ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΗΛΕΚΤΡΟΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ ΚΑΙ ΔΙΚΤΥΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Επεξεργασία ερωτήσεων σε Ασύρματα Δίκτυα Αισθητήρων με μνήμες στερεάς κατάστασης NAND

Φοιτητής:
Δημήτριος Σ. Αργυρίου

Επιβλέποντες:
Παναγιώτης Δ. Μποζάνης, Επίκουρος Καθηγητής
Δημήτριος Κατσαρός, Λέκτορας

ΒΟΛΟΣ
ΦΕΒΡΟΥΑΡΙΟΣ 2010

Πρόλογος

Τα Ασύρματα Δίκτυα Αισθητήρων αποτελούν την πλέον αναπτυσσόμενη τεχνολογία στον τομέα των Ασύρματων Επικοινωνιών και πιθανολογείται, ότι θα προκαλέσουν τεχνολογική επανάσταση στο μέλλον. Είναι χαρακτηριστικό το γεγονός, ότι στο ήδη μεγάλο υπάρχον πλήθος εφαρμογών των Ασύρματων Δικτύων Αισθητήρων, προστίθενται συνεχώς νέες εφαρμογές που αφορούν σχεδόν κάθε τομέα της ανθρώπινης δραστηριότητας. Εντούτοις, ένα σύνολο περιορισμών στους οποίους και υπόκεινται τα Ασύρματα Δίκτυα Αισθητήρων καθιστά επιτακτική την ανάπτυξη νέων καινοτόμων τεχνικών κι εργαλείων για την αποτελεσματικότερη λειτουργία και διαχείρισή τους. Αυτό ακριβώς είναι και το αντικείμενο της παρούσας μελέτης, η οποία κι εστιάζει το ενδιαφέρον της στα Ασύρματα Δίκτυα Αισθητήρων με μνήμη flash τύπου NAND.

Στο **πρώτο κεφάλαιο** παρουσιάζονται αναλυτικά τα Ασύρματα Δίκτυα Αισθητήρων. Πιο συγκεκριμένα, γίνεται αναφορά στη δομή τους, στα χαρακτηριστικά τους, στην τοπολογία τους, στα πλεονεκτήματα και τα μειονεκτήματά τους, ενώ επίσης παρουσιάζονται και μερικές από τις εφαρμογές τους.

Στο **δεύτερο κεφάλαιο**, αρχικά παρουσιάζουμε τη δομή, τα χαρακτηριστικά, τα πλεονεκτήματα και τα μειονεκτήματα της μνήμης NAND flash. Στη συνέχεια εξηγούμε τους λόγους, για τους οποίους η μνήμη NAND flash έχει επιλεγεί ως το καταλληλότερο μέσο αποθήκευσης στους SNs ενός WSN. Τέλος, παρουσιάζουμε τρεις καινοτόμες τεχνικές διαχείρισης αποθήκευσης στη μνήμη NAND flash των SNs ενός WSN.

Στο **τρίτο κεφάλαιο** μελετάμε τεχνικές ευρετηρίασης των δεδομένων, που προέρχονται από τις μετρήσεις των αισθητήρων σε ένα WSN. Για την ακρίβεια, παρουσιάζουμε αλγορίθμους για τη δημιουργία τοπικού ευρετηρίου των δεδομένων ενός SN, καθώς επίσης κι αλγορίθμους για τη δημιουργία κατανεμημένου ευρετηρίου των δεδομένων μίας ομάδας SNs. Όπως θα δούμε, οι αλγόριθμοι αυτοί αποσκοπούν στην ταχύτερη εύρεση των ζητούμενων δεδομένων.

Στο **τέταρτο κεφάλαιο** μελετάμε την κατανεμημένη επεξεργασία και βελτιστοποίηση των αιτημάτων, που εξυπηρετούνται από ένα WSN. Πιο συγκεκριμένα, αναλύουμε με ποιον τρόπο μπορούμε να κατανείμουμε την επεξεργασία ενός αιτήματος στους κόμβους ενός WSN. Επιπλέον, παρουσιάζουμε μία τεχνική βελτιστοποίησης των αιτημάτων πριν από την επεξεργασία τους από ένα WSN. Επίσης, προσθέτουμε κρυφή μνήμη στους κόμβους του WSN. Τέλος, εξετάζουμε τη δρομολόγηση των αιτημάτων που αποστέλλονται από τους SNs στη Sink σε ένα WSN.

Στο **πέμπτο κεφάλαιο** γίνεται μία επισκόπηση του συνόλου των τεχνικών που παρουσιάζονται στην παρούσα μελέτη σε ένα WSN.

Τέλος, αξίζει να επισημάνουμε το μεγάλο πλήθος παραδειγμάτων, εικόνων, πινάκων και σχημάτων, που χρησιμοποιούνται στην παρούσα εργασία, καθώς και την αναλυτική λεκτική περιγραφή όλων των αλγορίθμων με απώτερο σκοπό τη διευκόλυνση του αναγνώστη.

Περιεχόμενα

Κεφάλαιο 3^{ο} - Τοπική και κατανεμημένη ευρετηρίαση……

Παραρτήματα ……..

Βιβλιογραφία…….

Κεφάλαιο 1

Ασύρματα Δίκτυα Αισθητήρων

Περιεχόμενα

1.1 Ιστορία των Ασύρματων Επικοινωνιών

Η γέννηση των ασύρματων επικοινωνιών έλαβε χώρα το έτος 1888, οπότε και ο Hertz κατασκευάζοντας την πρώτη διπολική κεραία, επιβεβαίωσε τις προγενέστερες προβλέψεις των Maxwell και Faraday για την ύπαρξη των ηλεκτρομαγνητικών κυμάτων, τα οποία κι ως γνωστόν αποτελούν το αόρατο μέσο μετάδοσης πολλών διαφορετικών ειδών δεδομένων. Ο Hertz θεωρούσε, ότι τα ασύρματα κύματα που ανακάλυψε δεν θα είχαν ποτέ κάποια πρακτική εφαρμογή, όμως διαψεύστηκε μόλις πέντε χρόνια αργότερα, οπότε και κατασκευάστηκε το πρώτο ασύρματο σύστημα επικοινωνίας από τον Tesla. Την αμέσως επόμενη χρονιά και πιο συγκεκριμένα το 1894, ο Popov κατόρθωσε να μεταδώσει ραδιοκύματα μεταξύ κοντινών κτιρίων, ενώ το 1901 επετεύχθη η αποστολή του πρώτου υπερατλαντικού σήματος από τον Marconi. Ο τελευταίος ίδρυσε το 1903 την πρώτη εταιρία ασύρματου τηλέγραφου, τεχνολογία που έμελλε να χρησιμοποιηθεί ευρέως στις δεκαετίες που ακολούθησαν. Από τότε μέχρι και σήμερα έχουν πραγματοποιηθεί τεράστια άλματα στην ανάπτυξη των ασύρματων επικοινωνιών, με αποτέλεσμα την ευρεία χρήση της ασύρματης μεταφοράς δεδομένων στην καθημερινή ζωή, με μερικά αντιπροσωπευτικά παραδείγματα να αποτελούν το ραδιόφωνο, η τηλεόραση, η κινητή τηλεφωνία, οι δορυφορικές επικοινωνίες, η επικοινωνία μέσω υπέρυθρης ακτινοβολίας, η τεχνολογία Bluetooth και τα ασύρματα δίκτυα υπολογιστών ή άλλων συσκευών. Σήμερα, η πλέον αναπτυσσόμενη τεχνολογία στον τομέα των ασύρματων επικοινωνιών είναι τα Ασύρματα Δίκτυα Αισθητήρων (Wireless Sensor Networks – WSNs), τα οποία όπως πιθανολογείται θα προκαλέσουν τεχνολογική επανάσταση στο μέλλον.


Εικ 1.1: Tesla

Εικ 1.2: Hertz


Εικ 1.3: Marconi

1.2 Η γέννηση των Ασύρματων Δικτύων Αισθητήρων

Όπως έχει συμβεί και με πλήθος άλλων τεχνολογιών, η γέννηση και η εξέλιξη των Ασύρματων Δικτύων Αισθητήρων (WSNs) οφείλεται σε στρατιωτικά ερευνητικά προγράμματα. Πιο συγκεκριμένα, η πρώτη γνωστή εφαρμογή δικτύων αισθητήρων υπήρξε το SOSUS που χρησιμοποιήθηκε κατά τη διάρκεια του Ψυχρού Πολέμου για την ανίχνευση και τον εντοπισμό Σοβιετικών υποβρυχίων με τη βοήθεια ειδικών ακουστικών αισθητήρων. Το SOSUS παραμένει ακόμη σε λειτουργία για ειρηνικούς πλέον σκοπούς, όπως για παράδειγμα η παρακολούθηση φαλαινών και η μελέτη της σεισμικής δραστηριότητας. Το αμέσως επόμενο δίκτυο αισθητήρων ήταν το Smart Dust της DARPA και αναπτύχθηκε επίσης για στρατιωτική χρήση. Βασίστηκε στην ιδέα, ότι χιλιάδες μικροσκοπικοί αισθητήρες θα μπορούσαν να διασκορπιστούν από ένα μικρό, μη επανδρωμένο αεροπλάνο για να συλλέγουν πληροφορίες από τον εχθρό στο πεδίο της μάχης. Σταδιακά, τα WSNs άρχισαν να χρησιμοποιούνται σε πληθώρα εφαρμογών και χάρη στην έρευνα που γίνεται πάνω σε αυτά, αναπτύσσονται και βελτιώνονται με ραγδαίο ρυθμό.

1.3 Ασύρματα Δίκτυα Αισθητήρων

Ένα ασύρματο δίκτυο αισθητήρων (Wireless Sensor Network – WSN) χρησιμοποιείται για τη συλλογή κι ανάλυση φυσικών ή περιβαλλοντικών δεδομένων από μία περιοχή παρατήρησης χωρίς καμία εκ των προτέρων εγκατεστημένη δικτυακή υποδομή και γνώση της τοπολογίας, με σκοπό την παρακολούθηση κάποιου φαινομένου και μπορεί να χρησιμοποιηθεί σε πληθώρα εφαρμογών. Αποτελείται από μερικές δεκάδες, μέχρι και αρκετές χιλιάδες κόμβους αισθητήρων (Sensor Nodes - SNs), οι οποίοι είναι διασκορπισμένοι σε μία γεωγραφική περιοχή και υπόκεινται σε περιορισμούς όσον αφορά την ενέργεια και τους αποθηκευτικούς χώρους που διαθέτουν. Οι κόμβοι αυτοί λαμβάνουν διάφορα είδη δεδομένων από το περιβάλλον τους, τα οποία όμως αδυνατούν να διαχειριστούν οι ίδιοι, λόγω της περιορισμένης διαθέσιμης ενέργειας και μνήμης τους. Για αυτόν το λόγο, αποστέλλουν ασύρματα τις μετρήσεις τους μέσω των γειτονικών κόμβων σε μία ή περισσότερες κεντρικές βάσεις συγκέντρωσης δεδομένων (Sinks) προς επεξεργασία αυτών και λήψη αποφάσεων, γεγονός που όμως επιφέρει μεγάλο ενεργειακό κόστος για το δίκτυο.

1.3.1 Μέσο μετάδοσης

Σε ένα WSN η μετάδοση όλων των δεδομένων αυτού γίνεται ασύρματα, γεγονός που σημαίνει ότι όλες οι συνδέσεις μεταξύ των κόμβων του πρέπει να είναι ασύρματες. Το ασύρματο μέσο μετάδοσης των δεδομένων σε ένα WSN μπορεί να είναι είτε η ηλεκτρομαγνητική, είτε η οπτική, είτε η υπέρυθρη ακτινοβολία. Στην περίπτωση πάντως, κατά την οποία επιθυμούμε η λειτουργία ενός WSN να εκτείνεται σε παγκόσμιο επίπεδο, θα πρέπει να διαλέξουμε ένα ασύρματο μέσο μετάδοσης, το οποίο και θα είναι διαθέσιμο παντού στον κόσμο. Στην πράξη μία συνήθης επιλογή για τις ασύρματες ζεύξεις μέσω ραδιοσυχνοτήτων είναι η χρήση της Βιομηχανικής, Επιστημονικής, Ιατρικής Μπάντας (Industrial, Scientific, Medical ISM Band). Οι διαθέσιμες συχνότητες της μπάντας ISM, κάποιες από τις οποίες χρησιμοποιούνται ήδη σε ασύρματες συσκευές τηλεφώνου ή σε τοπικά ασύρματα δίκτυα (WLANs) παρουσιάζονται στον κάτωθι πίνακα:

6765 – 6965 kHz	433,05 – 434,79 MHz	24 – 24,25 GHz
13553 – 13567 kHz	902 – 928 MHz	61 – 61,5 GHz
26957 – 27283 kHz	2400 – 2500 MHz	122 – 123 GHz
40,66 – 40,70 MHz	5725 – 5875 MHz	244 – 246 GHz

Πίνακας 1.1: Συχνότητες της μπάντας ISM, που μπορεί να χρησιμοποιηθούν για τις ασύρματες ζεύξεις των WSNs.

Αξίζει να αναφέρουμε, ότι ειδικότερα για την Ευρώπη προτείνεται η χρήση της συχνότητας των 433 MHz, ενώ για τη Βόρεια Αμερική αυτή των 915 MHz. Τα πλεονεκτήματα της χρήσης των συχνοτήτων της μπάντας ISM είναι το τεράστιο φάσμα τους, η παγκόσμια διαθεσιμότητά τους και το ότι προσφέρονται χωρίς άδεια χρήσης, δηλαδή δωρεάν στα περισσότερα κράτη του πλανήτη. Επιπλέον, δεν περιορίζονται από κάποιο πρότυπο δίνοντας έτσι μεγαλύτερη ελευθερία στην υλοποίηση τεχνικών εξοικονόμησης ενέργειας στα WSNs.

1.3.2 Κόμβοι αισθητήρες (SNs)

Κάθε κόμβος αισθητήρας (SN) είναι στην ουσία μία χαμηλού κόστους, κινητή ή μη κινητή, μικροσκοπική ηλεκτρονική διάταξη, με ενσωματωμένους αισθητήρες για την παρακολούθηση και καταγραφή μετρήσιμων παραμέτρων, όπως η θερμοκρασία, η υγρασία, η φωτεινότητα, η δόνηση, η πίεση, ο ήχος, η κίνηση, η μόλυνση, οι σφυγμοί της καρδιάς, η ωρίμανση των καρπών και πολλές άλλες. Αποτελείται από τα ακόλουθα υποσυστήματα:

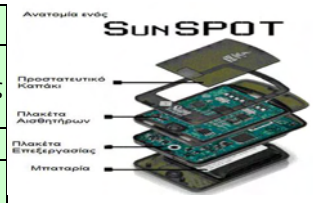
- Επεξεργασίας, το οποίο ενσωματώνει έναν μικροεπεξεργαστή, καθώς και μνήμη επεξεργασίας κι αποθήκευσης δεδομένων.
- Επικοινωνίας, με χρήση κάποιου πομποδέκτη.
- Αισθητήρων, για συλλογή των δεδομένων καθώς και τη μετατροπή τους από αναλογική σε ψηφιακή μορφή.
- Τροφοδοσίας, το οποίο αποτελείται από μία μικρή μπαταρία που παρέχει περιορισμένη ενεργειακή αυτονομία.

Επίσης, κάθε SN κι ανάλογα με την εφαρμογή, στην οποία χρησιμοποιείται μπορεί να περιλαμβάνει και τα ακόλουθα υποσυστήματα:

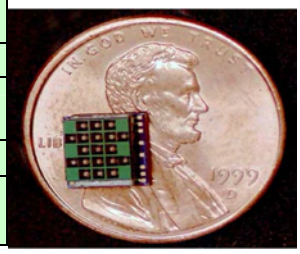
- Εύρεσης Γεωγραφικού Στίγματος, το οποίο και βοηθάει στην εύρεση της ακριβούς θέσης του SN στην περιοχή ανάπτυξης του δικτύου.
- Κίνησης, ώστε να μπορεί να μετακινείται αν αυτό χρειάζεται.
- Αναπλήρωσης Ενέργειας, για την επαναφόρτιση της μπαταρίας του.

ΥΠΟΣΥΣΤΗΜΑΤΑ SN	ΠΕΡΙΓΡΑΦΗ
ΕΠΕΞΕΡΓΑΣΙΑΣ	Μικροεπεξεργαστής, καθώς και μνήμη επεξεργασίας κι αποθήκευσης δεδομένων
ΕΠΙΚΟΙΝΩΝΙΑΣ	Πομποδέκτης
ΑΙΣΘΗΤΗΡΩΝ	Διατάξεις συλλογής δεδομένων και μετατροπής τους από αναλογική σε ψηφιακή μορφή
ΤΡΟΦΟΔΟΣΙΑΣ	Μικρή μπαταρία
ΕΥΡΕΣΗΣ ΓΕΩΓΡΑΦΙΚΟΥ ΣΤΙΓΜΑΤΟΣ	GPS
ΚΙΝΗΣΗΣ	Μηχανισμοί κίνησης
ΑΝΑΠΛΗΡΩΣΗΣ ΕΝΕΡΓΕΙΑΣ	Φωτοβολταϊκά

Πίνακας 1.2: Υποσυστήματα κόμβου αισθητήρα -SN



Εικόνα 1.4: Ανατομία ενός αισθητήρα



Εικόνα 1.5: Διατάξεις μικρότερες κι από ένα νόμισμα

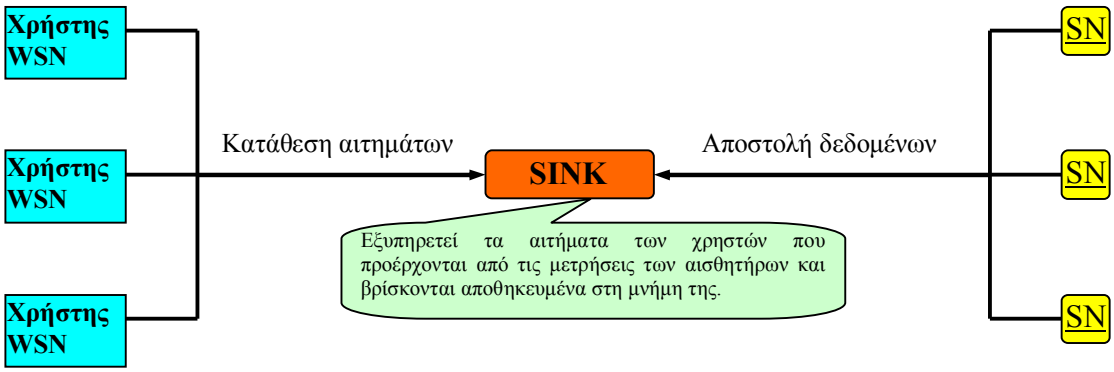
1.3.3 Κεντρική Βάση Συγκέντρωσης Δεδομένων (Sink)

Η Sink ενός WSN αποτελεί στην ουσία ένα σταθμό βάσης, ο οποίος κι αναλαμβάνει τη συνολική διαχείριση του WSN, μεσολαβώντας μεταξύ των χρηστών και των SNs αυτού. Πιο συγκεκριμένα, η Sink παραλαμβάνει συνεχώς από τους SNs τα δεδομένα που προέρχονται από τις μετρήσεις των αισθητήρων τους κι αποθηκεύει τα δεδομένα αυτά στη μνήμη της. Έτσι λοιπόν, όποτε η Sink παραλαμβάνει κάποιο αίτημα από κάποιο χρήστη, επεξεργάζεται τα δεδομένα των αισθητήρων που

βρίσκονται αποθηκευμένα στη μνήμη της, προκειμένου να εξυπηρετήσει το εν λόγω αίτημα. Όπως είναι προφανές η Sink, προκειμένου να εκτελέσει επιτυχώς το ρόλο που της έχει ανατεθεί σε ένα WSN, θα πρέπει να είναι επαρκής σε διαθέσιμους πόρους. Έτσι λοιπόν, σε αντίθεση με τους SNs που υπόκεινται σε περιορισμούς αναφορικά με τους διαθέσιμους πόρους, πρέπει να διαθέτει αυξημένη επεξεργαστική ισχύ, μεγάλης χωρητικότητας μνήμη και θεωρητικά απεριόριστη διαθέσιμη ενέργεια. Συνεπώς, η Sink πρέπει να βρίσκεται εγκατεστημένη σε μία ασφαλή τοποθεσία κι επιπλέον κοντά στο WSN, προκειμένου να είναι προσβάσιμη από αυτό. Συνήθως, η Sink είναι ένας σταθερός ή φορητός υπολογιστής, ενώ σε ένα WSN μπορεί να συνυπάρχουν περισσότερες από μία Sink. Πέραν του υλικού, η Sink θα πρέπει να διαθέτει και εξειδικευμένο για το WSN λογισμικό, μέσω του οποίου και θα δύναται:

- Να αποθηκεύει, να επεξεργάζεται και να απαντά στα αιτήματα των χρηστών
- Να αποθηκεύει και να επεξεργάζεται τα δεδομένα των αισθητήρων
- Να επικοινωνεί με ενδεχομένως υπάρχουσες άλλες Sink
- Να ανανήφει από σφάλματα
- Να ενημερώνεται δυναμικά για την τοπολογία του WSN
- Να εφαρμόζει αλγόριθμους βελτιστοποίησης, όπως για παράδειγμα αλγόριθμους δρομολόγησης, για την αύξηση της αποδοτικότητας του δικτύου και την ταχύτερη εξυπηρέτηση των αιτημάτων.

Συμπερασματικά λοιπόν, η Sink αποτελεί μεταφορικά τον εγκέφαλο και ταυτόχρονα το μέσο πρόσβασης ενός WSN κι ως εκ τούτου πρέπει να διαθέτει υλικό αυξημένων δυνατοτήτων, όπως επίσης και εξειδικευμένο για αυτό το WSN λογισμικό.



Σχήμα 1.1: Αλληλεπίδραση της Sink με τους SNs και τους χρήστες ενός WSN

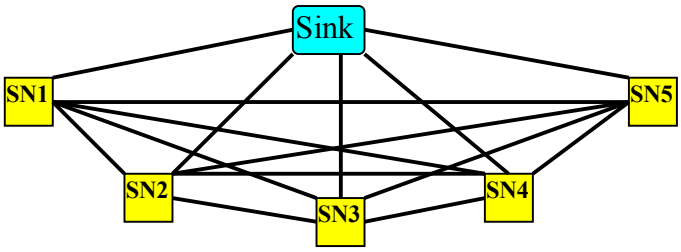
1.3.4 Τοπολογία

Μετά λοιπόν από την παρουσίαση των κόμβων ενός WSN, μπορούμε πλέον να αναφερθούμε και στην τοπολογία αυτού. Ως γνωστόν, με τον όρο τοπολογία ενός δικτύου, εννοούμε τη μορφή των συνδέσεων μεταξύ των κόμβων του δικτύου αυτού. Η τοπολογία ενός WSN δεν χρειάζεται να είναι εκ των προτέρων γνωστή, γεγονός που επιτρέπει την ταχεία ανάπτυξη ενός WSN σε δύσβατες ή ακατάλληλες για τον άνθρωπο περιοχές. Επιπλέον, η μορφή των συνδέσεων μεταξύ των SNs ενός WSN μπορεί να αλλάζει δυναμικά κατά τη λειτουργία του, προκειμένου να επιλυθούν πάσης φύσεως προβλήματα και να βελτιωθεί η συνολική απόδοση του WSN κατά την

εξυπηρέτηση των αιτημάτων των χρηστών αυτού. Υπάρχουν διάφορες προτεινόμενες τοπολογίες για ένα WSN, καθεμία από τις οποίες έχει τα δικά της χαρακτηριστικά και μπορεί να χρησιμοποιηθεί ανάλογα με τις ανάγκες του WSN. Κάποιες από τις τοπολογίες αυτές, είναι οι ακόλουθες:

α) Πλήρης τοπολογία

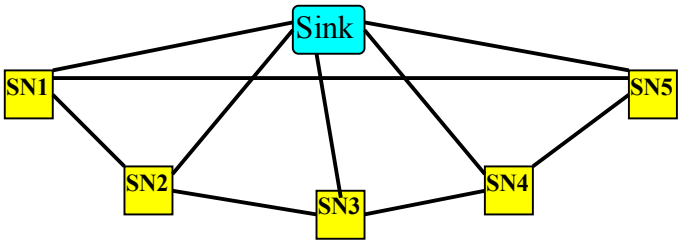
Όλοι οι SNs συνδέονται, τόσο μεταξύ τους, όσο και με τη Sink. Με την εφαρμογή κατάλληλων αλγορίθμων δρομολόγησης, η τοπολογία αυτή εξασφαλίζει την επικοινωνία της Sink με όλους τους SNs, ακόμη κι αν σε κάποια χρονική στιγμή καταστραφούν μερικές από τις ασύρματες ζεύξεις του δικτύου. Παρόλα αυτά, μπορεί να προκληθεί συμφόρηση και εν τέλει κατάρρευση του δικτύου, ενώ επίσης απαιτεί την ενσωμάτωση και διαχείριση παραπάνω του ενός πομποδεκτών σε κάθε SN και κρίνεται ακατάλληλη για WSNs με μεγάλο αριθμό SNs. Εντούτοις, μπορεί να χρησιμοποιηθεί σε ειδικές περιπτώσεις και για μικρό μόνο πλήθος SNs.



Σχήμα 1.2: Πλήρης τοπολογία. Είναι εμφανές, το μεγάλο πλήθος των ασύρματων ζεύξεων μεταξύ των SNs.

β) Τοπολογία πλέγματος

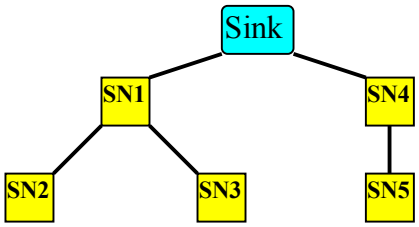
Με την τοπολογία πλέγματος, κάθε SN πρέπει να συνδέεται στο WSN με ένα μέγιστο πλήθος άλλων SNs. Για παράδειγμα, αν σε ένα WSN χρησιμοποιείται τοπολογία πλέγματος-3, τότε κάθε SN αυτού θα πρέπει να διαθέτει ασύρματο δίαυλο επικοινωνίας με τουλάχιστον τρεις άλλους SNs, συμπεριλαμβανομένης της Sink. Η συγκεκριμένη τοπολογία με χρήση κατάλληλων αλγορίθμων δρομολόγησης εξασφαλίζει την ανάνηψη του δικτύου από καταρρεύσεις των ασύρματων ζεύξεων ή των κόμβων αυτού. Αποτελεί στην ουσία μία εξειδίκευση της πλήρους τοπολογίας, η οποία αντιμετωπίζει μερικές μόνο από τις αδυναμίες αυτής.



Σχήμα 1.3: Τοπολογία πλέγματος-2. Κάθε SN συνδέεται με τη Sink κι άλλους 2 SNs.

γ) Τοπολογία δένδρου

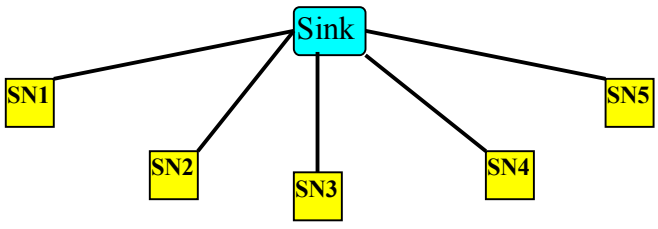
Στη ρίζα του δέντρου βρίσκεται η Sink, ενώ στους εσωτερικούς κόμβους και στα φύλλα αυτού οι SNs. Κάθε κόμβος του δέντρου επικοινωνεί μόνο με τους κόμβους-παιδιά αυτού, καθώς επίσης και με τον κόμβο που αποτελεί πατέρα του στο δέντρο. Για την ακρίβεια, κάθε κόμβος παραλαμβάνει αιτήματα από τον κόμβο-πατέρα του και τα αποστέλλει στους κόμβους-παιδιά του, ενώ αποστέλλει τα δεδομένα των μετρήσεων του, καθώς επίσης και των μετρήσεων των κόμβων-παιδιών του, στον κόμβο-πατέρα του. Η τοπολογία δένδρου είναι ιδανική για WSNs, στα οποία λόγω μεγάλης απόστασης μεταξύ της Sink και πολλών από τους SNs δεν καθίσταται εφικτή η ύπαρξη ασύρματων ζεύξεων μεταξύ τους. Παρουσιάζει το μειονέκτημα, ότι κάποιοι από τους SNs είναι επιφορτισμένοι με την αποστολή και λήψη δεδομένων που αφορούν άλλους SNs, γεγονός που μπορεί να επιφέρει συγκρούσεις και μεγάλες καθυστερήσεις κατά τη μετάδοση των δεδομένων. Επιπλέον, αν τεθεί εκτός λειτουργίας για κάποιο λόγο ένας SN, τότε αυτόματα διακόπτεται και η επικοινωνία της Sink με όλους τους SNs που αποτελούν παιδιά αυτού του SN στο δέντρο.



Σχήμα 1.4: Τοπολογία δέντρου. Ρίζα του δέντρου η Sink και κόμβοι του οι SNs

δ) Τοπολογία αστέρα

Όλοι οι SNs συνδέονται απευθείας με τη Sink, αλλά όχι και μεταξύ τους. Με τον τρόπο αυτό δημιουργείται ένας αστέρας με κέντρο του τη Sink και άκρα του, τους SNs. Είναι εξαιρετικά απλή η υλοποίησή της, αλλά δεν μπορεί να εφαρμοστεί όταν η απόσταση μεταξύ της Sink και κάποιων από τους SNs δεν επιτρέπει την ύπαρξη ασύρματων ζεύξεων μεταξύ τους. Επιπρόσθετα, εάν για οποιοδήποτε λόγο καταστραφεί ο ασύρματος δίαυλος επικοινωνίας, μεταξύ ενός SN και της Sink, τότε δεν υπάρχει εναλλακτική διαδρομή για τη συνέχιση της μεταξύ τους επικοινωνίας.

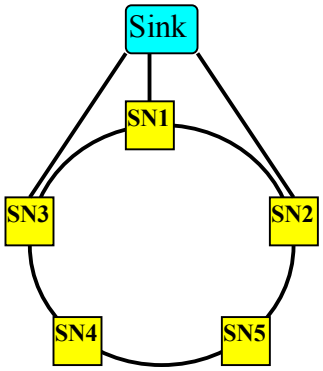


Σχήμα 1.5: Τοπολογία αστέρα. Κάθε SN συνδέεται μόνο με τη Sink.

ε) Τοπολογία δακτυλίου

Όλοι οι SNs συνδέονται σε μορφή δακτυλίου μεταξύ τους. Πιο συγκεκριμένα, οι SNs σχηματίζουν έναν κύκλο, στον οποίο καθένας από αυτούς συνδέεται με τους γειτονικούς σε αυτόν SNs, δηλαδή με τον επόμενο και τον προηγούμενο SN. Όσον

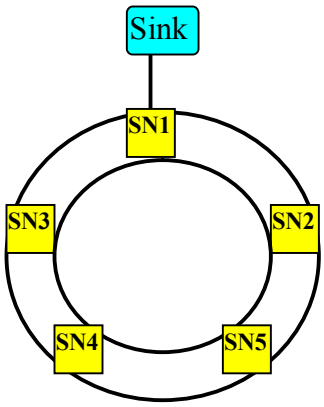
αφορά τη σύνδεση των SNs με τη Sink, μπορεί είτε όλοι είτε μερικοί μόνο από τους SNs να διαθέτουν ασύρματη ζεύξη με τη Sink. Το πιο σημαντικό πλεονέκτημα αυτής της τοπολογίας είναι η ύπαρξη εναλλακτικών διαδρομών μεταξύ της Sink και κάθε SN, γεγονός που εξασφαλίζει τη μεταξύ τους επικοινωνία, ακόμη κι αν καταστούν ανέφικτες κάποιες από τις ασύρματες ζεύξεις του WSN.



Σχήμα 1.6: Τοπολογία δακτυλίου. Με τη Sink συνδέονται οι SN1, SN2 και SN3. Θα μπορούσαν όλοι οι SNs να έχουν ζεύξη με τη Sink.

στ) Τοπολογία διπλού δακτυλίου

Είναι ίδια ακριβώς με τη τοπολογία δακτυλίου, με τη διαφορά πως αντί για έναν δακτύλιο υπάρχουν πλέον δύο δακτύλιοι. Δηλαδή, οι SNs σχηματίζουν δύο κύκλους σε καθέναν από τους οποίους κάθε SN συνδέεται με τους γειτονικούς σε αυτόν SN. Με αυτήν την τοπολογία, εξασφαλίζεται ακόμη περισσότερο η διαθεσιμότητα του WSN.

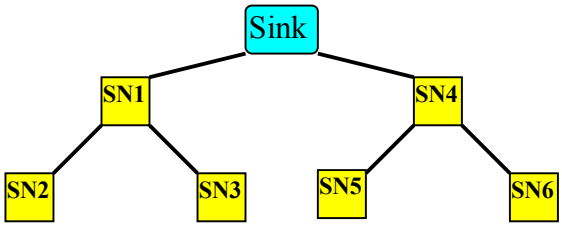


Σχήμα 1.7: Τοπολογία διπλού δακτυλίου. Με τη Sink συνδέονται μόνο ο SN1.Φυσικά θα μπορούσαν όλοι οι SNs να διαθέτουν ασύρματη ζεύξη με τη Sink.

Πέραν των παραπάνω τοπολογιών υπάρχει και πλήθος άλλων, ενώ επίσης δύο ή περισσότερες τοπολογίες μπορεί να συνδυαστούν σε ένα WSN για καλύτερα αποτελέσματα. Για παράδειγμα, θα εξετάσουμε στη συνέχεια, το συνδυασμό των τοπολογιών δέντρου και αστέρα.

ζ) Συνδυασμός τοπολογίας δέντρου και τοπολογίας αστέρα

Εάν, η τοπολογία αστέρα συνδυαστεί με την τοπολογία δέντρου, τότε θα δημιουργηθεί ένα ισοζυγισμένο δέντρο με ρίζα του τη Sink. Σε αυτό το δέντρο η ρίζα και κάθε εσωτερικός κόμβος αυτού, αποτελούν το κέντρο ενός αστέρα με άκρα τα παιδιά τους. Αποτελεί την πλέον χρησιμοποιούμενη τοπολογία στα WSNs.

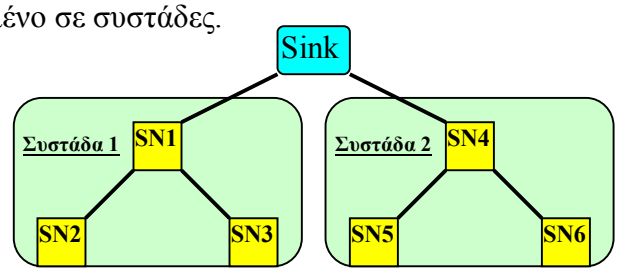


Σχήμα 1.8 Τοπολογία δέντρου-αστέρα.. Τρεις αστέρες, ένας με κέντρο τη Sink, ένας με κέντρο τον SN1 κι ένας με κέντρο τον SN4.

1.3.5 Ομαδοποίηση των κόμβων αισθητήρων (SNs) σε συστάδες

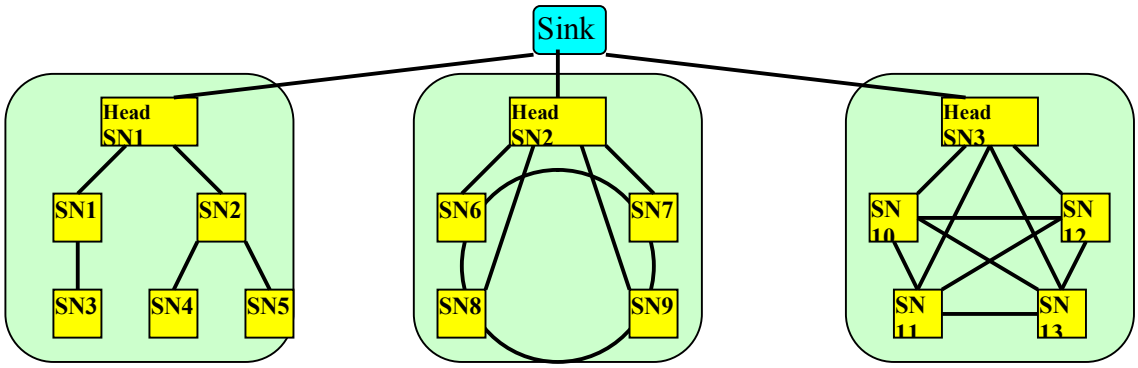
Στην πράξη κι ανεξάρτητα με τη χρησιμοποιούμενη σε ένα WSN τοπολογία είθισται οι SNs αυτού να χωρίζονται σε συστάδες. Κάθε συστάδα αποτελεί στην ουσία μία ομάδα SNs, ένας από τους οποίους τίθεται επικεφαλής αυτής κι ονομάζεται head SN της συστάδας. Ο head SN είναι ο μόνος SN της συστάδας, ο οποίος και διαθέτει ασύρματη ζεύξη με τη Sink, ενώ όλοι οι SNs της συστάδας συνδέονται ασύρματα μαζί του. Έτσι λοιπόν, αν ένας SN της συστάδας θέλει να αποστείλει τα δεδομένα των μετρήσεών του στη Sink, θα πρέπει να τα στείλει πρώτα στον head SN, ούτως ώστε αυτός να τα προωθήσει στη συνέχεια στη Sink. Με τον τρόπο αυτό, μπορούν να τοποθετηθούν στο WSN και SNs, οι οποίοι είναι εκτός της ασυρματικής εμβέλειας της Sink, αρκεί να διαθέτουν ασύρματη ζεύξη με κάποιον από τους head SNs του WSN. Από την άλλη, όποτε η Sink θέλει να στείλει κάποιο αίτημα σε κάποιον SN, τότε θα πρέπει να εντοπίσει τον head SN της συστάδας του και να στείλει το αίτημα σε αυτόν, προκειμένου αυτός με τη σειρά του να το μεταβιβάσει στον κατάλληλο SN. Όπως γίνεται κατανοητό, οι head SNs απαιτείται να έχουν αυξημένους διαθέσιμους πόρους για να διαχειριστούν τους SNs της συστάδας τους.

Τέλος, ένας head SN μπορεί να λειτουργεί και ως ένας απλός SN, δηλαδή να λαμβάνει δεδομένα από το περιβάλλον του και να τα αποστέλλει στη Sink. Δηλαδή, ως SN ενός WSN μπορεί να οριστεί οποιοσδήποτε SN αυτού. Ας δούμε για παράδειγμα το WSN του τελευταίου σχήματος, με συνδυασμένη τοπολογία δέντρου και αστέρα, χωρισμένο σε συστάδες.



Σχήμα 1.9: Τοπολογία αστέρα-δέντρου. Έχουμε δύο συστάδες, μία με head SN τον SN1 και μία με head SN τον SN4.

Σε ένα WSN, στο οποίο οι SNs αυτού είναι χωρισμένοι σε συστάδες μπορούν να εφαρμοστούν πολλές διαφορετικές τοπολογίες. Πιο συγκεκριμένα, μπορεί να χρησιμοποιηθεί διαφορετική τοπολογία για τους SNs κάθε συστάδας, όπως επίσης και για τους head SNs. Το γεγονός αυτό είναι ιδιαίτερα χρήσιμο, μιας και παρέχει μεγάλη ευελιξία στη σχεδίαση και διαχείριση του WSN.



Σχήμα 1.10: WSN με πολλές επί μέρους τοπολογίες. Χρησιμοποιείται τοπολογία αστέρα για τη σύνδεση των head SNs με τη Sink κι από μία διαφορετική τοπολογία για κάθε συστάδα SNs.

1.4 Αιτήματα και δεδομένα σε ένα WSN

Από τα μέχρι στιγμής γραφόμενα έχει γίνει κατανοητό, ότι η λειτουργία ενός WSN συνοψίζεται στη λήψη αιτημάτων από τους χρήστες του WSN και στην εξυπηρέτηση αυτών με βάση τα δεδομένα που βρίσκονται αποθηκευμένα στη Sink και τα οποία προέρχονται από τις μετρήσεις των αισθητήρων των SNs. Οι χρήστες ενός WSN, μπορεί να είναι είτε διάφορα προγράμματα, δηλαδή ειδικά σχεδιασμένο λογισμικό για τη διαχείριση του WSN, είτε φυσικά πρόσωπα, δηλαδή άνθρωποι στους οποίους κι έχει ανατεθεί η μελέτη κι εξαγωγή συμπερασμάτων από τα δεδομένα του WSN. Σε κάθε περίπτωση πάντως, θα πρέπει τα κατατιθέμενα προς τη Sink αιτήματα να έχουν μία συγκεκριμένη μορφή και γλώσσα γραφής, προκειμένου να γίνονται κατανοητά από τη Sink και τους SNs του WSN. Έτσι λοιπόν, τα αιτήματα αυτά έχουν παρόμοια μορφή με τα παραδοσιακά αιτήματα, που κατατίθενται σε μία Βάση Δεδομένων και τα οποία είναι γραμμένα στη γλώσσα SQL. Πιο συγκεκριμένα, η μορφή ενός αιτήματος που κατατίθεται προς τη Sink ενός WSN είναι η ακόλουθη:

SELECT <SN> FROM sensors WHERE < συνθήκη> ERROR TOLERANCE <όριο λάθους> WINDOW [<χρονική διάρκεια>]

Δηλαδή, σύμφωνα με την παραπάνω γενική μορφή ενός αιτήματος αναζητείται κάθε SN μεταξύ όλων των SNs ενός WSN (select <SN> from sensors), του οποίου οι μετρήσεις να ικανοποιούν μία συγκεκριμένη συνθήκη (where < συνθήκη>) και για τις μετρήσεις που έγιναν σε ένα συγκεκριμένο χρονικό διάστημα (window [<χρονική διάρκεια>]).

Τα αιτήματα, που κατατίθενται προς ένα WSN, διακρίνονται σε ακριβή και προσεγγιστικά. Τα ακριβή αιτήματα ενδιαφέρονται για τα πραγματικά δεδομένα των αισθητήρων, σε αντίθεση με τα προσεγγιστικά αιτήματα που μπορούν να ανεχτούν

ένα συγκεκριμένο περιθώριο λάθους στα δεδομένα των αισθητήρων. Έτσι λοιπόν, ένα προσεγγιστικό αίτημα είναι της μορφής:

SELECT <SN> FROM sensors WHERE < συνθήκη> ERROR TOLERANCE <όριο λάθους> WINDOW [<χρονική διάρκεια>]

Δηλαδή, σε ένα προσεγγιστικό αίτημα εισάγεται η έννοια του ανεκτού ορίου λάθους στα δεδομένα που λαμβάνονται ως απάντηση στο αίτημα αυτό. Οπότε, σε ένα προσεγγιστικό αίτημα αναζητείται κάθε SN μεταξύ όλων των SNs ενός WSN (select <SN> from sensors), του οποίου οι μετρήσεις να ικανοποιούν μία συγκεκριμένη συνθήκη (where < συνθήκη>) με ένα ανεκτό περιθώριο λάθους (error tolerance <όριο λάθους>) και για τις μετρήσεις που έγιναν σε ένα συγκεκριμένο χρονικό διάστημα (window [<χρονική διάρκεια>]). Τέλος, με δεδομένο ότι ένα WSN μπορεί να εξυπηρετεί ταυτόχρονα πολλά διαφορετικά αιτήματα, είναι πολύ σημαντικό να χρησιμοποιούνται διάφορες τεχνικές βελτιστοποίησης των αιτημάτων αυτών, προκειμένου να επιτευχθεί η ταχύτερη εξυπηρέτησή τους.

Πέραν των αιτημάτων που κατατίθενται σε ένα WSN, θα πρέπει και τα δεδομένα που προέρχονται από τις μετρήσεις των αισθητήρων των SNs, να έχουν κάποια συγκεκριμένη κοινή μορφή. Έτσι, η αποστολή των δεδομένων από τους SNs στη Sink γίνεται με εγγραφές δεδομένων, καθεμία από τις οποίες είναι της ακόλουθης μορφής:

{ti, vi1, vi2, ... , vim}

όπου ti είναι η χρονοσφραγίδα αυτής της εγγραφής, δηλαδή η χρονική στιγμή κατά την οποία δημιουργήθηκε αυτή η εγγραφή. Καθένα από τα vij (1<=j<=m) αναπαριστά μετρήσεις των αισθητήρων διαφόρων τύπων, όπως για παράδειγμα μετρήσεις θερμοκρασίας, υγρασίας και άλλες. Όλες οι εγγραφές δεδομένων πρέπει να έχουν το ίδιο μήκος, το οποίο και καθορίζεται από τους διαχειριστές του WSN.

1.5 Πλεονεκτήματα και μειονεκτήματα WSNs

Τα κυριότερα πλεονεκτήματα των WSNs είναι τα ακόλουθα:

1. Στα WSNs δεν υπάρχει η ανάγκη ύπαρξης δομημένης καλωδίωσης, κάτι που καθιστά την επικοινωνία εξαιρετικά ευέλικτη.

2. Λόγω του χαμηλού κόστους των SNs, οι τελευταίοι μπορούν να αναπτυχθούν με μεγάλη πυκνότητα στην περιοχή παρατήρησης.

3. Είναι εύκολη η εγκατάστασή τους, ακόμη και σε χώρους χωρίς καμία απολύτως υποδομή.

4. Είναι εύκολη η συντήρηση και η δυναμική τους επέκταση, ανάλογα με ρις ανάγκες της εκάστοτε εφαρμογής.

Πέρα από τα παραπάνω πλεονεκτήματα, τα WSNs παρουσιάζουν κι ένα σύνολο από περιορισμούς, οι οποίοι είναι εντελώς διαφορετικοί σε σύγκριση με αυτούς που παρουσιάζονται στα παραδοσιακά δίκτυα. Οι περιορισμοί αυτοί είναι οι ακόλουθοι:

1. Καταρχήν, το πιο σημαντικό μειονέκτημα των WSNs είναι αυτό της ενέργειας. Πιο συγκεκριμένα, η ενεργειακή αυτονομία των WSNs είναι περιορισμένη, δεδομένου ότι στις περισσότερες περιπτώσεις η τροφοδοσία αυτών γίνεται με απλές μπαταρίες τύπου ΑΑ ή ΑΑΑ. Έρευνες που έχουν γίνει τα τελευταία χρόνια, έδειξαν ότι αν ένας SN λειτουργεί συνεχώς με τη μέγιστη απαιτούμενη κατανάλωση ενέργειας, τότε η διάρκεια ζωής αυτού είναι κατά μέσο όρο μόλις τέσσερις μέρες. Για να εξασφαλιστεί λοιπόν, η μακροχρόνια λειτουργία ενός WSN, θα πρέπει αυτές οι τέσσερις μέρες να διασκορπιστούν σε αρκετά χρόνια ζωής. Αν και κάτι τέτοιο, για τα υπόλοιπα συστήματα είναι κάτι που δεν πρέπει να λαμβάνουν υπόψη, στα WSNs αποτελεί κάτι πολύ σημαντικό.

2. Ένα ακόμη μειονέκτημα των WSNs είναι η δυσκολία που παρουσιάζει η τοποθέτησή και η συντήρησή τους, μιας κι όπως έχει ήδη αναφερθεί σε πολλές περιπτώσεις απαιτείται η λειτουργία τους σε δυσπρόσιτα κι αφιλόξενα για τον άνθρωπο περιβάλλοντα. Ως αποτέλεσμα, κατά την εγκατάσταση ενός WSN απαιτείται η χρήση εξειδικευμένων τεχνικών και μηχανημάτων, προκειμένου να εξασφαλιστεί η σωστή εγκατάσταση και η απρόσκοπτη λειτουργία αυτού.

3. Επιπρόσθετα, τα WSNs πρέπει να αντιμετωπίσουν και το πρόβλημα του “coverage”, δηλαδή το πόσο καλά παρακολουθείται από τους SNs, η περιοχή στην οποία έχει εγκατασταθεί το WSN.

4. Μία ακόμη δυσκολία, που πρέπει να αντιμετωπιστεί στα WSNs είναι η απώλεια πακέτων δεδομένων, κατά την επικοινωνία μεταξύ των SNs. Πιο συγκεκριμένα, όποτε λαμβάνει χώρα απώλεια πακέτων δεδομένων θα πρέπει να αναζητείται η αιτία που την προκάλεσε ούτως ώστε να αντιμετωπιστεί και να μην επαναληφθεί μελλοντικά. Όμως, είναι πολύ δύσκολο να καθοριστεί για ποιον ακριβώς λόγο χάθηκε κάποιο πακέτο δεδομένων σε έναν SN. Εντούτοις, κάποιοι πιθανοί λόγοι μπορεί να είναι η έλλειψη ενέργειας ή ασφάλειας, η υπερχείλιση της ουράς μεταφοράς δεδομένων, η συμφόρηση στο δίκτυο, καθώς και διάφοροι εξωγενείς παράγοντες.

5. Οι SNs όντας μικροσκοπικές ηλεκτρονικές διατάξεις υφίστανται σημαντικούς περιορισμούς στους διαθέσιμους πόρους. Έτσι λοιπόν, ένα ακόμη ζήτημα που καλούμαστε να χειριστούμε στα WSNs είναι η περιορισμένη υπολογιστική ισχύς και η μικρή ενσωματωμένη μνήμη των SNs.

Λόγω όλων των ανωτέρω, καθώς και διαφόρων άλλων περιορισμών και προβλημάτων, που έχουν να αντιμετωπίσουν τα WSNs υπάρχει ανάγκη για καινοτόμα συστήματα, πρωτόκολλα κι αλγορίθμους, που να δίνουν λύση στα ζητήματα αυτά.

