ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Ασύγχρονη ελαστική μεταφορά δεδομένων για ερωτήματα αναλυτικής επεξεργασίας

Αναστάσιος Κ. Γιαννακόπουλος

Επιβλέπων: **Ιωάννης Ιωαννίδης,** Καθηγητής

ΑΘΗΝΑ

ΝΟΕΜΒΡΙΟΣ 2015

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Ασύγχρονη ελαστική μεταφορά δεδομένων για ερωτήματα αναλυτικής επεξεργασίας για εξόρυξη δεδομένων

Αναστάσιος Κ. Γιαννακόπουλος

Α.Μ.: 115200900053

ΕΠΙΒΛΕΠΟΝΤΕΣ: **Ιωάννης Ιωαννίδης,** Καθηγητής

ΠΕΡΙΛΗΨΗ

Λόγω του μεγάλου όγκου των δεδομένων που επεξεργάζονται οι σύγχρονες εφαρμογές η χρήση κατανεμημένων συστημάτων είναι απαραίτητη. Η αύξηση της απόδοσής τους είναι αναγκαία για τις μελλοντικές εφαρμογές. Όμως το κυριότερο πρόβλημα που διακρίνεται σε αυτά τα συστήματα είναι η κατανάλωση ενέργειας. Λόγω αυτού, η αρχιτεκτονική αυτών των συστημάτων θα πρέπει να αλλάξει δραματικά στο μέλλον.

Βασικό κομμάτι της λειτουργίας τους είναι ο διαχωρισμός μεγάλου όγκου δεδομένων σε κομμάτια και ο διαμοιρασμός τους ανά τους κόμβους του συστήματος, προκειμένου να είναι δυνατή η επεξεργασία τους. Όμως προβλέπεται ότι στο μέλλον το κύριο σημείο συμφόρησης (bottleneck) θα είναι η μεταφορά δεδομένων, αφού η υπολογιστική επεξεργασία θα απαιτεί λιγότερη ενέργεια. Επομένως υπάρχει η ανάγκη για ελαστική μεταφορά δεδομένων όπου θα ξεκλειδώνει περισσότερο εύρος ζώνης (bandwidth) με τις αντισταθμιστικές επιλογές (trade-offs) που θα μπορούν να γίνουν ανάμεσα στην υπολογιστική ισχύ και στο επιτευκτό εύρος ζώνης αποστολής δεδομένων.

Η μεταφορά δεδομένων μπορεί να γίνει με την μέθοδο αποθήκευση-αποστολή (store and forward) ή με την μέθοδο σωλήνωσης (pipelining). Κύριο μειονέκτημα της μεθόδου αποθήκευση-αποστολή είναι η αδυναμία επικάλυψης χρόνου επεξεργασίας με χρόνο αποστολής δεδομένων με αποτέλεσμα μικρότερο εύρος ζώνης σε σχέση με την μέθοδο σωλήνωσης. Η μέθοδος σωλήνωσης, παρόλο που βελτιώνει το συγκεκριμένο αρνητικό σημείο της μεθόδου αποθήκευση-αποστολή, εισάγει εξάρτηση μεταξύ των κόμβων του συστήματος. Η εξάρτηση, λόγω αργής μεταφοράς ή αργής επεξεργασίας δεδομένων σε κάποιον κόμβο, μπορεί να δημιουργήσει συμφόρηση σε όλους τους εξαρτημένους από αυτόν, κόμβους. Επομένως, ανάλογα με τις ιδιότητες των δεδομένων προς επεξεργασία, καθώς και με το εύρος ζώνης που προσφέρει το δίκτυο μεταφοράς του συστήματος, πρέπει να είμαστε σε θέση να επιλέξουμε δυναμικά ποια μέθοδο από τις δύο συμφέρει να χρησιμοποιηθεί ανά πάσα στιγμή.

Στην εργασία αυτή θα παρουσιάσουμε μια ελαστική μέθοδο μεταφοράς δεδομένων η οποία θα προσφέρει δύο διαστάσεις ελαστικότητας, την ελαστικότητα κατά την μεταφορά των δεδομένων και την ελαστικότητα στην συμπεριφορά της λειτουργίας της. Κινείται δηλαδή ομαλά μεταξύ όλου του εύρους συμπεριφορών που οριοθετούν οι δύο βασικές μέθοδοι μεταφοράς. Επιπλέον χρησιμοποιούμε ασύγχρονο, μη-παρεμποδιστικό (asynchronous, non-blocking) προγραμματισμό προκειμένου η μέθοδος που προτείνουμε να είναι πιο αποδοτική και να είναι λιγότερο απαιτητική σε πόρους σε σύγκριση με τον συμβατικό παράλληλο προγραμματισμό.

ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ: Κατανεμημένα συστήματα

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: Μεταφορά Δεδομένων, Ελαστική ενταμίευση, Κατανεμημένα συστήματα, Υπολογιστικά νέφη, Exareme, συμπίεση, προσαρμοστική μεταφορά δεδομένων, αποθήκευση-αποστολή, διοχέτευση.

ABSTRACT

Modern applications process large amounts of data, rendering the use of distributed systems a necessity. Increasing their performance is necessary for future scientific applications. The main problem of distributed systems is their power consumption, therefore their architecture will require some drastic changes in the future.

The main benefit of distributed systems is the segmentation and distribution of data to the appropriate system nodes for processing. But in the near future, the main bottleneck of these systems will be data transfer, as the power required for computing processing will be decreased. Therefore, there exists a need for elastic data transfer, which will unlock more bandwidth due to the trade-offs between computing power and data transfer bandwidth.

Data transfer can be implemented using two different approaches, store and forward or pipelining. A basic downside of the store and forward approach is its inability to overlap data processing and transfer, which introduces lower throughput than pipelined approach. Even though pipelining improves this, it introduces dependencies between system nodes. Such dependencies could create congestion, as a certain node could suffer from either low bandwidth or limited processing resources. Therefore, our goal is to be able to choose dynamically the approach that offers optimal performance depending on the current bandwidth/processing power combination.

In this paper we present an elastic data transfer approach which provides two dimensions of elasticity, elasticity in data transfer and elasticity in its behavior, smoothly alternating in the range of behaviors these two basic data transfer methods provide. Furthermore, we use asynchronous, non-blocking programming in order to be more efficient and limit the need in resources conventional parallel programming would require.

SUBJECT AREA: Distributed systems

KEYWORDS: Data transfer, Elastic Buffering, Distributed systems, Cloud computing, Exareme, Compression, Adaptive data transfer, Store and Forward, Pipelining

ΕΥΧΑΡΙΣΤΙΕΣ

Για τη διεκπεραίωση της παρούσας Πτυχιακής Εργασίας, θα ήθελα να ευχαριστήσω τους επιβλέποντες, καθ. Ιωάννη Ιωαννίδη, Herald Kllapi, Λευτέρη Σταματογιαννάκη και Ιωάννη Φουφούλα για την πολύτιμη συμβολή τους στην ολοκλήρωση της.

ΠΕΡΙΕΧΟΜΕΝΑ

ΚΑΤΑΛΟΓΟΣ ΣΧΗΜΑΤΩΝ

ΚΑΤΑΛΟΓΟΣ ΠΙΝΑΚΩΝ

ΠΡΟΛΟΓΟΣ

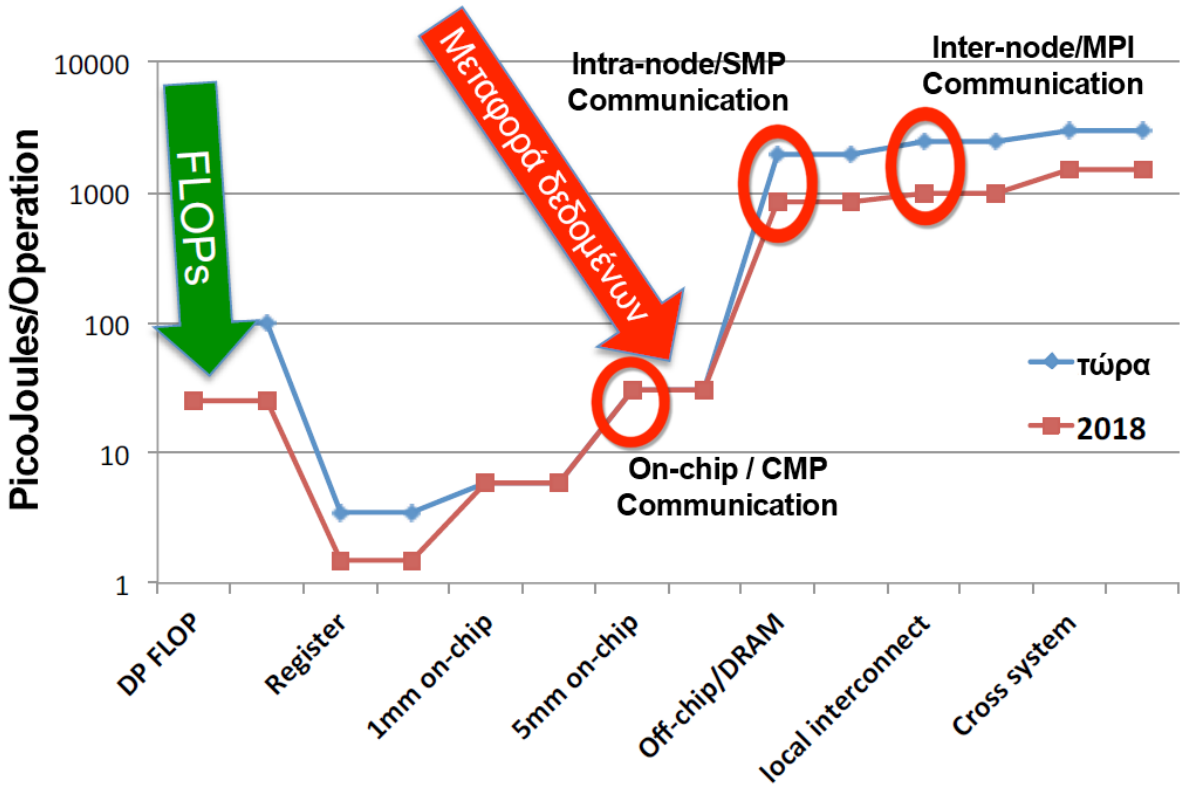
Η παρούσα πτυχιακή εργασία εκπονήθηκε στα πλαίσια του προγράμματος προπτυχιακών σπουδών του τμήματος Πληροφορικής και Τηλεπικοινωνιών, του Εθνικού και Καποδιστριακού Πανεπιστημίου Αθηνών. Αντικείμενο της εργασίας ήταν η σχεδίαση μιας ελαστικής μεθόδου μεταφοράς μεγάλου όγκου δεδομένων με βέλτιστο τρόπο, με σκοπό της χρήση της από ένα σύστημα αναλυτικής επεξεργασίας σαν το Exareme.

1. ΕΙΣΑΓΩΓΗ

Ο όγκος των δεδομένων και η πολυπλοκότητα των εφαρμογών έχουν επιβάλει τη χρήση κατανεμημένων συστημάτων, τα οποία μπορεί να αποτελούνται από εκατοντάδες ή χιλιάδες μηχανήματα κόμβους. Το κύριο πρόβλημά των κατανεμημένων συστημάτων είναι η μεγάλη τους ανάγκη για ηλεκτρική ενέργεια. Η αρχιτεκτονική των κόμβων πρόκειται να αλλάξει δραματικά στο μέλλον, αφού λόγω του κόστους ενέργειας και της δυσκολίας ψύξης, οι επεξεργαστές πρέπει να περιορίσουν της αύξηση του χρονισμού τους [1].

Από την εποχή του Gordon Earl Moore (ιδρυτή της Intel) ήταν δυνατόν κάποιος να μπορεί να προβλέψει την κατάσταση της επεξεργαστικής ισχύς που επιτυγχάνεται σήμερα. Σήμερα μπορούν να επιτευχθούν δεκάδες petaFLOPS χάρη στην μεγάλη προσπάθεια για συνεχή βελτίωση των τεχνολογιών. Ενώ στο πρόσφατο παρελθόν η προσπάθεια για συνεχή αύξηση της απόδοσης των συστημάτων επικεντρωνόταν στον διπλασιασμό του χρονισμού των επεξεργαστών κάθε 12-24 μήνες, σήμερα διπλασιάζεται η παραλληλία πάνω στο κύκλωμα (on-chip) [1]. Αν συνεχιστεί ο ίδιος ρυθμός αύξησης της απόδοσης μέχρι το 2020, θα μπορεί να επιτευχθεί ένα exaFLOPS [2]. Όμως υπάρχουν τρία μεγάλα ζητήματα που καθιστούν δύσκολο αυτό το εγχείρημα.

1. Το κόστος της ενέργειας. Το 2010 το 1 petaFLOPS χρειαζόταν 3 MW ηλεκτρική ενέργεια, όπου το 1 MW κοστίζει περίπου 1 εκατομμύριο δολάρια. Το 2020 με βάση την αναμενόμενη κλιμάκωση το ένα exaFLOPS θα χρειάζεται 200 MW, ενώ το άνω όριο που έχει τεθεί σαν λογικό κόστος είναι τα 20 MW [3]. Το όριο είναι κινητό, όμως θέτει σε ρίσκο τον σχεδιασμό των μελλοντικών συστημάτων.
2. Το κόστος των FLOPS στο παρελθόν ήταν το πιο δαπανηρό από άποψη κόστους σχεδιασμού και ενέργειας. Παρ'όλα αυτά στο μέλλον θα απαιτούνται 100 W για την επίτευξη 10 teraFLOPS σε ένα κύκλωμα, ενώ θα απαιτούνται 2000 W ενέργειας για την παροχή αρκετού ρυθμού μεταφοράς των δεδομένων για την αποθήκευσή τους στην κύρια μνήμη [1] με την υπάρχουσα τεχνολογία. Επομένως όπως φαίνεται στο **Σχήμα 1** το 2018 το κόστος των FLOPS θα βελτιωθεί κατά 5-10 φορές, ενώ το κόστος της μεταφοράς δεδομένων δεν θα βελτιωθεί αισθητά [1]. Με αποτέλεσμα το κόστος των FLOPS να είναι μικρότερο της μεταφοράς δεδομένων. Η μεταφορά δεδομένων θα είναι το κυριότερο σημείο συμφόρησης.
3. Το κόστος της μεταφοράς δεδομένων. Από τις έρευνες που έχουν γίνει προκύπτουν τα εξής συμπεράσματα [1]:
    1. Η ισχύς που καταναλώνεται αυξάνεται αναλογικά με τον ρυθμό μετάδοσης bit (bit-rate). Επομένως όσο κατευθυνόμαστε σε συνδέσεις πολύ υψηλούς εύρους ζώνης οι απαιτήσεις ισχύος θα είναι ένα συνεχώς αυξανόμενο ζήτημα.
    2. Η κατανάλωση ενέργειας είναι πολύ εξαρτώμενη από την απόσταση. Επομένως το εύρος ζώνης συνεχώς θα περιορίζεται τοπικώς, όσο η μείωση της κατανάλωσης ενέργειας θα γίνεται όλο και πιο δύσκολο πρόβλημα.



Σχήμα 1: Κόστος ενέργειας της μεταφοράς δεδομένων σε σχέση με το κόστος ενός flop στα συστήματα του σήμερα και στα συστήματα του 2018.

Πίνακας 1: Βελτίωση τμημάτων των συστημάτων ανά έτος. Τα κενά μεγαλώνουν εκθετικά με την πάροδο του χρόνου.

Βελτίωση ανά έτος			
Χρόνος για κάθε flop		Εύρος ζώνης	Χρόνος απόκρισης
59%	Δίκτυο	26%	15%
	DRAM	23%	5%

Όπως φαίνεται στον **Πίνακα 1** τα κενά μεταξύ των τεχνολογιών συνεχώς μεγαλώνουν ανά έτος και μάλιστα με εκθετικό ρυθμό. Είναι δύσκολο να αλλάξουν οι ρυθμοί αυτοί, διότι ο χρόνος απόκρισης εξαρτάται από την ταχύτητα του φωτός, και συνεχώς πλησιάζουμε τα όριά της όλο και περισσότερο ανά έτος, και είναι χρηματικά ακριβή η κάθε αύξηση του εύρους ζώνης [3].

Συμπερασματικά, στο μέλλον οι αλγόριθμοι, το λογισμικό και οι εφαρμογές θα πρέπει να μεριμνήσουν για την τοπικότητα των δεδομένων. Τα συστήματα αναλυτικής επεξεργασίας θα πρέπει να διαθέτουν επαρκείς πληροφορίες και έλεγχο ώστε να λαμβάνονται αποφάσεις που αξιοποιούν στο μέγιστο τις πληροφορίες σχετικά με την τοπολογία της επικοινωνίας και την τοπικότητα των δεδομένων. Χρειάζεται να είναι ελαστικά και να είναι ικανά να μεγιστοποιήσουν την απόδοση με της επιλογές τους. Χρειάζεται να μπορούν να προσφέρονται αντισταθμιστικές επιλογές μεταξύ

υπολογιστικής επεξεργασίας και αποστολής δεδομένων, ώστε να επιτυγχάνεται το μεγαλύτερο δυνατό εύρος ζώνης αποστολής δεδομένων εκτός κυκλώματος.

Σε αυτήν την εργασία θα αναπτύξουμε μια ασύγχρονη μέθοδο αποστολής και λήψης δεδομένων, η οποία θα είναι ελαστική και δυναμική. Η μέθοδος θα μπορεί να προσφέρει δύο διαστάσεις ελαστικότητας. Την ελαστικότητα στην μεταφορά δεδομένων. Καθώς και την ελαστικότητα στην ενταμίευση (buffering) των δεδομένων ανάμεσα στην κύρια μνήμη του συστήματος και στον δίσκο.

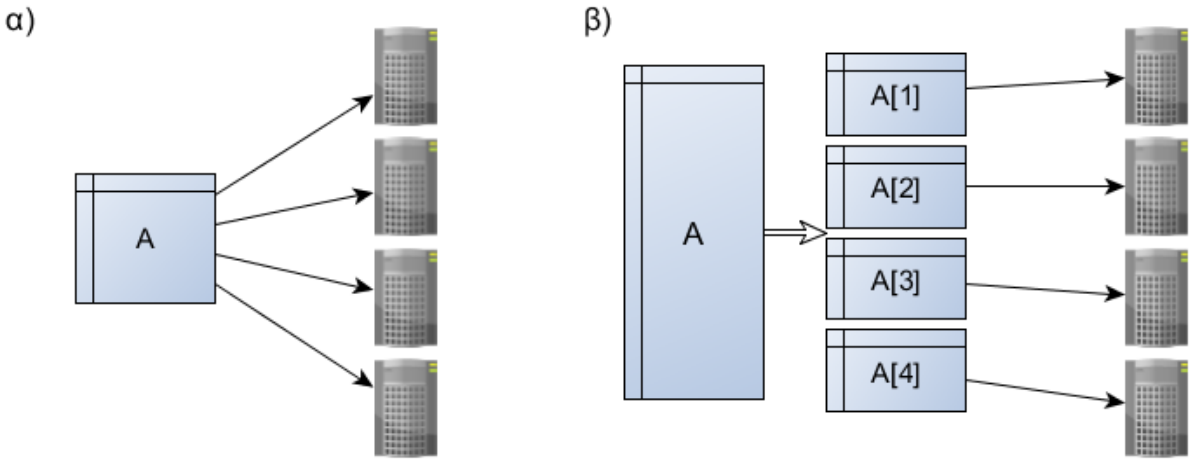
2. ΤΟ ΣΥΣΤΗΜΑ ΤΟΥ EXAREME

Το εργαστήριο του MADgIK [4] έχει αναπτύξει το κατανεμημένο σύστημα επεξεργασίας ροών δεδομένων Exareme [5]. Το Exareme υποστηρίζει παράλληλη επεξεργασία δεδομένων πάνω απο συστάδες υπολογιστών (clusters) ή υπολογιστικά νέφη (clouds). Το σύστημα προσφέρει μια δηλωτική γλώσσα βασισμένη στην SQL η οποία μπορεί να εκφράσει πολύπλοκες ροές που επεξεργάζονται αποδοτικά μεγάλου όγκου δεδομένα. Από τους πρωταρχικούς στόχους για την ανάπτυξη του συστήματος του Exareme, ήταν η επεξεργασία όσο το δυνατό μεγαλύτερου όγκου δεδομένων με το λιγότερο δυνατό κόστος.

