Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Τομέας Επικοινωνιών, ηλεκτρονικής και συστημάτων πληροφορικής

Μελέτη κι υλοποίηση αλγορίθμων για τη βέλτιστη δυνατή εκτέλεση ενός συνδέσμου πολλών σχέσεων στο περιβάλλον Map-Reduce

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Βασίλειος Κ. Παπαϊωάννου

Επιβλέπουσα : Φώτω Ν. Αφράτη
Καθηγήτρια Ε.Μ.Π.

Αθήνα, Μάρτιος 2014

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Τομέας Επικοινωνιών, ηλεκτρονικής και συστημάτων πληροφορικής

Μελέτη κι υλοποίηση αλγορίθμων για τη βέλτιστη δυνατή εκτέλεση ενός συνδέσμου πολλών σχέσεων στο περιβάλλον Map-Reduce

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Βασίλειος Κ. Παπαϊωάννου

Επιβλέπουσα : Φώτω Ν. Αφράτη
Καθηγήτρια Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 21η Ιανουαρίου 2014.

.....	.....	.....
Φώτω Ν. Αφράτη Καθηγήτρια Ε.Μ.Π.	Δημήτριος Φωτάκης Λέκτορας Ε.Μ.Π.	Νικόλαος Παπασπύρου Επίκουρος Καθηγητής Ε.Μ.Π.

Αθήνα, Μάρτιος 2014

(Υπογραφή)

.....

ΒΑΣΙΛΕΙΟΣ Κ. ΠΑΠΑΪΩΑΝΝΟΥ
Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Ευχαριστίες

Από του βήματος αυτού θα ήθελα να εκφράσω την ευγνωμοσύνη μου σε όλους τους ανθρώπους που συνετέλεσαν στην συγγραφή αυτής της διπλωματικής και κατά επέκταση συνέβαλαν στην επιτυχή ολοκλήρωση των σπουδών μου, αυτού του όμορφου ακαδημαϊκού ταξιδιού στην γνώση, στην μάθηση και στην επιστήμη του ηλεκτρολόγου μηχανικού και μηχανικού υπολογιστών.

Ξεκινώντας, αισθάνομαι βαθιά ευγνωμοσύνη για την επιβλέπουσα καθηγήτρια του Ε.Μ.Π. κ. Φώτω Αφράτη η οποία αφενός με επέλεξε να εκπονήσω διπλωματική στο αντικείμενο της κι αφετέρου μου άνοιξε τους ορίζοντες σε ένα πολύ ενδιαφέρον, καινοτόμο κι ωφέλιμο για την εποχή πεδίο της Πληροφορικής. Στην συνέχεια, θέλω να εκφράσω την ευγνωμοσύνη μου στους υποψήφιους διδάκτορες Φίλιππο Καλαμίδα και Νίκο Στασινόπουλο για την συνεχή, μεστή κι ουσιαστική καθοδήγηση τους στην παραγωγή του έργου που εκτίθεται στην παρούσα διπλωματική. Η συμβολή του ήταν παραπάνω από καίρια.

Συνεχίζοντας, χωρίς την κατάλληλη προετοιμασία κι απόκτηση δεξιοτήτων θα ήταν αδύνατη η παρούσα εργασία. Για αυτό θέλω να εκφράσω την ευγνωμοσύνη μου σε όλους τους καθηγητές της σχολής από τους οποίους είχα την τιμή να διδαχθώ τόσο τεχνικά όσο κι ευρύτερα θέματα του αντικειμένου της σχολής κι όχι μόνο. Κατά επέκταση, αισθάνομαι ευγνωμοσύνη απέναντι στην σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών και στο ίδιο το Εθνικό Μετσόβειο Πολυτεχνείο που μου παρείχαν ένα ευρύτερο υποστηρικτικό περιβάλλον υλικό και ακαδημαϊκό. Ενδεικτικά συστατικά αυτού του περιβάλλοντος αποτελούν τα διάφορα εργαστήρια, βιβλιοθήκες, κτίρια, εστίες κι υποδομές του ιδρύματος και της ΣΗΜΜΥ.

Στο ίδιο πλαίσιο θα ήταν παράλειψη να μην αναφέρω την αφανή αλλά σημαντικότατη συμβολή των συμφοιτητών μου μέσω του φόρουμ της σχολής, shmmy.ntua.gr, και του φόρουμ που κατέστησε δυνατή μια τέτοια συμβολή για όλους τους φοιτητές της ΣΗΜΜΥ.

Φτάνοντας προς το τέλος, στέκομαι ευγνώμων απέναντι σε όλους τους συμφοιτητές και ταυτόχρονα φίλους μου που αφενός με βοήθησαν με την σχολή αλλά μου πρόσφεραν και το δώρο της φιλίας τους. Ενδεικτικά και με χρονολογική σειρά αναφέρω τον Λευτέρη Κρητικό, Γιώργο Ανδρίτσο, Χάρη Μπαρνασά, Ευάνθη Παπαζαχαρίου και φυσικά τον Θωμά Φλώρο που πέρα από πολύ καλός φίλος συνιστά και θαυμαστός συνοδοιπόρος στο περιπετειώδες ταξίδι της ζωής.

Κλείνοντας, θέλω να ευχαριστήσω την οικογένεια μου. Σε αυτήν χρωστάω που υπάρχω, ζω κι έχω την τύχη να γράφω αυτήν την διπλωματική. Τέλος, θέλω να ευχαριστήσω τον εαυτό μου που ανέλαβα την ευθύνη και την πρωτοβουλία, κι ανέπτυξα όλη την απαραίτητη δράση ώστε σήμερα ο αναγνώστης να διαβάζει αυτήν την διπλωματική.

Τετάρτη 06. 12. 2012 Ν.Ε.Ε.Μ.Π, Αθήνα.

Μια καλή ζωή είναι αυτή που εμπνέεται από αγάπη και καθοδηγείται από την γνώση.
Bertrand Russell

Η σελίδα αυτή είναι σκόπιμα λευκή.

Περίληψη

Ένα πολύ σημαντικό πρόβλημα του σύγχρονου κόσμου αποτελεί η διαχείριση κι η επεξεργασία των δεδομένων όσο το δυνατόν γρηγορότερα. Τα δεδομένα αυτά ακόμα και στον χρονικό ορίζοντα μιας ημέρας έχουν έναν τεράστιο όγκο εμπεριέχοντας όμως χρησιμότατη πληροφορία για την πρόοδο κι εξέλιξη της ανθρωπότητας. Ένας τρόπος για την άντληση αυτής της πληροφορίας αποτελεί ο λεγόμενος σύνδεσμος και μάλιστα ο πολλαπλός. Ο σύνδεσμος δεν είναι τίποτα άλλο παρά μια τεχνική, μια μέθοδος, ένας αλγόριθμος που συνδυάζει τα δεδομένα, εξάγει την πληροφορία και την παρουσιάζει με κατανοητό τρόπο στον χρήστη. Ο πολλαπλός σύνδεσμος συνδυάζει πολλές πηγές δεδομένων ταυτόχρονα.

Από εκεί και πέρα ο σύνδεσμος για πολλούς λόγους δεν αποδίδει το μέγιστο δυνατό. Από την άλλη αποτελεί μια πολύ πρόσφορη λύση στο προαναφερθέν πρόβλημα γενικά κι ειδικά. Για αυτό ο σύνδεσμος και δη ο πολλαπλός σύνδεσμος, θα πρέπει να μελετηθούν ώστε να ξεπεραστούν τα προβλήματα τους και να γίνουν ακόμα πιο αποδοτικοί ως τεχνική.

Στην παρούσα διπλωματική γίνεται ακριβώς αυτό. Εξετάζεται αρχικά κι εν συντομία η έννοια του πολλαπλού συνδέσμου κι αναδεικνύεται η σημασία του. Έπειτα μελετώνται μερικές σημαντικές και σύγχρονες τεχνικές που είτε εφαρμόζονται ήδη είτε βρίσκονται σε στάδιο ανάπτυξης. Οι τεχνικές αυτές αφορούν σε ποικίλα περιβάλλοντα ώστε να καλυφθεί το θέμα πλήρως. Άλλωστε συνηθίζεται οι καλύτερες τεχνικές να συγκερνούν ένα πλήθος άλλων.

Η τελευταία από όλες τις τεχνικές μελετάται εκτεταμένα κι αποτελεί το κύριο αντικείμενο της παρούσας εργασίας. Η τεχνική αυτή αφορά στην απόδοση ενός συνδέσμου όταν αυτός υλοποιείται στο περιβάλλον του Map – Reduce. Μετά την θεωρητική ανάλυση της μεθόδου αυτής παρουσιάζονται μια σειρά από πειράματα όπου δοκιμάζεται η απόδοση της έναντι πιο παραδοσιακών τεχνικών σε σχέση με διάφορους παράγοντες. Το προγραμματιστικό περιβάλλον στο οποίο γίνονται τα πειράματα είναι αυτό του Hadoop της Apache, μια ανοικτού κώδικα υλοποίηση του Map – Reduce. Τέλος, παρουσιάζονται τα πειραματικά δεδομένα και τα συνεπαγόμενα συμπεράσματα προκειμένου να οριοθετηθεί το πεδίο της περαιτέρω βελτίωσης του συνδέσμου.

Λέξεις – κλειδιά : σύνδεσμος, πολλαπλός σύνδεσμος, Map – Reduce, Hadoop.

Abstract

Data management and processing as fast as possible is one very important problem of modern world. Data, even in the time horizon of a day, have a massive volume, including although very useful information for the progress and evolution of humanity. Join and particular multi – way join is one way to obtain this information. A join is nothing more than a technique, a method, an algorithm that combines data, extracts information and presents this information to the user in an understandable fashion. The multi – way join combines many data sources simultaneously.

Apart from that, the joins for many reasons does not output the maximum possible. On the other hand, the join is a suitable solution to the problem mentioned before generally and specifically. For this, the join and in particular the multi – way join, have to be studied in order their problems to be overcame and to become even more efficient as a technique.

In this diploma thesis this is exactly what happens. In the beginning and briefly the meaning of multi – way is examined and its value is exposed. Then some important and modern techniques are studied which either are already applied or are on a developing stage. These techniques concern various environments with the intention the subject to be fully covered. Nevertheless, is used to the best techniques to combine a set of other techniques.

The last of these techniques is studied extensively and it is the main subject of this thesis. This technique concerns the output of a multi – way join when it is implemented in a Map – Reduce environment. After the theoretical analysis of this method, a series of experiments is presented where the efficiency of this method is tested against more traditional techniques in dependence of various factors. The programing environment where these experiments take place is that of Hadoop, from Apache, which an open source implementation of Map – Reduce. In the end, the experimental data and the consequent conclusions are presented in order to set the field of join's further improvement.

Key – words : join, multi – way join, Map – Reduce, Hadoop.

Η σελίδα αυτή είναι σκόπιμα λευκή.

Αφιερώνεται στους γονείς μου

Κυριάκο κι Ουρανία

Η σελίδα αυτή είναι σκόπιμα λευκή.

Περιεχόμενα

Σχήματα

Πίνακες

Η σελίδα αυτή είναι σκόπιμα λευκή.

Η σελίδα αυτή είναι σκόπιμα λευκή.

ΚΕΦΑΛΑΙΟ 1

ΕΙΣΑΓΩΓΗ

Δεν φοβάμαι τις καταιγίδες που έρχονται όσο μαθαίνω να οδηγώ το πλοίο μου.

Αισχύλος

Με μια ματιά

1.1 Εισαγωγή

Κάθε μέρα που περνάει η ανθρώπινη γνώση αυξάνεται εκθετικά. Ακόμα ταχύτερα αυξάνονται τα δεδομένα που παράγει η ανθρώπινη γνώση. Υπάρχουν διάφοροι οργανισμοί και δράσεις από όλες τις πλευρές της ανθρώπινης δραστηριότητας που παράγουν ή επεξεργάζονται δισεκατομμύρια δεδομένων κάθε μέρα. Ενδεικτικά, αναφέρονται η Wikipedia, το πρόγραμμα του ανθρώπινου γονιδιώματος, η Ουράνια Ψηφιακή Έρευνα Sloan(SDSS), το Facebook κι ο καθένας μπορεί να σκεφτεί πολλά άλλα λιγότερο ή περισσότερο γνωστά παραδείγματα.

Μέσα σε όλα αυτά τα δεδομένα αυτό που ενδιαφέρει πάνω από όλα και δικαιολογεί την συλλογή κι επεξεργασία όλων αυτών των δεδομένων είναι η πληροφορία ή αλλιώς η γνώση. Με βάση αυτή την πληροφορία λαμβάνονται αποφάσεις που επηρεάζουν την ζωή πάρα πολλών ανθρώπων άλλοτε πιο αργά κι άλλοτε πιο γρήγορα. Για παράδειγμα κάθε νέα επιστημονική ανακάλυψη ή τεχνολογική καινοτομία επιδρά στην ζωή των ανθρώπων πολλές φορές με απρόβλεπτο τρόπο. Πριν από το 1994 το διαδίκτυο ήταν στη καλύτερη μια σκέψη, κάτι που ίσως θα ήταν καλό να γίνει. Στις μέρες το διαδίκτυο θεωρείται δεδομένο και γύρω από αυτό ζουν κι αναπτύσσονται εκατομμύρια ανθρώπων. Η παλαιά καθεστηκυία τάξη έχει αλλάξει. Η Wikipedia έχει πάρει την θέση της ακαδημαϊκής κοινότητας, με την έννοια ότι όσα γράφονται σε αυτήν θεωρούνται τουλάχιστον εξίσου αξιόπιστα με τα λεγόμενα της ακαδημαϊκής κοινότητας και λίγο πολύ όλοι για κάθε τι ανατρέχουν σε αυτήν. Παράγονται διπλωματικές εργασίες και πνευματικά έργα όπως ένα βιβλίο που περιλαμβάνουν την Wikipedia άμεσα ως πηγή.

Σαφέστατα λοιπόν, η πληροφορία κρίνεται ζωτικής σημασίας για την πορεία της ανθρωπότητας. Ενώ στην αρχή η πλεονάζουσα πληροφορία προέκυψε από την εξέλιξη της τεχνολογίας και της επιστήμης, η ίδια η πληροφορία εξωθεί τώρα την τεχνολογία και την επιστήμη προς νέες κατευθύνσεις ή να επανεξετάσει τις παλαιότερες. Μια τέτοια κατεύθυνση αποτελεί η επεξεργασία των δεδομένων σε ένα κατανεμημένο περιβάλλον όπου ένα σύμπλεγμα χιλιάδων υπολογιστών εργάζεται ακατάπαυστα προς την επίτευξη ενός κοινού σκοπού. Με άλλα λόγια, ενώ μέχρι ενός σημείου όλα γινόντουσαν σε ένα υπολογιστή πλέον αυτό δεν υπάρχει ως επιλογή. Αναγκαστικά, αν επιθυμείται κάποιο χρήσιμο αποτέλεσμα μέσα σε ένα εύλογο περιβάλλον θα πρέπει οι εργασίες να εκτελεστούν παράλληλα.

Για να γίνουν τα πράγματα ακόμα πιο πολύπλοκα, ενώ παλαιότερα ένα κατανεμημένο περιβάλλον εννοούνταν στα στενά και συγκεκριμένα γεωγραφικά όρια ενός δωματίου ή ενός κτηρίου πλέον ένα κατανεμημένο περιβάλλον μπορεί να απλώνεται σε όλη την υφήλιο. Πώς; Μια εταιρεία λαμβάνει τα δεδομένα προς επεξεργασία από όλον τον κόσμο μέσω του διαδικτύου ή έχει ένα τέτοιο περιβάλλον εγκατεστημένο στο διαδίκτυο. Για να γίνει αντιληπτό το τελευταίο λαμβάνεται ξανά το παράδειγμα της Wikipedia. Ο οργανισμός αυτός επιτρέπει στον καθένα να συγγράψει ένα άρθρο όπου και να βρίσκεται αυτός. Κατόπιν, με το πέρας της συγγραφής γίνεται ένας έλεγχος του άρθρου αυτού κι αναλόγως γίνονται κάποιες ενέργειες. Όπως φαίνεται το έργο της συγκέντρωσης της ανθρώπινης γνώσης στην ιστοσελίδα γίνεται από δισεκατομμύρια υπολογιστές απλωμένους σε όλο τον κόσμο, κατανέμεται σε κάθε έναν από τους κατοίκους της Γης, τουλάχιστον του αναπτυγμένου κόσμου, δυστυχώς.

Επιστρέφοντας στην πληροφορία, η τεχνολογία που επιτρέπει την άντληση της πληροφορίας από τα δεδομένα λέγεται σύνδεσμος. Ο σύνδεσμος αποτελεί μια τεχνική με βάση την οποία συνδυάζονται διαφορετικές πηγές δεδομένων παρέχοντας άλλα δεδομένα και τελικά την ζητούμενη γνώση. Όμως όταν γεννιέται το νέο δεν σημαίνει πως πεθαίνει το παλιό. Αυτό σημαίνει πως ο σύνδεσμος ως τεχνική θα πρέπει να αποδίδει εξίσου καλά σε έναν υπολογιστή, σε ένα σύμπλεγμα υπολογιστών και σε ένα σύστημα που αλληλεπιδρά με το διαδίκτυο και γενικά σε

οτιδήποτε θα μπορούσε να συμβάλλει.

Όπως αναμένεται, η πληθώρα των περιβαλλόντων στα οποία καλείται να δράσει ο σύνδεσμος αυξάνει σημαντικά την πολυπλοκότητά του. Θα πρέπει ο σύνδεσμος να αναπτύσσεται με βάση το πεδίο δράσης του. Επίσης, επηρεάζεται από διάφορους παράγοντες. Χοντρικά μιλώντας, κάτι που κρίνεται αποτελεσματικό στο νερό, δεν είναι εξίσου αποτελεσματικό στον αέρα ή στο έδαφος. Για αυτό διαφορετικές ερευνητικές ομάδες ακολουθούν διαφορετικές στρατηγικές ανάλογα με το πεδίο της έρευνας τους.

Άρα από όλα τα παραπάνω, εξάγεται ένα πρόβλημα, μια λύση και μια μεθοδολογία της λύσης αυτής. Το πρόβλημα είναι η άντληση της πληροφορία μέσα από δισεκατομμύρια δεδομένων. Η λύση συνίσταται στον σύνδεσμο. Η μεθοδολογία αναλύεται στους διαφορετικούς τρόπους με τους οποίους ο σύνδεσμος καθίσταται αποτελεσματικός σε ένα συγκεκριμένο περιβάλλον.

Η παρούσα διπλωματική εργασία εξετάζει τις διάφορες μέχρι τώρα μεθοδολογίες περί του συνδέσμου και καταλήγει σε μία την οποία εξετάζει εκτενώς θεωρητικά αλλά και την υποβάλλει σε διάφορα πειράματα προκειμένου να δοκιμαστεί η απόδοση της. Σημειώνεται πως όλες οι εξεταζόμενες μεθοδολογίες ερευνώνται σε ένα συγκεκριμένο περιβάλλον άλλοτε πιο γενικό κι άλλοτε πιο ειδικό. Με αυτό τον τρόπο επιτυγχάνεται μια σφαιρική κάλυψη του θέματος δίνοντας στον αναγνώστη την δυνατότητα να προχωρήσει προς όποια κατεύθυνση του φαίνεται πιο ενδιαφέρουσα.

1.2 Οργάνωση του τόμου

Η παρούσα διπλωματική επιμερίζεται σε 5 συνολικά κεφάλαια. Το **πρώτο** κεφάλαιο είναι εισαγωγικό ώστε να λάβει ο αναγνώστης μια γενική ιδέα για το αντικείμενο της εργασίας.

Το **δεύτερο** κεφάλαιο επισκοπεί τα κυριότερα προβλήματα και τις λύσεις στο συγκεκριμένο επιστημονικό πεδίο.

Το **τρίτο** κεφάλαιο παρουσιάζει την υλοποίηση της τελευταίας μεθόδου από αυτές που εκτέθηκαν στο προηγούμενο κεφάλαιο.

Το **τέταρτο** κεφάλαιο παρουσιάζει τα πειράματα και τα αποτελέσματα αυτών ώστε να φανεί κατά πόσο κι υπό ποιες συνθήκες η εξεταζόμενη μέθοδος συμφέρει να εφαρμοστεί.

Το **πέμπτο** κεφάλαιο συνοψίζει τα κυριότερα συμπεράσματα και προτείνει μελλοντικές κατευθύνσεις έρευνας.

Φυσικά, στο παράρτημα της εργασίας αυτής υφίσταται εκτενής βιβλιογραφία για περαιτέρω μελέτη.

Κλείνοντας, στο τέλος της διπλωματικής αυτής εργασίας υπάρχει ένα γλωσσάρι στο οποίο επεξηγούνται διάφοροι όροι για τους οποίους θα γίνεται λόγος εν γένει σε όλη την έκταση της εργασίας αυτής. Κάθε όρος που βρίσκεται στο γλωσσάρι, γράφεται με πλάγια γράμματα την πρώτη φορά που γίνεται λόγος για αυτόν στον κορμό της εργασίας.

Η σελίδα αυτή είναι σκόπιμα λευκή.

ΚΕΦΑΛΑΙΟ 2

ΣΥΓΧΡΟΝΕΣ ΤΕΧΝΙΚΕΣ ΠΟΛΛΑΠΛΟΥ ΣΥΝΔΕΣΜΟΥ

Ένας άνθρωπος μπορεί να κάνει την διαφορά κι ο καθένας θα πρέπει να δοκιμάσει.
John Fitzgerald Kennedy

Με μια ματιά

2.1 Βασικές έννοιες

2.1.1 Η έννοια του συνδέσμου

Τα διάφορα δεδομένα που χειρίζεται π.χ. μια επιχείρηση προέρχονται από διάφορες πηγές όπως μια βάση δεδομένων, ένα απλό αρχείο ή το διαδίκτυο. Η παραγωγή πληροφορίας από αυτά τα δεδομένα γίνεται με κάποιο ερώτημα(query). Το ερώτημα μπορεί να αφορά σε μια κατηγορία δεδομένων π.χ. στους πελάτες της επιχείρησης ή να αφορά σε πολλές κατηγορίες δεδομένων π.χ. στους πελάτες και στους λογαριασμούς που αυτοί μπορεί να έχουν στην επιχείρηση. Στην δεύτερη περίπτωση το ερώτημα συνδέει δεδομένα από δύο κατηγορίες για να παράξει την ζητούμενη πληροφορία. Ένα τέτοιο ερώτημα λέγεται σύνδεσμος(join query). Η χρήση του ρήματος “συνδέει” υποδηλώνει πως ο σύνδεσμος εννοείται ως τελεστής, πράξη, συνάρτηση ή αλγόριθμος. Ένας σύνδεσμος, δηλαδή, αποτελεί μια συνάρτηση που δέχεται κάποια δεδομένα ως μεταβλητές κι επιστρέφει κάποια άλλα δεδομένα ως τιμή ή αποτέλεσμα. Επίσης, ένας σύνδεσμος, συνδέει δεδομένα που έχουν κάτι κοινό μεταξύ τους ή που πληρούν κάποιες προϋποθέσεις. Σχετικά με τα δεδομένα, αυτά συνήθως οργανώνονται σε δομή πίνακα όπως σε μια βάση δεδομένων ή ένα απλό αρχείο κειμένου όπου τα δεδομένα διαχωρίζονται με κάποιον ειδικό χαρακτήρα όπως το κόμμα(comma separated values – CSV αρχεία) ή ο στηλοθέτης(tab separated values – TSV αρχεία). Τέλος, ισοδύναμος του όρου πίνακα είναι ο όρος σχέση.

Υπάρχουν δύο ειδών σύνδεσμοι, ο δυαδικός κι ο πολλαπλός. Ο δυαδικός σύνδεσμος(two – way ή binary join) εφαρμόζεται σε δύο σχέσεις μόνο. Ο πολλαπλός σύνδεσμος(multi – way join) εφαρμόζεται σε πολλές σχέσεις. Ένας πολλαπλός σύνδεσμος μπορεί να θεωρηθεί ως επαλληλία δυαδικών. Οι δύο πρώτες σχέσεις συνδέονται μεταξύ τους και το αποτέλεσμα συνδέεται με την τρίτη σχέση κι ούτω κάθε εξής. Από την άλλη ένας πολλαπλός σύνδεσμος μπορεί να υπολογιστεί κατευθείαν, χωρίς την ανάλυση του σε δυαδικούς.

2.1.2 Η σημασία του συνδέσμου

Γενικά, όσο ταχύτερα αντλείται η πληροφορία τόσο καλύτερα. Ένας σύνδεσμος λοιπόν, ως αντλία πληροφορίας θα πρέπει να λειτουργεί όσο τον δυνατό αποδοτικότερα. Βέβαια, αν η ανάγκη του συνδέσμου ως μια αντλία πληροφορίας προέκυπτε σπάνια, θα αρκούσε ίσως να λειτουργεί μέτρια ή κι αργά. Επειδή όμως, η πληροφορία που συγκερνά άλλες πληροφορίες και πολλά δεδομένα είναι αυτή που συνήθως ενδιαφέρει, ο σύνδεσμος χρειάζεται συχνά. Συνεπώς, επιβάλλεται αυτός να λειτουργεί όσο το δυνατόν καλύτερα. Ως αλγόριθμος ένας σύνδεσμος θα πρέπει να χαρακτηρίζεται από την ελάχιστη δυνατή πολυπλοκότητα.

Ένα παράδειγμα για την αναγκαιότητα του συνδέσμου αποτελεί το πρόγραμμα SDSS(Sloan Digital Sky Survey – Ουράνια Ψηφιακή Έρευνα Sloan) [01]. Ο ρυθμός συλλογής αστρονομικών δεδομένων ανέρχεται στα 200GB την ημέρα που ισοδυναμεί με το να βλέπει κάποιος 40 - 400 ταινίες την ημέρα. Το μέγεθος μιας μέσης ταινίας κυμαίνεται από 0.5 – 5GB. Επομένως, ο αποδοτικός σύνδεσμος αποτελεί πρόκληση.

2.1.3 Πολλαπλός ή δυαδικός σύνδεσμος;

Μεταξύ των δύο ειδών συνδέσμου, ποιο είναι καλύτερο; Εξαρτάται. Αυτό που έχει αποδειχθεί ωστόσο με σιγουριά, είναι οτι ο πολλαπλός σύνδεσμος αν και συνθετότερος του δυαδικού, κάποιες φορές και σε συγκεκριμένα περιβάλλοντα υπερτερεί.

Στο δυναμικό περιβάλλον εν γένει υπερτερεί ο πολλαπλός σύνδεσμος. Δεν εμφανίζει τουλάχιστον τα προβλήματα του δυαδικού ενώ έχει μερικές ενδιαφέρουσες ιδιότητες. Τι σημαίνει δυναμικό περιβάλλον; Δυναμικό περιβάλλον σημαίνει ένα περιβάλλον στο οποίο οι συμμετέχουσες στον σύνδεσμο σχέσεις ανανεώνονται συνεχώς με νέα δεδομένα σε ακανόνιστο ρυθμό(π.χ. δεδομένα διαδικτυακής προέλευσης), ή η διαθέσιμη μνήμη αυξομειώνεται απρόβλεπτα εγείροντας διάφορα ζητήματα όπως η υπερχείλιση μνήμης. Επίσης, σημαίνει οτι μπορεί να μην ενδιαφέρει τόσο ο υπολογισμός ολόκληρου του αποτελέσματος του συνδέσμου αλλά ο ταχύτερος δυνατός υπολογισμός ενός μέρους του αποτελέσματος στον δεδομένο χρονικό ορίζοντα.

2.1.4 Το πρόβλημα εν τέλει

Συμπερασματικά, το πρόβλημα είναι, να βρεθεί ένας αλγόριθμος που θα υπολογίζει βέλτιστα έναν σύνδεσμο σε ένα δεδομένο περιβάλλον ή ακόμα καλύτερα σε κάθε περιβάλλον.

Διάφορες ερευνητικές ομάδες έχουν ασχοληθεί με το ζήτημα αυτό. Στο υπόλοιπο μέρος του κεφαλαίου αυτού εξετάζονται μερικές από τις σημαντικότερες προσπάθειες επίλυσης του ζητήματος αυτού.

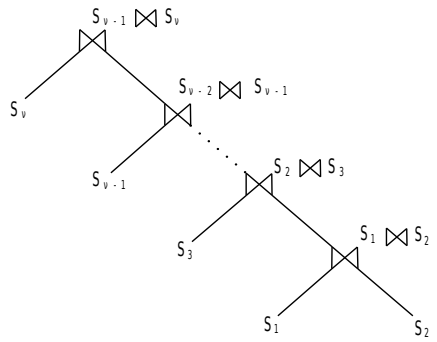
2.2 Υπάρχουσες λύσεις

2.2.1 MJoin

Τα αποτελέσματα της πρώτης υπό εξέταση ομάδας βρίσκονται στο άρθρο [02]. Λεπτομερέστερα, η ομάδα αυτή εισάγει έναν νέο τελεστή με το όνομα MJoin προσπαθώντας αφενός να εξαλείψει τα προβλήματα των μέχρι τώρα τεχνικών που βασίζονται στον δυαδικό σύνδεσμο κι αφετέρου να αυξήσει τον ρυθμό παραγωγής μερικών αποτελεσμάτων. Υπενθυμίζεται πως σε ένα δυναμικό περιβάλλον, ενδιαφέρει περισσότερο ο τάχιστος μερικός υπολογισμός του αποτελέσματος ενός συνδέσμου παρά το αποτέλεσμα συνολικά. Άλλωστε, όταν τα δεδομένα ρέουν ακατάπαυστα τότε χάνεται το νόημα του συνολικού αποτελέσματος κι έχει περισσότερο νόημα ένα μερικό αποτέλεσμα ενταγμένο σε έναν χρονικό ορίζοντα.

Για παράδειγμα, έστω πως το σύστημα μιας επιχείρησης εκτελεί έναν πολλαπλό σύνδεσμο που χειρίζεται απομακρυσμένα ρεύματα δεδομένων που ρέουν αέναα(π.χ. διαδίκτυο). Η παραδοσιακή τεχνική για τον υπολογισμό ενός τέτοιου συνδέσμου συνίσταται στην χρήση δέντρων από δυαδικούς, μερικώς παρεμποδιζόμενους συνδέσμους που ακολουθούν το σχήμα της διοχέτευσης[1]. Ο χαρακτηρισμός “παρεμποδιζόμενος” για τους δυαδικούς συνδέσμους, υποδηλώνει το γεγονός οτι αν κάποια είσοδος(σχέση) τους κολλήσει για κάποιο λόγο, τότε κι ο σύνδεσμος κολλάει. Ο σύνδεσμος, δηλαδή, παρεμποδίζεται από τις εισόδους(σχέσεις) του.

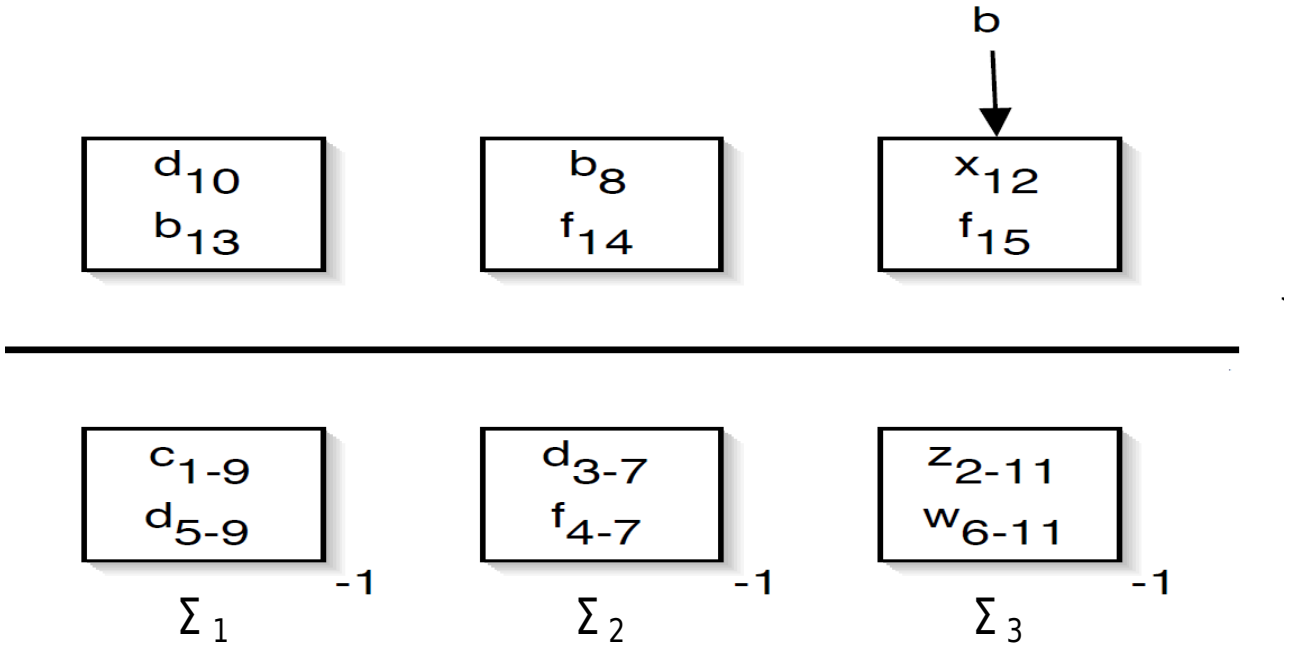
Συνεχίζοντας, κατά την παραδοσιακή τεχνική χτίζεται ένα δέντρο, όπου τα φύλλα του δέντρου αποτελούν τα ρεύματα εισόδου και κάθε ενδιάμεσος κόμβος είναι ο δυαδικός σύνδεσμος των παιδιών του. Θεωρώντας πως ο αλγόριθμος υπολογισμού του εκάστοτε συνδέσμου στο δέντρο αυτό, είναι ο σύνδεσμος κατακερματισμού(Hash Join) τότε διάφορα προβλήματα ανακύπτουν. Αρχικά, υπενθυμίζεται πως ο εν λόγω αλγόριθμος εκτελείται βασικά σε δύο φάσεις. Στην πρώτη φάση(build), για την μικρότερη από τις δύο σχέσεις του συνδέσμου χτίζεται ένας πίνακας κατακερματισμού(hash table). Στην δεύτερη φάση(probe), με βάση τον πίνακα κατακερματισμού εκτελείται η σύνδεση.



Σχήμα 01: Δυαδικό δέντρο υπολογισμού ενός πολλαπλού συνδέσμου

Από το σχήμα εκτέλεσης του πολλαπλού συνδέσμου, Σχήμα 1, κατά τον παραδοσιακό τρόπο, προκύπτει οτι τα αριστερά παιδιά είναι οι σχέσεις για τις οποίες συντίθεται ο πίνακας κατακερματισμού. Οι σχέσεις που κείνται στα δεξιά συμμετέχουν μόνο στην δεύτερη φάση του αλγορίθμου. Ένα πρώτο λοιπόν πρόβλημα, συνίσταται στο γεγονός οτι για να παραχθεί ένα οποιοδήποτε αποτέλεσμα θα πρέπει πρώτα από όλα να έχουν κατασκευαστεί οι πίνακες κατακερματισμού των αριστερών παιδιών S_1, S_3, ..., S_v. Αυτό από μόνο του καθυστερεί σημαντικά τον σύνδεσμο. Ακόμα χειρότερα, αν για κάποιο λόγο το δεξί παιδί – είσοδος κολλήσει, τότε όλη η διαδικασία κολλά και σταματά μέχρις ότου η κολλημένη είσοδος να επανέλθει. Βέβαια σε αυτή τη

1 Στην Πληροφορική η διοχέτευση εννοείται ως μια ακολουθία στοιχείων για την επεξεργασία δεδομένων. Η έξοδος κάθε ενός στοιχείου αποτελεί την είσοδο για το επόμενο αν αυτό υπάρχει. Τα στοιχεία αυτά, λειτουργούν παράλληλα κι ουσιαστικά είναι τα διάφορα, ανεξάρτητα βήματα εκτέλεσης μιας εργασίας. Το “ανεξάρτητα” σημαίνει πως η μόνη εξάρτηση ανάμεσα στα βήματα – στοιχεία είναι οτι το επόμενο περιμένει τα δεδομένα από το προηγούμενο.



Σχήμα 04 : Παράδειγμα διαχείρισης υπερχείλισης μνήμης - Χρονική στιγμή 15

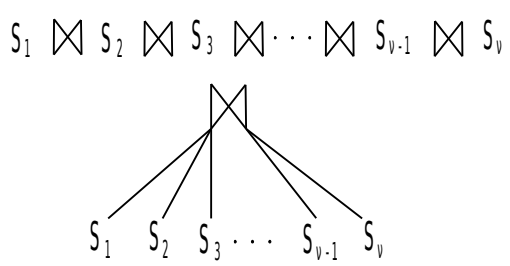
χειρότερη περίπτωση έχουν δοθεί ως λύση οι δυαδικοί, συμμετρικοί σύνδεσμοι όπως ο συμμετρικός σύνδεσμος κατακερματισμού(Symmetric Hash Join) αλγόριθμος που δεν παρεμποδίζεται από την λειτουργία των εισόδων του. Παρεμπιπτόντως, όταν ένας σύνδεσμος λέγεται συμμετρικός τότε αυτό σημαίνει πως η λειτουργία του δεν παρεμποδίζεται από τις εισόδους του.

Ένα δεύτερο πρόβλημα αποτελεί ο περίσσιος φόρτος εργασίας κι αποθήκευσης που εισάγει ο δυαδικός σύνδεσμος ακόμα κι όταν χαρακτηρίζεται συμμετρικός. Όταν κάποιος ενδιάμεσος σύνδεσμος παράγει το αποτέλεσμα του, τότε το αποτέλεσμα αυτό ανελίσσεται προς την ρίζα. Στην διάρκεια αυτής της ανέλιξης, το αποτέλεσμα συμμετέχει στους ενδιάμεσους συνδέσμους που συναντά αυξάνοντας τους απαιτούμενους πόρους. Για παράδειγμα, έστω οτι φτάνουν νέα δεδομένα από την είσοδο S_1 που συνδέονται με την ήδη διαβασμένη πλειάδα δεδομένων $X_{1,2}$ της εισόδου S_2. Αυτά τα $X_{1,2}$ δεδομένα, εφόσον συνεισφέρουν στο τελικό αποτέλεσμα, ανεβαίνουν προς την ρίζα του δέντρου, ξεκινώντας από τον σύνδεσμο των S_1, S_2 και καταλήγοντας στον σύνδεσμο των S_{v-1}, S_v περνώντας από όλα τα ενδιάμεσα βήματα εκτέλεσης του συνολικού, πολλαπλού συνδέσμου. Το πέρασμα από κάθε ένα ενδιάμεσο στάδιο περιλαμβάνει την εισαγωγή των δεδομένων αυτών στον πίνακα κατακερματισμού του τρέχοντος σταδίου. Αυτός ο μηχανισμός δημιουργεί πολλά νέα δεδομένα στην πορεία εκτέλεσης του πολλαπλού συνδέσμου που απαιτούν πόρους για την αποθήκευση τους και για την διάδοση τους προς την ρίζα του δέντρου. Όπως γίνεται σαφές υφίσταται μια σπατάλη πόρων για κάθε πλειάδα δεδομένων εξόδου που με την σειρά της εν δυνάμει καθυστερεί τον ρυθμό παραγωγής αποτελεσμάτων.

Ένα τρίτο πρόβλημα, είναι οι διάφορες παρενέργειες που οφείλονται στην φύση αυτή κάθε αυτή του δέντρου υπολογισμού του δυαδικού συνδέσμου και δεν αφορούν σε κάποια ιδιομορφία των εισόδων. Συγκεκριμένα, αν οι διάφορες είσοδοι εισάγουν δεδομένα με ένα διαφορετικό ρυθμό από των υπολοίπων, τότε, ο ρυθμός εξόδου εξαρτάται από την λειτουργία του βελτιωτικού μηχανισμού. Ο βελτιωτικός μηχανισμός ευθύνεται για την δημιουργία του δέντρου εκτέλεσης του πολλαπλού συνδέσμου όπως στο Σχήμα 1. Ο βελτιωτικός μηχανισμός επιλέγει το ύψος του δέντρου ή την θέση των εισόδων με τον ταχύτερο ρυθμό εισαγωγής δεδομένων στον πολλαπλό σύνδεσμο. Σε κάθε περίπτωση, ο ρυθμός εξόδου εξαρτάται από τον βελτιωτικό μηχανισμό. Αυτή η εξάρτηση επιδεινώνεται σε περιπτώσεις που μερικοί ή όλοι οι επιμέρους δυαδικοί σύνδεσμοι του δέντρου ξεπερνούν την διαθέσιμη τους μνήμη, μεταφέροντας ένα μέρος των δεδομένων τους στον δίσκο για

μετέπειτα επεξεργασία. Αυτή η εξάρτηση είναι κακή. Είναι κακή, γιατί απαιτεί τον υπολογισμό ενός βέλτιστου δέντρου υπολογισμού του πολλαπλού συνδέσμου, γεγονός που από μόνο του ως θέμα αποτελεί πρόκληση. Αλλά ακόμα κι αν βρεθεί αυτό το δέντρο, δεν σημαίνει πως είναι το βέλτιστο για όλο τον χρονικό ορίζοντα εκτέλεσης τους συνδέσμου. Αυτό το γεγονός επιβάλλει τον τακτικό επαναϋπολογισμό του δέντρου εκτέλεσης, δηλαδή, επιβάλλει ένα σημαντικό κόστος σε όλη την διαδικασία. Συνοψίζοντας, η φύση του δέντρου επιβαρύνει τον πολλαπλό σύνδεσμο.

Η υπό εξέταση λύση, ξεπερνά τα παραπάνω προβλήματα, εισάγοντας τον πολλαπλό σύνδεσμο MJoin. Ο MJoin αποτελεί κι αυτός έναν αλγόριθμο, μια συνάρτηση όπως και κάθε άλλος σύνδεσμος. Σε αντιδιαστολή με τον δυαδικό σύνδεσμο, εφαρμόζεται σε πολλές εισόδους – σχέσεις ταυτόχρονα. Ουσιαστικά επεκτείνει τον συμμετρικό, δυαδικό σύνδεσμο και τον XJoin [03] σε πολλές εισόδους – σχέσεις.



Σχήμα 02 : MJoin

Όπως έχει ήδη ειπωθεί ένας σύνδεσμος εκτελείται όταν συντρέχουν κάποιες συνθήκες. Ο MJoin συγκεντρώνει όλες αυτές τις συνθήκες μαζί, ουσιαστικά ενώνει τους επιμέρους συνδέσμους σε έναν. Σχηματικά, ο MJoin φαίνεται στο Σχήμα 2 όπου φαίνεται κι η συνένωση των επιμέρους συνδέσμων σε έναν. Στην συνένωση αυτή, κάθε μία από τις συνθήκες ταξιθετείται[2] ανάλογα με την πιθανότητα της να εκπληρωθεί. Η πιθανότητα αυτή λέγεται εκλεκτικότητα της συνθήκης και συμβολίζεται με σ. Όσο η σ → 0 τόσο πιο εκλεκτική θεωρείται η συνθήκη αφού τόσο πιο δύσκολα εκπληρώνεται. Για έναν σύνδεσμο, αυτό σημαίνει πως όσο πιο εκλεκτικές είναι οι συνθήκες του τόσο λιγότερα δεδομένα εξόδου παράγει. Αυτό είναι καλό και συνιστά την ιδέα κλειδί του αλγορίθμου. Είναι καλό, γιατί παράγονται μόνο τα αναγκαία ενδιάμεσα και τελικά αποτελέσματα. Αλγοριθμικά λοιπόν, για κάθε είσοδο δημιουργείται ένας πίνακας κατακερματισμού. Όταν έρχεται μια νέα πλειάδα δεδομένων σε κάποια είσοδο, αυτή εισάγεται στον κατάλληλο πίνακα και κατασκευάζεται ειδικά για αυτήν μια αλυσίδα από τις συνθήκες του συνδέσμου. Οι κρίκοι – συνθήκες συντάσσονται με τέτοιο τρόπο ώστε οι πιο εκλεκτικές συνθήκες να βρίσκονται στην αρχή. Η νεοεισερχόμενη πλειάδα καθώς περνά από κρίκο σε κρίκο συνδέεται με τους υπόλοιπους πίνακες κατακερματισμού. Συνεπώς παράγεται κάθε δυνατή σύνδεση, δηλαδή, πλειάδα δεδομένων, από την σύνδεση της νεοεισερχόμενης πλειάδας δεδομένων και των ήδη υπαρχουσών πλειάδων δεδομένων στην μνήμη των άλλων πινάκων κατακερματισμού. Φυσικά, η προαναφερθείσα σύνδεση δεν γίνεται με κάθε πίνακα κατακερματισμού κάθε φορά, αφού η συνθήκη σε έναν κρίκο μπορεί να μην συντρέχει. Λόγω της κατασκευής της αλυσίδας και του τρόπου που μια νέα πλειάδα συνδέεται με τις ήδη υπάρχουσες, εξασφαλίζεται η γρηγορότερη δυνατή εξάλειψη περιττών ενδιάμεσων αποτελεσμάτων κι υπολογισμών. Επιταχύνεται κατά επέκταση η εκτέλεση του συνδέσμου. Επισημαίνεται τέλος, πως η κατασκευαζόμενη αλυσίδα αφορά σε μια συγκεκριμένη πλειάδα εισόδου και καμία άλλη.

Προχωρώντας, ο MJoin ως αλγόριθμος, μεριμνά για τον υπολογισμό της αλυσίδας, την αποφυγή επαναϋπολογισμού των ενδιάμεσων αποτελεσμάτων, την υπερχείλιση της διαθέσιμης μνήμης και την αποφυγή παραγωγής πλεοναζόντων αντιτύπων των πλειάδων δεδομένων εξόδου. Η αλυσίδα υπολογίζεται σε χρόνο O(ν(ν – 1)log(ν – 1)) για ένα πολλαπλό σύνδεσμο ν εισόδων. Υπενθυμίζεται πως μια αλυσίδα αφορά σε μία από τις ν εισόδους. Η κ – οστή αλυσίδα για την είσοδο κ, υπαγορεύει με ποια σειρά θα πρέπει να γίνουν οι συνδέσεις ανάμεσα στην

2 Στην βιβλιογραφία, γίνεται χρήση του “ταξινομώ” για να δηλωθεί η τοποθέτηση κάποιων στοιχείων σε κάποια σειρά ή στην κατάλληλη θέση. Το “ταξινομώ” όμως αφορά στον χωρισμό των στοιχείων σε τάξεις και δεν σημαίνει αναγκαία οτι τα στοιχεία έχουν κάποια σειρά(ΜΕΙΖΟΝ ΕΛΛΗΝΙΚΟ ΛΕΞΙΚΟ, Τεγόπουλος Φυτράκης). Το “ταξιθετώ” σημαίνει την τοποθέτηση κάποιων στοιχείων στην κατάλληλη θέση, σειρά(sort στα αγγλικά). Στο κείμενο επομένως γίνεται χρήση του “ταξιθετώ” που πράγματι σημαίνει το ζητούμενο.

νεοεισερχόμενη πλειάδα δεδομένων από την είσοδο κ και στους υπόλοιπους ν – 1 πίνακες κατακερματισμού ώστε να παράγεται ο ελάχιστος αριθμός ενδιάμεσων αποτελεσμάτων. Έχοντας αυτό υπ' όψιν, ο χρόνος O((ν – 1)log(ν – 1)) αφορά σε κάθε μία από τις ν εισόδους. Συγκεκριμένα, για κάθε είσοδο ταξιθετούνται οι ν – 1 υπόλοιπες είσοδοι ως προς την εκλεκτικότητα τους. Ο συντελεστής ν στον χρόνο O(ν(ν – 1)log(ν – 1)) οφείλεται στο γεγονός ότι η εύρεση της αλυσίδας γίνεται για κάθε είσοδο, αφού από όλες τις εισόδους αναμένεται κάποια νέα πλειάδα δεδομένων.

Αλλάζοντας θέμα, ο MJoin σε κάποιες περιπτώσεις αναπόφευκτα επαναϋπολογίζει κάποια ενδιάμεσα αποτελέσματα αφού δεν τα αποθηκεύει. Οι περιπτώσεις αυτές, αφορούν κυρίως σε περιπτώσεις που μια πλειάδα δεδομένων από ένα ρεύμα εισόδου, συνδέεται με πολλές πλειάδες από τα εναπομείναντα ρεύματα εισόδου. Ως λύση σε αυτές τις περιπτώσεις προτείνεται η αποθήκευση των αποτελεσμάτων αυτών μέσω της διαίρεσης του MJoin σε μικρότερους ή οριακά στην μετατροπή του σε ένα δέντρο δυαδικών συνδέσμων. Συμπερασματικά, παρ' όλη αυτήν την εγγενή αδυναμία του, ο MJoin παράγει αποτελέσματα, τουλάχιστον τόσο γρήγορα όσο κι ένα δέντρο δυαδικών συνδέσμων χάρη σε ένα αριθμό άλλων παραγόντων. Ενδεικτικά, η καλή κατασκευή της αλυσίδας κάθε πλειάδας εισόδου κι η ορθολογικότερη χρήση της μνήμης με το να μην αποθηκεύονται ενδιάμεσα αποτελέσματα είναι μερικοί παράγοντες.

Η διαχείριση περιπτώσεων υπερχείλισης την διαθέσιμης μνήμης γίνεται παρόμοια με τον δυαδικό σύνδεσμο XJoin. Αναλυτικότερα, υπάρχουν τρία στάδια στην διάρκεια των οποίων γίνονται οι διάφορες συνδέσεις. Υπάρχει το στάδιο από μνήμη σε μνήμη όπου γίνονται όλες οι συνδέσεις, όταν δεν υπάρχει υπερχείλιση, όλες οι είσοδοι λειτουργούν κανονικά, δεν έχουν κολλήσει κι όταν υπάρχει διαθέσιμος χώρος για την νεοεισερχόμενη πλειάδα δεδομένων στην θέση μνήμης που απεικονίζεται. Αν δεν υπάρχει διαθέσιμος χώρος, τότε, η νεοεισερχόμενη πλειάδα δεδομένων στέλνεται στον δίσκο. Ένα άλλο στάδιο συνιστά το από δίσκο σε μνήμη όπου, όταν οι προς σύνδεση είσοδοι έχουν κολλήσει, γίνονται συνδέσεις μεταξύ πλειάδων δεδομένων στον δίσκο και στην μνήμη μέχρις ότου οι είσοδοι επανέλθουν. Η σύνδεση αυτή γίνεται όπως και στο στάδιο από μνήμη σε μνήμη αφού πρώτα μεταφερθούν κάποια δεδομένα από τον δίσκο στην μνήμη.

Το τρίτο στάδιο λέγεται από δίσκο σε δίσκο. Το στάδιο αυτό, εκτελείται όταν έχουν στερέψει οι είσοδοι. Στο στάδιο αυτό, όσες πλειάδες δεδομένων έχουν απομείνει και στην μνήμη και στον δίσκο συνδέονται καταλλήλως. Λεπτομερέστερα, για κάποια είσοδο επιλέγεται ένα ολόκληρο διαμέρισμα της από τον δίσκο, για το οποίο χτίζεται ένας πίνακας κατακερματισμού στην μνήμη. Μετά γίνονται συνδέσεις ανάμεσα στις πλειάδες των διαμερισμάτων του δίσκου που αφορούν στις άλλες σχέσεις και στον πίνακα κατακερματισμού που ήδη αναφέρθηκε.

Συνοψίζοντας την μόλις περιγραφείσα τεχνική για την υπερχείλιση μνήμης, τα τρία στάδια ουσιαστικά αποτελούν ένα στάδιο. Η τεχνική σύνδεσης δεν αλλάζει από στάδιο σε στάδιο. Αυτό που αλλάζει είναι η προέλευση των δεδομένων που συνδέονται κι η προέλευση του πίνακα κατακερματισμού. Στο από μνήμη σε μνήμη στάδιο, τόσο τα δεδομένα όσο κι ο πίνακας κατακερματισμού προέρχονται από την μνήμη. Ο πίνακας κατακερματισμού χτίζεται με βάση κάποιο “διαμέρισμα” της μνήμης. Στο από δίσκο σε μνήμη στάδιο, τα δεδομένα έρχονται από τον δίσκο κι πίνακας κατακερματισμού κείται στην μνήμη. Στο από δίσκο σε δίσκο στάδιο εξίσου τα δεδομένα κι ο πίνακας κατακερματισμού προέρχονται από τον δίσκο. Ο πίνακας κατακερματισμού χτίζεται με βάση κάποιο διαμέρισμα του δίσκου. Συμπερασματικά, στην σημειολογία από χ σε ψ το χ δηλώνει την προέλευση των δεδομένων που συνδέονται και το ψ δηλώνει την καταγωγή του διαμερίσματος για το οποίο χτίζεται ο πίνακας κατακερματισμού.

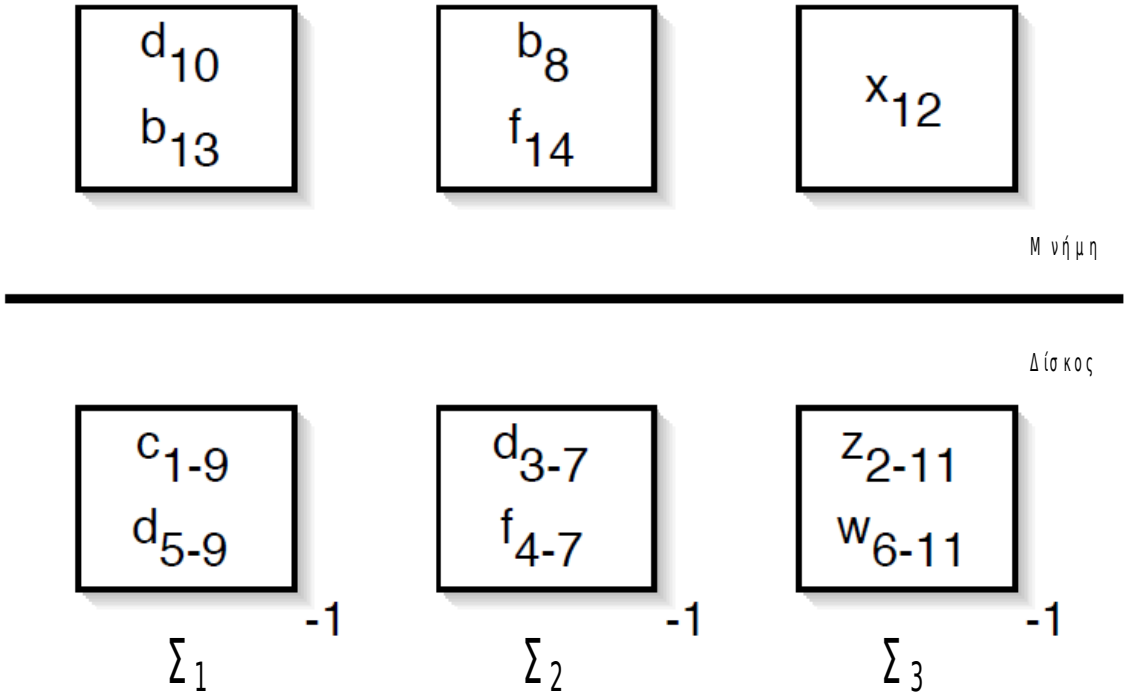
Τα παραπάνω στάδια είναι κοινά ανάμεσα στον πολλαπλό σύνδεσμο MJoin και στον XJoin. Ο MJoin όμως, προκειμένου να έχει καλό ρυθμό εξόδου, εισάγει μια τεχνική με το όνομα συντεταγμένη απόρριψη(coordinated flushing). Σύμφωνα με αυτήν την τεχνική, οι είσοδοι – συμμετέχουσες σχέσεις στον σύνδεσμο διαμερίζονται σύμφωνα με την κοινή ιδιότητα με βάση την

οποία γίνεται η σύνδεση. Κάθε φορά που εξαντλείται η διαθέσιμη μνήμη για κάποια σχέση, κι άρα θα πρέπει να απορριφθεί ένα μέρος της στον δίσκο, αυτό γίνεται από ένα συντεταγμένο, συγκεκριμένο διαμέρισμα της σχέσης π.χ. απορρίπτεται τα διαμέρισμα 1234321. Αυτό ισχύει για όλες τις σχέσεις. Όλες οι σχέσεις απορρίπτουν δεδομένα από το δικό τους διαμέρισμα 1234321 μέχρις ότου όλα τα διαμερίσματα 1234321 αδειάσουν. Με αυτόν τον τρόπο, όταν εισέρχεται μια νέα πλειάδα δεδομένων, τότε αυτή συνδέεται αν κι εφόσον υπάρχει μια ελεύθερη θέση μνήμης για αυτή. Αν δεν υπάρχει, τότε πάει στον δίσκο στο αντίστοιχο διαμέρισμα 1234321 της μνήμης κι αποφεύγονται οι όποιες περιπλοκές από την αναζήτηση μιας θέση μνήμης για την νεοεισερχόμενη πλειάδα. Συνεπώς, ο ρυθμός εξόδου δεν εμποδίζεται από την υπερχείλιση μνήμης.

Τέλος, ο MJoin αποφεύγει τα πολλαπλά αντίτυπα μιας πλειάδας δεδομένων εξόδου με την σφράγιση κάθε πλειάδας δεδομένων με τις χρονικές στιγμές που αυτή εισάγεται στο σύστημα κι εγκαταλείπει την μνήμη αντίστοιχα. Με βάση τις σφραγίδες αυτές γίνονται οι κατάλληλοι έλεγχοι κι επιτυγχάνεται η κάθε πλειάδα εξόδου να χαρακτηρίζεται μοναδική.

Για να γίνουν τα παραπάνω κατανοητά δίνεται ένα παράδειγμα. Επιθυμείται ο πολλαπλός σύνδεσμος μεταξύ των σχέσεων – ρευμάτων δεδομένων Σ_1, Σ_2 και Σ_3. Για κάθε μία σχέση δεσμεύεται ένα διαμέρισμα δεδομένων και τους διατίθεται μνήμη που χωράει το πολύ δύο πλειάδες δεδομένων. Κάθε πλειάδα δεδομένων σφραγίζεται με ένα χρονικό διάστημα που δηλώνει την χρονική στιγμή έλευσης της πλειάδας στην μνήμη και την στιγμή που απομακρύνθηκε από αυτή και μεταφέρθηκε στον δίσκο. Επίσης για κάθε διαμέρισμα δεδομένων στο δίσκο σημειώνεται η τελευταία φορά που συνδέθηκε με κάποια άλλα δεδομένα. Ένα διαμέρισμα δεδομένων που ποτέ δεν συνδέθηκε με κάποια άλλα δεδομένα, σημειώνεται με “-1”.

Για αρχή, η εικόνα της μνήμης και του δίσκου για τις τρεις σχέσεις φαίνεται στο σχήμα 3. Την χρονική στιγμή 15 εισάγεται στην μνήμη της σχέσης Σ_3 η πλειάδα f και την χρονική στιγμή 16 φτάνει στην Σ_3 η πλειάδα b. Το στιγμιότυπο της στιγμής 15 παρουσιάζεται στο σχήμα 4, όπου η πλειάδα b μόλις φτάνει στην Σ_3 αλλά δεν έχει μπει στην μνήμη. Την στιγμή 15 λοιπόν, ανάλογα με την σειρά που δοκιμάζεται η σύνδεση της f με κάθε μία από τις Σ_1, Σ_2 παράγεται και διαφορετικό



Σχήμα 03 : Παράδειγμα διαχείρισης υπερχείλισης μνήμης - Αρχική κατάσταση

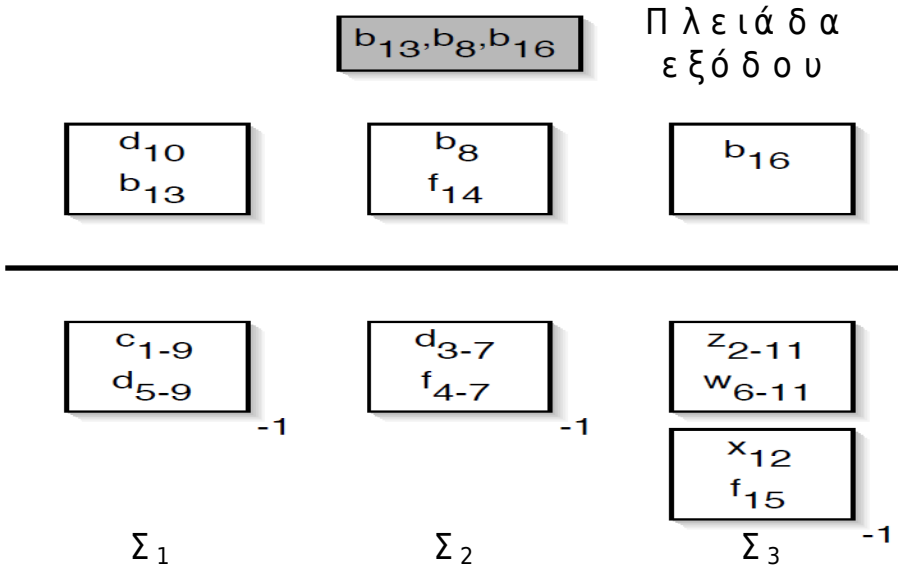
αποτέλεσμα. Αν η σειρά είναι η { Σ_1, Σ_2} τότε δεν παράγεται κάποιο αποτέλεσμα, γιατί η Σ_1 δεν έχει πλειάδα του τύπου f. Αν η σειρά είναι η { Σ_2, Σ_1} τότε παράγεται το ενδιάμεσο αποτέλεσμα { f_{14}, f_{15}} το οποίο τελικά απορρίπτεται αφού η Σ_1 δεν έχει πλειάδες τύπου f κι άρα δεν μπορεί να γίνει κάποια σύνδεση.

Την στιγμή 16, η b εισέρχεται στην μνήμη της Σ_3 και συμβαίνει υπερχείλιση. Τα περιεχόμενα της μνήμης της Σ_3 αδειάζονται στον δίσκο σύμφωνα με την τεχνική της συντεταγμένης απόρριψης, θεωρητικά τουλάχιστον. Πρακτικά, επειδή το διαμέρισμα για κάθε σχέση είναι μοναδικό η τεχνική αυτή είναι σαν μην εφαρμόζεται. Ύστερα από αυτό το γεγονός η εικόνα της μνήμης και του δίσκου είναι όπως αυτή του σχήματος 5. Στο σχήμα αυτό επίσης φαίνεται η παραγωγή της πλειάδας { b_{13}, b_8, b_{16}}.

