ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΜΕΤΑΠΤΥΧΙΑΚΟ ΔΙΠΛΩΜΑ ΕΙΔΙΚΕΥΣΗΣ (MSc)
στα ΠΛΗΡΟΦΟΡΙΑΚΑ ΣΥΣΤΗΜΑΤΑ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

«Συστήματα Διαχείρισης Ροών Δεδομένων»

Μπουγιούρης Παναγιώτης

Μ3040033

ΑΘΗΝΑ, ΜΑΪΟΣ 2006

ΜΕΤΑΠΤΥΧΙΑΚΟ ΔΙΠΛΩΜΑ ΕΙΔΙΚΕΥΣΗΣ (MSc)
στα ΠΛΗΡΟΦΟΡΙΑΚΑ ΣΥΣΤΗΜΑΤΑ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

«Συστήματα Διαχείρισης Ροών Δεδομένων»

Μπουγιούρης Παναγιώτης

M3040033

Επιβλέπων Καθηγητής: Βασίλης Βασσάλος
Εξωτερικός Κριτής: Δαμιανός Χατζηαντωνίου

ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΑΘΗΝΑ, ΜΑΪΟΣ 2006

ΠΕΡΙΕΧΌΜΕΝΑ

ΣΥΝΌΨΕΙΣ (EXTENDED ABSTRACTS)

i. ΣΥΝΟΨΗ ΣΤΑ ΕΛΛΗΝΙΚΆ
ii. ΣΥΝΟΨΗ ΣΤΑ ΑΓΓΛΙΚΆ

0. ΠΡΌΛΟΓΟΣ

(ΠΡΏΤΟ ΜΈΡΟΣ)

1. ΕΙΣΑΓΩΓΙΚΆ ΓΙΑ ΤΙΣ ΡΟΈΣ ΔΕΔΟΜΈΝΩΝ
    1. ΕΙΣΑΓΩΓΉ – ΟΡΙΣΜΟΊ
    2. ΚΑΤΗΓΟΡΊΕΣ ΡΟΏΝ ΔΕΔΟΜΈΝΩΝ
    3. ΠΑΡΆΔΕΙΓΜΑ ΣΥΣΤΉΜΑΤΟΣ ΔΙΑΧΕΊΡΙΣΗΣ ΡΟΏΝ ΔΕΔΟΜΈΝΩΝ (ΣΔΡΔ)
    4. ΣΥΣΧΕΤΊΣΕΙΣ ΚΑΙ ΔΙΑΦΟΡΈΣ ΜΕ ΠΑΡΑΔΟΣΙΑΚΆ ΣΥΣΤΉΜΑΤΑ ΔΙΑΧΕΊΡΙΣΗΣ ΒΆΣΕΩΝ ΔΕΔΟΜΈΝΩΝ (ΣΔΒΔ)
        1. ΣΤΑΤΙΚΌΤΗΤΑ ΤΩΝ ΕΠΕΡΩΤΉΣΕΩΝ
        2. Η ΈΝΝΟΙΑ ΤΟΥ ΧΡΌΝΟΥ
        3. ΜΗ ΠΕΠΕΡΑΣΜΈΝΑ ΣΎΝΟΛΑ ΔΕΔΟΜΈΝΩΝ
        4. ΑΣΤΆΘΜΗΤΟΙ ΠΑΡΆΓΟΝΤΕΣ ΤΩΝ ΡΟΏΝ ΔΕΔΟΜΈΝΩΝ
        5. ΔΥΝΑΤΌΤΗΤΑ ΑΝΆΔΡΑΣΗΣ
    5. ΧΑΡΑΚΤΗΡΙΣΤΙΚΈΣ ΠΡΟΔΙΑΓΡΑΦΈΣ ΣΥΣΤΗΜΆΤΩΝ ΔΙΑΧΕΊΡΙΣΗΣ ΡΟΏΝ ΔΕΔΟΜΈΝΩΝ (ΣΔΡΔ)
    6. ΤΟ ΜΟΝΤΈΛΟ ΤΗΣ ΡΟΉΣ
        1. ΠΕΡΙΓΡΑΦΉ
        2. ΧΡΟΝΟΣΉΜΑΝΣΗ (TIMESTAMPING)
        3. ΠΛΑΊΣΙΑ (WINDOWS)
    7. ΠΑΡΆΔΕΙΓΜΑ ΡΟΉΣ ΔΕΔΟΜΈΝΩΝ

2. ΣΥΝΕΧΕΊΣ ΕΠΕΡΩΤΉΣΕΙΣ (CONTINUOUS QUERIES)

1. ΕΙΣΑΓΩΓΉ ΣΤΙΣ ΣΥΝΕΧΕΊΣ ΕΠΕΡΩΤΉΣΕΙΣ
2. ΠΑΡΆΔΕΙΓΜΑ ΑΠΌ ΤΟ ΧΏΡΟ ΤΗΣ ΟΙΚΟΝΟΜΊΑΣ
3. ΕΚΤΈΛΕΣΗ ΣΥΝΕΧΏΝ ΕΠΕΡΩΤΉΣΕΩΝ
4. ΤΕΛΕΣΤΈΣ ΕΠΙΛΟΓΉ - ΠΡΟΒΟΛΉ
5. ΣΥΝΕΝΏΣΕΙΣ (JOINS)
    1. ΕΚΤΈΛΕΣΗ ΣΥΝΕΝΏΣΕΩΝ
    2. ΕΚΤΊΜΗΣΗ ΚΌΣΤΟΥΣ ΣΥΝΕΝΏΣΕΩΝ
    3. ΣΥΝΕΝΏΣΕΙΣ ΠΟΥ ΞΕΠΕΡΝΟΎΝ ΤΑ ΌΡΙΑ ΤΗΣ ΜΝΉΜΗΣ
    4. ΠΟΛΛΑΠΛΈΣ ΣΥΝΕΝΏΣΕΙΣ
6. ΔΙΑΧΕΊΡΙΣΗ ΠΌΡΩΝ
    1. ΔΙΑΧΕΊΡΙΣΗ ΜΝΉΜΗΣ
    2. ΔΙΑΜΟΙΡΑΣΜΌΣ ΠΌΡΩΝ
    3. ΠΡΟΓΡΑΜΜΑΤΙΣΜΌΣ (SCHEDULING)
    4. ΠΡΟΣΕΓΓΊΣΕΙΣ (APPROXIMATION)

3. ΒΕΛΤΙΣΤΟΠΟΊΗΣΗ ΚΑΙ ΤΕΧΝΙΚΈΣ
    1. ΠΡΟΣΑΡΜΟΣΤΙΚΉ ΕΚΤΈΛΕΣΗ ΕΠΕΡΩΤΗΣΕΩΝ
    2. ΜΕΘΟΔΟΛΟΓΊΕΣ ΠΡΟΣΑΡΜΟΣΤΙΚΉΣ ΕΚΤΈΛΕΣΗΣ ΕΠΕΡΩΤΉΣΕΩΝ
    3. ΤΉΡΗΣΗ ΣΤΑΤΙΣΤΙΚΏΝ
    4. ΤΕΧΝΙΚΈΣ ΚΑΤΆ ΤΗ ΒΕΛΤΙΣΤΟΠΟΊΗΣΗ
        1. ΒΕΛΤΙΣΤΟΠΟΊΗΣΗ ΒΆΣΕΙ ΡΥΘΜΟΎ (RATE-BASED)
        2. ΔΙΑΧΕΊΡΙΣΗ ΠΌΡΩΝ
        3. ΠΟΙΌΤΗΤΑ ΥΠΗΡΕΣΙΑΣ ΚΑΙ ΕΛΆΤΤΩΣΗ ΦΌΡΤΟΥ (LOAD SHEDDING)
        4. ΑΝΑΚΑΤΑΝΟΜΉ ΕΠΕΡΩΤΗΣΕΩΝ (QUERY SCRAMBLING)
        5. ΣΤΡΌΒΙΛΟΙ (EDDIES) ΚΑΙ ΚΛΆΔΟΙ ΕΥΡΕΤΗΡΊΩΝ (STEMS)
    5. ΕΙΔΙΚΆ ΘΈΜΑΤΑ ΚΑΤΆ ΤΗ ΒΕΛΤΙΣΤΟΠΟΊΗΣΗ
        1. ΦΙΛΤΡΆΡΙΣΜΑ ΡΟΏΝ (STREAM FILTERING)
        2. ΕΠΙΔΡΑΣΗ ΣΤΊΞΗΣ ΠΆΝΩ ΣΤΙΣ ΡΟΈΣ (PUNCTUATION)
        3. ΧΡΉΣΗ ΠΛΑΙΣΊΩΝ (WINDOWS)
        4. ΣΥΝΌΨΕΙΣ (SYNOPSES)

4. ΕΦΗΡΜΟΣΜΈΝΗ ΈΡΕΥΝΑ ΣΤΙΣ ΡΟΈΣ ΔΕΔΟΜΈΝΩΝ
    1. ΓΛΏΣΣΕΣ ΣΥΝΕΧΏΝ ΕΠΕΡΩΤΉΣΕΩΝ

    1. Η ΓΛΏΣΣΑ CQL (CONTINUOUS QUERY LANGUAGE)
    2. Η ΓΛΏΣΣΑ ESL (EXPRESSIVE STREAM LANGUAGE FOR TIME SERIES)
2. ΕΡΕΥΝΗΤΙΚΆ ΣΥΣΤΉΜΑΤΑ ΔΙΑΧΕΊΡΙΣΗΣ ΡΟΏΝ ΔΕΔΟΜΈΝΩΝ
    1. ΤΟ ΣΎΣΤΗΜΑ TELEGRAPH CQ
    2. ΤΟ ΣΎΣΤΗΜΑ STREAM (STANFORD STREAM DATA MANAGER)
    3. ΤΟ ΣΎΣΤΗΜΑ AURORA

(ΔΕΎΤΕΡΟ ΜΈΡΟΣ)

5. ΜΕΛΈΤΗ ΤΟΥ ΠΕΔΊΟΥ ΤΗΣ ΠΑΡΑΓΩΓΉΣ ΔΟΚΙΜΑΣΤΙΚΏΝ ΡΟΏΝ ΔΕΔΟΜΈΝΩΝ
    1. Η ΑΝΑΓΚΑΙΌΤΗΤΑ ΓΙΑ ΕΡΓΑΛΕΊΑ ΠΑΡΑΓΩΓΉΣ ΡΟΏΝ ΔΕΔΟΜΈΝΩΝ
    2. ΕΊΔΗ ΡΟΏΝ ΔΕΔΟΜΈΝΩΝ ΣΤΗΝ ΠΡΆΞΗ
        1. ΡΟΈΣ ΑΠΌ ΔΊΚΤΥΑ ΑΙΣΘΗΤΉΡΩΝ (SENSOR NETWORKS)
        2. ΡΟΈΣ ΑΠΌ ΔΊΚΤΥΑ ΔΕΔΟΜΈΝΩΝ (DATA NETWORKS)
        3. ΕΜΦΩΛΕΥΜΈΝΕΣ ΡΟΈΣ (NESTED STREAMS)
        4. ΡΟΈΣ ΑΠΌ ΑΡΧΕΊΑ ΙΣΤΟΤΌΠΩΝ (WEB LOGs)
        5. ΡΟΈΣ ΑΠΌ ΔΕΔΟΜΈΝΑ ΟΙΚΟΝΟΜΙΚΉΣ ΠΡΟΈΛΕΥΣΗΣ
    3. ΥΠΆΡΧΟΝΤΑ ΕΡΓΑΛΕΊΑ ΠΑΡΑΓΩΓΉΣ ΡΟΏΝ ΔΕΔΟΜΈΝΩΝ
        1. ΤΟ “LINEAR ROAD BENCHMARK”
        2. ΤΟ “NEXMARK BENCHMARK”

6. ΑΝΆΠΤΥΞΗ ΕΡΓΑΛΕΊΟΥ ΠΑΡΑΓΩΓΉΣ ΔΟΚΙΜΑΣΤΙΚΏΝ ΡΟΏΝ ΔΕΔΟΜΈΝΩΝ
    1. Η ΑΝΆΓΚΗ ΓΙΑ ΠΕΡΑΙΤΈΡΩ ΈΡΕΥΝΑ ΚΑΙ ΑΝΆΠΤΥΞΗ ΣΤΟ ΧΏΡΟ
    2. ΈΛΛΕΙΨΗ ΡΟΏΝ ΔΕΔΟΜΈΝΩΝ ΓΙΑ ΔΟΚΙΜΑΣΤΙΚΟΎΣ ΣΚΟΠΟΎΣ
        1. ΑΔΥΝΑΜΊΑ ΛΉΨΗΣ ΡΕΑΛΙΣΤΙΚΏΝ ΡΟΏΝ ΔΕΔΟΜΈΝΩΝ
        2. ΑΝΆΓΚΗ ΓΙΑ ΔΉΛΩΣΗ ΕΝΑΛΛΑΚΤΙΚΏΝ ΣΕΝΑΡΊΩΝ
    3. ΠΕΡΙΓΡΑΦΉ ΤΟΥ ΕΡΓΑΛΕΊΟΥ ΠΑΡΑΓΩΓΉΣ ΔΟΚΙΜΑΣΤΙΚΏΝ ΡΟΏΝ ΔΕΔΟΜΈΝΩΝ
    4. ΟΙ ΡΟΈΣ ΠΟΥ ΠΑΡΆΓΟΝΤΑΙ
        1. ΑΠΛΈΣ ΡΟΈΣ
        2. ΥΠΌ-ΡΟΈΣ
        3. ΔΙΑΔΟΧΈΣ ΡΟΏΝ
    5. ΕΝΝΟΙΟΛΟΓΙΚΌ ΠΛΑΊΣΙΟ

6. ΥΠΟΣΤΗΡΙΚΤΙΚΉ ΓΛΏΣΣΑ ΟΡΙΣΜΟΎ ΡΟΏΝ
7. ΣΧΕΔΊΑΣΗ ΚΑΙ ΑΝΆΠΤΥΞΗ ΤΟΥ ΛΟΓΙΣΜΙΚΟΎ
    1. ΤΡΌΠΟΣ ΣΧΕΔΊΑΣΗΣ
    2. ΠΕΡΙΓΡΑΦΉ ΤΩΝ ΑΡΘΡΩΜΆΤΩΝ
    3. ΠΑΡΟΥΣΊΑΣΗ ΛΟΓΙΣΜΙΚΟΎ
8. ΑΠΟΛΟΓΙΣΜΌΣ ΥΛΟΠΟΊΗΣΗΣ
9. ΕΝΔΙΑΦΕΡΟΝΤΑ ΣΗΜΕΙΑ ΥΛΟΠΟΊΗΣΗΣ
10. ΕΡΓΑΛΕΊΟ ΚΑΤΑΝΆΛΩΣΗΣ ΡΟΏΝ

7. ΕΠΊΛΟΓΟΣ

Προς όσους αισθάνονται δυνατοί:

“Προσπάθησα πολύ να σταθώ πλάι σας,
και δεν κατάλαβα πολλά!
Τώρα μπορείτε να αποσυρθείτε.
Δεν είμαι δικός σας, έχετε ηττηθεί,
ανήκω αλλού”

«Δεν ελπίζω τίποτα, δε φοβάμαι τίποτα, είμαι ελεύθερος»

Ν. Καζαντζάκης

Ευχαριστίες

Τούτη τη φορά αισθάνομαι την ανάγκη να ευχαριστήσω τους ανθρώπους εκείνους που με βοήθησαν σε αυτό το βήμα της ζωής μου. Κι αυτό γίνεται ενώ δεν το περιμένεις – όσο κι αν το ξέρεις. Η περίοδος που πέρασε στάθηκε ιδιαίτερα διδακτική για μένα. Το γεγονός ότι κάτι ολοκληρώθηκε δεν είναι καθόλου δικό μου επίτευγμα. Άλλα διάβαζα και άλλα έμαθα. Αλλά καλώς έχει και αυτό.

Ευχαριστώ τους ανθρώπους που με αγαπούν και δεν χρειάζονται να τους ευχαριστήσω.

“Χωρίς εσάς δε θα είχε κανένα απολύτως νόημα”

Σας αγαπώ πολύ κι εγώ.

Τάκης Κ. Μπουγιούρης

ΠΕΡΊΛΗΨΗ ΣΤΑ ΕΛΛΗΝΙΚΆ

Σε αυτή τη διπλωματική εργασία ασχολούμαστε με τη μελέτη των Συστημάτων Διαχείρισης Ροών Δεδομένων (ΣΔΡΔ – Data Stream Management Systems, DSMS). Μιλώντας για ΣΔΡΔ, εννοούμε τον τομέα εκείνο των Συστημάτων Διαχείρισης Βάσεων Δεδομένων (ΣΔΒΔ) που εξειδικεύονται στη διαχείριση όχι στατικών, αλλά δεδομένων που εμφανίζονται με τη μορφή ροών. Σε ένα ΣΔΒΔ τα δεδομένα υφίστανται αλλαγές έπειτα από κάποια εξωτερική ενεργοποίηση είτε του συστήματος, είτε ανθρώπινη. Αυτός ο τρόπος αντιμετώπισης δεν είναι ικανός για να αντιμετωπίσει τη νέα γενιά σύγχρονων εφαρμογών που συνεχώς αναδύονται και οι οποίες απαιτούν την επεξεργασία και διαχείριση δεδομένων που αδιάλειπτα καταφθάνουν σχηματίζοντας συνεχείς ροές.

Μια ¨ροή δεδομένων¨ είναι μια θεωρητικά απεριόριστη ακολουθία πλειάδων, μια απεριόριστη, δηλαδή, σειρά δεδομένων που παράγονται από κάποιο σύστημα και τις οποίες ένα ΣΔΡΔ οφείλει να διαχειριστεί με όσο πιο πλήρη και αποδοτικό τρόπο. Γίνεται γρήγορα κατανοητό ότι οι δυσκολίες που παρουσιάζονται σε ένα τέτοιο εγχείρημα, είναι αρκετά διαφορετικές από αυτές που παρουσιάζονται σε ένα ΣΔΒΔ, λόγω των ιδιαιτεροτήτων που χαρακτηρίζουν τις ροές. Μερικές από αυτές που αβίαστα εντοπίζονται είναι η στατικότητα των επερωτήσεων, σε αντίθεση με τη στατικότητα των δεδομένων στα κλασσικά ΣΔΒΔ, η μαζικότητα και οι υψηλοί ρυθμοί εμφάνισης των δεδομένων και η ανάγκη για επεξεργασία σε πραγματικό χρόνο.

Μερικές από τις πιο γνωστές εφαρμογές που παράγουν ροές στις μέρες μας είναι τα δίκτυα αισθητήρων (sensor networks), τα οποία αποτελούν μία από τις ταχύτερα εξελισσόμενες τεχνολογίες, τα κλασσικά δίκτυα δεδομένων (data networks), τα οποία παράγουν μεγάλες ποσότητες ροών, η καταγραφή αρχείων ιστοτόπων και εφαρμογών διαδικτύου (web logs) και εφαρμογές που χειρίζονται δεδομένα οικονομικής και ενημερωτικής προέλευσης (πχ. on-line δημοπρασίες). Τα παραδείγματα αυτά, αν και είναι χαρακτηριστικά, δεν είναι τα μόνα που μπορούν να καταγραφούν μιας και οι ποσότητες πληροφορίας που διακινούνται πλέον, διαρκώς και αυξάνονται. Στην παρούσα διπλωματική εργασία γίνεται αναφορά σ' αυτές ακριβώς τις εφαρμογές που παράγουν ροές δεδομένων καθώς και στις ιδιαιτερότητες που η κάθε μία παρουσιάζει.

Η εργασία αυτή ξεκινά με μια προσπάθεια για αποσαφήνιση των βασικών εννοιών που σχετίζονται με τις ροές δεδομένων και τα χαρακτηριστικά τους. Παράλληλα, γίνεται αναφορά στις απαιτήσεις που η σύγχρονη ερευνητική δραστηριότητα έχει προδιαγράψει για τα αντίστοιχα ΣΔΡΔ. Η μελέτη που πραγματοποιήθηκε θέτει αρχικά σε αντιπαραβολή τις ανάγκες και τις λειτουργικότητες των ΣΔΡΔ σε αντίθεση με τα παραδοσιακά ΣΔΒΔ. Έπειτα γίνεται εμβάθυνση στα πρώτα.

Μία από τις πολύ σημαντικές πτυχές της διαχείρισης ροών δεδομένων που αναλύονται, είναι η εκτέλεση συνεχών επερωτήσεων (continuous queries). Οι συνεχείς επερωτήσεις είναι επερωτήσεις που εκτελούνται συνεχόμενα και όχι μία φορά, είτε αποκλειστικά πάνω σε ροές δεδομένων, είτε πάνω σε συνδυασμούς ροών και στατικών δεδομένων. Γίνεται εύκολα κατανοητό ότι η κλασσική προσέγγιση στην εκτέλεση επερωτήσεων δεν ανταποκρίνεται παρά ελάχιστα στην περίπτωση αυτή. Αυτό συμβαίνει, επειδή, παρόλο που οι ροές που μελετάμε ακολουθούν το σχεσιακό μοντέλο, οι όγκοι των δεδομένων που συμμετέχουν σε αυτές είναι απρόβλεπτα μεγάλοι, ενώ οι ρυθμοί με τους οποίους εμφανίζονται είναι επίσης υψηλοί, με αποτέλεσμα, η διαχείριση που ένα ΣΔΡΔ προσπαθεί να επιτελέσει, να θέτει σε δοκιμασία τη διαθεσιμότητα των πόρων του. Η ίδια η διαχείριση των πόρων του, όσον αφορά τη εξοικονόμηση μνήμης και υπολογιστικής ισχύος, αποτελούν ένα ξεχωριστό πεδίο έρευνας των ΣΔΡΔ που επίσης αναφέρεται.

Συνεχίζοντας αναφερόμαστε σε μεθοδολογίες και τεχνικές που αποσκοπούν στην βελτιστοποίηση της εκτέλεσης των συνεχών επερωτήσεων. Η αναγκαιότητα για κάτι τέτοιο είναι εμφανής αν αναλογιστεί κανείς ότι όταν οι ρυθμοί εισόδου των δεδομένων σε ένα ΣΔΡΔ γίνουν μοιραία ασυνήθιστα υψηλοί, αυτό θα πρέπει να έχει τη δυνατότητα να αντιδρά με τον κατάλληλο τρόπο, ώστε να διακινδυνεύει όσο το δυνατόν λιγότερο τη διαθεσιμότητά του. Μια από τις ευρέως αποδεκτές μεθοδολογίες που έχουν αναπτυχθεί σε ερευνητικό επίπεδο στις μέρες μας είναι η προσαρμοστική εκτέλεση των συνεχών επερωτήσεων. Σύμφωνα με αυτή, ένα ΣΔΡΔ υποχρεώνεται να αναπροσαρμόζει διαρκώς τον τρόπο με τον οποίο διεκπεραιώνει τις επερωτήσεις, έτσι ώστε να μπορεί να ανταποκρίνεται στις εκάστοτε συνθήκες. Συχνά, είναι απαραίτητο να καταφύγει σε προσεγγίσεις που υποβαθμίζουν την ακρίβεια των αποτελεσμάτων του στην προσπάθειά του να εξοικονομήσει τους αναγκαίους πόρους για την εξυπηρέτηση του συνόλου των επερωτήσεων που έχει αναλάβει. Μερικοί τρόποι για την επίτευξη αυτού είναι η ελάττωση του φόρτου δεδομένων που δέχεται μέσω της απόρριψης πλειάδων, η προσεγγιστική εκτέλεση των επερωτήσεων, ο διαμοιρασμός των πόρων και η ανακατανομή των επερωτήσεων.

Ο χώρος της διαχείρισης ροών δεδομένων έχει αρχίσει να παρουσιάζει σημαντική ερευνητική δραστηριότητα τα τελευταία χρόνια, κάτι που αποδεικνύει τόσο το αυξανόμενο ακαδημαϊκό, όσο και εφηρμοσμένο τεχνολογικό ενδιαφέρον που αφιερώνεται σε αυτόν. Τεχνολογίες όπως για παράδειγμα αυτή των δικτύων αισθητήρων εξελίσσονται με τόσο γοργούς ρυθμούς και διευρύνονται σε τόσο πολλά πεδία εφαρμογής, ώστε είναι βέβαιο ότι θα συνεχίσουν να απασχολούν την ερευνητική κοινότητα και στα προσεχή χρόνια. Γι' αυτό τον λόγο, στην παρούσα εργασία, γίνεται αναφορά σε ορισμένες από τις πιο διακεκριμένες σύγχρονες ερευνητικές

προσπάθειες. Μέσα σε αυτές εντοπίζουμε τις γλώσσες συνεχών επερωτήσεων CQL (Continuous Query Language) και ESL (Expressive Stream Language). Η CQL βασίζει το συντακτικό της στην δηλωτική γλώσσα SQL και υποστηρίζει επερωτήσεις πάνω σε ροές και συνδυασμούς ροών με στατικά σχεσιακά δεδομένα. Η ESL είναι λιγότερο στενά δεμένη με την SQL, αν και στηρίζεται σε αυτήν, και επιχειρεί να δώσει μεγαλύτερη εκφραστικότητα στις επερωτήσεις.

Ακόμη, γίνεται αναφορά στα ερευνητικά ΣΔΡΔ STREAM, Aurora και TelegraphCQ. Το STREAM φιλοδοξεί να γίνει ένα γενικού σκοπού ΣΔΡΔ που θα υποστηρίζει συνεχείς επερωτήσεις που θα εκτελούνται πάνω σε πολλαπλές εισερχόμενες ροές και αποθηκευμένους πίνακες. Χρησιμοποιεί τη γλώσσα συνεχών επερωτήσεων CQL και εφαρμόζει την τεχνική της προσαρμογής των πλάνων εκτέλεσης των επερωτήσεων για το μέγιστο δυνατό διαμοιρασμό πόρων ανάμεσά τους. Το TelegraphCQ έχει την ιδιαιτερότητα ότι η ανάπτυξή του βασίστηκε πάνω στο πολύ γνωστό και ανοικτού κώδικα ΣΔΒΔ PostgreSQL. Η πρόκληση που αντιμετωπίστηκε στο εγχείρημα αυτό αφορούσε στη διαμόρφωση του υπάρχοντος ΣΔΒΔ ώστε να υποστηρίζει συνεχείς επερωτήσεις, ενώ χρησιμοποιεί τεχνικές όπως οι στρόβιλοι (eddies) και ο προγραμματισμός τελεστών (operator scheduling). Η στόχευση του Aurora είναι στο να αναπτυχθεί μια πλήρης εφαρμογή παρακολούθησης (monitoring) ροών δεδομένων, όπου εφαρμόζεται προσεγγιστική εκτέλεση των επερωτήσεων.

Έπειτα από τη βιβλιογραφική μελέτη που προηγήθηκε, αναγνωρίστηκε η αναγκαιότητα για ύπαρξη εξειδικευμένων εργαλείων παραγωγής δοκιμαστικών ροών δεδομένων, τα οποία να λειτουργούν υποβοηθητικά στην ανάπτυξη των εκκολαπτόμενων ΣΔΡΔ ή των αλγορίθμων που τα απαρτίζουν. Αυτό συμβαίνει διότι σε πολλές εφαρμογές παρουσιάζεται αδυναμία λήψης δοκιμαστικών ροών, αλλά ακόμη και όταν αυτό είναι δυνατό, παραμένει η ανάγκη για ποικιλία δεδομένων που θα εξυπηρετούν τις σκοπιμότητες όλων των δοκιμών. Δύο από τα ερευνητικά εργαλεία που έχουν κάνει την εμφάνισή τους στο πεδίο αυτό είναι το ¨Linear Road¨ και το ¨NEXMark¨. Το πρώτο προσομοιώνει ένα σενάριο παρακολούθησης της κυκλοφορίας σε μια μεγάλη πόλη και της μεταβαλλόμενης χρέωσης μέσω διοδίων των οχημάτων που κινούνται σε αυτή. Το δεύτερο προσομοιώνει μια on-line δημοπρασία. Και τα δύο αποσκοπούν στο να θέσουν σε δοκιμασία (benchmark) ένα ΣΔΡΔ που θα τα αξιοποιήσει.

Αναγνωρίζεται λοιπόν η έλλειψη για ένα εργαλείο παραγωγής δοκιμαστικών ροών δεδομένων που να τις προσεγγίζει με τρόπο γενικευμένο (generic). Δηλαδή, να δίνει τη δυνατότητα σε ένα χρήστη να ορίσει εκείνος τα είδη των ροών δεδομένων που χρειάζεται και να περιορίζεται κατά το δυνατόν λιγότερο από τη σημασιολογία που το εργαλείο του δίνει. Έτσι, στα πλαίσια της παρούσας

εργασίας, έγινε προσπάθεια για μια γενικευμένη θεώρηση της παραγωγής των ροών δεδομένων, η οποία να δίνει όσο το δυνατόν πιο μεγάλη ευελιξία στη δήλωσή τους από το χρήστη. Το εγχείρημα αυτό υποστηρίζεται από την παρατήρηση και καταγραφή αρκετών πραγματικών εφαρμογών που παράγουν ροές δεδομένων και έχει αντιμετωπιστεί με τρόπο που να είναι καλά σχεδιασμένος και εξελίξιμος στο μέλλον. Το αποτέλεσμα αυτού ήταν η πρωτογενής υλοποίηση ενός εργαλείου παραγωγής δοκιμαστικών ροών δεδομένων που να δηλώνονται σε XML μορφή.

Ο χώρος της διαχείρισης ροών δεδομένων μελετήθηκε σε σημαντικό εύρος, έτσι ώστε να εντοπιστούν τα στοιχεία εκείνα που διακρίνουν τις ροές και τις κάνουν να ξεχωρίζουν από τα δεδομένα που διαχειρίζονται τα παραδοσιακά ΣΔΒΔ. Έγινε αναφορά στις ιδιαιτερότητές τους, στα προβλήματα που ανακύπτουν κατά τη διαχείρισή τους, στους τρόπους με τους οποίους προτείνεται η βελτιστοποίηση της επεξεργασίας των επερωτήσεων πάνω τους καθώς και σε προγράμματα εφηρμοσμένης έρευνας εξειδικευμένων σε αυτές. Στα πλαίσια αυτά, έγινε προσπάθεια για μια πρώτη υλοποίηση ενός γενικευμένου εργαλείου παραγωγής δοκιμαστικών ροών δεδομένων. Είναι βέβαιο πως οι ροές δεδομένων παρουσιάζουν εξαιρετικό ενδιαφέρον στις μέρες μας και η ανάγκη για την καλύτερη κατανόησή τους δεν θα πάψει να υφίσταται.

EXTENDED ABSTRACT

In this master thesis we have dealt with the study of Data Stream Management Systems (DSMS). When we refer to DSMS's we mean the specific field of Database Management Systems (DBMS) that are specializing in the management of data that appear in the form of streams rather than static data. In a DBMS data are infrequently subject to changes, which occur as a result of an external trigger either human or system originated. This means of manipulating data is not appropriate for facing the new genre of contemporary applications, that are constantly emerging and demand the processing and management of data that are infinitely arriving forming streams.

A ¨data stream¨ is a theoretically infinite sequence of tuples, that is an infinite row of data which are produced by a certain system and which are to be managed by a DSMS in a most complete and efficient manner. It can be easily understood that the difficulties that such an endeavour entails, are fairly different from those appearing in a DBMS due to the individualities of streams. Some of the most distinct ones are query persistency in contrast with data persistency in classical DBMS's, the massiveness and high rates of data and the need for real time processing.

Some of the most noticeable applications that generate streams nowadays are sensor networks, one of the most rapidly developing technologies, classical data networks which produce large quantities of streams, web site and web application logs and applications that handle data of financial origin. (eg. on-line auctions). These examples, although typical, are not the only ones that can be recorded since the quantities of information that are exchanged, continually increase. The present master thesis mentions exactly the applications which produce data streams as well as the individualities each one presents.

This master thesis begins with an endeavour for clarifying the basic concepts that relate to data streams and their characteristics. At the same time we examine the features that contemporary research has recorded as demanding for the DSMS's. The conducted study juxtaposes the necessities and the functionalities of the DSMS's to those of DBMS's. Than we delve into DSMS's. One of the most important aspects of Data Stream Management that are analysed is continuous query execution. Continuous queries are queries which are executed in a constant manner and not only once either exclusively on data streams or on combinations of data streams and static data. It is easily understood that the traditional approach of executing queries is vaguely appropriate for this case. This happens because, although the streams we study do follow the relational model, the transmitted data volumes can be unexpectedly large, while the rates of the appearance are equally

high and as a result the management that the DSMS tries to achieve, challenges the availability of its resources. The actual management of its resources, as far as saving up memory and computational power, is a separate field of research of the DSMS's which is also mentioned herein.

Furthermore, we refer to methodologies and techniques that focus on the optimization of the continuous query execution. The need for such an optimisation is apparent considering that when data input rates in a DSMS fatally become too high, it should have the ability to react in an appropriate manner, so as not to endanger its availability. One of the most widely acceptable methodologies that have been developed in research nowadays is the adaptive query execution. According to this, a DSMS is obliged to continually reorganise the way in which query execution is planned, so that it can respond effectively in all circumstances. It is often necessary to resort to solution that degrade the quality of its results in an attempt to save up the necessary resources for serving all the queries it has undertaken. Some of the ways to achieve this are load shedding by discarding tuples, approximate query execution, resource sharing and rearrangement of query plans.

The field of data stream management has recently started to present important research activity. This proves not only the increasing academic interest but also the applied technological one. Technologies such as this of sensor networks evolve so rapidly and expand in so many application fields that it is safe to say they will continue to preoccupy a large part of the scientific community for years to come. That's why in the present master thesis we mention some of the most noticeable contemporary research efforts. Among them we locate the continuous query languages CQL (Continuous Query Languages) and ESL (Expressive Stream Language). CQL syntax is based on declarative language SQL and supports queries on streams and combinations of streams with static relational data. ESL is less related to SQL even though it is based on it and tries to give greater expressiveness to queries.

Moreover, we mention the research of DSMS's STREAM, Aurora and TelegraphCQ. STREAM has the ambition to become a DSMS of general aim which will support continuous queries conducted on multiple input streams and stored tables. It uses the Continuous Query Language CQL and materializes the technique of adaptive query execution for the greatest possible resource sharing. TelegraphCQ is unique in that its development was based on the well known and open source DBMS PostgreSQL. The challenge that this effort faced pertained to the rearrangement of the existing DBMS so that it can support continuous queries while it uses techniques such as eddies and operator scheduling. Aurora aims at developing a full monitoring of data streams where approximate query execution is adopted.

After the study of bibliography that preceded, the need for specialized data stream generating tools was identified. These would function as auxiliary tools in the development of DSMS's to be or the algorithms that constitute them. This happens because in many applications it is not possible to obtain testing data streams, but even when this is possible, there still lies the need for a variety of data which will serve the purposes of all kinds of tests. Two of these research tools that have been recorded in this field are the ¨Linear Road¨ and the ¨NEXMark¨. The first one simulates a scenario of traffic monitoring in a big city and the variable tolling of the vehicles moving within it. The second one simulates an on-line auction. Both of them aim at benchmarking DSMS's.

Thus we recognize the need for data stream generating tool which will approach streams in a generic way. That is, to give the ability to a user to define the kinds of data streams they need for their tests and not to constrain them to the semantics that the tools provides. So, as part of the present thesis, there has been an effort to develop a generic way of conceptually approaching data streams which will give the greatest possible flexibility in declaring them. This effort is backed up by the observation and recording of many real applications of data stream and it has been dealt with a well structured, evolvable way. The result of this was the elementary development of a data stream generating tool in XML language.

The field of data stream management was studied widely enough, so that the elements determining data streams from traditional relational data could be accented. We have also referred to their peculiarities, the problems emerging during their management, the ways the optimization of the execution of continuous queries is proposed as well as to projects of applied research specializing in them. Therefore, there has been an effort for a first development of a generic data stream generating tool. It is more than certain that data streams are presented with exceptional interest nowadays and the need for a better understanding will not cease to exist.

0. ΠΡΌΛΟΓΟΣ

Τα τελευταία χρόνια μια νέα γενιά εφαρμογών έχει αναδυθεί στην επιφάνεια και η οποία έχει την ανάγκη για διαχείριση ροών δεδομένων. Με τον όρο αυτό εννοούμε δεδομένα απαρτιζόμενα από συνεχή και πραγματικού χρόνου διαδοχή αντικειμένων. Τα αντικείμενα αυτά δεν είναι παρά δομημένη πληροφορία που παράγεται από ένα σύστημα και χρήζει επεξεργασίας, αποθήκευσης και ανάκτησης. Η λέξη κλειδί για αυτά τα δεδομένα είναι ότι είναι "συνεχή", καταφθάνουν, δηλαδή, σχεδόν ακατάπαυστα.

Τα κλασσικά συστήματα διαχείρισης βάσεων δεδομένων είχαν αρχικά αναπτυχθεί για να ανταποκριθούν στις ανάγκες εμπορικών εφαρμογών, όπου τα δεδομένα υφίστανται αλλαγές κατά κύριο λόγο σαν αποτέλεσμα διαδικασιών που εκκινούνται από ανθρώπινη παρέμβαση. Το ίδιο και η ανάκτησή τους. Έτσι, δρουν σε μεγάλο βαθμό σαν παθητικοί αποδέκτες δεδομένων, που ενεργοποιούνται εκτελώντας αποθηκευτικές λειτουργίες και λειτουργίες ανάκτησης όταν αυτό τους ζητηθεί. Οι ανάγκες των εφαρμογών που παράγουν – ή δέχονται - "συνεχείς ροές δεδομένων" δε μπορούν να αντιμετωπιστούν με τα κλασσικά συστήματα, όσο καλή και αποδοτική και αν είναι η διαχείριση που αυτά επιτελούν. Ο λόγος, είναι, φυσικά, γιατί δεν έχουν σχεδιαστεί για τέτοιες εφαρμογές.

Κάτι που ξεχωρίζει μια συνεχή ροή δεδομένων είναι το ότι πιθανότατα είναι απεριόριστη, ή, τουλάχιστο, μπορεί να θεωρηθεί απεριόριστη αν ιδωθεί υπό τις παραδοσιακές προϋποθέσεις διαχείρισης. Οι εφαρμογές που μπορούν να χαρακτηριστούν πως ανήκουν στο χώρο των "συνεχών ροών" έχουν βρεθεί σε σημαντική θέση σήμερα τόσο για εμπορικούς όσο και για τεχνικούς λόγους. Για παράδειγμα, από πλευράς τεχνολογίας, είναι εφικτή στις μέρες μας η ύπαρξη πλειάδας εφαρμογών παρακολούθησης (monitoring) μέσω δικτύων αισθητήρων (sensor networks), οι οποίοι είναι πλέον αρκετά φθηνοί και έτοιμοι για εμπορικούς σκοπούς. Δεδομένα που προέρχονται από τέτοια συστήματα είναι κατά πάσα πιθανότητα απεριόριστα σε αριθμό, καθώς συνεχώς νέα παράγονται, και πρέπει, ταυτόχρονα να υφίστανται επεξεργασία σε πραγματικό χρόνο. Οι χώροι στους οποίους παρόμοιες εφαρμογές βρίσκουν έδαφος είναι ποικίλοι: στρατιωτικές εφαρμογές, δίκτυα αισθητήρων, συστήματα παρακολούθησης, διαχείριση δικτύων, ανάλυση αρχείων log, ανίχνευση απάτης πιστωτικών καρτών σε πραγματικό χρόνο κα.

Επομένως, οι εφαρμογές που βασίζονται σε ροές δεδομένων εισάγουν νέες προκλήσεις στη διαχείριση των δεδομένων. Τα συστήματα διαχείρισης συνεχών ροών δεδομένων (data stream management systems) απαιτούν οι επερωτήσεις πάνω στα δεδομένα να εκτελούνται συνεχώς,

αντίθετα με τις επερωτήσεις των παραδοσιακών συστημάτων βάσεων δεδομένων όπου μια επερώτηση εκτελείται σε διακριτές χρονικές στιγμές. Τα δεδομένα που έρχονται σε "ροές" διογκώνονται σε κάθε μονάδα του χρόνου και οι επερωτήσεις που θέλουμε να εκτελέσουμε πάνω σε αυτά οφείλουν να λαμβάνουν υπόψη αυτή τους την ιδιότητα. Ακόμη, το γεγονός ότι ένα μεγάλο ποσοστό των εφαρμογών συνεχών ροών δεδομένων έχουν ως αντικείμενό τους την παρακολούθηση κάποιων οντοτήτων (monitoring), κάνει τα συστήματα που τις υποστηρίζουν να συμπεριλαμβάνουν στις προδιαγραφές τους τη δυνατότητα ανάδρασης ανάλογα με τα αποτελέσματα της επεξεργασίας της ροής.

Όλες αυτές οι ανάγκες και δυνατότητες, είτε εμπορικές είτε τεχνολογικές, έχουν οδηγήσει ένα μέρος της ερευνητικής δραστηριότητας στο χώρο της διαχείρισης δεδομένων στην επανεξέταση και επαναδιατύπωση των περισσότερων λειτουργιών ενός συστήματος διαχείρισης βάσεων δεδομένων ώστε να υποστηρίξει ροές δεδομένων. Με αυτές τις σκέψεις κατανοεί κανείς τη μεγάλη αύξηση της έρευνας στο χώρο των "ροών δεδομένων" και της διαχείρισής τους. Αυτό θα είναι και το θέμα της παρούσας εργασίας.

ΠΡΏΤΟ ΜΈΡΟΣ

1. ΕΙΣΑΓΩΓΙΚΆ ΓΙΑ ΤΙΣ ΡΟΈΣ ΔΕΔΟΜΈΝΩΝ

Τα παραδοσιακά Συστήματα Διαχείρισης Βάσεων Δεδομένων (ΣΔΒΔ) αναπτύχθηκαν αρχικά με γνώμονα την υποστήριξη εμπορικών εφαρμογών και τις ιδιαίτερες ανάγκες που εκείνες εμφανίζουν. Τα δεδομένα σε αυτού του τύπου τις εφαρμογές υφίστανται αλλαγές κυρίως έπειτα από κάποια ενεργοποίηση του συστήματος, συχνά ανθρώπινη. Οι αλλαγές ή οι ανακτήσεις πληροφορίας σε αυτά γίνονται σε συγκεκριμένες δοσοληψίες (transactions). Το Σύστημα Διαχείρισης λειτουργεί σαν ένα παθητικό αποθηκευτικό σύστημα για τα δεδομένα, εκτελώντας λειτουργίες όταν αυτές κατατεθούν από το χρήστη. Αυτή η φιλοσοφία αντιμετώπισης των δεδομένων δεν επαρκεί για να αντιμετωπίσει μια νέα γενιά εφαρμογών, οι οποίες απαιτούν την αδιάλειπτη επεξεργασία και διαχείριση δεδομένων που συνεχώς καταφθάνουν σε απεριόριστες ροές και μάλιστα σε πραγματικό χρόνο.

1.1 ΕΙΣΑΓΩΓΉ - ΟΡΙΣΜΟΊ

Μια "ροή δεδομένων" είναι μια θεωρητικά απεριόριστη ακολουθία πλειάδων, μια απεριόριστη, δηλαδή, σειρά δεδομένων που παράγονται από κάποιο σύστημα και ένα Σύστημα Διαχείρισης Ροών Δεδομένων - ΣΔΡΔ - (Data Stream Management System) οφείλει να διαχειριστεί με όσο πιο πλήρες και αποδοτικό τρόπο [4][3][47].

Ως "ακολουθία", τα στοιχεία της είναι διακριτά και διατεταγμένα – έχουν, επομένως, σειρά εμφάνισης και διάταξη. Και ακόμη, μιλώντας για "πλειάδες", εννοούμε ότι τα δεδομένα της ροής δεν είναι ακατάληπτα αλλά δομημένα σε συμφωνημένη ανάλογα με την εφαρμογή μορφή, όπως σε ένα οποιοδήποτε άλλο ΣΔΒΔ. Μερικά από τα σημεία που ξεχωρίζουν ένα Σύστημα Διαχείρισης Ροών Δεδομένων από ένα παραδοσιακό Σύστημα Διαχείρισης Βάσεων Δεδομένων είναι τα εξής:

1. **Στατικότητα** των επερωτήσεων (query persistency): Οι επερωτήσεις που γίνονται πάνω σε ροές δεδομένων είναι στατικές με την έννοια ότι προσδιορίζονται σε κάποια φάση ρύθμισης του συστήματος κι έπειτα εκτελούνται συνεχώς πάνω στα δεδομένα που καταφθάνουν [4]. Αυτό γιατί σε παρόμοια συστήματα αυτό που ενδιαφέρει είναι η παρακολούθηση ή/και καταγραφή κάποιας συγκεκριμένης μορφής της πληροφορίας και όχι όλη η πληροφορία καθαυτή [2], μιας και θεωρείται περαστική ή προσωρινή (temporal). Για παράδειγμα μπορεί να ενδιαφέρει η μέση ανά ώρα θερμοκρασία ενός αισθητήρα αλλά όχι όλες οι διαφορετικές θερμοκρασίες που αυτός έχει ανιχνεύσει. Αντίθετα στα παραδοσιακά ΣΔΒΔ τα δεδομένα είναι αυτά που είναι στατικά, δηλαδή καταχωρημένα

μόνιμα σε κάποιο αποθηκευτικό μέσο και οι επερωτήσεις που εκτελούνται πάνω σε αυτά μεταβάλλονται ανάλογα με τις εκάστοτε ανάγκες.

1. **Μαζικότητα** των δεδομένων: Τα δεδομένα καταφθάνουν στο σύστημα τόσο με συνεχή όσο και μαζικό τρόπο. Ο ρυθμός με τον οποίο γίνονται διαθέσιμα δεν είναι προκαθορισμένος και άρα μπορεί να μεταβάλλεται ανεξάρτητα από τις δυνατότητες σε ισχύ του συστήματος. Σε κάθε περίπτωση τα δεδομένα τέτοιων εφαρμογών έχουν συνεχή χαρακτήρα. Η αδυναμία πολλές φορές του συστήματος να γνωρίζει το ρυθμό εμφάνισης των δεδομένων δίνει στη διαχείρισή τους πολλές νέες πτυχές που πρέπει να ληφθούν υπόψη και σχετίζονται με την αυτόματη προσαρμογή του στις ανά τον χρόνο συνθήκες [3].

1. Επεξεργασία **σε πραγματικό χρόνο** (real time): Η επεξεργασία των ροών δεδομένων οφείλει να γίνεται όσο το δυνατόν πλησιέστερα στον πραγματικό χρόνο [1]. Ο λόγος για αυτό είναι ότι σε κάθε μονάδα χρόνου θα παράγονται νέα δεδομένα τα οποία θα πρέπει να εισέλθουν στο σύστημα για επεξεργασία και αυτά με τη σειρά τους. Επιπλέον, λόγω της συνεχούς ροής και της μαζικότητάς τους τα δεδομένα αυτά δεν είναι δυνατόν – και συχνά ούτε και αναγκαίο – να αποθηκεύονται ολόκληρα σε κάποιο αποθηκευτικό μέσο. Τις περισσότερες φορές η επεξεργασία τους παράγει κάποιο συγκεντρωτικό αποτέλεσμα και αυτό είναι το οποίο ενδιαφέρει το σύστημα και όχι οι ίδιες οι διακριτές τιμές των πλειάδων που δε σταματούν να έρχονται.

1.2 ΚΑΤΗΓΟΡΊΕΣ ΡΟΏΝ ΔΕΔΟΜΈΝΩΝ

Σαν μια πρώτη κατηγορία Συνεχών Ροών Δεδομένων μπορεί να διακρίνει κανείς [3] αυτές που χαρακτηρίζονται από τις δοσοληψίες (transactions). Οι συνεχείς ροές δοσοληψιών (transactional data streams) αποτελούν καταγραφές (logs) αλληλεπιδράσεων ανάμεσα σε διαφορετικές οντότητες. Σαν παραδείγματα θα μπορούσαμε να αναφέρουμε τις εξής εφαρμογές:

Σε σημαντικούς και με μεγάλη επισκεψιμότητα ιστοτόπους οι αλληλεπιδράσεις των χρηστών με το σύστημα καταγράφονται σε αρχεία (logs), τα οποία παρακολουθούνται από εφαρμογές που έχουν ως ευθύνη τους τον έλεγχο της απόδοσης του συστήματος και την εξατομίκευσή του για λογαριασμό των χρηστών του (personalisation). Η πληροφορία αυτών των αλληλεπιδράσεων συνεχώς καταγράφεται σε αυτό το αρχείο με τη μορφή νέων εγγραφών.

Οι τράπεζες που εκδίδουν πιστωτικές κάρτες παρακολουθούν την κίνηση των αγορών των πελατών

τους για να ανιχνεύσουν ανωμαλίες και να προλάβουν φαινόμενα απάτης. Οι καταγραφές των αγορών των πελατών δημιουργούν μια συνεχή ροή δεδομένων, η επεξεργασία των οποίων μπορεί να δώσει ενδείξεις σε περιπτώσεις απάτης.

Μια δεύτερη κατηγορία Συνεχών Ροών Δεδομένων [3] είναι αυτά που χρησιμοποιούνται για μετρήσεις. Στις εφαρμογές αυτές τα δεδομένα παράγονται σαν αποτέλεσμα της παρακολούθησης (monitor) της κατάστασης διαφόρων οντοτήτων ενδιαφέροντος. Μπορούμε να αναφέρουμε τα εξής παραδείγματα ενδεικτικά:

1. Καθώς οι τεχνολογία των αισθητήρων (sensors) γίνεται ολοένα και πιο προσιτή σε εμπορικές εφαρμογές ένας ολόκληρος νέος χώρος εφαρμογής γεννιέται, όπου αισθητήρες διασκορπισμένοι σε διάφορα φυσικά συστήματα θα μετρούν την κατάσταση συγκεκριμένων οντοτήτων που μας ενδιαφέρουν και θα παράγουν ροές δεδομένων με τις τιμές αυτές. Για παράδειγμα οι αισθητήρες θα μπορούν να μετρούν την κίνηση σε μια λεωφόρο ή τη θερμοκρασία σε ένα δάσος για την αντιμετώπιση πυρκαγιών.

1. Τα ευρείας κλίμακας δίκτυα επικοινωνιών απαιτούν τη συνεχή παρακολούθησή τους (monitoring) αναφορικά με διάφορες λειτουργίες τους, όπως ο εντοπισμός συμφορήσεων (bottlenecks) και η έγκαιρη διάγνωση επιθέσεων στο δίκτυο. Τα δεδομένα που παρακολουθούνται σε αυτή την περίπτωση είναι η πληροφορία στις επικεφαλίδες (headers) των πακέτων που πηγαινοέρχονται ανάμεσα σε μια ομάδα από δρομολογητές (routers). Αυτά τα δεδομένα μπορούν να αντιμετωπιστούν σα μια ροή.

Οι τυπικές επερωτήσεις που τα Συστήματα Διαχείρισης Συνεχών Ροών Δεδομένων συνήθως θέτουν πάνω στα δεδομένα είναι μακροχρόνιες και σταθερές. Αν η εφαρμογή καταθέσει στο σύστημα μια επερώτηση, τότε το σύστημα θα υπολογίζει συνεχόμενα το αποτέλεσμά της για μεγάλη χρονική περίοδο. Ο υπολογισμός αυτός μπορεί να γίνεται σε τακτά χρονικά διαστήματα και όχι κυριολεκτικά συνεχόμενα. Ας δούμε, όμως, ένα παράδειγμα συνεχούς επερωτήματος και τι αυτό τελικά σημαίνει.

Οι εφαρμογές που έχουν την ανάγκη ενός συστήματος διαχείρισης συνεχών ροών είναι ποικίλες, αν και πολλές από αυτές δεν αποτελούν πρωτοποριακές ιδέες, μιας και η αναγκαιότητά τους έχει παρατηρηθεί από καιρό [2]. Αντίθετα, πολλές από αυτές αναφέρονται σε συστήματα που υπήρξαν εδώ και καιρό ή σε εφαρμογές που λειτουργούν εδώ και χρόνια αλλά μέσω της αξιοποίησης της ιδιότητάς τους να διαθέτουν δεδομένα σε ροές φέρνουν εντελώς νέα αποτελέσματα. Για

παράδειγμα η διαχείριση των δεδομένων που παράγονται από τα στατιστικά στοιχεία ενός ιστοτόπου ή των δεδομένων που έρχονται σε μια μετεωρολογική υπηρεσία από τους διάφορους σταθμούς παρατήρησης είναι εφαρμογές που δεν εμφανίστηκαν καθόλου πρόσφατα αλλά τα στοιχεία του μπορούν με φυσικό τρόπο να ιδωθούν υπό το πρίσμα μιας ροής. Επιπλέον, εφαρμογές όπως τα δίκτυα αισθητήρων ή η παρακολούθηση στόλων μέσω GPS έχουν περισσότερο νεωτεριστικό χαρακτήρα, κυρίως λόγω του ότι η τεχνολογία τους έχει γίνει προσιτή μόλις τα τελευταία χρόνια σε μαζικότερο επίπεδο από τεχνολογική και οικονομική σκοπιά.

1.3 ΠΑΡΆΔΕΙΓΜΑ ΣΥΣΤΉΜΑΤΟΣ ΔΙΑΧΕΊΡΙΣΗΣ ΡΟΏΝ ΔΕΔΟΜΈΝΩΝ (ΣΔΡΔ)

Ας θεωρήσουμε ένα σύστημα που θα μπορούσε στην πραγματικότητα να υπάρχει. Μια λεωφόρος εξοπλισμένη με ένα σύστημα παρακολούθησης της κίνησης. Έστω ότι αφοσιωμένοι αισθητήρες διασκορπισμένοι κατά μήκος της λεωφόρου είναι επιφορτισμένοι με την ευθύνη να αναφέρουν τα στοιχεία που γνωρίζουν σχετικά με την κίνηση του δρόμου. Τα δεδομένα που θα παράγουν οι αισθητήρες αυτοί θα σχηματίζουν μια συνεχή ροή δεδομένων τόσο γιατί οι αναφορές τους δε θα πρέπει να διακόπτονται ποτέ όσο και γιατί το πλήθος τους θα οφείλει να είναι σημαντικό αν πρόκειται να παράγουν αξιόπιστα αποτελέσματα.

Έστω, λοιπόν, ότι ο κάθε αισθητήρας αποστέλλει για κάθε όχημα που διέρχεται από το χώρο δικαιοδοσίας του τα εξής στοιχεία: ένα διακριτικό για το όχημα (πχ αριθμός πινακίδας), την ταχύτητά του, ένα διακριτικό για τη λεωφόρο και συγκεκριμένα για το τμήμα της λεωφόρου στο οποίο ο αισθητήρας βρίσκεται, καθώς και την κατεύθυνση κίνησης του οχήματος. Έτσι κάθε μικρή αναφορά του αισθητήρα έχει τη μορφή μιας πλειάδας με τα τέσσερα προαναφερθέντα στοιχεία.

Οι επερωτήσεις που θα μπορούσαν να γίνουν πάνω σε αυτά τα δεδομένα ποικίλλουν:

- Για να βοηθήσει στη διαχείριση της ροής της κίνησης στη λεωφόρο μια επερώτηση θα μπορούσε να υπολογίζει τη μέση ταχύτητα των οχημάτων που περνούν από κάποιο τμήμα της λεωφόρου, μιας και για κάθε τέτοιο θα υπάρχει κάποιος υπεύθυνος αισθητήρας. Αν η μέση αυτή ταχύτητα πέσει κάτω από κάποιο κατώφλι τότε οι οδηγοί σε κάποιο σημείο στην αρχή της λεωφόρου θα μπορούν να ειδοποιούνται για να προτιμήσουν κάποια εναλλακτική διαδρομή.

- Βασισμένο πάνω στη μέση ταχύτητα ανά τμήμα της λεωφόρου του παραπάνω παραδείγματος, μια εφαρμογή θα μπορούσε να ειδοποιεί τους υπεύθυνους του οδικού

δικτύου για πιθανά τροχαία ατυχήματα που έχουν προκαλέσει κυκλοφοριακή συμφόρηση.

- Σε οδικά δίκτυα με σταθμούς διοδίων η ροή των δεδομένων που φέρει την πληροφορία των αυτοκινήτων που διέρχονται μπορεί να χρησιμοποιηθεί για την αυτόματη πληρωμή των διοδίων και την αφαίρεση του αντίστοιχου ποσού από το λογαριασμό του εκάστοτε οδηγού.

- Σε προχωρημένα μοντέλα διαχείρισης της κυκλοφορίας ένας από τους στόχους είναι η διαχείριση των κυκλοφοριακών συμφορήσεων. Για να επιτευχθεί αυτό συχνά χρησιμοποιείται η τακτική των κυμαινόμενων διοδίων, η οποία θα μπορούσε να υλοποιηθεί μέσω ενός αυτόματου συστήματος χρέωσης που θα εκμεταλλεύεται την ιδιότητα των δεδομένων να έρχονται σε ροές, όπως περιγράφηκε παραπάνω.

Παρόμοια παραδείγματα, είναι δυνατόν να συλλέξει κανείς σε πολλούς τομείς της σύγχρονης τεχνολογίας. Για παράδειγμα, στο χώρο της διαχείρισης δικτύων είναι έντονη ανάγκη για άμεση και ταχύτατη ανάλυση τεράστιων όγκων δεδομένων τα οποία παράγονται καθώς ένα δίκτυο βρίσκεται σε συνεχή χρήση. Επερωτήσεις όπως “ποια είναι η μέση χρήση σε εύρος ζώνης ενός δικτύου δεδομένων” ή “ποιες είναι οι πιο ογκώδεις μεταφορές δεδομένων που λαμβάνουν χώρα σε μια συγκεκριμένη χρονική στιγμή” ή “τι ποσοστό της πληροφορίας είναι κοινή ανάμεσα σε δύο διαφορετικούς δρομολογητές ” είναι κλασσικές στη διαχείριση δικτύων εδώ και καιρό.

1.4 ΣΥΣΧΕΤΊΣΕΙΣ ΚΑΙ ΔΙΑΦΟΡΈΣ ΜΕ ΠΑΡΑΔΟΣΙΑΚΆ ΣΥΣΤΉΜΑΤΑ ΔΙΑΧΕΊΡΙΣΗΣ ΒΆΣΕΩΝ ΔΕΔΟΜΈΝΩΝ (ΣΔΒΔ)

Στη συνέχεια θα δούμε πιο αναλυτικά το πώς πτυχές και λειτουργικότητες της φιλοσοφίας των παραδοσιακών Συστημάτων Διαχείρισης Βάσεων Δεδομένων έρχονται να επεκταθούν ή να παραλλαχθούν ώστε να καλύψουν τις ανάγκες της διαχείρισης των ροών δεδομένων.

1.4.1 ΣΤΑΤΙΚΌΤΗΤΑ ΤΩΝ ΕΠΕΡΩΤΉΣΕΩΝ

Καταρχήν, όπως αναφέρθηκε και προηγούμενα εν συντομία, οι επερωτήσεις που τίθενται πάνω σε μια ροή δεδομένων καταχωρούνται (register) στο σύστημα διαχείρισης και εκτελούνται από εκεί και πέρα συνεχόμενα πάνω στα δεδομένα. Τα δεδομένα εισέρχονται στο σύστημα διαχείρισης μία φορά το καθένα – η κάθε πλειάδα – χωρίς η εμφάνισή τους να μπορεί να επαναληφθεί και σε πραγματικό χρόνο το σύστημα πρέπει να έχει αποφασίσει πώς θα τα διαχειριστεί καθώς και να

ολοκληρώσει την επεξεργασία τους. Αυτό θα συνεχίσει να συμβαίνει έως ότου η κάθε επερώτηση αφαιρεθεί από το σύστημα (de-register) οπότε και σταματά να εκτελείται. Αντιθέτως [3], στα παραδοσιακά ΣΔΒΔ μια επερώτηση εκτελείται μόνο μια φορά και επιστρέφει τα αποτελέσματά της παύοντας έπειτα να υπάρχει. Θα εμφανιστεί ξανά μόνο αν σε επόμενη χρονική στιγμή το σύστημα ή κάποιος χρήστης του χρειαστεί να την εκτελέσει εκ νέου και θα εκτελεστεί ως μία νέα ανεξάρτητη επερώτηση. Ενδεχομένως το σύστημα να χρησιμοποιήσει την παλαιότερη εκτέλεση της επερώτησης ανάλογα με τις ρυθμίσεις που του έχουν γίνει για λόγους βελτιστοποίησης, όμως, η επερώτηση θα θεωρηθεί ως νέα και αυτόνομη.

Έτσι, λοιπόν, οι συνεχείς επερωτήσεις (continuous queries) εκτελούνται συνεχώς και μάλιστα πάνω στα δεδομένα που καταφθάνουν και όχι σε αποθηκευμένα δεδομένα [4]. Στο παραπάνω σενάριο του ελέγχου της κυκλοφοριακής κίνησης η μέση ωριαία ταχύτητα για παράδειγμα των αυτοκινήτων θα πρέπει να υπολογίζεται αδιάλειπτα από το σύστημα καθώς νέες τιμές θα παράγονται από τους αισθητήρες στις λεωφόρους αλλά και διότι είναι τέτοια η φύση της εφαρμογής που απαιτεί τα αποτελέσματα να έρχονται συνεχώς ή έστω με μικρές καθυστερήσεις. Παρατηρούμε, λοιπόν, ότι σε τέτοιες εφαρμογές η άμεση επεξεργασία των δεδομένων – το ποια και τι είδους θα είναι αυτή εξαρτάται από τη φύση της εφαρμογής – είναι επιτακτική, τόσο εξαιτίας της μαζικότητας των δεδομένων όσο και της ανάγκης για ταχύτατα αποτελέσματα. Για αυτό το λόγο, όπως θα συζητηθεί αργότερα κρίνεται συχνά αναγκαία η απόρριψη μερικών από τα δεδομένα για να εξοικονομηθούν πόροι και να διασφαλιστεί η διαθεσιμότητα του συστήματος.

1.4.2 Η ΈΝΝΟΙΑ ΤΟΥ ΧΡΌΝΟΥ

Στη συνέχεια, παρατηρεί κανείς πως στις παραδοσιακές εφαρμογές βάσεων δεδομένων η έννοια του χρόνου είναι είτε ανύπαρκτη είτε απλά περιορίζεται στο να αποτελεί και αυτή ένα ακόμη δεδομένο (ένα πεδίο, δηλαδή, ενός πίνακα). Έτσι, βλέπουμε ότι τα δεδομένα δεν έχουν εγγενή μέσα τους την έννοια του χρόνου. Ακόμη, όταν μια ποσότητα έχει κάποια τιμή και η τιμή της αυτή ανανεωθεί τότε μόνο η καινούρια είναι αυτή που ενδιαφέρει και μόνο σε εξαιρετικές περιπτώσεις χρησιμοποιείται το παρελθόν ενός μεγέθους πχ για στατιστικούς ή ιστορικούς λόγους. Είναι τέτοιες οι απαιτήσεις των εφαρμογών αυτών που ούτε εξαρτώνται ιδιαίτερα από τις τιμές των δεδομένων στη διάρκεια του χρόνου ούτε και είναι εφικτό από πλευράς κόστους όλες οι ιστορικές τιμές να καταγράφονται και να συντηρούνται.

Στις ροές δεδομένων, από την άλλη πλευρά, ο χρόνος αποκτά άλλη διάσταση και γίνεται αναπόσπαστη ιδιότητα των δεδομένων. Μια ροή αναπαριστά μια ακολουθία τιμών που

χαρακτηρίζουν μια οντότητα στην πάροδο του χρόνου. Μια οντότητα παρακολουθείται, δηλαδή, συνεχώς και οι τιμές που την χαρακτηρίζουν δημιουργούν πλειάδες για αυτήν και εισέρχονται στο σύστημα διαχείρισης ακολουθιακά και με απόλυτη χρονική σημασία. Με αυτό τον τρόπο [2] η ουσία της μελετώμενης οντότητας δε βρίσκεται σε κάποια τιμή της ή σε κάποιες τιμές της αλλά στη χρονική ακολουθία των τιμών που έχει λάβει και στα συμπεράσματα που μπορούν να συναχθούν από αυτήν. Οι επερωτήσεις πάνω στα συνεχή δεδομένα έχουν εξίσου "χρονικό" χαρακτήρα, μιας και εκτείνονται πάνω σε ένα ευρύ φάσμα της ιστορίας τους. Σε παραδείγματα σαν αυτό με την παρακολούθηση της κυκλοφορίας, κάτι τέτοιο εύκολα γίνεται αντιληπτό, καθώς η ανάγκη, για παράδειγμα, της καταγραφής της μέσης ταχύτητας των οχημάτων ανά δεκάλεπτο κατά της διάρκεια κάποιας ώρας αιχμής είναι προφανής.

Φυσικά, σκόπιμο θα ήταν να ξεκαθαριστεί σε αυτό το σημείο πως μια ροή δεδομένων, που αποτελεί, όπως έχει αναφερθεί μια ακολουθία, δεν είναι εξορισμού απαραίτητο να είναι μια χρονική ακολουθία. Είναι οπωσδήποτε δυνατόν οι τιμές της οντότητας που μελετούμε να εμφανίζονται σε συνάρτηση με κάποιο άλλο μέγεθος, όπως, για παράδειγμα, η θερμοκρασία κατά μήκος μιας μεγάλης απόστασης. Παρολαυτά η έννοια του χρόνου παραμένει ειδοποιός στις ροές δεδομένων γιατί είναι αυτή που αντικατοπτρίζει την αμεσότητα και την επιτακτικότητα της επεξεργασίας τους.

Έχει, επιπλέον διατυπωθεί [1] ότι η εξάρτηση αυτή των ροών δεδομένων από το χρόνο είναι κάτι απαραίτητο από την ίδια τους τη σημασιολογία. Για να επιτύχει σημασιολογική ορθότητα ένα Σύστημα Διαχείρισης Ροών έχει ανάγκη από τη χρονοσήμανση των δεδομένων (timestamping). Με αυτό τον τρόπο το σύστημα ποτέ δε θα επεξεργαστεί μια πλειάδα δεύτερη φορά. Η χρονοσήμανση επιτυγχάνεται με δύο τρόπους:

- Είτε οι πλειάδες χρονοσημαίνονται κατά την είσοδό τους στο ΣΔΡΔ από το ίδιο το σύστημα, οπότε και χαρακτηρίζονται από την ώρα του συστήματος και η χρονική διάκρισή τους μπορεί να επιτευχθεί εύκολα μιας και εκεί συνέβη.

- Είτε οι πλειάδες χρονοσημαίνονται στην πηγή στην οποία και παράγονται. Αυτές οι πλειάδες λαμβάνουν την χρονοσήμανση της πηγής και, αν οι πηγές είναι πολλαπλές, το ΣΔΡΔ οφείλει να διατηρεί κάποια πολιτική για να τις συγχρονίσει.

1.4.3 ΜΗ ΠΕΠΕΡΑΣΜΈΝΑ ΣΎΝΟΛΑ ΔΕΔΟΜΈΝΩΝ

Στα παραδοσιακά ΣΔΒΔ οι επερωτήσεις τίθενται πάνω σε δεδομένα αποθηκευμένα σε γνωστά αποθηκευτικά μέσα, είναι πεπερασμένου μήκους και τα αποτελέσματά τους έχουν νόημα απέναντι σε αυτά. Στις ροές δεδομένων τα δεδομένα δεν είναι πεπερασμένα [4]. Αυτό, βέβαια, δε σημαίνει ότι έχουν άπειρο μέγεθος αλλά ότι θεωρητικά ποτέ δε γνωρίζουμε το τέλος τους. Σε μια συνεχόμενη ακολουθία πλειάδων, η οποία εμπλουτίζεται ακατάπαυστα με νέες πλειάδες, είναι φυσικά αδύνατον να εκτελεστεί μια επερώτηση με καθολικό νόημα, μιας και τα δεδομένα πάνω στα οποία θα εκτελεστεί μεταβάλλονται με το χρόνο. Ακόμη και τα γνωστά κλειδώματα που ένα κλασσικό ΣΔΒΔ μπορεί να επιβάλλει πάνω στα δεδομένα για να διασφαλίσει την ακεραιότητα των επερωτήσεων είναι πλέον χωρίς αποτέλεσμα. Στις συνεχείς επερωτήσεις, επομένως, τα σύνολα δεδομένων αποτελούν ολοένα και διογκούμενες οντότητες, των οποίων ούτε το πέρας μπορούμε να εκτιμήσουμε με ασφάλεια ούτε και το ρυθμό αύξησης.

Έτσι, ένα ΣΔΡΔ οφείλει να αντιμετωπίσει τη ροή με πολύ διαφορετικό τρόπο [4]. Η αδυναμία αποθήκευσης και γνώσης του συνόλου της ροής, η αδυναμία χρήσης κλειδωμάτων πάνω στα δεδομένα, η επιτακτικότητα της επεξεργασίας και η ανάγκη για προσαρμοστικότητα του συστήματος στις εναλλαγές κάνουν τη σχεδίαση τέτοιων συστημάτων πολύ διαφορετική από ένα ΣΔΒΔ και δυσκολεύουν τη γενίκευσή τους.

Μια ακόμη παράμετρος που συνιστά σημαντική απόκλιση στο πως αντιλαμβανόμαστε τα δεδομένα μέσω των ροών σε σχέση με την κλασσική αντιμετώπιση [3] είναι ότι ούτε καν το ιστορικό τους δεν είναι να δυνατόν να είναι απόλυτα γνωστό. Σε μια μαζική και χωρίς υπολογίσιμο πέρας ροή είναι φύσει αδύνατον για ένα οποιοδήποτε σύστημα να μπορέσει να την αποθηκεύσει εξολοκλήρου και να την ανακτήσει μετέπειτα κατά βούληση. Συνεπώς, υπάρχει μια απροσδιοριστία σχετικά με τις παρελθούσες τιμές των μελετώμενων δεδομένων κάτι που δίνει στα ΣΔΡΔ μια αξιοσημείωτη ιδιαιτερότητα. Αν η φύση της εφαρμογής έχει ανάγκη από τη γνώση του παρελθόντος τότε πρέπει να είναι ξεκάθαρο το πως θα πραγματοποιηθεί αυτό, όσον αφορά την πυκνότητα των παλαιών τιμών που θα αποθηκευθούν - είτε μόνιμα είτε και προσωρινά. Η γνώση παλαιών τιμών είναι δυνατόν να είναι αναγκαία για διάφορους λόγους, όπως, για παράδειγμα, στον υπολογισμό της ανάδρασης του συστήματος συναρτήσει κάποιων συνθηκών ή στην εξαγωγή στατιστικών στοιχείων κα.

1.4.4 ΑΣΤΆΘΜΗΤΟΙ ΠΑΡΆΓΟΝΤΕΣ ΤΩΝ ΡΟΏΝ ΔΕΔΟΜΈΝΩΝ

Στις παραδοσιακές εφαρμογές βάσεων δεδομένων το σύστημα θεωρεί εν πολλοίς ότι διαχειρίζεται ακριβή και αξιόπιστα δεδομένα. Δεδομένα που του έχουν σταλεί από κάποιο λογισμικό και πάνω από κάποιο αξιόπιστο πρωτόκολλο - όπως πολύ συχνά το TCP/IP - και έχουν περάσει από αρκετούς ελέγχους για να διασφαλίσουν την ακεραιότητα και τη συνέπεια της βάσης. Ο τρόπος αντιμετώπισης καθώς και το ποσοστό ανοχής του ΣΔΒΔ σε δεδομένα που δεν ακολουθούν επακριβώς τον συμφωνημένο τύπο τους ή τον τρόπο εισαγωγής τους διαφέρει από σύστημα διαχείρισης σε σύστημα διαχείρισης.

Ένα Σύστημα Διαχείρισης Ροών Δεδομένων, από την άλλη πλευρά [3][9], πρέπει να είναι πιο προετοιμασμένο στο να υποδέχεται αναξιόπιστα δεδομένα. Η αναξιοπιστία αυτή μπορεί να προέλθει τόσο από την πιθανότητα να έχουν αλλοιωθεί κατά τη μεταφορά τους όσο και να καταφθάσουν με όχι απαραίτητα την αναμενόμενη χρονική τους σειρά εξαιτίας καθυστερήσεων του δικτύου ή, σε ένα δίκτυο αισθητήρων, για παράδειγμα, εξαιτίας της διακοπής λειτουργίας ορισμένων από τους αισθητήρες. Αν φέρουμε στο μυαλό μας ένα σύστημα παρακολούθησης καιρικών φαινομένων ή χρηματιστηριακών δεδομένων εύκολα κατανοούμε πόσο σημαντική είναι η συνέπεια στη χρονική διαδοχή των δεδομένων αλλά και η εξάρτηση από την αξιοπιστία των στοιχείων που έρχονται από τον κάθε αισθητήρα.

Στο πλαίσιο που προσδιορίσαμε ως "αστάθεια της ροής" πρέπει να σημειώσουμε ακόμη μία σημαντική πτυχή των ροών δεδομένων. Αυτή είναι ο ρυθμός με τον οποίο τα δεδομένα παράγονται και τελικά εισάγονται στο ΣΔΡΔ. Σε μια πραγματική εφαρμογή αυτός ο ρυθμός μπορεί να είναι πολύ λίγο υπολογίσιμος είτε από τα ίδια τα υπάρχοντα δεδομένα ή με στατιστικά στοιχεία ή, ακόμη, εντελώς άγνωστος [6]. Επιπλέον, πολλές φορές, η συγκυρία που χρήζει ιδιαίτερης προσοχής και προσαρμογής του συστήματος είναι εκείνη όπου τα δεδομένα εμφανίζονται σε απροσδόκητες ριπές (bursts), σε μεγάλους και απότομους, δηλαδή, ρυθμούς. Σε τέτοια περίπτωση το ΣΔΡΔ πρέπει να είναι έτσι σχεδιασμένο ώστε να μπορεί να αντιμετωπίσει την κατάσταση χωρίς να διακινδυνεύσει τη διαθεσιμότητά του. Η αστάθεια αυτή έχει παρατηρηθεί εκτεταμένα σε τομείς όπως η διαχείριση δικτύων [8] και μπορεί να οφείλεται τόσο σε παράγοντες που εξαρτώνται από τη φυσιολογία του μελετώμενου συστήματος - πχ υπερφορτωμένο δίκτυο σε συγκεκριμένες ώρες ή εξαιτίας συγκεκριμένων εφαρμογών - όσο και από εξωγενείς παράγοντες - πχ στο σύστημα παρακολούθησης της κυκλοφορίας οι αισθητήρες έχουν πολύ περισσότερα δεδομένα να αναφέρουν όταν πολλά αυτοκίνητα διέρχονται τη λεωφόρο στις ώρες αιχμής.

1.4.5 ΔΥΝΑΤΌΤΗΤΑ ΑΝΆΔΡΑΣΗΣ

Τα παραδοσιακά ΣΔΒΔ όσο πολύπλοκα και αν είναι στην υλοποίηση και τη βελτιστοποίησή τους έχουν κάνει εκ των προτέρων την υπόθεση ότι θα αντιμετωπίζουν τα δεδομένα τους με τον ίδιο τρόπο καθ' όλη τη διάρκεια της ζωής τους. Αυτό σημαίνει πώς ό,τι και αν αλλάξει στις συνθήκες του ίδιου του συστήματος ή των δεδομένων, το σύστημα θα συνεχίσει με απάθεια να εκτελεί τις συναλλαγές και τις επερωτήσεις που του τίθενται όπως θα έκανε και σε κάθε άλλη περίπτωση [10]. Αν, για παράδειγμα κάποια οι τιμές μιας οντότητας οι οποίες καταχωρούνται στη βάση δεδομένων αρχίσουν να εμφανίζονται με μεγάλο ρυθμό ή να λαμβάνουν συγκεκριμένες - έστω πολύ υψηλές - τιμές, τότε το ΣΔΒΔ δεν είναι κατασκευασμένο ώστε να αποφασίσει κάποια αλλαγή στρατηγικής. Κάτι τέτοιο δε συνιστά, σε καμία περίπτωση, ελάττωμα του ΣΔΒΔ αλλά χαρακτηριστικό των προδιαγραφών με τις οποίες έχει εξαρχής σχεδιαστεί.

Από την άλλη πλευρά σε ένα ΣΔΡΔ, αν, για παράδειγμα, μια ροή δεδομένων ξαφνικά αρχίσει να έρχεται με απροσδόκητα υψηλό ρυθμό τότε το σύστημα οφείλει να αντιδράσει με κάποιο τρόπο έτσι ώστε να μην διακυβευθεί η διαθεσιμότητά του, μιας και δεν είναι μία η ροή που παρακολουθεί. Στον αντίποδα αυτής της κατάστασης ένα ΣΔΡΔ είναι υποχρεωμένο να εκτελεί μια συνεχή επερώτηση ασταμάτητα ακόμη και όταν διαθέτει αραιά ή μη συγχρονισμένα δεδομένα. Γίνεται, λοιπόν, φανερό πως η ίδια η φύση των ροών, οι οποίες σε μεγάλο βαθμό απαιτούν την σε πραγματικό χρόνο (real time) επεξεργασία τους, προσδίδουν στα συστήματα που τις παρακολουθούν την αναγκαιότητα για ανάδραση. Την ικανότητα, δηλαδή, να αυτο-προσαρμόζονται κατά το δυνατόν καλύτερα στις εκάστοτε συνθήκες αντιδρώντας στις αλλαγές του περιβάλλοντός τους [12].