Σκοπός ενός συστήματος σαν το Exareme είναι να απαντάει βέλτιστα σε ερωτήματα αναλυτικής επεξεργασίας. Προκειμένου να γίνει η επεξεργασία πάνω σε μεγάλου όγκου δεδομένα, είναι αναγκαίο να γίνει διάσπαση των δεδομένων ανά τους κόμβους του συστήματος. Στην συνέχεια, τα κατανεμημένα δεδομένα επεξεργάζονται τοπικά από κάθε κόμβο.

2.1 Λειτουργίες αποστολής δεδομένων στο Exareme

Σε ένα σύστημα όπως το Exareme είναι αναγκαίες δύο ειδών λειτουργίες για την μεταφορά δεδομένων. Η μια είναι η λειτουργία διαμοιρασμού (broadcast) των ίδιων δεδομένων προς όλους τους κόμβους παραλήπτες, ενώ η άλλη είναι η διάσπαση (split) των δεδομένων, και η αποστολή των τμημάτων ανά τους κόμβους.

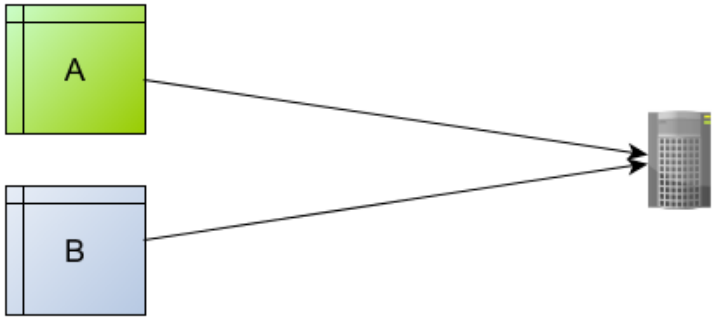


Σχήμα 2: α) Λειτουργία διαμοιρασμού (broadcast) ενός πίνακα. β) Λειτουργία διάσπασης (split) όπου ο αρχικός πίνακας διασπάται σε μικρότερους, ένας για τον κάθε κόμβο παραλήπτη.

Το Exareme επεξεργάζεται δεδομένα βάσεων, με πράξεις πάνω σε δύο ή και παραπάνω πίνακες την φορά. Μία χαρακτηριστική πράξη στην αναλυτική επεξεργασία δεδομένων, είναι η πράξη JOIN πάνω σε δύο πίνακες A και B, με M και N εγγραφές αντίστοιχα. Προκειμένου να γίνει η πράξη του JOIN θα πρέπει να ελεγχθεί η κάθε γραμμή του πίνακα A με την κάθε γραμμή του πίνακα B. Δηλαδή η συνολική πολυπλοκότητα της πράξης JOIN πάνω στους δυο πίνακες θα είναι $O(M \cdot N)$.

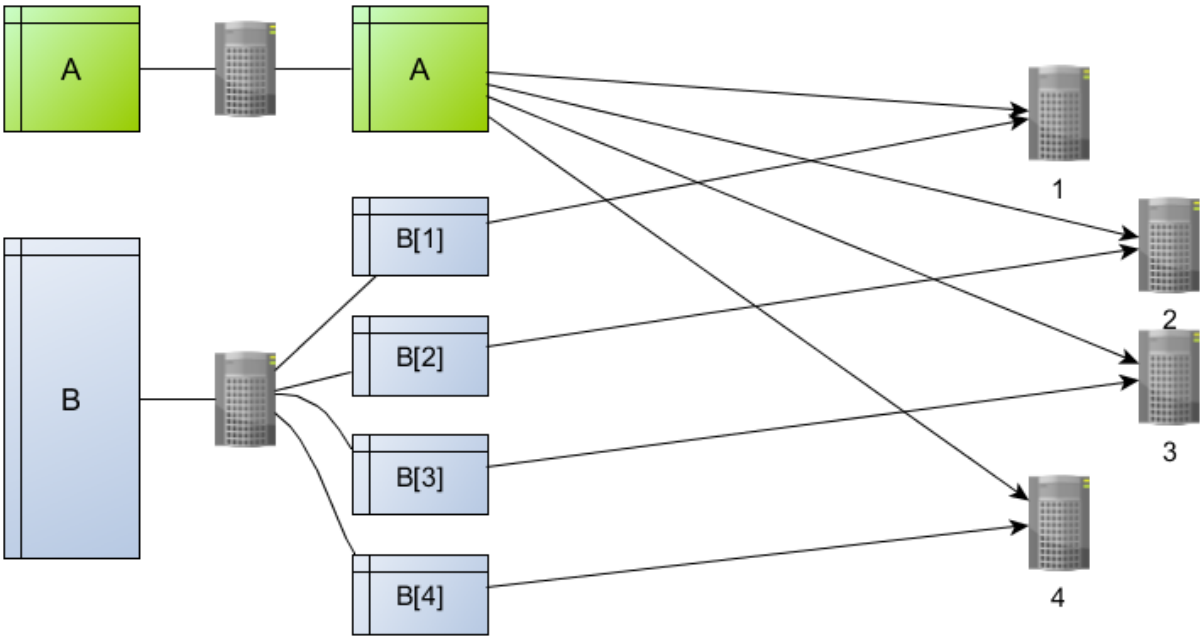
Το μέγεθος των πινάκων, σε τέτοιου είδους συστήματα, είναι συνήθως πάρα πολύ μεγάλο και δεν χωρούν να αποθηκευτούν και να επεξεργαστούν ολόκληροι σε κάποιον κόμβο του συστήματος. Επομένως η απλή JOIN δεν είναι πρακτική. Συμφέρει να γίνει η

HASH JOIN με πολυπλοκότητα $O(M+N)$ ή η MERGE JOIN με πολυπλοκότητα $O(M \cdot Log(M) + N \cdot Log(N))$. Όμως για να γίνουν αυτές θα πρέπει να διαχωριστούν τα δεδομένα ανά τους κόμβους. Συγκεκριμένα, διακρίνουμε τρείς περιπτώσεις.



Σχήμα 3: Και οι δύο πίνακες χωράνε στον κόμβο επεξεργασίας. Και οι δύο πίνακες στέλνονται αυτούσιοι στον κόμβο.

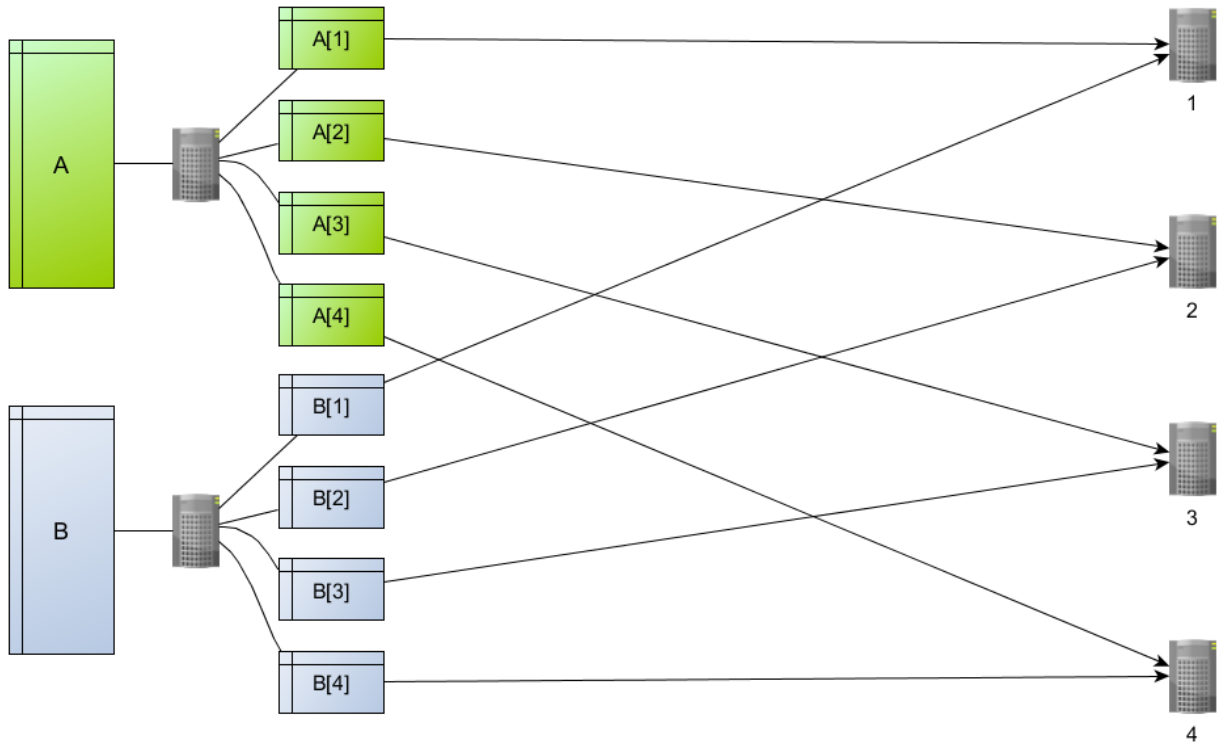
Όπως φαίνεται στο **Σχήμα 3**, ο πίνακας Α και ο πίνακας Β χωράνε να αποθηκευτούν και να επεξεργαστούν σε έναν κόμβο. Επομένως αρκεί να αποσταλούν ολόκληροι σε αυτόν τον κόμβο ώστε να τους επεξεργαστεί.



Σχήμα 4: Μόνο ο ένας πίνακας από τους δύο χωράει στον κόμβο επεξεργασίας. Ο μικρός πίνακας διαμοιράζεται ανά του κόμβους αυτούσιος, ενώ ο μεγάλος σε μέγεθος πίνακας διασπάται σε μικρότερους ώστε να αποσταλούν στους αντίστοιχους κόμβους επεξεργασίας.

Όπως φαίνεται στο **Σχήμα 4**, ο πίνακας Α χωράει ολόκληρος σε έναν κόμβο, όμως ο πίνακας Β δεν χωράει ολόκληρος. Έτσι πρέπει να γίνει διάσπαση του πίνακα Β, σε μικρότερους ώστε να χωρέσουν σε κόμβους επεξεργασίας. Ο πίνακας Α, αρκεί να

αποσταλεί αυτούσιος σε όλους του κόμβους. Έτσι θα γίνει σύγκριση των εγγραφών του πίνακα A με το κάθε τμήμα του πίνακα B (1, 2, …. k).



Σχήμα 5: Και οι δύο πίνακες δεν χωράνε στον κόμβο επεξεργασίας. Και οι δύο πίνακες διασπόνται σε μικρότερους ώστε να αποσταλούν στους αντίστοιχους κόμβους επεξεργασίας.

Όπως φαίνεται στο **Σχήμα 5**, κανένας από τους δύο πίνακες δεν χωράει ολόκληρος σε κάποιον κόμβο επεξεργασίας. Συνεπώς, πρέπει να γίνει διάσπαση και των δύο πινάκων, σε μικρότερους ώστε να χωρέσουν στους κόμβους επεξεργασίας. Έτσι θα γίνει η HASH JOIN όπου σε κάθε κόμβο γίνεται σύγκριση των εγγραφών των κομματιών, των δύο αρχικών πινάκων, που έχουν ληφθεί. Των κομματιών 1,2, ... ,κ του πίνακα A, με τα αντίστοιχα κομμάτια του πίνακα B.

2.2 Το σύστημα του MadIS

Το madIS [6] είναι ένα σύστημα βάσεων δεδομένων, γραμμένο σε Python [7] και η λειτουργία του έχει βασιστεί πάνω στο σύστημα SQLite [8]. Η εσωτερική γλώσσα που χρησιμοποιεί το σύστημα του madIS είναι επέκταση της SQL και ονομάζεται madQL. Αυτό που μας προσφέρει το madIS είναι η δυνατότητα επέκτασης της SQLite με την δημιουργία προσαρμοσμένων τελεστών από τους χρήστες. Οι τελεστές διακρίνονται σε row, aggregate και vtable τύπου τελεστές.

- Οι Row τύπου τελεστές, τροποποιούν τις εγγραφές μία-μία και τις εμφανίζουν στην έξοδο τροποποιημένες (π.χ. ο *UPPER* τελεστής μετατρέπει τις συμβολοσειρές σε κεφαλαία).
- Οι Aggregate τύπου τελεστές παράγουν ένα συνολικό αποτέλεσμα που βασίζεται στο σύνολο των εγγραφών του ερωτήματος εισόδου (π.χ. ο τελεστής *SUM* υπολογίζει το άθροισμα των τιμών του ερωτήματος εισόδου).

- Οι Vtable τύπου τελεστές παράγουν μία ροή δεδομένων (stream) στην μορφή πίνακα βάσης (π.χ. ο *FILE* τελεστής διαβάζει και επιστρέφει σαν πίνακα τα περιεχόμενα ενός αρχείου).

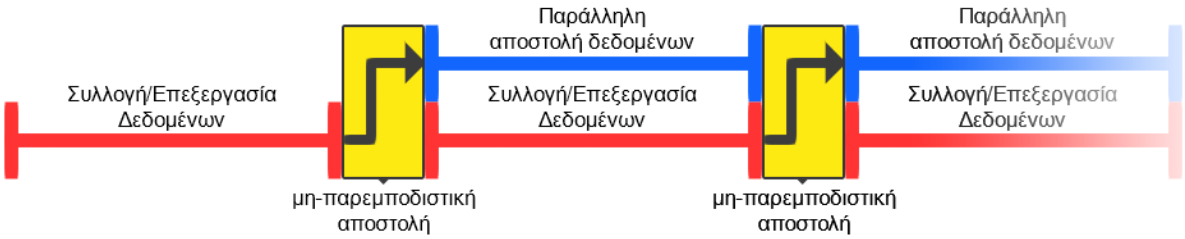
Το madIS χρησιμοποιείται ως ένα εργαλείο του συστήματος Exareme. Το Exareme δουλεύει πάνω από το σύστημα madIS που είναι εγκατεστημένο στους κατανεμημένους κόμβους του συστήματος. Το Exareme έχει την ευθύνη της επικοινωνίας των κόμβων μεταξύ τους και χρησιμοποιεί τους τελεστές του madIS προκειμένου να κάνει την όποια αναλυτική επεξεργασία πάνω στα δεδομένα στον κάθε κόμβο του συστήματος.

3. Ασύγχρονος, μη-παρεμποδιστικός προγραμματισμός (asynchronous, non-blocking)

Οι λειτουργίες αποστολής και λήψης δεδομένων μπορούν να είναι σύγχρονες ή ασύγχρονες. Μία σύγχρονη (synchronous) λειτουργία αποστολής παρεμποδίζει (blocking) μία διεργασία μέχρι να ολοκληρωθεί η αποστολή όλων των δεδομένων προς τον παραλήπτη. Μία ασύγχρονη λειτουργία είναι μη-παρεμποδιστική (non-blocking) και αρχικοποιεί μόνο την αποστολή.

Σε ένα σύγχρονο σύστημα η παραλληλία μπορεί να επιτευχθεί με την δημιουργία ξεχωριστών διεργασιών (forking) για κάθε παράλληλη λειτουργία, αφήνοντας το λειτουργικό σύστημα να συγχρονίσει και να διαχειριστεί τις παράλληλες διεργασίες, καθιστώντας αρκετά εύκολο τον προγραμματισμό ενός παράλληλου προγράμματος. Όμως αυτή η προσέγγιση συνεπάγεται το πρόσθετο κόστος διαχείρισης των επιπλέον διεργασιών και το κόστος της μεταγωγής περιβάλλοντος (context switching) που θα συμβαίνει σε κάθε εναλλαγή των διεργασιών.

Η ασύγχρονη μεταφορά δεδομένων επιτρέπει να υπάρξει παραλληλία μέσα σε μόνο ένα υπολογιστικό νήμα, αφού μια διεργασία που δεν παρεμποδίζεται, μέχρι να ολοκληρωθεί η αποστολή δεδομένων, μπορεί να κάνει παράλληλα κάποια υπολογιστική επεξεργασία. Για παράδειγμα, στην περίπτωση του αποστολέα δεδομένων, αυτό σημαίνει ότι ενώ στέλνονται τα δεδομένα που έχουν προηγουμένως δρομολογηθεί με μη-παρεμποδιστική αποστολή, μπορούν παράλληλα να επεξεργάζονται, να πακετάρονται ή και να συμπιέζονται τα επόμενα δεδομένα για αποστολή, όπως φαίνεται στο **Σχήμα 6**.



Σχήμα 6: Μη-παρεμποδιστική αποστολή δεδομένων.

Με βάση τις δυνατότητες που μας προσφέρει ο ασύγχρονος προγραμματισμός, είναι δυνατόν στον κάθε κόμβο του συστήματος Exareme να απαιτείται μόνο ένα υπολογιστικό νήμα για την αποστολή και λήψη δεδομένων. Έτσι κρατάμε τους απαιτούμενους πόρους τους συστήματος χαμηλά.

Συνέπεια της λειτουργίας ενός αλγόριθμου μέσα σε μόνο ένα υπολογιστικό νήμα είναι ο άμεσος διαμοιρασμός της μνήμης που χρειάζεται, χωρίς να επιβαρύνεται το λειτουργικό σύστημα με κοινόχρηστη μνήμη και τους ελέγχους για την συνέπειά της ανάμεσα στο πολλαπλά υπολογιστικά νήματα με σημαφόρους (semaphores). Έτσι μειώνονται ακόμη περισσότερο οι απαιτούμενοι πόροι.

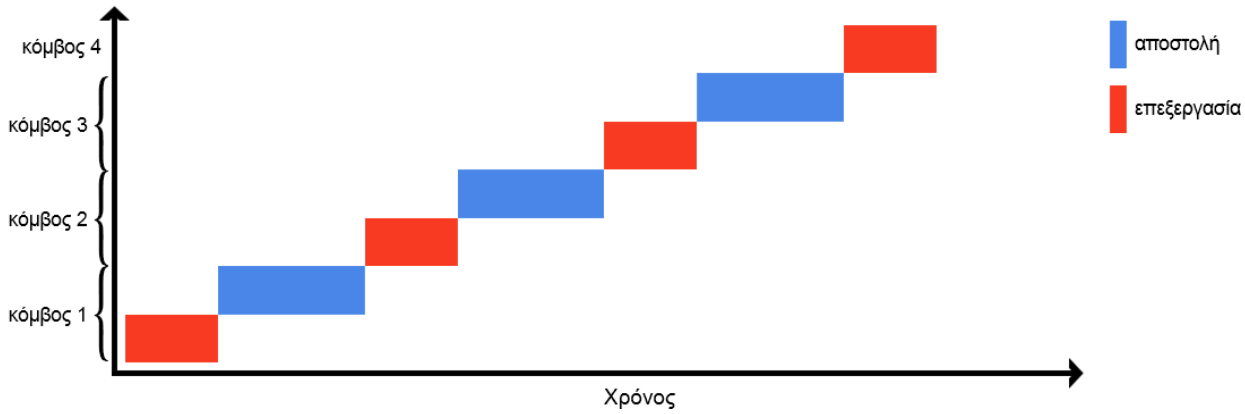
4. Βασικές μέθοδοι μεταφοράς

Υπάρχουν δύο βασικές μέθοδοι για την αποστολή και την λήψη δεδομένων μεταξύ δύο κόμβων. Η μία είναι η μέθοδος αποθήκευση-αποστολή και η άλλη είναι η μέθοδος σωλήνωσης. Η κύρια διαφορά τους είναι στην επιλογή του κύριου αποθηκευτικού μέσου για την προσωρινή αποθήκευση, δηλαδή την ενταμίευση των δεδομένων που πρόκειται να αποσταλούν.

4.1 Μέθοδος αποθήκευση-αποστολή

Η μέθοδος αποθήκευση-αποστολή χρησιμοποιεί σαν κύριο αποθηκευτικό μέσον τον σκληρό δίσκο. Κατά την μέθοδο αυτή, σε κάθε κόμβο, η σειρά που γίνονται τα πράγματα είναι η εξής:

- Παραγωγή/Επεξεργασία δεδομένων και εγγραφή αυτών στον δίσκο σε ένα αρχείο.
- Εκκίνηση αποστολής του αρχείου από τον αποστολέα και αναμονή λήψης του από τον παραλήπτη.



Σχήμα 7: Σχηματική απεικόνιση της σειριακής λειτουργίας της μεθόδου αποθήκευση-αποστολής.