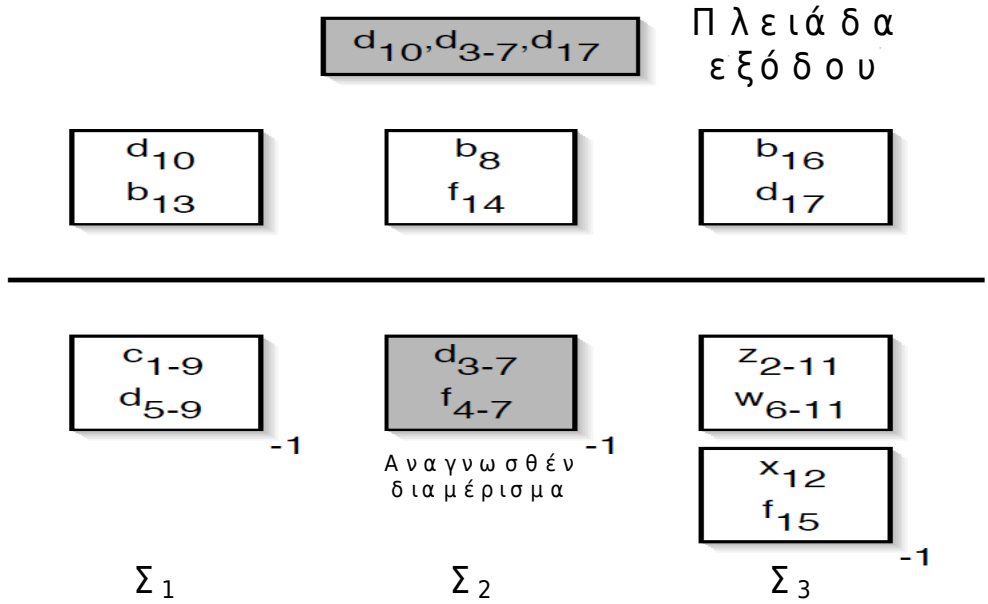
Την χρονική στιγμή 17, επίσης στην μνήμη της σχέσης Σ_3, εισέρχεται μια νέα πλειάδα, η d. Σε αυτό το σημείο όλες οι είσοδοι – ρεύματα – σχέσεις κολλάνε κι οπότε δοκιμάζονται συνδέσεις ανάμεσα σε πλειάδες της μνήμης και του δίσκου. Πράγματι, η νέα πλειάδα ταιριάζει με πλειάδες των σχέσεων Σ_1, Σ_2 και παράγεται στην έξοδο το αποτέλεσμα { d_{10}, d_{3-7}, d_{17}} όπως απεικονίζεται και στο σχήμα 6.

Τέλος, κι οι τρεις σχέσεις στέλνουν το σήμα πως έχουν στερέψει ως ρεύματα εισόδου και ο

αλγόριθμος περνά στην φάση από δίσκο σε δίσκο. Στην φάση αυτή συνδέονται, όσες πλειάδες έχουν απομείνει, αν είναι δυνατόν φυσικά. Η πλειάδα d μπορεί να συνδεθεί με δύο διαφορετικούς τρόπους. Με τον πρώτο τρόπο παράγεται το αποτέλεσμα { d_{10}, d_{3-7}, d_{17}} το όποιο όμως έχει ήδη παραχθεί και για αυτό δεν παράγεται εκ νέου. Με τον δεύτερο τρόπο, παράγεται η πλειάδα { d_{5-9}, d_{3-7}, d_{17}} η οποία είναι κι επιθυμητή.



Σχήμα 05 : Παράδειγμα διαχείρισης υπερχείλισης μνήμης - Χρονική στιγμή 16



Σχήμα 06 : Παράδειγμα διαχείρισης υπερχείλισης μνήμης - Χρονική στιγμή 17

Ένα άλλο αξιοσημείωτο χαρακτηριστικό του MJoin αφορά στην συμπεριφορά του όταν εκτελείται σε ένα συγκεκριμένο χρονικό ορίζοντα ή ορίζοντα πλήθους πλειάδων. Σε αυτή την περίπτωση ο MJoin λειτουργεί ως σύνδεσμος – παράθυρο(window join) και πρέπει να λειτουργεί ως τέτοιος, γιατί αλλιώς απαιτείται άπειρη μνήμη για να αποθηκευτούν οι άπειρες πλειάδες. Αποδεικνύεται πειραματικά, πως επειδή ο MJoin έχει σχεδιαστεί να συμπεριφέρεται βέλτιστα σε ρεύματα εισόδου που χωράνε στην μνήμη, παράγει τα καλύτερα αποτελέσματα ταχύτερα σε σχέση με τις παραδοσιακές τεχνικές.

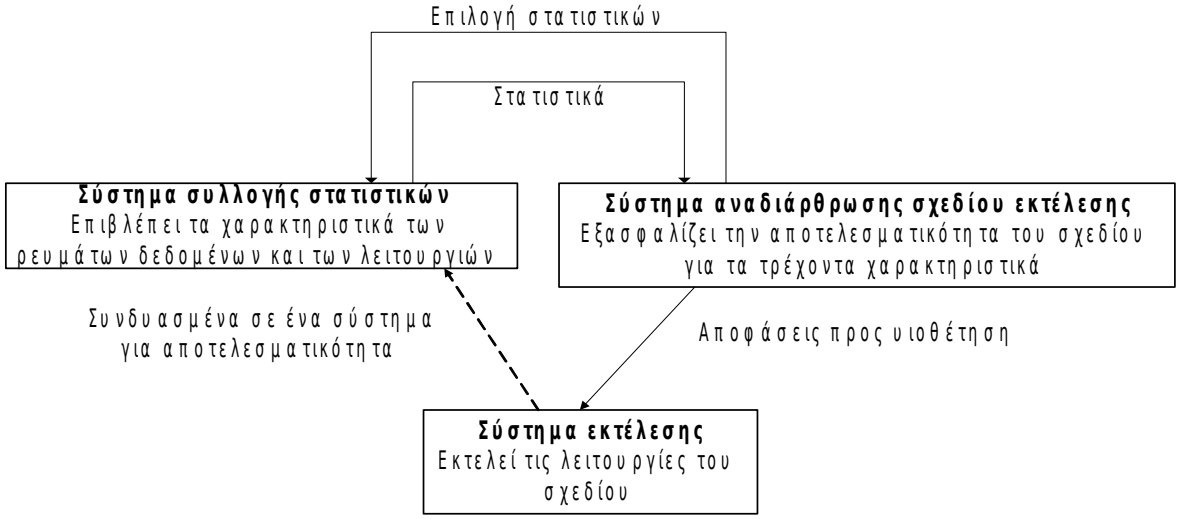
Κλείνοντας, την αναφορά σε αυτή την υπάρχουσα λύση, σύμφωνα με τα πειράματα, ο MJoin είναι καλός, γιατί δεν επηρεάζεται από τον ρυθμό εισαγωγής νέων δεδομένων, διαχειρίζεται καλά τα θέματα υπερχείλισης της μνήμης και δεν υποφέρει από τα προβλήματα του δέντρου δυαδικών συνδέσμων. Παρ' όλα αυτά, σύμφωνα με τα πειράματα υπάρχουν εναλλακτικές που αποδίδουν καλύτερα σε σχέση με τον αυτόν. Μια τέτοια εναλλακτική είναι ένα σύνολο από μικρότερους πολλαπλούς, παράλληλους συνδέσμους.

2.2.2 StreaMon

Η γενική ιδέα της λύσης αυτής είναι αφού υποβληθεί ο σύνδεσμος προς εκτέλεση, να παραχθεί ένα αρχικό σχέδιο υπολογισμού αυτού, που στην συνέχεια βελτιώνεται δυναμικά παράλληλα με τις αλλαγές του περιβάλλοντος εκτέλεσης.

Η ιδέα αυτή ενσαρκώνεται από ένα πρωτότυπο σύστημα διαχείρισης ρευμάτων δεδομένων(Data Stream Management System – DSMS) [04], [05] με το όνομα STREAM. Το STREAM αποτελεί ένα σχεσιακό σύστημα διαχείρισης ρευμάτων δεδομένων που υποστηρίζει συνεχείς συνδέσμους όπως αυτοί ορίζονται μέσω μιας δηλωτικής γλώσσας με το όνομα CQL. Τέλος, το STREAM διαθέτει ένα διαδραστικό γραφικό περιβάλλον ώστε ο χρήστης να εποπτεύει οπτικά το σχέδιο εκτέλεσης και την συμπεριφορά του συστήματος.

Ακόμη, το σύστημα αυτό διαθέτει μια μηχανή, το StreaMon, που επεξεργάζεται δυναμικά συνδέσμους. Η αρχιτεκτονική της μηχανής αυτής, Σχήμα 3, αναλύεται σε τρία συστατικά μέρη, το σύστημα εκτέλεσης του σχεδίου υπολογισμού του συνδέσμου, το σύστημα συλλογής στατιστικών για την αναδιάρθρωση του σχεδίου υπολογισμού του συνδέσμου και το σύστημα που επιτυγχάνει την εν λόγω αναδιάρθρωση.



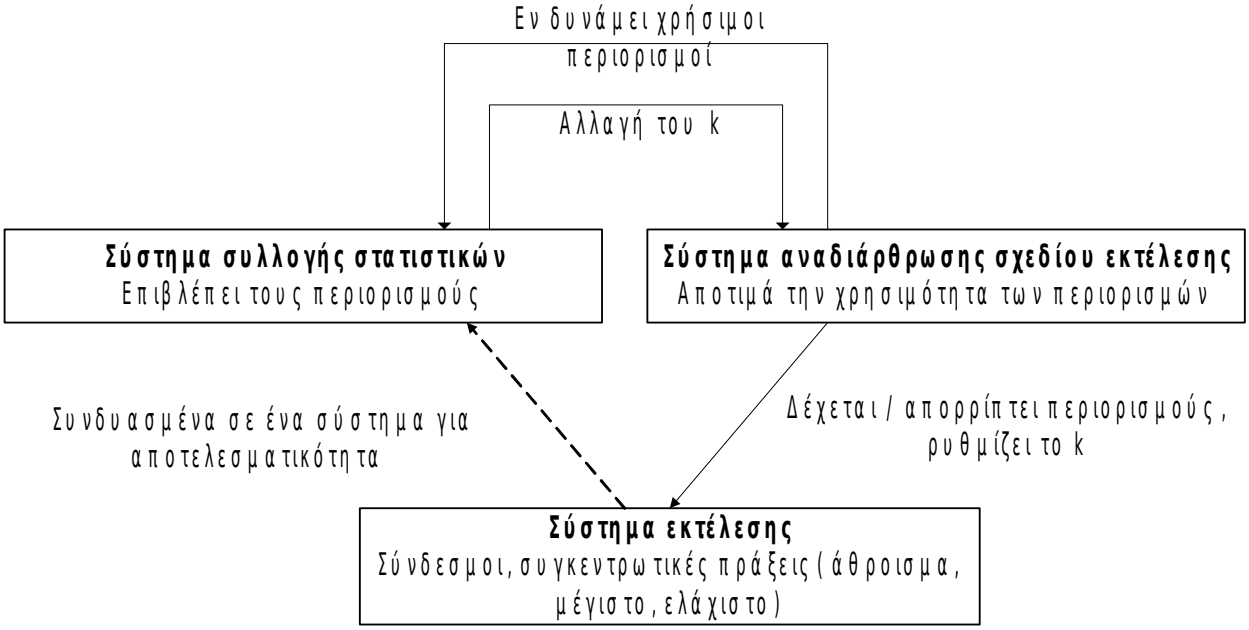
Σχήμα 07 : StreaMon

Η μηχανή αυτή υιοθετεί διάφορες τεχνικές ώστε να αντεπεξέλθει στις προκλήσεις εκτέλεσης ενός συνδέσμου σε ένα δυναμικό περιβάλλον. Η πρώτη από αυτές τις τεχνικές συνίσταται στην μείωση των απαιτήσεων για μνήμη κατά τον χρόνο εκτέλεσης ενός συνδέσμου εκμεταλλευόμενη τα ρεύματα δεδομένων αυτά κάθε αυτά και τα επαναλαμβανόμενα πρότυπα στην άφιξη τους. Μια δεύτερη τεχνική είναι η δυναμική αναδιάταξη των συνδέσμων προς επίτευξη ρευμάτων πολλαπλών συνδέσμων με δυνατά ποιοτικά εχέγγυα που ακολουθούν το σχήμα της διοχέτευσης, προς επίτευξη δηλαδή, αποδοτικών MJoin. Την τρίτη και τελευταία τεχνική αποτελεί η δυναμική αποθήκευση των μερικών αποτελεσμάτων που παράγονται σε κρυφές μνήμες(cache) στα παραλληλισμένα ρεύματα πολλαπλών συνδέσμων προς αποφυγή επαναϋπολογισμού των αποτελεσμάτων αυτών.

Κάθε μία από τις παραπάνω τεχνικές μοιράζονται την ίδια βασική αρχιτεκτονική που συνίσταται σε τρία μέρη, παρόμοια με αυτά του StreaMon του ίδιου. Το πρώτο μέρος είναι η

μηχανή εκτέλεσης της τεχνικής, το δεύτερο αποτελεί την συλλογή στατιστικών δεδομένων και το τρίτο μέρος αναλαμβάνει με βάση τα στατιστικά να αναδιαρθρώσει το σχέδιο υπολογισμού του συνδέσμου που αφορά στην τεχνική αυτή. Τονίζεται πως κάθε ένα από αυτά τα μέρη εξειδικεύεται σε κάθε μία από τις τεχνικές καταλλήλως.

Ξεκινώντας από την πρώτη τεχνική, που αποσκοπεί στην μείωση των απαιτήσεων για μνήμη, η αρχιτεκτονική της φαίνεται στο Σχήμα 4. Αυτή η τεχνική λέγεται k – Mon [06].



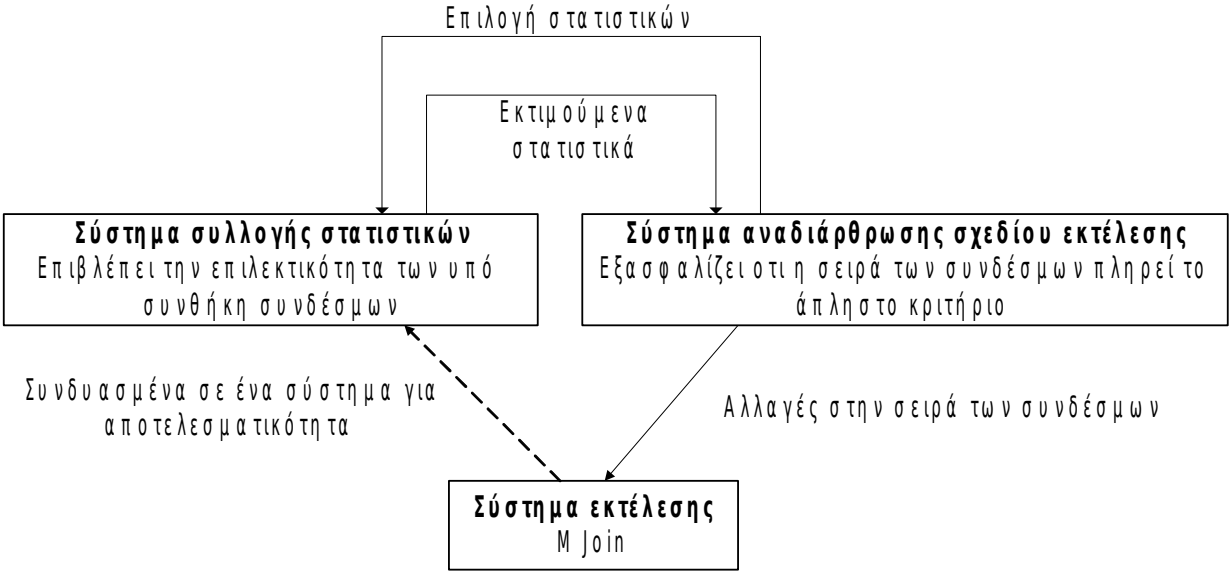
Σχήμα 08 : k - Mon

Επιτυγχάνει το ζητούμενο εισάγοντας έναν αριθμό περιορισμών(k) που το τρέχον ρεύμα δεδομένων πληρεί. Ακριβέστερα, ένα ρεύμα δεδομένων για κάποιο χρονικό διάστημα έχει κάποιες συγκεκριμένες ιδιότητες ή εμφανίζει κάποιο πρότυπο συμπεριφοράς. Αυτές οι ιδιότητες ή η συμπεριφορά εντοπίζονται από ένα σύνολο μηχανισμών που λέγονται περιορισμοί και συμβάλλουν στον περιορισμό της απαιτούμενες μνήμης. Ένα βασικό σύνολο περιορισμών αποτελούν οι σύνδεσμοι ένα σε πολλά, η διάταξη, κι η ακεραιότητα αναφοράς βασισμένη σε ρεύμα δεδομένων. Το StreaMon εφαρμόζει αυτούς τους περιορισμούς και τους εκμεταλλεύεται στον περιορισμό της απαιτούμενης μνήμης σε πράξεις όπως αυτή του συνδέσμου ή συγκεντρωτικών πράξεων όπως είναι το άθροισμα, η εύρεση μεγίστου ή ελαχίστου.

Η δεύτερη τεχνική, αναδιατάζει την σειρά εκτέλεσης των συνδέσμων με βάση κάποιο άπληστο κριτήριο ώστε να μειωθεί ο συνολικός φόρτος εργασίας στις παρούσες συνθήκες. Για αυτό η τεχνική αυτή λέγεται Adaptive – Greedy(A – Greedy) [07]. Το άπληστο κριτήριο είναι αυτό της εκλεκτικότητας των συνδέσμων. Δεδομένου οτι γίνεται λόγος για εκλεκτικότητα σε ένα δυναμικό περιβάλλον, οι σύνδεσμοι για τους οποίους γίνεται λόγος είναι ο MJoin [02]. Συνεχίζοντας, το StreaMon υποστηρίζει τρεις διαφορετικές εκδοχές του αλγορίθμου A – Greedy προκειμένου να συγκεράσει διάφορες αντικρουόμενες απαιτήσεις όπως η βέλτιστη αναδιάταξη, το κόστος υπολογισμού αυτής της αναδιάταξης και η ταχύτητα προσαρμογής στα νέα δεδομένα του περιβάλλοντος. Η αρχιτεκτονική της εν λόγω τεχνικής φαίνεται στο Σχήμα 5.

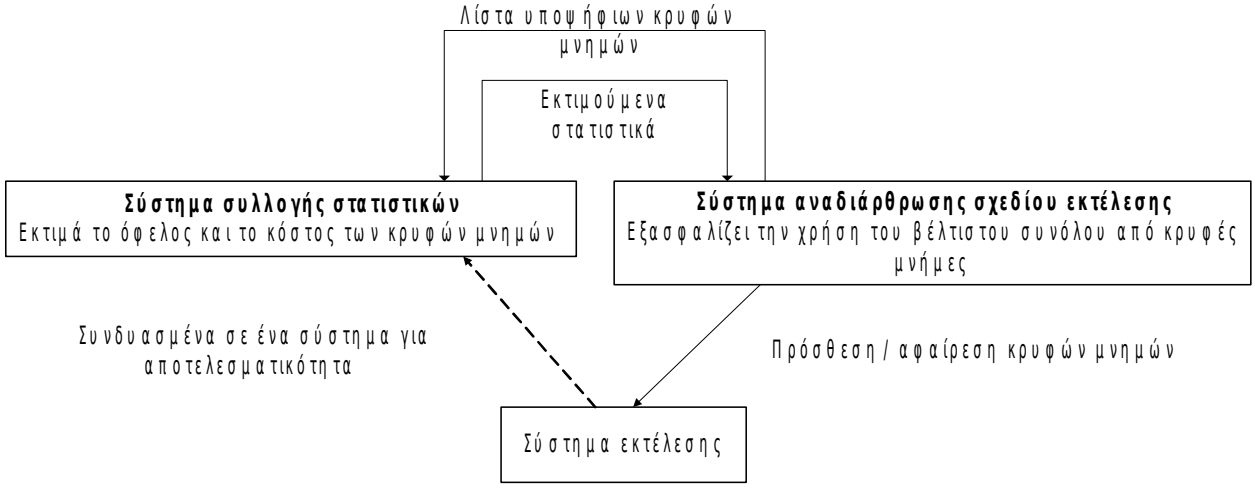
Η τρίτη και τελευταία τεχνική, καλείται Adaptive – Caching(A – Caching)[08] και

δυναμικά



Σχήμα 09 : A - Greedy

προσθέτει κι αφαιρεί κρυφές μνήμες σε MJoin οι οποίες περιέχουν τα διάφορα μερικά αποτελέσματα που παράγονται. Με αυτόν τον τρόπο αποφεύγεται ο επαναϋπολογισμός των ίδιων μερικών αποτελεσμάτων(εγγενής αδυναμία του MJoin). Πέρα από αυτό το όφελος, η τεχνική αυτή, δίνει στο σύστημα την δυνατότητα στους συνδέσμους να υπολογίζονται είτε ως MJoin είτε ως δέντρα συνδέσμων ή οτιδήποτε ανάμεσα τους ανάλογα με τις συνθήκες του περιβάλλοντος ή των ρευμάτων δεδομένων. Η αρχιτεκτονική της τεχνικής αυτής φαίνεται στο εξής σχήμα 6,



Σχήμα 10 : A - Cashing

2.2.3 Τεχνική βασισμένη σε δέντρα

Οι τεχνικές αυτές βασίζονται σε δύο γενικές ιδέες [09]. Η πρώτη αφορά στην αναγωγή ενός συνδέσμου σε ένα άλλο πρόβλημα για το οποίο υπάρχουν αποδοτικοί αλγόριθμοι. Η δεύτερη ιδέα αφορά στην χρήση μιας δομής δεδομένων η οποία αποθηκεύοντας παραπάνω πληροφορία μέσω της δομής της, μπορεί να βοηθήσει στον ταχύτερο υπολογισμό ενός συνδέσμου. Ο υπολογισμός είναι ταχύτερος για τρεις λόγους. Πρώτον, μειώνεται ο φόρτος εργασίας του υπολογισμού, δεύτερον μειώνονται περιττοί επαναϋπολογισμοί και τρίτον μειώνεται ο δαπανόμενος χρόνος για την επεξεργασία του τελικού αποτελέσματος για την απαλοιφή π.χ. πολλαπλών ίδιων εγγραφών αφού αυτές οι εγγραφές είναι λιγότερες με την χρήση των τεχνικών που βασίζονται στην χρήση μιας κατάλληλης δομής όπως τα δέντρα. Τέλος, η νέα δομή δεδομένων, αν απαιτεί παραπάνω χώρο για την αναπαράσταση της, τότε ο χώρος αυτός, δεν είναι κατά ανάγκη πολύ παραπάνω από τον χώρο της προηγούμενης δομής δεδομένων.

Πρακτικά μιλώντας, το πρόβλημα στο οποίο ανάγονται οι διάφοροι σύνδεσμοι είναι αυτό του *μεταβατικού κλεισίματος*(Transitive closure). Επιπροσθέτως, οι σύνδεσμοι που ανάγονται σε αυτό το πρόβλημα είναι μόνο οι μη αναδρομικοί. Αυτό δεν είναι τόσο περιοριστικό όσο φαίνεται, καθώς πολλά από τα υποβαλλόμενα ερωτήματα – σύνδεσμοι είναι τέτοιας φύσης. Για παράδειγμα ο υπολογισμός μιας οποιαδήποτε γραμμικής ακολουθίας σύνθεσης σχέσεων ανάγεται στον υπολογισμό ενός μερικού μεταβατικού κλεισίματος σε έναν γράφο. Οι ακμές του γράφου αυτού συνίστανται από την ένωση των συντιθέμενων σχέσεων. Οι κόμβοι του γράφου αυτού συνίστανται στις διάφορες σταθερές που εμφανίζονται σε κάμποσους συνδέσμους. Για παράδειγμα, για τον σύνδεσμο,

$$\pi_{X,Y}(R_1(X,Z) |\times| R_2(Z,W) |\times| R_3(W,Y))$$

η σειρά υπολογισμού $((R_1 \circ R_2) \circ R_3)$ αντιστοιχεί σε ένα μεταβατικό κλείσιμο που έχει ως κόμβους τις ιδιότητες X και Y των σχέσεων R_1 και R_3 αντίστοιχα. Συνεπώς, αποδοτικοί αλγόριθμοι για τον υπολογισμό του μεταβατικού κλεισίματος μπορούν να εφαρμοστούν για ερωτήματα – συνδέσμους του τύπου επιλογή – προβολή.

Περνώντας στην συζήτηση για την δεύτερη γενική ιδέα, η δομή δεδομένων που εφαρμόζεται για τον αποδοτικό υπολογισμό ενός συνδέσμου, είναι αυτή του *δέντρου*. Ειδικότερα, κατά τον υπολογισμό μιας πλειάδας δεδομένων, ακολουθούνται κάποια βήματα, ακολουθείται ένα μονοπάτι. Το δέντρο χρησιμεύει στην αποθήκευση αυτού του μονοπατιού, αυτής της σειράς υπολογισμών. Μάλιστα, στο δέντρο δεν αποθηκεύεται όλο αυτό μονοπάτι, παρά μόνο εκείνο το τμήμα το οποίο εξασφαλίζει την σύνδεση όλων των αρχικών πλειάδων δεδομένων με τις τελικές. Με αυτό τον τρόπο, αναπαράγονται γρήγορα ήδη υπολογισθέντα αποτελέσματα κι αποφεύγονται πολλαπλές, ίδιες πλειάδες δεδομένων στην έξοδο. Κλείνοντας, όσο περισσότερη είναι η απαιτούμενη μνήμη, τόσο πιο αποδοτική υφίσταται η χρήση δέντρων για την αποφυγή πολλαπλών, ίδιων πλειάδων εξόδου.

2.2.4 Συστήματα με πολλούς επεξεργαστές

Σε ένα σύστημα εκτέλεσης ερωτημάτων – πολλαπλών συνδέσμων υπάρχει ένας βελτιωτικός μηχανισμός ο οποίος με κάποιο τρόπο υπολογίζει ένα βέλτιστο σχέδιο αποτίμησης του συνδέσμου. Όπως ήδη έχει αναφερθεί, το περιβάλλον επιδρά στο σχέδιο αυτό. Η παρούσα λοιπόν λύση, μελετά τρόπους κατασκευής ενός βέλτιστου σχεδίου σε ένα περιβάλλον πολλών επεξεργαστών [10]. Ένα τέτοιο περιβάλλον επιτρέπει τον παραλληλισμό τόσο του ίδιου του ερωτήματος όσο και των διαφόρων, υποβαλλόμενων ερωτημάτων. Ο παραλληλισμός αυτός επιδρά ευεργετικά στην συνολική απόδοση του συστήματος που αναλαμβάνει την εκτέλεση των ερωτημάτων. Όμως αυξάνει την πολυπλοκότητα της εύρεσης του βέλτιστου σχεδίου. Ενδεικτικά, ο χώρος αναζήτησης ενός βέλτιστου σχεδίου αυξάνεται δραματικά. Ακόμη, η αναζήτηση του βέλτιστου σχεδίου δεν γίνεται να περιοριστεί στα γραμμικά, δηλαδή, ακολουθιακά σχέδια, γιατί ένα τέτοιο σχέδιο μπορεί να απέχει κατά πολύ από το βέλτιστο. Ένα γραμμικό σχέδιο για έναν πολλαπλό σύνδεσμο

$$\Sigma 1 \bowtie \Sigma 2 \bowtie ... \bowtie \Sigma_{v-1} \bowtie \Sigma_v$$

εκτελείται ως μια ακολουθία δυαδικών συνδέσμων της μορφής,

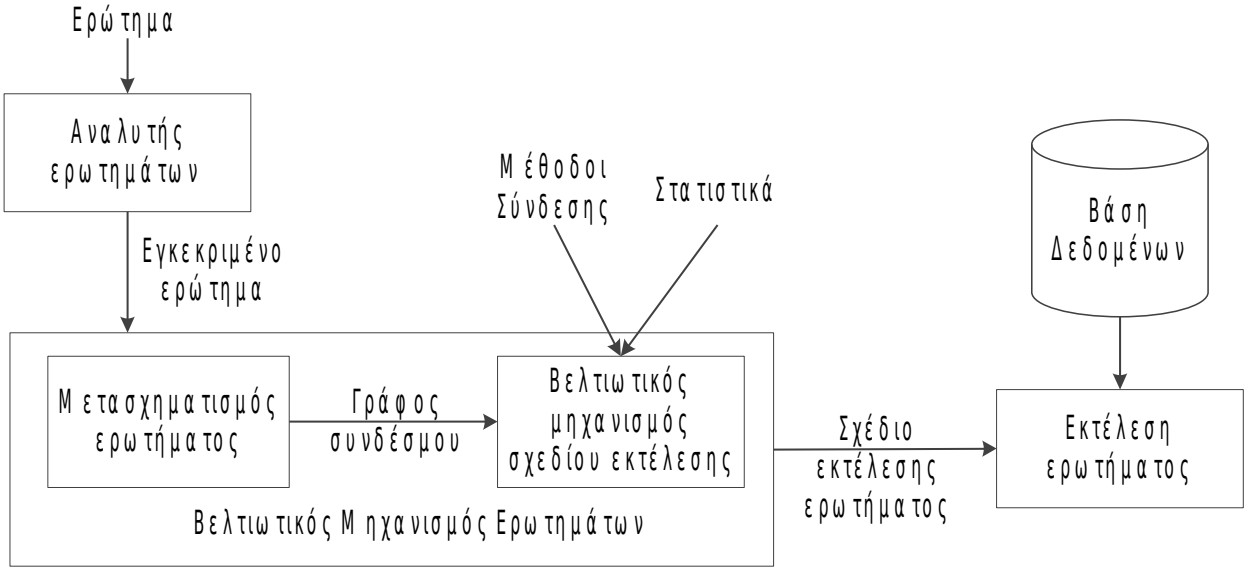
$$(((...(\Sigma_1 \bowtie \Sigma_2) \bowtie \Sigma_3) ...)) \bowtie \Sigma_v).$$

Για την επίλυση αυτής της διπλής πολυπλοκότητας διάφορες λύσεις έχουν προταθεί. Η μία από αυτές περιλαμβάνει την χρήση ευριστικών μεθόδων για τον υπολογισμό μη αναδρομικών ερωτημάτων [11]. Μια άλλη λύση προτείνει την μέθοδο της *προσομοιωμένης ανόπτησης*(simulated annealing)[12],[13] , [14]. Ακόμη, το καρτεσιανό γινόμενο υπό προϋποθέσεις [15] καθώς επίσης κι ο παραλληλισμός κάποιων καλών γραμμικών σχεδίων [16] μπορούν να παράξουν ένα βέλτιστο σχέδιο. Τέλος, επειδή η δομή του γράφου που αναπαριστά το σχέδιο εκτέλεσης επιδρά στην απόδοση του συστήματος, ως λύση έχουν εξεταστεί διάφορες δομές αυτού του γράφου όπως οι αριστερά και δεξιά ετεροβαρείς(left, right deep) δομές καθώς κι η θαμνώδης(bushy) δομή [17].

Περνώντας στην υπό εξέταση λύση, αυτή επικεντρώνεται σε μη αναδρομικά ερωτήματα. Η εξεταζόμενη λύση διαφέρει από τις προαναφερθείσες, στο οτι επιδιώκει να βελτιώσει τόσο την εκτέλεση του ερωτήματος αυτού κάθε αυτού(αυτό - βελτίωση) όσο και την ταυτόχρονη εκτέλεση πολλών ερωτημάτων(βελτίωση). Στο πλαίσιο της αυτό – βελτίωσης κάμποσοι επεξεργαστές αναλαμβάνουν να εκτελέσουν έναν σύνδεσμο. Τελικά, ο προτεινόμενος αλγόριθμος όχι μόνο προσδιορίζει την σειρά εκτέλεσης των συνδέσμων και την μέθοδο που θα πρέπει να ακολουθηθεί για την εκτέλεση αυτή, αλλά επιπλέον ορίζει τον αριθμό των συνδέσμων που θα πρέπει να εκτελεστούν παράλληλα και τον αριθμό των επεξεργαστών που θα αναλάβουν την εκτέλεση ενός συνδέσμου.

Το σύστημα που ενσαρκώνει την τρέχουσα λύση φαίνεται στο Σχήμα 11. Δέχεται ερωτήματα σε μια δηλωτική γλώσσα όπως η SQL και παράγει σχέδια εκτέλεσης ερωτημάτων η εκτέλεση των οποίων “απαντά” στο ερώτημα του χρήστη. Τα ερωτήματα περνούν από έλεγχο προτού ξεκινήσει η διαδικασία εκτέλεσης τους. Αφού περάσουν από τον έλεγχο αυτό, μετασχηματίζονται σε μια σημασιολογικά ισοδύναμη μορφή που αντιλαμβάνεται ο βελτιωτικός μηχανισμός ερωτημάτων. Μια τέτοια μορφή μπορεί να είναι ένας γράφος, ο επονομαζόμενος γράφος συνδέσμου. Στην διάρκεια αυτού του μετασχηματισμού, εφαρμόζονται διάφορες ευριστικές μέθοδοι όπως η ελάττωση της εκλεκτικότητας($\sigma \rightarrow 0$) του συνδέσμου όσο το δυνατόν. Όσο η εκλεκτικότητα ενός συνδέσμου τείνει προς το μηδέν τόσο γρηγορότερα εκτελείται. Έπειτα, ο γράφος συνδέσμου μαζί με τα διάφορα στατιστικά στοιχεία για τις συμμετέχουσες σχέσεις και τις διαθέσιμες μεθόδους για τον υπολογισμό ενός συνδέσμου αποστέλλονται στην καρδιά του συστήματος που είναι ο βελτιωτικός μηχανισμός του σχεδίου εκτέλεσης. Ο μηχανισμός αυτός παράγει το τελικό σχέδιο εκτέλεσης του συνδέσμου σύμφωνα με κάποια κριτήρια βελτίωσης. Σημειώνεται εδώ πως το παραγόμενο σχέδιο δεν συνιστά απαραίτητα το βέλτιστο δυνατό. Μπορεί να αποτελεί απλά το καλύτερο δυνατό σχέδιο

ανάμεσα σε όλα τα υποψήφια.



Σχήμα 11 : Λειτουργικές συνιστώσες στην επεξεργασία ερωτημάτων

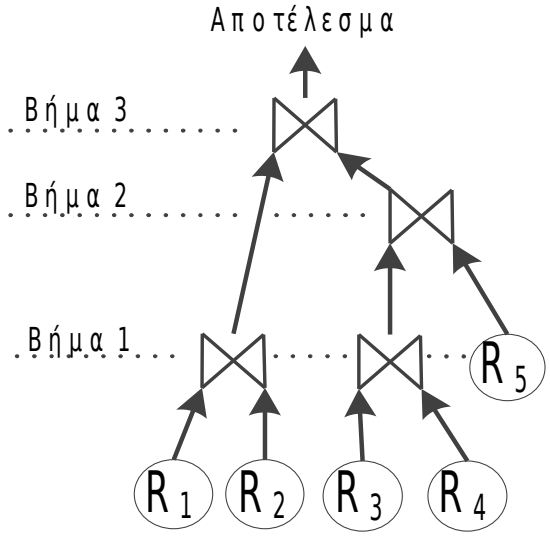
Συνεχίζοντας την ανάλυση, το σύστημα χαρακτηρίζεται γενικού σκοπού και δεν έχει κάποια ειδική σχεδίαση για την βάση δεδομένων. Έχει ένα μικρό πλήθος επεξεργαστών σε σχέση με άλλα συστήματα βάσεων δεδομένων που έχουν εκατοντάδες ή χιλιάδες επεξεργαστών. Οι επεξεργαστές με την σειρά τους μοιράζονται την διαθέσιμη μνήμη, έχοντας μια τοπική μνήμη για τις λειτουργίες εισόδου κι εξόδου τους. Επίσης, οι επεξεργαστές καθορίζουν την μνήμη που απονέμεται σε μια λειτουργία της βάσης δεδομένων όπως ένας σύνδεσμος. Η μνήμη αυτή συνεπώς μπορεί να διαφέρει από λειτουργία σε λειτουργία. Η σχεδιαστική επιλογή αυτή μολονότι δυσχεραίνει την ανάλυση απόδοσης του συστήματος βρίσκεται πιο κοντά στην πραγματικότητα και για αυτό προτιμάται. Επιπλέον, το σύστημα διαχείρισης της βάσης δεδομένων ευθύνεται για τον έλεγχο των επεξεργαστών και της μνήμης που έχουν ανατεθεί για την εκτέλεση ενός συνδέσμου. Ευθύνεται επίσης για την κατανομή των επεξεργαστών κατά την εκτέλεση ενός ερωτήματος και για την αποδοτική χρήση της μνήμης. Ακόμη, το σύστημα λειτουργεί με την υπόθεση οτι οι συμμετέχουσες σχέσεις στον σύνδεσμο δεν χωρούν εν γένει στην μνήμη. Το σύστημα διαθέτει επίσης κοινούς, γενικής χρήσης σκληρούς δίσκους ως δευτερεύουσα μονάδα αποθήκευσης στην οποία αποθηκεύονται οι συμμετέχουσες σχέσεις.

Κλείνοντας την αναφορά στο περιβάλλον για το οποίο προορίζεται η υπό εξέταση λύση, το κριτήριο με βάση το οποίο επιδιώκεται η βελτίωση είναι αυτό του συνολικού χρόνου εκτέλεσης. Ο χρόνος αυτός διαφέρει από τον συνολικό χρόνο επεξεργασίας του ερωτήματος. Η επεξεργασία ενός ερωτήματος περιλαμβάνει τόσο τις λειτουργίες εισόδου κι εξόδου όσο και τις λειτουργίες της κεντρικής μονάδας επεξεργασίας(CPU). Ο χρόνος εκτέλεσης λοιπόν ορίζεται ως ο μέγιστος χρόνος μεταξύ του χρόνου εισόδου – εξόδου και του χρόνου της κεντρικής μονάδας επεξεργασίας. Τέλος, αν ο εφαρμοζόμενος αλγόριθμος έχει διάφορες φάσεις, τότε, το κριτήριο με βάση το οποίο γίνεται η βελτίωση περιλαμβάνει το άθροισμα του χρόνου εκτέλεσης όλων των φάσεων.

Από εδώ και πέρα, παρουσιάζεται ο αλγόριθμος αυτός κάθε αυτός που κάνει πράξη την ιδέα της τρέχουσας μεθόδου επιτάχυνσης του υπολογισμού ενός συνδέσμου. Ξεκινώντας λοιπόν, ο αλγόριθμος χαρακτηρίζεται επαναληπτικός κι εκτελείται σε βήματα. Αυτός, σε κάθε βήμα παράγει κι ένα μέρος του τελικού σχεδίου εκτέλεσης του συνδέσμου. Τα σχέδια εκτέλεσης που παράγονται

είναι συγχρονισμένα με την έννοια οτι για να παραχθεί το επόμενο τους βήμα θα πρέπει να έχει ολοκληρωθεί το προηγούμενο. Ακόμη, τα εν λόγω σχέδια λέγονται θαμνώδη λόγω της δομής που έχουν. Για πέντε σχέσεις, μια από τις δυνατές δομές που μπορεί να έχει ένα σχέδιο απεικονίζεται στο Σχήμα 8.

Όπως φαίνεται και στο Σχήμα 12, η δομή των σχεδίων επιτρέπει να εκτελούνται πολλοί σύνδεσμοι παράλληλα σε κάθε βήμα. Αυτό αντανακλά τον παραλληλισμό ανάμεσα στα διάφορα ερωτήματα που αναφέρθηκε νωρίτερα[3]. Από την άλλη, ο συγχρονισμός παρουσιάζει δύο εν δυνάμει προβλήματα. Το πρώτο είναι οτι υπάρχει ένα επιπρόσθετο κόστος αποθήκευσης κι ανάκλησης των ενδιάμεσων αποτελεσμάτων. Αν δεν υπήρχε ο συγχρονισμός, τα ενδιάμεσα αποτελέσματα θα μπορούσαν να διατεθούν άμεσα για τον υπολογισμό του επόμενου ερωτήματος χωρίς να αποθηκευθούν και μετά να ανακληθούν. Το δεύτερο εν δυνάμει πρόβλημα συνίσταται στο ότι κάποιοι επεξεργαστές μπορεί να περιμένουν αδρανείς ενόσω περιμένουν να τελειώσουν οι υπόλοιποι το έργο τους. Αυτό το πρόβλημα μπορεί να μετριαστεί διαθέτοντας τους επεξεργαστές στους συνδέσμους ανάλογα με τον φόρτο εργασίας.



Σχήμα 12 : Συγχρονισμένο θαμνώδες σχέδιο εκτέλεσης

Ο αλγόριθμος αυτός κάθε αυτός, λέγεται Greedy Parallel(GP) κι έχει χρονική πολυπλοκότητα O(v^5logv) όπου ν το πλήθος των συμμετεχόντων σχέσεων. Η πολυπλοκότητα όμως αυτή αφορά στη χειρότερη δυνατή περίπτωση όπου οι ευριστικοί μηχανισμοί δεν μπορούν να συνεισφέρουν καθόλου στην εξεύρεση της λύσης. Όταν οι ευριστικοί μηχανισμοί αποδίδουν η χρονική πολυπλοκότητα μειώνεται στην O(v^3logv).

Ο αλγόριθμος δέχεται ως είσοδο έναν γράφο συνδέσμου που αναπαριστά τον σύνδεσμο. Ο γράφος αυτός έχεις ως κόμβους τις σχέσεις κι ως ακμές έχει τις συνθήκες μεταξύ των σχέσεων. Ο αλγόριθμος σε κάθε βήμα φτιάχνει ένα βήμα του σύγχρονου σχεδίου εκτέλεσης τους συνδέσμου. Σε κάθε βήμα, κατασκευάζει όσο το δυνατόν περισσότερα ζεύγη από σχέσεις τα οποία συνδέονται παράλληλα. Η απληστία του αλγορίθμου έγκειται στο γεγονός οτι επιδιώκει να συνδέσει όσα περισσότερα ζεύγη μπορεί σε ένα βήμα. Όσα περισσότερα ζευγάρια συνδέει τόσο ταχύτερα εκτελείται ο σύνδεσμος αφού χρειάζονται λιγότερα βήματα. Τελικά σε κάθε βήμα ο αλγόριθμος ζευγαρώνει τόσες σχέσεις ώστε να συμφέρει. Μπορεί σε ένα βήμα να γίνεται να υπάρξουν εξίσου ένα ή τέσσερα για παράδειγμα ζευγάρια. Όμως το κόστος των τεσσάρων ζευγών μπορεί να υπερβαίνει το κόστος του ενός. Για αυτό θα προτιμηθεί μόνο το ένα ζεύγος. Αν όμως τα τέσσερα ζεύγη κοστίζουν λιγότερο, τότε θα προτιμηθούν αυτά.

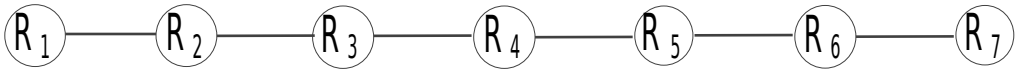
Η επιλογή των ζευγών δεν είναι τυχαία και φυσικά διαφέρει από βήμα σε βήμα. Σε ένα βήμα μπορεί να επιλεχθεί μόνο ένα ζεύγος και σε ένα άλλο να επιλεχθούν τρία. Ο αλγόριθμος λοιπόν, εξετάζει σε κάθε βήμα, επαναληπτικά όλους του δυνατούς συνδυασμούς. Με άλλα λόγια απαντά στο ερώτημα “πόσα ζευγάρια συμφέρει να συνδεθούν παράλληλα σε αυτό το βήμα, ένα, δύο ή περισσότερα”; Για να απαντήσει στο ερώτημα εξετάζει μία – μία όλες τις δυνατές περιπτώσεις. Αρχικά, θεωρεί ότι συμφέρει να συνδεθεί παράλληλα μόνο ένα ζευγάρι σε αυτό το βήμα. Για αυτό υπολογίζει το κόστος όλων των ζευγών μαζί με τα ζεύγη κι από αυτά κρατά το ζεύγος με το

3 Τα ερωτήματα σε αυτό το σημείο δεν είναι άσχετα μεταξύ τους αλλά αποτελούν μέρος ενός μεγαλύτερου ερωτήματος.

μικρότερο κόστος. Μετά, θεωρεί πως συμφέρει να συνδεθούν δύο ζευγάρια σε αυτό το βήμα. Υπολογίζει το κόστος όλων των δυνατών δυάδων από ζεύγη και κρατά εκείνα τα δύο ζευγάρια με το μικρότερο κόστος. Στην συνέχεια επαναλαμβάνει την διαδικασία θεωρώντας συμφέρον να συνδεθούν τρία ζευγάρια σε αυτό το βήμα κι ούτω κάθε εξής. Αν το κόστος των κ ζευγαριών είναι $K_κ$ τότε, μετά από κάθε τέτοια επανάληψη ο αλγόριθμος εξετάζει αν το $K_{κ+1} > K_κ$, συνθήκη (1). Επίσης, εξετάζει αν η επανάληψη όπου υπολογίστηκε το κόστος $K_{κ+1}$ περιλαμβάνει όλα τα εναπομείναντα ζευγάρια, συνθήκη (2). Αν ισχύει μία από τις δύο συνθήκες τότε ο αλγόριθμος σταματά έχοντας υπολογίσει το επιθυμητό σχέδιο. Επιπροσθέτως, τα ζευγάρια που επιλέγονται, επιλέγονται με το κριτήριο η νέα σχέση που θα προκύψει μετά την σύνδεση να έχει το μικρότερο δυνατό μέγεθος.

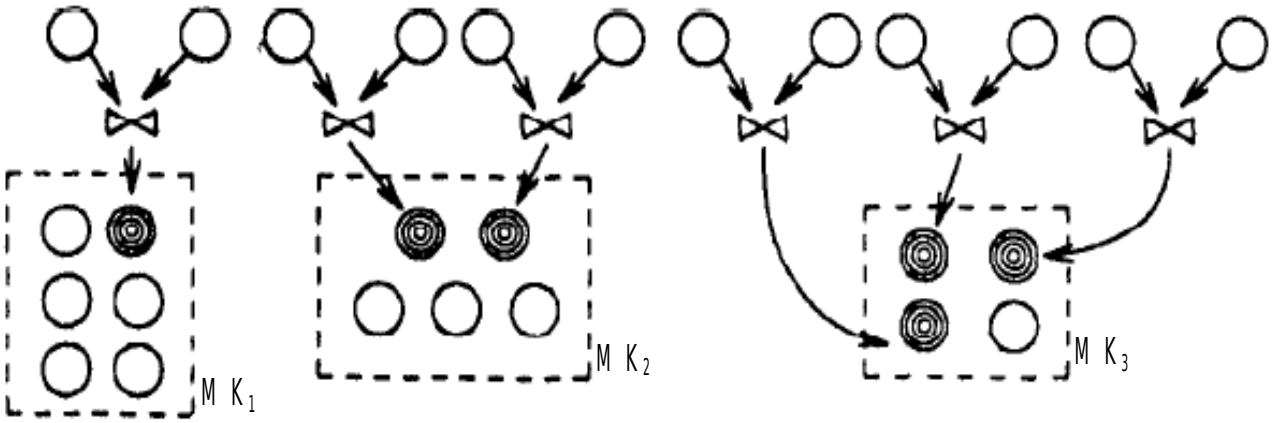
Από την μέχρι τώρα ανάλυση φαίνεται ο καθοριστικός ρόλος του κόστους στην όλη διαδικασία. Όμως η εύρεση του κόστους αυτού δεν γίνεται τετριμμένα καθώς υπάρχουν πάρα πολλά υποψήφια σχέδια για κ ζευγάρια. Επιπλέον, για κάθε σχέδιο υφίστανται πολλές μέθοδοι υπολογισμού του συνδέσμου ενός ζεύγους και διατίθενται πολλοί τρόποι ανάθεσης επεξεργαστών σε έναν σύνδεσμο. Για αυτό το λόγο ο αρχικός αλγόριθμος έχει δύο παραλλαγές ανάλογα με τον τρόπο υπολογισμού του κόστους ώστε να περιοριστεί το κόστος για τον υπολογισμό του κόστους. Η πρώτη λοιπόν παραλλαγή εκτιμά το συνολικό κόστος του σχεδίου εκτέλεσης συνυπολογίζοντας το κόστος όλων των βημάτων του σχεδίου. Για την ακρίβεια, κάθε φορά που αναζητείται το βέλτιστο πλήθος ζευγών, έστω κ, ο γράφος συνδέσμου χωρίζεται σε δύο συνιστώσες. Η μία είναι αυτή των κ ζευγών με κόστος $A_κ$ κι η άλλη είναι οι σχέσεις που περισσεύουν και θεωρείται πως συνδέονται μεταξύ τους γραμμικά με ένα κόστος $B_κ$. Το συνολικό κόστος για τα κ ζεύγη, $K_κ = A_κ + B_κ$. Το $K_κ$ υπολογίζεται για όλα τα δυνατά κ ζεύγη κι από όλα αυτά επιλέγεται το σύνολο των κ ζευγών με το μικρότερο κόστος, $MK_κ$.

Αν ο γράφος συνδέσμου για κάποιες εφτά σχέσεις είναι,

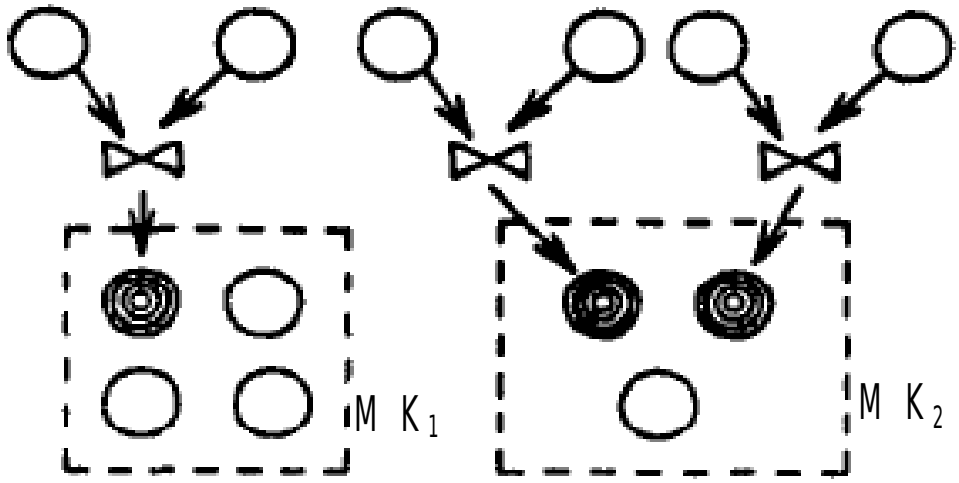


τότε, έστω οτι στο πρώτο βήμα του αλγορίθμου(Σχήμα 13) έχει βρεθεί ότι $MK_1 > MK_2$ και $MK_2 < MK_3$. Βάσει της συνθήκης (1), επιλέγονται τρία ζευγάρια. Στο δεύτερο(Σχήμα 14) επιλέγεται το ένα ζευγάρι θεωρώντας πως ισχύει $MK_1 < MK_2$. Στο τρίτο βήμα(Σχήμα 15) ισχύει $MK_1 > MK_2$ κι επιλέγονται τα δύο ζευγάρια. Ο αλγόριθμος τελειώνει δίνοντας το τελικό σχέδιο υπολογισμού(Σχήμα 16). Στα σχήματα παρακάτω για λόγους απλότητας δεν προσδιορίζονται τα ονόματα των σχέσεων που συμμετέχουν σε κάθε ζεύγος.

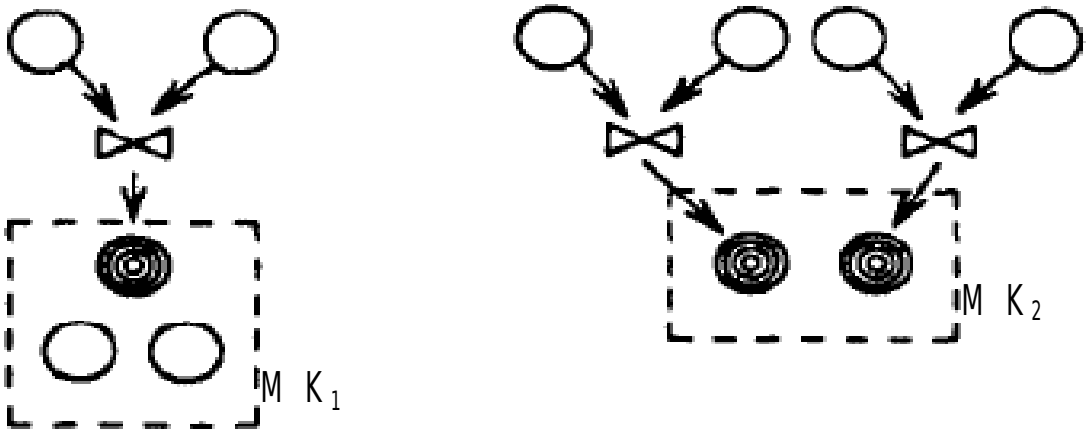
Όπως απεικονίζεται και στα προαναφερθέντα σχήματα, σε κάθε βήμα, οι σχέσεις που συνδέονται, αντικαθίστανται στο επόμενο βήμα από το αποτέλεσμα της σύνδεσης τους κι η διαδικασία επαναλαμβάνεται μέχρις ότου να μείνει μόνο μία σχέση, που συνιστά και το τελικό αποτέλεσμα.



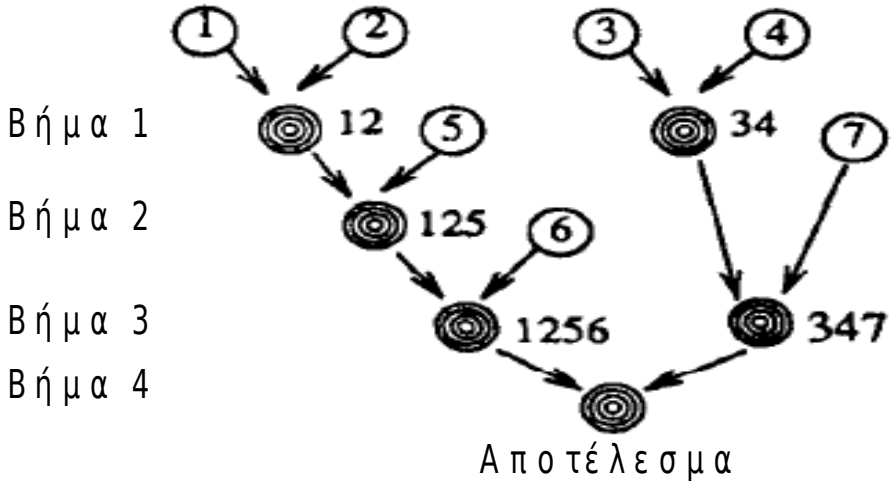
Σχήμα 13 : Greedy Parallel – εκτίμηση συνολικού κόστους, Βήμα 1



Σχήμα 14 : Greedy Parallel – εκτίμηση συνολικού κόστους, Βήμα 2



Σχήμα 15 : Greedy Parallel – εκτίμηση συνολικού κόστους, Βήμα 3



Σχήμα 16 : Greedy Parallel – εκτίμηση συνολικού κόστους, Αποτέλεσμα

Στην δεύτερη παραλλαγή, το κόστος για τα κ + 1 ζεύγη είναι $MK_{\kappa+1} = A_{\kappa+1}$ και για τα κ ζεύγη ισούται με $MK_{\kappa} = A_{\kappa} + E$,

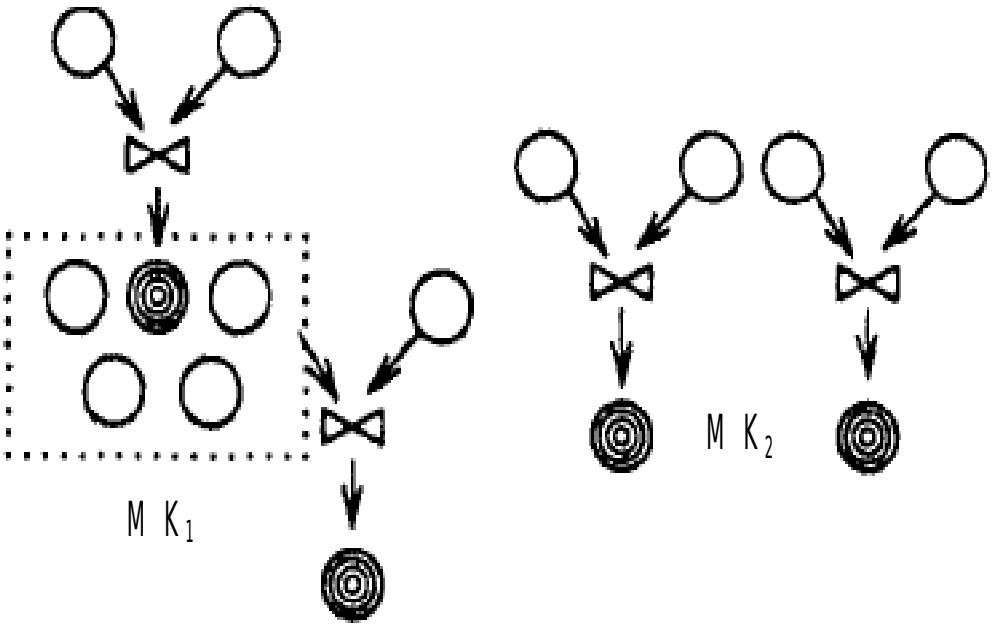
όπου, Ε είναι το ελάχιστο κόστος σύνδεσης δύο σχέσεων, μία από τις εναπομείνασες αρχικές σχέσεις και η άλλη σχέση συνιστά σύνδεση δύο άλλων σχέσεων. Τελικά, το πλήθος των σχέσεων που συνδέονται παράλληλα είναι,

$$\kappa = \begin{cases} 1 & \alpha\nu\, MK_1 < MK_2 \quad [1] \\ N & \alpha\nu\, MK_{\kappa-1} > MK_{\kappa}\, \forall \kappa,\ 1 < \kappa < N \quad [2] \\ \kappa & \alpha\nu\, MK_{\kappa-1} > MK_{\kappa} \wedge MK_{\kappa} < MK_{\kappa+1} \quad [3] \end{cases}$$

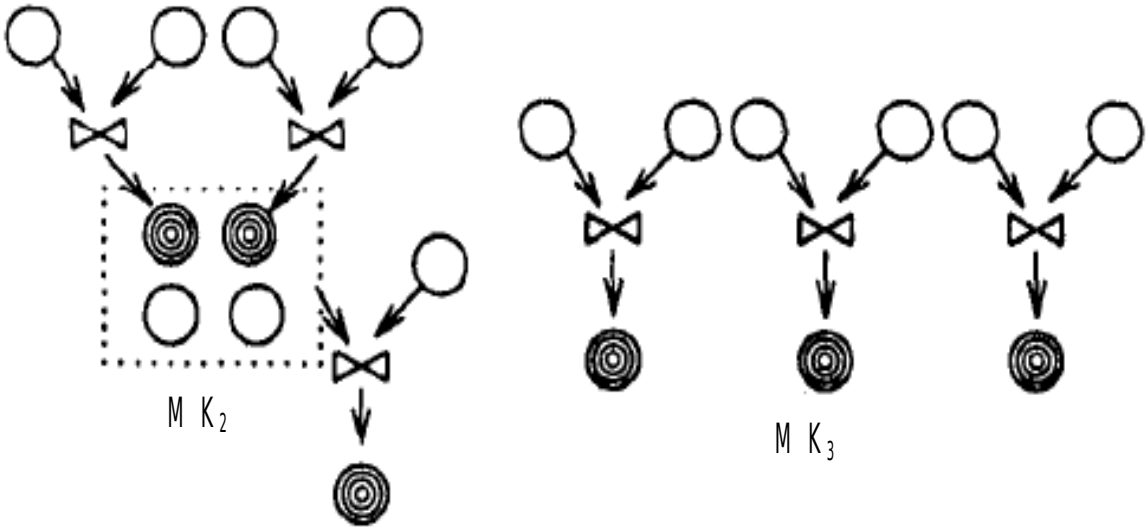
, όπου το N ισούται με το μέγιστο πλήθος ζευγών.

Για τον ίδιο γράφο συνδέσμου, οι εφτά σχέσεις μπορούν να παράξουν το πολύ τρία ζευγάρια. Αν υποτεθεί οτι $MK_1 > MK_2$(1η επανάληψη, Σχήμα 17) και $MK_2 > MK_3$(2η επανάληψη, Σχήμα 18), τότε από την συνθήκη [2], επιλέγονται αυτά τα τρία ζευγάρια. Για το επόμενο βήμα(Σχήμα 19) έχουν απομείνει μία σχέση από τις αρχικές κι οι τρεις νέες που παρήχθησαν. Στο δεύτερο βήμα επομένως, μπορούν υπάρχουν το πολύ δύο ζευγάρια κι αν ισχύει $MK_1 > MK_2$ τότε από την ίδια συνθήκη [2] επιλέγονται αυτά τα δύο ζευγάρια για να εκτελεστούν παράλληλα δίνοντας το τελικό αποτέλεσμα(Σχήμα 20).

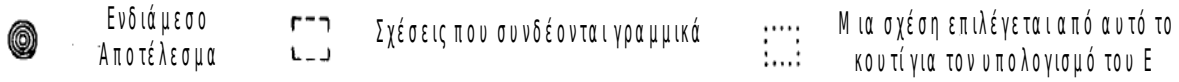
Τελειώνοντας, η πρώτη παραλλαγή δίνει καλύτερα πειραματικά αποτελέσματα. Ακριβολογώντας, η πρώτη παραλλαγή χαρακτηρίζεται πιο εύρωστη σε σχέση με την δεύτερη όταν ο αριθμός των συμμετεχουσών σχέσεων μεταβάλλεται από λίγες ως πολλές, όταν γενικά οι σχέσεις έχουν μεγάλο πλήθος, όταν ο αριθμός των συνδέσμων επίσης μεταβάλλεται. Η πρώτη παραλλαγή έχει επίσης το επιθυμητό χαρακτηριστικό πως ακόμα κι αν δεν βρει το βέλτιστο σχέδιο, βρίσκει ένα πολύ κοντά σε αυτό.