Ακόμη, λόγω της συχνής τους εφαρμογής σε συστήματα παρακολούθησης (monitoring), συχνά πρέπει να δέχονται παραμετροποίηση της συμπεριφοράς τους ανάλογα με τα αποτελέσματα των υπολογισμών τους. Χαρακτηριστικά, θα μπορούσαμε να αναφέρουμε μια χρηματιστηριακή εφαρμογή, η οποία ανάλογα με το συνδυασμό των τιμών κάποιων μετοχών και την πορεία τους μέσα σε κάποια δοσμένη χρονική περίοδο να μπορεί να αποστείλει μια ειδοποίηση σε έναν ενδιαφερόμενο ή να αναλάβει κάποια άλλου είδους δράση πχ αγοράς ή πώλησης. Παραδείγματα τέτοιων αναγκαιοτήτων ποικίλλουν σε εφαρμογές [4], όπως τα δίκτυα αισθητήρων, η ανάλυση και η διαχείριση δικτύων, οικονομικές εφαρμογές, ανάλυση και διαχείριση αρχείων καταγραφών (log files).

1.5 ΧΑΡΑΚΤΗΡΙΣΤΙΚΈΣ ΠΡΟΔΙΑΓΡΑΦΈΣ ΣΥΣΤΗΜΆΤΩΝ ΔΙΑΧΕΊΡΙΣΗΣ ΡΟΏΝ ΔΕΔΟΜΈΝΩΝ (ΣΔΡΔ)

Όπως έχουμε ήδη αναφέρει [8] οι εφαρμογές βάσεων δεδομένων και συστημάτων διαχείρισης όπου έχουν εντοπιστεί ανάγκες για μοντελοποίηση των δεδομένων ως ροές - κι επομένως ως τέτοιες πρέπει να αντιμετωπιστούν - υπάρχουν εδώ και αρκετό καιρό. Τις περισσότερες φορές αυτό που παλαιότερα είχε αναγνωριστεί ήταν ένα υποσύνολο των χαρακτηριστικών που επιτρέπουν να θεωρηθεί μια δομή πληροφορίας σαν ροή. Σήμερα, πια, κι ενώ η τεχνολογία έχει τη δυνατότητα να μας παράσχει τα μέσα για την εκμετάλλευση και την εφαρμογή των συνεχών ροών δεδομένων σε πραγματικά περιβάλλοντα, είναι εφικτή η κατά το δυνατόν συνολική θεώρηση των Συστημάτων Διαχείρισης Ροών Δεδομένων που να συμπεριλαμβάνει όλες τις προδιαγραφές που οι ανάγκες θέτουν και μάλιστα με δυνατότητες κλιμάκωσης [10].

Η τεχνολογία των αισθητήρων (sensors), για παράδειγμα, βρίσκεται σε τέτοιο επίπεδο αλλά και σε τόσο πυρετώδη εξέλιξη στις μέρες μας που σε λίγα μόνο χρόνια σχεδόν κάθε αντικείμενο θα έχει τη δυνατότητα να είναι εξοπλισμένο με μερικούς από αυτούς. Η νέα γενιά εφαρμογών που αναδύεται από αυτή μονάχα την προοπτική είναι τεράστια και ιδιαίτερα αναμενόμενη, ενώ υπάρχουν ήδη πολλές απόπειρες καταγεγραμμένες τόσο σε ερευνητικό αλλά και σε εφαρμοσμένο επίπεδο. Παράλληλα με την ερευνητική δραστηριότητα που αναπτύσσεται σε αυτό το χώρο πολύ σημαντικό είναι και το αίτημα για όσο το δυνατόν κοντινότερη στον πραγματικό χρόνο (real time) επεξεργασία που εγείρουν πολλά περιβάλλοντα πληροφοριακών συστημάτων. Αυτή η φιλοσοφία έρχεται σε μεγάλη αντίθεση με τα παραδοσιακά ΣΔΒΔ [13] όπου τα δεδομένα υφίστανται επεξεργασία ασύγχρονα και αφότου έχουν αποθηκευθεί σε σταθερό μέσο.

Οι προκλήσεις που φέρνουν τα ΣΔΡΔ απαιτούν θεμελιώδεις διαφοροποιήσεις όσον αφορά την αρχιτεκτονική και τη λειτουργικότητά τους. Μερικές από τις προδιαγραφές που τα κάνουν τόσο ξεχωριστά είναι συνοπτικά οι παρακάτω [10]:

- Ο τρόπος με τον οποίο τα δεδομένα μοντελοποιούνται οφείλει να εμπεριέχει την έννοια του χρόνου. Μάλιστα, όχι σαν μια παράμετρο αυτών αλλά σαν μια εγγενή τους ιδιότητα. Είναι δυνατόν, βέβαια, ανάλογα με τις ιδιαίτερες ανάγκες μιας εφαρμογής αυτό να μην ισχύει απόλυτα, αποτελεί, όμως, σημαντική προδιαγραφή που πηγάζει από το συνεχή και άμεσο χαρακτήρα των ροών. Έτσι, και ο μηχανισμός επερωτήσεων πάνω στα δεδομένα θα πρέπει να λαμβάνει υπόψη του το παραπάνω χαρακτηριστικό καθώς, φυσικά, και το ότι οι επερωτήσεις θα είναι και αυτές συνεχείς.

- Έχοντας εξηγήσει ότι από την ίδια τη φύση της μια ροή δεδομένων διαθέτει ένα βαθμό αβεβαιότητας τόσο λόγω φυσικών παραμέτρων (δίκτυο, πολλές πηγές δεδομένων κλπ) όσο και του απεριόριστου μεγέθους της. Το ΣΔΡΔ που θα τη διαχειρίζεται οφείλει, παρόλαυτα, να είναι σε θέση να απαντά στις επερωτήσεις με όσο καλύτερη ακρίβεια, γνωρίζοντας την αβεβαιότητα αυτή.

- Μια από τις επιπλοκές των παραπάνω είναι είναι ότι οι επερωτήσεις που εκτελούνται πάνω σε ροές πολύ δύσκολα μπορούν να δεχθούν καθολικούς τελεστές[1] (blocking operators) [14]. Τελεστές, δηλαδή, όπου όλη η ποσότητα των δεδομένων είναι αναγκαία ώστε να παραχθεί το επιθυμητό αποτέλεσμα. Η εκτέλεση, λοιπόν, παρόμοιων επερωτήσεων αποτελεί ιδιαίτερη πρόκληση για το σύστημα διαχείρισης ροών δεδομένων, μιας και οι καθολικοί τελεστές είναι εκ των πραγμάτων απαραίτητοι.

- Μια από τις ανάγκες που τίθενται από το γεγονός της "συνέχειας" των επερωτήσεων είναι ότι το σύστημα θα πρέπει να μπορεί να αντιμετωπίσει διακυμάνσεις των συνθηκών στις οποίες λειτουργεί. Κάτι τέτοιο σημαίνει ότι ο τρόπος εκτέλεσης των επερωτήσεων οφείλει να είναι κατά το δυνατόν προσαρμόσιμος σε αλλαγές όπως στο ρυθμό εμφάνισης των δεδομένων, ο οποίος μπορεί να είναι άλλοτε μικρός και άλλοτε πολύ υψηλός.

- Ακόμη, έχει γίνει κατανοητό πως η δυνατότητα αντίδρασης ενός σύγχρονου ΣΔΡΔ θα αποτελεί μια από τις απαραίτητες δυνατότητές τους. Το ζητούμενο εδώ είναι το σύστημα να μπορεί να παραμετροποιηθεί σε όσο το δυνατόν μεγαλύτερο βαθμό και να υποστηρίζει μεγάλο αριθμό αντιδράσεων (triggers) ανάλογα με τις συνθήκες που ανιχνεύει ή τα αποτελέσματα των υπολογισμών που το ίδιο κάνει πάνω στα δεδομένα του.

- Σε πολλές εφαρμογές ροών δεν είναι αρκετή μόνο η καλή διαχείριση των πόρων του συστήματος αλλά, ταυτόχρονα, η ικανότητά του να παρέχει και εγγυήσεις ποιότητας των υπηρεσιών του (quality of service).

- Τέλος, μπορούμε να παρατηρήσουμε πως ένα πολύ μεγάλος αριθμός εφαρμογών ροών δεδομένων έχουν εγγενώς κατανεμημένο χαρακτήρα στο περιβάλλον όπου μελετώνται. Χαρακτηριστικό το παράδειγμα της παρακολούθησης της κυκλοφορίας των λεωφόρων ή

1 Καθολικοί τελεστές (blocking operators) είναι εκείνοι οι τελεστές που απαιτούν το σύνολο των πλειάδων μιας σχέσης να είναι γνωστό για να παράγουν κάποιο αποτέλεσμα συνολικό. Πχ το άθροισμα όλων των πλειάδων μιας σχέσης R πάνω στο πεδίο a – sum(R.a). Η γνώση όλων των πλειάδων είναι εκ των πραγμάτων αδύνατη στις ροές.

της παρακολούθησης δασών για πυρκαγιές με δίκτυα αισθητήρων (sensor networks) κλπ. Αυτή η μεγάλη διασπορά δίνει καταρχάς σημασία στην αξιοποίηση του εύρους ζώνης (bandwidth) στην επεξεργασία των επερωτήσεων. Επιπλέον, οι αισθητήρες έχουν περιορισμένες δυνατότητες όσον αφορά το χρόνο ζωής της μπαταρίας τους, κάτι που απαιτεί ο ρυθμός ανάκρισής τους από το σύστημα να είναι προσεκτικά επιλεγμένος.

1.6 ΤΟ ΜΟΝΤΈΛΟ ΤΗΣ ΡΟΉΣ

1.6.1 ΠΕΡΙΓΡΑΦΉ

Έχουμε δει, λοιπόν, μέχρι τώρα τι είναι μια ροή δεδομένων και γιατί ένα Σύστημα Διαχείρισης Ροών Δεδομένων διαφέρει στην αντιμετώπισή του από την κλασσική αναπαράσταση και αρχιτεκτονική ενός παραδοσιακού Συστήματος Διαχείρισης Βάσεων Δεδομένων. Έχει γίνει ξεκάθαρο, πως οι ιδιαιτερότητες μιας ροής ταυτόχρονα με τη μεγάλη ανάγκη για εφαρμογές που τις αξιοποιούν στις μέρες μας, έχουν δώσει το έναυσμα για έντονη ερευνητική δραστηριότητα στο χώρο αυτό. Ακόμη, είδαμε ορισμένες από αυτές τις ιδιαιτερότητες των ροών και τα χαρακτηριστικά των ΣΔΡΔ που συνάγονται από αυτές.

Μπορούμε να συνοψίσουμε την περιγραφή του μοντέλου της ροής δεδομένων σε μερικές προτάσεις λέγοντας τα εξής [14][15][41]:

- Στο μοντέλο των ροών δεδομένων τα δεδομένα είναι "συνεχή", καταφθάνουν, δηλαδή, χωρίς διακοπές - αν και ίσως με αυθαίρετους/απροσδιόριστους ρυθμούς. Η ειδοποιός διαφορά σε σχέση με τις παραδοσιακές βάσεις δεδομένων, όπου και εκεί τα δεδομένα θεωρητικά δεν τερματίζουν, είναι ότι στα τελευταία οι πλειάδες καταφθάνουν σε αραιά χρονικά διαστήματα, κάτι που δίνει στο σύστημα περισσότερο χρόνο να τα διαχειριστεί. Αντίθετα σε ένα ΣΔΡΔ ο όγκος των δεδομένων, οι ρυθμοί άφιξής τους και η ανάγκη για άμεση επεξεργασία καθιστά μια τέτοια προσέγγιση ανώφελη.

- Ποτέ δεν είναι εξολοκλήρου διαθέσιμα στο σύστημα για υπολογισμούς, μιας και είναι απεριόριστα στο πλήθος. Άμεσα διαθέσιμο είναι μονάχα το υποσύνολο της ροής που βρίσκεται στην ενδιάμεση μνήμη του συστήματος (buffers) και παραμένει εκεί για περιορισμένο χρονικό διάστημα και το τμήμα της εκείνο που για λόγους διαχείρισης αποθηκεύεται προσωρινά στο δίσκο.

- Τα δεδομένα δεν είναι διαθέσιμα για τυχαία προσπέλαση εφόσον το μέγεθός τους είναι απροσδιόριστο και η συνολική τους αποθήκευση αδύνατη. Επομένως, η κύρια επεξεργασία τους λαμβάνει χώρα σε πραγματικό χρόνο (real time), καθώς αυτά καταφθάνουν. Αυτό δεν είναι απλά χαρακτηριστικό των ίδιων των ροών αλλά και λειτουργική απαίτηση των περιβαλλόντων στα οποία βρίσκουν εφαρμογή. Είναι, δηλαδή, ανάγκη των περισσοτέρων από αυτά η άμεση επεξεργασία των ροών και η παραγωγή αποτελεσμάτων.

- Το σύστημα δεν έχει έλεγχο πάνω στο ρυθμό, τη σειρά εμφάνισης ή την αξιοπιστία της/των πηγής/πηγών των δεδομένων, είτε μιλούμε για μια είτε για πολλές λογικά αλληλοσυνδεόμενες ροές. Έτσι οι ροές μπορεί να καταφθάνουν εντελώς ασυντόνιστα, με αυθαίρετους κι ίσως επικίνδυνα μεγάλους ρυθμούς και δίχως εγγυήσεις για τη σειρά εμφάνισης των πλειάδων τους[2]. Αυτό προσδίδει έναν επιπλέον φόρτο σε ένα σύστημα επεξεργασίας ροών δεδομένων

- Εξαιτίας του απεριόριστου μεγέθους τους κάθε πλειάδα αφότου εισέλθει στο σύστημα, στη συνέχεια είτε αποβάλλεται είτε αρχειοθετείται. Αν υπάρχει ανάγκη για μετέπειτα ανάκτησή τους ή ο ρυθμός τους αυξηθεί επικίνδυνα τότε τα ζητήματα διαχείρισης των πόρων του συστήματος γίνονται σοβαρά, μιας και η μνήμη του είναι πρακτικά ελάχιστη σε σύγκριση με το μέγεθος της ροής. Το τελευταίο είναι μια πρακτική άποψη για τις ροές που αναδεικνύει με ανάγλυφο τρόπο την αδυναμία του ΣΔΡΔ να διαχειριστεί με απλό τρόπο τις ροές.

1.6.2 ΧΡΟΝΟΣΗΜΑΝΣΗ (TIMESTAMPING)

Ένας από τους πιο κοινούς τρόπους αναπαράστασης μιας ροής δεδομένων είναι αυτός μιας κλασσικής σχεσιακής πλειάδας, όπως τη γνωρίζουμε από τα παραδοσιακά ΣΔΒΔ. Μια ακολουθία, δηλαδή, από πεδία συγκεκριμένου τύπου το κάθε ένα. Μια συχνά σχεδόν αναπόφευκτη προσθήκη στην πλειάδα αυτή είναι η χρονοσήμανσή της (timestamping), η προσθήκη, δηλαδή, ενός χρονικού πεδίου, ή η προσθήκη ενός αύξοντα αριθμού ως διακριτικό, ώστε η πλειάδα να είναι μονοσήμαντα αναγνωρίσιμη. Ειδικά η χρονοσήμανση αποτελεί για πολλούς [16] τόσο αναπόσπαστο τμήμα της

2 Ακόμη κι αν μια πηγή ροών εκπέμπει πλειάδες ροής διατεταγμένες ως προς κάποιο πεδίο τους είναι πολύ πιθανό ένα σύστημα επεξεργασίας να δέχεται τέτοιες πλειάδες από περισσότερες από μία πηγές, οι οποίες είναι αδύνατο να συντονιστούν μεταξύ τους. Για παράδειγμα σε ένα δίκτυο αισθητήρων θα υπάρχουν πολλοί αισθητήρες που θα αναφέρουν μετρήσεις θερμοκρασίας. Κι αν ακόμη κάποιοι γειτονικοί αισθητήρες συντονίζονται μεταξύ τους,το κεντρικό σύστημα που θα λαμβάνει το σύνολο των μετρήσεων δε θα έχει καμία εγγύηση για τη διαδοχή των πλειάδων που δέχεται.

ροής που να ορίζουν τη ροή ως μια ακολουθία ζευγών πλειάδας και χρονόσημου (timestamp).

Μπορεί να διακρίνει κανείς δύο είδη χρονοσήμανσης [14]:

- **Έμμεση χρονοσήμανση** (implicit timestamping): Προσαρτάται σε μια εισερχόμενη στο σύστημα πλειάδα όταν αυτή εμφανίζεται στο ίδιο το σύστημα. Ένα τέτοιο χρονόσημο θα μπορούσε πολύ απλά να είναι κάποιο διακριτικό, όπως ένας αύξων αριθμός, με σκοπό τόσο τη δυνατότητα πλήρους διάκρισης ανάμεσα στις πλειάδες όσο και τη χρονική διάκρισή τους - σειρά εμφάνισης.

- **Ρητή χρονοσήμανση** (explicit timestamping): Αυτού του είδους η χρονοσήμανση φέρεται από τα ίδια τα δεδομένα. Η κάθε πλειάδα, δηλαδή, διαθέτει σαν παράμετρό της ένα χρονικό διακριτικό, που σε πραγματικές εφαρμογές συνήθως είναι ο ίδιος ο χρόνος. Ο χρόνος αυτός, φυσικά, μπορεί να είναι είτε ο απόλυτος χρόνος (χρόνος, μήνας, μέρα, ώρα, λεπτό δευτερόλεπτο κοκ) είτε σχετικός (πχ από την εκκίνηση του συστήματος).

Ένα όχι τόσο ευδιάκριτο ζήτημα που γεννάται από τη χρήση των χρονοσήμων στις ροές είναι το εξής: Αν δύο ροές, εφόσον τις εννοούμε ως ακολουθίες σχεσιακών πλειάδων, ενωθούν σε κάποιο πεδίο τους (join) τότε οι παραγόμενες πλειάδες τι χρονόσημο θα πρέπει να διαθέτουν; Στην περίπτωση της έμμεσης χρονοσήμανσης μια προφανής λύση είναι οι παραγόμενες πλειάδες να λαμβάνουν ως χρονόσημο τη στιγμή που γίνεται η ένωση (join), ενώ μια λύση στην περίπτωση της ρητής είναι ο διαχειριστής του συστήματος να έχει προσδιορίσει τον τρόπο υπολογισμού παρόμοιων τιμών μέσω αρχικών ρυθμίσεων.

1.6.3 ΠΛΑΊΣΙΑ (WINDOWS)

Γνωρίζοντας πια καλά πως το μέγεθος της ροής καθεαυτής είναι πρακτικά απεριόριστο τα ΣΔΡΔ όταν εκτελούν επερωτήσεις πάνω σε αυτές συνήθως ενδιαφέρονται για ένα υποσύνολο ή ένα τμήμα τους. Έτσι, το εύρος της ροής πάνω στο οποίο θα εκτελεστεί μια επερώτηση πρέπει να είναι προσδιορίσιμο και πεπερασμένο. Για αυτό το σκοπό έχει καθιερωθεί η χρήση πλαισίων (windows). Τα πλαίσια αυτά εμφανίζονται ως επέκταση της SQL, της γνωστής γλώσσας περιγραφής επερωτήσεων σε σχεσιακές βάσεις δεδομένων και καθορίζουν μια συγκεκριμένη συνεχή περιοχή πάνω στη ροή, όπου και θα εκτελεστεί η επερώτηση [10].

Αν φέρουμε στο μυαλό μας το παράδειγμα με το σύστημα διαχείρισης της κυκλοφορίας θα

μπορούσαμε να θεωρήσουμε ότι μια ροή που να αποτελείται από πλειάδες με πεδία όπως ο αριθμός αυτοκινήτου, η ταχύτητά του, το αναγνωριστικό του αυτοκινητοδρόμου, η κατεύθυνσή του και η χρονική στιγμή της καταγραφής της πλειάδας. Μια εύλογη επερώτηση θα ήταν να υπολογιστεί η μέση ταχύτητα των αυτοκινήτων στα 15 τελευταία λεπτά. Αυτά τα 15 λεπτά που δίνουν το άνω και το κάτω χρονικό όριο του υπολογισμού αποτελούν το - χρονικό - "πλαίσιο" της συγκεκριμένης επερώτησης.

Ο λόγος χρήσης ενός παρόμοιου πλαισίου μπορεί να είναι τόσο ο περιορισμός του μεγέθους των δεδομένων της ροής, ώστε να είναι οι υπολογισμού εφικτοί σε - σχεδόν - πραγματικό χρόνο, όσο και η ανάγκη επεξεργασίας αυτών και μόνο των δεδομένων όταν αυτό επιτάσσει η σημασιολογία της επερώτησης. Επιπλέον, μία επερώτηση που κάνει χρήση πλαισίων είναι δυνατόν να εκτελείται με συνεχόμενο τρόπο δημιουργώντας με αυτό τον τρόπο την έννοια των ολισθαινόντων πλαισίων. Στο παράδειγμα με την παραπάνω επερώτηση στο σύστημα παρακολούθησης της κυκλοφορίας, η γνώση της μέσης ταχύτητας των αυτοκινήτων τα τελευταία 15 λεπτά δεν είναι χρήσιμη μόνο μια φορά αλλά σε διάρκεια. Επομένως κάθε τόσο, καθώς ο χρόνος περνά, το πλαίσιο των 15 λεπτών αναφέρεται σε μια διάρκεια που παραμένει σταθερή σε μήκος αλλά ολισθαίνει χρονικά προς της θετική φορά του χρόνου. Φυσικά, το πλαίσιο θα μπορούσε να μην προσδιοριστεί χρονικά αλλά και με κάποιο άλλο τρόπο, όπως, για παράδειγμα, με το πλήθος των πλειάδων που επιθυμούμε να λάβουν μέρος στην επεξεργασία της επερώτησης. Ακόμη, ένα πλαίσιο θα μπορούσε να έχει σταθερό το κάτω άκρο του και ανοικτό το άνω. Έτσι θα μπορούσε να υπολογιστεί, για παράδειγμα, η μέση ταχύτητα των οχημάτων από ένα συμβάν και έπειτα. Τότε μιλούμε για πλαίσια με ορόσημο (landmark), μιας και η ύπαρξή τους προσδιορίζεται από τη χρονική στιγμή του συμβάντος όπου ξεκινούν.

1.7 ΠΑΡΆΔΕΙΓΜΑ ΡΟΉΣ ΔΕΔΟΜΈΝΩΝ

Ας δούμε στη συνέχεια ένα παράδειγμα ροής δεδομένων που έρχεται από το χώρο της διαχείρισης δικτύων, όπου, όπως έχει ήδη αναφερθεί, οι ροές μπορούν να βρουν συχνά εφαρμογή [14][8]. Ας φανταστούμε ένα μεγάλο τηλεπικοινωνιακό οργανισμό που παρέχει υπηρεσίες διαδικτύου (ISP) που θέλει να παρακολουθεί την κυκλοφορία στο δίκτυό του. Το σύστημα που θα επιτελεί αυτή τη λειτουργία θα παρακολουθεί ένα μεγάλο αριθμό ροών δεδομένων των οποίων ο ρυθμός μπορεί να φτάσει σε πολύ υψηλά επίπεδα. Τα περισσότερα εργαλεία διαχείρισης κυκλοφορίας δικτύων μέχρι σήμερα είτε βασίζονται στη λογική του να εξυπηρετούν συγκεκριμένους σκοπούς, για τους οποίους και έχουν βελτιστοποιηθεί, είτε απλώς αποθηκεύουν τα δεδομένα τους σε αρχεία ή ειδικές βάσεις δεδομένων για να μπορέσουν αργότερα να τα επεξεργαστούν σε νεκρό χρόνο (off line). Είναι

φανερό πως ούτε η άμεση διαχείριση του δικτύου είναι εφικτή με αυτό τον τρόπο ούτε και η επεκτασιμότητα του συστήματος είναι σημαντική, μιας και τα αποτελέσματα παράγονται ασύγχρονα και το σύστημα επιτελεί συγκεκριμένες λειτουργίες στις οποίες είναι αφοσιωμένο.

Ένα Σύστημα Διαχείρισης Ροών Δεδομένων θα μπορούσε να αλλάξει ριζικά αυτή την εικόνα παρέχοντας σύγχρονη (on line) επεξεργασία συνεχών επερωτήσεων πάνω στις ροές των δεδομένων του δικτύου, ενώ οι διαχειριστές του θα μπορούσαν ανά πάσα στιγμή να θέσουν επί των ροών νέες συνεχείς επερωτήσεις ανάλογα με τις ανάγκες που θα εμφανιστούν. Για να γίνει περισσότερο κατανοητό το περιγραφόμενο σύστημα ας υποθέσουμε ότι το δίκτυο του οργανισμού διαθέτει πελάτες οι οποίοι συνδέονται με αυτό, για παράδειγμα μέσω τηλεφωνικών συνδέσεων (dial up) και κόμβους (routers) που, συνδεόμενοι μεταξύ τους απαρτίζουν το δίκτυο-κορμό (backbone) του οργανισμού. Πακέτα δεδομένων πηγαινοέρχονται προς κάθε κατεύθυνση ανάμεσα στους πελάτες και τους κόμβους του δικτύου.

Τα πακέτα που κυκλοφορούν στο δίκτυο θα μπορούσαν να περιγραφούν από το σύστημα διαχείρισης ως πλειάδες με τα παρακάτω πεδία:

- **src**: Η διεύθυνση από την οποία ένα πακέτο προέρχεται - η διεύθυνση της πηγής του πακέτου - που μπορεί να είναι η IP της πηγής.

- **dest**: Η διεύθυνση στην οποία ένα πακέτο προορίζεται - η διεύθυνση προορισμού του πακέτου - που μπορεί να είναι η IP του προορισμού.

- **id**: Κωδικός αριθμός που τίθεται στο πακέτο από τον αποστολέα έτσι ώστε ο παραλήπτης του πακέτου να μπορεί να το διακρίνει από τα υπόλοιπα.

- **len**: Το μήκος του πακέτου - πχ σε bytes, αν αυτό είναι επιθυμητό από τις ανάγκες.

- **time**: Η χρονική στιγμή στην οποία το πακέτο καταγράφηκε από το σύστημα παρακολούθησης του δικτύου.

Ο οργανισμός είναι πιθανόν να επιθυμεί να γνωρίζει το φόρτο στη σύνδεση ενός πελάτη του που είναι συνδεδεμένος στο δίκτυο κατά μέσο όρο ανά λεπτό της ώρας και να αποστέλλει μια ειδοποίηση στο διαχειριστή αν ο φόρτος αυτός ξεπεράσει κάποια συγκεκριμένη τιμή. Χρησιμοποιώντας αυτόματες ενεργοποιήσεις (triggers) ένα κλασσικό ΣΔΒΔ θα μπορούσε πιθανώς

να αποφέρει παρόμοια αποτελέσματα. Η επεξεργαστική ισχύς που θα χρειαζόταν το σύστημα όμως σε τέτοια περίπτωση, όπου τα δεδομένα πρώτα καταχωρούνται στη βάση κι έπειτα υφίστανται επεξεργασία, ενώ ταυτόχρονα συνεχίζουν να έρχονται με απρόβλεπτο ρυθμό, θα ήταν τεράστια και θα εξυπηρετούσε αυτή την επερώτηση και μόνο. Σε μια συγκυρία όπου ο ρυθμός μπορεί να αυξηθεί απότομα αλλά και όπου τα δεδομένα δεν είναι χρήσιμο να αποθηκεύονται εξολοκλήρου σε σταθερό μέσο χρειάζονται διαφορετικές τεχνικές αντιμετώπισης. Ενδεικτικά, το σύστημα παρακολούθησης θα μπορούσε, αντί να επιχειρεί να εκτελέσει την επερώτηση αδιαφορώντας για τη μεγάλη αύξηση του ρυθμού, να προσπαθήσει να υπολογίσει απλώς μια προσέγγισή της με τρόπο που να μην διακινδυνεύσει τη διαθεσιμότητά του. Άλλωστε θα πρέπει την ίδια ώρα να παρακολουθεί και άλλες ροές αλλά και να εκτελεί και άλλες επερωτήσεις.

Για να ανταποκριθεί στην παραπάνω επερώτηση το σύστημα θα έπρεπε να προβαίνει με συνεχόμενο και αδιάλειπτο τρόπο στις παρακάτω διαδικασίες:

i. Να παρακολουθεί όλες τις ροές πακέτων από όλους του τους πελάτες,

ii. Να διατηρεί το άθροισμα των μηκών των πακέτων που αντιστοιχούν στη ροή του κάθε πελάτη, ομαδοποιώντας τα αθροίσματα τόσο ανά πελάτη όσο και ανά λεπτό της ώρας και

iii. Να συγκρίνει τον υπολογιζόμενο φόρτο με το όριο πάνω από το οποίο να αποστέλλει ειδοποίηση στο διαχειριστή του δικτύου. Φυσικά, για να το επιτύχει αυτό θα πρέπει να υποστηρίζει και κάποια αλληλεπίδραση με το διαχειριστή, ο οποίος θα μπορεί να θέτει παρόμοιες επερωτήσεις στο σύστημα, καθώς και να τα αφαιρεί από αυτό κατά βούληση.

Μια άλλη αναγκαιότητα του οργανισμού θα μπορούσε να είναι η απομόνωση όλων των συνδέσεων - και λέγοντας "σύνδεση" εννοούμε τη ροή πακέτων από ένα συγκεκριμένο αποστολέα προς ένα συγκεκριμένο παραλήπτη - στο δίκτυο κορμό (backbone) του οργανισμού και ο προσδιορισμός του μεγέθους της κυκλοφορίας που προκαλείται από αυτήν. Και σε αυτή την περίπτωση μια συνεχής επερώτηση θα μπορούσε να κάνει τα εξής:

i. Να παρακολουθεί τις ροές πακέτων ανάμεσα στους κόμβους του δικτύου κορμού,

ii. Να ομαδοποιεί χρονικά τα πακέτα που έχουν κοινό ζευγάρι στα πεδία αποστολέα και παραλήπτη και

iii. Να υπολογίζει το άθροισμα των μηκών των πακέτων αυτών. Στη συγκεκριμένη επερώτηση η ομαδοποίηση που ζητείται δεν είναι απλή υπόθεση μιας και οι πράξεις ομαδοποίησης σε μια κλασσική αναπαράσταση δεδομένων απαιτούν τη γνώση του συνόλου των δεδομένων.

Ένα τελευταίο παράδειγμα στο παραπάνω σύστημα είναι ο υπολογισμός του ποσοστού της κυκλοφορίας του δικτύου που οφείλεται σε κάποιο συγκεκριμένο πελάτη του οργανισμού. Μια τέτοια επερώτηση ίσως να μην είναι αναγκαία σε σταθερή βάση αλλά μπορεί να αποδειχθεί χρήσιμη σε περιπτώσεις συμφορήσεων του δικτύου και να τεθεί για αυτό το σκοπό (ad hoc) στο σύστημα ώστε να ανιχνευθεί ποιοι είναι υπεύθυνοι για τη συμφόρηση. Για να επιτύχει κάτι τέτοιο το σύστημα θα πρέπει:

i. Να παρακολουθεί τις ροές ανάμεσα στους πελάτες του οργανισμού και τους κόμβους του δικτύου κορμού,

ii. Να ομαδοποιεί τις συνδέσεις των πελατών ανάλογα με τους κόμβους στους οποίους συνδέονται κάθε φορά,

iii. Να συγκρατεί το πλήθος των πακέτων που κυκλοφορούν για κάθε σύνδεση και

iv. Να διαιρεί το παραπάνω πλήθος με το σύνολο των πακέτων του δικτύου κορμού.

Στις παραπάνω παραγράφους, ουσιαστικά, δόθηκε σε μικρογραφία ένα Σύστημα Διαχείρισης Ροών Δεδομένων για τους συγκεκριμένους σκοπούς της διαχείρισης του δικτύου του υποθετικού τηλεπικοινωνιακού οργανισμού. Είναι, όμως, φανερές οι ιδιαιτερότητες που παρουσιάζουν οι ροές και σχετίζονται με τον όγκο τους, το βαθμό αβεβαιότητας που παρουσιάζουν, τις ανάγκες για πόρους (υπολογιστική ισχύ και μνήμη) του συστήματος που τις αναλαμβάνει και τις επερωτήσεις που θέτουμε πάνω σε αυτές. Σαν να μην έφτανε η εγγενής δυσκολία στην παρακολούθηση των ροών η πολυπλοκότητα των επερωτήσεων έρχεται να συμπληρώσει το τοπίο. Στο επόμενο κεφάλαιο θα μελετηθεί το θέμα των συνεχών επερωτήσεων ειδικότερα.

2. ΣΥΝΕΧΕΊΣ ΕΠΕΡΩΤΉΣΕΙΣ (CONTINUOUS QUERIES)

Όπως προαναφέρθηκε στο προηγούμενο κεφάλαιο, χωρίς πολλές λεπτομέρειες, η εκτέλεση επερωτήσεων πάνω σε ροές δεδομένων επανεξετάζει σχεδόν όλες τις πτυχές της εκτέλεσης επερωτήσεων σε σχέση με τις παραδοσιακές βάσεις δεδομένων. Η κύρια αιτία της διαφοροποίησης τους είναι η συνεχής και φευγαλέα φύση των δεδομένων των πρώτων - με ό,τι αυτό συνεπάγεται - σε αντίθεση με τη στατική φύση των τελευταίων.

2.1 ΕΙΣΑΓΩΓΉ ΣΤΙΣ ΣΥΝΕΧΕΊΣ ΕΠΕΡΩΤΉΣΕΙΣ

Όπως έχει αναφερθεί στην περιγραφή του μοντέλου των δεδομένων των ροών η κυρίαρχη οντότητα είναι η πλειάδα (tuple), έννοια όχι τυχαία δανεισμένη από το σχεσιακό μοντέλο. Τα δεδομένα περιγράφονται σαν γραμμές ενός πίνακα με πεδία συγκεκριμένου τύπου, που αποτελούν τις στήλες του. Η επιλογή αυτή είναι σχεδόν φυσική, μιας και το σχεσιακό μοντέλο κυριαρχεί στις εφαρμογές βάσεων δεδομένων που έχουν κατασκευαστεί μέχρι σήμερα και το υιοθετούν τα μεγαλύτερα συστήματα διαχείρισης βάσεων δεδομένων.

Από τη μία πλευρά, λοιπόν, το συγκεκριμένο μοντέλο δεδομένων είναι κυρίαρχο και, από την άλλη, ο τρόπος με τον οποίο αναπαριστά τις οντότητες είναι ιδιαίτερα κατανοητός από τον άνθρωπο, μιας και κάθε πλειάδα αποτελείται από τις τιμές των πεδίων της. Ταυτόχρονα, έχει υπάρξει τεράστια μελέτη στην επιστημονική κοινότητα τις τελευταίες δεκαετίες στο χώρο αυτό έτσι ώστε οι λειτουργίες που γίνονται βασισμένες σε αυτό το μοντέλο - καταχωρήσεις δεδομένων, ανάκτηση δεδομένων - να γίνονται με όσο το δυνατόν περισσότερο βελτιστοποιημένο τρόπο. Επιπλέον, οι σχεσιακές έννοιες είναι αναμφισβήτητα οικείες από το μεγάλο εύρος έρευνας και εφαρμογής που έχουν τύχει και η οικοδόμηση νέων θεωρήσεων πάνω σε αυτές αποδεικνύεται με αυτό τον τρόπο πιο κλιμακώσιμη.

Έτσι, και στις ροές δεδομένων η έρευνα κινείται προς αυτή την κατεύθυνση. Στον τρόπο, δηλαδή, με τον οποίο ένα σύστημα διαχείρισης ροών δεδομένων θα μπορεί να εκτελεί με αποδοτικό τρόπο λειτουργίες πάνω σε δεδομένα με σχεσιακή φύση τα οποία, όμως, εμφανίζονται με μορφή ροών. Αυτή η θεώρηση ανοίγει μια νέα γενιά προβλημάτων [17]. Παρόλο που στη γενική τους μορφή οι λειτουργίες που εκτελούνται πάνω στις ροές είναι αρκετά γνωστές, ως πράξεις της σχεσιακής άλγεβρας, η υλοποίησή τους διαφέρει πολύ από ότι στα παραδοσιακά ΣΔΒΔ. Ο κύριος λόγος είναι ότι η επεξεργασία μεταφέρεται από δεδομένα που βρίσκονται σε σκληρούς δίσκους σταθερά αποθηκευμένα περνούμε σε δεδομένα που ο μόνος τους περιορισμός ως προς το μέγεθος είναι η

ικανότητα του δικτύου στο οποίο ταξιδεύουν - το εύρος ζώνης (bandwidth). Δύο, λοιπόν από τις πιο στοιχειώδεις υποθέσεις που γνωρίζαμε ως τώρα καταρρίπτονται:

2. Τα δεδομένα δεν βρίσκονται σταθερά αποθηκευμένα σε σκληρούς δίσκους, ώστε να είναι στη διάθεση του συστήματος ανά πάσα στιγμή, άσχετα με το κόστος που η ανάκτησή τους επιφέρει.

1. Μια ροή έχει πρακτικά απεριόριστο μήκος και μόνο το τμήμα της το οποίο υφίσταται επεξεργασία είναι διαθέσιμο. Έτσι δεν είναι δυνατόν να υπάρξει ποτέ πλήρης γνώση, έστω και στιγμιαία, της οποιασδήποτε ροής.

2.2 ΠΑΡΆΔΕΙΓΜΑ ΑΠΌ ΤΟ ΧΏΡΟ ΤΗΣ ΟΙΚΟΝΟΜΊΑΣ

Θα χρησιμοποιήσουμε ένα ακόμη πραγματικό σενάριο που έρχεται από το χώρο των οικονομικών (financial) δεδομένων και έχει παρατηρηθεί στο [18], ώστε να καταδείξουμε την ουσία και την έννοια του τι σημαίνει μια συνεχής επερώτηση.

Τα οικονομικά δεδομένα στα οποία αναφερόμαστε είναι δεδομένα που προέρχονται από μια αγορά όπως αυτή του χρηματιστηρίου και περιλαμβάνουν πράξεις όπως αγοραπωλησίες μετοχών. Παρόμοιες αγορές είναι γνωστό ότι παράγουν τεράστιες ποσότητες δεδομένων και μάλιστα συχνά σε απότομους ρυθμούς που εξαρτώνται από παράγοντες καθαρά εξωγενείς από το οποιοδήποτε σύστημα τις παρακολουθεί. Αυτά τα δεδομένα αν υποστούν την κατάλληλη επεξεργασία είναι δυνατόν να αποφέρουν πολύ μεγάλα κέρδη. Βέβαια, πρόκειται για μια επεξεργασία που εξαρτάται σημαντικά από τη σημασιολογία των τιμών σε συνδυασμό με σωρεία άλλων παραγόντων και συγκυριών, είναι, όμως, αδύνατο να γίνει σε μεγάλη κλίμακα από ανθρώπους. Είναι, επομένως, κατανοητό πως συστήματα διαχείρισης ροών θα έβρισκαν σπουδαία εφαρμογή εκεί.

Η πιο απλή εφαρμογή που μπορεί να φανταστεί κανείς είναι η παρακολούθηση των τιμών επιλεγμένων μετοχών και η ειδοποίηση των ενδιαφερομένων όταν ικανοποιείται κάποια δοσμένη συνθήκη. Κάτι τέτοιο θα μπορούσε να υλοποιηθεί με μια επερώτηση - με την κλασσική έννοια της επερώτησης στις βάσεις δεδομένων -, η οποία με συνεχή τρόπο, καθώς τα στοιχεία από το χρηματιστήριο καταφθάνουν, θα επιλέγει τις μετοχές ενδιαφέροντος και θα τις συγκρίνει με τις τιμές-όρια που έχουν τεθεί. Είναι ξεκάθαρο πως, όσο απλή και αν είναι αυτή η εφαρμογή στη σύλληψή της, τα προβλήματα που γεννώνται είναι πολλά. Όλα, όμως, συμπυκνώνονται στο μέγεθος της ροής, το ρυθμό, δηλαδή, με τον οποίο η ροή γίνεται διαθέσιμη στο σύστημα

επεξεργασίας. Ας αναλογιστεί μόνο κανείς το φόρτο, όταν χιλιάδες τέτοιες επερωτήσεις από τους πολλούς χρήστες του συστήματος εκτελούνται σε συνεχή βάση, ταυτόχρονα και με το ρυθμό άφιξης είτε πολύ μεγάλο είτε ελάχιστα γνωστό και, μάλιστα, πρέπει να απαντηθούν άμεσα, γιατί, διαφορετικά, στο πλαίσιο μιας οικονομικής εφαρμογής δεν θα έχουν ουσιαστική αξία.

Σε ένα τέτοιο πλαίσιο διακρίνονται δύο ειδών τέτοιες επερωτήσεις. Αυτές που έχουν πρόσκαιρο χαρακτήρα και αυτές που έχουν μακροπρόθεσμο χαρακτήρα. Στην πρώτη κατηγορία οι ανάγκες που ικανοποιούνται είναι:

Η εύρεση μικρών χρονικών διαστημάτων όπου οι τιμές πώλησης και ζήτησης αυξομειώνονται και ενδιάμεσα υπάρχει συμφέρον για αγοραπωλησία.

Η ανίχνευση μετοχών που η αύξηση ή η μείωση των τιμών τους επηρεάζεται από κοινές ή παραπλήσιες συνθήκες.

Η ανίχνευση ογκωδών αγοραπωλησιών που, όμως, επιχειρείται να γίνονται τμηματικά σε μια προσπάθεια να μην τραβήξουν την προσοχή της υπόλοιπης αγοράς.

Στη δεύτερη κατηγορία - μακροπρόθεσμες - μερικές από τις ανάγκες που ικανοποιούνται είναι:

2. Η ανίχνευση των ιδιαίτερων τάσεων των τιμών κάποιων μετοχών - ή μετοχών κλάδων επιχειρήσεων κλπ -, συνδυάζοντας τις παρελθούσες τάσεις.

1. Η ανίχνευση μετοχών που ξεφεύγουν από τα αναμενόμενα άνω και κάτω όρια των τιμών τους, πληροφορία εξαιρετικά χρήσιμη στους συναλλασσόμενους.

Όλες αυτές οι αναγκαιότητες μπορούν σε πολλές περιπτώσεις να εξυπηρετηθούν από συνεχείς επερωτήσεις, οι οποίες θα εκτελούνται πάνω στα χρηματιστηριακά δεδομένα που θα εισέρχονται σε ροές στο σύστημα διαχείρισης. Η έννοια των ροών βρίσκει και εδώ απόλυτη εφαρμογή, τόσο λόγω του όγκου και του υψηλού ρυθμού εμφάνισης των δεδομένων όσο και λόγω της τεράστιας σημασίας που έχει η άμεση (on line) εκτέλεση των επερωτήσεων.

Μπορούμε από το παραπάνω παράδειγμα να παρατηρήσουμε τα παρακάτω γενικά είδη συνεχών επερωτήσεων που θα μας ενδιέφερε να υποστηρίζει ένα σύστημα διαχείρισης ροών δεδομένων και να γίνει εύκολα κατανοητό πως ισχύουν όχι ειδικά για το παράδειγμα αλλά γενικά για τα ΣΔΡΔ

[18]:

- **"Συνεχείς"**: Αποτελεί πλεονασμό αλλά πρόκειται για τις συνεχείς εκείνες επερωτήσεις που πραγματικά εκτελούνται αδιάλειπτα από το σύστημα για όσο διάστημα έχει προσδιοριστεί και οφείλουν να παράγουν αποτελέσματα ακούραστα, σε πραγματικό χρόνο και παρ' όλες τις δυσκολίες στη διαχείριση πόρων.

- **Περιοδικές**: Αυτό το είδος των συνεχών επερωτήσεων παράγει αποτελέσματα σε τακτά χρονικά διαστήματα. Αυτό σημαίνει ότι είτε η ανάγκη που ικανοποιούν είναι περιοδική, είτε ότι ο χρόνος που απαιτούν για να εκτελεστούν είναι υψηλός, είτε, ακόμη, ότι τα δεδομένα που λαμβάνουν μέρος κατά την εκτέλεσή τους χρειάζονται χρόνο για να συγκεντρωθούν.

- **Ειδικές (ad hoc)**: Αυτές οι επερωτήσεις εξυπηρετούν σκοπούς που εμφανίζονται σε όχι τακτική βάση και πρέπει να τεθούν στο σύστημα χωρίς αυτό να είναι αναγκαστικά προετοιμασμένο, από πλευράς ρυθμίσεων και βελτιστοποιήσεων, να τις δεχτεί. Επομένως, είναι επερωτήσεις που τίθενται στο σύστημα για πρώτη φορά και για συγκεκριμένες ανάγκες.

Κλείνοντας με το τελευταίο παράδειγμα, ας δούμε προκαταβολικά πώς θα μπορούσε να διατυπωθεί μια συνεχής επερώτηση σε μια γλώσσα σαν την SQL[3], τη γνωστή γλώσσα διατύπωσης επερωτήσεων σε σχεσιακές βάσεις δεδομένων. Ας υποθέσουμε πως θέλουμε να αναζητήσουμε αν υπάρχει σε κάποια χρονική στιγμή η πιθανότητα κάποιος αγοραστής με μεγάλη οικονομική δυνατότητα, ο οποίος, για να μην τραβήξει την προσοχή της αγοράς αμέσως, αγοράζει μεγάλες ποσότητες από κάποια μετοχή αλλά σε πολλές και μικρές δοσοληψίες.

Έστω ότι μια αγοραπωλησία (tick) έχει την παρακάτω γραμμογράφηση:

- **sID**: προσδιοριστικό οντότητας στην οποία ανήκει η μετοχή

- **MMID**: αναγνωριστικό της οντότητας για λογαριασμό της οποίας γίνεται η αγοραπωλησία.

3 Το παράδειγμα δεν είναι διατυπωμένο σε κάποια πρότυπη μορφή της SQL αλλά σε παραλλαγή της που αποσκοπεί να κάνει κατανοητή σε ένα πρώτο επίπεδο την έννοια της συνεχούς επερώτησης. Σε επόμενο κεφάλαιο θα συζητηθούν γλώσσες συνεχών επερωτήσεων που έχουν προταθεί.

- **price**: τιμή της μετοχής
- **volume**: ποσότητα της αγοραπωλησίας
- **type**: είδος (αγορά/πώληση/ερώτημα)
- **timestamp**: ο ακριβής χρόνος στον οποίο η πράξη εισήλθε στο σύστημα

Η παραπάνω επερώτηση, λοιπόν, θα μπορούσε να εκφραστεί ως εξής:

```
WITH  MaxPrice(bidPrice, timestamp) AS (
      SELECT maxw(range(90, timestamp), price ), timestamp
      FROM Ticks
      WHERE timestamp > now()-30minutes AND
            type      = 'bid'           AND
            sID       = 'ACME'
)
SELECT MMID, count(*)
FROM Ticks    t,
      MaxPrice mp
WHERE t.timestamp = mp.timestamp     AND
       t.timestamp > now()-30 minutes AND
       t.type      = 'bid'            AND
       t.sID       = 'ACME'           AND
       t.price     = mp.bidPrice
GROUP BY MMID
```

Το πρώτο μισό της παραπάνω επερώτησης υπολογίζει τις τιμές ζήτησης μιας μετοχής για τα τελευταία τριάντα λεπτά, θεωρώντας ότι το αίτημα για ζήτηση μιας μετοχής έχει χρόνο ζωής ενενήντα δευτερόλεπτα και προϋποθέτοντας ότι οι αγοραπωλησίες είναι ταξινομημένες χρονικά - στο πεδίο timestamp. Η δήλωση "WITH" προέρχεται από την SQL:1999 [19] και επιτρέπει σε μια επερώτηση να δημιουργήσει μια τοπική προσωρινή όψη (view), με εμβέλεια μόνο την ίδια την επερώτηση. Η συνάρτηση now() επιστρέφει την τρέχουσα χρονική στιγμή του συστήματος κι επομένως η δήλωση now()-30 minutes αναφέρεται στο τελευταίο μισάωρο.

2.3 ΕΚΤΕΛΕΣΗ ΣΥΝΕΧΩΝ ΕΠΕΡΩΤΗΣΕΩΝ

Οι δύο κυρίαρχες προσεγγίσεις στην εκτέλεση και βελτιστοποίηση επερωτήσεων [17] πάνω σε ροές δεδομένων μπορούν να κατηγοριοποιηθούν σε δύο μεγάλες κατηγορίες:

- Στατική προσέγγιση (**static**):
  Η υπόθεση που γίνεται σε αυτή την κατηγορία είναι ότι το σύστημα είναι αρκετά στατικό και, επομένως, ο σχεδιασμός της επεξεργασίας και της βελτιστοποίησης είναι επίσης στατικός και προδιαγεγραμμένος. Έτσι, κατά τη διάρκεια μιας συνεχούς επερώτησης το σύστημα αναμένεται να μην μεταβληθεί ιδιαίτερα. Για παράδειγμα μια ροή που καταφθάνει με σταθερό ρυθμό στο σύστημα όπου εκτελούνται οι επερωτήσεις είναι λογικό να αντιμετωπιστεί με αυτή την προσέγγιση. Το κατά πόσο κάτι τέτοιο μπορεί να ισχύει για τις ροές εξαρτάται από τις ιδιαιτερότητες της εκάστοτε εφαρμογής - πχ σε ένα δίκτυο αισθητήρων, οι αισθητήρες μπορούν να ρυθμιστούν ώστε να αναφέρουν ανά συγκεκριμένα χρονικά διαστήματα, επιτυγχάνοντας περίπου σταθερό ρυθμό δεδομένων.

- Προσαρμοστική προσέγγιση (**adaptive**):
  Σε αυτή την προσέγγιση το περιβάλλον επεξεργασίας θεωρείται εντελώς δυναμικό. Αυτό σημαίνει ότι ένα σύστημα διαχείρισης ροών δεν είναι δυνατό να κάνει υποθέσεις σε σχέση με τη φύση των ροών, όπως, για παράδειγμα οι ρυθμοί άφιξής τους. Το καλύτερο που έχει να κάνει είναι να προσπαθεί να προσαρμόζεται στις συνθήκες που εμφανίζονται κατά τη διάρκεια της ζωής του.

2.4 ΤΕΛΕΣΤΕΣ ΕΠΙΛΟΓΗ – ΠΡΟΒΟΛΗ

Σε ένα μεγάλο τους ποσοστό οι επερωτήσεις που μπορούν να γίνουν σε ροές δεδομένων ομοιάζουν με τις επερωτήσεις που γίνονται όπως γνωρίζουμε στις σχεσιακές βάσεις δεδομένων [6]. Άλλωστε και εδώ το μοντέλο που επιλέξαμε για την αναπαράσταση των δεδομένων είναι το σχεσιακό. Επομένως, κάνοντας χρήση συνεχών επερωτήσεων ένας χρήστης ενός ΣΔΡΔ θα μπορούσε να ανακρίνει τις εισερχόμενες ροές ανάλογα με τα κριτήρια που θα έθετε στην επερώτησή του. Εξαιτίας του ότι οι ροές δεδομένων συνιστούν τις περισσότερες φορές πληροφορία "περαστική" και ατέρμονη οι τελεστές που έχουμε στη διάθεσή μας λειτουργούν στις συνεχείς επερωτήσεις σαν φίλτρα. Δέχονται, δηλαδή, τη ροή σαν είσοδό τους και παράγουν μια άλλη ροή στην έξοδό τους που είναι είτε υποσύνολο της αρχικής ροής ή συνδυασμός στοιχείων που περιέχει.

Δύο από τους πιο συχνά χρησιμοποιούμενους τελεστές της σχεσιακής άλγεβρας και των σχεσιακών βάσεων δεδομένων είναι η επιλογή (selection) και η προβολή (projection) ή αλλιώς τα SELECT ... FROM ... WHERE ... όπως είναι ευρέως γνωστά στα παραδοσιακά συστήματα βάσεων δεδομένων και στη γλώσσα SQL. Οι τελεστές αυτοί μπορούν να υλοποιηθούν σε ένα ΣΔΡΔ με αρκετή ευκολία εξαιτίας της απλότητας στην επεξεργασία που απαιτούν [6].

Πιο συγκεκριμένα, για να υλοποιηθεί μια επιλογή και προβολή σε ένα ΣΔΡΔ το μόνο που είναι αναγκαίο είναι η έννοια του φίλτρου που προαναφέραμε. Αυτό σημαίνει ότι καθώς τα δεδομένα της ροής καταφθάνουν με τη μορφή πλειάδων το σύστημα δεν έχει παρά να απορρίψει τις πλειάδες που δεν ικανοποιούν τη συνθήκη της επερώτησης, αν αυτή υπάρχει, και να εμφανίσει τα πεδία που έχουν προσδιοριστεί στην επερώτηση για τις πλειάδες εκείνες που ικανοποιούν τη δοσμένη συνθήκη.

Για να γίνει αυτό εντελώς κατανοητό ας θυμηθούμε το παράδειγμα από το προηγούμενο κεφάλαιο από το χώρο της διαχείρισης δικτύων, όπου μία από τις πλειάδες που λαμβάνει μέρος στο σύστημα είναι αυτή που αντιστοιχεί στα πακέτα που πηγαινοέρχονται στο δίκτυο και έχουν την εξής γραμμογράφηση που έχουμε ήδη περιγράψει: [src, dest, id, len, time]. Ας θεωρήσουμε, ακόμη, μια απλή επερώτηση που κάνει χρήση μόνο επιλογής και προβολής: "Να επιλεγεί ο συνολικός φόρτος - συνολικό μέγεθος - των πακέτων που μεταφέρθηκαν από την IP 198.168.0.1 προς την IP 198.168.0.2".

Πρόκειται για μια συνεχόμενη επερώτηση που ανά πάσα στιγμή θα μας πληροφορεί για το συνολικό μέγεθος δεδομένων που μεταφέρθηκαν ανάμεσα στις προσδιοριζόμενες IPs, χωρίς να διατηρεί αρχείο των πακέτων που έχουν περάσει. Κάτι τέτοιο θα μπορούσε να εκφραστεί σε σύνταξη τύπου SQL ως εξής:

```
SELECT sum(P.len)
FROM Packets P
WHERE P.src  = '198.168.0.1' AND
      P.dest = '198.168.0.2'
```

Η παραπάνω επερώτηση είναι εξαιρετικά απλή τόσο στο να κατανοηθεί όσο και στο να υλοποιηθεί. Το ΣΔΡΔ που παρακολουθεί τα πακέτα του δικτύου μπορεί εύκολα καθώς δέχεται τη ροή των πλειάδων να ελέγχει σε κάθε μια την IP πηγής και προορισμού και αν η συνθήκη ικανοποιείται τότε να προσθέτει το μέγεθος του συγκεκριμένου πακέτου σε ένα μετρητή. Με αυτό τον τρόπο, δίχως να

έχει καταχωρημένες σε σταθερό αποθηκευτικό μέσο όλες τις πλειάδες, θα μπορεί πάντα να αποφανθεί για το συνολικό μέγεθος που μεταφέρθηκε από τη συγκεκριμένη πηγή προς τον συγκεκριμένο προορισμό. Το τελευταίο, αν και κάπως αυτονόητο σε ένα σύστημα που παρακολουθεί πακέτα δικτύου, αξίζει να σημειωθεί για άλλη μία φορά, μιας και καταδεικνύει μια από τις σημαντικότερες ιδιότητες των ροών.

Η κατάσταση διαφοροποιείται αρκετά αν στην παραπάνω επερώτηση προστεθεί η φράση "... στα τελευταία 30 λεπτά". Το ερώτημα, μολονότι και πάλι συνεχές, δεν αναφέρεται πλέον στο σύνολο των πλειάδων/πακέτων - από τη στιγμή πάντοτε που η επερώτηση εκκινήθηκε - αλλά στα τελευταία 30 λεπτά. Το λεπτό σημείο που πρέπει να κατανοήσει κανείς είναι ότι αναφερόμαστε ανά πάσα στιγμή στα 30 προηγούμενα λεπτά, κάτι που σημαίνει πως σε κάθε επόμενη χρονική στιγμή μιλούμε για 30 διαφορετικά λεπτά που περιέχουν διαφορετικές πλειάδες. Στην πραγματικότητα δεν περιέχουν εντελώς διαφορετικές πλειάδες παρά διαφέρουν στις νέες πλειάδες που μόλις εισήλθαν και στις γηραιότερες από 30 λεπτά - πλειάδες που δεν ενδιαφέρουν πια - και έχουν απορριφθεί. Αυτή η προσέγγιση βάζει τις πλειάδες σε μια FIFO λίστα με μέγιστο εύρος 30 λεπτών μέσα στην οποία μας ενδιαφέρει να πάρουμε τα αποτελέσματα που ζητούνται. Κάθε νέα πλειάδα εισέρχεται στην κορυφή της λίστας και, μετά την πάροδο των 30 λεπτών που έχει προσδιοριστεί – και καθώς νέες πλειάδες τοποθετούνται με τη σειρά τους στην κορυφή -, η πλειάδα απορρίπτεται.

Η πολυπλοκότητα γίνεται αισθητή όταν πολλές παρόμοιες, έστω και απλές επερωτήσεις, καταχωρούνται στο σύστημα με διαφορετικές ανάγκες για επεξεργασία και διαφορετικό εύρος η κάθε μια. Τότε το σύστημα οφείλει να εφαρμόσει αλγορίθμους για να εξοικονομήσει πόρους τόσο σε επεξεργαστική ισχύ όσο και σε μνήμη. Η επιλογή των αλγορίθμων εξαρτάται από το αν η προσέγγιση στην επεξεργασία της επερώτησης είναι στατική ή προσαρμοστική. Για παράδειγμα αν είχαν τεθεί δύο επερωτήσεις σαν την τελευταία αλλά με διαφορετικά εύρη ενδιαφέροντος η κάθε μία - έστω 30 και 60 λεπτά αντίστοιχα - το σύστημα δεν είναι ανάγκη να χρησιμοποιήσει διαφορετικές δομές για την κάθε μία, μιας και η δομή που θα διατηρούσε για τη δεύτερη θα κάλυπτε τις ανάγκες και για την πρώτη. Αυτή η τακτική θα χαρακτηριζόταν στατική εφόσον είναι εξαρχής καθορισμένη. Μια άλλη τακτική σε περίπτωση, για παράδειγμα, εμφάνισης μεγάλων ρυθμών άφιξης των πλειάδων, θα ήταν η επιλεκτική απόρριψη ορισμένων πλειάδων χωρίς να υποστούν επεξεργασία για λόγους εξομάλυνσης του φόρτου. Αυτή η τακτική θα είχε κάπως πιο δυναμικό χαρακτήρα. Ακόμη, μπορούμε να σημειώσουμε ότι αντί στην επερώτηση να προσδιοριστεί χρονικό εύρος είναι δυνατόν να προσδιοριστεί το εύρος σε πλήθος πρόσφατων πλειάδων. Με αυτό τον τρόπο, για παράδειγμα, η προαναφερθείσα επερώτηση θα μπορούσε να τελειώνει λέγοντας “... για τα τελευταία 10.000 πακέτα”.

2.5 ΣΥΝΕΝΏΣΕΙΣ (JOINS)

Σημαντικό ενδιαφέρον, στη συνέχεια, παρουσιάζουν οι τελεστές συνένωσης (joins), οι οποίοι είναι αυτοί που μας επιτρέπουν να εκτελούμε επερωτήσεις πάνω σε συσχετίσεις ροών, με τον ίδιο τρόπο με τον οποίο μας επιτρέπουν να εκτελούμε επερωτήσεις πάνω σε συσχετίσεις πινάκων στις παραδοσιακές βάσεις δεδομένων. Προτού μιλήσουμε για τις ίδιες τις συνενώσεις θα αναφερθούμε σε ορισμένες μετρικές που χρησιμοποιούνται στην εκτίμηση του κόστους εκτέλεσης επερωτήσεων με συνενώσεις. Είναι γνωστό ότι οι πράξεις συνένωσης μπορούν να αποβούν ιδιαίτερα χρονοβόρες [20][21] κι επομένως η εκτέλεσή τους πρέπει να βασίζεται σε προσεκτικά επιλεγμένους αλγορίθμους.

Θεωρούμε πως ένας αλγόριθμος δέχεται πλειάδες και παράγει πάλι πλειάδες. Μας ενδιαφέρει να εκτιμήσουμε όχι μόνο το κόστος στην επεξεργασία μιας επερώτησης - τον υπολογιστικό χρόνο που απαιτεί - αλλά και αλλά και το ρυθμό με τον οποίο απελευθερώνει τις παραγόμενες πλειάδες [17]. Χρησιμοποιούμε τις παρακάτω ποσότητες:

- $\mathbf{f_p}$ = Ο παράγοντας επιλεξιμότητας (selectivity) μιας ροής σε σχέση με μια επερώτηση. Δηλαδή, η πιθανότητα μια πλειάδα της να ικανοποιεί τη συνθήκη της επερώτησης κι επομένως να συμπεριληφθεί στα αποτελέσματα.

- $\boldsymbol{\lambda_S}$ = Ο μέσος ρυθμός άφιξης των πλειάδων μιας ροής, δηλαδή το πηλίκο των πλειάδων που καταφθάνουν προς τη μονάδα του χρόνου.

- $\mathbf{W_S}$ = Το μέγεθος του πλαισίου (window) που έχει προσδιοριστεί για μια ροή (για τα πλαίσια έγινε αναφορά στο προηγούμενο κεφάλαιο και θα ακολουθήσει εκτενέστερη σε επόμενο).

- $\boldsymbol{\mu_s(S)}$ = Το υπολογιστικό κόστος που χρειάζεται το σύστημα να δαπανήσει για την αποθήκευση μιας πλειάδας της S σε μια δομή δεδομένων στη μνήμη.

- $\boldsymbol{\mu_m(S)}$ = Το υπολογιστικό κόστος που χρειάζεται το σύστημα να δαπανήσει για βρει πλειάδες μέσα στη μνήμη όπου είναι αποθηκευμένη η S - να εντοπίσει, δηλαδή, πλειάδες που κάποιο τους πεδίο πληρεί κάποια δοσμένη συνθήκη.

- **$\mu_i(S)$ =** Το υπολογιστικό κόστος που χρειάζεται το σύστημα να δαπανήσει για να ακυρώσει μια πλειάδα της S μέσα στη μνήμη όπου είναι αποθηκευμένη η S – να την απορρίψει, δηλαδή, ως πεπαλαιωμένη από το πλαίσιο (window) της ροής.

Από τα παραπάνω μπορούμε επιπλέον να θεωρήσουμε τα εξής:

- **$\lambda_O = f_p \times \lambda_S$**, ο ρυθμός παραγωγής πλειάδων της επερώτησης.

- **$W_O = f_p \times W_S$**, το μέγεθος του πλαισίου της παραγόμενης από την επερώτηση ροής.

- Για μια επερώτηση με απλή επιλογή και προβολή χωρίς συνθήκη το f_p έχει τιμή 1.

Ένα από τα μεγαλύτερα προβλήματα που έχει να αντιμετωπίσει ένα ΣΔΡΔ στην εκτέλεση επερωτήσεων με συνενώσεις είναι καταρχάς το γεγονός ότι οι ροές που λαμβάνουν μέρος στη συνένωση καταφθάνουν ασύγχρονα. Δε μπορεί, δηλαδή να γίνει καμία υπόθεση για τη διαδοχή των πλειάδων των ροών, εκτός και αν κάτι τέτοιο ενυπάρχει στο σύστημα και στη σημασιολογία των συγκεκριμένων ροών. Αυτό δε δημιουργεί επιπλοκές στις επερωτήσεις με απλές επιλογές και προβολές παρά μόνο ως προς το ότι ότι η παραγωγή των αποτελεσμάτων θα είναι τόσο άμεση ή καθυστερημένη όσο και η ίδια η εμφάνιση των πλειάδων. Δηλαδή, αν κάποιες πλειάδες αργήσουν να φτάσουν στο σύστημα επεξεργασίας ή φτάσουν με άτακτη σειρά, τότε θα υπάρξει αντίστοιχη καθυστέρηση στην παραγωγή των αποτελεσμάτων, κάτι για το οποίο το σύστημα επεξεργασίας δεν μπορεί να κάνει κάτι.

Όσον αφορά, όμως, επερωτήσεις με συνενώσεις τα πράγματα είναι κάπως πιο περίπλοκα, καθώς για να μπορέσει να ολοκληρωθεί μια συνένωση είναι απαραίτητο να είναι διαθέσιμες όλες οι πλειάδες των ροών που λαμβάνουν μέρος στη συνένωση. Ή τουλάχιστον να είναι διαθέσιμο όλο το εύρος τους που έχει προσδιοριστεί να βρίσκεται στη μνήμη του συστήματος επεξεργασίας [17]. Στα παραδοσιακά ΣΔΒΔ μια τέτοια κατάσταση δεν αποτελεί πρόβλημα μιας και τα δεδομένα βρίσκονται όλα αποθηκευμένα στο δίσκο. Στις ροές το ζήτημα αντιμετωπίζεται κύρια με αλγορίθμους που επιχειρούν να παράγουν πλειάδες αποτελέσματος όσο το δυνατόν πιο σύντομα και όχι να ολοκληρώσουν το σύνολο της επερώτησης με συνενώσεις κι έπειτα να αποδώσουν το όλο αποτέλεσμα [22].

2.5.1 ΕΚΤΈΛΕΣΗ ΣΥΝΕΝΏΣΕΩΝ

Στην πιο γενική της μορφή μορφή η ιδέα αυτή της ασύγχρονης συνένωσης σχέσεων - ή, στη συγκεκριμένη περίπτωση, ροών - προσδιορίζεται στον αλγόριθμο της "συμμετρικής[4] συνένωσης με συνάρτηση σύνοψης" (symmetric hash join - SHJ). Η ιδέα έχει ως εξής [22][23]:

- Θεωρούμε τη συνένωση ισότητας (equi-join) της ροής R με τη ροή S πάνω στα πεδία R.a και S.b, τα οποία είναι, φυσικά, ίδιου τύπου. Δηλαδή τη συνένωση $R \bowtie_{R.a=S.b} S$.

- Θεωρούμε ότι τόσο η R όσο και η S χωρούν εξολοκλήρου στη μνήμη - ή τουλάχιστο ότι επεξεργαζόμαστε ένα εύρος τους τόσο μεγάλο όσο η μνήμη μας επιτρέπει. Μια τέτοια παραδοχή είναι αναπόφευκτη στη διαχείριση των ροών αν θέλει κανείς να μιλά για αποτελέσματα σε πραγματικό χρόνο (real time).

- Δημιουργείται ένας πίνακας σύνοψης (hash table) για κάθε μία εκ των R και S.

- Καθώς πλειάδες για την R και την S καταφθάνουν στο σύστημα επεξεργασίας:

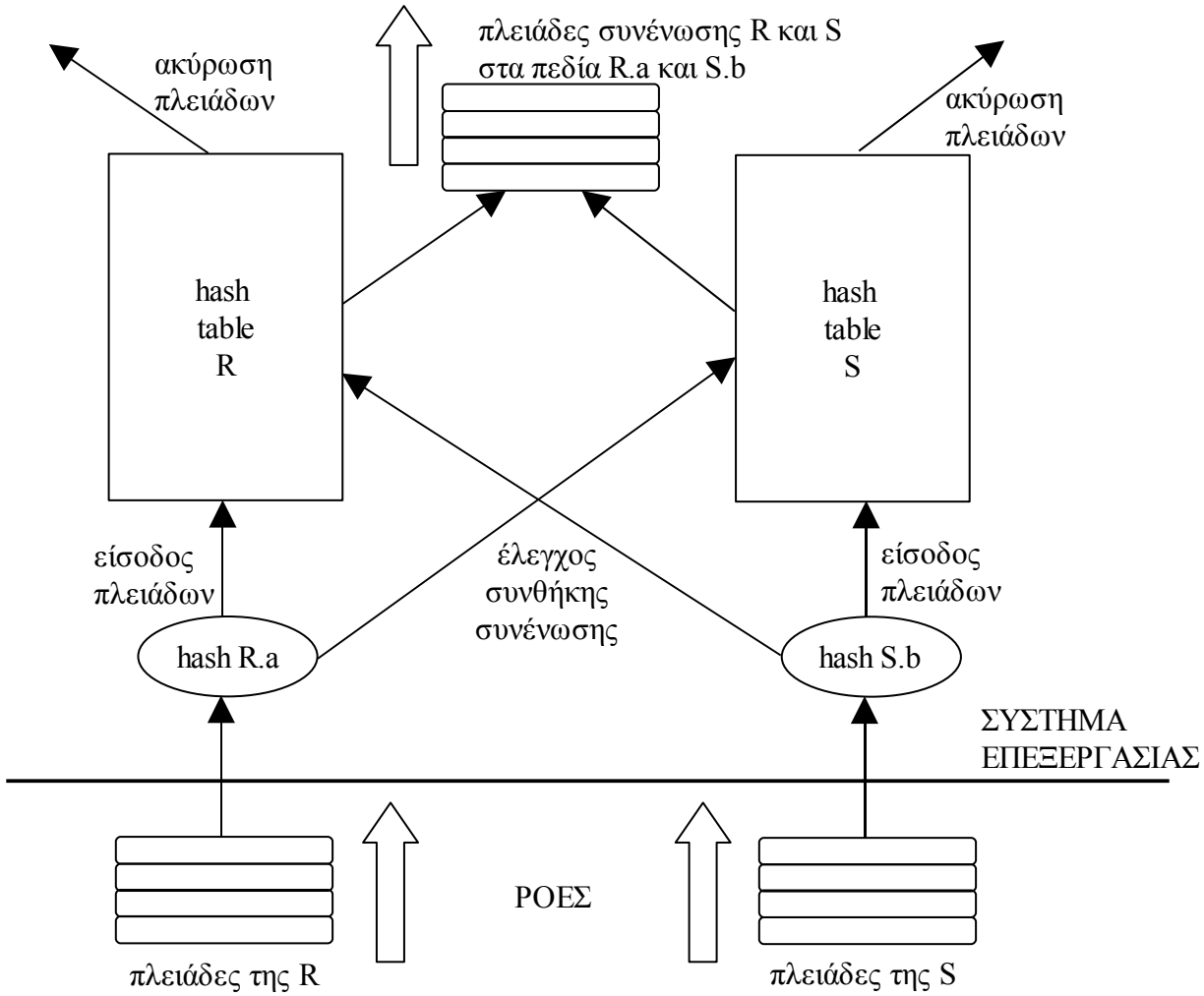
    - Η συνάρτηση σύνοψης (hash) επενεργεί πάνω στην κάθε πλειάδα της κάθε ροής στο πεδίο στο οποίο γίνεται η συνένωση και την εισάγει σε έναν αντίστοιχο πίνακα σύνοψης. Έτσι χτίζεται ένας πίνακας σύνοψης για κάθε μια από τις ροές.

    - Για κάθε πλειάδα της R γίνεται αναζήτηση για πλειάδες με ισότητα στο πεδίο συνένωσης μέσα στον πίνακα σύνοψης της S. Αντίστοιχη είναι η διαδικασία και για την S - για κάθε πλειάδα της γίνεται αναζήτηση για πλειάδες με ισότητα στο πεδίο συνένωσης μέσα στον πίνακα σύνοψης της R.

Όπως προαναφέραμε, η παραπάνω ιδέα βασίζεται στο γεγονός ότι επεξεργαζόμαστε ένα εύρος από την κάθε ροή τέτοιου μεγέθους που να μας το επιτρέπει τουλάχιστο η διαθέσιμη μνήμη. Την ποσότητα αυτή την έχουμε ονομάσει πλαίσιο (window) και αναφέρεται στις πιο πρόσφατες πλειάδες μιας ροής. Το μέγεθός του προσδιορίζεται είτε με ένα πλήθος πλειάδων είτε με ένα χρονικό όριο που δείχνει το βάθος στο παρελθόν πέρα από το οποίο οι πλειάδες δε μας ενδιαφέρουν

4 Συμμετρική διότι χρησιμοποιεί την ίδια δομή δεδομένων για την αναπαράσταση και των δύο μελών της συνένωσης – στην προκειμένη περίπτωση πίνακα σύνοψης (hash table). Στην πραγματικότητα είναι δυνατόν να χρησιμοποιηθούν οποιεσδήποτε δομές κριθούν απαραίτητες για την κάθε ροή – πχ πίνακας σύνοψης για τη μία και B-δέντρο για την άλλη.

πλέον. Πέρα, λοιπόν, από το κόστος που έχουν οι πράξεις καθεαυτές της ίδιας της συνένωσης υπάρχει και η ανάγκη για ακύρωση (invalidation) των πεπαλαιωμένων πλειάδων. Αν δούμε αυτό το πλαίσιο σαν μια FIFO λίστα τότε το κόστος αυτό αντιστοιχεί στο κόστος εισαγωγής νέων στοιχείων στην αρχή της λίστας και εξαγωγής των παλαιών στοιχείων από το τέλος της. Το σχήμα 2.1 αναπαριστά την όλη διαδικασία.



Σχήμα 2.1 - Αλγόριθμος SHJ

Μια χρήσιμη παρατήρηση πάνω στην παραπάνω διαδικασία είναι ότι ο πίνακας σύνοψης δεν είναι, βέβαια, η μόνη εναλλακτική λύση σε ένα τέτοιο αλγόριθμο. Οποιασδήποτε μορφής δομή δεδομένων με ευρετηρίαση στο πεδίο της συνένωσης, όπως δέντρα ή λίστες, μπορούν να χρησιμοποιηθούν για να αναπαραστήσουν τις πλειάδες της ροής αφότου αυτές εισέλθουν στο σύστημα επεξεργασίας. Η επιλογή της δομής εξαρτάται από τις δυνατότητες βελτιστοποίησης και τις ανάγκες της κάθε επερώτησης. Για παράδειγμα, ο πίνακας σύνοψης (hash table) μπορεί να

χρησιμοποιηθεί μόνο για συνενώσεις ισότητας και όχι για γενικού τύπου θήτα συνενώσεις (theta-join)[5]. Σε περίπτωση που κάτι τέτοιο είναι επιθυμητό τότε κάποια άλλη δομή θα πρέπει να προτιμηθεί. Ακόμη, στην επιλογή αυτή μπορεί να παίξει ρόλο η αποδοτικότητα της δομής στις μεταφορές των πλειάδων από ένα σημείο της μνήμης στο άλλο, πράξη που, όπως φαίνεται, είναι αρκετά συχνή καθώς οι πλειάδες εισέρχονται, ακυρώνονται και αναζητούνται συνεχώς.

2.5.2 ΕΚΤΙΜΗΣΗ ΚΌΣΤΟΥΣ ΣΥΝΕΝΏΣΕΩΝ

Ας προσπαθήσουμε, στη συνέχεια να εκτιμήσουμε την ποσότητα των πλειάδων που η συνένωση θα παράξει σαν αποτέλεσμα. Θεωρούμε ότι:

- Μετά από χρόνο t έχουν αναγνωσθεί $\mathbf{W_R(t) = \lambda_R \ x \ t}$ πλειάδες της ροής R.

- Μετά από χρόνο t έχουν αναγνωσθεί $\mathbf{W_S(t) = \lambda_S \ x \ t}$ πλειάδες της ροής S.

- Κατά τη διάρκεια της χρονικής μονάδας t θα υπάρξουν λ_R αφίξεις πλειάδων της ροής R, οι οποίες θα συνενωθούν με τις υπάρχουσες πλειάδες της S, που ως εκείνη τη στιγμή είναι $W_S(t)$.

- Κατά τη διάρκεια της ίδιας της χρονικής μονάδας t θα υπάρξουν λ_S αφίξεις πλειάδων της ροής S, οι οποίες θα συνενωθούν με τις υπάρχουσες πλειάδες της R, που ως εκείνη τη στιγμή είναι $W_R(t)$.

- Κατά τη διάρκεια της χρονικής μονάδας t θα παραχθούν $\mathbf{f_p \ x \ \lambda_R \ x \ W_s(t)}$ πλειάδες αποτελέσματος λόγω των αφίξεων της R, όπου f_p είναι ο παράγοντας επιλεξιμότητας (selectivity) της συνένωσης.

- Κατά τη διάρκεια της χρονικής μονάδας t θα παραχθούν $\mathbf{f_p \ x \ \lambda_S \ x \ W_R(t)}$ πλειάδες αποτελέσματος λόγω των αφίξεων της S, όπου f_p είναι ο παράγοντας επιλεξιμότητας (selectivity) της συνένωσης.

