και ειδικότερα στις βελτιώσεις μιας απεικόνισης κατανομής. Αυτό φαίνεται να μοιάζει αρκετά με τα ιστογράμματα haar wavelets τα οποία θεωρούνται από τα πιο απλά ιστογράμματα και χρησιμοποιούνται για μαθηματικούς υπολογισμούς.

Μια έννοια που πρέπει να αναλυθεί επειδή θα χρησιμοποιηθεί αργότερα αρκετά είναι η έννοια του data stream (ροή δεδομένων). Ένα data stream είναι μια διατεταγμένη ακολουθία σημείων που μπορούν να διαβαστούν μόνο μια ή λίγες φορές. Τα πολλαπλά περάσματα είναι πρακτικά απαγορευτικά λόγω του μεγάλου όγκου δεδομένων. Τυπικά ένα data stream είναι μια ακολουθία από σημεία $x_1,\ldots,x_i,\ldots,x_n$. Η εκτέλεση ενός αλγορίθμου που λειτουργεί σε data streams μετριέται από τον αριθμό των περασμάτων που ο αλγόριθμος θα πρέπει να αλλάξει το stream, όταν περιορίζεται από τους όρους της διαθέσιμης μνήμης. Τέτοιοι αλγόριθμοι είναι ιδιαίτερα σημαντικοί σε πολλές εφαρμογές δικτύων. Πολλές πηγές δεδομένων που ανήκουν σε συστατικά στοιχεία κρίσιμης σημασίας για το δίκτυο παράγουν τεράστιες ποσότητες δεδομένων σε ένα stream, οι οποίες θέλουν online ανάλυση και ερώτηση. Ένα από τα πρώτα αποτελέσματα στα data streams ήταν το αποτέλεσμα της μελέτης της απαίτησης χώρου επιλογής και ταξινόμησης ως συνάρτηση του αριθμού των περασμάτων στα δεδομένα. Όταν το μοντέλο επισημοποιήθηκε δόθηκαν αρκετοί αλγόριθμοι και αποτελέσματα πολυπλοκότητας σχετικά με γραφοθεωρητικά προβλήματα και εφαρμογές τους. Σκοπός είναι η συνοπτική καταγραφή των δεδομένων εισόδου σε ιστογράμματα, έτσι ώστε οι χαρακτηριστικές τιμές να κατηγοριοποιηθούν εγκαίρως.

Τα περισσότερα από αυτά τα προβλήματα ιστογραμμάτων μπορούν να λυθούν offline με τη χρήση δυναμικού προγραμματισμού. Τα πιο γνωστά αποτελέσματα απαιτούν τετραγωνικό χρόνο και γραμμικό χώρο για να παρέχουν την τέλεια λύση. Η εξοικονόμηση χώρου αποτελεί μια σημαντική πτυχή στην κατασκευή ιστογραμμάτων, αφού τα αντικείμενα που πρέπει να διαχειριστούν είναι πολλά. Κάτι τέτοιο πάντως κάνει απαραίτητη και την απαίτηση για μια προσεγγιστική περιγραφή.

V-optimal ιστόγραμμα

Στο σημείο αυτό πρέπει να ειπωθεί πως όλα τα ιστογράμματα για τα οποία θα γίνει λόγος είναι σειριακά. Ένας τύπος ιστογράμματος που δουλεύει πολύ καλά πάνω σε βάσεις δεδομένων ονομάζεται V-optimal ιστόγραμμα ή απλά V-ιστόγραμμα. Γενικά ο όρος V-optimality αναφέρεται ως ένας κανόνας χωρίσματος που δηλώνει ότι τα όρια ενός bucket πρέπει να τοποθετηθούν έτσι ώστε να ελαχιστοποιηθεί η αθροιστική σταθμισμένη διακύμανση των buckets. Η εφαρμογή αυτού του κανόνα είναι ένα πολύπλοκο θέμα και η κατασκευή των αντίστοιχων ιστογραμμάτων είναι επίσης μια περίπλοκη διαδικασία. Με βάση τα παραπάνω και το V-ιστόγραμμα προσπαθεί να επιτύχει την ελαχιστοποίηση της σταθμισμένης διακύμανσης και ορίζεται ως:

$$W=\sum_{j=1}^{J} n_j VAR_j \ ,$$

όπου το ιστόγραμμα έχει συνολικά J buckets, n_j είναι ο αριθμός των αντικειμένων που περιέχονται στο j bucket και VAR_j είναι η διακύμανση μεταξύ των τιμών που σχετίζονται με τα αντικείμενα στο j bucket. Κατά μια έννοια στόχος είναι να προσεγγιστεί μια καμπύλη από μια J-βημάτων συνάρτηση και το σφάλμα να είναι το ελάχιστο κατάλληλο τετραγωνικό.

Υπάρχουν και άλλα παρόμοια ιστογράμματα με το V-optimal που αναφέρθηκε παραπάνω τα οποία αξίζει να αναφερθούν. Ορισμένα από αυτά είναι το Equi-sum, το MaxDiff και το Compressed ιστόγραμμα. Στο equi-sum το άθροισμα των πηγαίων τιμών σε κάθε bucket είναι περίπου το ίδιο. Στο maxdiff υπάρχει ένα bucket όριο μεταξύ δυο πηγαίων παραμετρικών τιμών που είναι γειτονικές εάν η διαφορά μεταξύ αυτών των τιμών είναι μια από τις β-1 μεγαλύτερες διαφορές. Στο compressed οι h υψηλότερες πηγαίες τιμές αποθηκεύονται χωριστά σε h buckets με μόνο ένα αντικείμενο το κάθε bucket: τα υπόλοιπα χωρίζονται όπως σε ένα equi-sum ιστόγραμμα. Έχει επιλεγεί το h να είναι το νούμερο των πηγαίων τιμών που:

a) υπερβαίνει το άθροισμα όλων των πηγαίων τιμών διαιρεμένων με το νούμερο των buckets και
b) μπορεί να μπει σε ένα ιστόγραμμα με β buckets.

Όλα αυτά τα ιστογράμματα είναι αρκετά ευέλικτα, αφού κάνοντας διαφορετικές επιλογές στα χαρακτηριστικά τους μπορεί κάποιος να πάρει νέα ιστογράμματα.

Επιστρέφοντας πάλι στο V-optimal ιστόγραμμα, υπάρχει ένας ευθύς δυναμικός προγραμματιστικός αλγόριθμος που δόθηκε για την κατασκευή του. Αυτός ο αλγόριθμος τρέχει σε $O(n^2k)$ χρόνο και απαιτεί $O(n)$ χώρο. Το κεντρικό μέρος της δυναμικής προγραμματιστικής προσέγγισης είναι η εξίσωση που ακολουθεί:

$$OPT[k,n] = \min\{OPT[k-1,x] + VAR[(x+1)\ldots n]\}, \text{ για } x<n \qquad (1)$$

Για αυτή την εξίσωση το OPT[p,q] συμβολίζει το ελάχιστο κόστος αναπαράστασης του συνόλου τιμών καταταγμένες από 1...q από ένα ιστόγραμμα που έχει p levels. Το VAR[a…b] αναπαριστά τη διακύμανση του συνόλου τιμών που ταξινομούνται κατά a…b. Ο δείκτης x μπορεί να πάρει μέγιστο n τιμές κι έτσι σε κάθε είσοδο αρκεί O(n) χρόνος για να υπολογιστεί. Για O(nk) εισόδους, ο συνολικός χρόνος που απαιτείται είναι $O(n^2k)$ και ο συνολικός χώρος O(n).