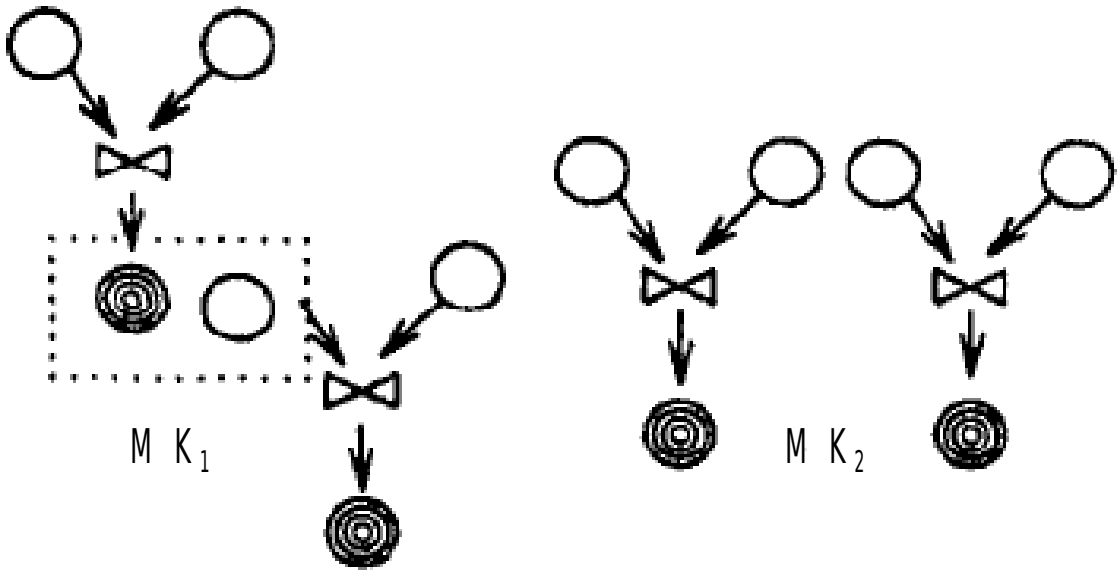


Σχήμα 17 : Greedy Parallel – εκτίμηση μερικού κόστους, Βήμα 1 - Επανάληψη 1 : Σύγκριση MK_1, MK_2

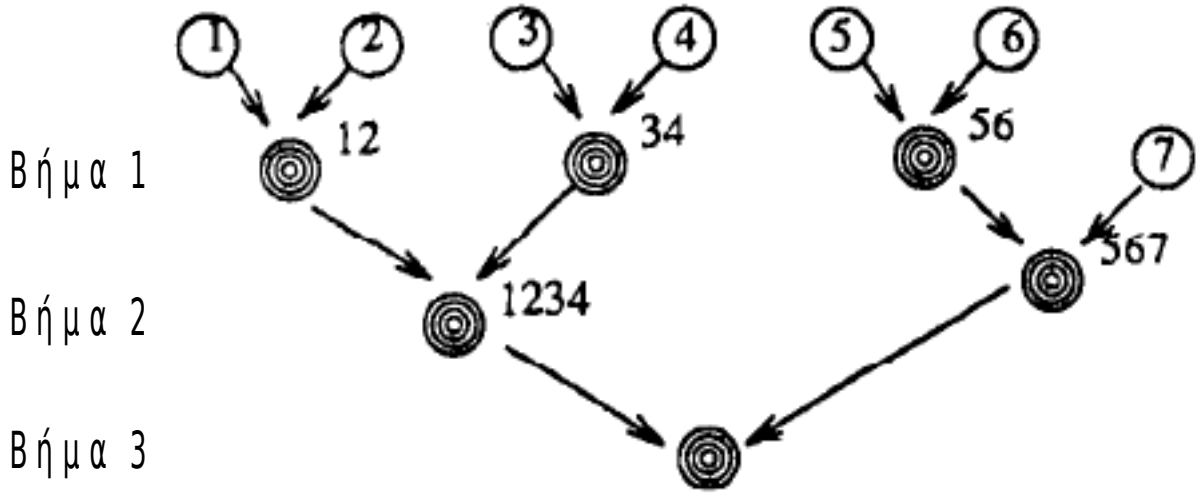


Σχήμα 18 : Greedy Parallel – εκτίμηση μερικού κόστους, Βήμα 1 - Επανάληψη 2 : Σύγκριση MK_2, MK_3





Σχήμα 19 : Greedy Parallel – εκτίμηση μερικού κόστους, Βήμα 2 - Επανάληψη 1 : Σύγκριση MK_1, MK_2



Σχήμα 20 : Greedy Parallel – εκτίμηση μερικού κόστους, Αποτέλεσμα

2.2.5 Βέλτιστος διαμερισμός δεδομένων

Στο σημείο αυτό, πριν από οτιδήποτε, υπενθυμίζεται πως το αντικείμενο διαπραγμάτευσης σε αυτήν την διπλωματική εργασία αποτελεί ο τάχιστος δυνατός υπολογισμός ενός συνδέσμου με έμφαση στον πολλαπλό σύνδεσμο.

Έχοντας αυτό κατά νου, ο βέλτιστος διαμερισμός[4] των δεδομένων [18] που συμμετέχουν φυσικά σε ένα σύνδεσμο συνιστά ένα παράγοντα επιτάχυνσης του εν λόγω συνδέσμου. Η επιτάχυνση οφείλεται σε δύο βασικούς λόγους. Πρώτον, δεδομένου ενός περιβάλλοντος παράλληλης επεξεργασίας, ο διαμερισμός των δεδομένων συμβάλλει στην καλύτερη εκμετάλλευση των επεξεργαστών μέσω της ισορροπημένης κατανομής του φόρτου εργασίας. Δεύτερον, οι απαιτήσεις σε μνήμη μειώνονται κάτι που με την σειρά του αυξάνει την *χωρική συγγένεια*(spatial locality) των δεδομένων και τα οφέλη από αυτήν. Επίσης ελαττώνονται οι εγγραφές από την μνήμη στο δίσκο αφού τα δεδομένα χωρούν σε κρυφές μνήμες πλησίον του επεξεργαστή. Συμπερασματικά, ο διαμερισμός των δεδομένων είναι σημαντικός για τον ταχύ υπολογισμό των δεδομένων και για αυτό θα πρέπει να γίνεται όσο το δυνατόν καλύτερα.

Μερικές από τις μέχρι τώρα τεχνικές διαμερισμού περιλαμβάνουν τον διαμερισμό κατακερματισμού[5](hash partitioning) και τον διαμερισμό εύρους[6](range – partitioning)[19]. Το πρόβλημα αυτών των δύο τεχνικών είναι οτι καμία από τις δύο δεν εγγυάται ένα ανώτατο όριο στο μέγεθος της διαμέρισης. Ειδικά αν τα κλειδιά της διαμέρισης χαρακτηρίζονται “βαριά”, δηλαδή, πολλά δεδομένα έχουν το ίδιο κλειδί, τότε ένα διαμέρισμα μπορεί να έχει πολύ μεγαλύτερο μέγεθος σε σχέση με το μέσο μέγεθος. Κάτι τέτοιο δημιουργεί προβλήματα. Δημιουργεί προβλήματα, αν η πολυπλοκότητα της επεξεργασίας των δεδομένων κάθε διαμερίσματος εξαρτάται υπέρ – γραμμικά από το μέγεθος του διαμερίσματος, όπως σε ένα πρόβλημα ταξιθέτησης. Επίσης, δημιουργεί προβλήματα όταν επιθυμείται ο παραλληλισμός κάποιας εργασίας, αφού οι πόροι του συστήματος διατίθενται ανισόρροπα.

Η εναλλακτική πρόταση προς υπερπήδηση των προαναφερθέντων δυσκολιών αποτελεί μια παραλλαγή του διαμερισμού εύρους. Συγκεκριμένα, ένας πίνακας, μια σχέση N εγγραφών διαμερίζεται σε 2κ + 1 διαμερίσματα δεδομένων κ διακριτών κλειδιών(διαμερισμού). Ακόμη, η παραλλαγμένη τεχνική διαμερισμού σε αντίθεση με την μέχρι τώρα τεχνική χαρακτηρίζεται πιο ευέλικτη, καθώς τα εύρη(διαστήματα) διαμερισμού μπορούν να είναι ανοιχτά, δηλαδή, της μορφής (α, β). Επιπρόσθετα, διατίθενται κ επιπλέον διαμερίσματα αφιερωμένα σε μοναδικά κλειδιά, δηλαδή, κλειδιά που ισούνται με κάποιο άκρο ενός διαμερίσματος. Για παράδειγμα, αν το εύρος ενός διαμερίσματος είναι (α, β), τότε ένα μοναδικό κλειδί θα έχει είτε την τιμή α είτε την τιμή β.

Χάρη στα παραπάνω υπάρχει η δυνατότητα ορισμού ενός ανώτατου ορίου στο μέγεθος διαμερισμάτων του τύπου (α, β) σε σχέση με το πλήθος των εγγραφών σε κάθε διαμέρισμα. Για την ακρίβεια, επιλέγοντας διαμελιστές[7] σε σχέση με τα “βαριά” κλειδιά, το πλήθος των στοιχείων

4 Διαμερισμός δεδομένων λέγεται το μοίρασμα των στοιχείων ενός διαμερίσματος(συνόλου) σε μικρότερα διαμερίσματα(σύνολα). Εν παραδείγματι, αν το αρχικό διαμέρισμα αποτελεί ένα πίνακα, τότε, ο διαμερισμός του έχει ως αποτέλεσμα ένα σύνολο μικρότερων πινάκων. Το μοίρασμα των στοιχείων γίνεται με βάση κάποιο κριτήριο, που μπορεί να είναι κάποια ιδιότητα των διαμεριζόμενων στοιχείων. Η ιδιότητα αυτή λέγεται κλειδί διαμερισμού ή απλά κλειδί. Ανάλογα με την ακολουθούμενη τεχνική ορίζεται και το κλειδί.

5 Τα δεδομένα διαμερίζονται σύμφωνα με την τιμή μιας συνάρτησης κατακερματισμού(hash function). Το κλειδί είναι η τιμή της εν λόγω συνάρτησης κατακερματισμού.

6 Εδώ εξετάζεται αν το κλειδί ανήκει σε κάποιο συγκεκριμένο εύρος και με βάση αυτό, το αντίστοιχο στοιχείο εισάγεται στο κατάλληλο διαμέρισμα. Αν το κλειδί ανήκει στο διάστημα [α, β] τότε το στοιχείο εισάγεται στο διαμέρισμα έστω Δ_1, αν το κλειδί ανήκει στο διάστημα [γ, δ] τότε το στοιχείο πάει στο διαμέρισμα έστω Δ_2 κι ούτω κάθε εξής.

7 Το κλειδί αφορά σε ένα στοιχείο ορίζοντας που θα διανεμηθεί το συγκεκριμένο στοιχείο. Ο διαμελιστής ορίζει τα διαμερίσματα. Στις εκλογές για παράδειγμα, το επίθετο κάθε ψηφοφόρου συνιστά το κλειδί με βάση το οποίο

κάθε διαμερίσματος δεν ξεπερνά τα $\lceil \frac{N-\kappa}{\kappa+1} \rceil$ στοιχεία στην χειρότερη περίπτωση. Ακόμη για συνδέσμους σταθερού πλήθους εγγραφών επειδή το κλειδί με το οποίο γίνεται η σύνδεση των εγγραφών, ταιριάζει με το κλειδί του διαμερισμού, δεν χρειάζεται να εξεταστεί κάθε εγγραφή χωριστά για να συνδεθεί με τις υπόλοιπες. Το διαμέρισμα μπορεί κατευθείαν να μεταφερθεί στην μνήμη. Με άλλα λόγια ένας σύνδεσμος μπορεί να υπολογιστεί κατευθείαν διαμέσου του διαμερισμού του αρχικού συνόλου δεδομένων λόγω του γεγονότος οτι έχουν κοινά βήματα εργασίας.

Στο σημείο αυτό μπορεί κάποιος να αναρωτηθεί για το κόστος εύρεσης των διαμελιστών μια κι οι διαμελιστές φαίνεται να καθορίζουν το πλήθος των στοιχείων σε ένα διαμέρισμα κι άρα να ευθύνονται για τα όποια πλεονεκτήματα της υπό συζήτηση τεχνικής. Απαντώντας στο ερώτημα αυτό, ο υπολογισμός των κατάλληλων διαμελιστών είναι αμελητέο. Στην χειρότερη περίπτωση ανέρχεται στο $O(\kappa lg^2 N)$. Πέρα όμως από το χαμηλό κόστος εύρεσης τους, οι διαμελιστές εξυπηρετούν με έναν ακόμη τρόπο. Όταν τα δεδομένα έχουν ένα μικρό πλήθος από “βαριά” κλειδιά, οι συνήθεις τεχνικές τείνουν να φτιάχνουν μεγάλα διαμερίσματα δεδομένων που περιλαμβάνουν τα ίδια τα “βαριά” κλειδιά. Κάτι τέτοιο όμως όπως ήδη ειπώθηκε δεν επιθυμείται. Άλλωστε μπορεί η μνήμη να μην το επιτρέπει όταν τα δεδομένα είναι πάρα πολλά. Δίνοντας ένα παράδειγμα, αν η είσοδος(ταξιθετημένη πάντα) είναι 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 4, 5, 6, 7, 8, τότε οι παραδοσιακές τεχνικές θα επέλεγαν ως διαμελιστές τα 2, 2, 5 και κατόπιν απαλοιφής των πολλαπλών, ίδιων διαμελιστών θα επέλεγαν τους διαμελιστές 2, 5. Όπως φαίνεται, αυτή η επιλογή οδηγεί σε μεγάλα διαμερίσματα. Η εξεταζόμενη τεχνική ωστόσο, θα επέλεγε ως διαμελιστές τα 1, 2, 6 κάτι που συνεπάγεται μικρότερο μέγεθος διαμερισμού. Ένα τελευταίο σημείο επί του θέματος αυτού αποτελεί το γεγονός οτι οι μέχρι τώρα τεχνικές δεν εγγυώνται για το αποτέλεσμα όταν εντός ενός διαμερίσματος υφίστανται πολλαπλές ίσες τιμές. Για αυτό η όποια επεξεργασία των μεγάλων διαμερισμάτων που δημιουργούν γίνεται με την υπόθεση ότι εμπεριέχονται τουλάχιστον δύο διαφορετικές τιμές. Η νέα προτεινόμενη τεχνική δεν υποφέρει από αυτό το πρόβλημα αφού υποστηρίζει διαμερίσματα μοναδικών κλειδιών.

Κλείνοντας με τα πλεονεκτήματα της εν λόγω τεχνικής, παρατίθενται μερικά ακόμα. Η τεχνική αυτή χαρακτηρίζεται εύρωστη σε σχέση με τα “βαριά” κλειδιά σε ένα περιβάλλον βάσης δεδομένων όπου τα δεδομένα ακολουθούν μια τυχαία κατανομή. Επιπλέον, χαρακτηρίζεται εύρωστη και σε σχέση με την οποιαδήποτε κατανομή των δεδομένων. Τέλος, το σύνολο των διαμελιστών εμφανίζει τις στατιστικές ιδιότητες ενός ισοβαθούς ιστογράμματος κάτι που χρησιμεύει στον προσεγγιστικό υπολογισμό ενός συνδέσμου.

Προχωρώντας, οι διαμελιστές ισούνται με κ κι ορίζουν 2κ + 1 διαμερίσματα. Τα κλειδιά διαμερισμού έχουν τέτοιο τύπο ώστε να μπορούν να ταξιθετηθούν. Γνωστοί τύποι με διάταξη συνιστούν οι πραγματικοί αριθμοί και τα αλφαριθμητικά(string). Δεδομένων λοιπόν αυτών των κ διαμελιστών, $s_1, s_2, \ldots, s_\kappa$ ορίζονται 2κ + 1 διαμερίσματα εκ των οποίων,

- κ από αυτά αφορούν σε μοναδικά κλειδιά, λέγονται διαμερίσματα ισότητας έχοντας την μορφή $\{ \chi \mid \chi = s_i \}$, $i = 1, \ldots, \kappa$
- κ – 1 από αυτά λέγονται διαμερίσματα ανισότητας έχοντας την μορφή $\{ \chi \mid s_i < \chi < s_{i+1} \}$, $i = 1, \ldots, \kappa - 1$
- 2 ακόμα διαμερίσματα ανισότητας της μορφής $\{ \chi \mid \chi < s_1 \}$ και $\{ \chi \mid \chi > s_\kappa \}$.

Τελειώνοντας, παρουσιάζεται ο τρόπος υπολογισμού των διαμελιστών για πολλαπλούς πίνακες – σχέσεις, δηλαδή, για έναν πολλαπλό σύνδεσμο. Η παρουσίαση αυτή γίνεται μέσω

ψηφίζει σε ένα εκλογικό κέντρο. Τα εκλογικά κέντρα όμως καθορίζονται από τους εκλογικούς διαμελιστές.

παραδείγματος για δύο σχέσεις Σ_1, Σ_2 που μοιράζονται μια κοινή ιδιότητα – στήλη I. Η Σ_2 έχει μικρότερο μέγεθος από την Σ_1. Αν ο σύνδεσμος των σχέσεων αυτών απαιτείται συχνά, τότε είναι γενικά καλή η εκμετάλλευση ενός προϋπολογισμένου συνόλου διαμελιστών εξομαλύνοντας κάποια πλευρά του υπολογισμού του συνδέσμου. Στην παρούσα παρουσίαση δίνεται έμφαση σε διαμερίσματα ανισότητας και θεωρώντας πως ο διαμερισμός αποτελεί ένα προηγούμενο βήμα πριν από έναν σύνδεσμο κατακερματισμού(hash join).

Έχοντας πει όλα τα παραπάνω, ένας εύλογος στόχος θα ήταν οι διαμελιστές να παράγουν διαμερίσματα των οποίων το μέγεθος δεν ξεπερνά το μέγεθος κάποιας μνήμης στην ιεραρχία των μνημών, π.χ. της L_2. Συνεπώς, κάποιος θα μπορούσε να βρει ένα σύνολο διαμελιστών για την μικρότερη από τις δύο σχέσεις(Σ_2) προκειμένου να διαμελίσει και τις δύο σχέσεις τελικά. Ο λόγος που προτιμάται η μικρότερη σχέση έγκειται στο γεγονός ότι αφενός ο χρόνος παραγωγής μιας πλειάδας δεδομένων υστερεί έναντι του χρόνου παραγωγής μιας εγγραφής κι αφετέρου απαιτείται λιγότερη μνήμη. Με αυτόν τον τρόπο εξασφαλίζεται ότι ακόμα κι αν το μέγεθος των διαμερισμάτων ποικίλει για την Σ_1, τα διαμερίσματα για την Σ_2 έχουν φραχτεί ως προς το μέγεθος.

Η παραπάνω προσέγγιση όμως μειονεκτεί. Δεν αρκεί ο υπολογισμός διαμελιστών για την Σ_2. Γιατί; Πρώτον, γιατί κάποια από τα διαμερίσματα της Σ_1 μπορεί να υπολείπονται αυτών της Σ_2. Κάτι τέτοιο σημαίνει πως το τρέχον σύνολο διαμελιστών δεν αποτελεί το βέλτιστο κι ότι έχει γίνει περισσότερο λεπτομερής διαμερισμός από όσο χρειαζόταν. Ένας πιο “χοντροκομμένος” διαμερισμός θα ήταν καλύτερος. Δεύτερον, αυτός ο τρόπος διαμερισμού μπορεί να οδηγήσει σε πολύ ανομοιόμορφα διαμερίσματα αν η κατανομή των τιμών της ιδιότητας I διαφέρει πολύ από σχέση σε σχέση. Κάτι τέτοιο δεν επιθυμείται καθώς σε επίπεδο παραλληλισμού μια τέτοια ανομοιομορφία επιφέρει μεγάλες καθυστερήσεις ή[8] αφήνει ανεκμετάλλευτους τους επεξεργαστές.

Κλείνοντας με τα θεωρητικά, ένα κατάλληλο σύνολο διαμελιστών λαμβάνει υπ' όψιν όλες τις τιμές της ιδιότητας I κι από τις δύο σχέσεις Σ_1, Σ_2. Οι τιμές αυτές θεωρούνται ταξιθετημένες. Θέτοντας N_1, N_2 τα μεγέθη των σχέσεων – πινάκων Σ_1, Σ_2 αντίστοιχα, τότε με $\tau_1[\,0...\,N_1-1]$, $\tau_2[\,0...\,N_2-1]$ συμβολίζονται οι ταξιθετημένες λίστες των στοιχείων της ιδιότητας I για τις Σ_1, Σ_2 αντίστοιχα. Τελικά, μπορεί να προκύψει μια αναπαράσταση του κόστους για τα διαμερίσματα των Σ_1, Σ_2. Το μέγεθος των διαμερισμάτων αυτών είναι δ_1 και δ_2 αντίστοιχα. Η εξίσωση του κόστους αυτού είναι,

$$\kappa\acute{o}\sigma\tau o\varsigma(\delta_1,\delta_2) = \kappa\acute{o}\sigma\tau o\varsigma\_\pi\acute{\iota}\nu\alpha\kappa\alpha(min(\delta_1,\delta_2)) + max(\delta_1,\delta_2)\,\kappa\acute{o}\sigma\tau o\varsigma\_\sigma\acute{\upsilon}\nu\delta\varepsilon\sigma\eta\varsigma(min(\delta_1,\delta_2))$$

όπου στο πλαίσιο υπολογισμού ενός συνδέσμου με την μέθοδο του κατακερματισμού, το κόστος_πίνακα ορίζεται ως το κόστος κατασκευής του πίνακα κατακερματισμού και το κόστος_σύνδεσης ορίζεται ως το κόστος σύνδεσης των δύο πινάκων. Η παραπάνω εξίσωση του κόστους αποτελεί μια απλουστευμένη εκδοχή του συνολικού κόστους. Για παράδειγμα το κόστος_σύνδεσης και το κόστος_πίνακα θα μπορούσαν να εξαρτώνται κι από κάποιες άλλες παραμέτρους όπως το μέγεθος του πίνακα κατακερματισμού για το κόστος_σύνδεσης. Σε κάθε περίπτωση όμως η παραπάνω εξίσωση κόστους έχει την βασική ιδιότητα μιας συνάρτησης κόστους. Η συνάρτηση είναι μονότονη ως προς το μέγεθος των διαμερισμάτων. Κάθε φορά που προστίθεται μια εγγραφή είτε στην φάση κατασκευής του πίνακα κατακερματισμού είτε στην φάση της σύνδεσης, το κόστος πρέπει να αυξάνεται αλλιώς να παραμένει ως έχει. Συμπερασματικά, ένα βέλτιστο σύνολο διαμελιστών οφείλει να έχει το ελάχιστο κόστος από κάθε άλλο δυνατό σύνολο όπως το κόστος υπολογίζεται με μια κατάλληλη εξίσωση κόστους.

Όσον αφορά στην πειραματική μελέτη του αλγορίθμου, αυτή επιβεβαιώνει την πολυπλοκότητα $O(\kappa lg^2 N)$. Επίσης, για μέγεθος εισόδου 1GB, το πολύ σε ένα δευτερόλεπτο υπολογίζεται ένα κατάλληλο πλήθος διαμελιστών καθιστώντας την όλη διαδικασία εύρεσης των

8 Η χρήση του διαζευκτικού “ή” σε μια πρόταση A ή B, σημαίνει είτε το A είτε το B ή και τα δύο. Η περίπτωση του είτε το A είτε το B αλλά όχι και τα δύο θα αναφέρεται ρητά.

διαμελιστών μια φθηνή εργασία βάσης δεδομένων. Επιπλέον, η αποθήκευση των διαμελιστών επιβαρύνει την μνήμη κατά 0,00076% κάτι που καθιστά την μέθοδο φθηνή κι ως προς την μνήμη. Αυτό επιτρέπει την ύπαρξη πολλών διαφορετικών προϋπολογισμένων διαμελιστών για διαφορετικές ιδιότητες ταξιθέτησης, γεγονός που επιταχύνει κατά πολύ την εκ νέου ταξιθέτηση των δεδομένων. Επιπρόσθετα επιβεβαιώνεται και πειραματικά η ευρωστία των διαμελιστών έναντι των βαριών κλειδιών πληρώνοντας ωστόσο ένα μικρό τίμημα κάποια διαστήματα ανισότητας να είναι άδεια. Ακόμη, ο εξεταζόμενος αλγόριθμος χαρακτηρίζεται πολύ αποδοτικός ανεξάρτητα της κατανομής της εισόδου, αν αποτελεί ομοιόμορφη κατανομή ή οποιαδήποτε άλλη.

Ο αλγόριθμος ωστόσο κοστίζει αρκετά καθώς το πλήθος των διαμελιστών αυξάνεται. Αυτό έγκειται στο γεγονός πως ο αλγόριθμος εξαρτάται από την θέση στην ιεραρχία μνήμης που έχουν οι δομές διαμελισμού όπως οι διαμελιστές και τα σημεία αρχής των διαμερισμάτων. Όταν το πλήθος των διαμελιστών αυξάνεται τότε αναγκαστικά ένα μέρος αυτών αποθηκεύεται εκτός της κρυφής μνήμης. Κατά επέκταση σε μια ομοιόμορφη κατανομή της εισόδου συμβαίνουν πολλές αστοχίες στην κρυφή μνήμη τόσο για την εύρεση του κατάλληλου διαμελιστή όσο και για την εύρεση του κατάλληλου διαμερίσματος. Συνεπώς, η απόδοση του αλγορίθμου μειώνεται δραματικά.

2.2.6 Χρήση δικτυακών υπηρεσιών

Ένας άλλος τρόπος για τον αποδοτικό υπολογισμό ενός συνδέσμου αποτελεί η χρήση δικτυακών υπηρεσιών ή υπηρεσιών δικτύου που από εδώ και στο εξής θα αναφέρονται ως υπηρεσίες [20]. Ο τρόπος αυτός ειδικεύεται στους συνδέσμους του τύπου επιλογή – προβολή. Σε γλώσσα SQL αυτά τα ερωτήματα – σύνδεσμοι γράφονται,

SELECT I_{Π}

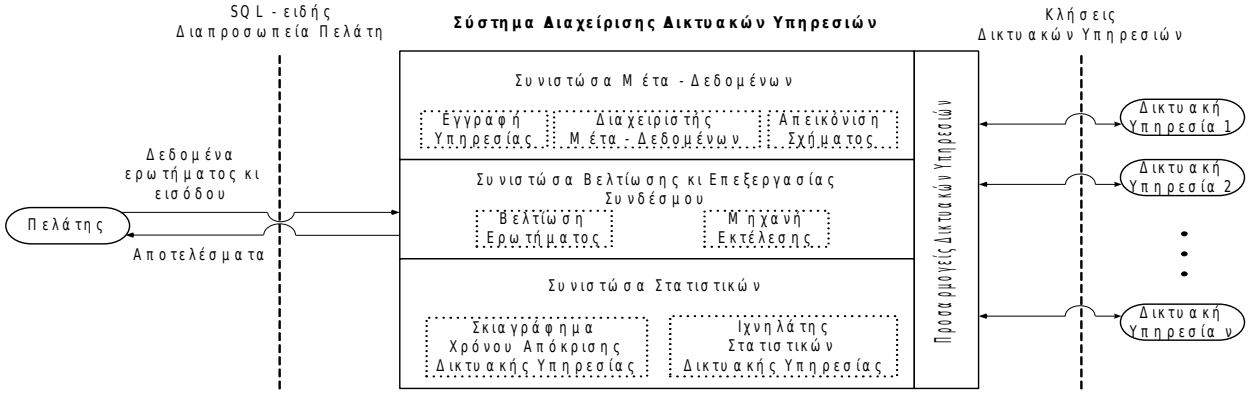
FROM $E(I_E) \bowtie Y_1(X_1^{\gamma}, Y_1^{\pi}) \bowtie \ldots \bowtie Y_N(X_v^{\gamma}, Y_v^{\pi})$

WHERE $K_1(I_1) \wedge K_2(I_2) \wedge \ldots \wedge K_{\mu}(I_{\mu})$

όπου, το I_{Π} αποτελεί το σύνολο των ζητούμενων – προβαλλόμενων ιδιοτήτων, I_E είναι το σύνολο των ιδιοτήτων των δεδομένων εισόδου E και K_1, ..., K_{μ} είναι κατηγορήματα που εφαρμόζονται επί των ιδιοτήτων I_1, ..., I_{μ}. Οι δείκτες γ και π στις ιδιότητες X και Y σημαίνουν πως εφόσον η αντίστοιχη υπηρεσία Y αναπαρίσταται ως ένας εικονικός πίνακας τότε, οι τιμές της ιδιότητας X πρέπει να είναι γνωστές ενώ οι τιμές της ιδιότητας Y παράγονται και για αυτό δεν χρειάζεται να είναι γνωστές. Η σημειολογία αυτή προέρχεται από την δημοσίευση [21]. Όπως φαίνεται η κατηγορία των συνδέσμων στους οποίους αφορά η υπό εξέταση μέθοδος χαρακτηρίζεται ευρεία.

Αναλυτικότερα, οι εν λόγω υπηρεσίες εκτελούν έναν σύνδεσμο στο περιβάλλον ενός συστήματος διαχείρισης δικτυακών υπηρεσιών(ΣΔΔΥ – Web Server Management System – WSMS, Σχήμα 17). Ένα τέτοιο σύστημα, συντονίζει την λειτουργία αυτών των υπηρεσιών και διαμεσολαβεί ανάμεσα στις υπηρεσίες και στον πελάτη. Επιγραμματικά, ένα σύστημα διαχείρισης υπηρεσιών, συνίσταται σε τρεις βασικές συνιστώσες. Η πρώτη συνιστώσα ασχολείται με τα μετά – δεδομένα, την εγγραφή των υπηρεσιών και προσφέρει στον πελάτη ένα γραφικό περιβάλλον. Μέσα από το γραφικό περιβάλλον, ο πελάτης έχει μια οπτική αναπαράσταση της δομής των υπηρεσιών. Επίσης, ο πελάτης μέσω μιας διαπροσωπείας του ίδιου γραφικού περιβάλλοντος, που μοιάζει με την SQL, μπορεί να θέτει ερωτήματα – συνδέσμους στο σύστημα. Η δεύτερη συνιστώσα αναλαμβάνει την εκτέλεση του συνδέσμου, όπως αυτός τίθεται μέσω της πρώτης συνιστώσας, μέσω ανάκλησης των κατάλληλων υπηρεσιών. Επιπλέον, βελτιώνει το σχέδιο εκτέλεσης του συνδέσμου με την χρήση στατιστικών. Η τρίτη συνιστώσα σκιαγραφεί τις υπηρεσίες ως προς τον χρόνο απόκρισης τους, διατηρεί σχετικά στατιστικά στοιχεία για τα δεδομένα κάθε υπηρεσίας για όσο πιο μακρό χρονικό διάστημα γίνεται. Αυτή η συνισταμένη του συστήματος συμβάλλει στην αναδιάρθρωση του σχεδίου υπολογισμού του συνδέσμου.

Ολοκληρώνοντας την εικόνα για ένα ΣΔΔΥ, οι διάφορες υπηρεσίες αποτελούν διεπαφές που



Σχήμα 21 : WSMS, Σύστημα Διαχείρισης Δικτυακών Υπηρεσιών

μοιάζουν με συναρτήσεις Α → Β όπου τα Α, Β είναι σύνολα ιδιοτήτων. Οι υπηρεσίες δέχονται τιμές για τις ιδιότητες του συνόλου Α και παράγουν τιμές για τις ιδιότητες του συνόλου Β. Επομένως, ο τρόπος λειτουργίας ενός ΣΔΔΥ μοιάζει με ένα ρεύμα εργασιών ή το σχήμα της διοχέτευσης. Εδώ, το ΣΔΔΥ δέχεται δεδομένα εισόδου από τον πελάτη τα οποία επεξεργάζεται μέσω μιας ακολουθίας από υπηρεσίες. Η έξοδος από κάθε υπηρεσία επιστρέφεται στο ΣΔΔΥ κι εξυπηρετεί ως είσοδος για την επόμενη υπηρεσία στην ακολουθία. Τελικά, υπολογίζεται ο σύνδεσμος. Λειτουργίες των υπηρεσιών αποτελούν η απόρριψη άσχετων δεδομένων με τον σύνδεσμο, ο μετασχηματισμός των δεδομένων κι η προσθήκη επιπλέον πληροφορίας σε κάθε ένα δεδομένο[9].

Στον υπολογισμό ενός συνδέσμου με αυτόν τον τρόπο συναντώνται τέσσερις βασικές προκλήσεις. Μία από αυτές είναι το γεγονός ότι οι υπηρεσίες διαφέρουν ως προς τον χρόνο απόκρισης τους. Επιπλέον διαφέρουν κι ως προς τον λόγο των πλειάδων εξόδου που παράγουν προς τις πλειάδες εισόδου που δέχονται. Με άλλα λόγια διαφέρουν ως προς την εκλεκτικότητα τους. Αυτό συνεπάγεται πως διαφορετικές διατάξεις των υπηρεσιών στην διοχέτευση μπορούν να έχουν σημαντικά διαφορετικούς, συνολικούς ρυθμούς επεξεργασίας. Για την βέλτιστη διάταξη των υπηρεσιών ευθύνεται ο βελτιωτικός μηχανισμός.

Μια δεύτερη πρόκληση απορρέει από τις εξαρτήσεις που ενδεχομένως υφίστανται ανάμεσα στις υπηρεσίες. Οι εξαρτήσεις αυτές ονομάζονται περιορισμοί προτεραιότητας. Σε αυτή την περίπτωση ο βελτιωτικός μηχανισμός οφείλει να βρει την βέλτιστη διάταξη δεδομένων των περιορισμών προτεραιότητας.

Η τρίτη πρόκληση έγκειται στο ενδεχόμενο, μια γραμμική διάταξη των υπηρεσιών στην διοχέτευση να μην είναι η βέλτιστη δυνατή. Για παράδειγμα, αν μεταξύ δύο υπηρεσιών α και β δεν υπάρχουν εξαρτήσεις, τότε δεν χρειάζεται η μία να περιμένει την άλλη. Οι α, β μπορούν να εκτελεστούν παράλληλα. Βέβαια η παράλληλη εκτέλεση δεν αποτελεί κατά ανάγκη την βέλτιστη λύση, όταν δεν υπάρχουν εξαρτήσεις φυσικά. Γιατί; Γιατί, αν έστω μία υπηρεσία μειώνει κατά πολύ τον όγκο των δεδομένων προς επεξεργασία τότε οι παραλληλισμένες υπηρεσίες, δεν μπορούν να επωφεληθούν από αυτό κι εξαναγκάζονται να επεξεργαστούν ένα πολύ μεγαλύτερο όγκο δεδομένων.

Η τέταρτη πρόκληση αφορά στην αποστολή των δεδομένων. Τα δεδομένα θα στέλνονται ένα – ένα ή ανά ομάδες; Σύμφωνα με τα πειράματα, ο χρόνος απόκρισης μιας υπηρεσίας εξαρτάται από τον τρόπο αποστολής των δεδομένων και μάλιστα όχι γραμμικά. Για αυτό ο βελτιωτικός μηχανισμός θα πρέπει να λάβει κι αυτό υπό όψιν του στην κατασκευή της διάταξης των υπηρεσιών.

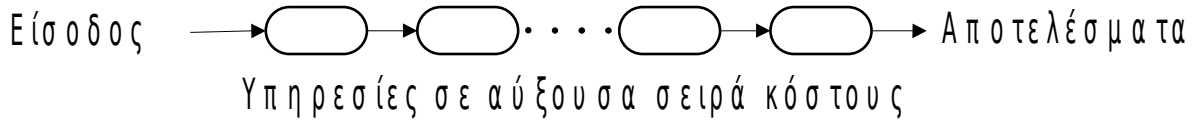
Μέχρις εδώ έγινε λόγος για την γενική λειτουργία κι αρχιτεκτονική του παρόντος τρόπου τάχιστου υπολογισμού ενός συνδέσμου. Τώρα, γίνεται λόγος για τους διάφορους αλγόριθμους που διαθέτει το σύστημα για την επίτευξη των στόχων του. Οι αλγόριθμοι αυτοί φυσικά ξεπερνούν τις προηγουμένως συζητηθείσες προκλήσεις. Αρχικά, οι αλγόριθμοι βασίζονται στην απλή πλην όμως σημαντική παρατήρηση ότι η απόδοση ενός σχεδίου υπολογισμού που ακολουθεί το σχήμα της διοχέτευσης, περιορίζεται από την υπηρεσία με τον μεγαλύτερο χρόνο απόκρισης. Για αυτό εισάγεται η μετρική ή ισοδύναμα το κριτήριο του κόστους συμφόρησης. Αυτό το κριτήριο έρχεται σε αντιδιαστολή με τα παραδοσιακά, συγκεντρωμένα συστήματα, όπου εκεί η απόδοση του αντίστοιχου σχεδίου περιορίζεται από το συνολικό κόστος των εργασιών(κριτήριο ή μετρική του συνολικού κόστους).

Συνεχίζοντας, ο αλγόριθμος παραλλάσσεται ανάλογα με το αν υπάρχουν περιορισμοί προτεραιότητας(εξαρτήσεις). Ο αλγόριθμος όταν δεν υπάρχουν περιορισμοί προτεραιότητας έχει με την σειρά του κι αυτός δύο παραλλαγές ανάλογα με το αν η εκλεκτικότητα όλων των υπηρεσιών ξεπερνά την μονάδα ή όχι. Διευκρινίζεται οτι η εκλεκτικότητα κάθε υπηρεσίας στο πλαίσιο των

9 Όπως φαίνεται από την μέχρι τώρα ανάλυση, στην υπό εξέταση μέθοδο υπολογισμού ενός συνδέσμου, οι υπηρεσίες μοιράζονται άμεσα ή έμμεσα μια κοινή δεξαμενή δεδομένων. Στις προηγούμενες μεθόδους, συνήθως υπήρχαν πολλές ανεξάρτητες είσοδοι.

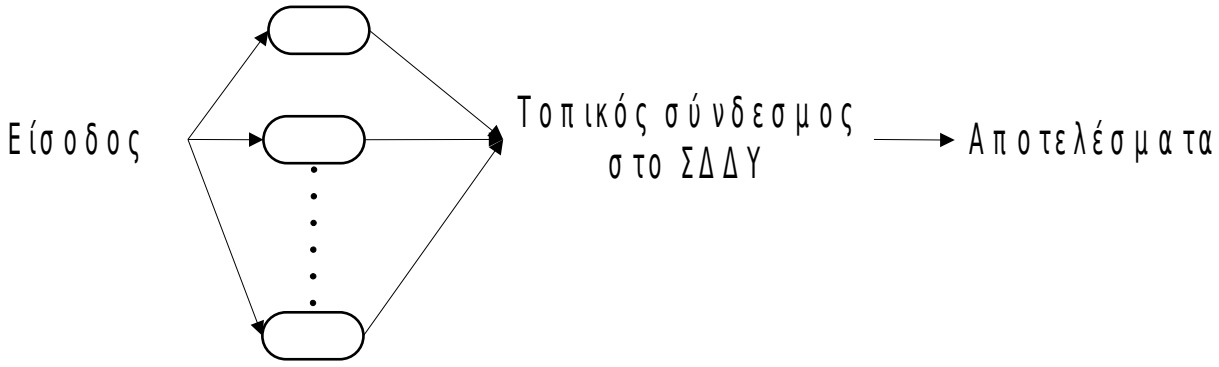
αλγορίθμων που παρουσιάζονται, θεωρείται ανεξάρτητη από αυτήν κάθε άλλης, ήδη κληθείσας υπηρεσίας. Ξεκινώντας, λοιπόν με την περιγραφή του αλγορίθμου όταν δεν υπάρχουν περιορισμοί προτεραιότητας και η εκλεκτικότητα όλων των υπηρεσιών ισούται το πολύ με 1, τότε, το βέλτιστο σχέδιο για την εκτέλεση του ερωτήματος απεικονίζεται στο Σχήμα 22.

Όπως φαίνεται από το σχήμα οι υπηρεσίες δρομολογούνται με αύξουσα σειρά κόστους. Αυτό συμβαίνει, γιατί οι πιο "αργές" υπηρεσίες(επιλεκτικότητα → 1) θα πρέπει να επεξεργάζονται όσο το δυνατόν λιγότερα δεδομένα προκειμένου να επιβαρύνουν όσο το δυνατόν λιγότερο την συνολική εκτέλεση. Διευκρινίζεται ότι καθώς τα δεδομένα υπόκεινται επεξεργασία από τις διάφορες υπηρεσίες, γενικά μειώνονται σε ποσότητα για το υπό εξέταση είδος συνδέσμων. Να παρατηρηθεί τέλος, οτι σε αυτήν την περίπτωση το βέλτιστο πλάνο είναι μια γραμμική διάταξη ανεξάρτητα από την ακριβή τιμή της εκλεκτικότητας των υπηρεσιών. Αρκεί η εκλεκτικότητα κάθε υπηρεσίας να μην ξεπερνά την 1.



Σχήμα 22 : Υπηρεσίες με εκλεκτικότητα ≤ 1

Όταν η εκλεκτικότητα κάθε υπηρεσίας ξεπερνά την 1, τότε όλες οι υπηρεσίες εκτελούνται παράλληλα έχοντας την ίδια είσοδο. Αυτή η τεχνική έχει το μειονέκτημα πως όλες οι υπηρεσίες αναλαμβάνουν την επεξεργασία ενός μεγάλου όγκου δεδομένων, κάτι που αυξάνει τον χρόνο απόκρισης τους. Παρόλα αυτά, αυτή η τεχνική αναδεικνύεται η βέλτιστη για την τρέχουσα περίπτωση. Το σχήμα εκτέλεσης αυτής της τεχνικής φαίνεται στο Σχήμα 23.

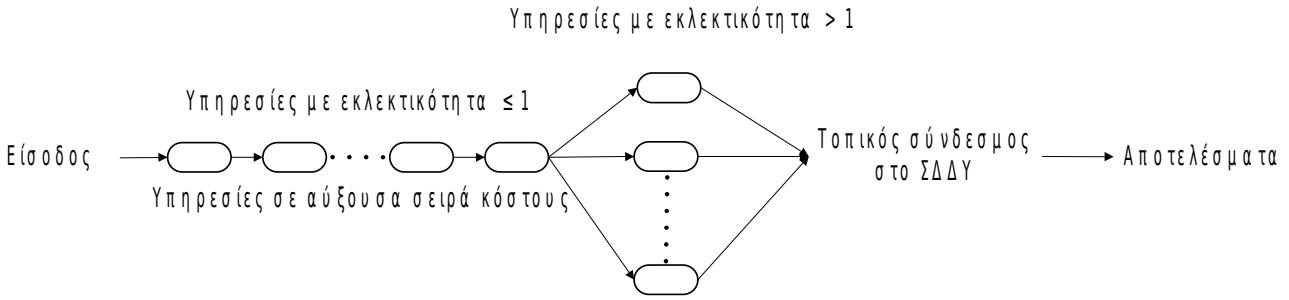


Σχήμα 23 : Υπηρεσίες με εκλεκτικότητα > 1

Τέλος στην περίπτωση που υπάρχουν υπηρεσίες τόσο με προτίμηση ≤ 1 όσο και > 1, το βέλτιστο σχέδιο για τον υπολογισμό του συνδέσμου είναι ο συνδυασμός των δύο προηγούμενων, δηλαδή(Σχήμα 24),

Σε κάθε περίπτωση, όταν δεν υπάρχουν περιορισμοί προτεραιότητας, τότε το βέλτιστο πλάνο είναι απλό στην σύνθεση του κι επίσης ανεξάρτητο από τις ακριβείς τιμές της κάθε εκλεκτικότητας. Αρκεί μόνο η σχετική τιμή τους ως προς την 1.

Από την άλλη όταν υπάρχουν περιορισμοί προτεραιότητας τότε τα πράγματα περιπλέκονται. Αρχικά οι διάφορες εξαρτήσεις των υπηρεσιών αναπαρίστανται με έναν κατευθυνόμενο, ακυκλικό



Σχήμα 24 : Μείξη υπηρεσιών διαφορετικής επιλεκτικότητας

γράφο Γ, του οποίου οι κόμβοι αποτελούν τις υπηρεσίες κι οι ακμές υποδηλώνουν τις εξαρτήσεις. Μια ακμή από τον κόμβο α στον κόμβο β, υποδηλώνει οτι η υπηρεσία β εξαρτάται από την υπηρεσία α. Στο σημείο αυτό τονίζεται πως ο γράφος Γ είναι ανεξάρτητος του ερωτήματος – συνδέσμου αυτού κάθε αυτού. Αποτελεί καθαρό προϊόν των εξαρτήσεων μεταξύ των υπηρεσιών, μία υπηρεσία έχει ως είσοδο την έξοδο κάποιας άλλης υπηρεσίας.

Τώρα, ένα βέλτιστο πλάνο αποτελεί ένα κατευθυνόμενο, ακυκλικό γράφο Η, διαφορετικό από τον Γ, που όμως τηρεί τις αναπαριστώμενες εξαρτήσεις από τον Γ, δηλαδή, οι ακμές διατηρούνται. Ο γράφος Η αναλυτικότερα, συνιστά μια διάταξη των υπηρεσιών που συμμετέχουν στον σύνδεσμο. Από τους κόμβους του Η μπορούν να εξέρχονται πολλές ακμές το οποίο αντανακλά τον παραλληλισμό των υπηρεσιών. Επίσης, στους κόμβους του Η δύνανται να εισέρχονται πολλές ακμές που υποδηλώνει τον σύνδεσμο των δεδομένων. Ο ίδιος ο γράφος Η κατασκευάζεται άπληστα ανά μία υπηρεσία την φορά. Η υπηρεσία αυτή πληρεί δύο προϋποθέσεις. Αφενός, όλες τις οι εξαρτήσεις έχουν πληρωθεί κι αφετέρου έχει το μικρότερο κόστος από όλες τις συνυποψήφιες. Η φράση “όλες τις οι εξαρτήσεις έχουν πληρωθεί” σημαίνει πως όλες οι υπηρεσίες από τις οποίες εξαρτάται η υποψήφια έχουν ήδη εισαχθεί στον Η.

Κλείνοντας, την περιγραφή του αλγορίθμου, αυτός έχει πολυπλοκότητα O(v^5), όπου ν είναι το πλήθος των υπηρεσιών. Η πολυπλοκότητα αυτή μπορεί να φαίνεται μεγάλη, αλλά από την μια είναι πολυωνυμικού χρόνου για αυθαίρετους περιορισμούς προτεραιότητας κι από την άλλη το ανάλογο πρόβλημα υπό το κριτήριο του συνολικού κόστους είναι NP – hard. Ακόμη, αυτή η χρονική πολυπλοκότητα χαρακτηρίζεται μάλλον απαισιόδοξη καθώς βασίζεται στην υπόθεση οτι το πλήθος των υποψήφιων υπηρεσιών σε κάθε επανάληψη του αλγορίθμου είναι σταθερό κι ίσο με ν. Κάτι τέτοιο στην πράξη όμως δεν ισχύει.

Ένα τελευταίο ζήτημα σχετικά με την υπό εξέταση λύση στο πρωταρχικό πρόβλημα συνιστά το πλήθος των πλειάδων εισόδου που δέχεται μια υπηρεσία. Επειδή κάθε κλήση μιας υπηρεσίας κοστίζει επιθυμείται ο περιορισμός αυτού του κόστους. Κάτι τέτοιο μπορεί να γίνει αν η καλούμενη υπηρεσία δέχεται ένα αριθμό από πλειάδες δεδομένων. Ωστόσο η υπηρεσία επεξεργάζεται την κάθε πλειάδα με τον ίδιο τρόπο που θα την επεξεργαζόταν αν λάμβανε μόνο αυτή. Η εύρεση του κατάλληλου μεγέθους αποτελεί ευθύνη του βελτιωτικού μηχανισμού. Όμως, η διαδικασία αυτή συνυπάρχει ανεξάρτητα από τον αλγόριθμο για την εύρεση του βέλτιστου πλάνου. Επιπροσθέτως, ο βέλτιστος αριθμός πλειάδων δεδομένων διαφέρει από υπηρεσία σε υπηρεσία κι η ακριβής τιμή του βασίζεται πολύ στην συνιστώσα των στατιστικών. Αυτό οφείλεται στο γεγονός οτι ο χρόνος απόκρισης μιας υπηρεσίας εξαρτάται από το πλήθος των πλειάδων εισόδου.

2.2.7 Διαχείριση διασποράς δεδομένων σε παράλληλους συνδέσμους

Ένας σύνδεσμος μπορεί να εκτελεστεί γρηγορότερα αν αυτός παραλληλιστεί, δηλαδή, σπάσει σε επιμέρους συνδέσμους που εκτελούνται παράλληλα. Κατά τον παραλληλισμό αυτόν απαιτείται ο διαμοιρασμός των δεδομένων στους επιμέρους, παραλληλισμένους συνδέσμους. Ο διαμοιρασμός αυτός δεν είναι πάντα εύκολος κι αποτελεί σημείο προβληματικό και καθυστέρησης. Το πρόβλημα συνίσταται στο πως ένας κακός διαμοιρασμός των δεδομένων, κατανέμει το φορτίο άνισα στους επιμέρους συνδέσμους κι αυτό με την σειρά του αναιρεί το όποιο όφελος από τον παραλληλισμό και επιφέρει απαράδεκτη απόδοση του συστήματος.

Πιο συγκεκριμένα το πρόβλημα προκύπτει από την διασπορά των δεδομένων. Η διασπορά των δεδομένων αφορά στην κατανομή των τιμών των δεδομένων. Με απλά λόγια οι τιμές των δεδομένων επηρεάζουν τον διαμοιρασμό των δεδομένων και κατά επέκταση την εκτέλεση του παραλληλισμένου συνδέσμου. Για παράδειγμα όταν μια τιμή εμφανίζεται πολλές φορές τότε υπάρχει πρόβλημα στον διαμοιρασμό των δεδομένων. Κάποιος σύνδεσμος θα έχει να συνδέσει πολλά περισσότερα δεδομένα από ότι κάποιοι άλλοι κι άρα θα αργήσει να παράξει αποτέλεσμα. Οπότε ο επιβαρυμένος σύνδεσμος τείνει να λειτουργεί σαν τον αρχικό σύνδεσμο που δεν έχει παραλληλιστεί κι εξού η αναίρεση της ωφέλειας του παραλληλισμού κι η κακή απόδοση του συστήματος.

Η αρχική λύση του προβλήματος σάρωνε τις σχέσεις προς σύνδεση προσπαθώντας να μειώσει την άνιση κατανομή του φορτίου. Όμως μια τέτοια λύση, αφενός προσθέτει σημαντικά μεγάλο κόστος για την πραγμάτωση της, αφετέρου επιβαρύνει σημαντικά την συνήθη περίπτωση έναντι μιας εξαιρετικής και σπάνιας περίπτωσης. Η συνήθης περίπτωση είναι αυτή στην οποία η διασπορά των δεδομένων είναι καλή, η κατανομή των τιμών των δεδομένων δεν τείνει προς μια τιμή.

Προσεγγίζοντας εναλλακτικά το πρόβλημα[27], προτείνεται η χρήση διαφορετικών αλγορίθμων ανάλογα με την ποιότητα της διασποράς των δεδομένων. Αν αυτή είναι καλή, τότε γίνεται χρήση του υβριδικού αλγορίθμου κατακερματισμού(hybrid hash join). Σε αντίθετη περίπτωση, γίνεται χρήση ενός άλλου αλγορίθμου, του εικονικού επεξεργαστή διαμοιρασμού εύρους(ΕΕΔΕ - virtual processor range partitioning VPRP). Ο δεύτερος αλγόριθμος κοστίζει ελάχιστα παραπάνω από τον πρώτο. Η επιλογή του εκάστοτε αλγορίθμου προϋποθέτει την εκ των προτέρων γνώση για την ποιότητα της διασποράς. Η γνώση αυτή αποκτιέται με την δειγματοληψία των σχέσεων προς σύνδεση. Η δειγματοληψία βέβαια αυτή, εξυπηρετεί και στον διαμοιρασμό των δεδομένων σε κάθε έναν από τους παραλληλισμένους συνδέσμους, και στην απεικόνιση των δεδομένων σε κάποιον επεξεργαστή(που έχει αναλάβει τον αντίστοιχο σύνδεσμο). Η προσέγγιση των δυο προαναφερθέντων αλγορίθμων πέραν του ότι λύνει το πρόβλημα, γίνεται επίσης εύκολα πράξη.

Προχωρώντας, υπάρχουν τρεις αιτίες για την άνιση κατανομή του φορτίου στους επιμέρους συνδέσμους. Η πρώτη αιτία συνίσταται στην χρήση κακής συνάρτησης κατακερματισμού και λέγεται διασπορά ανακατανομής, αφού συμβαίνει κατά την ανακατανομή των πλειάδων της σχέσης για την οποία παράγεται ο πίνακας κατακερματισμού του υβριδικού αλγόριθμου κατακερματισμού. Μια άλλη αιτία αποτελούν οι επαναλαμβανόμενες, πολλαπλές τιμές που αναγκαστικά κατακερματίζονται στον ίδιο σύνδεσμο. Η τρίτη αιτία λέγεται διασπορά συνδέσμου(product join skew) κι οφείλεται στην διαφορετική επιλεκτικότητα που μπορούν να έχουν ανάμεσα τους οι σύνδεσμοι. Ο σύνδεσμος 1 για παράδειγμα μπορεί να συνδέει πλειάδες που λιγότερο συχνά πληρούν την συνθήκη για να συνδεθούν, σε σχέση με τον σύνδεσμο 2.

Οι τρεις παραπάνω αιτίες αίρονται με την χρήση του ΕΕΔΕ, ο οποίος αντί για χρήση συνάρτησης κατακερματισμού, χειρίζεται κατακερματισμό εύρους όπου οι πλειάδες διανέμονται

στο κατάλληλο διαμέρισμα σύμφωνα με το εύρος στο οποίο εμπίπτει η τιμή τους. Ο ΕΕΔΕ λέγεται εικονικός, γιατί προκειμένου να αντεπεξέλθει στη διασπορά συνδέσμου δημιουργεί εικονικά διαμερίσματα που ξεπερνούν σε αριθμό τους πραγματικά διαθέσιμους επεξεργαστές και καταλλήλως αντιστοιχίζονται στους εν λόγω επεξεργαστές. Επίσης, ο ΕΕΔΕ έχει δύο εκδοχές ανάλογα με την τεχνική απεικόνισης των εικονικών επεξεργαστών στους πραγματικούς. Η πρώτη τεχνική χειρίζεται την αριθμητική υπολοίπου κι ο κ εικονικός επεξεργαστής απεικονίζεται στον κ mod ν πραγματικό επεξεργαστή όπου ν το πλήθος των πραγματικών επεξεργαστών. Η εκδοχή αυτή λέγεται ΕΕΔΕ εκ περιτροπής(VPRP – round robin, VPRP – RR). Η δεύτερη εκδοχή εισάγει μια συνάρτηση κόστους η οποία εκτιμά το κόστος υπολογισμού ενός εικονικού επεξεργαστή διαμέρισης. Η εκδοχή αυτή λέγεται ΕΕΔΕ δρομολόγησης επεξεργαστή(VPRP – processor scheduling, VPRP – PS).

Κλείνοντας, οι δύο παραπάνω εκδοχές του ΕΕΔΕ συγκρίνονται με τον υβριδικό αλγόριθμο κατακερματισμού, τον απλό αλγόριθμο κατακερματισμού εύρους(simple range partitioning) και τον σταθμισμένο αλγόριθμο κατακερματισμού εύρους(weighted range partitioning) καθώς μεταβάλλεται η διασπορά είτε στην σχέση για την οποία κατασκευάζεται ο πίνακας κατακερματισμού(εσωτερική σχέση – build) είτε στην σχέση η οποία συνδέεται στις εγγραφές του πίνακα κατακερματισμού(εξωτερική σχέση – probe) ή και στις δύο σχέσεις. Με βάση τα πειράματα αυτά, ο υβριδικός αλγόριθμος υπερτερεί ξεκάθαρα των άλλων όταν δεν υπάρχει διασπορά ή αυτή είναι πολύ μικρή. Όσο μεγαλώνει η διασπορά τόσο καλύτερα αποδίδει ο ΕΕΔΕ εκ περιτροπής έναντι των άλλων αλγορίθμων μέχρι η απόδοση του να κορεστεί. Τα συμπεράσματα αφορούν στην διασπορά που εμφανίζεται στην εσωτερική σχέση. Όταν η διασπορά υπάρχει στην εξωτερική σχέση τότε κερδίζει ο υβριδικός αλγόριθμος ενώ όταν και οι δύο σχέσεις εμφανίζουν διασπορά στις τιμές τους πάλι υπερέχει ο ΕΕΔΕ εκ περιτροπής.

2.2.8 Δυναμικό σύστημα εκτέλεσης ερωτημάτων για συνδυασμένα δεδομένα

Ως συνήθως πέρα από το περιεχόμενο μεγάλο ρόλο διαδραματίζει και η παρουσίαση. Επίσης είναι πολύ επιθυμητό να μην χρειάζεται ο χρήστης ρητά να εντοπίζει την πηγή της πληροφορίας αλλά αυτή να του προσφέρεται όσο πιο εύχρηστα γίνεται. Για αυτό πολλές φορές οι διάφορες πηγές πληροφοριών παρουσιάζονται με ενιαίο τρόπο. Ένα σύστημα που παρέχει αυτήν την ομοιόμορφη πρόσβαση σε μια πληθώρα πηγών πληροφορίας, λέγεται σύστημα ομοιόμορφης απεικόνισης των δεδομένων(ΣΟΑΔ, data integration system).

Ένα ΣΟΑΔ πρέπει να χαρακτηρίζεται δυναμικό και προσαρμοστικό, γιατί προορίζεται για ένα περιβάλλον όπου πρώτον, εκλείπουν στατιστικά στοιχεία για τα δεδομένα όπως η διασπορά αυτών, δεύτερον, ο ρυθμός άφιξης των δεδομένων στο σύστημα είναι απρόβλεπτος και τρίτον, πολλές πηγές πληροφορίας αλληλοεπικαλύπτονται ως προς το περιεχόμενο τους. Ένα ΣΟΑΔ που ανταποκρίνεται στις παραπάνω απαιτήσεις και χαρακτηριστικά είναι το Tukwila[28].

Το σύστημα αυτό αποσκοπεί στην ταχεία παραγωγή πρώτων αποτελεσμάτων, βασίζεται στον αλγόριθμο κατακερματισμού διπλής διοχέτευσης(double pipelined hash join), έχει έναν τελεστή ειδικά για την ένωση των δεδομένων από τις διάφορες πηγές, και μπορεί να παράγει είτε πλήρη είτε μερικά σχέδια εκτέλεσης κάποιου ερωτήματος ανάλογα με την διαθεσιμότητα των μετά – δεδομένων. Οπότε, μέσω των μερικών σχεδίων εκτέλεσης, το εν λόγω σύστημα προσαρμόζεται ακόμη και κατά την εκτέλεση ενός ερωτήματος.

Η αρχιτεκτονική του Tukwila περιλαμβάνει μηχανισμούς ώστε να αποκρύπτονται οι λεπτομέρειες για τις πηγές δεδομένων, περιέχει καταλόγους για τις πηγές πληροφορίας στους οποίους καταγράφονται διάφορα χρήσιμα στατιστικά στοιχεία και μετά – δεδομένα για τις σχέσεις. Επίσης το Tukwila έχει την ικανότητα να μετασχηματίζει το εκάστοτε ερώτημα κατάλληλα ώστε

να αντιστοιχίζεται το κάθε μέρος του ερωτήματος με την κατάλληλη πηγή πληροφορίας. Φυσικά, το Tukwila έχει κι έναν μηχανισμό που μετατρέπει ένα ερώτημα σε έναν όσο το δυνατό βέλτιστο σχέδιο εκτέλεσης του συνδέσμου. Τέλος, το εν λόγω ΣΟΑΔ εκτελεί τα ερωτήματα με τρόπο ώστε να παράγεται όσο το δυνατόν γρηγορότερα κάποιο μερικό αποτέλεσμα και διαθέτει επίσης προγράμματα(wrappers) που επιτυγχάνουν την επικοινωνία μεταξύ των σχέσεων και την μηχανή εκτέλεσης των ερωτημάτων.

Σύμφωνα με τα πειραματικά αποτελέσματα, ο αλγόριθμος κατακερματισμού, διπλής διοχέτευσης υπερτερεί έναντι του υβριδικού αλγορίθμου κατακερματισμού τόσο για ερωτήματα που γίνονται πλησίον του Tukwila όσο και για ερωτήματα που βρίσκονται σε πολύ μεγάλη απόσταση από αυτό. Σημειωτέον, ο υβριδικός αλγόριθμος κατακερματισμού συνιστά συνηθέστερη επιλογή για την επιλογή ενός συνδέσμου. Σε περιπτώσεις υπερχείλισης της μνήμης, ο αλγόριθμος κατακερματισμού, διπλής διοχέτευσης παράγει αποτελέσματα στην έξοδο κι ανάλογα με την στρατηγική που ακολουθείται ο ρυθμός παραγωγής αποτελεσμάτων μπορεί να μεταβάλλεται. Σε κάθε περίπτωση ο αλγόριθμος διπλής διοχέτευσης παραμένει εν γένει καλύτερος από τον υβριδικό αλγόριθμο.

Ακόμη τα πειράματα συνηγορούν υπέρ της στρατηγικής του Tukwila σύμφωνα με την οποία προκύπτει ένα βέλτιστο, μερικό σχέδιο εκτέλεσης του ερωτήματος για τις τρέχουσες συνθήκες κι έπειτα σχεδιάζεται ένα νέο μερικό σχέδιο εφόσον αυτό υπαγορεύεται από τα ενημερωμένα δεδομένα. Τέλος, η αποθήκευση της τρέχουσας κατάστασης του βελτιωτή των σχεδίων εκτέλεσης για μελλοντική χρήση, μπορεί υπό προϋποθέσεις να συνεισφέρει σημαντικά στην ταχεία εκτέλεση του συνδέσμου.

2.2.9 Συνεχώς προσαρμοζόμενα, συνεχή ερωτήματα επί ρευμάτων δεδομένων

Συμβαίνει κάποιες φορές, ένα ερώτημα(ένας σύνδεσμος) να έχει διαφορετική απάντηση ανάλογα με την χρονική στιγμή που αυτό ερωτάται. Ένα τέτοιο ερώτημα λέγεται συνεχές κι έχει αρκετή διάρκεια ώστε η απάντηση σε αυτό να αλλάζει καθώς το σύστημα στο οποίο αντιστοιχεί μεταβάλλεται. Για παράδειγμα η θερμοκρασία ενός χώρου αλλάζει συνέχεια ή ένταση του ήχου με συνέπεια τα αντίστοιχα ερωτήματα να έχουν διαφορετική απάντηση ανάλογα με την χρονική στιγμή που ερωτώνται. Όταν ένα συνεχές ερώτημα συνδυαστεί με έναν τελεστή συνεχώς προσαρμοζόμενο, τότε προκύπτει το συνεχώς προσαρμοζόμενο, συνεχές ερώτημα(ΣΠΣΕ, continuous adaptive, continuous query – CACQ). Ένας τελεστής συνεχώς προσαρμοζόμενος είναι ο τελεστής eddy.

Στο παρόν εδάφιο μελετάται η υλοποίηση μιας μηχανής εκτέλεσης ΣΠΣΕ με χρήση της μηχανής ροής δεδομένων Telegraph[29]. Η υλοποίηση αυτή ωφελεί ακόμα και σε περιπτώσεις που δεν υπάρχει κάποια ορατή μεταβολή, χάρη στην ικανότητά της να διαμοιράζει τους υπολογισμούς και την μνήμη στα διάφορα ΣΠΣΕ καλύτερα συγκριτικά με τεχνικές που χειρίζονται στατικά σχέδια εκτέλεσης ερωτημάτων.

Πιο συγκεκριμένα, η τρέχουσα υλοποίηση δίνει λύση σε τέσσερις προκλήσεις που μια μηχανή ΣΠΣΕ αντιμετωπίζει. Πρώτον, η επιλεκτικότητα ενός ερωτήματος μπορεί να αλλάζει κι άρα θα πρέπει κι η θέση του ερωτήματος εντός του σχεδίου εκτέλεσης των διαφόρων παράλληλων ερωτημάτων να αλλάζει. Όταν η επιλεκτικότητα χαρακτηρίζεται μικρή θα πρέπει το αντίστοιχο ερώτημα να τοποθετείται προς το τέλος της ουράς των προς εκτέλεση ερωτημάτων. Όταν η επιλεκτικότητα χαρακτηρίζεται μεγάλη θα πρέπει το αντίστοιχο ερώτημα να τοποθετείται προς την αρχή της ουράς των προς εκτέλεση ερωτημάτων. Προχωρώντας, πέρα από την αλλαγή αυτή κάθε αυτή, υπάρχει κι ένα θέμα σχετικά με το πως και πότε θα γίνεται η αλλαγή αυτή. Επιπλέον, κάποιες φορές απαιτείται να αλλάζει η σειρά με την οποία εφαρμόζονται οι διάφοροι τελεστές στις πλειάδες

δεδομένων. Με άλλα λόγια, μια πλειάδα μπορεί να περνά από κάποιους τελεστές με κάποια σειρά α αλλά μια άλλη πλειάδα μπορεί να πρέπει να περάσει από τους ίδιους τελεστές ακολουθώντας μια διαφορετική σειρά β. Αυτή η εξατομικευμένη πορεία για την κάθε πλειάδα επιθυμείται να γίνεται ακόμα κι ενώ το ερώτημα έχει υποβληθεί. Χάρη στον τελεστή eddy η παραπάνω πρόκληση απαντάται σε όλη την έκταση της.