Παρατηρούμε ότι οι πλειάδες αποτελέσματος είναι κατά τη χρονική μονάδα t:

---

5 Theta-join (Θ-συνένωση): συνένωση σχέσεων – ή, εν προκειμένω, ροών – πάνω σε κοινού τύπου πεδίο αλλά με σχέση ανισότητας - πχ συνένωση της σχέσης R με τη σχέση S με συνθήκη συνένωσης R.a ≥ S.b (και όχι R.a = S.b). Σε μια τέτοια συνένωση επιλέγονται οι πλειάδες του καρτεσιανού γινομένου των δύο σχέσεων που πληρούν τη συγκεκριμένη συνθήκη, όποια και αν είναι αυτή.

$$\mathbf{f_p \times \lambda_R \times W_s(t) + f_p \times \lambda_S \times W_R(t) - f_p \times \lambda_R \times \lambda_S},$$

το οποίο δείχνει τον αριθμό των πλειάδων αποτελέσματος που προστίθενται στο συνολικό αποτέλεσμα κατά τη χρονική μονάδα t.

Ο λόγος που αφαιρούμε τον τελευταίο παράγοντα $f_p \times \lambda_R \times \lambda_S$ είναι επειδή οι πλειάδες αποτελέσματος που αντιπροσωπεύει έχουν καταμετρηθεί δύο φορές από τον αλγόριθμο, μία κατά τις αφίξεις της R και μία κατά τις αφίξεις της S. Ακόμη, ο τρόπος με τον οποίο περιγράψαμε τον αλγόριθμο δεν εξηγεί το πώς υλοποιείται η έννοια του πλαισίου και του πώς αυτή επηρεάζει τα αποτελέσματα. Όταν αυτό – το πλαίσιο – λειτουργεί πραγματικά αυτό που συμβαίνει είναι ότι αντί για $W_R(t)$ και $W_s(t)$, τα οποία είναι συναρτήσεις του χρόνου, θα έχουμε απλώς W_R και W_s τα οποία είναι πλαίσια σταθερού μεγέθους. Σε αυτή την περίπτωση, όπου πράγματι οι παλαιές πλειάδες ακυρώνονται (invalidate) ο παράγοντας $f_p \times \lambda_R \times \lambda_S$ δε χρειάζεται να αφαιρεθεί και τότε το αναμενόμενο πλήθος του αποτελέσματος γίνεται $f_P \times (\lambda_R \times W_S + \lambda_s \times W_R)$.

Επιπλέον, πλαίσια είναι δυνατόν να εφαρμοστούν και στις ίδιες τις πλειάδες αποτελέσματος. Εφόσον δεν είναι επιθυμητό – ή, ίσως, ούτε και δυνατόν, ανάλογα με τους πόρους του συστήματος – δεν είναι απαραίτητο να διατηρούνται στη μνήμη όλες οι πλειάδες αποτελέσματος που έχουν παραχθεί μέχρι μια δεδομένη χρονική στιγμή. Είναι δυνατόν, επομένως, να υπάρχει όριο στο μέγεθος του αποτελέσματος. Το αποτέλεσμα θα λειτουργεί και αυτό σαν μια FIFO λίστα, όπου νέες πλειάδες αποτελέσματος εισέρχονται και παλαιότερες αφαιρούνται (ακυρώνονται), με αναμενόμενο μέγεθος $f_P \times W_R \times W_S$.

Ένα ενδιαφέρον σημείο των συνενώσεων με πλαίσια συνάγεται από την παρατήρηση ότι η συνένωση $R \bowtie_{R.a=S.b} S$ μπορεί να αποδομηθεί σε δύο παράλληλες συνενώσεις. Αυτές είναι οι:

- $\mathbf{R \bowtie_{R.a=S.b} S}$ που αντιπροσωπεύει το τμήμα της συνένωσης που θα εκτελεστεί από την πλευρά της R και

- $\mathbf{R \bowtie_{R.a=S.b} S}$ που αντιπροσωπεύει το τμήμα της συνένωσης που θα εκτελεστεί από την πλευρά της S.

Αυτό σημαίνει ότι η όλη συνένωση μπορεί να υλοποιηθεί και με μη συμμετρικό τρόπο. Δηλαδή, η δομή δεδομένων που θα αναπαριστά την κάθε μία από τις ροές ή το κάθε ένα από τα πλαίσια των ροών δεν είναι απαραίτητο να είναι το ίδιο και για τις δύο. Είναι δυνατόν να χρησιμοποιηθεί διαφορετική δομή για κάθε ροή κάτι που θα κριθεί από τις ανάγκες της εκάστοτε εφαρμογής. Για

παράδειγμα, σε μία ροή πάνω στην οποία γίνονται μόνο επερωτήσεις με συνθήκες ισότητας πάνω στο πεδίο ευρετηρίασης ένας πίνακας σύνοψης είναι ιδιαίτερα αποτελεσματικός. Αν, από την άλλη πλευρά, σε μια ροή γίνονται πιο περίπλοκες επερωτήσεις τότε μια πιο περιεκτική δομή, όπως ένα δέντρο, που υποστηρίζει επερωτήσεις με ανισότητες είναι αναγκαία.

Εύκολα κατανοεί κανείς ότι η ίδια δομή είναι επιθυμητό για λόγους οικονομίας να χρησιμοποιηθεί σε όσο το δυνατόν περισσότερες περιπτώσεις. Για παράδειγμα αν μια συνεχής επερώτηση με συνένωση απαιτεί τη δημιουργία ενός B-δέντρου πάνω σε ένα πεδίο a μιας ροής R, τότε η ίδια δομή είναι επιθυμητό να χρησιμοποιηθεί και σε μια άλλη ταυτόχρονη επερώτηση που ζητά πλειάδες της R τέτοιες ώστε να ικανοποιείται μια συνθήκη πάνω στο πεδίο a.

Στη συνέχεια ας δούμε πώς γίνεται η εκτίμηση του ίδιου του κόστους της συνένωσης που χρησιμοποιήθηκε ως παράδειγμα στον αλγόριθμο SHJ που προαναφέραμε. Μετά την παραπάνω διαίρεση της συνένωσης $R \bowtie_{R.a=S.b} S$ σε $R \ltimes_{R.a=S.b} S$ και $R \rhd_{R.a=S.b} S$ θα θεωρήσουμε ότι το συνολικό κόστος μ αυτής θα ισούται με τα επί μέρους κόστη της κάθε μιας από αυτές τις δύο.

Μπορούμε να πούμε ότι το κόστος για την $R \ltimes_{R.a=S.b} S$ είναι συνάρτηση των:

- Πλήθος αφίξεων της R στη μονάδα του χρόνου επί το κόστος αποθήκευσης των νέων πλειάδων στη δομή δεδομένων της και ακύρωσης των παλαιότερων.

- Πλήθος αφίξεων της S στη μονάδα του χρόνου επί το κόστος αναζήτησης στη δομή δεδομένων της R για ισότητα στο πεδίο της συνένωσης. Ίσως φαντάζει περίεργο να εξαρτάται το κόστος από τη μεριά της R από το πλήθος αφίξεων της S, όμως αυτό ακριβώς το πλήθος αφίξεων της S είναι που θα καθορίσει πόσες φορές θα γίνει πρόσβαση στη δομή δεδομένων της R προς αναζήτηση πλειάδων που ικανοποιούν τη συνθήκη της συνένωσης.

Αντίστοιχα ισχύουν και για το κόστος για την $R \rhd_{R.a=S.b} S$.

Επομένως το συνολικό κόστος θα είναι:

$$\boldsymbol{\lambda_S \times \mu_m(R) + \lambda_R \times [\mu_S(R) + \mu_i(R)]} \text{ (εξαιτίας της } R \ltimes_{R.a=S.b} S)$$

$$+$$

$$\boldsymbol{\lambda_R \times \mu_m(S) + \lambda_S \times [\mu_S(S) + \mu_i(S)]} \text{ (εξαιτίας της } R \rhd_{R.a=S.b} S)$$

2.5.3 ΣΥΝΕΝΏΣΕΙΣ ΠΟΥ ΞΕΠΕΡΝΟΎΝ ΤΑ ΌΡΙΑ ΤΗΣ ΜΝΉΜΗΣ

Μέχρι τώρα αναφερθήκαμε σε συνενώσεις των οποίων οι απαιτήσεις σε πόρους περιορίζονται στα μεγέθη της διαθέσιμης μνήμης του συστήματος επεξεργασίας. Η προσέγγιση έγινε μέσω του αλγορίθμου SHJ (symmetric hash join) που αποτελεί θεμελιώδη αλγόριθμο πάνω στον οποίο έχουν χτιστεί πολλές παραλλαγές με σκοπό τη βελτίωση της αποδοτικότητας. Παρολαυτά, παραμένει το ερώτημα του τι θα πρέπει το σύστημα να πράξει σε περιπτώσεις όπου η διαθέσιμη μνήμη δεν επαρκεί. Κάτι τέτοιο δεν είναι καθόλου απίθανο να συμβεί σε ένα περιβάλλον ροών δεδομένων όπου οι εγγυήσεις που έχουμε για τις συνθήκες είναι σε μεγάλο βαθμό μεταβλητές και απρόβλεπτες.

Χαρακτηριστικό παράδειγμα είναι περιπτώσεις όπου κάποια ροή εμφανίζεται σε μερικά χρονικά διαστήματα να καταφθάνει σε ιδιαίτερα υψηλούς ρυθμούς [4][8]. Είναι δυνατόν ο ρυθμός να γίνει τόσο μεγάλος που η διαθέσιμη μνήμη να μην επαρκεί από μόνη της για την αποθήκευση των δομών της ροής. Το πρόβλημα γίνεται ακόμη πιο έντονο και ορατό όταν μια τέτοια ροή απαιτείται να συνενωθεί με κάποια άλλη, η οποία μπορεί είτε να εμφανίζεται με εξίσου μεγάλες ριπές (bursts) είτε, αντίθετα, με πολύ μικρό ρυθμό. Και στις δύο περιπτώσεις υπάρχει ανάγκη για ιδιαίτερη αντιμετώπιση έτσι ώστε και να μην τίθεται σε κίνδυνο η διαθεσιμότητα του συστήματος επεξεργασίας αλλά και να είναι σε θέση – το σύστημα – να παράγει συνεχή αποτελέσματα.

Στην αντιμετώπιση αυτών των προβλημάτων ένας από τους πιο διαδεδομένους αλγορίθμους που έχουν παρουσιαστεί είναι ο XJoin [24][25], ο οποίος αποτελεί επέκταση της ιδέας του SHJ. Ο XJoin επεκτείνει τον SHJ έτσι ώστε να κάνει χρήση και σταθερού αποθηκευτικού μέσου (σκληρός δίσκος) πέρα από τη μνήμη, για να αντιμετωπίσει τα μεγάλα εισερχόμενα μεγέθη των ροών. Βέβαια, αυτό δεν είναι από μόνο του αρκετό. Δεν έχει, για παράδειγμα κανένα νόημα στην περίπτωση που μία από τις ροές εμφανίζεται με αφύσικες καθυστερήσεις. Για αυτό το λόγο ένα από τα κομβικά στοιχεία του αλγορίθμου είναι μια “ενεργά προγραμματιζόμενη” (reactively scheduled) διεργασία παρασκηνίου (background process), η οποία έχει σαν κύριο σκοπό της να εκμεταλλεύεται το νεκρό χρόνο των καθυστερήσεων για να μπορέσει να παράγει περισσότερες πλειάδες αποτελέσματος και μάλιστα όσο το δυνατόν νωρίτερα.

Η προσέγγιση του XJoin βασίζεται στις εξής δύο αρχές:

- Παραγωγή πλειάδων αποτελέσματος καθώς οι ροές καταφθάνουν, έτσι να υπάρχει κατά

το δυνατόν άμεση αλλά και συνεχόμενη απόκριση στην εκάστοτε επερώτηση. Αυτό σημαίνει ότι οι πλειάδες αποτελέσματος επιστρέφονται σταδιακά σαν απάντηση και καθώς υπολογίζονται.

- Δυνατότητα εξέλιξης της προόδου της επερώτησης ακόμη και όταν κάποια από τις ροές έρχεται με προβλήματα στη μετάδοση. Λέγοντας "προβλήματα" [26] χρειάζεται να συμπεριλάβουμε γενικότερα τόσο την περίπτωση η αρχική παραγωγή των πλειάδων να αργήσει να ξεκινήσει όσο και τις χρονικές περιόδους όπου ο ρυθμός τους μπορεί να πέσει πολύ χαμηλά. Ακόμη, στην ίδια κατηγορία προβλημάτων τοποθετούνται και οι πιθανές διακυμάνσεις του ρυθμού της ροής από φυσιολογικούς σε πολύ μεγάλους ρυθμούς και το αντίστροφο.

Όπως μπορεί να παρατηρηθεί, παρόμοιες είναι και οι ανάγκες που καλείται να ικανοποιήσει και ο SHJ. Στην περίπτωση του XJoin, όμως, είναι δυνατή η επεξεργασία μεγαλύτερων σε μέγεθος ροών, αφού υπάρχει η δυνατότητα προσωρινής αποθήκευσής τους στο δίσκο καθώς και πιο ευέλικτη η δυνατότητα πολλαπλών συνενώσεων.

Ας εξετάσουμε, στη συνέχεια τη λειτουργία του XJoin. Ας θεωρήσουμε για άλλη μια φορά τη συνένωση ισότητας $R \bowtie_{R.a=S.b} S$ και:

- $\mathbf{M_R}$: το πλήθος των πλειάδων της R που είναι δυνατόν να αποθηκευθούν στη μνήμη του συστήματος, δεδομένου ότι αυτή είναι περιορισμένη και υπάρχουν και άλλες ροές που πρέπει επίσης να αποθηκευθούν.

- $\mathbf{M_S}$: το πλήθος των πλειάδων της R που είναι δυνατόν να αποθηκευθούν στη μνήμη του συστήματος, αντίστοιχα.

- **|...|:** το συνολικό πλήθος των πλειάδων μιας ροής, πχ |R| και |S|, θεωρώντας ότι τόσο $M_R < |R|$ όσο και $M_S < |S|$, ώστε καμία από τις δύο ροές να μη χωρά εξολοκλήρου στη μνήμη.

Στον αλγόριθμο XJoin χρησιμοποιούνται και πάλι πίνακες σύνοψης (hash tables) για την αναπαράσταση των ροών. Κάθε, όμως, πίνακας σύνοψης διαμοιράζεται σε m διαμερίσματα (partitions) τα οποία συμβολίζουμε $\mathbf{p_1^R,..., p_m^R}$ για την R και $\mathbf{p_1^S,..., p_m^S}$ για την S, αντίστοιχα. Κάθε εισερχόμενη πλειάδα περνά δύο φορές από συνάρτηση σύνοψης. Μία για να προσδιοριστεί σε ποιο

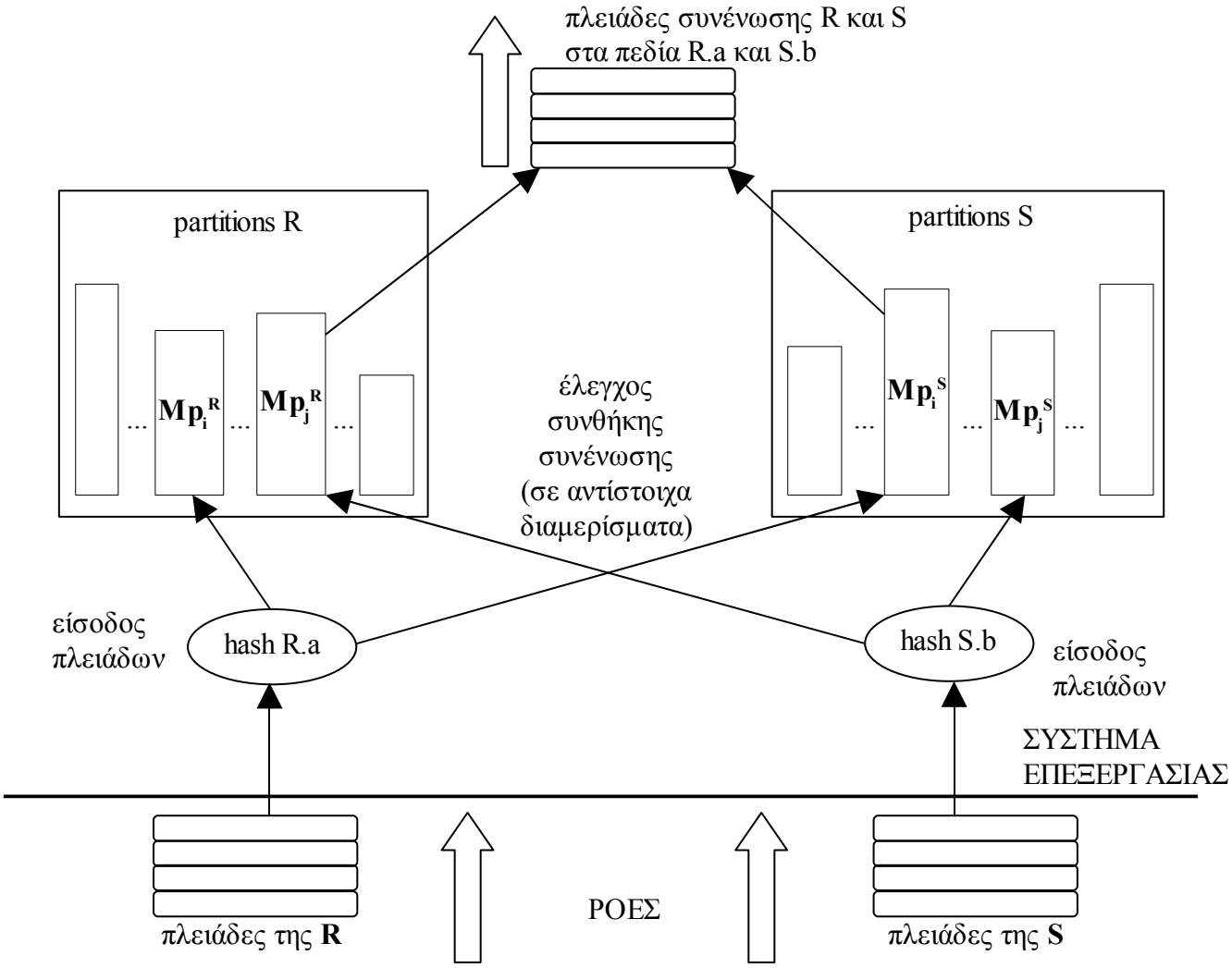
διαμέρισμα θα τοποθετηθεί και μία για να προσδιοριστεί σε ποια ακριβώς θέση μέσα στο συγκεκριμένο διαμέρισμα.

Κάθε διαμέρισμα (partition) είναι με τη σειρά του διασπασμένο σε δύο μέρη. Το ένα του μέρος βρίσκεται αποθηκευμένο στη μνήμη (memory resident) και το άλλο του κομμάτι στο δίσκο (disk resident). Επομένως, έχουμε πολλά διαμερίσματα που το καθένα βρίσκεται εν μέρει στο δίσκο και εν μέρει στη μνήμη. Το τμήμα του διαμερίσματος που βρίσκεται στη μνήμη θα περιέχει τις άρτι αφιχθείσες πλειάδες, ενώ το άλλο τις παλαιότερες. Η διαθέσιμη μνήμη - κατά τα άλλα – μοιράζεται ισότιμα ανάμεσα στις ροές της συνένωσης. Συμβολίζουμε το τμήμα του διαμερίσματος $\mathbf{p_i^R}$ που βρίσκεται στη μνήμη μιας σχέσης R σαν $\mathbf{Mp_i^R}$ και το τμήμα του που βρίσκεται στο δίσκο σαν $\mathbf{Dp_i^R}$. Πάντα θα ισχύει ότι $\mathbf{p_I^R = Mp_iR \cup Dp_iR}$. Στη συνέχεια, ο αλγόριθμος διαθέτει τις παρακάτω τρεις εναλλασσόμενες φάσεις.

Φάση 1

Καθώς οι εισερχόμενες ροές καταφθάνουν στο σύστημα, ο XJoin συμπεριφέρεται περίπου όπως ο SHJ. Δηλαδή, περνά κάθε πλειάδα κάθε ροής από συνάρτηση σύνοψης και την τοποθετεί στο αντίστοιχο διαμέρισμα (partition) που της αναλογεί, στο κομμάτι του διαμερίσματος που βρίσκεται στη μνήμη. Ακολούθως, κάνει έλεγχο για ισότητα στο πεδίο της συνένωσης με τις πλειάδες της δεύτερης ροής, οι οποίες βρίσκονται επίσης στη μνήμη για να παράγει πλειάδες αποτελέσματος της συνένωσης. Ο έλεγχος αυτός γίνεται στα διαμερίσματα που αντιστοιχούν μόνο και όχι σε όλα, εφόσον χρησιμοποιείται η ίδια συνάρτηση σύνοψης. Η διαφοροποίηση από τον SHJ έρχεται όταν η διαθέσιμη μνήμη μοιραία εξαντληθεί – ή έστω πέσει κάτω από ένα κατώφλι. Τότε, οι πλειάδες του διαμερίσματος (partition) που καταλαμβάνει το μεγαλύτερο ποσοστό μνήμης καταγράφονται στο δίσκο, έτσι ώστε να μη χαθούν αλλά και να ελευθερωθεί ένα μέρος της μνήμης. Το διαμέρισμα που καταγράφηκε στο δίσκο έχει πλέον μηδενικό μέγεθος.

Επισημαίνεται ότι ο αλγόριθμος δεν απασχολείται με το μέγεθος που θα αποκτήσει το τμήμα του κάθε διαμερίσματος που βρίσκεται στη μνήμη καθώς οι πλειάδες καταφθάνουν. Έτσι κι αλλιώς το μεγαλύτερο διαμέρισμα θα είναι και αυτό που θα περάσει στο δίσκο όταν η μνήμη εξαντληθεί. Αυτή η διαδικασία, στη συνέχεια, επαναλαμβάνεται οποτεδήποτε η μνήμη το απαιτεί. Η "Φάση 1" αναπαρίσταται στο Σχήμα 2.2.



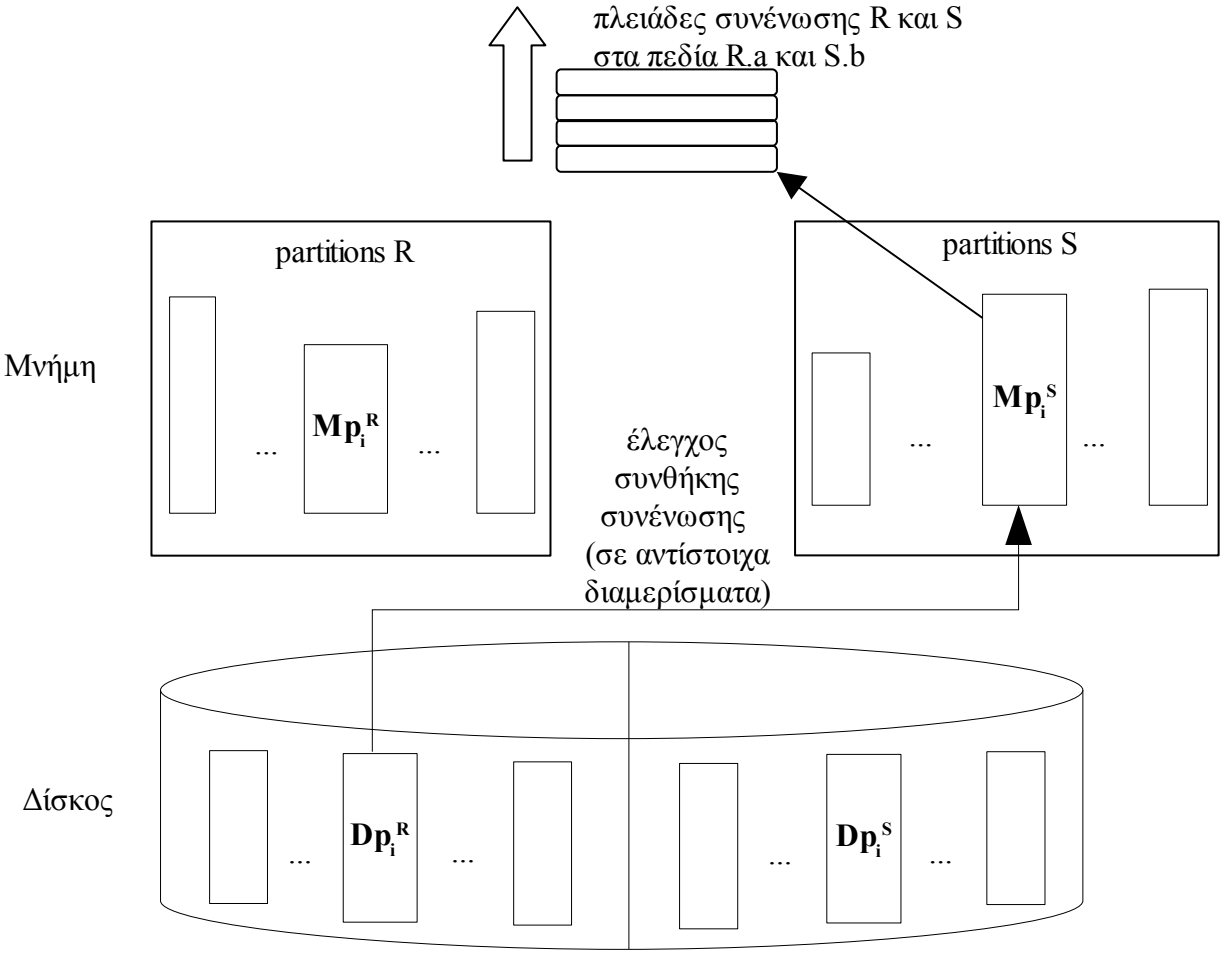
Σχήμα 2.2 - Αλγόριθμος XJoin, Φάση 1

Φάση 2

Στην περίπτωση που οι αφίξεις μίας εκ των ροών διακοπούν προσωρινά, για παράδειγμα λόγω καθυστερήσεων του δικτύου, τότε από τη μεριά της άλλης ροής ένα διαμέρισμα που βρίσκεται στο δίσκο ανασύρεται και οι πλειάδες του ελέγχονται για ισότητα στο πεδίο της συνένωσης με τις πλειάδες της πρώτης ροής που βρίσκονται στη μνήμη. Με αυτό τον τρόπο ο αλγόριθμος περνά από τη "Φάση 1" στη "Φάση 2". Και πάλι οι έλεγχοι για ισότητα γίνονται σε αντίστοιχα διαμερίσματα,

μιας και η συνάρτηση σύνοψης που πρωταρχικά έκανε την ευρετηρίαση είναι η ίδια. Οι όποιες πλειάδες βρεθούν να πληρούν τη συνθήκη της συνένωσης προωθούνται στα αποτελέσματα.
Μετά την επεξεργασία του διαμερίσματος που ανασύρθηκε από το δίσκο το σύστημα ελέγχει αν στην είσοδο υπάρχουν νέες πλειάδες και, αν ναι, τότε επανέρχεται στη "Φάση 1" και αρχίζει εκ νέου τη διαδικασία που περιγράψαμε παραπάνω. Διαφορετικά επιλέγεται ένα άλλο διαμέρισμα από

το δίσκο και η "Φάση 2" επαναλαμβάνεται. Σε αυτό το σημείο χρειάζεται προσοχή έτσι ώστε να μην παραχθούν διπλότυπες πλειάδες αποτελέσματος, οι οποίες πιθανώς να έχουν παραχθεί κατά τη διάρκεια της πρώτης φάσης του αλγορίθμου. Για να επιτευχθεί αυτό γίνεται έλεγχος πάνω στα χρονόσημα (timestamps) των πλειάδων καθώς και στα χρονικά διαστήματα ανάμεσα στα οποία χρειάστηκε να ανασυρθούν διαμερίσματα από το δίσκο. Σε αντίθεση, λοιπόν με την πρώτη φάση η δεύτερη δεν εξαρτάται από τη διαθεσιμότητα των ροών στην είσοδο του συστήματος. Εκμεταλλεύεται το νεκρό χρόνο για να παράγει περισσότερες πλειάδες αποτελέσματος ελέγχοντας πλειάδες καταχωρημένες στο δίσκο με εκείνες που βρίσκονται στη μνήμη ακόμη και αν αυτό προσδίδει στιγμιαία ένα μικρό επιπλέον υπολογιστικό κόστος (overhead) στο σύστημα επεξεργασίας. Η "Φάση 2" αναπαρίσταται στο Σχήμα 2.3.



Σχήμα 2.3 - Αλγόριθμος XJoin, Φάση 2

Φάση 3

Σε αυτή τη φάση βρίσκεται η ολοκλήρωση των φάσεων του αλγορίθμου όταν οι ροές της

συνένωσης φτάσουν στο τέρμα τους, όπου εκτελείται μια διαδικασία τελικής εκκαθάρισης (cleanup). Προσπαθεί να διασφαλίσει ότι όλες οι πλειάδες που τυχόν δεν έχουν ελεγχθεί μέχρι τότε θα ελεγχθούν για να είναι το αποτέλεσμα της συνένωσης πλήρες, μιας και οι πρώτες δύο φάσεις παράγουν μόνο μερικά αποτελέσματα. Η πρώτη φάση αποτυγχάνει στο να βρει πλειάδες αποτελέσματος από πλειάδες των ροών της συνένωσης που δεν βρίσκονταν ταυτόχρονα αποθηκευμένες στη μνήμη. Η δεύτερη φάση αποτυγχάνει, από την άλλη πλευρά, στο να βρει πλειάδες αποτελέσματος από πλειάδες που βρίσκονταν στο δίσκο όταν κάποιες άλλες ανασύρονταν από το δίσκο για να ελεγχθούν με αυτές που βρίσκονταν εκείνη την ώρα στη μνήμη.
Έτσι, η τρίτη φάση συνενώνει όλα τα διαμερίσματα, τόσο αυτά που βρίσκονται στη μνήμη όσο και αυτά που βρίσκονται στο δίσκο μεταξύ τους. Αυτό γίνεται επιλέγοντας τη ροή που είχε το μικρότερο πλήθος πλειάδων και φέρνοντας σταδιακά όλα τα Dp_i αυτής στη μνήμη για να ελεγχθούν για ισότητα με όλα τα Dp_i αλλά και τα Mp_i της άλλης ροής.

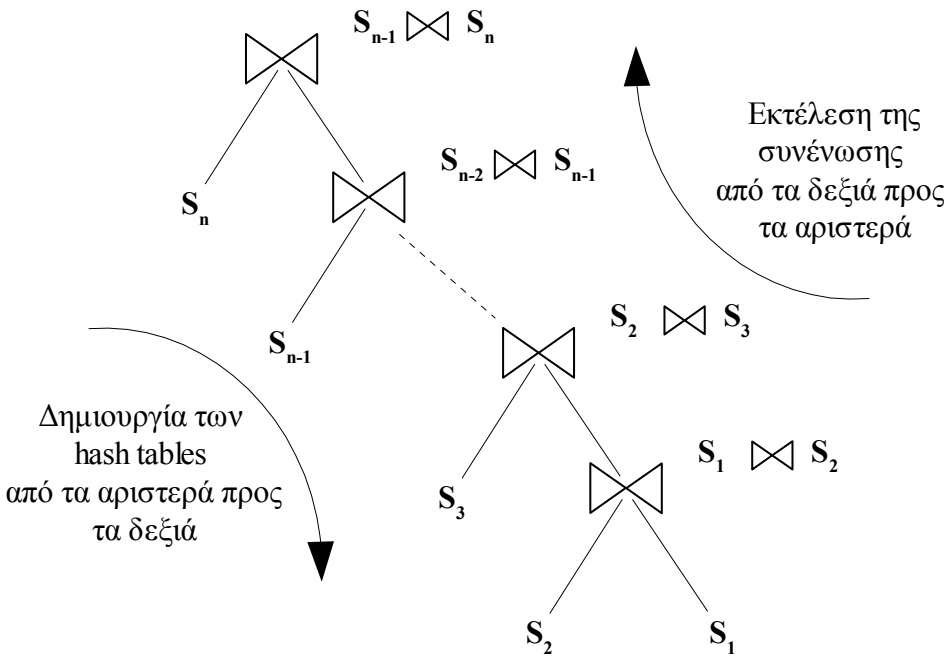
2.5.4 ΠΟΛΛΑΠΛΈΣ ΣΥΝΕΝΏΣΕΙΣ

Μέχρι στιγμής έχουμε αναφερθεί στην εκτέλεση συνεχών επερωτήσεων με απλές επιλογές/προβολές και με συνενώσεις που λαμβάνουν χώρα ανάμεσα σε δύο ροές. Συνενώσεις, όμως, είναι δυνατόν και επιθυμητό να εκφράζονται ανάμεσα σε πολλές ροές ταυτόχρονα, ακριβώς όπως και στις παραδοσιακές βάσεις δεδομένων και το σχεσιακό μοντέλο. Όπως έχει εντοπιστεί και παραπάνω, σημασία στα ΣΔΡΔ δε δίνεται πρώτιστα στην ταχύτατη ολοκλήρωση μιας επερώτησης, κάτι που ενδιαφέρει περισσότερο στις βάσεις δεδομένων [23][27] αλλά στην ικανότητα του συστήματος να ξεκινήσει να παράγει πλειάδες αποτελέσματος όσο το δυνατόν πιο άμεσα κατά την εκκίνηση της επερώτησης και όσο το δυνατόν πιο συνεχόμενα κατά τη διάρκειά της. Ένας από τους πιο σημαντικούς αλγόριθμους που έχουν προταθεί για την εκτέλεση επερωτήσεων με πολλαπλές συνενώσεις είναι ο MJoin [27][28], για τον οποίο θα μιλήσουμε στη συνέχεια.

Παραδοσιακά, οι πολλαπλές συνενώσεις αντιμετωπίζονταν με δυαδικά δέντρα όπου λάμβαναν χώρα διαδοχικές συνενώσεις ανά δύο σχέσεις/ροές. Αυτή η ιδέα αναπαρίσταται στο Σχήμα 2.4 όπου, όπως είναι φανερό, η εκτέλεση της επερώτησης ακολουθεί μια διαδοχή δυαδικών συνενώσεων. Ξεκινώντας από την πρώτες δύο σχέσεις, S_1 και S_2 οι οποίες συνενώνονται μεταξύ τους και το αποτέλεσμα – δηλαδή οι πλειάδες που πληρούν τη συνθήκη της πρώτης συνένωσης – συνενώνεται με την S_3 και το αποτέλεσμα αυτού θα συνενωθεί με τη σειρά του με την επόμενη σχέση έως ότου φτάσουμε μέχρι την τελευταία. Το σύστημα θα δημιουργούσε για κάθε μία σχέση ένα ευρετήριο - για παράδειγμα ένα πίνακα σύνοψης (hash table) – ξεκινώντας από την τελευταία σχέση που θα έπαιρνε μέρος στη συνένωση – αριστερά στο Σχήμα 2.4 – και θα εκτελούσε την

επερώτηση ξεκινώντας από τα δεξιά έχοντας, όλες τις σχέσεις προηγουμένως ευρετηριασμένες. Αν ήταν σε θέση να γνωρίζει την επιλεξιμότητα f_p της κάθε σχέσης στην επερώτηση τότε προηγουμένως θα τις ταξινομούσε κατά φθίνουσα f_p και θα δομούσε το δέντρο έτσι ώστε δεξιότερα να βρίσκονται οι σχέσεις με το μικρότερο f_p, έτσι ώστε να ξεκινήσει τις διαδοχικές συνενώσεις από αυτές, μειώνοντας, με αυτό τον τρόπο, το μέγεθος των ενδιάμεσων αποτελεσμάτων. Ακόμη, πρέπει να παρατηρηθεί ότι η παραγωγή των πλειάδων αποτελέσματος θα μπορεί να ξεκινήσει μόνο αφότου έχουν δημιουργηθεί όλα τα ευρετήρια. Αν, από την άλλη πλευρά, μιλούμε αποκλειστικά για ροές τότε σε ένα τέτοιο αλγόριθμο όταν μία ροή της συνένωσης διακόψει προσωρινά τις αφίξεις των πλειάδων της τότε θα διακοπεί και όλη η διαδικασία του του δέντρου των μερικών συνενώσεων.

Μια τέτοια προσέγγιση είναι ικανοποιητική για δεδομένα που είναι τοπικά διαθέσιμα, όχι όμως στην περίπτωση των ροών όπου οι πλειάδες εμφανίζονται συνεχόμενα και από διαφορετικές πηγές. Αυτό που πρέπει να αντιμετωπιστεί είναι η ανάγκη επεξεργασίας καθολικών πράξεων[6], κάτι που στους αλγορίθμους που αναφέραμε επιτυγχάνεται με την αντικατάσταση της καθολικότητας από πράξεις με ρέουσα μορφή (streaming operators). Το ίδιο ισχύει και για τον MJoin, ο οποίος επεκτείνει την ιδέα των συμμετρικών αλγορίθμων με δυαδική προσέγγιση στην εκτέλεση των πολλαπλών συνενώσεων και του XJoin.



Σχήμα 2.4 – Δυαδικό δέντρο εκτέλεσης συνένωσης

6 Καθολικές πράξεις είναι αυτές που περιέχουν καθολικούς τελεστές, στους οποίους έγινε σύντομη αναφορά στο προηγούμενο κεφάλαιο.

Το πρόβλημα έγκειται στο ότι ο αλγόριθμος θα πρέπει να είναι προετοιμασμένος να δεχθεί νέες πλειάδες για οποιαδήποτε από τις ροές που λαμβάνουν μέρος στη συνένωση και ταυτόχρονα για κάθε νέα πλειάδα μιας ροής να την αντιπαραβάλει με τα ευρετήρια των άλλων ροών για να παράγει αποτελέσματα όσο το δυνατόν συντομότερα και να εξασφαλίσει ότι κάθε πλειάδα αποτελέσματος παράγεται μόνο μια φορά. Αυτές οι απαιτήσεις γίνονται ακόμη περισσότερο επαχθείς για το σύστημα που θα αναλάβει να εκτελέσει την επερώτηση στην περίπτωση που μία ή περισσότερες ροές υπερβούν το χώρο που τους αναλογεί στη μνήμη και χρειαστεί να αποθηκευτούν προσωρινά στο δίσκο. Και βέβαια, μιλώντας, για ροές δεδομένων κάτι τέτοιο οπωσδήποτε θα συμβεί.

Ας θεωρήσουμε τις ροές S_1, S_2, ... S_n οι οποίες λαμβάνουν μέρος σε μία επερώτηση με πολλαπλή συνένωση. Ο αλγόριθμος MJoin ακολουθεί την εξής διαδικασία:

- Δημιουργούνται τόσα ευρετήρια όσες και οι ροές της συνένωσης. Έστω ότι τα ευρετήρια είναι πίνακες σύνοψης, όπως και στους άλλους αλγορίθμους που προαναφέρθηκαν μέχρι τώρα.

- Μόλις μια πλειάδα κάποιας ροής καταφθάνει στο σύστημα επεξεργασίας τοποθετείται στον αντίστοιχο πίνακα σύνοψής της.

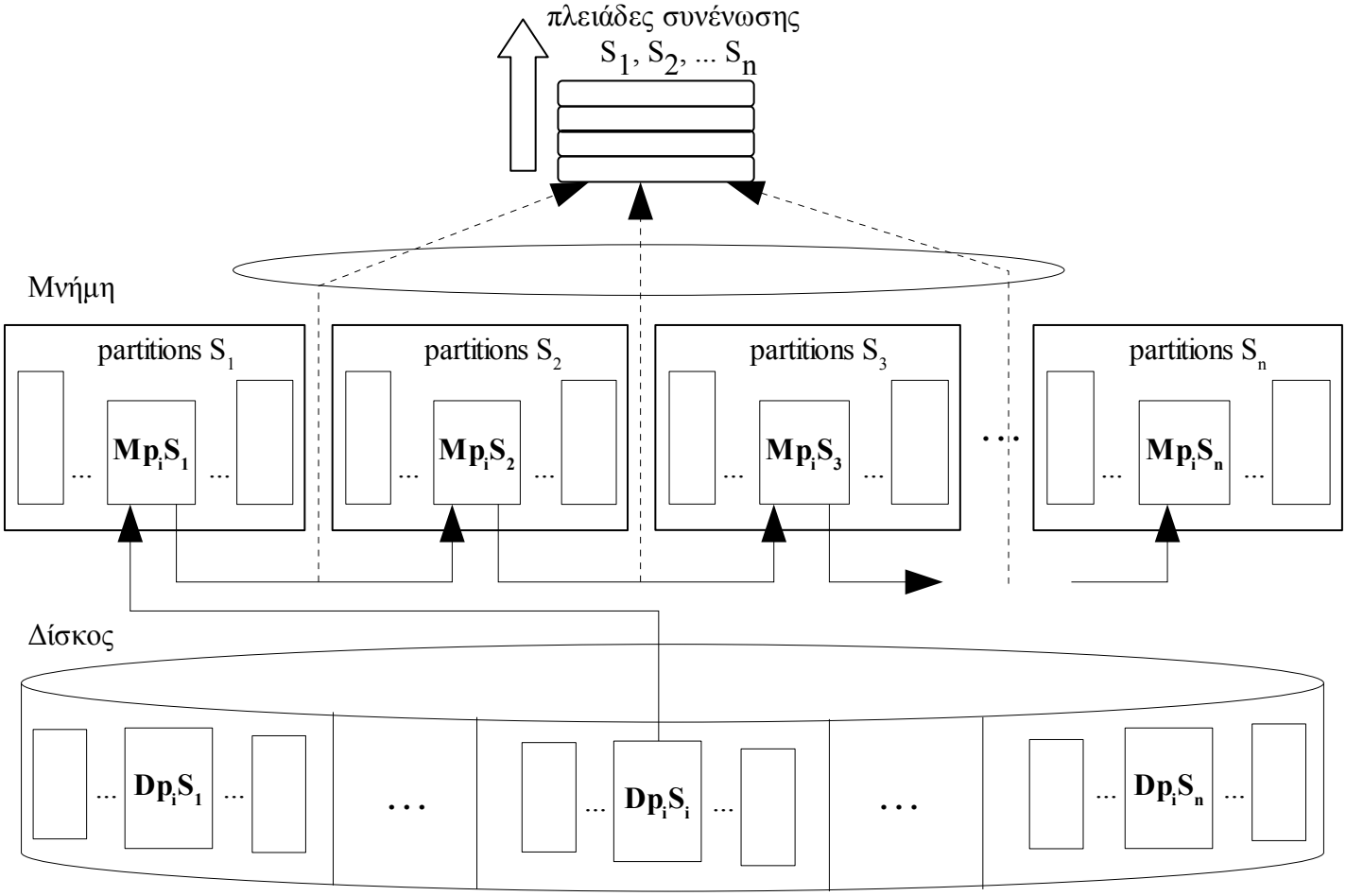
- Κατόπιν, η πλειάδα αντιπαραβάλλεται διαδοχικά με τις τις πλειάδες των άλλων ροών που υπάρχουν ήδη στο σύστημα με τους δικούς τους πίνακες σύνοψης. Με αυτό τον τρόπο θα εντοπιστεί κάθε πιθανή πλειάδα αποτελέσματος που μπορεί να προέλθει από τη συνένωση των τμημάτων των ροών που βρίσκονται αποθηκευμένες ανά πάσα χρονική στιγμή στην κύρια μνήμη. Όπως είναι προφανές σε μια συνένωση, οι δοκιμές διακόπτονται όταν σε έναν πίνακα σύνοψης κάποιας από τις άλλες ροές δεν βρεθεί ούτε μία κοινή πλειάδα αποτελέσματος. Η διαδικασία αυτή φαίνεται στο Σχήμα 2.5 για τις πλειάδες της S_1 και ομοίως ισχύει και για τις υπόλοιπες ροές.

- Η σειρά με την οποία γίνεται η αντιπαραβολή αυτή είναι κατά φθίνοντα παράγοντα επιλεξιμότητας, ο οποίος είναι δυνατόν να μην είναι καν σταθερός για κάθε ροή σε συνάρτηση με το χρόνο. Η σειρά αυτή έχει μεγάλη σημασία σε έναν αλγόριθμο που εκτελεί πολλαπλές συνενώσεις, όπως ο MJoin, μιας και η αποδοτικότητά του μπορεί να αυξηθεί πάρα πολύ όταν η εκτίμηση γίνει σωστά [27].

Εξαιτίας του ότι ο MJoin δεν αποθηκεύει μόνιμα τα ενδιάμεσα αποτελέσματα των επιμέρους

διαδοχικών συνενώσεων είναι πιθανό ορισμένες από τις πλειάδες αποτελέσματος να υπολογιστούν περισσότερες από μία φορές, κάτι που δε θα συνέβαινε σε ένα αλγόριθμο με δυαδικό δέντρο εκτέλεσης σαν αυτό του Σχήματος 2.4. Για παράδειγμα, αν κατά την είσοδο μιας πλειάδας της S_i βρεθούν n κοινές πλειάδες – μερικό αποτέλεσμα – με την S_{i+1} αλλά καμία με την S_{i+2} τότε το πρώτο μερικό αποτέλεσμα θα απορριφθεί. Αν, στη συνέχεια εμφανιστεί μία πλειάδα της S_{i+2} η οποία έχει κοινά στοιχεία με το προαναφερθέν μερικό αποτέλεσμα τότε αυτό θα πρέπει να υπολογιστεί εκ νέου, μιας και δε θα είναι πλέον διαθέσιμο. Στη συνέχεια θα δούμε πώς ο MJoin διαχειρίζεται τις καταστάσεις όπου τα μεγέθη των ροών ξεπερνούν τη διαθέσιμη μνήμη. Η λογική που ακολουθείται είναι όμοια με αυτή του XJoin με τις τρεις διαδοχικές φάσεις: i) καθώς οι πλειάδες καταφθάνουν και υπάρχει διαθέσιμη μνήμη οι νέες πλειάδες περνούν από τη συνάρτηση σύνοψης, εισέρχονται διαμέρισμα (partition) που τους αναλογεί – στο τμήμα του που βρίσκεται στη μνήμη – και αντιπαραβάλλονται με τα αντίστοιχα τμήματα των άλλων ροών, ii) όταν διακοπεί προσωρινά η άφιξη των ροών ο "νεκρός" χρόνος αυτός χρόνος χρησιμοποιείται για την αντιπαραβολή πλειάδων

που έχουν αποθηκευτεί στο δίσκο με πλειάδες που βρίσκονται εκείνη τη χρονική στιγμή στη μνήμη και iii) όταν οι ροές εξαντληθούν γίνεται αντιπαραβολή όλων των πλειάδων που βρίσκονται



Σχήμα 2.6 - Αλγόριθμος MJoin, Φάση 2

αποθηκευμένες στο δίσκο σε μία διαδικασία εκκαθάρισης.

Η πρώτη φάση είναι όμοια και στον MJoin. Στο Σχήμα 2.6 φαίνεται πώς ο MJoin γενικεύει αυτή τη λογική για τη δεύτερη φάση όταν έχουμε πολλαπλές συνενώσεις: επιλέγεται, δηλαδή, από το δίσκο ένα διαμέρισμα μιας ροής και οι πλειάδες του αντιπαραβάλλονται διαδοχικά με τις πλειάδες των αντίστοιχων διαμερισμάτων των υπολοίπων ροών που βρίσκονται στη μνήμη, έτσι ώστε να μη διακοπεί η παραγωγή πλειάδων αποτελέσματος. Ο MJoin γενικεύοντας τη λογική της τρίτης φάσης επιλέγει ένα διαμέρισμα μιας ροής από το δίσκο και το επαναφέρει στη μνήμη χτίζοντάς του τον πίνακα σύνοψης και έπειτα φέρνει στη μνήμη τα αντίστοιχα διαμερίσματα των άλλων ροών για να τα αντιπαραβάλει με το πρώτο.

2.6 ΔΙΑΧΕΊΡΙΣΗ ΠΌΡΩΝ

Παρόλο που η αντιμετώπιση μέσω των τριών φάσεων του XJoin λειτουργεί και στον MJoin, η ύπαρξη πολλαπλών ροών δυσχεραίνει πολύ τη διαχείριση της μνήμης κι επομένως ο ρυθμός παραγωγής πλειάδων μπορεί να επηρεαστεί εξίσου σημαντικά [27]. Η μεγιστοποίηση του ρυθμού παραγωγής αποτελέσματος αποτελεί ένα από τα μεγαλύτερα ζητήματα σε ένα τέτοιο αλγόριθμο και εξαρτάται από:

- το ρυθμό άφιξης των ίδιων των ροών: $\lambda(S_i)$, με i = 1..n (n το πλήθος των ροών)
- τους παράγοντες επιλεξιμότητας για τη συνένωση ανάμεσα στις ροές: f_{ij},[7] με i,j = 1..n και $i \neq j$,
- τον τρόπο με τον οποίο γίνονται οι καταχωρήσεις στο δίσκο όταν η διαθέσιμη μνήμη δεν επαρκεί.

Έχοντας στο μυαλό ότι δύο από τους σημαντικότερους στόχους είναι τόσο η μεγιστοποίηση του ρυθμού όσο και η αδιάλειπτη παραγωγή πλειάδων αποτελέσματος – όταν οι αφίξεις των ροών διακόπτονται προσωρινά – ας θεωρήσουμε μια πολλαπλή συνένωση $S_1 \bowtie S_2 \bowtie \ldots \bowtie S_n$ όπου κάθε ροή S_i καταφθάνει με τον ίδιο ρυθμό λ και ότι όλες οι επιλεξιμότητες είναι σταθερές και ίσες

7 f_{ij} : ο παράγοντας επιλεξιμότητας (selectivity) της επί μέρους συνένωσης ανάμεσα στις ροές S_i και S_j που λαμβάνουν μέρος σε μία πολλαπλή συνένωση

με f. Με την ίδια λογική ας θεωρήσουμε ότι μέχρι τη χρονική στιγμή t έχουν φτάσει στο σύστημα επεξεργασίας W_t πλειάδες από κάθε ροή. Τότε κατά την άφιξη μιας νέας πλειάδας έπειτα από εκείνη τη χρονική στιγμή θα αναμένεται να παραχθούν:

$$\mathbf{(W_t \, x \, f)x(W_t \, x \, f)x \ldots x(W_t \, x \, f) = W_t^{n-1} \, x \, f^{n-1}} \text{ – νέες πλειάδες αποτελέσματος}$$

Αν, τώρα, βρισκόμαστε στην περίπτωση που η διαθέσιμη μνήμη δεν επαρκεί, έστω ότι όλες οι ροές βρίσκονται σε ίσο ποσοστό κατανεμημένες ανάμεσα στο δίσκο και τη μνήμη. Αν το ποσοστό αυτό είναι σ τότε $\boldsymbol{\sigma \, x \, W_t}$ πλειάδες θα βρίσκονται στην κύρια μνήμη τη χρονική στιγμή t για κάθε ροή και έτσι κατά την άφιξη μιας νέας πλειάδας αμέσως μετά θα αναμένεται να παραχθούν:

$$\mathbf{(\sigma \, x \, W_t \, x \, f)x(\sigma \, x \, W_t \, x \, f)x \ldots x(\sigma \, x \, W_t \, x \, f) = \sigma^{n-1} \, x \, W_t^{n-1} \, x \, f^{n-1}} \text{ – νέες πλειάδες αποτελέσματος}$$

Η παραπάνω θεώρηση κατανέμει τις ροές στο δίσκο με τυχαίο και ομοιόμορφο τρόπο χωρίς να ασχολείται με τις τιμές των πεδίων πάνω στα οποία λαμβάνει χώρα η συνένωση. Ας εξετάσουμε, στη συνέχεια, πώς θα μπορούσε να ωφεληθεί ο ρυθμός παραγωγής αποτελεσμάτων από τον τρόπο με τον οποίο γίνονται οι καταχωρήσεις στο δίσκο στον αλγόριθμο MJoin [27]. Θεωρούμε μια συνένωση αστέρα (star join) $\mathbf{S_1} \bowtie_{\mathbf{S1.A1=S2.A2}} \mathbf{S_2} \bowtie_{\mathbf{S2.A2=S3.A3}} \ldots \bowtie_{\mathbf{Sn\text{-}1.An\text{-}1=Sn.An}} \mathbf{S_n}$. Σε μια συνένωση αστέρα απαιτείται διαδοχική ισότητα πάνω στο ίδιο πεδίο όλων των εμπλεκομένων ροών/σχέσεων. Δηλαδή, στο παράδειγμά μας, ισότητα $S_1A_1 = S_2A_2 = S_3A_3 = \ldots = S_nA_n$.

2.6.1 ΔΙΑΧΕΊΡΙΣΗ ΜΝΉΜΗΣ

Αυτή η συνένωση θα μπορούσε να υλοποιηθεί με τον αλγόριθμο MJoin όπως έχουμε αναφέρει. Μια ενδιαφέρουσα τροποποίηση του αλγορίθμου θα ήταν η εξής: Όταν θα έρθει η στιγμή να να αποθηκευθεί ένα μέρος μιας ροής S_i στο δίσκο τότε αντί αυτό να γίνει απλά και με τυχαίο τρόπο μπορεί να επιλεγεί η αποθήκευση στο δίσκο να γίνεται από μια συγκεκριμένη ομάδα διαμερισμάτων (partitions). Δηλαδή, αν γίνει αποθήκευση στο δίσκο του k-οστού διαμερίσματος της ροής S_i τότε να γίνει επίσης αποθήκευση στο δίσκο του k-οστού διαμερίσματος και των υπολοίπων ροών που συμμετέχουν στη συνένωση. Αν η επιλογή του διαμερίσματος που θα καταχωρηθεί στο δίσκο γίνει με βάση το ότι οι πλειάδες που θα αντιστοιχιστούν σε αυτό έχουν χαμηλή πιθανότητα συμμετοχής στο αποτέλεσμα τότε το κέρδος από αυτή την κίνηση είναι προφανές.

Με αυτή την τεχνική (coordinated flushing), όταν μία νέα πλειάδα καταφθάσει από οποιαδήποτε

ροή είτε καταχωρείται σε κάποιο διαμέρισμα που βρίσκεται εξολοκλήρου στη μνήμη και αντιπαραβάλλεται άμεσα με όλα τα αντίστοιχα διαμερίσματα για να παραχθούν πλειάδες αποτελέσματος, είτε καταχωρείται απευθείας σε ένα διαμέρισμα που βρίσκεται εξολοκλήρου στο δίσκο. Ακολουθώντας αυτή τη λογική και μετά την πάροδο κάποιου χρονικού διαστήματος όλα τα διαμερίσματα θα βρίσκονται το καθένα εξολοκλήρου είτε στο δίσκο είτε στη μνήμη. Με αυτό τον τρόπο οι αναμενόμενες πλειάδες αποτελέσματος κατά την άφιξη μίας νέας πλειάδας γίνονται:

- **0** με πιθανότητα **1-σ**, αν η νέα πλειάδα πέφτει σε διαμέρισμα αποθηκευμένο στο δίσκο ή

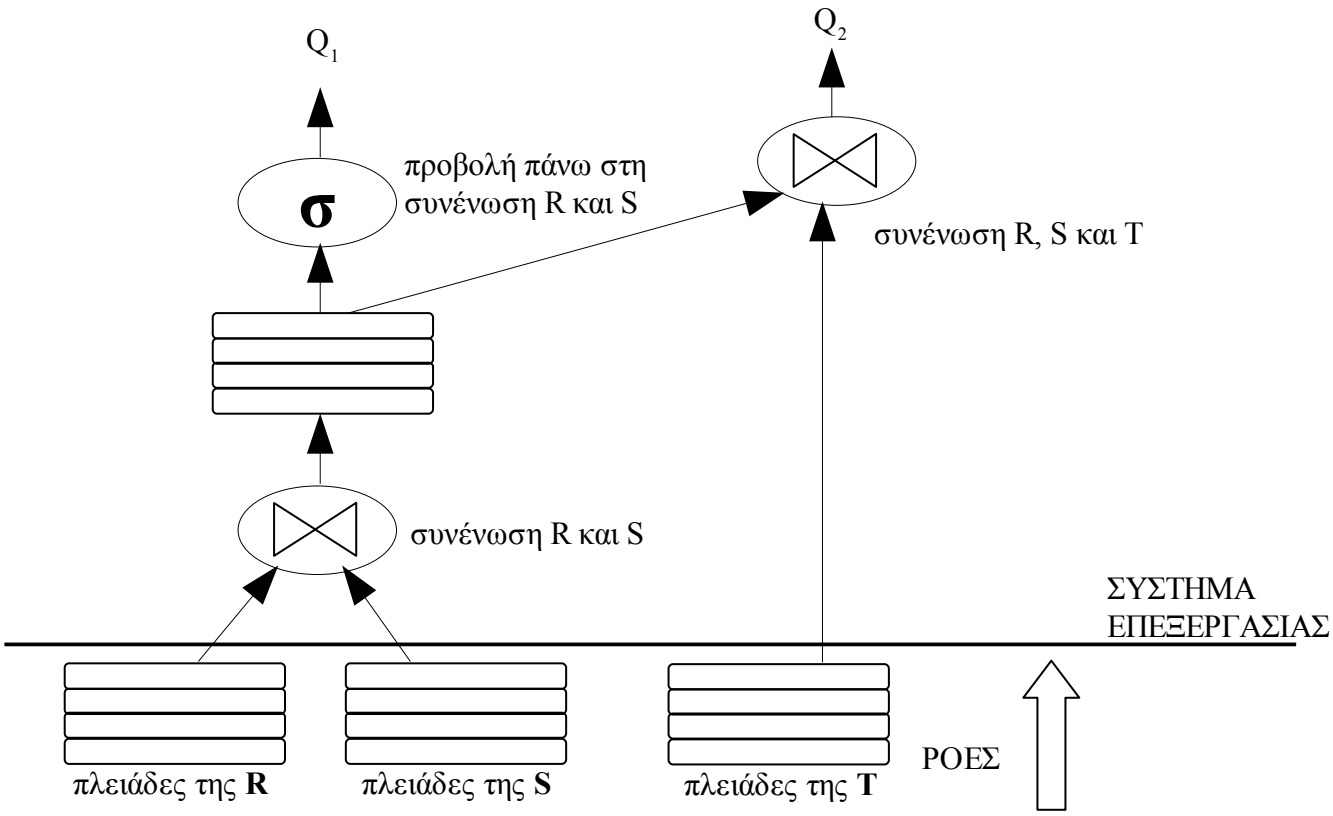
- $\mathbf{W_t^{n-1} \times f^{n-1}}$ με πιθανότητα **σ**, αν η νέα πλειάδα πέφτει σε διαμέρισμα που βρίσκεται στη μνήμη

Αν, δηλαδή, στο σύστημα μέχρι κάποια χρονική στιγμή υπάρχουν W_t πλειάδες κάποιας ροής και κατά την άφιξη μιας νέας πλειάδας γίνεται έλεγχος μέσα στη μνήμη – και όχι στο δίσκο – και αντιπαραβάλλουμε τη νέα πλειάδα με οποιοδήποτε από τα διαμερίσματα που είναι αποθηκευμένα στο δίσκο τότε δε θα παραχθούν καθόλου πλειάδες αποτελέσματος, εφόσον προέρχονται από διαφορετικά διαμερίσματα. Έτσι, οι πλειάδες αυτές που βρίσκονται στο δίσκο δε θα επηρεάσουν καθόλου το ρυθμό παραγωγής του αποτελέσματος, ενώ, παράλληλα, θα έχουν ελαφρύνει την κύρια μνήμη με την απουσία τους. Επομένως, ο αναμενόμενος αριθμός πλειάδων αποτελέσματος κατά την άφιξη μιας νέας πλειάδας θα γίνει $\mathbf{\sigma \times W_t^{k-1} \times f^{k-1}}$.

Η αποδοτική διαχείριση της μνήμης είναι ένα από τα σημαντικότερα προβλήματα που ένα Σύστημα Διαχείρισης Ροών Δεδομένων καλείται να αντιμετωπίσει – παράλληλα με τον επεξεργαστικό φόρτο. Έχουν αναφερθεί πολλές τεχνικές οι οποίες έχουν δοκιμαστεί και πειραματικά [23]. Οι περισσότερες αναφέρονται στη μελέτη μιας και μόνο επερώτησης – με συνενώσεις ή χωρίς – χωρίς να μπορούν να επεκτείνουν τα συμπεράσματά τους έτσι ώστε να είναι άμεσα χρήσιμα σε ένα πλήρες ΣΔΡΔ, όπου πολλές επερωτήσεις που πιθανώς χρησιμοποιούν κοινούς τελεστές εκτελούνται ταυτόχρονα. Οι απαιτήσεις σε ένα πραγματικό σύστημα που δέχεται τεράστιους και απρόβλεπτους όγκους ροών δεδομένων είναι ιδιαίτερα υψηλές και η μνήμη ενός συστήματος είναι σε κάθε περίπτωση πεπερασμένη. Μερικές από τις τεχνικές που προτείνονται ακολουθούν παρακάτω.

2.6.2 ΔΙΑΜΟΙΡΑΣΜΌΣ ΠΌΡΩΝ

Μια άλλη τακτική που θα εξυπηρετούσε στην προσπάθεια για εξοικονόμηση πόρων θα ήταν ο διαμοιρασμός πόρων ανάμεσα στις επερωτήσεις. Ας δούμε το εξής παράδειγμα [29]: Έστω ότι έχουμε δύο επερωτήσεις Q_1 και Q_2. Η Q_1 αποτελείται μια προβολή πάνω στη συνένωση δύο ροών R



Σχήμα 2.7

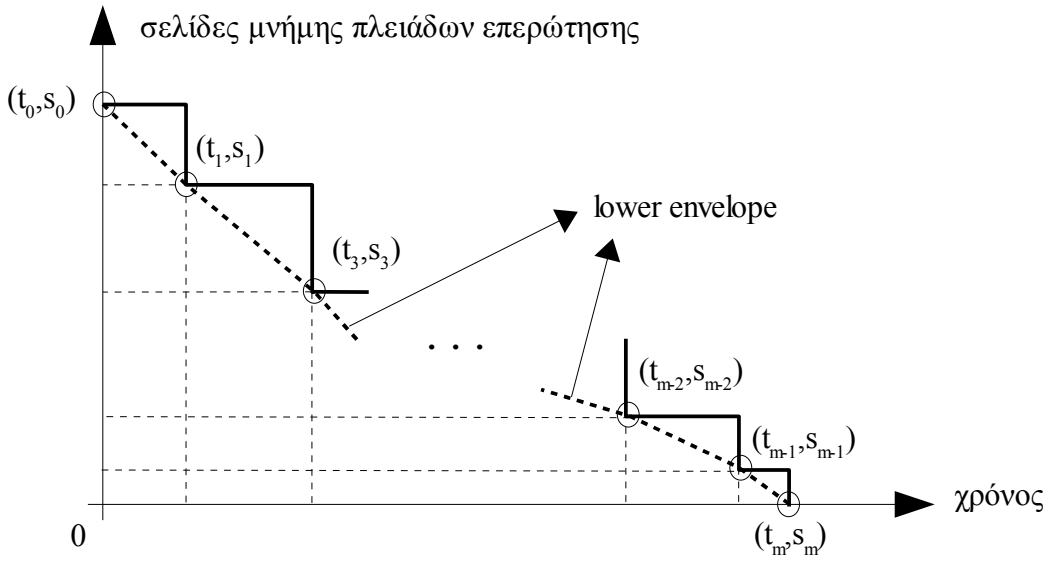
και S και η Q_2 αποτελείται από τη συνένωση τριών ροών R, S και T. Είναι προφανές ότι η συνένωση ανάμεσα στις R και S είναι κοινή και στις δύο επερωτήσεις και το σύστημα επεξεργασίας θα όφειλε να εκμεταλλευτεί αυτή την ιδιότητα με κάποιο τρόπο. Στο Σχήμα 2.7 φαίνονται οι δύο επερωτήσεις και το πλάνο εκτέλεσής τους. Αυτή η τακτική είναι αντίστοιχη με πάγιες τακτικές που ακολουθούνται στα παραδοσιακά ΣΔΒΔ και αφορούν τη βελτιστοποίηση πολλαπλών επερωτήσεων μέσω του διαμοιρασμού κοινών υπό-τμημάτων τους.

Η επιλογή του αν θα χρησιμοποιηθεί αυτή η δυνατότητα εξαρτάται από παράγοντες όπως ο ρυθμός άφιξης των ροών και η επιλεξιμότητα των ενδιάμεσων τελεστών. Ας θεωρήσουμε για παράδειγμα μια συνένωση J της οποίας η επιλεξιμότητα είναι ιδιαίτερα χαμηλή. Έστω, ακόμη δύο τελεστές P_1 και P_2 που ακολουθούν τον J σε κάποιες επερωτήσεις, όπου ο πρώτος έχει μεγάλη επιλεξιμότητα και ο δεύτερος χαμηλή. Αν ο ίδιος τελεστής J τοποθετείται τόσο πριν από τον P_1 όσο και πριν από τον P_2 τότε είναι φανερό ότι δεν ικανοποιούνται ισότιμα οι ανάγκες και των δύο. Επομένως, είναι

δυνατόν, παρόλο που δύο επερωτήσεις μοιράζονται τον ίδιο τελεστή πάνω στα ίδια πεδία, να μην είναι αποδοτικό κατά το σχεδιασμό της επερώτησης να αποτυπωθεί στην εκτέλεση αυτό το κοινό στοιχείο.

2.6.3 ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ (SCHEDULING)

Μία ακόμη πτυχή του διαμοιρασμού των πόρων ανάμεσα στις επερωτήσεις και τους τελεστές τους είναι η εξής [17]: Στις περιόδους όπου οι ρυθμοί άφιξης των ροών γίνονται ιδιαίτερα υψηλοί κι ακόμη και όταν η μνήμη είναι ικανή να αντεπεξέλθει στις απαιτήσεις είναι δυνατόν, εξαιτίας του μεγάλου υπολογιστικού φόρτου πολλές από τις εισερχόμενες πλειάδες να περιμένουν για σημαντικό χρόνο στην είσοδο του συστήματος έως ότου προωθηθούν για επεξεργασία. Για να



Σχήμα 2.8 – Progress Chart

αντιμετωπιστεί αυτό το γεγονός δημιουργούνται ουρές επεξεργασίας (queues) ανάμεσα στους τελεστές. Έτσι, οι πλειάδες, αφότου εισέλθουν στο σύστημα, μεταφέρονται από την ουρά επεξεργασίας του ενός τελεστή σε αυτή του επόμενου. Το πρόβλημα εντοπίζεται τότε στον αποδοτικό προγραμματισμό των τελεστών και τη μεταγωγή των πλειάδων ανάμεσά τους με τρόπο ώστε να ελαχιστοποιούνται οι ουρές αυτές. Όσο μικρότερες γίνονται οι ουρές τόσο πιο άμεσα θα γίνεται η πρόοδος στην επεξεργασία των επερωτήσεων.

Θεωρούμε, λοιπόν, ότι οι πλειάδες μετά την άφιξή τους στο σύστημα κατά κάποιον τρόπο δρομολογούνται ανάμεσα στους τελεστές ακολουθώντας, έτσι, μια διαδρομή. Οι ανάγκες σε μνήμη αυτής της προσέγγισης μπορούν να αποτυπωθούν σε ένα διάγραμμα προόδου (progress chart) [6]. Ένα τέτοιο διάγραμμα φαίνεται στο Σχήμα 2.8. Στον άξονα x βρίσκεται ο χρόνος. Στον άξονα y

βρίσκεται το μέγεθος της μνήμης που έχει δεσμευθεί από μία επερώτηση για να αποθηκεύσει τις πλειάδες τις σε ουρές μετρημένο σε σελίδες[8] (pages). Οι κύκλοι στο διάγραμμα αυτό είναι οι τελεστές που απαρτίζουν την επερώτηση. Αν αυτοί είναι m στο πλήθος τότε τότε θα υπάρχουν m+1 κύκλοι στο διάγραμμα. Σύμφωνα με το διάγραμμα ο i-οστός τελεστής απαιτεί t_i-t_{i-1} = μονάδες χρόνου για να επεξεργαστεί s_{i-1} σελίδες πλειάδων και παράγει s_i σελίδες πλειάδων. Ο παράγοντας επιλεξιμότητας του τελεστή θα είναι s_i/s_{i-1}. Στο τέλος όλες οι σελίδες θα έχουν υποστεί επεξεργασία οπότε το s_m θα είναι πάντοτε μηδέν.

Η κάθε σελίδα μνήμης που περιέχει πλειάδες που παίρνουν μέρος στην επερώτηση διατρέχει την έντονη γραμμή του διαγράμματος ανάμεσα στους τελεστές (κύκλοι). Όταν πλειάδες πλήθους που να γεμίζουν μία σελίδα μνήμης εισέρχονται προς επεξεργασία κατά τη χρονική στιγμή t_0 έχουν μέγεθος s_0. Διατρέχοντας τους τελεστές και λόγω της επιλεξιμότητας του κάθε ενός το μέγεθος αυτό ελαττώνεται έως ότου μηδενιστεί – μετά τον τελευταίο τελεστή –, μιας και τότε καμία πλειάδα δεν είναι πλέον απαραίτητη για την επερώτηση. Για να ελαχιστοποιεί τις ανάγκες σε μνήμη το πλάνο εκτέλεσης της επερώτησης προσπαθεί να διατηρεί τη δεσμευμένη από την επερώτηση μνήμη στα όρια της νοητής τεθλασμένης γραμμής που ενώνει στο διάγραμμα τους τελεστές μεταξύ τους (η έντονη διακεκομμένη γραμμή στο σχήμα). Ο λόγος είναι το ότι τελεστές που κινούνται κατά μήκος αυτής της γραμμής ελαττώνουν ταχύτερα τη δεσμευμένη μνήμη. Η γραμμή αυτή ονομάζεται κάτω κάλυμμα (lower envelope) του διαγράμματος.

Έτσι αναδύεται η ιδέα του αλυσιδωτού προγραμματισμού (chain scheduling strategy) [6]. Σύμφωνα με αυτόν επιλέγεται για επεξεργασία η σελίδα πλειάδων εκείνη που βρίσκεται στο πιο απότομο σημείο του κάτω καλύμματος του διαγράμματος προόδου. Ο αλγόριθμος αυτός αποδεικνύεται ότι είναι βέλτιστος στην περίπτωση όπου οι επιλεξιμότητες των τελεστών είναι γνωστές αλλά ακόμη και όταν δεν είναι γνωστές είναι δυνατόν να υπολογίζονται περιοδικά και αναλόγως να προσαρμόζεται το πλάνο της επεξεργασίας.

2.6.4 ΠΡΟΣΕΓΓΙΣΕΙΣ (APPROXIMATION)

Είναι κατανοητό πως η διαχείριση πολλών απεριόριστων ροών που καταφθάνουν με τυχαίους ρυθμούς, η επεξεργασία ταυτόχρονων, συνδυαστικών και σύγχρονων (on-line) επερωτήσεων πάνω σε αυτές και το πεπερασμένο των πόρων του συστήματος επεξεργασίας θα οδηγήσουν μοιραία ένα ΣΔΡΔ στην κατάσταση όπου δε θα μπορεί να αποδώσει τα επιθυμητά αποτελέσματα οσοδήποτε δοκιμασμένοι και αν είναι οι αλγόριθμοι που χρησιμοποιεί. Αυτό που μπορεί να κάνει, όμως, για να

8 Η σελίδα (page) είναι το ελάχιστο δομικό στοιχείο της μνήμης. Η όποια εγγραφή ή ανάγνωση από τη μνήμη μπορεί να γίνει σε πολλαπλάσια του μεγέθους της.

αντιμετωπίσει τέτοιες καταστάσεις είναι – αποδεχόμενο την αδυναμία του να ανταποκριθεί πλήρως στις απαιτήσεις – να προσπαθήσει να παράγει αποτελέσματα όσο το δυνατόν πιο κοντά στην πραγματικότητα [14][6]. Να δώσει, δηλαδή, όσο το δυνατόν καλύτερες προσεγγίσεις (approximations) στις απαντήσεις του.

Η προσπάθεια για προσεγγίσεις μπορεί να είναι στατική [29]. Αυτό σημαίνει ότι το πλάνο εκτέλεσης των επερωτήσεων λαμβάνει εκ των προτέρων υπόψη του το γεγονός ότι οι πόροι είναι δυσεύρετοι και είναι, με αυτό τον τρόπο προετοιμασμένο κατά την εκτέλεσή τους. Υιοθετώντας στατικές προσεγγίσεις η συμπεριφορά των επερωτήσεων μπορεί να είναι κατά το δυνατόν προβλέψιμη αν και δε λαμβάνονται υπόψη πιθανές ευριστικές ιδιότητες των ροών. Ενώ, δηλαδή, η εκτέλεση μιας επερώτησης δε θα χρειαστεί να ανασυνταχθεί κατά τη διάρκεια της εκτέλεσής της για να εφαρμόσει την τεχνική της προσέγγισης (στατικότητα) δε θα μπορεί και να εκμεταλλευτεί ιδιαιτερότητες κάποιας ροής, όπως, για παράδειγμα, το ότι εξαιτίας κάποιου συμβάντος τιμές συγκεκριμένων πεδίων παραμένουν σταθερές. Κάτι τέτοιο, φυσικά, εξαρτάται από τη σημασιολογία των εκάστοτε ροών.

Ένας τρόπος να εφαρμοστούν στατικές προσεγγίσεις είναι με τη συρρίκνωση του πλαισίου (window) πάνω στο οποίο γίνεται μια επερώτηση. Με τη μείωση, δηλαδή, του πλήθους των πιο πρόσφατων πλειάδων κάποιων ροών. Αυτό σημαίνει ότι στις επερωτήσεις όπου έχει προσδιοριστεί κάποιο τέτοιο πλαίσιο τότε αυτό συρρικνώνεται, ενώ σε επερωτήσεις όπου δεν έχει προσδιοριστεί τότε είναι δυνατόν εξαιτίας των συνθηκών να εφαρμοστεί.

Ένας δεύτερος τρόπος να εφαρμοστούν στατικές προσεγγίσεις είναι με δειγματοληψία των εισερχόμενων πλειάδων. Καθώς, δηλαδή, οι πλειάδες καταφθάνουν δε θα εισέρχονται όλες στο σύστημα επεξεργασίας παρά μόνο κάποια χαρακτηριστικά δείγματά τους. Η δειγματοληψία αυτή μπορεί να γίνει είτε αγνωστικά, επιλέγοντας δηλαδή από κάθε W πλειάδες μόνο τις $W_a < W$, πχ επιλέγοντας 80 στις 100, είτε λαμβάνοντας υπόψη σημασιολογικές ιδιαιτερότητες της ροής. Η ποιότητα του αποτελέσματος των επερωτήσεων εξαρτάται τότε από τη συχνότητα της δειγματοληψίας και την κατανομή με την οποία γίνεται.

Επιπλέον, οι προσεγγίσεις μπορούν να εφαρμοστούν και με περισσότερο δυναμικούς τρόπους [29]. Το πλάνο εκτέλεσης των επερωτήσεων παραμένει σταθερό, όμως, ανάλογα με τις ανάγκες για εξοικονόμηση πόρων το αποτέλεσμά τους γίνεται περισσότερο ή λιγότερο προσεγγιστικό. Το πλεονέκτημα είναι ότι ακριβώς εξαρτώμενο εκάστοτε συνθήκες το σύστημα τείνει να προσαρμόζεται αναλόγως. Σε περιόδους όπου ο φόρτος είναι μικρός τα αποτελέσματα των

επερωτήσεων θα είναι απόλυτα ακριβή, ενώ μόνο όταν ο φόρτος αυξάνεται σε επικίνδυνο για τη διαθεσιμότητα του συστήματος βαθμό θα υποβαθμίζεται η ποιότητα.

Και σε αυτή την περίπτωση οι τρόποι να υλοποιηθεί η προσέγγιση είναι με τη συρρίκνωση του πλαισίου (window) των επερωτήσεων και με μεθόδους δειγματοληψίας πάνω στις εισερχόμενες πλειάδες. Σε κάθε περίπτωση οι δυναμικά εφαρμοζόμενες προσεγγίσεις δίνουν μεγαλύτερη και προσαρμοστικότητα στο σύστημα και κάνουν την ισορροπία (trade-off) ανάμεσα σε απόδοση και ποιότητα περισσότερο ευέλικτη. Φυσικά, πάντοτε ο δέκτης του αποτελέσματος των επερωτήσεων πρέπει να είναι ενήμερος της όποιας τεχνικής χρησιμοποιείται που μειώνει την ακρίβεια και την ποιότητά του.