Η πρώτη διαπίστωση που μπορεί να γίνει πάνω στην εξίσωση (1) είναι ότι:
Αν a <= x <= b τότε VAR[a…n] >= VAR[x…n] >= VAR [b…n]
Και αφού κάθε λύση για το σύνολο δεικτών 1...b δίνει μια καλή λύση για το σύνολο δεικτών 1…x για κάθε x <= b, η δεύτερη διαπίστωση που μπορεί να γίνει είναι ότι:
Αν a <= x <= b τότε OPT[p,a] <= OPT[p,x] <= OPT[p,b]
Έτσι γίνεται κατανοητό ότι το επόμενο βήμα είναι να βρούμε το ελάχιστο του αθροίσματος δυο συναρτήσεων, μια από τις οποίες είναι μη αυξανόμενη και η άλλη είναι μη μειούμενη (OPT[p,x] όσο το x αυξάνεται). Μια αρχική ιδέα θα ήταν να χρησιμοποιηθεί η μονοτονία για να μειωθεί το ψάξιμο λογαριθμικών όρων, αλλά αυτή η λύση δεν είναι καλή επειδή απαιτεί πολύ χρόνο. Αυτό που προτείνεται είναι ο περιορισμός της έρευνας αν και ο στόχος είναι να βρεθεί ένας συντελεστής του 1+ε και όχι απλά μια προσέγγιση. Κάτι τέτοιο θα μπορούσε να γενικευτεί και σε βελτιστοποιήσεις σε συνεχόμενα διαστήματα δεικτών.

Σχετικά με την 1+ε προσέγγιση υπάρχει αλγόριθμος ο οποίος κατασκευάζει μια λύση με p levels για κάθε p, x με 1 <= x <= n και 1 <= p <= k. Ο αλγόριθμος αυτός ελέγχει τις τιμές οι οποίες είναι τοποθετημένες σε αυξανόμενη σειρά, δεδομένου ότι αρχικός δείκτης είναι n. Το u_i είναι η τιμή του i-οστού στοιχείου. Ο αλγόριθμος διατηρεί διαστήματα για κάθε 1 <= p <= k, $(a_1^p, b_1^p),\ldots,(a_l^p, b_l^p)$ και έτσι:

1. Τα διαστήματα έχουν χωριστεί και καλύπτουν 1…n. Ως εκ τούτου $a_1^p = 1$ και $b_i^p = n$. Εξάλλου, $b_j^p + 1 = a_{j+1}^p$ για j<l. Έτσι γίνεται $a_j^p = b_j^p$.
2. Διατηρείται η ανισοϊσότητα $SOL[p,b_j^p] <= (1+\delta)SOL[p,a_j^p]$.
3. Αποθηκεύονται $SOL[p,a_j^p]$ και $SOL[p,b_j^p]$ για όλα τα j και p. Αποθηκεύονται επίσης τα αθροίσματα u_i και u_i^2 από i=1 μέχρι r για κάθε r ε $U_{p,j}\{\{a_j^p\} U \{b_j^p\}\}$.
4. Ο αριθμός των διαστημάτων l εξαρτάται από το p οπότε θα χρησιμοποιηθεί ο όρος l αντί για l(p) για λόγους απλοποίησης. Η τιμή του l δεν ορίζεται και είναι τόσο μεγάλη όσο απαιτείται από τις παραπάνω συνθήκες.

Όταν ο αλγόριθμος δει την (n+1)οστή τιμή u_{n+1}, υπολογίζει το SOL[p,n+1] για κάθε p μεταξύ 1 και k και αναβαθμίζει τα διαστήματα $(a_1^p, b_1^p),\ldots,(a_l^p, b_l^p)$. Στην πραγματικότητα ο αλγόριθμος έχει να αναβαθμίσει μόνο το τελευταίο διάστημα (a_l^p, b_l^p) ή να κάνει το $b_l^p = n+1$ ή να δημιουργήσει ένα νέο διάστημα l+1 με $a_{l+1}^p = b_{l+1}^p = n+1$, που εξαρτάται από το SOL[p,n+1]. Έτσι φτάνει στον υπολογισμό του SOL[p,n+1].