Μια δεύτερη πρόκληση αποτελεί το γεγονός πως τελικά μια πλειάδα δεδομένων μπορεί να έχει πολλές διαφορετικές, δυνατές πορείες να ακολουθήσει ανάμεσα στους διαθέσιμους τελεστές. Για αυτό θα πρέπει για την κάθε πλειάδα να επιλέγεται η κατάλληλη πορεία κάθε φορά προκειμένου το τελικό αποτέλεσμα να λέγεται σωστό.

Η τρίτη πρόκληση συνίσταται στην επιδίωξη να αποφεύγεται περίσσιος υπολογισμός όταν οι διάφορες συνθήκες των διαφόρων ερωτημάτων έχουν κοινούς υπολογισμούς. Για την επίτευξη αυτής της επιδίωξης γίνεται χρήση της τεχνικής του ομαδικού φίλτρου(grouped filter) η οποία δέχεται ένα σύνολο συνθηκών και μια πλειάδα δεδομένων κι επιστρέφει τις συνθήκες που μπορούν να εφαρμοστούν στην πλειάδα δεδομένων. Οπότε αν π.χ. πρέπει να ρυθμιστεί η θερμοκρασία σε διάφορους χώρους τότε επειδή η θερμοκρασία άνεσης είναι λίγο – πολύ ίδια για όλους του ανθρώπους, μπορεί να εφαρμοστεί το ομαδικό φίλτρο και να ευρεθεί μια συνθήκη που ταιριάζει με την θερμοκρασία της αντίστοιχης πλειάδας. Με αυτό τον τρόπο όλες οι πλειάδες μπορούν να χωριστούν σε ομάδες ανάλογα με το αποτέλεσμα του ομαδικού φίλτρου κι άρα οι αντίστοιχοι τελεστές για την ρύθμιση της θερμοκρασίας να εφαρμοστούν σε ένα σύνολο από πλειάδες κι όχι σε κάθε μία πλειάδα ξεχωριστά. Προφανώς με αυτόν τον τρόπο εξοικονομούνται οι υπολογισμοί αφού με έναν υπολογισμό ικανοποιούνται πολλές πλειάδες δεδομένων.

Η τέταρτη πρόκληση τέλος αφορά στην περίπτωση που τα ερωτήματα ενός χρήστη σχετίζονται με διάφορες πηγές πληροφορίας των οποίων η πληροφορία είναι κοινή. Σε αυτήν την περίπτωση επιβάλλεται οι πλειάδες να συνδέονται μεταξύ τους εφόσον ανήκουν εντός του ίδιου χρονικού παραθύρου. Επίσης επιστρατεύεται ο αλγόριθμος κατακερματισμού διπλής διοχέτευσης στο πλαίσιο του τελεστή eddy. Για κάθε πηγή – εισόδου τίθεται ένας δείκτης που ενθυλακώνεται σε έναν τελεστή με το όνομα SteM. Αυτό επιτρέπει την εκτέλεση ενός πολλαπλού συνδέσμου διοχέτευσης κι άρα επιτρέπει τον σύνδεσμο οποιωνδήποτε υποσυνόλων των σχέσεων εισόδου.

Κλείνοντας, τα πειραματικά αποτελέσματα επιβεβαιώνουν την αποτελεσματικότητα των παραπάνω τεχνικών και δείχνουν την δυνατότητα προσαρμογής του συστήματος στις δυναμικές συνθήκες του περιβάλλοντος. Επίσης, το υπό εξέταση σύστημα, σε σχέση με τον υπάρχον(μέχρι το 2002), το NiagaraCQ αποδίδει καλύτερα.

2.3 Εναλλακτική προσέγγιση σε περιβάλλον Map - Reduce

Μέχρι τώρα εξετάστηκαν διάφοροι τρόποι για την βελτίωση της ταχύτητας υπολογισμού ενός συνδέσμου και μάλιστα πολλαπλού. Σε αυτό το σημείο εξετάζεται μια τελευταία μέθοδος, [22] η οποία παρουσιάζεται αρχικά εδώ θεωρητικά και μετά, μελετάται και πειραματικά στην έκταση όλης της υπόλοιπης εργασίας.

Ξεκινώντας, ο όγκος των δεδομένων κάποιων εφαρμογών όπως μια μηχανή αναζήτησης, ένα κοινωνικό δίκτυο, επιβάλλει την επεξεργασία τους σε ένα κατανεμημένο περιβάλλον, μολονότι τα ίδια τα δεδομένα υφίστανται απλές τεχνικές επεξεργασίας. Ένα κατανεμημένο περιβάλλον είναι αυτό στο οποίο τα δεδομένα έχουν κατανεμηθεί σε πολλά, διακριτά, κοινά υπολογιστικά συστήματα. Εξαιτίας αυτού του κατανεμημένου περιβάλλοντος κι άλλων σχετικών προβλημάτων, αναπτύχθηκε ένα νέο σύνολο εργαλείων λογισμικού στην θέση των παραδοσιακών συστημάτων αρχείων, λειτουργικών συστημάτων και συστημάτων διαχείρισης βάσεων δεδομένων. Κεντρικό ρόλο σε αυτά τα νέα εργαλεία έχει το σύστημα αρχείων όπως το σύστημα αρχείων της Google [23] (Google File System – GFS) ή το κατανεμημένο σύστημα αρχείων Hadoop[10] [24] (Hadoop Distributed File System – HDFS). Ένα τέτοιο σύστημα αρχείων σε αντίθεση με ένα παραδοσιακό, έχει πολύ μεγαλύτερες δομικές μονάδες αρχείων, έως και 1000 φορές. Λέγοντας δομική μονάδα αρχείων εννοείται ο ελάχιστος αριθμός αρχείων τον οποίο μπορεί ο υπεύθυνος μηχανισμός να διαχειριστεί. Επίσης, οι δομικές μονάδες των αρχείων υπάρχουν σε πολλαπλά αντίγραφα. Τα πολλαπλά αντίγραφα χρειάζονται ως πρόληψη στις συχνές αστοχίες που υφίστανται οι διάφορες λειτουργικές μονάδες ενός κατανεμημένου συστήματος(fault tolerance). Οπότε τα δεδομένα διατίθενται άμεσα από διάφορα σημεία του κατανεμημένου συστήματος. Τα πολλαπλά αντίγραφα όμως εξυπηρετούν και κατά την ομαλή λειτουργία του συστήματος αφού επιτρέπεται τοιουτοτρόπως η ταυτόχρονη πρόσβαση, εν παραλλήλω σε ένα αρχείο.

Έχοντας πλέον αυτό το νέο σύστημα αρχείων, οι εφαρμογές αναπτύσσονται με την βοήθεια ενός ισχυρού εργαλείου με το όνομα map – reduce [25] της Google ή το ισοδύναμο του στην κοινότητα του ανοιχτού κώδικα, Hadoop. Ξεφεύγοντας από τα όρια της κάθε προσέγγισης – υλοποίησης, Google, Hadoop ή ό,τι άλλο, το Map – Reduce αποτελεί “μια προγραμματιστική αναπαράσταση και μια υφιστάμενη υλοποίηση για την επεξεργασία και παραγωγή μεγάλων συνόλων δεδομένων”. Συνιστά ένα τρόπο προγραμματισμού και σκέψης για την αποπεράτωση εργασιών που αφορούν σε ένα κατανεμημένο περιβάλλον με τεράστιο όγκο δεδομένων με την απαίτηση της αντοχής σε βλάβες του υλικού. Ο τρόπος αυτός, εμπνευσμένος από τον συναρτησιακό προγραμματισμό, συνίσταται σε δύο συναρτήσεις, την map και την reduce. Η πρώτη συνάρτηση επεξεργάζεται τα δεδομένα και παράγει άλλα στην μορφή ζευγών κλειδιού – τιμής. Η δεύτερη συνάρτηση έχοντας λάβει όλες τις τιμές με το ίδιο κλειδί, παράγει με βάση τις τιμές αυτές μία και μόνο τιμή για το εν λόγω κλειδί. Εξίσου κι οι δύο συναρτήσεις έχουν πολλά στιγμιότυπα κατά την εκτέλεση του προγράμματος που τις κάλεσε κι όλα αυτά τα στιγμιότυπα συντονίζονται από έναν ελεγκτή(master controller). Ο ελεγκτής αυτός πέρα από το να συντονίζει τα στιγμιότυπα, ευθύνεται και για την επανεκκίνηση των διαφόρων εργασιών, όταν αυτές αποτυγχάνουν λόγω κατάρρευσης του υλικού.

Κλείνοντας με το σύστημα στο πλαίσιο του οποίου δρα η εξεταζόμενη μέθοδος, παρουσιάζεται το περιβάλλον στο οποίο λαμβάνει χώρα το Map – Reduce. Συγκεκριμένα, το περιβάλλον αυτό έχει τρεις κύριες συνιστώσες. Η πρώτη αφορά στα αρχεία. Ειδικότερα, τα δεδομένα ενός αρχείου κείνται τυχαία μέσα σε αυτό καθιστώντας τα αρχεία σχέσεις όπως σε ένα σύστημα διαχείρισης βάσεων δεδομένων. Επίσης πολλές διεργασίες δύνανται να διαβάσουν παράλληλα ένα αρχείο λόγω του πλήθους των υφιστάμενων αντιγράφων αυτών. Ακόμη πολλές

10 Το Hadoop συζητείται εκτενέστερα στο κεφάλαιο 3 μαζί με το Map – Reduce.

διεργασίες μπορούν να γράψουν σε ένα αρχείο την ίδια στιγμή αφού τα δεδομένα κάθε αρχείου δεν έχουν κάποια καθορισμένη θέση.

Η δεύτερη συνιστώσα σχετίζεται με τις διεργασίες. Κάθε διεργασία θεωρείται η δομική μονάδα υπολογισμού, δέχεται δεδομένα εισόδου από πολλές πηγές και μπορεί να παράξει δεδομένα εξόδου για ένα ή περισσότερα αρχεία. Τέλος, μια διεργασία ανατίθεται σε έναν οποιοδήποτε επεξεργαστή. Η τρίτη και τελευταία συνιστώσα αφορά στους επεξεργαστές οι οποίοι πρακτικά θεωρούνται άπειροι στο πλήθος. Επιπροσθέτως, οι επεξεργαστές αποτελούν κοινούς επεξεργαστές με CPU, κύρια μνήμη και δευτερεύουσα μονάδα αποθήκευσης. Πέρα από τα προηγούμενα κάθε ένας επεξεργαστής είναι γενικού σκοπού με την έννοια ότι επεξεργάζεται κάθε αρχείο και κάθε συνιστώσα ενός αρχείου.

Προχωρώντας προς τον αλγόριθμο, αυτός εκλαμβάνεται ως ένας ακυκλικός γράφος όπου οι κόμβοι του αποτελούν τις διεργασίες κι οι ακμές του υποδηλώνουν την μεταφορά δεδομένων από διεργασία σε διεργασία. Για την ακρίβεια μια διεργασία έχει ως είσοδο την έξοδο κάποιων άλλων διεργασιών και δεν μπορεί να λειτουργήσει αν δεν διαθέτει το σύνολο της απαιτούμενης εισόδου. Επειδή κάθε διεργασία μπορεί να ανατεθεί σε έναν οποιοδήποτε επεξεργαστή, μια διεργασία μπορεί να αρχίσει την λειτουργία της με το που είναι έτοιμα τα δεδομένα εισόδου της.

Έχοντας πει τα παραπάνω, το κριτήριο κόστους ή αλλιώς η μετρική του αλγορίθμου είναι αυτό του κόστους επικοινωνίας μεταξύ διεργασιών. Με άλλα λόγια ενδιαφέρει πόσο κοστίζει η μεταφορά της εξόδου από μια διεργασία Α στην είσοδο μιας διεργασίας Β. Αναλυτικότερα, υπάρχουν τρεις κατηγορίες αυτού του κόστους.

Αρχικά υπάρχει το κόστος επικοινωνίας μιας διεργασίας που ισούται με το μέγεθος της εισόδου της. Η έξοδος δεν προσμετράται στο κόστος καθώς είτε αποτελεί την είσοδο σε κάποια άλλη διεργασία κι άρα προσμετράται εκεί είτε συνιστά την συνολική έξοδο του αλγορίθμου πως ως μέγεθος οφείλει να χαρακτηρίζεται αμελητέο σε σχέση με το μέγεθος της όποιας εισόδου. Υπάρχει το συνολικό κόστος που ισούται με το άθροισμα του κόστους επικοινωνίας όλων των επιμέρους διεργασιών. Τέλος, γίνεται λόγος και για το διαβιβαστικό κόστος επικοινωνίας ή κόστος επικοινωνίας διάβασης που αφορά σε έναν ακυκλικό γράφο διεργασιών. Πιο αναλυτικά κάθε μονοπάτι στον εν λόγω γράφο έχει ένα κόστος διάβασης από ένα κόμβο σε ένα άλλο που ισούται με το άθροισμα του κόστους επικοινωνίας όλων των διεργασιών που βρίσκονται σε αυτό το μονοπάτι. Το μέγιστο κόστος από κάθε άλλο κόστος μονοπατιού ονομάζεται κόστος επικοινωνίας διάβασης.

Σημειωτέον, το κόστος επεξεργασίας δεν λογίζεται στο κόστος του αλγορίθμου καθώς τελικά αυτό μπορεί να αμεληθεί χάρη στην χρήση διάφορων τεχνικών μερικές εκ των οποίων αναφέρθηκαν προηγουμένως όπως η τεχνική βασισμένη σε δέντρα κι η τεχνική για το βέλτιστο διαμερισμό των δεδομένων.

Συνεχίζοντας, από εδώ και στο εξής θα αναπτυχθεί η τεχνική για την ταχύτερη εκτέλεση ενός συνδέσμου και δη πολλαπλού στο περιβάλλον του Map – Reduce κι ειδικότερα σε αυτό του Hadoop. Κάνοντας τα προηγούμενα πιο συγκεκριμένα, οι διεργασίες στο Map – Reduce, είναι τα εν παραλλήλω στιγμιότυπα των συναρτήσεων map και reduce[11] που αναφέρθηκαν προηγουμένως. Επομένως, θα πρέπει το κόστος επικοινωνίας μεταξύ των διεργασιών αυτών να περιοριστεί το μέγιστο δυνατό. Επίσης, νωρίτερα έγινε λόγος για ζεύγη κλειδιών – τιμών που αποστέλλονται σε μια ρ – διεργασία. Σημειωτέον, τα κλειδιά αυτά δεν χαρακτηρίζονται μοναδικά, απλά αποτελούν τιμές με βάση τις οποίες γίνεται η αποστολή των δεδομένων. Η έννοια του κλειδιού οδηγεί σε μια μέθοδο κατακερματισμού, μια αντίστοιχη συνάρτηση κατακερματισμού και στην έννοια του καλαθιού(bucket). Λέγοντας καλάθι εννοείται η μνήμη στην οποία αποθηκεύονται οι πλειάδες δεδομένων που φτάνουν σε μια διεργασία. Αν οι διεργασίες έχουν μόνο ένα τέτοιο καλάθι, τότε, το

11 Στο υπόλοιπο του κειμένου μια διεργασία map θα λέγεται μ – διεργασία και μια διεργασία reduce θα λέγεται ρ – διεργασία.

πλήθος των διεργασιών ταυτίζεται με αυτό των καλαθιών.

Για παράδειγμα, έστω οι σχέσεις Ρ(Α, Β) και Σ(Β, Γ) που έχουν αποθηκευτεί σε ένα σύστημα αρχείων όπως το HDFS. Οι σχέσεις αυτές συνδέονται με βάση το κοινό τους γνώρισμα που είναι η ιδιότητα Β. Η ιδιότητα Β, δηλαδή, είναι το κλειδί κάθε δυάδας δεδομένων ανεξαρτήτως της σχέσης στην οποία ανήκει αυτή η δυάδα δεδομένων. Επομένως μια μ – διεργασία για την σχέση Ρ παράγει τιμές της μορφής (α, Ρ) για το κατάλληλο κλειδί β της ιδιότητας Β. Το όνομα της σχέσης Ρ, χρησιμεύει ως διακριτικό για να διαχωρίζονται οι δυάδες που προέρχονται από την σχέση Ρ από αυτές της σχέσης Σ και τελικά να γίνεται μόνο σύνδεση μεταξύ των δύο σχέσεων. Με άλλα λόγια δεν επιθυμείται ο σύνδεσμος κάποιας σχέσης με τον εαυτό της. Αντίστοιχα για την Σ, παράγονται δυάδες (γ, Σ) με κλειδί την αντίστοιχη τιμή β της ιδιότητας Β.

Δεδομένου του κλειδιού β, οι πλειάδες από κάθε σχέση αποστέλλονται σε κάποια ρ – διεργασία ώστε να συνδυασθούν και να δώσουν ένα τελικό αποτέλεσμα. Η διεργασία που αναμένει να λάβει πλειάδες, δέχεται μόνο πλειάδες που αντιστοιχούν σε ένα συγκεκριμένο κλειδί. Αν λοιπόν, υπάρχουν ν ρ – διεργασίες τότε απαιτείται μια συνάρτηση κατακερματισμού κ[12] προκειμένου να γίνεται η αποστολή των πλειάδων στην κατάλληλη από αυτές. Συνεπώς, κάθε μ – διεργασία στέλνει τις πλειάδες που παράγει με βάση την τιμή κ(β) στην κατάλληλη ρ – διεργασία. Τελικά αυτές οι διεργασίες έχοντας λάβει τα ζεύγη (α, Ρ) και (γ, Σ) για το κλειδί β, παράγουν την τριάδα (α, β, γ) την οποία αποθηκεύουν μαζί με τις υπόλοιπες σε ένα κοινό αρχείο εξόδου.

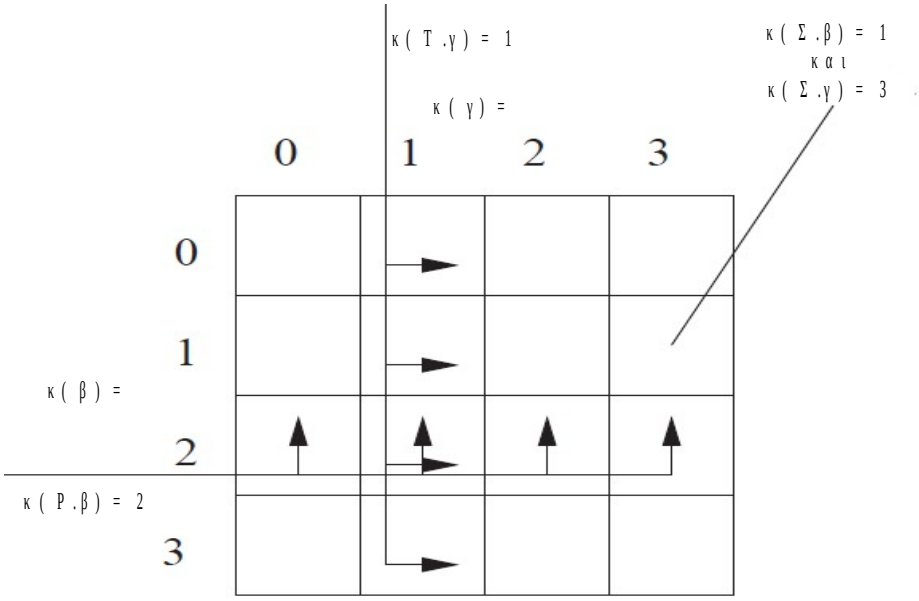
Τώρα ο σύνδεσμος των σχέσεων Ρ(Α, Β), Σ(Β, Γ) και Τ(Γ, Δ) ως γνωστόν μπορεί να υπολογιστεί όπως παραπάνω, συνδέοντας μία ακραία σχέση με την Σ και συνδέοντας το παραγόμενο αποτέλεσμα με την εναπομείνασα ακραία σχέση. Επίσης, ως γνωστόν, οι τρεις παραπάνω σχέσεις μπορούν να συνδεθούν μονομιάς αρκεί η προηγούμενη μέθοδος να επεκταθεί καταλλήλως.

Κάθε μ – διεργασία στέλνει κάθε μία δυάδα από τις σχέσεις Ρ και Τ σε πολλές διεργασίες reduce αλλά στέλνει κάθε δυάδα από την Σ σε μία μόνο ρ – διεργασία. Τα πολλαπλά αντίγραφα των δυάδων των σχέσεων Ρ και Τ αυξάνουν το κόστος επικοινωνίας αφού αυξάνεται το μέγεθος της εισόδου των διεργασιών που θα τις δεχτούν. Όμως τα αντίγραφα αυτά μειώνουν το κόστος επικοινωνίας καθώς δεν χρειάζεται να αποσταλεί το ενδιάμεσο αποτέλεσμα του πρώτου συνδέσμου όπως στην περίπτωση που ο σύνδεσμος υπολογιζόταν ως ακολουθία δύο απλούστερων. Επομένως, στην περίπτωση που μια πλειάδα από κάποια σχέση συνδέεται με πολλές πλειάδες από κάποια άλλη σχέση, ο πολλαπλός σύνδεσμος υπερτερεί έναντι της ακολουθίας των δυαδικών συνδέσμων.

Συνεχίζοντας το παράδειγμα, αν οι ρ – διεργασίες ανέρχονται σε $ν = σ^2$, τότε, επιλέγοντας τις ιδιότητες Β και Γ ως τις ιδιότητες κλειδιά, οι τιμές των Β και Γ θα αποστέλλονται σε σ ρ – διεργασίες. Οπότε, κάθε ρ – διεργασία έχει μια δυάδα από καλάθια, ένα για το κλειδί “τύπου” Β κι ένα για το κλειδί “τύπου” Γ. Κάθε δυάδα από καλάθια έχει το όνομα (η, ξ) όπου τα η, ξ = 1, 2, ..., σ. Ως αποτέλεσμα, οι δυάδες της σχέσης Ρ αποστέλλονται στις ρ – διεργασίες με δυάδα καλαθιών (κ(β), χ) όπου το χ = 1, 2, ..., σ, οι δυάδες της Τ αποστέλλονται στις διεργασίες με δυάδα καλαθιών (ζ, κ(γ)) όπου το ζ = 1, 2, ..., σ κι οι δυάδες της Σ αποστέλλονται στην διεργασία με την δυάδα καλαθιών (κ(β), κ(γ)), όπου τα β, γ αποτελούν συγκεκριμένες τιμές των κλειδιών Β, Γ. Με αυτόν τον τρόπο αν οι δυάδες Ρ(α, β), Σ(β, γ), Τ(γ, δ) μπορούν να συνδεθούν, τότε θα συμπέσουν στην ίδια ρ – διεργασία (κ(β), κ(γ)) και θα συνδεθούν. Μέσω του παραδείγματος αυτού φαίνεται η ορθότητα του αλγορίθμου για τον υπολογισμό ενός συνδέσμου. Σημειώνεται πως επειδή τα διάφορα κλειδιά των δεδομένων αφορούν σε μ – διεργασίες, τα κλειδιά αυτά λέγοντα μ – κλειδιά(map key). Τέλος, στο σχήμα 21 φαίνεται ένα γραφικό παράδειγμα της προαναφερθείσας τεχνικής.

12 Η συνάρτηση κατακερματισμού κ : Κ → Θ απεικονίζει ένα σύνολο κλειδιών Κ σε ένα σύνολο αντικειμένων Θ. Τα αντικείμενα αυτά στην υπό εξέταση περίπτωση είναι διεργασίες ενώ σε άλλες μπορεί να είναι αριθμοί, θέσεις μνήμης ή κάτι άλλο.

Παρά την όλη ανάλυση μένουν να απαντηθούν ακόμα ερωτήματα όπως ποια είναι η



Σχήμα 25 : Παράδειγμα για 16 ρ – διεργασίες

συνάρτηση κόστους, πως επιλέγονται οι ιδιότητες κλειδιά και ποιος είναι ο ρόλος των καλαθιών στην όλη διαδικασία. Πηγαίνοντας προς την απάντηση των ερωτημάτων αυτών, υπενθυμίζεται πως το κόστος του αλγορίθμου εξαρτάται από το μέγεθος των εισόδων των διεργασιών. Στο Map – Reduce περιβάλλον οι ρ – διεργασίες περιμένουν δεδομένα από τις μ – διεργασίες. Άρα, το κόστος του αλγορίθμου μειώνεται όσο τα δεδομένα από τις μ στις ρ – διεργασίες μειώνονται. Η ποσότητα των δεδομένων αυτών επηρεάζεται τόσο από τις ιδιότητες κλειδιά όσο κι από το πλήθος των καλαθιών ή ισοδύναμα από το πλήθος των ρ – διεργασιών.

Έστω ο κυκλικός σύνδεσμος Ρ(Α, Β) ⋈ Σ(Β, Γ) ⋈ Τ(Α, Γ) κι ότι ο επιθυμητός αριθμός ρ – διεργασιών είναι ν. Κάθε μία από τις ιδιότητες Α, Β, Γ συμμετέχει στο μ – κλειδί έχοντας ένα μερίδιο α, β, γ αντίστοιχα. Το μερίδιο αυτό ισούται με τον αριθμό των καλαθιών που διατίθενται σε κάθε ιδιότητα(σ στο προηγούμενο παράδειγμα). Κατά επέκταση υπάρχει μια συνάρτηση κατακερματισμού κ για κάθε ιδιότητα που απεικονίζει τις τιμές της στα καλάθια που τις αναλογούν. Σε ποια ιδιότητα αναφέρεται η συνάρτηση κ θα υποδηλώνεται από το όρισμα της κατά την χρήση της π.χ. το κ(α) αναφέρεται στην συνάρτηση της ιδιότητας Α, όπου το α είναι κάποια τιμή της Α. Επιπλέον, ισχύει η σχέση αβγ = ν, το γινόμενο των μεριδίων των ιδιοτήτων ισούται με το πλήθος των ρ – διεργασιών.

Από τα παραπάνω προκύπτει ότι κάθε ρ – διεργασία συσχετίζεται με μια τριάδα καλαθιών ή ισοδύναμα με ένα μ – κλειδί (η, θ, ξ) όπου το η = 1, 2, ..., α, το θ = 1, 2, ..., β και το ξ = 1, 2, ..., γ. Με άλλα λόγια το η αναπαριστά ένα καλάθι στο οποίο οι τιμές της ιδιότητας Α κατακερματίζονται και το ανάλογο ισχύει και για τα θ, ξ για τις αντίστοιχες ιδιότητες. Επομένως, μια δυάδα (χ, ψ) της σχέσης Ρ αποστέλλεται στις ρ – διεργασίες για τις οποίες ισχύει κ(χ) = η και κ(ψ) = θ ανεξάρτητα από την τιμή του ξ. Συνοψίζοντας, η δυάδα (χ, ψ) στέλνεται σε εκείνες τις ξ διεργασίες των οποίων το μ – κλειδί είναι (κ(χ), κ(ψ), ξ) με ξ = 1, ..., γ.

Με το ίδιο σκεπτικό, μια δυάδα της Σ, (ψ, ω) στέλνεται σε εκείνες τις η διεργασίες για τις οποίες το μ – κλειδί είναι το (η, κ(ψ), κ(ω)) με η = 1, ..., α και μια δυάδα της Τ, (χ, ω) προσλαμβάνεται από τις θ διεργασίες με μ – κλειδί (κ(χ), θ, κ(ω)), με θ = 1, ..., β. Αν λοιπόν, το

μέγεθος της σχέσης Ρ είναι ρ, της Σ είναι σ και της Τ είναι τ, τότε ο συνολικός αριθμός δυάδων που λαμβάνεται από τις ρ – διεργασίες ή αλλιώς το μέγεθος της εισόδου τους φτάνει τις ργ + σα + τβ δυάδες. Με άλλα λόγια το κόστος επικοινωνίας είναι

$$K(\alpha, \beta, \gamma) = \rho\gamma + \sigma\alpha + \tau\beta$$

κι όπως φαίνεται συναρτάται άμεσα από το πλήθος των καλαθιών και των ρ – διεργασιών. Τα μεγέθη των σχέσεων θεωρούνται γνωστά.

Συνεχίζοντας, θα πρέπει το κόστος $K(\alpha, \beta, \gamma) = \rho\gamma + \sigma\alpha + \tau\beta$ να ελαττωθεί το μέγιστο δυνατό υπό τον περιορισμό αβγ = ν. Από τα μαθηματικά, υπάρχει η μέθοδος των πολλαπλασιαστών του Lagrange για την εύρεση ακρότατων τιμών για συναρτήσεις που πρέπει να πληρούν περιορισμούς ισότητας. Σύμφωνα με την μέθοδο αυτή λαμβάνεται η εξίσωση Lagrange,

$$\Lambda(\alpha, \beta, \gamma, \lambda) = \rho\gamma + \sigma\alpha + \tau\beta - \lambda(\alpha\beta\gamma - \nu)$$

όπου το λ αναπαριστά τον πολλαπλασιαστή Lagrange. Η τελευταία εξίσωση με παραγώγιση ως προς τις μεταβλητές α, β,γ παράγει τις εξισώσεις,

$$\sigma = \lambda\beta\gamma$$

$$\tau = \lambda\alpha\gamma$$

$$\rho = \lambda\alpha\beta$$

που αν πολλαπλασιαστούν με την μεταβλητή η οποία απουσιάζει από αυτές τότε οι παραπάνω εξισώσεις δεδομένου ότι αβγ = ν, γράφονται,

$$\sigma\alpha = \lambda\nu$$

$$\tau\beta = \lambda\nu$$

$$\rho\gamma = \lambda\nu$$

από όπου προκύπτει με πολλαπλασιασμό και των τριών σχέσεων, στρ = $\lambda\nu^2$ και κατόπιν,

$$\lambda = \sqrt[3]{\frac{\sigma\tau\rho}{\nu^2}}$$

$$\alpha = \sqrt[3]{\frac{\nu\tau\rho}{\sigma^2}}$$

$$\beta = \sqrt[3]{\frac{\nu\sigma\rho}{\tau^2}}$$

$$\gamma = \sqrt[3]{\frac{\nu\sigma\tau}{\rho^2}}$$

και το ελάχιστο κόστος επικοινωνίας είναι $K(\alpha, \beta, \gamma)_{\text{ελάχιστο}} = 3\sqrt[3]{\nu\sigma\tau\rho}$.

Από τις παραπάνω εξισώσεις οι τιμές των α, β, γ δεν είναι απαραίτητα ακέραιες παρόλο που πρέπει να είναι λόγω της φυσικής τους σημασίας, αναπαριστούν καλάθια. Συνεπώς, οι προκύψασες τιμές των α, β, γ αν δεν είναι ακέραιες τότε μπορούν να εκληφθούν ως προσεγγίσεις των κατάλληλων ακέραιων τιμών. Επίσης, οι παραπάνω εξισώσεις ορίζουν τους λόγους μεταξύ των μεριδίων, για παράδειγμα ισχύει $\frac{\alpha}{\beta} = \frac{\tau}{\sigma}$, δηλαδή, το μερίδιο μιας ιδιότητας στο μ – κλειδί είναι αντιστρόφως ανάλογο του μεγέθους της σχέσης από την οποία η αντίστοιχη ιδιότητα λείπει. Στην τελευταία εξίσωση το α, μερίδιο της ιδιότητας Α, είναι αντιστρόφως ανάλογο του μεγέθους την Σ, από την οποία σχέση Σ, η ιδιότητα Α απουσιάζει. Αυτή η αντίστροφη αναλογία συστήνει τον ισοκατανομή του κόστους επικοινωνίας των σχέσεων στις ρ – διεργασίες. Τέλος, οι λόγοι των μεριδίων βοηθούν και στην αναπροσαρμογή του πλήθους των ρ – διεργασιών ν σε κάποιο εύρος,

προκειμένου να υπάρχει ένα ελάχιστο δυνατό κόστος.

Αν τα μεγέθη των τριών σχέσεων θεωρηθούν ίσα, τότε, $K(\alpha, \beta, \gamma)_{\text{ελάχιστο}} = 3\rho\sqrt[3]{\nu}$ και το κόστος για τις μ – διεργασίες είναι 3ρ. Άρα, η πολυπλοκότητα του αλγορίθμου είναι $O(\rho\sqrt[3]{\nu})$. Αν ο εξεταζόμενος πολλαπλός σύνδεσμος υπολογιστεί ως ακολουθία δύο δυαδικών συνδέσμων, τότε το κόστος επικοινωνίας για τον πρώτο σύνδεσμο για τις μ – διεργασίες είναι 2ρ που είναι και το κόστος των ρ – διεργασιών του πρώτου συνδέσμου. Η έξοδος αυτών των ρ – διεργασιών έχει μέγεθος $\rho^2\pi$, όπου το π ισούται με την πιθανότητα δύο πλειάδες από διαφορετικές σχέσεις να έχουν για την κοινή τους ιδιότητα την ίδια τιμή. Με άλλα λόγια το π ισούται με την πιθανότητα σύνδεσης δύο πλειάδων από δύο διαφορετικές σχέσεις. Συνεπώς, οι μ – διεργασίες του δεύτερου συνδέσμου έχουν ένα κόστος $\rho^2\pi + \rho$ αφού λαμβάνουν ως είσοδο την έξοδο των ρ – διεργασιών του πρώτου συνδέσμου και την εναπομείνασα σχέση. Με $\rho\pi > 1$, ο όρος $\rho^2\pi$ κυριαρχεί και η πολυπλοκότητα των δύο διαδοχικών συνδέσμων είναι $O(\rho^2\pi)$. Όπως φαίνεται για τιμές του $\nu < (\rho\pi)^3$, ο πολλαπλός σύνδεσμος επιτυγχάνει καλύτερα αποτελέσματα από την ακολουθία των διαδοχικών συνδέσμων.

Τελειώνοντας, από το παράδειγμα αυτό φαίνεται ο συμβιβασμός που γίνεται ανάμεσα στην ταχύτητα και στο κόστος του αλγορίθμου. Το συνολικό κόστος επικοινωνίας είναι $O(\sqrt[3]{\nu\sigma\tau\rho})$ ενώ το κόστος διαβίβασης είναι $O(\sqrt[3]{\frac{\sigma\tau\rho}{\nu^2}})$, δηλαδή, το $\frac{1}{\nu}$ του κόστους επικοινωνίας θεωρώντας πως η συνάρτηση κατακερματισμού κατανέμει τις πλειάδες των σχέσεων ισοπίθανα κι άρα ισοκατανέμει τις πλειάδες στις ρ – διεργασίες. Συνεπώς, όταν το συνολικό κόστος επικοινωνίας αυξάνεται κατά $\nu^{\frac{1}{3}}$ τότε το κόστος επικοινωνίας διαβίβασης μειώνεται κατά $\nu^{\frac{2}{3}}$. Με άλλα λόγια, όσο γρηγορότερα υπολογίζεται ο πολλαπλός σύνδεσμος, τόσο περισσότεροι πόροι απαιτούνται. Διευκρινίζεται πως στο παραπάνω κόστος επικοινωνίας διαβίβασης δεν συμπεριλαμβάνεται το κόστος επικοινωνίας διαβίβασης των μ – διεργασιών καθώς πρακτικά θεωρούνται άπειρες στο πλήθος εφόσον λαμβάνουν κάποιο μέρος της συνολικής εισόδου.

Τελικά αυτό που πρέπει αρχικά να μείνει στον αναγνώστη είναι ότι στο περιβάλλον του Map – Reduce μπορεί ένας πολλαπλός σύνδεσμος να εκτελεστεί αποδοτικά κατευθείαν.

Ένα δεύτερο αξιοσημείωτο συμπέρασμα από την παραπάνω ανάλυση αποτελεί το γεγονός το αρχικό πρόβλημα επιτάχυνσης του πολλαπλού συνδέσμου ανάγεται στην εύρεση των βέλτιστων μεριδίων των ιδιοτήτων στο μ – κλειδί. Η απόδοση του πολλαπλού συνδέσμου επηρεάζεται από το πλήθος των ρ – διεργασιών ν όπως φαίνεται από την πολυπλοκότητα του αλγορίθμου στο προηγούμενο παράδειγμα. Επειδή το ν ισούται με το γινόμενο των μεριδίων των ιδιοτήτων, η απόδοση του αλγορίθμου συναρτάται άμεσα από το ν. Τελικά, το αρχικό πρόβλημα επιτάχυνσης του πολλαπλού συνδέσμου ανάγεται στην εύρεση των βέλτιστων μεριδίων των ιδιοτήτων στο μ – κλειδί.

Ένα τρίτο αξιόλογο σημείο συνίσταται στο ότι, η παραπάνω μέθοδος για την εύρεση της εξίσωσης κόστους και των βέλτιστων καλαθιών για τις συμμετέχουσες ιδιότητες στο μ – κλειδί μπορεί γενικά να επεκταθεί σε οποιοδήποτε πολλαπλό σύνδεσμο. Αν χρειάζεται ο υπολογισμός του πολλαπλού συνδέσμου των σχέσεων X_κ, $\kappa = 1, ..., \sigma$ που μοιράζονται τις ιδιότητες I_μ, $\mu = 1, ..., \xi$, τότε η εξίσωση κόστους για ν ρ – διεργασίες δίνεται από την εξίσωση,

$$K(\iota_1, \iota_2, ..., \iota_\xi) = \sum_{\kappa=1}^{\sigma} \chi_\kappa \gamma_\kappa$$

όπου,

χ_κ : το μέγεθος της σχέσης X_κ που θεωρείται γνωστό $\forall \kappa$,

$\gamma_\kappa = \gamma_\kappa(\iota_1, \iota_2, ..., \iota_\xi)$: το γινόμενο των μεριδίων των ιδιοτήτων στο μ – κλειδί που δεν συμμετέχουν στην σχέση X_κ και

ι_μ : το μερίδιο της ιδιότητας I_μ.

Η εξίσωση Lagrange σύμφωνα με την μέθοδο των πολλαπλασιαστών του Lagrange γράφεται,

$$\Lambda(\iota_1, \iota_2, ..., \iota_\xi) = \sum_{\kappa=1}^{\sigma} \chi_\kappa \gamma_\kappa - \lambda\left(\prod_{\mu=1}^{\xi} \iota_\mu - \nu\right)$$

Κατόπιν η εξίσωση αυτή παραγωγίζεται ως προς κάθε μία μεταβλητή της, δίνοντας μία εξίσωση της μορφής,

$$\Lambda_\beta(\iota_1, \iota_2, ..., \iota_\xi) = \frac{\partial \Lambda(\iota_1, \iota_2, ..., \iota_\xi)}{\partial \iota_\beta} = \sum_{\kappa=1}^{\sigma} \chi_\kappa \frac{\partial \gamma_\kappa(\iota_1, \iota_2, ..., \iota_\xi)}{\partial \iota_\beta} - \lambda \prod_{\mu=1, \mu\neq\beta}^{\xi} \iota_\beta$$

όπου, ι_β είναι το β – οστό μερίδιο, αντίστοιχο της ιδιότητας I_β, β = 1, ..., ξ. Έπειτα, η προηγούμενη εξίσωση πολλαπλασιάζεται με την μεταβλητή ως προς την οποία παραγωγίστηκε δίνοντας την εξίσωση,

$$\Lambda_\beta^\beta(\iota_1, \iota_2, ..., \iota_\xi) = \sum_{\kappa=1}^{\sigma} \chi_\kappa \frac{\partial \gamma_\kappa(\iota_1, \iota_2, ..., \iota_\xi)}{\partial \iota_\beta} \iota_\beta - \lambda\nu$$

όπου το $\nu = \prod_{\mu=1}^{\xi} \iota_\beta$.

Ως αποτέλεσμα αυτής της διαδικασίας προκύπτει ένα σύστημα ξ εξισώσεων της μορφής,

$$\Sigma = \begin{cases} \sum_{\kappa=1}^{\sigma} \chi_\kappa \frac{\partial \gamma_\kappa(\iota_1, \iota_2, ..., \iota_\xi)}{\partial \iota_1} \iota_1 - \lambda\nu = 0 \\ \sum_{\kappa=1}^{\sigma} \chi_\kappa \frac{\partial \gamma_\kappa(\iota_1, \iota_2, ..., \iota_\xi)}{\partial \iota_2} \iota_2 - \lambda\nu = 0 \\ \vdots \\ \sum_{\kappa=1}^{\sigma} \chi_\kappa \frac{\partial \gamma_\kappa(\iota_1, \iota_2, ..., \iota_\xi)}{\partial \iota_\xi} \iota_\xi - \lambda\nu = 0 \end{cases} \Leftrightarrow \Sigma = \begin{cases} \sum_{\kappa=1}^{\sigma} \chi_\kappa \frac{\partial \gamma_\kappa(\iota_1, \iota_2, ..., \iota_\xi)}{\partial \iota_1} \iota_1 = \lambda\nu \\ \sum_{\kappa=1}^{\sigma} \chi_\kappa \frac{\partial \gamma_\kappa(\iota_1, \iota_2, ..., \iota_\xi)}{\partial \iota_2} \iota_2 = \lambda\nu \\ \vdots \\ \sum_{\kappa=1}^{\sigma} \chi_\kappa \frac{\partial \gamma_\kappa(\iota_1, \iota_2, ..., \iota_\xi)}{\partial \iota_\xi} \iota_\xi = \lambda\nu \end{cases}$$

που μαζί με την εξίσωση $\nu = \prod_{\mu=1}^{\xi} \iota_\beta$ αρκούν για τον υπολογισμό των ξ + 1 μεταβλητών, ξ μεταβλητές – μερίδια των ιδιοτήτων I_μ, μ = 1, ..., ξ και μία μεταβλητή ο πολλαπλασιαστής Lagrange λ. Έχοντας τις τιμές αυτών των μεταβλητών – μεριδίων υπολογίζεται το ελάχιστο κόστος $K(\iota_1, \iota_2, ..., \iota_\xi) = \sum_{\kappa=1}^{\sigma} \mu_\kappa \chi_\kappa$.

Όμως αυτή η μέθοδος δεν μπορεί να γενικευθεί κατευθείαν, γιατί υπάρχουν κάποια ανοιχτά

ζητήματα. Εν παραδείγματι, αν ο σύνδεσμος είναι ο

$$P(A, B, \Gamma) \bowtie \Sigma(A, B, \Delta) \bowtie T(A, \Delta, E) \bowtie Y(\Delta, \Phi)$$

τότε, η εξίσωση Lagrange ακολουθώντας την περιγραφείσα γενική μέθοδο είναι

$$\Lambda(\alpha, \beta, \gamma, \delta, \varepsilon, \varphi) = \rho\delta\varepsilon\varphi + \sigma\gamma\varepsilon\varphi + \tau\beta\gamma\varphi + \upsilon\alpha\beta\gamma\varepsilon - \lambda(\alpha\beta\gamma\delta\varepsilon\varphi - \nu)$$

για ν ρ – διεργασίες, όπου τα α,β, γ,δ ε, φ είναι οι μεταβλητές – μερίδια των αντίστοιχων ιδιοτήτων και ρ, σ, τ, υ τα μεγέθη των αντίστοιχων σχέσεων. Τελικά, λαμβάνεται το σύστημα των εξισώσεων,

$$\rho\delta\varepsilon\varphi = \lambda\nu \qquad (1)$$

$$\upsilon\alpha\beta\gamma\varepsilon = \lambda\nu \qquad (2)$$

$$\tau\beta\gamma\varphi + \upsilon\alpha\beta\gamma\varepsilon = \lambda\nu \qquad (3)$$

$$\rho\delta\varepsilon\varphi + \sigma\gamma\varepsilon\varphi + \tau\beta\gamma\varphi = \lambda\nu \qquad (4)$$

$$\sigma\gamma\varepsilon\varphi + \tau\beta\gamma\varphi + \upsilon\alpha\beta\gamma\varepsilon = \lambda\nu \qquad (5)$$

$$\rho\delta\varepsilon\varphi + \sigma\gamma\varepsilon\varphi + \upsilon\alpha\beta\gamma\varepsilon = \lambda\nu \qquad (6)$$

Αν από την σχέση (3) αφαιρεθεί η σχέση (2) τότε, τβγφ = 0 και τουλάχιστον μία από τις μεταβλητές β, γ, φ πρέπει να ισούται με το 0. Κάτι τέτοιο δεν έχει φυσικό νόημα προφανώς. Για κάθε ιδιότητα πρέπει να υπάρχει τουλάχιστον ένα καλάθι στο οποίο θα αποστέλλονται οι τιμές της. Αν από την σχέση (5) αφαιρεθεί η σχέση (3) τότε, σγεφ = 0 και παρουσιάζεται το ίδιο πρόβλημα.

Για να ξεπεραστεί αυτή η δυσκολία εισάγεται η έννοια της κυρίαρχης ιδιότητας. Σύμφωνα με αυτή κάποια ιδιότητα Χ κυριαρχεί κάποιας άλλης ιδιότητας Υ, όταν σε κάθε σχέση που συμμετέχει η Υ, συμμετέχει και η Χ. Η ιδιότητα Χ λέγεται κυρίαρχη ιδιότητα κι η Υ κυριαρχούμενη ιδιότητα.

Σε κάθε κυριαρχούμενη ιδιότητα, το μερίδιο της στο μ – κλειδί τίθεται ίσο με 1. Οι τιμές κάθε κυριαρχούμενης ιδιότητας αποστέλλονται σε ένα συγκεκριμένο καλάθι.

Επιστρέφοντας στο παράδειγμα, σε κυρίαρχες ιδιότητες αναδεικνύονται οι Α και Δ. Κατά επέκταση,

$$\Lambda(\alpha, \beta, \gamma, \delta, \varepsilon, \varphi) = \Lambda(\alpha, \delta) = \rho\delta + \sigma + \tau + \upsilon\alpha - \lambda(\alpha\delta - \nu)$$

και

$$\nu = \alpha\delta$$

$$\upsilon\alpha = \lambda\nu$$

$$\rho\delta = \lambda\nu$$

λαμβάνοντας τελικά, $\lambda = \sqrt{\frac{\rho\upsilon}{\nu}}$, $\alpha = \sqrt{\frac{\rho\nu}{\upsilon}}$ και $\delta = \sqrt{\frac{\upsilon\nu}{\rho}}$.

Το ελάχιστο κόστος είναι $K(\alpha, \beta, \gamma, \delta, \varepsilon, \varphi) = K(\alpha, \delta) = 2\sqrt{\upsilon\rho\nu} + \sigma + \tau$.

Από το παραπάνω παράδειγμα, φαίνεται πως η έννοια της κυρίαρχης ιδιότητας λύνει το πρόβλημα όπου κάποιο μερίδιο ισούται με το μηδέν. Όμως υπάρχουν περιπτώσεις που η κυρίαρχη ιδιότητα δεν εξαλείφει όλες τις περιπτώσεις που ένα μερίδιο ισούται με το 0. Για αυτό το λόγο, αφού εφαρμοστεί στο πρόβλημα η εξίσωση της κυρίαρχης ιδιότητας εφαρμόζεται μια γενικότερη

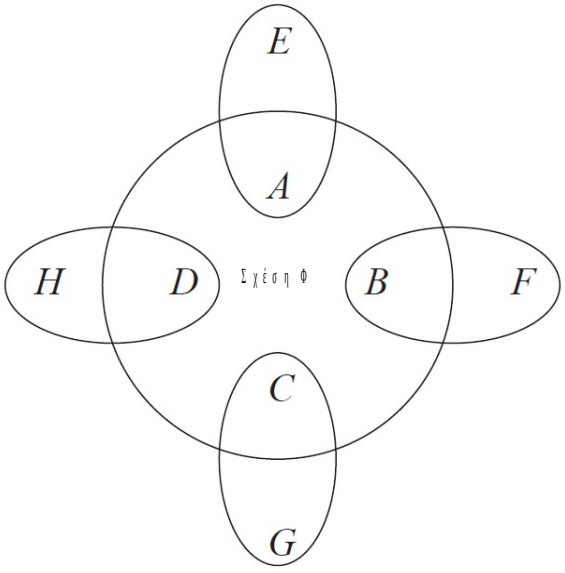
μέθοδος για την εξάλειψη όλων των υπόλοιπων προβληματικών περιπτώσεων. Πρακτικά σύμφωνα με την μέθοδο αυτή, αφού ληφθεί η εξίσωση Lagrange που περιέχει μόνο κυρίαρχες ιδιότητες, τότε κάθε ιδιότητα, μία την φορά, τίθεται ίση με την μονάδα και λύνεται το αντίστοιχο πρόβλημα. Στο νέο πρόβλημα που προκύπτει γίνεται το ίδιο μέχρις ότου να μείνει μόνο μία ιδιότητα όπου σταματά η αναδρομή. Σε κάθε ένα από τα προβλήματα που προκύπτουν καθώς και στο αρχικό υπολογίζεται ένα κόστος εφόσον η λύση του συστήματος των εξισώσεων έχει νόημα. Το πρόβλημα με το μικρότερο κόστος αποτελεί την λύση του αρχικού προβλήματος. Υπενθυμίζεται πως τελικά το αρχικό πρόβλημα αποτελεί η εύρεση των βέλτιστων μεριδίων των ιδιοτήτων στο μ – κλειδί.

Συγκεντρωτικά, ο αλγόριθμος αναλύεται σε πέντε βήματα,

1. βρίσκονται οι κυρίαρχες ιδιότητες. Αν δύο ή περισσότερες ιδιότητες συμμετέχουν σε όλες τις σχέσεις τότε κρατείται μία από όλες κι οι υπόλοιπες τίθενται να έχουν μερίδιο ίσο με 1.
2. Κατασκευάζεται η εξίσωση κόστους όπως έχει ήδη υποδειχθεί.
3. Κατασκευάζεται η εξίσωση Lagrange.
4. Εξαλείφονται οι εναπομείνασες περιπτώσεις όπου κάποια μερίδια ισούνται με το 0.
5. Επιλέγεται ως λύση εκείνη που ανάμεσα σε όλες τις παραχθείσες για τα διάφορα υπό – προβλήματα του βήματος 4, έχει το μικρότερο κόστος.

Εφαρμόζοντας τον παραπάνω αλγόριθμο σε δύο ειδικές και συχνές μορφές πολλαπλού συνδέσμου προκύπτουν για κάθε μία περίπτωση κλειστοί αναλυτικοί τύποι για τον υπολογισμό των μεριδίων.

Η πρώτη περίπτωση αφορά στον σύνδεσμο – αστέρα, όπου υπάρχει μια κεντρική σχέση και μια πληθώρα άλλων περιφερειακών, αρκετά μικρότερων σχέσεων που έχουν ακριβώς μία κοινή ιδιότητα με την κεντρική σχέση. Λέγοντας “ακριβώς μία κοινή ιδιότητα” εννοείται μια σύνθετη ιδιότητα η οποία συνδυάζει σε μία όλες τις κοινές ιδιότητες ανάμεσα στην κεντρική σχέση και σε μια συγκεκριμένη περιφερειακή σχέση. Επίσης, μια κεντρική σχέση μπορεί να έχει κι ιδιότητες που δεν συμμετέχουν σε καμία άλλη περιφερειακή σχέση. Προφανώς οι ιδιότητες αυτές χαρακτηρίζονται κυριαρχούμενες και δεν διαδραματίζουν κάποιον ρόλο. Ένα παράδειγμα φαίνεται στο σχήμα 22, όπου η σχέση Φ αποτελεί την κεντρική σχέση κι όλες οι άλλες σχέσεις χαρακτηρίζονται περιφερειακές.



Σχήμα 26 : Παράδειγμα συνδέσμου – αστέρα

Συμπερασματικά, για τον σύνδεσμο αστέρα, αν $\pi = \pi_1\pi_2...\pi_\sigma$ είναι το γινόμενο των μεγεθών των σ περιφερειακών σχέσεων κι α_κ είναι η κοινή ιδιότητα μεταξύ της κεντρικής σχέσης και της περιφερειακής σχέσης Π_κ, τότε, το $\alpha_\kappa = \pi_\kappa \sqrt[\sigma]{\frac{\nu}{\pi}}$, όπου το ν ισούται με το πλήθος των ρ – διεργασιών.

Η δεύτερη περίπτωση αφορά στον σύνδεσμο αλυσίδα που έχει την μορφή,

$$P_1(I_0, I_1) \bowtie P_2(I_1, I_2) \bowtie P_3(I_2, I_3) \bowtie \ldots \bowtie P_{\sigma-1}(I_{\nu-2}, I_{\nu-1}) \bowtie P_\sigma(I_{\nu-1}, I_\nu).$$

Φυσικά οι σχέσεις του συνδέσμου μπορούν να έχουν κι άλλες δικές τους μοναδικές ιδιότητες που επειδή κυριαρχούνται από τις άλλες, τις εμφανιζόμενες ιδιότητες, δεν επηρεάζουν το αποτέλεσμα και για αυτό δεν παρουσιάζονται. Σχετικά με το μέγεθος των μεριδίων των κυρίαρχων ιδιοτήτων αυτό εξαρτάται από το αν το πλήθος των συμμετεχόντων σχέσεων είναι περιττό ή άρτιο. Αν το πλήθος είναι περιττό και για αυθαίρετο μέγεθος των σχέσεων ο τύπος, είναι,

για μερίδια με άρτιο δείκτη

$$\alpha_{2\kappa} = (\alpha_2)^\kappa \prod_{\mu=2}^{\kappa} \frac{\rho_1 \rho_{2\mu}}{\rho_2 \rho_{2\mu-1}}$$

$$\alpha_{\nu-2} = \frac{1}{\alpha_2} \frac{\rho_2 \rho_{\nu-1}}{\rho_1 \rho_\nu}$$ και για μερίδια με περιττό δείκτη $\alpha_{2\kappa+1} = \alpha_1 \frac{\rho_{2\kappa+1}}{\rho_1 \alpha_{2\kappa}}$ ως προς

$$\alpha_2 = \left(\prod_{\mu=2}^{\frac{\sigma}{2}} \frac{\rho_2 \rho_{2\mu-1}}{\rho_1 \rho_{2\mu}} \right)^{\frac{2}{\sigma}}$$

την μεταβλητή α_1. Η μεταβλητή α_1 υπολογίζεται με την βοήθεια της σχέσης $\alpha_1\alpha_2\ldots\alpha_{\nu-1}\alpha_\nu = \nu$, ν το πλήθος των ρ – διεργασιών.

Αν το πλήθος των σχέσεων σ είναι περιττό, τότε, για μερίδια με άρτιο δείκτη $\alpha_{2\kappa} = (\alpha_2)^\kappa \prod_{\mu=2}^{\kappa} \frac{\rho_1 \rho_{2\mu}}{\rho_2 \rho_{2\mu-1}}$ και για μερίδια με περιττό δείκτη $\alpha_{\sigma-2\kappa} = (\alpha_2)^\kappa \prod_{\mu=1}^{\kappa} \frac{\rho_1 \rho_{\sigma-2\mu+1}}{\rho_2 \rho_{\sigma-2\mu+2}}$ όπου η μεταβλητή α_2 υπολογίζεται με την βοήθεια της σχέσης $\alpha_1\alpha_2\ldots\alpha_{\nu-1}\alpha_\nu = \nu$, ν το πλήθος των ρ – διεργασιών.

Στο σημείο αυτό ολοκληρώνεται η θεωρητική εξέταση της τρέχουσας μεθόδου. Απομένει ένα τελευταίο σχόλιο. Κατά την εφαρμογή της μεθόδου αυτής μπορεί να προκύψουν αρνητικές λύσεις ή λύσεις που έχουν τιμή μικρότερη της 1 δίνοντας ένα μικρότερο κόστος επικοινωνίας. Σε αυτές τις περιπτώσεις θα πρέπει ο χώρος αναζήτησης να περιοριστεί σε εκείνη την περιοχή όπου εγγυημένα όλα τα μερίδια των ιδιοτήτων στο μ – κλειδί να ισούνται τουλάχιστον με την μονάδα κι ας είναι το κόστος επικοινωνίας μεγαλύτερο. Επίσης, η απαίτηση κάθε μερίδιο να είναι τουλάχιστον ίσο με την 1 μπορεί να αποκλείσει βέλτιστες λύσεις όπου τα διάφορα μερίδια έχουν τιμή μεγαλύτερη της 1.

Τέλος, οι διάφορες λύσεις δεν είναι απαραίτητα ακέραιες. Όμως τα μερίδια των ιδιοτήτων στο μ – κλειδί θα πρέπει να έχουν ακέραια τιμή. Μια λύση σε αυτό αποτελεί η λήψη των κοντινότερων ακέραιων τιμών. Κάτι τέτοιο όμως δεν συνεπάγεται και την λήψη της βέλτιστης λύσης για ακέραιες τιμές. Επιπλέον, η λήψη των όποιων ακέραιων τιμών θα πρέπει να σέβεται και την απαίτηση το γινόμενο των μεριδίων να ισούται με μια σταθερά ν. Όπως φαίνεται το φυσικό νόημα των μεριδίων εισάγει περιορισμούς που δεν συμβαδίζουν απαραίτητα με την ευρεθείσα βέλτιστη λύση ή την απαίτηση το γινόμενο των μεριδίων να ισούται με ν. Για αυτό τελικά τόσο το ν όσο κι οι δεκαδικές τιμές των μεριδίων θα πρέπει να εκλαμβάνονται περισσότερο ως καθοδήγηση προς την βέλτιστη λύση παρά σαν θέσφατα.

Η πειραματική μελέτη της τρέχουσας μεθόδου και τα συμπεράσματα αυτής εκτίθενται στα κεφάλαια 4 και 5.

ΚΕΦΑΛΑΙΟ 3

ΥΛΟΠΟΙΗΣΗ ΠΟΛΛΑΠΛΟΥ ΣΥΝΔΕΣΜΟΥ

Ο κυριότερος στόχος της εκπαίδευσης δεν είναι η γνώση αλλά η δράση.
Herbert Spencer

Με μια ματιά

Στο τελευταίο μέρος του δεύτερου κεφαλαίου συζητήθηκε εκτενώς μια μέθοδος πολλαπλού συνδέσμου που στο περιβάλλον του Map – Reduce μπορεί να επιτύχει καλύτερη απόδοση σε σχέση με μια ακολουθία δυαδικών συνδέσμων. Στο κεφάλαιο αυτό συζητείται εκτενώς η πραγμάτωση αυτής της μεθόδου. Για την ακρίβεια δίνονται κάποιες παραπάνω λεπτομέρειες σχετικά με το περιβάλλον Map – Reduce, το Hadoop και τελικά την ίδια την υλοποίηση ώστε να δοθούν μια καλύτερη εικόνα της μεθόδου και μια μεγαλύτερη δυνατότητα κατανόησης των πειραματικών αποτελεσμάτων του επόμενου κεφαλαίου.

3.1 Map – Reduce

Με την έκρηξη των δεδομένων, η όποια διαχείριση κι επεξεργασία αυτών μεταφέρθηκε σε ένα κατανεμημένο περιβάλλον όπου υπήρχαν πολλά μηχανήματα που λειτουργούσαν εν παραλλήλω. Μόνο σε αυτό το περιβάλλον ήταν δυνατό μέσα σε ένα εύλογο χρονικό διάστημα να ολοκληρωθεί η κάθε λειτουργία επί των δεδομένων. Σε αυτό το περιβάλλον έπρεπε οι προγραμματιστές να αναλάβουν την διανομή, τον διαμερισμό των δεδομένων στις επεξεργαστικές μονάδες και τον ίδιο τον παραλληλισμό των μονάδων αυτών. Η ρύθμιση του κατανεμημένου περιβάλλοντος προφανώς εμπόδιζε την κύρια εργασία των προγραμματιστών που ήταν απλά και μόνο η επεξεργασία των προαναφερθέντων δεδομένων.

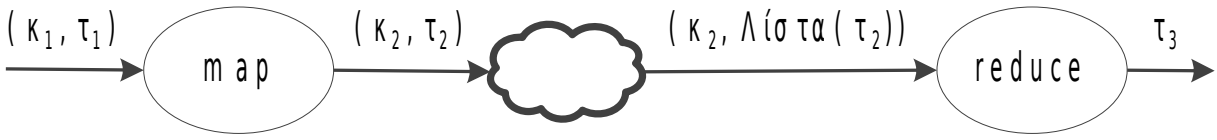
Το Map – Reduce λοιπόν αναπτύχθηκε προκειμένου να διευκολύνει τους προγραμματιστές να ασχοληθούν μόνο με κύρια εργασία τους. Ο διαμοιρασμός των δεδομένων, η διανομή αυτών κι ο παραλληλισμός των εργασιών σε ένα κατανεμημένο περιβάλλον, εύρωστο στις όποιες αστοχίες του υλικού, έγιναν ευθύνη του Map – Reduce. Το Map – Reduce ουσιαστικέ διαχώρισε τους υπολογισμούς σε ένα κατανεμημένο περιβάλλον από το περιβάλλον αυτό κάθε αυτό.

Πέρα από την παραπάνω διευκόλυνση το Map – Reduce διαθέτει ένα σύνολο χαρακτηριστικών που το καθιστούν ελκυστικό για κάποιες εφαρμογές. Τα χαρακτηριστικά αυτά είναι η αντοχή σε σφάλματα του υλικού(fault tolerance), η ευκολία επέκτασης του με την προσθήκη παραπάνω υλικού ή την αύξηση του όγκου των δεδομένων(scalability), το σύστημα αρχείων που προσφέρει για πάρα πολλά δεδομένα π.χ. GFS, HDFS και τέλος η δυνατότητα παραλληλισμού των εργασιών. Για όλα αυτά τα χαρακτηριστικά, το Map – Reduce αποτελεί πρώτη επιλογή για κάποιες εφαρμογές.

Για να επιτύχει το Map – Reduce το στόχο του, χωρίζει όλες τις εργασίες σε δύο είδη, τις map και τις reduce. Οι εργασίες αυτές εμπνευσμένες από τον συναρτησιακό προγραμματισμό, υποβάλλονται στο σύστημα και το σύστημα αναλαμβάνει την εκτέλεση τους από το κατανεμημένο περιβάλλον. Οι εργασίες αυτές κάθε αυτές δέχονται ως είσοδο δεδομένα που έχουν την μορφή ζεύγους κλειδί – τιμή. Το κλειδί στο πλαίσιο του Map – Reduce δεν είναι μοναδικό όπως σε μια βάση δεδομένων, αλλά χρειάζεται προκειμένου τα δεδομένα εξόδου των μ – διεργασιών[13] να αποσταλούν στις κατάλληλες ρ – διεργασίες.

Ακριβέστερα, μια μ – διεργασία δέχεται δεδομένα στην μορφή κλειδί – τιμή και παράγει ένα άλλο ζεύγος κλειδί – τιμή. Μετά, όλα τα ζεύγη που έχουν το ίδιο κλειδί συγχωνεύονται σε ένα ζεύγος από το ίδιο κλειδί και μια λίστα από όλες τις τιμές που αρχικά είχαν αυτό το κλειδί. Αυτό το νέο ζεύγος αποστέλλεται στην κατάλληλη ρ – διεργασία η οποία παράγει την τελική επιθυμητή τιμή. Όλο αυτό φαίνεται στο σχήμα 27, όπου το σύννεφο ανάμεσα στην μ – διεργασία και την ρ – διεργασία υποδηλώνει τον μηχανισμό μέσω του οποίου τα ζεύγη με το ίδιο κλειδί συγχωνεύονται σε ένα ζεύγος με το ίδιο κλειδί και μια λίστα από τιμές.