3. ΒΕΛΤΙΣΤΟΠΟΊΗΣΗ ΚΑΙ ΤΕΧΝΙΚΈΣ

Στα παραδοσιακά συστήματα διαχείρισης βάσεων δεδομένων η προσέγγιση στην εκτέλεση των επερωτήσεων ακολουθούσε τη διαδικασία πρώτα του σχεδιασμού της επερώτησης κι έπειτα της εκτέλεσής της. Ένα ειδικό τμήμα του συστήματος αναλάμβανε τη βελτιστοποίηση του πλάνου της επερώτησης και καθόριζε το πώς αυτό θα εκτελεστεί με τον καλύτερο δυνατό τρόπο. Το πλάνο αυτό δεν είχε λόγο να αλλάξει κατά τη διάρκεια της εκτέλεσης της επερώτησης παρά μόνο – αφότου η επερώτηση είχε ολοκληρωθεί και εκτελεστεί πολλές φορές – αν εντοπιζόταν κάποια ιδιότητα που να το απαιτούσε, συνήθως στατιστικής φύσης, όπως το ότι η επιλεξιμότητα κάποιων τελεστών λαμβάνει χαρακτηριστικά υψηλές ή χαμηλές τιμές. Οι βελτιστοποιητές, όμως, είναι δυνατόν και πιθανόν να κάνουν λανθασμένες εκτιμήσεις επειδή στηρίζονται είτε σε παρωχημένα στατιστικά είτε σε λάθος εκτιμήσεις [30].

Όταν η συζήτηση μεταφέρεται στις ροές δεδομένων τα παραπάνω προβλήματα γίνονται ακόμη πιο αισθητά και μπορούν να αποτελέσουν αιτία της αποτυχίας ενός καλοσχεδιασμένου κατά τα άλλα συστήματος. Ο λόγος είναι, καταρχάς, ο τεράστιος όγκος των δεδομένων που το σύστημα καλείται να διαχειριστεί ταυτόχρονα. Δεδομένα που, όπως έχουμε παρατηρήσει, καταφθάνουν με απρόσμενους και συχνά υψηλούς ρυθμούς και πάνω στα οποία τίθενται πολλές συνδυαστικές επερωτήσεις. Έτσι το σύστημα επεξεργασίας για να μπορέσει να αντεπεξέλθει υποχρεώνεται να εφαρμόσει μεθοδολογίες που να ελαφρύνουν τον υπολογιστικό φόρτο του χρησιμοποιώντας όσο το δυνατόν λιγότερη μνήμη.

Μια εξαιρετικά σημαντική παράμετρος που αποτελεί και ειδοποιό διαφορά ανάμεσα στις συνεχείς επερωτήσεις και στις παραδοσιακές επερωτήσεις είναι αυτή ακριβώς η συνέχειά τους. Ο όρος "συνεχείς" στις συνεχείς επερωτήσεις δεν αναφέρεται τόσο στο ότι εκτελούνται πάνω σε συνεχή δεδομένα αλλά στο ότι είναι και οι ίδιες συνεχείς. Εκτελούνται, δηλαδή, αδιάλειπτα και για μεγάλα χρονικά διαστήματα πάνω σε ένα σύνολο δεδομένων, απαντώντας σε ερωτήματα τα οποία τίθενται για να ανταποκριθούν σε ανάγκες εξίσου συνεχείς και μακροπρόθεσμες. Είναι αυτονόητο πως ένα σύστημα επεξεργασίας οφείλει να λαμβάνει υπόψη του το γεγονός ότι οι συνθήκες με την πάροδο του χρόνου μεταβάλλονται κι επομένως ο βέλτιστος τρόπος εκτέλεσης της εκάστοτε επερώτησης – αν είναι δυνατόν να υπολογιστεί επακριβώς – δε θα παραμείνει αναλλοίωτος στο χρόνο. Επομένως, το σύστημα οφείλει να αναθεωρεί το πλάνο εκτέλεσης των επερωτήσεων καθώς και την όλη διαχείριση των δεδομένων που δέχεται για να ανταποκρίνεται κατά το δυνατόν αποδοτικά στις αλλαγές του περιβάλλοντός του.

3.1 ΠΡΟΣΑΡΜΟΣΤΙΚΉ ΕΚΤΈΛΕΣΗ ΕΠΕΡΩΤΉΣΕΩΝ

Η προσαρμοστική εκτέλεση των επερωτήσεων (adaptive query processing) μέσα στο χρόνο είναι η αντιμετώπιση που αποσκοπεί στο να εξομαλυνθούν τα προαναφερθέντα προβλήματα και να αποφευχθούν το κόστος και οι παρενέργειες από τα πιθανά λάθη του βελτιστοποιητή των επερωτήσεων. Σύμφωνα με αυτήν ο σχεδιασμός της εκτέλεσης και η ίδια η εκτέλεση των επερωτήσεων είναι δύο παράλληλες και αλληλοσυνδεόμενες διαδικασίες. Η προσαρμοστικότητα κάνει την εκτέλεση περισσότερο συμπαγή και ανθεκτική σε σφάλματα του βελτιστοποιητή, σε αλλαγές των συνθηκών και στην αδυναμία άμεσης γνώσης ικανοποιητικών στατιστικών. Αναλυτικότερα τα προβλήματα που οδηγούν στην ανάγκη για αναπροσαρμογή των επερωτήσεων είναι τα εξής [30][31]:

- **Διόρθωση λάθος εκτιμήσεων του βελτιστοποιητή.**
  Υπάρχουν πολλές περιπτώσεις όπου ο βελτιστοποιητής επερωτήσεων μπορεί να σφάλλει καταρτίζοντας ένα πλάνο εκτέλεσης σημαντικά υποδεέστερο του πραγματικά βέλτιστου. Συνήθως η αιτία είναι η ανυπαρξία στατιστικών για τις συσχετίσεις των πεδίων των σχέσεων/ροών και η λανθασμένη εκτίμηση των κατανομών των πεδίων. Ιδίως στα δεδομένα με μορφή ροών το πρόβλημα είναι εντονότερο μιας και οι επερωτήσεις επενεργούν πάνω στα δεδομένα για μεγάλα χρονικά διαστήματα όπου οι συνθήκες μεταβάλλονται με το χρόνο. Αν, από την άλλη πλευρά, τέτοιες συσχετίσεις και κατανομές όντως υπάρχουν τότε, εξαιτίας της μακροχρόνιας δυνατότητας παρακολούθησης των δεδομένων, παρόμοιες εκτιμήσεις μπορούν να αποβούν εξαιρετικά χρήσιμες. Η διατήρηση, όμως, στατιστικών είναι μια επίπονη για το σύστημα διαδικασία που πρέπει να διατηρείται παράλληλα με τις υπόλοιπες λειτουργίες του. Έτσι, ο βελτιστοποιητής είναι εύκολα επιρρεπής σε απαρχαιωμένες πληροφορίες.

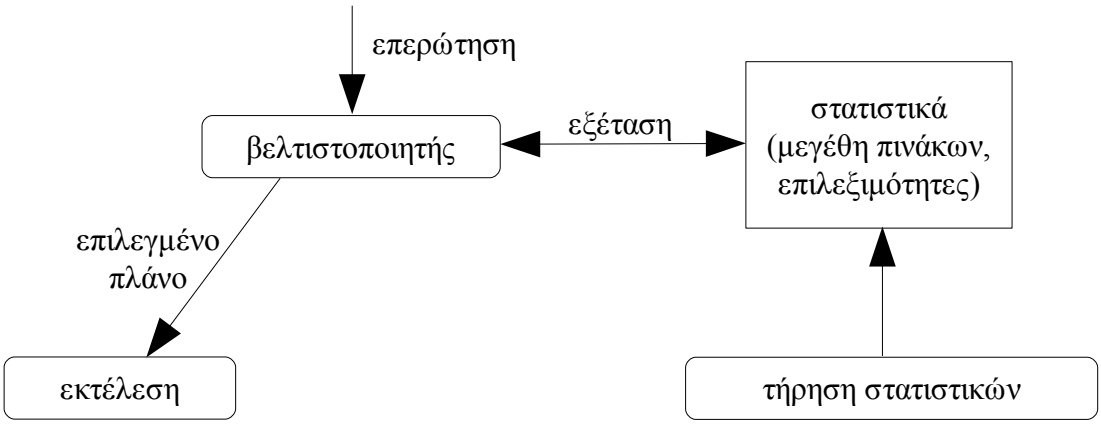
- **Αδυναμία γνώσης στατιστικών**
  Κατανοώντας το εύρος εφαρμογής τόσο των παραδοσιακών βάσεων δεδομένων όσο και των ροών δεδομένων η ποικιλία των επερωτήσεων που μπορούν να τεθούν είναι τόσο μεγάλη που η ίδια η ύπαρξη στατιστικών στοιχείων στην υποβοήθηση της βελτιστοποίησης αποτελεί πολλές φορές πολυτέλεια. Η κλασσική προσέγγιση που θέλει πρώτα τον πλήρη σχεδιασμό του πλάνου εκτέλεσης της επερώτησης κι έπειτα την εκτέλεσή της έχει πολύ λίγη απόδοση σε τέτοιες περιπτώσεις. Η προσαρμοζόμενη εκτέλεση αυτού του είδους των επερωτήσεων είναι η μόνη αντιμετώπιση όταν δεν

υπάρχει πρότερη γνώση για τα δεδομένα που συμμετέχουν σε αυτήν.

- **Μεταβολές στα χαρακτηριστικά της εισόδου και των συνθηκών**
  Όπως προαναφέρθηκε οι συνεχείς επερωτήσεις συχνά εξυπηρετούν σκοπούς μακροχρόνιους και κατά τη διάρκεια της εκτέλεσής τους πολλές από τις αρχικές συνθήκες είναι δυνατόν να αλλάξουν και μάλιστα όχι μόνο μία φορά. Ο ρυθμός άφιξης των πλειάδων μιας ροής είναι μία παράμετρος που μπορεί να αλλάζει ακόμη και αρκετά συχνά, όπως έχει παρατηρηθεί σε πολλές πραγματικές εφαρμογές. Δεν είναι μόνο, όμως, οι εξωγενείς συνθήκες αυτές που μεταβάλλονται. Το ίδιο το σύστημα επεξεργασίας περνά από πολλές φάσεις κατά τη διάρκεια της ζωής του, όπου παράγοντες όπως η διαθέσιμη μνήμη και η επεξεργαστική ισχύς του είναι σημαντικά δεσμευμένες.

3.2 ΜΕΘΟΔΟΛΟΓΙΕΣ ΠΡΟΣΑΡΜΟΣΤΙΚΉΣ ΕΚΤΈΛΕΣΗΣ ΕΠΕΡΩΤΉΣΕΩΝ

Η παραδοσιακή προσέγγιση πλάνου ακολουθούμενου από την εκτέλεση για μια επερώτηση υλοποιείται σε τρία στάδια. Καταρχάς ο βελτιστοποιητής επιλέγει το καλύτερο πλάνο εκτέλεσης που μπορεί να προσδιορίσει σύμφωνα με τα διαθέσιμα δεδομένα του. Στη συνέχεια η επερώτηση εκτελείται σύμφωνα με το πλάνο αυτό. Τέλος, παράλληλα με την εκτέλεση γίνεται και η παρακολούθηση στατιστικών πάνω στα δεδομένα, όπως επιλεξιμότητες τελεστών κλπ, τα οποία θα χρησιμοποιηθούν στην κατάρτιση του πλάνου των μετέπειτα επερωτήσεων. Η λογική αυτή φαίνεται στο Σχήμα 3.1. Η σημαντική διαφοροποίηση που επιτάσσει η προσαρμοστική εκτέλεση είναι ο περιοδικός – ή, καταχρηστικά, "συνεχής" - επαναπροσδιορισμός του βέλτιστου πλάνου εκτέλεσης και η εφαρμογή του. Επομένως, τα τρία παραπάνω στάδια διατηρούνται αλλά εκτελούνται παράλληλα και εναλλασσόμενα. Οι κυρίαρχες προσεγγίσεις στην προσαρμοστική εκτέλεση επερωτήσεων είναι οι εξής [30]: η βασισμένη στο πλάνο (plan-based), η βασισμένη στη

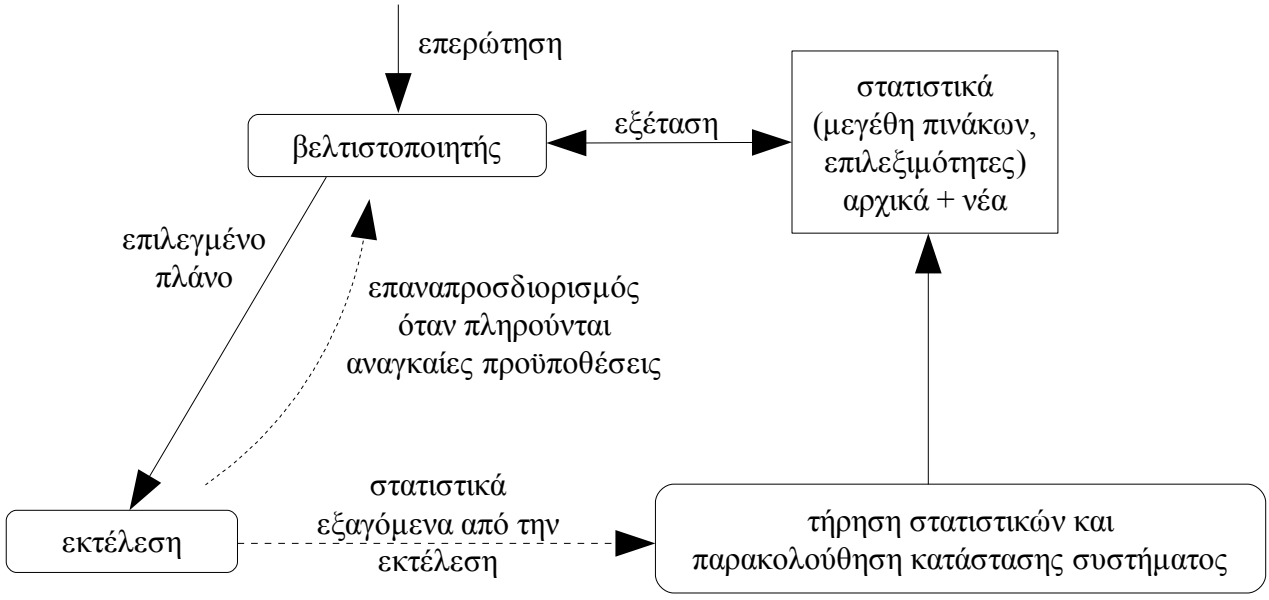


Σχήμα 3.1 – Κλασσική προσέγγιση

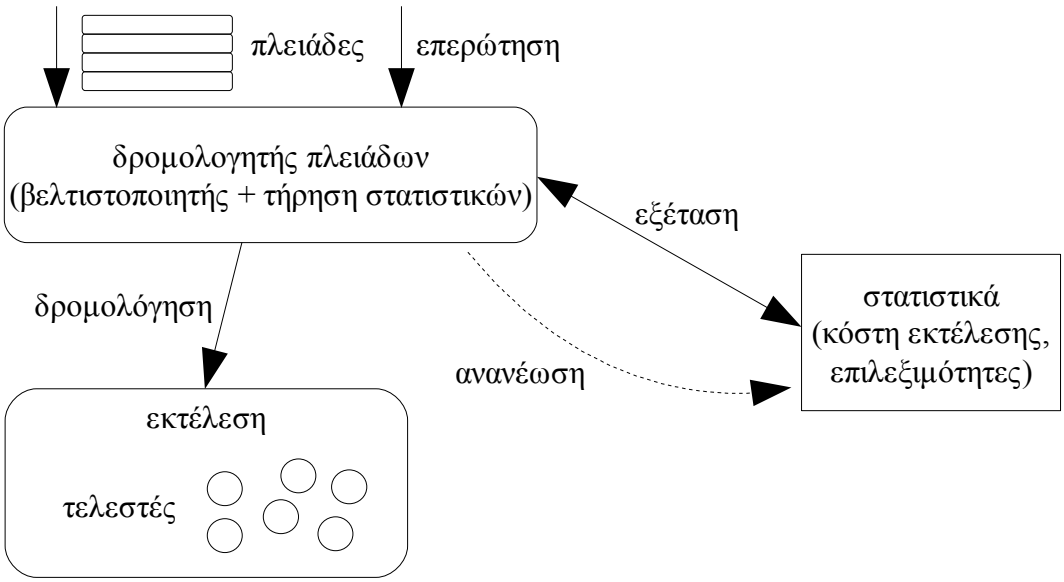
δρομολόγηση – πλειάδων – (routing-based) και η βασισμένη στη συνέχεια των επερωτήσεων (continuous-query-based).

Τα συστήματα που υιοθετούν τη λογική της "**βασισμένης στο πλάνο**" (plan-based) προσαρμοστικότητας επεκτείνουν την παραδοσιακή φιλοσοφία της κατασκευής πλάνου πρώτα και εκτέλεσης έπειτα παρακολουθώντας την πορεία της εκτέλεσης του πλάνου και επαναπροσδιορίζοντάς το όποτε συγκεκριμένες ιδιότητες μεταβάλλονται και παρεκκλίνουν σημαντικά από τις αρχικές εκτίμησης του βελτιστοποιητή. Τέτοιες ιδιότητες είναι το μέγεθος των ενδιάμεσων αποτελεσμάτων ανάμεσα στους διαδοχικούς τελεστές που απαρτίζουν μια επερώτηση και η διαθέσιμη μνήμη του συστήματος. Όπως φαίνεται στο Σχήμα 3.2 αυτό αποτυπώνεται επεκτείνοντας την "τήρηση στατιστικών" έτσι ώστε να συμπεριλάβει ιδιότητες του πλάνου και των συνθηκών του συστήματος μαζί με ορισμένα κριτήρια που θέτουν τις συνθήκες κάτω από τις οποίες ο επαναπροσδιορισμός του πλάνου ενεργοποιείται.

Η **βασισμένη στη δρομολόγηση – πλειάδων** – (routing-based) προσαρμογή αποτελεί μια αρκετά πρωτότυπη προσέγγιση. Συγκεκριμένα, αποφεύγεται η χρήση του παραδοσιακού βελτιστοποιητή και γενικότερα του οργανωμένου πλάνου εκτέλεσης σε μια επερώτηση. Αντίθετα, οι επερωτήσεις εκτελούνται μέσω της δρομολόγησης των πλειάδων μέσα σε ένα σύνολο από τελεστές, προσπαθώντας να εξασφαλιστεί ότι η διαδρομή που θα ακολουθήσουν είναι κάθε φορά η πιο αποδοτική. Αυτό αποτυπώνεται στο Σχήμα 3.3 όπου ο δρομολογητής πλειάδων εμπεριέχει ταυτόχρονα τόσο τη βελτιστοποίηση όσο και την τήρηση στατιστικών με σκοπό την επιλεκτική δρομολόγηση των πλειάδων στους τελεστές. Η πλειοψηφία των πλειάδων ακολουθεί το πιο



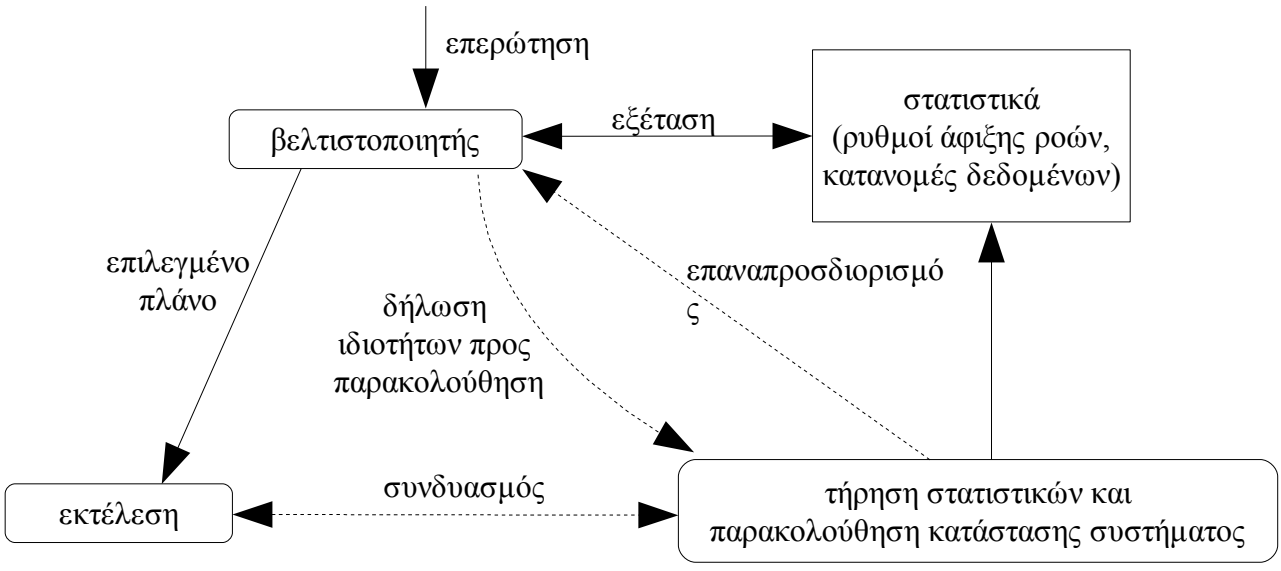
Σχήμα 3.2 – Προσέγγιση βασισμένη στο πλάνο



Σχήμα 3.3 – Προσέγγιση βασισμένη στη δρομολόγηση πλειάδων

αποδοτικό μέχρι στιγμής μονοπάτι ενώ οι υπόλοιπες εξερευνούν τα άλλα μονοπάτια. Η τεχνική των eddies (στρόβιλοι), για την οποία θα γίνει ιδιαίτερος λόγος παρακάτω, χρησιμοποιεί αυτή τη λογική.

Τα συστήματα που εφαρμόζουν **προσαρμοστικότητα συνεχών επερωτήσεων** (continuous-query-based), τέλος, ουσιαστικά υποστηρίζουν προσαρμοστικότητα στην εκτέλεση συνεχών επερωτήσεων, όπως τις έχουμε ορίσει. Χρησιμοποιούν τη δημιουργία πλάνου εκτέλεσης της επερώτησης και βελτιστοποιητή, όπως στη λογική που βασίζεται σε πλάνο. Παρολαυτά, επικεντρώνονται περισσότερο στην παρακολούθηση των μεταβολών στα χαρακτηριστικά των ροών



Σχήμα 3.4 – Προσαρμοστικότητα συνεχών επερωτήσεων

και της κατάστασης του συστήματος κατά την εκτέλεση των συνεχών επερωτήσεων από ότι στατιστικά που επιβάλλουν κάποιο πλάνο και μπορεί να είναι λαθεμένα. Για να ελαχιστοποιήσουν τον επιπλέον φόρτο που η προσαρμοστικότητα απαιτεί χρησιμοποιούν τεχνικές δειγματοληψίας πάνω στις ροές και συνδυάζουν τις παρατηρούμενες τιμές με την πορεία εκτέλεσης των επερωτήσεων. Αυτό φαίνεται στο Σχήμα 3.4.

Για να γίνει κατανοητή η έννοια και η αξία της αναπροσαρμογής μιας επερώτησης ας θεωρήσουμε μια συνένωση ανάμεσα σε τέσσερις ροές R, S, T και U που γίνεται πάνω σε ένα συγκεκριμένο πλαίσιο (window) των ροών. Ας υποθέσουμε ότι η συνένωση ανάμεσα στις R και T παράγει μεγάλο αριθμό ενδιάμεσων πλειάδων ενώ η συνένωση ανάμεσα στις R και U μικρό αλλά και τα δύο αυτά στοιχεία είναι άγνωστα στο σύστημα που αναλαμβάνει την επερώτηση. Γνωρίζοντάς τα, όμως, η βέλτιστη ακολουθία μερικών συνενώσεων είναι **[(R ⋈ U) ⋈ S] ⋈ T**. Ένας βελτιστοποιητής που εκτιμήσει λάθος τις επιλεξιμότητες των τελεστών θα μπορούσε να δώσει σαν προτεινόμενο πλάνο εκτέλεσης **[(R ⋈ S) ⋈ T] ⋈ U**. Ένα παραδοσιακό σύστημα που ακολουθεί το προδιαγεγραμμένο πλάνο εκτέλεσης δε θα αντιληφθεί το πρόβλημα στη βελτιστοποίηση κατά τη διάρκεια της εκτέλεσης διότι δεν είναι ικανό να συμπεράνει τις επί μέρους επιλεξιμότητες απλώς και μόνο από την ίδια τη διαδοχή των τελεστών. Αν μπορούσε να παρακολουθεί το μέγεθος του αποτελέσματος της συνένωσης **[(R ⋈ S) ⋈ T]** θα ήταν δυνατόν να εκτιμήσει ότι το αυτό είναι μεγαλύτερο από αυτό που είχε αρχικά προβλεφθεί και άρα δεν παράγει τα μικρότερα ενδιάμεσα αποτελέσματα. Επομένως θα ενεργοποιούσε τη διαδικασία για επαναπροσδιορισμό του πλάνου της επερώτησης. Αν ακόμη και τότε δεν επιλεγεί το βέλτιστο πλάνο θα υπάρξει δυνατότητα για μετέπειτα αναπροσαρμογή με βάση τις νέες παρατηρήσεις. Βέβαια, ακόμη και αν το βέλτιστο πλάνο εντοπιστεί, είναι πιθανόν οι νέες πλειάδες που καταφθάνουν στη συνέχεια να έχουν τέτοιες κατανομές που να εξαναγκάσουν το βελτιστοποιητή σε νέες προσαρμογές. Αυτή θα είναι μια διαδικασία που θα εφαρμόζεται για όσο χρονικό διάστημα η συνεχής επερώτηση χρειάζεται να παράγει αποτελέσματα – θα είναι μια διαδικασία τόσο συνεχής όσο και η επερώτηση.

Φυσικά, η αναπροσαρμογή της εκτέλεσης μιας επερώτησης δεν είναι εύκολο να γίνει απλά και χωρίς κόστος. Είναι βασικός σκοπός να διατηρηθεί κατά το δυνατόν η συνέχεια και η συνέπεια της εκτελούμενης επερώτησης όσον αφορά τόσο τις παραγόμενες πλειάδες του αποτελέσματος όσο και την ομαλότητα του ρυθμού παραγωγής τους. Ο νέος τρόπος εκτέλεσης οφείλει καταρχάς να συνεχίσει να δίνει έγκυρο αποτέλεσμα χωρίς να αναπαράγει μέρος του που έχει ήδη παραχθεί. Ακόμη, είναι ιδιαίτερα αποδοτικό η νέα εκτέλεση να επαναχρησιμοποιεί πόρους που χρησιμοποιούσε ο προηγούμενη, όπως, για παράδειγμα, ευρετήρια, ενδιάμεσους τελεστές και προσωρινές δομές, και να μην ξεκινά την εκτέλεση από το μηδέν. Βέβαια, η επαναχρησιμοποίηση

έχει και αυτή το δικό της κόστος και πρέπει να εφαρμόζεται όπου μπορεί να κριθεί ότι αυτό πραγματικά αξιοποιείται.

3.3 ΤΉΡΗΣΗ ΣΤΑΤΙΣΤΙΚΏΝ

Η ανάγκη για τήρηση στατιστικών είναι πλέον εμφανής τόσο σε ένα παραδοσιακό ΣΔΒΔ όσο και σε ένα ΣΔΡΔ. Η βέλτιστη εκτέλεση των επερωτήσεων και η αποδοτική διαχείριση των πόρων τους εξαρτάται σημαντικά από τα στατιστικά, έτσι ώστε να λαμβάνονται οι κατά το δυνατόν ορθότερες αποφάσεις αλλά και για να επιβεβαιώνονται ή να καταρρίπτονται αυτές ανάλογα με τις συνθήκες κατά τη διάρκεια της λειτουργίας. Ορισμένα από τα απαραίτητα στατιστικά μπορούν να παρατηρηθούν απευθείας από την εκτέλεση του πλάνου σε μια βασισμένη στο πλάνο εκτέλεση ή από τη ροή των πλειάδων σε μια βασισμένη στη δρομολόγηση πλειάδων εκτέλεση. Για παράδειγμα, για να υπολογιστεί η επιλεξιμότητα ενός τελεστή που παίρνει μέρος σε μια επερώτηση το σύστημα δεν έχει παρά να καταμετρήσει τις πλειάδες του μερικού αποτελέσματος.

Υπάρχουν, φυσικά, και στατιστικά που δεν μπορούν να συναχθούν από το πλάνο εκτέλεσης. Μια πιθανή λύση για το σύστημα είναι να χρησιμοποιήσει επί τούτου επιπλέον πλάνα εκτέλεσης πέραν του θεωρούμενου βέλτιστου. Οι επιπλέον αυτοί υπολογισμοί μπορούν να αξιοποιηθούν είτε και στην ίδια την παραγωγή του αποτελέσματος είτε αποκλειστικά και μόνο για το σκοπό της τήρησης χρήσιμων στατιστικών στοιχείων. Οι κυριότεροι γενικοί τρόποι τήρησης στατιστικών συνοψίζονται ως εξής [30]:

- **Παρατήρηση (observation).**
  Η "παρατήρηση" χρησιμοποιείται ευρύτατα στα συστήματα που υιοθετούν προσαρμοστικότητα βασισμένη στο πλάνο εκτέλεσης. Διάφορα στατιστικά, όπως επιλεξιμότητες, δειγματοληπτικά στοιχεία και ιστογράμμα[9]τα συλλέγονται από τις πλειάδες που συμμετέχουν στην εκτέλεση και μάλιστα σε διάφορα επιλεγμένα σημεία του πλάνου ανάμεσα στους τελεστές.

- **Αναζήτηση (exploration).**
  Η "αναζήτηση" αναφέρεται στην προσαρμοστικότητα βάσει δρομολόγησης πλειάδων όπου ένα μέρος των εισερχόμενων πλειάδων ακολουθεί εναλλακτικές διαδρομές για να υπολογιστούν επιλεξιμότητες και κόστη και σε αυτές, μιας και κάτι τέτοιο δε θα ήταν εφικτό αν τύγχανε προτίμησης μόνο η εκτιμώμενη βέλτιστη διαδρομή.

9 Για τα ιστογράμματα (histograms) θα γίνει ιδιαίτερος λόγος παρακάτω.

- **Συναγωνισμός (competition).**

  Ο "συναγωνισμός" είναι συναφής με την "αναζήτηση" αλλά δρομολογεί τις ίδιες εισερχόμενες πλειάδες ανάμεσα σε πολλές "συναγωνιζόμενες" διαδρομές και πλάνα εκτέλεσης για να συγκρίνει κόστη και επιλεξιμότητες – πέρα, πάντοτε, από το εκτιμώμενο βέλτιστο που ήδη εκτελείται και παράγει αποτελέσματα.

- **Περιγραφή (profiling).**

  Σε αυτή την τεχνική ένα μέρος των πλειάδων ακολουθεί εναλλακτικά πλάνα και περνά από διάφορους τελεστές αλλά με αποκλειστικό σκοπό την τήρηση στατιστικών στοιχείων.

Ανάλογα με την ποιότητα και την ποσότητα των στατιστικών που τηρούνται ο επιπλέον φόρτος που το σύστημα χρειάζεται να καταβάλλει διαφοροποιείται. Οι δύο πρώτες προσεγγίσεις δεν επιβάλλουν εγγενώς μεγάλο φόρτο στο σύστημα, παρά μόνο αν ζητούνται πολλά στατιστικά κάτι που μπορεί να εξισορροπηθεί ανάλογα με τις ανάγκες. Οι δύο τελευταίες προσεγγίσεις, από την άλλη, επιφέρουν μεγαλύτερο ποσοστό πλεονάζοντα φόρτου, ο οποίος αξιοποιείται περισσότερο για στατιστικά και λιγότερο ή καθόλου στην παραγωγή των αποτελεσμάτων. Ακόμη, οι δύο τελευταίες είναι φανερά πιο ενεργητικές προσεγγίσεις, μιας και είτε επεμβαίνουν στο πλάνο εκτέλεσης είτε χρησιμοποιούν επιπλέον πλάνα, ενώ οι δύο πρώτες παρακολουθούν περισσότερο παθητικά την εκτέλεση. Τέλος, η "παρατήρηση" και ο "συναγωνισμός" αποδίδουν με μεγαλύτερη βεβαιότητα ακριβείς μετρήσεις επειδή αναφέρονται σε πραγματικές παρατηρήσεις, ενώ οι άλλες δύο προσεγγίσεις εξαρτώνται από τις συσχετίσεις των τιμών των πλειάδων και τις εκάστοτε επιλογές του βελτιστοποιητή.

3.4 ΤΕΧΝΙΚΈΣ ΚΑΤΆ ΤΗ ΒΕΛΤΙΣΤΟΠΟΊΗΣΗ

Η βελτιστοποίηση και γενικότερα η διαχείριση των επερωτήσεων στηριζόταν παραδοσιακά στα μεγέθη των σχέσεων, στα μεγέθη των ενδιάμεσων αποτελεσμάτων, στον υπολογισμό του κόστους εκτέλεσης και του κόστους των αναγνώσεων από τα σταθερά μέσα αποθήκευσης στα οποία βρίσκονταν οι βάσεις δεδομένων. Ενώ κάποια από αυτά τα στοιχεία παραμένουν σημαντικά και στις ροές δεδομένων το κύριο αποθηκευτικό μέσο που αποτελούσαν οι δίσκοι δεν υπάρχει παρά σαν βοηθητικό μέσο – όταν η κύρια μνήμη δεν επαρκεί. Έτσι η όλη θεώρηση στην οποία βασίζεται η βελτιστοποίηση μπορεί και πρέπει να ιδωθεί από ένα διαφορετικό πρίσμα. Το κόστος των αναγνώσεων από το δίσκο δεν είναι το κυρίαρχο εμπόδιο στις ροές δεδομένων κι επομένως δεν

μπορεί να αποτελέσει τη βάση πάνω στην οποία η βελτιστοποίηση θα στηριχτεί [17]. Ο λόγος για αυτό έχει αναλυθεί στα προηγούμενα κεφάλαια και αφορά αιτίες όπως το γεγονός ότι οι ροές δεν έχουν πεπερασμένο μέγεθος και καταφθάνουν με αυθαίρετους ρυθμούς στο σύστημα επεξεργασίας.

3.4.1 ΒΕΛΤΙΣΤΟΠΟΊΗΣΗ ΒΆΣΕΙ ΡΥΘΜΟΎ

Είναι κατανοητό μέχρι στιγμής ότι ο ρυθμός άφιξης των πλειάδων των ροών παίζει καταλυτικό ρόλο στην όλη διαχείριση ενός ΣΔΡΔ. Έχουμε δει, ακόμη, πώς αντιμετωπίζουν το ρυθμό αυτό και τις διακυμάνσεις του οι διάφοροι αλγόριθμοι εκτέλεσης συνεχών επερωτήσεων (συνενώσεις, MJoin κλπ). Τον ίδιο κεντρικό ρόλο παίζει ο ρυθμός και σε πολλά μοντέλα μέτρησης του κόστους εκτέλεσης των επερωτήσεων στην προσπάθεια για το σχεδιασμό και τη βελτιστοποίησή τους [32][33]. Η κεντρική ιδέα πίσω από αυτά είναι ότι οι σημαντικότερες παράμετροι εκτέλεσης είναι:

- Ο αριθμός των πλειάδων που εισέρχονται στο σύστημα στη μονάδα του χρόνου – δηλαδή, ο ρυθμός άφιξης.

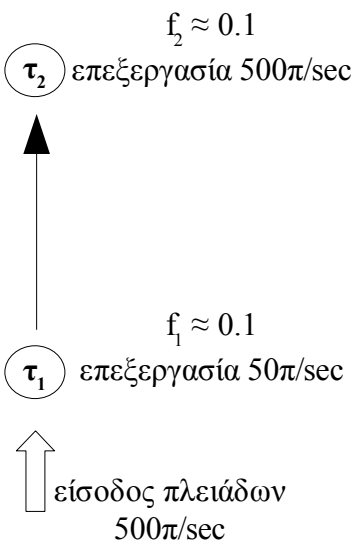
- Ο αριθμός των πλειάδων στη μονάδα του χρόνου που το σύστημα αναμένεται να καταφέρει να επεξεργαστεί.

- Ο αριθμός των πλειάδων αποτελέσματος στη μονάδα του χρόνου που το σύστημα αναμένεται να παράγει.

Ας δούμε το ακόλουθο παράδειγμα. Έστω μία επερώτηση με δύο διαδοχικούς δύο τελεστές τ_1 και τ_2 με ίσες περίπου επιλεξιμότητες – έστω $f_1 \approx f_2 \approx 0.1$ – αλλά με διαφορετικό ρυθμό παραγωγής του μερικού του αποτελέσματος ο καθένας. Έστω ότι ο αριθμός άφιξης των πλειάδων στο σύστημα

Χρόνος	Επεξ. Πλειάδ.	Παραγ. Πλειαδ.	Πλειαδ. σε Αναμονή
0	5	0.5	0
1	10	1.0	0
2	15	1.5	0
3	20	2.0	0
4	25	2.5	0
...	...	...	...

Χρόνος	Επεξ. Πλειάδ.	Παραγ. Πλειαδ.	Πλειαδ. σε Αναμονή
0	50	5	450
1	100	10	900
2	150	15	1350
3	200	20	1800
4	250	25	2250
...	...	...	...



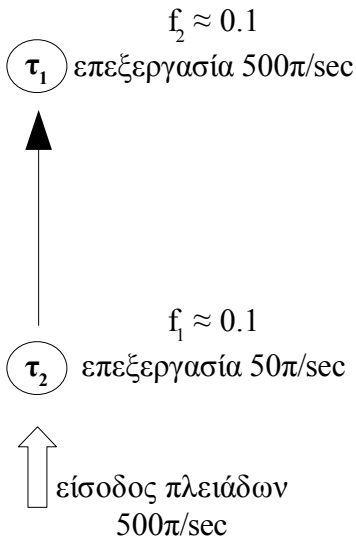
Σχήμα 3.5 – Ο τ_2 διαδέχεται τον τ_1

είναι 500 π/sec (πλειάδες ανά δευτερόλεπτο) και ότι ο τ_1 έχει την ικανότητα να επεξεργάζεται 50π/sec ενώ ο τ_2 μπορεί να επεξεργάζεται τις πλειάδες του άμεσα. Ας υποθέσουμε ότι υπάρχουν δύο πιθανά πλάνα εκτέλεσης Π_1 και Π_2. Στο Σχήμα 3.5 φαίνεται ένα στιγμιότυπο επεξεργασίας στην περίπτωση όπου ο τ_2 διαδέχεται τον τ_1. Στις στήλες βρίσκονται ο χρόνος, οι επεξεργασμένες πλειάδες, οι παραγόμενες πλειάδες και οι πλειάδες που βρίσκονται σε αναμονή εξαιτίας του ότι περισσότερες πλειάδες εισέρχονται από ότι εξέρχονται. Οι εισερχόμενες πλειάδες του τ_2 είναι οι παραγόμενες του τ_1. Στο Σχήμα 3.6 φαίνεται ένα στιγμιότυπο επεξεργασίας στην περίπτωση όπου ο τ_1 διαδέχεται τον τ_2. Αντίστοιχα, οι εισερχόμενες πλειάδες του τ_1 είναι οι παραγόμενες του τ_2.

Είναι φανερά άμεσα δύο πολύ σημαντικά στοιχεία. Πρώτον, όταν ο τ_2 διαδέχεται τον τ_1 ο ρυθμός των παραγόμενων πλειάδων είναι πάρα πολύ μικρότερος από ότι στη δεύτερη περίπτωση. Δεύτερον, ότι οι πλειάδες που παραμένουν σε αναμονή είναι πολύ λιγότερες στη δεύτερη περίπτωση. Μιας και το επεξεργαστικό κόστος είναι το ίδιο – σε χρόνο επεξεργασίας – και στις δύο περιπτώσεις μπορούμε με ασφάλεια να συμπεράνουμε ότι η δεύτερη είναι πολύ πιο αποδοτική. Μια παραδοσιακή θεώρηση βελτιστοποίησης που θα λάμβανε υπόψη της τα τρέχοντα μεγέθη των ροών και τις επιλεξιμότητες των τελεστών δε θα μπορούσε να προβλέψει κάτι τέτοιο. Άλλωστε τα μεγέθη των ροών είναι απροσδιόριστα και θεωρητικά απεριόριστα. Βασίζοντας τη θεώρησή του ο βελτιστοποιητής στο ρυθμό παραγωγής πλειάδων αποτελέσματος είναι δυνατόν να εντοπίσει το βέλτιστο πλάνο εκτέλεσης.

Χρόνος	Επεξ. Πλειάδ.	Παραγ. Πλειαδ.	Πλειαδ. σε Αναμονή
0	50	5	0
1	100	10	0
2	150	15	0
3	200	20	0
4	250	25	0
...	...	...	...

Χρόνος	Επεξ. Πλειάδ.	Παραγ. Πλειαδ.	Πλειαδ. σε Αναμονή
0	500	50	0
1	1000	100	0
2	1500	150	0
3	2000	200	0
4	2500	250	0
...	...	...	...



Σχήμα 3.6 – Ο τ_1 διαδέχεται τον τ_2

Η προσέγγιση που προτείνεται ως βέλτιστη [32] επικεντρώνεται στη ρυθμοαπόδοση (throughput) των παραγόμενων πλειάδων μιας επερώτησης Q, που ορίζεται ως:

$$\boldsymbol{\rho(Q) = \lambda(Q) / \mu(Q)},$$

με λ(Q) το ρυθμό παραγωγής πλειάδων αποτελέσματος και μ(Q) το συνολικό κόστος επεξεργασίας της επερώτησης Q. Ακόμη, ο παράγοντας 1/μ(Q) αναφέρεται ως παράγοντας αξιοποίησης (utilization). Δοθείσης μιας επερώτησης Q υπάρχει ένας αριθμός εναλλακτικών πλάνων εκτέλεσής της. Εάν ο σκοπός προσδιορίζεται στο να μεγιστοποιηθεί η ρυθμοαπόδοση ρ(Q) του συστήματος, τότε πρέπει είτε να μεγιστοποιηθεί ο ρυθμός παραγόμενων πλειάδων – αριθμητής του κλάσματος – είτε να ελαχιστοποιηθεί το κόστος επεξεργασίας – παρονομαστής του κλάσματος (το τελευταίο, ουσιαστικά ισοδυναμεί με μεγιστοποίηση του παράγοντα αξιοποίησης).

Σε πολλές περιπτώσεις αν επιτευχθεί έστω και μόνο το ένα από τα δύο ο σκοπός επιτυγχάνεται. Αν θεωρήσουμε μια επερώτηση όπου εφαρμόζονται πλαίσια (windows) σταθερού μεγέθους και ο ρυθμός εισόδου παραμένει σταθερός τότε ο παραγόμενος ρυθμός δε θα αλλάξει πολύ ανεξάρτητα από το πλάνο εκτέλεσης. Αυτό γιατί υπό αυτές τις συνθήκες το πρόβλημα έχει μεταφερθεί στο κλασσικό πρόβλημα βελτιστοποίησης επερωτήσεων όπου τα δεδομένα βρίσκονται αποθηκευμένα σε δίσκους, μόνο που εδώ τη θέση των δίσκων έχουν πάρει τα πλαίσια.

Ιδιαίτερο ενδιαφέρον υπάρχει όταν ο ρυθμός άφιξης δεν είναι σταθερός αλλά μεταβάλλεται με την πάροδο του χρόνου – μια θεώρηση αρκετά ρεαλιστική. Έστω μια επερώτηση Q και m πιθανά εναλλακτικά πλάνα εκτέλεσης P_i, i=1..m. Αν ο ρυθμός παραγωγής πλειάδων ανάλογα με το πλάνο εκτέλεσης είναι $\lambda_{Pi}(t)$ τότε το πλήθος των παραγόμενων πλειάδων για κάθε πλάνο P_i γίνεται:

$$\boldsymbol{n_{Pi}(t) = \int \lambda_{Pi}(t)\, dt}, \text{ με ολοκλήρωση από } 0..t$$

Αν είναι γνωστή η συσχέτιση (correlation) ανάμεσα στις πλειάδες αποτελέσματος $n_{Pi}(t)$ που παράγονται σε χρονικό διάστημα t από το πλάνο P_i με εκτιμώμενο ρυθμό πλειάδων αποτελέσματος $\lambda_{Pi}(t)$ είναι δυνατόν να γίνει η εξής βελτιστοποίηση: Θεωρώντας ένα συγκεκριμένο χρονικό διάστημα t_P μπορεί να επιλεγεί το πλάνο που μεγιστοποιεί τον αριθμό των παραγόμενων πλειάδων αποτελέσματος σε αυτό. Από την άλλη πλευρά, αν θεωρήσουμε το χρόνο μεταβλητό και σταθεροποιήσουμε το πλήθος των παραγόμενων πλειάδων σε n_Q μπορούμε να επιλέξουμε το πλάνο εκείνο που ελαχιστοποιεί το χρόνο που απαιτείται – άρα μεγιστοποιεί το ρυθμό παραγωγής – για να παραχθούν αυτές.

Η ολοκλήρωση που απαιτείται από την παραπάνω προσέγγιση είναι θεωρητικά ορθή αλλά

υπολογιστικά δύσκολη ιδίως όταν τα εναλλακτικά πλάνα εκτέλεσης είναι πολλά. Η πρακτική λύση που προτείνεται σε αυτή την περίπτωση είναι η βελτιστοποίηση να γίνεται όχι γενικά αλλά τοπικά και στοχευμένα με δύο εναλλακτικούς τρόπους που κάνουν χρήση ευριστικών (heuristics) [32].

Ο πρώτος τρόπος είναι η **μεγιστοποίηση τοπικού ρυθμού** (local rate maximization) που βασίζεται στην υπόθεση ότι για να είναι μέγιστος ο συνολικός – τελικός – ρυθμός παραγωγής πλειάδων αποτελέσματος μιας επερώτησης θα πρέπει να είναι μέγιστοι και οι ενδιάμεσοι ρυθμοί των τελεστών που την απαρτίζουν. Γι' αυτό προτείνεται να οργανωθεί το πλάνο εκτέλεσης με τέτοιο τρόπο ώστε να μεγιστοποιούνται οι εκτιμήσεις για κάθε τμήμα του. Σε μία συνένωση δύο ροών ο ρυθμός παραγωγής αποτελέσματος είναι:

$$\lambda = \frac{f \cdot \lambda_1 \cdot \lambda_2}{C(\lambda_1,\lambda_2)} \cdot t,$$

όπου f η επιλεξιμότητα της συνένωσης, λ_1 και λ_2 οι ρυθμοί εισόδου πλειάδων, $C(\lambda_1,\lambda_2)$ το κόστος επεξεργασίας των εισερχόμενων πλειάδων (ευρετηριασμός, αποθήκευση στη μνήμη κλπ) – που εξαρτάται από τους ρυθμούς εισόδου – και t η μελετώμενη χρονική διάρκεια. Αν το αποτέλεσμα της προαναφερθείσας συνένωσης συνενωνόταν με μία τρίτη ροή με επιλεξιμότητα f' τότε ο συνολικός ρυθμός γίνεται:

$$\lambda' = \frac{f' \cdot \lambda \cdot \lambda_3}{C(\lambda,\lambda_3)} \cdot t$$

και αντίστοιχα το αποτέλεσμα κλιμακώνεται όταν υπάρχουν στην ίδια επερώτηση περισσότεροι συνδυασμοί.

Ο δεύτερος τρόπος είναι **ελαχιστοποίηση τοπικού χρόνου** (local time minimization) αφορά το χρόνο που χρειάζεται ένας τελεστής συνένωσης για να παράξει ένα αποτέλεσμα. Αν το πλήθος των παραγόμενων πλειάδων είναι n και ο ρυθμός παραγωγής τους είναι r τότε ο απαιτούμενος χρόνος είναι **n/r**. Αν αυτό – η ελαχιστοποίηση χρόνου – επιτευχθεί, τότε είναι δυνατόν ένα σύστημα επεξεργασίας να μπορεί να βελτιστοποιήσει μια συγκεκριμένη επερώτηση μέχρις ότου αυτή παράγει N πλειάδες, πληρώνοντας προφανώς το τίμημα του ότι σημαντική υπολογιστική ισχύς αφιερώνεται σε αυτήν, κι έπειτα να προτιμήσει κάποιο πιο συντηρητικό πλάνο εκτέλεσης. Η λογική αυτή εξετάζει την βελτιστοποίηση με κατεύθυνση από το αποτέλεσμα προς τις ροές που το παράγουν. Δεν ξεκινά, δηλαδή, από τις ροές για να ανελιχθεί προς το επιθυμητό αποτέλεσμα αλλά αντίστροφα. Έτσι για μία συνένωση όπως η **A ⋈ (B ⋈ C)** μπορούν να τεθούν απαιτήσεις χρονικές όσον αφορά την τελική συνένωση μόνο – δηλαδή τη συνένωση του A με το αποτέλεσμα της συνένωσης του B με το C – χωρίς να καθοριστεί τίποτα σε περαιτέρω επίπεδο για την επί μέρους συνένωση του B με το C.

3.4.2 ΔΙΑΧΕΊΡΙΣΗ ΠΌΡΩΝ

Οι ρυθμοί στους οποίους βασίζεται η βελτιστοποίηση βάσει ρυθμού δεν είναι στην πραγματικότητα γνωστοί και ακόμη και όταν γίνονται εκτιμήσεις για αυτούς η αβεβαιότητα είναι μεγάλη. Η ανάγκη για αποδοτική διαχείριση των πόρων είναι ζωτικής σημασίας σε ένα ΣΔΡΔ, κάτι που έχουμε ήδη εντοπίσει πολλές φορές. Ταυτόχρονα, όμως, είναι και ένα μέσο για τη βελτιστοποίηση της απόδοσής του. Στο προηγούμενο κεφάλαιο, στην ενότητα "Διαχείριση Πόρων", έγινε αναφορά στο πώς ένα σύστημα επεξεργασίας θα μπορούσε να διαχειριστεί τους διαθέσιμους πόρους του έτσι ώστε αυτοί να διαμοιράζονται ανάμεσα στις μονάδες του και να μην σπαταλώνται. Όμοιες αναφορές μπορούν να γίνουν και εδώ, για αυτό δε θα επαναληφθούν.

3.4.3 ΠΟΙΌΤΗΤΑ ΥΠΗΡΕΣΊΑΣ ΚΑΙ ΕΛΆΤΤΩΣΗ ΦΌΡΤΟΥ

Ένα πολύ πιθανό σενάριο κατά την επεξεργασία επερωτήσεων πάνω σε ροές δεδομένων είναι όταν οι διαθέσιμοι επεξεργαστικοί πόροι, που το πλάνο εκτέλεσης απαιτεί για να ανταποκριθεί στους ρυθμούς άφιξης των ροών, δεν επαρκούν. Το πλάνο αυτό θεωρείται ανέφικτο (infeasible plan) και οδηγεί στη διόγκωση των ουρών των πλειάδων που βρίσκονται σε αναμονή ανάμεσα στους τελεστές. Σε αυτές τις περιπτώσεις, όταν πολλές ροές καταφθάνουν με μεγάλους ρυθμούς και, είτε αν οι υπόλοιπες τεχνικές διαχείρισης έχουν εξαντληθεί είτε συνδυαστικά με αυτές, προτείνεται η επιλεκτική απόρριψη[10] πλειάδων (απόρριψη φόρτου – load shedding) από τις ουρές αναμονής των τελεστών [34][35]. Η απόρριψη φόρτου είναι μια εναλλακτική που δε μπορεί να εφαρμοστεί δίχως προσοχή. Η εκλογή των πλειάδων που θα απορριφθούν πρέπει να είναι τέτοια ώστε να μην διακυβεύεται η συνολική ποιότητα των αποτελεσμάτων ή, έστω, αν αυτό συμβαίνει, να γίνεται ελεγχόμενα.

Έτσι, έχουν προταθεί δύο προσεγγίσεις στην ελάττωση φόρτου. Η τυχαία (random) όπου προσδιορίζεται μόνο το ποσοστό των πλειάδων που το σύστημα θα διατηρήσει – και δε θα απορρίψει – και η βασισμένη στη σημασιολογία (semantic) όπου η απόρριψη βασίζεται στην εκτιμώμενη χρησιμότητα των πλειάδων στην παραγωγή του αποτελέσματος. Ένα ΣΔΡΔ που εφαρμόζει ελάττωση φόρτου οφείλει να αποφασίζει σωστά για:

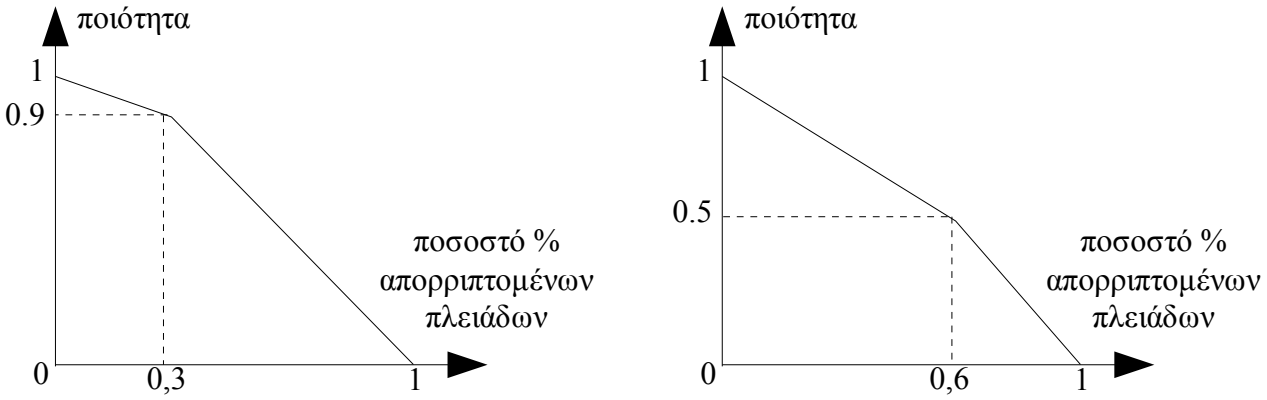
- για το πότε αυτή θα εφαρμοστεί,

10 Η τεχνική αυτή σχετίζεται άμεσα με την τεχνική των προσεγγίσεων (approximation) που αναφέρθηκε στο προηγούμενο κεφάλαιο.

- για το πού θα εφαρμοστεί (η απόρριψη μπορεί να γίνει είτε στην είσοδο του συστήματος είτε σε οποιοδήποτε σημείο του πλάνου εκτέλεσης μιας επερώτησης) και

- για το μέγεθος της απόρριψης (τι ποσοστό επί του συνόλου θα απορριφθεί).

Το μέτρο βάσει του οποίου γίνεται η απόρριψη είναι η ποιότητα υπηρεσίας (quality of service), η οποία εξαρτάται από το ποσοστό των πλειάδων που απορρίπτονται. Αν αυτές είναι πολλές τότε η πιθανότητα να αλλοιωθούν τα αποτελέσματα μεγαλώνει ανάλογα. Στο Σχήμα 3.7 φαίνονται μερικά παραδείγματα ποιότητας υπηρεσίας σε συνάρτηση με το ποσοστό αυτό. Όσο περισσότερο σημαντικές – σε σχέση με τη συμμετοχή τους στην παραγωγή του αποτελέσματος – είναι οι πλειάδες που απορρίπτονται τόσο πιο εύκολα επηρεάζεται η ποιότητα και τόσο περισσότερο μετατοπισμένη προς τα αριστερά είναι η καμπυλότητα της γραφικής παράστασης. Αν για παράδειγμα οι τιμές των πλειάδων μιας ροής είναι ισχυρά αλληλοσυσχετισμένες (correlated) τότε ακόμη και ένα υψηλό ποσοστό απόρριψης θα επηρέαζε λίγο την ποιότητα του αποτελέσματος.



Σχήμα 3.7 – Ποιότητα Υπηρεσίας και Απόρριψη Πλειάδων

Για να εντοπιστεί η ανάγκη για ενεργοποίηση της απόρριψης πλειάδων χρησιμοποιούμε [34] έναν παράγοντα φόρτου L (load coefficient) που δείχνει το τμήμα της υπολογιστικής ισχύος που είναι απαραίτητη για διέλθει μια πλειάδα ως την έξοδο του συστήματος. Αν κάθε τελεστής από τον οποίο η πλειάδα περνάει σπαταλά c_i υπολογιστικό κόστος και έχει επιλεξιμότητα s_i τότε θα έχουμε:

$$\mathbf{L} = \sum_{i=1}^{n}\left(\prod_{j=1}^{n} \mathbf{s}_j\right) * \mathbf{c}_i \quad \text{, όπου n το πλήθος τελεστών}$$

Για μια ροή I που καταφθάνει με ρυθμό $r_ι$ τότε ο συνολικός φόρτος $L_{ολ}$ θα είναι:

$$L_{ολ} = \sum_{i=1}^{m} L_i \, x \, r_i \quad \text{, όπου m το πλήθος των πλειάδων}$$

Ανάλογα με το τρέχοντα φόρτο το σύστημα μπορεί να επιλέξει να εκκινήσει τη διαδικασία απόρριψης πλειάδων. Αν η υπολογιστική ικανότητα του συστήματος είναι M, η συνάρτηση ποιότητας-ποσοστού απόρριψης u(P) για ένα πλάνο εκτέλεσης P και η συνολική επεξεργαστική ισχύς που απαιτείται είναι $L_{ολ} > M$, δηλαδή το πλάνο είναι ανέφικτο, το σύστημα οφείλει να εκτιμήσει ένα νέο πλάνο P' το οποίο να εφαρμόζει απόρριψη πλειάδων έτσι ώστε $L_{ολ}' \leq M < L_{ολ}$ και το u(P) – u(P') να ελαχιστοποιείται.

3.4.4 ΑΝΑΚΑΤΑΝΟΜΗ ΕΠΕΡΩΤΗΣΕΩΝ (QUERY SCRAMBLING)

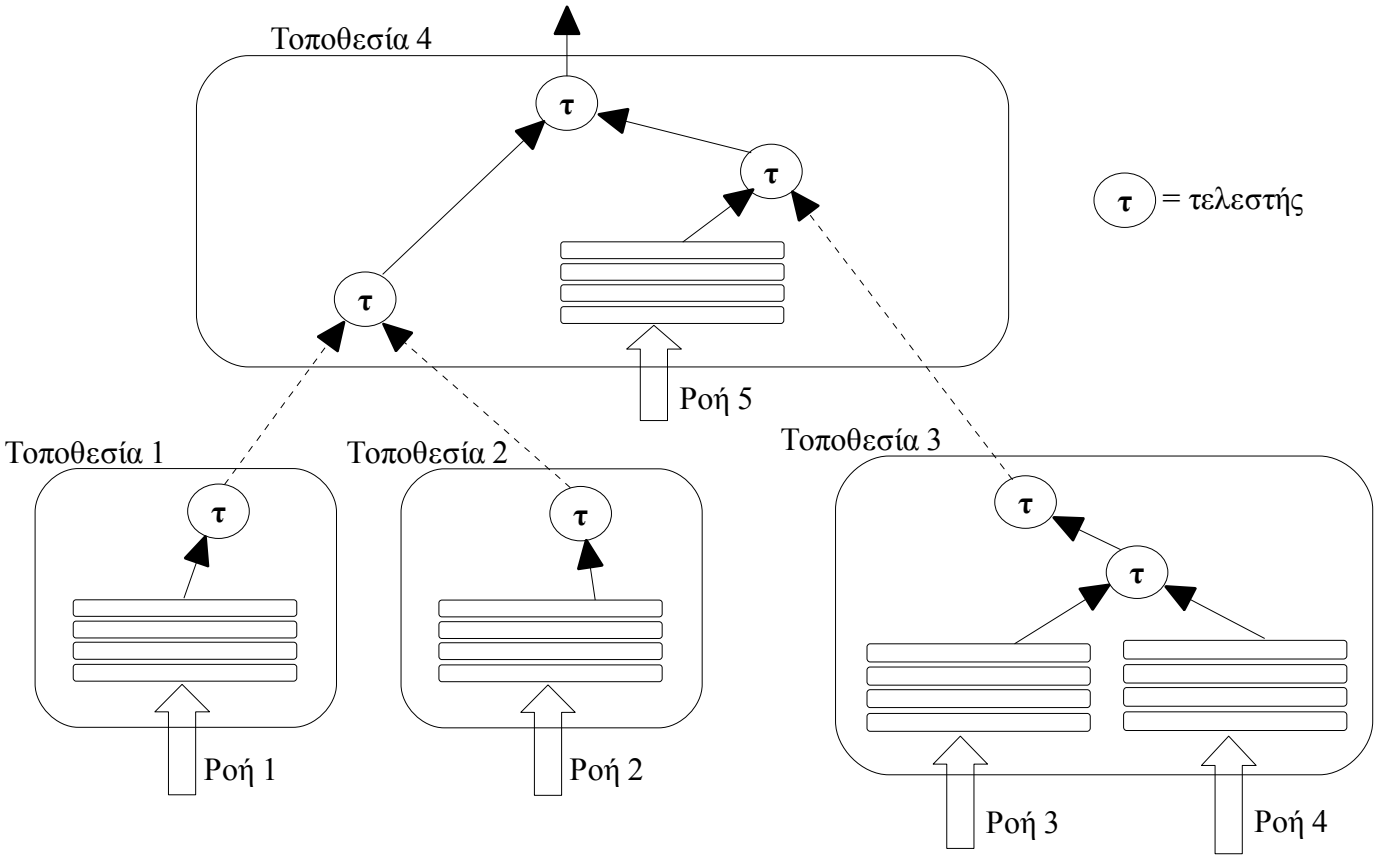
Η τεχνική της ανακατανομής επερωτήσεων (query scrambling) [26][36] είναι μια προσέγγιση που αποσκοπεί στην αντιμετώπιση της δεδομένης αβεβαιότητας που ενυπάρχει στους ρυθμούς άφιξης των ροών. Οι μικρές διακυμάνσεις στους ρυθμούς είναι δυνατόν να αντιμετωπιστούν, για παράδειγμα, με βελτιστοποίηση βασισμένη στο ρυθμό (rate-based), όμως οι μεγάλες εναλλαγές που ίσως συνοδεύονται από περιόδους όπου οι ροές καταφθάνουν και σε ισχυρές ριπές (bursty) δυσκολεύουν πολύ τη διαχείριση στην πράξη. Η τεχνική που συζητούμε σε αυτή την ενότητα προτείνει την ανακατανομή του πλάνου εκτέλεσης έτσι ώστε τα επακόλουθα μεγάλων ρυθμών ή αναπάντεχων καθυστερήσεων στις ροές να εξομαλύνονται. Επιπλέον λαμβάνει υπόψη το γεγονός ότι ένα σύστημα διαχείρισης ροών μπορεί να μη βρίσκεται εντοπισμένο σε ένα μοναδικό σύστημα επεξεργασίας αλλά να είναι κατανεμημένο κι επομένως οι ροές να διέρχονται όχι μόνο ανάμεσα από τους τελεστές των επερωτήσεων αλλά και ανάμεσα στα συμμετέχοντα υποσυστήματα.

Στο Σχήμα 3.8 φαίνεται ένα πλάνο εκτέλεσης που είναι κατανεμημένο σε παραπάνω από μία τοποθεσίες στην κάθε μία από τις οποίες μπορεί να εφαρμόζεται ένας συνδυασμός τελεστών. Αν μία από τις εισόδους παρουσιάζει καθυστερήσεις τότε όλο το πλάνο επηρεάζεται από αυτό το γεγονός και δεν παράγει αποτελέσματα. Ακόμη, αν, για παράδειγμα, η σύνδεση ανάμεσα στις τοποθεσίες 1 και 4 καθυστερεί ή διακοπεί προσωρινά τότε πάλι όλο το πλάνο μένει αιωρούμενο. Η ανακατανομή επερωτήσεων ακολουθεί δύο φάσεις, οι οποίες εκτελούνται επαναληπτικά και συνεχόμενα, για να αντιμετωπίσει αυτές τις καταστάσεις:

- **Επαναπρογραμματισμός επερώτησης (query rescheduling)**: Η σειρά με την οποία οι τελεστές εκτελούνται μεταβάλλεται δυναμικά. Αν η σύνδεση ανάμεσα στις τοποθεσίες 1 και 4 διακοπεί τότε οι τελεστές στις υπόλοιπες τοποθεσίες μπορούν να συνεχίσουν να παράγουν αποτελέσματα ανενόχλητοι. Το πλάνο εκτέλεσης δεν αλλάζει, όμως το

σύστημα προσαρμόζει τη σειρά εκτέλεσης έτσι ώστε οι τελεστές ή η συνδυασμοί τελεστών που έχουν διαθέσιμη είσοδο να εκτελούνται πρώτοι.

- **Σύνθεση τελεστών (operator synthesis)**: Σε αυτή τη φάση νέοι τελεστές εισάγονται στο πλάνο εκτέλεσης. Για παράδειγμα, μία συνένωση μπορεί να προστεθεί ενώ μία άλλη να αφαιρεθεί και έτσι το πλάνο μπορεί να μεταβληθεί σημαντικά.



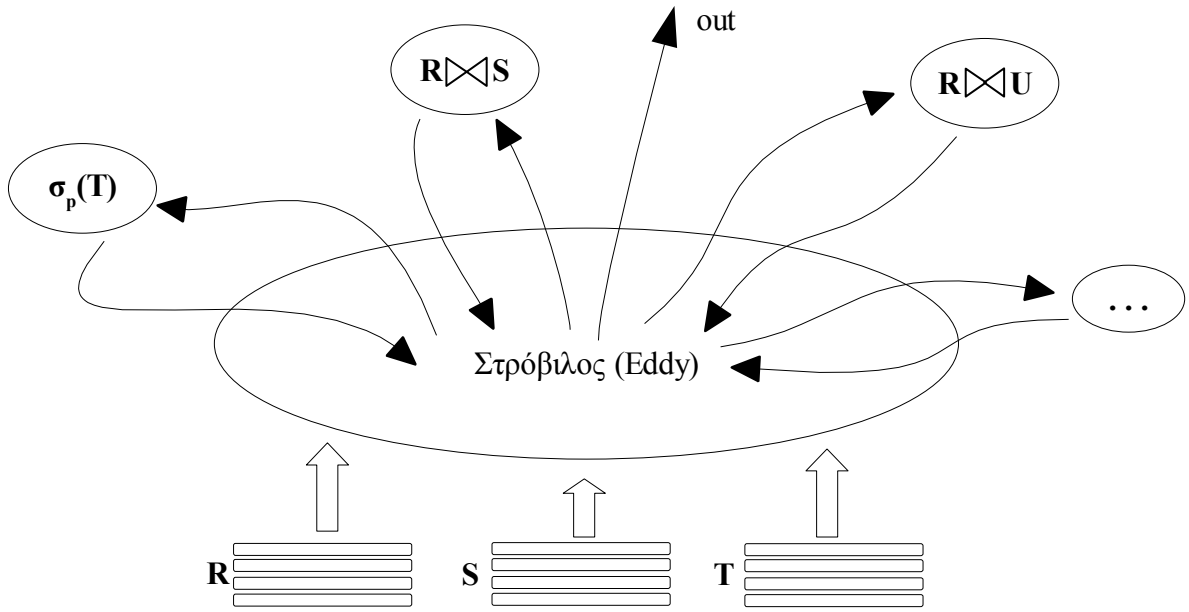
Σχήμα 3.8 – Κατανεμημένο πλάνο εκτέλεσης

3.4.5 ΣΤΡΌΒΙΛΟΙ (EDDIES) ΚΑΙ ΚΛΆΔΟΙ ΕΥΡΕΤΗΡΊΩΝ (STEMS)

Μία πολύ ενδιαφέρουσα τεχνική [37][38] που έχει προταθεί για να αντιμετωπίσει το πρόβλημα των διακυμάνσεων στους ρυθμούς των εισερχόμενων ροών αλλά και της απόδοσης του συστήματος επεξεργασίας είναι αυτή των στροβίλων (eddies). Η φιλοσοφία πίσω από αυτήν είναι να αναχθεί το πρόβλημα της βελτιστοποίησης του πλάνου εκτέλεσης σε ένα πρόβλημα δρομολόγησης – της δρομολόγησης των πλειάδων των εισερχομένων ροών. Προτείνεται, δηλαδή, οι τελεστές που συμμετέχουν στις επερωτήσεις να μη συνδέονται άμεσα μεταξύ τους, σχηματίζοντας με αυτό τον τρόπο το πλάνο εκτέλεσης, αλλά να υπάρχει μια κεντρική οντότητα που να αναλαμβάνει να δρομολογήσει τις πλειάδες ανάμεσα στους τελεστές μετά την είσοδό τους στο σύστημα. Η κεντρική

αυτή οντότητα ονομάστηκε “στρόβιλος” (eddy).

Ένας “στρόβιλος” δεν είναι παρά ένας σύνθετος τελεστής (operator) ο οποίος με δυναμικό και σύγχρονο τρόπο συνεχώς ανακατατάσσει τις διαδρομές των πλειάδων ανάμεσα στους άλλους τελεστές – αυτούς που έχουν τεθεί από τις συνεχείς επερωτήσεις. Είναι ένας δρομολογητής πλειάδων που παρεμβάλλεται ανάμεσα στις ροές δεδομένων που εισέρχονται και τους τελεστές της επεξεργασίας με σκοπό την άμεση (on the fly) βελτιστοποίηση των επερωτήσεων. Η ικανότητα αυτή βασίζεται σε μεγάλο βαθμό στην υπόθεση ότι ένα βέλτιστο πλάνο σπάνια υπολογίζεται με ακρίβεια κι επομένως η προσαρμογής στις αλλαγές είναι η βέλτιστη λύση [37]. Στο Σχήμα 3.9 παρουσιάζεται η γενική ιδέα του στροβίλου, όπου οι πλειάδες εισέρχονται στο σύστημα, δρομολογούνται στους τελεστές κι έπειτα από την επεξεργασία τους σε κάθε τελεστή επιστρέφουν στο στρόβιλο για να δρομολογηθούν στους επόμενους.



Σχήμα 3.9 – Στρόβιλος (Eddy)

Ο στρόβιλος, όπως περιγράφηκε, προσδιορίζει με ένα ιδιαίτερο τρόπο το πλάνο εκτέλεσης των επερωτήσεων με τους τελεστές και τις ροές που συμμετέχουν. Διατηρεί μια ενδιάμεση μνήμη (buffer) στην οποία οι πλειάδες εισέρχονται και δρομολογούνται προς τους τελεστές για επεξεργασία και κάθε τελεστής που συνεργάζεται με το στρόβιλο διατηρεί μία είσοδο για να δέχεται πλειάδες από αυτόν και μία έξοδο για να του τις επιστρέφει. Κάθε πλειάδα που εισέρχεται στο στρόβιλο διαθέτει δύο πίνακες από bits ελέγχου, έναν που δείχνει ποιοι τελεστές χρειάζεται να επεξεργαστούν την πλειάδα (Ready) και έναν που δείχνει ποιοι την έχουν ήδη επεξεργαστεί (Done). Ο στρόβιλος δρομολογεί πλειάδες μόνο προς τους τελεστές που έχουν ενεργοποιημένο το

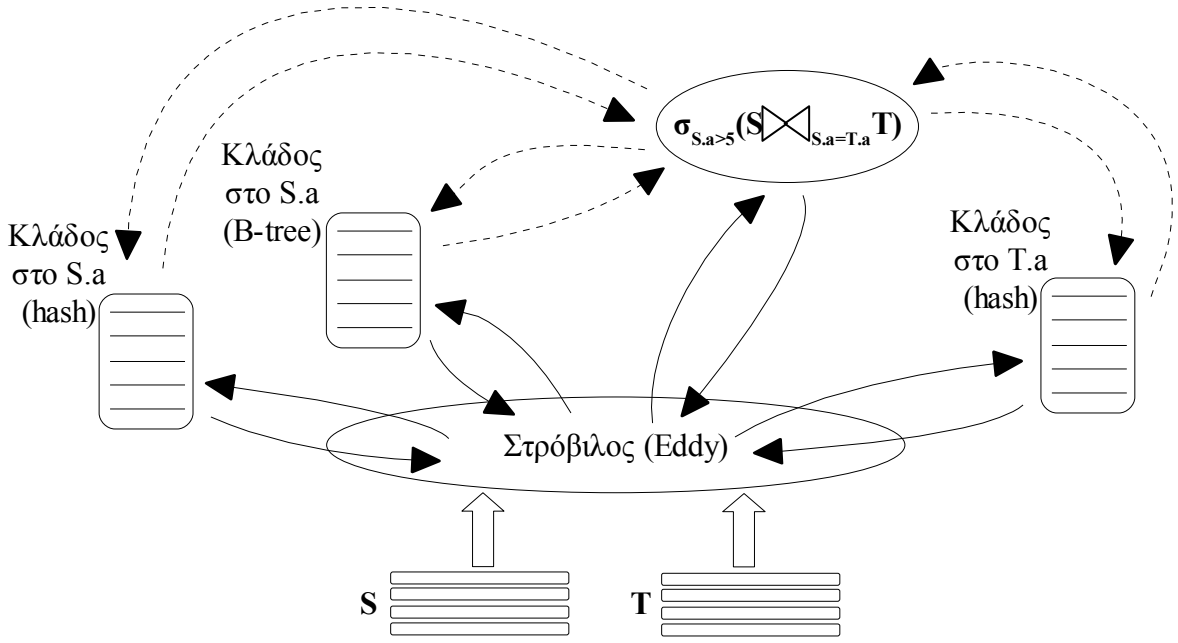
αντίστοιχο bit του πρώτου πίνακα και ενεργοποιεί το αντίστοιχο bit του δεύτερου πίνακα για τους τελεστές που επιστρέφουν μία πλειάδα αφότου τη χρησιμοποιήσουν. Όταν όλα τα bits του δεύτερου πίνακα ενεργοποιηθούν η πλειάδα δε χρειάζεται να παραμείνει άλλο στο σύστημα και απορρίπτεται. Για παράδειγμα, ο στρόβιλος μπορεί να επιτρέψει τη δρομολόγηση πλειάδων και ανάμεσα στους τελεστές – από τελεστή σε τελεστή. Ακόμη, μπορεί να προσδιορίσει η επιλογή των πλειάδων μέσα από την ενδιάμεση μνήμη του να γίνεται με τρόπο πρώτος-μέσα-πρώτος-έξω (FIFO) ή με κάποια άλλη λογική προτεραιότητας. Η τεχνική των στροβίλων εμφανίζεται ιδιαίτερα ευέλικτη και πρωτοποριακή αλλά είναι φανερό ότι από την πολιτική δρομολόγησης που θα ακολουθηθεί θα εξαρτηθεί και η αποδοτικότητα του συστήματος που τους υιοθετεί.

Η ίδια η φυσιολογία των στροβίλων και η έννοια της δρομολόγησης που εισαγάγουν μέσα στο πλάνο εκτέλεσης δίνει τροφή για εφαρμογές τους σε ΣΔΡΔ που μπορούν να εκτελούν κατανεμημένες επερωτήσεις. Μία τέτοια πρόταση έχει γίνει στο [38] και αφορά αλγορίθμους με κατανεμημένους στροβίλους. Οι μετρικές που προτείνονται για την απόδοση του συστήματος είναι ο μέσος χρόνος απόκρισης ενός τελεστή – ο μέσος, δηλαδή, χρόνος ανάμεσα στην είσοδο μιας πλειάδας στον τελεστή μέχρι την έξοδό της – και ο μέγιστος ρυθμός δεδομένων που μπορεί να ικανοποιηθεί χωρίς κανένας τελεστής να συμφορηθεί (bottleneck). Η κατανεμημένη επερώτηση μοντελοποιείται σαν ένα σύνολο από τελεστές $\mathbf{Op_i}$, i=1..n, που βρίσκονται σε ένα δίκτυο. Κάθε πλειάδα στον τελεστή $\mathbf{Op_i}$ καταναλώνει $\mathbf{Op_i \cdot r}$ πόρους (resources) – υπολογιστική ισχύ, μνήμη, εύρος ζώνης δικτύου – από τους $\mathbf{Op_i \cdot R}$ πόρους που διαθέτει συνολικά ο τελεστής. Άρα ο κάθε τελεστής μπορεί να επεξεργαστεί $\mathbf{Op_i \cdot R / Op_i \cdot r}$ πλειάδες ανά μονάδα χρόνου με μέσο χρόνο απόκρισης για κάθε πλειάδα $\mathbf{T_s = Op_i \cdot r / Op_i \cdot R}$. Για κάθε πλειάδα ο τελεστής θα παράγει **ρ** πλειάδες αποτελέσματος, το οποίο αποτελεί την επιλεξιμότητα του τελεστή, και, αφού την επεξεργαστεί πρέπει να αποφασίσει σε ποιον άλλο τελεστή θα την προωθήσει. Οι τελεστές που κάνουν τις περισσότερες προωθήσεις και άρα δεν επιστρέφουν τις πλειάδες άμεσα στο στρόβιλο, μπορούν να θεωρηθούν και ως αυτοί με τις μεγαλύτερες επιλεξιμότητες και να αυξηθεί η πιθανότητα δρομολόγησης των πλειάδων σε αυτούς. Με αυτές τις μετρικές είναι δυνατόν να ληφθούν δυναμικά εκτιμήσεις για τις καλύτερες δυνατές αποφάσεις όσον αφορά τα δρομολόγια εκτέλεσης (execution paths) της κάθε πλειάδας – που είναι και το ζητούμενο.