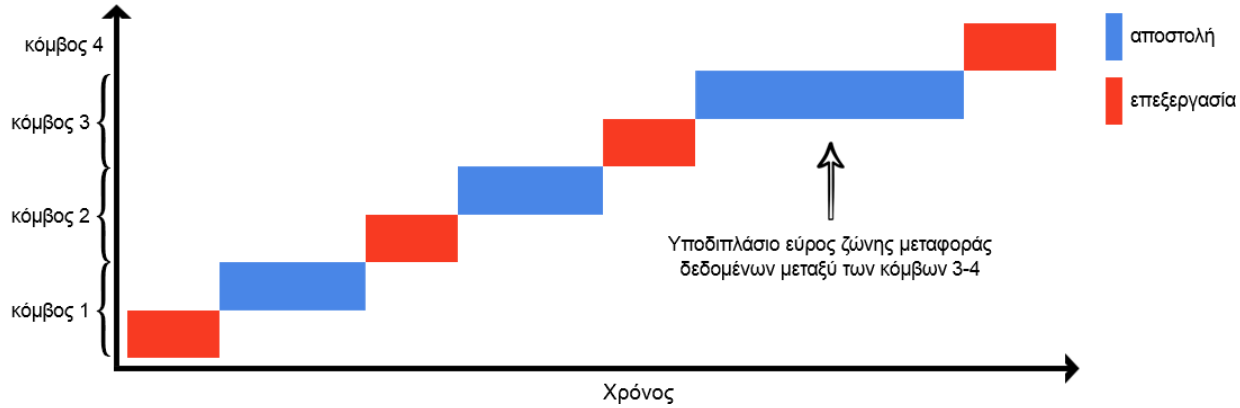
Η μέθοδος αυτή δουλεύει τμηματικά ανά δύο κόμβους. Αφού γίνει η επεξεργασία ή η δημιουργία των δεδομένων από τον αποστολέα, και στην συνέχεια ολοκληρωθεί η αποστολή των δεδομένων προς τον παραλήπτη, ο παραλήπτης γίνεται ο καινούριος αποστολέας, και ούτω καθεξής.

4.1.1 Θετικά στοιχεία της μεθόδου αποθήκευση-αποστολή

Λόγω του ότι η επεξεργασία και η αποστολή γίνεται τμηματικά ανά δύο κόμβους, οι κόμβοι χωρίζονται σε ζεύγη. Το κάθε ζεύγος δεν μπορεί να επηρεάσει ή να επηρεαστεί από τα υπόλοιπα ζεύγη κόμβων του συστήματος. Επομένως, σε ακανόνιστες συνδέσεις δικτύου, όπου από ζεύγος σε ζεύγος το εύρος ζώνης μπορεί να μην είναι πανομοιότυπο, δεν μπορεί να υπάρξει αλληλεξάρτηση μεταξύ των κόμβων του συστήματος που δεν ανήκουν στο ίδιο ζεύγος.

Για παράδειγμα στο **Σχήμα 8** στο τελευταίο ζευγάρι κόμβων 3-4 το εύρος ζώνης αποστολής των δεδομένων είναι υποδιπλάσιο από τα προηγούμενα ζεύγη, όμως δεν θα

επηρεαστούν από αυτό το γεγονός, αφού η διαδικασία ολοκληρώνεται τμηματικά. Ο χρόνος ολοκλήρωσης την συνολικής εργασίας θα είναι το άθροισμα του χρόνου ολοκλήρωσης όλων των τμημάτων ξεχωριστά.



Σχήμα 8: Σχηματική απεικόνιση της τμηματική επεξεργασία και αποστολή ανά ζεύγος κόμβων. Το εύρος ζώνης του δικτύου ανάμεσα στον τρίτο με τον τέταρτο κόμβο είναι υποδιπλάσιο από τους υπόλοιπους κόμβους. Ο συνολικός χρόνος ολοκλήρωσης της εργασίας θα είναι το άθρο

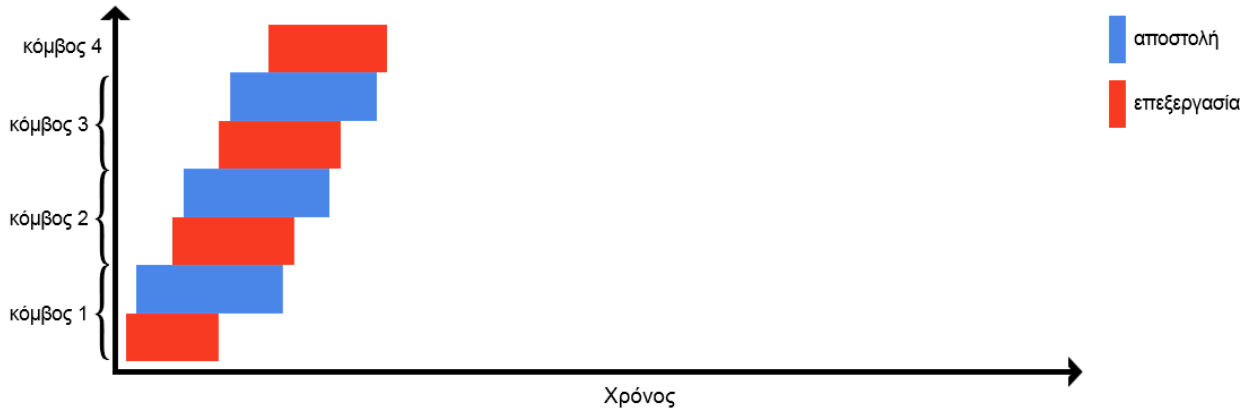
4.1.2 Αρνητικά στοιχεία της μεθόδου αποθήκευση-αποστολή

Λόγω του ότι η επεξεργασία και η αποστολή γίνεται τμηματικά ανά δύο κόμβους, με αυτήν την μέθοδο δεν μπορεί να επικαλυφθεί ο χρόνος αποστολής με χρόνο επεξεργασίας δεδομένων. Για να ξεκινήσει κάποιος κόμβος την επεξεργασία δεδομένων θα πρέπει πρώτα να έχει λάβει όλο το σύνολο των δεδομένων που πρόκειται να επεξεργαστεί. Επομένως δεν μπορεί να υπάρξει παραλληλία. Σε αντίθεση με την μέθοδο σωλήνωσης που το χαρακτηριστικό της παραλληλίας βρίσκεται στα θετικά της στοιχεία.

4.2 Μέθοδος σωλήνωσης

Η μέθοδος σωλήνωσης, ή αλλιώς διοχέτευσης, χρησιμοποιεί σαν κύριο αποθηκευτικό μέσον την κύρια μνήμη του συστήματος. Τα δεδομένα αποθηκεύονται κατά πακέτα στην κύρια μνήμη, στην συνέχεια επεξεργάζονται και προωθούνται ένα-ένα. Δεν απαιτείται να υπάρχει όλο το σύνολο των δεδομένων ώστε να μπορεί να γίνει επεξεργασία δεδομένων. Η επεξεργασία γίνεται σε κάθε ένα πακέτο ξεχωριστά.

Στο σύστημα του Exareme σωλήνωση μπορεί να γίνει όταν χρειάζεται να πραγματοποιηθεί μια σειρά από διαδικασίες επεξεργασίας πάνω σε δεδομένα, όπου η έξοδος της μιας διαδικασία είναι η είσοδος στην επόμενη σε σειρά. Μέσω της μεθόδου σωλήνωσης τα δεδομένα μπορούν να υφίστανται επεξεργασία παράλληλα από τις διάφορες διαδικασίες επεξεργασίας της σωλήνωσης.



Σχήμα 9: Σχηματική απεικόνιση της λειτουργίας της μεθόδου σωλήνωσης. Οι χρόνοι επεξεργασίας και αποστολής μπορούν να επικαλύπτονται λόγω της επεξεργασίας ανά πακέτο δεδομένων.

4.2.1 Θετικά στοιχεία της μεθόδου σωλήνωσης

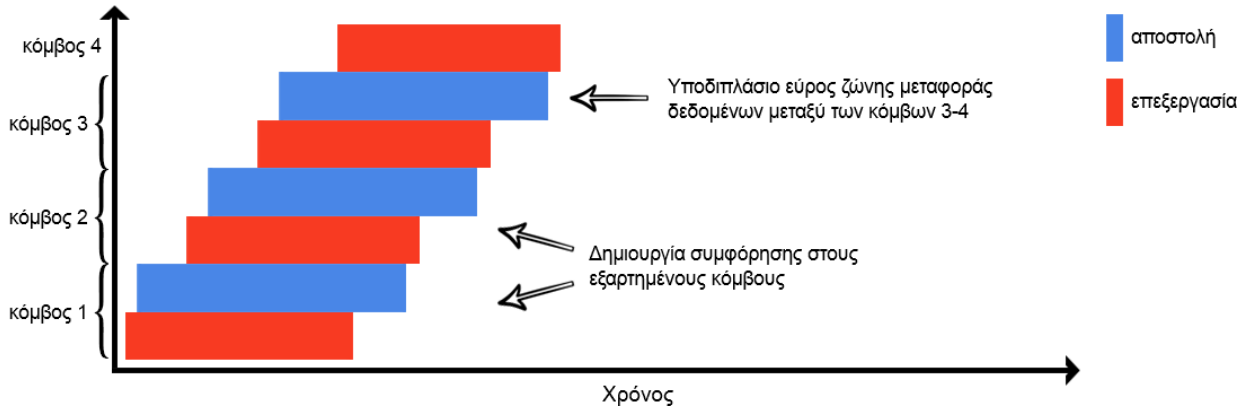
Στα θετικά στοιχεία, βρίσκεται η δυνατότητα επικάλυψης του χρόνου αποστολής με τον χρόνο επεξεργασίας δεδομένων. Επομένως και η ύπαρξη παραλληλίας μεταξύ επεξεργασίας και αποστολής δεδομένων.

Η διαδικασίας της σωλήνωσης δεν μειώνει τον χρόνο του κάθε τμήματος ξεχωριστά, αυξάνει όμως το εύρος ζώνης του συστήματος όταν αυτό επεξεργάζεται ένα συνεχόμενο ρεύμα από δεδομένα.

4.2.2 Αρνητικά στοιχεία της μεθόδου σωλήνωσης

Το αρνητικό στοιχείο αυτής της μεθόδου είναι ότι λόγω του τρόπου λειτουργίας της, υπάρχει μεγάλη εξάρτηση ανάμεσα στους κόμβους του συστήματος. Συγκεκριμένα, επειδή η κύρια μνήμη που μπορεί να χρησιμοποιηθεί για την ενταμιεύεση των δεδομένων είναι περιορισμένη, κάποιος κόμβος που έχει γεμίσει αυτόν τον χώρο δεν μπορεί να λάβει καινούρια δεδομένα. Επομένως θα δημιουργηθεί συμφόρηση σε όλους τους εξαρτημένους από αυτόν κόμβους. Δηλαδή η συμφόρηση μπορεί να παρουσιαστεί όταν οι χρόνοι επεξεργασίας είναι ακανόνιστοι ή το εύρος ζώνης του δικτύου ανάμεσα στους κόμβους είναι ανομοιογενές.

Για παράδειγμα, στο **Σχήμα 10** το εύρος ζώνης της μεταφοράς δεδομένων μεταξύ του κόμβου 3 με 4 είναι υποδιπλάσιο. Επομένως ο κόμβος 3 θα γεμίσει την μνήμη του, αφού τα δεδομένα που καταναλώνει δεν μπορεί να τα στείλει αρκετά γρήγορα στον κόμβο 4. Το ίδιο θα γίνει και στους κόμβους 1 και 2. Έτσι θα δημιουργηθεί συμφόρηση σε όλους τους κόμβους που εξαρτώνται από τον κόμβο 4.



Σχήμα 10: Δημιουργία συμφόρησης στους εξαρτημένους από τον κόμβο 4 κόμβους. Οι κόμβοι είναι σε σωλήνωση σε συνδεσμολογία μονοπατιού.

5. ΜΕΘΟΔΟΣ ΟΠΟΡΤΟΥΝΙΣΤΙΚΗΣ ΣΩΛΗΝΩΣΗΣ

5.1 Εισαγωγή

Στο κεφάλαιο αυτό θα παρουσιάσουμε τις βασικές δυνατότητες που θα έχει η ελαστική μέθοδος μεταφοράς δεδομένων που προτείνουμε, πάνω σε ένα σύστημα σαν το Exareme. Η μέθοδος αυτή θα πρέπει να μπορεί να προσφέρει τις αναγκαίες διαστάσεις ελαστικότητας, να μην είναι απαιτητική σε πόρους αλλά ταυτόχρονα να είναι πιο αποδοτική σε σχέση με τις βασικές μεθόδους.

Επίσης θα μελετήσουμε τις επιπτώσεις των αλλαγών στην απόδοση του συστήματος.

5.2 Πρόταση

Η κύρια ιδέα της μεθόδου οπορτουνιστικής σωλήνωσης είναι η εκμετάλλευση ανά πάσα στιγμή των συνθηκών τόσο του κόμβου αποστολέα, όσο και του δικτύου. Με βάση τις εκάστοτε συνθήκες, θα είναι δυνατόν να επιλέγονται κατά την διάρκεια εκτέλεσης οι κατάλληλες τιμές των παραμέτρων της μεθόδου με σκοπό την μεγιστοποίηση του συνολικού εύρους ζώνης.

Με αυτόν τον τρόπο θα μπορούν να γίνονται οι κατάλληλες αντισταθμιστικές επιλογές μεταξύ υπολογιστικής επεξεργασίας και εύρους ζώνης αποστολής. Επίσης θα μπορεί να υπάρξει ελαστικότητα στην ενταμίευση των δεδομένων, επιλέγοντας δυναμική αποθήκευση στην κύρια μνήμη ή στον σκληρό δίσκο.

5.3 Μετρικές επιλογής παραμέτρων

Οι δύο κύριες μετρικές που θα βοηθούν στις επιλογές των τιμών των παραμέτρων της μεθόδου είναι:

- Ο ρυθμός αποστολής των δεδομένων ως προς τους κόμβους παραλήπτες.
- Ο ρυθμός ενταμίευσης των δεδομένων εισόδου.

5.4 Ελαστικότητα

Η μέθοδός μας μπορεί να προσφέρει δύο διαστάσεις ελαστικότητας. Την ελαστικότητα στην μεταφορά των δεδομένων, όπου μπορούν να γίνουν οι κατάλληλες αντισταθμιστικές επιλογές μεταξύ υπολογιστικής ισχύς και του επιτευκτού εύρους ζώνης αποστολής δεδομένων. Και την ελαστικότητα στην ενταμίευση των δεδομένων, όπου επιλέγεται δυναμικά κατά την εκτέλεση, το πού θα αποθηκευτούν προσωρινώς τα δεδομένα στην ιεραρχία της μνήμης του συστήματος.

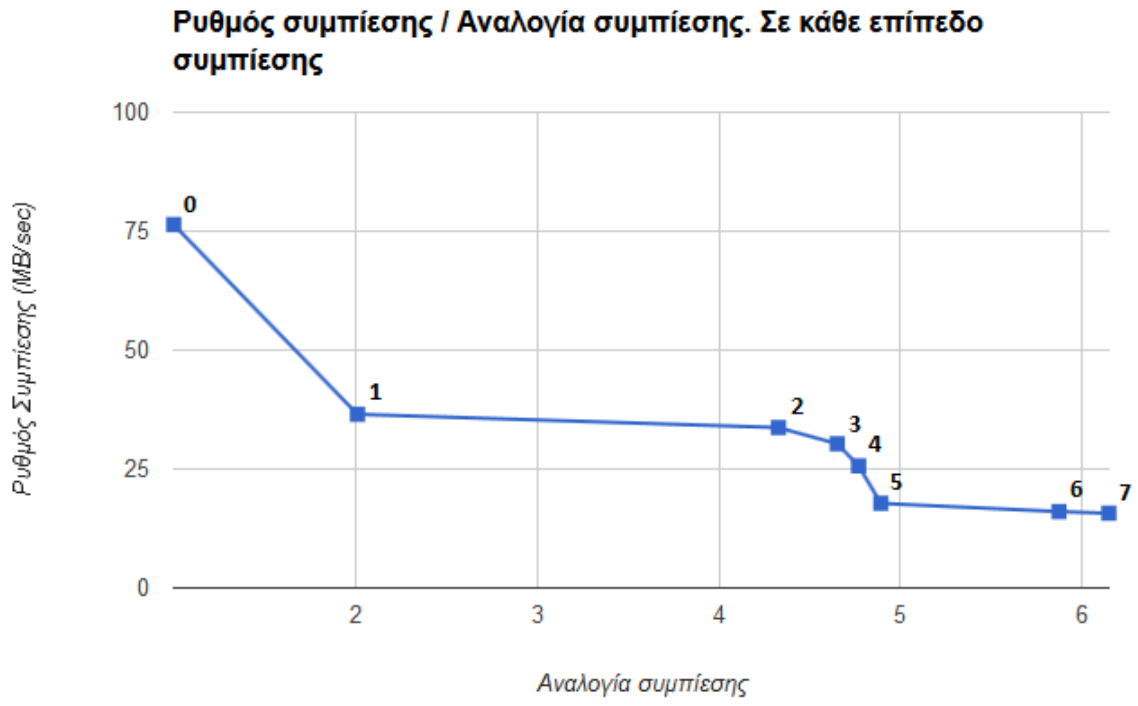
5.4.1 Ελαστικότητα στην μεταφορά δεδομένων

Στα συστήματα υψηλής απόδοσης στο σύντομο μέλλον το κυρίαρχο σημείο συμφόρησης θα είναι η μεταφορά δεδομένων. Αυτό θα γίνει διότι η υπολογιστική επεξεργασία θα είναι λιγότερο απαιτητική σε ενέργεια σε σχέση με την μεταφορά δεδομένων. Προκειμένου μια μέθοδος μεταφοράς δεδομένων να μπορεί να επιτυγχάνει το μεγαλύτερο δυνατό εύρος ζώνης αποστολής δεδομένων θα πρέπει να μπορούν να γίνουν αντισταθμιστικές επιλογές μεταξύ επεξεργαστικής ισχύς και αποστολής δεδομένων, ανάλογα τις συνθήκες που επικρατούν.

Θα πρέπει να μπορούμε να αυξήσουμε το εύρος ζώνης επιβαρύνοντας τον επεξεργαστή, όποτε αυτό χρειάζεται. Χρειαζόμαστε μια λογική που να μας επιτρέπει να το πετυχαίνουμε αυτό δυναμικά, ανάλογα τις συνθήκες του δικτύου και την φύση των δεδομένων. Πρέπει να ορίσουμε την υπολογιστική επεξεργασία που μπορεί να λειτουργήσει ως αντίβαρο του εύρους ζώνης αποστολής δεδομένων.

Για λόγους απόδοσης, τόσο του δικτύου όσο και υπολογιστικής επεξεργασίας, τα δεδομένα που συλλέγονται για αποστολή αποθηκεύονται σε πακέτα δεδομένων όπως και γίνεται στην μέθοδο σωλήνωσης. Τα πακέτα δεδομένων συμπιέζονται με χρήση βιβλιοθηκών που μας προσφέρει το madIS [6]. Ο αλγόριθμος συμπίεσης που χρησιμοποιείται παρέχει πολλαπλά επίπεδα συμπίεσης πάνω σε δεδομένα βάσης.

Τα επίπεδα συμπίεσης είναι κατάλληλα επιλεγμένα, ώστε να παρέχουν τις πιο συμφέρουσες επιλογές ανάμεσα στον ρυθμό συμπίεσης και στην αναλογία συμπίεσης. Στην τωρινή έκδοση του αλγορίθμου υπάρχουν 8 επίπεδα συμπίεσης. Χαρακτηριστικά, το επίπεδο 0, κάνει απλό πακετάρισμα των δεδομένων, το επίπεδο 1 κάνει κατάργηση διπλότυπων δεδομένων, και όσο το επίπεδο συμπίεσης αυξάνεται, βελτιώνεται η αναλογία συμπίεσης των δεδομένων, όμως ανάλογα αυξάνεται ο χρόνος συμπίεσης.



Σχήμα 11: Η καμπύλη, των επιπέδων συμπίεσης του συμπιεστικού αλγορίθμου, σχέσης ρυθμού συμπίεσης με αναλογία συμπίεσης.

Στο **Σχήμα 11** φαίνεται η καμπύλη που σχηματίζεται για τους πίνακες lineitemload του TPC-H [9].

Επομένως τις αντισταθμιστικές επιλογές, μεταξύ επεξεργαστικής ισχύς και αποστολής δεδομένων, μπορούμε να τις ανάγουμε στην επιλογή του επιπέδου συμπίεσης, τέτοιου ώστε να ελαχιστοποιείται ο απαιτούμενος χρόνος που χρειάζεται αθροιστικά η συμπίεση και η αποστολή του κάθε συμπιεσμένου πακέτου δεδομένων.