Για p=1, το SOL[p,n+1] έχει τιμή VAR[1…n+1] που υπολογίζεται από προκαθορισμένα αθροίσματα. Για να υπολογιστεί το SOL[p+1,n+1] θα πρέπει η τιμή του j να είναι τέτοια ώστε το:
$SOL[p, b_j^p] + VAR[(b_j^p+1)\ldots(n+1)]$
να ελαχιστοποιείται. Το ελάχιστο άθροισμα του παραπάνω θα είναι η τιμή που έχει το SOL[p+1,n+1]. Αυτός ο αλγόριθμος ισχύει στο data stream μοντέλο. Το θεώρημα 1 αποδεικνύει αυτή την προσέγγιση.

<u>Θεώρημα 1</u>
Ο παραπάνω αλγόριθμος δίνει μια $(1+\delta)^p$ προσέγγιση στο OPT[p+1,n+1].

Η απόδειξη είναι με διπλή επαγωγή και ως εκ τούτου πρέπει πρώτα να λυθεί στο n δοθέντος του p. Αν υποτεθεί ότι το θεώρημα είναι αληθές για όλα τα p <= k-1 θα φανεί παρακάτω πως αυτό ισχύει για όλα τα n με p=k. Για p=1 αν υπολογιστεί η διακύμανση θα υπάρξει ακριβής απάντηση. Έτσι το θεώρημα θα είναι αληθές για κάθε n, για p=1. Αν υποτεθεί ότι το θεώρημα είναι αληθές για κάθε n για p=k-1 πρέπει να δειχθεί ότι είναι αληθές για όλα τα n για p=k. Σχετικά με την εξίσωση (1) θα πρέπει να γίνει μια μικρή αλλαγή δηλαδή αντικατάσταση του n από το n+1, ενώ το x θα παραμείνει η τιμή που την ελαχιστοποιεί. Στην πραγματικότητα το (x+1)…(n+1) είναι το τελευταίο διάστημα της βέλτιστης λύσης στο OPT[k,n+1]. Για το OPT[k-1,x] και το διάστημα (a_j^{k-1}, b_j^{k-1}) που βρίσκεται το x, λόγω της μονοτονίας του OPT ισχύει ότι:
$OPT[k-1,a_j^{k-1}] <= OPT[k-1,x] <= OPT[k-1,b_j^{k-1}]$
Από επαγωγική υπόθεση ισχύει ότι:
$SOL[k-1,b_j^{k-1}] <= (1+\delta)SOL[k-1,a_j^{k-1}] <= (1+\delta)[(1+\delta)^{k-2}OPT[k-1,a_j^{k-1}]]$
Επίσης:
$VAR[(b_j^{k-1}+1)…(n+1)] <= VAR[(x+1)…(n+1)]$
Έτσι αν τοποθετηθούν όλα μαζί:
$OPT[k,n+1] = OPT[k-1,x] + VAR[(x+1)…(n+1)] >= (1+\delta)^{-k} SOL[k-1,b_j^{k-1}] + VAR[(b_j^{k-1}+1)…(n+1)]$
Η λύση θα είναι στο μέγιστο $SOL[k-1,b_j^{k-1}] + VAR[(b_j^{k-1}+1)…(n+1)]$. Αυτή είναι η απόδειξη του θεωρήματος.

Αν το δ τεθεί έτσι ώστε $(1+\delta)^k <= 1+\varepsilon$ αρκεί το $\log(1+\delta)=\varepsilon/k$. Ο χώρος που απαιτείται είναι $O(k\log OPT[k,n+1] / \log(1+\delta))$. Αν το καλύτερο είναι 0, θα αντιστοιχεί σε k διαφορετικά σετ από συνεχόμενους ακέραιους που θα αντιστοιχούν στα διαστήματα (a_1^p,b_1^p). Έτσι θα απαιτούνται $\log OPT[p,n+1] / \log(1+\delta)+1$ διαστήματα για αυτό το p και οι συνολικές απαιτήσεις χώρου θα είναι (k^2/ε) φορές $O(\log OPT[k,n+1])$. Το βέλτιστο μπορεί να είναι στο μέγιστο nR^2 όπου R είναι η μεγαλύτερη τιμή και logR είναι ο χώρος που απαιτείται για να αποθηκευτεί ο αριθμός. Άρα το $\log OPT[k,n+1]$ αντιστοιχεί στην αποθήκευση $\log(n)+2$ αριθμών και ο αλγόριθμος χρειάζεται $O((k^2 / \varepsilon)\log n)$ χρόνο για κάθε νέα τιμή. Έτσι προκύπτει το ακόλουθο θεώρημα:

<u>Θεώρημα 2</u>
Είναι δυνατό να δοθεί μια (1+ε) προσέγγιση για το βέλτιστο V-ιστόγραμμα με k levels σε $O((k^2 / \varepsilon)\log n)$ χώρο και $O((nk^2 / \varepsilon)\log n)$ χρόνο σε ένα data stream μοντέλο.

Μια σημαντική κατηγορία προβλημάτων που χρειάζεται να αναλυθεί είναι ότι δοθέντος ενός σετ από τιμές $u_1,…,u_n$ απαιτείται ένα χώρισμα του σετ σε k γειτονικά διαστήματα. Ο στόχος είναι να ελαχιστοποιηθεί μια συνάρτηση ενός μέτρου στο διάστημα. Για τα V-ιστογράμματα η συνάρτηση είναι το άθροισμα και το μέτρο κάθε διαστήματος είναι η διακύμανση των τιμών. Αν το μέτρο μπορεί να προσεγγιστεί μέχρι έναν παράγοντα ρ για το διάστημα, ο δυναμικός προγραμματισμός δίνει μια ρ προσέγγιση για το πρόβλημα, απαιτώντας μόνο τετραγωνικό χρόνο. Στην περίπτωση των V-ιστογραμμάτων η διακύμανση υπολογίστηκε ακριβώς και προέκυψε ένας εξαιρετικός αλγόριθμος.

<u>Θεώρημα 3</u>
Ο ευθύς δυναμικός προγραμματιστικός αλγόριθμος που αναφέρθηκε πιο πάνω γενικεύεται και δίνει έναν (1+ε)ρ αλγόριθμο σε $O(nk^2 / \varepsilon)$ χρόνο για data streams.

Πρώτα από όλα αυτό το θεώρημα θα χρησιμοποιηθεί για μια περίπτωση στην οποία δεν γίνεται να υπολογιστεί το λάθος σε ένα bucket ακριβώς. Για παράδειγμα αντί να ελαχιστοποιηθεί η διαφορά σε ένα διάστημα, να ελαχιστοποιηθεί το $\Sigma_i |u_i - z|$ όπου z είναι η τιμή που είναι αποθηκευμένη στο ιστόγραμμα για αυτό το διάστημα. Σε αυτή την περίπτωση η τιμή του z είναι η διάμεσος του μεγέθους των τιμών. Αυτό αντιστοιχεί στην ελαχιστοποίηση της περιοχής της συμμετρικής διαφοράς των δεδομένων και στην προσέγγιση. Μπορεί να βρεθεί ένα στοιχείο βαθμού $(n/2 + n\varepsilon)$ ή $(n/2 - n\varepsilon)$ (όπου n τα στοιχεία του διαστήματος) σε χώρο $O(\log n/\varepsilon)$. Έτσι με την αύξηση του χώρου οδηγείται σε μια $(1+\varepsilon)$ προσέγγιση.

Θεώρημα 4
Στην περίπτωση κάποιου λάθους l_1 το ιστόγραμμα μπορεί να προσεγγιστεί μέχρι ένα συντελεστή του $1+\varepsilon$ σε χώρο $O((k/\varepsilon)^2 \log^2 n)$ και χρόνο $O(n(k/\varepsilon)^2 \log^2 n)$ για data streams.