Οι κλάδοι ευρετηρίων (SteMs[11]) [40], που συχνά αναφέρονται στο ίδιο πλαίσιο με τους στροβίλους (eddies) είναι ευρετήρια δημιουργημένα στη στιγμή (on the fly) πάνω στις εισερχόμενες ροές ενός συστήματος ανάλογα με τις ανάγκες των εκάστοτε επερωτήσεων. Είναι και αυτοί, δηλαδή, τελεστές που περιέχουν από ένα ευρετήριο ο καθένας πάνω σε ένα συγκεκριμένο πεδίο μιας ροής.

11 Η λέξη SteMs είναι ακρωνύμιο του State Modules. Η λέξη stem εκτός των άλλων σημαίνει και κλαδί. Γι' αυτό και η απόδοση "κλάδοι ευρετηρίων".

Με αυτό τον τρόπο μία ροή μπορεί να έχει πολλούς κλάδους ευρετηρίων. Για παράδειγμα μια ροή S με πεδία [a, b, c, ...] μπορεί να έχει ένα κλάδο ευρετηρίου με πίνακα σύνοψης πάνω στο S.a και άλλον ένα με B-δέντρο πάνω στο S.b κοκ. Η επιλογή του τύπου του ευρετηρίου μπορεί να γίνει ανάλογα με τις ανάγκες των επερωτήσεων και το φόρτο που δέχεται ανά πάσα στιγμή το σύστημα. Ένας κλάδος ευρετηρίου σε συνδυασμό με ένα στρόβιλο [39] δέχεται πλειάδες από το στρόβιλο για να τις ευρετηριάσει και τις επιστρέφει σε αυτόν. Στη συνέχεια ο στρόβιλος χρησιμοποιεί τους κλάδους σαν οποιουσδήποτε άλλους τελεστές στέλνοντας και παίρνοντας πλειάδες από αυτούς για να εκτελέσει τις επερωτήσεις που έχει αναλάβει. Για παράδειγμα μπορεί να χρησιμοποιήσει τον κλάδο από το πεδίο S.a της ροής S για να την συνενώσει με μια άλλη ροή και τον κλάδο από το πεδίο S.b για να ικανοποιήσει μια συνθήκη στο αποτέλεσμα της συνένωσης. Στο Σχήμα 3.10 φαίνεται ένας τέτοιος συνδυασμός στροβίλου με κλάδους ευρετηρίων.



Σχήμα 3.10 – Στρόβιλος και Κλάδοι Ευρετηρίων

3.5 ΕΙΔΙΚΆ ΘΈΜΑΤΑ ΚΑΤΆ ΤΗ ΒΕΛΤΙΣΤΟΠΟΊΗΣΗ

Οι προκλήσεις που καλείται να αντιμετωπίσει ένα ΣΔΡΔ εμπεριέχουν δύο εξαιρετικά σημαντικές παραμέτρους από τις οποίες πηγάζουν οι πλειοψηφία των των προβλημάτων στην επεξεργασία:

- Τους **αυθαίρετους** και πιθανά πολύ **μεγάλους** ρυθμούς των δεδομένων και

- Τη συνεχή και **ατέρμονη** φύση των ροών που τις κάνουν πρακτικά **απεριόριστες**.

Οι δύο αυτές παράμετροι δεν είναι δυνατόν να παρακαμφθούν με κανέναν τρόπο, όπως ήδη έχουμε δει, παρά μόνο να εξομαλυνθούν κάπως όπου αυτό είναι δυνατόν. Οι εφαρμογές, όμως, που έχουν ανάγκη από την εκμετάλλευση των δεδομένων σαν ροές δεν είναι δυνατόν να απαιτηθεί να μειώσουν τις απαιτήσεις τους όσον αφορά τα μεγέθη των δεδομένων και τα ύψη των ρυθμών μεταφοράς. Αντιθέτως, αυτές ολοένα και αυξάνονται. Παρολαυτά, σε πολλές περιπτώσεις όπου τα μεγέθη και οι απαιτήσεις θα μπορούσαν να χαρακτηριστούν δυσθεώρητα, οι ανάγκες πολλών συστημάτων μπορούν να ικανοποιηθούν με προσεγγιστικές απαντήσεις και μεθοδολογίες. Κάτι τέτοιο αποσκοπεί στην αντιστάθμιση των μεγάλων φόρτων μια κατά το δυνατόν μικρή ελάττωση στην ακρίβεια των αποτελεσμάτων. Στη συνέχεια, θα δούμε μερικές ειδικές τεχνικές που έχουν προταθεί στη σύγχρονη βιβλιογραφία για να αντιμετωπίσουν αυτό το γεγονός και βασίζονται στην ελάττωση του μεγέθους των δεδομένων, την εκμετάλλευση της σημασιολογίας τους και την εύρεση προσεγγιστικών απαντήσεων.

3.5.1 ΦΙΛΤΡΆΡΙΣΜΑ ΡΟΏΝ (STREAM FILTERING)

Όταν ένα ΣΔΡΔ βρίσκεται αντιμέτωπο με τεράστιους όγκους δεδομένων τους οποίους εκτιμά ότι δε θα καταφέρει να επεξεργαστεί, τότε – προτού καταρρεύσει – μπορεί να επιλέξει να μειώσει το μέγεθος των δεδομένων που τελικά επεξεργάζεται σε επίπεδα που είναι ικανό να διαχειριστεί τη δεδομένη χρονική περίοδο. Ένας τρόπος για να γίνει αυτό είναι το φιλτράρισμα των εισερχόμενων ροών [41]. Το φιλτράρισμα ανακόπτει ένα μέρος της ροής στην οποία εφαρμόζεται, σαν ένα κλασσικό φίλτρο που επιτρέπει σε κάποια από τα δεδομένα να περάσουν και σε άλλα όχι, προσδοκώντας να μειώσει τον όγκο χωρίς να αλλοιώσει σημαντικά την ποιότητα των απαντήσεων των επερωτήσεων. Το φιλτράρισμα είναι επιθυμητό να υλοποιείται με απλό τρόπο για να μην επιφέρει υπερβολικό επεξεργαστικό φόρτο ή μεγάλες καθυστερήσεις κατά την εφαρμογή του. Μπορεί, δηλαδή, να είναι μια επιλογή ή μία προβολή πάνω σε μια ροή που φιλτράρει τις πλειάδες της βάσει κάποιας συνθήκης. Το σημείο της εφαρμογής του μπορεί να είναι είτε στην είσοδο του συστήματος επεξεργασίας είτε σε κάποιο σημείο μέσα σε αυτό – πχ, ανάμεσα σε κάποιους τελεστές – είτε στην ίδια την πηγή της ροής.

Ένας τρόπος να γίνει το φιλτράρισμα είναι η συγχώνευση των δεδομένων (data merging) σε μεγέθη διαχειρίσιμα από το σύστημα επεξεργασίας με τέτοιο τρόπο ώστε οι τελικές επερωτήσεις να εξακολουθούν να μπορούν να υπολογιστούν. Για παράδειγμα, σε ένα σύστημα παρακολούθησης δικτύου (network monitoring) μια χρήσιμη επερώτηση είναι οι k πιο ενεργές συνδέσεις στο δίκτυο. Σε αυτό το σενάριο το σύστημα δε χρειάζεται να διατηρεί το ιστορικό όλων των ροών δεδομένων που διέρχονται μέσα από μία σύνδεση ανάμεσα σε δύο κόμβους του δικτύου για να ικανοποιήσει τη

συγκεκριμένη επερώτηση. Διατηρώντας μόνο συνολικές τιμές (aggregate) των μεγεθών των ροών και συνδυάζοντάς τες είναι ικανό να απαντήσει και μάλιστα χωρίς να χάσει σε ακρίβεια. Βέβαια, άλλες ανάγκες, όπως η ανίχνευση εισβολών (intrusion detection) στο δίκτυο δεν είναι δυνατόν να εκτιμηθούν με παρόμοιες προσεγγίσεις, μιας και χρειάζονται να εφαρμόσουν αλγορίθμους που σχετίζονται με τη σημασιολογία των πακέτων.

Ακόμη, έχουμε ήδη αναφέρει την τεχνική της ελάττωσης φόρτου (load shedding) σε προηγούμενο κεφάλαιο κι έχουμε αγγίξει τις ιδιαιτερότητές της. Αποτελεί ένα σίγουρο τρόπο μείωσης του όγκου των δεδομένων και η αποτελεσματικότητά της έγκειται στο σωστό μοντέλο απόφασης σχετικά με το πότε το πού και το κατά πόσο θα εφαρμοστεί συναρτήσει πάντοτε του ποσοστού στο οποίο αλλοιώνει το αποτέλεσμα. Ένας τρόπος να γίνει αυτό είναι η αφελής (naïve) προσέγγιση, η οποία χωρίς να δίνει σημασία στο περιεχόμενο των δεδομένων εκτελεί την απόρριψη πλειάδων δίχως μεγάλη περίσκεψη. Η αφελής απόρριψη μπορεί να γίνει, για παράδειγμα με τρόπο τυχαίο (randomly) ή ομοιόμορφο (uniformly). Η μέθοδος αυτή μπορεί να φανεί χρήσιμη σε περιπτώσεις όπου το σύστημα υποφέρει από μεγάλο φόρτο και οι διαθέσιμες εναλλακτικές έχουν εξαντληθεί. Από την άλλη πλευρά οι περισσότερο εκλεπτυσμένες μέθοδοι φιλτραρίσματος εισάγουν και κάποιο κριτήριο σχετικά με τον τρόπο με τον οποίο θα γίνει το φιλτράρισμα λαμβάνοντας υπόψη τον αντίκτυπο στην ποιότητα του αποτελέσματος [34].

3.5.2 ΕΠΙΔΡΑΣΗ ΣΤΊΞΗΣ ΠΆΝΩ ΣΤΙΣ ΡΟΈΣ (PUNCTUATION)

Η επίδραση πλεονάζουσας σημασιολογίας πάνω στις εισερχόμενες ροές είναι μία τεχνική που αποσκοπεί στην απάλυνση του φόρτου του συστήματος επεξεργασίας δίνοντάς του επιπλέον πληροφορία για τη ροή, η οποία είναι ενσωματωμένη πάνω στην ίδια τη ροή. Η πιο χαρακτηριστική μορφή στίξης στις ροές είναι αυτή που δίνει ενδείξεις για την ολοκλήρωση συγκεκριμένων σημασιολογικών υπό-ομάδων δεδομένων μέσα στη ροή. Τυπικά ορίζεται [42] ότι αν S είναι μία ροή, S[i], i=1, 2, ..., οι πλειάδες της, S[i→j] είναι όλες οι πλειάδες της από την i+1 μέχρι και την j και match(p, S[i_0]) είναι μία λογική συνάρτηση που επιστρέφει αλήθεια όταν η πλειάδα S[i_0] ικανοποιεί τη συνθήκη στίξης p (punctuation) τότε η συγκεκριμένη στίξη διαχωρίζει ουσιαστικά τις πλειάδες σε αυτές που την ικανοποιούν και σε αυτές που δεν την ικανοποιούν. Η πλειάδα ορόσημο S[i_0] στην οποία η συνθήκη στίξης ικανοποιείται σημαίνει ότι για κάθε S[i_0→j] με $i_0 < j$ ικανοποιείται η συνθήκη και ότι για κάθε S[i_0→j] με $i_0 > j$ δεν ικανοποιείται η συνθήκη.

Για να γίνει αυτό κατανοητό ας δούμε το εξής παράδειγμα: Έστω ένα σύστημα που παρακολουθεί ένα δίκτυο αισθητήρων (sensor network) κι έστω μια επερώτηση που, συγκεντρώνοντας ανά τακτά

διαστήματα όλες τις θερμοκρασίες από τους αισθητήρες που δίνουν αυτή τη μέτρηση, επιστρέφει τη μέγιστη ανά ώρα θερμοκρασία. Η επερώτηση αυτή είναι συνεχής και μέσα σε αυτή την ιδιότητα κρύβεται μια εγγενής δυσκολία. Η επερώτηση ουσιαστικά περιέχει έναν τελεστή ομαδοποίησης - “ανά ώρα” - ο οποίος υπό κανονικές συνθήκες οφείλει να γνωρίζει το σύνολο των δεδομένων για να παράγει την απαιτούμενη ομαδοποίηση. Αυτό είναι, βέβαια, ανέφικτο στο παραπάνω σενάριο. Εναλλακτικά, αν δε γνωρίζει το σύνολο των δεδομένων, οφείλει να σπαταλήσει πόρους για να συμπεράνει την ομαδοποίηση. Μία λύση που θα ενσωμάτωνε την πληροφορία αυτή της ομαδοποίησης θα διευκόλυνε άμεσα την επερώτηση: εισάγοντας ως ειδική στίξη ανάμεσα στις πλειάδες την πληροφορία ολοκλήρωσης της κάθε ώρας, ώστε ο τελεστής να την γνωρίζει και να μη χρειάζεται να την υπολογίσει. Αν οι πλειάδες ακολουθούν τη μορφοποίηση [s_id, Θ, t], όπου s_id το αναγνωριστικό του αισθητήρα που στέλνει τη μέτρηση, Θ η θερμοκρασία και t η χρονική στιγμή της μέτρησης, τότε μια σειρά πλειάδων της μορφής:

{... , [8302, 35, 12:45], [1023, 36, 12:45], [1023, 36, 13:00], [8304, 34, 13:15], ...}

θα μπορούσε να γίνει:

{... , [8302, 35, 12:45], [1023, 36, 12:45], [1023, 36, 13:00], [*, *, 12:00], [8304, 34, 13:15], ...}

αναδεικνύοντας έτσι το σημείο στο οποίο η 12η ώρα ολοκληρώθηκε και υποβοηθώντας το πλάνο εκτέλεσης της επερώτησης, μιας και του υποδεικνύει το σημείο της ομαδοποίησης.

Ένα σημαντικό ζήτημα που εμφανίζεται κατά την εφαρμογή των στίξεων είναι το πώς επηρεάζεται η συμπεριφορά των διαφόρων τελεστών που παίρνουν μέρος στις επερωτήσεις. Για παράδειγμα, απλοί τελεστές επιλογής/προβολής (select) επηρεάζονται μόνο από το γεγονός ότι οφείλουν να αναμεταδίδουν τις στίξεις που ανιχνεύουν στους τελεστές που τους ακολουθούν. Από την άλλη, τελεστές ομαδοποίησης (group by) που εκμεταλλεύονται σχετικές στίξεις έχουν τη δυνατότητα να απελευθερώνουν τους πόρους που έχουν δεσμεύσει όταν η ολοκλήρωση των ομάδων καθορίζεται από τη στίξη, κάτι ιδιαίτερα σημαντικό. Τέλος, τελεστές συνένωσης πρέπει να είναι πολύ προσεκτικοί όσον αφορά την εκμετάλλευση των στίξεων μιας και οι πράξεις που επιτελούν αναφέρονται σε περισσότερες από μία ροές και πρέπει να διαφυλάσσουν τη γνώση της φάσης στην οποία βρίσκεται η συνένωση όταν πλειάδες στίξης καταφθάνουν.

3.5.3 ΧΡΉΣΗ ΠΛΑΙΣΊΩΝ (WINDOWS)

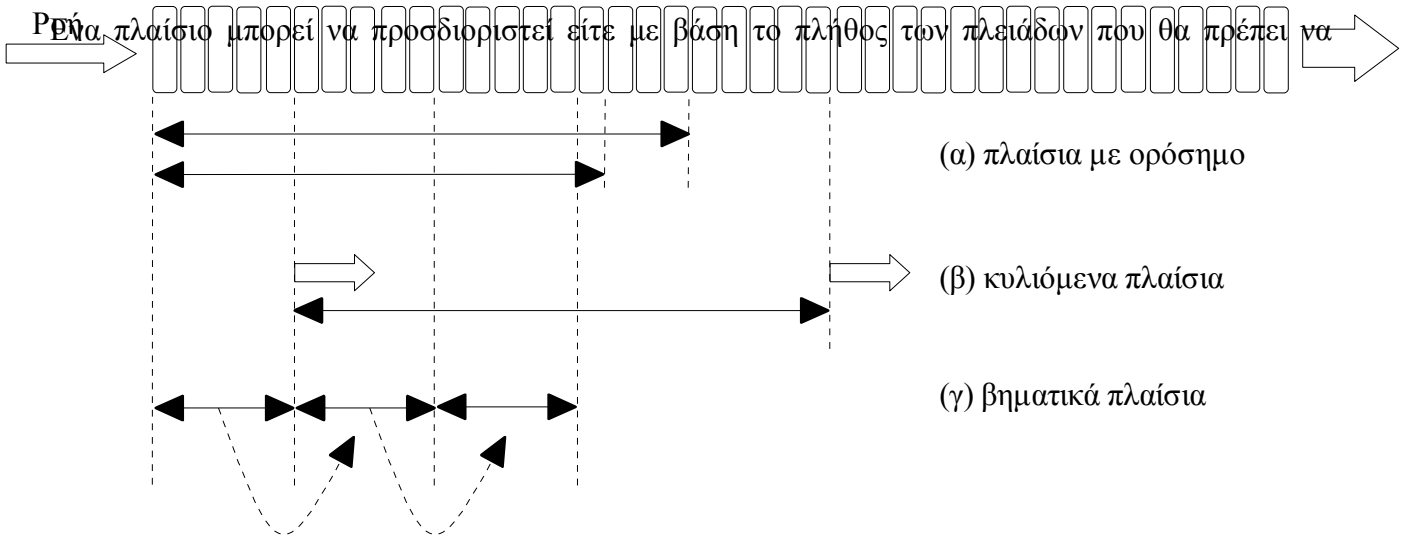
Η χρήση πλαισίων (windows) έχει ήδη αναφερθεί στο πρώτο κεφάλαιο στην ενότητα για το μοντέλο της ροής δεδομένων. Θυμίζουμε ότι ένα πλαίσιο σε ένα ΣΔΡΔ είναι μια περιορισμένου εύρους όψη μιας ροής. Η ανάγκη για τη χρήση πλαισίων εμφανίζεται τόσο εξαιτίας των προβλημάτων που εμφανίζονται λόγω των μεγάλων όγκων των ροών αλλά και εξαιτίας της ιδιαίτερης σημασιολογίας που διαθέτουν. Εφαρμόζοντας πλαίσια, από τη μία πλευρά, περιορίζονται οι πόροι που δεσμεύονται και, από την άλλη, ικανοποιούνται τελεστές και κατά συνέπεια επερωτήσεις που δεν έχουν ανάγκη από τη διατήρηση στη μνήμη μεγάλων ποσοτήτων των ροών.

Υπάρχουν τριών ειδών πλαίσια στη διαχείριση των ροών δεδομένων [11][41] που αναπαρίστανται όλα μαζί στο Σχήμα 3.11, α, β και γ:

- Με **ορόσημο** (landmark windows): Είναι πλαίσια που έχουν κοινό σημείο εκκίνησης αλλά ανεξάρτητο μεταξύ τους σημείο τέλους (α).

- **Κυλιόμενα** (sliding windows): Είναι πλαίσια πάνω στα οποία η ίδια η ροή ολισθαίνει συνεχώς και έτσι το πλαίσιο διατρέχει τις πλειάδες της μία προς μία (β). Μπορούν να θεωρηθούν σαν μια FIFO λίστα όπου πλειάδες εισέρχονται και εξέρχονται.

- **Βηματικά** (tumbling windows): Είναι πλαίσια που δεν έχουν καμία κοινή πλειάδα μεταξύ τους επειδή διατρέχουν τη ροή σε ενιαίες ποσότητες (γ).



Σχήμα 3.11 – Είδη πλαισίων (windows)

Όπως έχει ήδη σημειωθεί η έννοια του πλαισίου είναι εκ των πραγμάτων απαραίτητη σε ένα ΣΔΡΔ. Μία ροή δεδομένων έχει απεριόριστες θεωρητικά πλειάδες κι επομένως η διαχείρισή της θα γίνεται όπως και να 'χει σε δομές που εμπεριέχουν την έννοια του πλαισίου τουλάχιστο όσον αφορά την περιορισμένη όψη της ροής που μπορεί ένα σύστημα επεξεργασίας να έχει με τους πεπερασμένους του πόρους. Συχνά, κάτι τέτοιο συναντά και τις σημασιολογικές ιδιαιτερότητες των ροών. Για παράδειγμα, σε πολλά περιβάλλοντα δεν υπάρχει η ανάγκη για τήρηση στη μνήμη μεγάλου πλήθους παρελθουσών πλειάδων. Το ουσιαστικό πρόβλημα τίθεται στο πώς θα μπορέσει το σύστημα να εφαρμόσει τα πλαίσια, σε συνδυασμό με τις άλλες τεχνικές με τις οποίες είναι εξοπλισμένο, για να πετύχει τον επιθυμητό στόχο, δηλαδή την αποτελεσματικότητα.

3.5.4 ΣΥΝΌΨΕΙΣ (SYNOPSES)

Στο χώρο της βελτιστοποίησης και συγκεκριμένα της αποδοτικής διαχείρισης της διαθέσιμης μνήμης σε συνδυασμό με την ελαχιστοποίηση του επεξεργαστικού φόρτου έχει αναδυθεί ακόμη μία τεχνική που αποσκοπεί να αντιμετωπίσει τα δύο αυτά καίρια ζητήματα. Οι συνόψεις (synopses και μερικές φορές digests και sketches) είναι προσεγγίσεις που δομούν μια ροή με τρόπο περιληπτικό, έτσι ώστε να καταλαμβάνει λιγότερη διαθέσιμη μνήμη [4][41] και να απαιτεί λιγότερους επεξεργαστικού πόρους [14]. Το κόστος αυτής της προσέγγισης είναι η υποβάθμιση της ποιότητας και της ακρίβειας των αποτελεσμάτων των επερωτήσεων, εφόσον οι πλειάδες των ροών δεν είναι οι ίδιες εξολοκλήρου διαθέσιμες παρά μόνο συνόψεις ή περιλήψεις αυτών. Ένας απλός τρόπος, που έχουμε ήδη αγγίξει, για να δομηθεί μία σύνοψη μιας ροής είναι η δειγματοληψία πλειάδων πάνω στη ροή. Εκεί, ανάλογα με τη συχνότητα και με την πολιτική της δειγματοληψίας – αφελής (naïve) ή σημασιολογική (semantic) – η σύνοψη αντανακλά περισσότερο ή λιγότερο την πραγματική ποιότητα των πλειάδων της ροής, κάτι που διαχέεται και στην ποιότητα του αποτελέσματος των επερωτήσεων. Παρολαυτά, σε συστήματα όπου υπάρχει ανάγκη για ανάλυση τάσεων (trend analysis), όπως οικονομικές εφαρμογές, ή για ανίχνευση ανωμαλιών, όπως στην παρακολούθηση δικτύων οι συνόψεις βρίσκουν άμεση εφαρμογή, μιας και εκεί η ακρίβεια δεν είναι η σημαντικότερη προτεραιότητα. Ακόμη, μπορούμε να συμπληρώσουμε, μία σύνοψη μπορεί να τηρηθεί σε διαφορετικά σημεία του συστήματος επεξεργασίας: στην είσοδό του ή στην είσοδο κάποιου τελεστή του ανάλογα με τις απαιτήσεις του πλάνου εκτέλεσης.

Το μαθηματικό μοντέλο στο οποίο έχουν βασιστεί οι περισσότερες από τις σύγχρονες προτάσεις που αφορούν στις τυχαίες (random) συνόψεις περιγράφεται στο [43] και ονομάζεται τυχαιοκρατική σύνοψη με σκαριφήματα (randomized sketching). Βασίζεται στην εκτίμηση του μεγέθους της αυτο-συνένωσης (self-join) μιας σχέσης και επεκτείνεται στις πολλαπλές συνενώσεις. Συνοπτικά

ακολουθεί την εξής λογική:

- **$S = (x_1, x_2, \ldots, x_n)$**, μία σειρά πλειάδων της ροής,
- **$D = \{1 .. d\}$**, το πεδίο ορισμού των x_i,
- **$m_i = |j$ τ.ω. $x_j = i|$,** το πλήθος των εμφανίσεων του i στην S,
- **$F_k = \Sigma_{i=1..n}(m_i^k)$**, η k-οστή στιγμιαία συχνότητα (frequency moment) της S. Το F_0 είναι το πλήθος των διαφορετικών[12] τιμών της S, το F_1 είναι το μήκος – σε πλειάδες – της S και το F_2 είναι το μέγεθος της αυτο-συνένωσής της.

Η τεχνική των σκαριφημάτων αντιστοιχεί μέσω συνάρτησης σύνοψης (hash function) κάθε στοιχείο του πεδίου ορισμού D σε μία τιμή z_i στο $\{-1, +1\}$. Ορίζει ως $\mathbf{X = \Sigma_D(m_i z_i)}$ και εκτιμά την τιμή του $\mathbf{F_2}$ σε $\mathbf{X^2}$. Αν οι τιμές z_i είναι ανά τετράδες στατιστικά ανεξάρτητες (four-way independent) αποδεικνύεται ότι η αναμενόμενη τιμή $\mathbf{E(X^2)}$ του $\mathbf{X^2}$ είναι $\mathbf{F_2}$ και η διακύμανση του, $\mathbf{V(X^2)}$, είναι $\mathbf{2F_2^2}$. Η μεθοδολογία αυτή μπορεί να επεκταθεί από την αυτο-συνένωση και στις συνενώσεις δύο σχέσεων αλλά και στις πολλαπλές συνενώσεις.

Στη συνέχεια, τα ιστογράμματα (histograms) αποτελούν ακόμη μία σημαντική μέθοδο δημιουργίας περιληπτικών δομών και αποσκοπούν κυρίως στην απεικόνιση των κατανομών των τιμών σε σύνολα δεδομένων, τόσο σε βάσεις δεδομένων αλλά και σε ροές [44]. Χρησιμοποιούνται συχνά για την εκτίμηση του κόστους και την προσεγγιστική απάντηση των επερωτήσεων. Μπορούμε να διακρίνουμε τρεις κυρίαρχες κατηγορίες ιστογραμμάτων:

- **V-βέλτιστα (V-optimal)**: Προσεγγίζουν την κατανομή ενός συνόλου τιμών $v_1, v_2, \ldots, v_n$ με μία βηματική συνάρτηση u(i) έτσι ώστε να ελαχιστοποιείται το άθροισμα των τετραγώνων των αποκλίσεων της u(i) από τις πραγματικές τιμές, δηλαδή το $\Sigma_{i=1..n}(v_i - u(i))^2$.
- **Ισοκατανεμημένα (equi-width)**: Τμηματοποιούν το πεδίο ορισμού του συνόλου δεδομένων σε ενότητες (buckets) με τέτοιο τρόπο ώστε η κατανομή των τιμών κατά μήκος των ενοτήτων αυτών να είναι ομοιόμορφη.

12 Το πλήθος των διαφορετικών τιμών μιας σχέσης ή μιας ροής (F_0) ήταν ανέκαθεν σημαντικό στις εφαρμογές βάσεων δεδομένων και ιδιαίτερα στις περιπτώσεις όπου έπρεπε να γίνουν υπολογισμοί με ένα μόνο πέρασμα στα δεδομένα [14]. Το τελευταίο ισχύει, φυσικά, στις ροές δεδομένων.

- **Πολωμένα (end-biased)**: Αυτά τα ιστογράμματα κρατούν ακριβείς μετρήσεις για τα δεδομένα που εμφανίζονται πολύ συχνά, δηλαδή από μία συχνότητα και πάνω – και προσεγγιστικές με ομοιόμορφη κατανομή για τις υπόλοιπες.

Συνοπτικές περιγραφές των δεδομένων, ακόμη, δίνονται και μέσω της τεχνικής των κυματιδίων (wavelets). Τα κυματίδια εμφανίζονται σαν συντελεστές που συνθέτουν την απεικόνιση-μετασχηματισμό (transformation) ενός δοσμένου συνόλου δεδομένων [45]. Αποτελούν ένα μαθηματικό εργαλείο για την ιεραρχική αποδόμηση – ή από άλλη σκοπιά τη συμπίεση – συνόλων δεδομένων [46]. Είναι γεγονός ότι σε πολλές περιπτώσεις ένα σύνολο δεδομένων μπορεί να απεικονισθεί προσεγγιστικά αν επιλεγεί να διατηρηθεί ένα δοθέν ποσοστό ακρίβειας της απεικόνισης. Τότε το πλήθος των συντελεστών που διατηρούνται στη δομή οδηγεί στην αύξηση ή την ελάττωση της ακρίβειας. Με αυτό τον τρόπο, η ακρίβεια είναι δυνατόν να επιτευχθεί σε ένα μεγάλο βαθμό αν εκμεταλλευτεί κανείς ένα μικρό ποσοστό των ίδιων των δεδομένων, τα οποία μπορούν να θεωρηθούν μείζονος σημασίας, απορρίπτοντας ένα μεγάλο ποσοστό τους που δε συνεισφέρει ιδιαίτερα. Χαρακτηριστικό ανάλογο είναι οι μετασχηματισμοί που εφαρμόζονται στην επεξεργασία σημάτων, όπως, για παράδειγμα ο διακριτός μετασχηματισμός Fourier (discrete Fourier transform – DFT). Όταν αυτός εφαρμόζεται στον ήχο τότε είναι γνωστό το πλήθος των συντελεστών που πρέπει κανείς να διατηρήσει για να κρατήσει τον απεικονιζόμενο ήχο αναλλοίωτο. Διαφορετικά είναι γνωστό το πώς η ποιότητα υποβαθμίζεται καθώς η πυκνότητα των συντελεστών ελαττώνεται. Μια από τις επιθυμητές ιδιότητες των κυματιδίων είναι να μπορεί κανείς να αποκαταστήσει με όσο το δυνατόν μεγαλύτερη ακρίβεια τα δεδομένα χρησιμοποιώντας όσο το δυνατόν λιγότερους συντελεστές.

Τα απλούστερα κυματίδια (Haar wavelets) λειτουργούν ως εξής: Ας θεωρήσουμε ότι έχουμε ένα σήμα – ή μία ροή – S = [2, 2, 0, 2, 3, 5, 4, 4], πλήθους N=8. Σαν πρώτο βήμα συμπιέζουμε το σήμα παίρνοντας τη μέση τιμή των στοιχείων του ανά δύο καταλήγοντας στο [2, 1, 4, 4] και χάνοντας ήδη ένα τμήμα της αρχικής πληροφορίας. Για να μπορεί να ανασυσταθεί η αρχική πληροφορία θα πρέπει να τηρήσουμε κάποιους συντελεστές λεπτομέρειας (detail coefficients). Τα κυματίδια Haar αποθηκεύουν σαν τέτοιους συντελεστές τις διαφορές των αρχικών στοιχείων ανά δύο, δίνοντας έτσι [0, -1, -1, 0]. Η διαδικασία αυτή της μείωσης του μεγέθους στο μισό διατηρώντας τις μέσες τιμές και τις διαφορές ανά δύο των αρχικών στοιχείων οδηγεί στον παρακάτω πίνακα:

Πλήθος	***Μέσες Τιμές***	***Συντελεστές Λεπτομέρειας***
8	[2, 2, 0, 2, 3, 5, 4, 4]	-

Πλήθος	***Μέσες Τιμές***	***Συντελεστές Λεπτομέρειας***
4	[2, 1, 4, 4]	[0, -1, -1, 0]
2	[1.5, 4]	[0.5, 0]
1	[2.75]	[-1.25]

Ο μετασχηματισμός κυματιδίου τότε είναι η τελευταία μοναδική μέση τιμή και το σύνολο των συντελεστών λεπτομέρειας. Όλη η πληροφορία είναι διαθέσιμη αν αυτό είναι επιθυμητό, διαφορετικά μπορεί να επιλεγεί το πλήθος των συντελεστών που θα διατηρηθεί. Για να μειωθεί το μέγεθος των τελεστών αυτοί κανονικοποιούνται κάτι που σε πολλές περιπτώσεις τους κάνει πολύ μικρούς σε μέγεθος. Μπορούμε να παρατηρήσουμε πως αν το σήμα δεν έχει πολύ μεγάλες και απότομες εναλλαγές στις τιμές του και άρα οι μέσες τιμές και οι διαφορές αντίστοιχα είναι μικρές τότε η συμπύκνωση που επιτυγχάνεται με τα κυματίδια είναι αποδοτικότερη[13].

Βλέπουμε, λοιπόν, πως εξαιτίας των ιδιαιτεροτήτων που έχουν οι ροές δεδομένων και των περιορισμένων πόρων που διαθέτουν τα συστήματα επεξεργασίας υπάρχει αδήριτη η ανάγκη για μεθοδολογίες που να διαχειρίζονται και να απαλύνουν το φόρτο. Πολλές φορές επιλέγεται κάποιος συμβιβασμός ανάμεσα στην ποιότητα του αποτελέσματος των επερωτήσεων που ένα ΣΔΡΔ έχει την ικανότητα να υπολογίσει και την αποδοτική διαχείριση των πόρων του. Τα αποτελέσματα μπορούν υπό συνθήκες να είναι ικανοποιητικά και η έρευνα στο χώρο αυτό συνεχίζεται. Σε κάθε περίπτωση αυτό που διαφαίνεται να έχει τεράστια σημασία είναι η συνέπεια και η συνέχεια του ίδιου του συστήματος που οφείλει να χρησιμοποιεί ό,τι μέσα έχει στη διάθεσή του για να εξασφαλίσει τη διαθεσιμότητά του. Αναφέρθηκε ένα σημαντικό μέρος της σύγχρονης έρευνας γύρω από την εκτέλεση των συνεχών επερωτήσεων και τα θέματα σχετικά με τη βελτιστοποίησή τους και τη διαχείριση των πόρων. Τα αναμενόμενα, όμως, μεγέθη των ροών δεδομένων, όπως αυτό συμπεραίνεται από τα μέσα που η τεχνολογία προετοιμάζει για το κοντινό μέλλον, και οι απαιτήσεις σε επεξεργασία κάνουν εύκολα κατανοητή την παρατήρηση πως αν ένα σύστημα διαχείρισης ροών δεδομένων δεν είναι εξαιρετικά εξειδικευμένο στον τομέα του ή εντελώς σίγουρο εκ των προτέρων για τις παραμέτρους της επεξεργασίας, τότε το μόνο βέβαιο είναι ότι θα πρέπει να μπορεί να προσαρμόζεται στο περιβάλλον και τις συνθήκες επεξεργασίας που ανακύπτουν κάθε φορά με καλά ορισμένες προσεγγίσεις.

13 Κάτι τέτοιο είναι αναμενόμενο, μιας και ένα σήμα με μικρές εναλλαγές φέρει μικρή ποσότητα πληροφορίας – από τη Θεωρία Πληροφορίας.

4. ΕΦΗΡΜΟΣΜΈΝΗ ΈΡΕΥΝΑ ΣΤΙΣ ΡΟΈΣ ΔΕΔΟΜΈΝΩΝ

Στην ενότητα αυτή θα παρουσιαστούν συνοπτικά προσπάθειες που έχουν γίνει σε ερευνητικό πάντα επίπεδο στο χώρο της Διαχείρισης Ροών Δεδομένων. Έχουν υπάρξει αρκετές ερευνητικές απόπειρες δημιουργίας ενός πλήρους ΣΔΡΔ όπου η κάθε μία εφαρμόζει πολλές από τις τεχνικές και τις μεθοδολογίες που αναφέρθηκαν στα προηγούμενα κεφάλαια. Οι περισσότερες από αυτές έχουν αξιοποιηθεί σαν συστήματα πάνω στα οποίες νέες τεχνικές και αρχιτεκτονικές αναπτύσσονται και δοκιμάζονται και όχι σαν συστήματα με ορατή εξειδίκευση σε κάποιο τομέα. Ακόμη, έχουν προταθεί ειδικές γλώσσες δημιουργίας συνεχών επερωτήσεων κατά το μοντέλο των επερωτήσεων των παραδοσιακών βάσεων δεδομένων.

4.1 ΓΛΏΣΣΕΣ ΣΥΝΕΧΏΝ ΕΠΕΡΩΤΉΣΕΩΝ

Στην ενότητα αυτή θα παρουσιαστούν συνοπτικά μερικές από τις γλώσσες συνεχών επερωτήσεων που έχουν προταθεί σε ερευνητικό επίπεδο τα τελευταία χρόνια. Καμία από αυτές δε βρίσκεται ακόμη σε σημείο όπου να είναι κοντά σε προτυποποίηση (standard), μιας και όλες έχουν προκύψει από ερευνητικά προγράμματα. Θα δούμε συγκεκριμένα τη CQL [47] και την ESL [52] με το σκεπτικό ότι η κάθε μία από αυτές έχει να προτείνει κάτι νέο και διαφορετικό από τις υπόλοιπες σε συνδυασμό με την ευρύτητα τη βαρύτητα με την οποία διαφαίνεται να αναφέρονται και να σχολιάζονται στη σύγχρονη βιβλιογραφία.

4.1.1 Η ΓΛΏΣΣΑ CQL (CONTINUOUS QUERY LANGUAGE)

Η γλώσσα συνεχών επερωτήσεων CQL (Continuous Query Language) [47][48] στηρίζεται σε ένα συγκεκριμένο σημασιολογικό υπόβαθρο και προσδιορίστηκε στα πλαίσια του ερευνητικού προγράμματος για τη δημιουργία ενός συστήματος διαχείρισης ροών δεδομένων STREAM [49] και υλοποιήθηκε ως ένα βαθμό σε αυτό. Βασίζεται σε έναν τυπικό καταρχάς ορισμό των ροών και των σχέσεων (relations)[14], τις συσχετίσεις μεταξύ τους και μια δηλωτική γλώσσα επερωτήσεων του σχεσιακού μοντέλου – την SQL. Οι δημιουργοί της θεωρούν ότι η απλή αναπαραγωγή μιας γνωστής γλώσσας επερωτήσεων σε εφαρμογές ροών δε θα ήταν επαρκής, παρά μόνο σε ένα ελάχιστο υποσύνολο των επερωτήσεων που θα ήθελε κανείς να έχει τη δυνατότητα να θέσει, όπως, για παράδειγμα, απλές επιλογές/προβολές με κάποια συνθήκη. Στην περίπτωση, όμως, περισσότερο σύνθετων επερωτήσεων με πλαίσια (windows), ανάμιξη σχέσεων με ροές, εμφωλιασμένων επερωτήσεων (nested queries) κλπ, η αποτύπωση της "συνέχειας" γίνεται πιο περίπλοκη. Η CQL

14 Ο όρος σχέση (relation) αναφέρεται με την κλασσική έννοια της σχέσης στο σχεσιακό μοντέλο, δίνοντας έμφαση στην αντιδιαστολή του όρου στις παραδοσιακές βάσεις δεδομένων σε σχέση με τις ροές δεδομένων.

επιχειρεί να χρησιμοποιήσει την ευρέως διαδεδομένη σχεσιακή σημασιολογία με τέτοιο τρόπο ώστε να επιτύχει μία γλώσσα άμεση και εύληπτη. Επιπλέον, η γλώσσα αυτή θα πρέπει να είναι συμπαγής και με δηλωτικό τρόπο – όπως η SQL – και να εκφράζει με ακρίβεια την επιθυμητή ενέργεια.

Προτού διατυπωθεί η ίδια η γλώσσα διατυπώνεται ένας τυπικός (formal) ορισμός του μοντέλου των ροών δεδομένων, των σχέσεων και των αλληλοσυσχετίσεων μεταξύ ροών και σχέσεων. Θεωρήθηκε ορθό στον ορισμό αυτό να συμπεριληφθούν τόσο οι ροές όσο και οι σχέσεις για λόγους πρακτικούς, μιας και σε ένα πραγματικό σύστημα είναι πολύ πιθανόν να υπάρχει η ανάγκη για επεξεργασία επερωτήσεων όπου συμμετέχουν και οι δύο αλλά και επειδή το περιγραφόμενο μοντέλο είναι τέτοιο ώστε κάθε επερώτηση που αφορά αποκλειστικά ροές να μπορεί να αναχθεί σε μια επερώτηση μεταξύ σχέσεων. Ορίζονται:

Πεδίο T	Ένα καθολικό, διακριτό και διατεταγμένο χρονικό πεδίο.
Χρονική στιγμή τ	Μία οποιαδήποτε τιμή από το ***T***.
Ροή S	Ένα σύνολο στοιχείων **<*s*,*τ*>**, όπου ***s*** είναι μία πλειάδα που ακολουθεί το σχήμα της ροής και ***τ*** το χρονόσημο της πλειάδας.
Σχέση R	Μία αντιστοίχιση από το ***T*** προς ένα πεπερασμένο σύνολο πλειάδων που ακολουθούν το σχήμα της σχέσης.
Στοιχείο Ροής <s, τ>	Δηλώνει ότι μία πλειάδα ***s*** που ανήκει στη ροή ***S*** καταφθάνει τη χρονική στιγμή ***τ***, που είναι το χρονόσημό της.

Παρατηρείται ότι, όπως στο σχεσιακό μοντέλο, κάθε ροή και κάθε σχέση έχει συγκεκριμένο και γνωστό σχήμα. Ακόμη, για μια δεδομένη χρονική στιγμή ***τ*** από το ***T*** είναι δυνατόν να υπάρχουν μηδέν, ένα ή περισσότερα στοιχεία μιας ροής με το ίδιο αυτό χρονόσημο και το χρονόσημο αυτό δεν είναι μέρος του σχήματος της ροής αλλά επιπρόσθετη ιδιότητα πάνω στις πλειάδες της ροής.

Βασική Ροή	Η ίδια η ροή που καταφθάνει στην είσοδο ενός συστήματος επεξεργασίας όπως εστάλη από την πηγή της.
Παραγόμενη Ροή	Μια ροή που παράγεται σαν αποτέλεσμα μιας επερώτησης ή μιας υπό-επερώτησης (sub-query).
Στιγμιότυπο R(τ)	Το μη διατεταγμένο σύνολο πλειάδων της ***R*** τη χρονική στιγμή ***τ***.
Βασική Σχέση	Μια αποθηκευμένη σχέση.
Παραγόμενη Σχέση	Μια σχέση που παράγεται σαν αποτέλεσμα μιας επερώτησης ή μιας υπό-επερώτησης.

Στο παραδοσιακό σχεσιακό μοντέλο οι σχέσεις δεν έχουν κάποια χρονική σημασιολογία και δεν

είναι παρά απλά σύνολα πλειάδων. Διακρίνεται ως **στιγμιαία σχέση** (instantaneous relation) ή σχέση του παραδοσιακού μοντέλου που δεν εμπεριέχει την έννοια του χρόνου και η **σχέση** εκείνη που δηλώνει πως απαρτίζεται από μεταβλητό με το χρόνο σύνολο πλειάδων, όπως στον πιο πάνω ορισμό της σχέσης.

Στη συνέχεια προσδιορίζονται τρία είδη τελεστών ανάμεσα σε ροές και σχέσεις. Σημειώνεται πως δε συμπεριλαμβάνεται τελεστής που λαμβάνει σαν είσοδο ροές και παράγει ροές επειδή θα μπορεί να συντεθεί από τους υπόλοιπους.

Ροής-προς-Σχέση	'Ενας τελεστής ροής προς σχέση λαμβάνει μια ροή S σαν είσοδο και παράγει μια σχέση R με σχήμα ίδιο με αυτό της S. Σε κάθε χρονική στιγμή τ το $R(\tau)$ υπολογίζεται από τα στοιχεία της S με $\tau_i < \tau$.
Σχέσης-προς-Σχέση	Ένας τελεστής σχέσης προς σχέση λαμβάνει μία ή περισσότερες σχέσεις $R_1, R_2, ..., R_n$ σαν είσοδο και παράγει μία σχέση R. Σε κάθε χρονική στιγμή τ το $R(\tau)$ υπολογίζεται από τις στιγμιαίες σχέσεις $R_1(\tau), R_2(\tau), ..., R_n(\tau)$.
Σχέσης-προς-Ροή	Ένας τελεστής σχέσης προς ροή παίρνει μια σχέση R σαν είσοδο και παράγει μία ροή S με σχήμα ίδιο με αυτό της **R**. Σε κάθε χρονική στιγμή τ τα στοιχεία της S που έχουν χρονόσημο τ υπολογίζονται από το $R(\tau')$ με $\tau' \leq \tau$.

Ακολούθως, προτού καθοριστεί το συντακτικό της γλώσσας CQL προσδιορίζονται σημασιολογικά τα δομικά στοιχεία πάνω στα οποία αυτό θα βασιστεί. Αυτά είναι τα εξής:

- Μία δηλωτική **σχεσιακή γλώσσα επερωτήσεων**, η οποία μπορεί να ιδωθεί αφηρημένα σαν ένα σύνολο από τελεστές Σχέσης-προς-Σχέση.

- Ένας τρόπος **προσδιορισμού πλαισίων** (windows) ο οποίος χρησιμεύει στην εξαγωγή πλειάδων από τις ροές με τη λογική της Ροής-προς-Σχέση. Πρακτικά δίνουν τη δυνατότητα ορισμού πλαισίων αλλά τυπικά είναι ένας τρόπος για να παράγονται πεπερασμένα υποσύνολα πλειάδων από μη πεπερασμένα, όπως οι ροές. Ειδικά παράθυρα είναι το απεριόριστο (Unbounded) που δείχνει ό,τι λέει και το όνομά του και το τρέχον (Now) που αναφέρεται μόνο στην τρέχουσα χρονική στιγμή.

- Τρεις στοιχειώδεις τελεστές Σχέσης-προς-Ροή που μπορούν να αποδώσουν όλη την εκφραστικότητα των επιθυμητών επερωτήσεων. Ο ***Istream*** (insert stream) που όταν εφαρμόζεται σε μία σχέση R περιέχει το στοιχείο ροής $\langle s,\tau \rangle$ για πλειάδες s που

βρίσκονται στο ***R(τ) – R(τ-1)***, δηλαδή το τμήμα της ***R*** που εμφανίστηκε την τελευταία χρονική στιγμή – αυτό που προστέθηκε. Ο ***Dstream*** (delete stream) με παρόμοιο τρόπο όταν εφαρμόζεται σε μια σχέση ***R*** περιέχει το στοιχείο ροής ***<s,τ>*** για πλειάδες ***s*** που βρίσκονται στο ***R(τ-1) – R(τ)***, αυτό, δηλαδή, που αφαιρέθηκε. Ο ***Rstream*** (relation stream) όταν εφαρμόζεται σε μία σχέση περιέχει το στοιχείο ροής ***<s,τ>*** για πλειάδες ***s*** που περιέχονται στην ***R*** τη χρονική στιγμή ***τ***.

Μία συγκεκριμένη ιδιότητα που συνάγεται από τους παραπάνω ορισμούς είναι ότι όταν τίθεται το πλαίσιο σε μια επερώτηση απεριόριστο (Unbounded) και εφαρμόζεται ο τελεστής Istream τότε η επερώτηση μπορεί να μεταγραφεί με τον τελεστή Rstream και παράθυρο τρέχον (Now). Αυτό γιατί το να ζητηθεί απεριόριστο πλαίσιο με τελεστή που παράγει τις πλειάδες σε ροή είναι ισοδύναμο με το να ζητηθεί πλαίσιο τρέχον, δηλαδή, μόνο οι πλειάδες της τρέχουσας κάθε φορά χρονικής στιγμής οι οποίες λόγω του τελεστή σχέσης εισάγονται διαδοχικά στην παραγόμενη σχέση. Είναι, επομένως μια περίπτωση όπου ροή και σχέση ταυτίζονται ως προς το περιεχόμενό τους.

Όσον αφορά το συντακτικό, πλέον, της CQL αυτό περιέχει τις εξής τρεις διαφοροποιήσεις από την SQL, ενώ, κατά τα άλλα, περιέχει θεωρητικά όλη την υπόλοιπη δομή της:

- Οπουδήποτε σε μια επερώτηση αναφέρεται μία σχέση στην SQL είναι δυνατόν να αναφέρεται μία ροή στη CQL.

- Κάθε αναφορά σε μία βασική ροή και κάθε υπό-επερώτηση που παράγει μία παραγόμενη ροή πρέπει υποχρεωτικά να ακολουθείται από τη δήλωση του πλαισίου (window) πάνω στο οποίο γίνεται η αναφορά. Μέχρι στιγμής στη CQL υποστηρίζονται μόνο κυλιόμενα πλαίσια (sliding windows) βασισμένα στο χρόνο, στο πλήθος των πλειάδων και διαμερισμένα (partitioned). Για τα πρώτα δύο είδη πλαισίων έχει γίνει αναφορά, ενώ το τρίτο αποτελεί κάτι ανάλογο της ομαδοποίησης GROUP BY της SQL και σημαίνει ένα υποσύνολο των πεδίων μιας ροής δημιουργώντας ανεξάρτητες υπό-ροές (substreams).

- Κάθε αναφορά σε σχέση και κάθε υπό-επερώτηση που παράγει μια παραγόμενη σχέση μπορεί να μετατραπεί σε αναφορά σε ροή εφαρμόζοντας κάποιον από τους τελεστές Istream, Dstream και Rstream πάνω στην επιλογή (SELECT) της επερώτησης.

Για να δούμε ορισμένα παραδείγματα επερωτήσεων CQL ας θεωρήσουμε ένα σύστημα δημοπρασιών που απαρτίζεται από χρήστες και συναλλαγές ανάμεσά τους με τις επερωτήσεις που αυτοί και οι διαχειριστές θέτουν κατά τη λειτουργία του συστήματος. Το σχήμα που θα χρησιμοποιηθεί σε αυτό το απλό σενάριο είναι το εξής[15]:

Register(user_id, name, state)	Η καταχώρηση ενός χρήστη στο σύστημα.
Deregister(user_id)	Η αποχώρηση ενός χρήστη από το σύστημα.
Open(item_id, seller_id, start_price)	Η εκκίνηση της δημοπράτησης ενός αντικειμένου.
Close(item_id)	Η ολοκλήρωση της δημοπράτησης ενός αντικειμένου.
Bid(item_id, bidder_id, bid_price)	Η προσφορά ενός χρήστη για ένα αντικείμενο που βρίσκεται υπό δημοπράτηση.
User(user_id, name, state)	Παραγόμενη από τις Register(...) και Deregister(...).

Ακολουθούν ορισμένα παραδείγματα επερωτήσεων CQL βασισμένα στο παραπάνω σενάριο της δημοπρασίας όπου τα δεδομένα που "ρέουν" στο σύστημα επεξεργασίας είναι οι είσοδοι και έξοδοι των χρηστών από αυτό, η εκκινήσεις και ολοκληρώσεις των δημοπρασιών των αντικειμένων και οι προσφορές των χρηστών στα δημοπρατούμενα αντικείμενα.

Επερώτηση *Να επιλεγούν οι δημοπρασίες των οποίων η τιμή εκκίνησης είναι μεγαλύτερη από 100 και παράγουν το αποτέλεσμα ως μία ροή.*

CQL

```
SELECT *
FROM Open
WHERE start_price > 100
```

Σχόλια Η ροή Open αναφέρεται χωρίς πλαίσιο και αυτό σημαίνει προεπιλεγμένα απεριόριστο πλαίσιο (unbounded window), που σημαίνει όλες τις πλειάδες από τη χρονική στιγμή 0 μέχρι την τρέχουσα και μπορεί να δηλωθεί με το λεκτικό "Now". Το αποτέλεσμα της επερώτησης είναι όλες αυτές οι πλειάδες φιλτραρισμένες από τη συνθήκη WHERE. Η παραγόμενη σχέση είναι μονότονη – γίνονται μόνο προσθήκες στο σύνολο των στοιχείων της – κι επομένως θα μπορούσε να εφαρμοστεί ο τελεστής Istream ως εξής:

```
SELECT Istream(*)
FROM Open[Now]
WHERE start_price > 100
```

15 Οι ονομασίες των σχέσεων και των πεδίων τους είναι αυτονόητες και δεν υπονοούν τίποτε παραπάνω από το όνομά τους..

Επερώτηση *Να διατηρείται το τρέχον πλήθος των προσφορών κατά την τελευταία ώρα για τα αντικείμενα με item_id από 100 έως 200.*

CQL

```
SELECT Count(*)
FROM Bid[Range 1 hour]
WHERE item_id >= 100 AND
      item_id <= 200
```

Σχόλια Η επιλογή πάνω στη Bid γίνεται με προσδιορισμένο πλαίσιο. Αν σε αυτή τη δήλωση προσθέταμε τον τελεστή Istream τότε το αποτέλεσμα θα ήταν μια ροή με μία νέα πλειάδα αποτελέσματος κάθε φορά που το συνολικό πλήθος μεταβαλλόταν. Αν υπήρχε ανάγκη το αποτέλεσμα να εξάγεται σαν μία παραγόμενη ροή τότε θα χρησιμοποιούσαμε τον τελεστή Rstream.

Επερώτηση *Να διατηρείται ένας πίνακας με τις δημοπρασίες που είναι ανοικτές την τρέχουσα χρονική στιγμή.*

CQL

```
SELECT *
FROM Open
WHERE item_id NOT IN (
    SELECT *
    FROM Close)
```

Σχόλια Καταρχάς τόσο στην Open όσο και στην Close, έτσι όπως δηλώνονται εδώ, υπονοείται το προεπιλεγμένο απεριόριστο πλαίσιο. Το αποτέλεσμα της επερώτησης δεν είναι μονότονο – όταν μια δημοπρασία κλείσει τότε αυτή διαγράφεται από το αποτέλεσμα – και άρα δεν εφαρμόζεται ένας τελεστής Istream.

Επερώτηση *Να διατηρηθεί μία σχέση – παραγόμενη – User(user_id, name, state) με τους καταχωρημένους την τρέχουσα χρονική στιγμή χρήστες.*

CQL

```
SELECT user_id, name, state
FROM Register[Partition By user_id Rows 1]
WHERE user_id NOT IN (
    SELECT *
    FROM Deregister)
```

Σχόλια Το διαμερισμένο πλαίσιο της ροής Register λαμβάνει την τελευταία καταχώρηση στο σύστημα του κάθε χρήστη – αυτό προσδιορίζεται με το εύρος Rows 1 – από όπου στη συνέχεια φιλτράρονται οι αποχωρήσεις χρηστών που έχουν καταγραφεί.

Επερώτηση *Να αποδοθούν με μορφή ροής οι δημοπρασίες που εκκινήθηκαν από χρήστες που βρίσκονταν στην πόλη Α κατά τη στιγμή της εκκίνησης.*

CQL
```
SELECT Istream(Open.*)
FROM Open[Now], User
WHERE Open.seller_id = User.user_id AND
      state = "A"
```

Σχόλια Η παραγόμενη σχέση User είναι αυτή που υπολογίζεται από την προηγούμενη επερώτηση. Η εφαρμογή του τρέχοντος πλαισίου (Now) εξασφαλίζει ότι η συνένωση της σχέσης του χρήστη με τη σχέση της εκκίνησης δημοπρασιών θα γίνει μόνο για τη στιγμή εκκίνησης της δημοπρασίας. Αν εφαρμοζόταν απεριόριστο (Unbounded) πλαίσιο τότε κάθε φορά που ο χρήστης επανερχόταν στην πόλη Α τότε όλες οι δημοπρασίες που είχε εκκινήσει θα προωθούνταν στο αποτέλεσμα.

Επερώτηση *Να αποδοθούν ως ροή τα item_id όλων των δημοπρασιών μέσα σε 5 ώρες από την εκκίνησή τους.*

CQL
```
SELECT Istream(Close.item_id)
FROM Close[Now], Open[Range 5 Hours]
WHERE Close.item_id = Open.item_id
```

Σχόλια Η επερώτηση αυτή συνενώνει τις τρέχουσες κάθε χρονική στιγμή δημοπρασίες που κλείνουν με τις δημοπρασίες που εκκινήθηκαν 5 ώρες πριν και επιστρέφει σαν ροή το αποτέλεσμα. Αποτελεί παράδειγμα συνένωσης πάνω σε πλαίσια ροών.

Η CQL δημιουργήθηκε με σκοπό να αποτελέσει την πρώτη γενικού σκοπού δηλωτική γλώσσα επερωτήσεων πάνω σε σχέσεις και ροές δεδομένων, η οποία να έχει βασιστεί σε συγκεκριμένα σημασιολογικά στοιχεία. Δεν εξαρτάται από το συντακτικό καθεαυτό που χρησιμοποιεί – αν και αυτό είναι της SQL για προφανείς λόγους. Αναγνωρίζει την ανάγκη για τελεστές που δίνουν αντιστοιχίσεις ανάμεσα σε ροές και σχέσεις και τους ορίζει τυπικά. Επιχειρεί να έχει τη δυνατότητα να εκφράσει όλη τη σημασιολογία που έχει εντοπιστεί στη σύγχρονη έρευνα γύρω από τις συνεχείς επερωτήσεις και να είναι τυπικά και καλά ορισμένη σε ένα αφαιρετικό επίπεδο ανεξάρτητο της εκάστοτε υλοποίησής της. Έχει υλοποιηθεί σε ένα μικρό ένα βαθμό στο [49], γι' αυτό και η ύπαρξή της βρίσκεται περισσότερο σε θεωρητικό παρά σε εφηρμοσμένο επίπεδο.

4.1.2 Η ΓΛΏΣΣΑ ESL (EXPRESSIVE STREAM LANGUAGE)

Η γλώσσα ESL (Expressive Stream Language) [50][52] δημιουργήθηκε στα πλαίσια του ερευνητικού προγράμματος συστημάτων διαχείρισης ροών δεδομένων Stream Mill [53], έπειτα από την παρατήρηση ότι, ενώ έχει παρουσιαστεί μεγάλο ερευνητικό ενδιαφέρον στο πεδίο των ροών δεδομένων και της εκτέλεσης συνεχών επερωτήσεων, υπάρχουν κάποιες προκλήσεις που δεν έχουν αντιμετωπιστεί πλήρως. Αυτές είναι η περιορισμένη εκφραστικότητα των γλωσσών επερωτήσεων των παραδοσιακών βάσεων δεδομένων που δεν μπορούν να υποστηρίξουν την έννοια των ροών, παρά μόνο σε ένα ελάχιστο βαθμό[16], και η δυνατότητα παροχής ενδείξεων βελτιστοποίησης στο σύστημα διαχείρισης των ροών δεδομένων. Ακόμη, η ESL επιχειρεί να υποστηρίξει τόσο συνεχείς επερωτήσεις πάνω σε ροές όσο και σε όψεις που δημιουργούνται πάνω σε ροές και σε συνδυασμούς ροών και στατικών δεδομένων. Και εδώ σαν βάση χρησιμοποιείται η ευρύτατα διαδεδομένη γλώσσα SQL. Η ESL επεκτάθηκε αργότερα στην ESL-TS με τρόπο ώστε να υποστηρίζει σειριακές επερωτήσεις (sequence queries) πάνω σε εξαρτώμενες από το χρόνο ακολουθίες (time series)[17].

Η γλώσσα αυτή δεν έχει στηριχτεί σε κάποιο ιδιαίτερο θεωρητικό σημασιολογικό υπόβαθρο και έχει περισσότερο πρακτική σκοπιμότητα, χωρίς αυτό να είναι ένα στοιχείο που αναγκαστικά τη μειώνει λόγου χάριν απέναντι στη CQL, μιας και είναι κάπως διαφορετική η προσέγγισή της. Προσπαθεί να ενσωματώσει τις περισσότερες από τις λειτουργίες που υπάρχουν στις παραδοσιακές βάσεις δεδομένων και την SQL που έχουν νόημα και στις ροές δεδομένων. Λειτουργίες όπως η UPDATE ή η DELETE δεν έχουν νόημα σε μια ροή δεδομένων που εισέρχεται αδιάκοπα σε ένα ΣΔΡΔ. Ένα από τα στοιχεία που τη διαχωρίζουν από άλλες προσεγγίσεις είναι η δυνατότητα δήλωσης συναθροιστικών επερωτημάτων (aggregates) από το χρήστη.

Στα θεμελιώδη χαρακτηριστικά της ανήκουν το γεγονός ότι βλέπει τις ροές σαν μη φραγμένες ακολουθίες πλειάδων και ότι υποστηρίζει τρία είδη χρονοσήμων (timestamps): εξωτερικά (internal), εσωτερικά (external) και λανθάνοντα (latent). Τα εξωτερικά είναι χρονόσημα που έχουν τοποθετηθεί μέσα στις πλειάδες στο σημείο από όπου πηγάζουν. Τα εσωτερικά υπονοούν το χρόνο

16 Για παράδειγμα τελεστές απλής επιλογής/προβολής θα μπορούσαν αυτονόητα με την κλασσική τους έννοια από το σχεσιακό μοντέλο να εφαρμοστούν με όμοιο τρόπο και στις ροές δεδομένων. Αποτελούν, όμως, ένα πολύ μικρό υποσύνολο των επιθυμητών ενεργειών που θα ήθελε κανείς να θέσει πάνω στις ροές. Άλλα είδη επερωτήσεων που έχουν εντοπιστεί στις ροές δεδομένων είναι, για παράδειγμα, οι τελεστές της CQL που ήδη συζητήσαμε.

17 Με τον όρο χρονικές ακολουθίες (time series) [11][50] γίνεται αναφορά σε μία άποψη των ροών δεδομένων από την οπτική της παρακολούθησης συστημάτων (monitoring), εννοώντας ακολουθίες δεδομένων που έχουν εγγενώς μέσα τους την έννοια του χρόνου και οι οποίες δεν είναι δυνατόν να αποθηκευθούν εξολοκλήρου σε ένα σταθερό αποθηκευτικό μέσο, ενώ είναι επιθυμητό να υπάρχει η δυνατότητα της εκτέλεσης επερωτήσεων σε όσο δυνατόν ευρύτερο ιστορικό τους.

εισόδου μιας πλειάδας στο σύστημα επεξεργασίας. Τέλος, τα λανθάνοντα έχουν τοπική χροιά και παράγονται μόνο όταν κάτι τέτοιο απαιτείται από την εκάστοτε λειτουργία. Στην ESL θεωρητικά δημιουργείται μία ροή με λανθάνον χρονόσημο για κάθε ροή που διαθέτει εξωτερική χρονοσήμανση έτσι ώστε να αντιμετωπιστεί το φαινόμενο της μη χρονικά διατεταγμένης άφιξης των πλειάδων.

Επιπλέον, σε ένα σύστημα που υποστηρίζει τη γλώσσα αυτή μπορούν να δηλωθούν ροές με τα πεδία που διαθέτουν ακριβώς όπως θα γινόταν και στην SQL προσθέτοντας επιπλέον και τα χρονόσημά τους και το σημείο από το οποίο εισέρχονται στο σύστημα. Μπορούν να δηλωθούν όψεις ροών και πάλι σε αναλογία με τις όψεις της SQL και να δημιουργηθούν με αυτό τον τρόπο παράγωγες και σύνθετες ροές από τις υπάρχουσες. Ακόμη, υποστηρίζονται οι συνενώσεις ροών με σχέσεις βάσεων δεδομένων, οι ενώσεις (unions) μεταξύ ροών και πλαίσια προσδιοριζόμενα από εύρος χρόνου ή πλήθος πλειάδων. Σημειώνουμε ότι αξιωματικά στη γλώσσα αυτή μπορεί να υπάρχει μόνο μία ροή στο τμήμα FROM της επερώτησης, ενώ είναι δυνατόν να υπάρχουν μία ροή σε συνδυασμό με κάποιες σχέσεις. Γενικά, το συντακτικό που χρησιμοποιείται συμβαδίζει όσο το δυνατόν περισσότερο με αυτό της SQL, όσον αφορά τις δεσμευμένες λέξεις, τη σειρά των τελεστών, τις συνενώσεις και τις συνθήκες των επερωτήσεων.

Τα συναθροιστικά επερωτήματα (UDA – User Defined Aggregates) είναι καθολικοί τελεστές που μπορούν να προσδιοριστούν έως και με αρκετά σύνθετο τρόπο από το χρήστη και αποτελούν ένα από τα σημεία για τα οποία η γλώσσα ESL ισχυρίζεται ότι είναι ιδιαίτερα εκφραστική. Τα συναθροιστικά επερωτήματα μπορούν να οριστούν είτε με καθολικό (blocking) είτε με μη καθολικό τρόπο (non-blocking) και με τέτοιο τρόπο ώστε να υπάρχει η δυνατότητα συγγραφής ολόκληρων διαδικασιών που να τα παράγουν. Ακολουθούν ορισμένα παραδείγματα από τη γλώσσα ESL[18].

Επερώτηση *Δημιουργία Ροής*

ESL

```
CREATE STREAM ClosedAuction(
    itemID int,
    buyerID char(10),
    final_price real,
    current_time timestamp)
ORDER BY current_time
```

18 Και σε αυτά τα παραδείγματα δε θα ασχοληθούμε με τη σημασιολογία των ροών και των πεδίων τους, τα οποία θεωρούνται αυτονόητα από τις ονομασίες τους.

```
SOURCE 'POR4446'
```

Σχόλια Δήλωση στο σύστημα μιας ροής με πεδία itemID - ακέραιος, buyerID - αλφαριθμητικό 10 χαρακτήρων, final_price - πραγματικός, current_time – χρονόσημο, που έρχεται μέσω της θύρας 4446. Η λέξη current_time είναι δεσμευμένη και αναφέρεται σε ρητό εσωτερικό χρονόσημο – η χρονική στιγμή εισόδου κάθε πλειάδας στο σύστημα. Αν έθετε κανείς κάποια άλλη ονομασία στο χρονόσημο αυτό θα σήμαινε πως εκείνο το πεδίο είναι χρονόσημο ορισμένο στην πηγή της πλειάδας. Αν παρέλειπε το ORDER BY τότε μόνο λανθάνοντα χρονόσημα θα χρησιμοποιούνταν σε αυτή τη ροή.

Επερώτηση *Δημιουργία Όψης*

ESL

```
CREATE STREAM expensiveItems AS
SELECT itemID, sellerID, start_price, start_time
FROM OpenAuction
WHERE start_price > 1000
```

Σχόλια Δήλωση στο σύστημα μιας όψης ή παραγόμενης ροής από το φιλτράρισμα κάποιας άλλης. Εφόσον στα πεδία της επιλογής το πεδίο χρονόσημο start_time διατηρείται τότε αυτό θα είναι και το χρονόσημο της παραγόμενης ροής, διαφορετικά θα είχε μόνο λανθάνοντα χρονόσημα.

Επερώτηση *Συνένωση Ροής με Σχέση*

ESL

```
CREATE STREAM offersZC
SELECT ZipCode, itemID, start_price, start_time
FROM expensive_items AS I,
     sellerinfo      AS S
WHERE I.sellerID = S.SellerID
```

Σχόλια Δήλωση στο σύστημα μιας ροής που παράγεται από τη συνένωση μιας ροής με μία σχέση. Η expensive_items είναι μία ροή – παραγόμενη – από το προηγούμενο παράδειγμα, και η seller_info ένας πίνακας με στατικά δεδομένα.

Επερώτηση *Να τοποθετούνται σε ένα πίνακα όλες οι δημοπρασίες με τιμή πάνω από 1000 για τα τελευταία 10 λεπτά.*

ESL

```
CREATE TABLE high_priced AS (
SELECT ItemID, SellerID
FROM OpenedAuction
```

```
OVER (RANGE 10 MINUTE PRECEDING CURRENT)
WHERE start_price > 1000)
REFRESH IMMEDIATE
```

Σχόλια Δήλωση στο σύστημα ενός πίνακα με την επιθυμητή συνθήκη και τον προσδιορισμό ενός χρονικού πλαισίου 10 λεπτών από την τρέχουσα χρονική στιγμή και πίσω μέσω της δήλωσης OVER. Με την τελευταία δήλωση REFRESH υποδηλώνεται ότι ο πίνακας αυτός θα πρέπει να είναι πάντα ενημερωμένος με τα πιο πρόσφατα δεδομένα όποτε ζητηθεί από το χρήστη.

Επερώτηση *Επερώτηση με συνένωση ανάμεσα σε δύο ροές!*

ESL

```
INSERT INTO stdout
SELECT IdemID, current_time
FROM ClosedAuction AS C,
TABLE(OpenAuction
      OVER (RANGE 24 HOUR PRECEDING C)) AS O
WHERE O.itemID = C.itemID
```

Σχόλια Η πρώτη γραμμή δηλώνει την έξοδο της επερώτησης στην προκαθορισμένη έξοδο του συστήματος (standard output). Η συνένωση γίνεται πάνω σε μία ροή και στο χρονικά προσδιοριζόμενο πλαίσιο μιας δεύτερης. Το πλαίσιο της δεύτερης έχει προσδιοριστεί βάσει των πλειάδων της πρώτης (PRECEDING C) και εννοεί ότι για κάθε νέα πλειάδα της OpenAuction θα λαμβάνονται υπόψη στο πλαίσιο οι πλειάδες της μέχρι και 24 ώρες πριν. Έτσι ικανοποιείται και το αξίωμα της ESL να μην υπάρχουν πάνω από μία ροές στην περιοχή του FROM.

Επερώτηση *Ορισμός συναθροιστικής επερώτησης: μέσος όρος*

ESL

```
AGGREGATE myAvg(NextValue Real) : Real {
    TABLE state(sum Real, count Int);
    INITIALIZE : {
        INSERT INTO state VALUES (NextValue, 1);
    }
    ITERATE : {
        UPDATE state
        SET sum=sum+NextValue, count=count+1;
    }
    TERMINATE : {
```

```
            INSERT INTO RETURN
            SELECT sum/count
            FROM state;
        }
    }
```

Σχόλια Πρόκειται για ένα συναθροιστικό ισοδύναμο της γνωστής AVG που παράγει το μέσο όρο στην SQL. Ορίζεται ένας πίνακας που κρατά το άθροισμα sum και το πλήθος count των στοιχείων, με το πηλίκο των οποίων υπολογίζεται το αποτέλεσμα. Η συνάθροιση ξεκινά με δηλώσεις εκκίνησης (INITIALIZE), όπου ο πίνακας state αρχικοποιείται με την πρώτη τιμή NextValue και πλήθος 1. Στη συνέχεια ο state ανανεώνεται μέσω διαδοχικών επαναλήψεων (ITERATE), οι οποίες συμβαίνουν όποτε εμφανίζεται ένα νέο NextValue. Ο υπολογισμός τερματίζεται (TERMINATE) όταν δεν υπάρχει πλέον άλλη είσοδος. Αφήνοντας το πεδίο TERMINATE κενό η συνάθροιση αυτή μπορεί να εφαρμοστεί ελεύθερα σε ροές, οι οποίες θεωρητικά δεν τερματίζουν. Η λέξη RETURN υποδηλώνει την επιστροφή στο χρήστη του τελικού αποτελέσματος.

Ολοκληρώνοντας την αναφορά στην ESL μπορούμε να παρατηρήσουμε πως σαν γλώσσα επιχειρεί να κάνει αυτό που δηλωτικές γλώσσες όπως η SQL έκαναν στο παρελθόν, δηλαδή να δώσει μια φιλική στον άνθρωπο εκφραστικότητα πάνω στις συνεχείς επερωτήσεις. Υποστηρίζει όλες τις απλές επερωτήσεις πάνω σε ροές δεδομένων, όπως οι επιλογές/προβολές, αλλά και συνενώσεις ανάμεσα σε ροές, με τον περιορισμό το πολύ μία από τις συμμετέχουσες να παίρνει μέρος ολόκληρη, δηλαδή χωρίς τον προσδιορισμό κάποιου πλαισίου. Ακόμη, δίνει τη δυνατότητα ορισμού συναθροιστικών επερωτήσεων στο χρήστη αφήνοντάς τον να ορίσει ολόκληρες ρουτίνες υπολογισμού που δεν είναι εκ των προτέρων ενσωματωμένες στη γλώσσα χωρίς περιορισμούς. Αυτή θεωρείται πως είναι η πιο εξειδικευμένη δυνατότητά της που την διαχωρίζει.

4.2 ΕΡΕΥΝΗΤΙΚΆ ΣΥΣΤΉΜΑΤΑ ΔΙΑΧΕΊΡΙΣΗΣ ΡΟΏΝ ΔΕΔΟΜΈΝΩΝ

Στην ενότητα αυτή θα παρουσιαστούν συνοπτικά μερικά από τα σημαντικότερα Συστήματα Διαχείρισης Ροών Δεδομένων (ΣΔΡΔ) που εμφανιστεί στη σύγχρονη βιβλιογραφία. Στο σύνολό τους είναι ερευνητικές προσπάθειες που έχουν σαν πρώτο σκοπό να αποτελέσουν τη βάση για την εφαρμογή αλγορίθμων και μεθοδολογιών που θα μπορούσαν να ενσωματωθούν σε ένα πλήρες ΣΔΡΔ. Επομένως, δεν αποτελούν ώριμα συστήματα, αν και μερικά από αυτά χρονολογούνται εδώ

και μερικά χρόνια. Σε κάθε περίπτωση αποτέλεσαν και αποτελούν σημεία αναφοράς για τη σύγχρονη έρευνα στις ροές δεδομένων. Όπως και με τις γλώσσες συνεχών επερωτήσεων επιλέχθηκε να γίνει αναφορά τα συστήματα εκείνα που αναφέρονται συχνότερα στη βιβλιογραφία και φαίνεται να έχουν το μεγαλύτερο αντίκτυπο στην έρευνα στο πεδίο των ροών δεδομένων. Αυτά είναι το STREAM [49], η Aurora [55] και το TelegraphCQ [54].

4.2.1 ΤΟ ΣΥΣΤΗΜΑ TELEGRAPH CQ

Το σύστημα TelegraphCQ [54][57][58] είναι ένα ερευνητικό σύστημα διαχείρισης ροών δεδομένων που αποσκοπεί στην επεξεργασία συνεχών συνεχών επερωτήσεων παρόλες τις αντιξοότητες που οι ροές επιφέρουν – αυθαίρετοι ρυθμοί άφιξης, μεγάλοι όγκοι δεδομένων κλπ. Ένα πρώτο ιδιαίτερο σημείο του TelegraphCQ είναι η επιλογή να βασιστεί πάνω στην ανοικτού κώδικα και ευρέως γνωστή για τη σταθερότητα και αφοσίωσή της στα πρότυπα βάση δεδομένων PostgreSQL [59]. Μία τέτοια επιλογή δικαιολογείται, μιας και το σύστημα διαχείρισης βάσεων δεδομένων αυτό τυγχάνει μεγάλης εκτίμησης στο χώρο του ανοικτού λογισμικού και η διαμόρφωσή του με τρόπο ώστε να υποστηρίζει ροές αποτελεί πραγματική πρόκληση αλλά και συνεισφορά στην κοινότητα που το ανέδειξε[19].

Πρώτιστο μέλημα του TelegraphCQ είναι η αξιοποίηση τεχνικών διαμοιρασμού πόρων (resource sharing) και η προσαρμοστική (adaptive) εκτέλεση των συνεχών επερωτήσεων. Γι' αυτό το σκοπό η λειτουργικότητα του ΣΔΒΔ στο οποίο στηρίζεται χρειάστηκε να τροποποιηθεί αρκετά με την εφαρμογή τελεστών όπως οι στρόβιλοι (eddies) για τη δρομολόγηση πλειάδων και ο προγραμματισμός τελεστών (operator scheduling). Ακόμη, η παραδοσιακή αντιμετώπιση της εκτέλεσης κάθε επερώτησης ανεξάρτητα από τις άλλες έπρεπε να διαμορφωθεί ώστε οι πολλές ταυτόχρονα εκτελούμενες συνεχείς επερωτήσεις να χρησιμοποιούν όσο το δυνατόν περισσότερο κοινούς πόρους. Υποστηρίζονται, έτσι, η καταχώρηση και διαγραφή ροών από το σύστημα και η δυνατότητα να τίθενται συνεχείς επερωτήσεις πάνω σε αυτές, πάνω σε συνδυασμούς αυτών και κλασσικών σχεσιακών πινάκων.

Οι δηλώσεις που καταχωρούν και διαγράφουν μία ροή από το σύστημα είναι οι CREATE STREAM και DROP STREAM αντίστοιχα και κατ' αναλογία με τις σχετικές δηλώσεις της SQL. Είναι απαραίτητη προϋπόθεση ότι σε κάθε ροή ακριβώς ένα πεδίο θα είναι το χρονόσημο κάτι που ορίζεται με τον όρο TIMESTAPCOLUMN. Το σύστημα περιμένει ότι κάποια πηγή δεδομένων αφού δημοσιοποιήσει σε ποια καταχωρημένη ροή αναφέρεται θα ξεκινήσει να στέλνει δεδομένα

19 Η PostgreSQL διατίθεται υπό τη άδεια BSD, η οποία είναι μία από τις περισσότερο "ελεύθερες" άδειες ανοικτού λογισμικού που υπάρχουν.