Ο υπολογισμός του συνολικού χρόνου αποστολής ενός πακέτου ορίζεται ως:

$$C_i = \underbrace{\left(\frac{size}{compBandwidth_i}\right)}_{\text{Χρόνος που χρειάζεται η συμπίεση}} + \underbrace{\left(\frac{size \cdot compRatio_i}{bandwidth}\right)}_{\text{Χρόνος αποστολής του συμπιεσμένου πακέτου}}$$

Σχήμα 12: Μαθηματικός τύπος υπολογισμού του συνολικού χρόνου που χρειάζεται το επίπεδο συμπίεσης i να συμπιέσει ένα πακέτο δεδομένων και να το στείλει με κάποιο εύρος ζώνης.

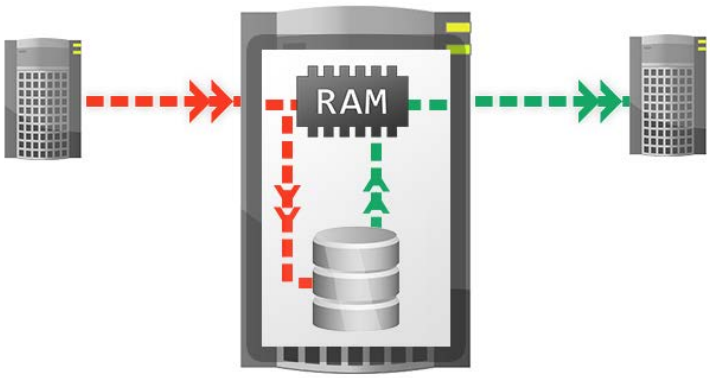
Στο **Σχήμα 12** έχουμε:

- C_i: Κόστος να συμπιέσουμε και να στείλουμε ένα πακέτο με το επίπεδο συμπίεσης i, και δεδομένου εύρους ζώνης αποστολής.
- $size$: Το αρχικό μέγεθος των δεδομένων προς συμπίεση.
- $compRatio_i$: Ο λόγος συμπίεσης του επιπέδου συμπίεσης i.
- $bandwidth$: Το τρέχον εύρος ζώνης αποστολής δεδομένων.
- $compBandwidth_i$: Το εύρος ζώνης συμπίεσης, του επιπέδου συμπίεσης i.

5.4.2 Ελαστικότητα στην ενταμίευση των δεδομένων

Ένας από τους κύριους στόχους της μεθόδου μας είναι η βέλτιστη κατανάλωση των δεδομένων εισόδου με σκοπό την μεγιστοποίηση του συνολικού εύρους ζώνης. Τα δεδομένα εισόδου θα πρέπει να χωριστούν σε πακέτα, να ενταμιευθούν στην μέθοδό μας και έπειτα να αποσταλούν στους παραλήπτες τους.

Η μέθοδος μας, λειτουργώντας αρχικά με την συμπεριφορά της μεθόδου σωλήνωσης, έχοντας ως μέσο αποθήκευσης μόνο την κύρια μνήμη. Δεν θα μπορούν να καταναλωθούν παραπάνω δεδομένα όταν η κύρια μνήμη γεμίσει. Τότε είναι που πρέπει να χρησιμοποιηθεί ο σκληρός δίσκος για την αποθήκευση των παραπάνω δεδομένων, και η συμπεριφορά της μεθόδου να αλλαχθεί σταδιακά σε αυτήν της μεθόδου αποθήκευση-αποστολή ώστε να συνεχιστεί η κατανάλωση των δεδομένων εισόδου. Η μέθοδός μας προσφέρει αυτήν την ελαστικότητα χρήσης της ιεραρχίας της μνήμης.



Σχήμα 13: Μοντέλο λειτουργίας της μεθόδου μας πάνω στην ιεραρχία της μνήμης.

Το μοντέλο της λειτουργίας της μεθόδου ως αναφορά την διαχείριση της ιεραρχίας της μνήμης φαίνεται στο **Σχήμα 13**. Αρχικά ο χρήστης θέτει ένα άνω όριο χρήσης της κύριας μνήμης. Μόλις τα σύνολο των δεδομένων που έχουν ενταμιευθεί στην μέθοδό μας ξεπεράσουν το όριο χώρου που έχει τεθεί, τα παραπάνω δεδομένα θα πρέπει να αποθηκευτούν στον σκληρό δίσκο σε λογική ουράς. Τα παραπάνω δεδομένα θα έρθουν ξανά στην κύρια μνήμη, με την σειρά που γράφτηκαν, ώστε να αποσταλούν όταν θα είναι αναγκαίο.

Όσο τα δεδομένα μπορούν να αποσταλούν με πιο γρήγορο ρυθμό απ' ό,τι ενταμιεύονται, τότε δεν πρόκειται να ξεπεράσουν σε σύνολο το όριο χώρου της κύριας μνήμης που έχει τεθεί, και στην ουσία λειτουργεί η μέθοδος σωλήνωσης, όπου το 100% των δεδομένων αποθηκεύονται στην κύρια μνήμη. Όταν ο ρυθμός αποστολής δεδομένων είναι πιο μικρός από τον ρυθμό κατανάλωσης δεδομένων της εισόδου, τότε κάποια στιγμή τα δεδομένα θα ξεπεράσουν το όριο χώρου της κύριας μνήμης. Τα παραπάνω δεδομένα θα ενταμιευθούν στον σκληρό δίσκο σε λογική ουράς. Αυτές οι συνθήκες αντιστοιχούν στην μέθοδο αποθήκευση-αποστολή.

Τα στάδια αλλαγής της συμπεριφοράς της μεθόδου δεν είναι μόνο τα δύο που αντιστοιχούν στις δύο βασικές μεθόδους. Ανάλογα το εύρος ζώνης του δικτύου και του ρυθμού κατανάλωσης δεδομένων εισόδου, η μέθοδός μας μπορεί να κινηθεί σε όλο το ενδιάμεσο εύρος σταδίων συμπεριφοράς που οριοθετούν η μέθοδος αποθήκευση-αποστολή και η μέθοδος σωλήνωσης. Η ομαλή εναλλαγή συμπεριφοράς της μεθόδου φαίνεται καθαρά στην περίπτωση όπου ο ρυθμός αποστολής δεδομένων ξεπεράσει τον ρυθμό κατανάλωσης δεδομένων εισόδου, μετά από ένα μεγάλο χρονικό διάστημα που συνέβαινε το αντίθετο. Τα καινούρια δεδομένα που καταναλώνονται από την είσοδο δεν θα αποθηκεύονται απευθείας στην κύρια μνήμη, αφού θα πρέπει πρώτα να καταναλωθούν τα δεδομένα που είναι αποθηκευμένα σε ουρά στον σκληρό δίσκο. Θα πρέπει σιγά σιγά να καταναλωθούν τα δεδομένα από τον δίσκο και να γίνει η μετάβαση προς την συμπεριφορά της μεθόδου σωλήνωσης ομαλά.

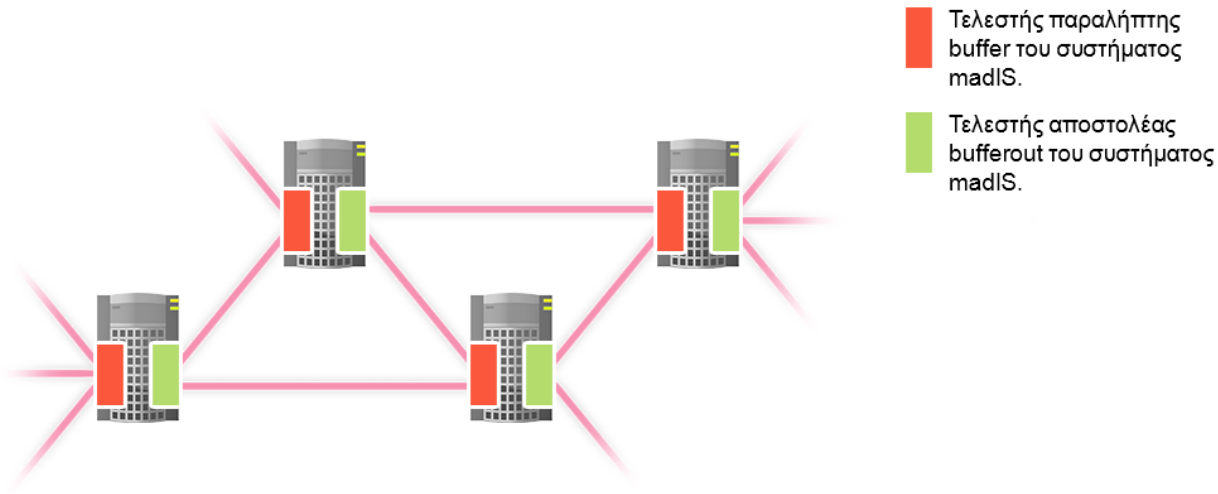
Συμπερασματικά, μέσω της ελαστικότητας χρήσης της ιεραρχίας της μνήμης του συστήματος μπορούμε να έχουμε και τις δύο βασικές μεθόδους σε μία. Με αυτόν τον τρόπο μπορούμε να κρατήσουμε τα θετικά στοιχεία και των δύο, και ταυτόχρονα να απαλειφθούν τα αρνητικά, αφού τα θετικά της μίας είναι ταυτόχρονα τα αρνητικά της άλλης.

5.5 Λεπτομέρειες υλοποίησης

5.5.1 Λειτουργία στο σύστημα Exareme

Η μέθοδός μας αποτελείται από δύο τελεστές VTABLE του συστήματος madIS.

- Τον τελεστή του αποστολέα, BUFFEROUT.
- Τον τελεστή του παραλήπτη, BUFFER.



Σχήμα 14: Σχηματική απεικόνιση ενός τμήματος του συστήματος Exareme όπου στον κάθε κόμβο είναι εγκατεστημένο το σύστημα madIS όπου βρίσκονται και οι δύο τελεστές της μεθόδου μας.

Το Exareme έχει υπό την εποπτεία του όλους τους κόμβους που διαθέτουν το σύστημα MadIS και έχει την ευθύνη επικοινωνίας των κόμβων μεταξύ τους. Το Exareme δίνει την δυνατότητα στο MadIS να επικοινωνήσει με τους άλλους κόμβους του συστήματος δίνοντάς πρόσβαση σε σωληνώσεις (pipes) του λειτουργικού συστήματος, που έχουν δημιουργηθεί αποκλειστικά για μεταφορά δεδομένων μεταξύ των κόμβων. Συγκεκριμένα, ο τελεστής αποστολέας θα έχει δικαίωμα εγγραφής στην σωλήνωση αποστολής, ενώ ο τελεστής παραλήπτης θα έχει δικαίωμα ανάγνωσης της αντίστοιχης σωλήνωσης για λήψη δεδομένων, όπως φαίνεται στο **Σχήμα 7**.

Σχήμα 15: Απεικόνιση τρόπου επικοινωνίας μεταξύ των κόμβων του συστήματος.

5.5.2 Λειτουργία στο σύστημα MadIS

Το MadIS δίνει τα δεδομένα του αποτελέσματος ενός ερωτήματος στον κάθε τελεστή, παρέχοντας του μια κλάση που τα περιέχει. Η κλάση αυτή δίνει την δυνατότητα στον

τελεστή να πάρει τα δεδομένα με την μέθοδο next(). Μέσω της μεθόδου next() ο τελεστής μπορεί να πάρει μία-μία τις εγγραφές (rows) του πίνακα του αποτελέσματος. Ο τελεστής θα πρέπει να καλέσει την μέθοδο next() τόσες φορές, όσες είναι και οι εγγραφές του πίνακα του αποτελέσματος.

5.5.3 Τρόπος κλήσης και ορίσματα

Ο τελεστής του αποστολέα καλείται ως εξής:

```
bufferout 'pipe_name' Parameters Query;
```

Όπου pipe_name είναι το όνομα της σωλήνωσης που θα μπορούν να γράφονται τα δεδομένα. Parameters είναι οι θέση όπου μπορούν να γραφτούν οι παράμετροι που αναφέρονται παρακάτω. Query είναι το ερώτημα σε γλώσσα madQL που θα παράγει τα δεδομένα εισόδου του τελεστή αποστολέα.

Οι παράμετροι που προσφέρει ο τελεστής του αποστολέα είναι:

- mode:[broadcast, split]. Με αυτήν την παράμετρο επιλέγεται η λειτουργία του αποστολέα, διαμοιρασμού ή διάσπασης αντίστοιχα. Η παράμετρος είναι προαιρετική. Ως προεπιλογή είναι η λειτουργία διάσπασης.
- split:[αριθμός]. Με αυτήν την παράμετρο επιλέγεται ο αριθμός των κόμβων παραλιπτών.
- comp_level:[αριθμός]. Με αυτήν την παράμετρο επιλέγεται το σταθερό επίπεδο συμπίεσης που θα χρησιμοποιηθεί κατά όλη την διάρκεια λειτουργία της μεθόδου. Αν δεν υπάρξει αυτή η παράμετρος θα επιλέγεται κατά την διάρκεια εκτέλεσης το βέλτιστο επίπεδο συμπίεσης για την μεγιστοποίηση του εύρους ζώνης.
- hms:[αριθμός]. Με αυτήν την παράμετρο ο χρήστης μπορεί να θέσει το άνω όριο χώρου της κύριας μνήμης που μπορεί να χρησιμοποιηθεί.

Ο τελεστής του παραλήπτη καλείται ως εξής:

```
buffer 'pipe_name_1' 'pipe_name_2'… Parameters;
```

Όπου πρώτα μπορούν να γραφτούν τα ονόματα των σωληνώσεων που θα μπορεί να γίνει η ανάγνωση των δεδομένων. Έπειτα μπορούν να γραφτούν οι παράμετροι της μεθόδου:

- file:[τοποθεσία αρχείου]. Με αυτήν την παράμετρο επιλέγεται το αρχείο βάσης που θα γίνεται η εγγραφή των δεδομένων που λαμβάνονται. Η παράμετρος είναι προαιρετική.
- ratio:[αριθμός]. Με αυτήν την παράμετρο δίνεται η επιλογή οριοθέτησης του εύρους ζώνης λήψης. Η παράμετρος είναι προαιρετική.

5.5.4 Πρωτόκολλο επικοινωνίας

Το πρωτόκολλο επικοινωνίας μεταξύ των δύο τελεστών ορίζεται ως εξής:

- Ο αποστολέας στέλνει πακέτα δεδομένων στον παραλήπτη.
- Πριν σταλεί το κάθε πακέτο δεδομένων, στέλνεται η τιμή του μεγέθους του μέσω ενός μικρού πακέτου μεταδεδομένων (metadata) προκαθορισμένου μεγέθους.
- Ως δήλωση του αποστολέα, ότι δεν έχει να στείλει άλλα πακέτα, στέλνεται πακέτο μεταδεδομένων με τιμή μηδέν.

5.5.5 Πολιτική χρήσης της κύριας μνήμης

Ο χρήστης μπορεί να θέσει ένα άνω όριο χώρου χρήσης της κύριας μνήμης. Όταν ο χώρος αυτός εξαντληθεί θα πρέπει να επιλεχθούν πολύ προσεκτικά ποια πακέτα δεδομένων θα βρίσκονται στην κύρια μνήμη, εφόσον δεν υπάρχει η δυνατότητα να υπάρχουν όλα. Η ανάγκη αυτή γίνεται ιδιαίτερα εμφανής στην λειτουργία διαμοιρασμού, όπου ο αποστολέας πρέπει να στείλει ακριβώς τα ίδια πακέτα δεδομένων σε όλους τους παραλήπτες, όμως την κάθε χρονική στιγμή οι παραλήπτες ενδέχεται να έχουν ανάγκη λήψης πακέτων που είναι αρκετά μακρινά από εκείνα τα οποία λαμβάνουν οι υπόλοιποι παραλήπτες. Χρειάζεται να υπάρξει ξεχωριστή πολιτική για τις δύο λειτουργίες μεταφοράς δεδομένων.

5.5.5.1 Λειτουργία διαμοιρασμού

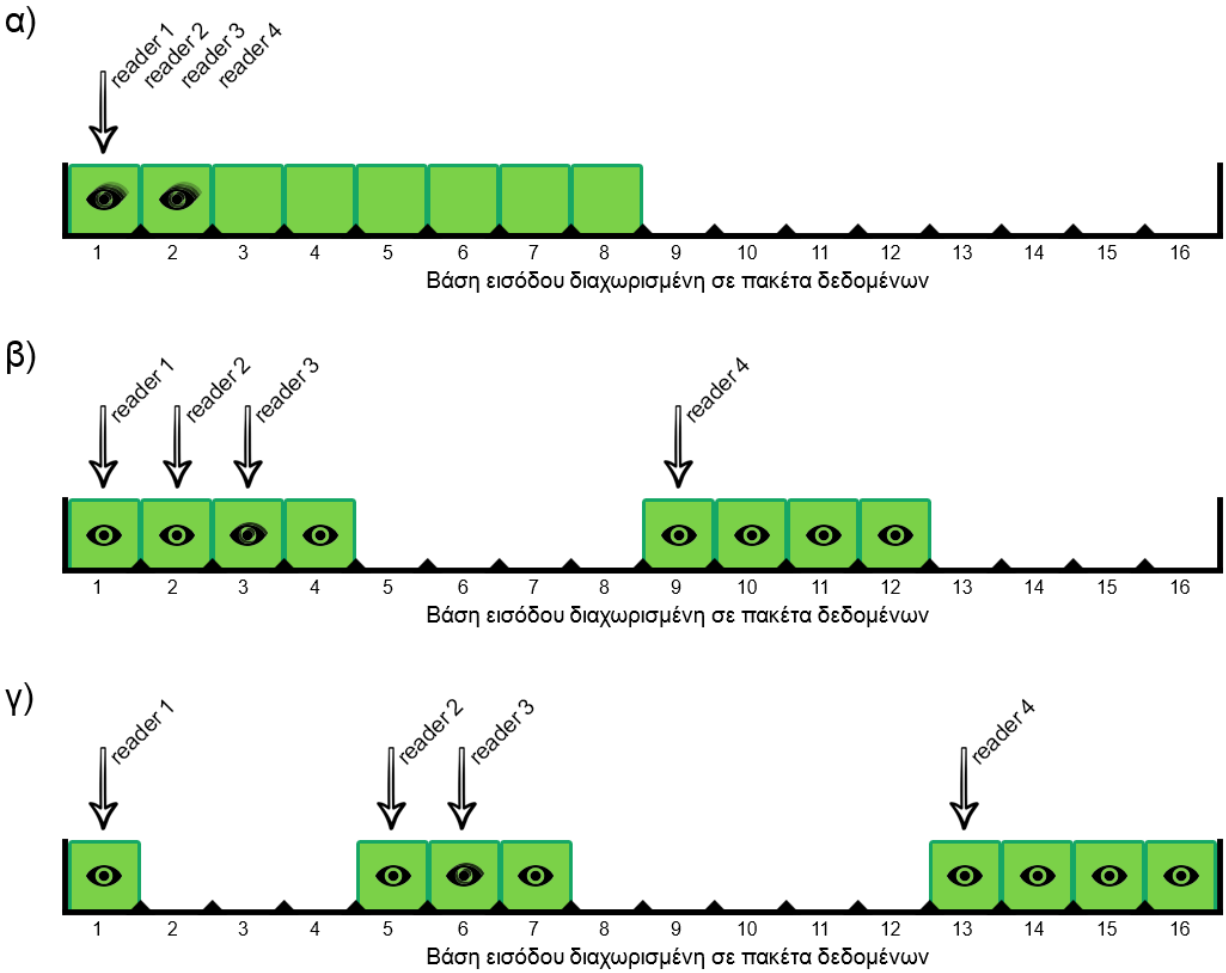
Κατά την λειτουργία διαμοιρασμού, τα δεδομένα που στέλνονται είναι τα ίδια για όλους τους παραλήπτες. Επομένως τόσο η κύρια μνήμη όσο και ο σκληρός δίσκος θα χρησιμοποιείται από κοινού για όλους τους παραλήπτες. Μπορούμε να ονομάσουμε τα πακέτα δεδομένων με έναν αύξων αριθμό σύμφωνα με την σειρά που παράγονται. Όλα τα πακέτα δεδομένων θα γράφονται εξαρχής στον σκληρό δίσκο με αυτήν την σειρά. Στην κύρια μνήμη θα βρίσκονται το πολύ όσα πακέτα δεδομένων χωράνε σε αυτήν.

Στην ιδανική περίπτωση, όλοι οι παραλήπτες θα λαμβάνουν δεδομένα με τον ίδιο ρυθμό λήψης. Επομένως, όλοι οι παραλήπτες θα διαβάζουν ένα, κοινό σε όλους, πακέτο δεδομένων κάθε φορά. Σε αυτήν την περίπτωση πάντα θα υπάρχει χώρος στην κύρια μνήμη ώστε να χωρέσει το ένα πακέτο που διαβάζεται, και μάλιστα περισσεύει χώρος και για επόμενα πακέτα ώστε να αρχίσουν να λαμβάνονται απευθείας μόλις χρειαστούν.