Ένα ακόμη παράδειγμα πάνω στο θεώρημα 3 είναι η προσέγγιση με τμηματικά splines (στα μαθηματικά η έννοια spline περιγράφει μια ειδική συνάρτηση που ορίζεται κατά τμήματα από πολυώνυμα και χρησιμοποιείται συχνά σε προβλήματα παρεμβολής) από μικρούς βαθμούς αντί για σταθερά τμήματα. Σκοπός είναι η ελαχιστοποίηση του αθροίσματος των τετραγώνων που είναι κατάλληλο για τα σημεία. Δοθέντος ενός σετ από n τιμές διατεταγμένες κατά ένα εύρος ακεραίων a,…,b μπορούν να παραχθούν οι συντελεστές της καλύτερης κατάλληλης καμπύλης για χρόνο ο οποίος είναι ανεξάρτητος του n.

Στην περίπτωση των μερικώς γραμμικών συναρτήσεων δοθέντος ενός σετ από τιμές u_i με το i να παίρνει τιμές a,…,b δεν είναι δύσκολο να ειπωθεί ότι τοποθετώντας t στο μέσο και αποθηκεύοντας το άθροισμα iu_i μαζί με τα αθροίσματα u_i και u_i^2 μικραίνει η έκφραση από το διάφορο s. Από τη γενίκευσή του μπορεί να διατυπωθεί το ακόλουθο θεώρημα:

Θεώρημα 5
Με την αποθήκευση $\Sigma_i i^d u_i, \ldots, \Sigma_i u_i$ και $\Sigma_i u_i^2$ ο παραπάνω αλγόριθμος επιτρέπει τον υπολογισμό μιας $1+\varepsilon$ προσέγγισης σε χρόνο $O((dnk^2/\varepsilon)\log n)$ και χώρο $O((dk^2/\varepsilon)\log n)$ για data streams.

Αν για κάποιο λόγο δεν γίνεται η υπόθεση ότι κάθε τιμή σε ένα bucket είναι το ίδιο πιθανή, τότε το κάθε bucket παίρνει μια λίστα με τις υπάρχουσες τιμές. Αυτό βοηθάει στη συμπίεση του stream επειδή κάπως έτσι είναι τυπικά η συμπιεσμένη προσέγγιση ιστογράμματος.

Θεώρημα 6
Το παραπάνω μπορεί να προσεγγιστεί μέχρι και $(1+\varepsilon)$ κλάσμα αν οι (ακέραιες) τιμές υπάρχουν σε μια οριοθετημένη περιοχή (για παράδειγμα 1...R) σε $O((Rk^2/\varepsilon)\log n)$ χώρο για ένα data stream.

Βέβαια σε αυτές τις προσεγγίσεις προτιμώνται συχνά συγκεκριμένα ιστογράμματα στα οποία οι πιο συχνά εμφανιζόμενες συχνότητες αποθηκεύονται και οι υπόλοιπες τιμές υποθέτονται ανάλογα. Ακόμη και αυτά τα ιστογράμματα μπορούν να κατασκευαστούν μέχρι και $1+\varepsilon$ συντελεστή, υπό την προυπόθεση ενός οριοθετημένου εύρους τιμών και με την υπόθεση ότι η εκτίμηση του αριθμού των ξεχωριστών τιμών είναι αρκετά δύσκολη. Στη

συμπίεση κειμένου είναι αρκετά ωφέλιμο το χώρισμα των δεδομένων σε blocks καθώς και η συμπίεση αυτών των χωρισμάτων. Το θεώρημα 3 επιτρέπει την αναγνώριση των ορίων (μέχρι και 1+ε) της απώλειας στη βέλτιστη απαίτηση χώρου.

Η συνολική συνάρτηση μπορεί να αντιμετωπιστεί σαν καμπύλη και μπορεί να υπάρξει για αυτή μια 1+ε προσέγγιση με ένα V-ιστόγραμμα με k levels. Επειδή, ωστόσο αυτή η προσέγγιση φαίνεται να εμφανίζει κάποια προβλήματα, οι περισσότερες προσεγγίσεις προτιμούν να προσεγγίζουν τα δεδομένα από ένα ιστόγραμμα και να τα χρησιμοποιούν για γενικούς σκοπούς. Όμως ένα V-ιστόγραμμα δεν είναι αναγκαίο να περιορίσει το λάθος. Έτσι αν προσεγγίσουμε $u_q,...,u_r$ από το z (στα δεδομένα) και με μια μετατόπιση t, το άθροισμα των τετραγωνικών λαθών για συνολικά ερωτήματα είναι:

$$\sum_{q<=i<=r} \left(\sum_{q<=j<=i} u_i - i*z - t\right)^2$$

Βέβαια αυτό προυποθέτει ότι κάθε σημείο είναι εξίσου πιθανό για το σύνολο. Κάπως έτσι μπορεί να προκύψει μια 1+ε προσέγγιση. Επίσης, μπορεί να βελτιστοποιηθεί ένας γραμμικός συνδυασμός λαθών συγκεκριμένων ερωτημάτων ή και συνολικών ερωτημάτων. Κάτι τέτοιο μπορεί να είναι χρήσιμο σε περιπτώσεις όπου η ένωση των τύπων των ερωτημάτων είναι γνωστά.

Hierarchical range queries

Τα παραπάνω ερωτήματα αναφέρονται ως prefix queries και ανήκουν σε ένα ευρύτερο πλαίσιο ιεραρχικού εύρους ερωτημάτων (hierarchical range queries). Αν έπρεπε να δοθεί ένας ορισμός για τα hierarchical range queries, αυτά θα μπορούσαν να περιγραφούν ως εξής: Ένα range query (στα ελληνικά ερώτημα εύρους δηλαδή ερώτημα στη βάση δεδομένων που ανακτά όλες τις εγγραφές όπου κάποια τιμή είναι μεταξύ άνω και κάτω ορίου) R_{ij} ζητάει το άθροισμα των τιμών s_{ij} = A[i]+…+A[j]. Ένα σετ S από range queries είναι ιεραρχικό αν για κάθε δυο ερωτήματα R_{ij} και R_{kl} στο S, είτε τα εύρη [i,j] και [k,l] είναι χωριστά είτε το ένα περιέχεται στο άλλο. Τα hierarchical range queries μπορούν εύκολα να απεικονίζονται σαν δέντρο στο οποίο κάθε κόμβος u έχει ένα εύρος r_u που σχετίζεται με αυτόν.

Το πρόβλημα που προκύπτει σε σχέση με τα optimal ιστογράμματα και τα hierarchical range queries και αξίζει να αναλυθεί είναι το εξής:
Δοθέντος ενός πίνακα A[1,n], B buckets και ενός workload W για hierarchical range queries με πιθανότητα p_{ij} για ένα ερώτημα R_{ij} να καθοριστούν τα όρια και οι μέσοι όροι bucket ενός optimal ιστογράμματος με B buckets.
Με τον όρο workload W περιγράφεται ένα σετ S από hierarchical range queries, μαζί με μια πιθανότητα p_{ij} που σχετίζεται με κάθε range query R_{ij} στο S.
Η πιθανότητα p_{ij} που σχετίζεται με κάθε εύρος μπορεί να ληφθεί με παρακολούθηση και καταγραφή ερωτημάτων στην αποθήκη.
Στο βασικό πρόβλημα που αναφέρεται παραπάνω υπάρχουν αρκετές παραλλαγές. Για παράδειγμα, ένα range query R_{ij} ενδέχεται να μην επιστρέψει το άθροισμα των A[k] τιμών σε αυτό το εύρος, αλλά να επιστρέψει το μεγαλύτερο $max_{k=i}^{j}$ A[k], ή οποιαδήποτε άλλη κατάλληλη συνάρτηση. Μια άλλη παραλλαγή είναι ότι κάποιος ίσως επιλέξει μια διαφορετική αναπαράσταση για το ιστόγραμμα, όπως για παράδειγμα να μην αποθηκευτεί η μέση τιμή αλλά κάτι άλλο παρόμοιο για τις τιμές των buckets. Μια δυνατότητα θα ήταν να αποθηκευτεί μια τιμή που βελτιστοποιεί το λάθος στα range queries. Γενικά, τέτοιες

παραλλαγές μπορούν να υπάρξουν πολλές αλλά κάθε μια από αυτές απαιτεί τη δική της μοναδική λύση ή συνδυασμό λύσεων.