που ακολουθούν το σχήμα αυτής. Φυσικά, περισσότερες από μία πηγές μπορούν να στέλνουν πλειάδες της ίδιας ροής. Η είσοδος των πλειάδων στο σύστημα γίνεται μέσω προπαρασκευαστικών συναρτήσεων (wrapper functions) που προσδιορίζονται από το χρήστη και έχουν ως ευθύνη τους τη διαχείριση της εισόδου των πλειάδων που στέλνει κάποια πηγή. Οι συναρτήσεις αυτές έχουν τρία τμήματα, τα init, next και done που καλούνται κατά την καταχώρηση της ροής, κατά την είσοδο των πλειάδων της και κατά την ολοκλήρωση της ροής αντίστοιχα. Επιπλέον, εφόσον υπάρχουν ροές που εισέρχονται στο σύστημα, ο χρήστης είναι δυνατόν να θέσει πάνω τους συνεχείς επερωτήσεις, οι οποίες θα εκτελούνται έως ότου αυτός τις ακυρώσει. Οι επερωτήσεις αυτές δεν υποστηρίζουν υπό-επερωτήσεις (subqueries) και μπορούν να προσδιορίζουν ένα πλαίσιο (window) εκτέλεσης. Ενδεικτικά:

ΚΑΤΑΧΏΡΗΣΗ ΡΟΉΣ

```
CREATE STREAM measurements (
    tcqtime TIMESTAMP TIMESTAMPCOLUMN,
    stationid INTEGER,
    speed REAL) TYPE ARCHIVED
```

ΔΗΜΙΟΥΡΓΊΑ ΠΡΟΠ/ΣΤΙΚΉΣ ΣΥΝΆΡΤΗΣΗΣ

```
CREATE FUNCTION measurements_init(INTEGER)
RETURNS BOOLEAN AS
'libmeasurements.so',measurements_init'
LANGUAGE 'C'
```

ΣΥΝΕΧΉΣ ΕΠΕΡΏΤΗΣΗ

```
SELECT ms.stationid, s.name, s.highway, s.mile,
       AVG(ms.speed)
FROM measurements ms, stations s
WHERE ms.stationid = s.stationid
GROUP BY ms.stationid, s.name, s.highway, s.mile
WINDOW ms ['10 minutes']
```

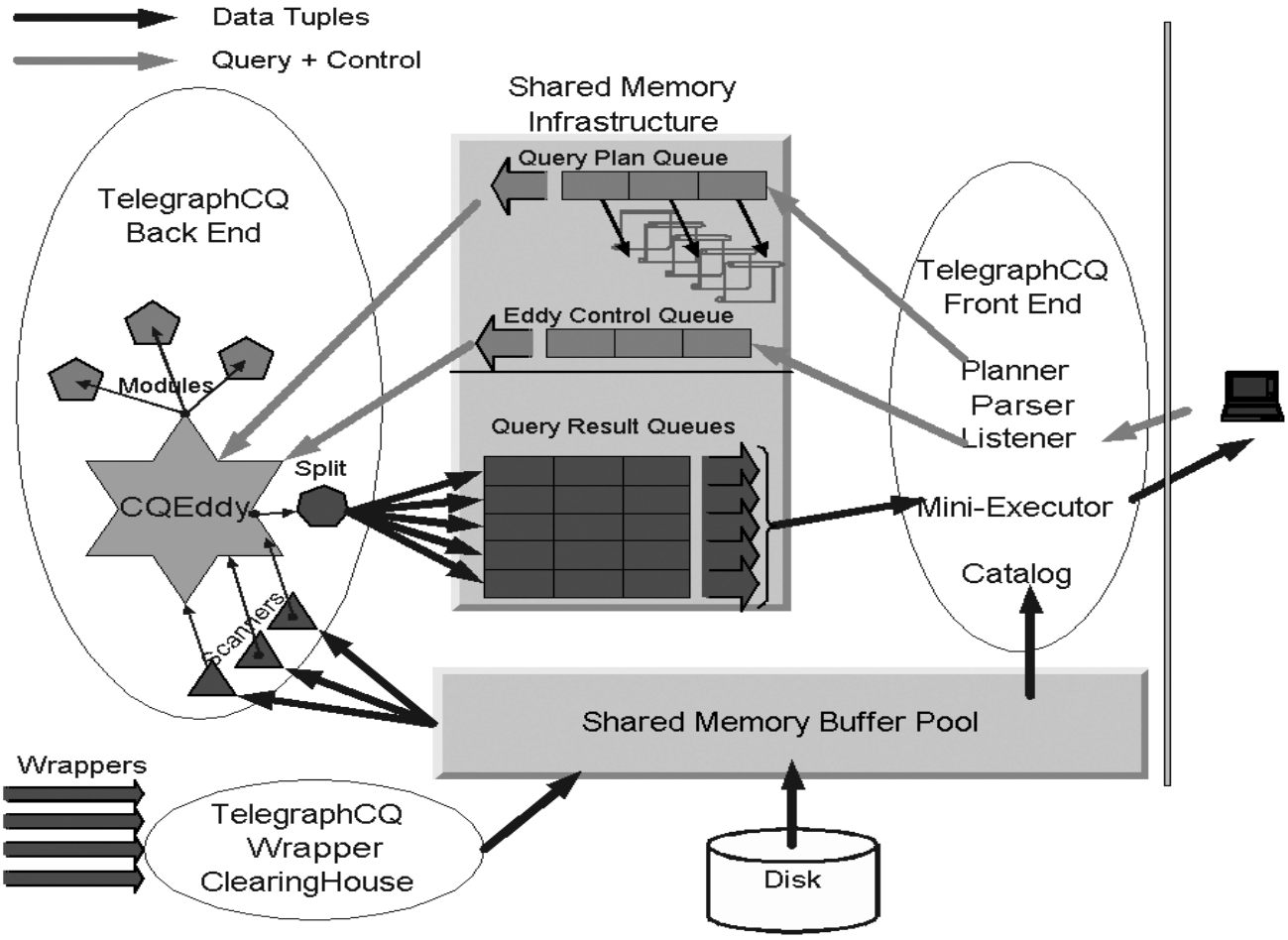
Μία συγκεκριμένη και πολύ ενδιαφέρουσα ιδιαιτερότητα της γλώσσας που χρησιμοποιεί το TelegraphCQ έγκειται στον τρόπο με τον οποίο εφαρμόζονται τα πλαίσια. Εκεί ο χρήστης έχει τη δυνατότητα μέσω ευέλικτων οδηγιών που θυμίζουν γλώσσα προγραμματισμού και WindowIs δηλώσεων να προσδιορίζει το επιθυμητό πλαίσιο της επερώτησης. Ενδεικτικά:

ΚΑΤΑΧΏΡΗΣΗ ΡΟΉΣ

```
SELECT closingPrice, timestamp
FROM ClosingStockPrices
WHERE stockSymbol  = 'MSFT' AND
```

```
        closingPrice > 50.00
for (t = 101; t <= 1000; t++ ) {
    WindowIs(ClosingStockPrices, 101, t);
}
```

Η γλώσσα που χρησιμοποιείται στο TelegraphCQ είναι σχετικά απλή, πηγάζει άμεσα από την SQL και είναι ιδιαίτερα εύληπτη. Ως εκ τούτου είναι εύκολα κατανοητή σε κάποιον που γνωρίζει SQL αλλά δε στηρίζεται σε κάποιο συγκεκριμένο θεωρητικό σημασιολογικό υπόβαθρο, όπως, για παράδειγμα, η CQL. Για το λόγο αυτό δεν αναφέρθηκε στην ενότητα των γλωσσών συνεχών επερωτήσεων. Παρολαυτά, η απλότητά της και η άμεση σύνδεσή της με την SQL είναι ενδεικτική του μοντέλου αναπαράστασης των ροών που χρησιμοποιείται, που δεν είναι άλλο από το σχεσιακό. Ακόμη είναι ενδεικτική της φιλοσοφίας με την οποία φαίνεται να αντιμετωπίζει τις ροές το TelegraphCQ, ως επέκταση της PostgreSQL, σαν μια πολύ ιδιαίτερη περίπτωση εφαρμογών βάσεων δεδομένων. Η προσέγγιση αυτή μοιάζει αρκετά φυσική εφόσον διατηρείται και στις ροές το σχεσιακό μοντέλο.



Σχήμα 4.1 – Η αρχιτεκτονική του TelegraphCQ

Η επεξεργασία επερωτήσεων που αφορούν μονάχα σε σχεσιακούς πίνακες γίνεται στο ίδιο το Front End, ενώ οι επερωτήσεις που αφορούν σε ροές ή σε πίνακες και ροές εκτελούνται στο Back End. Το πλάνο των συνεχών επερωτήσεων σχεδιάζεται στην ενδιάμεση υποδομή και τα αποτελέσματά τους διοχετεύονται πίσω στο χρήστη πάλι μέσω της υποδομής αυτής. Με αυτό τον τρόπο ένα παραδοσιακό σχεσιακό σύστημα τροποποιείται έτσι ώστε να υποστηρίζει επερωτήσεις πάνω σε ροές, πίνακες και συνδυασμούς αυτών, διαχωρίζοντας λειτουργικά τα υπό-τμήματα που αναλαμβάνουν την κάθε μία από αυτές.

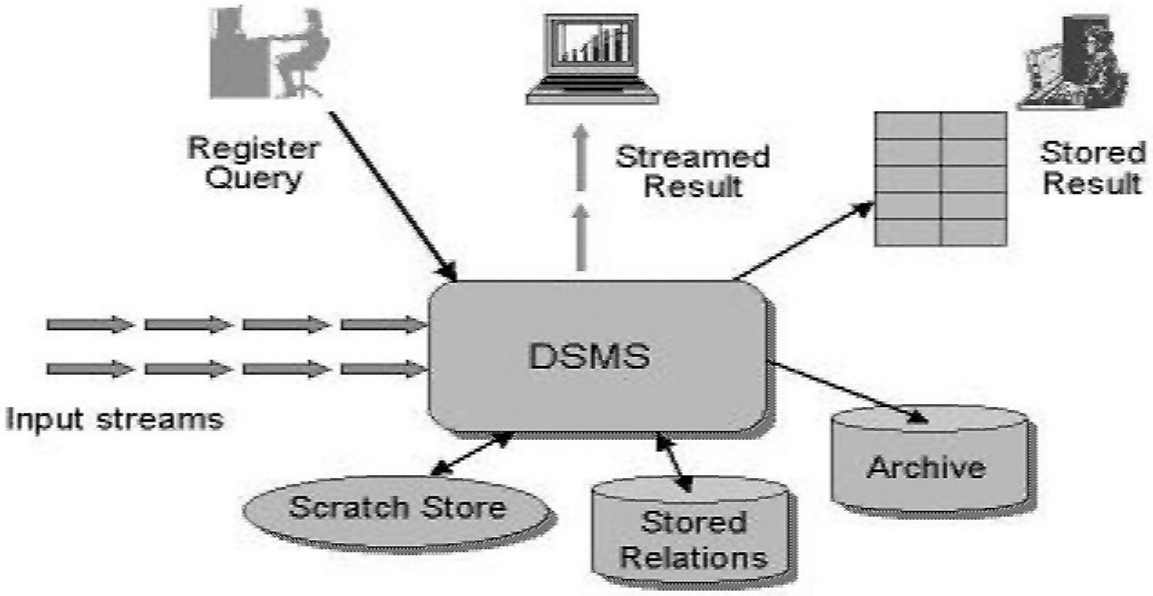
Το TelegraphCQ είναι ένα ερευνητικό πρόγραμμα που υλοποιείται ακόμη. Στηρίζεται όπως είδαμε σε ένα υπαρκτό και αποδεκτό ΣΔΒΔ και επιχειρεί να εφαρμόσει όσο το δυνατόν καλύτερα μεθοδολογίες προσαρμοστικότητας στη διαχείριση των ροών δεδομένων. Μερικές από αυτές είναι οι στρόβιλοι (eddies), οι κλάδοι ευρετηρίων (stems), ο διαμοιρασμός πόρων (resource sharing) των πλάνων των επερωτήσεων, ο προγραμματισμός τελεστών (operator scheduling) και τα πλαίσια (wndows). Μέσα στις προκλήσεις που ακόμη καλείται να αντιμετωπίσει είναι θέματα όπως η εξισορρόπηση της επεξεργαστικής ισχύος που απαιτείται για την ίδια την εφαρμογή της προσαρμοστικότητας, η δυνατότητα επεξεργασίας επερωτήσεων ανάμεσα σε ροές και ιστορικών στοιχείων αυτών (historical queries), η εφαρμογή τεχνικών συνόψεων, δειγματοληψίας και απόρριψης πλειάδων κατά την περίοδο μεγάλων φόρτων.

Η αρχιτεκτονική που υιοθετείται αποδεικνύεται ότι μπορεί να λειτουργήσει και παρέχει επιπλέον το πλεονέκτημα ότι τα τμήματα αυτά δεν πρέπει υποχρεωτικά να βρίσκονται στο ίδιο φυσικό επεξεργαστικό σύστημα αλλά θα μπορούσαν – με κάποια παραλλαγή της αρχιτεκτονικής – να είναι ανεξάρτητα μεταξύ τους και να συνεργάζονται. Κάτι τέτοιο θα έδινε τη δυνατότητα ύπαρξης ενός αφοσιωμένου και εξειδικευμένου υποσυστήματος για την κάθε λειτουργία. Με αυτό τον τρόπο η θεώρηση πάνω στην οποία στηρίχτηκε η υλοποίηση του TelegraphCQ κρίνεται αρκετά διορατική διότι δεν αγνοεί το γεγονός ότι ένα ΣΔΡΔ δεν είναι ξεκομμένο νοηματικά και λειτουργικά από ένα ΣΔΒΔ και το κάνει αυτό πράξη στηρίζοντας την υλοποίηση σε ένα ΣΔΒΔ όπως η PostgreSQL.

4.2.2 ΤΟ ΣΎΣΤΗΜΑ STREAM (STANFORD STREAM DATA MANAGER)

Το σύστημα STREAM (STanford stREam datA Manager) [49][60][61] φιλοδοξεί να γίνει ένα γενικού σκοπού σύστημα διαχείρισης ροών δεδομένων όπου συνεχείς επερωτήσεις θα εκτελούνται πάνω σε πολλαπλές εισερχόμενες ροές και αποθηκευμένους σχεσιακούς πίνακες. Έχει ήδη γίνει αναφορά στην αρχή αυτού του κεφαλαίου για τη γλώσσα συνεχών επερωτήσεων CQL, η οποία δημιουργήθηκε στα πλαίσια του ίδιου προγράμματος με το STREAM, και είναι βασισμένη σε

συγκεκριμένη σημασιολογία και τη γλώσσα SQL. Η CQL υποστηρίζει επερωτήσεις ανάμεσα σε ροές αλλά και συνδυασμούς πινάκων και ροών, μετασχηματισμούς πάνω στις ροές και τους πίνακες, πλαίσια κλπ. Η γενική αρχιτεκτονική του STREAM φαίνεται στο Σχήμα 4.2.



Σχήμα 4.2 – Η αρχιτεκτονική του STREAM

Στην αρχιτεκτονική του STREAM διακρίνει κανείς αρχικά τις εισερχόμενες ροές στα αριστερά και το τμήμα Scratch Store το οποίο είναι υπεύθυνο για την ενδιάμεση αποθήκευση των εισερχόμενων ροών. Η αποθήκευση αυτή ανάλογα με τις εκάστοτε ανάγκες της επεξεργασίας μπορεί να είναι είτε η μνήμη είτε και ο δίσκος. Ακόμη, φαίνεται η καταχώρηση συνεχών επερωτήσεων από τους χρήστες (Register Query), η επιστροφή των αποτελεσμάτων αυτών (Streamed Results), οι αποθηκευμένοι σχεσιακοί πίνακες (Stored Relations) και το κεντρικό σύστημα διαχείρισης (DSMS).

Η επεξεργασία των συνεχών επερωτήσεων στο STREAM ξεκινά με τη δημιουργία ενός πλάνου εκτέλεσης, το οποίο, στη συνέχεια, συγχωνεύεται με τα υπόλοιπα πλάνα των υπαρχόντων επερωτήσεων, έτσι ώστε να γίνει ο μεγαλύτερος δυνατός διαμοιρασμός πόρων. Ενδιαφέρον είναι το γεγονός ότι το πλάνο εκτέλεσης μπορεί να προταθεί από τον ίδιο το χρήστη σε XML μορφή. Το πλάνο εκτέλεσης περιέχει τριών ειδών τμήματα:

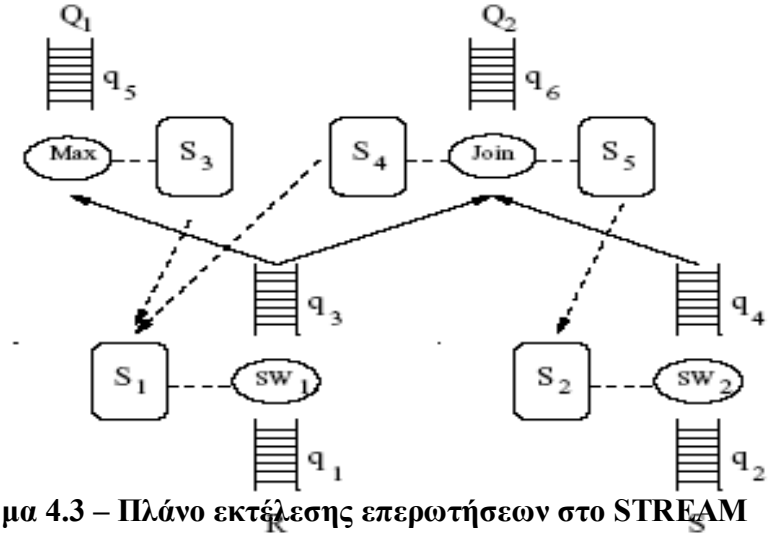
- Τελεστές επερωτήσεων: είναι οι τελεστές που έχουν ήδη αναφερθεί στη CQL (*Ροής-προς-Σχέση, Σχέσης-προς-Σχέση, Σχέσης-προς-Ροή*), οι τελεστές πλαισίων (προσδιοριζόμενα με χρόνο, με το πλήθος πλειάδων και τμηματοποιημένα) και τελεστές δειγματοληψίας για την προσεγγιστική απάντηση επερωτήσεων.

- Οι ουρές αναμονής ανάμεσα στους τελεστές, που αποθηκεύουν προσωρινά τις πλειάδες των ενδιάμεσων αποτελεσμάτων των επερωτήσεων. Μαζί με τις συνόψεις αποτελούν το τμήμα Scratch Store της αρχιτεκτονικής.

- Συνόψεις, που διατηρούν ενδιάμεσα τρέχοντα στοιχεία των πλειάδων που χρειάζεται να τροφοδοτούνται στους τελεστές. Λέγοντας συνόψεις στο STREAM εννοούνται γενικότερα οι δομές αναπαράστασης πλειάδων όπως ένα κυλιόμενο πλαίσιο ή μία περιληπτική αναπαράστασή τους, όπως αυτές που αναφέρθηκαν στο προηγούμενο κεφάλαιο.

Στο Σχήμα 4.3 φαίνεται ένα ενδεικτικό πλάνο εκτέλεσης δύο επερωτήσεων Q_1 και Q_2 πάνω στις ροές R και S. Οι τελεστές SW είναι κυλιόμενα πλαίσια, ο Max είναι η επιλογή του μεγίστου, τα S_i είναι συνόψεις, τα q_i είναι ουρές αναμονής πλειάδων και ο Join είναι η γνωστή συνένωση. Έτσι, η ουρά q_1 αναπαριστά ουσιαστικά τη ροή R και η ουρά q_3 το κυλιόμενο πλαίσιο της R. Η σύνοψη S_1 είναι η σύνοψη του κυλιόμενου πλαισίου της R. Όσον αφορά τον τελεστή Max, αυτός στην προσπάθειά του να υπολογίσει το μέγιστο έχει ανάγκη από την περιοδική χρήση παλαιότερων πλειάδων και γι' αυτό διατηρεί τη σύνοψη S_3. Επειδή, όμως η S_3 περιέχει ουσιαστικά στοιχεία που έχουν προέλθει από την S_1 είναι δυνατόν οι πόροι αυτών των δύο συνόψεων να αξιοποιηθούν με κοινό τρόπο και να διαμοιραστούν. Αυτό αναδεικνύει το διακεκομμένο βέλος από την S_3 προς την S_1. Αντίστοιχα ισχύουν για τις υπόλοιπες συνόψεις, ουρές και τελεστές.

Ο προγραμματισμός των τελεστών στο STREAM γίνεται μέσω μιας τεχνικής που καλείται αλυσιδωτή (Chain Scheduling). Ο πιο σημαντικός πόρος στην αντιμετώπιση αυτή είναι η διαθέσιμη μνήμη του συστήματος επεξεργασίας. Αυτό, γιατί στις περιόδους μεγάλων ρυθμών άφιξης των ροών εάν δεν είναι επιθυμητό να γίνει απόρριψη πλειάδων τμήματά τους διοχετεύονται στο δίσκο για να υποστούν επεξεργασία αργότερα, κάτι που επιβραδύνει την όλη απόκριση τους συστήματος.



Σχήμα 4.3 – Πλάνο εκτέλεσης επερωτήσεων στο STREAM

Ο αλυσιδωτός προγραμματισμός των τελεστών προσπαθεί να εκμηδενίσει αυτό το φαινόμενο. Η λογική που ακολουθείται είναι ότι τα πλάνα των επερωτήσεων διασπώνται σε ανεξάρτητες αλυσίδες από διαδοχικούς τελεστές με βάση την ικανότητα του καθενός να απορροφά γρήγορα πλειάδες. Αυτό σημαίνει ότι προτιμώνται οι τελεστές που σε σύντομο χρονικό διάστημα μπορούν να δεχθούν πολλές πλειάδες και να παράγουν σαν αποτέλεσμα λίγες.

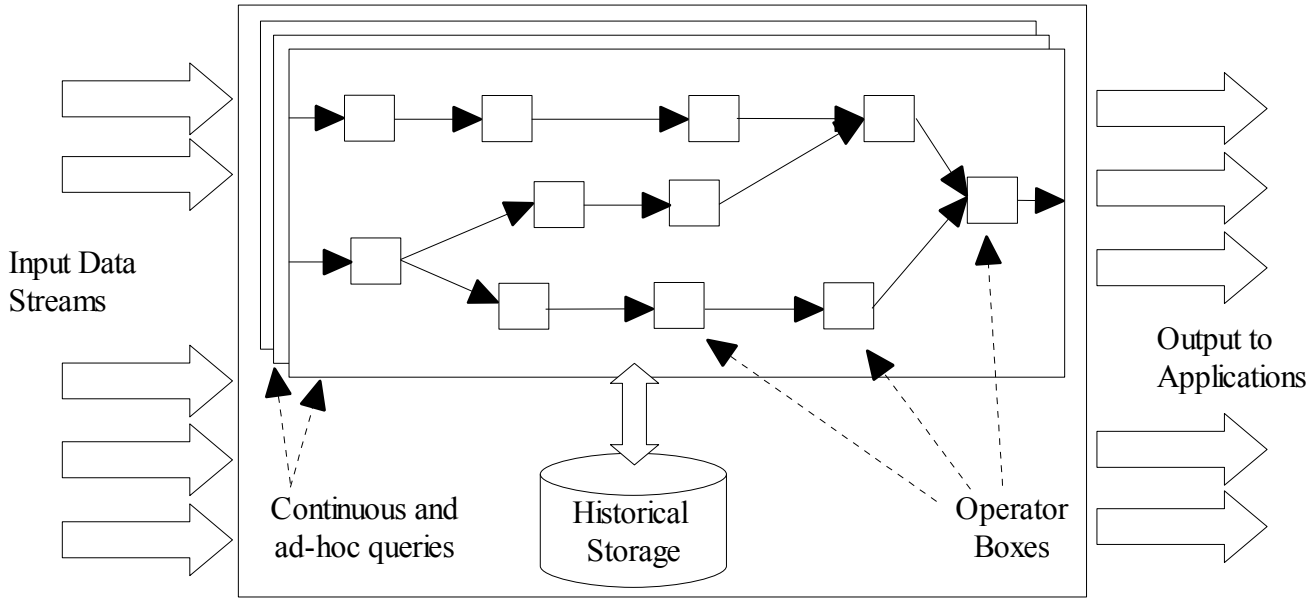
Το STREAM αναγνωρίζει τις δυσκολίες που υπάρχουν όσον αφορά τη διαχείριση μεγάλων όγκων ροών δεδομένων και εφαρμόζει κατά την εξέλιξή του όλο και περισσότερες τεχνικές βελτιστοποίησης βάσει κόστους, διαχείρισης πόρων, προγραμματισμού, προσεγγιστικών αποτελεσμάτων και κατανεμημένης επεξεργασίας. Μέχρι στιγμής έχει αποδειχθεί ένα πολύ σημαντικό ερευνητικό σύστημα διαχείρισης ροών, κατά τη διάρκεια της ανάπτυξης του οποίου έχουν προταθεί πολλές νέες ιδέες στο χώρο, ενώ στα πλαίσιά του έχει προταθεί και ως ένα βαθμό υλοποιηθεί η CQL. Ακόμη, βέβαια, δεν έχει βρεθεί σε ιδιαίτερα ώριμο επίπεδο. Σημειώνεται ότι το STREAM φαίνεται να έχει σταματήσει την ανάπτυξή του στις αρχές του 2006 κάτι που δεν ήταν γνωστό κατά τη σύνταξη των περιεχομένων της εργασίας αυτής. Παρολαυτά, παραμένει ένα ερευνητικό έργο που είχε σημαντική επιρροή στο πεδίο της διαχείρισης ροών δεδομένων, ενώ δεν είναι γνωστό ακόμη αν θα έχει συνέχεια.

4.2.3 ΤΟ ΣΎΣΤΗΜΑ AURORA

Το τελευταίο Σύστημα Διαχείρισης Ροών Δεδομένων στο οποίο θα γίνει αναφορά είναι το Aurora [55][9][11] το οποίο έχει εξελιχθεί σε μία κατανεμημένη εκδοχή του, το Borealis [62][63]. Το Aurora είναι ένα ερευνητικό πρόγραμμα που επιχειρεί να αναπτύξει μια πλήρη εφαρμογή παρακολούθησης (monitoring) ροών δεδομένων, κάτι που αναγνωρίζεται πως δεν είναι δυνατόν να επιτευχθεί με ένα παραδοσιακό σύστημα βάσεων δεδομένων όπου τα δεδομένα αποθηκεύονται στατικά, οι επερωτήσεις τίθενται σε διακριτές χρονικές στιγμές και τα αποτελέσματα των επερωτήσεων πρέπει να είναι πλήρη και ακριβή. Ο ρόλος ενός τέτοιου συστήματος ορίζεται ως Ενεργού-Συστήματος-Παθητικού-Χρήστη (DBMS-Active-Human Passive, DAHP) σε αντίθεση με τα παραδοσιακά ΣΔΒΔ που ορίζονται ως Ενεργού-Χρήστη-Παθητικού-Συστήματος (Human-Active-DBMS-Passive, HADP). Η θεμελιώδης προσέγγιση που ακολουθείται, επομένως, είναι αυτή του ενεργού συστήματος παρακολούθησης ροών δεδομένων με εξειδικευμένους σε ροές αλγορίθμους διαχείρισης και την αποδοχή της αναγκαιότητας για προσεγγιστικό υπολογισμό των συνεχών επερωτήσεων.

Τα δεδομένα που ρέουν στην είσοδο του Aurora θεωρούνται πως έρχονται από διάφορες

ετερογενείς πηγές, όπως άλλα υπολογιστικά συστήματα ή αισθητήρες. Μια ροή δεδομένων (data stream) νοείται ως η απεικόνιση των δεδομένων που εισέρχονται από πολλές πηγές δεδομένων (data sources). Κάθε πηγή δεδομένων έχει ένα μοναδικό αναγνωριστικό και κάθε πλειάδα που εισέρχεται στο ΣΔΡΔ χρονοσημαίνεται ξεχωριστά. Η διαχείριση των ροών γίνεται μέσω ενός γραφικού περιβάλλοντος διαχείρισης, όπου διάφοροι τελεστές σε μορφή κουτιών (boxes) σχηματίζουν έναν ακυκλικό γράφο που καθορίζει τη διαδρομή της κάθε ροής μέσα στο σύστημα. Η έξοδος του συστήματος αναφέρεται σε εφαρμογές (applications) που υποδέχονται τις εξερχόμενες ροές. Η αρχιτεκτονική αυτή φαίνεται στο Σχήμα 4.4.



Σχήμα 4.4 – Αρχιτεκτονική του Aurora

Πρώτιστο μέλημα στην εκτέλεση των επερωτήσεων είναι η ελαχιστοποίηση των περασμάτων που χρειάζεται να γίνουν σε μεγάλες ροές δεδομένων. Μια από τις μεθοδολογίες που το Aurora χρησιμοποιεί είναι η δυναμική βελτιστοποίηση των επερωτήσεων (dynamic query optimization), κάτι που σημαίνει ότι η εκτέλεση των επερωτήσεων ξεκινά με μη βέλτιστο τρόπο και βελτιστοποιείται στη συνέχεια χρησιμοποιώντας πληροφορίες όπως στατιστικά στοιχεία και επιλεξιμότητες τελεστών. Ακόμη υιοθετείται όπου αυτό είναι αναγκαίο και η απόρριψη πλειάδων είτε τυχαιοκρατικά, χωρίς εξέτασή τους, είτε μέσω κάποιου σημασιολογικού φιλτραρίσματος, όπως η απόρριψη πλειάδων που που δε συνεισφέρουν σημαντικά στο αποτέλεσμα, κάτι που εντοπίζεται μέσω πληροφορίας ποιότητας του αποτελέσματος.

ΔΕΎΤΕΡΟ ΜΈΡΟΣ

5. ΜΕΛΈΤΗ ΤΟΥ ΠΕΔΊΟΥ ΤΗΣ ΠΑΡΑΓΩΓΉΣ ΔΟΚΙΜΑΣΤΙΚΏΝ ΡΟΏΝ ΔΕΔΟΜΈΝΩΝ

Το επιστημονικό πεδίο της διαχείρισης των ροών δεδομένων είναι ταχύτατα και ευρύτατα εξελισσόμενο, όπως έχουμε ήδη διαπιστώσει. Υπάρχει πληθώρα εφαρμογών που χρήζουν ιδιαίτερης αντιμετώπισης εξαιτίας του ότι τα δεδομένα που εκμεταλλεύονται εμφανίζονται σε ροές και η ερευνητική δραστηριότητα είναι για αυτό το λόγο στο χώρο έντονη. Ως τώρα έχουμε εντοπίσει πολλές από τις ιδιότητες των ροών δεδομένων που αναδεικνύουν τη διαφορετικότητα τους από τα "στατικά" δεδομένα των παραδοσιακών εφαρμογών βάσεων δεδομένων και τις ξεχωριστές συνθήκες που τις διέπουν.

Ωστόσο, παρόλο που στην ακαδημαϊκή κοινότητα η έρευνα σημειώνει συνεχείς προόδους μπορεί να διακρίνει κανείς ακόμη συγκεκριμένους τομείς όπου υπάρχει ελεύθερος χώρος για καινοτομία και προσφορά. Με αυτή τη σκέψη, στα πλαίσια της παρούσας πτυχιακής εργασίας, επιλέχθηκε η μελέτη και η πρωτογενής ανάπτυξη ενός εργαλείου παραγωγής ροών δεδομένων. Η αναγκαιότητα για ένα παρόμοιο εργαλείο διαπιστώνεται από παρατηρήσεις όπως το ότι στα περισσότερα ερευνητικά Συστήματα Διαχείρισης Ροών Δεδομένων, σε μερικά από τα οποία έγινε αναφορά σε προηγούμενο κεφάλαιο, διακρίνεται έλλειψη δοκιμαστικών δεδομένων που να εμφανίζονται με μορφή ροών. Χωρίς ικανές ποσότητες δεδομένων που να μπορούν να παραμετροποιηθούν κατά βούληση, τα συστήματα αυτά έχουν ένα μειονέκτημα και τα αποτελέσματα που επιτυγχάνουν θα μπορούσαν να είναι περισσότερο πλήρη και αποδεδειγμένης αξίας.

Το εργαλείο για το οποίο γίνεται εδώ λόγος είναι ένα λογισμικό που σε πραγματικό χρόνο θα μπορεί να παράγει ροές δεδομένων με μια πληθώρα παραμέτρων αναγκαίων ώστε οι παραγόμενες ροές να ταιριάζουν στις ανάγκες των εκάστοτε δοκιμών. Στη συνέχεια του κεφαλαίου αυτού θα γίνει λόγος για τα είδη των ροών δεδομένων που εντοπίζονται σε ερευνητικό και εφηρμοσμένο επίπεδο στις μέρες μας. Ακόμη, θα γίνει αναφορά στο "Linear Road Project" και το “NEXMark”, δύο από τα ελάχιστα εργαλεία δοκιμών (benchmarking) για συστήματα διαχείρισης ροών δεδομένων σήμερα. Στη συνέχεια θα αναλυθεί τόσο από πλευράς ανάλυσης και προδιαγραφών όσο και από πλευράς υλοποίησης το εργαλείο που αναπτύχθηκε στα πλαίσια της παρούσας εργασίας. Τέλος, θα υπάρξει παρουσίαση των λειτουργιών αυτού καθώς και των αποτελεσμάτων που παράγει και θα γίνει σύντομη αναφορά σε μελλοντικές δυνατότητες ολοκλήρωσης και εξέλιξής του.

5.1 Η ΑΝΑΓΚΑΙΌΤΗΤΑ ΓΙΑ ΕΡΓΑΛΕΊΑ ΠΑΡΑΓΩΓΉΣ ΡΟΏΝ ΔΕΔΟΜΈΝΩΝ

Είναι άξιο παρατήρησης να αναζητήσει κανείς με τι είδους δοκιμαστικά δεδομένα γίνεται ο έλεγχος της απόδοσης των διαφόρων ερευνητικών συστημάτων διαχείρισης ροών δεδομένων αλλά και των αλγορίθμων που κατά καιρούς προτείνονται στο χώρο (πολλά από αυτά τα συστήματα και αλγορίθμους έχουν εξεταστεί και στα πλαίσια αυτής της εργασίας). Διαφαίνεται ότι πολλές φορές οι δοκιμές γίνονται με λίγα δοκιμαστικά δεδομένα τα οποία λαμβάνονται από κάποιο αρχείο και τροφοδοτούν το υπό δοκιμή σύστημα. Μια τέτοια προσέγγιση, παρόλο που μπορεί να βοηθήσει και να αποδώσει χρήσιμα συμπεράσματα υπολείπεται σημαντικά μιας πλήρους και συμπαγούς δοκιμής. Ένα εκκολαπτόμενο ΣΔΡΔ οφείλει να υποστεί εκτεταμένες δοκιμές σε συνθήκες όσο το δυνατόν πιο κοντά στις πραγματικές και σε μεγάλη ποικιλία δεδομένων που να του δίνουν τη δυνατότητα να γενικεύσει τα αποτελέσματά του.

Η αναγκαιότητα για ένα εργαλείο παραγωγής ροών δεδομένων έχει, επομένως, δύο κύριες συνιστώσες. Η πρώτη είναι η κατανόηση και καταγραφή των ειδών των ροών δεδομένων που παρουσιάζονται σήμερα τόσο σε ερευνητικό όσο και σε εφηρμοσμένο επίπεδο, των συσχετισμών ανάμεσά τους αλλά και του τρόπου με τον οποίο παράγονται. Αυτό σημαίνει πως όλες οι ροές δεν είναι εντελώς όμοιες μεταξύ τους και αυτό επειδή παρουσιάζονται διαφορές που αφορούν στις πηγές από τις οποίες παράγονται και στη σημασιολογία που τα δεδομένα εμπεριέχουν από τη φύση της κάθε εφαρμογής. Ενδεικτικά θα μπορούσε να αναφέρει κανείς ότι εφαρμογές όπως τα δίκτυα αισθητήρων (sensor networks) παράγουν δεδομένα που τις περισσότερες φορές δε συσχετίζονται μεταξύ τους. Αυτό γιατί κάθε αισθητήρας παράγει μετρήσεις ανεξάρτητες από τους υπόλοιπους και τις προωθεί στο σύστημα διαχείρισης. Η τεχνολογία, όμως, των αισθητήρων εξελίσσεται ραγδαία σε σημείο που σύντομα θα μπορούμε να μιλάμε για αισθητήρες που θα μπορούν να "τρέχουν" μέσα τους μικρά λειτουργικά συστήματα και να δημιουργούν αναμεταξύ τους δίκτυα δεδομένων και να επικοινωνούν. Έτσι η προηγούμενη θεώρηση, ότι στα δίκτυα αισθητήρων τα δεδομένα είναι ανεξάρτητα μεταξύ τους, αλλάζει μορφή, μιας και οι πηγές των ροών θα μπορούν να επικοινωνούν και να παράγουν δεδομένα με περισσότερο συνεργατικό τρόπο. Και οι δύο προσεγγίσεις στην προκειμένη περίπτωση έχουν η κάθε μία τις δικές της ιδιαιτερότητες.

Η δεύτερη συνιστώσα προέρχεται από την ανάγκη των ερευνητικών προγραμμάτων διαχείρισης δεδομένων για ύπαρξη δοκιμαστικών δεδομένων. Κάθε τέτοιο πρόγραμμα όσο εξειδικευμένο ή γενικού σκοπού και αν είναι έχει την ανάγκη να δεχθεί σαν είσοδο ροές δεδομένων που να πλησιάζουν τις πραγματικές συνθήκες για να μπορεί να εξαχθούν χρήσιμα συμπεράσματα.

Επιπλέον, είναι σημαντικό να μπορούν να έχουν στη διάθεσή τους δεδομένα που να ανταποκρίνονται σε προδιαγραφές που δεν έχουν καν ακόμη παρουσιαστεί σε πραγματικές συνθήκες, είτε επειδή δεν υπάρχει πρόσβαση σε παρόμοια δεδομένα είτε επειδή προβλέπεται ότι οι συνθήκες αυτές θα εμφανιστούν στο κοντινό μέλλον. Για παράδειγμα, η τροφοδότηση ενός ΣΔΡΔ με ελεγχόμενους ρυθμούς εισόδου των δεδομένων είναι μια πολύ θεμελιώδης ανάγκη για κάθε ΣΔΡΔ ώστε να μπορεί να δοκιμάσει τα όρια της αντοχής των αλγορίθμων του. Αυτού του είδους αλλά και άλλες ανάγκες επιχειρεί να καλύψει ένα εργαλείο παραγωγής ροών δεδομένων. Στη συνέχεια, θα ακολουθήσει μια επιγραμματική αναφορά στα είδη των ροών δεδομένων που εντοπίζονται μέχρι σήμερα και τις ιδιαιτερότητες που παρουσιάζουν αυτές και οι αντίστοιχες εφαρμογές τους.

5.2 ΕΊΔΗ ΡΟΏΝ ΔΕΔΟΜΈΝΩΝ ΣΤΗΝ ΠΡΆΞΗ

Στις μέρες μας διακρίνουμε πληθώρα εφαρμογών που δημιουργούν δεδομένα τα οποία εμφανίζονται σε μορφή ροών, δεδομένα, δηλαδή, που παράγονται ατέρμονα από διάφορες πηγές και κάποιο σύστημα που τα υποδέχεται καλείται να τα διαχειριστεί με το βέλτιστο τρόπο. Ανάλογα με τη φύση της εφαρμογής και το στάδιο εξέλιξης στο οποίο βρίσκεται η εκάστοτε τεχνολογία οι ροές αυτές ποικίλλουν όσον αφορά τη μορφή αλλά και τη μορφοποίηση της πληροφορίας που φέρουν. Έχουν η κάθε μία τις δικές της ιδιαιτερότητες αν και μοιράζονται πολλά κοινά σημεία. Η γνώση αυτών είναι πολύ σημαντική για την κατανόηση των ίδιων των ροών αλλά και εξαιρετικά γόνιμη στη διερεύνηση των προδιαγραφών για ένα εργαλείο παραγωγής δοκιμαστικών ροών δεδομένων.

5.2.1 ΡΟΈΣ ΑΠΌ ΔΊΚΤΥΑ ΑΙΣΘΗΤΉΡΩΝ (SENSOR NETWORKS)

Τα δίκτυα αισθητήρων (sensor networks) αποτελούν μια από τις ταχύτερα εξελισσόμενες αλλά και πιο πολυσυζητημένες τεχνολογίες στις μέρες μας. Πρόκειται για δίκτυα που δημιουργούνται από πολυάριθμες μικροσκοπικές συσκευές κάθε μία από τις οποίες ενσωματώνει πάνω της έναν ή περισσότερους αισθητήρες εξειδικευμένους για κάποιου είδους μέτρηση, όπως η θερμοκρασία, η πίεση, ο φωτισμός κλπ. Οι συσκευές αυτές, ακόμη, περιέχουν κάποιο λογισμικό που αναλαμβάνει τον έλεγχο της λειτουργίας τους και ένα υποσύστημα με μπαταρίες που τους κρατά σε λειτουργία. Επιπλέον, επικοινωνούν ασύρματα με κάποιο κεντρικό σύστημα διαχείρισης στο οποίο και ανακοινώνουν σε τακτικά διαστήματα τις μετρήσεις που ανιχνεύουν. Το τεράστιο ποσοστό ολοκλήρωσης τους σε συνδυασμό με το μικρό τους μέγεθος και την ολοένα και πιο εφικτή μαζική παραγωγή τους κάνουν τις εφαρμογές που τους αξιοποιούν εξαιρετικά κλιμακώσιμες και

ελπιδοφόρες [66].

Μερικές χαρακτηριστικές εφαρμογές δικτύων αισθητήρων είναι οι εξής:

- Παρακολούθηση εξ αποστάσεως δυσπρόσιτων περιοχών, όπως περιβάλλοντα οικολογικού ενδιαφέροντος, λίμνες, δάση κλπ [9].

- Παρακολούθηση μεγάλων εκτάσεων, όπως αυτοκινητόδρομοι, γέφυρες, εργοστασιακές εγκαταστάσεις, πυρηνικά εργοστάσια κλπ [66].

- Παρακολούθηση κτηριακών υποδομών τόσο από πλευράς ελέγχου λειτουργιών αλλά και από πλευράς ασφάλειας, ενεργειακής διαχείρισης κλπ [39].

- Εφαρμογές παρακολούθησης στρατιωτικών επιχειρήσεων [9].

- Εφαρμογές παρακολούθησης της εξέλιξης και προόδου των εκάστοτε λειτουργιών στις διάφορες φάσεις της εφοδιαστικής αλυσίδας (supply chain) [66].

- Ιατρικές εφαρμογές παρακολούθησης στοιχείων της υγείας ασθενών όταν αυτοί δε μπορούν να τύχουν συνεχούς ιατρικής παρουσίας [65].

Βέβαια, η παραπάνω λίστα μόνο από τα όρια της ανθρώπινης φαντασίας μπορεί να περιοριστεί, μιας και η τεχνολογία των αισθητήρων δείχνει πως όλο και περισσότερες δυνατότητες αναδεικνύονται και γίνονται διαθέσιμες στο χώρο. Κάποιες από αυτές τις δυνατότητες είναι η ενσωμάτωση μικρών λειτουργικών συστημάτων στους αισθητήρες, όπως αναφέρθηκε και παραπάνω, η δυνατότητα ασύρματης επικοινωνίας ανάμεσα στους ίδιους τους αισθητήρες – και όχι μόνο ανάμεσα στον καθένα και το κεντρικό σύστημα διαχείρισης – και η ικανότητα ενεργοποίησης δράσεων από αυτούς πέρα από την απλή αναφορά μετρήσεων.

Το πιο σύνηθες σενάριο σε ένα δίκτυο αισθητήρων είναι το εξής: οι αισθητήρες τοποθετούνται σε στρατηγικά σημεία του χώρου τον οποίο καλούνται να παρακολουθήσουν και κατά τακτικά χρονικά διαστήματα αναφέρουν τις μετρήσεις που ανιχνεύουν σε ένα τοπικό σύστημα υποδοχής με ασύρματη επικοινωνία. Το κεντρικό αυτό σύστημα συλλέγει τις μετρήσεις των αισθητήρων κι έπειτα από κάποια πιθανή επεξεργασία τις προωθεί σε κάποιο κεντρικό. Με αυτό τον τρόπο συνεχείς και τακτικές μετρήσεις καταφθάνουν στο κεντρικό σύστημα προς επεξεργασία με τη

μορφή μιας ροής δεδομένων.

Η ροή αυτή αποτελείται από πλειάδες (tuples) που αντιστοιχούν στις μετρήσεις των αισθητήρων. Ανάλογα με την κλιμάκωση της εφαρμογής οι ρυθμοί άφιξης των δεδομένων μπορούν να είναι και πολύ υψηλοί και τα δεδομένα να εμφανίζονται με μαζικό τρόπο. Για παράδειγμα σε μια εφαρμογή παρακολούθησης κτηρίων οι αισθητήρες είναι αναγκαίο να αναφέρουν μετρήσεις τακτικότερα από ότι σε μια εφαρμογή ανίχνευσης πυρκαγιών σε μια δασική έκταση. Στη δεύτερη περίπτωση, όμως, το πλήθος των αισθητήρων που θα είναι διεσπαρμένοι στο δάσος είναι δυνατόν να είναι πάρα πολύ μεγάλο στο σύνολό του. Σε κάθε περίπτωση, πάντως, ο όγκος των δεδομένων που πρέπει να υποστούν επεξεργασία κεντρικά μπορεί να είναι εξαιρετικά υψηλός και να εμφανίζεται και με μεγάλους ρυθμούς. Μερικές από τις ιδιότητες και ιδιαιτερότητες των δεδομένων σε παρόμοιες εφαρμογές είναι οι εξής:

- Τα δεδομένα από τέτοιες εφαρμογές τις περισσότερες φορές ακολουθούν το σχεσιακό μοντέλο όπου οι πλειάδες των μετρήσεων των αισθητήρων υπακούν σε κάποια δεδομένη γραμμογράφηση/σχήμα.

- Τα δεδομένα που παράγονται είναι συνήθως ανεξάρτητα μεταξύ τους μιας και οι αισθητήρες λειτουργούν οι ίδιοι ανεξάρτητα ο ένας από τον άλλον. Ο καθένας είναι υπεύθυνος για μία (και σε λίγες περιπτώσεις για περισσότερες από μία) μετρήσεις τις οποίες επαναλαμβάνουν τακτικά και χωρίς συνεννόηση με τους υπολοίπους. Οι εξελίξεις δείχνουν ότι αυτό σύντομα δε θα είναι, βέβαια, το μόνο σενάριο, καθώς οι αισθητήρες θα μπορούν να επικοινωνούν μεταξύ τους, να συνεργάζονται και να σχηματίζουν πραγματικά δίκτυα.

- Μία από τις ιδιαιτερότητες που εμφανίζονται, όμως, είναι φαινόμενα όπως η άτακτη χρονικά άφιξη των πλειάδων λόγω απρόβλεπτων καθυστερήσεων του δικτύου. Σε μία εφαρμογή όπου οι αισθητήρες είναι μοιρασμένοι σε μεγάλες εκτάσεις, και μάλιστα πολύ μακριά από το σύστημα διαχείρισης, οι μετρήσεις μπορεί εύκολα να καθυστερήσουν ή να μη φτάσουν με σωστή χρονική διαδοχή ή ακόμη και κάποιες από αυτές να χαθούν.

Είναι εύκολα κατανοητό ότι στην πρώτη αυτή κατηγορία ροών δεδομένων που διακρίνουμε αντιστοιχεί ένα αρκετά μεγάλο εύρος εφαρμογών, το οποίο, άλλωστε, και καταγράφηκε ως ένα βαθμό. Οι ροές αυτές αποτελούν μια αρκετά γενικευμένη κατηγορία με πρωταρχικό κοινό σημείο το γεγονός ότι οι παραγόμενες πλειάδες προέρχονται από τις μετρήσεις των αισθητήρων. Ακόμη, οι

εφαρμογές δικτύων αισθητήρων κλιμακώνονται σε πολύ μεγάλο βαθμό με συνέπεια τα παραγόμενα δεδομένα να φτάνουν σε ποσότητες πολύ δύσκολα διαχειρίσιμες κεντρικά. Ένα εργαλείο παραγωγής δοκιμαστικών ροών δεδομένων οφείλει να μπορεί να προσομοιώσει αυτή τη συμπεριφορά της μαζικής παραγωγής και της κλιμάκωσης με ενιαίο τρόπο.

5.2.2 ΡΟΈΣ ΑΠΌ ΔΊΚΤΥΑ ΔΕΔΟΜΈΝΩΝ (DATA NETWORKS)

Ακόμη ένα σημαντικό πεδίο όπου οι ροές δεδομένων βρίσκουν εφαρμογή είναι στη διαχείριση των δικτύων δεδομένων [56]. Πρόκειται για τα γνωστά σε όλους δίκτυα που επιτρέπουν την επικοινωνία ανάμεσα στους υπολογιστές και τα υπολογιστικά συστήματα εδώ και πολλές δεκαετίες. Τα δίκτυα αυτά βασίζονται πάνω στο γνωστό πρωτόκολλο διευθυνσιοδότησης υπολογιστών IP και τα πρωτόκολλα μεταφοράς TCP και UDP.

Είναι εύκολα κατανοητό πως σε αυτά τα δίκτυα τα δεδομένα που διακινούνται είναι τεράστια και μαζικά και μάλιστα ρέουν ασταμάτητα ακόμη και όταν ένα τέτοιο δίκτυο είναι φαινομενικά ανενεργό από πλευράς διαμεταγωγής ωφέλιμων δεδομένων. Επιπλέον, η μαζικότητα των δεδομένων που κυκλοφορούν στα δίκτυα δεδομένων γίνεται ακόμη πιο έντονη αν φανταστεί κανείς τα μεγέθη των δικτύων μεγάλων οργανισμών και των παροχέων του διαδικτύου (ISP's – Internet Service Providers). Όλα τα ιδιαίτερα προβλήματα που παρουσιάζουν οι ροές δεδομένων (μεγάλοι ρυθμοί άφιξης, αδυναμία συνολικής διαχείρισης των δεδομένων, πιθανότητα απώλειας δεδομένων, απροσδιόριστες καθυστερήσεις άφιξης κλπ) γίνονται περισσότερο αισθητά, ίσως από κάθε άλλη περίπτωση, λόγω αυτής ακριβώς της μαζικότητας των δεδομένων που δυσχεραίνουν πολύ τη διαχείρισής τους.

Ένας οργανισμός που χρησιμοποιεί ένα δίκτυο δεδομένων θέλει να γνωρίζει όσο το δυνατόν περισσότερα στοιχεία για την πληρέστερη διαχείριση του δικτύου του. Βέβαια, ο χώρος της διαχείρισης δικτύων δεν είναι νέος και εξειδικευμένα εργαλεία έχουν εμφανιστεί κατά καιρούς για να εξυπηρετήσουν τις ανάγκες του. Η "ρέουσα" φύση, όμως, της πληροφορίας που διατρέχει τα δίκτυα αυτά είναι δεδομένη μιας και τα δεδομένα πρέπει να υποστούν επεξεργασία σε πραγματικό χρόνο χωρίς δυνατότητα να αποθηκευθούν πλήρως για μετέπειτα επεξεργασία, λόγω του μεγάλου όγκους τους. Μερικά από τα συμπεράσματα που είναι χρήσιμο να εξάγονται για ένα δίκτυο δεδομένων για διαχειριστικούς και στατιστικούς σκοπούς είναι τα ακόλουθα [67]:

- Ο φόρτος του δικτύου τόσο ανά πάσα στιγμή όσο και ανά δοσμένες περιόδους. Η διακύμανση του φόρτου είναι εξαιρετικά σημαντική για τη διαχείριση ενός δικτύου, έτσι

ώστε να μπορούν να γίνουν αντιληπτά τα σημεία συμφόρησης ή/και υπό-λειτουργίας, τα τμήματα του δικτύου που παρουσιάζουν υψηλό φόρτο καθώς και οι περιοχές του δικτύου και οι χρονικές περίοδοι στις οποίες συμβαίνουν αυτά.

- Η ανίχνευση των συχνότερα χρησιμοποιούμενων εφαρμογών και κατά συνέπεια των θυρών (ports) που αυτές δεσμεύουν. Οι θύρες αυτές δίνουν τις περισσότερες φορές πληροφορία και για το πρωτόκολλο που χρησιμοποιείται (HTTP, FTP, SMTP κοκ).

- Ο προσδιορισμός της ποιότητας του δικτύου μέσω του εντοπισμού των χαμένων πακέτων και του ποσοστού τους επί του συνόλου της διακινούμενης πληροφορίας, της πολλαπλής αποστολής των ίδιων πακέτων, της διάρκειας των συνδέσεων (connections) και των συνόδων (sessions), της ταχύτητας διαμεταγωγής τω δεδομένων κλπ.

- Διάφορα ιδιάζοντα χαρακτηριστικά των των συνδέσεων ανάμεσα σε υπολογιστές πελάτες (clients) και υπολογιστές εξυπηρετητές (servers), όπως η διάρκεια των συνδέσεων μεταξύ τους, οι χρόνοι απόκρισης, οι ποσότητες των δεδομένων που διακινούνται ανά σύνδεση, οι εφαρμογές που τις εκκινούν κλπ.

- Εντοπισμός επιθέσεων κατά της ακεραιότητας ή διαθεσιμότητας του δικτύου μέσω της ανίχνευσης παράτυπων ή ασυνήθιστων συμπεριφορών, κάτι απαραίτητο στον τομέα της ασφάλειας κάθε δικτύου.

Τα δεδομένα που διακινούνται στα δίκτυα δεδομένων είναι, όπως προαναφέρθηκε, TCP και UDP πακέτα τα οποία περιέχουν επικεφαλίδες με πληροφορία χρήσιμη για την επικοινωνία ανάμεσα σε δύο ή περισσότερα συστήματα και το συγχρονισμό της αλλά και την ωφέλιμη πληροφορία που μεταφέρεται. Ένα σύστημα διαχείρισης ροών δεδομένων αν και παρακολουθεί όλη τη ροή αυτής της πληροφορίας εκμεταλλεύεται κυρίως τις επικεφαλίδες των πακέτων για να εξάγει τα επιθυμητά συμπεράσματα, μιας και οι επικεφαλίδες αυτές εμπεριέχουν εκ των πραγμάτων ό,τι χρειάζεται από πλευράς διαχείρισης της μεταφερόμενης πληροφορίας. Μια τέτοια αντιμετώπιση αλαφραίνει το φορτίο που ένα ΣΔΡΔ θα όφειλε να διαχειριστεί. Έχει, παρολαυτά, το μειονέκτημα ότι αδιαφορεί για την πληροφορία που καταγράφεται στις επικεφαλίδες των πρωτοκόλλων επιπέδου εφαρμογής (application layer). Με αυτό τον τρόπο δεν είναι δυνατή η παρακολούθηση της πληροφορίας που μεταφέρεται από εφαρμογές που χρησιμοποιούν το επίπεδο εφαρμογής, όπως εφαρμογές βάσεων δεδομένων, συνδέσεις HTTP, συνδέσεις μέσα σε εικονικά ιδιωτικά δίκτυα (VPN's) κλπ.

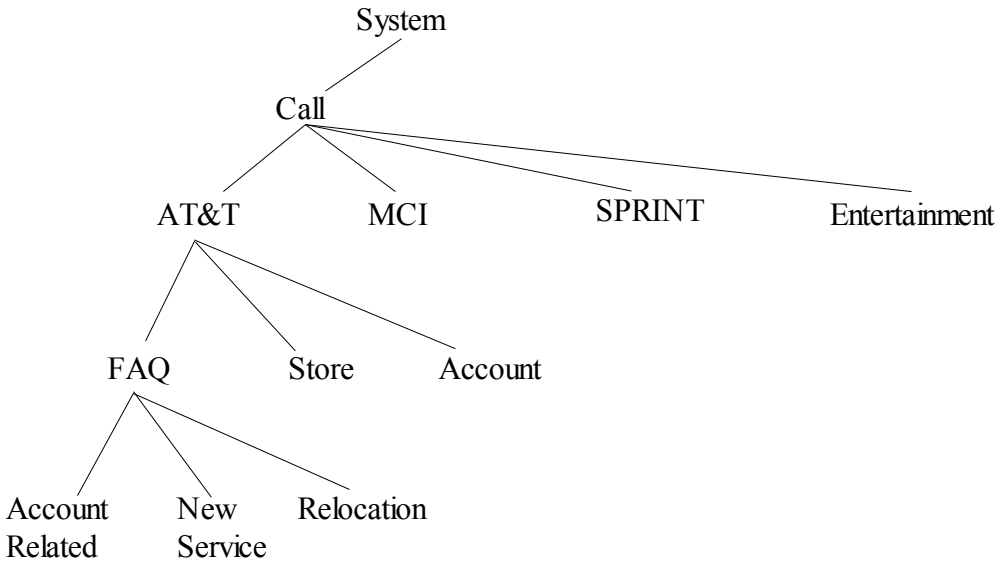
Πρέπει να παρατηρηθεί ότι η ανάγκη για ένα εργαλείο παραγωγής ροών δεδομένων είναι λιγότερο έντονη σε αυτό το χώρο, μιας και τα δεδομένα από ένα δίκτυο συνήθως αφθονούν και μπορούν εύκολα να ληφθούν από οποιονδήποτε έχει πρόσβαση σε αυτό. Αρκεί να να παρακολουθήσει κανείς το δίκτυο σε χρονικό διάστημα ικανού μεγέθους για τα συμπεράσματα που θέλει να συνάγει. Φυσικά, η πρόσβαση στα δεδομένα ενός μεγάλου δικτύου, όπως αυτό ενός παροχέα διαδικτύου (ISP), δεν είναι δυνατή στον καθένα. Από την άλλη πλευρά η σημασιολογία που τα δεδομένα ενός πραγματικού δικτύου φέρουν δεν είναι εύκολο να εξομοιωθεί υπολογιστικά.

5.2.3 ΕΜΦΩΛΕΥΜΈΝΕΣ ΡΟΈΣ (NESTED STREAMS)

Πρόκειται για μία ξεχωριστή κατηγορία ροών δεδομένων που μπορεί να διακριθεί από τις υπόλοιπες εξαιτίας συγκεκριμένων ιδιοτήτων που παρουσιάζουν ξεχωριστό ενδιαφέρον. Υπάρχουν ροές δεδομένων οι οποίες εμπεριέχουν στη σημασιολογία της πληροφορίας που φέρουν την έννοια της συνόδου (session) και των υπό-συνόδων (sub-sessions) που αυτές μπορεί να εμφανίζουν. Προτού αναφερθούμε σε αυτήν αξίζει να παρατηρήσουμε ότι η κατηγορία αυτή υφίσταται ως ιεράρχηση ακριβώς πάνω στη σημασιολογία των δεδομένων και όχι σε ιδιαιτερότητες των εφαρμογών που τις παράγουν ή τη φυσιολογία των τους. Για να γίνει αυτό περισσότερο ξεκάθαρο υπενθυμίζουμε πως η έννοια της συνόδου είναι δυνατόν να εμφανιστεί σε ποικίλες εφαρμογές, όπως, για παράδειγμα, στα προαναφερθέντα δίκτυα δεδομένων κατά την παρακολούθηση των ιχνών (traces) που καταγράφονται σε HTTP συνδέσεις με κάποιο εξυπηρετητή διαδικτύου (web server) ή με κάποιο εξυπηρετητή βάσεων δεδομένων (database server). Στο ίδιο παράδειγμα – των δικτύων δεδομένων – μπορεί κανείς να σημειώσει ακόμη ότι και η ίδια η φυσιολογία των δεδομένων – IP πακέτα με τις επικεφαλίδες τους – αποτελεί από μόνη της ιδιαιτερότητα που χρήζει ξεχωριστής αντιμετώπισης.

Η κατηγορία των ροών αυτών έχει εντοπιστεί και αναλυθεί στο [64] (“Querying Nested Streams” - σε αυτή την εργασία βασίζεται και η ενότητα αυτή), όπου δίνεται μεγάλη έμφαση στην πλούσια σημασιολογία τους. Οι ροές αυτές οργανώνονται σε μια ιεραρχική δομή όπου προσδιορίζονται σύνοδοι, οι οποίες μπορούν να περιέχουν υπό-συνόδους, και η παρακολούθηση που θα ήθελε να εκτελεί ένα σύστημα διαχείρισης επεκτείνεται σε όλο το εύρος της ιεραρχίας. Ένα πολύ χαρακτηριστικό παράδειγμα εμφωλευμένων ροών με συνόδους είναι τα αρχεία (logs) ενός μεγάλου τηλεφωνικού κέντρου. Το δέντρο των υπηρεσιών που ένα τέτοιο κέντρο μπορεί να παρέχει φαίνεται στο Σχήμα 5.1. Το δέντρο αυτό αναδεικνύει το γεγονός ότι μια σύνδεση με το κέντρο μπορεί να ιεραρχηθεί σημασιολογικά αλλά και λειτουργικά σε πιθανές φάσεις/στάδια – αυτά είναι οι κόμβοι του δέντρου – κάθε μία από τις οποίες μπορεί να ιεραρχηθεί περαιτέρω σε κάποια εκ των

προτέρων προσδιορισμένη πιθανή αλληλουχία βημάτων. Με αυτό τον τρόπο αναπαρίσταται η θεματική κατηγοριοποίηση που ο παροχέας του κέντρου έχει δημιουργήσει και ένας χρήστης του μπορεί να εκμεταλλευτεί. Για παράδειγμα, όταν η σύνδεση βρίσκεται στη φάση "Call" τότε οι μόνες δυνατές επιλογές, εκτός από τον τερματισμό της συνόδου, είναι οι "AT&T", η "MCI", η "SPRINT" και η "Entertainment". Ένα πιθανό σενάριο λειτουργίας είναι το εξής: Ένας χρήστης καλεί το κέντρο οπότε εκκινείται η "Call" του "System". Επιλέγει να ζητήσει υπηρεσίες της "AT&T", οπότε ανοίγει η ομώνυμη σύνοδος και από εκεί επιλέγει να ακούσει πληροφορίες ("FAQ") για το λογαριασμό του ("Account Related"). Ένα ενδεικτικό τμήμα του αρχείου (log file) που καταγράφεται κατά τη λειτουργία του τηλεφωνικού κέντρου και αντικατοπτρίζει με δεδομένα την παραπάνω περιγραφή φαίνεται στο Σχήμα 5.2. Στο αρχείο αυτό καταγράφονται το κανάλι που έχει δεσμευτεί για το χρήστη, η ημερομηνία και η ώρα που συνέβη κάποιο γεγονός, ο τύπος του γεγονότος αυτού και κάποιες περιγραφές που έχουν νόημα σε συγκεκριμένες μόνο περιπτώσεις.



Σχήμα 5.1 – Δέντρο Υπηρεσιών ενός τηλεφωνικού κέντρου

Είναι φανερό πλέον πως οι ροές που καταγράφονται στο αρχείο του τηλεφωνικού κέντρου κρύβουν μέσα τους ιεραρχική δομή, η οποία αντικατοπτρίζεται στις έννοιες των συνόδων και υπό-συνόδων (sessions και sub-sessions). Για λόγους είτε στατιστικούς είτε διαχειριστικούς αλλά είτε και κοστολόγησης των κλήσεων το τηλεφωνικό κέντρο θα ήθελε να μπορεί να θέτει ερωτήματα σαν τα παρακάτω πάνω στη ροή των γεγονότων που εμφανίζονται:

- Ποια είναι η τρέχουσα μέση τιμή της διάρκειας των κλήσεων που εξυπηρετούνται από το κέντρο;

- Ποια είναι η τρέχουσα μέση τιμή της διάρκειας των κλήσεων που έχουν ζητήσει

υπηρεσίες της “AT&T”, που βρίσκονται, δηλαδή, σε σύνοδο στο υπό-δέντρο της “AT&T”;

- Ποια είναι η τρέχουσα μέση τιμή της διάρκειας των κλήσεων που έχουν ζητήσει πληροφορίες (“FAQ”) για το σύνολο του συστήματος (“System”).

Όλα τα παραπάνω ερωτήματα είναι συνεχή (continuous queries) και απαιτούν κάποια γνώση της εμφωλευμένης φύσης των συνόδων που το σύστημα διαχειρίζεται. Επομένως, είναι απαραίτητη κάποιου είδους ξεχωριστή ομαδοποίηση μέσα στη ροή μιας και κάθε σύνοδος ή υπό-σύνοδος δεν εξαρτάται αποκλειστικά από την τιμή κάποιας συγκεκριμένης παραμέτρου αλλά από ένα σύνολο σημασιολογικών στοιχείων. Στο [64] δίνεται επίσης έμφαση στο ότι μια ροή μπορεί να αντιμετωπιστεί και ως μια αλληλουχία γεγονότων και ως εκ τούτου είναι μια οντότητα με κατάσταση (state) και συμπεριφορά (behaviour) που μπορεί να μοντελοποιηθεί με μια αντικειμενοστραφή προσέγγιση.

Channel	Date	Time	Step	Event	Expression	Description
21	2001-12-06	12:25:47	0	107	2126199903	CID Received
21	2001-12-06	12:25:47	0	102	1075	3 %
21	2001-12-06	12:25:47	5	500	2126199903	Caller_ID
21	2001-12-06	12:25:55	7	501	QBRICK	ServerName
21	2001-12-06	12:26:03	41	500	ATT	RecognizedCateg
21	2001-12-06	12:26:03	378	500	Starting	ATTCateg
21	2001-12-06	12:26:11	342	501	FAQ	ATTCateg
21	2001-12-06	12:26:15	380	501	FAQ_Categ=1	ATTCateg
21	2001-12-06	12:26:28	352	501	FAQ_Question=2	ATTCateg
21	2001-12-06	12:27:07	352	501	FAQ_Question=#	ATTCateg
21	2001-12-06	12:27:12	380	501	FAQ_Categ=2	ATTCateg
21	2001-12-06	12:27:16	354	501	FAQ_Question=3	ATTCateg
21	2001-12-06	12:27:41	342	501	FAQ	ATTCateg
21	2001-12-06	12:27:43	380	501	FAQ_Categ=3	ATTCateg
21	2001-12-06	12:27:46	356	501	FAQ_Question=2	ATTCateg
21	2001-12-06	12:28:04	342	501	ACCOUNT	ATTCateg
21	2001-12-06	12:28:36	342	501	Location	ATTCateg
21	2001-12-06	12:28:46	348	501	New York	RecognisedATTState
21	2001-12-06	12:28:53	381	500	ATTCenter0076.wav	CUSTOMER CENTER IS
21	2001-12-06	12:29:14	348	501	New Jersey	RecognisedATTState
21	2001-12-06	12:29:38	348	501	#	RecognisedATTState
21	2001-12-06	12:29:42	342	501	#	ATTCateg
21	2001-12-06	12:29:42	0	104	2	Disconnect break

Σχήμα 5.2 – Τμήμα από το αρχείο (log) ενός τηλεφωνικού κέντρου

Η παραγωγή δοκιμαστικών δεδομένων που να εμπεριέχουν πράγματι την έννοια της εμφώλευσης (nesting) αποτελεί μια επιπλέον πρόκληση για ένα εργαλείο σαν αυτό που επιχειρείται να αναπτυχθεί στα πλαίσια της παρούσας εργασίας.

5.2.4 ΡΟΈΣ ΑΠΌ ΑΡΧΕΊΑ ΙΣΤΟΤΟΠΩΝ (WEB LOGs)

Μια ακόμη κατηγορία ροών δεδομένων που αξίζει να αναφερθεί ξεχωριστά είναι αυτή που

αντιστοιχεί σε δεδομένα που καταγράφονται σε αρχεία όπως τα αρχεία καταγραφής ενός ιστοτόπου[20] (web logs) [4][14]. Το πλήθος και η περιπλοκότητα των ιστοτόπων που βρίσκονται εν ενεργεία στις μέρες μας είναι ολοένα και αυξανόμενα και το ίδιο ισχύει και για τις ποσότητες πληροφορίας που περιέχουν. Η ανάλυση των αρχείων τους είτε σε νεκρό χρόνο (offline) είτε σε πραγματικό χρόνο χρησιμεύει έτσι ώστε οι εφαρμογές αυτές να μπορούν να παρακολουθούν την ποιότητα της εξυπηρέτησης των χρηστών τους, την επισκεψιμότητα των περιεχομένων, να μπορούν να διαμοιράσουν το φόρτο τους (load balancing), να μπορούν να προσφέρουν προσωποποιημένες υπηρεσίες (personalization).

Τα δεδομένα που παράγονται από αυτές τις εφαρμογές είναι αλληλουχίες γεγονότων όπως αυτές καταγράφηκαν από τον εξυπηρετητή (server) ιστοτόπου και αφορούν στις απόπειρες πρόσβασης που κάθε χρήστης έκανε στα περιεχόμενα του ιστοτόπου, καθώς και στην έκβαση αυτών (επιτυχία, αποτυχία, κλπ). Τα δεδομένα αυτά είναι επίσης πολύ πλούσια σε σημασιολογικό περιεχόμενο μιας και η διαδοχή τους δεν είναι καθόλου τυχαία, εφόσον εξαρτάται από την πλοήγηση του χρήστη στην εφαρμογή και στις περιοχές αυτής που επιλέγει να πλοηγηθεί ή/και του επιτρέπεται να πλοηγηθεί. Στην απλούστερη μορφή τους είναι απλές πλειάδες που αποτυπώνουν όλη την πληροφορία για τη χρήση του ιστοτόπου όσον αφορά τους χρήστες που είχαν πρόσβαση σε αυτόν, τι περιεχόμενο ζήτησαν, τι όγκος πληροφορίας διακινήθηκε, πότε έγινε η κάθε πράξη πρόσβασης κοκ. Η προσομοίωση μιας τέτοιας ροής από ένα εργαλείο παραγωγής δοκιμαστικών ροών δεδομένων εμπεριέχει την πρόκληση της αναπαραγωγής της όλης σημασιολογίας που συζητούμε. Όπως και να 'χει είναι ιδιαίτερα σημαντικό για ένα ΣΔΡΔ που θέλει να δοκιμαστεί σε παρόμοιες ροές δεδομένων να έχει στη διάθεσή του όσο το δυνατόν πιο ρεαλιστικά δεδομένα. Γι' αυτό το λόγο ίσως να είναι προτιμότερο να εκμεταλλευτεί υπάρχοντα αρχεία που μπορούν να συλλεχθούν από πραγματικές εφαρμογές και να τροφοδοτηθεί με αυτά, προσομοιώνοντας την πραγματική χρονική διαδοχή.

5.2.5 ΡΟΈΣ ΑΠΌ ΔΕΔΟΜΈΝΑ ΟΙΚΟΝΟΜΙΚΉΣ ΠΡΟΈΛΕΥΣΗΣ

Η τελευταία ξεχωριστή κατηγορία ροών δεδομένων που συζητούμε είναι ροές που προέρχονται από εφαρμογές οικονομικής φύσεως [4]. Σαν τέτοιες μπορούμε να αναφέρουμε εφαρμογές παρακολούθησης των συναλλαγών που λαμβάνουν χώρα σε δημοπρασίες, παρακολούθησης των μετοχών του χρηματιστηρίου ή και άλλων πληροφοριών οικονομικού ενδιαφέροντος. Σκοπός

20 Με την έννοια ιστότοπος εδώ δεν αναφερόμαστε σε απλές ιστοσελίδες (web sites) αλλά στο σύνολο των εφαρμογών που είναι προσβάσιμες μέσω διαδικτύου και πλοηγητών (web browser). Το εύρος παρόμοιων εφαρμογών είναι φανερό πως εκτείνεται από πολύ απλές – απλοί ιστότοποι – έως και ιδιαίτερα περίπλοκες εφαρμογές – όπως φόρα, εφαρμογές βάσεων δεδομένων, εφαρμογές διαχείρισης περιεχομένου κοκ.

τέτοιων εφαρμογών είναι η συλλογή και αποθήκευση των οικονομικών δεδομένων, η εξαγωγή στατιστικών από αυτά, η ανίχνευση συσχετίσεων στις παρακολουθούμενες τιμές, η ανίχνευση τάσεων (trends) στις μετρήσεις και η απόπειρα πρόβλεψης μελλοντικών τιμών. Παρόμοια δεδομένα είναι ιδιαίτερα δύσκολο να παραχθούν υπολογιστικά από μια γενικής χρήσεως εφαρμογή με ρεαλιστικά αποτελέσματα, μιας και εξαρτώνται από πολλούς απρόβλεπτους παράγοντες της αγοράς. Μπορούν, όμως, να παραχθούν για τους υπόλοιπους λόγους που ένα σύστημα έχει ανάγκη για να δοκιμαστεί.

5.3 ΥΠΆΡΧΟΝΤΑ ΕΡΓΑΛΕΊΑ ΠΑΡΑΓΩΓΉΣ ΡΟΏΝ ΔΕΔΟΜΈΝΩΝ

Είναι γεγονός ότι στον ακαδημαϊκό χώρο μέχρι σήμερα δεν έχουν υπάρξει παρά ελάχιστα εργαλεία που να συνεισφέρουν σε ερευνητικό επίπεδο στις δοκιμές των επίδοξων συστημάτων διαχείρισης ροών δεδομένων με κάποιο τρόπο. Πολύ λίγες φορές, επιπλέον, οι δοκιμές των αλγορίθμων και των ΣΔΡΔ γίνονται με δεδομένα ρεαλιστικά και που να εμφανίζονται σε πραγματικό χρόνο. Συνήθως οι δοκιμές διεκπεραιώνονται με δεδομένα που λαμβάνονται από κάποιο αρχείο και τροφοδοτούν το σύστημα με πλειάδες σε διαφορετικούς ρυθμούς. Στην ενότητα αυτή θα παρουσιαστεί το "Linear Road Project" και το "NEXMark", ως τα δύο μοναδικά καταγεγραμμένα εργαλεία σε αυτό το χώρο.

5.3.1 ΤΟ "LINEAR ROAD BENCHMARK"

Το "Linear Road" [68][69] είναι κάτι παραπέρα από ένα εργαλείο παραγωγής δοκιμαστικών ροών δεδομένων. Πρόκειται για ένα εργαλείο που θέτει σε δοκιμασία (benchmarking) το σύνολο ενός Συστήματος Διαχείρισης Ροών Δεδομένων – και ως τέτοιο έχει γίνει αποδεκτό στον ερευνητικό χώρο σήμερα και θα αναφερθεί και εδώ. Φιλοδοξεί δε να αποτελεί μέτρο αναφοράς και σύγκρισης των ΣΔΡΔ μέσω των αποτελεσμάτων που παράγει.

Σκοπός του "Linear Road" είναι να προσομοιώσει τους αυτοκινητόδρομους μιας μεγάλης αστικής πόλης που λειτουργούν με ένα εξειδικευμένο σύστημα χρέωσης διοδίων. Το σύστημα χρέωσης έχει ονομαστεί "μεταβαλλόμενη χρέωση" (variable tolling), βασίζει τη χρέωση σε παράγοντες όπως η κυκλοφοριακή συμφόρηση και η εγγύτητα πιθανών ατυχημάτων και είναι ένα από τα τελευταία που έχουν προταθεί σε εφαρμοσμένο επίπεδο στις μέρες μας. Ένα τέτοιο σύστημα προσομοιώνεται, λοιπόν, σε μια φανταστική μητροπολιτική περιοχή και συμπεριλαμβάνει, εκτός από ειδικές χρεώσεις διοδίων, ανίχνευση και πρόβλεψη ατυχημάτων, ειδοποιήσεις για ατυχήματα, μετρήσεις συμφόρησης της κυκλοφορίας και ιστορικές[21] επερωτήσεις. Μερικές από τις θεμελιώδεις

21 Ιστορικές είναι οι επερωτήσεις που τίθενται όχι μόνο πάνω σε ροές αλλά σε συνδυασμούς ροών και δεδομένων καταχωρημένων σε αρχεία στο δίσκο.

παρατηρήσεις της ομάδας ανάπτυξής του είναι οι εξής:

- Το γεγονός ότι δεν υπάρχει κάποιος ενιαίος και συνολικός τρόπος για τη μέτρηση των επιδόσεων των ΣΔΡΔ που εμφανίζονται στον ακαδημαϊκό χώρο για να είναι δυνατόν να κριθούν ανεξάρτητα αλλά και να συγκριθούν μεταξύ τους. Όπως, προαναφέραμε, το "Linear Road" και το "NEXMark" είναι τα δύο μοναδικά εργαλεία αυτού του είδους που έχουν εντοπιστεί.

- Από τους βασικούς στόχους του θα πρέπει να είναι η μέτρηση και καταγραφή της απόδοσης ενός συστήματος στην επεξεργασία συνεχών – αλλά και ιστορικών – επερωτήσεων σε πραγματικό χρόνο (real time) με σαφώς προσδιορισμένες μετρικές.