1.6 Εφαρμογές Ασύρματων Δικτύων Αισθητήρων

Είναι πραγματικά ατελείωτη η λίστα των εφαρμογών των Ασύρματων Δικτύων Αισθητήρων (WSNs), ενώ πολλές από αυτές μας είναι ήδη οικείες καθώς ανταποκρίνονται στις συνήθεις δραστηριότητες και ανάγκες μας. Όπως αναφέρει άλλωστε χαρακτηριστικά ο καθηγητής του UCLA William Kaiser « Το Διαδίκτυο άλλαξε τον τρόπο με τον οποίο δουλεύουμε με τους υπολογιστές. Τα WSNs θα αλλάξουν τον τρόπο με τον οποίο ζούμε καθημερινά». Το πεδίο των εφαρμογών των WSNs περιλαμβάνει τις μετρήσεις ακριβείας πολλών ατμοσφαιρικών και μετεωρολογικών παραμέτρων, την επιτήρηση δασών, υδροβιοτόπων, θερμοκηπίων και γενικά αγροτικών καλλιεργειών για έλεγχο υγρασίας, θερμοκρασίας, πίεσης και ωρίμανσης των καρπών, την επιτήρηση υγρών στοιχείων για ρύπους ή έλεγχο ακραίων φαινομένων όπως οι πλημμύρες, την επιτήρηση του βιομηχανικού περιβάλλοντος για την εξασφάλιση επιθυμητών συνθηκών της παραγωγικής διαδικασίας, τις στοιχειώδεις ρυθμίσεις λειτουργιών σε κτίρια όπως θέρμανση, φωτισμός και συναγερμοί, την επιτήρηση στρατιωτικών εγκαταστάσεων, τον εντοπισμό αντικειμένων σε αρχαιολογικές έρευνες, τον έλεγχο της σεισμικής δραστηριότητας, τη δημιουργία περιβαλλόντων εικονικής πραγματικότητας, την επόπτευση των κατασκευών καθώς και πολλές άλλες. Ενδεικτικά περιγράφονται λεπτομερέστερα στη συνέχεια, μερικές από αυτές:

1.6.1 Έλεγχος τεχνικών κατασκευών

Οι κατασκευές υπόκεινται σε μακροπρόθεσμες καταπονήσεις λόγω εκτεταμένης λειτουργικής ζωής, διαβρώσεων, τριβών μεταξύ τους και σεισμικών δονήσεων. Είναι σημαντικό να επεκτείνουμε την ενεργή ζωή των υποδομών μας, μέσω της συλλογής ποιοτικών πληροφοριών για την κατάστασή τους. Οι μηχανικοί μπορούν έτσι να πραγματοποιούν προληπτικές επισκευές βασιζόμενοι περισσότερο σε μετρήσεις απόδοσης και λιγότερο σε προγραμματισμένες συντηρήσεις. Χαρακτηριστικό παράδειγμα αποτελεί η τοποθέτηση ενός WSN στη γέφυρα που συνδέει το Ρίο με το Αντίρριο για την παρακολούθηση μίας πληθώρας παραμέτρων που αφορούν τη συντήρηση και τη στατικότητα αυτής.

1.6.2 Περιβαλλοντολογική παρακολούθηση

Υπάρχει ένα τεράστιο πλήθος περιβαλλοντολογικών εφαρμογών των WSN, μία από τις οποίες αποτελεί η παρακολούθηση της χλωρίδας και της πανίδας. Για παράδειγμα, στο νησί Great Duck κοντά στην ακτή του Maine των ΗΠΑ, οι επιστήμονες χρησιμοποιούν ένα πρότυπο δίκτυο αισθητήρων για να ελέγχουν τις δυσπρόσιτες φωλιές των θαλάσσιων πουλιών. Αισθητήριοι κόμβοι σε μέγεθος νομίσματος τοποθετούνται στις σπηλιές, όπου βρίσκονται οι φωλιές και ελέγχουν συνεχώς μεγέθη, όπως η φωτεινότητα, η θερμοκρασία και η βαρομετρική πίεση. Οι μετρήσεις αναμεταδίδονται σε τοπικούς υπολογιστές και κατόπιν, σε πραγματικό χρόνο, στο εργαστήριο για την επεξεργασία και την εξαγωγή συμπερασμάτων. Με τον τρόπο αυτό, οι βιολόγοι παίρνουν τις πληροφορίες που θέλουν για την παρατήρηση των πουλιών και την προστασία του βιότοπου, με την ελάχιστη δυνατή ανθρώπινη παρέμβαση.

1.6.3 Γεωργία ακριβείας

Πολλά WSNs διαθέτουν αισθητήρες, που τους επιτρέπουν να παρακολουθούν τα ακριβή επίπεδα του πόσιμου νερού, τη διάβρωση του εδάφους, τη θερμοκρασία, την υγρασία και το βαθμό μόλυνσης του αέρα σε πραγματικό χρόνο. Αυτά τα WSNs μπορούν να χρησιμοποιηθούν στη γεωργία, προκειμένου να συντελέσουν στην ποιοτική και ποσοτική αναβάθμιση της αγροτικής παραγωγής. Ήδη σε κάποια περιοχή της Γαλλίας, έχει εγκατασταθεί ένα WSN με την ονομασία «έξυπνος αμπελώνας» που χρησιμοποιείται για το σκοπό αυτό. Στον «έξυπνο αμπελώνα», οι αισθητήρες ελέγχουν τις θρεπτικές ουσίες σε φυτά και έδαφος, κρατούν τις αμπέλους απαλλαγμένες από τα παράσιτα, εντοπίζουν την υγρασία και ειδοποιούν για τις περιοχές όπου απαιτείται πότισμα. Επίσης, το πρόγραμμα λαμβάνει πληροφορίες από εκείνους που φροντίζουν τα αμπέλια, καλλιεργούν το χώμα και μαζεύουν τα σταφύλια καθώς και δεδομένα για τις ανάγκες των ιδιοκτητών των αμπελώνων, των οινοπαραγωγών και των πωλητών κρασιού. Με τον τρόπο αυτό επιτυγχάνεται, τόσο η αύξηση της παραγωγής, όσο και η ποιοτική βελτίωση του παραγόμενου κρασιού.

1.6.4 Πυρανίχνευση

Ένα WSN εγκατεστημένο σε μία δασική έκταση, με κόμβους αισθητήρες (SNs) ιδανικούς για την ανίχνευση της φωτιάς μπορεί να προσφέρει έγκαιρη και έγκυρη πυρανίχνευση και πυρασφάλεια. Πιο συγκεκριμένα, κάθε SN διαθέτει αισθητήρες μέτρησης της θερμοκρασίας και της σχετικής υγρασίας, καθώς και αισθητήρες ανίχνευσης της φλόγας. Η ενεργειακή τροφοδοσία του κάθε SN μπορεί να παρέχεται είτε μέσω του φλοιού του δέντρου, στο οποίο και βρίσκεται εγκαταστημένος είτε μέσω φωτοβολταϊκών συστημάτων. Αν μάλιστα λάβουμε υπόψιν μας, το γεγονός ότι οι SNs αυτού του WSN μπορεί να παραμείνουν αδρανείς για μεγάλο χρονικό διάστημα, μπορούμε να εφαρμόσουμε διάφορους μηχανισμούς εξοικονόμησης ενέργειας εξασφαλίζοντας τη μακροχρόνια λειτουργία του δικτύου. Τέλος το ανωτέρω WSN μπορεί να συνδυαστεί και με ένα μη επανδρωμένο αεροσκάφος (U.A.V.) , το οποίο και θα πραγματοποιεί πτήσεις πάνω από τη δασική περιοχή με σκοπό την επιβεβαίωση της πυρκαγιάς.

1.6.5 Ανίχνευση πλημμύρων

Οι πλημμύρες μπορεί να προκληθούν από τις παρατεταμένες βροχοπτώσεις μιας θύελλας ή μιας καταιγίδας, τη γρήγορη τήξη μεγάλων ποσοτήτων χιονιού, φουσκωμένους ποταμούς εξαιτίας μεγάλων βροχοπτώσεων στις πηγές αυτών, καθώς και από κατάρρευση φραγμάτων ή αναχωμάτων κατασκευασμένων από τον άνθρωπο. Ανάλογα με το πιθανό αίτιο πρόκλησης μίας πλημμύρας σε κάποια συγκεκριμένη γεωγραφική περιοχή, μπορεί να εγκατασταθεί σε αυτήν ένα WSN για την έγκαιρη ανίχνευσή κι αντιμετώπισή της. Ένα παράδειγμα συστήματος ανίχνευσης πλημμύρων είναι το σύστημα ALERT, το οποίο και αναπτύχθηκε στις ΗΠΑ. Στο σύστημα αυτό χρησιμοποιήθηκαν διάφοροι τύποι αισθητήρων, όπως μέτρησης της στάθμης της βροχόπτωσης, της θερμοκρασίας, της υγρασίας και της στάθμης του νερού. Τα δεδομένα από αυτές τις μετρήσεις αποστέλλονται σε μία ή περισσότερες κεντρικές αποθήκες δεδομένων, όπου και υφίστανται επεξεργασία για την αποτελεσματική πρόληψη, καθώς και την αντιμετώπιση των πλημμύρων.

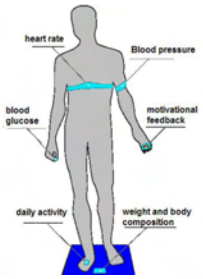
1.6.6 Εφαρμογές υγείας

Κάποιες από τις εφαρμογές των WSNs στον τομέα της υγείας περιλαμβάνουν τον εντοπισμό και την παρακολούθηση των γιατρών και των ασθενών σε ένα νοσοκομείο, την τηλεπρακολούθηση των φυσιολογικών δεδομένων ενός ανθρώπου, την παροχή μέσων αλληλεπίδρασης για άτομα με ειδικές ανάγκες, την παρακολούθηση ασθενών, τη διάγνωση, τη διαχείριση φαρμάκων σε νοσοκομεία, καθώς και πολλές άλλες. Για παράδειγμα, μπορεί να προσαρτηθεί σε κάθε ασθενή ενός νοσοκομείου ένας ή περισσότεροι SNs. Καθένας από αυτούς επιτελεί κάποιο συγκεκριμένο καθήκον, όπως για παράδειγμα η παρακολούθηση της καρδιακής λειτουργίας ή της αναπνοής ή της αρτηριακής πίεσης του ασθενή. Οι παρεχόμενες από τους SNs πληροφορίες μπορούν να παρέχουν άμεση προειδοποίηση στο νοσηλευτικό προσωπικό σε περίπτωση έκτακτου περιστατικού, όπως επίσης και να βοηθήσουν τους ιατρούς στην επιλογή της καταλληλότερης θεραπευτικής αγωγής.

1.6.7 Στρατιωτικές εφαρμογές

Όπως αναφέραμε και νωρίτερα, η γέννηση των WSNs προήλθε από διάφορα στρατιωτικά ερευνητικά προγράμματα. Στις μέρες μας, τα WSNs χρησιμοποιούνται για την επιτήρηση στρατιωτικών εγκαταστάσεων και των συνόρων ενός κράτους, όπως επίσης και για την αναγνώριση των φίλιων πυρών στο πεδίο της μάχης, την εκτίμηση των ζημιών μετά από μία μάχη, τη στόχευση, την παρακολούθηση του πεδίου της μάχης, την αναγνώριση των εχθρικών δυνάμεων και του εδάφους και πολλά άλλα. Πιο συγκεκριμένα, ένα WSN μπορεί να χρησιμοποιηθεί στο πεδίο της μάχης για τη συλλογή δεδομένων που αφορούν την κατάσταση, τη θέση και τον οπλισμό των στρατευμάτων σε πραγματικό χρόνο. Τα δεδομένα αυτά αποστέλλονται στους αρχηγούς του στρατεύματος, βοηθώντας τους στη λήψη των κατάλληλων αποφάσεων, κατά τη διάρκεια της μάχης.

Εικόνα 1.6 :
SNs σε γέφυρα για έλεγχο της στατικότητας αυτής



Εικόνα 1.7 :
SNs σε ανθρώπινο σώμα για παρακολούθηση υγείας

Εικόνα 1.8 :
Στιγμιότυπο από WSN για στρατιωτική χρήση

1.6.8 Αυτοκίνητο του μέλλοντος

Τα WSNs θα αλλάξουν ριζικά την εικόνα και τη χρήση των αυτοκινήτων. Πιο συγκεκριμένα, κάθε αυτοκίνητο θα διαθέτει ένα μεγάλο πλήθος SNs, μέσω των οποίων θα έχει την ικανότητα να μετακινείται από μόνο του χωρίς να χρειάζεται η ανθρώπινη παρέμβαση. Επιπλέον, θα επικοινωνεί με άλλα οχήματα, με δίκτυα διαχείρισης της κυκλοφορίας, με δίκτυα ασφαλείας και φυσικά με το Διαδίκτυο. Με τον τρόπο αυτό, θα επιτευχθεί σημαντική μείωση των τροχαίων ατυχημάτων και θα εξοικονομηθούν εκατομμύρια ανθρωποώρες. Είναι χαρακτηριστική η πρόσφατη ανακοίνωση της αυτοκινητοβιομηχανίας Toyota, σύμφωνα με την οποία μέσα στα επόμενα χρόνια θα υπάρχουν πάνω από πενήντα δισεκατομμύρια αισθητήρες στα κατασκευαζόμενα από αυτήν οχήματα.

1.6.9 Ιχθυοκαλλιέργεια

Μία από τις πολλές εφαρμογές των WSNs είναι και αυτή της ιχθυοκαλλιέργειας. Πιο συγκεκριμένα, η εταιρία AKVA Group έχει κατασκευάσει ένα WSN για τον έλεγχο του ταΐσματος σε μία μονάδα ιχθυοκαλλιέργειας. Αυτό το WSN αποτελείται από μία κάμερα επιφανείας, δύο υποβρύχιες κάμερες και αισθητήρες θερμοκρασίας, θαλασσίων ρευμάτων, οξυγόνου και κόκκων. Οι παρεχόμενες από τους αισθητήρες μετρήσεις παρέχονται στους διαχειριστές της μονάδας ιχθυοκαλλιέργειας βοηθώντας τους στη λήψη των αποφάσεων, που αφορούν την ώρα που πρέπει να λαμβάνει χώρα το τάισμα των ψαριών, καθώς επίσης την ποσότητα και τη διασπορά της παρεχόμενης τροφής. Με τον τρόπο αυτό, επιτυγχάνεται η υλοποίηση της απλής αλλά σημαντικής διαδικασίας του ταΐσματος των ψαριών, με επιστημονικό τρόπο βελτιώνοντας τόσο ποσοτικά, όσο και ποιοτικά την παραγωγή.

Η προαναφερθείσα αναφορά μερικών μόνο από τις εφαρμογές των WSNs, καθιστά σαφή τη ραγδαία αύξηση του ρυθμού διείσδυσης της τεχνολογίας αυτής, σε ένα ευρύ φάσμα βιομηχανικών εφαρμογών. Όμως, παρά τη σημαντική αυτή αύξηση , το ποσοστό ανάπτυξης των WSNs σε σύγκριση με τις ανάγκες της αγοράς παραμένει μικρότερο του πέντε τοις εκατό, γεγονός που εν πολλοίς οφείλεται στους περιορισμούς της τεχνολογίας των ασύρματων κόμβων που συνθέτουν ένα WSN.

1.7 Λειτουργία ενός WSN

Συμπερασματικά λοιπόν, η λειτουργία ενός WSN έχει ως εξής: Οι SNs βρίσκονται εγκατεστημένοι σε μία γεωγραφική περιοχή και λαμβάνουν μέσω των αισθητήρων τους, μετρήσεις διάφορων φυσικών ή περιβαλλοντικών παραμέτρων. Εν συνεχεία, αποθηκεύουν τα δεδομένα αυτά στην ενσωματωμένη μνήμη, που διαθέτουν για το σκοπό αυτό και τα αποστέλλουν στη Sink. Η Sink συλλέγει τα δεδομένα που προέρχονται από τις μετρήσεις των αισθητήρων όλων των SNs και τα αποθηκεύει σε μία Βάση Δεδομένων. Έτσι, όποτε η Sink παραλαμβάνει ένα καινούργιο αίτημα από κάποιο χρήστη, ανατρέχει στη μνήμη της για να βρει τα δεδομένα που αποτελούν απάντηση στο αίτημα αυτό. Σε περίπτωση, κατά την οποία τα αποθηκευμένα δεδομένα στη Βάση Δεδομένων της Sink είναι ανεπαρκή για την απάντηση κάποιου αιτήματος, τότε η Sink διαβιβάζει το συγκεκριμένο αίτημα στους SNs, προκειμένου αυτοί να της αποστείλουν τα ζητούμενα δεδομένα.

1.8 Γενική επισκόπηση του WSN της παρούσας μελέτης

Όπως έχει ήδη αναφερθεί, σκοπός της εργασίας μας είναι η παρουσίαση διαφόρων τεχνικών, οι οποίες μπορούν να εφαρμοστούν σε ένα WSN, με απώτερο σκοπό την αντιμετώπιση των περιορισμών αυτού και τη συνολική βελτιστοποίηση της λειτουργίας του. Προτού όμως προχωρήσουμε στην παρουσίαση κι ανάλυση των βελτιστοποιήσεων αυτών, θα παρουσιάσουμε πρώτα μία γενική επισκόπηση του WSN, πάνω στο οποίο θα εργαστούμε.

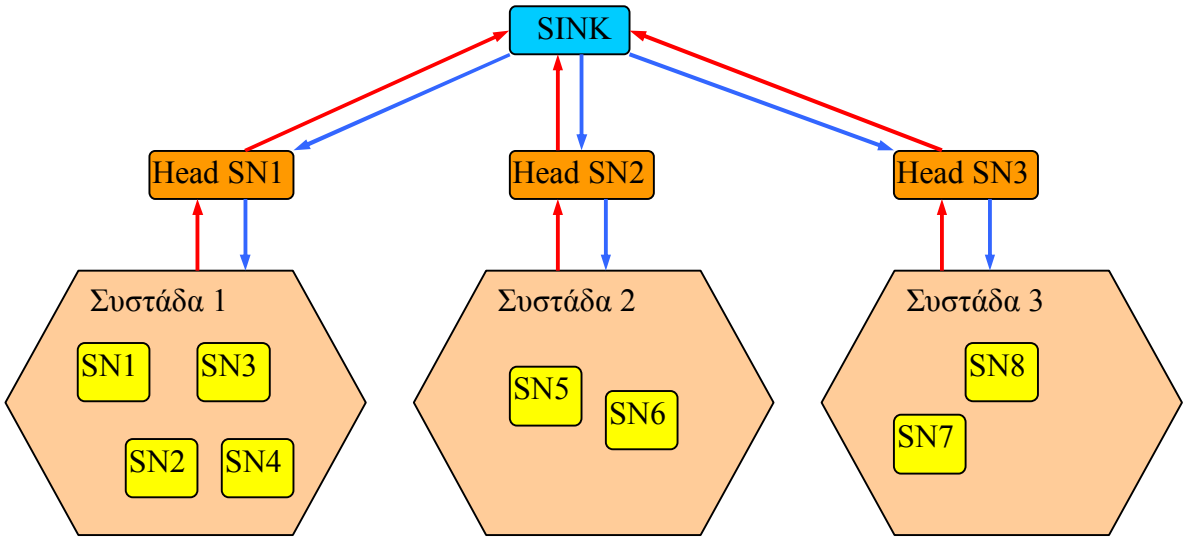
1.8.1 Αρχιτεκτονική

Πιο συγκεκριμένα, θεωρούμε ένα WSN αποτελούμενο από μία Sink και από ένα μεγάλο πλήθος SNs.

- Οι SNs βρίσκονται διάσπαρτα εγκατεστημένοι στην περιοχή παρακολούθησης και υφίστανται περιορισμούς όσον αφορά τους διαθέσιμους πόρους. Λαμβάνουν ανά προκαθορισμένα τακτά χρονικά διαστήματα τις μετρήσεις των αισθητήρων τους, τις οποίες κι αποθηκεύουν στη μεγάλης χωρητικότητας μνήμη NAND flash, που διαθέτουν για το σκοπό αυτό. Τα δεδομένα αυτά διατηρούνται στη μνήμη, όσο υπάρχει ελεύθερος χώρος σε αυτήν και για ένα συγκεκριμένο μέγιστο χρονικό διάστημα. Παράλληλα, αποστέλλουν τα δεδομένα των μετρήσεών τους στη Sink. Όλοι οι SNs είναι χωρισμένοι σε συστάδες, με καθεμία από τις συστάδες να διαθέτει έναν head SN. Ο τελευταίος έχει αυξημένη διαθεσιμότητα σε πόρους, σε σχέση με τους απλούς SNs.

- Η Sink έχει πλήρη επάρκεια σε διαθέσιμους πόρους και βρίσκεται σε μία εύκολα προσβάσιμη κι ασφαλή τοποθεσία, πλησίον του WSN. Εναλλακτικά, μπορεί να συνδέεται με τους SNs μέσω του Διαδικτύου ή μέσω κάποιου WAN. Παραλαμβάνει τα δεδομένα που προέρχονται από τις μετρήσεις των αισθητήρων των SNs, τα οποία κι αποθηκεύει στη μεγάλης χωρητικότητας μνήμη της. Επίσης παραλαμβάνει τα αιτήματα των χρηστών, τα οποία και προσπαθεί να εξυπηρετήσει με βάση τα δεδομένα που βρίσκονται αποθηκευμένα στη μνήμη της. Αν τα δεδομένα αυτά είναι επαρκή για την απάντηση κάποιου αιτήματος, τότε η Sink εξυπηρετεί άμεσα το αίτημα αυτό χωρίς να το προωθήσει στο WSN. Ειδάλλως, προωθεί το αίτημα στους head SNs, προκειμένου να λάβει από αυτούς όλα τα αιτούμενα δεδομένα.

Τα όσα αναφέραμε απεικονίζονται στο σχήμα, που ακολουθεί, στο οποίο τα μπλε βέλη αντιστοιχούν σε αποστολή δεδομένων από τους SNs στη Sink και τα κόκκινα βέλη αντιστοιχούν στην αποστολή αιτημάτων από τη Sink στους SNs. Οι SNs κάθε συστάδας μπορούν να έχουν οποιαδήποτε από τις τοπολογίες, που παρουσιάσαμε νωρίτερα.



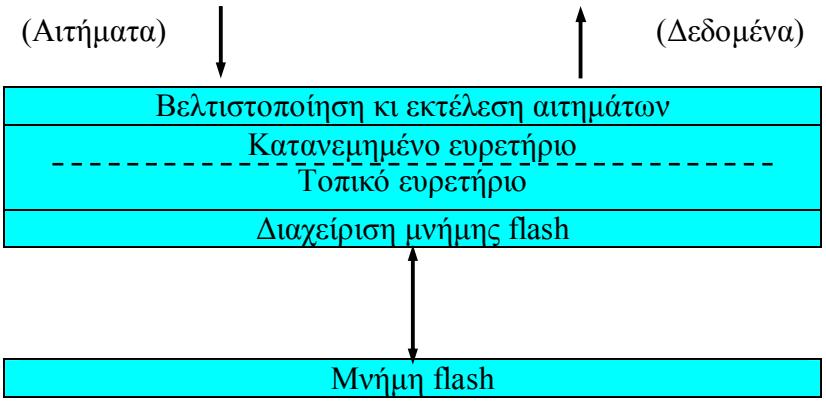
Σχήμα 1.11: Αρχιτεκτονική WSN

1.8.2 Στοίβα Διαχείρισης Δεδομένων

Πρώτα θα περιγράψουμε αναλυτικά τον τρόπο με τον οποίο λαμβάνει χώρα η επεξεργασία ενός αιτήματος από το σύστημά μας. Έτσι λοιπόν, κάθε φορά, που η Sink παραλαμβάνει ένα καινούργιο αίτημα από κάποιο χρήστη εφαρμόζει πρώτα διάφορες τεχνικές βελτιστοποίησης πάνω σε αυτό. Στη συνέχεια, η Sink επεξεργάζεται το αίτημα αυτό κι εάν μπορεί να το εξυπηρετήσει με βάση τα αποθηκευμένα στη μνήμη της δεδομένα, τότε επιστρέφει άμεσα τα αιτούμενα δεδομένα. Διαφορετικά μεταβιβάζει το αίτημα αυτό στους head SNs, προκειμένου να λάβει από αυτούς τα ζητούμενα δεδομένα. Προκειμένου μάλιστα, να μειωθεί το πλήθος των αποστολών του αιτήματος εντός του WSN υπάρχει μεταξύ όλων των head SNs ένα κατανεμημένο ευρετήριο, το οποίο κι αποτελεί στην ουσία έναν κατάλογο των δεδομένων που βρίσκονται αποθηκευμένα στους head SNs. Έτσι, το αίτημα αποστέλλεται μόνον στους head SNs, οι οποίοι κι ενδεχομένως σύμφωνα με το κατανεμημένο ευρετήριο, διαθέτουν τα ζητούμενα δεδομένα. Κάθε head SN που λαμβάνει ένα αίτημα, το αποστέλλει στους SNs της συστάδας του. Και πάλι η αποστολή του αιτήματος γίνεται μόνον στους SNs που μπορεί να διαθέτουν τα ζητούμενα δεδομένα, με βάση ένα κατανεμημένο ευρετήριο μεταξύ όλων των SNs της συστάδας. Κάθε SN που παραλαμβάνει το αίτημα χρησιμοποιεί ένα τοπικό ευρετήριο για να διαπιστώσει αν τα ζητούμενα από το αίτημα δεδομένα υπάρχουν αποθηκευμένα στη NAND flash μνήμη του κι εάν ισχύει κάτι τέτοιο, τότε χρησιμοποιεί τεχνικές διαχείρισης της μνήμης flash για να ανακτήσει τα δεδομένα αυτά.

Για να γίνει περισσότερο κατανοητή η παραπάνω διαδικασία θα την περιγράψουμε ξανά με βάση τη στοίβα διαχείρισης αιτημάτων και δεδομένων, η οποία και απεικονίζεται στο επόμενο σχήμα. Πιο συγκεκριμένα, κάθε φορά που ένας κόμβος του WSN παραλαμβάνει ένα αίτημα αρχίζει κι εκτελεί από πάνω προς τα κάτω τις ενέργειες, που καθορίζονται στα επίπεδα της συγκεκριμένης στοίβας.

Σχήμα 1.12 : Στοίβα Διαχείρισης Αιτημάτων και Δεδομένων σε κόμβο ενός WSN



Έτσι λοιπόν, όταν ένας κόμβος του SN παραλάβει ένα καινούργιο αίτημα, αρχικά εκτελεί το ανώτερο επίπεδο της στοίβας, δηλαδή εφαρμόζει διάφορες τεχνικές βελτιστοποίησης του αιτήματος κι εν συνεχεία αρχίζει την εκτέλεση αυτού. Στη συνέχεια, χρησιμοποιώντας ένα κατανεμημένο ευρετήριο αναζητεί ποιοι από τους κόμβους του WSN, με τους οποίους συνδέεται μπορεί να διαθέτουν τα ζητούμενα δεδομένα κι αποστέλλει το αίτημα αυτό στους κόμβους αυτούς. Παράλληλα, χρησιμοποιώντας ένα τοπικό ευρετήριο εξετάζει αν διαθέτει τα αιτούμενα δεδομένα στη μνήμη του κι εάν ισχύει κάτι τέτοιο χρησιμοποιεί διάφορες μεθόδους διαχείρισης της μνήμης flash για να ανακτήσει από την τελευταία τα αιτούμενα δεδομένα.

Με βάση την παραπάνω στοίβα, θα μελετήσουμε στην παρούσα εργασία το πρόβλημα της επεξεργασίας των αιτημάτων σε ένα WSN, από τρεις διαφορετικές σκοπιές:

1. Βελτιστοποίηση και κατανεμημένη επεξεργασία αιτημάτων
2. Τεχνικές τοπικής και κατανεμημένης ευρετηρίασης
3. Διαχείριση αποθήκευσης σε μνήμη flash

Κεφάλαιο 2

Διαχείριση Αποθήκευσης σε Μνήμη Flash

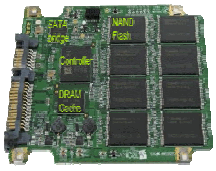
Περιεχόμενα

Όπως αναφέρθηκε στο προηγούμενο κεφάλαιο, οι SNs ενός WSN είναι εξοπλισμένοι με μνήμη flash τύπου NAND. Σε αντίθεση με τα παραδοσιακά μέσα αποθήκευσης, η μνήμη flash επιδεικνύει ένα σύνολο ιδιαίτερων χαρακτηριστικών, τα οποία κι απαιτούν νέες εξειδικευμένες τεχνικές διαχείρισης αποθήκευσης. Σε αυτό το κεφάλαιο, θα προτείνουμε τρεις τεχνικές διαχείρισης αποθήκευσης σε μνήμη flash, οι οποίες κι όπως θα δούμε στη συνέχεια βασίζονται στη διαμέριση της μνήμης, στην εύρεση συσχετίσεων μεταξύ των δεδομένων και στην επαναχρησιμοποίηση των δεδομένων. Καθεμία από αυτές τις τεχνικές, αποσκοπεί στη βελτίωση της λειτουργικότητας ενός WSN. Πρώτα όμως, θα πρέπει να παρουσιάσουμε τη μνήμη flash τύπου NAND και τα χαρακτηριστικά αυτής, που την καθιστούν τόσο διαφορετική από τα υπόλοιπα μέσα αποθήκευσης.

2.1 Μνήμη flash τύπου NAND

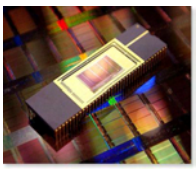
Από το 1987, οπότε και παρουσιάστηκε για πρώτη φορά το πρωτότυπο της μνήμης flash τύπου NAND, αυτός ο τύπος μνήμης χρησιμοποιείται ολοένα και περισσότερο ως μέσο αποθήκευσης δεδομένων σε ένα ευρύ φάσμα υπολογιστικών συσκευών, όπως είναι τα MP3 players, τα PDAs, τα κινητά τηλέφωνα και οι ψηφιακές κάμερες. Πρόσφατα μάλιστα, αρκετοί κατασκευαστές H/Y άρχισαν να λανσάρουν διάφορους καινούργιους φορητούς και σταθερούς υπολογιστές, στους οποίους κι αντικατέστησαν τους μαγνητικούς δίσκους από τη μνήμη flash τύπου NAND. Παράλληλα, συνεχίζεται με αμείωτο ρυθμό η εξέλιξη των τεχνικών χαρακτηριστικών της, όπως επίσης και η ανάπτυξη πληθώρας νέων τεχνικών διαχείρισης της. Η ευρεία αυτή αποδοχή της μνήμης flash οφείλεται σε ένα σύνολο χαρακτηριστικών της, τα οποία την καθιστούν ελκυστικότερη από όλα τα υπόλοιπα διαθέσιμα μέσα αποθήκευσης. Όσο μάλιστα η χωρητικότητά της αυξάνεται και η τιμή της μειώνεται, τόσο περισσότερο γίνεται σαφές, ότι αυτή η αυξανόμενη τάση στη χρησιμοποίηση της μνήμης flash τύπου NAND θα συνεχιστεί και τα επόμενα χρόνια.

Εικόνα 2.1: Μητρική κάρτα με μνήμη flash



Εικόνα 2.2: USB Flash Memory Stick

Εικόνα 2.3: NAND Flash μνήμη



Εικόνα 2.4: Κάρτα μνήμης flash 64GB

2.1.1 Δομή

Πιο συγκεκριμένα, η μνήμη NAND flash αποτελείται από ημιαγωγούς, που αποθηκεύουν bits πληροφορίας «0» ή «1», ακόμη κι όταν δεν τροφοδοτούνται με ρεύμα. Κάθε μνήμη flash τύπου NAND είναι χωρισμένη σε μπλοκ, καθένα από τα οποία είναι χωρισμένο σε τομείς. Κάθε τομέας με τη σειρά του είναι χωρισμένος σε σελίδες, οι οποίες και συνιστούν τη δομική μονάδα αποθήκευσης της μνήμης flash. Κάθε σελίδα μνήμης απαρτίζεται από τα δεδομένα που βρίσκονται αποθηκευμένα σε αυτήν, πληροφορίες για τα δεδομένα αυτά και τη σελίδα γενικότερα και τέλος

κάποιον δείκτη σε θέση μνήμης. Οι σελίδες μνήμης διακρίνονται σε σελίδες δεδομένων και σελίδες ευρετηρίου. Έτσι λοιπόν και η μνήμη flash χωρίζεται σε δύο μέρη: ένα, στο οποίο αποθηκεύονται οι σελίδες δεδομένων κι ένα στο οποίο κι αποθηκεύονται οι σελίδες ευρετηρίου.

2.1.1 Πλεονεκτήματα

Τα κυριότερα πλεονεκτήματα της μνήμης flash, τα οποία και την καθιστούν τόσο ελκυστική ως μέσο αποθήκευσης σε σύγκριση με τον κυριότερο ανταγωνιστή της που είναι ο μαγνητικός δίσκος, είναι τα παρακάτω:

- έχει μικρό βάρος
- παρουσιάζει αυξημένη αντοχή στις δονήσεις
- απαιτεί ελάχιστη κατανάλωση ενέργειας
- έχει χαμηλές στάθμες θορύβου
- είναι ταχεία σε εκτέλεση
- είναι μη πτητική και διατηρεί τα δεδομένα της, ακόμη κι όταν δεν τροφοδοτείται με ενέργεια

2.1.2 Περιορισμοί και ιδιαίτερα χαρακτηριστικά

Πέρα των παραπάνω συγκριτικών πλεονεκτημάτων της, η μνήμη flash τύπου NAND παρουσιάζει ένα σύνολο ιδιαίτερων χαρακτηριστικών, τα οποία και χρήζουν περαιτέρω ανάλυσης. Τα χαρακτηριστικά αυτά είναι τα ακόλουθα:

α) Περιορισμός της διαγραφής πριν από την εγγραφή

Ως γνωστόν, στα περισσότερα διαθέσιμα μέσα αποθήκευσης υπάρχει η δυνατότητα της άμεσης ενημέρωσης των δεδομένων που βρίσκονται αποθηκευμένα σε μία θέση μνήμης αυτών, μέσω της επανεγγραφής τους. Αντίθετα, στη μνήμη flash δεν υπάρχει η δυνατότητα αυτή, αλλά για να ενημερωθεί η τιμή κάποιου δεδομένου σε αυτήν, θα πρέπει πρώτα να λάβει χώρα η χρονοβόρα λειτουργία της διαγραφής του από τη μνήμη και στη συνέχεια η εγγραφή του, με τη νέα πλέον τιμή του, σε αυτήν. Ακόμη χειρότερα μάλιστα, η λειτουργία της διαγραφής των υπό ενημέρωση δεδομένων δεν αρκεί να περιλαμβάνει μόνο τα δεδομένα αυτά, αλλά και ολόκληρο το μπλοκ στο οποίο και βρίσκονται αυτά αποθηκευμένα. Για αυτόν το λόγο, όταν πρόκειται να ενημερωθεί η τιμή κάποιων δεδομένων σε μία μνήμη flash, το μπλοκ στο οποίο ανήκουν τα δεδομένα αυτά ονομάζεται μπλοκ διαγραφής. Αυτός ο περιορισμός της διαγραφής πριν από την εγγραφή στη μνήμη flash τύπου NAND, απαιτεί ειδικές τεχνικές διαχείρισης αποθήκευσης, προκειμένου να μειωθούν στο ελάχιστο οι λειτουργίες διαγραφής και να μην επηρεάζεται έτσι η συνολική αποδοτικότητα του συστήματος από τη μνήμη flash.

β) Περιορισμένο πλήθος διαγραφών/ επανεγγραφών

Κάθε σελίδα μνήμης σε μία μνήμη NAND flash μπορεί να αντέξει ένα περιορισμένο πλήθος διαγραφών, το οποίο και κυμαίνεται συνήθως από 10.000 μέχρι 100.000 λειτουργίες. Αν μάλιστα λάβουμε υπόψιν μας, ότι για κάθε επανεγγραφή δεδομένων στη μνήμη flash απαιτείται και μία λειτουργία διαγραφής, είναι προφανές ότι και το πλήθος των επανεγγραφών που μπορούν να πραγματοποιηθούν σε αυτήν είναι επίσης περιορισμένο.

γ) Δεν διαθέτει μηχανικά μέρη

Η μνήμη flash λόγω της ηλεκτρονικής της φύσης δεν διαθέτει κανένα μηχανικό μέρος, όπως είναι για παράδειγμα οι κεφαλές στους μαγνητικούς δίσκους. Έτσι λοιπόν κι ενώ στους μαγνητικούς δίσκους η καθυστέρηση αναζήτησης και περιστροφής επιφέρει το μεγαλύτερο κόστος κατά την ανάγνωση κι εγγραφή δεδομένων, στη μνήμη flash ο χρόνος πρόσβασης στα δεδομένα είναι ανάλογος του όγκου των δεδομένων που βρίσκονται αποθηκευμένα σε αυτήν κι ανεξάρτητος της θέσης μνήμης στην οποία βρίσκονται. Δηλαδή, η μνήμη flash πραγματοποιεί τυχαίες και σειριακές προσβάσεις με την ίδια ταχύτητα. Η ικανότητα αυτή της μνήμης flash τύπου NAND να εκτελεί ταχύτατα αναγνώσεις κι εγγραφές δεδομένων αποτελεί ένα από τα σημαντικότερα προς εκμετάλλευση χαρακτηριστικά της.

δ) Ασύμμετρο κόστος ανάγνωσης/ εγγραφής

Στη μνήμη flash, τα κόστη ανάγνωσης κι εγγραφής είναι ασύμμετρα μεταξύ τους. Μάλιστα, η ταχύτητα ανάγνωσης είναι τυπικά διπλάσια της ταχύτητας εγγραφής, σε αντίθεση για παράδειγμα με τους μαγνητικούς δίσκους, όπου οι ταχύτητες ανάγνωσης κι εγγραφής είναι ίδιες. Από το γεγονός αυτό, κρίνεται σημαντικό να μειώσουμε το πλήθος των εγγραφών, για να μην επηρεάζεται η αποδοτικότητα του συστήματος.

Πίνακας 2.1: Χρόνοι αποθήκευσης ανάγνωσης/ εγγραφής/ διαγραφής μνήμης NAND flash και μαγνητικού δίσκου

Μέσο αποθήκευσης	Ανάγνωση	Εγγραφή	Διαγραφή
Μαγνητικός Δίσκος	12,70 ms (2KB)	13,7 ms (2KB)	
Flash τύπου NAND	0,08 ms (2KB)	0,2 ms (2KB)	1,5 ms (128KB)

2.1.3 Κατάλληλο μέσο αποθήκευσης για τους SNs

Μετά από αυτήν τη σύντομη παρουσίαση της μνήμης flash και για να επανέλθουμε στο θέμα μας, θυμίζουμε ότι η είναι θεμιτό οι SNs σε ένα WSN να είναι εξοπλισμένοι με κάποιου τύπου μνήμη, ούτως ώστε τα λαμβανόμενα δεδομένα από τους αισθητήρες των SNs να μπορούν να αποθηκευτούν και να διαχειριστούν ενδοδικτυακά. Με βάση τα χαρακτηριστικά της μνήμης flash είναι πασιφανές, ότι ο ενδεικτικότερος τύπος μνήμης για τους SNs δεν μπορεί να είναι άλλος από τη μνήμη flash τύπου NAND. Για παράδειγμα το μικρό της μέγεθος, εξυπηρετεί την εγκατάστασή της στους επίσης μικρού μεγέθους κόμβους αισθητήρες. Επιπρόσθετα, η ταχεία λειτουργία της κρίνεται αναγκαία για να ισοσκελιστούν άλλες αδυναμίες των

WSNs. Εξίσου σημαντική κρίνεται και η ανθεκτικότητά της, δεδομένου ότι πολλές φορές τα SNs είναι εγκατεστημένα σε αφιλόξενα περιβάλλοντα. Τέλος, η δυνατότητα διατήρησης των αποθηκευμένων σε αυτήν δεδομένων χωρίς καμία απολύτως ενεργειακή τροφοδοσία εξασφαλίζει τη διαθεσιμότητα των μετρήσεων που έχουν πραγματοποιηθεί από τους αισθητήρες, ακόμη κι αν σταματήσει η τροφοδοσία τους με ενέργεια. Πράγματι λοιπόν, η μνήμη flash τύπου NAND χρησιμοποιείται ως το καταλληλότερο μέσο αποθήκευσης στους SNs ενός WSN.

2.2 Τεχνικές διαχείρισης αποθήκευσης σε μνήμη flash

Υπάρχει μία πληθώρα τεχνικών διαχείρισης αποθήκευσης, οι οποίες κι εφαρμόζονται στα υπάρχοντα αποθηκευτικά μέσα και οι οποίες μπορούν κάλλιστα να εφαρμοστούν και σε συστήματα με μνήμη NAND flash. Τα ιδιαίτερα χαρακτηριστικά όμως της τελευταίας και σε συνδυασμό με τη χρησιμοποίησή της ως αποθηκευτικού μέσου στους SNs των WSNs καθιστούν επιτακτική την ανάπτυξη κι εφαρμογή νέων τεχνικών. Σε αυτήν την ενότητα εστιάζουμε λοιπόν, στη μελέτη νέων τεχνικών διαχείρισης αποθήκευσης στη μνήμη NAND flash των SNs ενός WSN. Οι τεχνικές αυτές είναι οι ακόλουθες:

- Διαμέριση της μνήμης
- Εύρεση συσχετίσεων μεταξύ των δεδομένων
- Επαναχρησιμοποίηση των δεδομένων

2.3 Διαμέριση της μνήμης

Σε πρώτη φάση λοιπόν, θα εξετάσουμε μία τεχνική διαχείρισης αποθήκευσης σε μνήμη flash, η οποία και βασίζεται στη δημιουργία διαμερίσεων σε αυτήν.

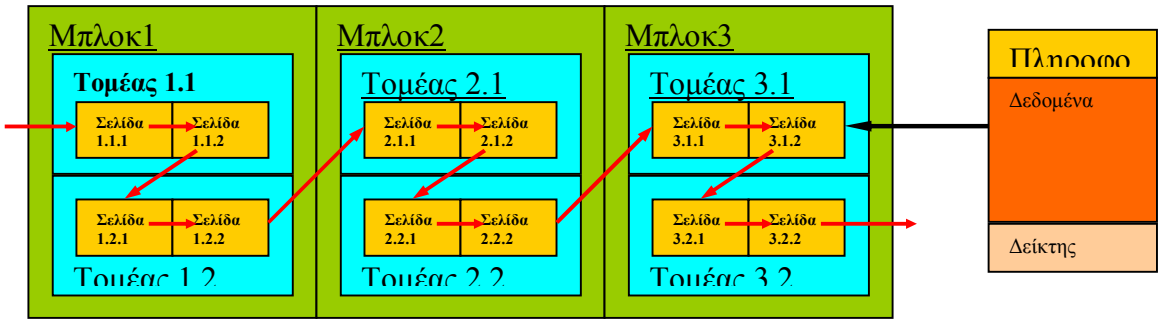
2.3.1 Μνήμη flash χωρίς διαμερίσεις

Ως γνωστόν, κάθε αισθητήρας ενός SN λαμβάνει περιοδικά μετρήσεις της τιμής κάποιας παραμέτρου. Τα δεδομένα που λαμβάνει ο αισθητήρας αποθηκεύονται στη μνήμη flash του SN, στον οποίο και βρίσκεται εγκατεστημένος, προκειμένου να χρησιμοποιηθούν όποτε υπάρξει σχετικό αίτημα από κάποιο χρήστη. Το ζήτημα που προκύπτει είναι με ποιον ακριβώς τρόπο λαμβάνει χώρα η αποθήκευση των δεδομένων από τις μετρήσεις των αισθητήρων ενός SN στη μνήμη flash αυτού. Δεδομένου λοιπόν, ότι η δομική μονάδα αποθήκευσης της μνήμης flash είναι η σελίδα, μία απλή λύση είναι η σειριακή προσάρτηση νέων σελίδων δεδομένων στη μνήμη flash του SN για νέες εγγραφές από τους αισθητήρες αυτού.

Πιο συγκεκριμένα κι επανερχόμενοι στην πρότερη περιγραφή της μνήμης flash, υπενθυμίζουμε ότι αυτή χωρίζεται σε μπλοκ, καθένα από τα οποία χωρίζεται σε τομείς. Κάθε τομέας με τη σειρά του αποτελείται από ένα πλήθος σελίδων, οι οποίες και είναι δομημένες σε λίστες εντός αυτού. Για κάθε λίστα σελίδων εντός ενός τομέα, υπάρχει ένας δείκτης προς τη διεύθυνση της πρώτης σελίδας που βρίσκεται αποθηκευμένη σε αυτήν, δηλαδή ένας δείκτης που δείχνει στην αρχή αυτής της λίστας. Κάθε σελίδα της λίστας διαθέτει επίσης ένα δείκτη, ο οποίος και περιέχει τη διεύθυνση της επόμενης μετά από αυτήν σελίδας στη λίστα. Ο δείκτης της τελευταίας σελίδας της λίστας σηματοδοτεί το τέλος της λίστας περιέχοντας είτε τη διεύθυνση

της επόμενης λίστας σελίδων του ιδίου τομέα είτε τη διεύθυνση του επόμενου τομέα είτε τη διεύθυνση του επόμενου μπλοκ. Φυσικά, όταν μία λίστα σελίδων είναι κενή, τότε ο δείκτης που δείχνει στην αρχή αυτής ορίζεται να έχει το ίδιο περιεχόμενο με το δείκτη που δείχνει στο τέλος της. Όλα αυτά απεικονίζονται και στο σχήμα που ακολουθεί.

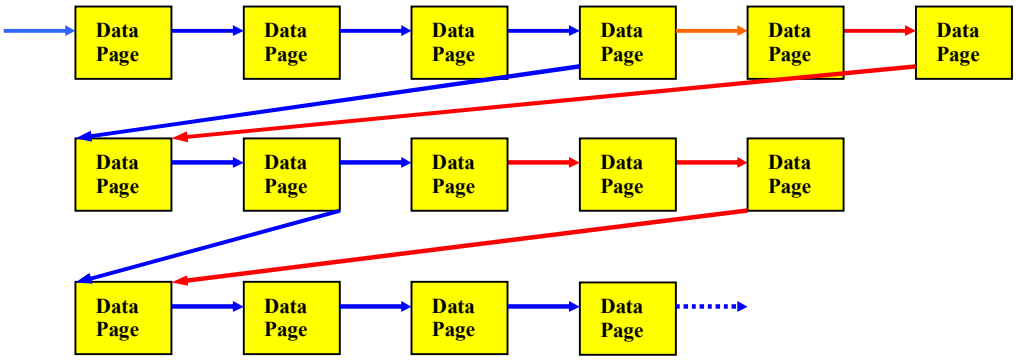
Σχήμα 2.1: (Δομή μνήμης NAND flash) (Σελίδα)



Αρχικά, θα εξετάσουμε την περίπτωση στην οποία δεν υπάρχουν διαμερίσεις στη μνήμη flash, με αποτέλεσμα η προσθήκη νέων σελίδων δεδομένων σε αυτήν, να γίνεται σειριακά. Κατά τη σειριακή προσάρτηση νέων σελίδων δεδομένων, κάθε φορά που απαιτείται μία καινούργια εγγραφή, η μνήμη flash ανατρέχει στην πρώτη διαθέσιμη λίστα αυτής, η οποία έχει τουλάχιστον μία κενή θέση και προσαρτά σε αυτήν μία ή περισσότερες σελίδες δεδομένων που αντιστοιχούν στη συγκεκριμένη εγγραφή. Σε περίπτωση, που οι ελεύθερες θέσεις στη συγκεκριμένη λίστα δεν αρκούν για όλες τις σελίδες δεδομένων που αντιστοιχούν στην εγγραφή αυτή, τότε η μνήμη flash συνεχίζει στην επόμενη λίστα δεδομένων που διαθέτει κενές θέσεις. Ας δούμε κι ένα σχετικό αποσαφηνιστικό παράδειγμα:

Έστω, ότι σε κάποια δεδομένη χρονική στιγμή ένας SN παραλαμβάνει από έναν από τους αισθητήρες του μία νέα εγγραφή των μετρήσεων αυτού κι ας υποθέσουμε επιπλέον, ότι για την αποθήκευση αυτής της εγγραφής στη μνήμη flash του SN απαιτούνται 4 σελίδες δεδομένων. Τότε, ο SN θα αρχίσει να διατρέχει σειριακά τη μνήμη flash ξεκινώντας από την πρώτη λίστα του πρώτου τομέα που ανήκει στο πρώτο μπλοκ, προκειμένου να εντοπίσει σε αυτήν, 4 κενές σελίδες δεδομένων. Αν οι λίστες του τομέα αυτού έχουν την μορφή του παρακάτω σχήματος κι επιπλέον κάθε λίστα έχει χωρητικότητα 6 σελίδων, τότε αρχικά ο SN θα διατρέξει την πρώτη λίστα. Μόλις φτάσει στην $5^{η}$ θέση αυτής θα διαπιστώσει ότι είναι κενή και θα εισάγει σε αυτήν την πρώτη σελίδα δεδομένων και θα θέσει το δείκτη αυτής να δείχνει στην αρχή της επόμενης λίστας. Παράλληλα, θα θέσει το δείκτη της $4^{ης}$ σελίδας δεδομένων που πριν έδειχνε στην αρχή της επόμενης λίστας δεδομένων να δείχνει πλέον στη διεύθυνση της σελίδας της $5^{ης}$ θέσης, την οποία και μόλις εισάγαμε. Στη συνέχεια και με παρόμοιο τρόπο θα εισαχθούν στην $6^{η}$ θέση της πρώτης λίστας και στις θέσεις4 και 5 της δεύτερης λίστας, οι υπόλοιπες τρεις σελίδες δεδομένων που αντιστοιχούν στην εγγραφή μας. Τέλος, θα γίνουν και όλες οι απαραίτητες σχετικές ενημερώσεις στο ευρετήριο της μνήμης flash. Για περαιτέρω κατανόηση των προαναφερθέντων, ο αναγνώστης μπορεί να ανατρέξει στο σχήμα, που ακολουθεί:

Σχήμα 2.2 Σειριακή προσθήκη δεδομένων σε μνήμη flash χωρίς διαμερίσεις



Συνεπώς κατά τη σειριακή προσάρτηση των σελίδων στη μνήμη flash, κάθε εγγραφή και για την ακρίβεια οι σελίδες που αντιστοιχούν σε αυτήν, τοποθετείται με βάση τη χρονική στιγμή της άφιξής της, στις πρώτες διαθέσιμες κενές θέσεις της μνήμης flash. Το γεγονός αυτό έχει ως αποτέλεσμα, σε γειτονικές σελίδες μνήμης να βρίσκονται αποθηκευμένες ως επί το πλείστον εγγραφές των αισθητήρων με παρόμοιες χρονικές στιγμές άφιξης. Έτσι λοιπόν, η παραπάνω τεχνική κρίνεται μεν ικανοποιητική για την ταχεία εξυπηρέτηση αιτημάτων που αφορούν το χρόνο των μετρήσεων, όχι όμως και αιτημάτων που αφορούν την τιμή των μετρήσεων των αισθητήρων του SN, καθώς παρόμοιες τιμές δεδομένων μπορεί να βρίσκονται διασκορπισμένες σε πολλές διαφορετικές θέσεις της μνήμης flash. Αν μάλιστα λάβουμε υπόψιν μας το μεγάλο πλήθος των αιτημάτων που αφορούν τις τιμές των δεδομένων, γίνεται σαφής η καθυστέρηση που εισάγεται στο WSN κατά την εξυπηρέτηση αυτών των αιτημάτων, λόγω του υπέρογκου πλήθους των προσπελάσεων μνήμης που απαιτούνται για το σκοπό αυτό.