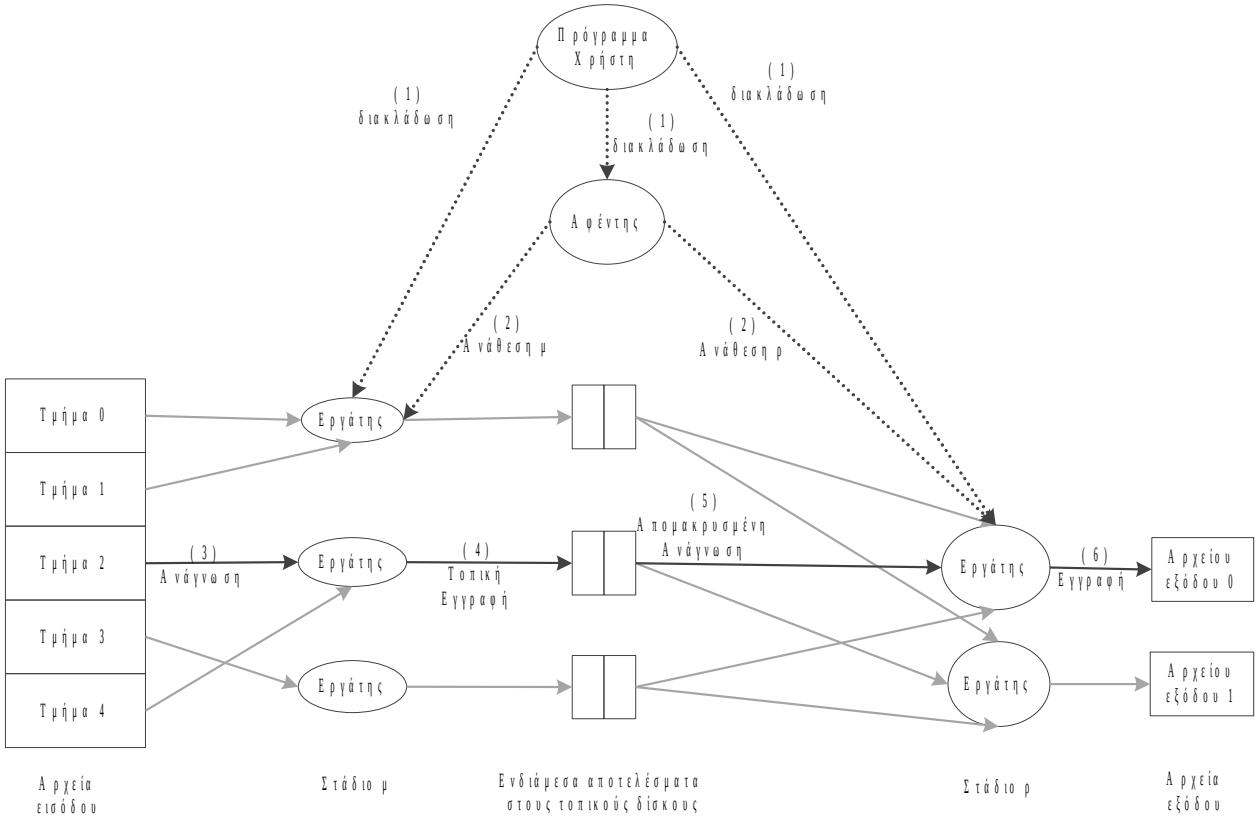
13 Ακολουθείται η ίδια σύμβαση με την παράγραφο 2.4 του δεύτερου κεφαλαίου, όπου οι διεργασίες map λέγονται μ – διεργασίες κι οι διεργασίες reduce λέγονται ρ – διεργασίες.



Σχήμα 27 : Διάγραμμα ροής από μια μ σε μια ρ – διεργασία

Όπως αναφέρθηκε, το Map – Reduce αφορά σε μια συστάδα από υπολογιστές που όλοι μαζί παράλληλα εργάζονται για την επίτευξη μιας εργασίας. Κάθε ένας από αυτούς τους υπολογιστές έχει εν γένει τα ίδια χαρακτηριστικά με τους υπόλοιπους και μπορεί εξίσου καλά να αναλάβει μια μ – διεργασία όσο και μια ρ – διεργασία. Για αυτό, κάθε ένας υπολογιστής εννοείται ως εργάτης ή εργαζόμενος ανεξάρτητα από το είδος της διεργασίας που διεκπεραιώνει. Όλοι οι εργάτες ή οι εργαζόμενοι εργάζονται παράλληλα κάτω από την καθοδήγηση ενός υπολογιστή – επόπτη που λέγεται αφέντης(master). Ο αφέντης αφενός συντονίζει τους εργάτες, αφετέρου έχει την ευθύνη για την επιτέλεση του έργου. Επίσης, σε περίπτωση που κάποιος εργαζόμενος βγει εκτός λειτουργίας ο αφέντης επιφορτίζεται με τις συνέπειες αυτού του γεγονότος. Τέλος, ο αφέντης διαμεσολαβεί ανάμεσα στο αίτημα του χρήστη και του εργάτες που θα εκτελέσουν το αίτημα του χρήστη. Προφανώς, οι εργάτες των μ – διεργασιών διαβάζουν τα δεδομένα τους από κάποια είσοδο κι οι εργάτες των ρ – διεργασιών γράφουν τα δεδομένα του σε κάποια έξοδο. Όλα τα προηγούμενα απεικονίζονται στο σχήμα 28. Στο σχήμα αυτό, τα στάδια μ και ρ αναφέρονται στις διεργασίες μ και ρ αντίστοιχα. Επίσης, οι αριθμοί δείχνουν την διαδικασία που ακολουθείται κατά τους διάφορους υπολογισμούς. Τα τμήματα των αρχείων εισόδου υποδηλώνουν τον διαμερισμό που υφίσταται η είσοδος.

Τέλος, όσον αφορά στον προγραμματισμό σε ένα σύστημα Map – Reduce, αυτό που θα πρέπει να μείνει από την παραπάνω παρουσίαση συνίσταται στο ότι για να εκτελεστεί μια εργασία θα πρέπει αυτή να διασπαστεί κατάλληλα σε μια μ – διεργασία και σε μια ρ – διεργασία. Η διάσπαση αυτή θα φανεί και πρακτικά στις επόμενες σελίδες αυτού του κεφαλαίου.



Σχήμα 28 : Αρχιτεκτονική του περιβάλλοντος Map – Reduce [25]

3.2 Hadoop

Το Hadoop αποτελεί μια ανοιχτού κώδικα ενσάρκωση του Map – Reduce από τον μη κερδοσκοπικό οργανισμό λογισμικού Apache. Είναι υλοποιημένο σε Java.

Η γενική αρχιτεκτονική του Hadoop παρουσιάζεται στο σχήμα 29 και διαποτίζεται από την γενική αρχή του διαχωρισμού. Σύμφωνα με αυτήν την αρχή, διαφορετικά πράγματα κρατιούνται χώρια. Η αρχή αυτή εφαρμόζεται κατά κόρον στην Πληροφορική. Ενδεικτικά, το Map – Reduce διαχωρίζει το κατανεμημένο περιβάλλον από τους υπολογισμούς που γίνονται σε αυτό. Ή στο διαδίκτυο το περιεχόμενο κι η λειτουργία των ιστοσελίδων διαχωρίζονται από την εμφάνιση αυτών. Το Hadoop εφαρμόζει την αρχή αυτή διαχωρίζοντας τα δεδομένα από τους όποιους υπολογισμούς γίνονται σε αυτά.

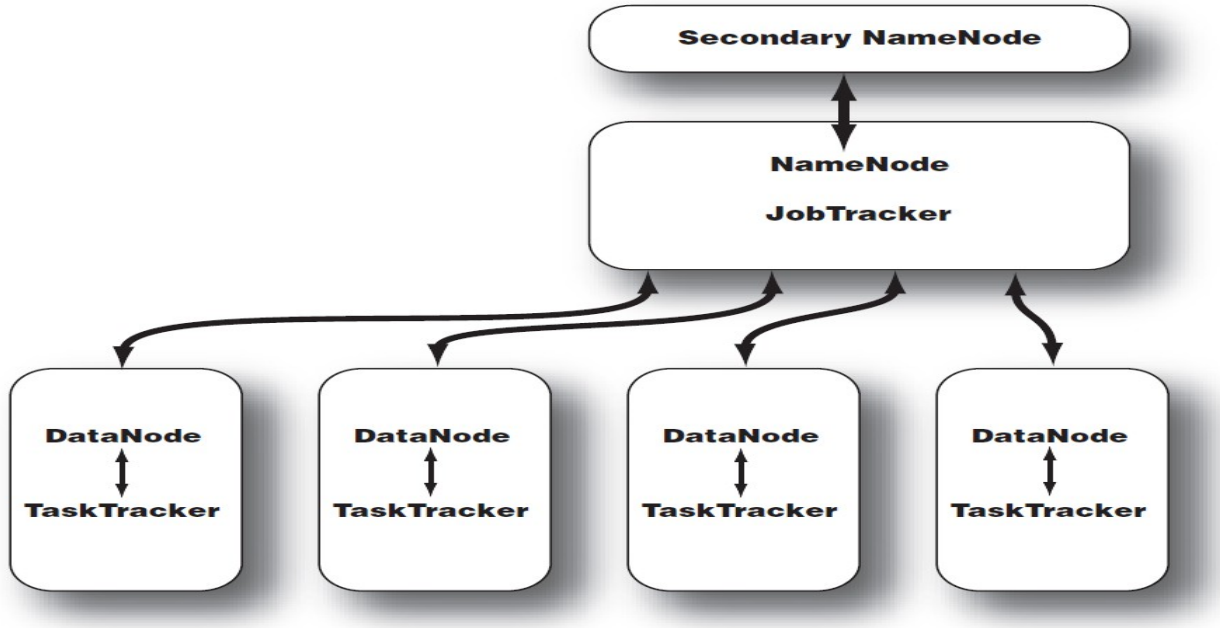
Όπως ειπώθηκε προηγουμένως, στο Map – Reduce υπάρχει ένας υπολογιστής αφέντης. Αυτός στο Hadoop χωρίζεται στον αφέντη των δεδομένων και στον αφέντη των υπολογισμών. Ο πρώτος λέγεται NameNode κι ευθύνεται για την διαχείριση των δεδομένων τα οποία φυλάσσονται στο κατανεμημένο σύστημα αρχείων του Hadoop, το HDFS. Η διαχείριση των δεδομένων περιλαμβάνει λίγο πολύ την δουλεία ενός βιβλιοθηκάριου. Ο NameNode γνωρίζει την κατάσταση του HDFS(βιβλιοθήκης) και πως τα δεδομένα(βιβλία) έχουν διαμεριστεί και κατανεμηθεί στους εργάτες(αναγνώστες). Ο δεύτερος αφέντης λέγεται JobTracker κι ευθύνεται για την εκτέλεση της εργασίας. Σχεδιάζει το σχέδιο εκτέλεσης της εργασίας ορίζοντας ποια αρχεία θα χρειαστούν, ποιοι εργάτες θα αναλάβουν την εργασία και παρακολουθεί την όλη διαδικασία. Αν κάποιος εργάτης – υπολογιστής καταρρεύσει φροντίζει για την επανεκκίνηση της εργασίας από κάποιον άλλον εργάτη.

Με παρόμοιο τρόπο κι ο προαναφερθείς εργάτης χωρίζεται στον εργάτη των δεδομένων και στον εργάτη των υπολογισμών. Ο πρώτος λέγεται DataNode κι ο δεύτερος λέγεται TaskTracker. Ο TaskTracker εργάζεται επί των δεδομένων του DataNode. Για αυτό υπάρχει στενή αλληλεπίδραση μεταξύ των δύο. Ο πρώτος δίνει αναφορά μόνο στον NameNode και ευθύνεται για τις λειτουργίες εισόδου – εξόδου, αναγνώσεις κι εγγραφές στο HDFS. Επίσης, συνδιαλέγεται απευθείας με τον κώδικα – πελάτη ώστε να διατεθούν τα απαραίτητα δεδομένα. Ακόμα οι εργάτες αυτού του είδους συνεργάζονται προκειμένου να αντιγραφούν κάποια δεδομένα προς όφελος της όλης εργασίας. Ο δεύτερος εργάτης λογοδοτεί μόνο στον JobTracker κι αναλαμβάνει την εκτέλεση των όποιων υπολογισμών, δηλαδή, την εκτέλεση των μ και ρ – διεργασιών. Υπενθυμίζεται πως αυτές οι διεργασίες μπορούν να έχουν πολλά στιγμιότυπα ταυτόχρονα. Για αυτό ένας τέτοιος εργάτης μπορεί να ξεκινήσει πολλά JVM για να “τρέχει” αυτά τα στιγμιότυπα παράλληλα. Τέλος, ο TaskTracker ανά τακτά χρονικά διαστήματα επικοινωνεί με τον αφέντη του, γιατί αυτό αποτελεί ένδειξη ότι όλα πάνε καλά. Αν διακοπεί αυτή η επικοινωνία ο JobTracker αναλαμβάνει δράση. Στο σχήμα 30 φαίνεται η αλληλεπίδραση των TaskTracker, JobTracker και κώδικα – πελάτη όταν ο τελευταίος αναθέτει μια εργασία στον δεύτερο, κι ο δεύτερος στην συνέχεια αναθέτει την εργασία σε πολλές μ και ρ – διεργασίες σε κάθε έναν TaskTracker του συμπλέγματος.

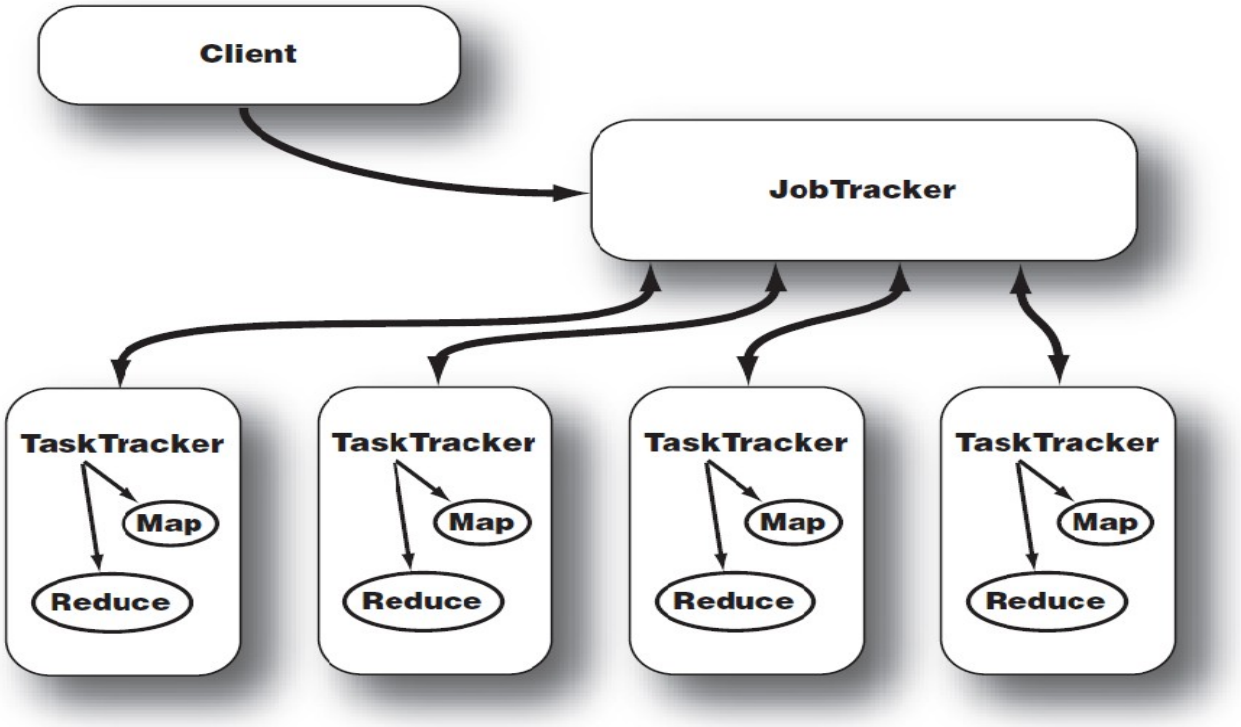
Σε όλη αυτή την αρχιτεκτονική, υπάρχουν πάρα πολλοί εργάτες ενδεχομένως και κατά χιλιάδες, ενώ οι αφέντες χαρακτηρίζονται μοναδικοί. Επειδή κι ο αφέντης μπορεί να καταρρεύσει υπάρχει ένας “φωτογράφος”, ο SecondaryNameNode(SNN) ο οποίος περιοδικά κρατάει ένα στιγμιότυπο των μετά – δεδομένων του HDFS. Αν ο NameNode “πέσει” τότε χάρη στον SecondaryNameNode μπορεί το σύστημα χειροκίνητα να ανακάμψει.

Κλείνοντας, τόσο οι αφέντες όσο κι οι εργάτες ως φυσικές οντότητες εντός του συμπλέγματος(cluster) των υπολογιστών του κατανεμημένου περιβάλλοντος, αντιστοιχούν σε κάποιο υπολογιστή. Ένας DataNode κι ένας TaskTracker συνήθως βρίσκονται στο ίδιο μηχάνημα. Ο NameNode κι ο JobTracker για ένα μικρό σύμπλεγμα κείνται στο ίδιο μηχάνημα ενώ για ένα

μεγαλύτερο κείνται σε διαφορετικά. Ο SecondaryNameNode για μικρά συμπλέγματα συνυπάρχει σε έναν υπολογιστή με τους τοπικούς DataNode, TaskTracker.



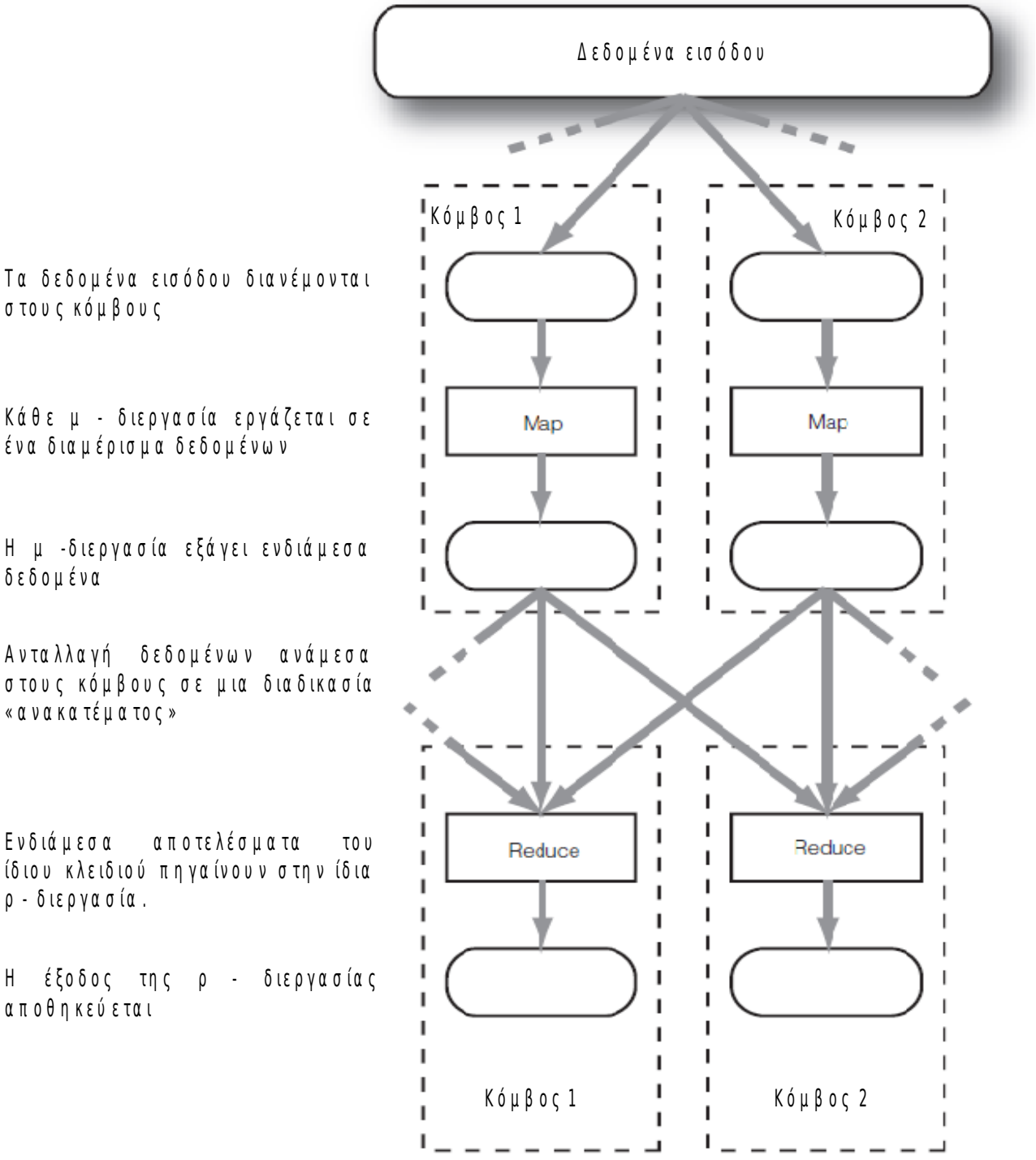
Σχήμα 29 : Αρχιτεκτονική ενός συμπλέγματος Hadoop [26]



Σχήμα 30 : Αλληλεπίδραση πελάτη, JobTracker, TaskTracker [26]

3.3 Map – Reduce και Hadoop

Στο σημείο αυτό παρουσιάζεται αφαιρετικά το γενικό σχήμα υλοποίησης του Map – Reduce από το Hadoop ως διαδικασία εκτέλεσης κάποιας εργασίας. Αυτό φαίνεται στο σχήμα 31 όπου κάθε κόμβος αντιστοιχεί σε ένα μηχάνημα – υπολογιστή του συμπλέγματος.

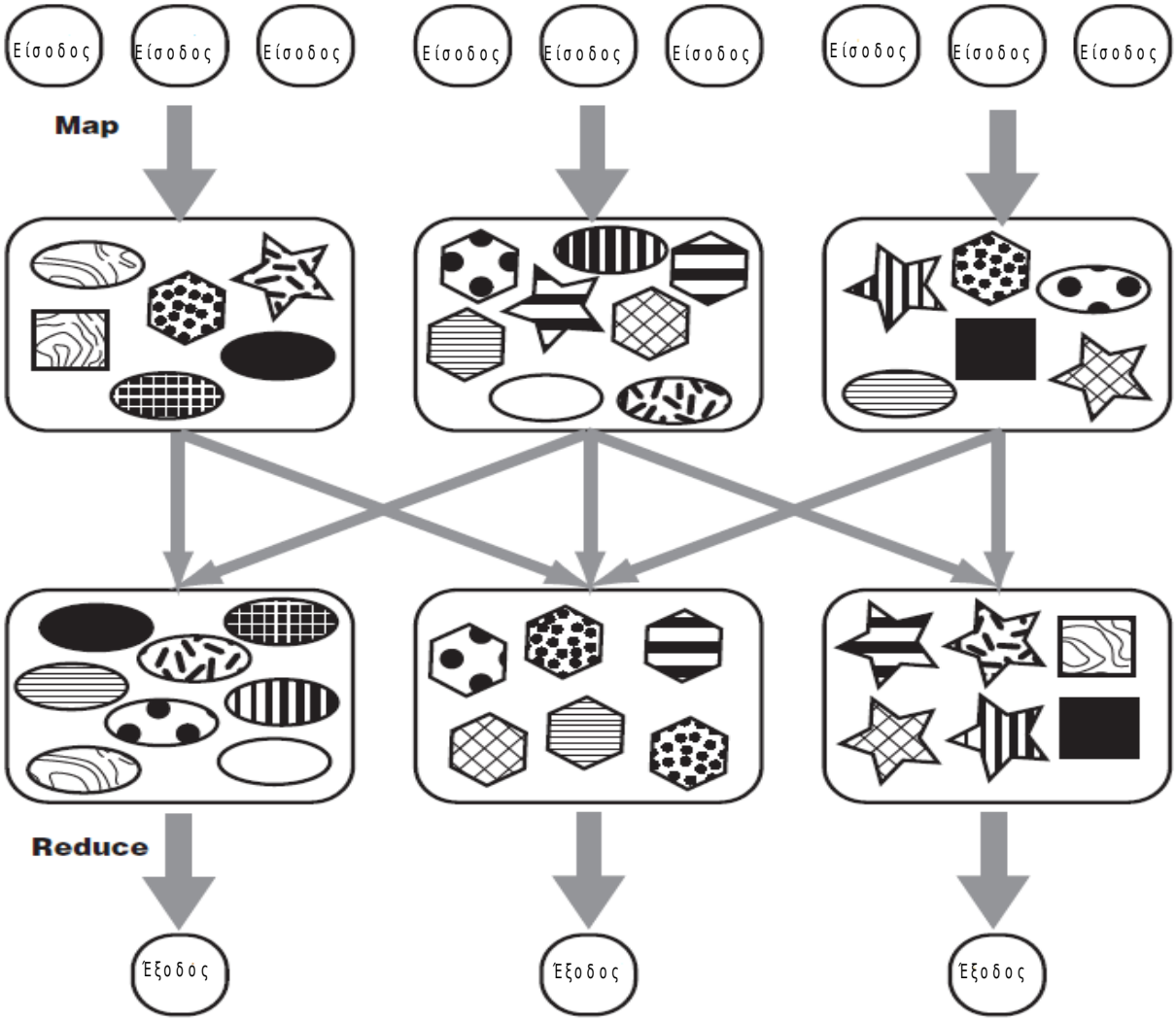


Σχήμα 31 : Το Map - Reduce στο Hadoop [26]

Ξεκινώντας, τα δεδομένα εισόδου αποθηκεύονται ως μια ακολουθία αρχείων στο σύστημα αρχείων του Hadoop(Hadoop Data File System – HDFS)[14]. Κάθε ένα από αυτά τα αρχεία έχει ένα συγκεκριμένο μέγεθος με εξαίρεση το τελευταίο αρχείο που έχει μικρότερο μέγεθος. Ο διαμελισμός των δεδομένων σε αρχεία κάποιου μεγέθους, ευνοεί την παράλληλη επεξεργασία τους και τα γενικότερα χαρακτηριστικά του συστήματος όπως η ευρωστία αυτού(fault tolerance). Βέβαια, αν το μέγεθος των αρχείων είναι πολύ μικρό, τότε ο χρόνος εκκίνησης και διακοπής της επεξεργασίας ενός τέτοιου αρχείου αποτελούν μια σημαντική προσαύξηση στον χρόνο εκτέλεσης. Τα δεδομένα εισόδου και γενικώς όλα τα δεδομένα που διαχειρίζεται το Map – Reduce έχουν την μορφή ζεύγους κλειδί – τιμή.

Κάθε μ – διεργασία λαμβάνει ένα ή κάποια από τα προαναφερθέντα αρχεία τα οποία επεξεργάζεται καταλλήλως. Η έξοδος αυτής της επεξεργασίας αποτελεί ένα πλήθος από ζεύγη κλειδιών – τιμών που δεν έχουν κάποια συγκεκριμένη σειρά. Προτού τα δεδομένα εξόδου των μ – διεργασιών φτάσουν στις ρ – διεργασίες, ανακατανέμονται με βάση το κλειδί τους μέσω μιας διαδικασίας “ανακατέματος”. Το “ανακάτεμα” δηλώνει το γεγονός πως η έξοδος μιας μ – διεργασίας σε έναν κόμβο μπορεί να σταλεί σε ρ – διεργασίες που βρίσκονται σε πολλούς άλλους κόμβους του συμπλέγματος κι οπότε κατά μία έννοια έρχονται τα “πάνω – κάτω”. Η ανακατανομή των δεδομένων εξόδου των μ – διεργασιών απεικονίζεται πιο παραστατικά στο σχήμα 32.

14 Στην παράγραφο 2.4 του δεύτερου κεφαλαίου, γίνεται μια αναλυτικότερη αναφορά στο HDFS και των όμοιων συστημάτων αρχείων.



Σχήμα 32 : Ανακατανομή και διανομή της εξόδου των μ – διεργασιών [26]

Στο σχήμα 32, κάθε εικόνα αντιστοιχεί σε ένα ζεύγος κλειδί – τιμή. Κάθε σχήμα ισοδυναμεί με το κλειδί και το περιεχόμενο κάθε σχήματος ισοδυναμεί με την τιμή. Μετά από το “ανακάτεμα” όλα τα σχήματα του ίδιου τύπου εντοπίζονται στην ίδια ρ – διεργασία. Διαφορετικά κλειδιά μπορεί να αποστέλλονται στην ίδια ρ – διεργασία π.χ. τα κλειδιά – αστέρια παραλαμβάνονται από την ίδια ρ – διεργασία με αυτήν των κλειδιών – ορθογώνιων. Με πιο επίσημη ορολογία, το Hadoop διαθέτει επομένως ένα διαμελιστή(partitioner) που ανακατανέμει τις εξόδους των μ – διεργασιών με βάση το κλειδί των δεδομένων εξόδου των μ – διεργασιών.

Κατόπιν της περιγραφείσας ανακατανομής, οι ρ – διεργασίες παραλαμβάνουν και ταξιθετούν τα δεδομένα, με βάση το κλειδί τους. Τα δεδομένα που έχουν το ίδιο κλειδί συγχωνεύονται σε ένα συγκεκριμένο, ζεύγος κλειδί – λίστα τιμών. Η λίστα τιμών συγκροτείται από τις τιμές όλων των δεδομένων, ζευγών που είχαν το ίδιο κλειδί. Τονίζεται πως οι περιγραφείσες διαδικασίες, ταξιθέτηση και συγχώνευση, δεν αποτελούν μέρος της επεξεργασίας που κάνουν οι ρ – διεργασίες, δεν αποτελούν μέρος του κώδικα που γράφει ο προγραμματιστής. Το Hadoop έχει σχεδιαστεί με αυτόν τον τρόπο κι ευθύνεται για την ταξιθέτηση και την συγχώνευση. Αυτό σε κάποιες περιπτώσεις είναι καλό, σε κάποιες άλλες όχι. Συνεχίζοντας, μετά αναλαμβάνουν δράση οι ρ – διεργασίες παράγοντας μία λίστα από ζεύγη κλειδιών – τιμών. Η λίστα ενδέχεται να είναι και κενή. Όταν οι ρ – διεργασίες γράφουν τα τελικά τους αποτελέσματα, τα γράφουν σε ένα μόνο αρχείο που έχει το όνομα part – χχχχχ, όπου το μέρος χχχχχ αφορά στην ταυτότητα του

διαμερίσματος που επεξεργάστηκε η ρ – διεργασία.

Όλα τα παραπάνω συνιστούν την τετριμμένη οδό κατά την εκτέλεση μιας εργασίας στο Hadoop. Σε αυτή την οδό κάποιες φορές υπάρχει ένα ακόμη προαιρετικό βήμα ανάμεσα στις μ και ρ – διεργασίες. Το βήμα αυτό σαν μία τοπική ρ – διεργασία λέγεται συνδυάσμων(combiner) και συνδυάζει τα διάφορα δεδομένα σε ένα, προκειμένου να ελαττωθούν αφενός το φορτίο στο δίκτυο και στην παραλήπτη ρ -διεργασία κι αφετέρου το συνεπαγόμενο κόστος του όποιου επιπρόσθετου φορτίου. Για παράδειγμα, αν ένα πρόγραμμα μετράει το πλήθος των λέξεων μέσα σε ένα αρχείο, τότε συμφέρει περισσότερο να αποσταλεί μια φορά η χ λέξη ως (χ, ψ) αν εμφανίζεται ψ φορές παρά να αποσταλεί η λέξη χ, ψ φορές ως (χ, 1). Όπως φαίνεται και στο παράδειγμα με την λέξη χ, ο συνδυάσμων διατηρεί την μορφή των δεδομένων. Είτε υπάρχει είτε όχι ο συνδυάσμων η ρ – διεργασία λαμβάνει την ίδια μορφή δεδομένων.

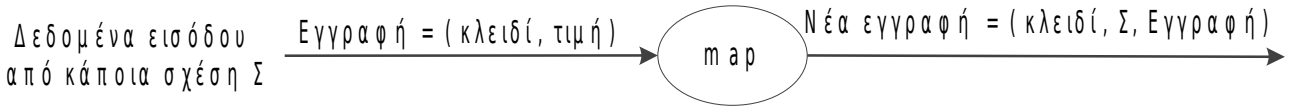
3.4 Αλγόριθμος συνδέσμου

Έχοντας ορίσει ή περιγράψει το προγραμματιστικό περιβάλλον στο οποίο θα εκτελεστεί ο όποιος σύνδεσμος, ήρθε η ώρα να εξεταστεί ο ίδιος ο σύνδεσμος ως αλγόριθμος. Στην παρούσα παράγραφο εξετάζεται η υλοποίηση του συνδέσμου από το ίδιο το Hadoop. Η υλοποίηση αυτή εξετάζεται πειραματικά στο τέταρτο(4^{0}) κεφάλαιο και συγκρίνεται με τον αλγόριθμο της παραγράφου 2.3 του δεύτερου($2^{ου}$) κεφαλαίου.

3.4.1 Αλγόριθμος συνδέσμου του Hadoop

Το Hadoop περιλαμβάνει ένα πακέτο για την εκτέλεση συνδέσμων με το όνομα datajoin(hadoop-datajoin-*.jar). Σύμφωνα με αυτό το πακέτο, ένας αλγόριθμος για την σύνδεση των δεδομένων λέγεται Reduce – side join κι εκτελεί έναν εσωτερικό σύνδεσμο στην reduce πλευρά, εξ ου και το όνομα του αλγορίθμου. Ο σύνδεσμος ονομάζεται εσωτερικός(inner join), γιατί παράγει ως έξοδο την τομή των συμμετεχουσών σχέσεων στον σύνδεσμο. Στην map πλευρά γίνεται η κατάλληλη προετοιμασία των δεδομένων εισόδου ώστε να συνδεθούν έπειτα.

Ακολουθώντας το σχήμα 31, οι συμμετέχουσες σχέσεις αποτελούν τα δεδομένα εισόδου τα οποία διαβάζονται από τις μ – διεργασίες ως αρχεία. Κάθε μ – διεργασία διαβάζει το αρχείο – σχέση, εγγραφή προς εγγραφή, δηλαδή, το διαβάζει γραμμή προς γραμμή. Για κάθε εγγραφή η μ – διεργασία παράγει μία νέα εγγραφή. Η νέα εγγραφή σε σχέση με την παλιά εγγραφή, έχει επιπλέον μια ετικέτα που είναι το όνομα της σχέσης από την οποία προέκυψε κι έχει κι ένα κλειδί, το κλειδί της παλιάς εγγραφής. Το όνομα της σχέσης απαιτείται ώστε να συνδέονται εγγραφές μόνο από διαφορετικές σχέσεις. Πιο παραστατικά η διαδικασία αυτή παρουσιάζεται στο σχήμα 33.



Σχήμα 33 : Δεδομένα εξόδου μ - διεργασιών

Όλες οι νέες εγγραφές στέλνονται στις ρ – διεργασίες. Δεδομένου όμως του σταδίου της ανακατανομής των εξόδων των μ – διεργασιών, κάθε ρ – διεργασία δέχεται ως είσοδο ομάδες από νέες εγγραφές. Κάθε ομάδα έχει ένα μοναδικό κλειδί. Με την σειρά της μια ρ – διεργασία για κάθε ομάδα υπολογίζει όλους τους δυνατούς συνδυασμούς που μπορούν να προκύψουν συνδυάζοντας τις εγγραφές της τρέχουσας ομάδας που επεξεργάζεται. Ουσιαστικά μια ρ – διεργασία εκτελεί μια πράξη καρτεσιανού γινομένου. Ένας έγκυρος συνδυασμός περιλαμβάνει μόνο εγγραφές που προέρχονται από διαφορετικές σχέσεις. Δεν υπάρχουν εγγραφές προερχόμενες από την ίδια σχέση. Άνευ αυτής της προϋπόθεσης, η ρ – διεργασία δεν παράγει κάποιο δεδομένο εξόδου. Η ρ – διεργασία παράγει τις πλειάδες εξόδου μέσω μιας μεθόδου με το όνομα combine(). Η μέθοδος αυτή δέχεται έναν συνδυασμό, εξετάζει αν περιλαμβάνει ακριβώς τόσες εγγραφές όσες κι οι συμμετέχουσες σχέσεις στον σύνδεσμο και μετά προσπαθεί να συνδυάσει αυτές τις εγγραφές.

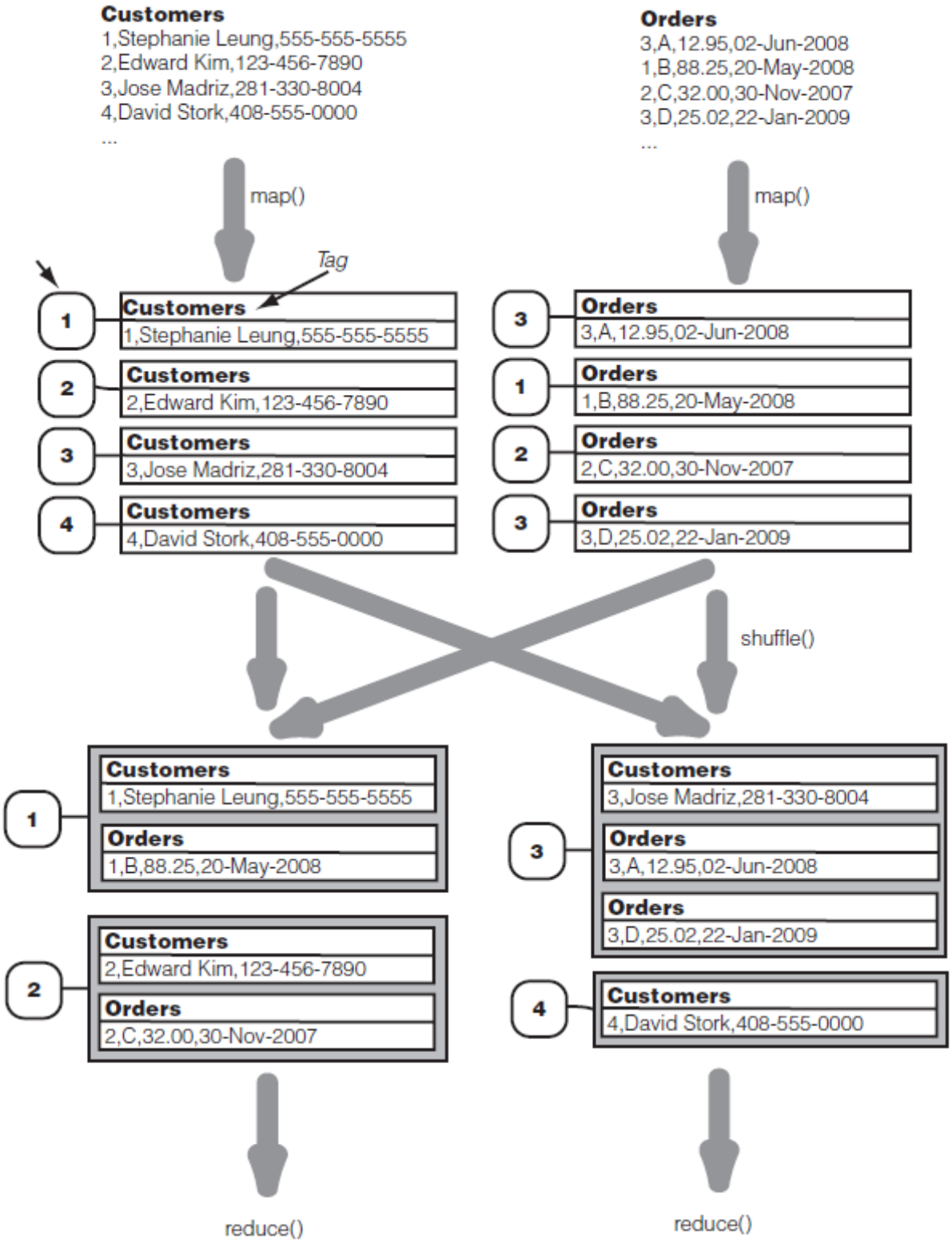
Για παράδειγμα αν ζητείται ο σύνδεσμος Customers $\bowtie$ Orders, κι οι σχέσεις Customers, Orders έχουν τα δεδομένα του πίνακα 1, τότε το σχήμα 34 αναπαριστά το διάγραμμα ροής για τις μ – διεργασίες και το σχήμα 35 απεικονίζει το διάγραμμα ροής για τις ρ – διεργασίες.

Πίνακας 1	
Customers	Orders
1, Stephanie Leung, 555-555-555	3, A, 12.95, 02-Jun-2008
2,Edward Kim, 123-456-7890	1, B, 88.25, 20-May-2008
3. Jose Madriz, 281-330-8004	2, C, 32.00, 30-Nov-2007
4. David Stork, 408-555-0000	3, D, 25.02, 22-Jan-2009

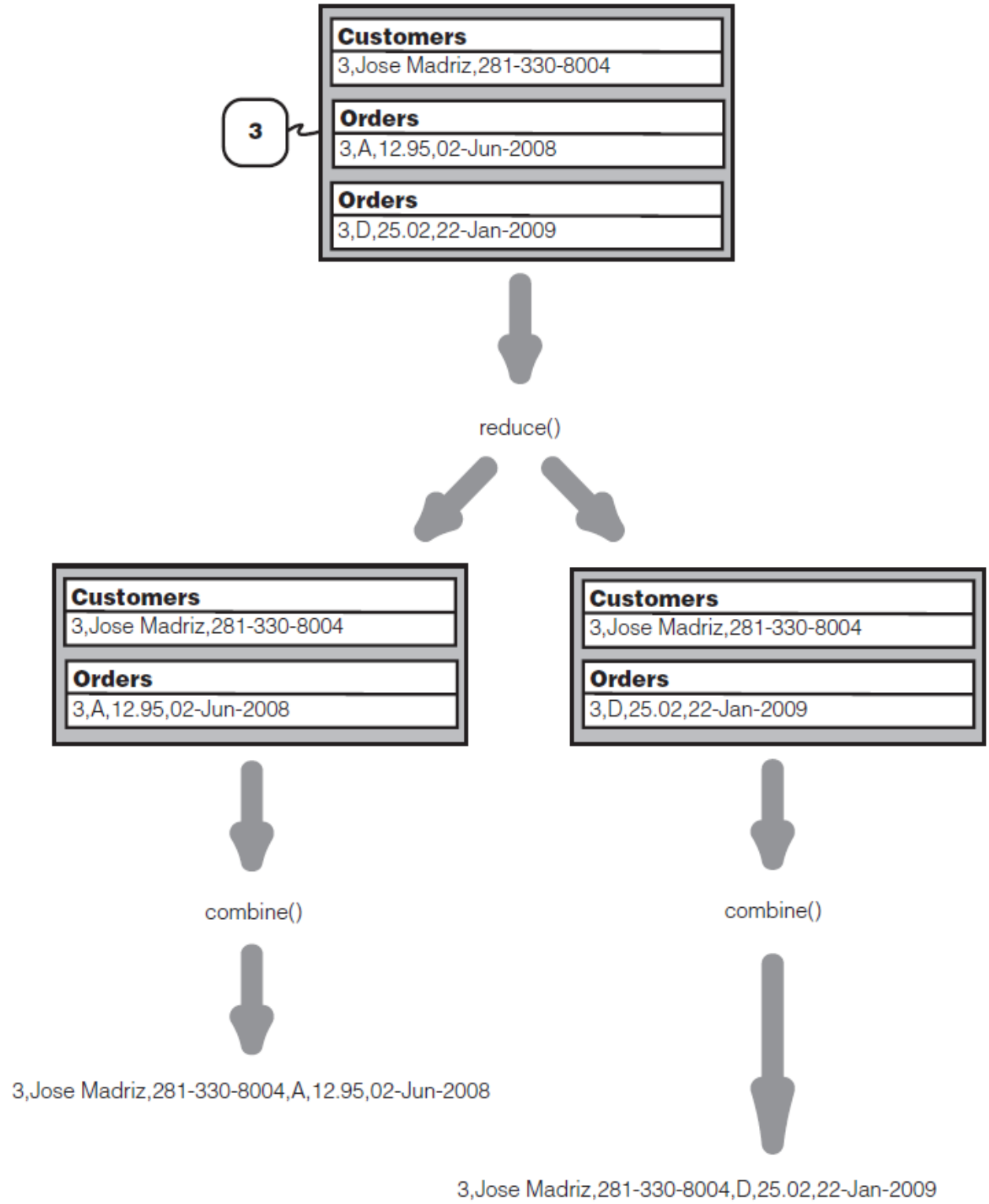
Στο σχήμα 34 φαίνεται καθαρά πως η κάθε νέα εγγραφή προκύπτει από την παλαιά. Ουσιαστικά, με βάση το σχήμα, κάθε νέα εγγραφή συνιστά ένα περιτύλιγμα για την παλαιά ώστε αυτή να μπορέσει να συνδεθεί με τις εγγραφές των άλλων σχέσεων εφόσον έχουν το ίδιο κλειδί. Επίσης, στο ίδιο σχήμα φαίνεται η ανακατανομή των εγγραφών εξόδου των μ – διεργασιών κι η αποστολή τους με βάση το κλειδί τους στις ρ – διεργασίες όχι ως μεμονωμένες εγγραφές αλλά ταξινομημένες σε ομάδες.

Στο σχήμα 35, για την ομάδα με το κλειδί 3 φαίνεται η παραγωγή όλων των δυνατών συνδυασμών τηρουμένης της προϋπόθεσης που ειπώθηκε παραπάνω.

Κλείνοντας σημειώνεται πως στο συνολικό διάγραμμα ροής, σχήματα 34 και 35, τηρείται το γενικό διάγραμμα ροής του σχήματος 31.



Σχήμα 34 : Reduce - side join, μ – διεργασίες [26]



Σχήμα 35 : Reduce - side join, ρ – διεργασίες [26]

Η σελίδα αυτή είναι σκόπιμα λευκή.

ΚΕΦΑΛΑΙΟ 4

ΑΛΓΟΡΙΘΜΟΙ

Η γνώση δεν έχει καμία αξία αν δεν την εφαρμόσεις κάπου.

Anton Chekhov

Με μια ματιά

Όπως ειπώθηκε στο πρώτο κεφάλαιο, ".... Η παρούσα διπλωματική εργασία εξετάζει τις διάφορες μέχρι τώρα μεθοδολογίες περί του συνδέσμου και καταλήγει σε μία την οποία εξετάζει εκτενώς θεωρητικά αλλά και την υποβάλλει σε διάφορα πειράματα προκειμένου να δοκιμαστεί η απόδοση της".

Σε αυτό το κεφάλαιο λοιπόν εξετάζεται θεωρητικά αυτή η μία τεχνική συνδέσμου και συγκρίνεται η απόδοση της με την απόδοση άλλων πιο "πατροπαράδοτων" αλγορίθμων. Αναλυτικότερα, ενδιαφέρει ο αποδοτικός σύνδεσμος πολλών σχέσεων σε ένα σύμπλεγμα υπολογιστών. Διερευνάται αν ο σύνδεσμος υπολογίζεται αποδοτικότερα με την σύνδεση όλων των σχέσεων μαζί ή με την χρήση μιας ακολουθίας διπλών συνδέσμων. Η πρώτη προσέγγιση συνιστά την νέα, προτεινόμενη τεχνική ενώ η δεύτερη τεχνική αποτελεί τον "πατροπαράδοτο" τρόπο. Το κύριο κριτήριο απόδοσης συνίσταται στο κόστος επικοινωνίας, δηλαδή, στον όγκο των δεδομένων που διακινούνται στο σύμπλεγμα των υπολογιστών εξαιρώντας τα δεδομένα εισόδου κι εξόδου.

Αρχικά παρατίθενται τεχνικές και θεωρητικές λεπτομέρειες των δύο αλγορίθμων, έπειτα μελετάται θεωρητικά το κόστος επικοινωνίας των δύο αλγορίθμων κι έπειτα παρουσιάζονται τα πειραματικά αποτελέσματα της σύγκρισης των δύο αλγορίθμων στο 5ο Κεφάλαιο.

4.1 Αλγόριθμος Hadoop – Ακολουθία διπλών συνδέσμων

Κατά την χρήση μιας ακολουθίας διπλών συνδέσμων, δύο οποιεσδήποτε, διαδοχικές σχέσεις συνδέονται και το αποτέλεσμα τους συνδέεται με μια τρίτη σχέση, διαδοχική όμως των δύο προηγούμενων. Το ίδιο επαναλαμβάνεται για το νέο αποτέλεσμα και την επόμενη διάδοχη σχέση. Για παράδειγμα, ο σύνδεσμος $P_1 \bowtie P_2 \bowtie P_3 \bowtie ... \bowtie P_v$ μπορεί να υπολογιστεί ως $(... ((P_1 \bowtie P_2) \bowtie P_3) \bowtie ...) \bowtie P_v$ όπου η διάταξη των παρενθέσεων ορίζει την σειρά των υπολογισμών.

Κάθε ένας από τους διπλούς συνδέσμους υπολογίζεται με χρήση του συνδέσμου του Hadoop που περιγράφηκε στο προηγούμενο κεφάλαιο, στην παράγραφο 3.4.

Στο πλαίσιο της υλοποίησης του εν λόγω αλγορίθμου υπάρχει ένα προ – επεξεργαστικό στάδιο κατά το οποίο βρίσκονται οι κοινές ιδιότητες ανάμεσα στις σχέσεις του συνδέσμου που χρησιμεύουν ως το κλειδί σύνδεσης του αντίστοιχου διπλού συνδέσμου δεδομένης μιας σειράς υπολογισμού. Στο προηγούμενο παράδειγμα, $(... ((P_1 \bowtie P_2) \bowtie P_3) \bowtie ...) \bowtie P_v$, βρίσκονται οι κοινές ιδιότητες μεταξύ των σχέσεων P_1, P_2 προς υπολογισμό του συνδέσμου $P_{12} = P_1 \bowtie P_2$. Έπειτα βρίσκονται οι κοινές ιδιότητες της νέας σχέσης P_{12} με την σχέση P_3 προκειμένου να υπολογιστεί ο σύνδεσμος $P_{12} \bowtie P_3 = (P_1 \bowtie P_2) \bowtie P_3)$ κ.ο.κ. Το εν λόγω προ – επεξεργαστικό βήμα έχει γραφτεί σε Python[15].

Τέλος, ακολουθεί το διάγραμμα κλάσεων, σχήμα 36, του υπό συζήτηση αλγορίθμου που έχει γραφτεί σε Java. Στο διάγραμμα αυτό δεν αναφέρονται με λεπτομέρεια όλες οι μέθοδοι και τα πεδία των κλάσεων και των διαπροσωπειών. Ο λόγος για αυτό είναι οτι δεν θα προσέθεταν στην κατανόηση του αλγορίθμου. Ίσα – ίσα που λόγω του όγκου τους θα μπορούσαν να συντελέσουν και προς την αντίθετη κατεύθυνση. Όμως φαίνονται όλα όσα χρειάζονται για την υλοποίηση του αλγορίθμου. Ένας δεύτερος λόγος αποτελεί ότι δεν είναι όλες οι κλάσεις κι οι διαπροσωπείες άμεσα σχετικές με τον αλγόριθμο αλλά παρουσιάζονται για λόγους πληρότητας. Περισσότερες πληροφορίες για αυτές τις κλάσεις και τις διαπροσωπείες υπάρχουν στο διαδίκτυο.

Κλείνοντας, επεξηγείται η λειτουργία των κύριων κλάσεων στο παρακάτω διάγραμμα όπως αυτή εξηγήθηκε νωρίτερα στην γενική ανάλυση του αλγορίθμου, σχήματα 30 και 31 στο τρίτο

15 Το Hadoop χάρη στο πακέτο streaming(hadoop-streaming-*.jar) που διαθέτει μπορεί να εκτελέσει προγράμματα γραμμένα και σε άλλες γλώσσες πέρα από την Java.

κεφάλαιο. Συγκεκριμένα,

1 MapperClass : κληρονομεί την DataJoinMapperBase, υλοποιεί την μ – διεργασία και με την μέθοδο,

1.1 generateInputTag : δημιουργεί την ετικέτα που προσδιορίζει μοναδικά την προέλευση της τρέχουσας εγγραφής.

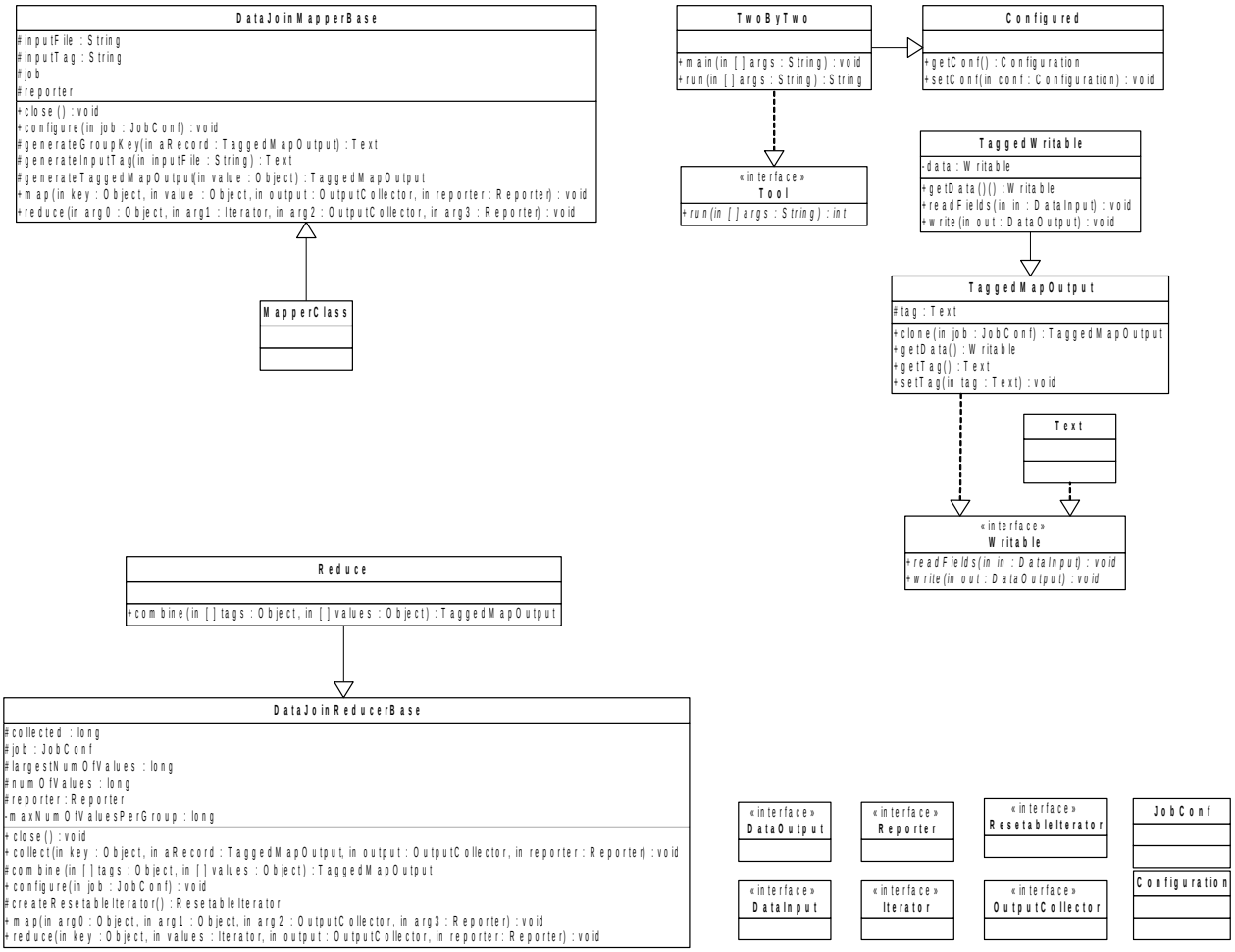
1.2 generateGroupKey : δημιουργεί το κλειδί της νέας εγγραφής.

1.3 generateTaggedMapOutput : παράγει την νέα εγγραφή καλώντας τις μεθόδους generateInputTag και generateGroupKey. Η νέα εγγραφή αποτελεί ένα αντικείμενο της κλάσης TaggedMapOutput.

2 Reduce : κληρονομεί την DataJoinReducerBase, ενσαρκώνει την ρ – διεργασία και με την μέθοδο,

2.1 combine : κάνει την σύνδεση μεταξύ των δεδομένων.

TaggedMapOutput : υλοποιεί την διαπροσωπεία Writable, αποτελεί μια κλάση – περιτύλιγμα αντικειμένων της κλάσης Text. Κάθε εγγραφή όταν διαβάζεται από το αρχείο - σχέση είναι ένα αντικείμενο String και μετατρέπεται σε αντικείμενο Text ώστε να μπορεί να δεχτεί την ετικέτα και το κλειδί της νέας εγγραφής.



Σχήμα 36 : Διάγραμμα κλάσεων για την ακολουθία διπλών συνδέσμων

4.2 Πολλαπλός σύνδεσμος με την χρήση της τεχνικής του άρθρου

Σύμφωνα με την τεχνική της ακολουθίας των διπλών συνδέσμων τα τελικά αποτελέσματα παράγονται σταδιακά από διπλό σύνδεσμο σε διπλό σύνδεσμο. Κατά την διαδρομή όμως αυτή παράγονται ενδιάμεσα αποτελέσματα τα οποία όμως τελικά μπορεί να μην δώσουν κάποιο τελικό αποτέλεσμα. Τέτοια στείρα αποτελέσματα προφανώς συνιστούν περιττούς υπολογισμούς και περιττό κόστος επικοινωνίας. Ένας τρόπος για να μην υπάρχουν τέτοια ενδιάμεσα αποτελέσματα, είναι να παραχθούν κατευθείαν τα τελικά αποτελέσματα.

Ο μελετώμενος πολλαπλός σύνδεσμος πετυχαίνει ακριβώς αυτό, την άμεση παραγωγή των τελικών αποτελεσμάτων, κατευθείαν από τις αρχικές σχέσεις. Όμως επειδή οι αρχικές σχέσεις είθισται να ανέρχονται σε μεγάλες ποσότητες εγγραφών, επιβάλλεται ο παραλληλισμός του πολλαπλού συνδέσμου. Αυτό όμως αποξενώνει τις εγγραφές των αρχικών σχέσεων και συνεπώς δεν γίνεται να γίνει ο συνδυασμός τους και να εξαχθούν τα τελικά αποτελέσματα. Για αυτό χρειάζεται, οι αποξενωμένες εγγραφές να συγκεντρωθούν στην κατάλληλη υπολογιστική μονάδα με κάποιο τρόπο.

Ο τρόπος αυτός συνίσταται στον προτεινόμενο αλγόριθμο της § 2.3 που καταφέρνει να συγκεντρώσει τις εγγραφές εκ νέου στην κατάλληλη υπολογιστική μονάδα. Στο πλαίσιο του Hadoop, μια υπολογιστική μονάδα αντιστοιχεί σε μια ρ – διεργασία κι οπότε η σύνδεση των αλγορίθμων γίνεται στην πλευρά της ρ – φάσης(reduce – side join algorithm). Η συγκέντρωση των κατάλληλων εγγραφών σε κάθε μία ρ – διεργασία εισάγει ένα προ – επεξεργαστικό βήμα προς υπολογισμό των βέλτιστων καλαθιών. Κατά τα άλλα όμως αφήνει ανοιχτό το θέμα του αλγορίθμου του συνδέσμου. Με άλλα λόγια η επιλογή του αλγορίθμου του συνδέσμου επαφίεται στον προγραμματιστή και στις ανάγκες αυτού.

Προχωρώντας, προς περαιτέρω βελτίωση του αλγορίθμου, εισάγεται ένα δεύτερο προ – επεξεργαστικό βήμα το οποίο επιταχύνει τις μ – διεργασίες. Το βήμα αυτό εξηγείται παρακάτω. Τέλος, υφίσταται ένα τρίτο προ – επεξεργαστικό βήμα που διαδραματίζει παρόμοιο ρόλο με αυτό της ακολουθίας των διπλών συνδέσμων προσαρμοσμένο βέβαια στην νέα τεχνική. Το βήμα αυτό αφενός βοηθά στην σύνδεση των σχέσεων αφετέρου βοηθά στον ορισμό των ρ – διεργασιών που κάθε μία εγγραφή θα αποσταλεί. Κι αυτό το βήμα εξηγείται παρακάτω.

Τέλος, σημειώνεται πως όλα τα προ – επεξεργαστικά βήματα αφενός έχουν γραφεί σε Python, αφετέρου εκτελούνται ανεξάρτητα από τον κύριο αλγόριθμο.

4.2.1 Εύρεση βέλτιστων καλαθιών

Σύμφωνα με το άρθρο(§ 2.3), ένας πολλαπλός σύνδεσμος μπορεί να επιτύχει καλύτερα αποτελέσματα σε σχέση με μια ακολουθία διπλών συνδέσμων αρκεί να γίνει ο διαμερισμός των δεδομένων εξόδου των μ – διεργασιών καταλλήλως. Ο διαμερισμός φυσικά γίνεται στην βάση κάποιων βέλτιστων καλαθιών για τις ρ – διεργασίες ή ισοδύναμα βέλτιστων μεριδίων των ιδιοτήτων που συμμετέχουν στο μ – κλειδί.

Ο αλγόριθμος για την εύρεση των βέλτιστων καλαθιών υλοποιήθηκε σε Python κι εκτελείται πριν από τον αλγόριθμο του πολλαπλού συνδέσμου ως μια αλυσίδα τριών φάσεων – εργασιών. Υπενθυμίζεται, πως ο αλγόριθμος αναλύεται σε πέντε βήματα,

1. βρίσκονται οι κυρίαρχες ιδιότητες. Αν δύο ή περισσότερες ιδιότητες συμμετέχουν σε όλες τις σχέσεις τότε κρατείται μία από όλες κι οι υπόλοιπες τίθενται να έχουν μερίδιο ίσο με 1.

2. Κατασκευάζεται η εξίσωση κόστους όπως έχει ήδη υποδειχθεί.

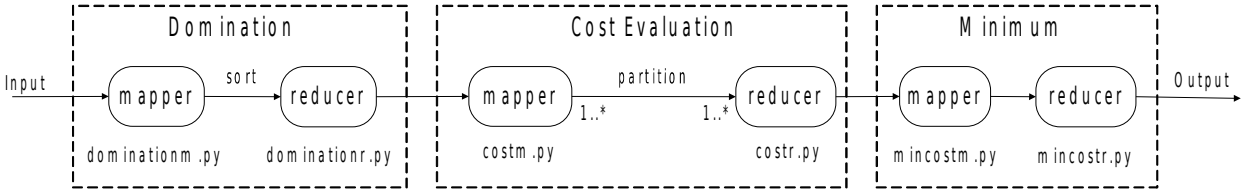
3. Κατασκευάζεται η εξίσωση Lagrange.

4. Εξαλείφονται οι εναπομείνασες περιπτώσεις όπου κάποια μερίδια ισούνται με το 0.

5. Επιλέγεται ως λύση εκείνη που ανάμεσα σε όλες τις παραχθείσες για τα διάφορα υπό – προβλήματα του βήματος 4, έχει το μικρότερο κόστος.

Το βήμα 1 αποτελεί μια φάση από μόνο του κι ονομάζεται domination. Τα βήματα 2, 3, 4 συγκροτούν την δεύτερη φάση που ονομάζεται cost evaluation καθώς σε αυτή εκτιμάται το κόστος όλων των δυνατών προβλημάτων που παράγονται. Το βήμα 5 συνιστά την τελευταία φάση, λέγεται minimumcost κι εκτελείται σε δύο στάδια. Στο πρώτο στάδιο, βρίσκεται το ελάχιστο κόστος της εξόδου κάθε ρ – διεργασίας από την προηγούμενη φάση και στέλνεται σε ένα αρχείο εξόδου. Στο δεύτερο στάδιο βρίσκονται το ολικό ελάχιστο κόστος και οι τιμές των καλαθιών που το κάνουν ελάχιστο. Αν οι τιμές των καλαθιών δεν είναι ακέραιες τότε λαμβάνεται ειδική μέριμνα. Ως τελικές βέλτιστες τιμές λαμβάνονται εκείνες οι ακέραιες τιμές ώστε το γινόμενο των καλαθιών να είναι όσο το δυνατόν πλησιέστερα στο επιθυμητό, στην αρχική προδιαγραφή.