Στην πραγματικότητα όμως, οι παραλήπτες πιθανότατα θα έχουν διαφορετικό ρυθμό λήψης μεταξύ τους. Επομένως κάποιοι θα λαμβάνουν πιο γρήγορα τα δεδομένα και θα αφήσουν πίσω τους υπόλοιπους παραλήπτες. Έτσι σαν εύκολη λύση θα χρειαστεί ένα παράθυρο πακέτων που βρίσκονται στην κύρια μνήμη, τέτοιο ώστε όλα τα πακέτα που λαμβάνουν οι παραλήπτες να βρίσκονται μέσα σε αυτό. Αυτό όμως μπορεί να περιορίσει τους αρκετά γρήγορους παραλήπτες που θα έχουν ανάγκη για λήψη πακέτα που βρίσκονται έξω από το περιορισμένο παράθυρο στην μνήμη. Αν η απόστασή ανάμεσα στους ταχύρρυθμους παραλήπτες και τους αργούς είναι μεγάλη, τότε δεν θα μπορεί να υπάρχει ένα παράθυρο πακέτων που να βρίσκονται μέσα όλα τα πακέτα που διαβάζουν όλοι οι παραλήπτες γιατί η κύρια μνήμη δεν θα τα χωράει. Επομένως θα πρέπει να υπάρξει μία λογική που να αντιστοιχεί διαφορετικό χώρο της κύριας μνήμης στον κάθε παραλήπτη, έτσι ώστε να μην χρειάζεται να υπάρχουν πακέτα στην κύρια μνήμη, που δεν θα αποσταλούν άμεσα. Με αυτόν τον τρόπο οι ταχύρρυθμοι παραλήπτες θα μπορούν να λαμβάνουν πακέτα που βρίσκονται αρκετά μακριά από τους υπόλοιπους παραλήπτες. Επίσης, με αυτήν την λογική θα υπάρχει κάποια βοήθεια για τους ταχύρρυθμους παραλήπτες ώστε να λάβουν τα δεδομένα τους όσο πιο γρήγορα γίνεται.

Η λογική αποτελείται από τους εξής κανόνες:

- Όλα τα πακέτα που δημιουργούνται αποθηκεύονται πάντα στον σκληρό δίσκο, άσχετα με την κύρια μνήμη.
- Η κύρια μνήμη είναι δομημένη έτσι ώστε όλα τα πακέτα δεδομένων να βρίσκονται με την σειρά την οποία κατασκευάζονται. Έχει την μορφή πίνακα. Αν το όριο χρήσης της κύριας μνήμης που έχει τεθεί δεν επιτρέπει την ύπαρξη πακέτων σε κάποιες θέσεις της κύριας μνήμης, τότε οι θέσεις αυτές θα υπάρχουν με κενά δεδομένα (null).
- Κατά την εκκίνηση, όλα τα πακέτα δεδομένων που κατασκευάζονται, αποθηκεύονται και στην κύρια μνήμη μέχρι να συμπληρωθεί το όριο χρήσης της κύριας μνήμης, όπως φαίνεται στο **Σχήμα 72 α)**.
- Ο κάθε παραλήπτης έχει έναν προσωπικό μεταβλητό αριθμό πακέτων που μπορεί να “δει”, δηλαδή που του αντιστοιχούν. Αρχικά, όλοι οι παραλήπτες ξεκινούν να λαμβάνουν το πρώτο πακέτο δεδομένων. Επομένως δεν είναι γνωστός ο ρυθμός λήψης του καθενός ακόμα. Η κύρια μνήμη θα ισοκατανεμηθεί σε όλους τους παραλήπτες. Όλοι παραλήπτες θα “βλέπουν” τον ίδιο αριθμό από πακέτα. Όπως στο **Σχήμα 72 α)** όπου το όριο πακέτων που χωράει η κύρια μνήμη είναι 8.
- Ο αριθμός των πακέτων που “βλέπει” ο κάθε παραλήπτης είναι τέτοιος ώστε τα δεδομένα τους να μην ξεπερνούν σε σύνολο τα bytes που αντιστοιχούν στον συγκεκριμένο παραλήπτη, που είναι κάποιο μέρος του συνολικού ορίου χώρου χρήσης της κύριας μνήμης.
- Το ποσοστό των bytes που αντιστοιχούν σε κάποιον παραλήπτη σε σχέση με το συνολικό όριο χρήσης της κύριας μνήμης, ισούται με το ποσοστό του ρυθμού λήψης του παραλήπτη σε σχέση με τον συνολικό ρυθμό αποστολής δεδομένων από τον αποστολέα. Αυτό θα έχει σημασία αφού γνωστοποιηθεί ο ρυθμός λήψης του κάθε παραλήπτη. Αυτό φαίνεται στο **Σχήμα 72 β), γ)** όπου αντιστοιχούν διαφορετικά μεγέθη μνήμης στον κάθε παραλήπτη, που είναι ανάλογα του ρυθμού λήψης τους.
- Τα πακέτα μεταφέρονται από τον σκληρό δίσκο στην κύρια μνήμη, μόνο όταν το πακέτο που χρειάζεται να λάβει κάποιος παραλήπτης δεν υπάρχει στην κύρια μνήμη. Συγκεκριμένα μεταφέρονται μαζικά όσα πακέτα “βλέπει” ο συγκεκριμένος παραλήπτης εκείνη την χρονική στιγμή ώστε να μειωθούν οι αναγνώσεις του δίσκου. Αυτό φαίνεται στο **Σχήμα 72 β)** όπου ο παραλήπτης 4 χρειάζεται να διαβάσει το πακέτο 9.
- Προκειμένου να περιέχονται στην κύρια μνήμη τουλάχιστον ο αριθμός τον πακέτων που βλέπουν όλοι οι παραλήπτες, θα χρειάζεται να απελευθερώνονται κάποια πακέτα που υπάρχουν από πριν στην κύρια μνήμη. Τα πακέτα που θα απελευθερώνονται είναι αυτά που δεν τα βλέπει κανείς παραλήπτης, και συγκεκριμένα αυτά τα οποία έχουν “ιδωθεί” λιγότερο πρόσφατα. Αυτό φαίνεται στο **Σχήμα 72 β)** όπου για να φορτωθούν τα πακέτα 9-12 θα πρέπει να απελευθερωθεί κάποια μνήμη και εφόσον δεν βλέπει κανείς 5-8, θα απελευθερωθούν αυτά.



Σχήμα 16: Απεικόνιση πολιτικής χρήσης της κύριας μνήμης κατά την λειτουργία διαμοιρασμού. α) Αρχικά όλοι οι παραλήπτες θα πρέπει να λάβουν το πρώτο πακέτο δεδομένων, όλοι οι παραλήπτες "βλέπουν τον ίδιο αριθμό από πακέτα". β) Γνώση των ρυθμών λήψης του κάθε παραλ παραλήπτη, αλλάζοντας έτσι τον αριθμό των πακέτων που "βλέπουν". γ) Χρονική εξέλιξη του β), παρατήρηση του πως ζητούνται τα νέα πακέτα από τον δίσκο.

5.5.5.2 Λειτουργία διάσπασης

Κατά την λειτουργία διάσπασης, η πρώτη στήλη των δεδομένων εισόδου χρησιμοποιείται σαν συνάρτηση κατακερματισμού ώστε η κάθε εγγραφή των δεδομένων να σταλθεί στον κατάλληλο παραλήπτη. Επομένως ο κάθε παραλήπτης έχει αποκλειστικά τα δικά του δεδομένα να παραλάβει. Αυτή η ανεξαρτησία μας επιτρέπει μια εύκολα διαχειρίσιμη πολιτική χρήσης της κύριας μνήμης.

- Για τον κάθε παραλήπτη θα υπάρχουν ξεχωριστές δομές στην κύρια μνήμη.
- Για τον κάθε παραλήπτη θα υπάρχει ένα αρχείο στον σκληρό δίσκο ώστε να ενταμιεύονται τα παραπάνω δεδομένα.
- Όταν ένα παραγόμενο πακέτο δεδομένων που αντιστοιχεί σε κάποιον παραλήπτη δεν χωράει να αποθηκευτεί στην κύρια μνήμη, θα αποθηκευτεί στο αντίστοιχο αρχείο στον δίσκο με λογική ουράς.
- Τα πακέτα δεδομένων κάποιου παραλήπτη μεταφέρονται από τον δίσκο στην κύρια μνήμη μόνο όταν έχουν διαβαστεί όλα τα πακέτα που βρίσκονται στην

κύρια μνήμη. Θα μεταφέρονται τόσα πακέτα ώστε να γεμίσει πάλι πλήρως η κύρια μνήμη.

5.5.6 Επικάλυψη χρόνου αποστολής

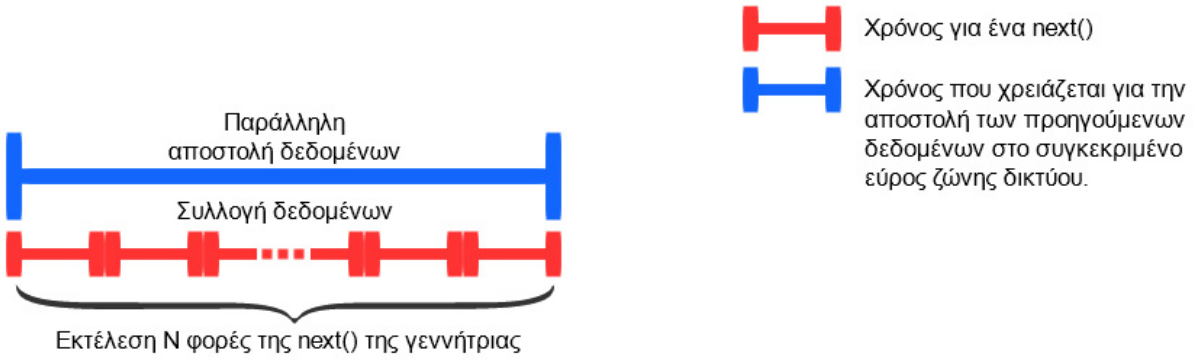
Όταν τα δεδομένα στέλνονται ασύγχρονα μέσω σωληνώσεων δεν στέλνονται απευθείας, αλλά ενταμιεύονται προσωρινά σε έναν χώρο που τον διαχειρίζεται το λειτουργικό σύστημα. Το λειτουργικό σύστημα, προκειμένου οι εφαρμογές να μην μπορούν να υπερχειλίσουν την μνήμη του, έχει θέσει ένα άνω όριο χώρου για την κάθε σωλήνωση που δημιουργείται. Η εγγραφή δεδομένων στις σωληνώσεις που λειτουργούν μη-παρεμποδιστικά γίνεται με την συνάρτηση write(). Η συνάρτηση write() επιστρέφει αμέσως μετά την κλήση της και επιστρέφει σαν τιμή τον αριθμό των bytes που μπόρεσαν να ενταμιευτούν στο λειτουργικό σύστημα. Αν ο χώρος ενταμίευσης του λειτουργικού συστήματος γεμίσει, τότε η συνάρτηση write() επιστρέφει την τιμή 0 χωρίς να καταφέρει να γράψει δεδομένα. Θα πρέπει να καταναλωθεί τουλάχιστον κάποιο μέρος των δεδομένων που έχουν ενταμιευθεί στην σωλήνωση ώστε να είναι δυνατόν να ενταμιευθούν καινούρια δεδομένα ξανά.

Προκειμένου να γνωρίζουμε πότε υπάρχει ελεύθερος χώρος για ενταμίευση δεδομένων σε κάποια σωλήνωση πρέπει να χρησιμοποιηθεί η λειτουργία select του συστήματος UNIX. Η λειτουργία select του συστήματος UNIX, κάνοντας γνωστοποίηση στην κλάση της τις σωληνώσεις του λειτουργικού συστήματος που μας ενδιαφέρουν, μας δίνει την δυνατότητα να γνωρίζουμε ποιες σωληνώσεις από αυτές διαθέτουν ελεύθερο χώρο για εγγραφή δεδομένων ανά πάσα στιγμή.

Ο χρόνος που τα δεδομένα θα παραμείνουν ενταμιευμένα στο λειτουργικό σύστημα εξαρτάται από το εύρος ζώνης του δικτύου. Δηλαδή ισούται με τον χρόνο που χρειάζεται μέχρι να αποσταλούν όλα τα δεδομένα στον παραλήπτη. Επομένως μπορούμε να προβλέψουμε το χρονικό διάστημα που αναμένεται να αδειάσει ο χώρος του λειτουργικού, εάν γνωρίζουμε το εύρος ζώνης αποστολής και το μέγεθος των δεδομένων που είναι ενταμιευμένα στο λειτουργικό σύστημα.

Εφόσον προβλέψουμε τον χρόνο που θα χρειαστεί μέχρι να καταναλωθούν όλα τα δεδομένα μιας σωλήνωσης, μπορούμε να τον επικαλύψουμε με κάτι χρήσιμο πέρα από την αναμονή. Θα πρέπει να οριστεί η υπολογιστική επεξεργασία εκείνη, που να είναι μικρή, τετριμμένη και επαναλαμβανόμενη η οποία θα πρέπει ευέλικτα να επικαλύψει τον χρόνο που γίνεται η αποστολή δεδομένων, ανεξάρτητα με την διάρκειά του. Επίσης, είναι επιθυμητό να καταναλώνονται τα δεδομένα εισόδου με βέλτιστο τρόπο από την μέθοδό μας.

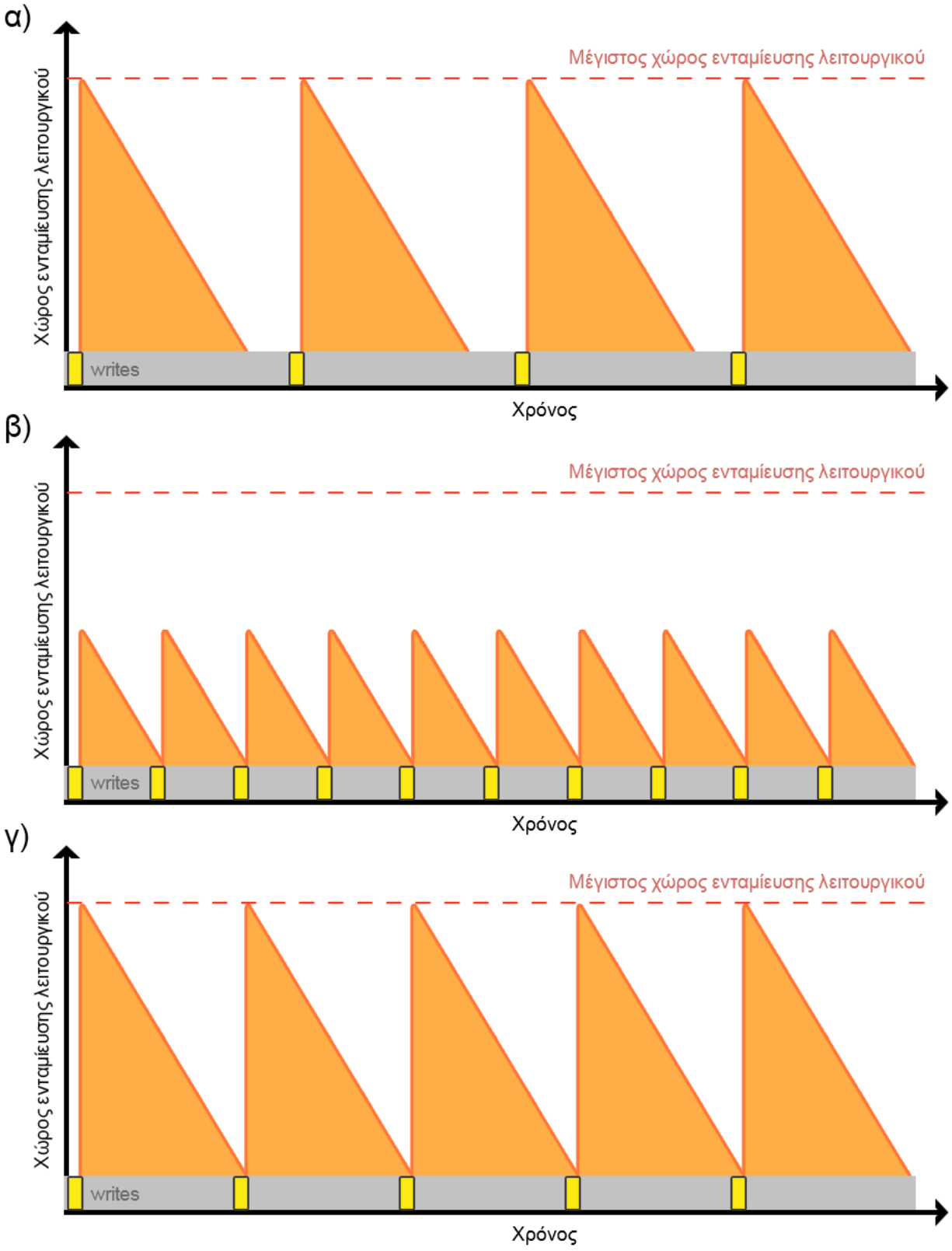
Το MadIS παρέχει στους τελεστές του μία κλάση που περιέχει τα δεδομένα εισόδου και μας παρέχει την μέθοδο next() για να τα συλλέγει η μέθοδός μας. Η κάθε κλήση της μεθόδου next(), όπως είναι αναμενόμενο, κοστίζει υπολογιστικά. Όμως είναι αναγκαίο να την καλέσουμε ακριβώς τόσες φορές όσες είναι και οι εγγραφές των δεδομένων εισόδου. Έτσι μετρώντας τον χρόνο που χρειάζεται η κάθε κλήση της μεθόδου next(), μπορούμε να καλέσουμε τόσες φορές την next() ώστε να επικαλυφθεί ο χρόνος που έχει προβλεφθεί ότι θα ολοκληρωθεί η αποστολή των δεδομένων που έχουν ενταμιευθεί στο λειτουργικό σύστημα.



Σχήμα 17: Κλήσεις της συνάρτησης next() προκειμένου να συμπληρωθεί ο χρόνος αποστολής δεδομένων.

Είναι πολύ σημαντικό να καλούμε τον ακριβή αριθμό των next() που χρειάζονται, ώστε να πετύχουμε ακριβώς την στιγμή που θα καταναλωθούν τα δεδομένα που είναι ενταμιευμένα στο λειτουργικό σύστημα. Έτσι ώστε να ξανά ενταμιευθούν άμεσα καινούρια δεδομένα.

Όπως φαίνεται στο **Σχήμα 18** α) αν καλούνται αργά οι write() ώστε να ενταμιευθούν καινούρια δεδομένα στις σωληνώσεις τότε θα υπάρχει χρόνος που δεν θα καταναλώνονται δεδομένα από τον παραλήπτη. Στο β) αν καλούνται οι write() πολύ γρήγορα τότε δεν θα εκμεταλλευτεί όλος ο χώρος ενταμίευσης που μας παρέχει το λειτουργικό σύστημα επίσης θα κληθούν πολλές παραπάνω write() με το κόστος των κλήσεων συναρτήσεων (callbacks). Επομένως πρέπει να υπολογίζεται ακριβώς η στιγμή που καταναλώνονται τα ενταμιευμένα δεδομένα του λειτουργικού όπως στο γ).