Τέλος, θα μελετηθεί το βασικό πρόβλημα που αναφέρθηκε παραπάνω δηλαδή η κατασκευή optimal ιστογράμματος με συγκεκριμένο πλαίσιο κατασκευής και θα μελετηθεί ιδιαίτερα η συμπεριφορά των V-optimal ιστογραμμάτων σε αυτό το πρόβλημα. Φαίνεται λοιπόν ότι τα V-optimal ιστογράμματα που ελαχιστοποιούν το αναμενόμενο σφάλμα για τα point queries, μπορούν να είναι sub-optimal για την περίπτωση των hierarchical range queries. Για να προσδιοριστεί το V-optimal ιστόγραμμα απαιτεί γνώση των πιθανοτήτων πρόσβασης σε μεμονωμένα σημεία. Αυτό μπορεί να αλιευθεί από το range query workload. Για δυο συγκεκριμένα range queries το V-optimal ιστόγραμμα φαίνεται να έχει πολύ χαμηλό συνολικό προσδοκώμενο σφάλμα. Όμως, το sub-optimality του V-optimal ιστογράμματος για range queries αποδεικνύεται από ένα άλλο ιστόγραμμα το H-optimal, το οποίο έχει μηδενικό συνολικό προσδοκώμενο σφάλμα για αυτά τα δυο hierarchical range queries. Βέβαια η παραπάνω διαπίστωση δεν σημαίνει απαραίτητα ότι κάποιος θα μπορούσε να πάρει ένα H-optimal ιστόγραμμα για έναν δεδομένο αριθμό buckets, ευθυγραμμίζοντας τα όρια του bucket κατά μήκος μερικών hierarchical range query ορίων.

Επίλογος

Κλείνοντας, μπορεί να ειπωθεί ότι η εξόρυξη, ανάλυση και παρουσίαση δεδομένων είναι τομείς που παρουσιάζουν ιδιαίτερο ενδιαφέρον και για περαιτέρω μελέτη ειδικά αν σκεφτεί κανείς ότι η εποχή των μεγάλων όγκων δεδομένων έχει έρθει. Πάντως, όλοι αυτοί οι τομείς είναι δεδομένο ότι χρειάζονται βελτιώσεις για να φτάσουν σε ένα ακόμη πιο ικανοποιητικό επίπεδο. Για παράδειγμα στην εξόρυξη δεδομένων μια καλή ιδέα θα ήταν η γενίκευση των σχέσεων σε παραπάνω από δυο χαρακτηριστικά ή η βελτίωση της στρατηγικής βελτίωσης των πλειάδων και των προτύπων, ενώ στην ανάλυση και παρουσίαση δεδομένων μια πρόταση θα ήταν η μείωση των λαθών ή η μείωση του κόστους σε χρόνο και χώρο. Είναι εντυπωσιακό πόσοι άλλοι κλάδοι ωφελούνται από την ανάπτυξή τους. Από πολύ απλές κατασκευές βάσεων δεδομένων μέχρι συγκρίσεις για αγορές και πολύπλοκες ιατρικές εφαρμογές.

Βιβλιογραφία

[1] Sudipto Guha, Nick Koudas, Kyuseok Shim “Data-Streams and Histograms”

[2] Sergey Brin “Extracting Patterns and Relations from the World Wide Web”

[3] Praveena Mettu “Pattern extraction from the world wide web”

[4] Eugene Agichtein, Luis Gravano “Snowball: Extracting Relations from Large Plain-Text Collections”

[5] Nick Koudas, S. Muthukrishnan, Divesh Srivastava “Optimal Histograms for Hierarchical Range Queries” (Extended Abstract)

[6] Viswanath Poosala, Peter J. Haas, Yannis E. Ioannidis, Eugene J. Shekita “Improved Histograms for Selectivity Estimation of Range Predicates”

[7] www.wikipedia.org