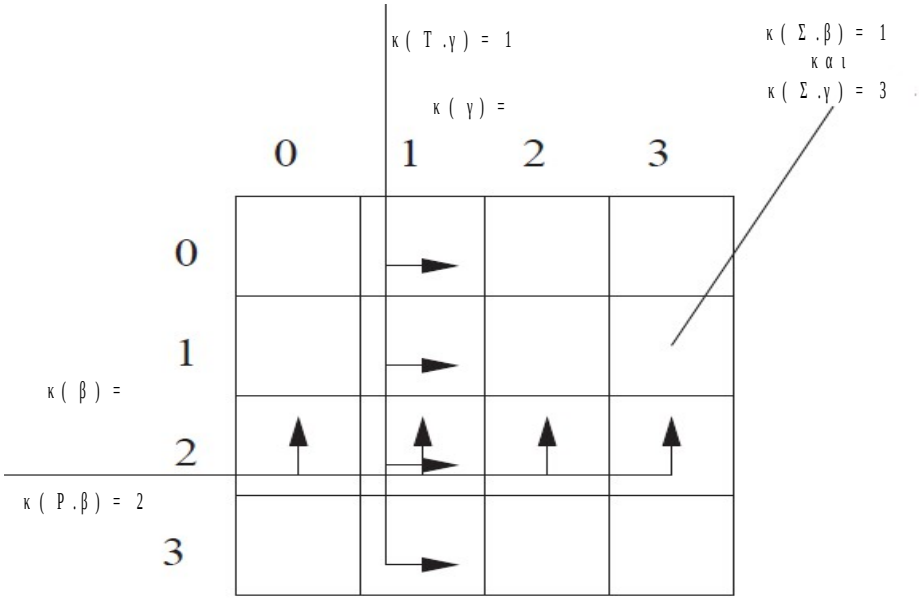
Οι παραπάνω φάσεις απεικονίζονται στο σχήμα 37 όπου τα ονόματα που τελειώνουν σε .py συνιστούν αρχεία γραμμένα σε Python και τα οποία αναλαμβάνουν την διεκπεραίωση της αντίστοιχης εργασίας.



Σχήμα 37 : Φάσεις υπολογισμού των βέλτιστων καλαθιών

4.2.2 Μαθηματική ανάλυση της νέας τεχνικής

Ο εξεταζόμενος αλγόριθμος πρέπει να υπολογίζει ορθά έναν πολλαπλό σύνδεσμο. Στην § 2.3, εξετάστηκε ο σύνδεσμος των σχέσεων Ρ(Α, Β), Σ(Β, Γ) και Τ(Γ, Δ) όπου οι ιδιότητες Β, Γ αποτελούν το μ – κλειδί. Για 16 ρ – διεργασίες προέκυπτε το σχήμα



Σχήμα 38 : Παράδειγμα για 16 ρ – διεργασίες

όπου κάθε συνάρτηση κ, αντιπροσώπευε μια συνάρτηση κατακερματισμού για την εκάστοτε εγγραφή της εκάστοτε σχέσης. Από το σχήμα απορρέει πως ο σωστός υπολογισμός ενός πολλαπλού συνδέσμου, ισοδυναμεί με την αποστολή των σωστών εγγραφών στις σωστές ρ – διεργασίες όπου γίνεται η σύνδεση των κατάλληλων εγγραφών. Άρα, υπάρχει μία μ – φάση(map phase) όπου υπολογίζονται οι ρ – διεργασίες στις οποίες θα σταλεί κάθε μία εγγραφή. Υπάρχει επίσης και μια ρ – φάση(reduce phase) όπου γίνεται η σύνδεση των εγγραφών και παράγεται η έξοδος.

Ξεκινώντας με την μ – φάση, το παραπάνω σχήμα απεικονίζει έναν πίνακα όπου κάθε θέση αφορά σε μία και μόνο ρ – διεργασία. Η θέση κάθε ρ – διεργασίας αποτελεί επίσης την διεύθυνση της μέσα στον πίνακα. Κάθε γραμμή ή στήλη αφορά σε μία και μόνο ιδιότητα, συμμετέχουσα στο μ – κλειδί του συνδέσμου. Ο αύξων αριθμός μιας γραμμής ή στήλης αντιστοιχεί στον αύξοντα αριθμό κάποιου μεριδίου της αντίστοιχης ιδιότητας. Η ιδιότητα Β π.χ. έχει 4 μερίδια με αύξοντες αριθμούς από 0 – 3. Κάθε ιδιότητα του μ – κλειδιού του συνδέσμου συμβάλλει στον σχηματισμό της διεύθυνσης μιας ρ – διεργασίας. Οπότε, οι διευθύνσεις των ρ – διεργασιών σχετίζονται με τις ιδιότητες του μ – κλειδιού κάτι που εξυπηρετεί στην αποστολή των σωστών εγγραφών στις

πρέπουσες ρ – διεργασίες.

Με μια πιο μαθηματική θεώρηση του προβλήματος, οι ν συνολικές, διαφορετικές ιδιότητες των σχέσεων είναι μοναδιαία διανύσματα τα οποία ορίζουν ένα χώρο ν διαστάσεων. Το πλήθος των καλαθιών μιας ιδιότητας είναι το μέτρο, η έκταση του ν – διάστατου χώρου στην διεύθυνση που ορίζει αυτή η ιδιότητα – διάνυσμα. Οι ρ – διεργασίες σε αυτό το χώρο έχουν τον ρόλο των σημείων κι οι εγγραφές αποτελούν τιμές για κάποιες “τοπικές ιδιότητες” των σημείων αυτών. Ο πίνακας του σχήματος 38 συνιστά έναν τέτοιο χώρο 2 διαστάσεων.

Συνεχίζοντας, το μ – κλειδί καθίσταται το μέγιστο σύνολο ιδιοτήτων – διανυσμάτων που χρειάζονται για να οριστεί ο συγκεκριμένος χώρος, δηλαδή, καθίσταται η βάση του χώρου αυτού. Μια ιδιότητα με μοναδιαίο πλήθος καλαθιών δεν συμβάλλει ουσιαστικά στην δημιουργία του υπό συζήτηση χώρου, αλλά ανήκει στον μηδενικό χώρο(null space). Το γεγονός οτι ανήκει στον μηδενικό χώρο συνεπάγεται πως η εν λόγω ιδιότητα δεν συμβάλλει στο κόστος του αλγορίθμου, και διατηρεί την συνοχή του γράφου του συνδέσμου.

Χαρακτηριστικό παράδειγμα της αξίας των ιδιοτήτων του μηδενικού χώρου αποτελεί ο σύνδεσμος αλυσίδα. Σε αυτόν, αν όλες οι σχέσεις έχουν το ίδιο μέγεθος, το πλήθος των καλαθιών των ιδιοτήτων με άρτιο δείκτη($\alpha_{2\kappa}$) ισούται με την 1. Αν οι ιδιότητες αυτές απαλειφθούν από το μ – κλειδί τότε απαλείφονται κρίκοι από τον σύνδεσμο αλυσίδα και δυσχεραίνεται σημαντικά η απόφαση για την σύνδεση ή όχι των εγγραφών σε μια ρ – διεργασία. Για την ακρίβεια απαιτείται η πληροφορία που χάθηκε με την απαλοιφή των προαναφερθέντων κρίκων. Οπότε, οι ιδιότητες κρίκοι κρίνονται απαραίτητοι για την διαδικασία του συνδέσμου.

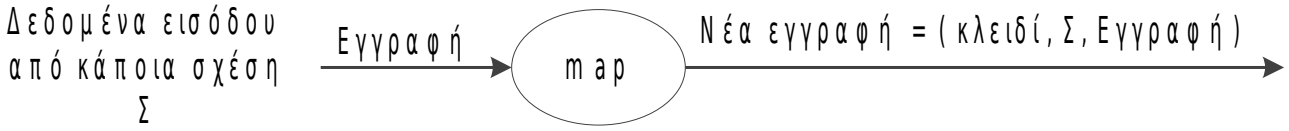
Τέλος, μέσα από την μαθηματική θεώρηση, το πρόβλημα ανάγεται στην κατασκευή αυτού του χώρου, με την έννοια οτι υπολογίζονται οι διαστάσεις του με τον αλγόριθμο της § 2.3 καθώς και τα σημεία του – ρ – διεργασίες κατά την φάση του κατακερματισμού των εγγραφών στην μ – φάση.

Συνεχίζοντας με την ρ – φάση, ο συσχετισμός των διευθύνσεων των ρ – διεργασιών με τις ιδιότητες του μ – κλειδιού επιτυγχάνεται μέσω συναρτήσεων κατακερματισμού, όπως φαίνεται και στο άνωθεν σχήμα. Μια συνάρτηση κατακερματισμού για διαφορετικές τιμές εισόδου μπορεί να παράγει την ίδια τιμή εξόδου. Συνεπώς, όταν κάποιες εγγραφές στέλνονται στην ίδια ρ – διεργασία δεν σημαίνει πως αυτές θα πρέπει να συνδεθούν απαραίτητα. Διευκρινίζεται πως το κλειδί με βάση το οποίο μια εγγραφή στέλνεται σε μια ρ – διεργασία ταυτίζεται με τον αύξοντα αριθμό της ρ – διεργασίας. Προφανώς το εν λόγω κλειδί(κλειδί αποστολής) διαφέρει από το κλειδί με βάση το οποίο η αντίστοιχη εγγραφή συνδέεται με άλλες εγγραφές για την παραγωγή κάποιας πλειάδας εξόδου(κλειδί εγγραφής). Επομένως, η αποστολή δύο εγγραφών στην ίδια ρ – διεργασία αποτελεί ικανή συνθήκη για να συνδεθούν κι όχι αναγκαία. Για αυτό χρειάζεται μια παραπάνω φροντίδα στην πραγμάτωση της ρ – φάσης.

Μέχρι εδώ δόθηκε μια γενική εικόνα και των δύο φάσεων κι από εδώ και πέρα δίνεται μια πιο λεπτομερής και μαθηματική εικόνα.

Συνεχίζοντας, μια μ – διεργασία διαβάζει μια εγγραφή από την είσοδο και την μετασχηματίζει σε μια νέα εγγραφή τύπου κλειδί – τιμή που έχει το όνομα της σχέσης προέλευσης ως ετικέτα για να ξεχωρίζει από τις υπόλοιπες εγγραφές. Συνδέονται μόνο εγγραφές με διαφορετική σχέση προέλευσης. Ο μετασχηματισμός αυτός απεικονίζεται στο σχήμα 39. Η νέα εγγραφή σε σχέση με την αρχική έχει κάποιο κλειδί και μια τιμή που περιλαμβάνει τόσο την αρχική εγγραφή όσο και κάποια δεδομένα[16] που χρειάζονται στην ρ – φάση. Το κλειδί της νέας εγγραφής ταυτίζεται με τον αύξοντα αριθμό της ρ – διεργασίας στην οποία θα αποσταλεί. Με αυτόν τον τρόπο, όλες οι νέου τύπου εγγραφές με τον ίδιο αύξοντα αριθμό πηγαίνουν στην ίδια ρ – διεργασία.

16 Τα δεδομένα αυτά συζητούνται στο πλαίσιο της ρ – φάσης.



Σχήμα 39 : Δεδομένα εξόδου μ - διεργασιών

Πώς υπολογίζεται ο αύξων αριθμός μιας ρ – διεργασίας; Όπως ειπώθηκε νωρίτερα, οι ν ιδιότητες – διανύσματα του μ – κλειδιού ορίζουν ένα χώρο ν διαστάσεων, που με όρους γραμμικής άλγεβρας, ο χώρος αυτός αποτελεί έναν πίνακα μεγέθους ν που λέγεται ρ – πίνακας. Κάθε θέση του πίνακα αυτού συνιστά και μια ρ – διεργασία, εξ ου και το όνομα ρ – πίνακας. Δεδομένου οτι ένας ν – διάστατος πίνακας μπορεί να μετασχηματιστεί σε ένα μονοδιάστατο πίνακα, η κ – οστή ρ – διεργασία αντιστοιχίζεται άμεσα σε κάποιο κελί του ρ – πίνακα κι αντίστροφα.

Αν ένας πίνακας Π έχει ν διαστάσεις $ι_1, ι_2, ..., ι_κ, ..., ι_{ν-1}, ι_ν$ με μέγεθος $Δ_1, Δ_2, ..., Δ_κ, ..., Δ_{ν-1}, Δ_ν$ η κάθε μία, τότε η θέση $[ι_1][ι_2] ...[ι_κ] ...[ι_{ν-1}][ι_ν]$ στον Π αντιστοιχεί σε μια θέση ενός πίνακα γραμμή Γ που δίνεται από την σχέση,

$$Γ(ι_1, ι_2, \dots, ι_κ, \dots, ι_{ν-1}, ι_ν) = ι_1 \prod_{λ=2}^{ν} Δ_λ + ι_2 \prod_{λ=3}^{ν} Δ_λ + \dots + ι_κ \prod_{λ=κ+1}^{ν} Δ_λ + \dots + ι_{ν-1} \prod_{λ=ν}^{ν} Δ_λ + ι_ν$$

που ισοδύναμα γράφεται,

$$Γ(ι_1, ι_2, \dots, ι_κ, \dots, ι_{ν-1}, ι_ν) = ι_1 \prod_{λ=2}^{ν} Δ_λ + ι_2 \prod_{λ=3}^{ν} Δ_λ + \dots + ι_κ \prod_{λ=κ+1}^{ν} Δ_λ + \dots + ι_{ν-1} Δ_ν + ι_ν \quad (1)$$

όπου $\prod_{λ=ν}^{ν} Δ_λ = Δ_ν$ κι ο συντελεστής της ιδιότητας $ι_ν$ είναι ο $\prod_{λ=ν+1}^{ν} Δ_λ = 1$.

Σε κάθε εγγραφή μιας σχέσης εφαρμόζεται μια συνάρτηση κατακερματισμού. Η συνάρτηση αυτή αποτελεί τον γραμμικό συνδυασμό των συναρτήσεων κατακερματισμού που εφαρμόζονται στα επιμέρους μέρη της εγγραφής. Κάθε ένα από τα μέρη αυτά αφορά σε μια και μόνο ιδιότητα της εγγραφής. Αν για παράδειγμα ληφθεί η σχέση Σ(Α, Β, Γ), η συνάρτηση κατακερματισμού της είναι

$$κ_Σ = κ(Α) + κ(Β) + κ(Γ),$$

όπου κ(Ι) η συνάρτηση κατακερματισμού της ιδιότητας Ι, Ι = Α, Β, Γ. Δεδομένου οτι οι τιμές των ιδιοτήτων Α, Β, Γ είναι αλφαριθμητικά(string) εφαρμόζεται ως συνάρτηση κατακερματισμού κάθε μίας ιδιότητας μια συνάρτηση κατακερματισμού για αλφαριθμητικά. Προφανώς, μια συνάρτηση κ(Ι) έχει ένα πεδίο τιμών που ξεπερνά το μέγεθος της οποιασδήποτε διάστασης του πίνακα Π. Για αυτό η επιστρεφόμενη τιμή από την κ(Ι) με μια λειτουργία υπολοίπου, κ(Ι) modulo μ(Ι), περιορίζεται στο εύρος 0, ..., μ(Ι) – 1, όπου μ(Ι) το μέγεθος της ιδιότητας – διάστασης Ι. Στο εξεταζόμενο πρόβλημα ως μέγεθος της ιδιότητας Ι ορίζεται το πλήθος των καλαθιών της εφόσον συμμετέχει στο μ – κλειδί. Αν δεν συμμετέχει στο μ – κλειδί του συνδέσμου τότε έχει μοναδιαίο

μέγεθος. Τελικά,

$$\kappa_{\Sigma} = M_{A} + M_{B} + M_{\Gamma} \quad (2)$$

για την σχέση Σ κι όπου M_{I} = κ(I) modulo μ(I), I = A, B, Γ.

Ως συμπέρασμα των σχέσεων (1), (2) εξάγεται πως όταν για μια ιδιότητα I μιας εγγραφής Ε υπολογίζεται μια τιμή κ(I), τότε, η τιμή της αντίστοιχης διάστασης στην σχέση (1) είναι γνωστή και δεσμευμένη. Όταν όμως δεν ισχύει κάτι τέτοιο, η τιμή της αντίστοιχης διάστασης στην σχέση (1) είναι ελεύθερη λαμβάνοντας τιμές στο διάστημα [0, μ(I) – 1]. Επιστρέφοντας στην σχέση Σ, όταν για όλες τις ιδιότητες A, B, Γ μιας εγγραφής της, υπολογίζεται μια τιμή $M_{I}, I = A, B, \Gamma$, τότε η κ_{Σ} λαμβάνει μια συγκεκριμένη τιμή αυτής της σχέσης (2). Αν όμως για παράδειγμα η ιδιότητα Α εξαιρεθεί από την σχέση (2) τότε, η κ_{Σ} έχει ως πεδίο τιμών το $\Pi_{T} = [M_{B} + M_{\Gamma}, M_{B} + M_{\Gamma} + \mu(A) - 1]$ αφού $M_{A} = 0, \ldots, \mu(A) - 1$.

Επομένως, όταν μια εγγραφή Ε έχει ως κλειδί ένα γνήσιο υποσύνολο του μ – κλειδιού του συνδέσμου, τότε, η εν λόγω εγγραφή δεν αποστέλλεται μόνο σε μία ρ – διεργασία αλλά σε ένα

σύνολο αυτών, λόγω των ελεύθερων διαστάσεων. Κάθε μία ρ – διεργασία από το σύνολο των ρ – διεργασιών αντιστοιχεί σε ένα συνδυασμό των ελεύθερων διαστάσεων στην σχέση (1). Για κάθε ελεύθερη διάσταση κ = $ι_1$, ..., $ι_ν$, υπολογίζεται το σύνολο

$$A_\kappa = \{ \lambda_\kappa^\xi = \xi \prod_{\kappa+1}^{\nu} \Delta_\lambda ,\ 0 \leqslant \xi \leqslant \iota_\kappa - 1 \} \quad (3)$$

όπου, $\prod_{\lambda=\nu}^{\nu} \Delta_\lambda = \Delta_\nu$ και $\prod_{\lambda=\nu+1}^{\nu} \Delta_\lambda = 1$ που περιέχει όλες τις θέσεις του πίνακα Π όταν μεταβάλλεται μόνο η κ διάσταση κι όλες οι άλλες τίθενται ίσες με το 0. Προκειμένου να ληφθούν όλες οι δυνατές θέσεις του Π, υπολογίζεται το καρτεσιανό γινόμενο των συνόλων $A_κ$. Στο καρτεσιανό γινόμενο αυτό, όταν συνδυάζονται οι τιμές δύο συνόλων, αυτές προστίθενται δίνοντας την καινούργια θέση. Τέλος, στις τιμές των καρτεσιανών γινομένων προστίθεται η σταθερή συνεισφορά των δεσμευμένων διαστάσεων και προκύπτουν οι διευθύνσεις των επιθυμητών ρ – διεργασιών.

4.2.3 Επιτάχυνση των μ - διεργασιών

Συμπερασματικά, η διεύθυνση μιας ρ – διεργασίες στην οποία αποστέλλεται μια εγγραφή Ε, είναι ο γραμμικός συνδυασμός των ελεύθερων και των δεσμευμένων διαστάσεων – ιδιοτήτων. Οι δεσμευμένες ιδιότητες συμμετέχουν εξίσου στην Ε και στο μ – κλειδί, ενώ οι ελεύθερες ιδιότητες συμμετέχουν μόνο στο μ – κλειδί. Να σημειωθεί εδώ, πως ο χαρακτηρισμός μιας ιδιότητας ως ελεύθερης ή δεσμευμένης συνδέεται άρρηκτα με την σχέση προέλευσης της υπό εξέταση εγγραφής. Με αλλά λόγια κάποια ιδιότητα μπορεί να χαρακτηρίζεται ελεύθερη για μια εγγραφή E_1 και να χαρακτηρίζεται δεσμευμένη για μια εγγραφή E_2 ανάλογα με την σχέση προέλευσης.

Για κάθε εγγραφή ξεχωριστά μπορούν να υπολογιστούν όλες οι ρ – διεργασίες στις οποίες πρέπει να σταλεί. Δεδομένου όμως πως το μ – κλειδί συγκροτείται από πεπερασμένο αριθμό ιδιοτήτων, υπάρχει πεπερασμένος αριθμός συνδυασμών των ιδιοτήτων αυτών ως ελεύθερες και δεσμευμένες ιδιότητες. Εν παραδείγματι, έστω πως το μ – κλειδί είναι το A_1, A_2, πως τα μεγέθη των καλαθιών είναι $α_1 = 3$, $α_2 = 4$ κι οτι η ιδιότητα – διάσταση A_1 συνιστά την δεύτερη διάσταση του χώρου που ορίζουν οι διαστάσεις A_1, A_2. Κατ' επέκταση μπορεί να κατασκευαστεί ο ακόλουθος πίνακας όπου το 0 υποδηλώνει την απουσία της αντίστοιχης ιδιότητας από την εγγραφή ενώ ο 1 υποδηλώνει το αντίθετο. Με άλλα λόγια, το 0 σημαίνει πως η αντίστοιχη ιδιότητα είναι ελεύθερη για την εν λόγω εγγραφή ενώ ο 1 σημαίνει πως η αντίστοιχη ιδιότητα είναι δεσμευμένη για την ίδια εγγραφή. Επίσης, τα σύμβολα $A_κ$, κ = 1, 2 είναι τα σύνολα της σχέσης (3)

$$A_\kappa = \{ \lambda_\kappa^\xi = \xi \prod_{\kappa+1}^{\nu} \Delta_\lambda ,\ 0 \leqslant \xi \leqslant \iota_\kappa - 1 \}$$

όπως αυτή ορίστηκε νωρίτερα.

Αύξων αριθμός	A_1	A_2	Συνεισφορά ελεύθερων ιδιοτήτων με μηδενική συνεισφορά των δεσμευμένων
0	0	0	Αδύνατο να απουσιάζουν κι οι δύο ιδιότητες από μια εγγραφή.
1	0	1	$A_1 = \{ \lambda_1^\xi = \xi \prod_{2}^{2} \Delta_\lambda = \xi\Delta_2 = 4\xi ,\ 0 \leqslant \xi \leqslant 2 \} = \{ 0, 4, 8 \}$
2	1	0	$A_2 = \{ \lambda_2^\xi = \xi \prod_{3}^{2} \Delta_\lambda = \xi ,\ 0 \leqslant \xi \leqslant 3 \} = \{ 0, 1, 2, 3 \}$
3	1	1	Καμία συνεισφορά αφού δεν υπάρχουν ελεύθερες ιδιότητες

Πίνακας 1: Εξαγωγή ρ - διεργασιών από ένα μ – κλειδί δύο ιδιοτήτων

Από τον παραπάνω πίνακα, παρατηρείται πως για 2 ιδιότητες υπάρχουν 3 συνδυασμοί αυτών ως ελεύθερες και δεσμευμένες. Μια εγγραφή μπορεί να περιέχει μόνο την ιδιότητα A_2 του μ – κλειδιού, όπως στην περίπτωση με αύξοντα αριθμό 1 ή να περιέχει μόνο την ιδιότητα A_1 όπως στην περίπτωση με αύξοντα αριθμό 2 ή τέλος δύναται να περιέχει όλες τις ιδιότητες του μ – κλειδιού όπως στην περίπτωση με αύξοντα αριθμό 3. Συμπερασματικά, για πεπερασμένο αριθμό ιδιοτήτων στο μ – κλειδί υπάρχουν πεπερασμένοι συνδυασμοί αυτών των ιδιοτήτων ως ελεύθερες κι ως δεσμευμένες. Για ν ιδιότητες ανέρχονται σε $2^{\nu} - 1$ αφού ο συνδυασμός να απουσιάζουν όλες οι ιδιότητες από μια εγγραφή δεν έχει πρακτικό νόημα.

Επιπλέον, από τον ίδιο πίνακα φαίνεται πως για κάθε συνδυασμό των ιδιοτήτων, η συνεισφορά των ελεύθερων μεταβλητών είναι συγκεκριμένη και δίνεται από το καρτεσιανό γινόμενο των συνόλων $A_{\kappa} = \{ \lambda_{\kappa}^{\xi} = \xi \prod_{\kappa+1}^{\nu} \Delta_{\lambda}, 0 \leqslant \xi \leqslant \iota_{\kappa} - 1 \}$ των ιδιοτήτων που δεν συμμετέχουν στην τρέχουσα εγγραφή. Για παράδειγμα, αν το μ – κλειδί συγκροτείται από τις ιδιότητες A_1, A_2, A_3 με $\alpha_1 = 3$, $\alpha_2 = 4$ κι $\alpha_3 = 2$, κι επιπλέον η ιδιότητα A_1 συνιστά την τρίτη διάσταση του οριζόμενου χώρου, η A_2 αποτελεί την δεύτερη διάσταση του χώρου κι η A_3 χαρακτηρίζεται ως η πρώτη διάσταση του χώρου, τότε προκύπτει ο Πίνακας 2. Στον πίνακα αυτόν, φαίνονται κάποια καρτεσιανά γινόμενα τα οποία έχουν ως αποτέλεσμα ένα νέο σύνολο. Στα σύνολα αυτά, έχουν προστεθεί οι συνιστώσες κάθε στοιχείου τους δίνοντας ένα μοναδικό αριθμό που αντιστοιχεί σε κάποια ρ – διεργασία. Το γεγονός αυτό σημειώνεται με το σύμβολο (+) στις κατάλληλες γραμμές.

Αυτό που πρέπει να γίνει κατανοητό από τον Πίνακα 2, αφορά στο γεγονός πως η συνεισφορά των ελεύθερων διαστάσεων είναι συγκεκριμένη και δύναται να προϋπολογιστεί για κάθε συνδυασμό ελεύθερων και δεσμευμένων διαστάσεων, δηλαδή, για κάθε εγγραφή. Δεν χρειάζεται για κάθε εγγραφή ξεχωριστά να υπολογίζεται εκ νέου η συνεισφορά των ελεύθερων διαστάσεων. Αυτό προφανώς επιταχύνει τις μ – διεργασίες καθώς εκμηδενίζονται οι περιττοί υπολογισμοί.

Αύξων αριθμός	A_1	A_2	A_3	Συνεισφορά ελεύθερων ιδιοτήτων με μηδενική συνεισφορά των δεσμευμένων
0	0	0	0	Αδύνατο να απουσιάζουν κι οι τρεις ιδιότητες από μια εγγραφή.
1	0	0	1	(+) $A_1 \times A_2 = \{ 0, 8, 16 \} \times \{ 0, 2, 4, 6 \} = \{ 0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22 \}$
2	0	1	0	(+) $A_1 \times A_3 = \{ 0, 8, 16 \} \times \{ 0, 1 \} = \{ 0, 1, 8, 9, 16, 17 \}$
3	0	1	1	$A_1 = \{ 0, 8, 16 \}$
4	1	0	0	(+) $A_2 \times A_3 = \{ 0, 2, 4, 6 \} \times \{ 0, 1 \} = \{ 0, 1, 2, 3, 4, 5, 6, 7 \}$
5	1	0	1	$A_2 = \{ 0, 2, 4, 6 \}$
6	1	1	0	$A_3 = \{ 0, 1 \}$
7	1	1	1	Καμία συνεισφορά αφού δεν υπάρχουν ελεύθερες ιδιότητες

Πίνακας 2: Εξαγωγή ρ - διεργασιών από ένα μ - κλειδί τριών ιδιοτήτων

Βέβαια κάποιος μπορεί να ρωτήσει, και πώς γίνεται για κάθε εγγραφή να λαμβάνεται η

σωστή συνεισφορά από όλες τις δυνατές; Η απάντηση στο ερώτημα αυτό δίνεται κι από τους δύο πίνακες 1 και 2. Ειδικότερα, θεωρώντας την διάταξη των ιδιοτήτων του μ – κλειδιού αυστηρή, δηλαδή, η πρώτη ιδιότητα του μ – κλειδιού αντιστοιχεί στην τελευταία διάσταση του οριζόμενου χώρου, η δεύτερη ιδιότητα αναπαριστά την προτελευταία διάσταση του οριζόμενου χώρου κ.ο.κ., τότε κάθε συνδυασμός αντιστοιχεί σε μια συγκεκριμένη συνεισφορά. Ο συνδυασμός { 0, 0, 1} αντιστοιχεί πάντα στην συνεισφορά με αύξοντα αριθμό 1 του πίνακα 2 λαμβάνοντας υπό όψιν και την διάταξη των ιδιοτήτων. Αν όμως, η διάταξη αγνοηθεί, τότε ο συνδυασμός { 0, 0, 1} αντιστοιχεί εξίσου στις συνεισφορές με αύξοντες αριθμούς 1 και 2.

Εν κατακλείδι, κάθε εγγραφή δέχεται μια συγκεκριμένη συνεισφορά από τις ελεύθερες ιδιότητες η οποία μπορεί να υπολογιστεί εκ των προτέρων, γεγονός που επιταχύνει τις μ – διεργασίες. Αυτός ο εκ των προτέρων υπολογισμός αφορά στο δεύτερο προ – επεξεργαστικό βήμα. Τέλος, η συνεισφορά των δεσμευμένων ιδιοτήτων αφορά σε κάθε εγγραφή ξεχωριστά και συνάγεται από την συνάρτηση κατακερματισμού.

4.2.4 Αλγόριθμος σύνδεσης

Στο παρόν εδάφιο εξετάζονται οι διάφοροι αλγόριθμοι οι οποίοι εφαρμόσθηκαν ως αλγόριθμοι συνδέσμου στο πλαίσιο της νέας τεχνικής για τον διαμοιρασμό των εγγραφών στις ρ – διεργασίες. Υπενθυμίζεται πως η νέα τεχνική υπαγορεύει μόνο πως ο αλγόριθμος σύνδεσης θα εκτελείται στην ρ – φάση του συνολικού αλγορίθμου(reduce side algorithm) αφήνοντας κατά τα άλλα ανοιχτό το θέμα της υλοποίησης του.

Ο αλγόριθμος σύνδεσης λοιπόν υλοποιήθηκε με τρεις βασικούς τρόπους. Κάθε ένας από αυτούς τους τρόπους έχει το δικό του πλήθος παραλλαγών. Αυτές οι παραλλαγές δεν εξετάζονται εδώ πέρα καθώς δεν προσθέτουν στην κατανόηση.

Και οι τρεις τρόποι υλοποίησης του αλγορίθμου σύνδεσης έχουν δύο κοινά χαρακτηριστικά. Το πρώτο χαρακτηριστικό τους είναι η χρήση του καρτεσιανού γινομένου κάποια στιγμή ώστε να συνδεθούν οι εγγραφές δίνοντας την εγγραφή εξόδου. Η χρήση του καρτεσιανού γινομένου γίνεται με τον ίδιο τρόπο και στους τρεις τρόπους υλοποίησης του αλγορίθμου σύνδεσης.

Το δεύτερο χαρακτηριστικό τους είναι η ανάγκη για λήψη ειδικής μέριμνας ώστε οι εγγραφές εισόδου των ρ – διεργασιών να ταξιθετηθούν με βάση το κλειδί με το οποίο συνδέονται με άλλες εγγραφές από άλλες σχέσεις. Το εν λόγω χαρακτηριστικό προέρχεται από το γεγονός πως οι εγγραφές αυτές διαμοιράζονται στις ρ – διεργασίες με βάση τον αύξοντα αριθμό της ρ – διεργασίας στην οποία θα σταλούν. Στην περίπτωση της ακολουθίας των διπλών συνδέσμων η αποστολή των εγγραφών στις ρ – διεργασίες γίνεται σύμφωνα με το κλειδί σύνδεσης κι οπότε οι εγγραφές ταξιθετούνται αυτόματα και καθίστανται έτοιμες για σύνδεση.

Συνεχίζοντας, ο τρόπος με τον οποίο εξυπηρετείται η ανάγκη για ταξιθέτηση των εγγραφών διακρίνει τους τρεις τρόπους υλοποίησης του αλγορίθμου σύνδεσης τον έναν από τον άλλο. Μάλιστα, η διαδικασία της ταξιθέτησης επηρεάζει σημαντικότατα την απόδοση και την πολυπλοκότητα του αλγορίθμου καθώς και την χρήση της μνήμης.

Οι τρεις τρόποι υλοποίησης επομένως του αλγόριθμου σύνδεσης αναλύονται στο καρτεσιανό γινόμενο, στην γραμμική αναζήτηση και στην χρήση συνάρτησης κατακερματισμού. Οι τρεις αυτοί τρόποι παρουσιάζονται ένας – ένας με ιδιαίτερη έμφαση στον τρίτο τρόπο ο οποίος αποτελεί και την τελική επιλογή για την διεξαγωγή των πειραμάτων καθώς έχει την βέλτιστη απόδοση.

4.2.4.1 Καρτεσιανό γινόμενο

Αυτός ο αλγόριθμος ταυτίζεται με τον αλγόριθμο συνδέσμου του Hadoop που παρουσιάστηκε στην § 3.4. Σε εκείνη την παρουσίαση το κλειδί αποστολής των εγγραφών στις ρ – διεργασίες και το κλειδί σύνδεσης τους ταυτίζονται και για αυτό οι εγγραφές ταξιτίθεντο κατευθείαν σε έναν αριθμό από ομάδες. Στην τρέχουσα περίπτωση το κλειδί αποστολής ταυτίζεται με τον αύξοντα αριθμό της ρ – διεργασίας παραλήπτη και προφανώς δεν ταυτίζεται με το κλειδί σύνδεσης. Συνεπώς, κάθε ρ – διεργασία δέχεται μόνο μία ομάδα εγγραφών στην οποία ομάδα κατά τα γνωστά εκτελείται μια πράξη καρτεσιανού γινομένου λαμβάνοντας έναν – έναν όλους τους δυνατούς συνδυασμούς εγγραφών. Για την ακρίβεια το καρτεσιανό γινόμενο λαμβάνει χώρα μεταξύ της ομάδας εισόδου και του εαυτού της. Έπειτα κάθε συνδυασμός εξετάζεται κατά πόσον μπορεί να παράξει μια έγκυρη εγγραφή εξόδου κι αν μπορεί, παράγεται αυτή η εγγραφή εξόδου.

Ο τελεστής του καρτεσιανού γινομένου αποδίδει ικανοποιητικά όταν τα σύνολα στα οποία εφαρμόζεται είναι σχετικά μικρά. Όταν τα σύνολα γίνονται μεγάλα τότε αποδίδει πενιχρά αφού έχει εκθετική πολυπλοκότητα. Πιο παραστατικά, για δύο σύνολα Α και Β που έχουν ως πλήθος στοιχείων κάποια δύναμη του 2, έστω $\Pi_A = 2^\alpha$ και $\Pi_B = 2^\beta$, οι δυνατοί συνδυασμοί ανέρχονται σε $\Pi_A * \Pi_B = 2^\alpha * 2^\beta = 2^{\alpha + \beta}$. Αν τα α, β = 10, τότε υπάρχουν 2^{20} = 1.048.576 συνδυασμοί. Με άλλα λόγια για μόνο 2^{10} = 1024 εγγραφές εισόδου σε μια ρ – διεργασία πρέπει να εξεταστούν 1.048.576 συνδυασμοί! Προφανώς, αυτό χαρακτηρίζεται επιεικώς ασύμφορο όταν οι εγγραφές εισόδου αυξάνονται έστω και λίγο εννοώντας μια αύξηση της τάξης του MB όπως έδειξαν τα πειράματα.

4.2.4.2 Γραμμική αναζήτηση

Σε αυτήν την περίπτωση, αρχικά οι εγγραφές εισόδου χωρίζονται με βάση την σχέση προέλευσης τους. Έπειτα οι εγγραφές της πρώτης σχέσης του συνδέσμου, έστω P_1, εισάγονται σε έναν δισδιάστατο πίνακα στον οποίο σταδιακά δημιουργούνται και συσσωρεύονται οι πλειάδες εξόδου. Κάθε εισαγόμενη εγγραφή, εισάγεται κατά στήλη, όπου κάθε ιδιότητα καταλαμβάνει μια συγκεκριμένη γραμμή. Ο προαναφερθείς πίνακας λέγεται πίνακας εξόδου μια και συγκεντρώνει τις πλειάδες εξόδου. Επίσης, έχει αρκετές στήλες ώστε να χωρούν όλες οι πλειάδες εξόδου εν τέλει. Οι γραμμές του πίνακα εξόδου είναι τόσες, όσες κι οι ιδιότητες κάθε πλειάδας εξόδου. Για παράδειγμα, αν μια πλειάδα εξόδου έχει 6 ιδιότητες τότε ο πίνακας έχει 6 γραμμές.

Αρχικά λοιπόν, ο πίνακας εξόδου περιέχει μόνο τις εγγραφές της P_1. Έπειτα για κάθε σχέση P_κ, κ = 2, ..., ν όπου ν το πλήθος των σχέσεων προς σύνδεση, ακολουθείται η ίδια διαδικασία. Σύμφωνα με αυτή την διαδικασία, κάθε εγγραφή σαρώνει τον πίνακα εξόδου εξετάζοντας αν μπορεί να ταιριάξει με την μέχρι τώρα συσσωρευθείσα πλειάδα στην υπό εξέταση στήλη. Ανεξάρτητα από το αποτέλεσμα της εκάστοτε εξέτασης για ταίριασμα η κάθε εγγραφή εξετάζει όλον τον πίνακα εξόδου όπως έχει διαμορφωθεί εκείνη την στιγμή. Ωστόσο αν υπάρξει ταίριασμα, το γεγονός αυτό σημειώνεται για την τρέχουσα στήλη. Η σημείωση αυτή συνεισφέρει σημαντικά στην επιτάχυνση του αλγορίθμου, γιατί κάθε στήλη που δεν σημειώνεται, δεν εξετάζεται από τις εγγραφές της επόμενης σχέσης. Κατά συνέπεια κάποιοι από τους συνδυασμούς που παράγει το καρτεσιανό γινόμενο αποφεύγονται προς εξοικονόμηση χρόνου.

Βέβαια ενώ εξοικονομείται χρόνος, σπαταλάται μνήμη καθώς οι μη σημειωμένες στήλες εξακολουθούν να καταλαμβάνουν την μνήμη που καταλάμβαναν. Προς ανακύκλωση της μνήμης των σημειωμένων στηλών αναπτύχθηκαν διάφορες τεχνικές – παραλλαγές του εν λόγω αλγορίθμου αλλά δεν παρατηρήθηκε κάποια βελτίωση.

Επιπλέον, να σημειωθεί πως επειδή κάποιες εγγραφές της ίδιας σχέσης μπορεί να

ταιριάζουν με την ίδια στήλη συμβαίνουν αντιγραφές των εν λόγω στηλών ώστε να παραχθούν οι αντίστοιχες εγγραφές εξόδου. Για αυτό ο πίνακας εξόδου πρέπει να έχει αρκετές στήλες ώστε να χωράει τις εγγραφές εξόδου εξ αντιγραφής. Αυτό βέβαια αυξάνει τις απαιτήσεις σε μνήμη.

Τέλος, η πολυπλοκότητα του αλγορίθμου είναι μεταβλητή. Πιο συγκεκριμένα, αν ν είναι το πλήθος των εγγραφών που λαμβάνει μια ρ – διεργασία τότε απαιτείται χρόνος ν ώστε να διαχωρισθούν οι εγγραφές σύμφωνα με την σχέση προέλευσης τους. Για την περαιτέρω ανάλυση, το μέγεθος της κ – οστής σχέσης συμβολίζεται με $ν_κ$. Κατά την πρώτη εκτέλεση του αλγορίθμου, για τις δύο πρώτες σχέσεις χρειάζεται χρόνος $ν_1ν_2$. Στο τέλος της πρώτης εκτέλεσης υφίστανται ως εγγραφές εξόδου ένα ποσοστό των αρχικών εγγραφών $ν_1$, $π_1ν_1$. Προχωρώντας στην δεύτερη εκτέλεση του αλγορίθμου ο αναγκαίος χρόνος ανέρχεται σε $(π_1ν_1)ν_3$. Στο πέρας αυτής της εκτέλεσης απομένουν $π_2(π_1ν_1)$ εγγραφές, δηλαδή, ένα ποσοστό των διαθέσιμων εγγραφών κατά την έναρξη της πρώτης επανάληψης. Γενικεύοντας, για την εκτέλεση μιας οποιασδήποτε *επανάληψης* ε χρειάζεται χρόνος $π_{ε+1}(π_ε(π_{ε-1}...(π_2(π_1ν_1)...)))ν_{ε+2} = ν_1ν_{ε+2}\prod_1^{ε+1} π_λ$ όπου το $π_λ$ συμβολίζει το ποσοστό των εγγραφών που έχουν απομείνει από την προηγούμενη επανάληψη, $ν_1$ είναι οι εγγραφές της πρώτης σχέσης και το $ν_{ε+2}$ αναπαριστά το πλήθος των εγγραφών της τρέχουσας, συνδεόμενης σχέσης. Σημειώνεται πως η πρώτη εκτέλεση του αλγορίθμου, αντιστοιχεί στην επανάληψη με αύξοντα αριθμό ε = 0 κι απαιτεί χρόνο, $ν_1ν_2$ όπου $\prod_1^1 π_λ = 1$. Τελικώς, η πολυπλοκότητα του αλγορίθμου για λ + 1 σχέσεις ισούται με $O(max(ν, ν_1 max(ν_{ε+2}\prod_1^{ε+1} π_λ)))$ όπου το ν ταυτίζεται με το πλήθος των εγγραφών που παραλαμβάνει μια ρ – διεργασία αρχικά.

Στην παραπάνω πολυπλοκότητα η εσωτερική συνάρτηση μεγίστου αντανακλά το γεγονός πως οι χρόνοι από επανάληψη σε επανάληψη προστίθενται ο ένας στον άλλο. Οπότε η πολυπλοκότητα των επαναλήψεων ισούται με την πιο χρονοβόρα επανάληψη. Η εξωτερική συνάρτηση μεγίστου αναδεικνύει μια εν δυνάμει γραμμική συμπεριφορά του αλγορίθμου και την επιλεκτικότητα του συνδέσμου. Για αρχή το γινόμενο $ν_1ν_2 < ν$ σε κάποιες περιπτώσεις. Αν $ν_1 = 10$, $ν_2 = 10$, $ν_3 = 40$ και $ν_4 = 60$ τότε, $ν_1ν_2 = 100$ και $ν = 120$. Αν επιπλέον, κάθε άλλο γινόμενο $ν_1ν_{ε+2}\prod_1^{ε+1} π_λ < ν$ τότε, ο αλγόριθμος έχει γραμμική πολυπλοκότητα. Βέβαια, για να ισχύει η τελευταία ανισότητα θα πρέπει τα ποσοστά $π_λ$ να έχουν κάποια κατάλληλη τιμή. Στο προηγούμενο παράδειγμα, για το γινόμενο $π_1ν_1ν_3$ πρέπει $π_1 < 0,3$ και για το γινόμενο $π_2π_1ν_1ν_4$ πρέπει $π_2π_1 < 0,2$, δηλαδή, $π_2 < 0,66$ περίπου. Από την άλλη, αν έστω κι ένα γινόμενο, είτε κάποιο από τα $ν_1ν_{ε+2}\prod_1^{ε+1} π_λ$ για ε = 1, 2 είτε το $ν_1ν_2$ ξεπερνά το ν, τότε ο αλγόριθμος έχει τετραγωνική πολυπλοκότητα για δεδομένα μεγέθη σχέσεων.

Ακόμη, τα ποσοστά $π_λ$ αναδεικνύουν την επιλεκτικότητα του αλγορίθμου. Αυτή μάλιστα μπορεί να μεταβάλλεται από επανάληψη σε επανάληψη. Σε μια επανάληψη το $π_λ > 1$ και σε μια άλλη ≤ 1. Όσο το $π_λ$ μιας επανάληψης βρίσκεται χαμηλότερα της μονάδας τόσο πιο επιλεκτικός χαρακτηρίζεται ο αντίστοιχος σύνδεσμος.

4.2.4.3 Συνάρτηση κατακερματισμού

Οι γενικές αρχές που εφαρμόστηκαν στον αλγόριθμο γραμμικής αναζήτησης ισχύουν κι

εδώ. Στην αρχή οι εγγραφές εισόδου μιας ρ – διεργασίας ξεχωρίζονται με βάση την σχέση προέλευσης τους. Έπειτα οι εγγραφές της πρώτης σχέσης του συνδέσμου, έστω P_1 εισάγονται σε μια κατάλληλη δομή όπου συσσωρεύονται οι εγγραφές εξόδου. Η εν λόγω δομή λέγεται δομή εξόδου.

Εν προκειμένω, η δομή εξόδου συνιστά έναν πίνακα κατακερματισμού(ΠΚ). Το κλειδί σε αυτόν τον πίνακα συγκροτείται από τις κοινές ιδιότητες των σχέσεων P_1 και P_2, όπου η P_2 αφορά στην δεύτερη σχέση του συνδέσμου όπως αυτός διαβάζεται από τα αριστερά προς τα δεξιά. Η τιμή που αντιστοιχεί σε ένα κλειδί αποτελείται από εγγραφές των διαφόρων σχέσεων του συνδέσμου κι εν δυνάμει μπορούν να παράξουν μία εγγραφή εξόδου. Στην αρχή, τα κλειδιά κι οι τιμές του ΠΚ είναι αυτά της P_1.

Στην συνέχεια χρειάζεται να συγκεντρωθούν οι κατάλληλες εγγραφές από τις άλλες σχέσεις στον ΠΚ στην κατάλληλη θέση. Για να γίνει αυτό απαιτείται ένας δεύτερος, βοηθητικός πίνακας κατακερματισμού(ΒΠΚ). Η ορθή λειτουργία του ΒΠΚ προϋποθέτει την σωστή λειτουργία του τρίτου προ – επεξεργαστικού βήματος που εξηγείται παρακάτω.

Το κλειδί του ΒΠΚ συνίσταται από τις κοινές ιδιότητες της σχέσης $P_κ$ με την $P_{κ+1}$. Η σχέση $P_κ$ είναι η σχέση της οποίας οι εγγραφές εισάγονται στον ΠΚ στην τρέχουσα φάση του αλγορίθμου. Η $P_{κ+1}$ αποτελεί την σχέση της οποίας οι εγγραφές θα εισαχθούν στον ΠΚ στην επόμενη φάση του αλγορίθμου. Η τιμή του ΒΠΚ είναι μία λίστα από κλειδιά του ΠΚ. Εν παραδείγματι, αν μια εγγραφή Ε της $P_κ$ έχει εισαχθεί στον ΠΚ σε τρεις θέσεις με κλειδιά $κ_1$, $κ_2$, $κ_3$ τότε, στον ΒΠΚ στο κλειδί της Ε αντιστοιχεί ως τιμή η λίστα των κλειδιών $κ_1$, $κ_2$, $κ_3$. Με αυτό τον τρόπο μια εγγραφή Φ της $P_{κ+1}$ που έχει το ίδιο κλειδί με την Ε θα εισαχθεί στον ΠΚ στις ίδιες θέσεις που έχει εισαχθεί κι η Ε. Τελικά, με αυτόν τον τρόπο γίνεται η συσσώρευση των εγγραφών. Προς βαθύτερη κατανόηση του αλγορίθμου, ο αναγνώστης ενθαρρύνεται να μελετήσει το τρίτο προ επεξεργαστικό βήμα που αναλύεται στην § 4.2.4.3.1.

Όσον αφορά στην πολυπλοκότητα του αλγορίθμου είναι γραμμική. Αυτό οφείλεται αρχικά στον αρχικό διαχωρισμό των ν εγγραφών εισόδου. Έπειτα οφείλεται στην συνάρτηση κατακερματισμού που στην χειρότερη περίπτωση έχει γραμμική πολυπλοκότητα.

4.2.4.3.1 Τρίτο προ επεξεργαστικό βήμα – Ανάλυση των εγγραφών στα κλειδιά τους

Σε έναν σύνδεσμο τριών και πλέον σχέσεων, π.χ. $P_1 \bowtie P_2 \bowtie P_3 \bowtie ... \bowtie P_ν$, κάθε ενδιάμεση σχέση έχει δύο κλειδιά, το προ – κλειδί και το μετά – κλειδί. Το προ – κλειδί είναι το σύνολο των κοινών ιδιοτήτων μιας σχέσης με την προηγούμενη της σχέση. Ως μετά – κλειδί χαρακτηρίζεται το σύνολο των ιδιοτήτων μιας σχέσης με την επόμενη της σχέση. Οι πρώτη και τελευταία σχέσεις θεωρούνται καταχρηστικά πως έχουν κι αυτές δύο κλειδιά εκ των οποίων το ένα όμως είναι κενό. Για την μεν πρώτη σχέση το προ – κλειδί είναι κενό και για την δε τελευταία σχέση το μετά – κλειδί είναι το κενό.

Το τρίτο προ – επεξεργαστικό βήμα εξάγει για κάθε μία από τις σχέσεις ενός συνδέσμου αυτά τα κλειδιά. Χάρη σε αυτό το βήμα, ένας πολλαπλός σύνδεσμος μπορεί να υπολογιστεί απευθείας. Αναλυτικότερα, έχοντας κατά νου την ανάλυση του αλγορίθμου σύνδεσης με χρήση συνάρτησης κατακερματισμού, υπάρχουν δύο πίνακες, ο πίνακας κατακερματισμού εξόδου(ΠΚ) κι ο βοηθητικός πίνακας κατακερματισμού(ΒΠΚ). Για λόγους κατανόησης, ας υποτεθεί πως θέλουμε να υπολογίσουμε τον σύνδεσμο $P_1(A_1, A_2, A_3) \bowtie P_2(A_1, A_2, A_4) \bowtie P_3(A_1, A_4, A_5) \bowtie P_4(A_4, A_6)$. Τα προ – κλειδιά και τα μετά – κλειδιά των σχέσεων του συνδέσμου φαίνονται στον Πίνακα 3, ενώ στον Πίνακα 4 φαίνονται ενδεικτικές εγγραφές των σχέσεων.

P_1		P_2		P_3		P_4	
Προ – κλειδί	Μετά - κλειδί	Προ – κλειδί	Μετά - κλειδί	Προ – κλειδί	Μετά - κλειδί	Προ – κλειδί	Μετά - κλειδί
Ø	A_1, A_2	A_1, A_2	A_1, A_4	A_1, A_4	A_4	A_4	Ø

Πίνακας 3: Προ – κλειδί και μετά – κλειδί ενός συνδέσμου 4 σχέσεων

P_1	P_2	P_3	P_4
1, α, Α	1, α, ΑΒ	1, ΑΒ, Ε	ΑΒ, ΕΖ
2, β, Β	2, β, ΒΓ	2, ΒΓ, Ζ	ΒΓ, ΖΗ
3, γ, Γ	3, γ, ΓΔ	3, ΓΔ, Η	ΓΔ, ΗΘ

Πίνακας 4: Ενδεικτικές εγγραφές ενός συνδέσμου 4 σχέσεων

Πώς συνδέονται οι παραπάνω σχέσεις; Αρχικά ο ΠΚ έχει όπως στον Πίνακα 5. Στον πίνακα αυτόν να παρατηρηθεί πως η σχέση προέλευσης των εγγραφών σημειώνεται ώστε αργότερα να παραχθούν σωστά οι εγγραφές εξόδου. Ο ΒΠΚ στην αρχή περιέχει ως κλειδιά τα μετά – κλειδιά της P_1 κι ως τιμές τα κλειδιά του ΠΚ.

Κλειδί	Τιμή
1, α	P_1(1, α, Α)
2, β	P_1(2, β, Β)
3, γ	P_1(3, γ, Γ)

Πίνακας 5: Αρχική κατάσταση του ΠΚ

Κλειδί	Τιμή
1, α	1, α
2, β	2, β
3, γ	3, γ

Πίνακας 6: Αρχική κατάσταση του ΒΠΚ

Έπειτα, μία – μία οι εγγραφές της P_2 εξετάζονται ως προς το προ – κλειδί τους κι εισάγονται στον ΠΚ κι ο ΠΚ διαμορφώνεται όπως στον Πίνακα 6. Ο ΒΠΚ με βάση το μετά – κλειδί των εγγραφών της P_2 διαμορφώνεται όπως στον Πίνακα 7.

Κλειδί	Τιμή
1, α	P_1(1, α, Α), P_2(1, α, ΑΒ)
2, β	P_1(2, β, Β), P_2(2, β, ΒΓ)
3, γ	P_1(3, γ, Γ), P_2(3, γ, ΓΔ)

Πίνακας 7: Κατάσταση του ΠΚ μετά την εισαγωγή των εγγραφών της P_2

Κλειδί	Τιμή
1, ΑΒ	1, α
2, ΒΓ	2, β
3, ΓΔ	3, γ

Πίνακας 8: Κατάσταση του ΒΠΚ μετά την επεξεργασία των εγγραφών της P_2

Όταν έρχεται η σειρά των εγγραφών της σχέσης P_3, τότε αυτές εισάγονται στον ΠΚ χάρη στον ΒΠΚ. Για την εγγραφή 1, ΑΒ, Ε, το προ – κλειδί είναι το 1, ΑΒ. Οπότε σύμφωνα με τον ΒΠΚ του Πίνακα 7, η εγγραφή 1, ΑΒ, Ε, θα πρέπει να εισαχθεί στην θέση 1,α του ΠΚ. Η ίδια διαδικασία ακολουθείται και για τις άλλες εγγραφές της P_3 λαμβάνοντας τον Πίνακα 8 για τον ΠΚ. Έπειτα, ο ΒΠΚ ανανεώνεται όπως στον Πίνακα 9 δεδομένων των μετά – κλειδιών των εγγραφών της P_3. Σημειώνεται πως στον Πίνακα 9 η σειρά των εγγραφών διαφέρει επίτηδες από την σειρά του Πίνακα 7. Αυτό συμβαίνει ώστε να γίνει σαφές πως ο ΒΠΚ ανανεώνεται στην βάση μιας συνάρτησης κατακερματισμού κι άρα η θέση των εγγραφών του είναι τυχαία μέσα σε αυτόν.

Τέλος, έρχεται η σειρά των εγγραφών της σχέσης P_4 οι οποίες πάλι με την βοηθεια του ΒΠΚ, Πίνακας 9, εισάγονται στον ΠΚ, Πίνακας 10. Δεδομένου πως το μετά – κλειδί των εγγραφών της P_4 είναι κενό, ο ΒΠΚ αδειάζει έχοντας εκπληρώσει την αποστολή του.

Έχοντας πλέον στον ΠΚ συγκεντρώσει όλες τις εγγραφές των σχέσεων του συνδέσμου που μπορούν εν δυνάμει να παράξουν μια εγγραφή εξόδου, ξεκινά το στάδιο παραγωγής αυτών των εγγραφών εξόδου.

Κλειδί	Τιμή
1, α	P_1(1, α, Α), P_2(1, α, ΑΒ), P_3(1, ΑΒ, Ε)
2, β	P_1(2, β, Β), P_2(2, β, ΒΓ), P_3(2, ΒΓ, Ζ)
3, γ	P_1(3, γ, Γ), P_2(3, γ, ΓΔ), P_3(3, ΓΔ, Η)

Πίνακας 9: Κατάσταση του ΠΚ μετά την εισαγωγή των εγγραφών της P_3

Κλειδί	Τιμή
ΑΒ	1, α
ΓΔ	3, γ
ΒΓ	2, β

Πίνακας 10: Κατάσταση του ΒΠΚ μετά την επεξεργασία των εγγραφών της P_3

Κλειδί	Τιμή
1, α	P_1(1, α, A), P_2(1, α, AB), P_3(1, AB, E), P_4(AB, EZ)
2, β	P_1(2, β, B), P_2(2, β, ΒΓ), P_3(2, ΒΓ, Z), P_4(ΒΓ, ZH)
3, γ	P_1(3, γ, Γ), P_2(3, γ, ΓΔ), P_3(3, ΓΔ, H), P_4(ΓΔ, HΘ)

Πίνακας 11: Τελική κατάσταση του ΠΚ μετά την εισαγωγή των εγγραφών της P_4

4.2.4.4 Διάγραμμα κλάσεων αλγορίθμου σύνδεσης πολλαπλού συνδέσμου

Παρουσιάζεται στο Σχήμα 40 το διάγραμμα κλάσεων του αλγορίθμου σύνδεσης του πολλαπλού συνδέσμου που είναι κοινό και για τις τρεις εν δυνάμει υλοποιήσεις αυτού. Στο διάγραμμα αυτό,

1 MapperClass : κληρονομεί την DataJoinMapperBase, υλοποιεί την μ – διεργασία και με την μέθοδο,

   1.1 generateInputTag : δημιουργεί την ετικέτα που προσδιορίζει μοναδικά την προέλευση της τρέχουσας εγγραφής.

   1.2 generateGroupKey : δημιουργεί το κλειδί της νέας εγγραφής.

   1.3 generateTaggedMapOutput : παράγει την νέα εγγραφή καλώντας τις μεθόδους generateInputTag και generateGroupKey. Η νέα εγγραφή αποτελεί ένα αντικείμενο της κλάσης TaggedMapOutput.

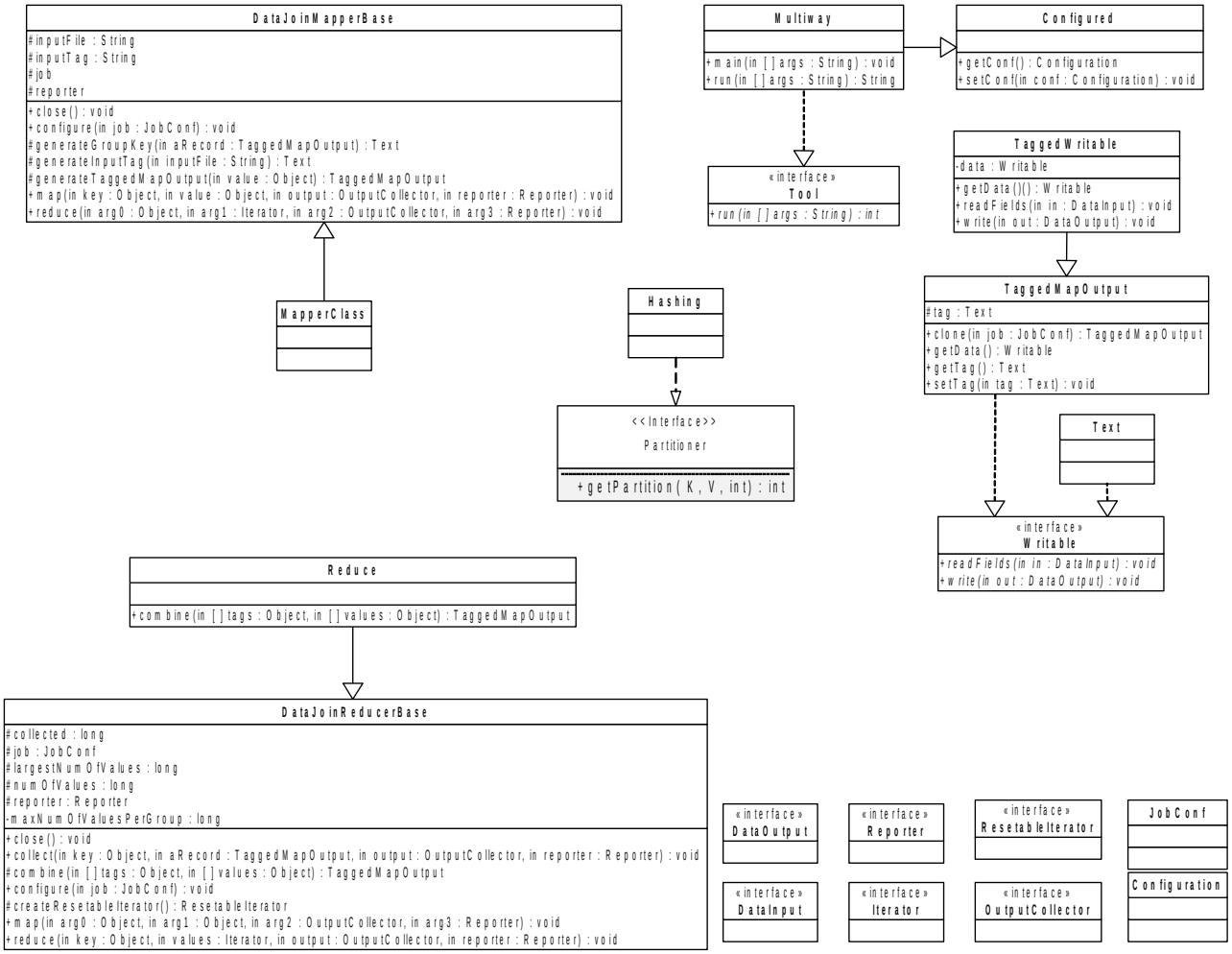
2 Reduce : κληρονομεί την DataJoinReducerBase, ενσαρκώνει την ρ – διεργασία και με την μέθοδο,

   2.1 combine : κάνει την σύνδεση μεταξύ των δεδομένων.

3 TaggedMapOutput : υλοποιεί την διαπροσωπεία Writable, αποτελεί μια κλάση – περιτύλιγμα αντικειμένων της κλάσης Text. Κάθε εγγραφή όταν διαβάζεται από το αρχείο - σχέση είναι ένα αντικείμενο String και μετατρέπεται σε αντικείμενο Text ώστε να μπορεί να δεχτεί την ετικέτα και το κλειδί της νέας εγγραφής.

4 Hashing : υλοποιεί την διαπροσωπεία Partitioner και επιτελεί την λειτουργία της διανομής των εγγραφών εξόδου της μ – φάσης στις ρ – διεργασίες.

Τέλος, όπως και στο Σχήμα 36 περισσότερες λεπτομέρειες για τις κλάσεις του διαγράμματος μπορούν να βρεθούν στο διαδίκτυο. Οποιαδήποτε παράθεση επιπρόσθετης πληροφορίας δεν προσθέτει κάτι επιπλέον στην κατανόηση.



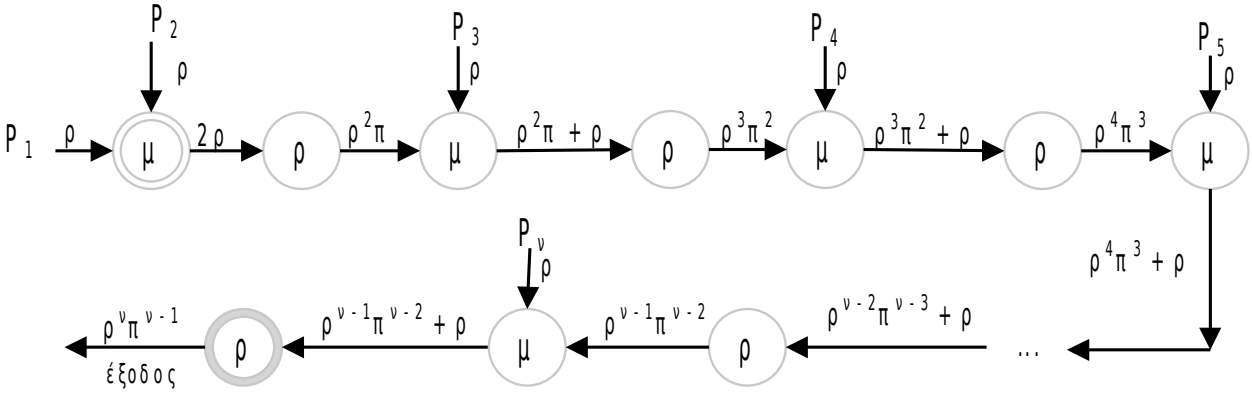
Σχήμα 40 : Διάγραμμα κλάσεων για τον πολλαπλό σύνδεσμο

4.3 Κόστος επικοινωνίας

Τώρα εξετάζεται θεωρητικά το κόστος επικοινωνίας της ακολουθίας συνδέσμων δύο σχέσεων και του πολλαπλού συνδέσμου. Για αρχή, ένας σύνδεσμος σχέσεων έχει την μορφή $P_1 \bowtie P_2 \bowtie P_3 \bowtie ... \bowtie P_v$ αδιαφορώντας για τις συγκεκριμένες ιδιότητες των σχέσεων του συνδέσμου. Χωρίς βλάβη της γενικότητας υποθέτουμε πως όλες οι σχέσεις έχουν το ίδιο μέγεθος.