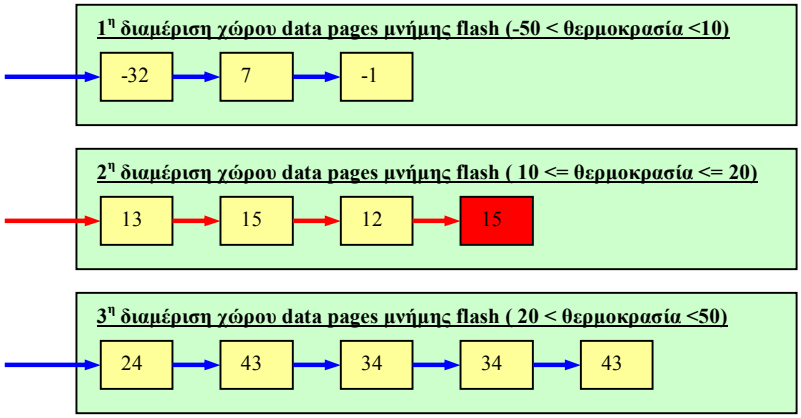
2.3.2 Μνήμη flash χωρισμένη σε διαμερίσεις

Προκειμένου λοιπόν, να πετύχουμε μία αποτελεσματική και ταχεία εξυπηρέτηση, τόσο των αιτημάτων που αφορούν το χρόνο, όσο επίσης και των αιτημάτων που αφορούν την τιμή, θα προτείνουμε μία βελτιωμένη τεχνική αποθήκευσης των δεδομένων στη μνήμη flash, η οποία και στηρίζεται στη διαμέριση του τμήματος, στο οποίο αποθηκεύονται οι σελίδες δεδομένων της μνήμης flash. Πιο συγκεκριμένα, το τμήμα αποθήκευσης δεδομένων της μνήμης flash χωρίζεται σε ένα πλήθος διαμερίσεων, καθεμία από τις οποίες αντιστοιχεί σε ένα συγκεκριμένο σύνολο τιμών. Φυσικά, η ένωση (U) όλων των διαμερίσεων θα πρέπει να αντιστοιχεί στο σύνολο όλων των δυνατών τιμών των δεδομένων. Έτσι λοιπόν, όταν πρέπει να γίνει μία καινούργια εγγραφή στη μνήμη flash, τότε πρώτα εξετάζεται η τιμή των δεδομένων αυτής της εγγραφής και με βάση αυτήν προκύπτει σε ποια διαμέριση θα πρέπει αυτή να αποθηκευτεί. Εν συνεχεία, πραγματοποιείται μετάβαση σε αυτήν τη διαμέριση, όπου και λαμβάνει χώρα η σειριακή αποθήκευση της εν λόγω εγγραφής και για την ακρίβεια η σειριακή προσάρτηση των σελίδων δεδομένων που αντιστοιχούν σε αυτήν την εγγραφή. Προτού παρουσιάσουμε ένα σχετικό παράδειγμα, θα πρέπει να τονίσουμε, ότι η διαμέριση της μνήμης flash θα πρέπει να μεταβάλλεται δυναμικά σε συνάρτηση με το φόρτο εργασίας στο WSN κατά τη λειτουργία αυτού, ούτως ώστε να έχουμε τα βέλτιστα επιθυμητά αποτελέσματα. Τέλος για τη διαμέριση του χώρου δεδομένων της μνήμης flash μπορεί να χρησιμοποιηθεί ένας οποιοσδήποτε από τους γνωστούς αλγόριθμους κατακερματισμού, ανάλογα και με τις ανάγκες της εκάστοτε εφαρμογής που εξυπηρετεί το WSN.

2.3.3 Παράδειγμα 2.1 – Αποθήκευση δεδομένων σε μνήμη flash χωρισμένη σε διαμερίσεις

Ας θεωρήσουμε λοιπόν, ότι ένας SN είναι εφοδιασμένος με έναν αισθητήρα μέτρησης της θερμοκρασίας. Ο αισθητήρας αυτός λαμβάνει περιοδικά την τιμή της θερμοκρασίας του περιβάλλοντος, στο οποίο είναι εγκατεστημένος κι εν συνεχεία αποδίδει στον SN μία εγγραφή με τη μετρούμενη τιμή. Το πλήθος των σελίδων δεδομένων που απαιτούνται για την αποθήκευση κάθε εγγραφής ισούται με 1, ενώ το τμήμα δεδομένων της μνήμης flash του SN έχει διαμεριστεί σε τρία τμήματα, καθένα από τα οποία αντιστοιχεί στα ακόλουθα διαστήματα θερμοκρασιών (-50,10) , [10,20], (20,50). Έστω, ότι σε κάποια οποιαδήποτε χρονική στιγμή, ο αισθητήρας αποδίδει στον SN μία εγγραφή με τιμή θερμοκρασίας ίση με 15 βαθμούς Κελσίου. Τότε, ο SN χρησιμοποιώντας κάποιον αλγόριθμο κατακερματισμού θα εντοπίσει πρώτα τη διαμέριση, στην οποία και πρέπει να αποθηκευτεί η εν λόγω εγγραφή. Εν συνεχεία, θα πραγματοποιήσει μετάβαση σε αυτήν τη διαμέριση, η οποία στην προκειμένη περίπτωση είναι η δεύτερη και θα αποθηκεύσει στην πρώτη διαθέσιμη θέση αυτής τη σελίδα δεδομένων που απαιτείται για τη συγκεκριμένη εγγραφή.

Σχήμα 2.3: Προσθήκη δεδομένων σε μνήμη flash χωρισμένη σε διαμερίσεις



Τελικά λοιπόν, με την παραπάνω τεχνική επιτυγχάνουμε την ταχύτερη εξυπηρέτηση, τόσο των αιτημάτων που αφορούν το χρόνο, όσο και των αιτημάτων που αφορούν την τιμή και κυρίως, χρησιμοποιώντας χαμηλής πολυπλοκότητας αλγορίθμους διαμέρισης και κατακερματισμού, γεγονός που κρίνεται ιδιαίτερα σημαντικό αν λάβουμε υπόψιν μας τη μικρή υπολογιστική ισχύ ενός SN.

2.4 Εύρεση συσχετίσεων μεταξύ των δεδομένων

Ο σημαντικότερος ίσως περιορισμός της μνήμης flash, τον οποίο κι αναλύσαμε σε προηγούμενη ενότητα είναι ο περιορισμός της διαγραφής πριν από την εγγραφή. Σύμφωνα με αυτόν το περιορισμό, στη μνήμη flash δεν υπάρχει η δυνατότητα της άμεσης ενημέρωσης μιας θέσης μνήμης, όπως συμβαίνει για παράδειγμα στους μαγνητικούς δίσκους. Αντίθετα, όποτε πρέπει να λάβει χώρα ενημέρωση της τιμής κάποιων δεδομένων που βρίσκονται αποθηκευμένα σε αυτήν, πρέπει πρώτα να λάβει χώρα η λειτουργία της διαγραφής ολοκλήρου του μπλοκ, στο οποίο ανήκουν αυτά τα

δεδομένα και το οποίο για αυτόν το λόγο ονομάζεται μπλοκ διαγραφής κι εν συνεχεία να λάβει χώρα η λειτουργία της εγγραφής ξανά ολοκλήρου του μπλοκ διαγραφής σε μία νέα θέση μνήμης με τα ενημερωμένα πλέον δεδομένα. Αν μάλιστα λάβουμε υπόψιν μας και τον περιορισμό του πλήθους των διαγραφών/ επανεγγραφών μίας σελίδας μνήμης, σύμφωνα με τον οποίο σε κάθε μπλοκ της μνήμης flash μπορεί να λάβει χώρα περιορισμένο πλήθος διαγραφών και συνεπώς κι επανεγγραφών, τότε κατανοούμε απόλυτα το μεγάλο κόστος της διαδικασίας της επανεγγραφής. Το γεγονός αυτό μας υποχρεώνει να αναζητήσουμε νέες τεχνικές διαχείρισης της αποθήκευσης σε μνήμες flash, μέσω των οποίων και θα μειώσουμε το πλήθος και το κόστος των απαιτούμενων λειτουργιών διαγραφής και κατ' επέκταση και των διαδικασιών επανεγγραφής, χωρίς να επηρεάσουμε τη λειτουργικότητα και την αποδοτικότητα της μνήμης flash. Στην παρούσα ενότητα θα περιγράψουμε μία τεχνική, η οποία επιτυγχάνει το σκοπό αυτό εκμεταλλευόμενη τις συσχετίσεις μεταξύ των δεδομένων που βρίσκονται αποθηκευμένα στη μνήμη flash.

Η βασική φιλοσοφία αυτής της τεχνικής έγκειται στην τοποθέτηση στο ίδιο μπλοκ διαγραφής σελίδων μνήμης, οι οποίες ενημερώνονται ταυτόχρονα. Για να το πετύχουμε αυτό, θα πρέπει να εντοπίσουμε συσχετίσεις μεταξύ των ενημερώσεων των δεδομένων που βρίσκονται αποθηκευμένα στη μνήμη flash. Δηλαδή, θα πρέπει να αναζητήσουμε κάποια δεδομένα Α, η ενημέρωσή των οποίων λαμβάνει χώρα πάντα ή σχεδόν πάντα ταυτόχρονα με την ενημέρωση κάποιων άλλων δεδομένων Β, οπότε και θα πρέπει να αποθηκεύσουμε αυτά τα δύο σύνολα δεδομένων Α και Β, στο ίδιο μπλοκ διαγραφής. Προκειμένου να γίνει κατανοητή αυτή η τεχνική θα χρησιμοποιήσουμε για πολλοστή φορά τη δύναμη του παραδείγματος.

2.4.1 Παράδειγμα 2.2 – Ενημέρωση δεδομένων μνήμης flash με εύρεση συσχετίσεων μεταξύ των δεδομένων

Ας θεωρήσουμε, ότι έχουμε 4 σελίδες δεδομένων, τις P1, P2, P3 και P4. Έχουμε εντοπίσει κάποια συσχέτιση κατά την ενημέρωση των σελίδων P1 και P2, δηλαδή έχουμε διαπιστώσει ότι οι ενημερώσεις των σελίδων P1 και P2 λαμβάνουν χώρα πολύ συχνά κατά τις ίδιες χρονικές στιγμές. Αντίθετα, οι σελίδες P3 και P4 ενημερώνονται πολύ σπάνια. Ας θεωρήσουμε, ότι έχουμε δύο μπλοκ διαγραφής B1 και B2 από τα οποία στο B1 είναι αποθηκευμένες οι σελίδες P1 και P3, ενώ στο B2 είναι αποθηκευμένες οι σελίδες P2 και P4.

Σχήμα 2.4: Μπλοκ διαγραφής 1ης περίπτωσης

Έστω, ότι κάποια χρονική στιγμή πρέπει να ενημερωθεί η τιμή της σελίδας P1. Τότε λόγω της συσχέτισης που αναφέραμε παραπάνω είναι πολύ πιθανόν να πρέπει να ενημερωθεί και η τιμή της σελίδας P2. Αντίθετα, οι σελίδες P3 και P4, λογικά δεν θα πρέπει να ενημερωθούν. Σύμφωνα λοιπόν, με τα όσα αναφέραμε και νωρίτερα για τη διαδικασία της ενημέρωσης στη μνήμη flash, προκειμένου να ενημερωθούν οι τιμές των P1 και P2, θα πρέπει πρώτα να διαγραφούν ολόκληρα τα μπλοκ διαγραφής, στα οποία αυτές ανήκουν. Συνεπώς θα πρέπει να διαγραφούν ολόκληρα τα μπλοκ B1

και B2, με αποτέλεσμα τη διαγραφή και των σελίδων P2 και P4, παρότι αυτές δεν πρόκειται να ενημερωθούν. Εν συνεχεία, τα δύο μπλοκ B1 και B2, θα πρέπει να ξαναγραφούν με τις ενημερωμένες πλέον τιμές για τις P1 και P2. Προκειμένου λοιπόν να ολοκληρωθεί η διαδικασία της ενημέρωσης των P1 και P2, θα πρέπει συνολικά να λάβουν χώρα 4 λειτουργίες διαγραφής σελίδας, μία για καθεμία από τις P1, P2, P3 και P4 κι επιπλέον 4 λειτουργίες εγγραφής σελίδας, μία και πάλι για καθεμία από τις P1, P2, P3 και P4.

Ας θεωρήσουμε τώρα για το ίδιο παράδειγμα, ότι οι P1 και P2 βρίσκονται αποθηκευμένες στο μπλοκ B1, ενώ οι P3 και P4 στο μπλοκ B2.

Σχήμα 2.4: Μπλοκ διαγραφής 2^{ης} περίπτωσης

Σε αυτήν τη δεύτερη περίπτωση, κατά την ενημέρωση των P1 και P2, θα πρέπει να διαγραφεί μόνο το μπλοκ B1 κι όχι το μπλοκ B2, γεγονός που σημαίνει ότι για να ολοκληρωθεί η διαδικασία της ενημέρωσης των P1 και P2, θα πρέπει συνολικά να λάβουν χώρα 2 λειτουργίες διαγραφής σελίδας, μία για καθεμία από τις P1 και P2 κι επιπλέον 2 λειτουργίες εγγραφής σελίδας, μία και πάλι για καθεμία από τις P1 και P2. Άρα είναι προφανές, ότι τοποθετώντας τις P1 και P2 στο ίδιο μπλοκ υποδιπλασιάσαμε το πλήθος των απαιτούμενων λειτουργιών εγγραφής κι ανάγνωσής, κατά τη διαδικασία της ενημέρωσής τους, εκμεταλλευόμενοι τη συσχέτιση που παρουσιάζουν οι δύο αυτές σελίδες κατά την ενημέρωσή τους.

2.4.2 Μειονέκτημα

Πριν κλείσουμε την αναφορά σε αυτήν την τεχνική, θα πρέπει να επισημάνουμε και ένα βασικό μειονέκτημά της. Πιο συγκεκριμένα, για να μπορεί η εν λόγω τεχνική να προβλέπει τις μελλοντικές ενημερώσεις, θα πρέπει να βασίζεται στις παλαιότερες ενημερώσεις, γεγονός που σημαίνει ότι θα πρέπει να καταλαμβάνεται κάποιος χώρος στη μνήμη για τη διατήρηση της ιστορίας των ενημερώσεων. Εντούτοις και με βάση όλα τα παραπάνω, είναι σαφής η βελτιστοποίηση που εισάγεται από την τεχνική της διαχείρισης αποθήκευσης σε μνήμη flash με χρήση συσχετίσεων κατά την ενημέρωση δεδομένων. Φυσικά, υπάρχουν και πολλών ειδών άλλων συσχετίσεις μεταξύ των δεδομένων, οι οποίες και μπορούν να χρησιμοποιηθούν σε παρόμοιες τεχνικές.

2.5 Επαναχρησιμοποίηση δεδομένων

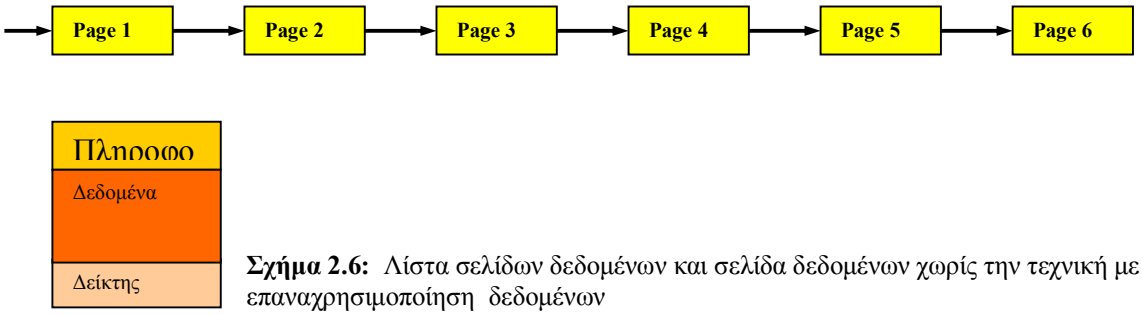
Έχει παρατηρηθεί, ότι στα περισσότερα WSNs η συντριπτική πλειοψηφία των δεδομένων ανήκει σε ένα περιορισμένο εύρος τιμών. Το γεγονός αυτό έχει ως αποτέλεσμα, να είναι πολύ πιθανή η επαναληπτική εμφάνιση των ίδιων τιμών δεδομένων στις μετρήσεις των αισθητήρων. Με βάση λοιπόν αυτήν την παρατήρηση και θέλοντας να ξεπεράσουμε τους περιορισμούς της διαγραφής πριν από την εγγραφή και του περιορισμένου πλήθος διαγραφών/ επανεγγραφών, θα παρουσιάσουμε στην παρούσα ενότητα μία ακόμη τεχνική διαχείρισης αποθήκευσης

στη μνήμη flash των SNs ενός WSN, η οποία και στηρίζεται στην επαναχρησιμοποίηση δεδομένων.

Η βασική φιλοσοφία της τεχνικής αυτής είναι αντί να αποθηκεύουμε κάθε νέα τιμή που προέρχεται από κάποια μέτρηση ενός αισθητήρα σε μία νέα σελίδα δεδομένων, να συγκρίνουμε τη νέα τιμή αυτή με τις υπάρχουσες τιμές στις ήδη αποθηκευμένες σελίδες δεδομένων. Αν η νέα τιμή είναι παρόμοια με κάποια ήδη υπάρχουσα τιμή, τότε θα προσθέτουμε τη νέα τιμή σε ένα αρχείο καταγραφής, το οποίο και θα διασυνδέεται μέσω δείκτη σε διεύθυνση με τη σελίδα δεδομένων που περιέχει την παρόμοια ήδη υπάρχουσα τιμή. Δηλαδή, η τεχνική αυτή χρησιμοποιεί ένα όριο, το οποίο και συγκρίνει τη διαφορά μεταξύ της νέας τιμής και των υπαρχουσών τιμών στις ήδη αποθηκευμένες σελίδες δεδομένων. Αν το όριο αυτό είναι μικρότερο της διαφοράς της νέας τιμής με την τιμή κάποιας ήδη υπάρχουσας σελίδας δεδομένων, τότε τίθεται ένας δείκτης προς εγγραφή κάποια καταγραφής, τον οποίο και διαθέτει αυτή η υπάρχουσα σελίδα δεδομένων, να δείχνει προς μία νέα εγγραφή καταγραφής. Παράλληλα, αποθηκεύεται σε αυτήν την εγγραφή καταγραφής η εν λόγω διαφορά. Όλα τα ανωτέρω, μελετώνται και διευκρινίζονται πλήρως προς ευκολία του αναγνώστη στις παραγράφους που ακολουθούν.

2.5.1 Δομή της μνήμης flash χωρίς επαναχρησιμοποίηση δεδομένων

Πιο συγκεκριμένα, όπως έχουμε ήδη αναφέρει η μνήμη flash χωρίζεται σε μπλοκ, καθένα από τα οποία χωρίζεται σε τομείς. Κάθε τομέας με τη σειρά του αποτελείται από ένα πλήθος σελίδων δεδομένων, οι οποίες και είναι δομημένες σε λίστες εντός αυτού. Κάθε σελίδα δεδομένων απαρτίζεται από τα δεδομένα που βρίσκονται αποθηκευμένα σε αυτήν, πληροφορίες για τα δεδομένα αυτά και τη σελίδα γενικότερα και τέλος κάποιον δείκτη προς τη διεύθυνση μίας άλλης σελίδας δεδομένων. Έτσι λοιπόν μία σελίδα δεδομένων και μία λίστα σελίδων δεδομένων της μνήμης flash ενός SN, όταν δεν λαμβάνει χώρα επαναχρησιμοποίηση δεδομένων, θα έχουν τη μορφή που φαίνεται στο παρακάτω σχήμα:

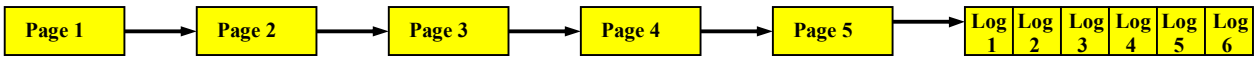


Σχήμα 2.6: Λίστα σελίδων δεδομένων και σελίδα δεδομένων χωρίς την τεχνική με επαναχρησιμοποίηση δεδομένων

2.5.2 Δομή μνήμης flash με επαναχρησιμοποίηση δεδομένων

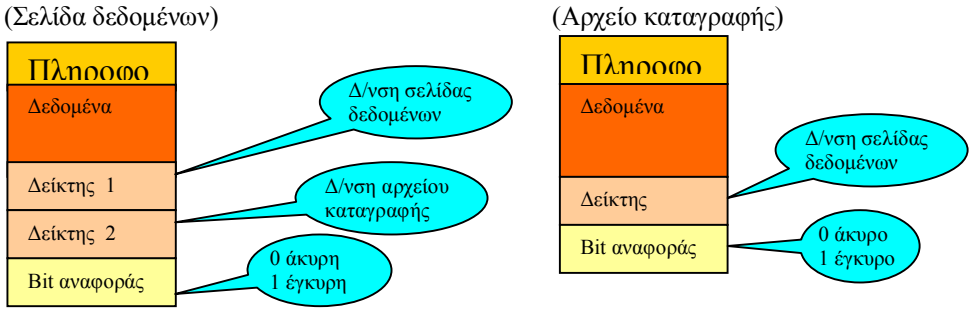
Η τεχνική, την οποία κι εξετάζουμε στην παρούσα ενότητα αντικαθιστά την τελευταία σελίδα δεδομένων κάθε λίστας σελίδων δεδομένων από ένα σύνολο αρχείων καταγραφής (log records). Το πλήθος των αρχείων καταγραφής ισούται με το πλήθος των σελίδων δεδομένων της λίστας δεδομένων. Δηλαδή κι αν θεωρήσουμε τη λίστα δεδομένων του παραπάνω σχήματος, η οποία κι αποτελείται από 6 σελίδες

δεδομένων, τότε με την εν λόγω τεχνική οι 5 πρώτες σελίδες δεδομένων αυτής θα παραμείνουν ως έχουν, ενώ η τελευταία θα αντικατασταθεί από 6 αρχεία καταγραφής. Έτσι λοιπόν, αυτή η λίστα δεδομένων θα έχει πλέον τη μορφή του παρακάτω σχήματος:



Σχήμα 2.7: Λίστα σελίδων δεδομένων με την τεχνική με επαναχρησιμοποίηση δεδομένων

Επιπλέον, προσθέτει σε καθεμία από τις υπόλοιπες σελίδες δεδομένων έναν δείκτη προς τη διεύθυνση κάποιου αρχείου καταγραφής κι έναν μετρητή αναφοράς. Δηλαδή, κάθε σελίδα δεδομένων θα περιέχει πλέον δύο δείκτες προς κάποια διεύθυνση. Έναν, ο οποίος μπορεί να περιέχει τη διεύθυνση κάποιας άλλης σελίδας δεδομένων κι έναν που ενδεχομένως θα περιέχει τη διεύθυνση κάποιου αρχείου καταγραφής. Ο μετρητής ή αλλιώς δείκτης ή αλλιώς bit αναφοράς είναι στην ουσία ένα bit, το οποίο κι έχει την τιμή 1 όταν η τιμή που βρίσκεται αποθηκευμένη στη σελίδα δεδομένων είναι έγκυρη και 0, όταν είναι άκυρη.



Σχήμα 2.8: Σελίδα δεδομένων με την τεχνική με επαναχρησιμοποίηση δεδομένων και αρχείο καταγραφής

Κάθε αρχείο καταγραφής είναι μία μικρότερη μονάδα αποθήκευσης από μία σελίδα δεδομένων και το οποίο αποτελείται από ένα κατάλληλο πρόσημο (+ ή -) ακολουθούμενο από κάποια τιμή. Επιπλέον διαθέτει έναν δείκτη, ο οποίος και περιέχει είτε τη διεύθυνση κάποιου άλλου αρχείου καταγραφής της ίδιας λίστας σελίδων δεδομένων είτε τη διεύθυνση της πρώτης σελίδας δεδομένων της επόμενης λίστας ή του επόμενου τομέα ή του επόμενου μπλοκ. Τέλος, περιέχει επίσης κι ένα bit αναφοράς.

Μέχρι στιγμής λοιπόν, έχουμε εξετάσει τα ιδιαίτερα χαρακτηριστικά, τα οποία εισάγει στη μνήμη NAND flash η τεχνική που μελετάμε σε σχέση με τα όσα έχουμε δει μέχρι τώρα. Στη συνέχεια, θα εξετάσουμε τον τρόπο με τον οποίο γίνεται η αναζήτηση, η αποθήκευση και η ενημέρωση των δεδομένων με τη συγκεκριμένη τεχνική, καθώς επίσης και τα πλεονεκτήματα που προκύπτουν από τη χρήση αυτής. Για το σκοπό αυτό θα παρουσιάσουμε τρεις αλγορίθμους για την αναζήτηση, την εισαγωγή και την ενημέρωση της τιμής των σελίδων δεδομένων που βρίσκονται αποθηκευμένες στη μνήμη flash ενός SN. Οι αλγόριθμοι αυτοί είναι οι ακόλουθοι:

- Αλγόριθμος αναζήτησης σελίδων δεδομένων
- Αλγόριθμος εισαγωγής νέων σελίδων δεδομένων
- Αλγόριθμος ενημέρωσης της τιμής μίας σελίδας δεδομένων

2.5.3 Αλγόριθμος αναζήτησης σελίδων δεδομένων

1. Άφιξη ενός αιτήματος, το οποίο μας καλεί να αναζητήσουμε την ύπαρξη κάποιας συγκεκριμένης τιμής στις σελίδες δεδομένων που βρίσκονται αποθηκευμένες στη μνήμη flash του SN.

2. Ξεκινάμε να διατρέχουμε τη μνήμη flash από την πρώτη σελίδα του πρώτου τομέα του πρώτου μπλοκ αυτής.

   a. Αν η σελίδα αυτή εμπεριέχει τη ζητούμενη τιμή κι επιπλέον το bit αναφοράς αυτής ισούται με 1, τότε επιστρέφουμε αυτήν την τιμή ως αποτέλεσμα και τερματίζουμε την εκτέλεση του αλγορίθμου.

   b. Αν η σελίδα αυτή δεν εμπεριέχει τη ζητούμενη τιμή ή το bit αναφοράς της ισούται με 0, τότε εξετάζουμε τους δείκτες αυτής.

3. Αν διαθέτει δείκτη προς κάποια εγγραφή καταγραφής, τότε μεταβαίνουμε σε αυτήν την εγγραφή, προσθέτουμε ή αφαιρούμε αναλόγως με το πρόσημο την τιμή αυτής της εγγραφής καταγραφής με την τιμή αυτής της σελίδας δεδομένων και υπολογίζουμε εκ νέου τη διαφορά του αποτελέσματος με την υπό αποθήκευση σελίδα δεδομένων.

   a. Αν η διαφορά που προκύπτει είναι μικρότερη του συγκεκριμένου ορίου κι επιπλέον το bit αναφοράς αυτής της εγγραφής καταγραφής ισούται με 1, τότε την επιστρέφουμε ως αποτέλεσμα.

   b. Ειδάλλως, κοιτάμε αν η εν λόγω επανεγγραφή έχει δείκτη προς κάποια άλλη επανεγγραφή κι επαναλαμβάνουμε την ίδια διαδικασία.

4. Αν δεν διαθέτει δείκτη προς κάποια εγγραφή καταγραφής, τότε ελέγχουμε αν διαθέτει δείκτη προς επόμενη σελίδα δεδομένων. Αν ναι τότε μεταβαίνουμε στην παραπάνω σελίδα κι εκτελούμε για αυτήν ξανά το δεύτερο βήμα του αλγορίθμου. Ενώ τέλος, αν όχι τότε θα πρέπει να διαθέτει δείκτη προς την πρώτη σελίδα δεδομένων της επόμενης λίστας ή προς τον επόμενο τομέα ή προς το επόμενο μπλοκ, οπότε και συνεχίζουμε την εκτέλεση του αλγορίθμου από εκεί, επαναλαμβάνοντας από το δεύτερο βήμα αυτού.

2.5.4 Αλγόριθμος εισαγωγής νέων σελίδων δεδομένων

1. Άφιξη στη μνήμη flash μίας νέας σελίδας δεδομένων προς αποθήκευση

2. Εύρεση του πρώτου τομέα, ο οποίος διαθέτει τουλάχιστον μία ελεύθερη σελίδα δεδομένων ή μία ελεύθερη εγγραφή καταγραφής

2.1 Αν ο τομέας αυτός, δεν διαθέτει καμία ελεύθερη σελίδα δεδομένων, αλλά διαθέτει τουλάχιστον μία ελεύθερη εγγραφή καταγραφής, τότε συγκρίνουμε την τιμή της υπό αποθήκευσης σελίδας δεδομένων με την τιμή καθεμίας από τις σελίδες δεδομένων αυτού του τομέα και

i. Αν η διαφορά της τιμής της υπό αποθήκευσης σελίδας δεδομένων από την τιμή καθεμίας εκ των σελίδων δεδομένων του τομέα αυτού, καθώς και από τις εγγραφές καταγραφής που αντιστοιχούν σε αυτές είναι μεγαλύτερη ενός συγκεκριμένου ορίου (στην προκειμένη περίπτωση ίσου με 10 μονάδες), τότε προχωράμε στην εξέταση του επόμενου τομέα, δηλαδή επιστρέφουμε στο δεύτερο βήμα.

ii. Αν όμως, η διαφορά αυτή βρεθεί να είναι μικρότερη του συγκεκριμένου ορίου για κάποια ήδη υπάρχουσα σελίδα δεδομένων στον τομέα αυτό, τότε μεταβαίνουμε στο τρίτο βήμα.

2.2 Αν ο τομέας αυτός, δεν διαθέτει καμία ελεύθερη εγγραφή καταγραφής, αλλά διαθέτει τουλάχιστον μία ελεύθερη θέση στις σελίδες δεδομένων αυτού, τότε αποθηκεύουμε την υπό αποθήκευση σελίδα δεδομένων στην πρώτη ελεύθερη θέση στις σελίδες δεδομένων και θέτουμε το bit αναφοράς αυτής ίσο με 1.

2.3 Αν ο τομέας αυτός, διαθέτει τουλάχιστον μία ελεύθερη σελίδα δεδομένων και τουλάχιστον μία εγγραφή καταγραφής, τότε συγκρίνουμε την τιμή της υπό αποθήκευσης σελίδας δεδομένων με την τιμή καθεμίας από τις σελίδες δεδομένων αυτού του τομέα και

i. Αν η προκύπτουσα διαφορά της υπό αποθήκευση σελίδας δεδομένων από όλες τις σελίδες δεδομένων, καθώς και από τις εγγραφές καταγραφής που αντιστοιχούν σε αυτές είναι μεγαλύτερη του συγκεκριμένου ορίου, τότε προσθέτουμε την υπό αποθήκευση σελίδα δεδομένων στην πρώτη ελεύθερη θέση στις σελίδες δεδομένων και θέτουμε το bit αναφοράς αυτής ίσο με 1.

ii. Αν όμως, η διαφορά αυτή βρεθεί μικρότερη του συγκεκριμένου ορίου, τότε μεταβαίνουμε στο τρίτο βήμα.

3.

i. Αν βρεθεί ότι η υπό αποθήκευση σελίδα δεδομένων διαφέρει λιγότερο από το συγκεκριμένο όριο από μία ήδη υπάρχουσα σελίδα δεδομένων, τότε προσθέτουμε σε αυτήν τη σελίδα δεδομένων, η οποία έχει διαφορά μικρότερη του συγκεκριμένου ορίου από την υπό αποθήκευση σελίδα δεδομένων έναν επιπλέον δείκτη, στον οποίο κι αποθηκεύουμε τη διεύθυνση της πρώτης ελεύθερης εγγραφής καταγραφής του τομέα. Εν συνεχεία, μεταβαίνουμε σε αυτήν την εγγραφή καταγραφής κι αποθηκεύουμε σε αυτήν την τιμή της διαφοράς με κατάλληλο πρόσημο, ενώ παράλληλα θέτουμε το bit καταγραφής αυτής ίσο με 1.

ii. Αν βρεθεί ότι η υπό αποθήκευση σελίδα δεδομένων διαφέρει λιγότερο από το συγκεκριμένο όριο από μία ήδη υπάρχουσα εγγραφή καταγραφής τότε προσθέτουμε σε αυτήν την εγγραφή καταγραφής, η οποία έχει διαφορά μικρότερη του συγκεκριμένου ορίου από την υπό αποθήκευση σελίδα δεδομένων έναν δείκτη, στον οποίο κι αποθηκεύουμε τη διεύθυνση της πρώτης ελεύθερης εγγραφής καταγραφής του τομέα. Εν συνεχεία, μεταβαίνουμε σε αυτήν την εγγραφή καταγραφής κι αποθηκεύουμε σε αυτήν την τιμή της διαφοράς με κατάλληλο πρόσημο, ενώ παράλληλα θέτουμε το bit καταγραφής αυτής ίσο με 1.

2.5.5 Αλγόριθμος ενημέρωσης της τιμής μίας σελίδας δεδομένων

1. Εφαρμόζουμε τον αλγόριθμο αναζήτησης, προκειμένου να βρούμε τη σελίδα δεδομένων, της οποίας την τιμή επιθυμούμε να αλλάξουμε, δηλαδή να ενημερώσουμε. Έτσι βρίσκουμε και μεταβαίνουμε στο μπλοκ διαγραφής, στο οποίο αυτή ανήκει.

2. Υπολογίζουμε τη διαφορά μεταξύ της παλιάς μη ενημερωμένης τιμής αυτής της σελίδας δεδομένων και της νέας ενημερωμένης τιμής αυτής και συγκρίνουμε τη διαφορά αυτή με το όριο που έχει τεθεί για την αποθήκευση μίας σελίδας δεδομένων σε μία εγγραφή καταγραφής και

   i. Αν η διαφορά αυτή είναι μεγαλύτερη του ορίου ή ναι μεν είναι μικρότερη του ορίου, αλλά δεν υπάρχει καμία ελεύθερη θέση εγγραφής καταγραφής στη λίστα στην οποία ανήκει η σελίδα δεδομένων με την παλιά μη ενημερωμένη τιμή, τότε εφαρμόζουμε τον αλγόριθμο εισαγωγής μίας νέας σελίδας δεδομένων για τη νέα ενημερωμένη τιμή. Παράλληλα, ακυρώνουμε κι άρα ελευθερώνουμε τη σελίδα δεδομένων που περιέχει την παλιά μη ενημερωμένη τιμή, θέτοντας το bit αναφοράς αυτής ίσο με 0.

   ii. Αν όμως, η διαφορά αυτή είναι μικρότερη του ορίου κι επιπλέον υπάρχει μία ελεύθερη θέση εγγραφής καταγραφής στη λίστα στην οποία ανήκει η σελίδα δεδομένων με την παλιά με ενημερωμένη τιμή, τότε εφαρμόζουμε το βήμα (3.ii) του αλγορίθμου εισαγωγής μίας νέας σελίδας δεδομένων για τη νέα ενημερωμένη τιμή , ενώ παράλληλα θέτουμε το bit αναφοράς της σελίδας δεδομένων της παλιάς μη ενημερωμένης τιμής ίσο με 0.

Προκειμένου να γίνουν ακόμη πιο κατανοητά τα προαναφερθέντα θα παρουσιάσουμε στη συνέχεια δύο σχετικά παραδείγματα.

2.5.6 Παράδειγμα 2.3 – Αποθήκευση δεδομένων σε μνήμη flash με επαναχρησιμοποίηση δεδομένων

Ας θεωρήσουμε ένα WSN, το οποίο χρησιμοποιείται για τη μέτρηση της ατμοσφαιρικής ρύπανσης σε μία σύγχρονη μεγαλούπολη και το οποίο αποτελείται από μία πληθώρα SNs που βρίσκονται εγκατεστημένα σε διάσπαρτα σημεία εντός της πόλης. Κάθε SN αυτού του WSN, διαθέτει έναν ενσωματωμένο αισθητήρα για τη

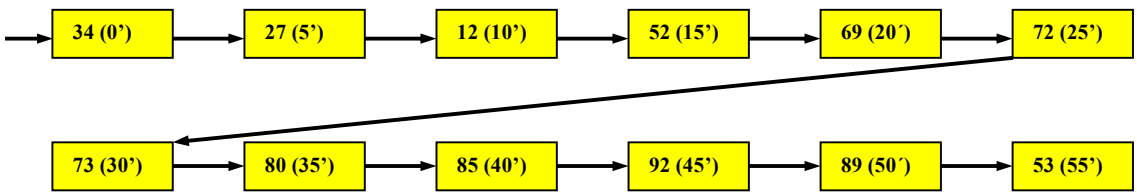
μέτρηση της συγκέντρωσης των μικροσωματιδίων στην ατμόσφαιρα. Ο αισθητήρας αυτός λαμβάνει μετρήσεις της συγκέντρωσης των αιωρούμενων σωματιδίων κάθε 5 λεπτά, οι οποίες κι αποθηκεύονται στη συνέχεια στη μνήμη NAND flash του αντίστοιχου SN. Κάθε SN είναι υποχρεωμένος να διατηρεί στη μνήμη του, τα δεδομένα των μετρήσεων που έχουν πραγματοποιηθεί εντός της τελευταίας ώρας, ενώ αποστέλλει περιοδικά τα δεδομένα αυτά σε μία κεντρική βάση αποθήκευσης δεδομένων (Sink). Στη συνέχεια, θα επικεντρωθούμε στις μετρήσεις ενός συγκεκριμένου αισθητήρα και για την ακρίβεια, στον τρόπο με τον οποίο αποθηκεύονται αυτές στη μνήμη NAND flash του SN, στον οποίο κι έχει εγκατασταθεί ο αισθητήρας αυτός. Απώτερος σκοπός μας είναι να γίνουν κατανοητά τα πλεονεκτήματα, τα οποία μας παρέχει η τεχνική διαχείρισης αποθήκευσης σε μνήμη flash με επαναχρησιμοποίηση δεδομένων, την οποία κι εξετάζουμε στην παρούσα ενότητα. Χάριν ευκολίας, θα θεωρήσουμε ότι κάθε τομέας της μνήμης NAND flash του SN αποτελείται από 6 σελίδες δεδομένων.

Έστω λοιπόν, ότι οι μετρήσεις του αισθητήρα που εξετάζουμε κατά τη διάρκεια της τελευταίας ώρας είναι αυτές που εμφανίζονται στον ακόλουθο πίνακα.

Α/Α Μέτρησης	Τιμή μέτρησης	Χρόνος μέτρησης
1	34 (mg/m^3)	0’ (λεπτά)
2	27 (mg/m^3)	5’ (λεπτά)
3	12 (mg/m^3)	10’ (λεπτά)
4	52 (mg/m^3)	15’ (λεπτά)
5	69 (mg/m^3)	20’ (λεπτά)
6	72 (mg/m^3)	25’ (λεπτά)
7	73 (mg/m^3)	30’ (λεπτά)
8	80 (mg/m^3)	35’ (λεπτά)
9	85 (mg/m^3)	40’ (λεπτά)
10	92 (mg/m^3)	45’ (λεπτά)
11	89 (mg/m^3)	50’ (λεπτά)
12	53 (mg/m^3)	55’ (λεπτά)

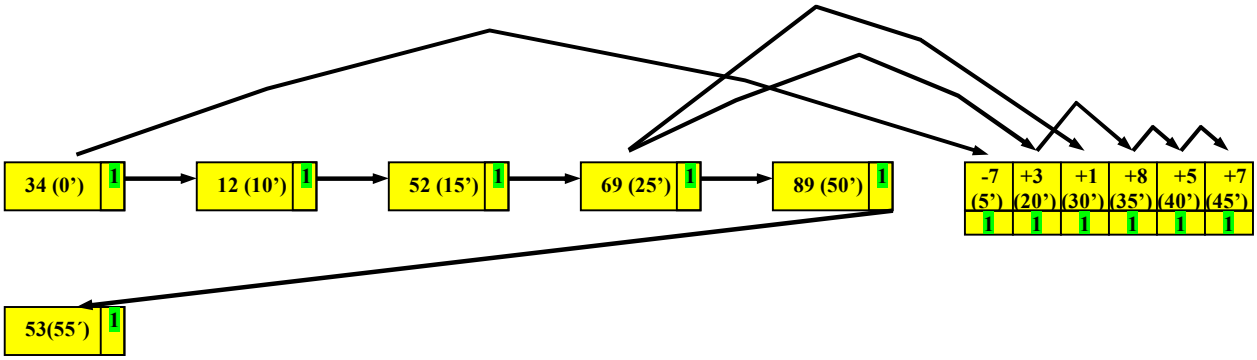
Πίνακας 2.2: Μετρήσεις αισθητήρων παραδείγματος 2.3

Σε πρώτη φάση θα εξετάσουμε τον τρόπο αποθήκευσης των δεδομένων των μετρήσεων του αισθητήρα στον SN, χωρίς επαναχρησιμοποίηση δεδομένων. Όπως είναι προφανές από το παρακάτω σχήμα, για την αποθήκευση των δεδομένων του αισθητήρα στη μνήμη NAND flash του SN, απαιτείται χρήση 2 τομέων και 10 σελίδων δεδομένων αυτής.



Σχήμα 2.9: Αποθήκευση των δεδομένων στη μνήμη NAND flash χωρίς επαναχρησιμοποίηση δεδομένων

Στη συνέχεια, θα εξετάσουμε τον τρόπο αποθήκευσης των δεδομένων των μετρήσεων του αισθητήρα στον SN, με επαναχρησιμοποίηση δεδομένων. Για το σκοπό αυτό, θα θεωρήσουμε ότι η σε κάθε τομέα η έκτη σελίδα αυτού χρησιμοποιείται για την αποθήκευση των εγγραφών καταγραφής. Επιπλέον, δεχόμαστε ότι η αποθήκευση μίας σελίδας δεδομένων σε μία εγγραφή καταγραφής λαμβάνει χώρα όταν η εμπεριεχόμενη τιμή σε αυτήν τη σελίδα δεδομένων διαφέρει από την τιμή κάποιας από τις υπάρχουσες σελίδες δεδομένων του τομέα αυτού το πολύ μέχρι 10 μονάδες. Πιο συγκεκριμένα, σε αυτήν την περίπτωση θα χρησιμοποιήσουμε την έκτη σελίδα δεδομένων κάθε τομέα για την αποθήκευση των αρχείων καταγραφής (log αρχείων).



Σχήμα 2.9: Αποθήκευση των δεδομένων στη μνήμη NAND flash με επαναχρησιμοποίηση δεδομένων

- 1η μέτρηση) 34

34 μπαίνει στην πρώτη θέση της πρώτης λίστας

- 2η μέτρηση) 27

27 - 34 = -7. Άρα η διαφορά -7< =|10| μπαίνει στην πρώτη εγγραφή καταγραφής. Παράλληλα, τοποθετείται στην πρώτη σελίδα δεδομένων, η οποία κι εμπεριέχει την τιμή 34, ένας δείκτης προς την πρώτη εγγραφή καταγραφής.

- 3η μέτρηση) 12

12 μπαίνει στην πρώτη θέση της πρώτης λίστας
12 –34 = -22 > 10 και 12 – 34 + 7 = 15 > 10. Άρα μπαίνει στην δεύτερη σελίδα δεδομένων της πρώτης λίστας.

- 4η μέτρηση) 52

52 –34 = 18 > 10 και 52 – 34 + 7 = 25 > 10. Άρα μπαίνει στην δεύτερη σελίδα δεδομένων της πρώτης λίστας.

- 5η μέτρηση) 69

69 –34 = 35 > 10 και 69 – 34 + 7 = 42 > 10 και 69 –52 = 17 < 10. Άρα μπαίνει στην τρίτη σελίδα δεδομένων της πρώτης λίστας

- 6η μέτρηση) 72

72 –34 = 38 > 10 και 72 – 34 + 7 = 45 > 10 και 72 –52 = 20 > 10 και 72 – 69 = 3 < 10. Άρα, η προκύπτουσα διαφορά +3 μπαίνει στη δεύτερη εγγραφή καταγραφής.

Παράλληλα, τοποθετείται στην τρίτη σελίδα δεδομένων, η οποία κι εμπεριέχει την τιμή 69, ένας δείκτης προς τη δεύτερη εγγραφή καταγραφής.

- $7^{η}$ μέτρηση) 73

73 –34 = 39 > 10 και 73 – 34 + 7 = 46 > 10 και 73 –52 = 20 > 10 και 73 – 69 = 4 <10. Άρα, η προκύπτουσα διαφορά +4 μπαίνει στην τρίτη εγγραφή καταγραφής. Παράλληλα, τοποθετείται στην τρίτη σελίδα δεδομένων, η οποία κι εμπεριέχει την τιμή 69, ένας δείκτης προς την τρίτη εγγραφή καταγραφής.

- $8^{η}$ μέτρηση) 80

80 –34 = 46 > 10 και 80 – 34 + 7 = 53 > 10 και 80 –52 = 38 > 10 και 80 – 69 = 11 >10 και 80 – 69 – 3 = 8 <10. Άρα, η προκύπτουσα διαφορά +8 μπαίνει στην τέταρτη εγγραφή καταγραφής. Παράλληλα, τοποθετείται στη δεύτερη εγγραφή καταγραφής, η οποία κι εμπεριέχει την τιμή 73, ένας δείκτης προς την τέταρτη εγγραφή καταγραφής.

- $9^{η}$ μέτρηση) 85

85 –34 = 51 > 10 και 85 – 34 + 7 = 58 > 10 και 85 –52 = 43 > 10 και 85 – 69 = 16 >10 και 85 – 69 – 3 = 8 > 1015 και 85 – 69 –3 –6 = 5 <10 . Άρα, η προκύπτουσα διαφορά +5 μπαίνει στην πέμπτη εγγραφή καταγραφής. Παράλληλα, τοποθετείται στην τέταρτη εγγραφή καταγραφής, η οποία κι εμπεριέχει την τιμή 80, ένας δείκτης προς την πέμπτη εγγραφή καταγραφής.

- $10^{η}$ μέτρηση) 92

92 –34 = 58 > 10 και 92 – 34 + 7 = 65 > 10 και 92 –52 = 40 > 10 και 92 – 69 = 23 >10 και 92 – 69 – 3 = 20 > 10 και 92 –69 –1 = 22 >10 και 92 – 69 –3 –8 = 12 > 10 και 92 –69 –3 –8 –5 = 7 < 10. Άρα, η προκύπτουσα διαφορά +7 μπαίνει στην έκτη εγγραφή καταγραφής. Παράλληλα, τοποθετείται στην πέμπτη εγγραφή καταγραφής, η οποία κι εμπεριέχει την τιμή 85, ένας δείκτης προς την έκτη εγγραφή καταγραφής.

- $11^{η}$ μέτρηση) 89

Ή πρώτη λίστα διαθέτει μόνον μία ελεύθερη θέση δεδομένων και καμία ελεύθερη εγγραφή καταγραφής. Άρα, θα μπει στην πέμπτη θέση της πρώτης λίστας.

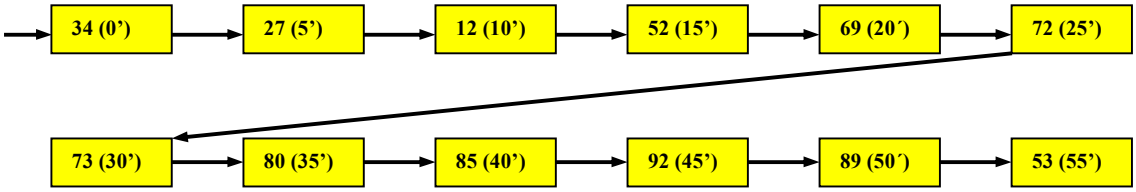
- $12^{η}$ μέτρηση) 53

Ή πρώτη λίστα δεν διαθέτει καμία ελεύθερη θέση δεδομένων και καμία ελεύθερη εγγραφή καταγραφής. Άρα, θα μπει στην πρώτη θέση της δεύτερης λίστας.

Από το παραπάνω παράδειγμα είναι προφανές, ότι χωρίς την τεχνική με την επαναχρησιμοποίηση των δεδομένων, καταναλώθηκαν 12 σελίδες δεδομένων για την αποθήκευση των μετρήσεων του αισθητήρα. Αντιθέτως, χρησιμοποιώντας την τεχνική αυτή επιτύχαμε να καταναλώσουμε μόλις 7 θέσεις μνήμης για την αποθήκευση των δεδομένων. Άρα, καθίσταται σαφές, ότι ένα από τα σημαντικότερα πλεονεκτήματα της εν λόγω τεχνικής είναι η εξοικονόμηση θέσεων μνήμης, γεγονός ιδιαίτερα σημαντικό για την περιορισμένης χωρητικότητας μνήμη των SNs.

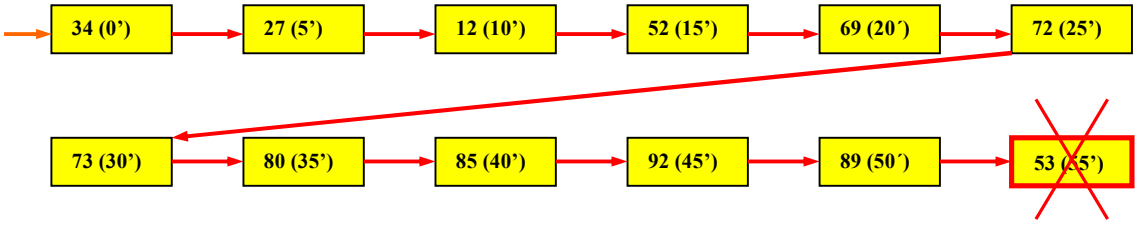
2.5.7 Παράδειγμα 2.4 – Ενημέρωση δεδομένων σε μνήμη flash με επαναχρησιμοποίηση δεδομένων

Επανερχόμενοι στο προηγούμενο παράδειγμα και προκειμένου να γίνει κατανοητό ένα ακόμη σημαντικό πλεονέκτημα που μας παρέχει η τεχνική διαχείρισης αποθήκευσης σε μνήμη flash με επαναχρησιμοποίηση δεδομένων, θα εξετάσουμε στη συνέχεια τον τρόπο με τον οποίο λαμβάνει χώρα η ενημέρωση μίας θέσης μνήμης σε αυτήν. Πιο συγκεκριμένα, ας θεωρήσουμε την πρώτη περίπτωση του παραπάνω παραδείγματος, κατά την οποία η αποθήκευση των δεδομένων στη μνήμη NAND flash έχει πραγματοποιηθεί χωρίς επαναχρησιμοποίηση δεδομένων. Τότε, όπως είδαμε προηγουμένως το περιεχόμενο του μπλοκ μνήμης που εξετάζουμε, θα έχει ως εξής:



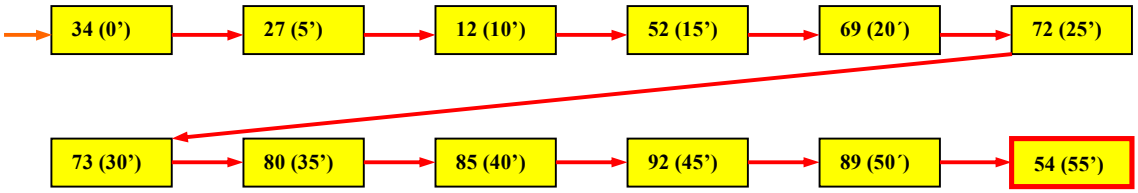
Έστω τώρα, ότι θέλουμε να ενημερωθεί, δηλαδή να αλλάξει η τιμή της τελευταίας μέτρησης του αισθητήρα κι από 53 να μετατραπεί σε 54. Θα μπορούσε κανείς να ισχυριστεί, ότι κάτι τέτοιο μπορεί εύκολα να πραγματοποιηθεί αλλάζοντας απλά την τιμή της αντίστοιχης σελίδας δεδομένων, όπως συμβαίνει για παράδειγμα σε ανάλογες περιπτώσεις στους μαγνητικούς δίσκους. Όμως, λόγω του περιορισμού της διαγραφής πριν από την εγγραφή στη μνήμη flash, κάτι τέτοιο δεν είναι εφικτό στην πράξη. Αντιθέτως, θα πρέπει πρώτα να διαγραφεί η σελίδα δεδομένων, η οποία και περιέχει την τιμή 53, όπως επίσης και ολόκληρο το μπλοκ δεδομένων, στο οποίο ανήκει η σελίδα αυτή. Εν συνεχεία, θα πρέπει ολόκληρο το μπλοκ αυτό, το οποίο και για ευνόητους λόγους ονομάζεται μπλοκ διαγραφής να ξαναγραφεί με την ενημερωμένη πλέον τιμή αυτής της σελίδας δεδομένων σε κάποια άλλη θέση μνήμης. Δηλαδή, προκειμένου να γίνει η ενημέρωση της συγκεκριμένης σελίδας μνήμης θα πρέπει να λάβουν χώρα οι ακόλουθες ενέργειες:

1^{ον}) Εύρεση και διαγραφή της σελίδας δεδομένων που περιέχει την τιμή 53.