- Για να έχουν νόημα τα αποτελέσματα που ένα τέτοιο εργαλείο δοκιμασίας (benchmarking) παράγει θα πρέπει τα δεδομένα που τροφοδοτεί στα ΣΔΡΔ να έχουν κατά το δυνατόν σημασιολογική εγκυρότητα. Μια οποιαδήποτε ροή αποτελεί τη διακριτή αναπαράσταση μιας φυσικής συνεχούς διαδικασίας και θα όφειλε να είναι συνεπής με αυτή. Αυτό είναι αληθές σε μεγάλο βαθμό αν και υπάρχουν περιπτώσεις που διαφοροποιούνται. Για παράδειγμα, ενώ σε ένα σύστημα δημοπρασιών δεν είναι δυνατόν να υπάρξει αγορά ενός αντικειμένου αν αυτό δεν έχει προηγουμένως εισαχθεί στη δημοπρασία, σε ένα δίκτυο αισθητήρων οι τιμές που ανιχνεύονται σε διαφορετικές άκρες μιας έκτασης δεν είναι άμεσα εξαρτώμενες

- Ακριβώς εξαιτίας του γεγονότος ότι τα επερωτήματα είναι συνεχή, οι μετρήσεις της απόδοσης δε μπορούν να βασίζονται στο χρόνο ολοκλήρωσής τους (completion time) αλλά στο χρόνο απόκρισης του συστήματος (response time). Παράλληλα, θα πρέπει να λαμβάνεται σοβαρά υπόψη και ο φόρτος δεδομένων υπό τον οποίο τα εκάστοτε αποτελέσματα επιτυγχάνονται, μιας και κάποιος χρόνος απόκρισης πιθανόν να είναι αποδεκτός σε περίοδο μεγάλου φόρτου αλλά μη αποδεκτός σε περίοδο μικρού φόρτου.

- Ακόμη, η όλη δοκιμασία (benchmark) θα πρέπει να είναι επαληθεύσιμη παρόλο που τα σωστά αποτελέσματα δεν είναι μοναδικά μιας και θα εξαρτώνται από την τρέχουσα κάθε φορά κατάσταση των παραγόμενων δοκιμαστικών δεδομένων, τα οποία θα ενέχουν με τον ένα ή τον άλλο τρόπο κάποιο βαθμό τυχαιότητας.

- Τέλος, οι επερωτήσεις που απαιτεί η δοκιμασία θα πρέπει να προσδιορίζονται σε μια

γλώσσα γενικόλογη μιας και δεν υπάρχει κάποια ευρέως αποδεκτή γλώσσα διατύπωσης συνεχών επερωτήσεων για να προτιμηθεί.

Η έμπνευση για την προσομοίωση που κάνει το “Linear Road” προέρχεται από την ολοένα και ευρύτερα εφαρμοζόμενη τακτική της μεταβλητής χρέωσης διοδίων που υιοθετείται σε μεγάλα αστικά κέντρα. Η τακτική αυτή καθορίζει ότι η χρέωση που θα γίνεται θα είναι συνάρτηση του επιπέδου της κυκλοφοριακής συμφόρησης και της εγγύτητας σε τόπους τροχαίων ατυχημάτων με απώτερο σκοπό την αποθάρρυνση των οδηγών από το να χρησιμοποιήσουν οδικές αρτηρίες που παρουσιάζουν προβλήματα. Με αυτό τον τρόπο, ένας οδηγός που επιθυμεί να διασχίσει ένα δρόμο με μεγάλη κίνηση θα χρεωθεί περισσότερο από ότι αν χρησιμοποιούσε ένα λιγότερο πολυσύχναστο δρόμο.

Τα δεδομένα που το “Linear Road” απαιτεί για να τρέξει είναι μια σειρά από αναφορές θέσης αυτοκινήτων, που παράγονται με το μορφή ροής με ρυθμό μία πλειάδα κάθε 30 sec για κάθε αυτοκίνητο, και ορισμένα καταχωρημένα σε δίσκο στατικά δεδομένα. Το σύστημα που θα υποδεχθεί τις ροές του “Linear Road” θα πρέπει:

- να διατηρεί στατιστικά σχετικά με τον αριθμό των οχημάτων που βρίσκονται στους αυτοκινητοδρόμους ανά κάθε λεπτό,

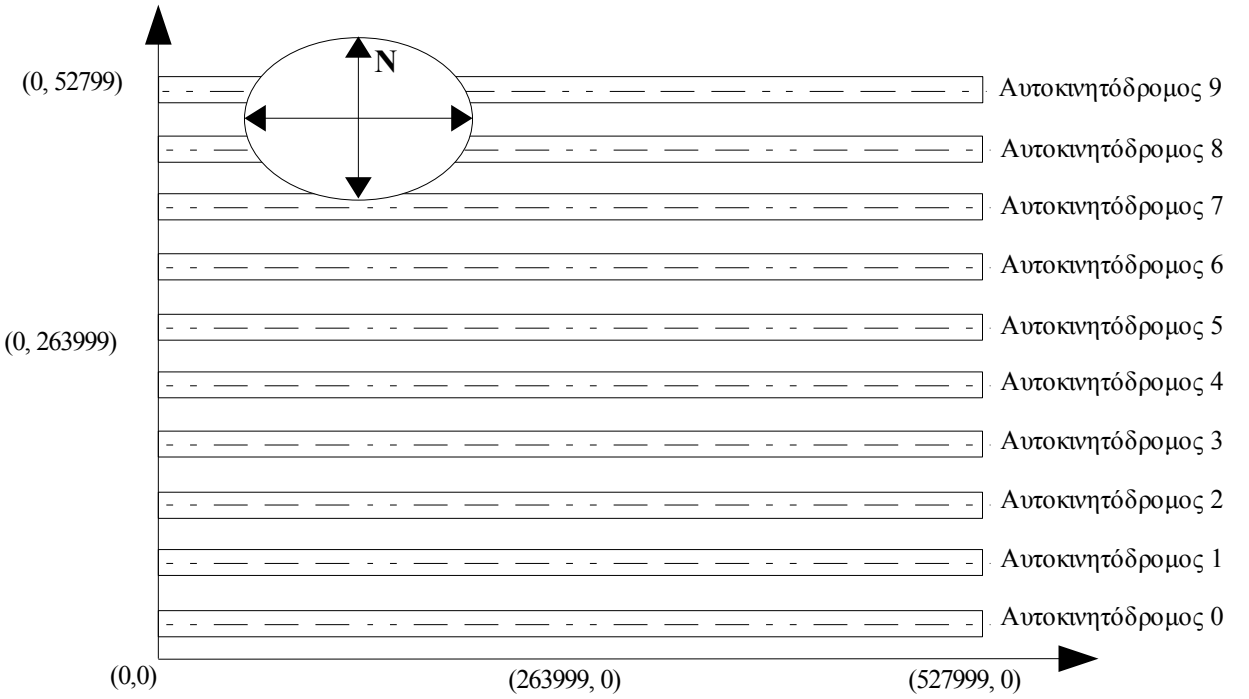
- να ανιχνεύει τροχαία ατυχήματα και να ειδοποιεί τους οδηγούς για αυτά,

- να υπολογίζει τις χρεώσεις των διοδίων (toll charges) βάσει των παραμέτρων που αναφέρθηκαν και

- να χρεώνει ανάλογα τους λογαριασμούς (accounts) των οδηγών, οι οποίοι θεωρείται ότι χρεώνονται με αυτόματο τρόπο καθώς διασχίζουν τη φανταστική πόλη.

Τα παραπάνω είναι επερωτήσεις και λειτουργίες πάνω σε ροές δεδομένων. Για να συμπληρωθεί η δοκιμασία υπάρχουν και επερωτήσεις πάνω σε συνδυασμούς ροών και στατικών δεδομένων (historical queries). Τα στατικά δεδομένα αφορούν πληροφορία σχετική με τον τρόπο χρέωσης, το ιστορικό των χρεώσεων που έχουν καταγραφεί για δεδομένο χρονικό διάστημα στο κοντινό παρελθόν, την τοπολογία των αυτοκινητοδρόμων, τα στοιχεία των οχημάτων που κυκλοφορούν κλπ.

Αυτοκινητόδρομος 8 Για να μετρηθεί η ικανότητα ενός ΣΔΡΔ στα δεδομένα που το "Linear Road" του διοχετεύει έχει προσδιοριστεί μια συγκεκριμένη μετρική. Αυτή έχει ονομαστεί **L** και αντιστοιχεί στην ποσότητα των δεδομένων που ένα ΣΔΡΔ μπορεί να αντέξει χωρίς να πέσει κάτω από τα ανεκτά όρια χρόνου απόκρισης και ορθότητας των αποτελεσμάτων των επερωτήσεων. Ας θυμηθούμε ότι μία από τις πιο συχνές τακτικές που εφαρμόζονται όταν ο φόρτος ενός ΣΔΡΔ είναι ιδιαίτερα υψηλός είναι η απόρριψη πλειάδων και ο προσεγγιστικός υπολογισμός των επερωτήσεων. Επομένως, η μετρική αυτή συμψηφίζει τόσο την ακρίβεια των αποτελεσμάτων των επερωτήσεων όσο και την αντοχή του συστήματος σε υψηλούς ρυθμούς δεδομένων με το χρόνο απόκρισής του.



Σχήμα 5.3 – Η γεωμετρία της πόλης του "Linear Road"

Η φανταστική "ευθύγραμμη πόλη" του "Linear Road" απεικονίζεται στο Σχήμα 5.3. Έχει έκταση 100x100 μίλια και περιέχει 10 παράλληλους αυτοκινητόδρομους που απέχουν 10 μίλια ο ένας από τον άλλο, ενώ δεν έχε καθόλου κάθετους αυτοκινητόδρομους. Κάθε αυτοκινητόδρομος έχει 2 κατευθύνσεις και 4 λωρίδες σε κάθε κατεύθυνση, από τις οποίες οι 3 είναι για την κυκλοφορία και η $4^{η}$ για την είσοδο και έξοδο των οχημάτων. Επίσης, κάθε αυτοκινητόδρομος έχει 100 εισόδους και 100 εξόδους σε κάθε κατεύθυνση κάτι που τον χωρίζει νοητά σε 100 τμήματα του ενός μιλίου. Κάθε όχημα διαθέτει έναν αισθητήρα που εκπέμπει μια αναφορά για τη θέση του αυτοκινήτου κάθε 30 sec. Κάθε τόσο ένα όχημα, αφότου αναφέρει τη θέση του, ζητά να μάθει πληροφορίες, όπως η κατάσταση του λογαριασμού του, το σύνολο της χρέωσής του εκείνη την ημέρα, πρόβλεψη χρόνου ταξιδιού κλπ. Τα ατυχήματα συμβαίνουν με τυχαίο – και όχι συχνό – τρόπο προκαλώντας προβλήματα στην ταχύτητα της κυκλοφορίας και λίγο χρόνο αργότερα ξεπερνιούνται.

Τα δεδομένα που το “Linear Road” χρησιμοποιεί προέρχονται από τον προσομοιωτή οδικής κυκλοφορίας MITSIM [70] από όπου παράγονται, ακολούθως αποθηκεύονται σε αρχείο και εν συνεχεία διοχετεύονται έτσι ώστε να προσομοιώνεται ο πραγματικός χρόνος. Έτσι, υπάρχουν τα εξής δεδομένα που διακινούνται: οι αναφορές θέσης των οχημάτων, οι επερωτήσεις των οχημάτων για το λογαριασμό τους, την ημερήσια χρέωσή τους και την πρόβλεψη χρόνου ταξιδιού και τα αποτελέσματά τους, αλλά και το αρχείο των χρεώσεων του παρελθόντος.

Η “δοκιμασία” στην οποία βάζει ένα ΣΔΡΔ το “Linear Road” αποτελείται από ένα σύνολο εκ των προτέρων προσδιορισμένων επερωτήσεων και λειτουργιών των οποίων η ταχύτητα απόκρισης και η ακρίβεια του αποτελέσματος προσμετρώνται και κρίνονται. Μερικές από τις πιο σημαντικές είναι οι εξής:

- Λειτουργίες σχετικές με τη χρέωση των διοδίων.
  Πρέπει να γίνεται υπολογισμός της χρέωσης ενός οχήματος κάθε φορά που αυτό περνά από ένα τμήμα (μήκους ενός μιλίου) στο επόμενο και σύμφωνα με τη λογική της “μεταβαλλόμενης χρέωσης”. Ο υπολογισμός της χρέωσης γίνεται πρώτα με την ενημέρωση του οχήματος για την επικείμενη χρέωση κι έπειτα με την ίδια τη χρέωση του λογαριασμού του.

- Λειτουργίες σχετικές με την κυκλοφορία.
  Πρέπει να γίνεται η όσο πιο έγκαιρη πρόγνωση των ατυχημάτων που συμβαίνουν έτσι ώστε κάθε όχημα που εισέρχεται κοντά στην περιοχή εκείνη να ειδοποιείται κατάλληλα για να μπορεί να την αποφύγει.

- Επερωτήσεις προς επεξεργασία.
  Εκτός από τα παραπάνω, τα οποία έτσι κι αλλιώς εκτελούνται διηνεκώς με τη μορφή συνεχών επερωτήσεων, εκτελούνται, επιπλέον, και επερωτήσεις που αφορούν τα ερωτήματα των οχημάτων για την τρέχουσα κατάσταση του λογαριασμού τους, τη συνολική ημερήσια χρέωσή τους και την εκτίμηση του χρόνου ταξιδιού τους μέχρι κάποιον προορισμό.

Όπως προαναφέρθηκε η μετρική που έχει καθοριστεί για να αναδείξει την απόδοση ενός ΣΔΡΔ που δοκιμάζεται με το “Linear Road” ονομάζεται **L** και είναι ο μέγιστος παράγοντας φόρτου δεδομένων κάτω από τον οποίον το σύστημα μπορεί να ανταποκρίνεται στις δοσμένες επερωτήσεις μέσα σε

ανεκτά πλαίσια χρόνου απόκρισης και ακρίβειας των αποτελεσμάτων (ο προσομοιωτής οδικής κυκλοφορίας δημιουργεί L στο πλήθος αρχεία κυκλοφορίας διάρκειας τριών ωρών και στο πλήθος αυτό αντανακλά η μετρική **L**). Οι δοκιμές εκτελούνται με αυξανόμενο παράγοντα φόρτου μέχρις ότου βρεθεί αυτός όπου οι προδιαγραφές δε μπορούν πλέον να τηρηθούν.

Το "Linear Road" χρησιμοποιείται ως εργαλείο δοκιμών στα ερευνητικά ΣΔΡΔ Aurora [55] και STREAM [60], τα οποία αναφέραμε στο προηγούμενο κεφάλαιο, και είναι, επομένως, αποδεκτό από ένα σημαντικό φάσμα της ακαδημαϊκής κοινότητας. Οι προδιαγραφές που θέτει είναι αρκετές ώστε να δοκιμάσουν ένα επίδοξο ΣΔΡΔ σημαντικά σε μεγάλο εύρος λειτουργιών και να προκαλέσουν πραγματικά την αντοχή του σε μεγάλη ποσότητα δεδομένων. Παρολαυτά, είναι κατανοητό ότι η δοκιμασία που θέτει είναι αρκετά προσδιορισμένη και δε δίνει πολλούς βαθμούς ελευθερίας για πειραματισμούς. Οι ροές που παράγονται είναι συγκεκριμένες τόσο όσον αφορά το πλήθος τους όσο και ως προς τους τύπους δεδομένων που συμμετέχουν στις πλειάδες. Επίσης, απολύτως καθορισμένη είναι και η σημασιολογία των παραγόμενων δεδομένων. Το τελευταίο είναι επιθυμητό με σκεπτικό ότι τα δεδομένα δε θα πρέπει να είναι εντελώς τυχαία αλλά να αντικατοπτρίζουν τη λογική της εφαρμογής που προσομοιώνεται. Από την άλλη πλευρά, όμως, κάτι τέτοιο περιορίζει τις δυνατές δοκιμές που μπορούν να εκτελεστούν σε αυτό το σενάριο μονάχα που έχει εκ των προτέρων προδιαγραφεί.

Σκεπτόμενοι τα παραπάνω μπορεί κανείς να παρατηρήσει ότι το "Linear Road" είναι πράγματι ένα αξιόλογο εργαλείο δοκιμών (benchmarking) για ΣΔΡΔ, το οποίο έχει καταφέρει να ενσωματώσει μια δεδομένη σημασιολογία – αυτή της διαχείρισης των μεταβαλλόμενων χρεώσεων της κυκλοφοριακής κίνησης. Παρόλο, βέβαια, που έχει εκ προθέσεως δοθεί αυτή η έμφαση στη σημασιολογία, οι δοκιμές που υποστηρίζονται δίνουν περισσότερο βάρος στο να προκαλέσουν τα όρια αντοχής ενός συστήματος σε μεγάλους φόρτους εισερχόμενων δεδομένων (stress test). Και αυτό επιτυγχάνεται χωρίς τελικά να συνεισφέρει ουσιαστικά σε αυτή τη διαδικασία το γεγονός ότι τα δεδομένα φέρουν τη συγκεκριμένη σημασιολογία, μιας και τα όρια αντοχής δοκιμάζονται με απλή αύξηση των ρυθμών εισόδου των δεδομένων, ανεξάρτητα με το τι αυτά εσωτερικά σημαίνουν. Η αξία της δοσμένης σημασιολογίας, εκτός από σύνθετη στο να επιτευχθεί, φυσικά, είναι αναμφισβήτητη, όσον αφορά κυρίως την επαλήθευση των αποτελεσμάτων των επερωτήσεων. Βέβαια δεν υπάρχει η δυνατότητα ορισμού εναλλακτικών σεναρίων εκτέλεσης και η δοκιμασία περιορίζεται στις συνθήκες που έχουν προσδιοριστεί με μόνη δυνατότητα την παραμετροποίηση του φόρτου εισόδου.

5.3.2 ΤΟ "NEXMARK BENCHMARK"

Το δεύτερο εργαλείο δοκιμασίας Συστημάτων Διαχείρισης Ροών Δεδομένων για το οποίο θα γίνει εδώ αναφορά είναι το "NEXMark" [72][71] (Niagara Extension to XMark benchmark). Μέσα στους στόχους του είναι να αποτελέσει ένα εργαλείο δοκιμασίας (benchmarking), να παράσχει ένα εργαλείο παραγωγής ροών δεδομένων (stream generator) και να ορίσει μετρικές για την εκτίμηση της απόδοσης συνεχών επερωτήσεων. Βασίζεται πάνω στο "XMark" [73], που είναι εργαλείο δοκιμών για τη μέτρηση της απόδοσης αποθηκών XML δεδομένων (XML repositories[22]) που παράγει XML δεδομένα από την προσομοίωση δημοπρασιών. Το "NEXMark" επεκτείνει το "XMark" έτσι ώστε τα παραγόμενα δεδομένα να εμφανίζονται με την απαραίτητη μορφή ροών και μάλιστα όχι απαραίτητα XML μορφοποίησης.

Σε μία δημοπρασία που εκτελείται on-line δημιουργείται μια πληθώρα από δεδομένα που αφορούν σε πράξεις της δημοπρασίας όπως η εισαγωγή νέων αντικειμένων προς πώληση, η εισαγωγή νέων χρηστών στο σύστημα, οι προσφορές των χρηστών πάνω στα διαθέσιμα αντικείμενα, το κλείσιμο αγορών κλπ. Με αυτό τον τρόπο ορίζονται τριών ειδών κύριες ροές που εισάγουν δεδομένα στο σύστημα της δημοπρασίας και αναφέρονται στην αλληλεπίδραση που έχουν οι χρήστες με αυτό:

- Η ροή που αφορά στους χρήστες (users) των δημοπρασιών και τις εγγραφές τους σε αυτές σαν πωλητές ή αγοραστές.

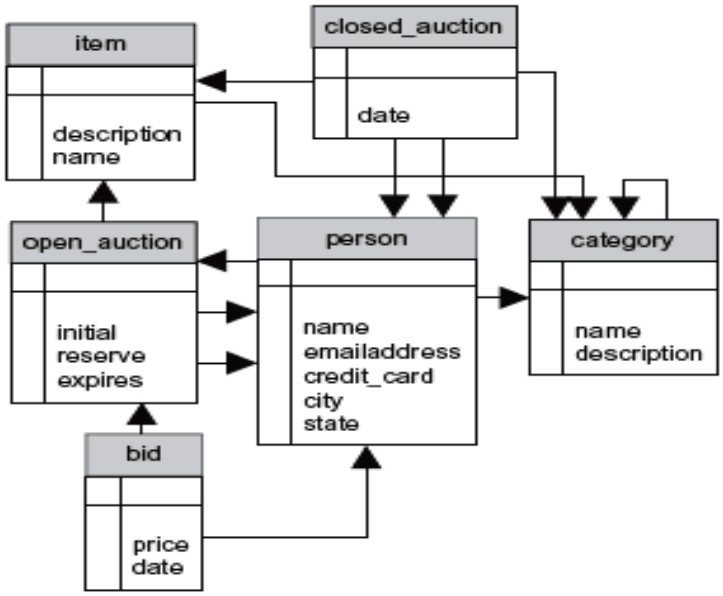
- Η ροή που αφορά στην ίδια τη δημοπράτηση αντικειμένων (items) και περιέχει τα αντικείμενα που εισέρχονται προς δημοπράτηση – μαζί με πληροφορίες όπως η ποσότητα, ο χρόνος έναρξης και λήξης – και τις πράξεις της εισαγωγής αντικειμένου και της ολοκλήρωσης μιας δημοπρασίας.

- Η ροή που περιέχει τις προσφορές (bids) των χρηστών πάνω στα δημοπρατούμενα αντικείμενα. Η ροή αυτή μπορεί να πολλαπλασιαστεί ανάλογα με την ένταση της δοκιμασίας.

Επιπλέον των παραπάνω ροών υπάρχουν και στατικά δεδομένα παίρνουν μέρος στη δοκιμασία, τα οποία βρίσκονται αποθηκευμένα σε συνηθισμένα αρχεία. Αυτά αφορούν υποστηρικτικά δεδομένα των δημοπρασιών, όπως πληροφορίες σχετικές με τις κατηγορίες των δημοπρατούμενων

22 Ένα XML repository είναι ένα σύστημα διαχείρισης δεδομένων που βρίσκονται σε XML μορφή. Η διαχείριση μεγάλων όγκων XML δεδομένων είναι πολύ σημαντική στις μέρες μας και τα ιδιαίτερα ζητήματα που ανακύπτουν από τις ιδιότητες της XML έρχονται να αντιμετωπίσουν τα συστήματα αυτά.

αντικειμένων που δε μεταβάλλονται τακτικά. Στο Σχήμα 5.4 φαίνεται το σχήμα (schema) των δεδομένων που παίρνουν μέρος στο "NEXMark".



Σχήμα 5.4 – Το σχήμα (schema) των ροών του "NEXMark"

Το "NEXMark" προσδιορίζει ένα σύνολο από οκτώ συνεχείς επερωτήσεις που απαιτεί να επεξεργαστεί το υπό δοκιμασία ΣΔΡΔ και τις εκφράζει σε γλώσσα τύπου SQL. Οι πρώτες τέσσερις είναι απλές επερωτήσεις επιλογών (SELECT) που επιστρέφουν πληροφορία για την κατάσταση της δημοπράτησης συγκεκριμένων αντικειμένων, υπολογίζουν κάποια μέση τιμή (πχ. τη μέση τρέχουσα τιμή πώλησης των αντικειμένων κάποιας κατηγορίας) ή εκτελούν κάποια απλή συνένωση (JOIN) (πχ. ποιοι είναι οι νέοι χρήστες που έκαναν προσφορά για κάποια δημοπρασία). Οι υπόλοιπες τέσσερις επερωτήσεις είναι επερωτήσεις πάνω σε πλαίσια (windows) των ροών με ομαδοποιήσεις (GROUP BY) και συνενώσεις. Ενδεικτικά δύο από τις επερωτήσεις που ζητούνται:

- Τα δημοπρατούμενα αντικείμενα που έχουν δεχτεί τις μεγαλύτερες προσφορές τα τελευταία 10 λεπτά. Η επερώτηση διατυπώνεται ως εξής:

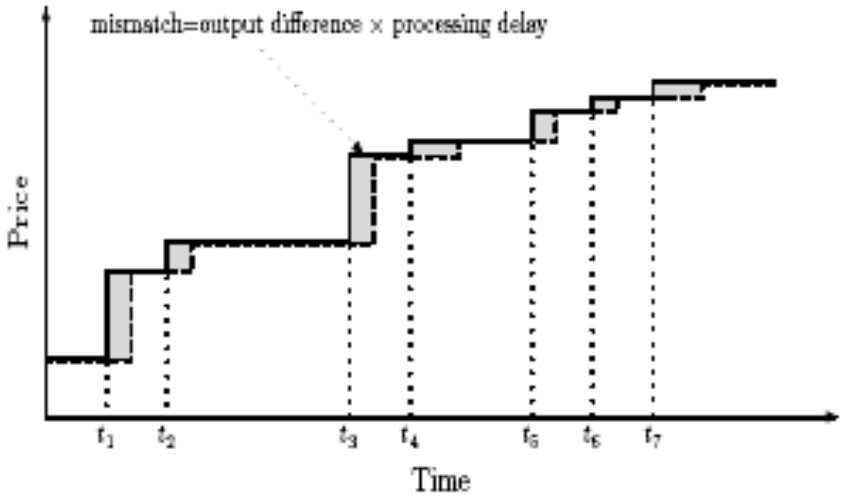
```
SELECT bid.price, bid.itemid
FROM bid where bid.price =
    (SELECT MAX(bid.price)
     FROM bid [RANGE 10 MINUTES PRECEDING]);
```

- Οι νέοι χρήστες που καταχώρησαν κάποιο αντικείμενο προς πώληση μέσα σε 24 ώρες από την εγγραφή τους στο σύστημα. Η επερώτηση αυτή απαιτεί συνένωση πάνω σε δύο πλαίσια – χρήστες και δημοπρασίες – και διατυπώνεται ως εξής:

```
SELECT person.id, person.name
FROM person [RANGE 12 HOURS PRECEDING],
     open auction [RANGE 12 HOURS PRECEDING]
WHERE person.id = open auction.sellerId;
```

Η δοκιμασία στην οποία το "NEXMark" υποβάλλει ένα ΣΔΡΔ κλιμακώνεται σε μέγεθος καθώς το αρχείο με τα στατικά δεδομένα μεγαλώνει, το πλήθος των ροών των προσφορών των χρηστών για αντικείμενα αυξάνεται, ο ρυθμός εισαγωγής νέων προς πώληση αντικειμένων μεγαλώνει και καθώς η διάρκεια της δοκιμασίας επεκτείνεται. Όπως και το "Linear Road" έτσι και το "NEXMark" καταμετρά τόσο την ταχύτητα απόκρισης των επερωτήσεων που θέτει όσο και την ακρίβεια των αποτελεσμάτων τους.

Η μετρική που προτείνεται είναι ο συνδυασμός του ρυθμού δεδομένων που εισήχθησαν στο σύστημα και της ακρίβειας των επερωτήσεων που επιτεύχθηκε με αυτόν (output matching) και είναι ενδιαφέρον να εξετάσει κανείς πώς υπολογίζεται. Όπως φαίνεται στο Σχήμα 5.5 γραφική παράσταση δείχνει την μέτρηση μιας τιμής από μια επερώτηση ανά χρονικές στιγμές. Η μαύρη – μη διακεκομμένη – γραμμή δείχνει το αποτέλεσμα που θα έφερνε ένα ιδεατό σύστημα που θα εκτελούσε την επερώτηση με μηδενική καθυστέρηση και με απόλυτη ακρίβεια. Η διακεκομμένη γραμμή δείχνει το αποτέλεσμα που φέρνει ένα πραγματικό σύστημα. Έτσι ανάλογα με το πόσο καθυστερημένα έρχεται μία απάντηση ή πόσο ανακριβής είναι αυτή τόσο ευρύτερη γίνεται η γκρίζα περιοχή και τόσο χαμηλότερη είναι η απόδοση. Στο συγκεκριμένο παράδειγμα για λόγους γενίκευσης η τιμή που μετριέται έχει αύξουσα τάση αλλά η μετρική μπορεί να εφαρμοστεί παραλλαγμένη και στις υπόλοιπες περιπτώσεις.



Σχήμα 5.5 – Υπολογισμός μετρικής "output matching" του "NEXMark"

Στο σημείο αυτό ολοκληρώνεται και η αναφορά στο σύστημα δοκιμασίας ΣΔΡΔ "NEXMark", όπου μπορούμε να κάνουμε όμοιες παρατηρήσεις με αυτές που έγιναν και για το "Linear Road". Δηλαδή, ότι το σενάριο που το εν λόγω εργαλείο εφαρμόζει αν και είναι πλήρως προσδιορισμένο είναι περιορισμένο σημασιολογικά σε μία συγκεκριμένη περίπτωση όπου μονάχα λειτουργικές παράμετροι μεταβάλλονται. Και εδώ, επομένως, η δοκιμασία (benchmark) στηρίζεται κατά βάθος περισσότερο στο να προκαλέσει το δοκιμαζόμενο ΣΔΡΔ ως προς την αντοχή του σε μεγάλους όγκους δεδομένων χρησιμοποιώντας το γεγονός ότι εφαρμόζει τα σημασιολογικά στοιχεία των on-line δημοπρασιών υποστηρικτικά και όχι ουσιαστικά.

Είναι, κατά συνέπεια, κατανοητό πως υπάρχει αρκετό ακόμη ανεξερεύνητο ερευνητικό πεδίο στο χώρο της μελέτης των ροών δεδομένων, και συγκεκριμένα στο πεδίο των εργαλείων δοκιμασίας και παραγωγής δοκιμαστικών ροών δεδομένων. Ένα περισσότερο γενικού σκοπού εργαλείο, που θα βοηθά στην βαθύτερη κατανόηση των ιδιαιτεροτήτων των ροών από τα επίδοξα συστήματα διαχείρισης ροών αλλά και θα λειτουργεί πραγματικά και αυτό σαν υποστηρικτικό ερευνητικό εργαλείο στην ανάπτυξη και εξέλιξή τους, θα παρουσιαστεί στη συνέχεια της παρούσας εργασίας κυρίως ως προς τη μελέτη και το σχεδιασμό του και στη συνέχεια σε μια πρωτογενή υλοποίησή του.

6. ΑΝΆΛΥΣΗ, ΣΧΕΔΊΑΣΗ ΚΑΙ ΑΝΆΠΤΥΞΗ ΕΡΓΑΛΕΊΟΥ ΠΑΡΑΓΩΓΉΣ ΔΟΚΙΜΑΣΤΙΚΏΝ ΡΟΏΝ ΔΕΔΟΜΈΝΩΝ

Στο προηγούμενο κεφάλαιο είδαμε πως πράγματι στο ερευνητικό πεδίο της διαχείρισης ροών δεδομένων υπάρχει αναγκαιότητα για εργαλεία παραγωγής δοκιμαστικών ροών δεδομένων και εργαλεία δοκιμασίας (benchmarking) Συστημάτων Διαχείρισης Ροών Δεδομένων (ΣΔΡΔ). Μάλιστα εξετάσαμε εν συντομία και δύο από τα ελάχιστα αυτού του είδους εργαλεία. Η μεγάλη προσφορά αυτών των εργαλείων έγκειται κυρίως στο ότι παρέχουν – στο βαθμό που το επιτυγχάνουν, βέβαια, αυτό – ένα κοινό και ενιαίο τρόπο για τη μέτρηση της αποδοτικότητας των επίδοξων ΣΔΡΔ. Αυτό, μπορεί να ισχυριστεί κανείς, είναι οπωσδήποτε επιθυμητό σαν σκοπιμότητα και τα δύο εργαλεία που εξετάσαμε το καταφέρνουν υπό τις συνθήκες που τα ίδια καθορίζουν.

6.1 Η ΑΝΆΓΚΗ ΓΙΑ ΠΕΡΑΙΤΈΡΩ ΈΡΕΥΝΑ ΚΑΙ ΑΝΆΠΤΥΞΗ ΣΤΟ ΧΏΡΟ

Έχουν εντοπιστεί όμως και ελλείψεις ή, τουλάχιστο, δυνατότητες για περαιτέρω ερευνητική δραστηριότητα στο χώρο αυτό. Όπως έχουμε παρατηρήσει, τα εργαλεία δοκιμασίας αυτά προσδιορίζουν το καθένα ένα αρκετά συγκεκριμένο και καθορισμένο σενάριο το οποίο και εφαρμόζουν κατά της εκτέλεσή τους για να παράξουν ροές δεδομένων που θα εισάγουν σε ένα ΣΔΡΔ. Στη μία περίπτωση το σενάριο είναι η διαχείριση της κυκλοφορίας στην "ευθύγραμμη πόλη" του "Linear Road" και στην άλλη η διεξαγωγή on-line δημοπρασιών. Και στις δύο περιπτώσεις η σημασιολογία των παραγόμενων ροών είναι υπαρκτή, δηλαδή, τα δεδομένα υπόκεινται πράγματι σε σημασιολογικούς κανόνες τους οποίους το εργαλείο εγγυάται ότι ακολουθούν. Ακόμη, τα σχήματα (schemas) των ροών είναι εκ των προτέρων καθορισμένα βάσει της σημασιολογίας του κάθε εργαλείου και κάποιες επιλεγμένες επερωτήσεις επιλέγονται για να εκτελεστούν, τα αποτελέσματα των οποίων κρίνονται βάσει συγκεκριμένων μετρικών που οι ομάδες ανάπτυξης των εργαλείων έχουν θέσει.

Η σκοπιμότητα που τα εργαλεία αυτά εξυπηρετούν είναι δεδομένη και η αποδοχή τους στην ακαδημαϊκή κοινότητα είναι εξίσου σημαντική (τα ερευνητικά ΣΔΡΔ “STREAM” και ”Aurora” έχουν επιλέξει το “Linear Road” ως εργαλείο για benchmarking). Υπάρχει, μολαταύτα, ένα λεπτό σημείο που δεν ξεκαθαρίζεται πλήρως. Και τα δύο αυτά εργαλεία σημειώνουν με έμφαση ότι βασικός σκοπός τους είναι η παραγωγή ροών που να περιέχουν σημασιολογική αξία, να μην είναι, δηλαδή, εντελώς τυχαία, κατανοώντας πάντα ότι εφόσον τα δεδομένα παράγονται υπολογιστικά πάντοτε θα υπάρχει κάποιος βαθμός τυχαιότητας, ο οποίος δε θα βλάπτει ουσιαστικά την ποιότητα των παραγόμενων συμπερασμάτων. Και είναι γεγονός ότι την επιθυμητή για αυτά σημασιολογία

την επιτυγχάνουν. Όμως, η σημασιολογία αυτή δε φαίνεται να παίζει ουσιαστικό ρόλο στην ίδια τη δοκιμασία στην οποία υποβάλλουν ένα ΣΔΡΔ κατά την εκτέλεσή τους. Η δοκιμασία καταλήγει στη μέτρηση της αποδοτικότητας των επερωτήσεων – συνδυασμός απόκρισης και ακρίβειας αποτελέσματος – χωρίς να διαφαίνεται πώς έπαιξε σπουδαίο ρόλο στη δοκιμασία καθεαυτή η ύπαρξη της σημασιολογίας.

Η σημασιολογία αυτή είναι αναμφισβήτητα σημαντική όσον αφορά στο να θεωρηθεί ένα τέτοιο εργαλείο ολοκληρωμένο και συμπαγές, όμως υπάρχει, με αυτό τον τρόπο, πάντα και ο περιορισμός του ότι το συγκεκριμένο σενάριο είναι και το μοναδικό που μπορεί να δοκιμαστεί. Δεν υπάρχει, επομένως, στις ομάδες ανάπτυξης ΣΔΡΔ ή αλγορίθμων που εφαρμόζονται εσωτερικά σε ένα ΣΔΡΔ η δυνατότητα προσδιορισμού σεναρίων εκτέλεσης των δοκιμών που να εξειδικεύονται στις εκάστοτε ανάγκες και κατευθύνσεις τους. Οι μόνες παραμετροποιήσεις που είναι δυνατό να γίνουν αφορούν στην κλιμάκωση του φόρτου των δεδομένων που εισάγονται στο υπό δοκιμασία ΣΔΡΔ. Από αυτά τα συμπεράσματα αναδύεται η αναγκαιότητα για την ύπαρξη ενός εργαλείου που να δίνει περισσότερους βαθμούς ελευθερίας στις ερευνητικές ομάδες που επιθυμούν να δοκιμάσουν τα αποτελέσματα της έρευνάς τους στο χώρο της διαχείρισης των ροών δεδομένων

6.2 ΈΛΛΕΙΨΗ ΡΟΏΝ ΔΕΔΟΜΈΝΩΝ ΓΙΑ ΔΟΚΙΜΑΣΤΙΚΟΎΣ ΣΚΟΠΟΎΣ

Επομένως, είναι κατανοητό ότι υπάρχει ακόμη η ανάγκη για την περαιτέρω ανάπτυξη εργαλείων που να συνδράμουν στις δοκιμές των ΣΔΡΔ αλλά και να συνεισφέρουν στην καλύτερη κατανόηση των ροών δεδομένων. Οι σημαντικότεροι παράγοντες που υποστηρίζουν αυτό το σκεπτικό και λειτουργούν ως μία πρώτη καταγραφή προδιαγραφών για το εργαλείο που αναπτύχθηκε στα πλαίσια της παρούσας εργασίας περιγράφονται εκτενέστερα στο υπόλοιπο αυτής της παραγράφου.

6.2.1 ΑΔΥΝΑΜΊΑ ΛΉΨΗΣ ΡΕΑΛΙΣΤΙΚΏΝ ΡΟΏΝ ΔΕΔΟΜΈΝΩΝ

Για να μπορέσει ένα ΣΔΡΔ να δοκιμάσει τις δυνατότητές του στη διαχείριση ροών δεδομένων ή, ακόμη, ένας απλός αλγόριθμος να ελέγξει πρωτογενώς ή και σε έκταση την αποδοτικότητα των αποτελεσμάτων του έχουν την ανάγκη να διαθέτουν ποσότητες ροών για δοκιμαστικούς σκοπούς. Η ευκολότερη λύση που μπορεί να προτιμηθεί είναι η τροφοδότηση του υπό δοκιμή λογισμικού με μια σειρά από ασύνδετες μεταξύ τους ροές που είτε έχουν δημιουργηθεί επί τούτου είτε έχουν ληφθεί από κάποιο πραγματικό περιβάλλον και βρίσκονται προσωρινά αποθηκευμένες σε στατικά αρχεία. Μια τέτοια προσέγγιση δεν επιτρέπει παρά το ελάχιστο και μόνο των δοκιμών που θα μπορούσαν να εκτελεστούν. Επιπλέον, η λήψη ρεαλιστικών δεδομένων από πραγματικά

περιβάλλοντα που παράγουν ροές δεδομένων δεν είναι πάντοτε εφικτή ή επιθυμητή για πολλούς λόγους:

- Συχνά τα δεδομένα αυτά παράγονται σε περιβάλλοντα ιδιόκτητα (proprietary) και ο δανεισμός ή η παραχώρησή τους δεν είναι επιτρεπτά. Για παράδειγμα, δεδομένα που προέρχονται από την παρακολούθηση της κίνησης στρατευμάτων (battalion monitoring).

- Το ίδιο ισχύει και όταν απλά δεν υπάρχει πρόσβαση σε τέτοια δεδομένα εξαιτίας περισσότερο φυσικών αιτίων. Για παράδειγμα σε χώρες όπως η Ελλάδα είναι δύσκολο να βρει κανείς δεδομένα από δίκτυα αισθητήρων, μιας και η τεχνολογία αυτή δεν είναι ιδιαίτερα διαδεδομένη.

- Ακόμη και όταν η λήψη τέτοιων δεδομένων είναι εφικτή τότε και πάλι τα δεδομένα που θα ληφθούν θα είναι τα μόνα που θα είναι διαθέσιμα για τις δοκιμές. Αν χρειαστούν και άλλα που να εμπεριέχουν άλλες ιδιότητες από τα προηγούμενα τότε θα πρέπει να ληφθούν εκ νέου από τη φυσική τους πηγή.

- Συχνά υπάρχει η ανάγκη για δοκιμαστικά δεδομένα που να έχουν συγκεκριμένες ιδιότητες στις οποίες ένα ΣΔΡΔ ή ένας αλγόριθμος χρειάζεται να δοκιμαστεί. Για παράδειγμα για να δοκιμαστεί η ικανότητα ενός αλγορίθμου να εκτελέσει μεγάλες συνενώσεις πάνω σε ροές θα πρέπει να έχει στη διάθεσή του δεδομένα που να εξασφαλίζουν ότι οι συνενώσεις θα έχουν πράγματι τη μεγάλη επιλεξιμότητα (selectivity) στην οποία είναι επιθυμητό να γίνει η δοκιμή. Κάτι τέτοιο δε θα μπορούσε να το εγγυηθεί ούτε καν η λήψη δεδομένων από το ίδιο το περιβάλλον στο οποίο ένα σύστημα στοχεύει, μιας και αντιστοιχούν σε μια ακραία περίπτωση η οποία πιθανώς να μην εμφανίζεται τακτικά.

6.2.2 ΑΝΆΓΚΗ ΓΙΑ ΔΉΛΩΣΗ ΕΝΑΛΛΑΚΤΙΚΏΝ ΣΕΝΑΡΊΩΝ

Τα δύο εργαλεία εξετάσαμε υλοποιούν από ένα συγκεκριμένο σενάριο περιβάλλοντος όπου παράγονται ροές δεδομένων ξεχωριστού ενδιαφέροντος το καθένα. Αν και οι υλοποιήσεις αυτές στοχεύουν στη σκοπιμότητα της ενιαίας δοκιμασίας (benchmarking) για ερευνητικά ΣΔΡΔ, υπάρχουν πολλές ανάγκες που δεν καλύπτονται επαρκώς ή δεν εξετάζονται.

Καταρχήν, τα σενάρια εκτέλεσης είναι εκ των προτέρων πλήρως προσδιορισμένα. Αυτό όσο και αν προσδίδει σαφήνεια στην προσέγγιση θέτει και πολλούς περιορισμούς. Είναι ξεκάθαρα διαφορετική η αναγκαιότητα για μια δοκιμασία – όπως ένα benchmark – και διαφορετική η αναγκαιότητα για την ύπαρξη χρήσιμων δοκιμαστικών ροών δεδομένων. Παρόλο που η λειτουργικότητες στις δύο αυτές περιπτώσεις επικαλύπτονται σε κάποιο βαθμό πολλά από τα στοιχεία που η μία καλύπτει συμπληρώνονται από την άλλη. Χαρακτηριστικό είναι το παράδειγμα του "Linear Road" όπου τα ίδια τα δεδομένα που αυτό διοχετεύει κατά τη διάρκεια της δοκιμασίας δεν τα έχει παράξει το ίδιο αλλά ένας άλλος προσομοιωτής κυκλοφοριακής κίνησης εξειδικευμένος για αυτή τη δουλειά. Αλλά και το "NEXMark" αποτελεί το ίδιο επέκταση ενός εργαλείου παραγωγής XML δεδομένων για XML αποθήκες (XML repositories).

Η δυνατότητα εκμετάλλευσης ενός μονάχα σεναρίου εκτέλεσης δοκιμών είναι ιδιαίτερα χρήσιμη όταν αυτό το σενάριο δοκιμάζεται σε πολλά ΣΔΡΔ ή όμοιους αλγορίθμους έτσι ώστε να παρέχει ένα κοινό σημείο αναφοράς για συγκρίσεις. Παρολαυτά, σε μια ερευνητική δραστηριότητα που διαρκεί για μεγάλη χρονική περίοδο κάτι τέτοιο δεν είναι χρήσιμο παρά σε τελικά στάδια της ανάπτυξης. Μια ερευνητική ομάδα που αναπτύσσει είτε ένα ολόκληρο ΣΔΡΔ είτε ένα τμήμα του, είτε απλά ένα συγκεκριμένο αλγόριθμο, έχει τη συνεχή ανάγκη από δοκιμαστικές ροές δεδομένων και μάλιστα με μεγάλη ποικιλία επιλογών. Μερικές από τις επιλογές αυτές που μπορούμε να καταγράψουμε εδώ είναι οι εξής:

- **Δυνατότητα ορισμού των σχημάτων (schemas).**
  Είναι χρήσιμο να υπάρχει σε πολλές περιπτώσεις μέχρι και πλήρης ελευθερία όσον αφορά τα σχήματα και τις γραμμογραφήσεις των παραγόμενων ροών έτσι ώστε να μπορούν άμεσα και κατά απαίτηση να δηλωθούν και να παραχθούν οι οι επιθυμητές ροές.

- **Δυνατότητα ορισμού οποιουδήποτε πλήθους ροών.**
  Ανάλογα με την κλιμάκωση των δοκιμών ή τις συγκεκριμένες ιδιαιτερότητές τους θα πρέπει να είναι δυνατός ο ορισμός οποιουδήποτε πλήθους ανεξάρτητων ροών που να υπακούν σε δοσμένα σχήματα.

- **Δυνατότητα ορισμού σεναρίων εκτέλεσης.**
  Ένα συγκεκριμένο σενάριο εκτέλεσης δεν είναι αρκετό για να καλύψει κάθε ανάγκη δοκιμών. Πάντοτε παρουσιάζονται συνδυασμοί που οδηγούν σε περιπτώσεις απρόβλεπτες και μόνο με τη δυνατότητα δήλωσης και δοκιμής πολλών συνδυαστικών

σεναρίων μπορούν πολλές από αυτές να καλυφθούν έγκαιρα. Επίσης, πολλές φορές είναι δυνατόν να υπάρχει ανάγκη για εκτέλεση σεναρίων περιορισμένης λειτουργικότητας και όχι σαν και αυτό μιας πλήρους δοκιμασίας (benchmark) και αυτό θα πρέπει να μπορεί να ορίζεται εξίσου εύκολα.

- **Δυνατότητά ορισμού σημασιολογικών στοιχείων.**
  Η μεγάλη πρόκληση για ένα εργαλείο σαν και αυτό που συζητούμε εδώ είναι η δυνατότητα ορισμού περισσότερων του ενός σημασιολογικών στοιχείων των ροών που θα παραχθούν. Κάτι τέτοιο είναι κατά τα άλλα απόλυτα επιθυμητό και θα έδινε τη μέγιστη ευελιξία στη δήλωση και παραγωγή δοκιμαστικών ροών.

Πρέπει να σημειωθεί ότι το εργαλείο που θα συζητήσουμε παρακάτω δεν αποσκοπεί στον ίδιο ερευνητικό χώρο με τα δύο εργαλεία που συζητήσαμε στο προηγούμενο κεφάλαιο. Η λειτουργικότητα που το κάθε ένα προσφέρει είναι σε μερικά σημεία κοινή, όμως έχει γίνει, πλέον, κατανοητό ότι η εστίαση είναι εν γένει διαφορετική.

6.3 ΠΕΡΙΓΡΑΦΉ ΤΟΥ ΕΡΓΑΛΕΊΟΥ ΠΑΡΑΓΩΓΉΣ ΔΟΚΙΜΑΣΤΙΚΏΝ ΡΟΏΝ ΔΕΔΟΜΈΝΩΝ

Μια πρώτη εκτίμηση των απαιτήσεων που θα είχε κανείς από ένα εργαλείο παραγωγής δοκιμαστικών ροών δεδομένων (stream generator) έγινε στην προηγούμενη παράγραφο. Επίσης η γνώση και η κατανόηση του χώρου και των προβλημάτων που αποκομίζουμε τόσο από αυτό το κεφάλαιο (6ο) όσο και από το προηγούμενο (5ο) λειτουργεί σαν οδηγός στη μελέτη για την καταγραφή των προδιαγραφών για το υπό ανάπτυξη εργαλείο υποβοηθητικά και συμπληρωματικά. Είδαμε ότι μεγάλη σημασία θα πρέπει να δοθεί στη διαδικασία της ανάλυσης έτσι ώστε το εργαλείο αυτό να προσκομίζει κάτι νέο στις προσπάθειες που γίνονται στο ερευνητικό πεδίο της διαχείρισης ροών δεδομένων. Η σκοπιμότητά του είναι διττή. Από τη μία πλευρά φιλοδοξεί να αποτελέσει πράγματι ένα χρήσιμο εργαλείο παραγωγής ροών δεδομένων αλλά και, από την άλλη, να δώσει τροφή για την βαθύτερη κατανόηση των ιδιαιτεροτήτων τους. Το τελευταίο είναι εξίσου σημαντικό με το πρώτο, μιας και ο χώρος των Συστημάτων Διαχείρισης Ροών Δεδομένων παρουσιάζει αυξημένο ενδιαφέρον, όπως έχουμε διαπιστώσει, και οι εφαρμογές των οποίων άπτεται είναι ποικίλες και προκαλούν ξεχωριστούς προβληματισμούς.

Στην παρούσα εργασία μελετάται και υλοποιείται πρωτογενώς ένα εργαλείο παραγωγής δοκιμαστικών ροών δεδομένων οι οποίες ορίζονται σε XML γλώσσα με γραφικό περιβάλλον.

Καταβλήθηκε προσπάθεια κατά την διερεύνηση και την ανάλυση του πεδίου αυτού ώστε η προσέγγιση που θα ακολουθηθεί να μην δίνει έμφαση σε καμία συγκεκριμένη εφαρμογή ροών δεδομένων και να μην εστιάζει σε καμία από αυτές, που ούτως ή άλλως έχουμε προηγούμενα εντοπίσει. Το έργο αυτό, άλλωστε, το έχουν αναλάβει ερευνητικά προγράμματα όπως το "Linear Road" και το "NEXMark", για το είδος της εφαρμογής που το καθένα τους έχει επιλέξει να εστιάσει. Αντίθετα, δόθηκε έμφαση στο να είναι αυτή κατά το δυνατόν ευέλικτη και γενικόλογη (generic). Στα αποτελέσματα της προσπάθειας αυτής δεν εξετάζεται το ερώτημα αν η γενικόλογη αυτή προσέγγιση μπορεί να εφαρμοστεί για κάθε είδος ροών δεδομένων. Κάτι τέτοιο θα ήταν κάπως ουτοπικό. Είναι βέβαιο πως πολλά από τα είδη ροών δεδομένων που έχουμε εντοπίσει μπορούν να παραχθούν υπολογιστικά με κάποιο εργαλείο αυτού του είδους. Η πρόκληση έγκειται πρώτα στην προσπάθεια οι ροές αυτές να είναι δυνατόν να οριστούν και να παραχθούν και, βέβαια, στη συνέχεια, να έχουν και σημασιολογική αξία. Επιπλέον, δεν είναι στους στόχους της προσπάθειας αυτής να αναπτύξει ένα εργαλείο που να αποδομεί τις έννοιες των ροών δεδομένων τόσο εξαντλητικά ώστε να μπορεί να τις παράξει όλες. Οπωσδήποτε, δεν είναι καν δυνατόν να μιλά κανείς για το σύνολο των ειδών των ροών, μιας και ολοένα και περισσότερες εφαρμογές κάνουν την εμφάνισή τους, καθώς η πληροφορία που διακινείται στις μέρες μας είναι τεράστια και πολυποίκιλη.

Σημαντική βαρύτητα, επίσης, έχει δοθεί στο σχεδιασμό του εργαλείου αυτού. Όπως προαναφέραμε δεν είναι στους στόχους της παρούσας φάσης της ανάπτυξης η υποστήριξη πολλών ειδών ροών δεδομένων ή περίπλοκων λειτουργιών (features). Είναι, όμως, πρώτιστο μέλημα η δομημένη και ευέλικτη προσέγγιση τόσο σε σχεδιαστικό όσο και σε προγραμματιστικό επίπεδο. Αυτό γιατί το εγχείρημα δεν είναι απλά ένα εργαλείο που υλοποιείται για μία φορά και ολοκληρώνεται αλλά ένα ερευνητικό πρόγραμμα που οφείλει να προβλέψει ότι μπορεί να έχει και συνέχεια. Επομένως, έχει καταναλωθεί σημαντική προσπάθεια στο να είναι η σχεδίασή του όσο το δυνατόν πιο δομημένη και εξελίξιμη μελλοντικά. Έτσι έχει δημιουργηθεί ένα ευέλικτο υπόβαθρο από πλευράς σχεδιαστικής προσέγγισης που επιτρέπει να επεκταθούν πάνω του νέες λειτουργικότητες ή, το λιγότερο, να μεταποιηθεί προς κάτι καινούριο με το μικρότερο κόπο.

6.4 ΟΙ ΡΟΈΣ ΠΟΥ ΠΑΡΆΓΟΝΤΑΙ

6.4.1 ΑΠΛΈΣ ΡΟΈΣ

Η πρώτη σημαντική λειτουργικότητα του εργαλείου αυτού είναι ο ορισμός και η παραγωγή απλών ροών δεδομένων. Λέγοντας απλές ροές δεδομένων εννοούμε ροές που δεν έχουν μεταξύ τους

σημασιολογικές εξαρτήσεις. Ροές, δηλαδή που παράγονται ανεξάρτητα η μία από την άλλη και χωρίς να επηρεάζουν η μία την άλλη. Δείγματα τέτοιων ροών υπάρχουν πολλά. Για παράδειγμα σε ένα δίκτυο αισθητήρων όπου πολλοί αισθητήρες που είναι εξειδικευμένοι σε διαφορετικά φυσικά μεγέθη ο καθένας και αναφέρουν τις μετρήσεις που ανιχνεύουν, οι μετρήσεις του καθενός είναι ανεξάρτητες από τις μετρήσεις των υπολοίπων. Κάτι τέτοιο, επομένως, ανταποκρίνεται σε μια πραγματική αναγκαιότητα που συναντούμε σε υπαρκτές εφαρμογές ροών δεδομένων.

Οι ροές αυτές είναι δυνατόν να οριστούν πλήρως από το εργαλείο αυτό. Αυτό σημαίνει ότι το σχήμα (schema) στο οποίο ανήκουν και η γραμμογράφησή τους βρίσκονται στη διάθεση εκείνου που τις ορίζει μέσω του εργαλείου. Για αυτό το λόγο το εργαλείο παρέχει ένα σύνολο από δυνατότητες ορισμού ροών έτσι ώστε να μπορεί να καλύπτεται μια πληθώρα συνδυασμών που να ανταποκρίνεται σε πολλές ανάγκες. Έτσι μία ροή μπορεί να περιέχει τα πεδία που είναι επιθυμητό να περιέχει και τα οποία ανήκουν σε κάποιο από τους γνωστούς τύπους δεδομένων. Στην παρούσα φάση της υλοποίησης έχουν ολοκληρωθεί οι πιο θεμελιώδεις μόνο από τους τύπους δεδομένων (ακέραιος-integer, αλφαριθμητικό-string κλπ), έχει, όμως, γίνει τέτοια σχεδίαση ώστε πολύ εύκολα να μπορούν να προστεθούν και νέοι.

6.4.2 ΥΠΌ-ΡΟΈΣ

Ένα ακόμη είδος γενικών ροών (generic) ροών που επιλέχθηκε να υποστηριχθούν από το εργαλείο είναι είναι οι εμφωλευμένες ροές για τις οποίες έγινε λόγος στο προηγούμενο κεφάλαιο. Οι εμφωλευμένες ροές (nested streams ή υπό-ροές) είναι ροές δεδομένων που εκκινούνται από άλλες ροές όταν κάποιο γεγονός συμβαίνει. Η ιδέα για αυτή την κατηγορία των ροών έχει εμπνευστεί από το [64] όπου οι ροές αυτές καθώς και κάποιες από τις εφαρμογές στις οποίες εντοπίζονται αναλύονται σε βάθος και τρόποι διαχείρισής τους προτείνονται. Ένα επιπλέον σχετικό παράδειγμα που θα μπορούσαμε να αναφέρουμε από το χώρο των δικτύων αισθητήρων είναι η ανάγκη για ενεργοποίηση άλλων αισθητήρων όταν κάποιος από αυτούς ανιχνεύει κάποια τιμή ενδιαφέροντος. Για παράδειγμα όταν ένας αισθητήρας θερμοκρασίας σε ένα κτήριο ανιχνεύει τιμές μεγαλύτερες από ένα κατώφλι θα θέλαμε ενδεχομένως να ενεργοποιήσουμε και τους αισθητήρες φωτισμού, οι οποίοι έμεναν ανενεργοί για να μη σπαταλούν τη μπαταρία τους, για να διαπιστώσουμε αν υπάρχει κίνδυνος πυρκαγιάς. Έτσι εξαιτίας των μετρήσεων κάποιας ροής δημιουργούνται και κάποιες άλλες υπό-ροές αυτής.

Στην παρούσα προσέγγιση αυτού του είδους οι ροές μοντελοποιούνται μέσω συνθηκών. Οι συνθήκες αυτές καθορίζουν τη στιγμή όπου οι εμφωλευμένες ροές ξεκινούν να παράγονται και,

φυσικά, ποιες είναι αυτές οι ίδιες. Η λογική στην οποία βασίζονται οι συνθήκες λειτουργεί και αντίστροφα. Δηλαδή, όταν η συνθήκη σταματήσει να ισχύει τότε παύει και να παράγεται η εμφωλευμένη ροή. Ακόμη, οι συνθήκες αυτές μπορούν να προκαλέσουν την εκκίνηση οποιουδήποτε τύπου ροής έχει δηλωθεί στο εργαλείο ότι συμμετέχει στην εκάστοτε δοκιμή. Η σχεδίαση είναι τέτοια ώστε σε μεταγενέστερη υλοποίηση να μπορούν να παράγονται και περισσότερες από μία υπό-ροές σαν αποτέλεσμα ενός γεγονότος που συμβαίνει σε κάποια άλλη.

6.4.3 ΔΙΑΔΟΧΈΣ ΡΟΏΝ

Το τελευταίο είδος ροών που μπορεί να δηλωθεί από το υπό ανάπτυξη εργαλείο είναι αυτό που ονομάστηκε “διαδοχές ροών”. Με τον όρο αυτό εννοούμε τη δυνατότητα ορισμού της διαδοχής με την οποία συγκεκριμένες ροές θα παράγονται για να τροφοδοτήσουν ένα ΣΔΡΔ. Έτσι εάν στο εργαλείο έχουν δηλωθεί n-ροές τότε με τον τρόπο αυτό θα είναι δυνατό να καθοριστεί με ποια σειρά (διαδοχή) θα εμφανίζονται οι πλειάδες της κάθε μιας καθώς θα παράγονται. Η δυνατότητα αυτή είναι αρκετά συγκεκριμένη και αντανακλά σε ροές δεδομένων με επίσης αρκετά συγκεκριμένη σημασιολογία. Η μοντελοποίηση που γίνεται προσπάθεια να επιτευχθεί σε αυτό το σημείο είναι για την παραγωγή σύνθετων ροών που έχουν την ανάγκη να προσδιορίσουν τη διαδοχή αυτή ανάμεσα στις ροές που εμπερικλείουν. Στο παράδειγμα των δικτύων αισθητήρων θα μπορούσαμε να αναφέρουμε το εξής παράδειγμα: σε ένα σύστημα παρακολούθησης ενός κτηρίου μέσω αισθητήρων είναι δυνατόν να είναι επιθυμητό ομάδες αισθητήρων να αναφέρουν τιμές με συγκεκριμένη σειρά και ομαδοποίηση σε τακτά χρονικά διαστήματα για λόγους αρχιτεκτονικής του συστήματος.

Τα τρία αυτά είδη ροών συνοψίζονται στον παρακάτω πίνακα. Η επιλογή τους έχει γίνει με γνώμονα το πόσο γενική και θεμελιώδης είναι η φύση τους και άρα πόσο χρήσιμες θα φανούν στην οικοδόμηση και εξέλιξη του εργαλείου αυτού. Δεν είναι δηλαδή άμεσος σκοπός του εργαλείου να παράγει απόλυτα τα συγκεκριμένα είδη ροών αλλά βασιζόμενο σε αυτά να θέσει τη βάση για μια περισσότερο γενική μελέτη της παραγωγής δοκιμαστικών ροών δεδομένων (stream generation) που να μπορεί να εξελιχθεί περαιτέρω.

Είδος Ροής	***Περιγραφή***
Απλές ροές	Ροές που δεν έχουν σημασιολογικές εξαρτήσεις η μία από την άλλη.

Υπό-ροές	Ροές που παράγονται σαν αποτέλεσμα ενός γεγονότος κάποιας άλλης ροής.
Διαδοχές ροών	Ροές από σύνθεση άλλων με προσδιορισμό της σειράς ανάμεσά τους.

6.5 ΕΝΝΟΙΟΛΟΓΙΚΌ ΠΛΑΊΣΙΟ

Είδαμε, λοιπόν, μέχρι στιγμής τον τρόπο με τον οποίο οδηγούμαστε στην αναγκαιότητα για τη μελέτη ενός εργαλείου παραγωγής ροών δεδομένων προσδιορισμένων από το χρήστη, έχουμε διερευνήσει τους χώρους στους οποίους οι ροές εμφανίζονται ως κυρίαρχες οντότητες και έχουμε διατυπώσει ποιες είναι οι γενικές απαιτήσεις που θα είχε κανείς από ένα τέτοιο εργαλείο. Εννοιολογικά οι ιδέες στις οποίες η προσέγγιση αυτή στηρίζεται αποτυπώνεται στα δύο σχήματα που ακολουθούν (Σχήμα 6.1 και Σχήμα 6.2).

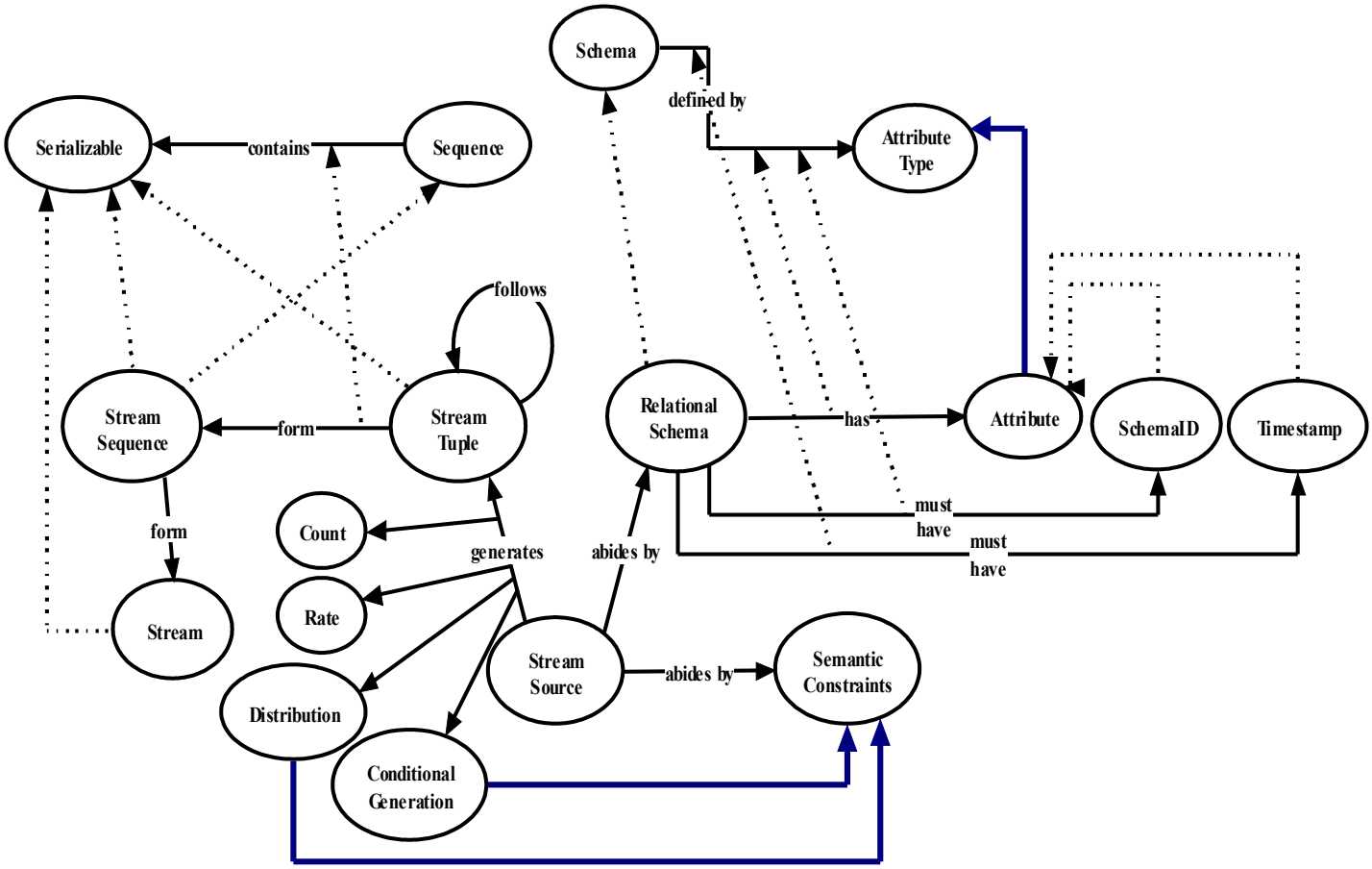
Στο Σχήμα 6.1 έχει γίνει προσπάθεια να αποτυπωθούν γενικευμένα οι ιδέες αυτές σε οντολογικό επίπεδο και η – γραφική – γλώσσα απεικόνισης που έχει επιλεχθεί είναι η SIS-Telos [74]. Στο σχήμα αυτό οι κυριότερες κλάσεις που συμμετέχουν και αξίζουν να αναφερθούν ανεξάρτητα είναι οι εξής:

- **Schema – Relational Schema**

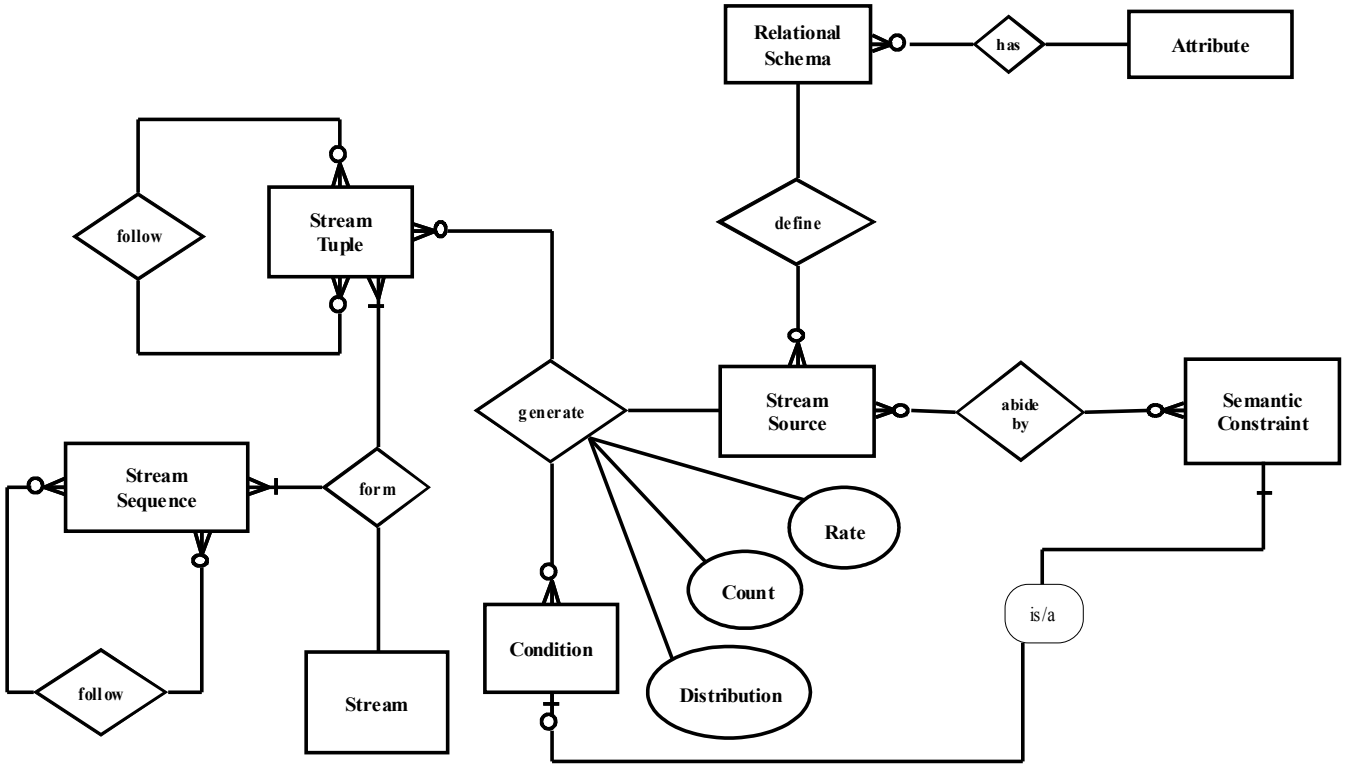
  Η κλάση Schema βρίσκεται σε υψηλό επίπεδο αφαίρεσης/γενίκευσης (επίπεδο M1-Class στη SIS-Telos) και ομαδοποιεί όλα τα σχήματα (schemas) των ροών δεδομένων. Έτσι ένα Schema καθορίζεται (defined by) από τύπους πεδίων (Attribute Type). Η κλάση Relational Schema είναι πραγμάτωση (instance) της κλάσης Schema και έτσι αποτελεί το σημείο αναφοράς για τα σχεσιακά σχήματα που αφορούν τις ροές που θα παράγονται (και βρίσκεται σε επίπεδο S-Class στη SIS-Telos). Ως πραγμάτωση της Schema διαθέτει (has) πεδία (Attribute) τα οποία είναι πραγματώσεις των τύπων πεδίων (Attribute Type).

- **Attribute Type – Attribute**

  Η κλάση Attribute Type ομαδοποιεί τους τύπους πεδίων που ένα σχεσιακό σχήμα ροών δεδομένων επιτρέπεται να έχει και η Attribute αποτελεί πραγμάτωση αυτής. Τα SchemaID και Timestamp αποτελούν ενδεικτικά πεδία που ανήκουν στην κλάση Attribute Type και είναι τα ίδια πεδία σχεσιακού σχήματος.



Σχήμα 6.1 – Εννοιολογικό πλαίσιο – οντολογική προσέγγιση (διάγραμμα SIS-Telos)



Σχήμα 6.2 – Εννοιολογικό πλαίσιο – διάγραμμα οντοτήτων συσχετίσεων (E-R)

- **Serializable**

  Η κλάση Serializable (σε M1-Class επίπεδο) αναφέρεται σε αντικείμενα που έχουν τη δυνατότητα να μπορούν να μεταφερθούν από ένα σύστημα σε ένα άλλο[23]. Για παράδειγμα, μια πηγή ροών δεδομένων δεν είναι "μεταφέρσιμη" οντότητα, ενώ μια πλειάδα που αυτή παράγει είναι.

- **Sequence**

  Η κλάση Sequence ομαδοποιεί τις οντότητες που έχουν τη δυνατότητα να τεθούν σε μία συγκεκριμένη διαδοχική σειρά. Τέτοιες οντότητες είναι αντικείμενα της κλάσης Serializable τα οποία μπορούν να είναι μεταφέρσιμα.

- **Stream Source**

  Αποτελεί αρκετά κομβική οντότητα της οντολογικής αναπαράστασης και παρολαυτά επιλέχθηκε να τοποθετηθεί σε επίπεδο πραγμάτωσης S-Class μιας και αντικατοπτρίζει περισσότερο λειτουργικές παρά αφαιρετικές έννοιες. Η κλάση Stream Source ομαδοποιεί τις οντότητες εκείνες που είναι παραγωγοί ροών δεδομένων, δηλαδή τις ίδιες τις πηγές δεδομένων. Όπως φαίνεται στο σχήμα μία πηγή ροών (Stream Source) ακολουθεί κάποιο σχεσιακό σχήμα (Relational Schema), υπόκειται σε σημασιολογικούς περιορισμούς (Semantic Constraints) και παράγει πλειάδες (generates). Η ιδιότητα "generates" της Stream Source, που αντιστοιχεί στην πράξη καθεαυτή της παραγωγής πλειάδων, έχει ως ιδιότητες τον αριθμό (count) και το ρυθμό (rate) με τον οποίο οι πλειάδες παράγονται, την κατανομή των τιμών που οι πλειάδες λαμβάνουν (distribution) και τις πιθανές συνθήκες (Conditional Generation) που καθορίζουν την παραγωγή των ροών. Για τις κατανομές θα γίνει λόγος παρακάτω. Οι συνθήκες αυτές αναφέρονται στη δυνατότητα παραγωγής υπό-ροών, που συζητήθηκαν στην προηγούμενη παράγραφο.

- **Stream Tuple**

  Η κλάση Stream Tuple αντιπροσωπεύει την πλειάδα που παράγεται (generates) από μια πηγή ροής δεδομένων (Stream Source). Μια πλειάδα ακολουθεί το σχήμα της πηγής που την παράγει και πολλές πλειάδες μαζί σχηματίζουν σειρές πλειάδων ή διαδοχές πλειάδων (Stream Sequence). Ακόμη, μια πλειάδα είναι πραγμάτωση της κλάσης Serializable μιας και είναι οντότητα που μεταφέρεται από το εργαλείο παραγωγής ροών προς το σύστημα διαχείρισής ροών.

23 Ο όρος Serializable είναι δανεισμένος από την ορολογία της γλώσσας Java και σημαίνει αυτά ακριβώς τα αντικείμενα που έχουν τη δυνατότητα να μετατραπούν σε σειριακή μορφή, να μετατραπούν, δηλαδή, σε σειρά από μεταφέρσιμα δεδομένα.

- **Stream Sequence**

  Η κλάση Stream Sequence αντιπροσωπεύει σειρές πλειάδων που σχηματίζονται (form) από τις παραγόμενες πλειάδες (Tuple). Είναι και αυτή μέλος της Serializable για όμοιους λόγους. Μια Stream Sequence σχηματίζεται είτε από πλειάδες της ίδιας πηγής είτε από πλειάδες πολλών διαφορετικών ροών. Έτσι μπορεί να μοντελοποιηθεί και η έννοια της διαδοχής που συζητήθηκε στην προηγούμενη παράγραφο.

- **Stream**

  Τέλος, η κλάση Stream που αντιπροσωπεύει την ίδια τη ροή δεδομένων, που είναι κατά κάποιο τρόπο και αυτή στην οποία στοχεύει και η όλη θεώρηση. Αντικείμενα της κλάσης Stream σχηματίζονται (form) από σειρές πλειάδων, οι οποίες μπορεί, όπως αναφέρθηκε, να είναι είτε μεμονωμένες πλειάδες κάποιας πηγής ροών, είτε ομάδες πλειάδων διαφορετικών ροών. Και αυτή είναι μέλος της Serializable μιας μια ροή κατεξοχήν είναι οντότητα μεταφέρσιμη από ένα σύστημα (το εργαλείο παραγωγής) σε ένα άλλο (το σύστημα διαχείρισης).

Στο Σχήμα 6.2 έχει καταβληθεί προσπάθεια να γίνει ένα βήμα παραπέρα προς την κατεύθυνση του να αποκτήσει υπόσταση το παραπάνω οντολογικό σχήμα με ένα διάγραμμα οντοτήτων συσχετίσεων (entity relationship – ER). Το διάγραμμα οντοτήτων συσχετίσεων είναι περισσότερο εύληπτο από μια οντολογία μιας και παρουσιάζει περισσότερο λειτουργικά ένα περιβάλλον, αλλά και για αυτό το λόγο είναι και πιο περιορισμένο εννοιολογικά. Με δύο λόγια, στο Σχήμα 6.2 φαίνεται μία πηγή ροών δεδομένων (Stream Source) που ορίζεται από το σχήμα (Relational Schema) στο οποίο ανήκει και παράγει πλειάδες (Stream Tuple) με συγκεκριμένο ρυθμό (Rate) και κατανομή (Distribution). Οι πλειάδες αυτές ακολουθούνται (follows) από άλλες πλειάδες ή σχηματίζουν σειρές πλειάδων (Stream Sequence), οι οποίες ακολουθούνται από άλλες σειρές πλειάδων ή απλές πλειάδες, και όλες μαζί συνιστούν μια ροή (Stream).

Το σημείο στο οποίο υπήρξε η σημαντικότερη πρόκληση από πλευράς ερευνητικού ενδιαφέροντος είναι η προσπάθεια να είναι η προσέγγιση αυτή όσο το δυνατόν περισσότερο γενικευμένη και, επομένως, ευέλικτη. Έχουν εντοπιστεί τα σημεία στα οποία τα εργαλεία benchmarking που είδαμε δίνουν έμφαση και προτιμήθηκε να μη γίνει το ίδιο και σε αυτή την περίπτωση. Η παρούσα προσέγγιση αντιμετωπίζει τις ροές σαν αντικείμενα με ιδιότητες οι οποίες καθορίζουν τη συμπεριφορά τους και προς αυτό είναι αρκετά αντικειμενοστραφής. Αυτό θα γίνει περισσότερο κατανοητό στη συνέχεια όπου θα μιλήσουμε για την προσέγγιση της ίδιας της ανάπτυξης του

εργαλείου.