Για την ακολουθία των διπλών συνδέσμων υποθέτουμε πως η σειρά υπολογισμού των συνδέσμων αυτών γίνεται ως $(... ((P_1 \bowtie P_2) \bowtie P_3) \bowtie ...) \bowtie P_v$ χωρίς πάλι βλάβη της γενικότητας. Συνεπώς προκύπτει το Σχήμα 41, όπου το μ συμβολίζει την μ – φάση του εκάστοτε διπλού συνδέσμου και το ρ εντός ενός κύκλου συμβολίζει την αντίστοιχη ρ – φάση. Το ρ εκτός κύκλου αναπαριστά το μέγεθος όλων των σχέσεων. Το π αναπαριστά το ποσοστό των εγγραφών που απομένουν στο τέλος της τρέχουσας ρ – φάσης και χωρίς βλάβη της γενικότητας θεωρείται το ίδιο για κάθε ρ – φάση της ακολουθίας των διπλών συνδέσμων.

Στο Σχήμα 41 η κάθε μ – φάση της ακολουθίας των διπλών συνδέσμων έχει ως εγγραφές εξόδου τις εγγραφές εισόδου της που έχουν υποστεί την επεξεργασία της § 3.4.1. Το πλήθος επομένως των εγγραφών εξόδου ισούται με το πλήθος των εγγραφών εισόδου. Στην κάθε ρ – φάση οι εγγραφές εισόδου της δέχονται μία πράξη καρτεσιανού γινομένου δίνοντας τις εγγραφές εξόδου του αντίστοιχου συνδέσμου. Βέβαια από αυτές τις εν δυνάμει εγγραφές εξόδου μόνο το π ποσοστό αυτών αποτελούν όντως εγγραφές εξόδου. Για παράδειγμα, στην πρώτη ρ – φάση που εμφανίζεται στο Σχήμα 41, το καρτεσιανό γινόμενο παράγει εν δυνάμει ρ^2 εγγραφές εξόδου εκ των οποίων μόνο το π ποσοστό αυτών είναι όντως εγγραφές εξόδου συνεχίζοντας στον επόμενο σύνδεσμο της ακολουθίας.



Σχήμα 41 : Ροή δεδομένων σε μια ακολουθία διπλών συνδέσμων ν σχέσεων

Το κόστος επικοινωνίας της ακολουθίας διπλών συνδέσμων

$$
\begin{aligned}
K_2 &= 2\rho + \rho^2\pi + (\rho^2\pi + \rho) + \rho^3\pi^2 + (\rho^3\pi^2 + \rho) + \rho^4\pi^3 \\
&\quad + (\rho^4\pi^3 + \rho) + ... + (\rho^{v-2}\pi^{v-3} + \rho) + \rho^{v-1}\pi^{v-2} + (\rho^{v-1}\pi^{v-2} + \rho) \\
&= 2\rho + \rho + \rho + ... + \rho + \rho + 2\rho^2\pi + 2\rho^3\pi^2 + 2\rho^4\pi^3 + ... + 2\rho^{v-1}\pi^{v-2} \\
&= 2\rho + \sum_{\lambda=3}^{v} \rho + 2\sum_{\lambda=3}^{v} \rho^{\lambda-1}\pi^{\lambda-2} = 2\rho + \rho(v-2) + 2\rho^2\pi \sum_{\lambda=0}^{v-3} \rho^{\lambda}\pi^{\lambda} \\
&= \rho v + 2\rho^2\pi \frac{1-(\rho\pi)^{v-2}}{1-\rho\pi}
\end{aligned}
$$

είναι σταθερό, ανεξάρτητο από το πλήθος των ρ – διεργασιών κι εξαρτημένο τόσο από το μέγεθος

των σχέσεων όσο κι από το ποσοστό π των εγγραφών εξόδου. Επίσης όπως φαίνεται, το εν λόγω κόστος επικοινωνίας δεν εξαρτάται από την δομή του συνδέσμου μια και επεξεργάζεται τους συνδέσμους ανά δύο. Τέλος, το εν λόγω κόστος μπορεί να εκτιμηθεί εκ των προτέρων. Αυτά το θεωρητικά συμπεράσματα επιβεβαιώνονται πειραματικά στο 5^{ο} Κεφάλαιο που έπεται. Στην παραπάνω εξίσωση οι αρχικές σχέσεις θεωρούνται αποθηκευμένες τοπικά κι άρα δεν επιβαρύνουν το δίκτυο σαν κόστος επικοινωνίας.

Το κόστος επικοινωνίας του πολλαπλού συνδέσμου εξαρτάται από το πλήθος των ρ – διεργασιών αφού η συνάρτηση κόστους καθώς και τα μερίδια των ιδιοτήτων του μ – κλειδιού εξαρτώνται από το πλήθος αυτό. Κι αυτό το γεγονός επιβεβαιώνεται πειραματικά. Προφανώς, το εν λόγω κόστος επικοινωνίας εξαρτάται κι από το μέγεθος των σχέσεων και μάλιστα ανάλογα. Όσο αυξάνεται το μέγεθος των σχέσεων τόσο πιο πολλές εγγραφές πρέπει να σταλούν στις ρ – διεργασίες κι άρα τόσο μεγαλύτερο θα είναι το αντίστοιχο κόστος επικοινωνίας. Το κόστος επικοινωνίας εξαρτάται από την δομή του συνδέσμου αφού ανάλογα με την δομή προκύπτει άλλη συνάρτηση κόστους. Τέλος, από τον Πίνακα 2 της σελίδας 92 που αντιστοιχεί σε έναν σύνδεσμο μπορεί να υπολογιστεί ακριβώς το κόστος επικοινωνίας εκ των προτέρων για ένα δεδομένο πλήθος ρ – διεργασιών.

Η σελίδα αυτή είναι σκόπιμα λευκή.

ΚΕΦΑΛΑΙΟ 5

ΠΕΙΡΑΜΑΤΑ

Η γνώση είναι το ισχυρότερο όπλο το οποίο μπορείς να χειριστείς για να αλλάξεις τον κόσμο.

Nelson Mandela

Με μια ματιά

5.1. Πειραματικό περιβάλλον
5.2. Μετρήσεις
5.3. Δομές συνδέσμων
5.4. Πειράματα κάθε εαυτά

Στο παρόν κεφάλαιο συγκρίνονται οι δύο αλγόριθμοι σύνδεσης πάνω σε τρεις διαφορετικές δομές συνδέσμου. Ο πρώτος αλγόριθμος αφορά στην ακολουθία διπλών συνδέσμων στο πλαίσιο του Hadoop κι ο δεύτερος αφορά στον πολλαπλό σύνδεσμο με χρήση μιας συνάρτησης κατακερματισμού. Οι εν λόγω αλγόριθμοι συγκρίνονται ως προς τον χρόνο εκτέλεσης και στο κόστος επικοινωνίας όπως αυτό περιγράφεται στην § 2.3. Ωστόσο με λίγα λόγια, το κόστος επικοινωνίας ισούται με το ποσό των ενδιάμεσων αποτελεσμάτων στην διάρκεια εκτέλεσης ενός αλγόριθμου σύνδεσης. Ο χρόνος μετριέται σε δευτερόλεπτα και το κόστος επικοινωνίας μετριέται σε εγγραφές. Οι τρεις δομές συνδέσμου είναι ο σύνδεσμος αλυσίδα, ο σύνδεσμος αστέρας κι ο απλός σύνδεσμος. Ο απλός σύνδεσμος δεν έχει κάποια συγκεκριμένη δομή όπως οι δύο άλλοι σύνδεσμοι.

5.1 Πειραματικό περιβάλλον

Τα πειράματα εκτελέστηκαν στο Hadoop 1.2.1 σε ένα σύμπλεγμα 11 κόμβων(master και σκλάβοι). Τα χαρακτηριστικά αυτών των κόμβων συνοψίζονται στον Πίνακα 1.

Κόμβος	**RAM(GB)**	**HDD(GB)**	**CPUs**	**OS**
Master, Slave1 - Slave4	8	100	8 at 2.1 GHz	Ubuntu / Linux 13.04, kernel version 3.11
Slave5 - Slave10	8	40	4 at 2.1 GHz	

Table 1: Προδιαγραφές υλισμικού των κόμβων του συμπλέγματος

5.2 Μετρήσεις

Εξίσου και για τους δύο αλγόριθμους σύνδεσης, ο χρόνος εκτέλεσης και το κόστος επικοινωνίας έχουν μετρηθεί με διάφορους τρόπους.

5.2.1 Χρόνος εκτέλεσης

Όνομα μέτρησης	**Μέτρηση**
Πολλαπλός σύνδεσμος και μεμονωμένες φάσεις της ακολουθίας διπλών συνδέσμων	
Συνολικός χρόνος	task_finish_time – task_launch_time
Χρόνος μ – φάσης	last_finished_map_task – task_launch_time
Ακριβής χρόνος μ – φάσης	last_finished_map_task – first_started_map_task
Χρόνος ρ – φάσης	task_finish_time - first_started_reduce_task
Ακριβής χρόνος ρ - φάσης	last_finish_reduce_task - first_started_reduce_task

Table 2: Μετρήσεις χρόνου εκτέλεσης

5.2.2 Κόστος επικοινωνίας

Όνομα μέτρησης	Μέτρηση
Πολλαπλός σύνδεσμος και μεμονωμένες φάσεις της ακολουθίας διπλών συνδέσμων	
Είσοδος μ – φάσης	Εγγραφές εισόδου για κάθε μ – φάση
Έξοδος μ – φάσης	Εγγραφές εξόδου για κάθε μ – φάση
Είσοδος ρ – φάσης	Εγγραφές εισόδου για κάθε ρ – φάση
Έξοδος ρ – φάσης	Εγγραφές εξόδου για κάθε ρ – φάση

Table 3: Μετρήσεις κόστους επικοινωνίας

Στους Πίνακες 2 και 3 οι μετρήσεις που έχουν γραφτεί σε πλάγια γραφή είναι κι οι μετρήσεις που παρουσιάζονται στην ακόλουθη πειραματική ανάλυση.

5.3 Δομές συνδέσμων

Οι δομές των συνδέσμων στις οποίες εφαρμόζονται οι αλγόριθμοι σύνδεσης είναι,

Όνομα Δομής	Δομή
Σύνδεσμος αλυσίδα	$R_1(A_1, A_2)$ |X| $R_2(A_2, A_3)$ |X| $R_3(A_3, A_4)$ |X| $R_4(A_4, A_5)$
Σύνδεσμος απλός	$R_1(A_1, A_2, A_3)$ |X| $R_2(A_1, A_2, A_4)$ |X| $R_3(A_1, A_4, A_5)$ |X| $R_4(A_4, A_6)$
Σύνδεσμος αστέρα	$R_1(A_1, A_2, A_3, A_4)$ |X| $R_2(A_1, A_5, A_6)$ |X| $R_3(A_2, A_7)$ |X| $R_4(A_3, A_8, A_9)$

Table 4: Δομές συνδέσμων

και λέγονται “δομές συνδέσμων”, γιατί όταν τοποθετούνται στον χώρο έχουν μια συγκεκριμένη γεωμετρική δομή όπως και τα ονόματα τους υποδηλώνουν. Όσο για τον απλό σύνδεσμο, δεν έχει κάποια συγκεκριμένη γεωμετρική δομή όπως μια αλυσίδα ή ένας αστέρα, αλλά μπορεί να αναχθεί σε αλυσίδα. Εν κατακλείδι, το σημαντικό σχετικά με έναν σύνδεσμο είναι να υπάρχει ένα μονοπάτι από κάθε σχέση προς κάθε άλλη σχέση. Άνευ αυτής της ιδιότητας ένας αλγόριθμος συνδέσμου δεν μπορεί να εφαρμοστεί καταλλήλως.

5.4 Τα πειράματα κάθε εαυτά

Εκτελέστηκαν δύο σειρές πειραμάτων όπου οι εξαρτημένες μεταβλητές είναι ο χρόνος εκτέλεσης και το κόστος επικοινωνίας. Στην πρώτη σειρά πειραμάτων η ανεξάρτητη μεταβλητή είναι το μέγεθος των σχέσεων εισόδου μετρημένο σε εγγραφές / σχέση. Στην δεύτερη σειρά πειραμάτων υπάρχουν δύο ανεξάρτητες μεταβλητές, το πλήθος των επεξεργαστών και το πλήθος των ρ – διεργασιών.

5.4.1 Πειράματα ως προς το μέγεθος των σχέσεων

Για αυτά τα πειράματα το πλήθος των ρ – διεργασιών ανέρχεται σε 20.

Σύμφωνα με το άρθρο [22], σελίδα 9, § '4.1 Star Joins', "...It is expected that the fact table is very large, while the dimension tables are smaller"(… αναμένεται ότι ο κεντρικός πίνακας είναι πολύ μεγάλος, ενώ οι περιφερειακοί πίνακες είναι μικρότεροι). Επομένως, για τον σύνδεσμο αστέρα, ο κεντρικός πίνακας είναι 7 φορές μεγαλύτερος συγκριτικά με τους περιφερειακούς. Ο λόγος 7 : 1 αποτελεί πιο πολύ μια προσέγγιση για μικρά σύνολα δεδομένων π.χ. 1000 εγγραφές / σχέση αλλά, είναι σχεδόν ακριβής (απόκλιση < 2%) για μεγαλύτερα σύνολα δεδομένων. Επιπλέον, όταν λέγεται πως ο σύνδεσμος αστέρας έχει για παράδειγμα 1.000.000 εγγραφές / σχέση αυτό που ισχύει στην πραγματικότητα είναι πως ο κεντρικός πίνακας έχει 1.000.000 εγγραφές κι οι υπόλοιπες σχέσεις έχουν το 1 / 7 των 1.000.000 εγγραφών.

Όσο για τις υπόλοιπες δομές συνδέσμων, ο απλός σύνδεσμος κι ο σύνδεσμος αλυσίδα, όταν λέγεται πως περιέχουν 1.000.000 εγγραφές / σχέση, τότε όλες οι σχέσεις κάθε μίας δομής έχουν 1.000.000 εγγραφές.

Συνεχίζοντας, τα μεγαλύτερα πειράματα αφορούν σε 1.000.000 εγγραφές / σχέση. Αυτό το γεγονός ευθυγραμμίζεται με τα προηγούμενα σχετικά πειράματα από τον Βίκτωρα Κυρίτση τα οποία παρουσιάζονται στο [22]. Για τις δομές συνδέσμων του Πίνακα 4, το μέγεθος κάθε μίας δομής δίνεται στον ακόλουθο Πίνακα 5. Στον πίνακα αυτόν, οι στήλες "Ίδιο" και "Μισό" αναφέρονται στις ενδιάμεσες σχέσεις μιας δομής συνδέσμου. Ειδικότερα, "λένε" αν οι ενδιάμεσες σχέσεις έχουν το ίδιο μέγεθος με τις ακραίες ή το μισό. Αυτό ισχύει ακόμα και για τον σύνδεσμο αστέρα. Ωστόσο, επειδή ο κεντρικός πίνακας ήδη κυριαρχεί στην δομή αστέρα, η επίδραση του μεγέθους των ενδιάμεσων σχέσεων στο μέγεθος του συνδέσμου είναι πολύ μικρότερη από ότι στις άλλες δομές.

εγγραφές / σχέση	**Σύνδεσμος αλυσίδα**		**Απλός σύνδεσμος**		**Σύνδεσμος αστέρα**	
	Ίδιο	**Μισό**	**Ίδιο**	**Μισό**	**Ίδιο**	**Μισό**
10	566 B	426 B	759 B	547 B	392 B	366 B
10^2	5,6 KB	4,2 KB	7,7 KB	5,7 KB	3,6 KB	3,4 KB
10^3	56,2 KB	42,2 KB	77,2 KB	56,3 KB	36 KB	33,6 KB
10^4	560 KB	420,2 KB	770 KB	559,8 KB	360,1 KB	335,2 KB
10^5	5,6 MB	4,2 MB	7,7 MB	5,6 MB	3,6 MB	3,4 MB
10^6	56 MB	42 MB	77 MB	56 MB	36 MB	33,5 MB
$5 * 10^6$	280 MB	210 MB	385 MB	280 MB	180 MB	167, 5 MB
10^7	560 MB	420 MB	770 MB	560 MB	360 MB	335 MB
10^8	5,6 GB	4.2 GB	7,7 GB	5,6 GB	3,6 MB	3,4 GB

Table 5: Μέγεθος των δομών συνδέσμων καθώς ο λόγος εγγραφές / σχέση μεταβάλλεται κι όταν κάθε εγγραφή έχει μήκος από 2 – 10 χαρακτήρες.

Καταλήγοντας, τα παρουσιαζόμενα πειράματα αφορούν σε 1.000.0000 εγγραφές / σχέση το πολύ, για κάθε δομή συνδέσμου.

Τα δεδομένα που χρειάστηκαν κατά την εκτέλεση των πειραμάτων αποτελούν αλφαριθμητικά(strings) μήκους από 2 ως και 10 χαρακτήρων και δεν έχουν κάποια ιδιαίτερο

σχήμα. Κάθε αλφαριθμητικό μπορεί να περιέχει οποιονδήποτε εκτυπώσιμο χαρακτήρα εκτός από την κάτω παύλα, “_” και το κόμμα, “,” γιατί αυτοί οι χαρακτήρες επιτελούν συγκεκριμένες, αλγοριθμικές λειτουργίες.

Όλα τα ακόλουθα γραφήματα είναι σε λογαριθμική κλίμακα.

5.4.2 Πειραματική αποτίμηση και στατιστική ανάλυση

Σε ότι ακολουθεί εμφανίζονται, δεν εμφανίζονται γραφήματα και για τους τρεις συνδέσμους κατά την μελέτη είτε του χρόνου εκτέλεσης είτε του κόστους επικοινωνίας. Αυτό συμβαίνει, γιατί τα γραφήματα που λείπουν είναι παρόμοια με τα γραφήματα που εμφανίζονται κι η παρουσίαση τους επόμενως δεν θα προσέθετε κάτι.

5.4.2.1 Χρόνος εκτέλεσης

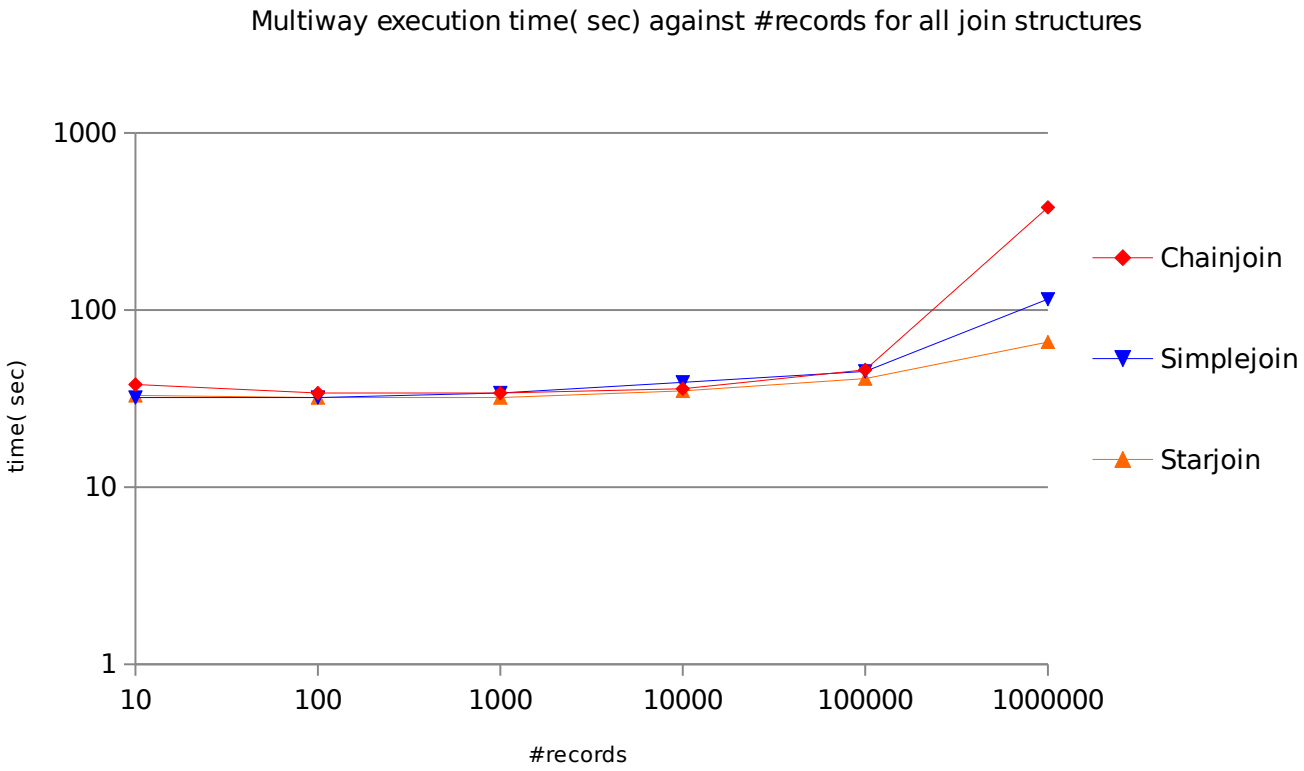


Illustration 1 : Χρόνος εκτέλεσης(sec) πολλαπλού συνδέσμου έναντι του #εγγραφών για όλες τις δομές συνδέσμων

Από το παραπάνω γράφημα συνάγεται το συμπέρασμα πως εξίσου ο λόγος εγγραφές / σχέση(από εδώ και πέρα λόγος) κι η δομή ενός συνδέσμου επιδρούν στον χρόνο εκτέλεσης του πολλαπλού συνδέσμου. Η επίδραση αυτών των παραγόντων θα φανεί στην συζήτηση για τον χρόνο εκτέλεσης των μ, ρ – διεργασιών. Επίσης, από ένα κατώφλι και πάνω η δομή συνδέσμου επιδρά περισσότερο στον χρόνο εκτέλεσης όπως φαίνεται όταν ο λόγος μεταβαίνει από 100.000 σε 1.000.000. Επιπλέον, η δομή αλυσίδα επιδρά το περισσότερο παρόλο που έχει μικρότερο όγκο δεδομένων εισόδου σύμφωνα με τον Πίνακα 5. Γιατί συμβαίνει αυτό;

Για τον σύνδεσμο αλυσίδα από τον Πίνακα 4 συμπεραίνουμε πως οι ιδιότητες A_2, A_4

κυριαρχούν. Η ιδιότητα A_3 εξαιρείται καθώς έχει πλήθος καλαθιών ίσο με 1 όταν όλες οι σχέσεις χαρακτηρίζοντας ισομεγέθεις όπως στην περίπτωση μας. Συνεχίζοντας, για έναν λόγο r, για 20 ρ – διεργασίες έχουμε $a_2 = 5$, $a_4 = 4$ καλάθια αντίστοιχα. Οπότε, 4r + 4r + 5r + 5r = 18r εγγραφές πρέπει να σταλούν σε 20 ρ – διεργασίες. Κάθε όρος στην προηγούμενη εξίσωση αντιστοιχεί σε μια σχέση του συνδέσμου αλυσίδα του Πίνακα 4, 4r → R_1, 4r → R_2, 5r → R_3, 5r → R_4.

Για τον απλό σύνδεσμο, A_1, A_4 κυριαρχούν με καλάθια $a_1 = 5$, $a_4 = 4$. Οι σταλθείσες εγγραφές στις 20 ρ – διεργασίες για τον ίδιο λόγο r, είναι, 4r + r + r + 5r = 11r.

Άρα, ενώ οι σύνδεσμοι αλυσίδα και απλός έχουν το ίδιο πλήθος εγγραφών εισόδου, 4r, στην ρ – φάση τους έχουν να επεξεργαστούν ένα διαφορετικό πληθος εγγραφών που επιδρά σημανιτκά στην απόδοση τους. Αυτό συμβαίνει εξαιτίας της δομής των συνδέσμων· ο σύνδεσμος αλυσίδα παράγει περισσότερα ενδιάμεσα αποτελέσματα και για αυτό χρειάζεται περισσότερο χρόνο εκτέλεσης.

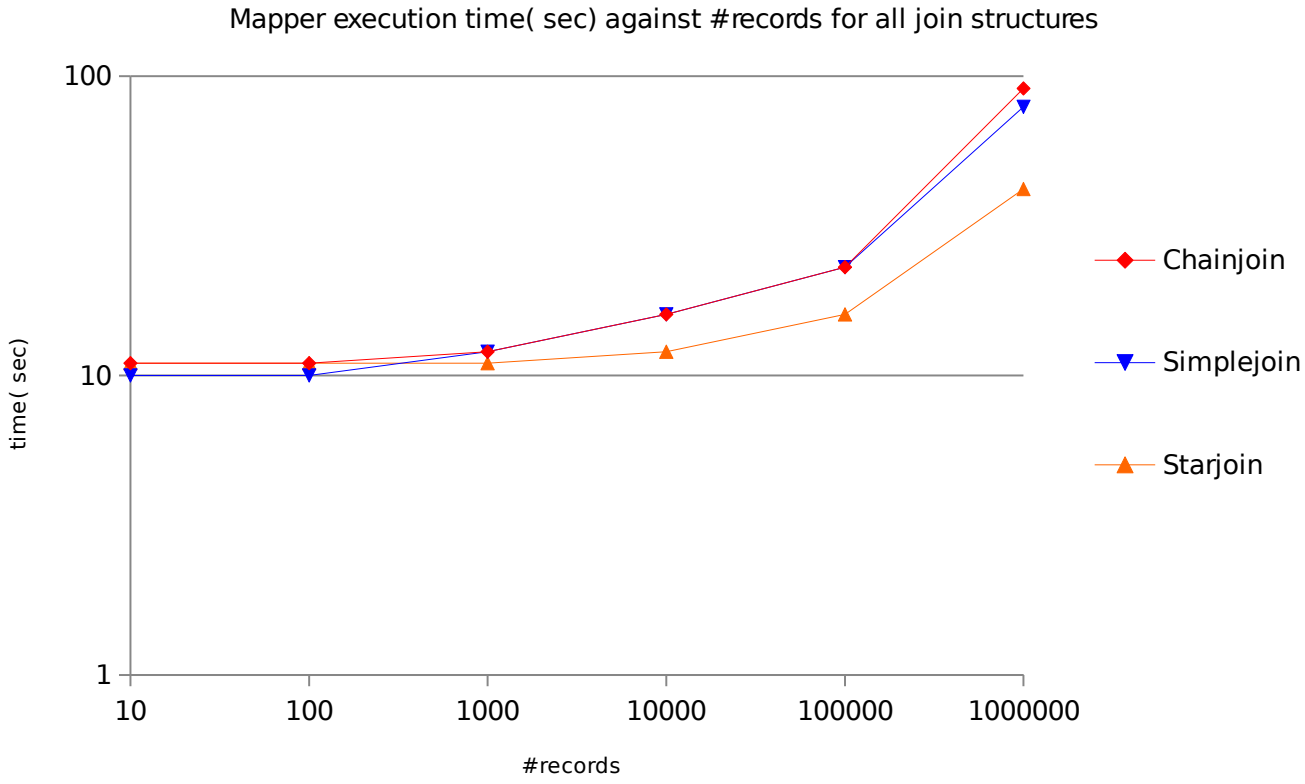


Illustration 2 : Χρόνος εκτέλεσης μ – φάσης(sec) έναντι του #εγγραφών για όλες τις δομές συνδέσμων

Στο γράφημα αυτό φαίνεται καθαρά πως η είσοδος κι η συνεπαγόμενη επεξεργασία επηρεάζουν τον χρόνο εκτέλεσης. Ο σύνδεσμος αστέρας, επεξεργάζεται πολύ λιγότερα δεδομένα εισόδου σε σχέση με τους άλλους δύο συνδέσμους και για αυτό ο χρόνος εκτέλεσης είναι πολύ λιγότερος.

Ο σύνδεσμος αλυσίδα κι ο απλός σύνδεσμος δέχονται το ίδιο ποσό δεδομένων εισόδου έχοντας ένα παραπλήσιο χρόνο εκτέλεσης όσο η δομή του συνδέσμου δεν επιδρά στην απόδοση.

Μια τελική παρατήρηση είναι πως η μ – φάση είναι σχετικά αναίσθητη στην δομή του συνδέσμου μέχρι ενός ορίου ενώ είναι πιο ευαίσθητη στον λόγο.

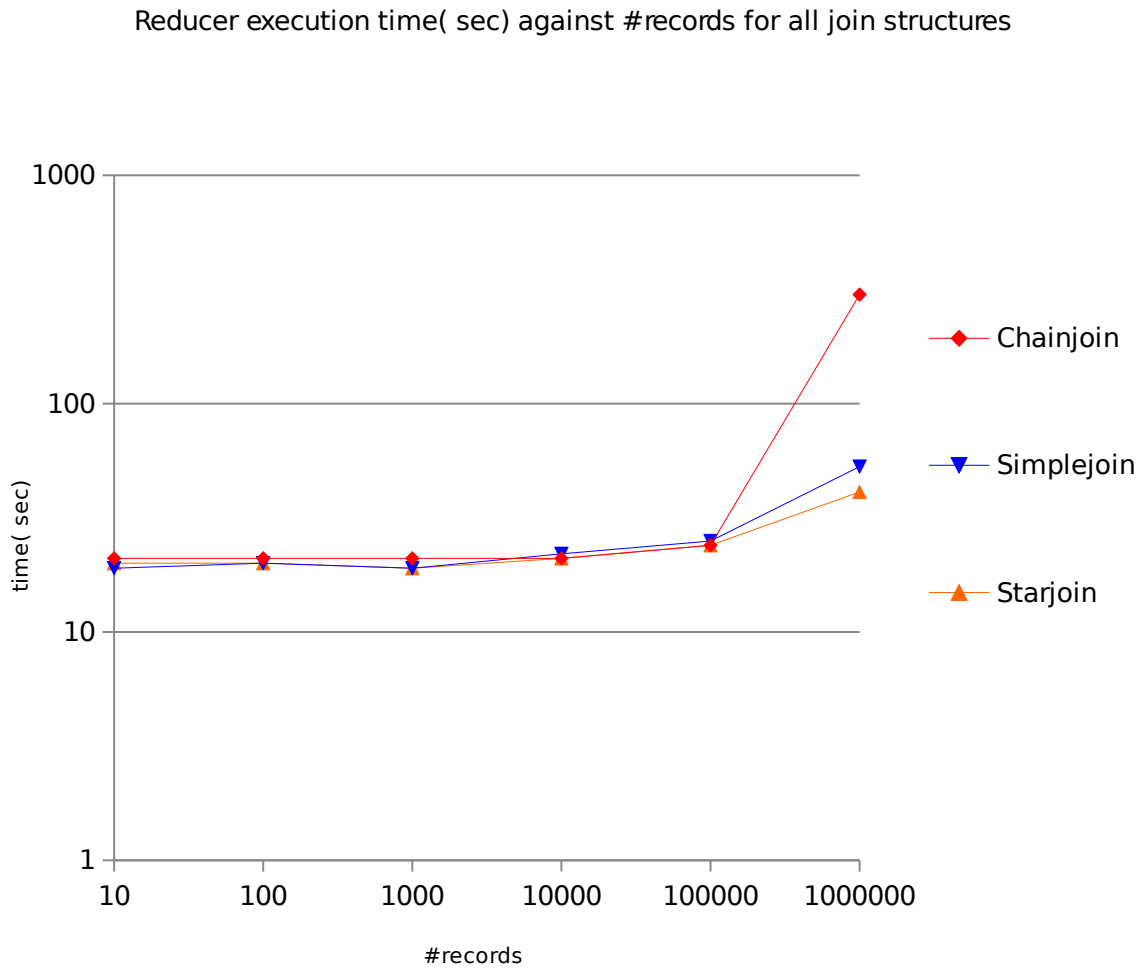


Illustration 3 : Χρόνος εκτέλεσης ρ – φάσης(sec) έναντι του #εγγραφών για όλες τις δομές συνδέσμων

Σε αυτό το γράφημα, φαίνεται πως η ρ – φάση είναι αναίσθητη στα δεδομένα εισόδου σε αντίθεση με την μ – φάση καθώς ο χρόνος εκτέλεσης δεν διαφέρει σημαντικά από σύνδεσμο σε σύνδεσμο μέχρι τις 100.000 εγγραφές / σχέση συμπεριλαμβανομένων. Ωστόσο, η ρ – φάση εξαρτάται πολύ περισσότερο από την δομή του συνδέσμου. Όταν ο λόγος ισούται με 1.000.000 ο σύνδεσμος αλυσίδα χρειάζεται πολύ περισσότερο χρόνο επεξεργασίας. Οι άλλοι δύο σύνδεσμοι επιδρούν στην ρ – φάση περίπου το ίδιο κι όχι τόσο δραστικά όσο ο σύνδεσμος αλυσίδα. Το φαινόμενο αυτό έχει ήδη εξηγηθεί στην συζήτηση για το πρώτο γράφημα(Illustration 1), και θα εξηγηθεί πιο εμπεριστατωμένα και παραστατικά στην ανάλυση των γραφημάτων του κόστους επικοινωνίας.

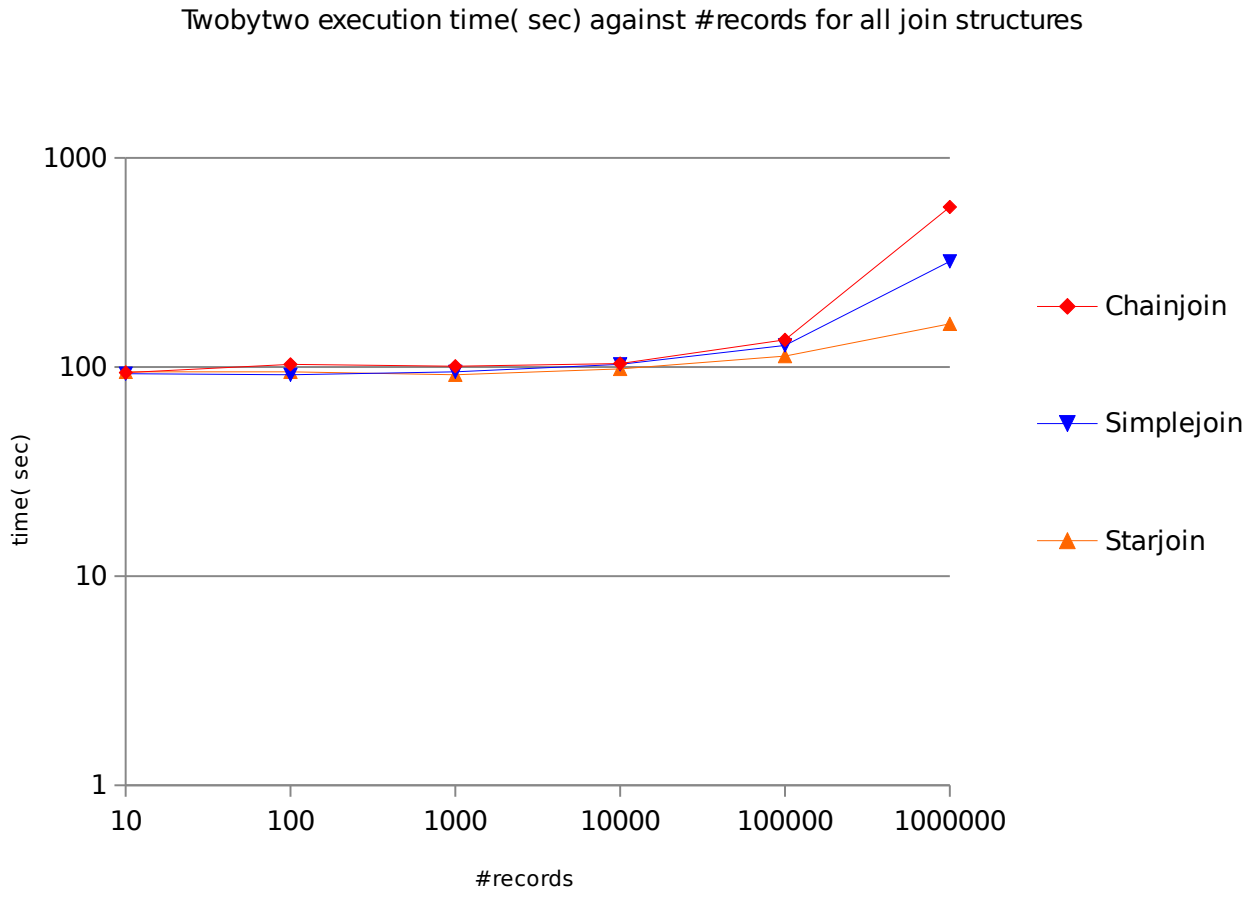
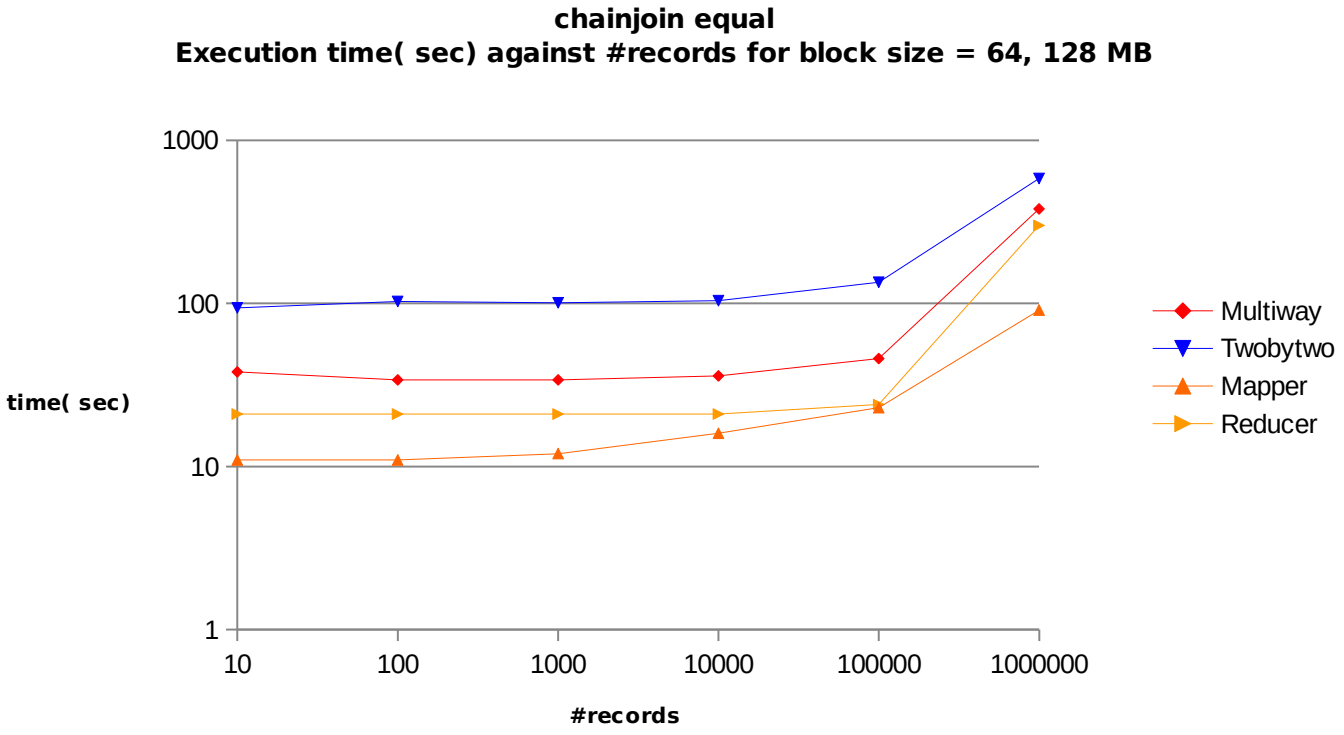


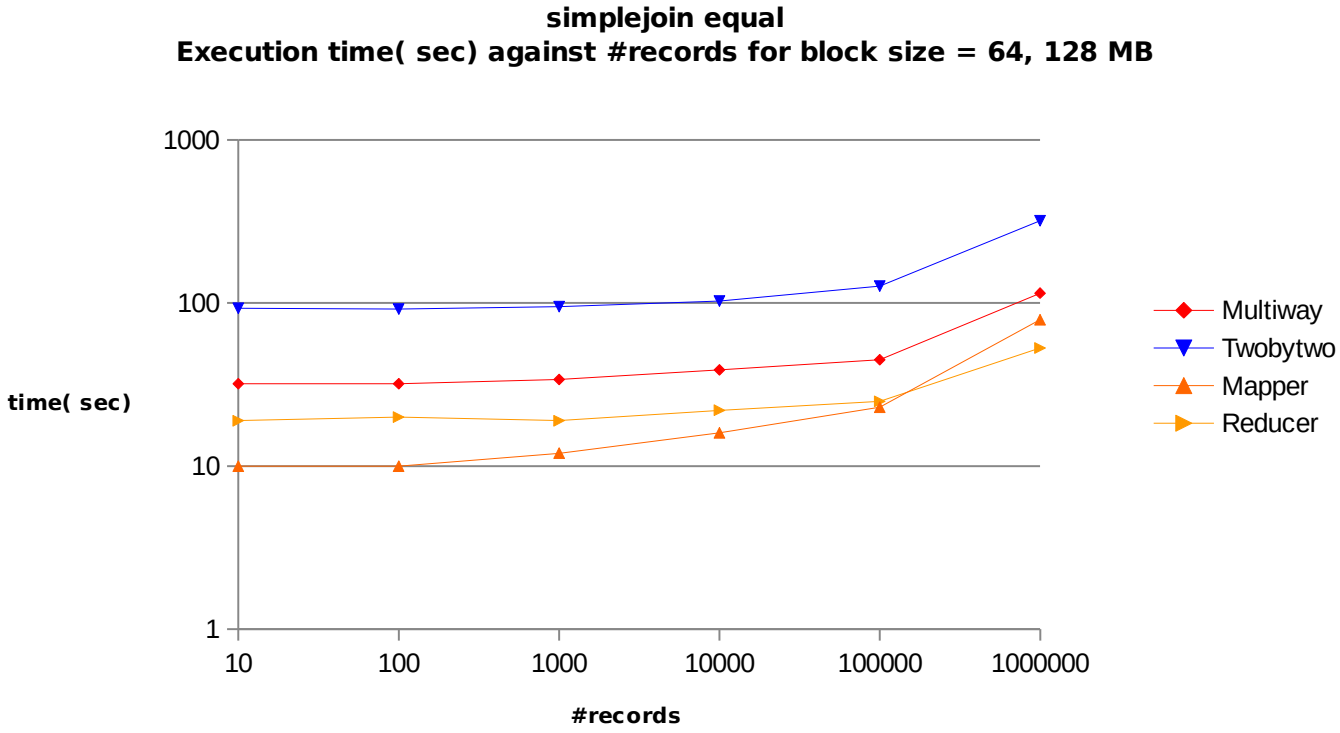
Illustration 4 Χρόνος εκτέλεσης ακολουθίας διπλών συνδέσμων(sec) έναντι του #εγγραφών για όλες τις δομές συνδέσμων.

Ξεκινώντας, η ακολουθία των διπλών συνδέσμων(ανά δύο από εδώ και στο εξής) επηρεάζεται περισσότερο από τον λόγο σε σχέση με τον πολλαπλό σύνδεσμο αφού χρειάζεται περισσότερο χρόνο να εκτελεστεί. Επηρεάζεται επίσης από την δομή του συνδέσμου αλλά όχι τόσο πολύ όσο ο πολλαπλός σύνδεσμος. Παρατηρούμε μια μικρότερη κλίση όταν ο λόγος μεταβάλλεται από το 100.000 στο 1.000.000. Επιπλέον, ξανά, ο σύνδεσμος αλυσίδα έχει την μεγαλύτερη επίδραση στην απόδοση του αλγορίθμου μολονότι έχει μικρότερο μέγεθος δεδομένων εισόδου.

Προχωρώντας στα επόμενα διαγράμματα φαίνονται οι μελετηθείσες γραφικές παραστάσεις του χρόνου εκτέλεσης συγκεντρωτικά για τους συνδέσμους αλυσίδα κα απλό. Συγκρίνοντας τον πολλαπλό σύνδεσμο με τον ανά δύο, συνάγεται εύκολα πως ο πολλαπλός σύνδεσμος είναι ταχύτερος από τον ανά δύο. Μάλιστα ο πολλαπλός σύνδεσμος είναι κατά πολύ ταχύτερος από τον ανά δύο από οτι φαίνεται όπως θα φανεί από τα γραφήματα του κόστους επικοινωνίας. Εν τέλει, η υπεροχή του πολλαπλού συνδέσμου επί του ανά δύο θα αναδειχθεί ξανά για κάθε δομή συνδέσμου ξεχωριστά.

Ωστόσο, ανάλογα με την δομή που μελετάμε παρατηρούμε πως ο χρόνος εκτέλεσης του πολλαπλού συνδέσμου είτε συνγκλίνει στην χρόνο του ανα δύο συνδέσμου όπως στον σύνδεσμο αλυσίδα είτε παρατηρούμε οι αντίστοιχοι χρόνοι εκτέλεσης να πηγαίνουν παράλληλα όπως στον απλό σύνδεσμο.





Κλείνοντας, από την σύγκριση της μ και της ρ – φάσης, η δεύτερη απαιτεί περισσότερο χρόνο για να ολοκληρωθεί σε σχέση με την πρώτη γεγονός που αναμενόταν λόγω της απαιτητικότερης επεξεργασίας που κάνει. Συνεπώς, η ρ – φάση αποτελεί το σημείο συμφόρησης του πολλαπλού συνδέσμου. Επιπροσθέτως παρατηρούμε πως η γραφική παράσταση της ρ – φάσης ταυτίζεται με την γραφική παράσταση του πολλαπλού συνδέσμου ως προς την μορφή. Αυτό το

γεγονός υποδηλώνει η ρ – φάση ορίζει την απόδοση του πολλαπλού συνδέσμου.

Ένα άλλο συμπέρασμα είναι πως ο λόγος επηρεάζει περισσότερο την μ – φάση σε σχέση με την δομή του συνδέσμου ενώ η δομή του συνδέσμου επηρεάζει περισσότερο την ρ – φάση συγκριτικά με τον λόγο.

Τελικά, στην μέχρι τώρα συζήτηση, δεν έγινε κάποια αναφορά στις ενδιάμεσες σχέσεις, αν είναι ισομεγέθεις με τις ακριανές ή αν είναι οι μισές αυτών. Τα πειράματα έδειξαν πως η μόνη διαφορά ανάμεσα στις δύο περιπτώσεις είναι πως όταν οι ενδιάμεσες σχέσεις έχουν το μισό μέγεθος των ακριανών, η απόδοση του πολλαπλού συνδέσμου δεν αλλάζει σημαντικά λόγω του κόστους επικοινωνίας ενώ στον ανά δύο αλγόριθμο η απόδοση του γίνεται σημαντικά καλύτερη. Περισσότερες λεπτομέρειες δίνονται στην ενότητα του κόστους επικοινωνίας.

5.4.2.2 Κόστος επικοινωνίας

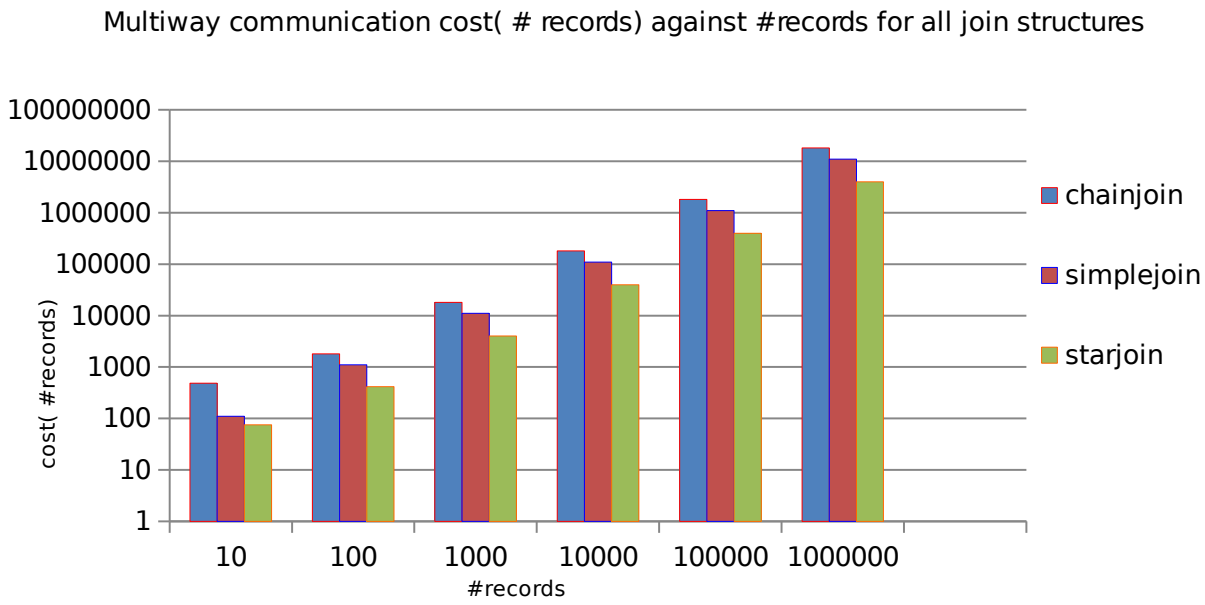


Illustration 5: Κόστος επικοινωνίας(#εγγραφών) πολλαπλού συνδέσμου για όλους τους συνδέσμους

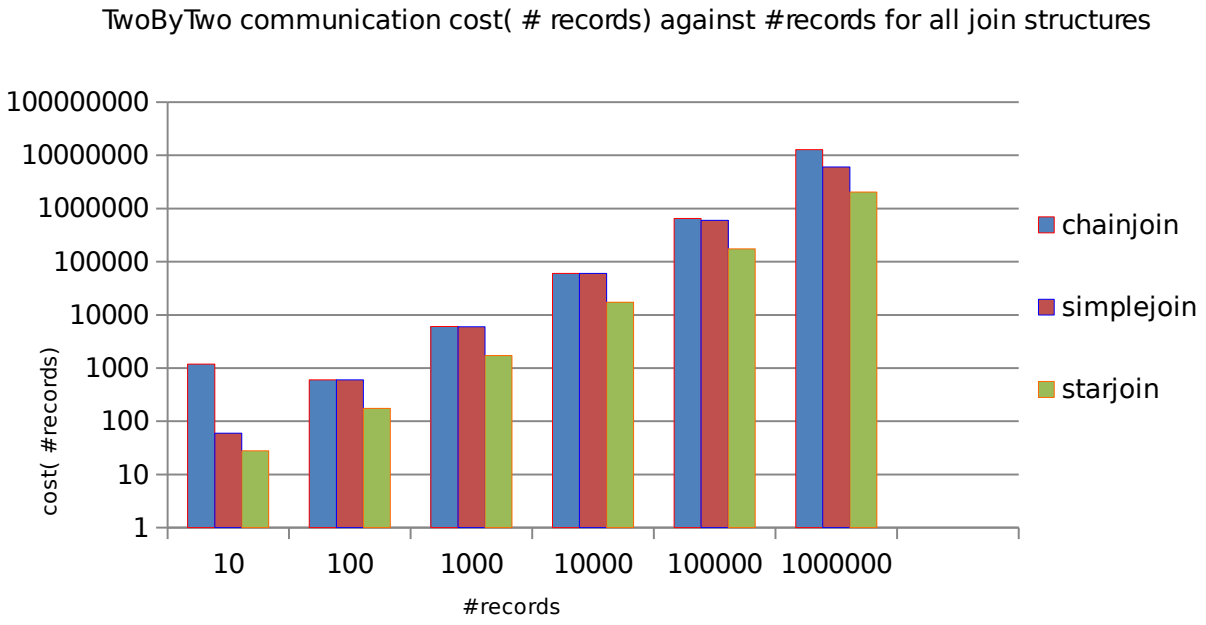


Illustration 6: Κόστος επικοινωνίας(#εγγραφών) ανά δύο συνδέσμου για όλους τους συνδέσμους

Τα δύο τελευταία γραφήματα επιβεβαιώνουν οτι έχει ήδη ειπωθεί. Ο σύνδεσμος αλυσίδα όντας ο πιο απαιτητικός οδηγεί σε μεγαλύτερους χρόνους εκτέλεσης.

Συγκρίνοντας τα δύο τελευταία γραφήματα, κάτι ενδιαφέρον αναδεικνύεται. Τί είναι αυτό;

Οι στήλες στο γράφημα του ανά δύο συνδέσμου(Illustration 6) βρίσκονται κάτω από την γραμμή. Για παράδειγμα, οι στήλες για 10.000 εγγραφές βρίσκονται κάτω από την γραμμή των 10.000 στον κάθετο άξονα. Οι στήλες στο γράφημα του πολλαπλού συνδέσμου(Illustration 5) κείνται άνω της γραμμής. Για το ίδιο παράδειγμα των 10.000 εγγραφών, οι στήλες κείνται άνω την γραμμής των 10.000 εγγραφών στον κάθετο άξονα.

Τί σημαίνει αυτό;

Σημαίνει πολλά πράγματα γιατί τα γραφήματα βρίσκονται σε *λογαριθμική κλίμακα* κι άρα το πάνω και το κάτω από την γραμμή, αφορούν σε μια εκθετική μεταβολή. Επομένως, μια φαινομενικά μικρή, λογαριθμική διαφορά είναι μεγαλύτερη στην πραγματικότητα. Επανερχόμενοι, το "πάνω και κάτω από την γραμμή" πρώτον, εξηγεί γιατί η δομή των συνδέσμων επιδρά περισσότερο στον πολλαπλό σύνδεσμο παρά στον ανά δύο σύνδεσμο. Η μ – φάση εξαρτώμενη από την δομή παράγει περισσότερα ενδιάμεσα αποτελέσματα και συνεπώς η ρ – φάση χρειάζεται περισσότερο χρόνο να εκτελεστεί. Ο ανά δύο σύνδεσμος από φάση σε φάση παράγει πολύ λιγότερα ενδιάμεσα αποτελέσματα και για αυτό επηρεάζεται λιγότερο από την δομή του συνδέσμου. Ο σύνδεσμος αλυσίδα έχει την μεγαλύτερη επιρροή γιατί πυροδοτεί την μεγαλύτερη παραγωγή ενδιάμεσων εγγραφών.

Συνεχίζοντας, σημαίνει πως ο πολλαπλός σύνδεσμος είναι πιο απαιτητικός από τον ανά δύο. Με άλλα λόγια ο πολλαπλός σύνδεσμος παράγει το ίδιο αποτέλεσμα με τον ανά δύο αλλά επεξεργάζεται πολλές περισσότερες εγγραφές που ο ίδιος παράγει. Αυτή η μαζική παραγωγή νέων, ενδιάμεσων εγγραφών μπορεί να προκαλέσει προβλήματα μνήμης ειδικά σε περιβάλλοντα με περιορισμένη μνήμη. Στην διάρκεια των πειραμάτων προβλήματα μνήμης αντιμετωπίστηκαν αρκετές φορές. Αναλυτικά, 200 MB μνήμης RAM ήταν αρκετά για τον ανά δύο σύνδεσμο όταν ό λόγος ήταν 1.000.000. Για τον ίδιο λόγο ο πολλαπλός σύνδεσμος χρειαζόταν 1GB μνήμης για να εκτελεστεί παρά τις όποιες βελτιώσεις στον κώδικα ως προς την χρήση της μνήμης.

Μια άλλη συνέπεια της μαζικής παραγωγής εγγραφών είναι η ανάγκη μιας εκλεπτυσμένης υλοποίησης του πολλαπλού συνδέσμου για δύο λόγους. Πρώτον, έχοντας γράψει τους αλγορίθμους σε Java, δύο συχνά σφάλματα σχετικά με την μνήμη ήταν java.lang.OutOfMemoryError: GC overhead limit exceeded και java.lang.OutOfMemoryError: Java heap space. Το πρώτο σφάλμα οφείλεται στην μεγάλη παραγωγή προσωρινών αντικειμένων με μικρό χρόνο ζωής. Η αποκομιδή τους ωστόσο καθίσταται ασύμφορη από ένα σημείο και μετά, αφού απαιτεί πολύ περισσότερο χρόνο από την κύρια εργασία. Το σφάλμα αυτό ξεπεράστηκε με την χρήση StringBuilder και με την αντικατάσταση συναρτήσεων – όπως οι String.split() που δημιουργούν νέα ενδιάμεσα αποτελέσματα μέρος των οποίων χρειάζονται εν τέλει – με απλούστερο κώδικα Java που είναι μεν πιο πολύς σε ποσότητα αλλά καλύτερος από την άποψη της μνήμης.

Σχετικά με το δεύτερο σφάλμα, κι αυτό οφείλεται στην παραγωγή ενδιάμεσων, προσωρινών αντικειμένων κατά την ρ – φάση. Το πρόβλημα αυτό υπερπηδήθηκε εν μέρει μέσα από την επίλυση του πρόβληματος με τον garbage collector, καθώς και με την αύξηση του μεγέθους του σωρού.

Δεύτερον, οι απαιτήσεις σε μνήμη κι ενδιάμεσα αποτελέσματα κατά την ρ – φάση, καθιστούν τον πολλαπλό σύνδεσμο εν δυνάμει πολύ. Αργό. Στην διάρκεια των πειραμάτων η υλοποίηση του πολλαπλού συνδέσμου άλλαξε 11 φορές για να επιτευχθεί μια πολύ καλή απόδοση αυτού για αυθαίρετο μέγεθος των δομών των συνδέσμων με τα μέχρι τώρα πειραματικά αποτελέσματα. Ειδικότερα, ο ανά δύο αλγόριθμος έχει υλοποιηθεί εκμεταλλευόμενοι τον αλγόριθμο του Hadoop(reduce side join) που αποδίδει σχετικά καλά. Η χρήση του ίδιου αλγορίθμου για τον πολλαπλό σύνδεσμο ήταν πολύ κακή. Αναλυτικότερα, ο αλγόριθμος του Hadoop χρειαζόταν ώρες για να εκτελεστεί ως πολλαπλός σύνδεσμος και για είσοδο 10.000 εγγραφών. Με την τελευταία υλοποίηση του πολλαπλού συνδέσμου, η περίπτωση των *100.000* εκτελείται *σε* 37 δευτερόλεπτα. Αυτά τα 37 δευτερόλεπτα αφορούν μόνο στον χρόνο εκτέλεσης των μ και ρ φάσεων.

Συνεχίζοντας, η εκλεπτυσμένη υλοποίηση του πολλαπλού συνδέσμου έρχεται με κάποιο κόστος., Τα προβλήματα με την ανάγκη μιας εκλεπτυσμένης υλοποίησης του πολλαπλού

συνδέσμου είναι η αδυναμία εκμετάλλευσης ήδη καλά μελετημένων αλγορίθμων όπως αυτός του Hadoop κι η χρήση απλών αλγορίθμων γενικότερα με όλα τα οφέλη που συνεπάγονται. Επίσης το σύστημα γίνεται πιο δύσκολο να συντηρηθεί και να επεκταθεί.

Προχωρώντας, το γεγονός πως ο πολλαπλός σύνδεσμος είναι πιο απαιτητικός, προκαλεί επίσης δικτυακά προβλήματα γιατί, το φορτίο επί του δικτύου μεγαλώνει. Σχηματικά και για την περίπτωση του απλού συνδέσμου,

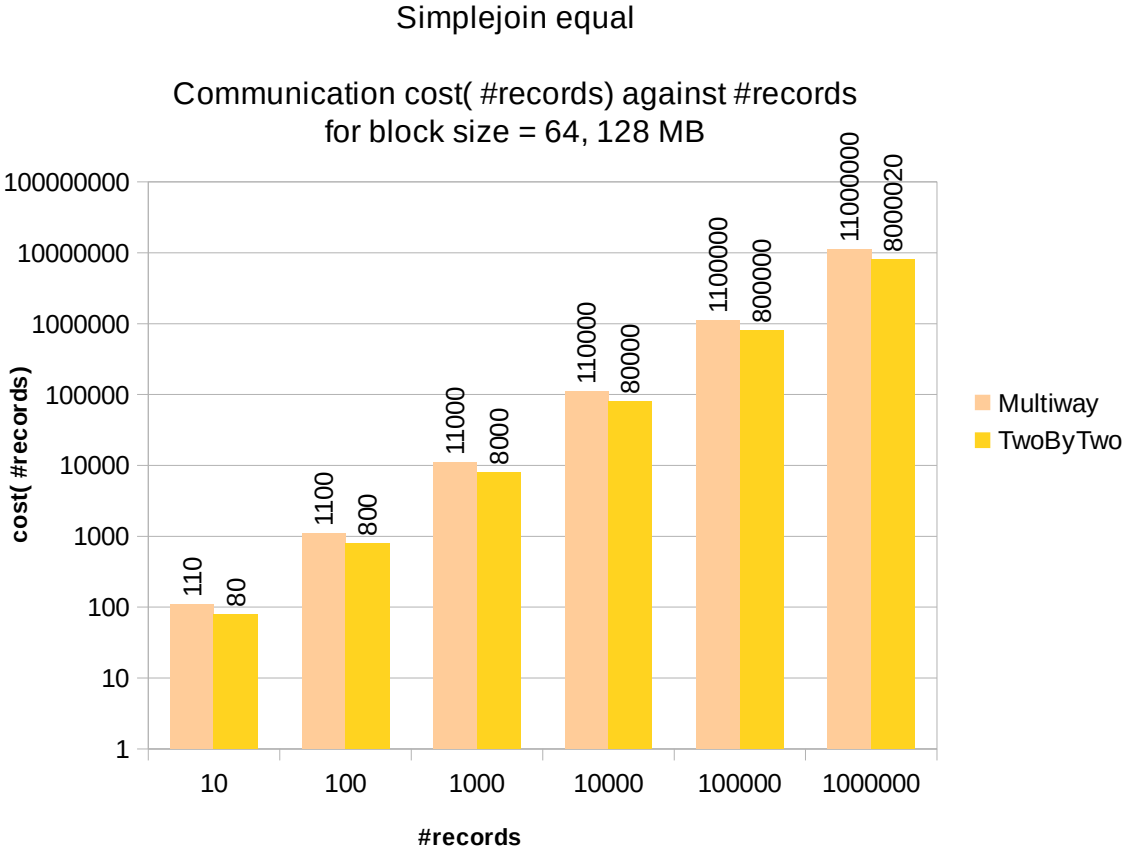


Illustration 7: Κόστος επικοινωνίας απλού συνδέσμου για τους πολλαπλό και ανά δύο συνδέσμους όταν οι ενδιάμεσες σχέσεις χαρακτηρίζονται ισομεγέθεις με τις ακραίες.

όπου οι αναγραφόμενοι αριθμοί αναφέρονται στο συνολικό κόστος επικοινωνίας. Στην περίπτωση του ανά δύο συνδέσμου οι 8.000 εγγραφές για παράδειγμα ισούνται με το συνολικό κόστος επικοινωνίας και των τριών φάσεων. Επίσης, η λέξη “equal” στον τίτλο του γραφήματος υποδηλώνει πως όλες οι σχέσεις του συνδέσμου έχουν το ίδιο μέγεθος.