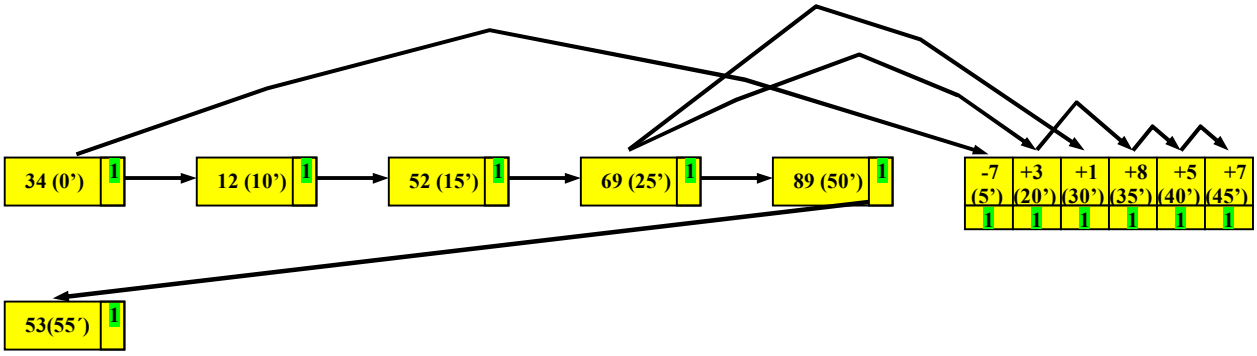
2^{ον}) Διαγραφή ολόκληρου του μπλοκ, στο οποίο ανήκει αυτή η σελίδα δεδομένων, δηλαδή ολόκληρου του μπλοκ διαγραφής από τη θέση μνήμης στην οποία βρίσκεται κι εν συνεχεία εγγραφή αυτού σε μία νέα θέση μνήμης.

3^{ον}) Εγγραφή στο μπλοκ διαγραφής και για τη νέα θέση μνήμης, στη οποία βρίσκεται αυτό της ενημερωμένης πλέον τιμής 54.



Σχήμα 2.11: Ενημέρωση των δεδομένων στη μνήμη NAND flash χωρίς επαναχρησιμοποίηση δεδομένων

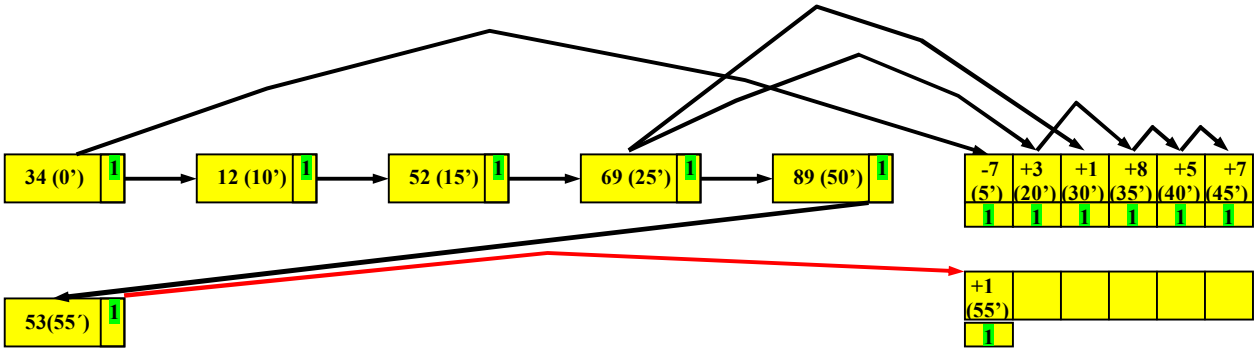
Δηλαδή κι όπως παρατηρούμε, για να ολοκληρωθεί η διαδικασία της ενημέρωσης της τιμής μίας σελίδας δεδομένων στο παραπάνω παράδειγμα απαιτείται να λάβουν χώρα 12 λειτουργίες εγγραφής και 12 λειτουργίες διαγραφής σελίδων δεδομένων.
Ας επαναλάβουμε τώρα την ίδια διαδικασία για την περίπτωση, κατά την οποία η αποθήκευση των δεδομένων στη μνήμη flash έχει πραγματοποιηθεί με επαναχρησιμοποίηση δεδομένων. Τότε και όπως είδαμε παραπάνω το περιεχόμενο της μνήμης flash, θα έχει ως εξής:



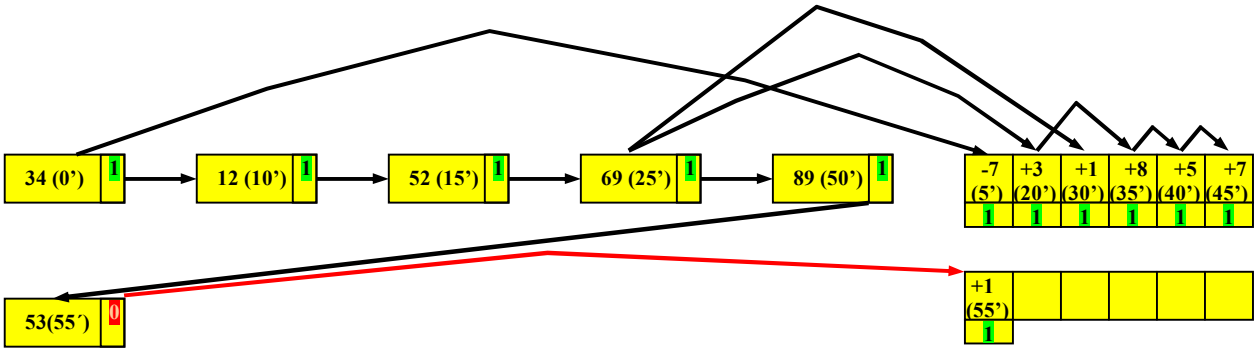
Σε αυτήν την περίπτωση προκειμένου να ενημερώσουμε τη σελίδα δεδομένων, η οποία περιέχει την τιμή 53, μετατρέποντας την τελευταία σε 54, θα εκτελέσουμε τις παρακάτω ενέργειες:

1^{ov}) Υπολογισμός της διαφοράς μεταξύ της παλιάς και της νέας τιμής της υπό ενημέρωση σελίδας δεδομένων και σύγκριση αυτής με το όριο των 10 μονάδων που έχει τεθεί για την προσθήκη μίας εγγραφής καταγραφής σε μία σελίδα δεδομένων.
$54 - 53 = 1 < 10$.

2^{ov}) Αφού η παραπάνω διαφορά είναι μικρότερη του 10, μπορούμε να προσθέσουμε μία εγγραφή καταγραφής για τη νέα τιμή της σελίδας δεδομένων χωρίς να διαγράψουμε την παλαιά τιμή αυτής, με δεδομένο φυσικά ότι υπάρχουν ελεύθερες θέσεις εγγραφής στη λίστα, στην οποία ανήκει η συγκεκριμένη σελίδα δεδομένων. Έτσι λοιπόν, προσθέτουμε μία νέα εγγραφή καταγραφής για την πρώτη σελίδα της δεύτερης λίστας δεδομένων, η οποία θα περιέχει την τιμή +1 και στην οποία θα δείχνει η πρώτη σελίδα της δεύτερης λίστας δεδομένων.



3ον) Με τον τρόπο αυτό λοιπόν, θα έχουμε τόσο την τιμή 53, όσο και την τιμή 54 αποθηκευμένες στο μπλοκ δεδομένων. Άρα, θα πρέπει με κάποιον τρόπο να καταστήσουμε έγκυρη την ενημερωμένη τιμή 53 κι άκυρη την παλαιά τιμή 54. Για να το πετύχουμε αυτό θα θέσουμε απλά το δείκτη εγκυρότητας της σελίδας δεδομένων με την παλιά τιμή 53 ίσο με 0, με αποτέλεσμα όταν διατρέξουμε κατά την αναζήτηση δεδομένων, το εν λόγω μπλοκ δεδομένων, να παρακάμψουμε αυτήν τη σελίδα δεδομένων θεωρώντας την άκυρη και να μεταβούμε απευθείας στην εγγραφή καταγραφής με την έγκυρη τιμή 53 + 1 = 54.



Σχήμα 2.12: Ενημέρωση των δεδομένων στη μνήμη NAND flash με επαναχρησιμοποίηση δεδομένων

Δηλαδή κι όπως παρατηρούμε, για να ολοκληρωθεί η διαδικασία της ενημέρωσης της τιμής μίας σελίδας δεδομένων όταν χρησιμοποιείται επαναχρησιμοποίηση δεδομένων απαιτείται να λάβει χώρα μόλις μία διαδικασία εγγραφής.

Από όλα τα ανωτέρω, καθίσταται σαφές το ότι με την τεχνική βελτιώνουμε την αποδοτικότητα ενός WSN ανταλλάσσοντας τις αυξημένου κόστους επανεγγραφές με τις μειωμένου κόστους αναγνώσεις κι εξοικονομώντας παράλληλα πολύτιμους πόρους μνήμης.

Κεφάλαιο 3

Τοπική και κατανεμημένη ευρετηρίαση σε ένα WSN

Περιεχόμενα

Στο προηγούμενο κεφάλαιο επικεντρώσαμε τη μελέτη μας σε τεχνικές διαχείρισης της αποθήκευσης στη μνήμη flash των SNs ενός WSN, οι οποίες κι αποσκοπούν στη βελτιστοποίηση της λειτουργικότητας και της απόδοσης ενός WSN. Στο τρέχον κεφάλαιο, θα μελετήσουμε την ευρετηρίαση των δεδομένων που βρίσκονται αποθηκευμένα σε ένα WSN. Πιο συγκεκριμένα, τόσο η Sink όσο και κάθε SN διατηρούν ένα τοπικό ευρετήριο, το οποίο κι αποτελεί στην ουσία έναν κατάλογο με τα δεδομένα που βρίσκονται αποθηκευμένα σε αυτόν τον SN. Επιπλέον, υπάρχει ένα κατανεμημένο ευρετήριο μεταξύ όλων των SNs κάθε συστάδας, στο οποίο και περιέχονται τα δεδομένα όλων των SNs της συστάδας αυτής. Επίσης, υπάρχει ένα κατανεμημένο ευρετήριο μεταξύ όλων των head SNs με όλα τα δεδομένα των head SNs. Σκοπός μας στο παρών κεφάλαιο είναι να προτείνουμε τοπικές και κατανεμημένες τεχνικές ευρετηρίασης, μέσω των οποίων θα επιτευχθεί η εύρεση των αιτούμενων δεδομένων στο συντομότερο χρονικό διάστημα και με το ελάχιστο δυνατό πλήθος αποστελλόμενων μηνυμάτων στο WSN. Πρώτα όμως, θα ξεκαθαρίσουμε την έννοια των όρων ευρετήριο κι ευρετηρίαση.

3.1 Ευρετήριο κι ευρετηρίαση

Ο όρος ευρετήριο αποδίδεται σε κάθε κατάλογο, στον οποίο είναι συστηματικά καταχωρημένα διάφορα στοιχεία, καθώς και η θέση όπου βρίσκεται καθένα από αυτά. Η ευρετηρίαση αντιστοιχεί στη σύνταξη ενός ευρετηρίου κι αποτελεί μία κοινά χρησιμοποιούμενη μέθοδο για την επίσπευση της πρόσβασης σε αποθηκευμένα δεδομένα.

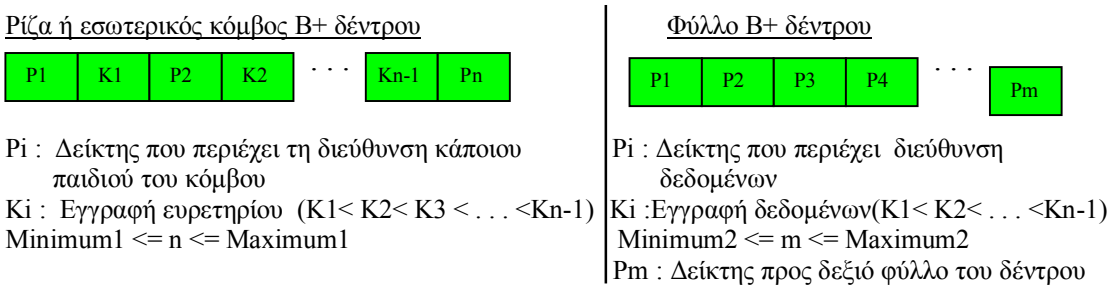
3.2 Τοπικό ευρετήριο

Όπως είπαμε, η Sink και κάθε SN ενός WSN διαθέτουν ένα τοπικό ευρετήριο, το οποίο κι αποτελεί στην ουσία έναν κατάλογο με τα δεδομένα που έχουν αποθηκευμένα στη μνήμη τους. Στην ενότητα αυτή, θα εξετάσουμε την καταλληλότερη δομή δεδομένων για την υλοποίηση του τοπικού ευρετηρίου. Μία πρώτη σκέψη είναι να χρησιμοποιήσουμε ως ευρετήριο των δεδομένων που βρίσκονται αποθηκευμένα στη μνήμη NAND flash ενός SN, ένα B+ δέντρο, το οποίο κι αποτελεί τη δομή ευρετηρίου των μαγνητικών δίσκων.

3.2.1 B+ δέντρο

Ένα B+ δέντρο είναι ένα ισοζυγισμένο δέντρο αναζήτησης, γεγονός που σημαίνει ότι όλα τα μονοπάτια που ξεκινούν από τη ρίζα αυτού και καταλήγουν σε οποιοδήποτε φύλλο αυτού έχουν το ίδιο μήκος. Σε ένα B+ δέντρο, δεν είναι απαραίτητο όλοι οι κόμβοι αυτού να έχουν το ίδιο πλήθος παιδιών μεταξύ τους. Η ρίζα και κάθε εσωτερικός κόμβος ενός B+ δέντρου αποτελείται από ένα σύνολο εγγραφών ευρετηρίου (keys) κι από ένα σύνολο δεικτών (pointers). Κάθε εγγραφή ευρετηρίου αποτελεί μία τιμή, η οποία μας βοηθά κατά την αναζήτηση των δεδομένων που βρίσκονται αποθηκευμένα σε ένα B+ δέντρο. Κάθε δείκτης εμπεριέχει τη διεύθυνση της ρίζας ενός από τα υπόδεντρα αυτού του κόμβου, δηλαδή τη διεύθυνση ενός από τα παιδιά αυτού του κόμβου. Οι εγγραφές ευρετηρίου της ρίζας και κάθε εσωτερικού κόμβου ενός B+ δέντρου είναι ταξινομημένες σε αύξουσα σειρά από αριστερά προς τα δεξιά, ενώ είναι πάντα λιγότερες κατά 1 από τους δείκτες

του κόμβου αυτού. Επιπλέον, το πλήθος των εγγραφών ευρετηρίου και των δεικτών ενός εσωτερικού κόμβου έχει ένα ελάχιστο κι ένα μέγιστο όριο. Κάθε φύλλο ενός B+ δέντρου αποτελείται από ένα σύνολο εγγραφών δεδομένων κι από ένα πλήθος δεικτών. Η μόνη διαφορά με τους εσωτερικούς κόμβους είναι ότι οι δείκτες ενός φύλλου περιέχουν τη διεύθυνση των αποθηκευμένων δεδομένων, των οποίων ευρετήριο αποτελεί το B+ δέντρο. Τα πλήθη των εγγραφών ευρετηρίου και των δεικτών ενός φύλλου ενός B+ δέντρου διαθέτουν επίσης ένα ελάχιστο κι ένα μέγιστο όριο, τα οποία και μπορεί να διαφέρουν από τα αντίστοιχα όρια ενός εσωτερικού κόμβου.



Σχήμα 3.1: Περιεχόμενο κόμβων ενός B+ δέντρου

Στη συνέχεια, θα παρουσιάσουμε έναν αλγόριθμο αναζήτησης κι έναν αλγόριθμο εισαγωγής μίας εγγραφής σε ένα B+ δέντρο. Με την έννοια εγγραφή εννοούμε ουσιαστικά τη διεύθυνση κάποιων από τα αποθηκευμένα δεδομένα, των οποίων ευρετήριο αποτελεί το B+ δέντρο.

3.2.2 Αλγόριθμος αναζήτησης εγγραφής δεδομένων σε ένα B+ δέντρο

1.Ξεκινάμε από τη ρίζα του δέντρου.

- Αν αυτή δεν έχει ούτε παιδιά ούτε κάποια εγγραφή δεδομένων, τότε τερματίζεται ο αλγόριθμος χωρίς να επιστραφεί αποτέλεσμα.

- Αν έχει δεν έχει παιδιά, αλλά έχει κάποια εγγραφή δεδομένων, τότε εξετάζουμε αν αυτή είναι ίδια με την εγγραφή που αναζητούμε. Αν ναι, τότε την επιστρέφουμε ως αποτέλεσμα και τερματίζουμε τον αλγόριθμο, ενώ διαφορετικά τερματίζουμε τον αλγόριθμο χωρίς να επιστραφεί αποτέλεσμα.

- Αν έχει παιδιά, τότε βρίσκουμε την εγγραφή ευρετηρίου με ίση ή την αμέσως μικρότερη τιμή από την εγγραφή δεδομένων που αναζητούμε, πάμε στο δείκτη που ακολουθεί αυτή την εγγραφή ευρετηρίου και μεταβαίνουμε στο παιδί της ρίζας στο οποίο δείχνει αυτός ο δείκτης. Δηλαδή, αν η εγγραφή ευρετηρίου με την αμέσως μικρότερη τιμή είναι η Ki, τότε πάμε στο δείκτη Pi+1 και μεταβαίνουμε στον κόμβο στον οποίο δείχνει αυτός.

2. Εξετάζουμε τον τρέχοντα κόμβο

- Αν ο κόμβος αυτός είναι εσωτερικός κόμβος, τότε βρίσκουμε την εγγραφή ευρετηρίου με ίση ή την αμέσως μικρότερη τιμή από την εγγραφή δεδομένων που αναζητούμε, πάμε στο δείκτη που ακολουθεί αυτή την εγγραφή ευρετηρίου και μεταβαίνουμε στο παιδί αυτού του εσωτερικού κόμβου, στο οποίο και δείχνει αυτός ο δείκτης. Δηλαδή, αν η εγγραφή ευρετηρίου με την αμέσως μικρότερη τιμή είναι η Ki, τότε πάμε στο δείκτη Pi+1 και μεταβαίνουμε στον κόμβο στον οποίο δείχνει αυτός.

- Αν ο κόμβος αυτός είναι τερματικός κόμβος, δηλαδή φύλλο του δέντρου, τότε εξετάζουμε αν κάποια από τις εγγραφές δεδομένων αυτού είναι αυτή που αναζητούμε κι αν ισχύει κάτι τέτοιο την επιστρέφουμε ως αποτέλεσμα και τερματίζουμε τον αλγόριθμο, ενώ διαφορετικά τερματίζουμε τον αλγόριθμο χωρίς επιστροφή αποτελέσματος.

3. Επαναλαμβάνουμε το βήμα 3 του αλγορίθμου, μέχρι να τερματιστεί ο αλγόριθμος.

3.2.3 Απλός αλγόριθμος εισαγωγής εγγραφής δεδομένων σε ένα B+ δέντρο

Καταρχήν, οι είσοδοι στο συγκεκριμένο αλγόριθμο, οι οποίες και θα δίνονται κατά την εκτέλεσή του θα είναι οι ακόλουθες:

1. Το μέγιστο πλήθος των κόμβων ενός επιπέδου του B+ δέντρου, το οποίο και θα συμβολίζουμε στη συνέχεια με max1

2. Το μέγιστο πλήθος των εγγραφών ευρετηρίου σε έναν εσωτερικό κόμβο του δέντρου, το οποίο και θα συμβολίζουμε με max2.

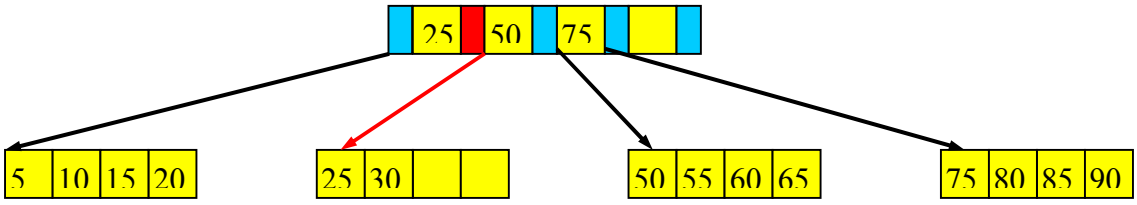
3. Το μέγιστο πλήθος των εγγραφών δεδομένων σε ένα φύλλο του δέντρου, το οποίο και θα συμβολίζουμε με max3.

Όπως είναι προφανές, το μέγιστο πλήθος των εγγραφών ευρετηρίου ενός εσωτερικού κόμβου (max2) μπορεί να διαφέρει από το μέγιστο πλήθος των εγγραφών δεδομένων ενός φύλλου του δέντρου (max3). Σε γενικές γραμμές, ο αλγόριθμος αρχίζει να εκτελείται από τη ρίζα του δέντρου και πραγματοποιώντας μεταβάσεις μεταξύ των εσωτερικών κόμβων αυτού επιλέγει το κατάλληλο φύλλο του δέντρου, στο οποίο και πρέπει να τοποθετηθεί η υπό εισαγωγή εγγραφή δεδομένων. Στη συνέχεια τοποθετείται η εγγραφή δεδομένων στο φύλλο αυτό και ελέγχεται αν το δέντρο είναι ισοζυγισμένο, καθώς κι αν ικανοποιούνται οι περιορισμοί του μέγιστου πλήθους εγγραφών ευρετηρίου σε έναν εσωτερικό κόμβο του δέντρου (max2), του μέγιστου πλήθους των εγγραφών δεδομένων σε ένα φύλλο του δέντρου (max3) και του μέγιστου πλήθους κόμβων σε ένα επίπεδο του δέντρου (max1), ούτως ώστε αν κάποια από τις προηγούμενες προϋποθέσεις δεν ισχύει, να γίνουν οι όποιοι απαραίτητοι μετασχηματισμοί στο δέντρο για την ικανοποίηση αυτής. Παράλληλα με την παρουσίαση του αλγορίθμου αυτού, θα παρουσιάσουμε για κάθε βήμα αυτού ή καλύτερα για κάθε περίπτωση αυτού κι ένα σχετικό αποσαφηνιστικό παράδειγμα. Έτσι λοιπόν, ο αλγόριθμος αυτός έχει ως εξής:

Σε κάθε επανάληψη του αλγορίθμου, εξετάζουμε τον τρέχων κόμβο, στον οποίο βρισκόμαστε και :

1)
Αν είναι εσωτερικός κόμβος, τότε βρίσκουμε την εγγραφή ευρετηρίου αυτού που έχει την αμέσως μικρότερη ή ίση τιμή από την εγγραφή δεδομένων που θέλουμε να εισάγουμε, πάμε στο δείκτη που ακολουθεί αυτή την εγγραφή ευρετηρίου και μεταβαίνουμε στο παιδί της ρίζας στο οποίο δείχνει ο δείκτης αυτός. Δηλαδή, αν η εγγραφή ευρετηρίου με την ίση ή την αμέσως μικρότερη τιμή είναι η Ki, τότε πάμε στο δείκτη Pi+1 και μεταβαίνουμε στον κόμβο στον οποίο δείχνει αυτός.

Σχήμα 3.2: Απλός αλγόριθμος εισαγωγής εγγραφής δεδομένων σε B+ δέντρο



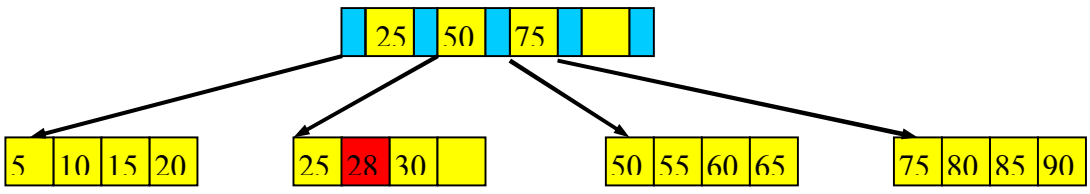
Θέλουμε να εισάγουμε την εγγραφή με τιμή 28. Στον κόμβο που εξετάζουμε, δηλαδή τη ρίζα στην προκειμένη περίπτωση, η εγγραφή με την αμέσως μικρότερη τιμή από το 28, είναι η 25. Άρα, πάμε στο δείκτη που την ακολουθεί για να μεταβούμε στον επόμενο κόμβο.

2)
Αν είναι τερματικός κόμβος, δηλαδή φύλλο του δέντρο , τότε

2.1)
Αν το πλήθος των εγγραφών δεδομένων στο φύλλο αυτό είναι μικρότερο του ανώτερου ορίου εγγραφών δεδομένων σε ένα φύλλο (max3), τότε προσθέτουμε την εγγραφή μας στην κατάλληλη θέση σε αυτό, δηλαδή δεξιά από τις μικρότερες από αυτήν εγγραφές κι αριστερά από τις μεγαλύτερες από αυτήν εγγραφές και τερματίζουμε τον αλγόριθμο.

Σχήμα 3.3: Απλός αλγόριθμος εισαγωγής εγγραφής δεδομένων σε B+ δέντρο

Προσθήκη εγγραφής 28.



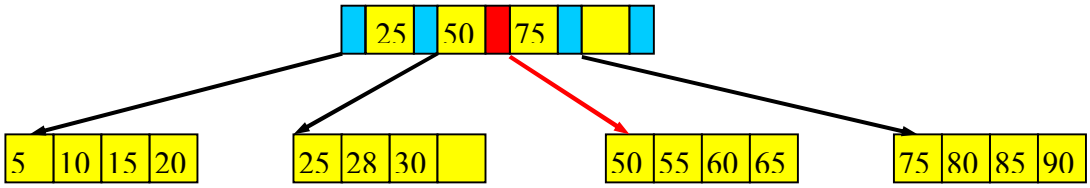
Η εγγραφή 28 τοποθετείται στην κατάλληλη θέση του κόμβου, στον οποίο βρισκόμαστε.

2.2)
Αν το πλήθος των εγγραφών δεδομένων στο φύλλο αυτό είναι ίσο με το ανώτερο όριο εγγραφών δεδομένων σε ένα φύλλο, δηλαδή είναι ίσο με το max3, τότε

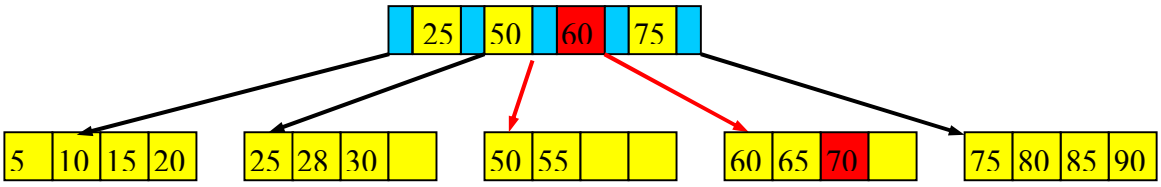
2.2.1)
Αν το πλήθος των φύλλων του δέντρου σε αυτό το επίπεδο του δέντρου όπου βρισκόμαστε είναι μικρότερο του ανώτερου πλήθους φύλλων σε ένα επίπεδο, δηλαδή είναι μικρότερο του max1, τότε εκτελούμε τη διαδικασία του διαχωρισμού σε αυτό το φύλλο του δέντρου, στο οποίο βρισκόμαστε.

Σχήμα 3.4: Απλός αλγόριθμος εισαγωγής εγγραφής δεδομένων σε B+ δέντρο

Προσθήκη εγγραφής 70



Η αμέσως μικρότερη εγγραφή είναι η 60 κι άρα με βάση το δείκτη που την ακολουθεί μεταβαίνουμε στον ανάλογο κόμβο.

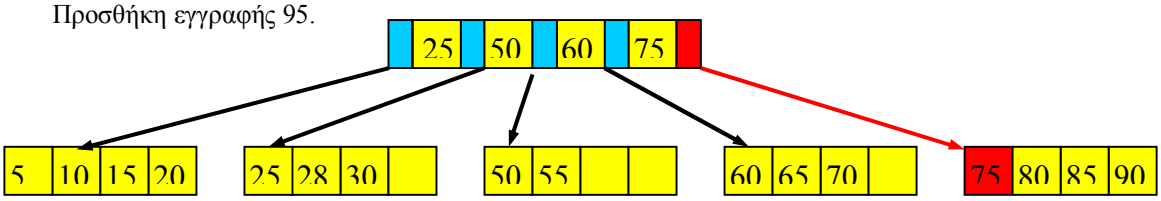


Από τη διαδικασία του διαχωρισμού, προκύπτουν τα δύο νέα φύλλα που υποδεικνύονται στο παραπάνω σχήμα, ενώ παράλληλα τοποθετείται η μεσαία εγγραφή 60 του κόμβου που διαχωρίστηκε στον κόμβο πατέρα αυτού. Η εγγραφή 70 τοποθετείται στην κατάλληλη θέση του κόμβου, στον οποίο βρισκόμαστε.

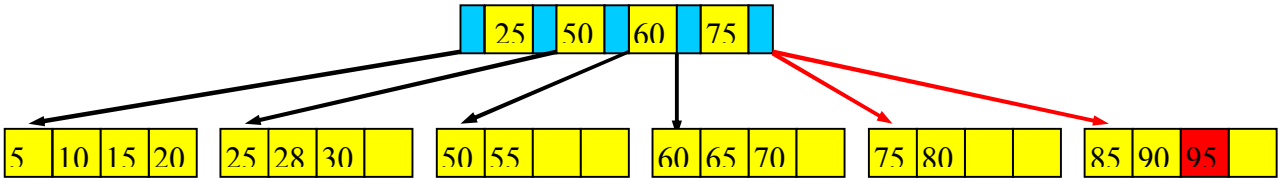
2.2.2)
Αν το πλήθος των φύλλων του δέντρου σε αυτό το επίπεδο του δέντρου όπου βρισκόμαστε είναι ίσο με το ανώτερο πλήθος φύλλων σε ένα επίπεδο, δηλαδή είναι ίσο με το max1, τότε πρώτα εκτελούμε τη διαδικασία του διαχωρισμού σε αυτό το φύλλο του δέντρου, στο οποίο βρισκόμαστε.

Σχήμα 3.5: Απλός αλγόριθμος εισαγωγής εγγραφής δεδομένων σε B+ δέντρο

Προσθήκη εγγραφής 95.



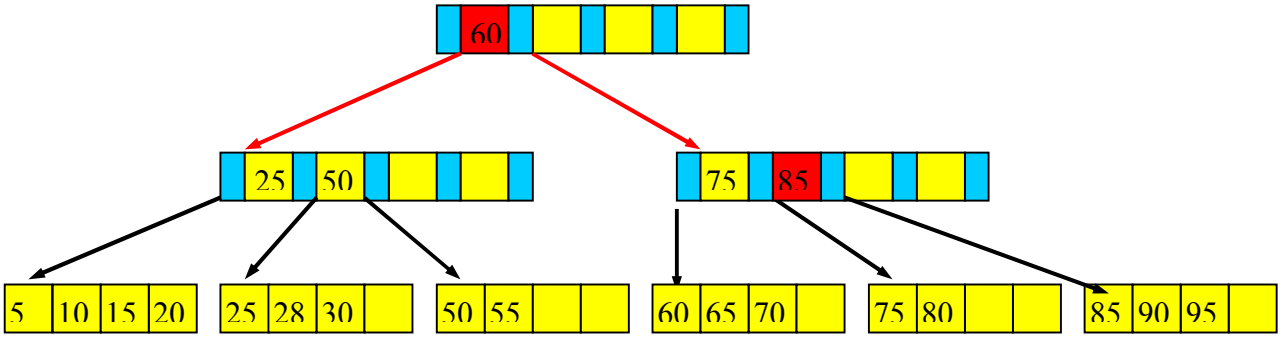
Η αμέσως μικρότερη εγγραφή είναι η 75 κι άρα με βάση το δείκτη που την ακολουθεί μεταβαίνουμε στον ανάλογο κόμβο.



Ο κόμβος αυτός διαχωρίζεται και η εγγραφή 95 τοποθετείται στην κατάλληλη θέση. Από τη διαδικασία του διαχωρισμού, προκύπτουν τα δύο νέα φύλλα που υποδεικνύονται στο παραπάνω σχήμα, ενώ παράλληλα πρέπει να τοποθετηθεί η μεσαία εγγραφή 85 του κόμβου που διαχωρίστηκε στον κόμβο πατέρα αυτού. Είναι βέβαια προφανές, ότι πλέον το πλήθος των κόμβων στο επίπεδό μας είναι μεγαλύτερο του max1. Άρα μεταβαίνουμε στη ρίζα για να τη διαχωρίσουμε.

Στη συνέχεια κι αφού έχουμε ολοκληρώσει το διαχωρισμό αυτού του φύλλου του δέντρου μεταβαίνουμε στον εσωτερικό κόμβο του δέντρου από τον οποίο και μεταβήκαμε στο φύλλο όπου βρισκόμαστε, δηλαδή από τον εσωτερικό κόμβο του δέντρου που διαθέτει δείκτη προς το φύλλο του δέντρου, στο οποίο βρισκόμαστε, προκειμένου να εισάγουμε σε αυτόν τη μεσαία εγγραφή του φύλλου του δέντρου, το οποίο και μόλις διαχωρίσαμε. Για να το πετύχουμε αυτό επαναλαμβάνουμε την εκτέλεση του αλγορίθμου και πιο συγκεκριμένα του βήματος 2.2 αυτού, για αυτόν τον εσωτερικό κόμβο.

Σχήμα 3.6: Απλός αλγόριθμος εισαγωγής εγγραφής δεδομένων σε B+ δέντρο



Ο εσωτερικός κόμβος πατέρας του κόμβου στον οποίο εισάγαμε την εγγραφή 95 διαχωρίστηκε με αποτέλεσμα να προκύψουν οι δύο νέοι κόμβοι που υποδεικνύονται από τα κόκκινα βέλη. Οι μεσαίες εγγραφές των δύο προηγηθέντων διαχωρισμών τοποθετούνται στις κατάλληλες θέσεις και πλέον το δέντρο είναι ισοζυγισμένο και ικανοποιούνται όλοι οι σχετικοί περιορισμοί.

- Διαδικασία διαχωρισμού ενός κόμβου του δέντρου

Στον παραπάνω αλγόριθμο, έχουμε καλέσει αρκετές φορές τη διαδικασία διαχωρισμού ενός κόμβου του δέντρου. Είσοδος της διαδικασίας αυτής είναι ένας κόμβος του δέντρου, ο οποίος μπορεί να είναι είτε ένας εσωτερικός κόμβος του δέντρου είτε ένα φύλλο αυτού. Έξοδος της διαδικασίας είναι οι δύο νέοι κόμβοι, οι οποίοι και προέκυψαν από το διαχωρισμό του αρχικού κόμβου, καθώς επίσης και η μεσαία εγγραφή του αρχικού κόμβου, η οποία και δίνεται προς εισαγωγή στον κόμβο-πατέρα αυτού του κόμβου στο δέντρο. Η διαδικασία αυτή, η οποία είναι στην ουσία μία ρουτίνα, δηλαδή μία ενσωματωμένη στον αλγόριθμο συνάρτηση έχει ως εξής:

1. Βρίσκουμε τη μεσαία εγγραφή του κόμβου αυτού.

2. Διαχωρίζουμε αυτόν τον κόμβο του δέντρου σε δύο νέους κόμβους

3. Τοποθετούμε στον αριστερό νέο κόμβο όλες τις εγγραφές δεδομένων με τιμή μικρότερη της τιμής της μεσαίας εγγραφής.

4. Τοποθετούμε δεξί νέο κόμβο πρώτα τη μεσαία εγγραφή και δεξιά από αυτήν, όλες τις υπόλοιπες εγγραφές με τιμή μεγαλύτερη από αυτήν.

5. Τοποθετούμε ανάλογα με την τιμή της, την εγγραφή που θέλουμε να εισάγουμε στον κατάλληλο από αυτούς τους δύο κόμβους και στην κατάλληλη θέση αυτού.

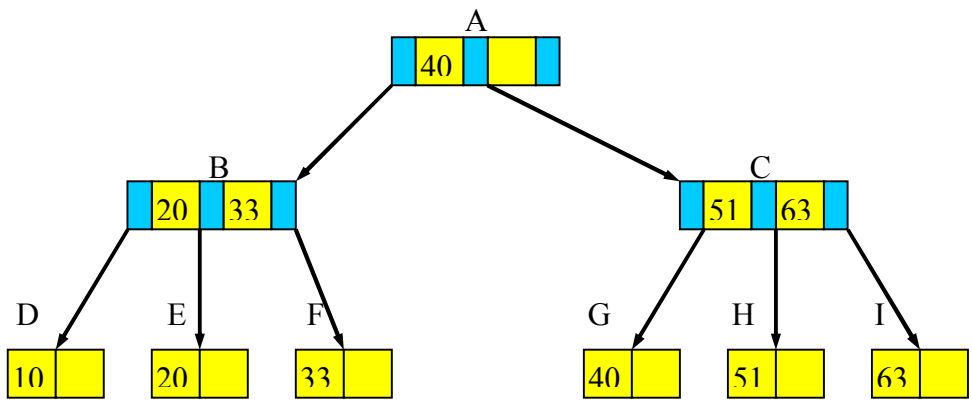
6. Τοποθετούμε τη μεσαία εγγραφή στην κατάλληλη θέση του εσωτερικού κόμβου του δέντρου από τον οποίο και μεταβήκαμε στον κόμβο όπου βρισκόμαστε, δηλαδή στον εσωτερικό κόμβο του δέντρου που διαθέτει δείκτη προς τον κόμβο του δέντρου, στο οποίο βρισκόμαστε.

Αφού λοιπόν, έχουμε εξηγήσει αναλυτικά τον παραπάνω αλγόριθμο εισαγωγής μίας εγγραφής δεδομένων σε ένα B+ δέντρο, θα μελετήσουμε στη συνέχεια την εφαρμογή του στην πράξη για τα δεδομένα που βρίσκονται αποθηκευμένα σε ένα SN ενός WSN.

3.2.4 Παράδειγμα 3.1 – Εκτέλεση του απλού αλγορίθμου εισαγωγής μίας εγγραφής δεδομένων σε ένα B+ δέντρο

Ας θεωρήσουμε λοιπόν, ότι στη μνήμη flash του SN βρίσκονται αποθηκευμένες οι εγγραφές, δηλαδή οι μετρήσεις των αισθητήρων αυτού, με τις ακόλουθες τιμές : 10, 20, 33, 40, 51, 63. Τότε, το B+ δέντρο, το οποίο χρησιμοποιείται ως ευρετήριο των εγγραφών της μνήμης flash, θα έχει ως εξής:

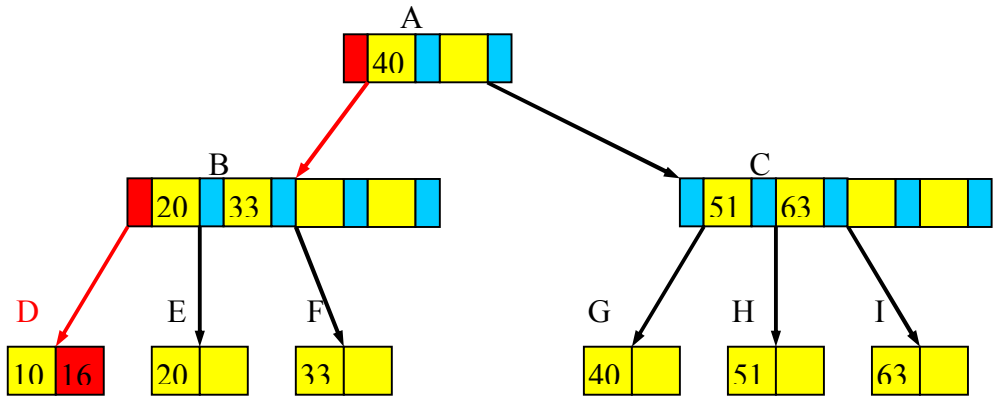
Σχήμα 3.7: Παράδειγμα εισαγωγής εγγραφής δεδομένων σε B+ δέντρο με τον απλό αλγόριθμο



Οι επόμενες μετρήσεις που καταφθάνουν στον SN από τους αισθητήρες αυτού, προκειμένου να αποθηκευτούν στη μνήμη flash αυτού έχουν τιμές: 16, 23, 18 και 25. Πρώτα λοιπόν, θα εισαχθεί η εγγραφή με τιμή 16, με αποτέλεσμα την ενημέρωση της σελίδας D.

Σχήμα 3.8: Παράδειγμα εισαγωγής εγγραφής δεδομένων σε B+ δέντρο με τον απλό αλγόριθμο

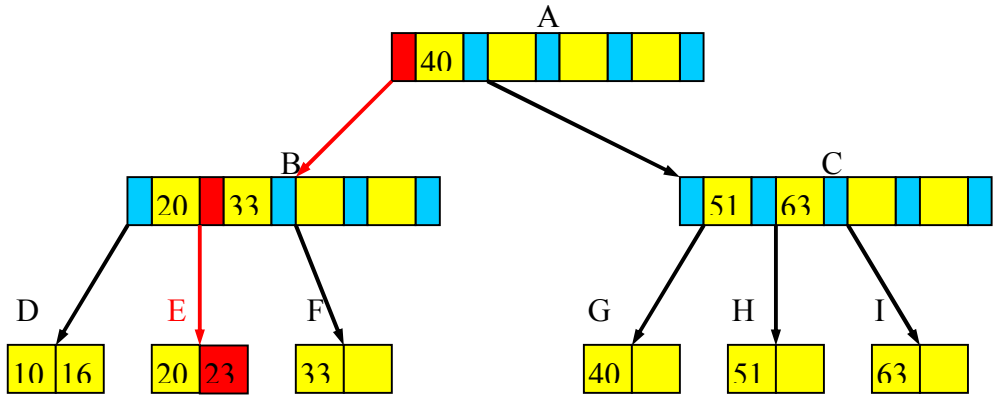
Προσθήκη εγγραφής με τιμή 16.



Στη συνέχεια θα προστεθεί η εγγραφή με τιμή 23, με αποτέλεσμα την ενημέρωση της σελίδας E.

Σχήμα 3.9: Παράδειγμα εισαγωγής εγγραφής δεδομένων σε B+ δέντρο με τον απλό αλγόριθμο

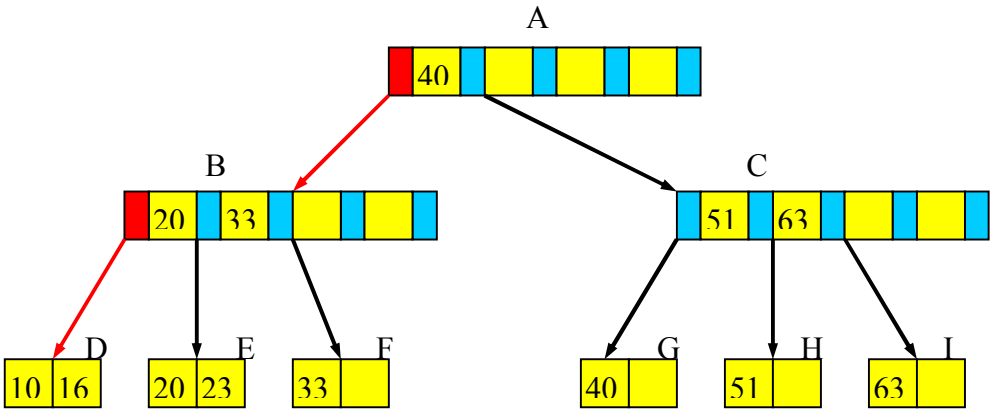
Προσθήκη εγγραφής με τιμή 23.



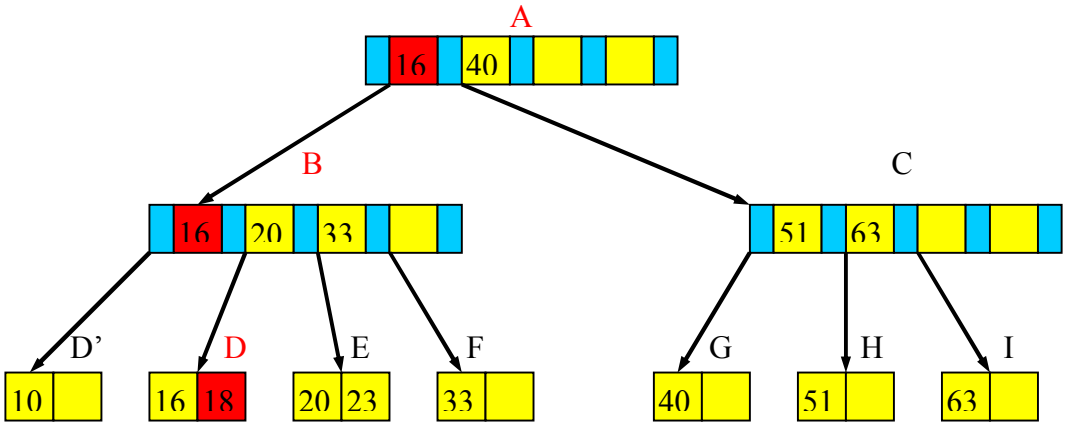
Στη συνέχεια, θα προστεθεί η εγγραφή 18, με αποτέλεσμα τους διαδοχικούς διαχωρισμούς των σελίδων D, B και τη διαδοχική ενημέρωση τελικά των σελίδων D, B και A.

Σχήμα 3.10: Παράδειγμα εισαγωγής εγγραφής δεδομένων σε B+ δέντρο με τον απλό αλγόριθμο

Προσθήκη εγγραφής με τιμή 18.



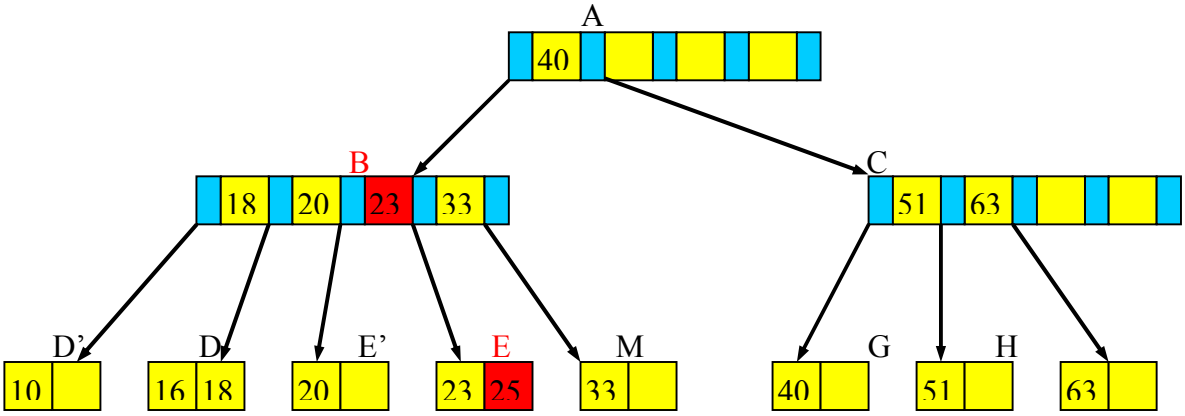
Διαχωρισμός σελίδων B, D και διαδοχική ενημέρωση σελίδων D, B και A.



Στη συνέχεια, θα προστεθεί η εγγραφή 25, με αποτέλεσμα τους διαδοχικούς διαχωρισμούς των σελίδων E, B και τη διαδοχική ενημέρωση τελικά των σελίδων E και B.

Σχήμα 3.11: Παράδειγμα εισαγωγής εγγραφής δεδομένων σε B+ δέντρο με τον απλό αλγόριθμο

Προσθήκη εγγραφής με τιμή 25.



Είναι οφθαλμοφανές λοιπόν, ότι για την προσθήκη των παραπάνω 4 εγγραφών απαιτείται η διαδοχική ενημέρωση της τιμής των σελίδων D, E, D, D, B, A, E, B. Δηλαδή, απαιτείται η ενημέρωση 8 σελίδων συνολικά.

3.2.5 Βελτιστοποιημένος αλγόριθμος εισαγωγής εγγραφής δεδομένων σε ένα B+ δέντρο

Γνωρίζοντας, ότι το κόστος της ενημέρωσης της τιμής μίας σελίδας δεδομένων που βρίσκεται αποθηκευμένη στη μνήμη flash ενός SN είναι ιδιαίτερα μεγάλο λόγω του περιορισμού της διαγραφής πριν από την εγγραφή, καθώς και του περιορισμένου πλήθους επανεγγραφών/ διαγραφών, η τεχνική που ευρετηρίου βασισμένου στη μνήμη flash, που παρουσιάζουμε στην παρούσα ενότητα προτείνει έναν βελτιωμένο αλγόριθμο εισαγωγής. Ο αλγόριθμος αυτός είναι ίδιος με αυτόν που μελετήσαμε προηγουμένως με τη βασική διαφορά, ότι οι μετρήσεις των αισθητήρων ενός SN δεν εισάγονται στη μνήμη flash απευθείας κατά την άφιξή τους σε αυτόν, αλλά πρώτα συγκεντρώνονται σε έναν χώρο άφιξης κι εν συνεχεία, αφού ο χώρος αυτός γεμίσει ταξινομούνται κατά αύξουσα σειρά και μετά εισάγονται στο B+ δέντρο. Κατά την εισαγωγή των σελίδων στο B+ δέντρο πραγματοποιούνται διαχωρισμοί, μόνο στο επίπεδο των φύλλων αυτού. Όταν και μόνον όταν ολοκληρωθεί η εισαγωγή και της τελευταίας σελίδας δεδομένων, τότε πραγματοποιούμε τους όποιους απαραίτητους διαχωρισμούς στα υπόλοιπα ανώτερα επίπεδα εσωτερικών κόμβων του δέντρου. Δηλαδή, προσθέτουμε επιπλέον στον παραπάνω αλγόριθμο έναν χώρο άφιξης των νέων σελίδων δεδομένων, στον οποίο κι αποθηκεύονται τα δεδομένα των μετρήσεων που λαμβάνονται από τους αισθητήρες του SN. Μόλις ο χώρος αυτός γεμίσει, τότε ταξινομούνται οι εντός αυτού σελίδες δεδομένων κι εν συνεχεία εισάγονται όλες μαζί στο B+ δέντρο με τους όποιους απαραίτητους διαχωρισμούς να περιορίζονται στο επίπεδο των φύλλων αυτού. Αφού ολοκληρωθεί, η εισαγωγή και της τελευταίας σελίδας δεδομένων, τότε πραγματοποιούνται οι διαχωρισμοί των εσωτερικών κόμβων των ανώτερων επιπέδων. Δηλαδή, ο βελτιστοποιημένος αλγόριθμος έχει ως εξής:

1. Μόλις ο χώρος άφιξης των σελίδων γεμίσει τότε τον ταξινομούμε σε αύξουσα σειρά από αριστερά προς τα δεξιά.