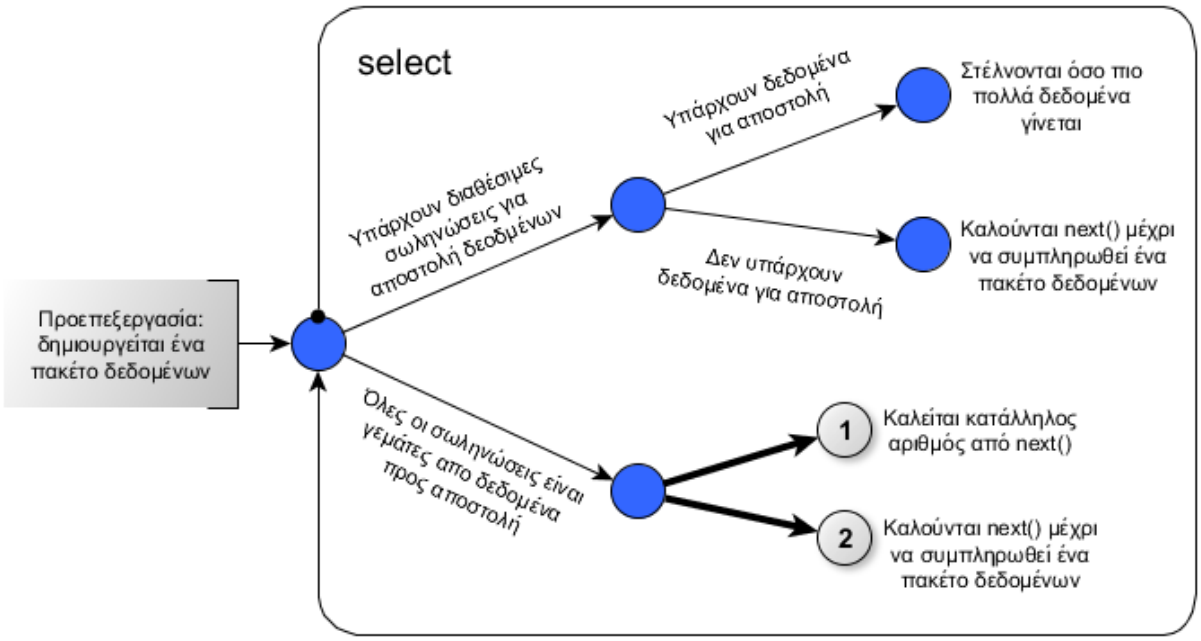


Σχήμα 18: Περιπτώσεις συγχρονισμού των write() με το άδειασμα του χώρου ενταμίευσης του λειτουργικού συστήματος. α) Τα write() καλούνται πολύ γρήγορα. β) Τα write() καλούνται ακριβώς την σωστή στιγμή. γ) Τα write() καλούνται καθηστερημένα.

5.5.7 Μηχανή καταστάσεων της μεθόδου

Ο τελεστής του αποστολέα της μεθόδου θα χρειαστεί μια μηχανή καταστάσεων προκειμένου να μπορεί να αποφασίζει πώς θα κινηθεί με βάση τις συνθήκες που επικρατούν στον κόμβο που εκτελείται αλλά και στο δίκτυο. Οι δύο λειτουργίες που υποστηρίζει η μέθοδός μας θέλουν ξεχωριστή μεταχείριση και λογική.

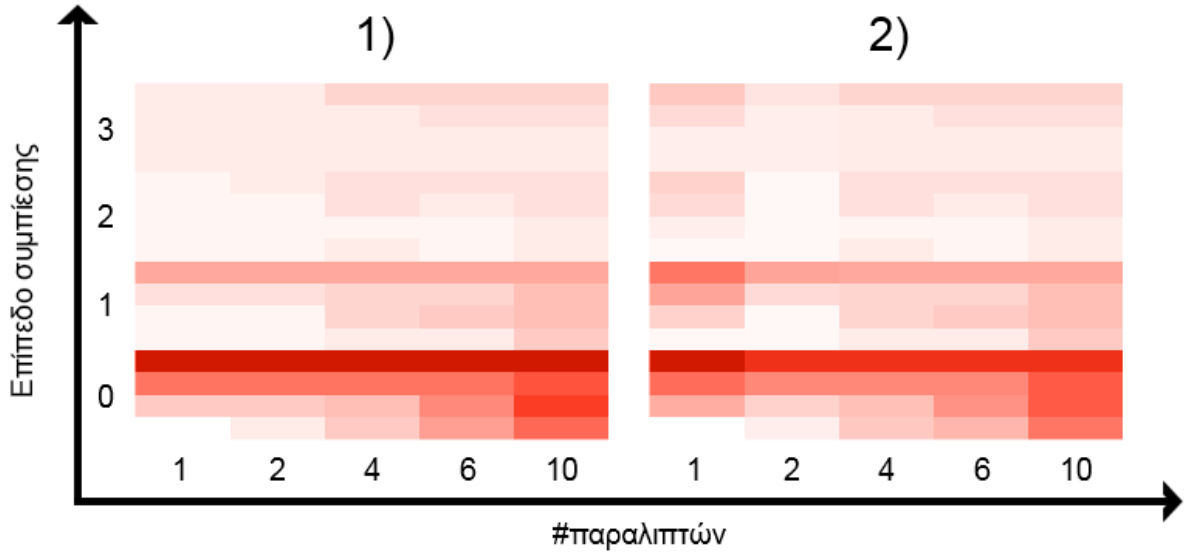
5.5.7.1 Λειτουργία διαμοιρασμού



Σχήμα 19: Μηχανή καταστάσεων στην λειτουργία διαμοιρασμού με μία κατάσταση που έχει δύο πιθανές επιλογές.

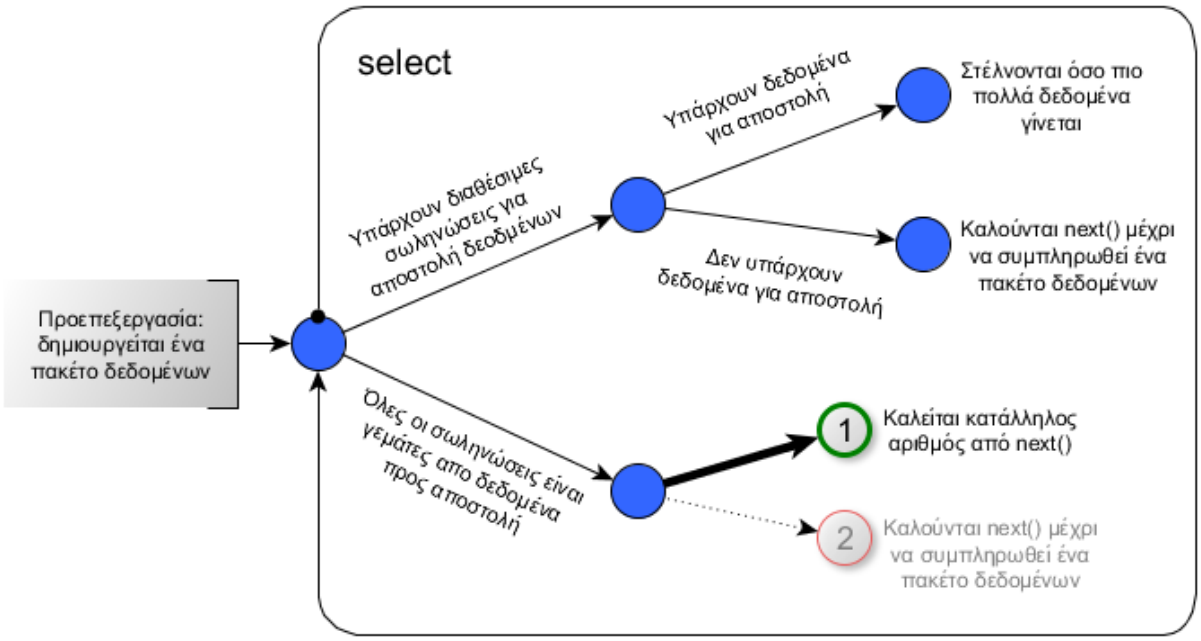
Η μηχανή καταστάσεων της μεθόδου κατά την λειτουργία διαμοιρασμού φαίνεται στο **Σχήμα 19**. Η μηχανή καταστάσεων αποτελείται από μια επανάληψη όπου κάθε φορά γίνεται έλεγχος με την λειτουργία select του συστήματος UNIX για το ποιες σωληνώσεις είναι διαθέσιμες, δηλαδή έχουν ελεύθερο χώρο ώστε να εγγραφούν δεδομένα σε αυτές. Μετά τις τελικές καταστάσεις η επανάληψη αρχίζει από την αρχή. Πριν ξεκινήσει αυτή η επανάληψη, υπάρχει μια προ-επεξεργασία όπου δημιουργείται ένα πακέτο δεδομένων προκειμένου να υπάρχουν δεδομένα για αποστολή.

Στο σχήμα φαίνεται μία κατάσταση, που δεν είναι γνωστό σε πια επόμενη κατάσταση συμφέρει να μεταφερθούμε. Επομένως θα πρέπει να γίνουν μετρήσεις και με τις δύο περιπτώσεις, ώστε να αποφανθούμε το πια έχει ως αποτέλεσμα καλύτερους χρόνους ολοκλήρωσης της συνολικής εργασίας. Η κατάσταση 1 καλεί κατάλληλο αριθμό από next() ώστε να μην ξεπεραστεί ο χρόνος που χρειάζεται μέχρι να καταναλωθούν τα ήδη ενταμιευμένα δεδομένα των σωληνώσεων, όπως αναλύθηκε στην **Ενότητα 5.5.6**.



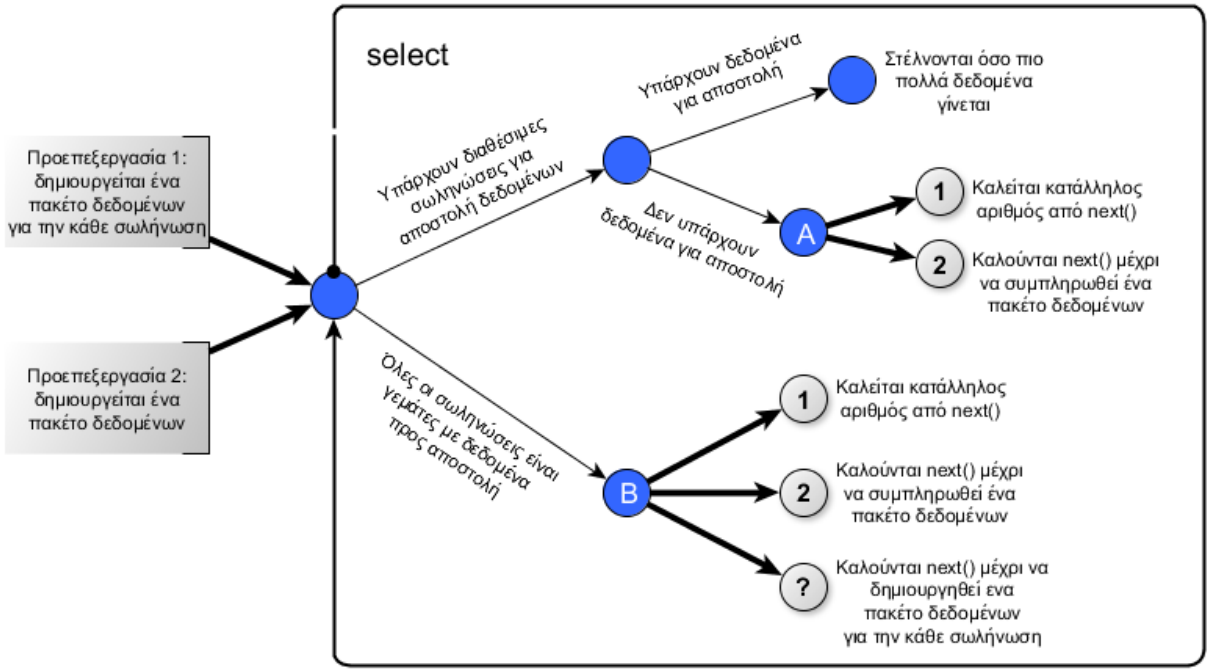
Σχήμα 20: Αποτελέσματα μετρήσεων για τις δύο πιθανές επιλογές. Όσο πιο κόκκινο το χρώμα τόσο μεγαλύτεροι είναι οι χρόνοι.

Όπως φαίνεται στο **Σχήμα 20** οι μετρήσεις δείχνουν ότι όταν υπάρχουν δεδομένα για αποστολή και όλες οι σωληνώσεις είναι γεμάτες, μας συμφέρει να επιλέξουμε την κατάσταση 1, δηλαδή να καλέσουμε τον κατάλληλο αριθμό από next() ανάλογα το εύρος ζώνης του δικτύου. Αυτό γιατί, όταν υπάρχει μόνο ένας παραλήπτης, ο συνολικός χρόνος είναι περισσότερος, σε σχέση με το να καλούνται next() μέχρι να γεμίσει ένα πακέτο δεδομένων. Επομένως η τελική εικόνα της μηχανής καταστάσεων κατά την λειτουργία διαμοιρασμού φαίνεται στο **Σχήμα 18**.



Σχήμα 21: Τελική μηχανή καταστάσεων της λειτουργίας διαμοιρασμού.

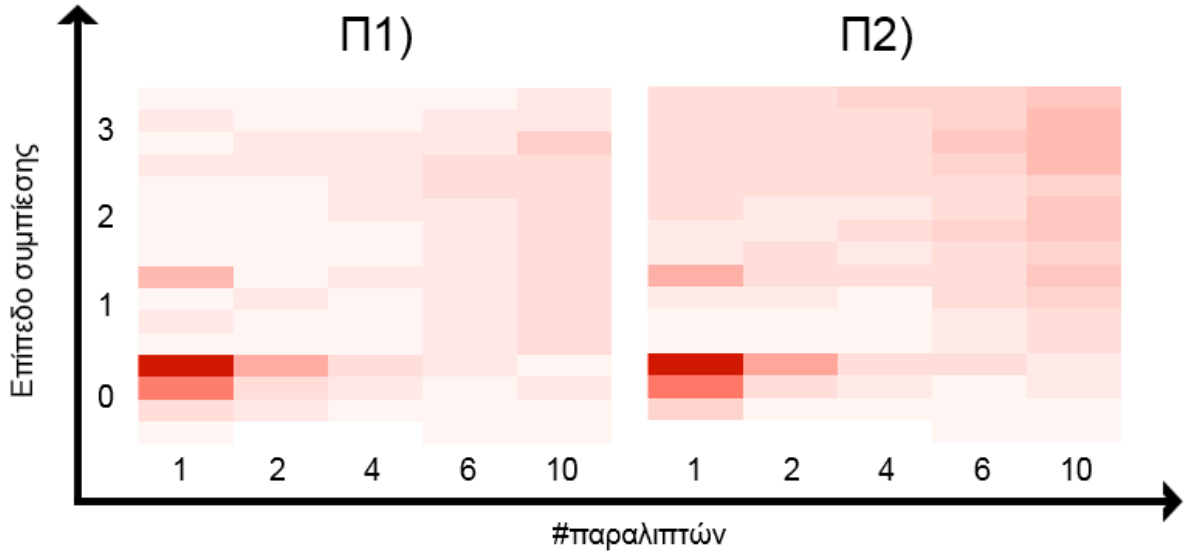
5.5.7.2 Λειτουργία διάσπασης



Σχήμα 22: Μηχανή καταστάσεων στην λειτουργία διάσπασης, με τρείς αμφιλεγόμενες καταστάσεις.

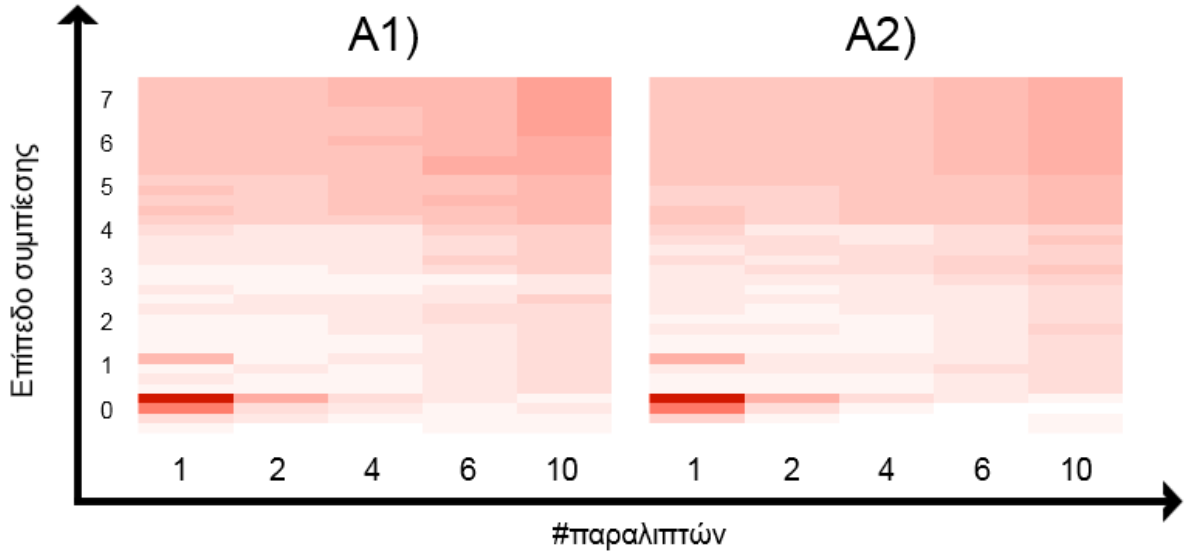
Η μηχανή καταστάσεων της μεθόδου κατά την λειτουργία διάσπασης φαίνεται στο **Σχήμα 22**. Η μηχανή καταστάσεων αποτελείται από μια επανάληψη όπου κάθε φορά γίνεται έλεγχος με την λειτουργία select του UNIX συστήματος για το ποιες σωληνώσεις είναι διαθέσιμες, δηλαδή έχουν ελεύθερο χώρο ώστε να εγγραφούν δεδομένα σε αυτές. Μετά τις τελικές καταστάσεις η επανάληψη αρχίζει από την αρχή. Πριν ξεκινήσει αυτή η επανάληψη, υπάρχει μια προ-επεξεργασία που πρέπει να γίνει. Όμως σε αυτήν την λειτουργία δεν είναι σαφές ποια από τις δύο επιλογές είναι καλύτερη για την προ-επεξεργασία, έτσι θα πρέπει να γίνουν μετρήσεις για να αποφασιστεί.

Στο σχήμα φαίνονται επίσης τρεις καταστάσεις, που δεν είναι γνωστό σε πια επόμενη κατάσταση συμφέρει να μεταφερθούμε. Επομένως θα πρέπει να γίνουν μετρήσεις για όλες τις περιπτώσεις, ώστε να αποφανθούμε το πια έχει ως αποτέλεσμα καλύτερους χρόνους ολοκλήρωσης της συνολικής εργασίας.



Σχήμα 23: Αποτελέσματα μετρήσεων για τις δύο πιθανές επιλογές της Προεπεξεργασίας. Όσο πιο κόκκινο το χρώμα τόσο μεγαλύτεροι είναι οι χρόνοι.

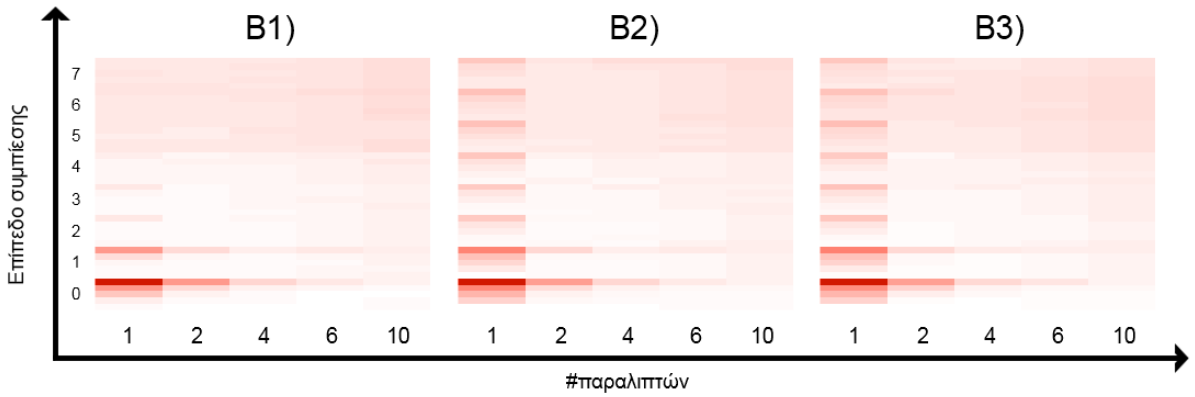
Όπως φαίνεται στο **Σχήμα 23** οι μετρήσεις δείχνουν ότι πριν ξεκινήσει η μέθοδός μας στην λειτουργία διάσπασης, συμφέρει η επιλογή 1, δηλαδή να ζητηθούν τόσες εγγραφές από την βάση ώστε να συμπληρωθεί ένα πακέτο δεδομένων για τον κάθε αποστολέα. Αυτό γιατί, παρατηρήθηκε ότι όσο αυξάνονται οι αποστολείς, και όσο αυξάνεται το επίπεδο συμπίεσης, ο συνολικός χρόνος είναι καλύτερος σε σύγκριση με την άλλη επιλογή. Αυτό οφείλεται στο ότι όλοι οι παραλήπτες έχουν να λάβουν δεδομένα από την πρώτη στιγμή.



Σχήμα 24: Αποτελέσματα μετρήσεων για τις δύο πιθανές επιλογές της κατάστασης Α. Όσο πιο κόκκινο το χρώμα τόσο μεγαλύτεροι είναι οι χρόνοι.