Ξεκάθαρα, ο ανά δύο αλγόριθμος σύνδεσης προτιμάται από την σκοπιά του δικτύου. Παρόλο που φαίνεται ο ανά δύο αλγόριθμος να επιβαρύνει το δίκτυο λόγω των πολλών του φάσεων, είναι καλύτερος από τον πολλαπλό σύνδεσμο λαμβάνοντας υπό όψιν το επικοινωνιακό κόστος στο προηγούμενο γράφημα. Ακόμη, ο ανά δύο σύνδεσμος υπερέχει γιατί το κόστος επικοινωνίας του διανέμεται πιο ομοιόμορφα επί του δικτύου. Από την άλλη ο πολλαπλός σύνδεσμος έχει ένα κόστος επικοινωνίας που μοιάζει με ριπή, πάρα πολλά δεδομένα σε σύντομο χρόνο, και για αυτό απαιτεί ειδική μέριμνα. Ποσοτικότερα, $\frac{\textit{κόστος επικοινωνίας πολλαπλού συνδέσμου}}{\textit{κόστος επικοινωνίας ανά δύο συνδέσμου}} = \frac{110}{80} = 1.375$ που σημαίνει πως το κόστος

επικοινωνίας του πολλαπλού συνδέσμου είναι σχεδόν 40% μεγαλύτερο από το κόστος του ανά δύο.

Οτι ελέχθη μέχρι τώρα για το κόστος επικοινωνίας σε σχέση με το δίκτυο χειροτερεύει όταν οι ενδιάμεσες σχέσεις μειώνονται κατά μισό συγκριτικά με τις ακριανές σχέσεις του συνδέσμου. Αυτό λέγεται με την έννοια πως ο ανά δύο σύνδεσμος γίνεται ακόμα καλύτερος για το δίκτυο. Εδώ,

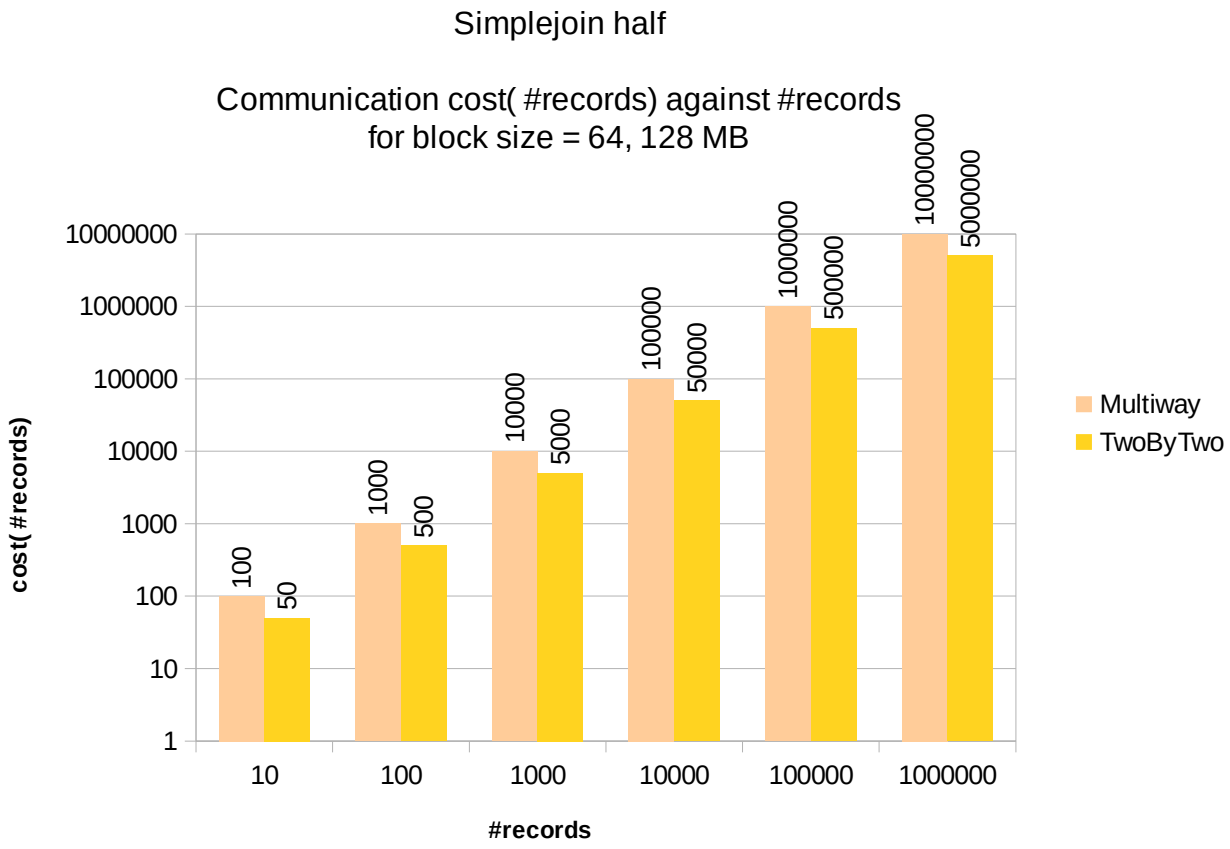


Illustration 8 Κόστος επικοινωνίας απλού συνδέσμου για τους πολλαπλό και ανά δύο συνδέσμους όταν οι ενδιάμεσες σχέσεις έχουν το μισό μέγεθος από τις ακραίες.

$\frac{\textit{κόστος επικοινωνίας πολλαπλού συνδέσμου}}{\textit{κόστος επικοινωνίας ανά δύο συνδέσμου}} = \frac{100}{50} = 2$ που σημαίνει πως ο ανα δύο σύνδεσμος έχει το μισό κόστος από τον πολλαπλο σύνδεσμο. Μάλιστα συγκρίνοντας τις δύο εκδοχές $\frac{\textit{κόστος επικοινωνίας ανά δύο συνδέσμου , μισές σχέσεις}}{\textit{κόστος επικοινωνίας ανά δύο συνδέσμου , ίσες σχέσεις}} = \frac{500}{800} = 62{,}5\,\%$, Για αυτό έχει και μικρότερο χρόνο εκτέλεσης. Το κόστος επικοινωνίας του πολλαπλού συνδέσμου παράμενει σχεδόν το ίδιο και είναι ανεξάρτητα από το μέγεθος των ενδιάμεσων σχέσεων ως ένα βαθμό. Αυτό συμβαίνει, γιατί η μείωση των ενδιάμεσων σχέσεων αντισταθμίζεται από τον πολλαπλασιασμό των εγγραφών κατά την μ – φάση. Συνεπώς κι ο χρόνος εκτέλεσης δεν διαφέρει σημαντικά.

Τώρα ήρθε η ώρα να εξηγηθεί, γιατί η μ – φάση επηρεάζεται περισσότερο από τον λόγο εγγραφές / σχέση σε σχέση με την ρ – φάση. Ο λόγος είναι πως στην διάρκεια της μ – φάσης, η μαζική παραγωγή νέων εγγραφών λαμβάνει χώρα. Συνεπώς, η μ – φάση εργάζεται σε ένα μεγάλο σύνολο εγγραφών ενώ η ρ – φάση λειτουργεί σε ένα μέρος αυτού του μεγάλου συνόλου και για αυτό επηρεάζεται λιγότερο από τον λόγο. Αναλυτικά, κάποιες ρ – διεργασίες, δεν εργάζονται καθόλου, με την έννοια πως αν δεν λάβουν εγγραφές από όλες τις σχέσεις, τότε σταματούν.

Κάποιες άλλες ρ – διεργασίες, τελικά εργάζονται επί ενός μέρους της εισόδου τους ύστερα από κάποια επεξεργασία αυτής. Συμπερασματικά, η ρ – φάση έχει μεγαλύτερο κόστος λόγω των υπολογισμών που κάνει κι όχι του πλήθους των εγγραφών. Η μ – φάση έχει μεγαλερο κόστος λόγω του πλήθους των εγγραφών που διαχειρίζεται κι όχι τνω υπολογισμών που κάνει.

Πηγαίνοντας παραπέρα, έχει ήδη αποδειχθεί πως ο πολλαπλός σύνδεσμος είναι πιο γρήγορος από τον ανα δύο. Αν κάποιος λάβει υπό όψιν του πως ο πολλαπλός σύνδεσμος διαχειρίζεται ένα μεγαλύτερο φορτίο από τον ανά δύο, τότε αναπόφευκτα μπορεί να καταλήξει πως ο πολλαπλός σύνδεσμος είναι πολύ γρηγορότερος από ότι αρχικά δείχνει. Κατ' επέκταση, έχοντας κατά νου πως ο ανά δύο αλγόριθμος χειρίζεται καρτεσιανό γινόμενο, ενώ ο πολλαπλός σύνδεσμος χειρίζεται κατακερματισμό, καταλήγουμε πως ο κατακερματισμός είναι μεν ταχύτερος αλλά δαπανητότερος ως προς την μνήμη.

5.4.2.3 Πρώτη πειραματική σύνοψη και λύσεις

Πολλαπλός σύνδεσμος	Ανά δύο σύνδεσμος
Πολύ γρήγορος	Μέτρια γρήγορος
Μεγάλο κόστος επικοινωνίας	Μικρό κόστος επικοινωνίας
Αμετάβλητο κόστος επικοινωνίας ακόμα κι αν οι ενδίαμεσες σχέσεις μειωθούν στο μισό.	Το κόστος επικοινωνίας μειώνεται καθώς οι ενδιάμεσες σχέσεις μειώνονται
Επιβαρύνει το δίκτυο	Επιβαρύνει πολύ λιγότερο το δίκτυο
Εκλεπτυσμένη αλγοριθμική υλοποίηση	Απλή αλγοριθμική υλοποίηση
Προβλήματα μνήμης ακόμα και για 56 MB δομές συνδέσμων κι άρα μη ανεκτός σε περιβάλλοντα περιορισμένης μνήμης.	Μη ορατά προβλήματα μνήμης ακόμα και για 7,7 GB δομές δεδομένων

Πολλαπλός σύνδεσμος	
Πρόβλημα	**Εν δυνάμει λύση**
Μεγάλο κόστος επικοινωνίας	Χρήση συνδυάσμωνα(combiner) αφού ένα μέρος της εξόδου της μ – φάσης είναι άχρηστο. Συμπίεση εξόδου μ - φάσης
Αμετάβλητο κόστος επικοινωνίας ακόμα κι αν οι ενδίαμεσες σχέσεις μειωθούν στο μισό.	
Επιβαρύνει το δίκτυο	
Εκλεπτυσμένη αλγοριθμική υλοποίηση	Αναζήτηση απλούστερης υλοποίησης
Προβλήματα μνήμης ακόμα και για 56 MB δομές συνδέσμων κι άρα μη ανεκτός σε περιβάλλοντα περιορισμένης μνήμης.	Εκλέπτυνση τρέχοντος αλγορίθμου και κώδικα. Μεταφορά του υπολογισμού του συνδέσμου στις μ – διεργασίες π.χ. οι μ – διεργασίες να στέλνουν ένα μέρος της εξόδου τους στον εαυτό τους και να λειτουργούν ως ρ – διεργασίες για να μειωθεί η μαζική παραγωγή νέων εγγραφών. Αύξηση των ρ – διεργασιών ώστε να μειωθεί ο λόγος φορτίο / ρ – διεργασία.

5.4.3 Πειράματα έναντι των επεξεργαστών(cpus) και των ρ – διεργασιών

Σε αυτά τα πειράματα, εξετάζεται η σχέση του χρόνου εκτέλεσης και του κόστους επικοινωνίας σε σχέση με το πλήθος επεξεργαστών και των ρ – διεργασιών ενώ η είσοδος μένει σταθερή στις 1.000.000 εγγραφές / σχέση.

Το πλήθος των ρ – διεργασιών μεταβάλλεται από 20 έως 100. Το πειραματικό πλήθος των ρ – διεργασιών που φαίνεται παρακάτω, κάποιες φορές αποτελεί μια προδιαγραφή και κάποιες φορές αποτελεί ένας ακριβής αριθμός.

Δομές συνδέσμων	P - διεργασίες	
	Προδιαγραφή	Πραγματική τιμή
Σύνδεσμοι αλυσίδα κι απλός	20	20
	40	42
	60	56
	80	81
	100	100
Σύνδεσμος αστέρας	20	18
	40	36
	60	63
	80	80
	100	100

Το πλήθος των επεξεργαστών μεταβάλλεται από 24 έως και 64.

5.4.4 Χρόνος εκτέλεσης

5.4.4.1 Χρόνος εκτέλεσης σε σχέση με τις ρ – διεργασίες όταν οι επεξεργαστές = 64

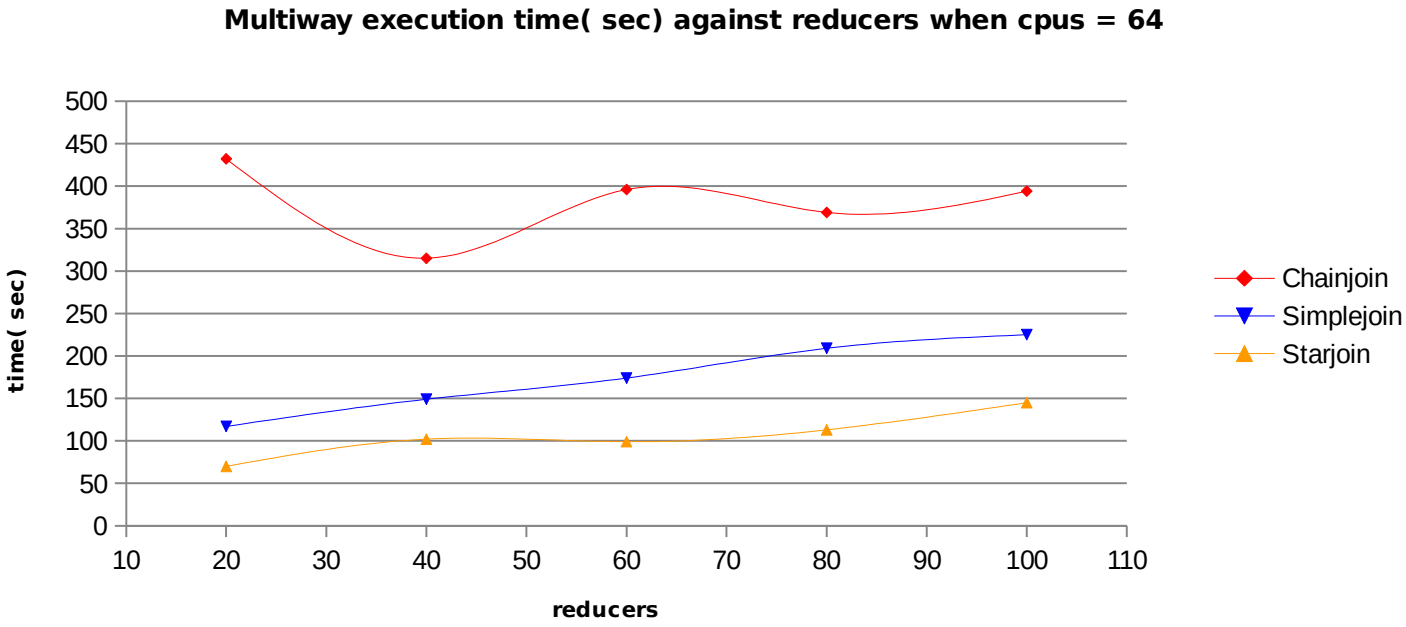


Illustration 9: Χρόνος εκτέλεσης του πολλαπλού συνδέσμου(sec) σε σχέση με τις ρ - διεργασίες

Παρατηρούμε δύο διαφορετικές σχέσεις ανάμεσα στον χρόνο εκτέλεσης και στο πλήθος των ρ – διεργασιών ανάλογα με την δομή του συνδέσμου. Για τον απλό σύνδεσμο και τον σύνδεσμο αστέρα η σχέση είναι σχετικά γραμμική. Επιπλέον, καθώς οι ρ – διεργασίες αυξάνονται, ο χρόνος εκτέλεσης αυξάνεται αναλόγως. Όπως όμως δείχνει η γραφική παράσταση του απλού συνδέσμου ενδέχεται να υφίσταται κορεσμός. Ο χρόνος εκτέλεσης, δηλαδή, να μην εξαρτάται από τις ρ – διεργασίες.

Επίσης παρατηρούμε πως ο χρόνος εκτέλεσης αυξάνεται με την αύξηση των ρ – διεργασιών λόγω του κόστους επικοινωνίας που συνεπάγονται ενώ κάποιος θα μπορούσε να περιμένει να μειωθεί λόγω του παραλληλισμού που επιφέρουν οι παραπάνω ρ – διεργασίες.

Για τον σύνδεσμο αλυσίδα η εξεταζόμενη σχέση μπορεί να χαρακτηριστεί ως αποσβενύμμενη ημιτονοειδής που συγκλίνει γύρω από τα 400 δευτερόλεπτα. Ξανά, η αύξηση στις ρ – διεργασίες έχει μια περιορισμένη επίδραση στον χρόνο εκτέλεσης. Μέχρι το κατώφλι των 100 ρ – διεργασιών, η αύξηση του πλήθους των ρ – διεργασιών είτε βελτιώνει είτε χειροτερεύει τον χρόνο εκτέλεσης. Πέρα από τις 100 ρ – διεργασίες η επιρροή του πλήθους τους είναι σχεδόν ασήμαντη.

Στην γραφική παράσταση του συνδέσμου αλυσίδα διαπιστώνουμε την δράση δύο αντίθετων δυνάμεων. Η μία συνίσταται στο κόστος επικονωνίας που τείνει να αυξήσει τον χρόνο εκτέλεσης. Η άλλη αφορά στο φορτίο ανα ρ – διεργασία που τείνει να μειώσει τον χρόνο εκτέλεσης, αφού αυτό όσο μικρότερο είναι τόσο πιο γρήγορα γίνεται η επεξεργασία του. Καθώς οι ρ – διεργασίες μεταβάλλονται αλλάζει κι ο συσχετισμός των δυνάμεων αυτών βλέποντας διαφορετκούς χρόνους εκτέλεσης. Όταν οι δυνάμεις αυτές ισορροπούν, τότε φτάνουμε στο σημείου κόρου. Επιπρόσθετα, παρατηρούμε πως για κάποιο πλήθος ρ – διεργασιών κάθε μία από αυτές τις δυνάμεις έχει την

μέγιστη επιρροή της. Για το κόστος επικοινωνίας αυτό συμβαίνει για 20 ρ – διεργασίες ενώ για το φορτίο / ρ – διεργασία συμβαίνει για 40 ρ – διεργασίες. Βέβαια τα σημεία αυτά γενικά πρέπει να υπάρχουν αλλά η τιμή τους πρέπει να είναι αντίστοιχη των προδιαγραφών του συμπλέγματος των υπολογιστών.

Ως ένα γενικό συμπέρασμα, η επιρροή της αύξησης των ρ – διεργασιών στον χρόνο εκτέλεσης εξαρτάται από την δομή του συνδέσμου. Ως γενικός κανόνας ωστόσο, μια αύξηση των ρ – διεργασιών ωφελεί τον σύνδεσμο αλυσίδα αλλά ζημιώνει τον απλό σύνδεσμο και τον σύνδεσμο αστέρα. Συνεπώς, η απόδοση του συνδέσμου αλυσίδα μπορεί να βελτιωθεί ρυθμίζοντας απλά το Hadoop χωρίς την ανάγκη για περισσότερο ή καλύτερο υλικό.

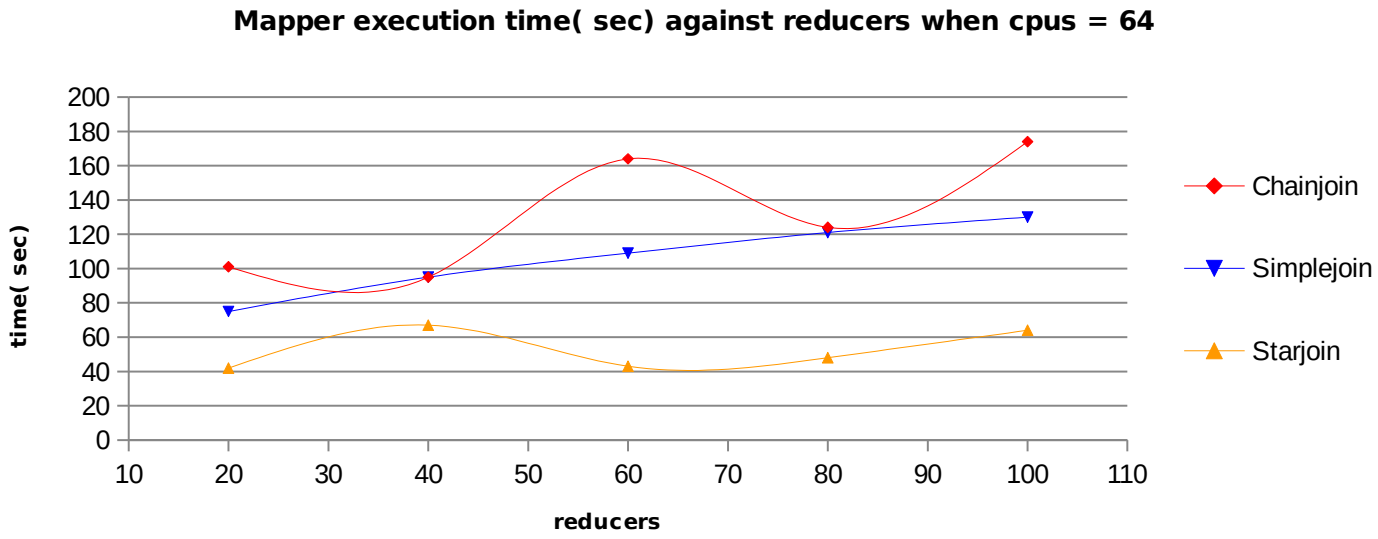


Illustration 10: Χρόνος εκτέλεσης μ – φάσης(sec) σε σχέση με τις ρ - διεργασίες

Εδώ, δύο διαφορετικά πρότυπα εμφανίζονται ξανά. Ο χρόνος εκτέλεσης της μ – φάσης του απλού συνδέσμου έχει την ίδια γραμμική σχέση με τις ρ – διεργασίες όπως ο αντίστοιχος πολλαπλός σύνδεσμος.

Οι σύνδεσμοι αλυσίδα και αστέρας έχουν χρόνο εκτέλεσης μ – φάσης που μεταβάλλεται ημιτονοειδώς σε σχέση με το πλήθος των ρ – διεργασιών. Ειδικότερα, και ο σύνδεσμος αστέρας και ο σύνδεσμος αλυσίδα ακολουθούν ένα αύξον ημιτονοειδές πρότυπο χωρίς κάποιο σημείο κόρου. Στον σύνδεσμο αλυσίδα οι μεταβολές χαρακτηρίζονται προφανώς εντονότερες.

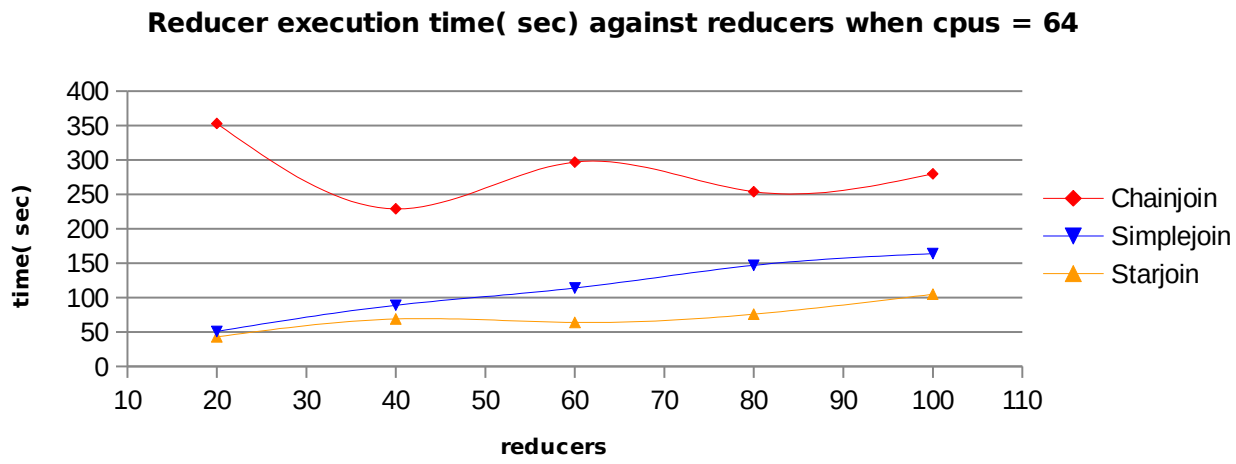


Illustration 11: Χρόνος εκτέλεσης ρ – φάσης(sec) σε σχέση με τις ρ - διεργασίες

Από την τελευταία γραφική παράσταση μπορούμε να σημειώσουμε πως ταυτίζεται με αυτή του πολλαπλού συνδέσμου(Illustration 9) με την διαφορά πως οι χρόνοι εκτέλεσης είναι μικρότεροι. Για αυτό συμπεραίνουμε πως η ρ – φάση υποδεικνύει την απόδοση του αλγορίθμου και πάλι.

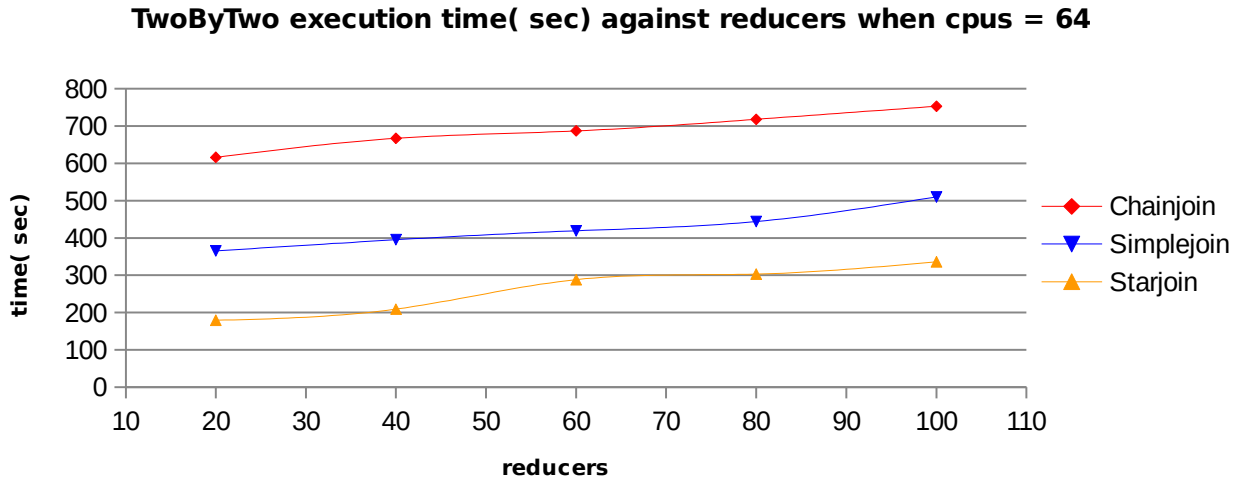


Illustration 12: Χρόνος εκτέλεσης του ανά δύο συνδέσμου(sec) σε σχέση με τις ρ - διεργασίες

Σε αυτή την περίπτωση ξεκάθαρα, βλέπουμε ο χρόνος εκτέλεσης να σχετίζονται γραμμικά με τις ρ – διεργασίες ανεξάρτητα της δομής του συνδέσμου. Η κλίση αυτής της σχέσης χαρακτηρίζεται σχετικά μικρή που σημαίνει πως ο ανά δύο σύνδεσμος είναι σχετικά ανεξάρτητος από την ποσότητα των ρ – διεργασιών.

5.4.4.2 Χρόνος εκτέλεσης σε σχέση με τις ρ – διεργασίες και τους επεξεργαστές

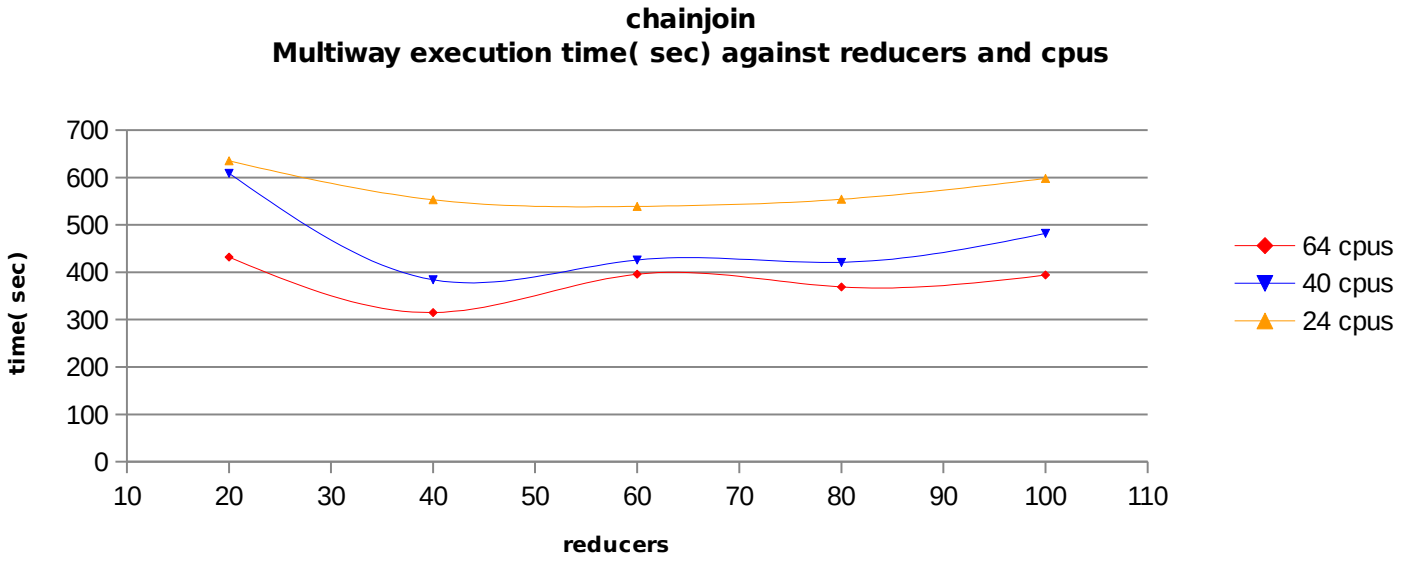


Illustration 13: Χρόνος εκτέλεσης πολλαπλού συνδέσμου(sec) σε σχέση με την ποσότητα των επεξεργαστών και των ρ – διεργασιών για τον σύνδεσμο αλυσίδα.

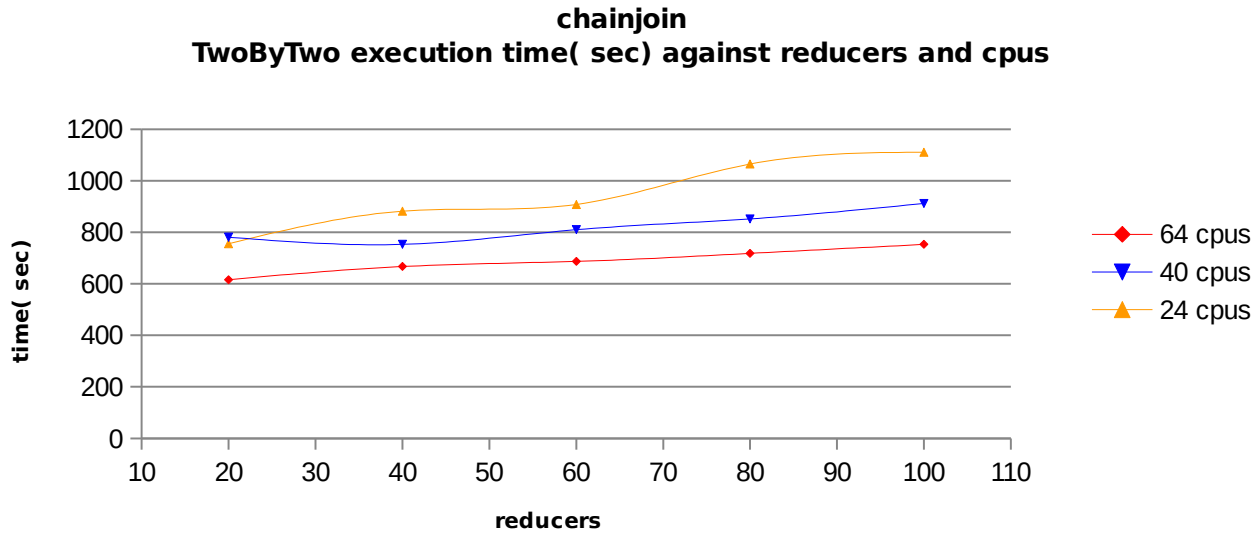


Illustration 14: Χρόνος εκτέλεσης του ανά δύο συνδέσμου(sec) σε σχέση με την ποσότητα των επεξεργαστών και των ρ – διεργασιών για τον σύνδεσμο αλυσίδα.

Από τις δύο τελευταίες απεικονίσεις προκύπτει κατηγορηματικά πως η αύξηση των επεξεργαστών βελτιώνει τον χρόνο εκτέλεσης. Ωστόσο αυτή η ευεργετική επίδραση έχει κάποια όρια. Η βελτίωση στον χρόνο εκτέλεσης από τους 24 στους 40 επεξεργστές είναι σημαντικά μεγαλύτερη από τους 40 στους 64 επεξεργαστές στον γράφημα του πολλαπλού συνδέσμου.

Το πλήθος των επεξεργαστών, επίσης επηρεάζει την επίδραση των ρ – διεργασιών και στους δύο αλγορίθμους σύνδεσης. Για παράδειγμα, για τον ανά δύο σύνδεσμο όταν οι ρ – διεργασίες αυξάνονται από 20 σε 40, για 40 επεξεργαστές υφίσταται μια βελτίωση στον χρόνο εκτέλεσης. Ωστόσο, για 24 και 64 επεξεργαστές συμβαίνει μια χειροτέρευση. Για τον πολλαπλό σύνδεσμο το ίδιο συμβαίνει όταν οι ρ – διεργασίες αυξάνουν από 40 σε 60 , όπου για 24 επεξεργαστές παρατηρείται μια βελτίωση αλλά για 40 και 64 επεξεργαστές δεν παρατηρείται κάτι τέτοιο.

Επιπρόσθετα, όταν οι επεξεργαστές αυξάνονται, τότε η επίδραση των ρ – διεργασιών καλυτερεύει. Στην περίπτωση του πολλαπλού συνδέσμου, υπάρχουν βαθύτερες κοιλάδες, δηλαδή, ο χρόνος εκτέλεσης μειώνεται γρηγορότερα όταν συμβαίνει να μειώνεται. Στον ανά δύο σύνδεσμο το κόστος των επιπρόσθετων ρ – διεργασιών μειώνεται. Ειδικότερα, καθώς οι επεξεργαστές αυξάνονται ο ανά δύο αλγόριθμος σύνδεσης γίνεται πιο ανεξάρτητος από το πλήθος των ρ – διεργασιών. Όταν οι επεξεργαστές λιγοστεύουν, ο ανά δύο σύνδεσμος εξαρτάται αρκετά περισσότερο από τον αριθμό των ρ – διεργασιών.

5.4.4.3 Κόστος επικοινωνίας

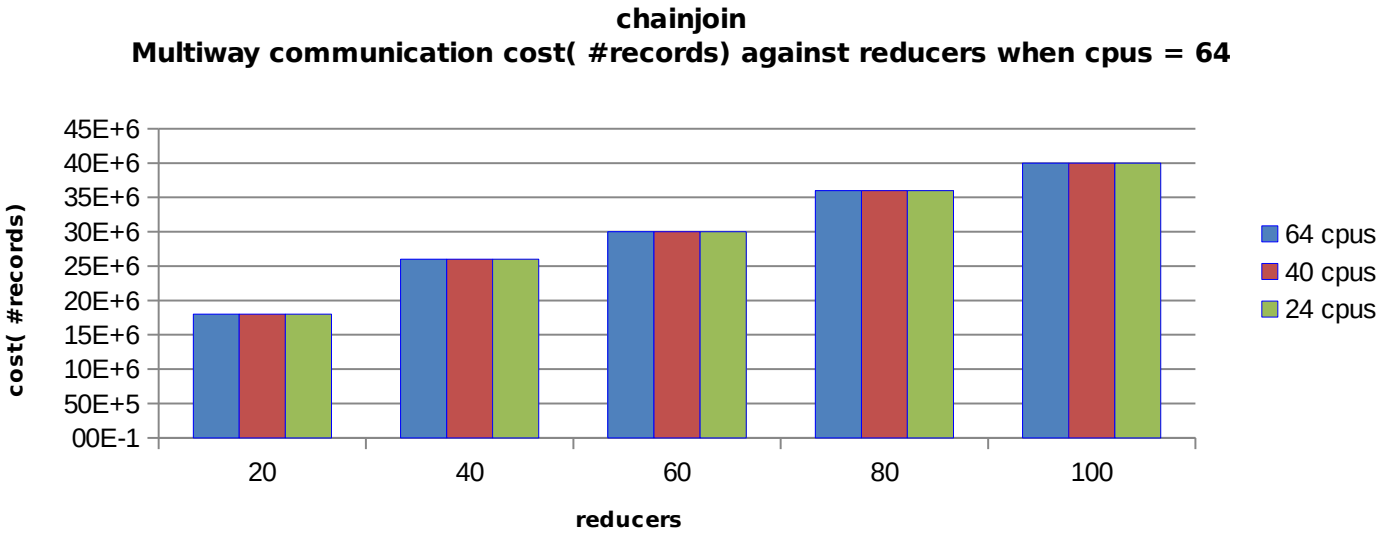


Illustration 15: Κόστος επικοινωνίας πολλαπλού συνδέσμου(#εγγραφές) για τον σύνδεσμο αλυσίδα σε σχέση με τους επεξεργαστές και τις ρ - διεργασίες

Το κόστος επικοινωνίας αυξάνεται ανάλογα με το πλήθος των ρ – διεργασιών αλλά δεν εξαρτάται από το πλήθος των επεξεργαστών. Το γεγονός αυτό αναμενόταν. Ωστόσο, η αύξηση του κόστους επικοινωνίας καθώς αυξάνονται οι ρ – διεργασίες δεν είναι η ίδια, μάλιστα είναι περιοδική με εξαίρεση όταν οι ρ – διεργασίες αυξάνονται από 20 σε 40 όπου τότε σχεδόν 10^7 εγγραφές παράγονται. Όταν οι ρ – διεργασίες γίνονται από 40 σε 60 έχουμε μια αύξηση των $4 * 10^6$ εγγραφών, από 60 σε 80 έχουμε μια αύξηση των $6 * 10^6$ εγγραφών κι από 80 σε 100 έχουμε μια αύξηση των $4 * 10^6$ εγγραφών. Με άλλα λόγια, το κόστος επικοινωνίας από την αύξηση των ρ – διεργασιών αυξάνετται με περιοδικό τρόπο κι αυτή η αύξηση είναι λίγη συγκριτικά με το πλήθος των εγγραφών π.χ. $4 * 10^6$ εγγραφές από 40 σε 60 σε ένα πλήθος εγγραφών 25.000.000 .

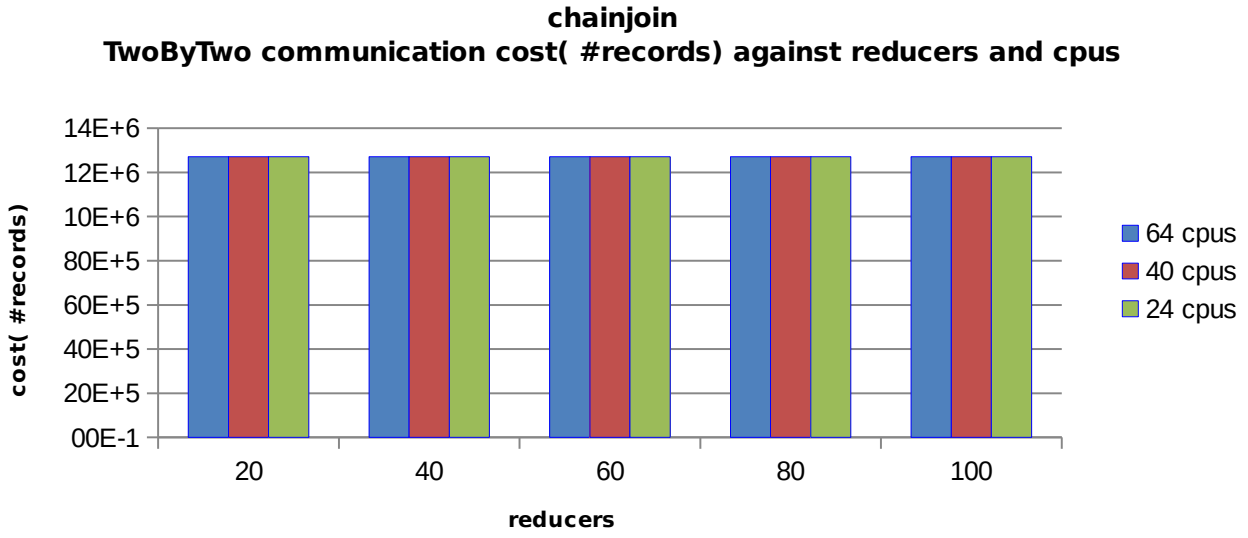


Illustration 16: Κόστος επικοινωνίας ανά δύο συνδέσμου(#εγγραφές) για τον σύνδεσμο αλυσίδα σε σχέση με τους επεξεργαστές και τις ρ - διεργασίες

Παραπάνω βλέπουμε πως το κόστος επικοινωνίας του ανά δύο συνδέσμου δεν εξαρτάται ούτε από τους επεξεργαστές αλλά κι ούτε από τις ρ – διεργασίες. Επίσης, βλέπουμε πως το κόστος επικοινωνίας του ανά δύο αλγορίθμου είναι πολύ λιγότερο από το αντίστοιχο κόστος του πολλαπλού συνδέσμου ειδικότερα για πολλές ρ – διεργασίες π.χ. 100, 12.000.000 έναντι 40.000.000 εγγραφές.

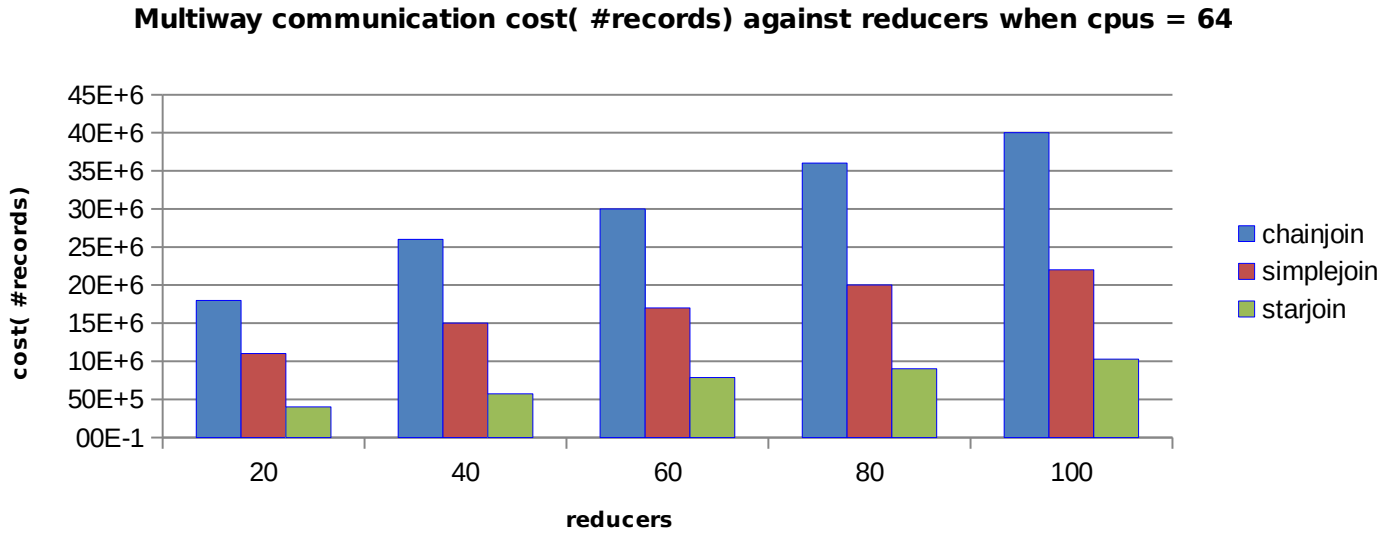


Illustration 17: Κόστος επικοινωνίας πολλαπλού συνδέσμου(#εγγραφές) για τον σύνδεσμο αλυσίδα σε σχέση με τους επεξεργαστές, τις ρ – διεργασίες και τις δομές συνδέσμων

Το παραπάνω γράφημα δείχνει πως ο σύνδεσμος αλυσίδα έχει το μεγαλύτεο κόστος επικοινωνίας συγκριτικά με τους άλλους δύο συνδέσμους. Το κόστος του συνδέσμου αστέρα είναι

το μικρότερο όπως αναμενόταν, γιατί οι περιφερειακές σχέσεις υστερούν σημαντικά συγκριτικά με την κεντρική σχέση κι όλες τις άλλες σχέσεις των άλλων συνδέσμων και γιατί οι εγγραφές της κεντρικής σχέσης δεν πολλαπλασιάζονται σε πλήθος μια και η κεντρική σχέση έχει όλες τις ιδιότητες του μ – κλειδιού. Ο πολλαπλασιασμός των περιφερειακών σχέσεων κατά την μ – φάση είναι ο μικρότερος δυνατός και κατά συνέπεια και το κόστος επικοινωνίας είναι μικρό.

Τελικά, από την παρακάτω απεικόνιση, επιβεβαιώνουμε το απαιτητικό του συνδέσμου αλυσίδα στην περίπτωση του ανά δύο συνδέσμου και πως το κόστος επικοινωνίας δεν εξαρτάται από τις ρ – διεργασίες.

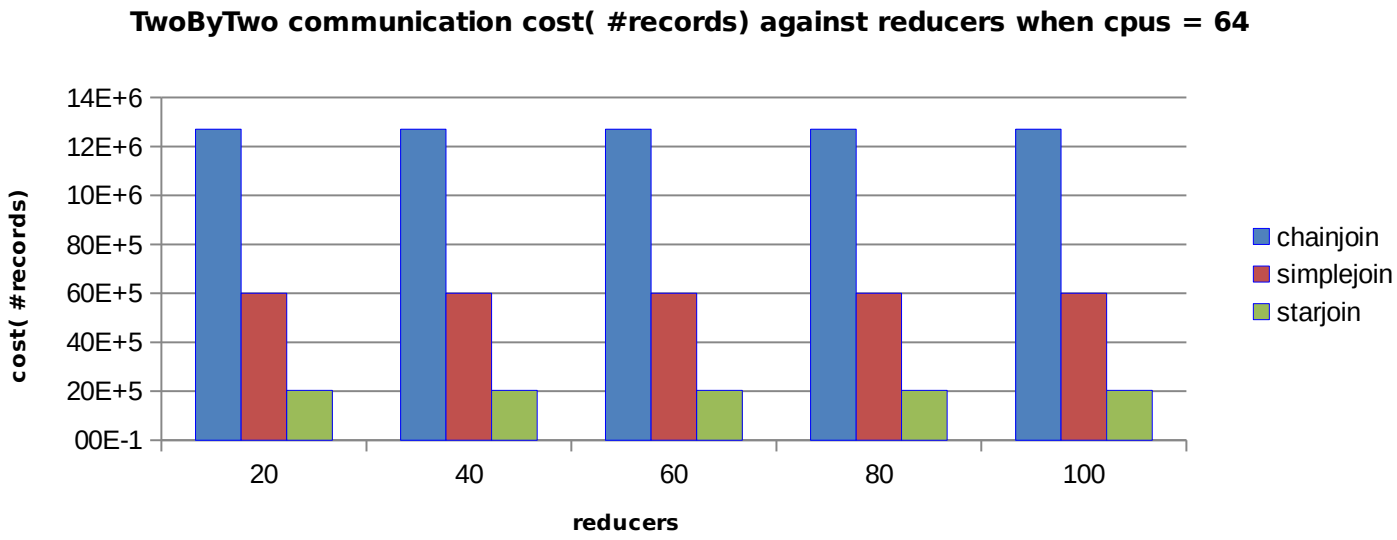


Illustration 18: Κόστος επικοινωνίας ανά δύο συνδέσμου(#εγγραφές) για τον σύνδεσμο αλυσίδα σε σχέση με τους επεξεργαστές, τις ρ – διεργασίες και τις δομές συνδέσμων

5.4.4.4 Σύνοψη πειράματος και λύσεις

Συνοψίζοντας, ο χρόνος εκτέλεσης επηρεάζεται τόσο από τους επεξεργαστές όσο κι από τις ρ – διεργασίες. Αυτή η επιρροή είναι γραμμική και ημιτονοειδής ανάλογα με τον σύνδεσμο και για τους δύο αλγόριθμους σύνδεσης. Ωστόσο, στην περίπτωση του ανά δύο συνδέσμου η μη γραμμική επιρροή γρήγορα μετατρέπεται σε γραμμική με την αύξηση των επεξεργαστών. Επιπλέον, η επίδραση των επεξεργαστών είναι πάντα ωφέλιμη κι επηρεάζει και την επίδραση των ρ – διεργασριών καθώς αυτές μεταβάλλονται σε πλήθος. Η επίδραση των ρ – διεργασιών δεν χαρακτηρίζεται πάντα εποικοδομητική. Ειδικότερα στην περίπτωση του ανά δύο συνδέσμου είναι πάντα επιζήμια.

Το κόστος επικοινωνίας από την άλλη, επηρεάζεται μόνο από την ποσότητα των ρ – διεργασιών και μόνο στην περίπτωση του πολλαπλού συνδέσμου. Αυτό ήταν αναμενόμενο λόγω της φύσης του πολλαπλού συνδέσμου. Ο πολλαπλός σύνδεσμος προκειμενου να στείλει τις κατάλληλες εγγραφές στις κατάλληλες ρ – διεργασίες, πρέπει να πολλαπλασιάσει τις εγγραφές εισόδου στην μ – φάση. Ενώ οι ρ – διεργασίες αυξάνονται κι ο πολλαπλασιαστικός παράγοντας αυξανεται αναπόφευκτα καθώς οι νέες ρ – διεργασίες πρέπει να λάβουν εγγραφές εισόδου. Εν κατακλείδι, επειδή αυτή είναι η φύση του πολλαπλού συνδέσμου δεν υπάρχει κάποια λύση σε αυτό

το πρόβλημα, δηλαδή, η αύξηση του κόστους επικοινωνίας ενώ αυξάνονται οι ρ – διεργασίες. Αυτό που μπορεί να γίνει είναι η χρήση ενός συνδυάσμωνα(combiner) και να συμπιεστεί η έξοδος της μ – φάσης ώστε να περιοριστεί όσο το δυνατόν το εν λόγω κόστος.

ПАРАРТНМА

Γλωσσάρι

Στο παρόν μέρος επεξηγούνται διάφοροι όροι που υπάρχουν σε όλη την έκταση της εργασίας αυτής. Ο κάθε όρος παρέχεται στα Ελληνικά και εντός παρενθέσεων και στα Αγγλικά.

Μεταβατικό κλείσιμο(Transitive closure) : στη θεωρία γράφων αποτελεί ένα γράφο Μ που δείχνει για όλους του κόμβους Κ ενός άλλου αρχικού γράφου Γ, ποιοι κόμβοι του Γ γειτνιάζουν άμεσα ή έμμεσα(μέσω μονοπατιού) με κάθε έναν κόμβο που ανήκει στο Κ.

Πολλαπλός σύνδεσμος(multi way join) : ένας σύνδεσμος πολλών σχέσεων.

Προσομοιωμένη ανόπτηση(simulated annealing) : αποτελεί έναν αλγόριθμο αναζήτησης κατά βάθος που χειρίζεται ευριστικές μεθόδους και πιθανότητες. Μέσω των πιθανοτήτων ο αλγόριθμος αυτός επιλύει το πρόβλημα των ακρότατων. Το πρόβλημα των ακρότατων είναι αυτό κατά το οποίο η αναζήτηση περιορίζεται σε κάποια συγκεκριμένα σημεία του χώρου που λέγονται ακρότατα. Ο αλγόριθμος λοιπόν, χάρη στις πιθανότητες, απεγκλωβίζεται από τα ακρότατα και μπορεί να συνεχίσει την αναζήτηση σε άλλα σημεία του χώρου.

Οντότητα(entity) : ένα διακριτό αντικείμενο του φυσικού κόσμου σε επίπεδο βάσεων δεδομένων. Μια οντότητα έχει χαρακτηριστικά που λέγονται κι ιδιότητες. Αναπαρίσταται ως ένας πίνακας με όνομα αυτό της οντότητας και τόσες στήλες όσο και τα χαρακτηριστικά της οντότητας. Δεδομένου οτι μια οντότητα αποτελεί ένα διακριτό αντικείμενο, δηλαδή, ξεχωρίζει από τον υπόλοιπο κόσμο, πρέπει να έχει ένα χαρακτηριστικό που την καθιστά μοναδική.

Σύνδεσμος(join) : είναι μια πράξη, ένας τελεστής, μια συνάρτηση που εφαρμόζεται πάνω σε σχέσεις δίνοντας ως αποτέλεσμα άλλες σχέσεις. Συγχωνεύει τις σχέσεις σε μία, υπό προϋποθέσεις. Η κύρια προϋπόθεση είναι οι σχέσεις που συγχωνεύονται, να έχουν κάποια κοινά χαρακτηριστικά πρώτον, και δεύτερον, για αυτά τα χαρακτηριστικά να έχουν την ίδια τιμή. Για παράδειγμά αν το κοινό χαρακτηριστικό είναι το χρώμα, τότε θα πρέπει να έχουν το ίδιο χρώμα(τιμή), π.χ. κόκκινο.

Σχέση(relationship) : στο πλαίσιο της εργασίας αυτής, μια σχέση αποτελεί την οργάνωση σχετικών μεταξύ τους δεδομένων σε μορφή πίνακα. Με άλλα λόγια, μια σχέση είναι ένας πίνακας από δεδομένα. Σε μια βάση δεδομένων, μια σχέση πέρα από την έννοια του πίνακα έχει και μια σημασιολογική έννοια, καθώς αντικατοπτρίζει τον συσχετισμό κάποιων αντικειμένων – οντοτήτων του φυσικού κόσμου.

Χωρική συγγένεια(spatial locality) : εκφράζει το γεγονός ότι όταν προσπελάζεται μια θέση μνήμης η πιθανότητα, να προσπελαστούν στο κοντινό μέλλον οι γειτονικές θέσης μνήμης, είναι αυξημένη.

Βιβλιογραφία

01: Wikipedia, , , http://en.wikipedia.org/wiki/Sloan_Digital_Sky_Survey
02: S. D. Viglas, J. F. Naughton, and J. Burger, Maximizing the output rate of multi-way join queries over streaming information sources, 2003
03: T. Urhan and M. J. Franklin, XJoin: A reactively - scheduled pipelined join operator, 2000
04: S. Babu and J. Widom, StreaMon: an adaptive engine for stream query processing, 2004
05: R. Motwani, J. Widom, et al, Query processing, approximation, and resource management in a data stream management system, 2003
06: S. Babu, U. Srivastava, and J. Widom, Exploiting k - constraints to reduce memory overhead in continuous queries over data streams, 2002
07: S. Babu, R. Motwani, K. Munagala, I. Nishizawa, and J. Widom, Adaptive ordering of pipelined stream filters, 2004
08: S. Babu, K. Munagala, J. Widom, and R. Motwani, Adaptive caching for continuous queries, 2004
09: H. Jacobsson, Tree - based techniques for query evaluation, 1993
10: Hongjun Lu, Ming - Chein Shan, Kian - Lee Tan, Optimization of Multi - way join queries for parallel execution, 1991
11: Krishnamurthy, R., Boral, H., and Zaniolo,C. , Optimization of Nonrecursive Queries, 1986
12: Swami, A. and Gupta, A., Optimization of Large Join Queries, 1988
13: Swami, A. , Optimization of Large Join Queries : Combining Heuristics and Combinatorial Techniques, 1989
14: Ioannidis, Y. E. and Kang, Y., Randomized Algorithms for Optimizing Large Join Queries, 1990
15: Ono, K. and Lohman, G. M., Measuring the Complexity of Join Enumeration in Relational Query Optimization, 1990
16: Stonebraker, M., Katz, R., Patterson. D., and Ousterhout, J., The Design of XPRS, 1988
17: Schneider, D. A. and Dewitt, D. J, Tradeoffs in Processing Complex Join Queries via Hashing in Multi processor Database Machines, 1990
18: K. A. Ross and J. Cieslewicz, Optimal splitters for database partitioning with size bounds, 2009
19: S. Ghandeharizadeh and D. J. DeWitt, Hybrid-range partitioning strategy: A new declustering strategy for multiprocessor database machines, 1990
20: U. Srivastava, K. Munagala, J. Widom, and R. Motwani, Query optimization over web services, 2006
21: D. Florescu, A. Levy, I. Manolescu, and D. Suciu, Query optimization in the presence of limited access patterns, 1999
22: F. N. Afrati and J. D. Ullman, Optimizing joins in a Map - Reduce Environment, 2009
23: S. Ghemawat, H. Gobioff, and S.-T. Leung, The google File system, 2003
24: Apache, Hadoop, 2006, http://hadoop.apache.org/
25: J. Dean and S. Ghemawat, Mapreduce: simplified data processing on large clusters, 2008
26: Chuck Lam, Hadoop in action, 2011
27: D. J. DeWitt, J. F. Naughton, D. A. Schneider, and S. Seshadri, Practical skew handling in parallel joins, 1992
28: Z. G. Ives, D. Florescu, M. Friedman, A. Y. Levy, and D. S. Weld, An adaptive query execution system for data integration, 1999
29: S. Madden, M. A. Shah, J. M. Hellerstein, and V. Raman, Continuously adaptive continuous queries over streams, 2002
30: F. N. Afrati, D. Fotakis, and J. D. Ullman. Enumerating subgraph instances using map-reduce. Technical report, Stanford InfoLab, January 2012. Available at http://ilpubs.stanford.edu:8090/1020/.
31: F. N. Afrati, A. D. Sarma, D. Menestrina, A. Parameswaran, and J. D. Ullman. Fuzzy joins using

mapreduce. In ICDE, 2012.
32: N. Alon. On the number of subgraphs of prescribed type of graphs with a given number of edges. Israel Journal of Mathematics, 38(1-2):116{130, 1981.
33: Amazon. Amazon Elastic Compute Cloud (Amazon EC2). Amazon Inc., 2008.
34: A. Atserias, M. Grohe, and D. Marx. Size bounds and query plans for relational joins. In FOCS, pages 739{748, 2008.
35: R. L. F. Cordeiro, C. T. Jr., A. J. M. Traina, J. Lopez, U. Kang, and C. Faloutsos. Clustering very large multi-dimensional datasets with mapreduce. In KDD, 2011.
36: J. Dean and S. Ghemawat. Mapreduce: simplied data processing on large clusters. In OSDI, pages 137{150, 2004.
37: W. Feller. An Introduction to Probability Theory and Its Applications. Vol. 1, 3rd ed. New York: Wiley, 1968. (Stirling's Formula in Section 2.9).
38: M. Grohe and D. Marx. Constraint solving via fractional edge covers. In SODA, pages 289{298, 2006.
39: W. Hasan and R. Motwani. Optimization algorithms for exploiting the parallelism - communication tradeo in pipelined parallelism. In VLDB, 1994.
40: R. Ikeda, H. Park, and J. Widom. Provenance for generalized map and reduce workows. In CIDR, 2011.
41: H. J. Karlo, S. Suri, and S. Vassilvitskii. A model of computation for mapreduce. In SODA, pages 938 - 948, 2010.
42: P. Koutris and D. Suciu. Parallel evaluation of conjunctive queries. In PODS, pages 223 - 234, 2011.
43: Y. Kwon, M. Balazinska, B. Howe, and J. A. Rolia. Skewtune: mitigating skew in mapreduce applications. In SIGMOD Conference, pages 25 - 36, 2012.
44: A. C. McKellar and E. G. Coman. Organizing matrices and matrix operations for paged memory systems. Commun. ACM, 12(3):153{165, 1969.
45: A. Okcan and M. Riedewald. Processing theta-joins using mapreduce. In SIGMOD Conference, pages 949 - 960, 2011.
46: C. Olston and B. Reed. Inspector gadget: A framework for custom monitoring and debugging of distributed dataows. In Proc. of VLDB, 2011.
47: A. Rajaraman and J. D. Ullman. Mining of Massive Datasets. Cambridge University Press. Also available on-line at i.stanford.edu/~ullman/mmds.html, 2011.
48: T. Schank. Algorithmic Aspects of Triangle-Based Network. University of Karlsruhe (TH), 2007.
49: S. Suri and S. Vassilvitskii. Counting triangles and the curse of the last reducer. In WWW, 2011.
50: W.C. Tan. Provenance in Databases: Past, Current, and Future. IEEE Data Engineering Bulletin, 2008.
51: R. Vernica, M. J. Carey, and C. Li. Ecient parallel set-similarity joins using mapreduce. In SIGMOD Conference, 2010.
52: T. White. Hadoop - The Denitive Guide: Storage and Analysis at Internet Scale (2. ed.). O'Reilly, 2011.
53: M. S. Bernstein et al. Soylent: a word processor with a crowd inside. In UIST, pages 313 – 322, 2010.
54: J. P. Bigham et al. Vizwiz: nearly real-time answers to visual questions. In UIST, pages 333 – 342, 2010.
55: A. P. Dawid and A. M. Skene. Maximum likelihood estimation of observer error-rates using the em algorithm. Journal of the Royal Statistical Society., 28(1):pages 20 – 28, 1979.
56: J. L. Fleiss. Measuring nominal scale agreement among many raters. Psychological Bulletin, 76(5):pages 378–382, 1971.
57: M. Franklin, D. Kossmann, T. Kraska, S. Ramesh, and R. Xin. CrowdDB: Answering queries

with crowdsourcing. In SIGMOD, pages 61–72, 2011.
58: P. G. Ipeirotis, F. Provost, and J. Wang. Quality management on amazon mechanical turk. In HCOMP, pages 64–67, 2010.
59: M. G. Kendall. A new measure of rank correlation. Biometrika, 30(1-2):pages 81–93, 1938.
60: A. Kittur, B. Smus, and R. E. Kraut. CrowdForge: Crowdsourcing Complex Work. Technical report, 2011.
61: R. Likert. A technique for the measurement of attitudes. Archives of Psychology, 22(140):pages 1–55, 1932.
62: G. Little et al. Turkit: human computation algorithms on mechanical turk. In UIST, pages 57–66, 2010.
67: A. Marcus, E. Wu, et al. Crowdsourced databases: Query processing with people. In CIDR, 2011.
68: W. Mason and D. J. Watts. Financial incentives and the “performance of crowds”. In HCOMP, pages 77–85, 2009.
69: A. Parameswaran and N. Polyzotis. Answering queries using databases, humans and algorithms. In CIDR, 2011.
70: Foto N. Afrati, Anish Das Sarma, Semih Salihoglu, Jeffrey D. Ullman, Upper and Lower Bounds on the Cost of a Map-Reduce Computation, IEEE 28th International *Conference* on Data Engineering, p.498-509, April 01-05, 2012. 4.