2. Εισάγουμε τις σελίδες του χώρου αφίξεων με τη σειρά στο δέντρο. Κατά την εισαγωγή αυτών πραγματοποιούμε διαχωρισμούς, μόνο στο επίπεδο των φύλλων του δέντρου.

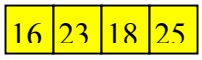
3. Μόλις εισαχθεί και η τελευταία σελίδα δεδομένων στο δέντρο, τότε αν χρειάζεται διαχωρίζουμε τους εσωτερικούς κόμβους, του αμέσως ανώτερου επιπέδου και μόνον αυτού.

4. Συνεχίζουμε προς τα πάνω διαχωρίζοντας τους εσωτερικούς κόμβους ενός επιπέδου του δέντρου κάθε φορά, μέχρι να φτάσουμε στη ρίζα του δέντρου.

Στη συνέχεια, θα εκτελέσουμε για τα δεδομένα του προηγούμενου παραδείγματος τον βελτιστοποιημένο αλγόριθμο εισαγωγής μίας εγγραφής δεδομένων σε ένα B+ δέντρο.

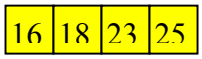
Ας δούμε λοιπόν, την εισαγωγή των σελίδων δεδομένων του τελευταίου παραδείγματος με βάση αυτόν το νέο αλγόριθμο. Θεωρούμε, ότι ο χώρος άφιξης των μετρήσεων των αισθητήρων είναι μεγέθους 4, οπότε μετά την άφιξη και της τελευταίας αυτός θα έχει την ακόλουθη μορφή:

Χώρος άφιξης νέων σελίδων δεδομένων



Αφού ο χώρος άφιξης έχει γεμίσει, η πρώτη μας δουλειά είναι η ταξινόμηση των σελίδων αυτού κατά αύξουσα σειρά από τα αριστερά προς τα δεξιά, οπότε θα έχουμε:

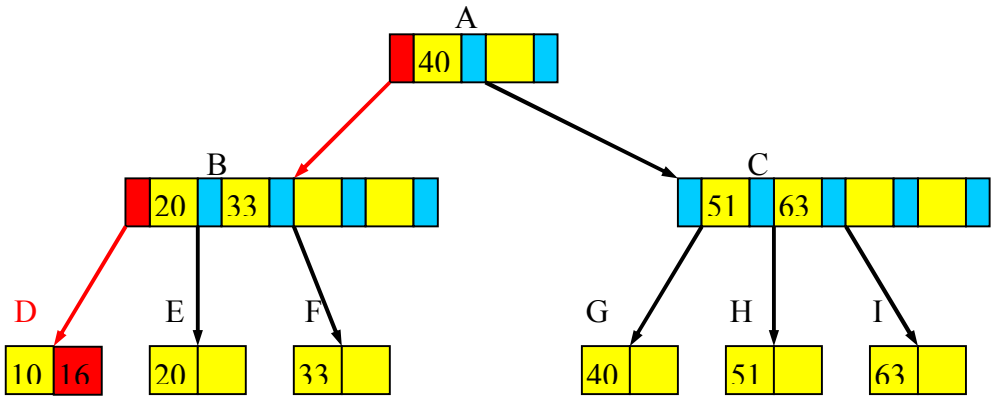
Ταξινομημένος χώρος άφιξης νέων σελίδων δεδομένων



Πρώτα θα εισάγουμε την εγγραφή με τιμή 16, με αποτέλεσμα την ενημέρωση της σελίδας δεδομένων D.

Σχήμα 3.12:Παράδειγμα εισαγωγής εγγραφής δεδομένων σε B+ δέντρο με το βελτιστοποιημένο αλγόριθμο

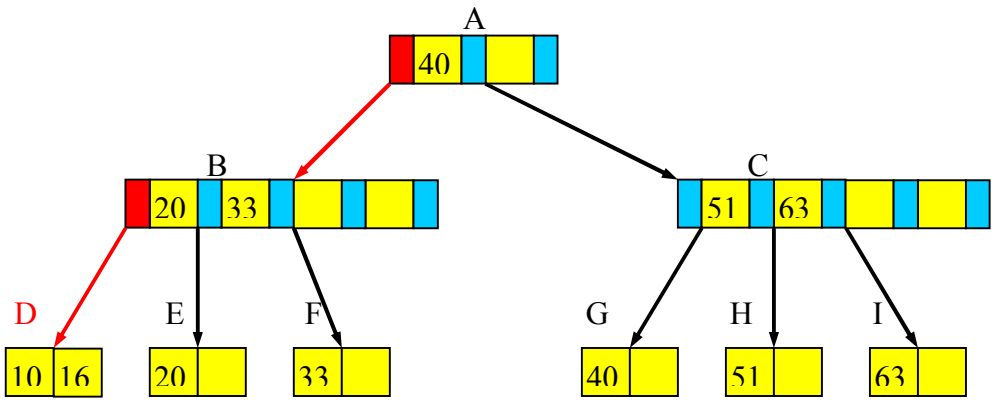
Προσθήκη εγγραφής με τιμή 16.



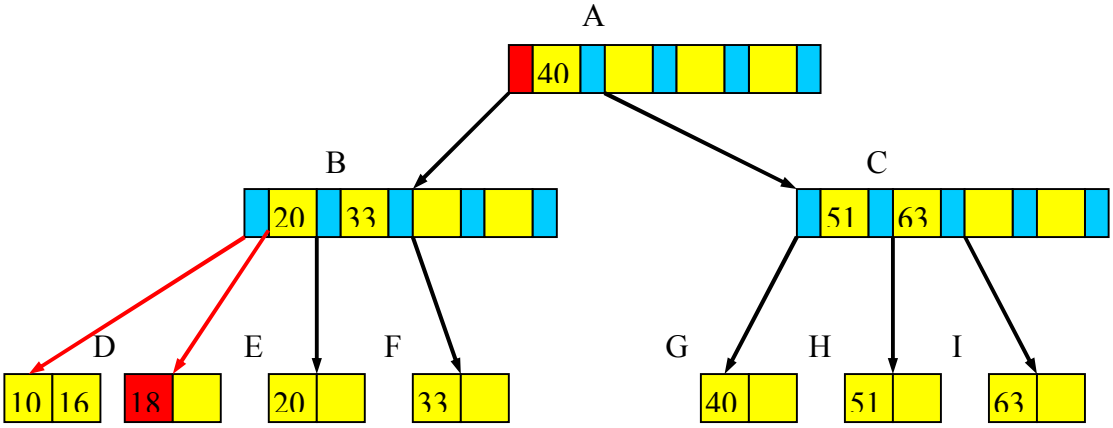
Στη συνέχεια, θα εισάγουμε την εγγραφή με τιμή 18, με αποτέλεσμα το διαχωρισμό και την ενημέρωση της σελίδας δεδομένων D.

Σχήμα 3.13:Παράδειγμα εισαγωγής εγγραφής δεδομένων σε B+ δέντρο με το βελτιστοποιημένο αλγόριθμο

Προσθήκη εγγραφής με τιμή 18.



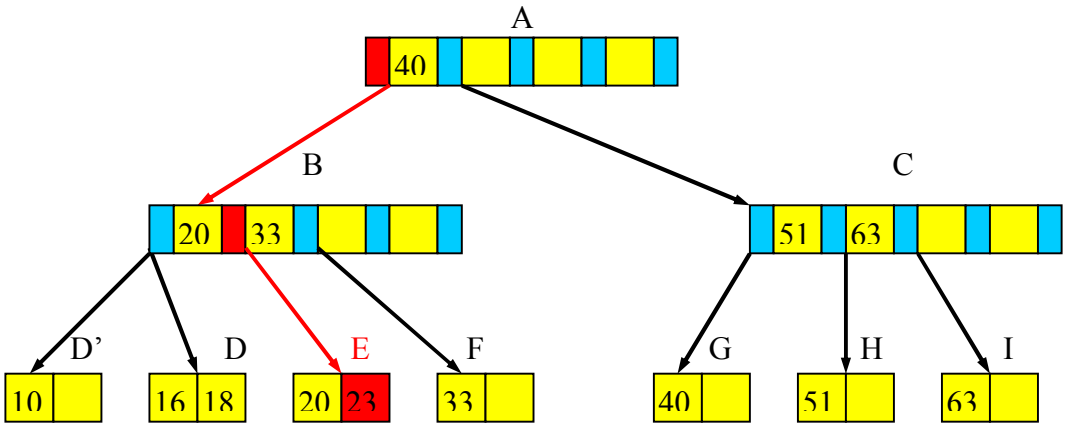
Διαχωρισμός της σελίδας δεδομένων D και ενημέρωση της τιμής αυτής.



Μετά, θα εισάγουμε την εγγραφή με τιμή 23, με αποτέλεσμα την ενημέρωση της σελίδας δεδομένων E.

Σχήμα 3.14:Παράδειγμα εισαγωγής εγγραφής δεδομένων σε B+ δέντρο με το βελτιστοποιημένο Αλγόριθμο

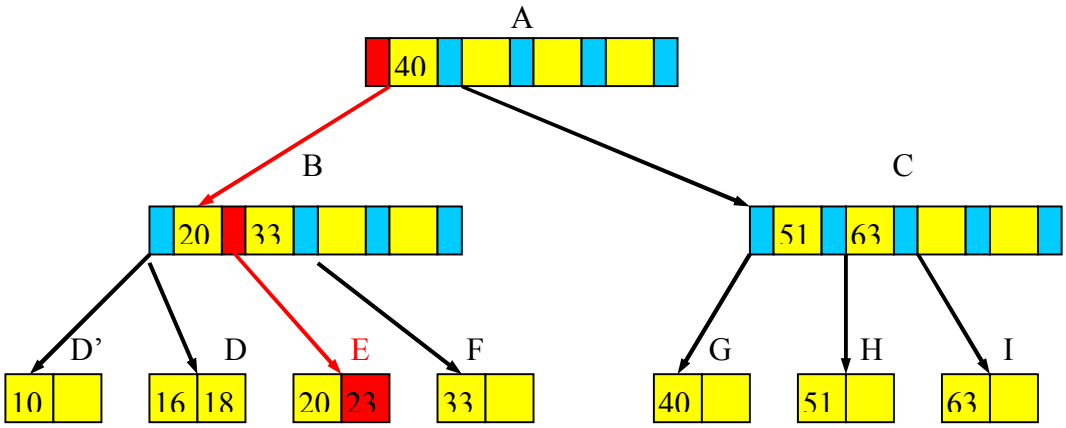
Προσθήκη εγγραφής με τιμή 23.



Στη συνέχεια, θα εισάγουμε την εγγραφή με τιμή 25, με αποτέλεσμα το διαχωρισμό και την ενημέρωση της σελίδας δεδομένων E.

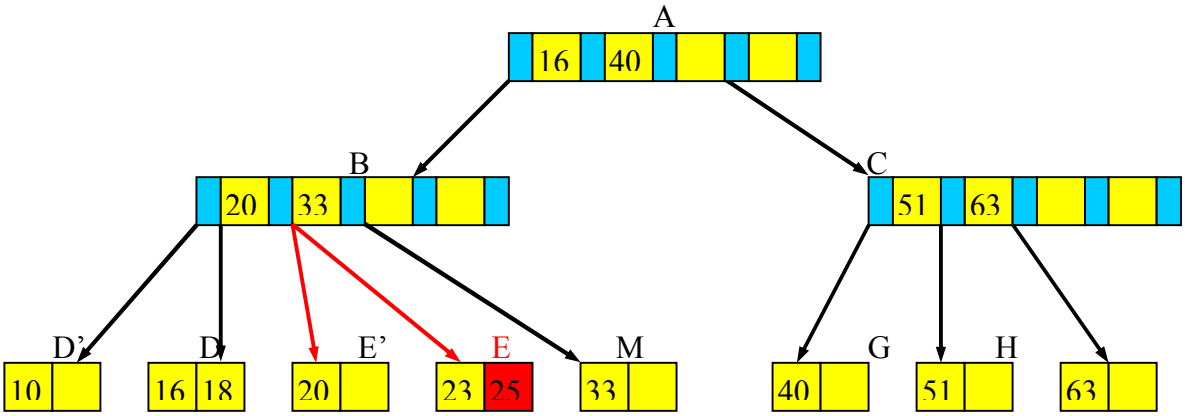
Σχήμα 3.15:Παράδειγμα εισαγωγής εγγραφής δεδομένων σε B+ δέντρο με το βελτιστοποιημένο αλγόριθμο

Προσθήκη εγγραφής με τιμή 25



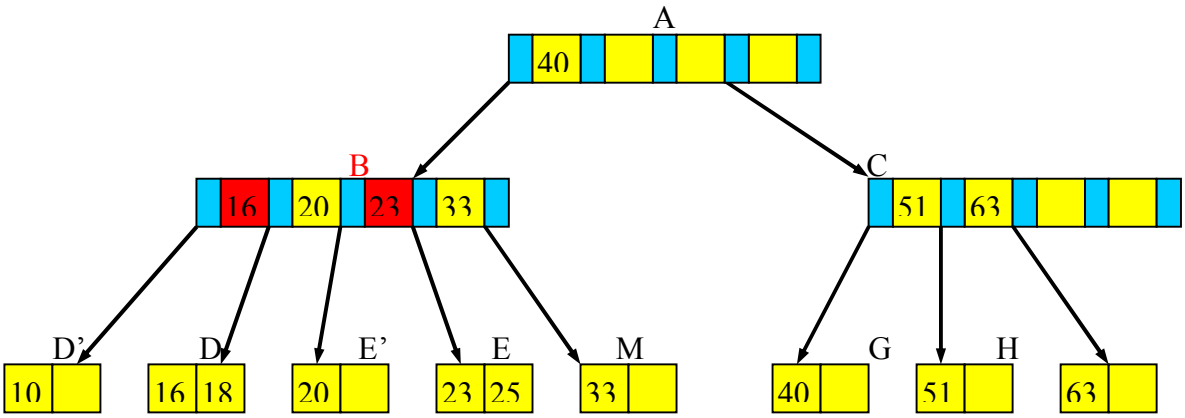
Σχήμα 3.16:Παράδειγμα εισαγωγής εγγραφής δεδομένων σε B+ δέντρο με το βελτιστοποιημένο αλγόριθμο

Διαχωρισμός της σελίδας δεδομένων E και ενημέρωση της τιμής αυτής.



Μετά κι αφού έχει ολοκληρωθεί η εισαγωγή όλων των σελίδων δεδομένων στο B+ δέντρο, θα διαχωρίσουμε τον εσωτερικό κόμβο B του αμέσως ανώτερου επιπέδου, με αποτέλεσμα την ενημέρωση της τιμής αυτού. Έτσι τελικά, θα έχουμε:

Σχήμα 3.17:Παράδειγμα εισαγωγής εγγραφής δεδομένων σε B+ δέντρο με το βελτιστοποιημένο αλγόριθμο



Όπως παρατηρούμε λοιπόν, με το βελτιστοποιημένο αλγόριθμο πετύχαμε για την εισαγωγή των μετρήσεων των αισθητήρων να απαιτείται η ενημέρωση διαδοχικά των σελίδων δεδομένων D, E και B, δηλαδή η ενημέρωση μόλις 3 σελίδων δεδομένων, σε αντίθεση με τον απλό αλγόριθμο, όπου χρειάστηκε να ενημερωθούν οι τιμές 8 σελίδων δεδομένων. Καθίσταται ολοφάνερο λοιπόν, ότι για την ευρετηρίαση των δεδομένων που βρίσκονται αποθηκευμένα σε έναν κόμβο ενός WSN εξυπηρετεί η χρήση του βελτιστοποιημένου αλγορίθμου εισαγωγής εγγραφών δεδομένων σε ένα B+ δέντρο.

3.3 Κατανεμημένο ευρετήριο

Έχει ήδη αναφερθεί, ότι μεταξύ όλων των head SNs σχηματίζεται ένα κατανεμημένο ευρετήριο των δεδομένων που βρίσκονται καταχωρημένα σε όλους τους head SNs. Επιπρόσθετα και μεταξύ όλων των head SNs μίας συστάδας δημιουργείται ένα κατανεμημένο ευρετήριο, στο οποίο και είναι καταχωρημένα τα δεδομένα όλων των SNs της συστάδας του. Στην ενότητα αυτή, θα εξετάσουμε την καταλληλότερη δομή δεδομένων για την υλοποίηση του κατανεμημένου ευρετηρίου. Πιο συγκεκριμένα, θα παρουσιάσουμε έναν αλγόριθμο δημιουργίας του κατανεμημένου ευρετηρίου κι έναν αλγόριθμο αναζήτησης μίας εγγραφής δεδομένων στο κατανεμημένο ευρετήριο, καθώς και μία βελτιστοποιημένη υλοποίηση αυτού. Πρώτα όμως, θα αποσαφηνίσουμε μερικές έννοιες, οι οποίες και είναι χρήσιμες για την κατανόηση του αλγορίθμου αυτού.

3.3.1 Λογικό άλμα και συνάρτηση hop( )

Προτού προχωρήσουμε στην παρουσίαση του αλγορίθμου δημιουργίας του κατανεμημένου ευρετηρίου, θα πρέπει να αναφέρουμε εν συντομία μερικά πράγματα για την έννοια του λογικού άλματος και τη συνάρτηση hop (number, number2). Έτσι λοιπόν, ένα λογικό άλμα ισοδυναμεί με τη μετάβαση από έναν head SN στον από δεξιά γειτονικό head SN αυτού, ενώ , η συνάρτηση hop(number, number2) παρέχει στην έξοδό της ένα σύνολο head SNs, τους οποίους και μπορούμε να επισκεφτούμε από τον τρέχοντα SN εκτελώντας από number1 μέχρι number2 άλματα. Πιο συγκεκριμένα, οι είσοδοι της συνάρτησης hop( ) είναι οι ακόλουθες:

- ένας κόμβος του δακτυλίου των head SNs, δηλαδή ένας head SN
- ένας αριθμός number1, ο οποίος κι αποτελεί το αριστερό όριο του διαστήματος [number1, number2]
- ένας αριθμός number2, ο οποίος κι αποτελεί το δεξιό όριο του διαστήματος [number1, number2] . Η μέγιστη τιμή του number2 θα πρέπει να ισούται με το πλήθος των κόμβων του δακτυλίου.

Η δε έξοδος της συνάρτησης hop( ) είναι η κάτωθι:

- ένα πλήθος κόμβων του δακτυλίου, δηλαδή ένα υποσύνολο του συνόλου όλων των head SNs, καθώς και η μεγαλύτερη τιμή εγγραφής που βρίσκεται στους κόμβους αυτούς.

Η λειτουργία της συνάρτησης hop( ), έχει ως εξής:

- Ξεκινάει από τον τρέχοντα κόμβο του δακτυλίου κι εκτελεί ένα πλήθος λογικών αλμάτων ίσο με number1, δηλαδή εκτελεί number1 λογικά άλματα προς τα δεξιά μεταβαίνοντας με τον τρόπο αυτό σε έναν άλλο κόμβο του δακτυλίου, δηλαδή σε έναν άλλο head SN, τον οποίο και παρέχει στην έξοδό της ως αποτέλεσμα.

Στη συνέχεια συγκρίνει τους αριθμούς number1 και number2 κι εάν αυτοί είναι ίσοι μεταξύ τους τερματίζει. Ειδάλλως, αυξάνει κατά μία μονάδα τον number1, δηλαδή θέτει τον number1 ίσο με (number1)+1 κι επαναλαμβάνει για τη νέα αυτή τιμή του number1, το προηγούμενο βήμα 1.

3.3.2 Αλγόριθμος δημιουργίας του κατανεμημένου ευρετηρίου ενός WSN

1. Αριθμούνται οι επικεφαλείς κόμβοι αισθητήρες (head SNs) του WSN σε αύξουσα σειρά.

2. Οι αριθμημένοι πλέον head SNs σχηματίζουν έναν εικονικό δακτύλιο, στον οποίο και καθένας από αυτούς συνδέεται από αριστερά του με τον αμέσως μικρότερο από αυτόν head SN κι από δεξιά του με τον αμέσως μεγαλύτερο από αυτόν head SN. Εξαίρεση αποτελεί μόνο ο μεγαλύτερος head SN, ο οποίος και συνδέεται από δεξιά του με τον μικρότερο head SN. Αξίζει να αναφέρουμε, ότι στην πράξη οι γειτονικοί κατά μήκους του δακτυλίου head SNs μπορεί είτε να είναι απευθείας συνδεδεμένοι μεταξύ τους είτε να συνδέονται μέσω άλλων SNs.

3. Βρίσκουμε το πεδίο τιμών των μετρήσεων των αισθητήρων που βρίσκονται ενσωματωμένοι στους SNs αυτού του WSN και το χωρίζουμε σε τόσα υποδιαστήματα, όσοι είναι και οι head SNs. Τα υποδιαστήματα αυτά θα πρέπει να έχουν όσο το δυνατόν παρόμοιο μέγεθος μεταξύ τους. Εν συνεχεία, αναθέτουμε καθένα από τα υποδιαστήματα αυτά στους head SNs με αύξουσα σειρά.

4. Δημιουργούμε ένα ευρετήριο για καθέναν από τους head SNs. Το ευρετήριο αυτό είναι στην ουσία ένας πίνακας 2 στηλών και 3 γραμμών.

   - Η πρώτη στήλη αυτού ονομάζεται distInt και προσδιορίζει την απόσταση των head SNs στους οποίους μπορούμε να μεταβούμε από τον τρέχοντα head SN μετρημένη σε λογικά άλματα. Κάθε στοιχείο της στήλης αυτής είναι της μορφής hop(number1,number2) και σημαίνει ότι γίνονται από number1 μέχρι number2 πλήθος αλμάτων από τον τρέχοντα head SN.

   - Η δεύτερη στήλη ονομάζεται maxKey και αντιστοιχεί στην μέγιστη τιμή εγγραφής που μπορεί να βρεθεί από τον τρέχοντα head SN, αν πραγματοποιηθούν τόσα άλματα όσα ορίζονται από την πρώτη στήλη.

Έτσι λοιπόν, οι γραμμές του ευρετηρίου-πίνακα κάθε head SN, θα έχουν ως ακολούθως:

- Η πρώτη γραμμή του πίνακα θα μας δείχνει τη μέγιστη τιμή εγγραφής που μπορεί να βρει ο τρέχων head SN, αν δεν πραγματοποιήσει κανένα άλμα (hop(0,0)), δηλαδή τη μέγιστη τιμή εγγραφής αυτού, δηλαδή το δεξιό όριο του υποδιαστήματος τιμών που έχει αντιστοιχηθεί σε αυτόν.

- Κάθε επόμενη γραμμή του πίνακα μας δείχνει τη μέγιστη τιμή εγγραφής που μπορεί να βρεθεί από τον τρέχοντα head SN, αν πραγματοποιηθούν τόσα άλματα όσα ορίζονται από τη συνάρτηση hop(), δηλαδή τη μέγιστη τιμή εγγραφής που βρίσκεται αποθηκευμένη σε κάποιον από τους head SNs, που αποτελούν έξοδο της συνάρτησης hop( ) της πρώτης στήλης.

3.3.3 Απλός αλγόριθμος αναζήτησης μίας εγγραφής δεδομένων στο κατανεμημένο ευρετήριο

1. Μεταβαίνουμε στον πίνακα-ευρετήριο του head SN, στον οποίο βρισκόμαστε δηλαδή του head SN, ο οποίος κι έχει λάβει το αίτημα εύρεσης της εγγραφής.

2. Ξεκινάμε από την πρώτη γραμμή του ευρετηρίου και συγκρίνουμε την τιμή της υπό αναζήτηση εγγραφής με το maxKey αυτής.

- Αν η εγγραφή, την οποία κι αναζητούμε είναι μικρότερη ή ίση του maxKey της πρώτης γραμμής, τότε γνωρίζουμε ότι η εγγραφή αυτή θα πρέπει να βρίσκεται καταχωρημένη στον τρέχοντα κόμβο, δηλαδή στον κόμβο όπου βρισκόμαστε. Έτσι λοιπόν, αναζητούμε την εν λόγω εγγραφή στον κόμβο αυτό κι αν πράγματι υπάρχει την επιστρέφουμε ως αποτέλεσμα, ενώ διαφορετικά εξετάζουμε την επόμενη γραμμή του ευρετηρίου.

- Αν η εγγραφή, την οποία αναζητούμε είναι μεγαλύτερη του maxKey της πρώτης γραμμής, τότε γνωρίζουμε ότι η εγγραφή αυτή δεν ανήκει στον κόμβο αυτό και προχωράμε κατευθείαν στην εξέταση της επόμενης γραμμής του ευρετηρίου.

3. Εξετάζουμε την τρέχουσα γραμμή του ευρετηρίου συγκρίνοντας κάθε φορά την τιμή της υπό εισαγωγή εγγραφής με το maxKey αυτής της γραμμής.

    - Αν η τιμή της υπό εισαγωγή εγγραφής είναι μικρότερη του maxKey αυτής της γραμμής, τότε καλούμε για τον τρέχοντα κόμβο, στον οποίο βρισκόμαστε την εκτέλεση της συνάρτησης hop( ), που βρίσκεται στην πρώτη στήλη αυτής της γραμμής. Ως αποτέλεσμα της εκτέλεσης της συνάρτησης hop( ), θα λάβουμε ένα πλήθος head SNs, σε καθένα από τους οποίους κι ελέγχουμε αν υπάρχει η υπό αναζήτηση εγγραφή. Αν ναι, τότε την επιστρέφουμε ως αποτέλεσμα, ενώ διαφορετικά εξετάζουμε την επόμενη γραμμή του ευρετηρίου.

    - Αν η εγγραφή, την οποία αναζητούμε είναι μεγαλύτερη του maxKey της τρέχουσας γραμμής, τότε γνωρίζουμε ότι η εγγραφή αυτή δεν ανήκει στον κόμβο αυτό και προχωράμε κατευθείαν στην εξέταση της επόμενης γραμμής του ευρετηρίου.

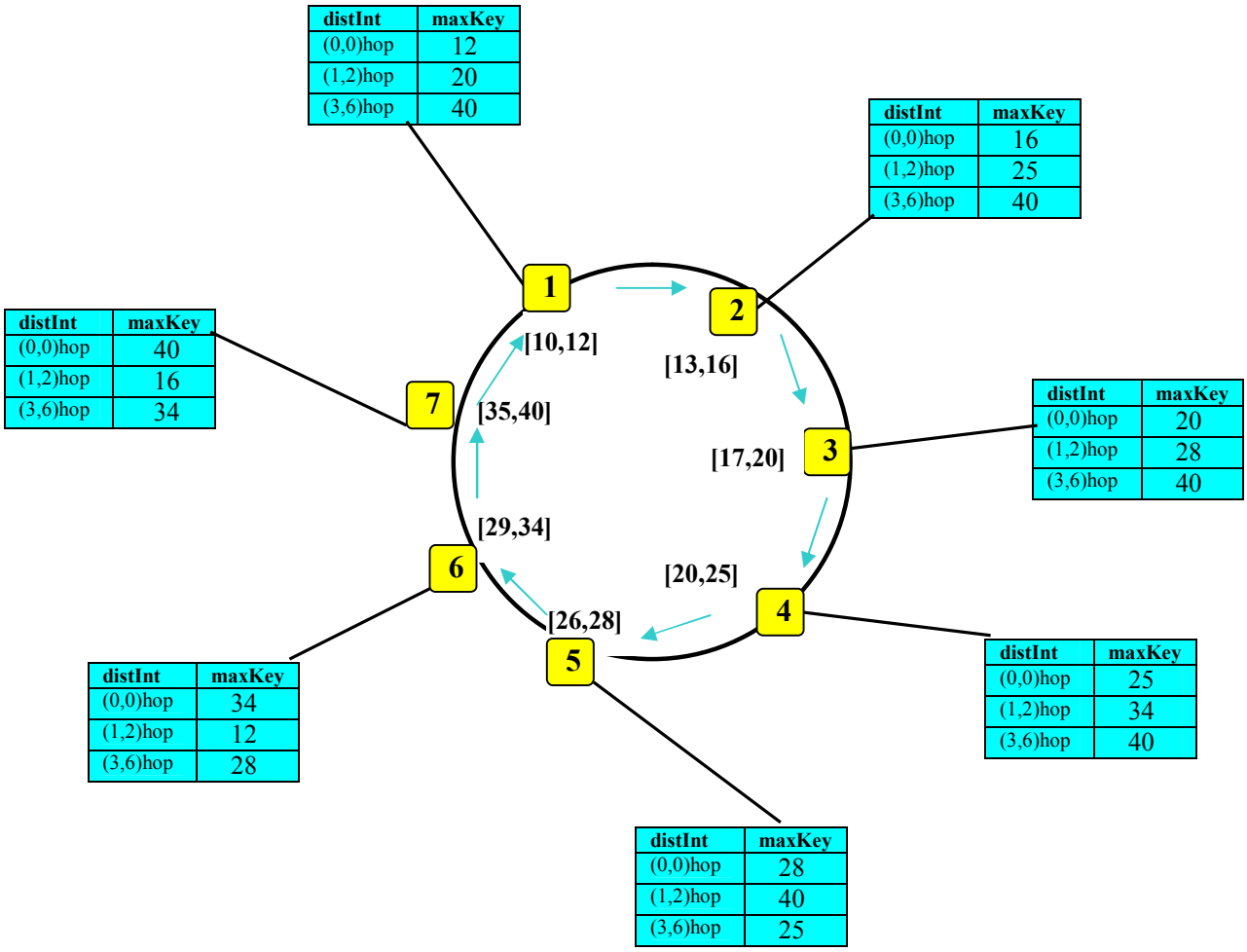
4. Επαναλαμβάνουμε το βήμα 3 και για καθεμία από τις υπόλοιπες γραμμές του πίνακα-ευρετήριο του head SN, στον οποίο βρισκόμαστε.

3.3.4 Παράδειγμα 3.3 - Δημιουργία του κατανεμημένου ευρετηρίου κι αναζήτηση της τιμής μίας εγγραφής στο κατανεμημένο ευρετήριο με τον απλό αλγόριθμο αναζήτησης

Ας θεωρήσουμε ένα WSN, το οποίο και διαθέτει στις τάξεις του 7 επικεφαλείς κόμβους αισθητήρες (head SNs), ενώ οι τιμές των μετρήσεων που γίνονται από τους αισθητήρες των SNs αυτού του WSN ανήκουν στο διάστημα [10, 40]. Πρώτα,

αριθμούμε τους head SNs από το 1 μέχρι το 7 και στη συνέχεια τους εισάγουμε σε έναν εικονικό δακτύλιο. Στη συνέχεια, χωρίζουμε το πεδίο τιμών των μετρήσεων των αισθητήρων, δηλαδή το διάστημα [10, 40] στα ακόλουθα υποδιαστήματα [10, 12], [13, 16], [17, 20], [20, 25], [26, 28], [29, 34], [35, 40] κι αναθέτουμε με τη σειρά καθένα από τα υποδιαστήματα αυτά στους head SNs του WSN. Στη συνέχεια δημιουργούμε για καθέναν από τους head SNs, το αντίστοιχο ευρετήριο-πίνακα αυτού. Έτσι προκύπτει τελικά, ο εικονικός δακτύλιος που φαίνεται στο παρακάτω σχήμα:

Σχήμα 3.18: Εικονικός δακτύλιος του WSN του παραδείγματος 3.3



- Αναζήτηση εγγραφής με τιμή 15 από τον κόμβο 1.

Έστω τώρα, ότι σε κάποια χρονική στιγμή αφικνύεται στον κόμβο 1, ένα αίτημα αναζήτησης της εγγραφής με τιμή 15. Η πρώτη μας κίνηση είναι να μεταβούμε στον πίνακα-ευρετήριο του head SN 1, στον οποίο βρισκόμαστε. Η υπό αναζήτηση εγγραφή 15 είναι μεγαλύτερη του maxKey της πρώτης γραμμής του ευρετηρίου, όποτε μεταβαίνουμε στη δεύτερη γραμμή αυτού. Εκεί διαπιστώνουμε, ότι η υπό αναζήτηση εγγραφή με τιμή 15 είναι μικρότερη του maxKey αυτής. Συνεπώς, θα εκτελέσουμε για τον head SN1, τη συνάρτηση hop( ) της δεύτερης γραμμής του ευρετηρίου αυτού. Ως αποτέλεσμα, θα λάβουμε τους head SNs 2 και 3. Στέλνουμε σε

καθέναν από αυτούς το αίτημα για την εγγραφή με τιμή 15 και διαπιστώνουμε, ότι αυτή βρίσκεται στον head SN2. Δηλαδή, όπως παρατηρούμε για την απάντηση του αιτήματος χρειάστηκε να σταλεί το αίτημα 3 φορές στο δίκτυο, μία αρχική στον SN1, μία από τον SN1 στον SN2 και μία στον SN1 από τον SN3. Αν δεν χρησιμοποιούσαμε το κατανεμημένο ευρετήριο, τότε το αίτημα θα έπρεπε να σταλεί και στους 7 SNs, δηλαδή θα έπρεπε να σταλεί συνολικά 7 φορές. Είναι προφανής λοιπόν, η αποσυμφόρηση του δικτύου που επιτυγχάνουμε με τη χρήση του κατανεμημένου ευρετηρίου.

- Αναζήτηση εγγραφής με τιμή 12 από τον κόμβο 2.

Έστω τώρα, ότι σε κάποια άλλη χρονική στιγμή ο κόμβος 2 παραλαμβάνει ένα αίτημα για την τιμή 37. Η πρώτη μας κίνηση είναι να μεταβούμε στον πίνακα-ευρετήριο του head SN 1, στον οποίο βρισκόμαστε. Η υπό αναζήτηση εγγραφή 12 είναι μικρότερη του maxKey της πρώτης γραμμής του ευρετηρίου, όποτε την αναζητούμε στις εγγραφές αυτού. Όμως, όπως παρατηρούμε η εγγραφή με τιμή 12 δεν συμπεριλαμβάνεται στις εγγραφές του τρέχοντα κόμβου 2, οπότε μεταβαίνουμε στη δεύτερη γραμμή του ευρετηρίου. Εκεί διαπιστώνουμε, ότι η υπό αναζήτηση εγγραφή με τιμή 15 είναι μικρότερη του maxKey αυτής. Συνεπώς, θα εκτελέσουμε για τον head SN2, τη συνάρτηση hop( ) της δεύτερης γραμμής του ευρετηρίου αυτού. Ως αποτέλεσμα, θα λάβουμε τους head SNs 3 και 4. Στέλνουμε σε καθέναν από αυτούς το αίτημα για την εγγραφή με τιμή 12 και διαπιστώνουμε, ότι αυτή δεν βρίσκεται αποθηκευμένη σε κανένα από τους δυο τους κι έτσι μεταβαίνουμε στην επόμενη γραμμή του ευρετηρίου. Εκεί διαπιστώνουμε, ότι η υπό αναζήτηση εγγραφή με τιμή 15 είναι μικρότερη του maxKey αυτής. Συνεπώς, θα εκτελέσουμε για τον head SN2, τη συνάρτηση hop( ) της τρίτης γραμμής του ευρετηρίου αυτού. Ως αποτέλεσμα, θα λάβουμε τους head SNs 5, 6, 7 και 1. Στέλνουμε σε καθέναν από αυτούς το αίτημα για την εγγραφή με τιμή 15 και διαπιστώνουμε, ότι τελικά αυτή είναι αποθηκευμένη στον head SN1.

Δηλαδή, όπως παρατηρούμε για την απάντηση του αιτήματος χρειάστηκε να σταλεί το αίτημα 7 φορές στο δίκτυο, οι οποίες και είναι οι ακόλουθες: μία αρχική στον SN2, SN2→SN3, SN2→SN4, SN2→SN5, SN2→SN6, SN2→SN7, SN2→SN1. Αυτή είναι μία ακραία περίπτωση, κατά την οποία το πλήθος των αποστολών του αιτήματος εντός του δικτύου είναι ίδιο είτε χρησιμοποιήσουμε το κατανεμημένο ευρετήριο είτε όχι. Λαμβάνοντας υπόψιν μας, ότι δεν είναι και τόσο πιθανό να συμβεί κάτι τέτοιο, εξακολουθούμε να δεχόμαστε την αποδοτικότητα στη χρήση του κατανεμημένου ευρετηρίου. Εντούτοις και λόγω της επιθυμίας μας, να επιτύχουμε ακόμη καλύτερα αποτελέσματα, θα προτείνουμε στη συνέχεια μία ακόμη βελτιστοποιημένη υλοποίηση του κατανεμημένου ευρετηρίου.

3.3.5 Διαφορές μεταξύ απλού και βελτιστοποιημένου αλγορίθμου αναζήτησης μίας εγγραφής δεδομένων στο κατανεμημένο ευρετήριο

Προτού προχωρήσουμε στην παρουσίαση του νέο βελτιστοποιημένου αλγορίθμου, αξίζει να αναφέρουμε τις κυριότερες διαφορές αυτού με τον απλό αλγόριθμο, οι οποίες και είναι οι ακόλουθες:

- Πέρα από τη συνάρτηση hop( ), χρησιμοποιείται και μία παρόμοια με αυτήν συνάρτηση alm( ). Μοναδική διαφορά μεταξύ τους είναι το γεγονός, ότι ενώ στη συνάρτηση hop( ), τα λογικά άλματα από κάθε κόμβο του δακτυλίου

πραγματοποιούνται προς τα δεξιά, στη συνάρτηση alm( ), τα λογικά άλματα γίνονται προς τα αριστερά, καθώς επίσης και το γεγονός ότι ενώ η hop( ) δίνει στην έξοδό της ένα σύνολο κόμβων και τη μεγαλύτερη τιμή εγγραφής σε αυτούς, η alm( ) δίνει στην έξοδό της επίσης ένα σύνολο κόμβων με τη μικρότερη όμως τιμή εγγραφής σε αυτούς.

- Ο δακτύλιος δεν είναι ένας όπως πριν, αλλά δύο. Ο ένας δακτύλιος είναι ο ίδιος δεξιόστροφος δακτύλιος με τον απλό αλγόριθμο, ενώ ο άλλος είναι ένας ίδιος ακριβώς αλλά αριστερόστροφος δακτύλιος. Οι μεταβάσεις στο δεξιόστροφο δακτύλιο γίνονται μέσω της συνάρτησης hop( ), ενώ οι μεταβάσεις στον αριστερόστροφο δακτύλιο γίνονται μέσω της συνάρτησης alm( ).

- Το ευρετήριο κάθε SN επεκτείνεται και παράλληλα χωρίζεται σε δύο νέα τμήματα Α και Β. Το τμήμα Α είναι το ίδιο ακριβώς με το ευρετήριο κάθε κόμβου στον απλό αλγόριθμο με τη διαφορά ότι λείπει η πρώτη γραμμή αυτού, ενώ το τμήμα Β είναι παρόμοιο με το τμήμα Α, με τη διαφορά ότι αντί της συνάρτησης hop( ), χρησιμοποιούμε σε αυτό τη συνάρτηση alm( ).

3.3.6 Βελτιστοποιημένος αλγόριθμος αναζήτησης μίας εγγραφής δεδομένων στο κατανεμημένο ευρετήριο

1. Μεταβαίνουμε στον πίνακα-ευρετήριο του head SN, στον οποίο βρισκόμαστε, δηλαδή του head SN, ο οποίος κι έχει λάβει το αίτημα εύρεσης της εγγραφής.

2. Αν η υπό αναζήτηση εγγραφή έχει τιμή μικρότερη ή ίση του maxKey της πρώτης γραμμής και μεγαλύτερη ή ίση του minKey της πρώτης γραμμής, τότε εγγραφή αυτή βρίσκεται καταχωρημένη στον τρέχοντα κόμβο, δηλαδή στον κόμβο όπου βρισκόμαστε. Έτσι λοιπόν, αναζητούμε την εν λόγω εγγραφή στον κόμβο αυτό κι αν πράγματι υπάρχει την επιστρέφουμε ως αποτέλεσμα, ενώ διαφορετικά τερματίζουμε τον αλγόριθμο χωρίς αποτέλεσμα.

3. Αν η εγγραφή, την οποία αναζητούμε είναι μεγαλύτερη του maxKey της πρώτης γραμμής του τμήματος Α, οπότε προφανώς και θα είναι μεγαλύτερη του minKey του τμήματος Β, τότε μεταβαίνουμε στο τμήμα Α του ευρετηρίου του τρέχοντος κόμβου και

3.1 Εξετάζουμε δεύτερη γραμμή του τμήματος Α συγκρίνοντας την τιμή της υπό εισαγωγή εγγραφής με το maxKey αυτής της γραμμής.

- Αν η τιμή της υπό εισαγωγή εγγραφής είναι μικρότερη του maxKey αυτής της γραμμής, τότε καλούμε για τον τρέχοντα κόμβο, στον οποίο βρισκόμαστε την εκτέλεση της συνάρτησης hop( ), που βρίσκεται στην πρώτη στήλη αυτής της γραμμής του τμήματος Α. Ως αποτέλεσμα της εκτέλεσης της συνάρτησης hop( ), θα λάβουμε ένα πλήθος head SNs, σε καθένα από τους οποίους κι ελέγχουμε αν υπάρχει η υπό αναζήτηση εγγραφή. Αν ναι, τότε την επιστρέφουμε ως αποτέλεσμα, ενώ διαφορετικά εξετάζουμε την επόμενη γραμμή του ευρετηρίου.

- Αν η εγγραφή, την οποία αναζητούμε είναι μεγαλύτερη του maxKey της τρέχουσας γραμμής, τότε γνωρίζουμε ότι η εγγραφή αυτή δεν ανήκει στον κόμβο αυτό και προχωράμε κατευθείαν στην εξέταση της επόμενης γραμμής του ευρετηρίου.

3.2 Επαναλαμβάνουμε το βήμα 3.1 και για καθεμία από τις υπόλοιπες γραμμές του πίνακα-ευρετήριο του head SN, στον οποίο βρισκόμαστε.

4. Αν η εγγραφή, την οποία αναζητούμε είναι μικρότερη του minKey της πρώτης γραμμής του τμήματος Β, οπότε προφανώς και θα είναι μικρότερη και του maxKey του τμήματος Α, τότε μεταβαίνουμε στο τμήμα Β του ευρετηρίου του τρέχοντος κόμβου και

4.1 Εξετάζουμε τη δεύτερη γραμμή του τμήματος Β συγκρίνοντας την τιμή της υπό εισαγωγή εγγραφής με το maxKey αυτής της γραμμής.

- Αν η τιμή της υπό εισαγωγή εγγραφής είναι μεγαλύτερη του minKey αυτής της γραμμής, τότε καλούμε για τον τρέχοντα κόμβο, στον οποίο βρισκόμαστε την εκτέλεση της συνάρτησης alm( ), που βρίσκεται στην πρώτη στήλη αυτής της γραμμής του τμήματος Β. Ως αποτέλεσμα της εκτέλεσης της συνάρτησης alm( ), θα λάβουμε ένα πλήθος head SNs, σε καθένα από τους οποίους κι ελέγχουμε αν υπάρχει η υπό αναζήτηση εγγραφή. Αν ναι, τότε την επιστρέφουμε ως αποτέλεσμα, ενώ διαφορετικά εξετάζουμε την επόμενη γραμμή του ευρετηρίου.

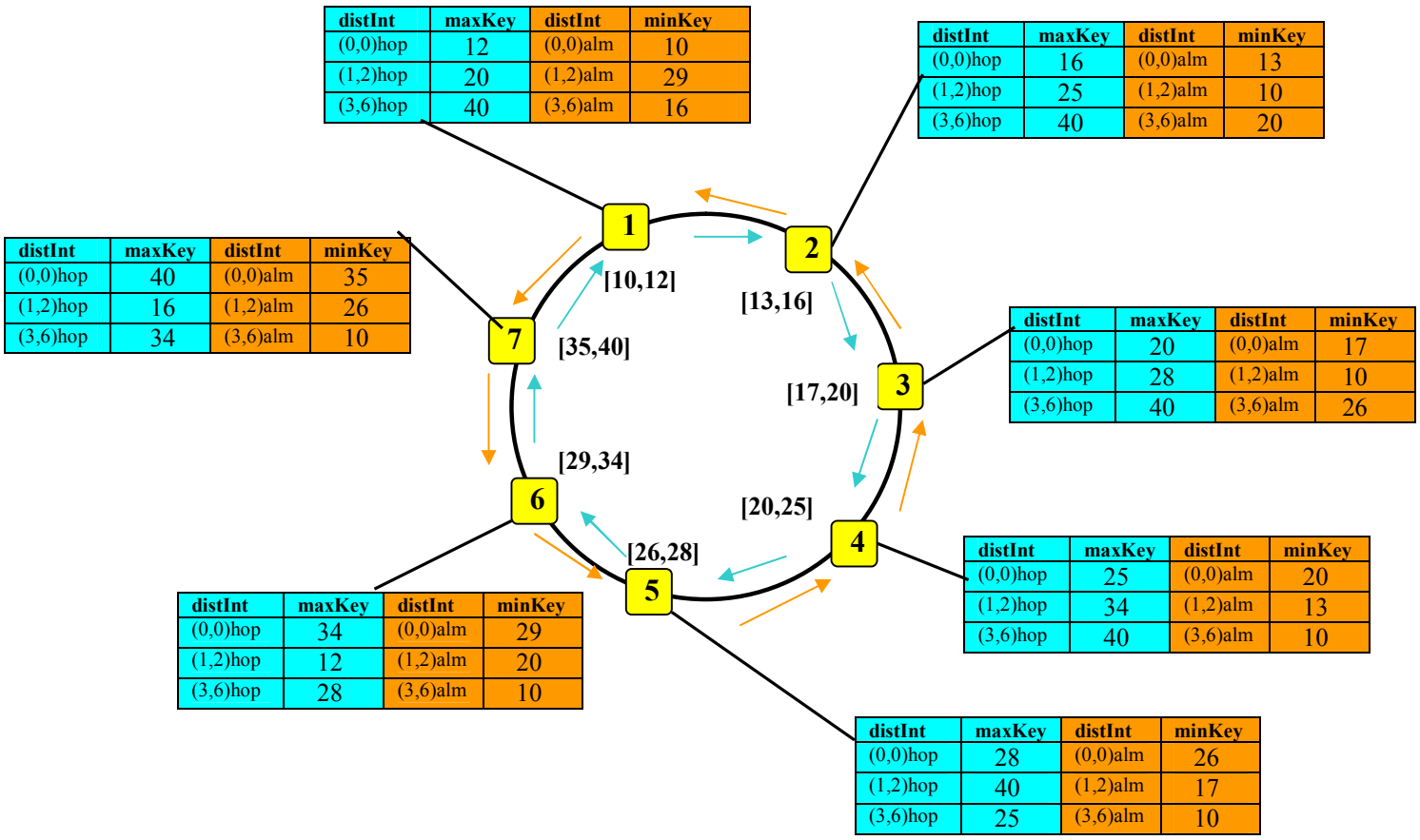
- Αν η εγγραφή, την οποία αναζητούμε είναι μικρότερη του minKey της τρέχουσας γραμμής, τότε γνωρίζουμε ότι η εγγραφή αυτή δεν ανήκει στον κόμβο αυτό και προχωράμε κατευθείαν στην εξέταση της επόμενης γραμμής του ευρετηρίου.

4.2 Επαναλαμβάνουμε το βήμα 4.1 και για καθεμία από τις υπόλοιπες γραμμές του πίνακα-ευρετήριο του head SN, στον οποίο βρισκόμαστε.

3.3.7 Παράδειγμα 3.4 - Βελτιστοποιημένος αλγόριθμος αναζήτησης μίας εγγραφής δεδομένων στο κατανεμημένο ευρετήριο

Πρώτα δημιουργούμε τον πίνακα-ευρετήριο κάθε SN. Πιο συγκεκριμένα, με βάση τη συνάρτηση alm( ), δημιουργούμε το δεξιό υποπίνακα - ευρετήριο Β και με βάση τη συνάρτηση hop( ), δημιουργούμε τον αριστερό υποπίνακα – ευρετήριο Α. Έτσι λοιπόν, το κατανεμημένο ευρετήριο, θα έχει πλέον τη μορφή του παρακάτω σχήματος. Η διαφορά μεταξύ αυτού του εικονικού δακτυλίου σε σύγκριση με τον αντίστοιχο εικονικό δακτύλιο του παραδείγματος 3.3 έγκειται στο γεγονός, ότι η αποστολή μηνυμάτων σε αυτόν τον εικονικό δακτύλιο, κατά την αναζήτηση κάποιας εγγραφής δεδομένων μπορεί να γίνει και προς τις δύο κατευθύνσεις, δηλαδή τόσο προς αριστερά όσο και προς τα δεξιά. Δηλαδή, είναι σαν να έχουμε δύο εικονικούς δακτυλίους, έναν αριστερόστροφο κι έναν δεξιόστροφο.

Σχήμα 3.19: Εικονικός δακτύλιος του WSN του παραδείγματος 3.4



- Αναζήτηση εγγραφής με τιμή 11 από τον κόμβο 1.

Η πρώτη μας κίνηση είναι να μεταβούμε στον πίνακα-ευρετήριο του head SN 1, στον οποίο βρισκόμαστε. Εκεί διαπιστώνουμε, ότι η υπό αναζήτηση εγγραφή 11 βρίσκεται μεταξύ του maxKey και του minKey της πρώτης γραμμής, οπότε το αίτημα απαντάται απευθείας από τον κόμβο αυτό, χωρίς να χρειαστεί περαιτέρω επαναποστολή του.

- Αναζήτηση εγγραφής με τιμή 15 από τον κόμβο 1.

Ισχύει, ότι ακριβώς και με την αναζήτηση της εγγραφής με τιμή 15 στον κόμβο 1, με τον απλό αλγόριθμο.

- Αναζήτηση εγγραφής με τιμή 14 από τον κόμβο 2.