Όπως φαίνεται στο **Σχήμα 24** οι μετρήσεις δείχνουν ότι για τον κάθε αποστολέα που δεν είναι γεμάτος, και δεν υπάρχουν διαθέσιμα δεδομένα για αυτόν, μας συμφέρει η

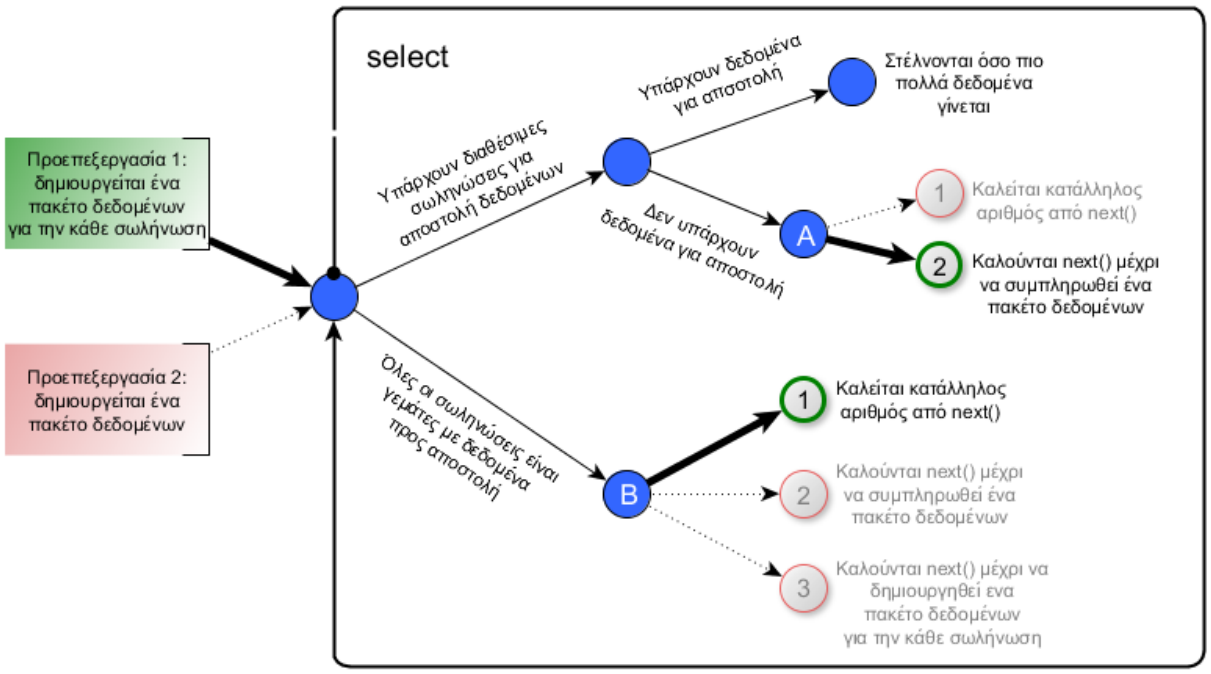
επιλογή της κατάστασης 2, δηλαδή να ζητήσουμε τον κατάλληλο αριθμό από εγγραφές της βάσης μέχρι να συμπληρωθεί ένα πακέτο δεδομένων. Αυτό γιατί, παρατηρήθηκε ότι όσο αυξάνονται οι αποστολείς, και όσο αυξάνεται το επίπεδο συμπίεσης, ο συνολικός χρόνος είναι καλύτερος σε σύγκριση με την άλλη επιλογή. Αυτό οφείλεται στο ότι αφού δεν υπάρχουν δεδομένα για αποστολή θα πρέπει να δημιουργηθούν άμεσα.



Σχήμα 25: Αποτελέσματα μετρήσεων για τις τρείς πιθανές επιλογές της κατάστασης B. Όσο πιο κόκκινο το χρώμα τόσο μεγαλύτεροι είναι οι χρόνοι.

Όπως φαίνεται στο **Σχήμα 25** οι μετρήσεις δείχνουν ότι όταν όλοι οι αποστολείς είναι γεμάτοι, μας συμφέρει η επιλογή της κατάστασης 1, δηλαδή να κληθεί ο κατάλληλος αριθμός από next(), ανάλογα το στιγμιαίο εύρος ζώνης του δικτύου. Αυτό οφείλεται στο ότι όλοι οι παραλήπτες θα πρέπει να καταναλώσουν τα ήδη ενταμιευμένα δεδομένα του λειτουργικού. Όμως επειδή είναι άγνωστος ο ακριβής χρόνος που θα ολοκληρωθεί η πλήρης λήψη των δεδομένων θα πρέπει να προβλεφθεί ο κατάλληλος αριθμός από next() που θα πρέπει να κληθούν με τον τρόπου που περιγράψαμε.

Επομένως η τελική εικόνα της μηχανής καταστάσεων κατά την λειτουργία διαμοιρασμού φαίνεται στο **Σχήμα 26**.



Σχήμα 26: Τελική μηχανή καταστάσεων της λειτουργίας διάσπασης.

6. ΠΕΙΡΑΜΑΤΙΚΗ ΑΞΙΟΛΟΓΗΣΗ

6.1 Πειραματική διάταξη

Οι δοκιμές έγιναν σε μια μηχανή με επεξεργαστή i7-4700MQ με χρονισμό στα 3.4 GHz και 16 GB RAM. Ο αποστολέας και οι παραλήπτες βρίσκονταν στο ίδιο μηχάνημα ώστε η αποστολή να γίνεται μέσω του localhost. Αυτό έγινε ώστε να μην υπάρχει περιορισμός στο συνολικό εύρος ζώνης λόγο του δικτύου, και επομένως να μπορούμε να μετρήσουμε τις μέγιστες δυνατότητες της μεθόδου μας. Όμως χρειάστηκε ένα είδος προσομοίωσης του εύρους ζώνης δικτύου προκειμένου να γίνουν οι απαραίτητοι πειραματισμοί με διαφορετικές τιμές εύρους ζώνης.

6.2 Προσομοίωση ρύθμισης εύρους ζώνης δικτύου

Εφόσον τα πειράματα γίνονται με έναν αποστολέα και πολλούς παραλήπτες, τα δεδομένα στην πραγματικότητα θα έπρεπε να αποστέλλονται από την ίδια κάρτα δικτύου. Η κάρτα δικτύου θα συνδεόταν μέσω μιας γραμμής δικτύου με έναν διαχωριστή πακέτων (splitter), ο οποίος θα ανακατευθύνει τα δεδομένα στους αντίστοιχους παραλήπτες. Αυτό έχει ως αποτέλεσμα η γραμμή που ενώνει την κάρτα δικτύου του αποστολέα με τον διαχωριστή πακέτων, να πρέπει να μοιραστεί, στην ιδανική περίπτωση, σε ίσα μερίδια εύρους ζώνης για τον κάθε παραλήπτη. Για παράδειγμα, όταν στις μετρήσεις μας θα αναφέρουμε ότι το εύρος ζώνης της γραμμής αποστολής δεδομένων είναι $10\,MB/s$, και υπάρχουν 2 παραλήπτες, τότε σημαίνει ότι στον κάθε παραλήπτη αντιστοιχίζεται εύρος ζώνης λήψης δεδομένων $10/2 = 5\,MB/s$.

Αυτό μπορεί να εξομοιωθεί ρυθμίζοντας τον κάθε παραλήπτη να δέχεται δεδομένα με συγκεκριμένο ρυθμό. Δηλαδή να γίνεται εσωτερικά, στον παραλήπτη, ρύθμιση εύρους ζώνης (bandwidth throttling) λήψης.

Το παραπάνω επιτυγχάνεται εύκολα ως εξής:

- Μετράται ανά τακτά χρονικά διαστήματα το σύνολο των δεδομένων που έχουν ληφθεί μέσα σε ένα χρονικό διάστημα.
    - Αν τα δεδομένα που λήφθηκαν είναι περισσότερα από τα δεδομένα που θα μπορούσαν να ληφθούν με το περιορισμένο εύρος ζώνης, τότε η διεργασία παγώνει (sleep) για το χρονικό διάστημα που θα χρειαζόταν, ώστε με το δεδομένο εύρος ζώνης, να γινόταν λήψη των παραπάνω δεδομένων που λήφθηκαν
    - Αλλιώς συνεχίζεται η λήψη δεδομένων.

6.3 Διαδικασία πειραμάτων

Για τις μετρήσεις μας χρειάστηκε να παράγουμε έναν πίνακα βάσης με την βοήθεια του εργαλείου παραγωγής τυχαίων δεδομένων TPC-H. Η βάση περιέχει ~12M εγγραφές, και είναι μεγέθους 1,7 GB.

Χρειάστηκε η σύγκριση της μεθόδου μας με τις δύο άλλες μεθόδους μεταφοράς. Η σύγκριση έγινε και για τις δύο λειτουργίες αποστολής, διαμοιρασμού και διάσπασης.

Η επικοινωνία αυτών γίνεται μέσω μίας κοινής δικτυακής γραμμής. Επομένως θεωρείται ότι το εύρος ζώνης της γραμμής ισοκατανέμεται ανάμεσα στους παραλήπτες.

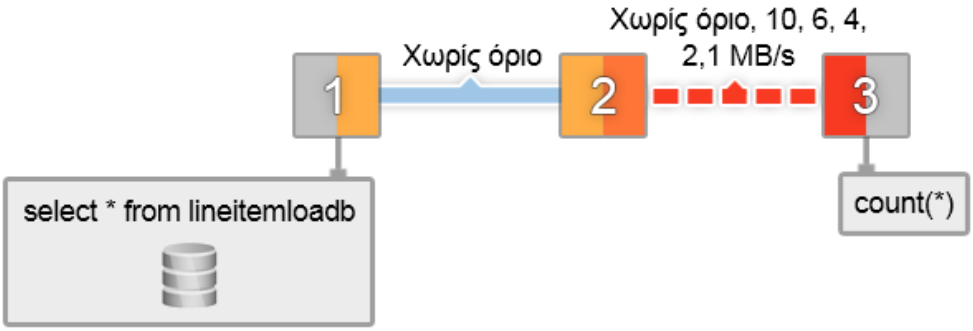
Οι μετρήσεις έγιναν με το εύρος ζώνης της γραμμής αποστολής να είναι στα 1, 2, 4, 6, 10 MB/s και χωρίς όριο.

Έγιναν δύο είδη πειραμάτων. Το πείραμα με τους τρεις κόμβους ώστε να φανούν τα πλεονεκτήματα και τα μειονεκτήματα όλων των μεθόδων. Και το πείραμα απόδοσης των δύο λειτουργιών μεταφοράς δεδομένων, διαμοιρασμού και διάσπασης, ώστε να μετρηθούν οι δυνατότητας του τελεστή του αποστολέα.

6.4 Αποτελέσματα

6.4.1 Πείραμα με τρεις κόμβους

Το πείραμα αυτό διεξάγεται έχοντας τρεις κόμβους στην σειρά, όπου ο ένας στέλνει στον άλλον τα ίδια δεδομένα χωρίς επεξεργασία. Με αυτό το πείραμα θέλουμε να γίνουν εμφανή τα μειονεκτήματα και τα πλεονεκτήματα των δύο βασικών μεθόδων σε σύγκριση με την μέθοδο που προτείνουμε.

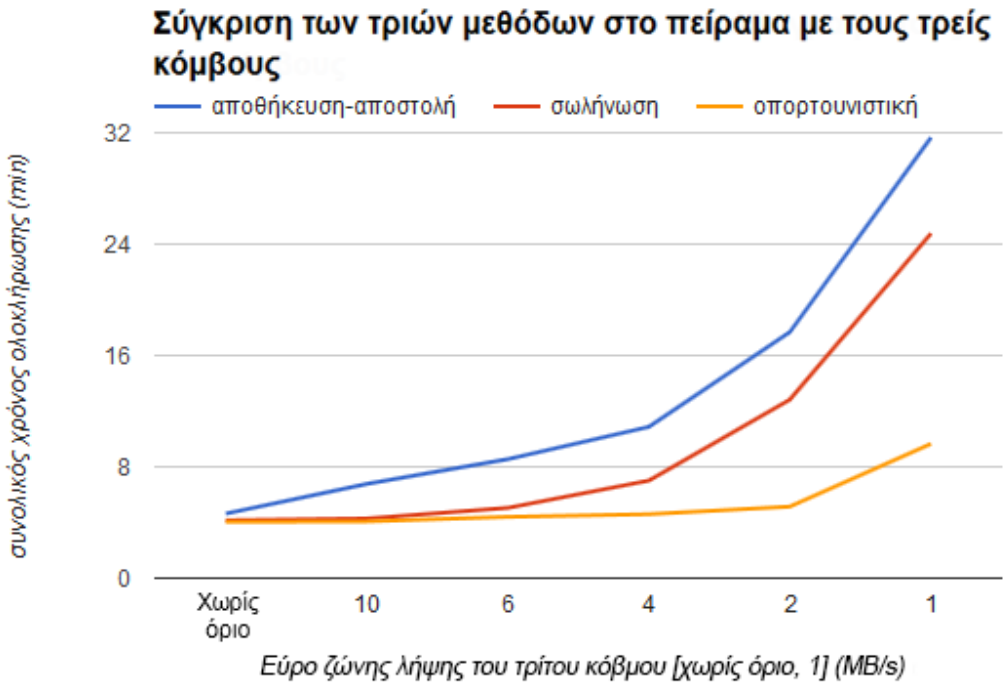


Σχήμα 27: Σχεδιάγραμμα πειράματος με τρεις κόμβους στην σειρά, όπου μεταβάλλεται το εύρος ζώνης αποστολής του δεύτερου.

Συγκεκριμένα στο πείραμα, ο πρώτος κόμβος διαβάζει την βάση με την εντολή

```
select * from lineitemloadb;
```

και την αποστέλλει στον δεύτερο κόμβο χωρίς όριο εύρους ζώνης. Ο δεύτερος κόμβος λαμβάνει την βάση και, την προωθεί στον τρίτο κόμβο. Όμως το εύρος ζώνης που δέχεται δεδομένα ο τρίτος κόμβος μεταβάλλεται και περιορίζεται, αρχίζοντας χωρίς περιορισμό και φτάνοντας μέχρι το 1 MB/s, με βήματα 10, 6, 4, 2, 1 MB/s.



Σχήμα 28: Σύγκριση των τριών μεθόδων πάνω στο πείραμα των τριών κόμβων.

Τα αποτελέσματα του πειράματος φαίνονται στο **Σχήμα 28**. Το διάγραμμα παρουσιάζει τον συνολικό χρόνο εκτέλεσης του πειράματος σε κάθε μέθοδο, με κάθε τιμή περιορισμού εύρους ζώνης λήψης στον τρίτο κόμβο.

Παρατηρούμε ότι η μέθοδος αποθήκευση-αποστολή είναι η λιγότερο αποδοτική από όλες. Αυτό οφείλεται στην μη ύπαρξη επικάλυψης χρόνου αποστολής με επεξεργασία. Και επομένως ο συνολικός χρόνος στο κάθε στάδιο του πειράματος θα είναι το άθροισμα της αποστολής του αρχείου από τον πρώτο στον δεύτερο κόμβο με 56 MB/s και από τον δεύτερο στον τρίτο κόμβο με τον κάθε περιορισμό εύρους ζώνης.

Παρατηρούμε ότι στην μέθοδος σωλήνωσης είναι όντως πιο αποδοτική από την μέθοδο αποθήκευση-αποστολή. Όμως, καθώς μειώνεται το εύρος ζώνης λήψης του τρίτου κόμβου, γίνεται εμφανές το αρνητικό χαρακτηριστικό της μεθόδου, όπου ο περιορισμός εύρους ζώνης μεταξύ του δεύτερου και του τρίτου κόμβου προκαλεί συμφόρηση και στο ζεύγος του πρώτου με τον δεύτερο κόμβο. Ενώ σε σχετικά υψηλούς περιορισμούς εύρους ζώνης φαίνεται το πλεονέκτημα της επικάλυψης χρόνου αποστολής μεταξύ των δύο ζευγαριών, στους χαμηλούς περιορισμούς οι χρόνοι πλησιάζουν όλο και πιο πολύ την μέθοδο αποθήκευση-αποστολή όπως ήταν αναμενόμενο.

Η μέθοδος οπορτουνιστική σωλήνωσης επιλέγει τον σωστό τρόπο διαχείρισης της μνήμης, και πραγματοποιεί τις κατάλληλες αντισταθμιστικές επιλογές μεταξύ επεξεργαστικής ισχύς και αποστολής δεδομένων, επιλέγοντας κάθε φορά το κατάλληλο επίπεδο συμπίεσης των δεδομένων, ώστε να μεγιστοποιηθεί το επιτευκτό εύρος ζώνης μεταφοράς δεδομένων. Έτσι φαίνεται να μην επηρεάζεται από τον περιορισμό εύρους ζώνης που επιβάλλεται στον τρίτο κόμβο.

Πίνακας 2: Επίτευξη εύρους ζώνης με κάθε μέθοδο στο κάθε στάδιο του πειράματος.

Εύρος ζώνης λήψης του τρίτου κόμβου (MB/s)	Συνολική επίτευξη εύρους ζώνης (MB/s)		
	αποθήκευση-αποστολή	σωλήνωσης	οπορτουνιστική
Χωρίς όριο	6,27	6,98	**7,19**
10	4,29	6,79	**7,12**
6	3,40	5,76	**6,60**
4	2,67	4,15	**6,33**
2	1,64	2,27	**5,65**
1	0,92	1,17	**3,01**

Στον **Πίνακα 2** φαίνεται το αντίστοιχο συνολικό εύρος ζώνης που επιτυγχάνεται σε αυτό το πείραμα με την κάθε μέθοδο στο κάθε στάδιο. Παρατηρούμε ότι η μέθοδος οπορτουνιστικής σωλήνωσης πετυχαίνει συνολικό εύρος ζώνης που είναι πολλαπλάσιο από τον περιορισμό εύρους ζώνης στο δεύτερο ζεύγος των κόμβων, ειδικά στους χαμηλούς περιορισμούς.

6.4.2 Πείραμα απόδοσης των λειτουργιών

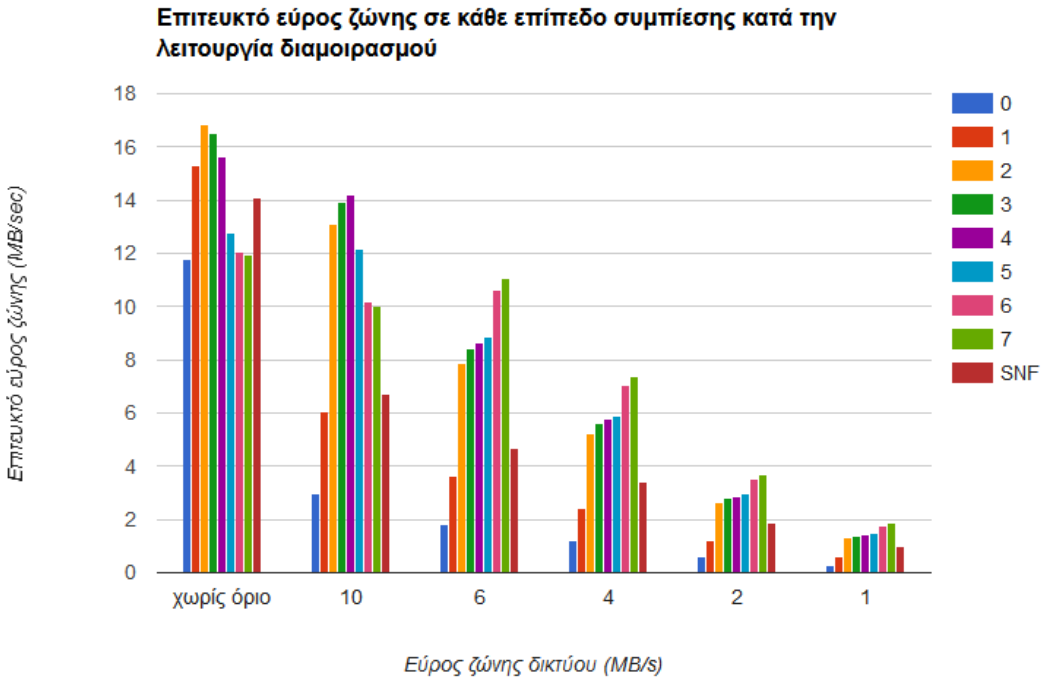
Σε αυτό το πείραμα θα προσπαθήσουμε να αναδείξουμε τις δυνατότητες που έχει ο τελεστής του αποστολέα στην κάθε λειτουργία μεταφοράς. Στην λειτουργία διαμοιρασμού και στην λειτουργία διάσπασης.