6.6 ΥΠΟΣΤΗΡΙΚΤΙΚΉ ΓΛΏΣΣΑ ΟΡΙΣΜΟΎ ΡΟΏΝ

Για να είναι δυνατόν από ένα χρήστη να οριστούν οι επιθυμητές ροές δεδομένων το εργαλείο οφείλει να διαθέτει κάποιο δομημένο και εύληπτο τρόπο για αυτή τη διαδικασία. Έτσι αναπτύχθηκε στα πλαίσια της ανάπτυξης του εργαλείου μια υποστηρικτική γλώσσα ορισμού ροών δεδομένων σε XML μορφή. Η XML επιλέχθηκε γιατί είναι πραγματικά μια εξαιρετικά δομημένη προσέγγιση στην μορφοποίηση οποιουδήποτε είδους πληροφορίας και, εκτός των άλλων, είναι αρκετά φιλική προς τον άνθρωπο για να γράψει και να διαβάσει από αυτήν εάν χρειαστεί.

Με XML, λοιπόν, είναι δυνατόν σε ένα χρήστη να ορίσει τις ροές που έχει ανάγκη για τις δοκιμές που επιθυμεί να διεξάγει. Η λογική της γλώσσας αυτής ακολουθεί τη διαδικασία της δημιουργίας ενός project ροών που καλείται “STREAM_PROJECT”. Με αυτό τον τρόπο ο χρήστης καλείται να δημιουργήσει κάθε φορά ένα project ροών δεδομένων το οποίο θα μπορεί να έχει τους δικούς του τύπους ροών και τις δικές του ροές. Αναλυτικά τα XML tags (XML δηλώσεις) που απαρτίζουν αυτή τη γλώσσα φαίνονται παρακάτω.

<STREAM_PROJECT> ... </STREAM_PROJECT>
Περιέχει όλες τις υπόλοιπες XML δηλώσεις του project. Η χρήση του είναι απλά δομική έτσι ώστε όλες οι δηλώσεις να ομαδοποιούνται και να περικλείονται σε ένα κοινό νοηματικό πλαίσιο.

<SCHEMA name="xxx"> ... </SCHEMA>
Δηλώνει ένα σχεσιακό σχήμα και του προσδίδει ένα όνομα (name), το οποίο πρέπει να είναι μοναδικό, δηλαδή να μην αντιστοιχεί σε κανένα άλλο σχεσιακό σχήμα που ορίζεται στο ίδιο project. Ένα σχεσιακό σχήμα μπορεί να διαθέτει διάφορους τύπους πεδίων και αποτελεί ουσιαστικά τον ορισμό των τύπων στους οποίους θα μπορούν να ανήκουν οι ροές που θα παράγονται από το εργαλείο. Η δήλωση αυτή περιέχεται μέσα σε ένα STREAM_PROJECT και μπορεί να επαναληφθεί οσεσδήποτε φορές με τον περιορισμό του μοναδικού ονόματος.

<ATTRIBUTE name="xxx"> ... </ATTRIBUTE>
Δηλώνει ένα τύπο ορίσματος που ένα σχεσιακό σχήμα περιέχει και του προσδίδει ένα χαρακτηριστικό όνομα. Περιέχεται μέσα σε ένα SCHEMA και μπορεί να επαναληφθεί εκεί οσεσδήποτε φορές χρειαστεί με τον περιορισμό ότι ένα SCHEMA δεν επιτρέπεται να περιέχει δύο τύπους ορισμάτων με το ίδιο όνομα.

<TYPE> ... </TYPE>

Δηλώνει τον τύπο δεδομένων στον οποίο ανήκει ένας τύπος ορίσματος ενός σχήματος. Περιέχεται μέσα σε ένα ATTRIBUTE και είναι μοναδικό για το καθένα. Οι υποστηριζόμενοι μέχρι στιγμής τύποι δεδομένων είναι ο ακέραιος (integer), το αλφαριθμητικό (string) και το χρονόσημο (timestamp).

<RANGE min="x" max="y"> ... </RANGE>

Δηλώνει το εύρος τιμών που ένας τύπος ορίσματος ενός σχήματος επιτρέπεται να πάρει. Περιέχεται μέσα σε ένα ATTRIBUTE και είναι μοναδικό για αυτό. Το εύρος τιμών ορίζεται μέσα από τις ιδιότητες min και max αν πρόκειται για αριθμητικό τύπο δεδομένων. Αν πρόκειται για αλφαριθμητικό τότε το RANGE θα πρέπει να έχει την ιδιότητα delim (από το delimeter=διαχωριστικό) και να περιέχει τις δυνατές αλφαριθμητικές τιμές μέσα στο σώμα του RANGE διαχωρισμένες από το χαρακτήρα που βρίσκεται μέσα στο delim. Γίνεται έλεγχος έτσι ώστε οι τιμές του RANGE να είναι επιτρεπτές.

<DISTRIBUTION> xxx </DISTRIBUTION>

Δηλώνει την κατανομή των τιμών που ένα όρισμα/πεδίο ενός σχήματος λαμβάνει μέσα στο εύρος τιμών που έχει δηλωθεί. Η επιτρεπόμενη τιμή μέχρι στιγμής είναι μόνο η τυχαία κατανομή που μια προγραμματιστική γλώσσα μπορεί να επιτύχει και ουσιαστικά αντιστοιχεί σε ομοιόμορφη (uniform) κατανομή. Η ύπαρξη αυτής της δήλωσης βρίσκεται εδώ για καθαρά μελλοντικούς λόγους όπου νέες κατανομές θα μπορούν να προστεθούν στο εργαλείο εφόσον αυτές ήδη θα υποστηρίζονται. Με την υποστήριξη των κατανομών είναι δυνατόν να δοθεί μια ισχυρή σημασιολογική δυνατότητα στη δήλωση των ροών. Η ύπαρξη των κατανομών μπορεί να προσδώσει ρεαλισμό στις παραγόμενες ροές και να αποτελέσει και μια ξεχωριστή λειτουργικότητα για τους συσχετισμούς των ροών. Για παράδειγμα αν ανάμεσα σε δύο ροές επιθυμεί κανείς να εκτελέσει συνενώσεις τότε ορίζοντας διαφορετικές κατανομές στις ροές αυτές θα μπορέσει να επιτύχει διαφορετικές επιλεξιμότητες (selectivities) στο αποτέλεσμα των συνενώσεων και να δώσει με αυτό τον τρόπο νέα τροπή στο σενάριο εκτέλεσης.

<STREAM_SOURCE type="xxx" name="yyy"> ... </STREAM_SOURCE>

Δηλώνει μια πηγή ροής δεδομένων η οποία αντιστοιχεί σε ένα από τα υπάρχοντα σχεσιακά σχήματα (SCHEMA's) που έχουν δηλωθεί πρωτύτερα στο project και έχει ένα μοναδικό αναγνωριστικό όνομα. Περιέχεται μέσα σε ένα STREAM_PROJECT και μπορεί να εμφανιστεί οσεσδήποτε φορές είναι επιθυμητό με τον περιορισμό του μοναδικού ονόματος. Το σχήμα στο

οποίο αντιστοιχεί η πηγή ροής αυτή ορίζεται από την ιδιότητα type, της οποίας το περιεχόμενο οφείλει να είναι το όνομα ενός υπαρκτού για το project ονόματος ενός σχήματος.

<RATE type="xxx" value="y"> ... </RATE>

Ορίζει το ρυθμό παραγωγής μιας ροής δεδομένων από μια πηγή ροής. Περιέχεται μέσα σε ένα STREAM_SOURCE μία μοναδική φορά. Η ιδιότητα type μπορεί να πάρει τις τιμές second, minute και hour και η ιδιότητα value μπορεί να πάρει ακέραιες τιμές και η σημασιολογία αυτού είναι ότι ο ρυθμός παραγωγής είναι "value" πλήθος πλειάδων ανά "type" ποσό χρόνου (πχ 5 πλειάδες το λεπτό, ή 10 πλειάδες το δευτερόλεπτο).

<CONDITION origin="xxx.yyy" type="www" than="z"> ... </CONDITION>

Η δήλωση CONDITION προσδιορίζει μία συνθήκη υπό την οποία παράγεται μια εμφωλευμένη ροή. Η ιδιότητα origin προσδιορίζει το πεδίο (ATTRIBUTE) κάποιας υπαρκτής πηγής ροής (STREAM_SOURCE) για το οποίο ορίζεται η συνθήκη. Η ιδιότητα type ορίζει τον τύπο της συνθήκης ο οποίος μπορεί να είναι "greater" ή "less" και η ιδιότητα than την τιμή της συνθήκης. Για παράδειγμα ο συνδυασμός origin="terrace.pressure_value" type="greater" than="200" δηλώνει μία συνθήκη πάνω στο πεδίο pressure_value της ροής terrace όταν η τιμή που λάβει αυτό γίνει μεγαλύτερη από 200.

<NESTED_STREAM_SOURCE type="xxx" name="yyy"> ...
</NESTED_STREAM_SOURCE>

Η δήλωση αυτή προσδιορίζει μια εμφωλευμένη ροή που παράγεται κάτω από μία δηλωμένη συνθήκη. Περιέχεται μέσα σε μια CONDITION και έχει ένα αναγνωριστικό όνομα name και έναν τύπο type που δηλώνει το σχήμα της εμφωλευμένης ροής, ο οποίος πρέπει να είναι ένα από τα υπαρκτά σχήματα που έχουν δηλωθεί στο project. Πρέπει να σημειωθεί ότι κατά τα άλλα μια NESTED_STREAM_SOURCE για λόγους σχεδιασμού και ομοιομορφίας δεν είναι τίποτε άλλο παρά μια STREAM_SOURCE και αυτή. Ακόμη περιέχει και ένα ρυθμό παραγωγής με τον οποίο παράγεται και ο οποίος είναι πανομοιότυπος με το RATE που αναφέρθηκε παραπάνω. Στην παρούσα φάση της υλοποίησης υποστηρίζεται συντακτικά αλλά δεν παράγεται στις παραγόμενες ροές ακόμη.

Οι παραπάνω δηλώσεις είναι αρκετές για να ορίσουν και να εκκινήσουν ένα απλό project παραγωγής ροών δεδομένων. Ακόμη δηλώσεις που δίνουν τη δυνατότητα ορισμού επιπλέον στοιχείων είναι οι παρακάτω.

<SCENARIO> ... </SCENARIO>

Περιέχεται μέσα στο STREAM_PROJECT και ομαδοποιεί τις επιπλέον δηλώσεις που αναφέραμε.

<STREAM_SEQUENCE name="xxx"> ... </STREAM_SEQUENCE>

Δηλώνει μια διαδοχή πλειάδων με όνομα name το οποίο είναι μοναδικό μέσα στο ίδιο SCENARIO. Περιέχεται μέσα σε ένα SCENARIO. Οι διαδοχές – σειρές – πλειάδων έχουν οριστεί παραπάνω στους υποστηριζόμενους τύπους ροών του εργαλείου. Μια STREAM_SEQUENCE περιέχει και ένα ρυθμό παραγωγής με τον οποίο παράγεται και ο οποίος είναι πανομοιότυπος με το RATE που αναφέρθηκε παραπάνω.

<TUPLES from="rear_window_light" count="1"> ... </TUPLES>

Η δήλωση αυτή ορίζει από πού παίρνει πλειάδες μια σειρά ροών STREAM_SEQUENCE. Η ιδιότητα from ορίζει μια υπαρκτή πηγή ροής STREAM_SOURCE από την οποία λαμβάνει πλειάδες και η ιδιότητα count πόσο πλήθος πλειάδων λαμβάνει από αυτήν. Περιέχεται μέσα σε ένα STREAM_SEQUENCE και μπορεί να εμφανιστεί εκεί όσες φορές χρειάζεται ανάλογα με τις ανάγκες τις εκάστοτε δοκιμής. Στην παρούσα φάση της υλοποίησης του εργαλείου οι δηλώσεις αυτές υποστηρίζονται συντακτικά όμως οι αντίστοιχες πλειάδες δεν παράγονται ακόμη.

Ας δούμε ένα παράδειγμα ενός ολοκληρωμένου STREAM_PROJECT στη συνέχεια.

```
<?xml version="1.0" encoding="UTF-8"?>

<!-- Stream project declaration -->
<STREAM_PROJECT>
    <!-- Stream schemata declarations -->
    <SCHEMA name="Light">
        <ATTRIBUTE name="light_value">
            <TYPE>integer</TYPE>
            <RANGE min="40" max="120" />
            <DISTRIBUTION>random</DISTRIBUTION>
        </ATTRIBUTE>
    </SCHEMA>
    <SCHEMA name="Temperature">
        <ATTRIBUTE name="temperature_value">
            <TYPE>integer</TYPE>
            <RANGE min="10" max="35" />
            <DISTRIBUTION>random</DISTRIBUTION>
        </ATTRIBUTE>
```

```
    </SCHEMA>
    <SCHEMA name="Pressure_Wind">
        <ATTRIBUTE name="pressure_value">
            <TYPE>integer</TYPE>
            <RANGE min="100" max="500" />
            <DISTRIBUTION>random</DISTRIBUTION>
        </ATTRIBUTE>
        <ATTRIBUTE name="wind_value">
            <TYPE>integer</TYPE>
            <RANGE min="0" max="10" />
            <DISTRIBUTION>random</DISTRIBUTION>
        </ATTRIBUTE>
    </SCHEMA>

    <!-- Stream sources that abide by the above defined schemata -->
    <STREAM_SOURCE type="Pressure_Wind" name="terrace1">
        <RATE type="second" value="2" />
    </STREAM_SOURCE>
    <STREAM_SOURCE type="Light" name="rear_window_light">
        <RATE type="minute" value="10" />
    </STREAM_SOURCE>
    <STREAM_SOURCE type="Light" name="front_window_light">
        <RATE type="minute" value="20" />
    </STREAM_SOURCE>
    <STREAM_SOURCE type="Temperature" name="living_room_temperature">
        <RATE type="hour" value="120" />
    </STREAM_SOURCE>
    <STREAM_SOURCE type="Temperature" name="kitchen_temperature">
        <RATE type="hour" value="240" />
    </STREAM_SOURCE>
    <STREAM_SOURCE type="Temperature" name="dining_room_temperature">
        <RATE type="hour" value="60" />
    </STREAM_SOURCE>
    <STREAM_SOURCE type="Pressure_Wind" name="terrace">
        <RATE type="second" value="10" />
    </STREAM_SOURCE>
</STREAM_PROJECT>
```

Μπορούμε να θεωρήσουμε ότι το παραπάνω παράδειγμα μοντελοποιεί ένα μικρό δίκτυο αισθητήρων που βρίσκεται εγκατεστημένο σε ένα κτήριο. Φυσικά, για τις ανάγκες του

παραδείγματος το δίκτυο αυτό είναι αρκετά μικρό αλλά είναι ενδεικτικό. Βλέπουμε, λοιπόν, ότι ορίζονται τριών ειδών σχήματα (SCHEMA) τα οποία στη συγκεκριμένη περίπτωση ορίζουν τριών ειδών αισθητήρες και αυτά είναι τα Light, Temperature και Pressure_Wind. Η σημασία τους ας θεωρήσουμε ότι είναι αυτονόητη από τα ονόματά τους. Το κάθε ένα από αυτά περιέχει έναν αριθμό από πεδία (ATTRIBUTE). Το Light περιέχει ένα πεδίο light_value ακέραιου τύπου και εύρους τιμών από 40 μέχρι 120 (οι τιμές είναι ενδεικτικές και όχι πραγματικές) και ορίζει ένα τύπο αισθητήρων φωτισμού. Αναλόγως και το Temperature. Το Pressure_Wind διαφέρει στο ότι περιέχει δύο πεδία, το pressure_value και το wind_value, οπότε θεωρούμε ότι αντιστοιχεί σε έναν αισθητήρα που μετρά ταυτόχρονα την πίεση και τον άνεμο. Στο project αυτό ορίζονται ακόμη επτά πηγές ροών (STREAM_SOURCE). Αυτές οι πηγές αντιστοιχούν σε αισθητήρες του project που θα βρίσκονται στο κτήριο. Έτσι, δηλώνονται δύο αισθητήρες φωτός, ο front_window_light και ο rear_window_light που θα εκπέμπουν ροές με ρυθμούς 10 και 20 πλειάδες το λεπτό αντίστοιχα. Ακόμη, δηλώνονται δύο αισθητήρες τύπου Pressure_Wind, ο terrace και ο terrace1, που παράγουν πλειάδες με ρυθμούς 2 και 10 πλειάδες το δευτερόλεπτο. Τέλος, δηλώνονται τρεις αισθητήρες θερμοκρασίας, ο living_room_temperature, ο kitchen_temperature και ο dining_room_temperature, που παράγουν πλειάδες με ρυθμούς 120, 240 και 60 το δευτερόλεπτο αντίστοιχα.

6.7 ΣΧΕΔΊΑΣΗ ΚΑΙ ΑΝΆΠΤΥΞΗ ΤΟΥ ΛΟΓΙΣΜΙΚΟΎ

6.7.1 ΤΡΌΠΟΣ ΣΧΕΔΊΑΣΗΣ

Το εργαλείο που αναπτύχθηκε στα πλαίσια του παρόντος εγχειρήματος ονομάστηκε κωδικά "Ηώς"[24] (Eos – Eos Originator of Streams). Έχει αναπτυχθεί στη γλώσσα προγραμματισμού Java με πλήρως αντικειμενοστραφή προσέγγιση. Η μεγάλη πρόκληση στην προσπάθεια αυτή ήταν να γίνει η όλη ανάλυση και ανάπτυξη με τρόπο καλά δομημένο έτσι ώστε να είναι εξελίξιμος στο μέλλον. Η παραγωγή δοκιμαστικών ροών δεδομένων όσον αφορά στην καλύτερη κατανόηση του πώς οι ροές παράγονται και αλληλεπιδρούν στάθηκε πραγματικά ένα ιδιαίτερα δύσκολο σημείο στη φάση αυτή. Είναι καλά γνωστό και έχει εντοπιστεί σε πολλά σημεία της παρούσας εργασίας πως οι εφαρμογές που εξαρτώνται από ροές δεδομένων είναι πολλές και ποικίλες.

Σε αυτό ακριβώς το σημείο αξίζει να σταθεί κανείς παραπάνω. Οι ιδιαιτερότητες που εμπεριέχουν οι ροές δεδομένων σε κάθε ξεχωριστό πεδίο εφαρμογής είναι τόσο ποικίλες και διαφορετικές που κάνουν μια γενικευμένη θεώρησή τους αρκετά δύσκολη. Σε αυτή την εργασία δεν απαντάται το

24 Η Ηώς είναι αρχαιοελληνική θεότητα. Ανήκει στους Τιτάνες και ήταν η θεότητα της αυγής η οποία καλούσε κάθε πρωί τον Ήλιο και άνοιγε για αυτόν τις πύλες του ουρανού, ώστε να τον διασχίσει με το άρμα του κατά τη διάρκεια της ημέρας.

ερώτημα αν αυτή η γενικευμένη θεώρηση είναι όντως δυνατή. Δε θα είχε ενδεχομένως και μεγάλη σημασία αυτό, καθώς ολοένα και νέες εφαρμογές πληροφοριακών συστημάτων που διαχειρίζονται ροές δεδομένων κάνουν την εμφάνισή τους. Αυτό που απαντάται είναι ότι σε πρώτο επίπεδο πολλές από τις ιδιότητές τους μπορούν να εντοπισθούν σε πολλές από αυτές και, επομένως να ομαδοποιηθούν.

Μερικές από αυτές τις ιδιότητες υπενθυμίζουμε ότι είναι στοιχεία όπως ο ρυθμός παραγωγής τους, και η υπόθεση ότι συζητούμε για ροές που υπακούν στο σχεσιακό μοντέλο και άρα διαθέτουν κάποιο δεδομένο σχήμα (schema). Ακόμη, έχει γίνει η προσπάθεια να μοντελοποιηθούν φαινόμενα όπως η εξάρτηση ροών από άλλες με την έννοια των εμφωλευμένων ροών και να αποδοθούν σημασιολογικά στοιχεία όπως αυτές οι υπό-ροές, οι κατανομές των τιμών των πεδίων των ροών και η δυνατότητα διαδοχής των πλειάδων τους. Αν και έχουν εντοπιστεί και περιγραφεί όλα αυτά εκτενώς παραπάνω ένα μόνο μέρος τους έχει υλοποιηθεί.

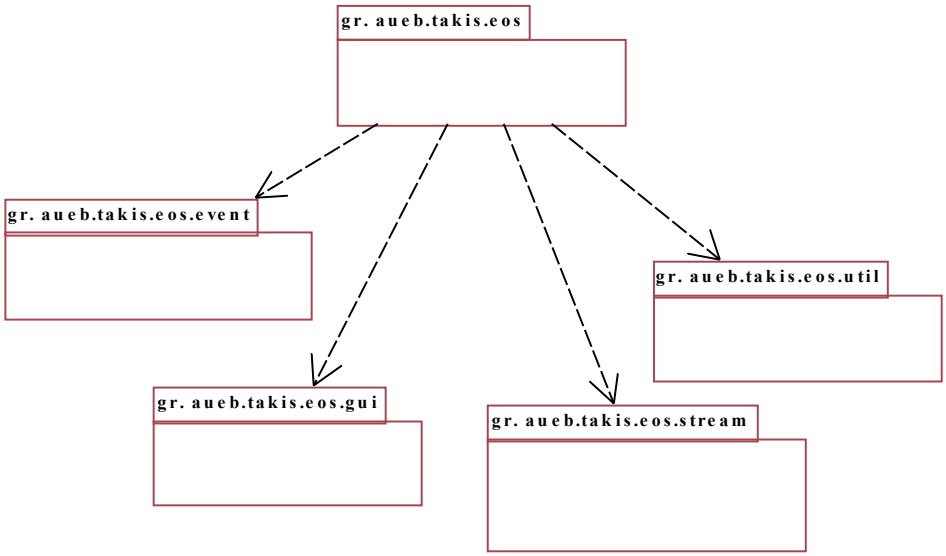
Η τρέχουσα υλοποίηση, λοιπόν, είναι μία πρωτογενής προσπάθεια στην κατεύθυνση που περιγράψαμε. Ταυτόχρονα, το εργαλείο που υλοποιήθηκε και θα περιγραφεί ως λογισμικό στην υπόλοιπη ενότητα διαθέτει ένα γραφικό περιβάλλον κατά το δυνατόν εύχρηστο για το σκοπό αυτό αλλά και σύνθετο στην υλοποίησή του. Η σχεδίασή του είναι τέτοια ώστε να μην δένει σφιχτά τα αρθρώματα (components) που σχετίζονται με λειτουργικά στοιχεία του με τα αρθρώματα που σχετίζονται με τις ίδιες τις ροές. Αυτό από μόνο του αποτελεί μία πρόσθετη πρόκληση σε σχεδιαστικό και προγραμματιστικό επίπεδο. Στη συνέχεια ακολουθεί αναφορά στην προσπάθεια αυτή και τα αποτελέσματά της. Η αναφορά αναπόφευκτα χρησιμοποιεί ορολογία που σχετίζεται με την ανάπτυξη λογισμικού και ιδιαίτερα της γλώσσας Java, η οποία είναι μια πλήρως αντικειμενοστραφής γλώσσα της οποίας ο εκτελέσιμος κώδικας μπορεί να εκτελεστεί σε πλήθος λειτουργικών συστημάτων χωρίς να χρειάζεται να υποστεί αλλαγές.

6.7.2 ΠΕΡΙΓΡΑΦΉ ΤΩΝ ΑΡΘΡΩΜΆΤΩΝ

Όπως μόλις αναφέρθηκε το εργαλείο έχει σχεδιαστεί αρθρωτά. Αυτό σημαίνει ότι ανεξάρτητα υπό-τμήματά του, καθένα από τα οποία είναι υπεύθυνο για διαφορετική λειτουργικότητα, είναι σχεδιασμένα και υλοποιημένα ξεχωριστά και ανεξάρτητα από τα υπόλοιπα. Τα αρθρώματα αυτά έχουν ομαδοποιηθεί σε ξεχωριστά πακέτα Java (Java packages) και φαίνονται στο Σχήμα 6.3. Αναλυτικά τα πακέτα αυτά είναι τα εξής:

- **gr.aueb.takis.eos**

Το κεντρικό πακέτο της υλοποίησης που περιέχει κλάσεις για την εκκίνηση του λογισμικού και ομαδοποιεί όλα τα υπόλοιπα πακέτα.



Σχήμα 6.3 – Τα Java packages του λογισμικού

- **gr.aueb.takis.eos.util**
  Περιέχει κλάσεις βοηθητικές για τη γενικότερη λειτουργικότητα του γραφικού περιβάλλοντος που δεν έχουν σχέση με την παραγωγή των ροών.

- **gr.aueb.takis.eos.gui**
  Περιέχει τις κλάσεις που απαρτίζουν το γραφικό περιβάλλον του λογισμικού.

- **gr.aueb.takis.eos.event**
  Περιέχει τις κλάσεις που σχετίζονται με τα συμβάντα που λαμβάνουν χώρα κατά τη λειτουργία του γραφικού περιβάλλοντος.

- **gr.aueb.takis.eos.stream**
  Περιέχει τις κλάσεις που υλοποιούν τις έννοιες των ροών δεδομένων και ό,τι άλλο υποβοηθητικό χρειάζονται αυτές.

Είναι φανερό πως το πακέτο που σχετίζεται με τις ίδιες τις ροές δεδομένων είναι πλήρως

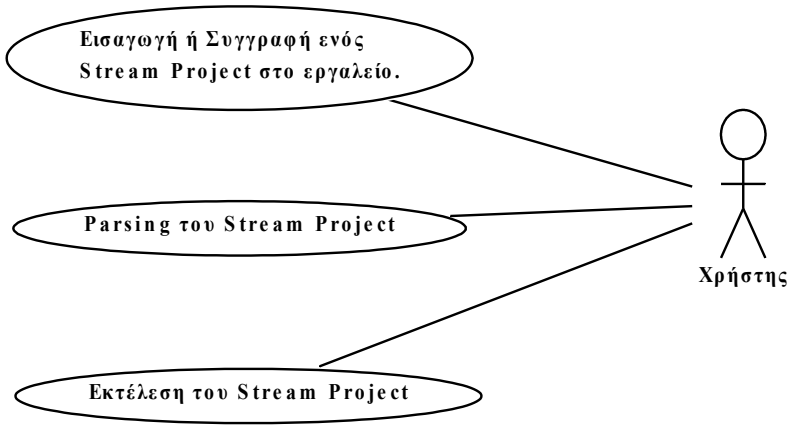
ανεξάρτητο από το υπόλοιπο λογισμικό. Με αυτό τον τρόπο τα υπόλοιπα τμήματα του λογισμικού έχουν σχεδιαστεί προσεκτικά ώστε να χρησιμοποιούν το πακέτο αυτό σαν ένα ξεχωριστό τμήμα το οποίο εισάγουν (import) στη λειτουργικότητά τους. Με τον ίδιο τρόπο το πακέτο αυτό είναι έτσι σχεδιασμένο ώστε να μπορεί να χρησιμοποιηθεί ως έχει και από άλλα λογισμικά. Αυτό επιτυγχάνεται μέσα από την καλή εκμετάλλευση των διεπαφών (interfaces) του αντικειμενοστραφούς προγραμματισμού και της Java. Για την καλύτερη κατανόηση των συσχετίσεων ανάμεσα στα πακέτα και τις κλάσεις έχει γραφτεί και το συνοδευτικό javadoc API (τεκμηρίωση του κώδικα ειδική για τη γλώσσα Java). Στο API αυτό γίνεται φανερό πως η αντικειμενοστραφής προσέγγιση των κλάσεων που απαρτίζουν το πακέτο των ροών δίνει τη δυνατότητα το πακέτο αυτό να εμπλουτιστεί ή να τροποποιηθεί μελλοντικά ώστε να συμπεριλαμβάνει για παράδειγμα περισσότερων ειδών πεδία και συνδυασμών ροών.

6.7.3 ΠΑΡΟΥΣΙΑΣΗ ΤΟΥ ΛΟΓΙΣΜΙΚΟΎ

Το ίδιο το εργαλείο είναι ένα γραφικό περιβάλλον στο οποίο η όλη διαδικασία της δήλωσης και παραγωγής ροών γίνεται με φιλικό τρόπο. Η φιλοσοφία του γραφικού περιβάλλοντος ακολουθεί τη λογική της διαχείρισης ενός project των προγραμματιστικών IDE's (Integrated Development Environment). Με αυτό τον τρόπο ο διαθέσιμος χώρος χωρίζεται σε τρία κύρια τμήματα:

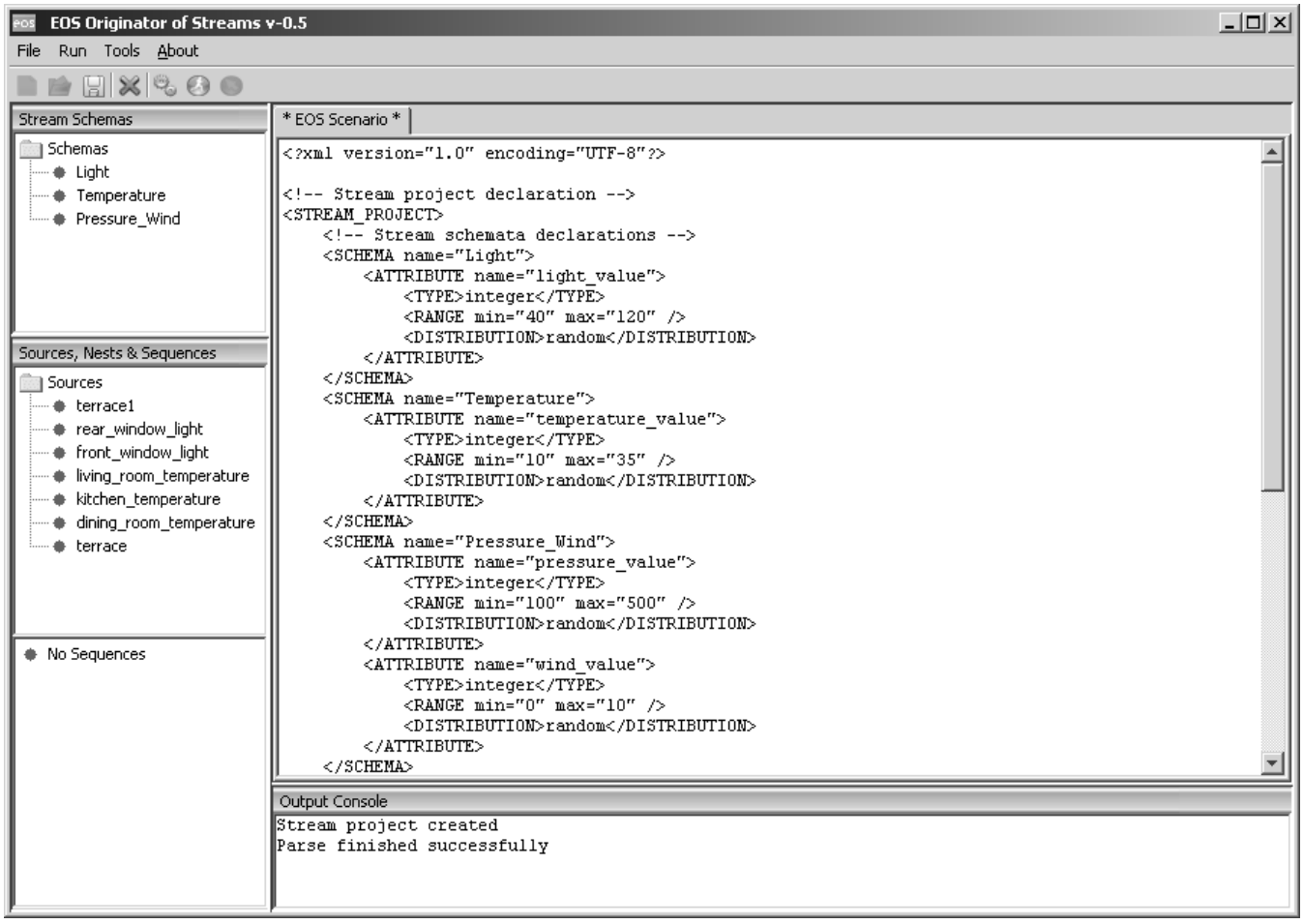
- Το τμήμα όπου ο χρήστης επεξεργάζεται το project.
- Το τμήμα όπου εμφανίζονται τα παραγόμενα στοιχεία που ο χρήστης ορίζει.
- Το τμήμα όπου εμφανίζονται διαγνωστικά μηνύματα.

Στο Σχήμα 6.4 φαίνονται τα κυρίαρχα use-cases αλληλεπίδρασης του χρήστη με το λογισμικό.



Σχήμα 6.4 – Κυρίαρχα use-cases

Στο Σχήμα 6.5 φαίνεται η απεικόνιση του γραφικού περιβάλλοντος του λογισμικού.



Σχήμα 6.5 – Γενικό περιβάλλον εργαλείου

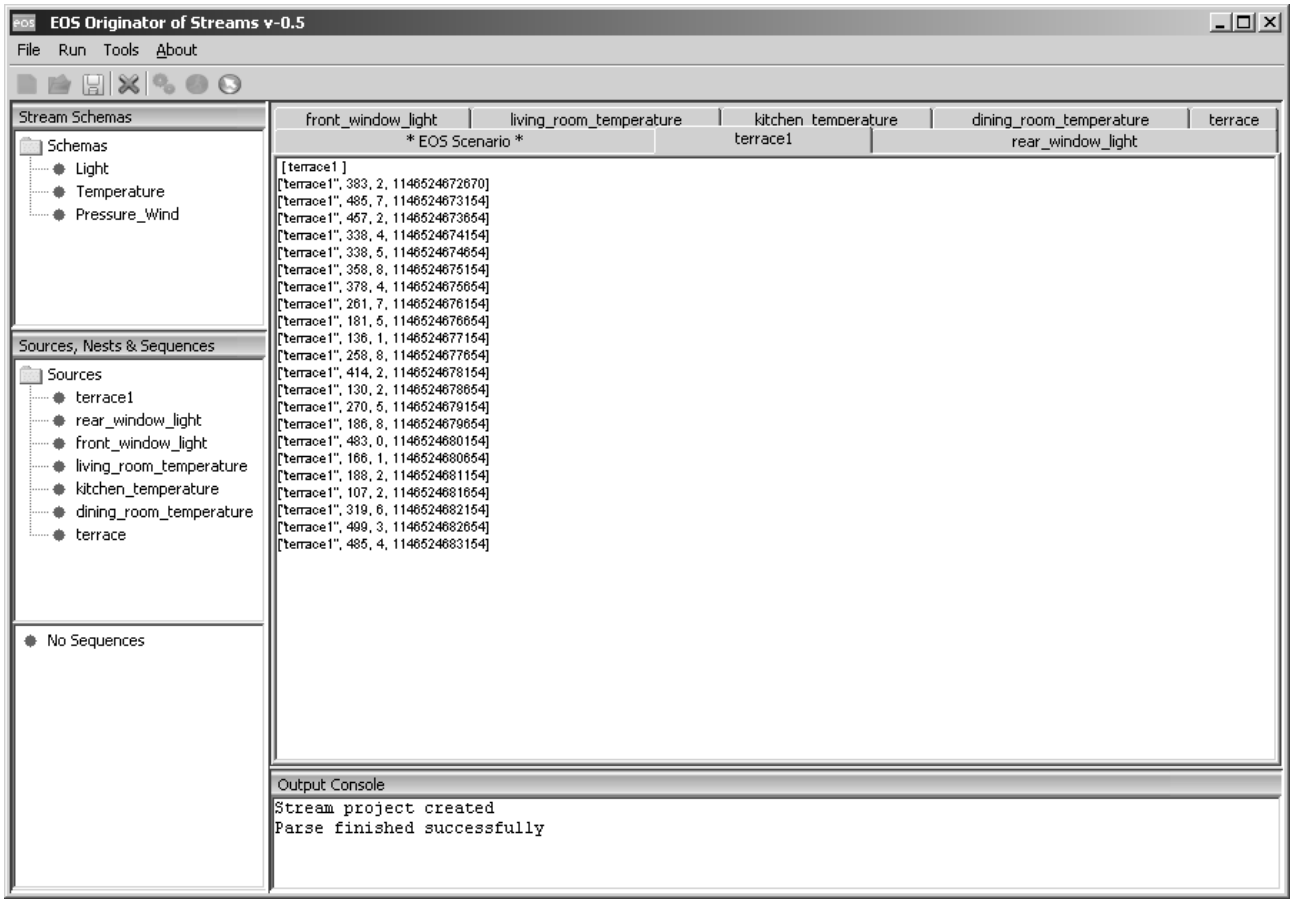
Στο γραφικό περιβάλλον του λογισμικού διακρίνονται το κεντρικό μενού και η κεντρική μπάρα εργαλείων πάνω πάνω, η κύρια περιοχή όπου γίνεται η συγγραφή του stream project σε XML, η

κονσόλα εκπομπής διαγνωστικών μηνυμάτων κάτω από την κύρια περιοχή και το τμήμα απεικόνισης των αντικειμένων που ορίζει ο χρήστης στα αριστερά..

Οι λειτουργίες που το εργαλείο δίνει στο χρήστη είναι οι εξής:

- Δημιουργία ενός νέου stream project (New)
- Άνοιγμα ενός υπάρχοντος stream project (Open)
- Αποθήκευση του τρέχοντος project εφόσον έχει γίνει κάποια επεξεργασία (Save)
- Κλείσιμο του τρέχοντος project (Close)
- Ανάλυση και αποκωδικοποίηση του τρέχοντος project (Parse)
- Εκτέλεση του τρέχοντος project (Run)
- Διακοπή της εκτέλεσης του τρέχοντος project εφόσον αυτό εκτελείται (Stop)

Κατά τη διαδικασία της εκτέλεσης εμφανίζεται μία ξεχωριστή καρτέλα για κάθε μία από τις δηλωμένες από το χρήστη ροές για να είναι δυνατή η παρακολούθηση της εξέλιξής της. Ταυτόχρονα το εργαλείο ανοίγει μία θύρα (port, την 8888) στην οποία ακούει για clients που ενδιαφέρονται για τις ροές που παράγει. Όποιος client συνδεθεί στη θύρα αυτή δέχεται άμεσα τις παραγόμενες ροές, οι οποίες συγχρονίζονται έτσι ώστε οι πλειάδες τους να μεταφέρονται ακέραιες. Στο Σχήμα 6.6 φαίνεται και η λειτουργία αυτή.



Σχήμα 6.6 – Εκτέλεση του stream project

6.8 ΑΠΟΛΟΓΙΣΜΌΣ ΥΛΟΠΟΊΗΣΗΣ

Το εργαλείο που περιγράφηκε παραπάνω αποτελεί μια πρωτογενή υλοποίηση ενός παραγωγού ροών (stream generator) που στηρίζεται σε μια γενικευμένη προσέγγιση της παραγωγής ροών δεδομένων με αφορμή τα όσα παρατηρήθηκαν στο προηγούμενο κεφάλαιο σχετικά με τα υπάρχοντα εργαλεία δοκιμασίας ΣΔΡΔ (DSMS benchmarks) και τα είδη των ροών δεδομένων που κυρίαρχα εντοπίζονται στη σύγχρονη βιβλιογραφία αλλά και εφηρμοσμένη έρευνα.

Η συγκεκριμένη υλοποίηση δεν έχει ολοκληρώσει όλη την αναφερθείσα λειτουργικότητα στην παρούσα φάση. Αυτό που αρχικά έχει γίνει είναι η μελέτη των ειδών ροών δεδομένων με τις οποίες καταπιάνεται και η μοντελοποίησή τους στο σχετικό λογισμικό. Έχουν, επομένως, καταγραφεί τα τρία είδη ροών που προαναφέραμε, οι απλές ροές, οι υπό-ροές και οι διαδοχές ροών. Από αυτές έχουν υλοποιηθεί μέχρι στιγμής πλήρως οι απλές ροές τόσο όσον αφορά τη δήλωση όσο και την παραγωγή τους. Για τα δύο άλλα είδη ροών έχουν υλοποιηθεί μόνον οι δηλώσεις τους και η

απεικόνισή τους στο γραφικό περιβάλλον του εργαλείου. Οι δηλώσεις, όπως περιγράφηκε, έχουν γίνει σε XML, η οποία διαβάζεται από το εργαλείο, ελέγχεται για το αν οι οντότητες που περιγράφει είναι έγκυρες, σύμφωνα με τη σημασιολογία που έχει διατυπωθεί, και ακολούθως το δοσμένο σενάριο εκτελείται.

Σημαντική εργασία έχει γίνει στο ίδιο το εργαλείο το οποίο είναι γραφικού περιβάλλοντος με όλες τις δυσκολίες που αυτό εμπεριέχει. Οι κρίσιμες λειτουργίες της παραγωγής ροών εκτελούνται σε ξεχωριστά νήματα (threads) έτσι ώστε να είναι ανεξάρτητες η μία πηγή ροών από την άλλη και όλες μαζί από το υπόλοιπο λογισμικό. Ακόμη, το εργαλείο υλοποιεί και πολλές από τις λειτουργικότητες που περιμένει κανείς από ένα τέτοιου είδους εργαλείο, όπως το άνοιγμα/κλείσιμο/αποθήκευση αλλά και η επεξεργασία των αρχείων του και η διαχείριση του σεναρίου που ο χρήστης θέτει προς ανάλυση και εκτέλεση. Το εργαλείο είναι έτσι σχεδιασμένο ώστε τα ανεξάρτητα τμήματά του να είναι οργανωμένα σε ξεχωριστά αρθρώματα και να λειτουργούν ανεξάρτητα. Ακόμη, το περιβάλλον επιχειρήθηκε να είναι όσο το δυνατόν πιο απλό και φιλικό χωρίς να θυσιάζει το σκοπό για τον οποίο δημιουργήθηκε.

Συνεχίζοντας κανείς την υλοποίηση του λογισμικού αυτού θα έβαζε ως πρώτο στόχο την ολοκλήρωση της παραγωγής των υπό-ροών και των διαδοχών ροών που δεν έχουν ακόμη υλοποιηθεί. Είναι δυνατόν οι ροές αυτές να υλοποιηθούν βασιζόμενοι πάνω στο ίδιο το API (Application Programming Interface) που έχει μέχρι στιγμής παραδοθεί, μιας και αυτό έχει σχεδιαστεί ώστε να είναι κατά το δυνατόν επεκτάσιμο. Ακόμη, χρήσιμη θα ήταν και η διατύπωση ενός XSD Schema, ενός XML αρχείου, δηλαδή, που εξυπηρετεί σε θεμελιώδεις λειτουργίες επικύρωσης XML εγγράφων. Κάτι τέτοιο στην παρούσα φάση κρίθηκε πρώιμο, μιας και περισσότερο κρίσιμες λειτουργικότητες του ίδιου του εργαλείου ως λογισμικό έπρεπε να υλοποιηθούν κατά προτεραιότητα.

6.9 ΕΝΔΙΑΦΕΡΟΝΤΑ ΣΗΜΕΙΑ ΥΛΟΠΟΊΗΣΗΣ

Η όλη προσέγγιση που έχουμε ως τώρα περιγράψει βασίζεται στις παρατηρήσεις που έχουν εντοπιστεί όσον αφορά μερικά από τα σημαντικότερα είδη ροών που εμφανίζονται σε εφαρμογές στις μέρες μας αλλά και τη μελέτη των δύο εργαλείων δοκιμασίας (benchmarking) Συστημάτων Διαχείρισης Ροών Δεδομένων. Παρατηρήσαμε ότι οι εφαρμογές που παράγουν και διαχειρίζονται δεδομένα σε μορφές ροών είναι ποικίλες και προέρχονται από πληθώρα ερευνητικών και εφηρμοσμένων πεδίων. Θα εξετάσουμε σε αυτή την παράγραφο τα διάφορα σημεία που στάθηκαν ως προκλήσεις στο εγχείρημα αυτό.

Καταρχήν, εντοπίσαμε ένα πολύ σημαντικό και σύγχρονο πεδίο εφαρμογής, αυτό των δικτύων αισθητήρων (sensor networks), όπου οι ροές δεδομένων χρήζουν άμεσης αντιμετώπισης, και το οποίο τεχνολογικά αναπτύσσεται ραγδαία παρέχοντας ολοένα και περισσότερες δυνατότητες αλλά και δημιουργώντας περισσότερα ζητήματα προς επίλυση. Στη συνέχεια, ένα ακόμη πεδίο εμφάνισης ροών ήταν αυτό των δικτύων δεδομένων (data networks) με τις δικές του ξεχωριστές ιδιαιτερότητες, όπως το γεγονός ότι η ωφέλιμη πληροφορία για ένα ΣΔΡΔ σε αυτό τον τομέα είναι η παρακολούθηση των IP επικεφαλίδων και όχι ολόκληρα τα IP πακέτα. Ακόμη, οι "εμφωλευμένες" ροές, ή υπό-ροές, όπως τις ονομάσαμε, που είναι ροές που εμπεριέχουν στη σημασιολογία τους έννοιες όπως αυτή της συνόδου (session), οι ροές που παράγονται από συμβάντα εφαρμογών διαδικτύου (web logs) καθώς και οι ροές με δεδομένα οικονομικής προέλευσης.

Είναι φανερό ότι η **ετερογένεια των ίδιων των δεδομένων** που μεταφέρονται σε κάθε μία από τις προαναφερθείσες περιπτώσεις είναι μεγάλη. Αυτό είναι ένας πρώτος λόγος που κάνει την καθολική προσέγγιση της παραγωγής αυτών των ροών δύσκολη υπόθεση τουλάχιστον σε λειτουργικό επίπεδο. Οι ροές αυτές μπορούν να περιέχουν απλά σχεσιακά δεδομένα, δηλαδή σχεσιακές πλειάδες που ακολουθούν κάποιο σχήμα (schema), αλλά και δεδομένα IP των οποίων οι επικεφαλίδες μπορούν να αντιμετωπιστούν ως διακριτές πλειάδες αλλά το σώμα των οποίων αποτελείται από κάθε είδους δεδομένα που εξαρτώνται από την εκάστοτε εφαρμογή. Ακόμη, σε πολλές από αυτές τις περιπτώσεις τα δεδομένα έχουν μορφή XML, όπως για παράδειγμα τα δεδομένα οικονομικής προέλευσης – άλλωστε, η XML είναι έτσι κι αλλιώς αρκετά δημοφιλής τρόπος απεικόνισης δομημένης πληροφορίας και μπορεί να εφαρμοστεί σε ποικιλία εφαρμογών.

Ένα πολύ σημαντικό σημείο που αποτελεί ανεξάρτητα τεράστια πρόκληση είναι η **σημασιολογία των ροών.** Κάθε εφαρμογή παράγει δεδομένα με διαφορετικό τρόπο και για διαφορετικό σκοπό από τις υπόλοιπες. Το νόημα καθεαυτό που κουβαλά το κάθε κομμάτι πληροφορίας είναι αυτό που κρίνεται δύσκολο να μοντελοποιηθεί και να αναπαραχθεί υπολογιστικά. Μερικές μόνο από τις διαφορετικές σημασιολογίες που έχουμε εντοπίσει είναι, για παράδειγμα, οι εξής:

- Αυτές που αφορούν στα δίκτυα αισθητήρων. Οι αισθητήρες καταγράφουν συνήθως τις τιμές φυσικών μεγεθών. Στην απλή μορφή τους είναι εξειδικευμένοι σε ένα φυσικό μέγεθος, πχ. μέτρηση θερμοκρασίας, ενώ μπορεί να είναι και περισσότερο σύνθετοι μετρώντας πολλά μεγέθη ταυτόχρονα. Ακόμη, οι μετρήσεις που καταγράφουν είναι ανεξάρτητες μεταξύ τους, μιας και η αλληλεπίδραση μεταξύ αισθητήρων ανάλογα με τις

συνθήκες είναι δυνατότητα που μόλις στις μέρες μας συζητείται από πλευράς τεχνολογίας και πρωτοκόλλων.

- Στα IP δίκτυα οι συσχετίσεις των πακέτων της διακινούμενης πληροφορίας και τα μεγέθη που παρατηρούνται όσον αφορά τους ρυθμούς μεταγωγής εξαρτώνται ισχυρά από τη φύση της κάθε εφαρμογής αλλά και από τη χρονική στιγμή κατά την οποία παρατηρούνται.

- Ακόμη, σε πολλές εφαρμογές παρατηρήσαμε την ύπαρξη της έννοιας της συνόδου (session). Η σύνοδος ουσιαστικά ορίζει ένα σύνολο από – έστω και μόνο μία – υπό-ροές οι οποίες έχουν ένα κοινό σημείο αφετηρίας. Αυτή συναντάται σε πληθώρα εφαρμογών, όπως τα αρχεία καταγραφής εφαρμογών διαδικτύου (web logs), όπου οι πράξεις των χρηστών καταγράφονται κατά τη διάρκεια της συνόδου αλληλεπίδρασής τους με την εφαρμογή.

- Ακόμη, εξειδικευμένες περιπτώσεις που συναντήθηκαν είναι αυτή των εργαλείων δοκιμασίας ΣΔΡΔ “Linear Road” και “NEXMark”, όπου το πρώτο μοντελοποιεί την κυκλοφορία και το σύστημα διοδίων σε μια μεγάλη πόλη, ενώ το δεύτερο μια on-line δημοπρασία.

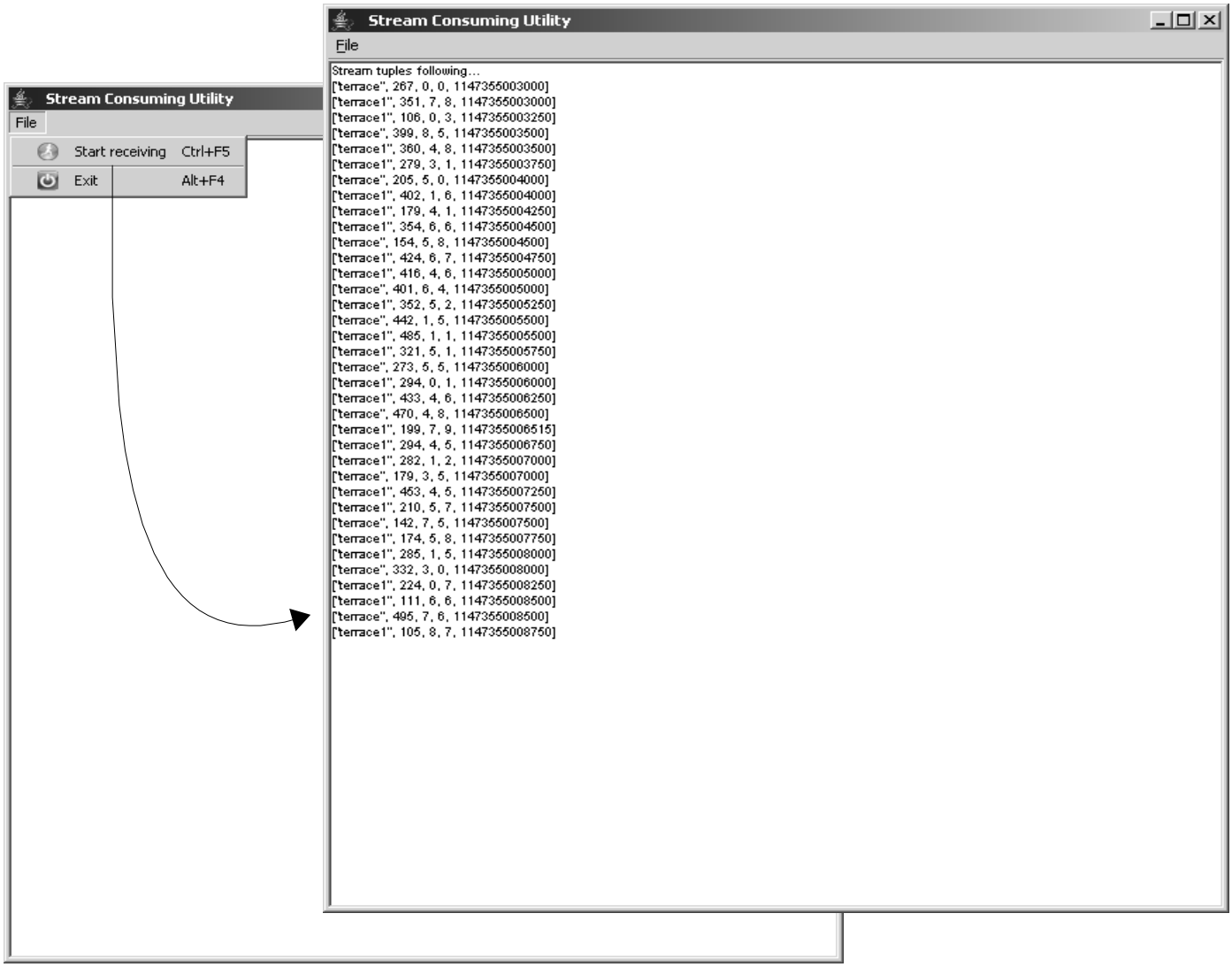
Οφείλουμε να παρατηρήσουμε ότι ενώ το ερευνητικό πεδίο των ροών δεδομένων είναι ιδιαίτερα πλούσιο σε σημασιολογικά στοιχεία, τα δύο προαναφερθέντα εργαλεία επέλεξαν να υλοποιήσουν πολύ συγκεκριμένα στοιχεία αυτών, κάτι που φαίνεται και από την κριτική που τους έγινε στα αντίστοιχα κεφάλαια. Η επιλογή αυτή είναι ενδεικτική της δυσκολίας να μοντελοποιηθεί η σημασιολογία των ροών. Στην παρούσα εργασία προτιμήθηκε μια αρκετά πιο γενικόλογη προσέγγιση, η οποία φτάνει μόνο να αγγίξει σε πρακτικό επίπεδο το ζήτημα αυτό.

Η προσπάθεια που έγινε είναι η μελέτη σε βιβλιογραφικό επίπεδο του χώρου των ροών δεδομένων, της παραγωγής ροών, των Συστημάτων Διαχείρισης ροών Δεδομένων και των εργαλείων δοκιμασίας αυτών. Εφόσον, αναγνωρίστηκαν τα ζητήματα που ανακύπτουν στο χώρο αυτό έγινε προσπάθεια πολλά από τα κοινά στοιχεία που παρουσιάζουν οι ροές σε κάθε σχεδόν πεδίο εφαρμογής τους να ομαδοποιηθούν και να αποτυπωθούν σε ένα εργαλείο παραγωγής δοκιμαστικών ροών. Η διαφορά του εργαλείου αυτού και της προσέγγισής του από παρόμοια εγχειρήματα στα οποία αναφερθήκαμε είναι ότι προσπαθεί να αντιμετωπίσει τις ροές με γενικό τρόπο και όχι να παράξει ροές με περιορισμένα σημασιολογικά στοιχεία. Η προσπάθεια δεν έχει ακόμη

ολοκληρωθεί σε αναπτυξιακό επίπεδο και οπωσδήποτε είναι πάντοτε ανοικτή σε περαιτέρω πρόοδο καθώς ο χώρος στον οποίο ανήκει είναι ολοένα και περισσότερο εξελισσόμενος.

6.10 ΕΡΓΑΛΕΊΟ ΚΑΤΑΝΆΛΩΣΗΣ ΡΟΏΝ

Σαν συμπλήρωμα του εργαλείου παραγωγής ροών που περιγράψαμε, υλοποιήθηκε και ένα απλό χρηστικό εργαλείο (utility) που να επιτρέπει την παρακολούθηση των παραγόμενων ροών, η οποία είναι έτσι κι αλλιώς εφικτή και μέσα από το ίδιο το περιβάλλον του κεντρικού εργαλείου. Στην



Σχήμα 6.7 – Utility κατανάλωσης ροών

περίπτωση αυτή πρόκειται για ένα utility που επιτρέπει δικτυακά την παρακολούθηση των ροών που το εργαλείο παραγωγής εκπέμπει, εφόσον δοθούν η IP διεύθυνση του μηχανήματος όπου αυτό εκτελείται και η θύρα στην οποία "ακούει". Στο Σχήμα 6.7 φαίνεται το εργαλείο αυτό.

Το εργαλείο αυτό λειτουργεί ως εξής: Αφότου εκκινηθεί και εμφανιστεί το γραφικό περιβάλλον ο

χρήστης επιλέγει από το κεντρικό μενού την επιλογή “Start Receiving”. Έπειτα εμφανίζεται ένας διάλογος όπου ο χρήστης καλείται να εισάγει την IP του server όπου εκτελείται ο stream generator, αλλά και τη θύρα στην οποία αυτός είναι συνδεδεμένος. Αν τα στοιχεία που εισάγει ο χρήστης είναι σωστά τότε στο κύριο τμήμα του εργαλείου αυτού θα εμφανιστούν οι παραγόμενες από το stream generator ροές. Φυσικά τα δύο αυτά λογισμικά δεν είναι απαραίτητο να βρίσκονται στο ίδιο φυσικό μηχάνημα μιας και η επικοινωνία τους είναι δικτυακή πάνω από TCP/IP. Σημειώνεται, τέλος, ότι το εργαλείο αυτό της κατανάλωσης των ροών βρίσκεται σε ένα επιπλέον ξεχωριστό πακέτο (Java package) το οποίο είναι ανεξάρτητο από τα υπόλοιπα πακέτα του stream generator αλλά έχει ενσωματωθεί στο ίδιο jar (Java εκτελέσιμο αρχείο) με τα υπόλοιπα.

Είδαμε, λοιπόν, σε αυτό το κεφάλαιο την ολοκλήρωση της ανάλυσης τη σχεδίαση και την ανάπτυξη του εργαλείου παραγωγής ροών δεδομένων. Το εργαλείο αυτό μέσα από δηλώσεις που προσδιορίζονται σε γλώσσα XML καθιστά δυνατό τον ορισμό και την παραγωγή ροών μέσα από ένα γραφικό περιβάλλον. Είδαμε, ακόμη, τις προκλήσεις και τις δυσκολίες που αυτό το εγχείρημα περιείχε, στην προσπάθεια για ανάπτυξη μιας μεθοδολογίας που να αντιμετωπίζει με γενικευμένο τρόπο την παραγωγή ροών δεδομένων. Αντίθετα με άλλες προσεγγίσεις, όπως αυτές του “Linear Road” ή του “NEXMark” καταβλήθηκε προσπάθεια η προσέγγιση αυτή να είναι σχεδιασμένη με όσο το δυνατόν περισσότερο δομημένο και άρα εξελίξιμο τρόπο. Οπωσδήποτε, μεγάλος χρόνος έρευνας απομένει ακόμη για την πλήρη περάτωση μιας τέτοιας προσπάθειας.

7. ΕΠΙΛΟΓΟΣ

Στα πλαίσια της παρούσας εργασίας ασχοληθήκαμε με τη μελέτη Συστημάτων Διαχείρισης Ροών Δεδομένων (Data Stream Management Systems). Το ερευνητικό πεδίο αυτό ανήκει εννοιολογικά στο χώρο της διαχείρισης βάσεων δεδομένων αλλά εξειδικεύεται στη διαχείριση δεδομένων που εμφανίζονται με τη μορφή ροών, σε αντίθεση με τα παραδοσιακά ΣΔΒΔ τα οποία διαχειρίζονται, ως επί το πλείστον, στατικά δεδομένα. Οι ροές δεδομένων προκαλούν πληθώρα νέων ζητημάτων στα συστήματα που τις αξιοποιούν εξαιτίας ιδιαιτεροτήτων τους όπως οι μεγάλοι ρυθμοί άφιξης των δεδομένων. Όλες αυτές οι έννοιες παρουσιάστηκαν με σκοπό να καταγραφεί ένα ευρύ φάσμα γύρω από τα κομβικά σημεία της διαχείρισης των ροών, με όσο το δυνατόν πιο ολοκληρωμένο τρόπο.

Κατόπιν, περιγράφηκαν οι συνεχείς επερωτήσεις οι οποίες τίθενται πάνω σε ροές δεδομένων και οι δυσκολίες που αντιμετωπίζονται κατά την εκτέλεση αυτών. Η σύγχρονη έρευνα έχει να παρουσιάσει πληθώρα μεθοδολογιών και τεχνικών που αποσκοπούν στην αποδοτικότερη επεξεργασία τους. Έτσι, έχουν αναπτυχθεί αλγόριθμοι για την βέλτιστη διαχείριση των πόρων του υπολογιστικού συστήματος του ΣΔΡΔ, όπως η ανακατανομή των επερωτήσεων, η προσαρμοστική εκτέλεση των επερωτήσεων, η ελάττωση φόρτου, η προσεγγιστική εκτέλεση και η χρήση στατιστικών πληροφοριών. Είναι φανερό ότι παρόλα τα ενδιαφέροντα αποτελέσματα που κατά καιρούς εμφανίζονται στη βιβλιογραφία, είναι τέτοια η ποικιλία των προβλημάτων που αντιμετωπίζονται ώστε ακόμη να μην έχει υπάρξει μια ενιαία και ολοκληρωμένη προσέγγιση. Παρόλα αυτά, έχουν προταθεί διάφορα ΣΔΡΔ σε ερευνητικό επίπεδο, μερικά από τα οποία παρατίθενται και εδώ.

Σε βιβλιογραφικό επίπεδο, στην παρούσα εργασία, έχει γίνει ευρεία έρευνα του χώρου για την οποία καταβλήθηκε προσπάθεια να αποκρυσταλλωθεί σε ένα συνεκτικό κείμενο. Με τα δεδομένα αυτά αναγνωρίστηκε η έλλειψη για ένα εργαλείο παραγωγής δοκιμαστικών ροών δεδομένων που να τις προσεγγίζει με τρόπο γενικευμένο. Ενώ υπάρχουν εργαλεία δοκιμασίας (benchmarking), αυτά δεν δίνουν τη δυνατότητα ορισμού σεναρίων εκτέλεσης διαφορετικών από αυτά που υποστηρίζουν. Για αυτό τον λόγο, επιλέχθηκε να γίνει μια πιο γενικευμένη θεώρηση της παραγωγής των ροών με σκοπό την κατασκευή ενός εργαλείου που να δίνει την κατά το δυνατόν μεγαλύτερη ευελιξία δηλώσεων. Ο τρόπος με τον οποίο προσεγγίστηκε η ανάπτυξη του εργαλείου αυτού είναι τέτοιος ώστε αυτό να είναι καλά σχεδιασμένο και να έχει δυνατότητες εξέλιξης.

Η εργασία αυτή φιλοδοξεί να αποτελέσει μια μικρή συνδρομή στην καλύτερη κατανόηση του

επιστημονικού πεδίου των ροών δεδομένων. Το πεδίο αυτό είναι βέβαιο πως θα απασχολήσει την ερευνητική κοινότητα για πολύ καιρό ακόμη μιας και οι εφαρμογές του συνεχώς αυξάνονται σε πλήθος και χρησιμότητα, προσδίδοντάς του ολοένα και περισσότερη αξία.

ΑΝΑΦΟΡΈΣ

[1] "Flexible Time Management in Data Stream Systems"
Utkarsh Srivastava, Jennifer Widom, Stanford University (2004)

[2] "Real-Time Databases"
Krithi Ramamritha, Dpt of Computer and Information Science, University of Massachussets (1996)

[3] "Introduction to Stream Data Management"
Nauman A. Chaudri, Dpt of Computer Science, University of New Orleans (2000)

[4] "Issues in Data Stream Management"
Lukasz Golab, M. Tamer Ozsu, University of Waterloo, Canada (2003)

[5] "Continuous Queries over Data Streams"
Shivnath Babu, Jennifer Widom, Stanford University (2001)

[6] "Operator Scheduling in Data Stream Systems"
Brian Babcock, Shivnath Babuy, Mayur Datarz, Rajeev Motwanix, Dilys Thomas, Stanford University (2003)

[7] "On Demand Classification of Data Streams"
Charu C. Aggarwal IBM T. J. Watson Research Center, Jiawei Han, Jianyong Wang UIUC, Philip S. Yu IBM T. J. Watson Research Center (2004)

[8] "Wide-Area Traffic: The Failure of Poisson Modeling"
Vern Paxson and Sally Floyd, Lawrence Berkeley Laboratory and EECS Division, University of California, Berkeley (1995)

[9] "Retrospective on Aurora"
Hari Balakrishnan, Magdalena Balazinska, Don Carney, Ugur Cetintemel, Mitch Cherniack, Christian Convey, Eddie Galvez, Jon Salz, Michael Stonebraker, Nesime Tatbul, Richard Tibbetts, Stan Zdonik, Brown University, MIT Cambridge, Brandeis University

[10] "Stream Data Management"
Nauman Chaudri, University of New Orleans, Kevin Shaw, Naval Research Lab, Mahdi Abdelguerfi, University of New Orleans (2005)

[11] "Monitoring Streams – A New Class of Data Management Applications"
Don Carney, Ugur Çetintemel, Mitch Cherniack, Christian Convey, Sangdon Lee, Greg Seidman, Michael Stonebraker, Nesime Tatbul, Stan Zdonik, Brown University, MIT Cambridge, Brandeis University (2002)

[12] "The Design of the Borealis Stream Processing Engine"

Daniel J. Abadi, Yanif Ahmad, Magdalena Balazinska, Ugur Cetintemel, Mitch Cherniack, Jeong-Hyon Hwang, Wolfgang Lindner, Anurag S. Maskey, Alexander Rasin, Esther Ryvkina, Nesime Tatbul, Ying Xing, Stan Zdonik, Brown University, MIT Cambridge, Brandeis University (2005)

[13] "Advances in Real-Time Database Research"

Azer Bestavros, Boston University (1995)

[14] "Models and Issues in Data Stream Systems"

Brian Babcock, Shivnath Babu, Mayur Datar, Rajeev Motwani, Jennifer Widom, Department of Computer Science Stanford University

[15] "Computing on data streams"

Monika Rauch Henzinger, Prabhakar Raghavan, Sridar Rajagopalan (1998)

[16] "An Abstract Semantics and Concrete Language for Continuous Queries over Streams and Relations"

Arvind Arasu, Shivnath Babu, Jennifer Widom, Stanford University

[17] "Query Execution and Optimization"

Stratis Viglas, School of Informatics, University of Edinburgh (2005)

[18] "The Virtues and Challenges of Ad Hoc + Streams Querying in Finance"

Alberto Lerner, Ecole Nationale Superieure de Telecommunications, Paris, Dennis Shasha, New York University, New York (2003)

[19] "SQL:1999, formerly known as SQL3"

Andrew Eisenberg, Sybase, Jim Melton, Sandy UT

[20] "STREAM: The Stanford Data Stream Management System"

Arvind Arasu, Brian Babcock, Shivnath Babu, John Cieslewicz, Mayur Datar, Keith Ito, Rajeev Motwani, Utkarsh Srivastava and Jennifer Widom, Department of Computer Science, Stanford University

[21] "Scalable Distributed Stream Processing"

Mitch Cherniack, Hari Balakrishnan, Magdalena Balazinska, Don Carney, Ugur Cetintemel, Ying Xing and Stan Zdonik, Brandeis University, Brown University, M.I.T. (2003)

[22] "A Model for Pipelined Query Execution"

Annita Wilschut, Stephan A. van Gils, University of Twente, Netherlands (1995)

[23] "Evaluating Window Joins over Unbounded Streams"

Jaewoo Kang, Jeffrey F. Naughton, Stratis D. Viglas, University of Wisconsin-Madison

[24] "Xjoin: Getting Fast Answers from Slow and Bursty Nerworks"

Tolga Urhan, Michael J. Franklin, Institute for Advanced Computer Studies Computer Science Department University of Maryland (1999)

[25] "XJoin: A Reactively-Scheduled Pipelined Join Operator"

Tolga Urhan, University of Maryland, Michael J. Franklin, University of California, Berkeley (2000)

[26] "Scrambling Query Plans to Cope With Unexpected Delays"

Laurent Amsaleg, Michael J. Franklin, Tolga Urhan, University of Maryland, Anthony Tomasic, Institut National de Recherche en Informatique

[27] "Maximizing the Output Rate of Multi-Way Join Queries over Streaming Information Sources"

Stratis D. Viglas, Jeffrey F. Naughton, Josef Burger, University of Wisconsin (2003)

[28] "MJoin: A Metadata Aware Stream Join Operator"

Luping Ding, Elke A. Rundensteiner, George T. Heineman, Worcester Polytechnic Institute

[29] "Query Processing, Approximation, and Resource Management in a Data Stream Management System"

Rajeev Motwani, Jennifer Widom, Arvind Arasu, Brian Babcock, Shivnath Babu, Mayur Datar, Gurmeet Manku, Chris Olston, Justin Rosenstein, Rohit Varma, Stanford University

[30] "Adaptive Query Processing in the Looking Glass"

Shivnath Babu, Stanford University, Pedro Bizarro, University of Wisconsin (2005)

[31] "An Adaptive Query Execution System for Data Integration"

Zachary G. Ives, Marc Friedman, Alon Levy, Daniel S. Weld, University of Washington, Daniela Florescu, INRIA Roquencourt

[32] "Rate-Based Query Optimization for Streaming Information Sources"

Stratis D. Viglas, Jeffrey F. Naughton, Department of Computer Sciences University of Wisconsin-Madison (2002)

[33] "Dynamic Pipeline Scheduling for Improving Interactive Query Performance"

Tolga Urhan, Michael J. Franklin, University of California, Berkeley

[34] "Load Shedding in a Data Stream Manager"

Nesime Tatbul, Ugur Cetintemel, Stan Zdonik, Brown University, Mitch Cherniak, Brandeis University, Michael Stonebraker, MIT

[35] "Load Shedding Techniques for Data Stream Systems"

Brian Babcock, Mayur Datar, Rajeev Motwani, Department of Computer Science Stanford University

[36] "Cost-based Query Scrambling for Initial Delays"

Tolga Urhav, Michael J. Franklin, University of Maryland, Laurent Amsaleg, IRISA

[37] "Eddies: Continuously Adaptive Query Processing"

Ron Avnur, Joseph M. Hellerstein, University of California, Berkeley

[38] "Tuple Routing Strategies for Distributed Eddies"

Feng Tian, David J. DeWitt, Department of Computer Sciences, University of Wisconsin, Madison (2003)

[39] "Continuously Adaptive Continuous Queries over Streams"

Samuel Madden, Mehul Shah, Joseph M. Hellerstein, UC Berkeley, Vijayshankar Raman, IBM Almaden Research Center (2002)

[40] "Using State Modules for Adaptive Query Processing"

Vijayshankar Rama, IBM Almaden Research Center, Amol Deshpande, Joseph M. Hellerstein University of California, Berkeley

[41] "Filtering, Punctuation, Windows and Synopses"

David Maier, OGI School of Science and Engineering, Peter A. Tucker, Whitworth College, Minos Garofalakis, Bell Laboratories (2005)

[42] "Applying Punctuation Semantics over Continuous Data Streams"

Petes A. Tucker, David Maier, Tim Sheard, OGI School of Science and Engineering (2003)

[43] "Tracking Join and Self-Join Sizes in Limited Storage"

Noga Alon, Tel Aviv University, Phillip P. Gibbons, Intel Research, Yossi Mattias, Tel Aviv University, Mario Szegedy, Pittsburgh University (2002)

[44] "DataStreams and Histograms"

Sudipto Guha, Nick Koudas, AT&T Research, Kyuseok Shim, Computer Science Department, Korean Advanced Institut of Science and Technology (2001)

[45] "Surfing Wavelets on Streams: One-Pass Summaries for Approximate Aggregate Queries"

Anna C. Gilbert, Yannis Kotidis, S. Muthukrishnan, Martin J. Strauss, AT&T Labs-Research (2001)

[46] "Dynamic Maintenance of Wavelet-Based Histograms"

Yossi Matias, Tel Aviv University, Jeffrey Scott Vitter, Duke University, Min Wang, Data Management Department IBM T. J. Watson Research Center

[47] "The CQL Continuous Query Language: Semantic Foundations and Query Execution"
Arvind Arasu, Shivnath Babu, Jennifer Widom, Stanford University

[48] "An Abstract Semantics and Concrete Language for Continuous Queries over Streams and Relations"
Arvind Arasu, Shivnath Babu, Jennifer Widom, Stanford University

[49] "STREAM – Stream Data Manager"
http://www-db.stanford.edu/stream/, Stanford University

[50] "Efficient Support for Time Series Queries in Data Stream Management Systems"
Yijian Bai, Chang R. Luo, Hetal Thakkar, Carlo Zaniolo, Computer Science Department, University of California (2005)

[51] "Query Languages and Data Models for Database Sequences and Data Streams"
Yan-Nei Law, Haixun Wang, Computer Science Department, University of California, Carlo Zaniolo, IBM T. J. Watson Research

[52] "An Introduction to the Expressive Stream Language (ESL)"
Web Information System Laboratory, UCLA, CS Department

[53] "Stream Mill: Bringing Power and Generality to Data Stream Management Systems"
http://wis.cs.ucla.edu/stream-mill/index.html

[54] "Telegraph (Berkeley Database Research): Adaptive Dataflow for Querying Streams, Deep Web, and Beyond "
http://telegraph.cs.berkeley.edu/

[55] "The Aurora Project"
http://www.cs.brown.edu/research/aurora/

[56] "The Gigascope Stream Database"
Chuck Cranor, Theodore Johnson, Oliver Spatscheck, AT&T Labs, Vladislav Shkapenyuk, Department of Computer Science CMU

[57] "TelegraphCQ: An Architectural Status Report"
Sailesh Krishnamurthy, Sirish Chandrasekaran, Owen Cooper, Amol Deshpande, Michael J. Franklin, Joseph M. Hellerstein, Wei Hong, Samuel R. Madden, Fred Reiss, Mehul A. Shah, University of California, Berkeley (2003)

[58] "TelegraphCQ: Continuous Dataflow Processing for an Uncertain World"

Sirish Chandrasekaran, Owen Cooper, Amol Deshpande, Michael J. Franklin, Joseph M. Hellerstein, Wei Hong, Sailesh Krishnamurthy, Sam Madden, Vijayshankar Raman, Fred Reiss, and Mehul Shah University of California, Berkeley

[59] "PostgreSQL: The World's most Advanced Open Source Database"

http://www.postgresql.org/

[60] "STREAM: The Stanford Stream Data Manager"

The STREAM Group, Stanford University (2003)

[61] "STREAM: The Stanford Data Stream Management System"

Arvind Arasu, Brian Babcock, Shivnath Babu, John Cieslewicz, Mayur Datar, Keith Ito, Rajeev Motwani, Utkarsh Srivastava, and Jennifer Widom, Department of Computer Science, Stanford University

[62] "Borealis: Second Generation Stream Processing Engine"

http://www.cs.brown.edu/research/db/borealis/

[63] "The Design of the Borealis Stream Processing Engine"

Daniel J. Abadi, M.I.T Cambridge, Yanif Ahmad, Brown University, Magdalena Balazinska, M.I.T Cambridge, Ugur Cetintemel, Brown University, Mitch Cherniack, Brandeis University, Jeong-Hyon Hwang, Brown University, Wolfgang Lindner, M.I.T Cambridge, Anurag S. Maskey, Brandeis University, Alexander Rasin, Brown University, Esther Ryvkina, Brandeis University, Nesime Tatbul, Ying Xing, Stan Zdonik, Brown University (2005)

[64] "NESTREAM: Querying Nested Streams"

Damianos Chatziantoniou, Achilleas Anagnostopoulos, Department of Management Science and Technology, Athens University of Economics and Business (2004)

[65] "Sensor Data Stream Processing in Health Monitoring"

Heiko Schuldt, Gert Brettlecker, University for Health Informatics and Technology Tyrol, Austria

[66] "The Sensor Spectrum: Technology, Trends, and Requirements"

Joseph M. Hellerstein, Wei Hong, Samuel R. Madden, UC Berkeley, Intel Research Berkeley, MIT

[67] "Using Data Stream Management Systems for Traffic Analysis – A Case Study", Thomas Plagemann, Vera Goebel, Institut Eurecom, Andrea Bergamini, Giacomo Tolu, Guillaume Urvoy-Keller, Ernst W. Biersack, University of Oslo, Department of Informatics

[68] "Linear Road: A Stream Data Management Benchmark"

http://www.cs.brandeis.edu/~linearroad/

[69] "Linear Road: A Stream Data Management Benchmark"

Arvind Arasu, Stanford University, Mitch Cherniack, Brandeis University, Eduardo Galvez, Brandeis University, David Maier, OHSU/OGI, Anurag S. Maskey, Brandeis University, Esther Ryvkina, Brandeis University, Michael Stonebraker, MIT, Richard Tibbetts, MIT (2004)

[70] "MITSIMLab: MIT Intelligent Transportation Systems Program"

http://mit.edu/its/mitsimlab.html

[71] "NEXMark – A Benchmark for Queries over Data Streams"

Pete Tucker, Kristin Tufte, Vassilis Papadimos, David Maier, OGI School of Science & Engineering at OHSU

[72] "NEXMark Benchmark"

http://datalab.cs.pdx.edu/niagara/NEXMark/

[73] "**XMark - An XML Benchmark Project"**

http://www.xml-benchmark.org/ - http://monetdb.cwi.nl/xml/

[74] "SIS – Data Entry Language User's Manual"

Version 2.2, Institute of Computer Science Foundation for Research and Technology - Hellas