Πρώτα μεταβαίνουμε στον πίνακα ευρετήριο του κόμβου, στον οποίο βρισκόμαστε, δηλαδή στον πίνακα-ευρετήριο του κόμβου 2. Εκεί διαπιστώνουμε, ότι η τιμή της υπό αναζήτηση εγγραφής 12 είναι μικρότερη του minKey της πρώτης γραμμής του πίνακα-ευρετήριο B, όμως επίσης και του maxKey της πρώτης γραμμής του πίνακα-ευρετήριο A. Άρα, μεταβαίνουμε στον πίνακα-ευρετήριο B και για την ακρίβεια στη δεύτερη γραμμή αυτού, όπου και διαπιστώνουμε ότι η υπό αναζήτηση εγγραφή 12 είναι μικρότερη του minKey αυτής της γραμμής. Άρα, εκτελούμε τη συνάρτηση alm() αυτής της γραμμής για τον τρέχοντα κόμβο, στον οποίο βρισκόμαστε, δηλαδή για τον κόμβο 2. Ως αποτέλεσμα παίρνουμε τους SNs 1 και 7, στους οποίους κι αποστέλλουμε το αίτημα για την εγγραφή με τιμή 12. Τελικά, η

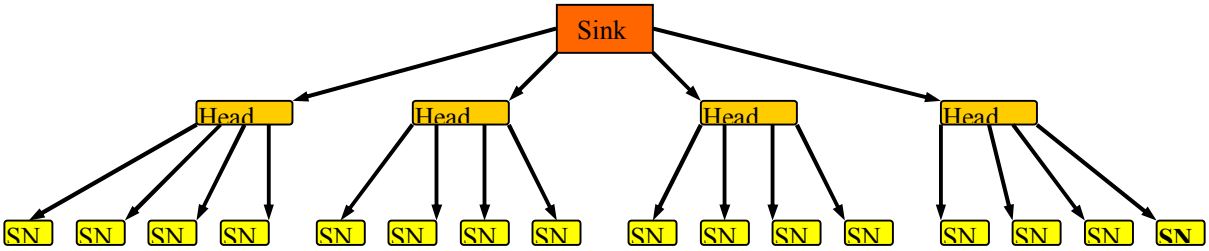
εγγραφή αυτή βρίσκεται στον SN1 από τον οποίο κι επιστρέφεται ως απάντηση στο εν λόγω αίτημα. Έτσι, παρατηρούμε, ότι για τη εύρεση της εν λόγω εγγραφής απαιτήθηκε να σταλεί το σχετικό αίτημα μόλις 3 φορές, σε αντίθεση με την αντίστοιχη περίπτωση με τον απλό αλγόριθμο, όπου το αίτημα έπρεπε να σταλεί συνολικά 7 φορές. Καθίσταται προφανής λοιπόν, η βελτίωση που επιφέρει η χρήση του βελτιστοποιημένου αλγόριθμου, σε σύγκριση τόσο με τον απλό αλγόριθμο δημιουργίας κατανεμημένου ευρετηρίου, όσο φυσικά και με την περίπτωση που στο WSN δεν χρησιμοποιείται κατανεμημένο ευρετήριο.

Μέσα από το παραπάνω παράδειγμα φαίνεται κι ένα ακόμη σημαντικό πλεονέκτημα του βελτιστοποιημένου αλγορίθμου αναζήτησης σε σχέση με τον απλό αλγόριθμο. Πιο συγκεκριμένα, αν στον απλό αλγόριθμο αναζήτησης διακοπεί κάποια ασύρματη ζεύξη μεταξύ δύο κόμβων του εικονικού δακτυλίου των SNs, τότε αυτόματα τίθεται εκτός λειτουργίας και το κατανεμημένο ευρετήριο. Αυτό οφείλεται στο γεγονός, ότι λόγω της ύπαρξης ενός και μοναδικού δακτυλίου θα υπάρχει μία και μοναδική διαδρομή επικοινωνίας, μεταξύ των SNs. Αντίθετα, εάν στο βελτιστοποιημένο αλγόριθμο διακοπεί μία ασύρματη ζεύξη μεταξύ δύο SNs του εικονικού δακτυλίου, τότε η επικοινωνία μεταξύ των SNs θα συνεχιστεί κανονικά, δηλαδή το κατανεμημένο ευρετήριο θα συνεχίσει να λειτουργεί κανονικά. Αυτό οφείλεται στο γεγονός, ότι υπάρχουν δύο δακτύλιοι με αποτέλεσμα να υπάρχουν περισσότερες από μία εναλλακτικές διαδρομές επικοινωνίας μεταξύ των SNs.

3.3.8 Σχήμα εξυπηρέτησης αιτημάτων σε ένα WSN, χωρίς κατανεμημένο ευρετήριο

Συμπερασματικά λοιπόν, το κατανεμημένο ευρετήριο των δεδομένων που προέρχονται από τις μετρήσεις των αισθητήρων των SNs ενός WSN και το οποίο εξετάσαμε ενδελεχώς στην παρούσα ενότητα αποτελεί μία ιδανική μέθοδο για την επίσπευση της πρόσβασης στα δεδομένα που βρίσκονται αποθηκευμένα σε ένα WSN. Πιο συγκεκριμένα, το σχήμα εξυπηρέτησης ενός αιτήματος από ένα WSN χωρίς τη χρήση κατανεμημένου ευρετηρίου, περιλαμβάνει αρχικά την παραλαβή του αιτήματος αυτού από τη Sink. Στη συνέχεια η Sink αποστέλλει το αίτημα αυτό συνολικά σε όλους τους head SNs του WSN κι αυτοί με τη σειρά τους σε όλους τους SNs της συστάδας τους. Όσοι από τους SNs διαθέτουν δεδομένα που αποτελούν απάντηση στο εν λόγω αίτημα τα επιστρέφουν στους head SNs της συστάδας τους, οι οποίοι με τη σειρά τους τα προωθούν στη Sink.

Σχήμα 3.20: Εξυπηρέτηση αιτημάτων σε WSN χωρίς κατανεμημένο ευρετήριο

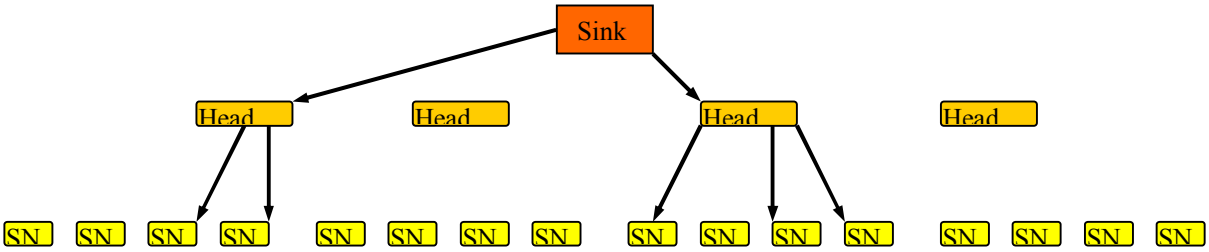


Αποστολή ενός αιτήματος σε ένα WSN, στο οποίο δεν χρησιμοποιείται κατανεμημένο ευρετήριο. Συνολικά θα λάβει χώρα η αποστολή του αιτήματος αυτού 20 φορές.

3.3.9 Εξυπηρέτηση αιτημάτων σε ένα WSN, με κατανεμημένο ευρετήριο

Από την άλλη, με το σχήμα εξυπηρέτησης ενός αιτήματος από ένα WSN με τη χρήση κατανεμημένου ευρετηρίου, η Sink μπορεί να αποστείλει το αίτημα αυτό με τη βοήθεια ενός κατανεμημένου ευρετηρίου των δεδομένων των head SNs, μόνο σε όσους head SNs διαθέτουν στις συστάδες τους SNs, στους οποίους και μπορεί να βρίσκονται αποθηκευμένα τα ζητούμενα δεδομένα. Έπειτα, οι head SNs αυτοί με τη βοήθεια και πάλι ενός κατανεμημένου ευρετηρίου των δεδομένων των SNs της συστάδας τους, εντοπίζουν τους SNs της συστάδας τους που μπορεί να διαθέτουν τα δεδομένα τα ζητούμενα δεδομένα, ούτως ώστε να αποστείλουν το αίτημα αυτό μόνο σε αυτούς τους συγκεκριμένους SNs της συστάδας τους. Εν συνεχεία, όσοι τελικά από τους SNs διαθέτουν αποθηκευμένα τα ζητούμενα δεδομένα, τα αποστέλλουν στους head SNs της συστάδας τους, οι οποίοι με τη σειρά τους τα προωθούν στη Sink, ούτως ώστε αυτή να τα επεξεργαστεί καταλλήλως και να απαντήσει στο εν λόγω αίτημα.

Σχήμα 3.21: Εξυπηρέτηση αιτημάτων σε WSN χωρίς κατανεμημένο ευρετήριο



Αποστολή ενός αιτήματος σε ένα WSN, στο οποίο χρησιμοποιείται κατανεμημένο ευρετήριο, τόσο μεταξύ των head SNs, όσο και μεταξύ των SNs της συστάδας κάθε head SN. Αν τα ζητούμενα δεδομένα του αιτήματος αυτού βρίσκονται στους SN3, SN4 του Head SN1 και στους SN1,SN3,SN4 του Head SN3, τότε θα λάβει χώρα η αποστολή του αιτήματος αυτού 7 φορές.

Έτσι λοιπόν, με τη χρήση του κατανεμημένου ευρετηρίου επιτυγχάνουμε την αποσυμφόρηση του δικτύου, την ελαχιστοποίηση της πιθανότητας λαθών κατά τη μετάδοση των δεδομένων και τη συνολική βελτίωση στην αποδοτικότητα και τη λειτουργία του WSN. Εντούτοις, θα πρέπει να αποσαφηνίσουμε, ότι παρά την επίσπευση της πρόσβασης στα δεδομένα των SNs, η επεξεργασία αυτών εξακολουθεί να λαμβάνει χώρα μόνο στην Κεντρική Βάση Συγκέντρωσης Δεδομένων (Sink) και όχι στους SNs.

Κεφάλαιο 4

Κατανεμημένη επεξεργασία και βελτιστοποίηση αιτημάτων

Περιεχόμενα

Σε αυτό το κεφάλαιο, θα ασχοληθούμε με τεχνικές βελτιστοποίησης και κατανεμημένης επεξεργασίας των αιτημάτων, που εξυπηρετούνται από ένα WSN. Πιο συγκεκριμένα, τα ζητήματα που θα εξετάσουμε εκτενέστερα είναι τα ακόλουθα:

- Επεξεργασία αιτημάτων
- Βελτιστοποίηση αιτημάτων
- Κρυφή μνήμη
- Δεδομενοκεντρική βελτιστοποίηση δρομολόγησης

4.1 Επεξεργασία αιτημάτων

Στη μέχρι στιγμής μελέτη μας, έχουμε θεωρήσει ότι η επεξεργασία των αιτημάτων των χρηστών μπορεί να λάβει χώρα στην Κεντρική Βάση Συγκέντρωσης Δεδομένων (Sink) και όχι στους απλούς κόμβους-αισθητήρες SNs. Είναι προφανές λοιπόν, ότι ο ρόλος των SNs περιορίζεται στη λήψη κι αποθήκευση των δεδομένων, καθώς και στην αποστολή τους στη Sink, όποτε αυτό χρειάζεται. Προκειμένου όμως να βελτιώσουμε την αποδοτικότητα και τη συνολική λειτουργία ενός WSN, μπορούμε να μεταφέρουμε τουλάχιστον ένα κομμάτι της επεξεργασίας των αιτημάτων στους SNs. Αυτό ακριβώς αποτελεί και το αντικείμενο της μελέτης μας στην παρούσα ενότητα, δηλαδή με ποιον τρόπο μπορεί να πραγματοποιηθεί η επεξεργασία των αιτημάτων που αποστέλλονται σε ένα WSN απευθείας από τους SNs αυτού ή για να το πούμε ακόμη καλύτερα, με ποιον τρόπο οι SNs ενός WSN μπορούν να συμμετέχουν στην επεξεργασία των αιτημάτων που παραλαμβάνονται από ένα WSN. Στη συνέχεια, θα μελετήσουμε τις δυνατότητες επεξεργασίας των αιτημάτων απευθείας από τους SNs ενός WSN, ξεχωριστά για τα ακριβή και για τα προσεγγιστικά αιτήματα.

4.2 Τεχνικές εξυπηρέτησης αιτημάτων σε ένα WSN

Υπάρχουν 3 τεχνικές εξυπηρέτησης αιτημάτων σε ένα WSN, τις οι οποίες και είναι οι ακόλουθες:

- Κεντρική Εξυπηρέτηση από τη Sink
- Τοπική Εξυπηρέτηση από τους SNs
- Εξυπηρέτηση σε δύο επίπεδα

4.3 Κεντρική εξυπηρέτηση αιτημάτων από τη Sink

Είναι η τεχνική εξυπηρέτησης αιτημάτων σε ένα WSN, την οποία και παρουσιάσαμε συνοπτικά στην εισαγωγή της παρούσας ενότητας και την οποία έχουμε χρησιμοποιήσει στη μέχρι στιγμής μελέτη μας. Σύμφωνα με αυτήν, η εξυπηρέτηση των αιτημάτων των χρηστών και η διαχείριση των δεδομένων των αισθητήρων σε ένα WSN λαμβάνει χώρα αποκλειστικά στη Sink. Πιο συγκεκριμένα, οι SNs περιορίζονται στη λήψη των δεδομένων από τις μετρήσεις των αισθητήρων, τα οποία κι αποστέλλουν ανά τακτά χρονικά διαστήματα στους head SNs της συστάδας, στην οποία ανήκουν. Εν συνεχεία, οι head SNs προωθούν με τη σειρά τους τα δεδομένα αυτά στη Sink. Έτσι λοιπόν ,όποτε η Sink παραλαμβάνει ένα καινούριο αίτημα, δεν το προωθεί στο δίκτυο, αλλά το εξυπηρετεί άμεσα με βάση τα

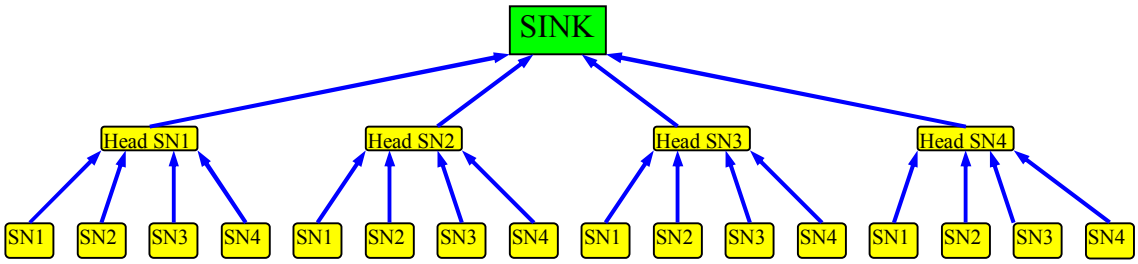
αποθηκευμένα σε αυτήν δεδομένα, τα οποία κι αντιστοιχούν στα πραγματικά δεδομένα των αισθητήρων.

Για παράδειγμα, ας θεωρήσουμε ένα WSN, στο οποίο η επεξεργασία των αιτημάτων των χρηστών λαμβάνει χώρα αποκλειστικά στη Sink. Όπως είπαμε, το γεγονός αυτό θα έχει ως αποτέλεσμα, οι SNs αυτού του WSN να αποστέλλουν τα δεδομένα τους ανά περιοδικά χρονικά διαστήματα στη Sink, ούτως ώστε αυτή να τα επεξεργαστεί κατάλληλα, όποτε κριθεί απαραίτητο για την εξυπηρέτηση κάποιου αιτήματος. Έστω λοιπόν, ότι η Sink αυτού του WSN παραλαμβάνει ένα αίτημα εύρεσης όλων των SNs, οι οποίοι έχουν θερμοκρασία μεγαλύτερη των 80 βαθμών Κελσίου κι επιπλέον η θερμοκρασία τους αυξήθηκε κατά το χρονικό διάστημα των τελευταίων 3 ωρών. Το αίτημα αυτό θα είναι της μορφής:

```
Select s1.sensor_id, s2.sensor_id from sensors s1, sensors s2 where s1.temperature > 80
and s2.temperature>s1.temperature and s2.time>s1.time window [-3,0].
```

Το παραπάνω αίτημα, θα έχει ως αποτέλεσμα να αναζητήσει η Sink στα δεδομένα των μετρήσεων που έχει λάβει από τους SNs, αν κάποιος από αυτούς έχει την τρέχουσα χρονική στιγμή, θερμοκρασία πάνω από 80 βαθμούς Κελσίου (s1.temperature >80) κι επιπλέον η θερμοκρασία του σε κάποια προγενέστερη χρονική στιγμή (s2.time>s1.time) εντός της διάρκειας των τελευταίων 3 ωρών (window [-3,0]) ήταν μεγαλύτερη της τρέχουσας θερμοκρασίας (s2.temperature>s1.temperature). Μόλις η Sink εντοπίσει τους SNs, οι οποίοι και πληρούν τις παραπάνω προϋποθέσεις, τότε θα τους παρέχει ως απάντηση στο παραπάνω αίτημα.

Σχήμα 4.1: Κεντρική εξυπηρέτηση αιτημάτων από τη Sink



Η Sink δεν αποστέλλει το αίτημα σε κανέναν SN, αλλά δέχεται συνεχώς τις μετρήσεις των δεδομένων των αισθητήρων από όλους τους SNs (μπλε βέλη), τι οποίες κι αποθηκεύει προς μελλοντική επεξεργασία. Έτσι, όποτε παραλάβει κάποιο αίτημα ψάχνει στα αποθηκευμένα σε αυτήν δεδομένα για να βρει την απάντηση στο εν λόγω αίτημα. Είναι εμφανής η συνεχής επιβάρυνση του δικτύου και η αυξημένη πιθανότητα λαθών και συγκρούσεων λόγω της συνεχούς αποστολής των δεδομένων από τους SNs στους head SNs κι από αυτούς στη Sink.

Καθίσταται προφανές, ότι το μεγαλύτερο μειονέκτημα της παραπάνω μεθόδου είναι υψηλό κόστος της συνεχούς μετάδοσης των δεδομένων από τους SNs στη Sink. Δηλαδή, οι SNs αποστέλλουν συνεχώς κι ανά τακτά χρονικά διαστήματα τα δεδομένα που προέρχονται από τις μετρήσεις των αισθητήρων τους στη Sink, γεγονός που προκαλεί συμφόρηση ή ακόμη και κατάρρευση στο δίκτυο.

4.4 Τοπική εξυπηρέτηση αιτημάτων από τους SNs

Σε αυτήν την τεχνική, η εξυπηρέτηση των αιτημάτων μεταφέρεται από τη Sink στους SNs. Πιο συγκεκριμένα, οι SNs αποθηκεύουν τα λαμβανόμενα από αυτούς δεδομένα χωρίς να τα αποστέλλουν στη Sink. Έτσι, όποτε η Sink παραλαμβάνει ένα και νέο αίτημα από κάποιο χρήστη, το προωθεί στους SNs, προκειμένου αυτοί να το επεξεργαστούν. Στη συνέχεια, οι SNs αποστέλλουν τα ακριβή δεδομένα των μετρήσεων των αισθητήρων που αποτελούν απάντηση στο αίτημα αυτό στη Sink, ούτως ώστε να δοθούν στο χρήστη που κατέθεσε το συγκεκριμένο αίτημα.

Για παράδειγμα, ας θεωρήσουμε ένα WSN στο οποίο η επεξεργασία των αιτημάτων των χρηστών δεν λαμβάνει χώρα αποκλειστικά στη Sink όπως προηγουμένως, αλλά ένα μεγάλο μέρος αυτής εκτελείται στους ίδιους τους SNs. Το γεγονός αυτό σημαίνει, ότι οι SNs δεν αποστέλλουν τα δεδομένα των μετρήσεων τους συνεχώς στη Sink παρά μόνον όταν χρειαστεί να απαντηθεί κάποιο αίτημα. Έστω τώρα, ότι η Sink παραλαμβάνει το ίδιο αίτημα με προηγουμένως. Σε αυτήν την περίπτωση, η Sink θα πρέπει να αποστείλει 2 αιτήματα. Το πρώτο αίτημα θα αναζητεί όλους τους SNs των οποίων η θερμοκρασία στη δεδομένη χρονική στιγμή είναι μεγαλύτερη από 80 βαθμούς Κελσίου και θα είναι το ακόλουθο:

```
Select s1.sensor_id from sensors s1 into r1 where s1.temperature > 80
```

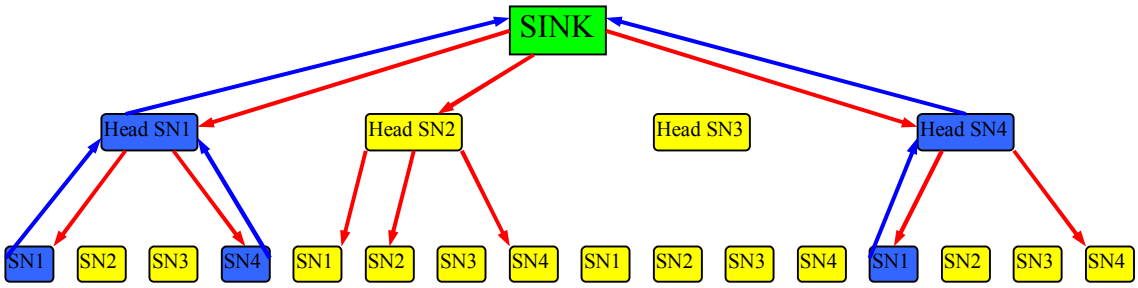
Το παραπάνω αίτημα αποστέλλεται είτε σε όλους τους SNs είτε αν χρησιμοποιείται κατανεμημένο ευρετήριο σε όσους από αυτούς ενδέχεται να έχουν θερμοκρασία μεγαλύτερη των 80 βαθμών Κελσίου. Οι SNs, οι οποίοι επιστρέφουν τον αύξοντα αριθμό τους (sensor_id), δηλαδή την ταυτότητά τους ως απάντηση στο συγκεκριμένο αίτημα τοποθετούνται σε έναν προσωρινό πίνακα r1 στη Sink. Παράλληλα, καθένας από τους αισθητήρες αυτούς αποθηκεύει την τρέχουσα θερμοκρασία που είναι μεγαλύτερη των 80 βαθμών Κελσίου και την τρέχουσα χρονική στιγμή σε έναν ειδικό πίνακα στη μνήμη του και μπαίνει σε κατάσταση αναμονής, περιμένοντας ένα δεύτερο αίτημα που θα αφορά τα δεδομένα που αποθήκευσε στον παραπάνω ειδικό πίνακα. Εν συνεχεία κι αφού η Sink έχει εντοπίσει όλους τους SNs με θερμοκρασία μεγαλύτερη των 80 βαθμών Κελσίου στη δεδομένη χρονική στιγμή, θα ψάξει να βρει σε ποιους από αυτούς αυξήθηκε η θερμοκρασία κατά τις τελευταίες 3 ώρες. Για το σκοπό αυτό θα στείλει σε όλους τους αισθητήρες που βρίσκονται αποθηκευμένοι στον πίνακα r1 το ακόλουθο αίτημα:

```
Select s1.sensor_id from sensors r1 where s1.temperature>r1.temperature
and  s1.time>r1.time window [-3,0]
```

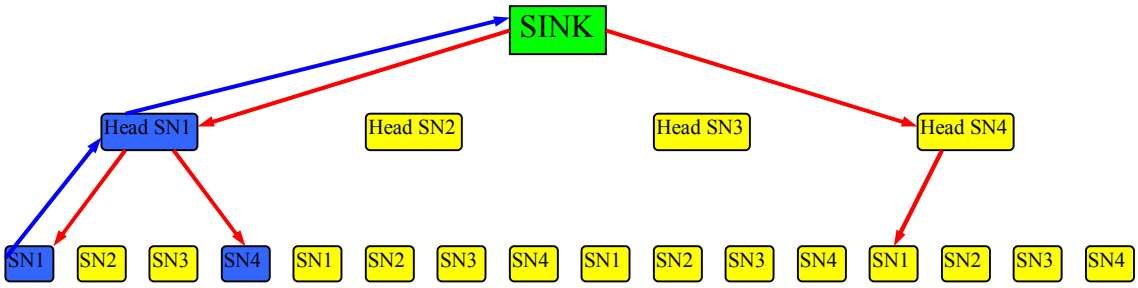
Καθένας από τους SNs, που παραλαμβάνει το παραπάνω αίτημα αναζητά στα δεδομένα του, εάν διαθέτει κάποια μέτρηση θερμοκρασίας (s1.temperature), η οποία να είναι μεγαλύτερη από την τιμή της θερμοκρασίας στον ειδικό πίνακα (r1.temperature) και η οποία να έχει ληφθεί χρονικά (s1.time) μεταγενέστερα από τη χρονική στιγμή που βρίσκεται αποθηκευμένη στον ειδικό πίνακα και η οποία είναι η χρονική στιγμή, κατά την οποία ελήφθη η θερμοκρασία του ειδικού πίνακα, εντός πάντα του χρονικού ορίου των τελευταίων 3 ωρών (window [-3,0]). Με τον τρόπο

αυτό, καθένας από τους SNs του πίνακα r1 βρίσκει αν η θερμοκρασία του αυξήθηκε κατά το χρονικό διάστημα των τελευταίων 3 ωρών. Έτσι, αν πράγματι διαπιστώσει πως ισχύει κάτι τέτοιο, θα αποστείλει τον αύξοντα αριθμό του στη Sink.

Σχήμα 4.2: Τοπική εξυπηρέτηση αιτημάτων από τους SNs



Αποστολή πρώτου αιτήματος. Η Sink αποστέλλει το αίτημα σε όσους από τους head SNs, ενδεχομένως να διαθέτουν στη συστάδά τους και σύμφωνα πάντα με το κατανεμημένο ευρετήριο, SNs με θερμοκρασία άνω των 80 βαθμών Κελσίου (κόκκινα βέλη). Οι head SNs με τη σειρά τους αποστέλλουν το αίτημα αυτό σε όσους SNs της συστάδας τους διαθέτουν σύμφωνα με το κατανεμημένο ευρετήριο θερμοκρασία άνω των 80 βαθμών Κελσίου (κόκκινα βέλη). Όσοι SNs (μπλε χρώμα) διαθέτουν πράγματι την κατάλληλη θερμοκρασία αποστέλλουν τα id τους στους head SNs τους (μπλε χρώμα), οι οποίοι με τη σειρά τους στέλνουν τα id τους στη Sink.



Αποστολή δεύτερου αιτήματος. Η Sink αποστέλλει το αίτημα μόνον στους head SNs που διαθέτουν στη συστάδά τους SNs με θερμοκρασία άνω των 80 βαθμών Κελσίου κι αυτοί με τη σειρά τους προωθούν το αίτημα στους εν λόγω SNs. Όσοι από τους τελευταίους ικανοποιούν τις προϋποθέσεις του δεύτερου αιτήματος αποστέλλουν τα ζητούμενα δεδομένα στο head SN της συστάδας τους, ο οποίος και τα προωθεί στη Sink.

Μέσα από το παραπάνω παράδειγμα, καθίσταται προφανές, ότι η επεξεργασία του αρχικού αιτήματος δεν λαμβάνει αποκλειστικά χώρα πλέον στη Sink, αλλά το μεγαλύτερο μέρος αυτής εκτελείται στους SNs του WSN. Το γεγονός αυτό έχει το πλεονέκτημα, ότι δεν χρειάζεται το WSN να επιβαρύνεται συνεχώς από τις πολλαπλές αποστολές των δεδομένων των αισθητήρων στη Sink, ούτως ώστε αυτή να τα επεξεργαστεί, μιας και η επεξεργασία των ζητούμενων δεδομένων εκτελείται πλέον στους ίδιους τους SNs. Μπορεί βέβαια σε αυτήν την περίπτωση, το WSN να επιβαρύνεται από την αποστολή των αιτημάτων, αλλά λαμβάνοντας υπόψιν μας το σημαντικά μικρό μέγεθος των αιτημάτων σε σύγκριση με το πολύ μεγαλύτερο μέγεθος των δεδομένων από τις μετρήσεις των SNs καθίσταται προφανής η αποσυμφόρηση του δικτύου που επιτυγχάνεται με τον τρόπο αυτό. Είναι δηλαδή, σαν να ανταλλάσσουμε τις μεγάλου όγκου αποστολές δεδομένων από τους SNs προς τη

Sink, οι οποίες και λαμβάνουν χώρα συνεχώς ανά τακτικά χρονικά διαστήματα με τις αισθητά μικρότερου όγκου αποστολές αιτημάτων από τη Sink προς τους SNs, οι οποίες και λαμβάνουν χώρα μόνο όταν κατατεθεί κάποιο αίτημα από κάποιο χρήστη. Έτσι λοιπόν και όπως φαίνεται και στα παραπάνω σχήματα, για την εξυπηρέτηση του αιτήματος στο πρώτο WSN απαιτήθηκε η αποστολή των δεδομένων των αισθητήρων συνολικά 20 φορές, ενώ δεν χρειάστηκε να γίνει καμία αποστολή του αιτήματος, ενώ στο δεύτερο WSN χρειάστηκε να γίνουν 15 αποστολές αιτημάτων και μόλις 7 αποστολές δεδομένων. Εντούτοις, το κόστος μετάδοσης των αιτημάτων εντός του WSN αποτελεί ένα σημαντικό μειονέκτημα αυτής τεχνικής.

4.5 Εξυπηρέτηση αιτημάτων σε δύο επίπεδα – Κατανεμημένη εξυπηρέτηση αιτημάτων

Πριν προχωρήσουμε στην παρουσίαση της τεχνικής αυτής, θα πρέπει πρώτα να υπενθυμίσουμε το διαχωρισμό των αιτημάτων των χρηστών σε ακριβή και προσεγγιστικά. Πιο συγκεκριμένα, τα ακριβή αιτήματα ενδιαφέρονται για τα πραγματικά δεδομένα των αισθητήρων, σε αντίθεση με τα προσεγγιστικά αιτήματα που μπορούν να ανεχτούν ένα συγκεκριμένο περιθώριο λάθους στα δεδομένα των αισθητήρων. Η τεχνική αυτή αποτελεί στην ουσία κάτι ενδιάμεσο, ανάμεσα στις δύο προαναφερθείσες τεχνικές με βασικό της χαρακτηριστικό, το γεγονός ότι υποστηρίζει τα προσεγγιστικά αιτήματα των χρηστών, δηλαδή επιτρέπει την ύπαρξη κάποιου ανεκτού περιθωρίου λάθους στα δεδομένα των αισθητήρων που αποτελούν απάντηση στα αιτήματα των χρηστών. Με βάση αυτήν την τεχνική, η εξυπηρέτηση των αιτημάτων πραγματοποιείται σε δύο επίπεδα και για την ακρίβεια, σε ένα πρώτο προσεγγιστικό επίπεδο από τη Sink και σε ένα δεύτερο ακριβές επίπεδο από τους SNs.

Σύμφωνα λοιπόν με την τεχνική εξυπηρέτησης των αιτημάτων σε δύο επίπεδα, οι SNs λαμβάνουν τις τιμές των μετρήσεων, που πραγματοποιούν οι αισθητήρες τους και τις αποθηκεύουν στη μνήμη NAND flash, που διαθέτουν για το σκοπό αυτό. Εν συνεχεία, αποστέλλουν τα δεδομένα αυτά στη Sink, όχι όμως ανά τακτά χρονικά διαστήματα, αλλά όποτε κρίνουν πως πρέπει να κάνουν κάτι τέτοιο. Για να γίνει αυτό πιο κατανοητό παρουσιάζουμε στη συνέχεια, τη διαδικασία της λήψης κι αποστολής μίας μέτρησης από έναν SN.

4.5.1 Διαδικασία λήψης κι αποστολής μιας μέτρησης με κάποιο ανεκτό περιθώριο λάθους από έναν SN

Πιο συγκεκριμένα, κάθε SN χρησιμοποιεί ένα ανεκτό περιθώριο λάθους e, προκειμένου να αποφασίσει αν θα στείλει ή όχι μία καινούρια μέτρηση στη Sink. Έστω λοιπόν ui η νέα μέτρηση του αισθητήρα και ui', η τελευταία μέτρηση του αισθητήρα. Αν η διαφορά μεταξύ τους είναι είτε μεγαλύτερη από e/2 είτε μικρότερη από e/2, δηλαδή αν είναι μεγαλύτερη από το περιθώριο λάθους του SN, τότε ο SN θα στείλει τη ui στη Sink, ενώ σε διαφορετική περίπτωση όχι. Δηλαδή, είναι:

- εάν ui'-ui>e/2 ή ui'-ui<e/2 τότε ο SN θα αποστείλει την ui' στη Sink
- εάν -e/2<ui'-ui<e/2 τότε ο SN δεν θα αποστείλει την ui' στη Sink

Δηλαδή και για να γίνουν τα ανωτέρω ακόμη πιο σαφή, ορίζεται ένα κλειστό διάστημα [-e/2, e/2], το οποίο ονομάζεται φράγμα κι εγκαθίσταται σε κάθε SN. Αν η διαφορά (ui – ui'), μεταξύ της νέας (ui') και της αμέσως προηγηθείσας μέτρησης του SN ανήκει στο φράγμα, τότε η νέα μέτρηση του SN δεν θα αποσταλεί από τον SN στη Sink, ενώ αν η διαφορά αυτή παραβιάζει το φράγμα, τότε θα αποσταλεί. Συνεπώς, είναι :

- Εάν ui'-ui ανήκει στο φράγμα [-e/2, e/2], τότε ο SN δεν θα αποστείλει την ui' στη Sink

- Ενώ, εάν ui'-ui δεν ανήκει στο φράγμα [-e/2, e/2] ο SN θα αποστείλει την ui' στη Sink

Συνεπώς, αν και κάθε SN αποθηκεύει κάθε νέα μέτρηση που κάνει στη NAND flash μνήμη του, δεν την αποστέλλει απευθείας στη Sink παρά μόνον αν αυτή διαφέρει από την τελευταία μέτρηση που έκανε περισσότερο από κάποιο ανεκτό περιθώριο λάθους για αυτόν τον SN. Μέχρι στιγμής λοιπόν , έχουμε δείξει τον τρόπο με τον οποίο γίνεται η αποθήκευση των δεδομένων στους SNs ενός WSN, καθώς και η αποστολή τους από τους SNs στη Sink με τη συγκεκριμένη τεχνική. Στη συνέχεια, θα ασχοληθούμε με την ουσία αυτής της τεχνικής, δηλαδή με το πώς γίνεται η επεξεργασία των αιτημάτων σε αυτήν.

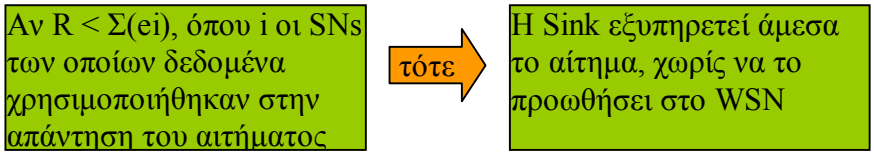
4.5.2 Επεξεργασία αιτήματος από WSN με κατανεμημένη επεξεργασία αιτημάτων

Εκτελούνται τα ακόλουθα βήματα:

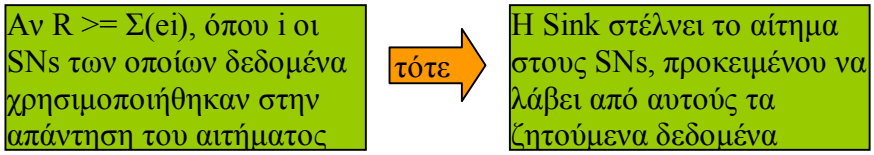
1. Όπως είπαμε λοιπόν, η τεχνική αυτή υποστηρίζει προσεγγιστικά αιτήματα, δηλαδή αιτήματα που ανέχονται κάποιο συγκεκριμένο περιθώριο λάθους στα δεδομένα των μετρήσεων των αισθητήρων, το οποίο και για κάθε αίτημα συμβολίζεται με R. Κάθε φορά που η Sink παραλαμβάνει ένα νέο προσεγγιστικό αίτημα, προσπαθεί πρώτα να το εξυπηρετήσει με βάση τα δεδομένα που βρίσκονται αποθηκευμένα σε αυτήν. Βρίσκει λοιπόν την απάντηση στο εν λόγω αίτημα, χωρίς να το προωθήσει στο δίκτυο κι

2. Εν συνεχεία ελέγχει αν το αποτέλεσμα που βρήκε είναι εντός του ανεκτού περιθωρίου λάθους του αιτήματος αυτού R. Για να το πετύχει αυτό, συγκρίνει το ανεκτό περιθώριο λάθους του αιτήματος R με το άθροισμα των γνωστών σε αυτήν περιθωρίων λάθους e όλων των SNs από τους οποίους και λήφθηκαν δεδομένα για την απάντηση του αιτήματος αυτού.

    - Αν το R είναι μεγαλύτερο του Σ (ei), όπου ei κάθε SN δεδομένα του οποίου χρησιμοποιήθηκαν για την απάντηση του αιτήματος, τότε πάει να πει ότι το περιθώριο λάθους των μετρήσεων που έστειλαν οι αισθητήρες στη Sink δεν υπερβαίνει το ανεκτό περιθώριο λάθους του αιτήματος κι άρα το αίτημα μπορεί να απαντηθεί άμεσα από τη Sink, χωρίς να απαιτείται αποστολή αυτού στο δίκτυο.



- Ειδάλλως, εάν το R είναι μικρότερο του Σ (ei), τότε το περιθώριο λάθους των μετρήσεων που έστειλαν οι αισθητήρες στη Sink υπερβαίνει το ανεκτό περιθώριο λάθους του αιτήματος κι άρα το αίτημα πρέπει να αποσταλεί στο δίκτυο, προκειμένου να ληφθούν οι πραγματικές τιμές των δεδομένων που αποτελούν απάντηση σε αυτό.



4.5.3 Παράδειγμα 4.1 – Κατανεμημένη επεξεργασία αιτήματος σε WSN

Ας θεωρήσουμε ένα WSN, το οποίο αποτελείται από μία 4 SNs καθένας από τους οποίους διαθέτει έναν ενσωματωμένο αισθητήρα μέτρησης της θερμοκρασίας. Το ανεκτό περιθώριο λάθους στις μετρήσεις των αισθητήρων θεωρείται χάριν ευκολίας ίδιο για όλους τους SNs και ίσο με 6 βαθμούς Κελσίου. Οι μετρήσεις των αισθητήρων των SNs έχουν ως εξής:

SN1		**SN2**		**SN3**		**SN4**	
ΧΡΟΝΟΣ	ΒΑΘΜΟΙ ΚΕΛΣΙΟΥ	ΧΡΟΝΟΣ	ΒΑΘΜΟΙ ΚΕΛΣΙΟΥ	ΧΡΟΝΟΣ	ΒΑΘΜΟΙ ΚΕΛΣΙΟΥ	ΧΡΟΝΟΣ	ΒΑΘΜΟΙ ΚΕΛΣΙΟΥ
1 min	20	1 min	35	1 min	23	1 min	20
2 min	25	2 min	37	2 min	29	2 min	25
3 min	27	3 min	33	3 min	27	3 min	29
4 min	26	4 min	38	4 min	29	4 min	24

Πίνακας 4.1: Τιμές μετρήσεων SNs

Αυτό που πρέπει να κοιτάξουμε σε πρώτη φάση είναι ποιες από τις παραπάνω μετρήσεις των αισθητήρων θα σταλούν στη Sink και ποιες όχι, δηλαδή ποιο θα είναι το περιεχόμενο της Sink τη χρονική στιγμή που έπεται των μετρήσεων αυτών. Έχουμε λοιπόν, αφού e=6 => e/2=3, οπότε θα είναι:

- 1 min

Όλοι οι SNs θα στείλουν τις πρώτες μετρήσεις τους στη Sink, οπότε θα είναι:

SINK	SN1 -------- 20	SN2 -------- 35	SN3 -------- 23	SN4 -------- 20

- 2 min

SN1: 25 – 20 = 5 > 3, άρα αποστέλλεται στη Sink
SN2: 37 – 35 = 2 < 3, άρα δεν αποστέλλεται στη Sink
SN3: 29 – 23 = 6 > 3, άρα αποστέλλεται στη Sink
SN4: 25 – 20 = 5 > 3, άρα αποστέλλεται στη Sink, οπότε θα είναι:

SINK	SN1 -------- 25	SN2 -------- 35	SN3 -------- 29	SN4 -------- 25

- 3 min

SN1: 26 – 25 = 1 < 3, άρα δεν αποστέλλεται στη Sink
SN2: 38 – 37 = 1 < 3, άρα δεν αποστέλλεται στη Sink
SN3: 29 – 29 = 0 < 3, άρα δεν αποστέλλεται στη Sink
SN4: 24 – 25 = 0 < 3, άρα δεν αποστέλλεται στη Sink, οπότε θα είναι:

SINK	SN1 -------- 25	SN2 -------- 33	SN3 -------- 29	SN4 -------- 29

- 4 min

SN1: 25 – 25 = 5 > 3, άρα αποστέλλεται στη Sink
SN2: 37 – 37 = 2 < 3, άρα δεν αποστέλλεται στη Sink
SN3: 29 – 29 = 6 > 3, άρα αποστέλλεται στη Sink
SN4: 25 – 25 = 5 > 3, άρα αποστέλλεται στη Sink, οπότε θα είναι:

SINK	SN1 -------- 25	SN2 -------- 33	SN3 -------- 29	SN4 -------- 29

Δηλαδή κι όπως παρατηρούμε μετά και την τέταρτη χρονική στιγμή τα δεδομένα των μετρήσεων των αισθητήρων, όπως αυτά είναι αποθηκευμένα στη Sink, διαφέρουν από τα πραγματικά δεδομένα των μετρήσεων των αισθητήρων, όπως αυτά είναι αποθηκευμένα σε αυτούς.

4min Προσεγγιστικά δεδομένα στη Sink	SN1- 25	SN2- 33	SN3- 29	SN4- 29
4min Πραγματικά δεδομένα στους SNs	SN1- 26	SN2- 38	SN3- 29	SN4- 24

Έστω τώρα, ότι η Sink παραλαμβάνει ένα αίτημα υπολογισμού της μέσης τιμής της θερμοκρασίας που μέτρησαν οι αισθητήρες κατά τη χρονική στιγμή 4 min με ανεκτό περιθώριο λάθους του αιτήματος ίσο με 5 βαθμούς Κελσίου, το οποίο και είναι το εξής:

Select avg(temperature) from sensors tolerance 5

Η Sink πρώτα θα υπολογίσει τη μέση θερμοκρασία με βάση τα προσεγγιστικά δεδομένα που έχει αποθηκευμένα στη μνήμη της, οπότε θα είναι:

- Avg = (25 + 33 + 29 +29) / 4 = 29 βαθμούς Κελσίου

Στη συνέχεια και προκειμένου να αποφασίσει αν θα δεχτεί το παραπάνω αποτέλεσμα ή όχι, θα χρησιμοποιήσει τον παραπάνω τύπο για να συγκρίνει το ανεκτό όριο λάθους του αιτήματος με το άθροισμα τον ορίων λάθους των SNs, οπότε θα έχουμε:

- $R = 5 < \Sigma(ei) – 2*M = 6 + 6 + 6 + 6 – (2*4) = 24 – 8 = 16$

Οπότε, η Sink θα αποστείλει αίτημα στους SNs για να λάβει τις πραγματικές τιμές των δεδομένων τους και να υπολογίσει με βάση αυτές, τη μέση τιμή.

Ας θεωρήσουμε την περίπτωση, κατά την οποία το ανεκτό περιθώριο λάθους R του αιτήματος ισούται με 17 βαβμούς Κελσίου. Τότε, θα είναι:

- $R = 5 > \Sigma(ei) – 2*M = 6 + 6 + 6 + 6 – (2*4) = 24 – 8 = 16.$

Οπότε, σε αυτήν την περίπτωση η Sink θεωρεί ικανοποιητικά ακριβές το αποτέλεσμα και το παρέχει ως απάντηση στο χρήστη που κατέθεσε το αίτημα, χωρίς να το προωθήσει στο WSN κι αποσυμφορώντας με τον τρόπο αυτό το δίκτυο.

Μετά την παρουσίαση και των τριών προαναφερθείσων τεχνικών εξυπηρέτησης αιτημάτων, η προτεινόμενη από την παρούσα μελέτη τεχνική είναι αυτή της κατανεμημένης επεξεργασίας αιτημάτων, επειδή παρουσιάζει μεγαλύτερη ευελιξία κατά τη διαχείριση των αιτημάτων.

4.6 Βελτιστοποίηση αιτημάτων

Ένα WSN μπορεί να επεξεργάζεται ταυτόχρονα σε κάθε χρονική στιγμή ένα μεγάλο πλήθος αιτημάτων, επιστρέφοντας ξεχωριστά σε καθένα από αυτά τις αντίστοιχες μετρήσεις των αισθητήρων, γεγονός το οποίο μπορεί να οδηγήσει σε υπερφόρτωση του δικτύου. Με δεδομένο όμως, ότι πολλά από τα αιτήματα αυτά, ενδεχομένως να αφορούν κοινά δεδομένα των αισθητήρων, τα αιτήματα θα μπορούσαν να επωφεληθούν μεταξύ τους μέσω του διαμοιρασμού των κοινών τους δεδομένων. Με τον τρόπο αυτό θα αποσυμφορηθεί το δίκτυο από την επαναμετάδοση των ίδιων δεδομένων και θα εξοικονομηθούν πολύτιμοι ενεργειακοί πόροι. Αυτό ακριβώς αποτελεί και το αντικείμενο της τρέχουσας ενότητας, δηλαδή το πώς μπορεί να οικοδομηθεί κάποια μορφή συνεργασίας μεταξύ των αιτημάτων, με απώτερο σκοπό τη βελτιστοποίησή τους και την αύξηση της αποδοτικότητας του WSN κατά την εξυπηρέτηση τους.

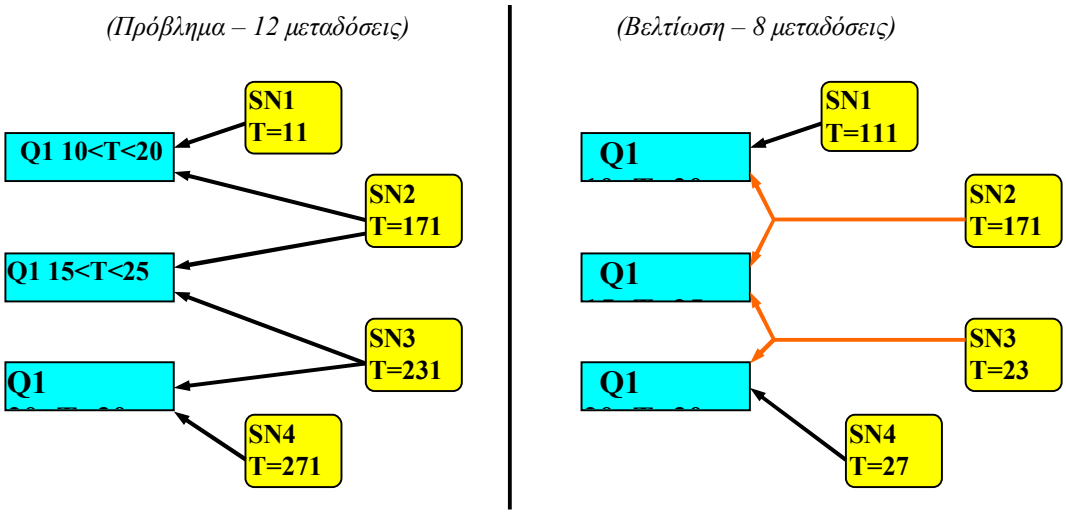
4.6.1 Ταυτόχρονη διαχείριση πολλαπλών αιτημάτων από ένα WSN

Για να γίνουμε πιο σαφείς θα παρουσιάσουμε ένα σχετικό παράδειγμα, μέσω του οποίου και αναδεικνύεται το πρόβλημα της ταυτόχρονης διαχείρισης των αιτημάτων από ένα WSN. Ας θεωρήσουμε λοιπόν ένα WSN εφοδιασμένο με αισθητήρες μέτρησης της θερμοκρασίας, στο οποίο σε κάποια χρονική στιγμή εισάγονται ταυτόχρονα τα ακόλουθα τρία αιτήματα:

```
1: select sensor_id from sensors where 10 < temperature <20 error tolerance 1 window [-3,0]
2: select sensor_id from sensors where 15 < temperature <25 error tolerance 1 window [-3,0]
3: select sensor_id from sensors where 20 < temperature <30 error tolerance 1 window [-3,0]
```

Το πρώτο αίτημα αναζητεί όλους τους αισθητήρες, οι οποίοι μέτρησαν κατά τις τελευταίες τρεις ώρες θερμοκρασίες από 10 μέχρι 20 βαθμούς Κελσίου, το δεύτερο αίτημα όσους μέτρησαν θερμοκρασίες από 15 μέχρι 25 βαθμούς Κελσίου και το τρίτο αίτημα όσους μέτρησαν θερμοκρασίες από 20 μέχρι 30 βαθμούς Κελσίου.

Είναι προφανές λοιπόν, ότι εάν ένας αισθητήρας έχει λάβει κατά τις προηγούμενες τρεις ώρες θερμοκρασία της τάξης των 17 βαθμών Κελσίου, τότε θα μεταδόσει τα δεδομένα του δύο φορές, από μία για καθένα από τα αιτήματα 1 και 2. Αντίστοιχα, εάν ένας αισθητήρας έχει λάβει κατά τις προηγούμενες τρεις ώρες θερμοκρασία της τάξης των 23 βαθμών Κελσίου, τότε θα μεταδόσει τα δεδομένα του δύο φορές, από μία για καθένα από τα αιτήματα 2 και 3.



Σχήμα 4: Πολλαπλή αποστολή δεδομένων από αισθητήρα

Το γεγονός αυτό μπορεί να οδηγήσει σε συγκρούσεις κατά τη μετάδοση των δεδομένων με άμεση συνέπεια, τόσο τη συμφόρηση των ασύρματων ζεύξεων του δικτύου, όσο και ενδεχόμενη απώλεια κάποιων εκ των δεδομένων, με αποτέλεσμα την ανάγκη επαναμετάδοσης τους, με ότι αυτό συνεπάγεται σε κόστος ενέργειας κι επικοινωνίας για το δίκτυο. Έτσι λοιπόν, στη συνέχεια θα εξετάσουμε έναν αλγόριθμο βελτιστοποίησης των αιτημάτων, ο οποίος κι επιτρέπει σε παρόμοια αιτήματα να μοιράζονται τους περιορισμένους διαθέσιμους επικοινωνιακούς κι ενεργειακούς πόρους.

Προτού προχωρήσουμε στην παρουσίαση αυτού του αλγορίθμου, θα πρέπει να αναφέρουμε σε ποιο σημείο λαμβάνει χώρα, η εκτέλεση αυτού. Έτσι λοιπόν, όπως έχει πολλάκις αναφερθεί ένα WSN αποτελείται από μία κεντρική βάση συγκέντρωσης δεδομένων (Sink), καθώς κι ένα μεγάλο πλήθος κόμβων αισθητήρων (SNs). Τα αιτήματα των χρηστών εισάγονται στη Sink κι εν συνεχεία διαχέονται στο WSN προκειμένου να επεξεργαστούν. Σε κάθε χρονική στιγμή η Sink διατηρεί μία λίστα με όλα τα αιτήματα, τα οποία έχουν ήδη υποβληθεί σε αυτήν, τα δεδομένα που αποτελούν απάντηση στα αιτήματα αυτά, καθώς και το κόστος καθενός εξ' αυτών. Το κόστος ενός αιτήματος ορίζεται ως το πλήθος των μεταδόσεων που πρέπει να πραγματοποιηθούν εντός του δικτύου για την εξυπηρέτηση του αιτήματος αυτού. Δεδομένης της αυξημένης υπολογιστικής ισχύος, ενεργειακής αυτονομίας και αποθηκευτικών χώρων της Sink σε σχέση με τους SNs είναι προφανές, ότι ο αλγόριθμος βελτιστοποίησης των αιτημάτων είναι βολικό να λαμβάνει χώρα στη Sink και πριν αυτά διαδοθούν στο WSN.