Το πείραμα θα διεξαχθεί με έναν αποστολέα και 4 παραλήπτες, όπου θα ανταλλάσσονται δεδομένα μέσω κοινής δικτυακής γραμμής. Θεωρείται ότι το εύρος ζώνης της γραμμής ισοκατανέμεται στους παραλήπτες. Το εύρος ζώνης που προσφέρει συνολικά η γραμμή θα μεταβάλλεται και θα περιορίζεται, αρχίζοντας χωρίς όριο και φτάνοντας μέχρι το 1 MB/s, με βήματα 10, 6, 4, 2, 1 MB/s.

Από την μεριά του αποστολέα θα επιλέγεται σταθερά κάθε φορά ένα επίπεδο συμπίεσης του αλγορίθμου, από το 0 μέχρι το 7. Η μετρική του πειράματος θα είναι το εύρος ζώνης που επιτυγχάνεται με την ολοκλήρωση του κάθε σταδίου.

Τα αποτελέσματα συγκρίνονται με την αντίστοιχη μέτρηση με την μέθοδο αποθήκευση-αποστολή.

6.4.2.1 Λειτουργία διαμοιρασμού



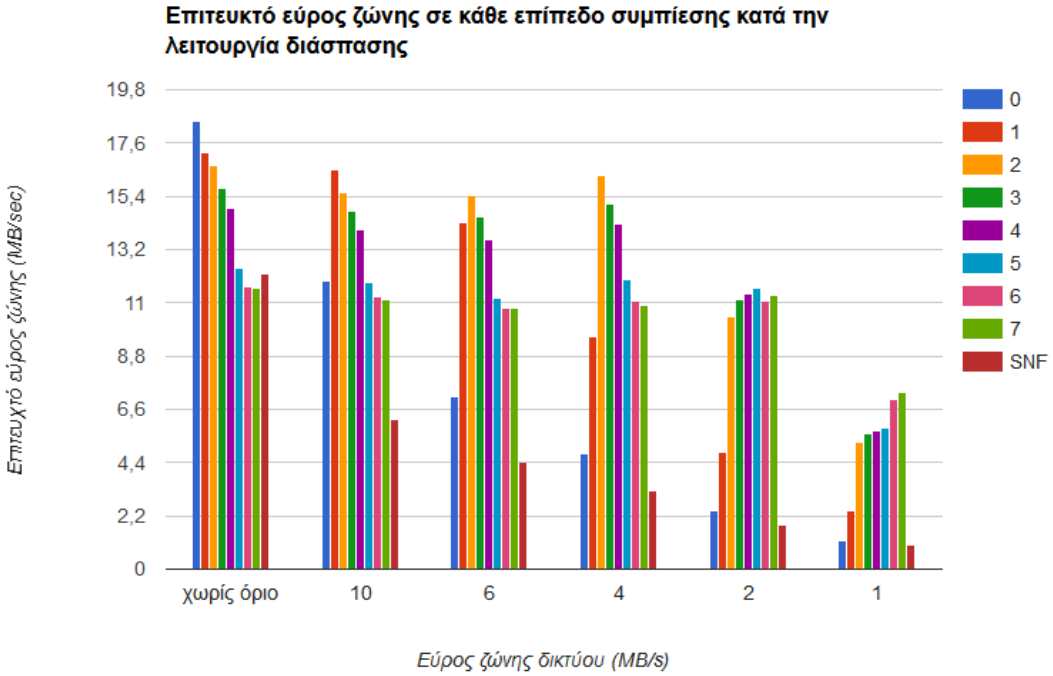
Σχήμα 29: Μέτρηση της απόδοσης του αποστολέα κατά την λειτουργία διαμοιρασμού σε 4 παραλήπτες. Στο διάγραμμα φαίνεται το εύρος ζώνης που επιτυγχάνεται στο κάθε στάδιο περιορισμού του εύρους ζώνης του δικτύου, για το κάθε επίπεδο συμπίεσης της μεθόδου οπορτουνιστικής σωλήνωσης σε σύγκριση με την μέθοδο αποθήκευση-αποστολή (SNF). Όσο μεγαλύτερη η τιμή τόσο καλύτερα.

Στο **Σχήμα 29** φαίνεται η επιλογή του καταλληλότερου επιπέδου συμπίεσης για τον κάθε περιορισμό εύρους ζώνης του δικτύου. Το καταλληλότερο είναι αυτό που επιτυγχάνει το μεγαλύτερο εύρος ζώνης. Επίσης φαίνεται και το εύρος ζώνης που επιτυγχάνει η μέθοδος αποθήκευση-αποστολή (SNF).

Αυτό που μπορούμε να παρατηρήσουμε είναι την αλλαγή της καταλληλότητας του κάθε επιπέδου συμπίεσης ανάλογα με το εύρος ζώνης του δικτύου. Επίσης παρατηρούμε ότι για περιορισμό δικτύου από 6 MB/s και κάτω επιλέγεται το επίπεδο 7 ως το καλύτερο. Φαίνεται δηλαδή ότι υπάρχει η ανάγκη για περισσότερα επίπεδα υψηλής συμπίεσης καλύτερα από το επίπεδο 7.

Συγκρίνοντας την μέθοδό μας με την μέθοδο αποθήκευσης-αποστολής φαίνεται το πολλαπλάσιο εύρος ζώνης που μπορεί να επιτευχθεί. Ειδικότερα όταν το εύρος ζώνης του δικτύου περιορίζεται όλο και πιο πολύ.

6.4.2.2 Λειτουργία διάσπασης



Σχήμα 30: Μέτρηση της απόδοσης του αποστολέα κατά την λειτουργία διάσπασης σε 4 παραλήπτες. Στο διάγραμμα φαίνεται το εύρος ζώνης που επιτυγχάνεται στο κάθε στάδιο περιορισμού του εύρους ζώνης του δικτύου, για το κάθε επίπεδο συμπίεσης της μεθόδου οπορτουνιστικής σωλήνωσης σε σύγκριση με την μέθοδο αποθήκευση-αποστολή (SNF). Όσο μεγαλύτερη η τιμή τόσο καλύτερα.

Αντίστοιχα με την μέθοδο διαμοιρασμού, στο **Σχήμα 30** φαίνεται η επιλογή του καταλληλότερου επιπέδου συμπίεσης για τον κάθε περιορισμό εύρους ζώνης του δικτύου. Το καταλληλότερο είναι αυτό που επιτυγχάνει το μεγαλύτερο εύρος ζώνης. Επίσης φαίνεται και το εύρος ζώνης που επιτυγχάνει η μέθοδος αποθήκευση-αποστολή (SNF).

Αυτό που παρατηρούμε και εδώ είναι η αλλαγή της καταλληλότητας του κάθε επιπέδου συμπίεσης ανάλογα με το εύρος ζώνης του δικτύου. Όμως παρατηρούμε ότι η αλλαγή κατά αυτήν την λειτουργία αποστολής είναι αρκετά πιο ομαλή. Φαίνεται ότι μπορούν να αξιοποιηθούν όλα τα επίπεδα συμπίεσης για περισσότερες τιμές του εύρους ζώνης του δικτύου.

Συγκρίνοντας την μέθοδό μας με την μέθοδο αποθήκευσης-αποστολής φαίνεται το πολλαπλάσιο εύρος ζώνης που μπορεί να επιτευχθεί. Ειδικότερα όταν το εύρος ζώνης του δικτύου περιορίζεται όλο και πιο πολύ. Ειδικότερα, φαίνεται ότι μπορεί να επιτευχθεί μέχρι και 7 φορές μεγαλύτερο εικονικό εύρος ζώνης αποστολής, όταν το εύρος ζώνης του δικτύου περιορίζεται στο 1 MB/s.

7. ΜΕΛΛΟΝΤΙΚΗ ΕΡΓΑΣΙΑ

7.1 Ενσωμάτωση στο σύστημα Exareme

Η μέθοδός μεταφοράς οπορτουνιστικής σωλήνωσης που προτείνουμε σε αυτήν την εργασία σχεδιάστηκε για το σύστημα Exareme. Θα πρέπει να προσαρμοστεί όμως κατάλληλα ώστε να λειτουργήσει πραγματικά μέσα στο σύστημα. Το Exareme θα πρέπει επίσης να προσαρμόσει κατάλληλα την δομή του ώστε να λειτουργήσει με την μέθοδό μας σαν τμήμα του.

7.2 Υποστήριξη περισσότερων επίπεδων συμπίεσης

Όπως είδαμε στο πείραμα ελέγχου της απόδοσης της λειτουργίας διαμοιρασμού στην **Ενότητα 6.4.2.1**, η επιλογή του τελευταίου επιπέδου συμπίεσης ως βέλτιστο έγινε πιο απότομα όσο το εύρος ζώνης του δικτύου μειωνόταν, σε σχέση με την λειτουργία διάσπασης. Επομένως, φαίνεται πως υπάρχει ανάγκη ύπαρξης περισσότερων επιπέδων συμπίεσης. Θα πρέπει επομένως να γίνουν δοκιμές και μετρήσεις και με άλλα ισχυρότερα επίπεδα συμπίεσης που θα προστεθούν στην βιβλιοθήκη MadComp του συστήματος madIS.

7.3 Βελτιώσεις χαμηλού επιπέδου

Προκειμένου να αυξηθεί και άλλο η απόδοση της μεθόδου μεταφοράς οπορτουνιστικής σωλήνωσης θα χρειαστεί να γίνει πιο λεπτομερής ανάλυση του προφίλ της εκτέλεσης της μεθόδου. Έτσι θα εντοπιστούν πιθανά σημεία στον κώδικα, τα οποία χρειάζονται τις κατάλληλες βελτιώσεις χαμηλού επιπέδου.

7.4 Πολιτική χρήσης της κύριας μνήμης με βάση πιθανοτικές κατανομές

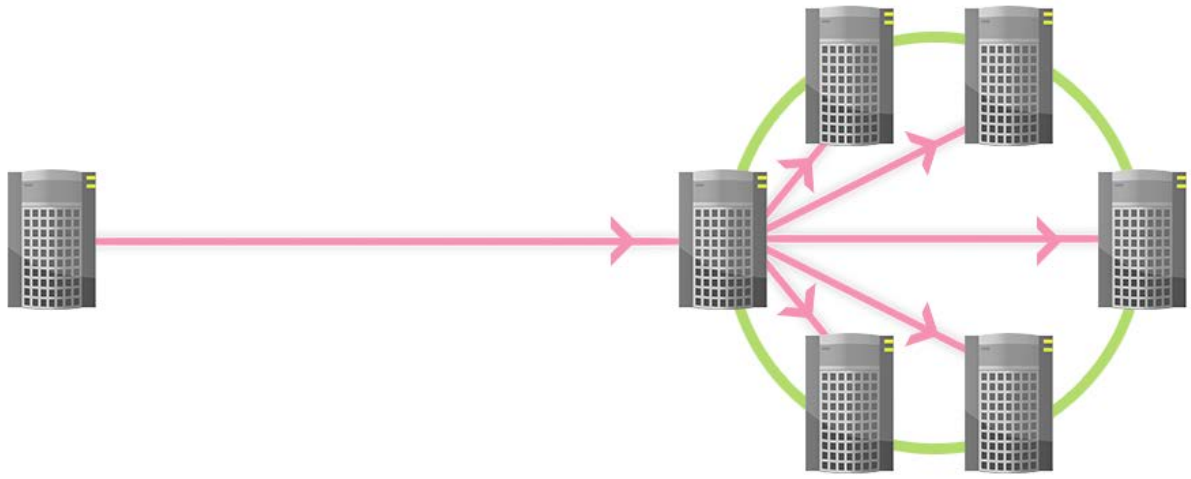
Η πολιτική χρήσης της κύριας μνήμης, που αναφέραμε στην **Ενότητα 5.6** θα πρέπει να εμπλουτιστεί με την υποστήριξη πιθανοτικών κατανομών. Δηλαδή, το κύριο μέρος των πακέτων που βρίσκονται στην μνήμη να συγκεντρώνεται γύρω από την μέση τιμή των ρυθμών λήψης των παραληπτών. Η οποία μέση τιμή θα βασίζεται σε κάποια πιθανοτική κατανομή που θα ταιριάζει με την συμπεριφορά των παραληπτών του συστήματος. Η κατάλληλη πιθανοτική κατανομή θα πρέπει να βρεθεί πειραματικά σε ένα πραγματικό σύστημα αναλυτικής επεξεργασίας. Έτσι δεν θα επωφελούνται μόνο οι πιο γρήγοροι παραλήπτες, αλλά όσοι βρίσκονται γύρω από την μέση τιμή.

7.5 Πρώιμη εξισορρόπηση συστήματος

Σε ένα σύστημα αναλυτικής επεξεργασίας θα πρέπει να γίνονται, όσο πιο γρήγορα γίνεται, οι καλύτερες επιλογές για την επίτευξη της βέλτιστης απόδοσής του. Είναι σημαντικό να μπορούν να εντοπιστούν γρήγορα οι χρονικές στιγμές που θα πρέπει να γίνει κάποια αλλαγή στα τμήματα του συστήματος. Από την σκοπιά της μεθόδου μας, θα πρέπει να μελετηθούν τρόποι για την άμεση ή και πρόωρη αλλαγή των παραμέτρων ώστε να μείνει εξισορροπημένο το σύστημα ακόμα και σε απότομες αλλαγές που μπορούν να προκύψουν.

7.6 Εκμετάλλευση της τοπολογίας των κόμβων του συστήματος

Αν κάποιος κόμβος γνωρίζει ότι πρέπει να διαμοιράσει τα ίδια δεδομένα σε πολλούς κόμβους, κάποιοι από τους οποίους λειτουργούν σαν ομάδα και έχουν μεταξύ τους αρκετά μεγαλύτερο εύρος δικτύου σε σχέση με το εύρος ζώνης που μπορεί να τους στείλει ο κόμβος που έχει τα δεδομένα διαμοιρασμού, τότε θα μπορούσε να εκμεταλλευτεί την τοπολογία τους και να στείλει τα δεδομένα μόνο σε έναν κόμβο από αυτούς. Ο κόμβος της ομάδας που έλαβε τα δεδομένα θα αναλάβει να διαμοιράσει ο ίδιος τα δεδομένα στους υπόλοιπους κόμβους της ομάδας του. Έτσι το συνολικό εύρος ζώνης θα βελτιωθεί. Η λειτουργία αυτή απεικονίζεται στο **Σχήμα 31**. Ο πρώτος κόμβος από αριστερά θέλει να διαμοιράσει τα ίδια δεδομένα στους υπόλοιπους κόμβους. Έτσι θα τα στείλει στον έναν από αυτούς, και αυτός με την σειρά του θα αναλάβει να τα διαμοιράσει στους υπόλοιπους, εφόσον έχει μεγαλύτερο εύρος ζώνης.



Σχήμα 31: Απεικόνιση του τρόπου εκμετάλλευσης της τοπολογίας των κόμβων που είναι πολύ κοντά μεταξύ τους.

8. ΣΥΜΠΕΡΑΣΜΑΤΑ

Χρησιμοποιώντας τον παραμετροποιήσιμο αλγόριθμο συμπίεσης δεδομένων πολλαπλών επιπέδων και την γνώση του εύρους ζώνης του δικτύου καταφέραμε να πετύχουμε ελαστική μεταφορά δεδομένων. Συμβάλλοντας έτσι στην βελτίωση του συνολικού εύρους ζώνης ενός συστήματος σαν το Exareme.

Καταφέραμε επίσης να χρησιμοποιούμε ελαστικά την κύρια μνήμη και τον σκληρό δίσκο σαν μέσα αποθήκευσης των δεδομένων. Με αυτόν τον τρόπο συνδυάστηκαν οι δύο βασικές μέθοδοι αποστολής δεδομένων σε μία. Η μέθοδος αποθήκευση-αποστολή και η μέθοδος σωλήνωσης, όπου η κύρια διαφορά τους είναι το αποθηκευτικό μέσο που χρησιμοποιούν. Αποτέλεσμα αυτού είναι να συνυπάρχουν σε μία μέθοδο μεταφοράς όλα τα θετικά στοιχεία και των δύο βασικών μεθόδων. Και επίσης να απαλειφθούν τα αρνητικά τους, αφού τα θετικά της μίας είναι τα αρνητικά της άλλης.

Προκειμένου η μέθοδός μας να ενσωματωθεί στο σύστημα του Exareme, στον κάθε κόμβο που διαθέτει το σύστημα madIS, επιλέχθηκε να χρησιμοποιηθεί ασύγχρονος, μη παρεμποδιστικός προγραμματισμός. Με τον τρόπο αυτό δεν απαιτούνται πολλαπλά υπολογιστικά νήματα για την παράλληλη αποστολή δεδομένων σε πολλαπλούς παραλήπτες. Έτσι οι απαιτήσεις σε πόρους του συστήματος είναι σχετικά πιο χαμηλές σε σύγκριση με τον συμβατικό παράλληλο προγραμματισμό

Από τις μετρήσεις βρέθηκε ότι η μέθοδος οπορτουνιστικής σωλήνωσης που προτείνουμε μπορεί να πετύχει μέχρι και 7 φορές πολλαπλάσιο εύρος ζώνης σε σχέση με τις δύο βασικές μεθόδους αποστολής δεδομένων. Επίσης φάνηκε ότι μπορεί να πετύχει το βέλτιστο εύρος ζώνης που μπορεί να επιτευχθεί από την μέθοδό μας, ακόμη και σε περιπτώσεις μη σταθερού εύρους ζώνης αποστολή δεδομένων στο δίκτυο.

Ο προγραμματισμός της μεθόδου μας αποδείχθηκε αρκετά δύσκολος για δύο κυρίως λόγους. Πρώτον, ο ασύγχρονος προγραμματισμός είναι αρκετά πιο δύσκολος από τον συμβατικό πολυνηματικό προγραμματισμό για την επίτευξη παραλληλίας. Δεύτερον, χρειάζεται αρκετός χρόνος για την ελαστικότητα χρήσης της ιεραρχίας της μνήμης, λόγω του ότι απαιτούνται πολλά σημεία και δομές ελέγχου για την κατάσταση της μνήμης, ειδικά σε ένα περιβάλλον υποστήριξης παραλληλίας.

ΠΙΝΑΚΑΣ ΟΡΟΛΟΓΙΑΣ

Ξενόγλωσσος όρος	**Ελληνικός Όρος**
Bottleneck	Κύριο σημείο συμφόρησης
Bandwidth	Εύρος ζώνης
Trade-offs	Αντισταθμιστικές επιλογές
Store and forward	Αποθήκευση-αποστολή
Pipe	Σωλήνωση
Synchronous/Asynchronous	Σύγχρονος/Ασύγχρονος
Blocking/non-blocking	Παρεμποδιστικός/μη-παρεμποδιστικός
On-chip	Μέσα στο κύκλωμα
Semaphore	Σημαφόρος
Bit-rate	Ρυθμός μετάδοσης bit
Buffering	Ενταμίευση
Stream	Ροή δεδομένων
Clusters	Συστάδες υπολογιστών
Clouds	Υπολογιστικά νέφη
Broadcast	Διαμοιρασμός
Split	Διάσπαση
Forking	Δημιουργία νέων διεργασιών
Context switching	Μεταγωγής περιβάλλοντος
Row	Εγγραφή βάσης
Callback	Κλήση συνάρτησης
Bandwidth throttling	Ρύθμιση εύρους ζώνης
Splitter	Διαχωριστής πακέτων
Sleep	Πάγωμα διεργασίας
Compression	Συμπίεση
Ram	Κύρια μνήμη συστήματος
Metadata	Μεταδεδομένα

ΣΥΝΤΜΗΣΕΙΣ – ΑΡΚΤΙΚΟΛΕΞΑ – ΑΚΡΩΝΥΜΙΑ

FLOPS	Floating-point operations per second
SNF	Store and Forward

ΑΝΑΦΟΡΕΣ

[1] John Morrison, John Shalf, Supid Dosanjh: Exascale Computing Technology CHallenges.
[2] Horst Simon: No Exascale for you.
[3] Kathy Yelick: Algorithmic Challenges of Exascale Computing.
[4] MADgIK; http://www.madgik.di.uoa.gr/. [Προσπελάστηκε 6/11/15]
[5] Exareme; http://madgik.github.io/exareme/. [Προσπελάστηκε 6/11/15]
[6] madIS; https://github.com/madgik/madis. [Προσπελάστηκε 6/11/15]
[7] Python; https://www.python.org/. [Προσπελάστηκε 6/11/15]
[8] SQLite; https://www.sqlite.org/. [Προσπελάστηκε 6/11/15]
[9] tpch-kit; https://github.com/alexpap/tpch-kit. [Προσπελάστηκε 6/11/15]