(Αίτημα)	(Κόστος)	(Δεδομένα)
Query q1	Cost(q1)	Data(q1)
Query q2	Cost(q2)	Data(q2)
Query q3	Cost(q3)	Data(q3)
Query q4	Cost(q4)	Data(q4)
Query q5	Cost(q5)	Data(q5)

Σχήμα 4.4: Λίστα αιτημάτων της SINK

4.6.2 Αλγόριθμος Βελτιστοποίησης Αιτημάτων στη Sink

Ο αλγόριθμος βελτιστοποίησης των αιτημάτων στη Sink, έχει ως εξής:

1)
Όποτε παραλαμβάνεται ένα καινούργιο αίτημα (qi) από κάποιον χρήστη, τότε υπολογίζεται πρώτα το κόστος (cost) του αιτήματος αυτού. Δηλαδή, όπως είπαμε και πριν υπολογίζεται το πλήθος των μεταδόσεων που πρέπει να πραγματοποιηθούν εντός του δικτύου για την εξυπηρέτηση του εν λόγω αιτήματος.

2)
Στη συνέχεια, διατρέχουμε σειριακά τη λίστα των ήδη υποβληθέντων αιτημάτων, προκειμένου να διαπιστώσουμε, αν υπάρχει σε αυτήν κάποιο αίτημα (qj) που να είναι παρόμοιο με το αίτημά μας (qi). Πιο συγκεκριμένα, παίρνουμε αρχικά το πρώτο αίτημα της λίστας της Sink. Με βάση λοιπόν το αίτημα αυτό και το νεοεισελθέν αίτημά μας (qi), φτιάχνουμε δύο ή και περισσότερα νέα αιτήματα (qk) και (qm) και υπολογίζουμε το κόστος αυτού. Εν συνεχεία, αφαιρούμε από το κόστος των νέων αιτημάτων (qk) και (qm), το άθροισμα του κόστους του αιτήματός μας (qi) και του τρέχοντος πρώτου αιτήματος (q1) της λίστας Sink. Το αποτέλεσμα καλείται όφελος (benefit) μεταξύ των αιτημάτων (qi) και (q1). Δηλαδή, είναι:

$$\text{Όφελος} = (\text{κόστος}(qi) + \text{κόστος}(q1)) - (\text{κόστος}(qk) + \text{κόστος}(qm))$$

Δηλαδή, το όφελος μας δίνει τη μείωση του κόστους που προκύπτει από την αντικατάσταση του αιτήματος (qi) από τα αιτήματα (qk) και (qm).

3)
Αν όλα τα υπολογιζόμενα οφέλη που προκύπτουν είναι ίσα ή μικρότερα του μηδενός, τότε τερματίζουμε την τρέχουσα εκτέλεση του αλγορίθμου και διαδίδουμε το αίτημα (qi) στο δίκτυο. Μόλις ολοκληρωθεί η επεξεργασία του αιτήματος (qi), το εισάγουμε στη λίστα των αιτημάτων της Sink μαζί με τα δεδομένα που αποτελούν απάντηση σε αυτό. Ειδάλλως, προχωράμε στην εκτέλεση του επόμενου βήματος (4).

4)
Επιλέγουμε το αίτημα της λίστας (qj) με το μεγαλύτερο όφελος και τότε

- Αν όλα τα ζητούμενα δεδομένα του αιτήματός μας (qi) έχουν ήδη ζητηθεί κι απαντηθεί από το αίτημα (qj), τότε προφανώς και λαμβάνουμε τα ζητούμενα δεδομένα του αιτήματος (qi) από τα δεδομένα που αποτελούν απάντηση στο αίτημα (qj). Με τον τρόπο αυτό επιτυγχάνουμε την άμεση εξυπηρέτηση του αιτήματος (qi) χωρίς καμία απολύτως επιβάρυνση του φόρτου του WSN. Μόλις ολοκληρωθεί η επεξεργασία του αιτήματος qi, το εισάγουμε στη λίστα των αιτημάτων της Sink μαζί με τα δεδομένα που αποτελούν απάντηση σε αυτό.

Τα δεδομένα που ζητάει το αίτημα qi έχουν ήδη παραληφθεί από το αίτημα qj.

- Αν ένα υποσύνολο των ζητούμενων δεδομένων του αιτήματος μας (qi) έχουν ήδη ζητηθεί κι απαντηθεί από το αίτημα (qj), τότε κατασκευάζουμε από τα αιτήματα (qi) και (qj) δύο νέα αιτήματα (qk) και (qm). Το ένα νέο αίτημα (qk) θα ζητάει όλα εκείνα τα δεδομένα που ζητάει και το αίτημα (qi) και τα οποία δεν έχουν ήδη απαντηθεί στο αίτημα (qj). Ως εκ τούτου, το αίτημα (qk) θα διαδίδεται στο WSN, προκειμένου να εξυπηρετηθεί από αυτό. Το άλλο αίτημα (qm) θα ζητάει όλα εκείνα τα δεδομένα που έχουν ήδη απαντηθεί από το αίτημα (qj), οπότε και δεν θα διαδίδεται στο WSN, αλλά θα τα λαμβάνει απευθείας από το αίτημα (qj). Έτσι επιτυγχάνουμε να αναζητήσουμε στο WSN μόνο ένα τμήμα από το σύνολο των δεδομένων που ζητάει το αίτημα qi επιταχύνοντας την εξυπηρέτηση αυτού κι αποσυμφορώντας παράλληλα το δίκτυο. Μόλις ολοκληρωθεί η επεξεργασία του αιτήματος (qk), τότε το εισάγουμε στη λίστα των αιτημάτων της Sink μαζί με τα δεδομένα που αποτελούν απάντηση σε αυτό. Το ίδιο πράττουμε φυσικά και για τα αιτήματα (qi) και (qm).

Κάποια από τα δεδομένα που ζητάει qi τα έχει λάβει το qj.

5)
Τέλος και μετά την ολοκλήρωση της εκτέλεσης του αλγορίθμου για το αίτημα (qi) και την ενημέρωση της λίστας της Sink, επανεκτελούμε τον αλγόριθμο για τη νέα ενημερωμένη λίστα αιτημάτων της Sink.

4.6.3 Παράδειγμα 4.2 – Βελτιστοποίηση αιτημάτων στη Sink

Ας παρουσιάσουμε όμως, μία εκτέλεση του παραπάνω αλγορίθμου ειδικά για την πιο πολύπλοκη περίπτωση, κατά την οποία εισάγονται ταυτόχρονα 3 διαφορετικά αιτήματα στη Sink για το προαναφερθέν παράδειγμα αιτημάτων. Ας θεωρήσουμε λοιπόν, ότι έχουμε το WSN του σχήματος 3 κι ότι σε κάποια δεδομένη χρονική στιγμή δέχεται στην είσοδό του, τα ακόλουθα τρία αιτήματα:

1:select sensor_id from sensors where 10 < temperature <20 error tolerance 1 window [-3,0]
2:select sensor_id from sensors where 15 < temperature <25 error tolerance 1 window [-3,0]
3:select sensor_id from sensors where 20 < temperature <30 error tolerance 1 window [-3,0]

Τα κόστη καθενός εξ' αυτών των αιτημάτων ισούνται με:

- cost(q1)= 4 (Λόγω της αμφίδρομης επικοινωνίας με τους SN1 και SN2)
- cost(q2)= 4 (Λόγω της αμφίδρομης επικοινωνίας με τους SN2 και SN3)
- cost(q3)= 4 (Λόγω της αμφίδρομης επικοινωνίας με τους SN3 και SN4)

$1^{η}$ εκτέλεση)

Ο αλγόριθμος βελτιστοποίησης αιτημάτων θα εκτελεστεί πρώτα για το αίτημα q1. Παίρνουμε λοιπόν αρχικά κι εξετάζουμε το αίτημα q1 με το αίτημα q2.Από τα αιτήματα q1 και q2 μπορούν να κατασκευαστούν τα ακόλουθα δύο αιτήματα q4 και q5:

4:select sensor_id from sensors where 10 < temperature <15 error tolerance 1 window [-3,0]
5:select sensor_id from sensors where 15 < temperature <20 error tolerance 1 window [-3,0]

με κόστη ίσα με

- cost(q4)= 2 (Λόγω της αμφίδρομης επικοινωνίας με τον SN1)
- cost(q5)= 2 (Λόγω της αμφίδρομης επικοινωνίας με τον SN2)

Σε αυτήν την περίπτωση το αίτημα q1 θα λάβει το ένα μέρος των ζητούμενων δεδομένων και πιο συγκεκριμένα τα δεδομένα που αφορούν θερμοκρασίες από 15 μέχρι 20 βαθμούς Κελσίου από το αίτημα q2, ενώ για το άλλο μέρος των ζητούμενων δεδομένων και πιο συγκεκριμένα τα δεδομένα που αφορούν θερμοκρασίες από 10 μέχρι 15 βαθμούς Κελσίου θα πρέπει να διαδοθεί το αίτημα q στο WSN. Κατά συνέπεια το όφελος από την παραπάνω ενέργεια θα ισούται με:

- benefit(q1,q2) = cost(q1) + cost(q2) – cost(q3) – cost(q4) = 4 + 4 - 2 – 2 = 4

Εν συνεχεία παίρνουμε κι εξετάζουμε το αίτημα q1 με το αίτημα q3 και διαπιστώνουμε, ότι τα αιτήματα αυτά δεν μπορούν να συνδυαστούν προς βελτιστοποίηση. Συνεπώς, μετά από αυτήν την πρώτη εκτέλεση του αλγορίθμου στη λίστα των αιτημάτων της Sink θα περιέχονται τα αιτήματα q4 και q5.

$2^{η}$ εκτέλεση)

Στη συνέχεια, αλγόριθμος βελτιστοποίησης αιτημάτων θα εκτελεστεί για το αίτημα q2. Η εκτέλεση του αλγορίθμου για τα αιτήματα q1 και q2 έχει ήδη πραγματοποιηθεί, οπότε παίρνουμε κι εξετάζουμε το αίτημα q2 με το αίτημα q3. Από τα αιτήματα αυτά μπορούν να κατασκευαστούν τα ακόλουθα δύο αιτήματα q6 και q7:

```
6:select sensor_id from sensors where 20 < temperature <25 error tolerance 1 window
[-3,0]
7:select sensor_id from sensors where 15 < temperature <20 error tolerance 1 window
[-3,0]
```

με κόστη ίσα με

- cost(q6)= 2 (Λόγω της αμφίδρομης επικοινωνίας με τον SN3)
- cost(q7)= 2 (Λόγω της αμφίδρομης επικοινωνίας με τον SN2)

Σε αυτήν την περίπτωση το αίτημα q2 θα λάβει το ένα μέρος των ζητούμενων δεδομένων και πιο συγκεκριμένα τα δεδομένα που αφορούν θερμοκρασίες από 20 μέχρι 25 βαθμούς Κελσίου από το αίτημα q3, ενώ για το άλλο μέρος των ζητούμενων δεδομένων και πιο συγκεκριμένα τα δεδομένα που αφορούν θερμοκρασίες από 15 μέχρι 20 βαθμούς Κελσίου θα πρέπει να διαδοθεί το αίτημα q7 στο WSN. Κατά συνέπεια το όφελος από την παραπάνω ενέργεια θα ισούται με:

- benefit(q2,q3) = cost(q2) + cost(q3) – cost(q6) – cost(q7) = 4 + 4 - 2 – 2 = 4

Συνεπώς, μετά από αυτήν τη δεύτερη εκτέλεση του αλγορίθμου, στη λίστα των αιτημάτων θα περιέχονται τα αιτήματα q4, q5, q6 και q7. Όπως παρατηρούμε όμως τα αιτήματα q5 και q7 είναι ίδια μεταξύ τους, γεγονός που μας επιτρέπει να διαγράψουμε το q7.

$3^{η}$ εκτέλεση)

Στη συνέχεια, αλγόριθμος βελτιστοποίησης αιτημάτων θα εκτελεστεί για το αίτημα q3. Όμως, η εκτέλεση του αλγορίθμου για το q3, τόσο με το q1 όσο και με το q2 έχει ήδη πραγματοποιηθεί προηγουμένως. Οπότε θα εκτελέσουμε τον αλγόριθμο για το αίτημα q3 σε συνδυασμό με τα αιτήματα της λίστας αιτημάτων. Έτσι λοιπόν, το αίτημα q3 διαπιστώνουμε, ότι δεν μπορεί να συνδυαστεί προς βελτιστοποίηση με κανένα εκ των q4 και q5. Αντιθέτως μπορεί να συνδυαστεί με το q6, οπότε και θα προκύψει ένα νέο αίτημα το q7 και θα έχουμε:

```
6:select sensor_id from sensors where 20 < temperature <25 error tolerance 1 window
[-3,0]
7:select sensor_id from sensors where 25 < temperature <30 error tolerance 1 window
[-3,0]
```

- cost(q6)= 2 (Λόγω της αμφίδρομης επικοινωνίας με τον SN3)
- cost(q7)= 2 (Λόγω της αμφίδρομης επικοινωνίας με τον SN4)

Σε αυτήν την περίπτωση το αίτημα q3 θα λάβει το ένα μέρος των ζητούμενων δεδομένων και πιο συγκεκριμένα τα δεδομένα που αφορούν θερμοκρασίες από 20 μέχρι 25 βαθμούς Κελσίου από το αίτημα q6, ενώ για το άλλο μέρος των ζητούμενων δεδομένων και πιο συγκεκριμένα τα δεδομένα που αφορούν θερμοκρασίες από 25 μέχρι 30 βαθμούς Κελσίου θα πρέπει να διαδοθεί το αίτημα q7 στο WSN. Κατά συνέπεια το όφελος από την παραπάνω ενέργεια θα ισούται με:

- benefit(q3,q6) = cost(q3) + cost(q6) – cost(q6) – cost(q7) = 4 + 2 - 2 – 2 = 2

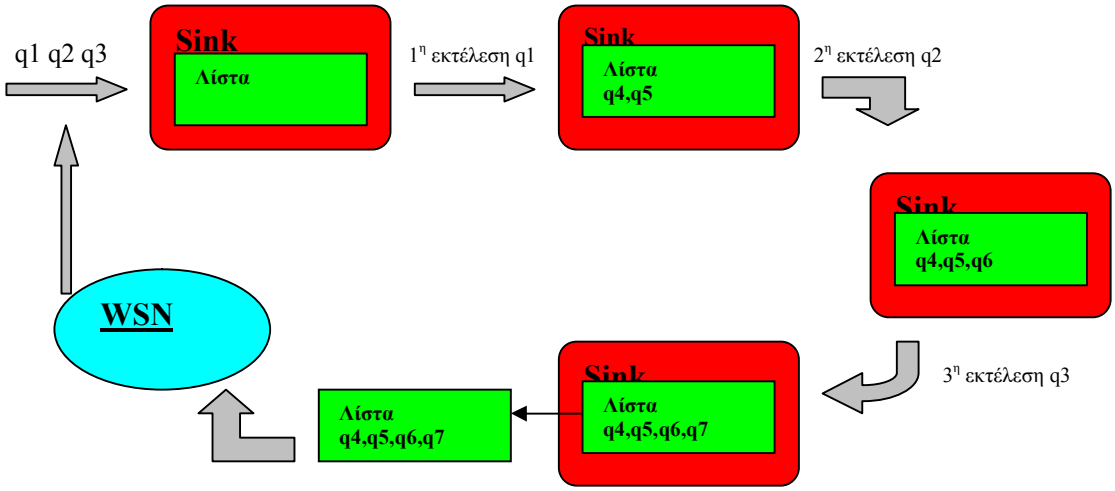
Συνεπώς, μετά από αυτήν τη δεύτερη εκτέλεση του αλγορίθμου, στη λίστα των αιτημάτων θα περιέχονται τα αιτήματα q4, q5, q6 και q7.

Στη συνέχεια επανεκτελούμε τον αλγόριθμο για τη νέα λίστα αιτημάτων και διαπιστώνουμε ότι δεν μπορεί να υπάρξει κάποιο περαιτέρω κέρδος, οπότε και τερματίζουμε. Έτσι τελικά από τα τρία αιτήματα q1, q2 και q3, το κόστος επεξεργασίας των οποίων ισούται με 12, προέκυψε η λίστα αιτημάτων Q το κόστος επεξεργασίας των οποίων ισούται με 8 και η οποία θα περιλαμβάνει τελικά τα ακόλουθα αιτήματα:

```
4:select sensor_id from sensors where 10 < temperature <15 error tolerance 1
window [-3,0]
5:select sensor_id from sensors where 15 < temperature <20 error tolerance 1
window [-3,0]
6:select sensor_id from sensors where 20 < temperature <25 error tolerance 1
window [-3,0]
7:select sensor_id from sensors where 25 < temperature <30 error tolerance 1
window [-3,0]
```

Τα παραπάνω αιτήματα της λίστας Q είναι τελικά αυτά που θα διαδοθούν προς επεξεργασία στο WSN και οι απαντήσεις στα οποία θα χρησιμοποιηθούν για να απαντηθούν και τα αιτήματα q1, q2 και q3.Από το παραπάνω παράδειγμα γίνεται σαφές λοιπόν, το πώς η βελτιστοποίηση των αιτημάτων μπορεί να εξοικονομήσει πόρους και να βελτιώσει την αποδοτικότητα ενός WSN.

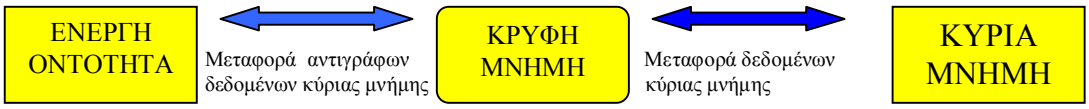
Σχήμα 4.5: Βελτιστοποίηση αιτημάτων στη Sink (Παράδειγμα 4.2)



Ο αλγόριθμος, τον οποίο και παρουσιάσαμε μετράει το κόστος κάθε αιτήματος με το πλήθος των μεταδόσεων που απαιτούνται για την εξυπηρέτησή του και δεν λαμβάνει υπόψιν του άλλες παραμέτρους όπως για παράδειγμα το εύρος ζώνης των ασύρματων ζεύξεων. Μελετώνται κι αναπτύσσονται συνεχώς διάφορες παραλλαγές και βελτιώσεις του αλγορίθμου αυτού, με βάση και τις ιδιαιτερότητες του κάθε WSN και τις ανάγκες της εφαρμογής που αυτό εξυπηρετεί. Πάντως για μια ακόμη φορά καθίσταται σαφές, ότι η βελτιστοποίηση των αιτημάτων λαμβάνοντας χώρα στη Sink μπορεί να αποσυμφορήσει σε μεγάλο βαθμό ένα WSN και να βελτιώσει συνολικά την αποδοτικότητά του.

4.7 Κρυφή μνήμη

Ένας ακόμη τρόπος για τη μείωση του χρόνου επεξεργασίας των αιτημάτων, καθώς και του κόστους μετάδοσης των δεδομένων σε ένα WSN είναι η χρησιμοποίηση κρυφής μνήμης (cache memory) στους κόμβους αισθητήρες. Στην επιστήμη των υπολογιστών ως κρυφή μνήμη ορίζεται η μνήμη υπολογιστή που αποθηκεύει μία συλλογή από δεδομένα, τα οποία κι αποτελούν αντίγραφα τιμών που βρίσκονται κάπου αλλού. Στη γενικότερη περίπτωση υπάρχει μία ενεργή οντότητα που ενεργεί πάνω σε ένα σύνολο δεδομένων, τα οποία και βρίσκονται αποθηκευμένα σε μία μονάδα κύριας μνήμης. Η κρυφή μνήμη παρεμβάλλεται ανάμεσα στην ενεργή οντότητα και στην κύρια μνήμη περιέχοντας αντίγραφα τιμών των δεδομένων της τελευταίας. Η κρυφή μνήμη ναι μεν έχει μεγαλύτερο κόστος από την κύρια μνήμη, αλλά είναι σημαντικά μικρότερη σε μέγεθος και ταχύτερη σε λειτουργία από αυτήν. Έτσι, τα δεδομένα της κύριας μνήμης που χρησιμοποιούνται περισσότερο αποθηκεύονται στην κρυφή μνήμη, ούτως ώστε να είναι ευκολότερα και ταχύτερα προσβάσιμα από την ενεργή οντότητα, αυξάνοντας με τον τρόπο αυτό τη συνολική αποδοτικότητα του συστήματος.



Σχήμα 4.6: Βασική οργάνωση συστήματος με κρυφή μνήμη

Για την κατανόηση της βασικής λειτουργίας της κρυφής μνήμης μπορεί να βοηθήσει το παράδειγμα της χρήσης του ψυγείου σε μία κουζίνα. Αν δεν υπήρχε το ψυγείο για την αποθήκευση και τη συντήρηση των τροφίμων εντός της κουζίνας μας, τότε θα έπρεπε να μεταβαίνουμε σχεδόν σε καθημερινή βάση σε κάποιο κατάστημα για την αγορά των απαραίτητων προϊόντων, γεγονός που θα μείωνε την αποδοτικότητα μας. Η λειτουργία όμως του ψυγείου ως κρυφής μνήμης της κουζίνας, μας βοηθάει να προμηθευτούμε τα υλικά για μεγαλύτερο χρονικό διάστημα εξοικονομώντας με τον τρόπο αυτό πολύτιμο χρόνο. Το γνωστότερο παράδειγμα κρυφής μνήμης είναι η κρυφή μνήμη του επεξεργαστή σε έναν ηλεκτρονικό υπολογιστή, η οποία και μεσολαβεί μεταξύ της κεντρικής μνήμης RAM και της Κεντρικής Μονάδας Επεξεργασίας (CPU), ενώ το πρώτο τσιπ με ενσωματωμένη κρυφή μνήμη υπήρξε το τσιπ 80486 της Intel, δηλαδή ο μετέπειτα Pentium.

4.7.1 Λειτουργία συστήματος με κρυφή μνήμη

Σε γενικές γραμμές η λειτουργία ενός συστήματος με κρυφή μνήμη γίνεται ως εξής: Όταν υπάρχει αίτηση για κάποια εγγραφή στη μνήμη, τότε εξετάζεται πρώτα αν η εγγραφή αυτή υπάρχει στην κρυφή μνήμη. Αν ναι, τότε ανακτάται από εκεί και λέγεται πως σημειώθηκε επιτυχία (hit). Αν όχι, τότε σημειώνεται αποτυχία (miss) και λαμβάνει χώρα ανάκτηση της εγγραφής από την πιο αργή κύρια μνήμη, ενώ παράλληλα τοποθετείται η εγγραφή στην κρυφή μνήμη, αντικαθιστώντας κάποια άλλη ήδη υπάρχουσα εκεί εγγραφή. Το κόστος μπορεί να μετρηθεί από το πλήθος των αποτυχιών που σημειώνονται.

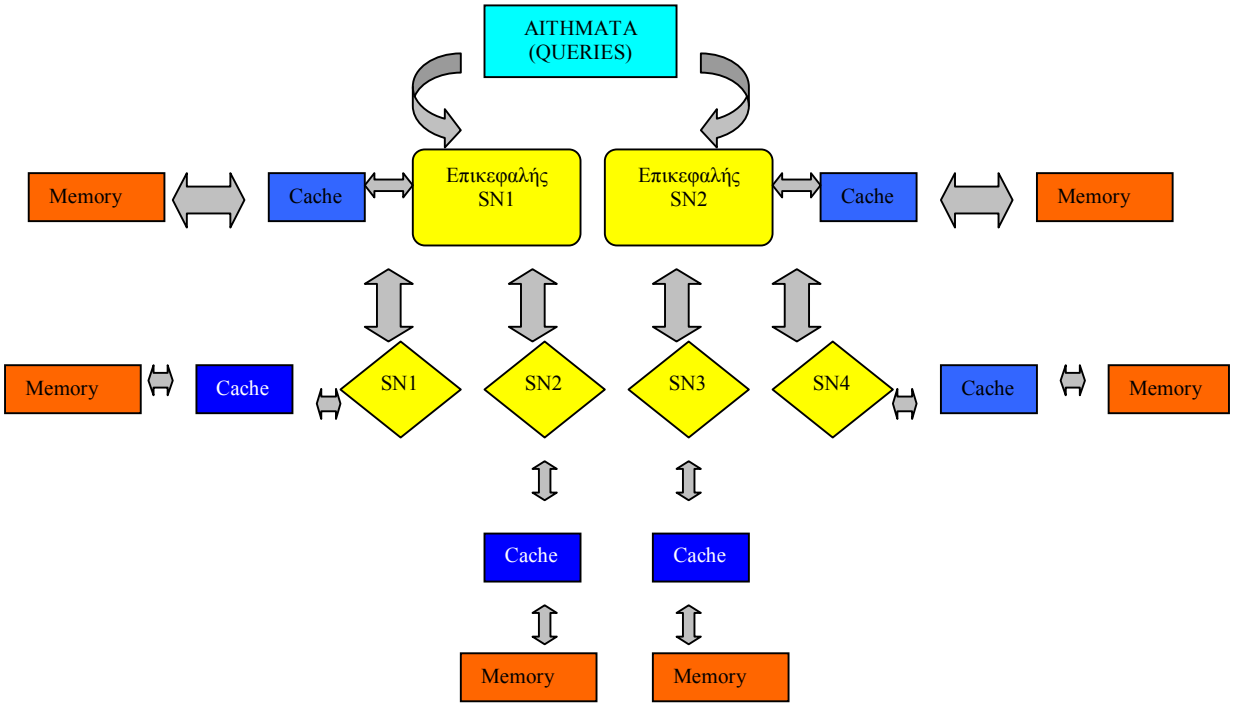
4.7.2 Αρχή της τοπικότητας της αναφοράς

Τέλος και προτού αναφερθούμε εκτενέστερα στη χρησιμοποίηση της κρυφής μνήμης σε WSNs αξίζει να κάνουμε μία σύντομη αναφορά στην αρχή της τοπικότητας της αναφοράς ή απλά αρχή της τοπικότητας, σύμφωνα με την οποία διαδοχικές αναφορές στη μνήμη γίνονται μέσα από ένα σύνολο γειτονικών διευθύνσεων. Η αρχή της τοπικότητας διακρίνεται σε δύο τύπους, την τοπικότητα του χρόνου και την τοπικότητα του χώρου.

4.7.3 Κρυφή μνήμη σε WSN

Στην υλοποίηση μας, η κρυφή μνήμη ενσωματώνεται τόσο στους επικεφαλείς κόμβους αισθητήρες των συστάδων αισθητήρων, όσο και στους απλούς κόμβους αισθητήρες. Όπως έχουμε ήδη αναφέρει, όταν καταφθάνει στο WSN ένα νέο αίτημα από κάποιο χρήστη τότε αυτό μεταβιβάζεται πρώτα προς επεξεργασία σε ένα κατάλληλο σύνολο επικεφαλών κόμβων αισθητήρων. Αν δεν υπήρχε η κρυφή μνήμη, τότε καθένας από αυτούς τους επικεφαλείς SNs θα έπρεπε να μεταβεί στην κύρια μνήμη αυτού για να εντοπίσει τα δεδομένα που αποτελούν απάντηση σε αυτό το αίτημα, προκειμένου να τα ανακτήσει από αυτήν και να τα επιστρέψει ως απάντηση στο εν λόγω αίτημα. Ενσωματώνοντας σε κάθε επικεφαλή SN μία κρυφή μνήμη, μπορούμε να επισπεύσουμε σημαντικά την παραπάνω διαδικασία.

Σε αυτήν την περίπτωση, ο επικεφαλής SN θα ανατρέξει πρώτα στην κρυφή μνήμη για να εντοπίσει και να ανακτήσει ταχύτερα τα ζητούμενα δεδομένα με υψηλή πιθανότητα επιτυχίας, λόγω και των όσων αναφέραμε προηγουμένως. Άλλωστε, ακόμη κι αν δεν βρεθούν τα ζητούμενα δεδομένα στην κρυφή μνήμη η καθυστέρηση που θα έχει εισαχθεί θα είναι ελάχιστη με δεδομένη την ταχεία προσπέλαση αυτής. Αν τα ζητούμενα δεδομένα δεν βρίσκονται αποθηκευμένα ούτε στην κύρια ούτε στην κρυφή μνήμη του επικεφαλή SN, τότε αυτός θα πρέπει να μεταβιβάσει το αίτημα αυτό απευθείας σε όλους τους SNs που ανήκουν στη συστάδα τους. Οι τελευταίοι είναι επίσης εφοδιασμένοι με κρυφή μνήμη, οπότε η ίδια διαδικασία αναζήτησης και ανάκτησης των αιτούμενων δεδομένων θα λάβει χώρα και σε αυτούς. Ενσωματώνοντας λοιπόν κρυφή μνήμη σε όλους τους SNs ενός WSN επιτυγχάνουμε την ταχύτερη επεξεργασία των αιτημάτων μέσω της ταχύτερης πρόσβασης κάθε SN στη μνήμη αυτού.



Σχήμα 4.7: Παρουσίαση χρήσης cache memory σε WSN

4.7.4 Επιλογή δεδομένων προς αποθήκευση στην κρυφή μνήμη

Εντούτοις, στην παραπάνω απλή διαδικασία εμπίπτουν ένα σύνολο ζητημάτων, τα οποία και χρήζουν περαιτέρω ανάλυσης. Πιο συγκεκριμένα, το πρώτο ζήτημα που θα μας απασχολήσει είναι η επιλογή των κατάλληλων δεδομένων που πρέπει να αποθηκευτούν στην κρυφή μνήμη ενός SN. Είναι προφανές, ότι λόγω του μεγάλου όγκου δεδομένων που μετρώνται και διακινούνται μέσω ενός SN σε ένα WSN, δεν είναι δυνατόν να αποθηκευτούν όλα τα δεδομένα στην κρυφή μνήμη αυτού. Επομένως, κάθε SN θα πρέπει με κάποιον τρόπο να επιλέγει τα δεδομένα εκείνα, τα οποία και θα αποθηκεύσει στην κρυφή του μνήμη. Υπάρχουν διάφορες τεχνικές για το σκοπό αυτό, μερικές από τις οποίες είναι και οι ακόλουθες:

1. Μία εγγραφή δεδομένων, η οποία μετριέται ή παραλαμβάνεται από κάποιον SN έχει μεγαλύτερη πιθανότητα να χρησιμοποιηθεί στο άμεσο μέλλον από ότι κάποια παλαιότερη εγγραφή. Έτσι, ο SN επιλέγει να αποθηκεύει στην κρυφή του μνήμη τις πιο πρόσφατες χρονικά εγγραφές, αντικαθιστώντας όπως θα δούμε παρακάτω τις παλαιότερες. Δηλαδή, στην προκειμένη περίπτωση, το κριτήριο της αποθήκευσης μιας εγγραφής στην κρυφή μνήμη είναι η χρονική στιγμή της άφιξής της.

2. Αν κατά τη λειτουργία ενός WSN ένας SN αυτού διαπιστώσει ότι κάποια συγκεκριμένα δεδομένα του συμμετέχουν σε ένα μεγάλο αριθμό αιτημάτων, τότε μπορεί εύκολα να συνειδητοποιήσει ότι αυτή η δραστηριότητα θα συνεχιστεί και στο μέλλον. Έτσι επιλέγει να αποθηκεύσει τα δεδομένα αυτά στην κρυφή του μνήμη, προκειμένου να ελαχιστοποιήσει το χρόνο ανάκτησής

τους. Δηλαδή, στην περίπτωση αυτή η επιλογή των δεδομένων που θα αποθηκευτούν στην κρυφή μνήμη γίνεται με βάση τη συχνότητα εμφάνισης τους στα αιτήματα των χρηστών και κατ' επέκταση, τη συχνότητα ανάκτησής τους από τη μνήμη.

3. Κάθε SN διαθέτει ένα μεγάλο πλήθος αισθητήρων για την καταγραφή των τιμών πολλών διαφορετικών παραμέτρων. Ενδεχομένως κάποιοι από τους αισθητήρες αυτούς να απαιτούν πολύ μεγαλύτερη κατανάλωση ενέργειας κατά τη μέτρηση της τιμής της αντίστοιχης παραμέτρου από ότι οι υπόλοιποι. Με δεδομένη λοιπόν, την περιορισμένη ενεργειακή αυτάρκεια των SNs, ενδεχομένως σε κάποιες περιπτώσεις να είναι βολικό να αποθηκεύουμε στην κρυφή μνήμη του SN τις μετρήσεις των αισθητήρων εκείνων που απαιτούν μεγάλη κατανάλωση ενέργειας. Η περίπτωση αυτή βρίσκει εφαρμογή ιδιαίτερα σε WSNs, στα οποία μας ενδιαφέρει η αναζήτηση όχι τόσο ιστορικών, αλλά κυρίως σύγχρονων μετρήσεων των SNs. Κριτήριο λοιπόν, σε αυτήν την περίπτωση αποτελεί η ενέργεια που καταναλώνεται από κάθε αισθητήρα ενός SN κατά τη μέτρηση της τιμής της αντίστοιχης παραμέτρου.

4. Οι SNs σε ένα WSN επικοινωνούν μεταξύ τους μέσω ασύρματων ζεύξεων με χρήση ειδικών πομποδεκτών και διαφόρων πρωτοκόλλων δρομολόγησης. Αν ένας SN διαπιστώσει ότι η ασύρματη ζεύξη του με κάποιον άλλο SN είτε λόγω περιορισμένου εύρους εισάγει μεγάλες χρονικές καθυστερήσεις είτε απαιτεί μεγάλη κατανάλωση ενέργειας, τότε μπορεί να επιλέξει να αποθηκεύει τα δεδομένα πού λαμβάνει από αυτόν τον SN στην κρυφή του μνήμη. Για παράδειγμα, ένας επικεφαλής SN μπορεί να ελέγχει περιοδικά την ποιότητα των ασύρματων ζεύξεων με όλους τους SN της συστάδας του και να αποθηκεύει στην κρυφή του μνήμη, τα δεδομένα που προέρχονται από SNs με χαμηλό εύρος μετάδοσης. Ένα λοιπόν ακόμη κριτήριο αποθήκευσης στην κρυφή μνήμη είναι η ποιότητα των ασύρματων ζεύξεων μεταξύ των SNs και το ενεργειακό κόστος που καθεμία από αυτές εισάγει.

5. Η επιλογή των δεδομένων που εισάγονται στην κρυφή μνήμη μπορεί να γίνει και με έναν προκαθορισμένο τρόπο από τους σχεδιαστές ή τους χρήστες του WSN. Πιο συγκεκριμένα, έχουμε τη δυνατότητα να αντιστοιχίσουμε σε κάθε αισθητήρα ενός SN μία τιμή προτεραιότητας. Έτσι, κατά την εγγραφή στην κύρια μνήμη της τιμής της παραμέτρου, την οποία και λαμβάνει αυτός ο αισθητήρας, θα μπορεί να γίνει σύγκριση της προτεραιότητας αυτής της παραμέτρου με τις παραμέτρους που βρίσκονται ήδη αποθηκευμένες στην κρυφή μνήμη κι αναλόγως να αποφασιστεί αν θα δημιουργηθεί αντίγραφο της τιμής της παραμέτρου στην κρυφή μνήμη.

6. Τέλος, η επιλογή των δεδομένων που είναι αποθηκευμένα στην κύρια μνήμη και των οποίων αντίγραφα διατηρούνται στην κρυφή μνήμη, μπορεί να γίνει και με τυχαίο τελείως τρόπο, εάν κάτι τέτοιο είναι φυσικά επιθυμητό.

Είναι προφανές, ότι η επιλογή του τρόπου με τον οποίο θα καθορίζονται τα δεδομένα που βρίσκονται αποθηκευμένα στην κύρια μνήμη κάποιου SN και των οποίων αντίγραφα διατηρούνται στην κρυφή μνήμη αυτού, αποτελεί έναν κρίσιμο παράγοντα για την ταχύτερη δυνατή απόκριση στα αιτήματα των χρηστών ενός WSN και μπορεί να υλοποιηθεί με πολλές διαφορετικές τεχνικές, οι οποίες και

παρουσιάστηκαν προηγουμένως. Στην πράξη, σε ένα WSN μπορεί να εφαρμοστεί οποιαδήποτε από τις παραπάνω τεχνικές ή οποιοσδήποτε δυνατός συνδυασμός αυτών, ανάλογα με τις ανάγκες και τις ιδιαιτερότητες του κάθε SN και της εφαρμογής που εξυπηρετεί το WSN γενικότερα. Τέλος, μπορούμε με βάση το ποσοστό των επιτυχών ανακτήσεων από την κρυφή μνήμη να αλλάξουμε τη χρησιμοποιούμενη τεχνική εισαγωγής δεδομένων σε αυτήν για να πετύχουμε τη βέλτιστη αποδοτικότητα.

4.7.5 Πολιτικές αντικατάστασης δεδομένων από την κρυφή μνήμη

Το δεύτερο ζήτημα που θα μελετήσουμε αναφορικά με την κρυφή μνήμη είναι η εφαρμοζόμενη πολιτική «εξώσεως» από αυτήν. Το ζήτημα αυτό έγκειται στο ποια εγγραφή θα επιλέγεται προς «έξωση» από την κρυφή μνήμη, δηλαδή ποια εγγραφή θα διαγράφεται από την κρυφή μνήμη, όταν αυτή είναι και γεμάτη και πρέπει να εισαχθεί κάποια καινούργια εγγραφή σε αυτήν. Υπάρχουν διάφορες πολιτικές εξώσεως, οι βασικότερες από τις οποίες είναι οι κάτωθι:

- Ελάχιστης Τελευταίας Χρήσεως (last recently used – LRU)

  Αφαιρεί από την κρυφή μνήμη που χρησιμοποιήθηκε παλαιότερα από όλες τις υπόλοιπες, που δηλαδή η τελευταία της ανάκτηση είναι η παλαιότερη από όλες. Ενδείκνυται, στην περίπτωση που η τεχνική εισαγωγής των δεδομένων στη κρυφή μνήμη γίνεται με το κριτήριο της χρονικής άφιξής τους σε αυτήν. Για την υλοποίησή της απαιτείται η αντιστοίχιση σε κάθε εγγραφή της κρυφής μνήμης μίας χρονοσφραγίδας, η οποία και θα αντιστοιχεί στην τελευταία χρονική στιγμή, κατά την οποία και ανακτήθηκε η εν λόγω εγγραφή.

- Ελάχιστης συχνότητας χρήσεως(least frequently used – LFU)

  Επιτάσσει την έξωση από την κρυφή μνήμη της σελίδας με τη μικρότερη συχνότητα ανάκτησης. Ενδείκνυται, όταν η επιλογή της εισαγωγής των δεδομένων στην κρυφή μνήμη, γίνεται με βάση τη συχνότητα εμφάνισής τους στα αιτήματα των χρηστών. Για την υλοποίηση αυτής της τεχνικής αρκεί να αντιστοιχηθεί σε κάθε εγγραφή ένας μετρητής και κάθε φορά που ανακτάται αυτή η εγγραφή από τη μνήμη να αυξάνεται και η τιμή του μετρητή κατά μία μονάδα. Τα δεδομένα με τη μεγαλύτερη τιμή στον εν λόγω μετρητή θα είναι αυτά, που θα παρουσιάζουν το μεγαλύτερο ρυθμό ανάκτησης.

- Μεγαλύτερου Χρόνου Παραμονής(first in first out – FiFo)

  Αφαιρείται από την κρυφή μνήμη η εγγραφή που έχει παραμείνει σε αυτήν το μεγαλύτερο χρονικό διάστημα. Ενδείκνυται, στην περίπτωση που η τεχνική εισαγωγής των δεδομένων στη κρυφή μνήμη γίνεται με το κριτήριο της χρονικής άφιξής τους σε αυτήν. Για την υλοποίησή της απαιτείται η αντιστοίχιση σε κάθε εγγραφή της κρυφής μνήμης μίας χρονοσφραγίδας, η οποία και θα αντιστοιχεί στην χρονική στιγμή, κατά την οποία και εισήχθη στην κρυφή μνήμη η εν λόγω εγγραφή.

- Τελευταίας Εισαγωγής(last in first out – LiFo)

  Αφαιρείται από την κρυφή μνήμη, η εγγραφή που έχει εισαχθεί τελευταία σε αυτήν. Για την υλοποίησή της απαιτείται και πάλι η αντιστοίχιση σε κάθε εγγραφή της κρυφής μνήμης μίας χρονοσφραγίδας, η οποία και θα αντιστοιχεί

στην τελευταία χρονική στιγμή, κατά την οποία και ανακτήθηκε η εν λόγω εγγραφή.

- Τυχαίας Αντικατάστασης(random selection – RS)
  Επιλέγει με εντελώς τυχαίο τρόπο μία εγγραφή για έξωση. Προφανώς και χρησιμοποιείται, όταν και η είσοδος των εγγραφών στην κρυφή μνήμη γίνεται επίσης με τυχαίο τρόπο.

Καθίσταται σαφές λοιπόν, ότι απαιτείται η χρήση κάποιας εκ των ανωτέρω πολιτικών εξώσεων από την κρυφή μνήμη σε συνδυασμό πάντα με την αντίστοιχη κατάλληλη τεχνική εισόδου εγγραφών στην κρυφή μνήμη, προκειμένου να βελτιστοποιείται η εξυπηρέτηση των αιτημάτων σε ένα WSN. Στη συνέχεια θα εξετάσουμε ένα παράδειγμα, προκειμένου να αποσαφηνιστούν τόσο η διαδικασία της εισαγωγής δεδομένων στην κρυφή μνήμη, όσο και η διαδικασία της αντικατάστασης δεδομένων από την κρυφή μνήμη.

4.7.6 Παράδειγμα 4.3 – Κρυφή μνήμη

Ας θεωρήσουμε ένα WSN αποτελούμενο από ένα μεγάλο πλήθος SNs, καθένας από τους οποίους διαθέτει ενσωματωμένο έναν αισθητήρα μέτρησης της θερμοκρασίας. Οι SNs αυτού του WSN είναι οργανωμένοι σε συστάδες των τριών SNs, ενώ σε κάθε συστάδα αντιστοιχίζεται επιπλέον κι ένας head SN. Ο αισθητήρας κάθε SN λαμβάνει την τιμή της θερμοκρασίας κάθε πέντε λεπτά κι εν συνεχεία, η τιμή αυτή αποθηκεύεται στην ενσωματωμένη μνήμη του SN. Ταυτόχρονα, ο SN αποστέλλει αυτή την τιμή της θερμοκρασίας στον head SN της συστάδας, στην οποία ανήκει. Τόσο οι SNs, όσο και οι head SNs διαθέτουν κάποια κρυφή μνήμη.

Εμείς στο παράδειγμά μας, θα εξετάσουμε το περιεχόμενο της κρυφής μνήμης του head SN μίας συστάδας. Για το σκοπό αυτό, θεωρούμε ότι η τεχνική εισαγωγής των δεδομένων στην κρυφή μνήμη του εν λόγω SN είναι η πρώτη από τις τεχνικές εισαγωγής που περιγράψαμε νωρίτερα, δηλαδή κάθε εγγραφή δεδομένων που παραλαμβάνεται από τον head SN αποθηκεύεται άμεσα στην κρυφή μνήμη αυτού. Επιπλέον, θεωρούμε ότι χρησιμοποιούνται δύο πολιτικές αντικατάστασης των δεδομένων της κρυφής μνήμης και πιο συγκεκριμένα, η πολιτική ελάχιστης συχνότητας χρήσεως (LFU) και η πολιτική μεγαλύτερου χρόνου παραμονής (FiFo). Για την εφαρμογή της LFU, αντιστοιχίζεται σε κάθε εγγραφή της κρυφής μνήμης, ένας μετρητής που ονομάζεται δείκτης χρησιμοποίησης. Πιο συγκεκριμένα, κάθε φορά που αποθηκεύεται μία νέα εγγραφή στην κρυφή μνήμη, ο δείκτης χρησιμοποίησης αυτής τίθεται ίσος με 0. Στη συνέχεια κι όποτε υπάρχει ένα αίτημα από τη Sink για αυτή την εγγραφή της κρυφής μνήμης, ο δείκτης χρησιμοποίησής αυτής αυξάνεται κατά1. Ως αποτέλεσμα, όσο μεγαλύτερος είναι ο δείκτης χρησιμοποίησης μίας εγγραφής της κρυφής μνήμης, τόσο μεγαλύτερη είναι και η συχνότητα εμφάνισης της εγγραφής αυτής στα αιτήματα των χρηστών. Για την εφαρμογή της FiFo, αντιστοιχίζεται σε κάθε εγγραφή της κρυφής μνήμης, η χρονική στιγμή της αποθήκευσης αυτής στην κρυφή μνήμη. Με τον τρόπο αυτό κι εξετάζοντας τις χρονικές στιγμές άφιξης των εγγραφών στην κρυφή μνήμη, μπορούμε να εντοπίσουμε ποια από αυτές έχει το μεγαλύτερο χρόνο παραμονής στην κρυφή μνήμη. Μεγαλύτερη προτεραιότητα από αυτές τις δύο πολιτικές έχει η LFU, γεγονός

που σημαίνει ότι στο παράδειγμά μας η επιλογή των προς αντικατάσταση δεδομένων της κρυφής μνήμης, γίνεται ως εξής:

- Εξετάζονται οι δείκτες χρησιμοποίησης των εγγραφών της κρυφής μνήμης κι επιλέγεται προς αντικατάσταση η εγγραφή με τη μικρότερη τιμή στο δείκτη χρησιμοποίησης, δηλαδή η εγγραφή με τη μικρότερη συχνότητα εμφάνισης στα αιτήματα των χρηστών.

- Αν υπάρχουν δύο ή περισσότερες εγγραφές με την ίδια μικρότερη τιμή στο δείκτη χρησιμοποίησης, τότε επιλέγεται προς αντικατάσταση από αυτές, όποια διαθέτει το μεγαλύτερο χρόνο παραμονής στην κρυφή μνήμη, δηλαδή επιλέγεται προς αντικατάσταση η εγγραφή της οποίας η χρονική στιγμή άφιξης στην κρυφή μνήμη είναι προγενέστερη των υπολοίπων.

Συνεπώς, πρώτα εφαρμόζεται η πολιτική αντικατάστασης LFU κι εάν η πολιτική αυτή μας οδηγεί σε συγκρούσεις, τότε εφαρμόζεται και η πολιτική αντικατάστασης FiFo, για την επίλυση των συγκρούσεων αυτών. Επανερχόμενοι στο παράδειγμά μας, ας θεωρήσουμε ότι οι μετρήσεις που παραλαμβάνει ο head SN, του οποίου την κρυφή μνήμη εξετάζουμε, από τους SNs της συστάδας του είναι αυτές που φαίνονται στον παρακάτω πίνακα:

Πίνακας 4.2: Μετρήσεις που παραλαμβάνει ο head SN από τους SNs της συστάδας του

Βαθμοί Κελσίου	$1^η$ μέτρηση (t0=0’)	$2^η$ μέτρηση (t1=5’)	$3^η$ μέτρηση (t2=10’)	$4^η$ μέτρηση (t3=15’)
SN1	20	22	23	24
SN2	25	23	23	21
SN3	15	16	17	19

(t1=5’)

Κελί 1	Κελί 2	Κελί 3	Κελί 4	Κελί 5	Κελί 6
SN1 20	SN2 25	SN3 15	SN1 22	SN2 23	SN3 16
0’ 0	0’ 0	0’ 0	5’ 0	5’ 0	5’ 0

Το περιεχόμενο της κρυφής μνήμης, μετά από τις 2 πρώτες μετρήσεις (0’ και 5’). Ο head SN δεν έχει αποστείλει κανένα δεδομένο στη Sink κι έτσι όλοι οι δείκτες ισούνται με 0.

(t1=5’)

Κελί 1	Κελί 2	Κελί 3	Κελί 4	Κελί 5	Κελί 6
SN1 20	SN2 25	SN3 15	SN1 22	SN2 23	SN3 16
0’ 0	0’ 0	0’ 0	5’ 1	5’ 1	5’ 0

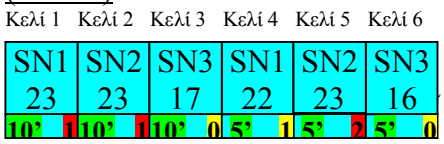
Ο head SN δέχεται αίτημα από τη Sink, για τα δεδομένα των μετρήσεων των SN1 και SN2, στα 5’ κι αποστέλλει τα δεδομένα αυτά στη Sink. Οι δείκτες των αντίστοιχων κελιών στην κρυφή μνήμη αυξάνονται κατά 1. Άρα, έχουμε 2 hits.

(t2=10’)

Κελί 1	Κελί 2	Κελί 3	Κελί 4	Κελί 5	Κελί 6
SN1 23	SN2 23	SN3 17	SN1 22	SN2 23	SN3 16
0’ 0	0’ 0	0’ 0	5’ 1	5’ 1	5’ 0

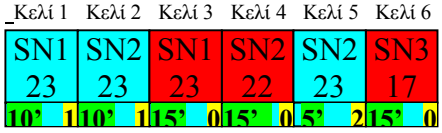
Ο head SN παραλαμβάνει τις μετρήσεις των SN1, SN2 και SN3, στα 10’. Αντικαθίσταται ένα εκ των 1,2,3 και 6. Το 6 είναι μεταγενέστερο των 1,2,3 κι άρα αντικαθίστανται τα 1,2 και 3.

(t2=10')



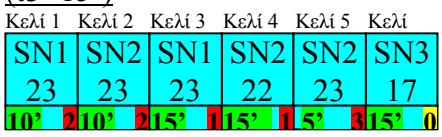
Ο head SN δέχεται αίτημα από τη Sink, για τα δεδομένα των μετρήσεων των αισθητήρων που έχουν τιμή ίση με 23, στα 5' και στα 10' κι αποστέλλει τα δεδομένα αυτά στη Sink. Οι δείκτες των αντίστοιχων κελιών αυξάνονται κατά 1. Άρα, έχουμε 3 hits.

(t3=15')



Παραλαμβάνονται μετρήσεις των SNs, στα 15'. Θα αντικατασταθούν τα κελιά 3 και 6, που έχουν τους μικρότερους δείκτες (0), καθώς επίσης κι ένα εκ των 1,2 και 4, που έχουν τον αμέσως μικρότερο δείκτη (1). Από τα τελευταία επιλέγεται το προγενέστερο χρονικά κελί 4.

(t3=15')



Έρχεται αίτημα από τη Sink, για τα δεδομένα όλων των μέχρι τώρα μετρήσεων των αισθητήρων που έχουν τιμή>20 κι αποστέλλει τα δεδομένα αυτά στη Sink.Οι δείκτες των αντίστοιχων κελιών αυξάνονται κατά 1. Για όσα δεδομένα δεν γνωρίζει την τιμή τους, επειδή δεν υπάρχουν στην Cache στέλνει το αίτημα στους αντίστοιχους SNs. Άρα, έχουμε 5 hits κι όπως φαίνεται από τον πίνακα, 3 miss.

Στο παράδειγμα, που εξετάσαμε είχαμε 10 περιπτώσεις επιτυχίας (hits) και 3 περιπτώσεις αποτυχίας (miss), κατά την ανάκτηση των αιτούμενων δεδομένων από την κρυφή μνήμη. Δηλαδή, το ποσοστό επιτυχίας της κρυφής μνήμης ισούται με 10/13= 76,9 % και είναι εξαιρετικά ικανοποιητικό, μιας και ανακτήσαμε τα περισσότερα δεδομένα τάχιστα από την κρυφή μνήμη, χωρίς να χρειαστεί να προσπελάσουμε την πιο αργή κύρια μνήμη του SN, παρά μόνο τρεις φορές.

4.8 Δεδομενοκεντρική βελτιστοποίηση δρομολόγησης

Όπως είδαμε στην ενότητα, ο πιο αποτελεσματικός κι αποδοτικός τρόπος εξυπηρέτησης των αιτημάτων σε ένα WSN είναι μέσω της υλοποίησης της εξυπηρέτησης σε δύο επίπεδα και για την ακρίβεια, σε ένα πρώτο προσεγγιστικό επίπεδο από τη Sink και σε ένα δεύτερο πιο ακριβές επίπεδο από τους SNs. Μέσω της κατανεμημένης εξυπηρέτηση αιτημάτων, οι SNs δεν απαιτείται να αποστέλλουν τα δεδομένα των μετρήσεών τους συνεχώς κι ανά τακτά χρονικά διαστήματα στη Sink, αλλά όποτε κρίνουν πως πρέπει να κάνουν κάτι τέτοιο. Το γεγονός αυτό έχει ως αποτέλεσμα να μειώνεται σημαντικά το πλήθος των μηνυμάτων ενημέρωσης που αποστέλλονται από τους SNs στη Sink, βελτιώνοντας με τον τρόπο αυτό την αποδοτικότητα του WSN. Στην παρούσα ενότητα, θα εξειδικεύσουμε τη μελέτη μας στον τρόπο, με τον οποίο γίνεται η δρομολόγηση των δεδομένων που αποστέλλονται από τους SNs στη Sink σε ένα WSN με κατανεμημένη εξυπηρέτηση αιτημάτων,

έχοντας ως στόχο μας την περαιτέρω μείωση του πλήθους των αποστελλόμενων μηνυμάτων ενημέρωσης από τους SNs στη Sink.

Καταρχήν, θα πρέπει να διευκρινίσουμε την έννοια της δρομολόγησης, η οποία και ουσιαστικά σημαίνει καθορισμός δρομολογίου, δηλαδή καθορισμός διαδρομής. Στη μέχρι στιγμής μελέτη μας, έχουμε θεωρήσει ότι όλοι οι SNs ενός WSN αποστέλλουν τα δεδομένα των μετρήσεών τους απευθείας είτε στη Sink είτε στον head SN της συστάδας τους, χωρίς δηλαδή να απαιτείται η μεσολάβηση κάποιου άλλου SN. Όμως, σε πολλές περιπτώσεις διάφοροι παράγοντες, όπως για παράδειγμα η ποιότητα της ασύρματης ζεύξης μεταξύ ενός SN και της Sink ή μεταξύ ενός SN και του head SN της συστάδας του, συνηγορούν στη δρομολόγηση των μετρήσεων ενός SN, μέσω κάποιου μονοπατιού ενός ή περισσοτέρων άλλων SNs. Σκοπός μας λοιπόν στην παρούσα ενότητα, είναι η εύρεση του βέλτιστου δέντρου δρομολόγησης για τα δεδομένα, που αποστέλλουν οι SNs ενός WSN είτε στη Sink είτε στον head SN της συστάδας, στην οποία ανήκουν.

Πριν όμως προχωρήσουμε στην εύρεση του βέλτιστου δέντρου δρομολόγησης, θα πρέπει πρώτα να αναφερθούμε σε κάποια άλλα σχετικά ζητήματα, τα οποία και είναι τα ακόλουθα:

1. Βάρος ασύρματης ζεύξης σε ένα WSN
2. Συνάθροιση δεδομένων σε έναν SN
3. Υπολογισμός του συνολικού κόστους ενημερώσεων σε ένα δέντρο δρομολόγησης

4.8.1 Βάρος ασύρματης ζεύξης σε ένα WSN

Σε ένα WSN δεν είναι υποχρεωτικό όλοι οι SNs αυτού να διαθέτουν κάποια απευθείας ασύρματη ζεύξη με τη Sink ή ακόμη και τον επικεφαλή της συστάδας, στην οποία ανήκουν. Επιπλέον, όλες οι ασύρματες ζεύξεις δεν έχουν την ίδια ποιότητα μεταξύ τους, καθώς το εύρος ζώνης μετάδοσης των δεδομένων μπορεί να διαφέρει μεταξύ τους. Πρέπει λοιπόν με κάποιον τρόπο να καθορίσουμε την ύπαρξη και την ποιότητα μίας ασύρματης ζεύξης μεταξύ δύο κόμβων ενός WSN. Για το σκοπό αυτό αντιστοιχίζουμε σε όλα τα ζεύγη μεταξύ της Sink κι ενός SN, όπως επίσης και σε όλα ζεύγη μεταξύ δύο SNs ενός WSN, μία μετρήσιμη παράμετρο που ονομάζεται βάρος του ζεύγους και συμβολίζεται με w. Όταν λοιπόν, το βάρος μεταξύ 2 SNs ή μεταξύ ενός SN και της Sink είναι ίσο με 0, τότε δεν υπάρχει ασύρματη ζεύξη μεταξύ τους. Ειδάλλως, εάν είναι διαφορετικό του μηδενός, τότε θα υπάρχει ασύρματη ζεύξη και μάλιστα, όσο μεγαλύτερη είναι η τιμή του βάρους της, τόσο καλύτερη είναι η ποιότητα αυτής. Στο δέντρο δρομολόγησης, όταν το βάρος μίας ακμής αυτού ισούται με 0, τότε πολύ απλά η ακμή αυτή δεν υφίσταται, ενώ σε διαφορετική περίπτωση το βάρος μίας ακμής σημειώνεται πάνω σε αυτήν.

Στα (β) και (δ) δεν υπάρχει ασύρματη ζεύξη. Αντίθετα στα (α) και (δ), υπάρχουν ασύρματες ζεύξεις βάρους 2 και 3 αντίστοιχα, γεγονός που συνεπάγεται μάλιστα, ότι η ποιότητα της ασύρματης ζεύξης στο (γ) είναι καλύτερη αυτής στο (α).

4.8.2 Συνάθροιση δεδομένων σε έναν SN

Μέχρι στιγμής, θεωρούσαμε ότι κάθε SN έστελνε τα δεδομένα των μετρήσεων του απευθείας είτε στη Sink είτε στον head SN της συστάδας του, όταν χρειαζόταν βέβαια να συμβεί κάτι τέτοιο και τα δεδομένα αυτά αφορούσαν μόνο τις δικές του μετρήσεις. Πλέον, αν ένας SN αποστέλλει στη Sink πέρα από τα δεδομένα των δικών του μετρήσεων και τα δεδομένα των μετρήσεων ενός ή περισσοτέρων άλλων SNs, τότε θα πρέπει με κάποιο τρόπο να γίνεται συνάθροιση όλων αυτών των δεδομένων στον SN. Ο τρόπος, με τον οποίο λαμβάνει χώρα σε έναν SN, η ανωτέρω συνάθροιση δεδομένων έχει ως εξής:

1. Η έξοδος ενός SN, θα ισούται με:

- το άθροισμα προγενέστερων μεταβολών της τιμής αυτού που δεν μεταδόθηκαν λόγω του γεγονότος ότι περιορίστηκαν από το εγκατεστημένο στον SN φίλτρο
- τη διαφορά της τιμής της πλέον πρόσφατης μέτρησής του από την προηγούμενη μέτρησή του
- το άθροισμα των τιμών όλων των εισόδων αυτού.

2. Εν συνεχεία κι αφού έχουμε υπολογίσει την έξοδο του SN:

- Αν η έξοδος του SN παραβιάζει το εγκατεστημένο σε αυτόν φράγμα, τότε αυτή μεταδίδεται είτε στη Sink είτε σε έναν άλλο SN. Παράλληλα, διαγράφονται από τη μνήμη του SN όλες οι έξοδοι αυτού που είχαν αποθηκευτεί σε προγενέστερες χρονικές στιγμές.
- Ειδάλλως, δεν μεταδίδεται και διατηρείται από τον SN σε κάποια κατάλληλη θέση της μνήμης αυτού.

4.8.3 Υπολογισμός του συνολικού κόστους των ενημερώσεων σε ένα δέντρο δρομολόγησης

Καταρχήν, θα πρέπει να διευκρινίσουμε τι ακριβώς αποτελεί το συνολικό κόστος των ενημερώσεων σε ένα δέντρο δρομολόγησης. Είναι στην ουσία μία παράμετρος, η οποία μας δείχνει τον όγκο των ενημερώσεων που αποστέλλουν οι SNs ενός WSN στη Sink αυτού, όταν χρησιμοποιείται η δρομολόγηση που καθορίζεται από το συγκεκριμένο δέντρο. Δηλαδή, όσο μεγαλύτερες είναι οι μεταβολές των τιμών στις μετρήσεις των δεδομένων που λαμβάνουν οι αισθητήρες των SNs σε ένα WSN, τόσο περισσότερα και ογκωδέστερα θα είναι τα μηνύματα ενημέρωσης που αποστέλλονται σε αυτό το WSN και τόσο μεγαλύτερο θα είναι το κόστος των ενημερώσεων σε αυτό το WSN. Ο υπολογισμός του συνολικού κόστους γίνεται, ως εξής:

1. Υπολογίζουμε αρχικά το κόστος των ενημερώσεων μεταξύ της δεύτερης και της πρώτης χρονικής στιγμής, δηλαδή ασχολούμαστε αρχικά με τις μεταβολές στις τιμές των μετρήσεων των αισθητήρων μεταξύ της δεύτερης και της πρώτης χρονικής στιγμής.

2. Ξεκινάμε λοιπόν, από το κατώτερο επίπεδο κόμβων του δέντρου, δηλαδή από το επίπεδο των φύλλων αυτού. Παίρνουμε ένα προς ένα και με τη σειρά τους κόμβους του επιπέδου αυτού και για καθένα από αυτά υπολογίζουμε την έξοδό του, σύμφωνα και με τα όσα αναφέραμε παραπάνω.

- Αν η έξοδος αυτή, παραβιάζει το φίλτρο που έχει εγκατασταθεί σε αυτό το φύλλο του δέντρου, τότε την πολλαπλασιάζουμε με το βάρος της ακμής που συνδέει αυτό το φύλλο του δέντρου με τον κόμβο του δέντρου, που είναι πατέρας αυτού του φύλλου του δέντρου κι αποστέλλουμε το αποτέλεσμα στον τελευταίο.

- Ειδάλλως, δεν αποστέλλουμε την έξοδο του δέντρου στον κόμβο πατέρα του, αλλά την αποθηκεύουμε σε κατάλληλη θέση στη μνήμη αυτού

3. Επαναλαμβάνουμε την παραπάνω διαδικασία, δηλαδή το βήμα (2) και για όλους τους υπόλοιπους κόμβους του τρέχοντος επιπέδου.

4. Εν συνεχεία, επαναλαμβάνουμε τα παραπάνω, δηλαδή τα βήματα (1), (2) και (3) και για όλους τους υπόλοιπα επίπεδα κόμβων του δέντρου, μέχρι και τη Sink.

5. Τέλος, επαναλαμβάνουμε το σύνολο των προηγούμενων βημάτων και για τις επόμενες χρονικές στιγμές, ασχολούμενοι κάθε φορά με τις ενημερώσεις που λαμβάνουν χώρα μεταξύ της τρέχουσας και της αμέσως προηγούμενης χρονικής στιγμής.

6. Αθροίζουμε όλα τα κόστη που έχουμε υπολογίσει και το αποτέλεσμα θα ισούται με το συνολικό κόστος των ενημερώσεων στο συγκεκριμένο δέντρο δρομολόγησης.

Αφού λοιπόν έχουμε εξηγήσει αναλυτικά όλα τα ανωτέρω ζητήματα, είμαστε έτοιμοι να παρουσιάσουμε στη συνέχεια τον αλγόριθμο εύρεσης του βέλτιστου δέντρου δρομολόγησης των δεδομένων σε ένα WSN από τους SNs στη Sink αυτού.

4.8.4 Αλγόριθμος εύρεσης του βέλτιστου δέντρου δρομολόγησης των δεδομένων σε ένα WSN από τους SNs στη Sink

Οι είσοδοι του αλγορίθμου αυτού είναι οι ακόλουθες:

- ένα αρχικά κενό σύνολο SNs που συμβολίζεται με S και το οποίο ονομάζεται υποψήφιο σύνολο
- η Sink, οι SNs, καθώς το εγκατεστημένο σε κάθε SN φίλτρο
- όλες οι δυνατές ασύρματες ζεύξεις μεταξύ των κόμβων του WSN, καθώς και το κόστος καθεμίας από αυτές.

Έξοδος του είναι το βέλτιστο δέντρο δρομολόγησης. Η εκτέλεση του αλγορίθμου αρχίζει από τη Sink, δηλαδή αρχικά θεωρούμε ότι το δέντρο μας αποτελείται μόνο από τη Sink. Σε κάθε βήμα του αλγορίθμου και μέχρι τον τερματισμό αυτού επαναλαμβάνουμε τις παρακάτω ενέργειες:

1. Παίρνουμε το τρέχον μη ολοκληρωμένο δέντρο δρομολόγησης, το οποίο όπως είπαμε αρχικά αποτελείται μόνο από τη Sink.

2. Βρίσκουμε όλους τους SNs, οι οποίοι δεν ανήκουν ήδη στο σύνολο S και οι οποίοι διαθέτουν ασύρματη ζεύξη επικοινωνίας με το τρέχων μη ολοκληρωμένο δέντρο δρομολόγησης και τους εισάγουμε στο σύνολο S. Δηλαδή, βρίσκουμε όλους τους SNs, οι οποίοι μπορούν να συνδεθούν ασύρματα με τουλάχιστον έναν από τους SNs τρέχοντος δέντρου δρομολόγησης και τους εισάγουμε στο σύνολο S.

3. Παίρνουμε τον πρώτο SN του συνόλου S και τον συνδέουμε με έναν από τους SNs του υπάρχοντος δέντρου, με τον οποίο και διαθέτει ασύρματο δίαυλο μετάδοσης δεδομένων. Έτσι προκύπτει ένα νέο δέντρο δρομολόγησης.

4. Υπολογίζουμε για αυτό το νέο δέντρο δρομολόγησης τα κόστη όλων των δυνατών μονοπατιών, που οδηγούν από τα φύλλα αυτού στη Sink του, χρησιμοποιώντας για το σκοπό αυτό τον αλγόριθμο υπολογισμού της εξόδου ενός SN και τον αλγόριθμο υπολογισμό του κόστους ενός μονοπατιού από τη Sink στον SN.

5. Επαναλαμβάνουμε τα βήματα (3) και (4) και για καθέναν από τους υπόλοιπους SNs του τρέχοντος SN, με τους οποίους και διαθέτει ασύρματη ζεύξη ο πρώτος SN του συνόλου S, δηλαδή και για όλα τα υπόλοιπα πιθανά δέντρα δρομολόγησης, που μπορεί να προκύψουν από την προσθήκη αυτού του SN στο τρέχον δέντρο δρομολόγησης, υπολογίζοντας και για καθένα από αυτά το κόστος του.

6. Επαναλαμβάνουμε τα βήματα (3), (4) και (5) του αλγορίθμου και για όλους τους υπόλοιπους SNs του συνόλου S.

7. Από όλα τα δέντρα δρομολόγησης, που προέκυψαν από το τρέχον δέντρο δρομολόγησης, επιλέγουμε αυτό με το μικρότερο συνολικό κόστος μονοπατιών. Αυτό θα είναι και το νέο τρέχον δέντρο δρομολόγησης.

8. Παίρνουμε τον SN από την προσθήκη του οποίου προέκυψε το νέο τρέχον μη ολοκληρωμένο δέντρο δρομολόγησης και το αφαιρούμε από το σύνολο S.

9. Αν το σύνολο S είναι κενό, τότε τερματίζουμε τον αλγόριθμο παρέχοντας στην έξοδό του το τρέχων δέντρο δρομολόγησης, που είναι και το βέλτιστο δέντρο δρομολόγησης. Ειδάλλως, επαναλαμβάνουμε τον αλγόριθμο για το τρέχον δέντρο δρομολόγησης.

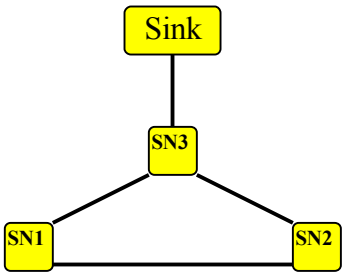
4.8.5 Παράδειγμα 4.3 – Εύρεση βέλτιστου δέντρου δρομολόγησης

Ας θεωρήσουμε ένα WSN, το οποίο βρίσκεται εγκατεστημένο σε ένα συγκεκριμένο ύψος μίας λεωφόρου τριών λωρίδων μονής κατεύθυνσης. Είναι επιφορτισμένο με την παρακολούθηση και καταμέτρηση του όγκου των οχημάτων που διέρχονται, τόσο από κάθε λωρίδα, όσο και συνολικά από το συγκεκριμένο

σημείο της λεωφόρου κάθε 60 δευτερόλεπτα. Για το σκοπό αυτό, το WSN αποτελείται από 3 SNs, καθένας από τους οποίους κι είναι εγκατεστημένος σε μία από τις τρεις λωρίδες της λεωφόρου. Κάθε SN διαθέτει έναν ενσωματωμένο οπτικό αισθητήρα για την καταγραφή του πλήθους των οχημάτων που διέρχονται από την αντίστοιχη λωρίδα της λεωφόρου στη διάρκεια 60 δευτερολέπτων. Επιπλέον διαθέτει και μνήμη flash τύπου NAND, για την αποθήκευση των μετρούμενων δεδομένων. Έτσι λοιπόν, κάθε αισθητήρας μετράει αρχικά το πλήθος των οχημάτων που διέρχονται από τη λωρίδα της λεωφόρου, στην οποία κι έχει εγκατασταθεί. Εν συνεχεία, αποστέλλει την τιμή της μέτρησής του στον SN, στον οποίο και βρίσκεται ενσωματωμένος, ούτως ώστε ο τελευταίος να την αποθηκεύσει στη μνήμη του κι έπειτα να την αποστείλει στη Sink.

Σε αυτό το σημείο θα πρέπει να αναφέρουμε, πως σε κάθε SN έχει εγκατασταθεί ένα φίλτρο [-1, +1]. Το συγκεκριμένο φίλτρο περιορίζει κατά τα γνωστά την αποστολή της πιο πρόσφατης μέτρησής του SN στη Sink, μόνο αν αυτή είναι είτε μεγαλύτερη είτε μικρότερη από την τιμή της αμέσως προηγηθείσας από αυτήν μέτρησης, κατά 1 μονάδα. Δηλαδή, αν η διαφορά της τιμής της πιο πρόσφατης μέτρησης από την τιμή της αμέσως προηγούμενης μέτρησης ανήκει στο διάστημα [-1, +1], τότε το φίλτρο θα αποτρέψει την ενημέρωση της Sink με τη νέα τιμή. Επιπρόσθετα, σε αυτό το WSN θεωρούμε ότι η επεξεργασία των αιτημάτων λαμβάνει χώρα σε δύο επίπεδα, σύμφωνα δηλαδή με το σχήμα που αναφέραμε στην ενότητα. Τέλος, όταν ένας SN παραλαμβάνει στην είσοδό του τη μεταβολή της τιμής ενός άλλου ή και περισσοτέρων SNs, τότε η έξοδός του θα ισούται με το άθροισμα όλων των εισόδων του συν τη μεταβολής της δικής του τιμής συν προηγούμενες μεταβολές τιμών αυτού του SN, που βρίσκονται αποθηκευμένες στη μνήμη αυτού.

Στη συνέχεια, θα εξετάσουμε το πλήθος των μηνυμάτων ενημέρωσης που αποστέλλονται στη Sink από τους SNs του παραπάνω WSN για 3 διαφορετικές περιπτώσεις δρομολόγησης των δεδομένων εντός αυτού. Σημειωτέον, ότι υπάρχει μία ασύρματη ζεύξη επικοινωνίας μεταξύ κάθε ζεύγους SNs (SN1-SN2, SN1-SN3, SN2-SN3), αλλά μόνο ο SN3 διαθέτει ασύρματο δίαυλο επικοινωνίας με τη Sink. Δηλαδή, ο γράφος συνδεσιμότητας του WSN είναι αυτός που ακολουθεί:



Σχήμα 4.8: Γράφημα συνδεσιμότητας των SNs του WSN

Έστω λοιπόν, ότι οι μετρήσεις των SNs στα τρία πρώτα λεπτά της λειτουργίας του WSN, είναι αυτές που φαίνονται στον πίνακα που ακολουθεί:

Διερχόμενα οχήματα από τη λωρίδα όπου βρίσκεται ο SNi στη διάρκεια του j λεπτού-min.

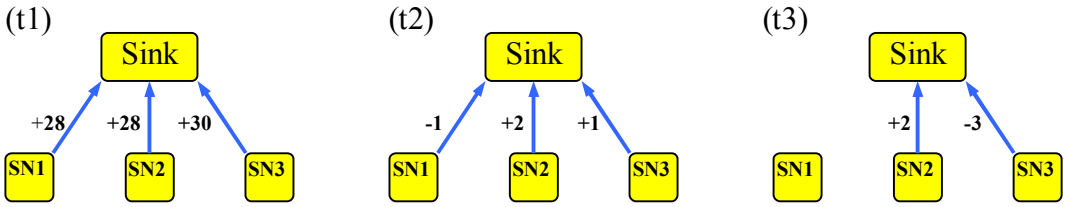
(SNi, j-min)	1-min t1	2-min t2	3-min t3
SN3	30	31	31
SN2	28	30	32
SN1	28	27	24

Πίνακας 4.4: Τιμές μετρήσεων SNs παραδείγματος 4.3

- 1η περίπτωση)

Καθένας από τους SNs μεταδίδει τις μετρήσεις του απευθείας στη Sink, χωρίς να απαιτείται η μεσολάβηση κάποιου άλλου SN για το σκοπό αυτό. Ως αποτέλεσμα, σε κάθε χρονική στιγμή όλοι οι SNs αποστέλλουν τα δεδομένα των μετρήσεών τους απευθείας στη Sink. Επομένως, θα σταλούν από τους SNs στη Sink 8 συνολικά ενημερώσεις, μιας και κάθε φορά που αλλάζει η τιμή κάποιου SN, αυτός θα ενημερώνει άμεσα τη Sink.

Σχήμα 4.8: Εύρεση βέλτιστου δέντρου δρομολόγησης



1η περίπτωση: Συνολικά, θα σταλούν 8 μηνύματα ενημέρωσης (μπλε βέλη) από τους SNs στη Sink.

Συνεπώς, όταν όλοι οι SNs αποστέλλουν τα δεδομένα των μετρήσεών τους απευθείας στη Sink, χωρίς να χρησιμοποιείται κάποια μορφή δρομολόγησης, τότε το πλήθος των μηνυμάτων ενημέρωσης που αποστέλλονται από τους SNs στη Sink ισούται με 8. Προκειμένου λοιπόν, να μειώσουμε το πλήθος των αποστελλόμενων μηνυμάτων ενημέρωσης, θα επιλέξουμε τυχαία στην επόμενη περίπτωση ένα πιθανό δέντρο δρομολόγησης αυτών.

- 2η περίπτωση)

Οι SN1 και SN2 αποστέλλουν τις ενημερώσεις των δεδομένων τους στη Sink μέσω του SN3, ο οποίος και συνδέεται απευθείας σε αυτήν. Αρχικά θεωρούμε, ότι η τιμή καθενός εκ των SNs στη Sink ισούται με 0, δηλαδή είναι:

SN1	0	SN2	0	SN3	0

(1-min, t1)

Αρχικά λοιπόν κατά τη χρονική στιγμή t1 , όλοι οι SNs θα αποστείλουν τα δεδομένα των μετρήσεών τους στη Sink. Πιο συγκεκριμένα, θα είναι:

SN1 → SN3 : Ο SN1 ενημερώνει τον SN3, ότι η τιμή του αυξήθηκε από 0 σε 28.

SN2 → SN3 : Ο SN2 ενημερώνει τον SN3, ότι η τιμή του αυξήθηκε από 0 σε 28

SN3 → Sink : Ο SN3 αθροίζει τις ενημερώσεις που έλαβε από τους SN1 (+28) και SN2 (+28) με τη μεταβολή της δικής του τιμής (+30) κι αποστέλλει το αποτέλεσμα, το οποίο και ισούται με +86 στη Sink.

Έτσι λοιπόν, κατά τη χρονική στιγμή t1 θα σταλούν 3 μηνύματα ενημέρωσης στη Sink και το περιεχόμενο της τελευταίας θα έχει ως εξής:

SN1	28	SN2	28	SN3	30

(2-min, t2)
SN1 → SN3 : Η τιμή του SN1 μειώθηκε κατά 1, με αποτέλεσμα το εγκατεστημένο σε αυτόν φίλτρο να αποτρέπει την αποστολή ενημέρωσης για τη νέα τιμή του SN1 στον SN3. Ως αποτέλεσμα, ο SN1 δεν θα στείλει κανένα μήνυμα ενημέρωσης στον SN3 και θα αποθηκεύσει την τρέχουσα μεταβολή της τιμής του στη μνήμη του.

SN2 → SN3 : Ο SN2 ενημερώνει τον SN3, ότι η τιμή του αυξήθηκε κατά 2.

SN3 → Sink : Ο SN3 αθροίζει την ενημέρωση που έλαβε από τον SN2 (+2) με τη μεταβολή της δικής του τιμής (+1) κι αποστέλλει το αποτέλεσμα, το οποίο και ισούται με +3 στη Sink.

Έτσι λοιπόν, κατά τη χρονική στιγμή t2 θα σταλούν 2 μηνύματα ενημέρωσης στη Sink και το περιεχόμενο της τελευταίας θα έχει ως εξής:

SN1	28	SN2	30	SN3	31

(3-min, t3)
SN1 → SN3 : Η τιμή του SN1 μειώθηκε κατά 3. Σε αυτήν την τρέχουσα μεταβολή της τιμής του SN1, προσθέτουμε και τη μεταβολή της τιμής αυτού, κατά την προηγούμενη χρονική στιγμή (t2), η οποία βρίσκεται αποθηκευμένη στη μνήμη αυτού και ισούται με –1. Ως αποτέλεσμα, η συνολική μεταβολή της τιμής του SN1, θα ισούται με –4. Έτσι, ο SN1 διαγράφει από τη μνήμη του όλες τις προηγούμενες μεταβολές της τιμής αυτού κι ενημερώνει τον SN3, ότι η τιμή του μειώθηκε κατά 4.

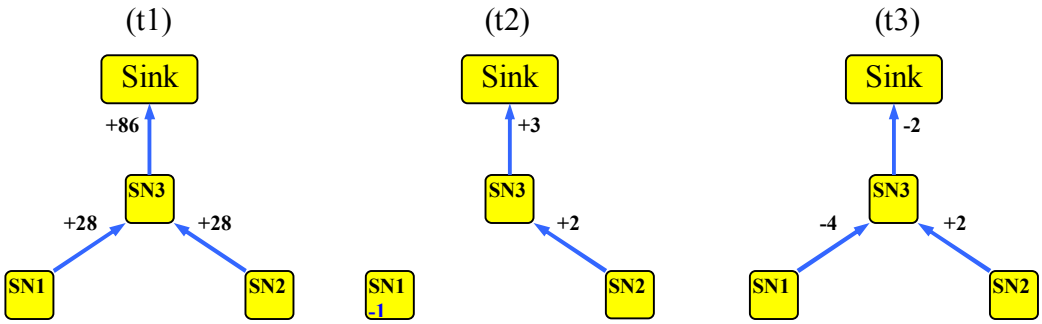
SN2 → SN3 : Ο SN2 ενημερώνει τον SN3, ότι η τιμή του μειώθηκε κατά 2.

SN3 → Sink : Ο SN3 αθροίζει τις ενημερώσεις που έλαβε από τους SN1 (-4) και SN2 (+2) με τη μεταβολή της δικής του τιμής (0) κι αποστέλλει το αποτέλεσμα, το οποίο και ισούται με +2 στη Sink.

Έτσι λοιπόν, κατά τη χρονική στιγμή t2 θα σταλούν 3 μηνύματα ενημέρωσης στη Sink και το περιεχόμενο της τελευταίας θα έχει ως εξής:

SN1	24	SN2	32	SN3	31

Σχήμα 4.9: Εύρεση βέλτιστου δέντρου δρομολόγησης



$2^{η}$ περίπτωση: Συνολικά, θα σταλούν και πάλι 8 μηνύματα ενημέρωσης (μπλε βέλη) από τους SNs στη Sink.

Συνολικά λοιπόν, θα σταλούν σε αυτήν την περίπτωση δρομολόγησης 8 μηνύματα ενημέρωσης από τους SNs στη Sink. Δηλαδή, η παρούσα δρομολόγηση δεν μειώνει το πλήθος των μηνυμάτων ενημέρωσης στο WSN. Προκειμένου λοιπόν να αποσυμφορίσουμε το WSN, θα εφαρμόσουμε για το παραπάνω WSN τον αλγόριθμο εύρεσης της βέλτιστου μονοπατιού δρομολόγησης των δεδομένων από τους SNs στη Sink αυτού.

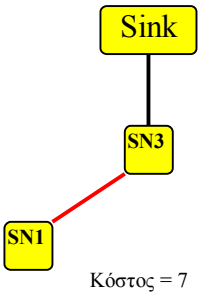
- $3^{η}$ περίπτωση

Αρχικά λοιπόν, εισάγουμε στο δέντρο τη Sink και το σύνολο S των υποψηφίων προς εισαγωγή στο δέντρο SNs, θα απαρτίζεται από όλους τους SNs που μπορούν να έχουν απευθείας επικοινωνία με αυτήν, δηλαδή μόνο από τον SN3, οπότε θα είναι: S={SN3}.
Sink αυτού.

Είναι προφανές λοιπόν, ότι θα προσθέσουμε στο δέντρο δρομολόγησης τον SN3, ως παιδί της Sink. Παράλληλα, θα αφαιρέσουμε από το σύνολο S τον SN3 και θα προσθέσουμε σε αυτό τους SNs, που μπορούν συνδεθούν μέσω κάποιας ασύρματης ζεύξης με τον SN3, δηλαδή τους SN1 και SN2.

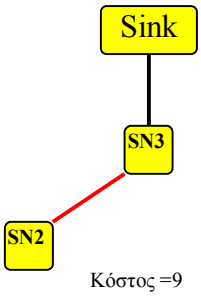
Στη συνέχεια από τους SNs του S, επιλέγουμε αυτόν με το μικρότερο κόστος και τον εισάγουμε στο δέντρο και πιο συγκεκριμένα στον SN3 . Οι δυνατές επιλογές επέκτασης του δέντρου με κάποιον από τους SNs του συνόλου S είναι οι ακόλουθες:

Δέντρο δρομολόγησης (α)



Κόστος = 7

Δέντρο δρομολόγησης (β)



Κόστος =9

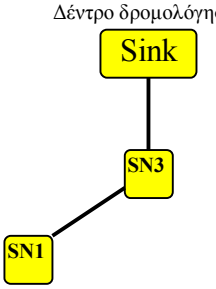
Το κόστος καθενός εκ των παραπάνω πιθανών δέντρων δρομολόγησης (α) και (β), θα ισούται με:

(α) - Κόστος SN1 → SN3 → Sink :
(Κόστος SN1→SN3→ Sink (t1→t2)) + (Κόστος SN1→SN3→ Sink (t2→t3)) =
= (Κόστος SN1→SN3(t1→t2)) + (Κόστος SN3→ Sink (t1→t2)) +
+ (Κόστος SN1→SN3(t2→t3)) + (Κόστος SN3→ Sink (t2→t3)) =
= |1*(-1)| + |1*(0+1)| + |1*(-1-3)| + |1*(0+1-4)| = 0 + 0 + 4 + 3 = 7

(β) - Κόστος SN2 → SN3 → Sink:
(Κόστος SN2→SN3→ Sink (t1→t2)) + (Κόστος SN2→SN3→ Sink (t2→t3)) =
= (Κόστος SN2→SN3(t1→t2)) + (Κόστος SN3→ Sink (t1→t2)) +
+ (Κόστος SN2→SN3(t2→t3)) + (Κόστος SN3→ Sink (t2→t3)) =
|1*2| + |1*(2+1)| + |1*(2)| + |1*(2+0)| = 2 + 3 + 2 + 2 = 9

Συνεπώς, από τους SNs του S, αυτός με το μικρότερο κόστος είναι ο SN1, οπότε και τον εισάγουμε στο δέντρο ως παιδί του SN3. Παράλληλα, θα αφαιρέσουμε από το σύνολο S τον SN3, οπότε και θα παραμείνει σε αυτό, μόνο ο SN2, δηλαδή θα είναι: S={SN2}

Δέντρο δρομολόγησης

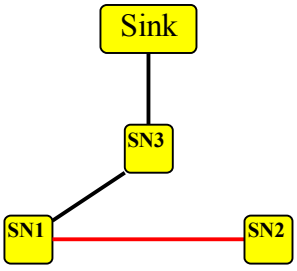


Σύνολο υποψηφίων κόμβων

S={SN2}

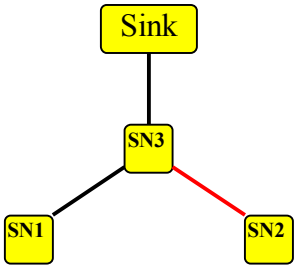
Ο SN2, μπορεί να προστεθεί είτε στον SN1 είτε στον SN3. Οι δυνατές επιλογές επέκτασης του δέντρου με κάποιον από τους SNs του συνόλου S είναι οι ακόλουθες:

Δέντρο δρομολόγησης (α)



(Κόστος = 4)

Δέντρο δρομολόγησης (β)



(Κόστος =11)

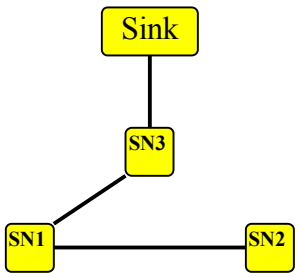
Οπότε, θα υπολογίσουμε το κόστος της προσθήκης του SN2 στον SN3 και του SN2 στον SN1, προκειμένου να αποφασίσουμε με ποιον από τους 2, θα συνδεθεί. Έχουμε λοιπόν:

(α) - Κόστος SN2 → SN1 → SN3 → Sink :
(Κόστος SN2→ SN1 → SN3→ Sink (t1→t2)) +
+ (Κόστος SN2→SN1→ SN3 → Sink (t2→t3)) =
(Κόστος SN2→SN1(t1→t2)) + (Κόστος SN1→ SN3 (t1→t2)) +
+ (Κόστος SN3→Sink(t1→t2)) + (Κόστος SN2→SN1(t2→t3)) +
+ (Κόστος SN1→ SN3 (t2→t3)) + (Κόστος SN3→Sink(t2→t3)) =
= |1*2| + |1*(2-1)| + |1*(0+1)| +|1*2| + |1*(2+1-3)| + |1*1|=
2 + 0 +0 +2 + 0 + 0 = 4

(β) – (Κόστος (SN1 Λ SN2) → SN3 → Sink):
(Κόστος (SN1 Λ SN2) → SN3 → Sink) =
= |1*(-1)| + |1*2| + |1*(2+1+0)| + |1*(-1-3)| + |1*2| + |1*0|=
= 0 + 2 + 3 + 4 + 2 + 0 = 11

Άρα ο SN2, θα εισαχθεί στο δέντρο ως παιδί του SN1. Το δε νέο σύνολο S των υποψηφίων προς εισαγωγή στο δέντρο SNs, θα είναι πλέον το κενό σύνολο, οπότε τερματίζεται ο αλγόριθμος δίνοντας στην έξοδό του, το ακόλουθο δέντρο δρομολόγησης:

Βέλτιστο δέντρο δρομολόγησης



Συνεπώς, με βάση το παραπάνω βέλτιστο δέντρο δρομολόγησης, ο SN2 στέλνει τις ενημερώσεις του στον SN1, ο SN1 στέλνει τις ενημερώσεις του στον SN3 κι ο SN3 στη Sink. Άρα, θα έχουμε:

(1-min, t1)
Αρχικά λοιπόν κατά τη χρονική στιγμή t1 , όλοι οι SNs θα αποστείλουν τα δεδομένα των μετρήσεών τους στη Sink. Πιο συγκεκριμένα, θα είναι:
SN2 → SN1 : Ο SN2 ενημερώνει τον SN1, ότι η τιμή του αυξήθηκε από 0 σε 28.

SN1 → SN3 : Ο SN1 αθροίζει την ενημέρωση που έλαβε από τον SN2 (+28) με τη μεταβολή της δικής του τιμής (+28) κι αποστέλλει το αποτέλεσμα, το οποίο και ισούται με +56 στον SN3.

SN3 → Sink : Ο SN3 αθροίζει την ενημέρωση που έλαβε από τον SN1 (+56) με τη μεταβολή της δικής του τιμής (+30) κι αποστέλλει το αποτέλεσμα, το οποίο και ισούται με +86 στη Sink.

Έτσι λοιπόν, κατά τη χρονική στιγμή t1 θα σταλούν 3 μηνύματα ενημέρωσης στο WSN και το περιεχόμενο της Sink θα έχει ως εξής:

SN1	28	SN2	28	SN3	30

(2-min, t2)
SN2 → SN1 : Ο SN2 ενημερώνει τον SN1, ότι η τιμή του αυξήθηκε κατά 2.

SN1 → SN3 : Ο SN1 αθροίζει την ενημέρωση που έλαβε από τον SN2 (+2) με τη μεταβολή της δικής του τιμής (-1). Το άθροισμα αυτό, θα ισούται με +1, με αποτέλεσμα το εγκατεστημένο στον SN1 φίλτρο να αποτρέπει την αποστολή ενημέρωσης για τη νέα τιμή του SN1 στον SN3. Ως αποτέλεσμα, ο SN1 δεν θα στείλει κανένα μήνυμα ενημέρωσης στον SN3 και θα αποθηκεύσει την τρέχουσα μεταβολή της τιμής του(+1) στη μνήμη του.

SN3 → Sink : Η τιμή του SN3 αυξήθηκε κατά 1, με αποτέλεσμα το εγκατεστημένο σε αυτόν φίλτρο να αποτρέπει την αποστολή ενημέρωσης για τη νέα τιμή του SN3 στη Sink. Ως αποτέλεσμα, ο SN3 δεν θα στείλει κανένα μήνυμα ενημέρωσης στη Sink και θα αποθηκεύσει την τρέχουσα μεταβολή της τιμής του(+1) στη μνήμη του.

Έτσι λοιπόν, κατά τη χρονική στιγμή t2 θα σταλεί μόνο 1 μήνυμα ενημέρωσης στο WSN και το περιεχόμενο της Sink θα έχει ως εξής:

SN1	28	SN2	28	SN3	30

(3-min, t3)
SN2 → SN1 : Ο SN2 ενημερώνει τον SN1, ότι η τιμή του αυξήθηκε κατά 2.

SN1 → SN3 : Ο SN1 αθροίζει την ενημέρωση που έλαβε από τον SN2 (+2) με τη μεταβολή της δικής του τιμής (-3), καθώς και με τη μεταβολή της τιμής αυτού, κατά την προηγούμενη χρονική στιγμή (t2), η οποία βρίσκεται αποθηκευμένη στη μνήμη αυτού και ισούται με +1. Ως αποτέλεσμα, η συνολική μεταβολή της τιμής του SN1, θα ισούται με 0, με αποτέλεσμα να μην χρειάζεται ο SN1 να αποστείλει μήνυμα ενημέρωσης στον SN3. Παράλληλα ο SN1, διαγράφει από τη μνήμη του όλες τις προηγούμενες μεταβολές τιμών που βρίσκονται αποθηκευμένες σε αυτήν.

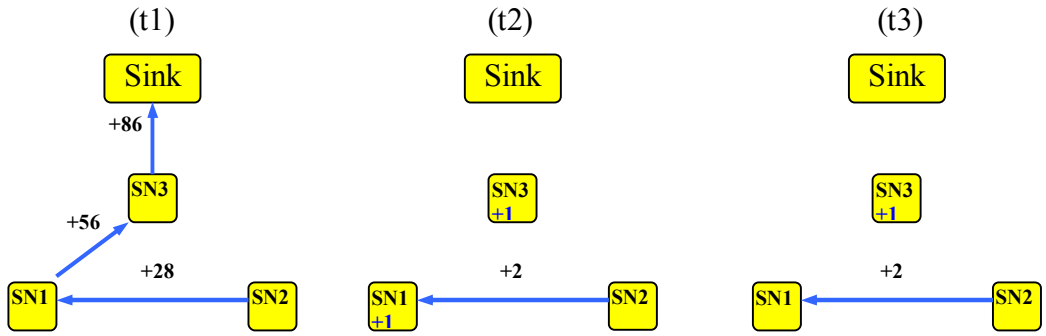
SN3 → Sink : Η τιμή του SN3 δεν μεταβάλλεται κατά την τρέχουσα χρονική στιγμή, ούτε επίσης λαμβάνει κάποια ενημέρωση από τον SN1. Οπότε, η μόνη ενημέρωση που θα έχει είναι αυτή που βρίσκεται αποθηκευμένη στη μνήμη του από την προηγούμενη χρονική στιγμή t2, η οποία και ισούται με +1, με αποτέλεσμα λόγω του εγκατεστημένου φίλτρου να μην αποστέλλεται στη Sink και να παραμένει στη μνήμη του SN3.

Έτσι λοιπόν, κατά τη χρονική στιγμή t3 θα σταλεί μόνο 1 μήνυμα ενημέρωσης στο WSN και το περιεχόμενο της Sink, θα έχει ως εξής:

SN1	28	SN2	28	SN3	30

Συνολικά λοιπόν, θα σταλούν σε αυτήν την περίπτωση δρομολόγησης 5 μηνύματα ενημέρωσης από τους SNs στη Sink. Δηλαδή, η παρούσα δρομολόγηση μειώνει αισθητά το πλήθος των μηνυμάτων ενημέρωσης στο WSN.

Σχήμα 4.10: Εύρεση βέλτιστου δέντρου δρομολόγησης



3η περίπτωση: Συνολικά, θα σταλούν μόνον 5 μηνύματα ενημέρωσης (μπλε βέλη) από τους SNs στη Sink.

Έτσι λοιπόν και μετά το βέλτιστο δέντρο δρομολόγησης, το οποίο και προέκυψε από την εφαρμογή στο παραπάνω WSN του αλγορίθμου εύρεσης του βέλτιστου δέντρου δρομολόγησης για την αποστολή των δεδομένων των SNs στη Sink σε ένα WSN, στο οποίο η εξυπηρέτηση των αιτημάτων υλοποιείται σε 2 επίπεδα, το πλήθος των μηνυμάτων ενημέρωσης περιορίστηκε από 8 σε μόλις5. Όπως παρατηρούμε λοιπόν, ο τρόπος με τον οποίο δρομολογούνται τα δεδομένα που αποστέλλονται από τους SNs στη Sink σε ένα WSN αποτελεί καθοριστικό παράγοντα για το πλήθος των ενημερώσεων της Sink από τους SNs. Πιο συγκεκριμένα, επιλέγοντας την κατάλληλη δρομολόγηση επιτυγχάνουμε το ελάχιστο πλήθος μηνυμάτων ενημέρωσης, αποσυμφορίζοντας με τον τρόπο αυτό το WSN, με αποτέλεσμα τη συνολική βελτίωση της αποδοτικότητας του WSN.

Κεφάλαιο 5

Συμπερασματικά – Παρουσίαση βελτιστοποιημένου WSN

Περιεχόμενα

Σε αυτό το κεφάλαιο, θα παρουσιάσουμε ένα ολοκληρωμένο WSN, το οποίο ενσωματώνει τις τεχνικές βελτιστοποίησης και τα χαρακτηριστικά, που παρουσιάσαμε στην παρούσα εργασία.

5.1 Αρχιτεκτονική

Το WSN μας αποτελείται από μία **Sink** κι από ένα πλήθος **SNs** ομαδοποιημένων σε συστάδες. Πιο συγκεκριμένα, υπάρχουν **τρεις συστάδες SNs** από τις οποίες η πρώτη αποτελείται από 4 SNs, ενώ η δεύτερη και η τρίτη από 5 SNs. Ένας από τους SNs κάθε συστάδας έχει οριστεί ως **head SN** δηλαδή επικεφαλής αυτής, γεγονός που στην πράξη σημαίνει ότι έχει μεγαλύτερη διαθεσιμότητα σε πόρους από τους υπόλοιπους SNs της συστάδας αυτής.

5.2 Τοπολογία

Οι head SNs συνδέονται με τη Sink σε τοπολογία αστέρα. Στην πρώτη συστάδα χρησιμοποιείται η τοπολογία δέντρου, στη δεύτερη η τοπολογία διπλού δακτυλίου και στην τρίτη η τοπολογία-πλέγματος 3. Όλοι οι κόμβοι του WSN, συμπεριλαμβανομένης και της Sink είναι εξοπλισμένοι με **μνήμη flash τύπου NAND**, ενώ διαθέτουν επίσης και μία μονάδα **κρυφής μνήμης**.

5.3 Ευρετήρια

Κάθε κόμβος του WSN διαθέτει ένα **τοπικό ευρετήριο** των δεδομένων που βρίσκονται αποθηκευμένα στη μνήμη του. Επιπλέον υλοποιούνται τρία **κατανεμημένα ευρετήρια**. Ένα μεταξύ των head SNs με τη μορφή διπλού δακτυλίου ένα μεταξύ των SNs της πρώτης συστάδας με τη μορφή απλού δακτυλίου και τέλος ένα μεταξύ των SNs της τρίτης συστάδας επίσης με τη μορφή απλού δακτυλίου.

5.4 Λήψη δεδομένων από τους SNs κι αποστολή τους στη Sink

Οι SNs λαμβάνουν τις τιμές των μετρήσεων, που πραγματοποιούν οι αισθητήρες τους και τις αποθηκεύουν στην ενσωματωμένη μνήμη NAND flash, που διαθέτουν για το σκοπό. Κατά την αποθήκευση των δεδομένων χρησιμοποιούνται οι τεχνικές της **διαμέρισης της μνήμης**, της **εύρεσης συσχετίσεων μεταξύ των δεδομένων** και της **επαναχρησιμοποίησης δεδομένων**. Εν συνεχεία, όποτε ζητηθεί τα δεδομένα αυτά αποστέλλονται στη Sink με χρήση κάποιου βέλτιστου δέντρου δρομολόγησης.

5.5 Εξυπηρέτηση αιτημάτων

Σε αυτό το WSN λαμβάνει χώρα εξυπηρέτηση των αιτημάτων σε δύο επίπεδα, δηλαδή **κατανεμημένη εξυπηρέτηση αιτημάτων**. Επιπρόσθετα, όποτε η Sink λαμβάνει κάποιο καινούργιο αίτημα εφαρμόζει σε αυτό τον **αλγόριθμο βελτιστοποίησης αιτημάτων.**

Παράρτημα Ι

Ευρετήριο Εικόνων

Κεφάλαιο 1

Κεφάλαιο 2

Παράρτημα ΙΙ

Ευρετήριο Πινάκων

Κεφάλαιο 1

Κεφάλαιο 2

Κεφάλαιο 4

Παράρτημα ΙΙΙ

Ευρετήριο Σχημάτων

Κεφάλαιο 1

Κεφάλαιο 2

Κεφάλαιο 3

Κεφάλαιο 4

Κεφάλαιο 5

Βιβλιογραφία

Δικτυακοί Τόποι

[1] http://www.thelab.gr
«Ασύρματα Δίκτυα Αισθητήρων»
(Διαδικτυακό Forum)

[2] http:// www2.enthesis.net
«Αισθητήρες όλου του κόσμου, ενωθείτε»
(Παναγιώτης Τσακαλίδης)

[3] http:// www.wikipedia.org
«Cache»
«Wireless Sensor Network»
(Διαδικτυακή εγκυκλοπαίδεια)

[4] http:// www.akvagroup.com
«Ασύρματο Δίκτυο Αισθητήρων για μονάδες Ιχθυοκαλλιέργειας»
(Ιστοσελίδα Ανώνυμης Εταιρίας)

Βιβλιογραφία

[1] «Ιστορία της Τεχνολογίας»
(Στ. Γ. Φραγκόπουλου)

[2] «10 Emerging Technologies That Will Change The World»
(Άρθρο του MIT Technology Review)

[3] «Multiple Query Optimization for Wireless Sensor Networks»
(Shili Xiang, Hock Beng Lim, Kian Lee Tan)

[4] «Design of Flash-Based DBMS: An In-Page Logging Approach»
(Sang-Won Lee, Bongki Moon)

[5] «B+ trees»
(Susan Anderson-Freed)

[6] «In-Network Execution of Monitoring Queries in Sensor Networks»
(Xiaoyan Yang, Hock Beng Lim, M.Tamer Ozsu, Kian Lee Tan)

[7] «On Distributed Query Processing in Flash-based Sensor Network»
(Jianliang Xu, Xueyan Tang, Wang-Chien Lee)

[8] «Processing Precision-Constrained Approximate Queries in Wireless Sensor Networks»
(Minji Wu, Jianliang Xu, Xueyan Tang)

[9] «Processing Approximate Aggregate Queries in Wireless Sensor Networks»
(Antonios Deligiannakis, Yannis Kotidis, Nick Roussopoulos)

[10] «How Flash Memory Changes the DBMS World»
(Hans Olav Norheim)

[11] « Έλεγχος συμφόρησης στα Ασύρματα Δίκτυα Αισθητήρων»
(Παναγιώτα Κούλλουρου, Τμήμα Πληροφορικής, Πανεπιστήμιο Κύπρου)
[12] «Compendium of Applications for Wireless Sensor Network»
(Ankit Mehta, Deepak T.J, Arpit Mehta)
[13] «Ασφάλεια στα Ασύρματα Δίκτυα Αισθητήρων»
(Αλέξανδρος Κ. Καπετανάκης)
[14] « Ασύρματα δίκτυα αισθητήρων για τη μη επεμβατική παρακολούθηση βιοσήματος»
(Λοϊζος Λοϊζου)
[15] «Δομές Δεδομένων – Ταξινόμηση και Αναζήτηση με Java»
(Παναγιώτης Δ. Μποζάνης)
[16] « Wireless Sensor Networks : A Survey Revisited»
(Ian F. Akyildiz, Mehmet C. Vuran, Ozgur B. Akan, Weilian Su)
[17] «Balancing Energy Efficiency and Quality of Aggregate Data in Sensor Networks»
(Mohamed Sharaf, Jonathan Beaver, Alexandros Lamprinidis, Panos Chrysanthis)
[18] «Two-Tier Multiple Query Optimization for Sensor Networks»
(Shili Xiang, Hock Beng Lim, Kian-Lee Tan, Yongluan Zhou)