Title

Ευρετηριοποίηση Χωροχρονικών δεδομένων με το hBπ-tree

Author Information

Ευάγγελος Κανούλας, Τμήμα Εφαρμοσμένης Πληροφορικής, Πανεπιστήμιο Μακεδονίας, Εγνατίας 156, 54006 Θεσσαλονίκη, τηλ. 0310-891844, it9916@uom.gr

Γεώργιος Ευαγγελίδης, Τμήμα Εφαρμοσμένης Πληροφορικής, Πανεπιστήμιο Μακεδονίας, Εγνατίας 156, 54006 Θεσσαλονίκη, τηλ. 0310-891844, gevan@uom.gr

Abstract

Database applications become more and more complex and the data they operate on often has to be indexed using more than one attribute. In addition, there exist and continuously emerge new applications that require a DBMS to manipulate both current and historical data. In other words, there is a pressing demand for indexing data using the time dimension. To efficiently use a temporal database one needs specialized indexing structures. In this paper, we present the modifications that need to be performed on a multi-dimensional index method, the hB$^{\pi}$-tree, so that it becomes appropriate for indexing temporal data. We propose certain modifications on the structure of the index and on the node splitting algorithm that lend the hB$^{\pi}$-tree the characteristics of the TSB-tree and make it suitable for indexing temporal or spatio-temporal data.

Περίληψη

Οι εφαρμογές των βάσεων δεδομένων γίνονται ολοένα πιο πολύπλοκες. Τα δεδομένα που αποθηκεύονται στην πλειονότητα αυτών είναι δεδομένα που απαιτούν ταυτόχρονη ευρετηριοποίηση με βάση περισσότερα από ένα χαρακτηριστικά. Παράλληλα, σε πολλές εφαρμογές υπάρχει η ανάγκη να διατηρούνται δεδομένα που αναφέρονται τόσο στο παρόν όσο και στο παρελθόν. Με άλλα λόγια, υπάρχει η ανάγκη καταχώρισης ακόμη μιας διάστασης, της διάστασης του χρόνου. Για την αποδοτική χρήση μιας τέτοιας βάσης χρονικών δεδομένων απαιτείται και η κατάλληλη δομή ευρετηριοποίησης. Στην παρούσα εργασία παρουσιάζουμε τις τροποποιήσεις που πρέπει να γίνουν σε μια δομή ευρετηριοποίησης πολυδιάστατων δεδομένων, το hB$^{\pi}$-tree, ώστε αυτή να υποστηρίζει και χρονικά δεδομένα. Προτείνουμε κάποιες τροποποιήσεις στη δομή του δένδρου και στον αλγόριθμο διάσπασης των κόμβων ώστε το hB$^{\pi}$-tree να αποκτήσει τα χαρακτηριστικά του TSB-tree και να γίνει κατάλληλο για την ευρετηριοποίηση χρονικών ή χωροχρονικών δεδομένων.

Ευρετηριοποίηση Χωροχρονικών δεδομένων με το hB$^{\pi}$-tree

Ευάγγελος Κανούλας, Γεώργιος Ευαγγελίδης

Τμήμα Εφαρμοσμένης Πληροφορικής, Πανεπιστήμιο Μακεδονίας

{it9916, gevan}@uom.gr

Abstract

Database applications become more and more complex and the data they operate on often has to be indexed using more than one attribute. In addition, there exist and continuously emerge new applications that require a DBMS to manipulate both current and historical data. In other words, there is a pressing demand for indexing data using the time dimension. To efficiently use a temporal database one needs specialized indexing structures. In this paper, we present the modifications that need to be performed on a multi-dimensional index method, the hB$^{\pi}$-tree, so that it becomes appropriate for indexing temporal data. We propose certain modifications on the structure of the index and on the node splitting algorithm that lend the hB$^{\pi}$-tree the characteristics of the TSB-tree and make it suitable for indexing temporal or spatio-temporal data.

Περίληψη

Οι εφαρμογές των βάσεων δεδομένων γίνονται ολοένα πιο πολύπλοκες. Τα δεδομένα που αποθηκεύονται στην πλειονότητα αυτών είναι δεδομένα που απαιτούν ταυτόχρονη ευρετηριοποίηση με βάση περισσότερα από ένα χαρακτηριστικά. Παράλληλα, σε πολλές εφαρμογές υπάρχει η ανάγκη να διατηρούνται δεδομένα που αναφέρονται τόσο στο παρόν όσο και στο παρελθόν. Με άλλα λόγια, υπάρχει η ανάγκη καταχώρισης ακόμη μιας διάστασης, της διάστασης του χρόνου. Για την αποδοτική χρήση μιας τέτοιας βάσης χρονικών δεδομένων απαιτείται και η κατάλληλη δομή ευρετηριοποίησης. Στην παρούσα εργασία παρουσιάζουμε τις τροποποιήσεις που πρέπει να γίνουν σε μια δομή ευρετηριοποίησης πολυδιάστατων δεδομένων, το hB$^{\pi}$-tree, ώστε αυτή να υποστηρίζει και χρονικά δεδομένα. Προτείνουμε κάποιες τροποποιήσεις στη δομή του δένδρου και στον αλγόριθμο διάσπασης των κόμβων ώστε το hB$^{\pi}$-tree να αποκτήσει τα χαρακτηριστικά του TSB-tree και να γίνει κατάλληλο για την ευρετηριοποίηση χρονικών ή χωροχρονικών δεδομένων.

1. Εισαγωγή

Οι συνηθισμένες βάσεις δεδομένων διατηρούν μόνο την *τρέχουσα κατάσταση* (current state) των δεδομένων που αποθηκεύονται σε αυτές. Όταν μια συναλλαγή μεταβάλει την κατάσταση της βάσης δεδομένων τότε η προηγούμενη κατάσταση δεν διατηρείται. Δεν υπάρχει δηλαδή κάποιο είδος μνήμης στη βάση δεδομένων ώστε να είναι προσβάσιμη παλιότερη κατάσταση της. Αυτού του είδους οι βάσεις δεδομένων, δηλαδή αυτές που διατηρούν ένα *στιγμιότυπο* (snapshot) μόνο της πραγματικότητας που απεικονίζεται σε αυτές, ονομάζονται βάσεις δεδομένων στιγμιότυπου.

Ωστόσο, σε πολλές εφαρμογές υπάρχει η ανάγκη της υποστήριξης δεδομένων που αναφέρονται στο παρελθόν, το παρόν ή ακόμη και το μέλλον. Μια τέτοια εφαρμογή θα μπορούσε, για παράδειγμα, να αφορά τα στοιχεία των υπαλλήλων μιας επιχείρησης. Σε αυτή τη βάση δεδομένων είναι ανάγκη, πρώτα από όλα, να διατηρούνται εκτός των συνηθισμένων, ανεξάρτητων από το χρόνο, δεδομένων (όνομα, επίθετο, κλπ.) και κάποια δεδομένα που είναι συνάρτηση του χρόνου, αλλάζουν δηλαδή με το χρόνο (μισθός υπαλλήλου, θέση υπαλλήλου στην επιχείρηση, κλπ.). Επίσης, υπάρχει η ανάγκη να διατηρούνται δεδομένα του παρελθόντος ώστε να μπορούν να απαντώνται αιτήματα του τύπου «ποιοι ήταν οι διευθυντές της επιχείρησης τον Μάιο του 2000;» ή «Ποια η μισθολογική πορεία ενός υπαλλήλου της επιχείρησης έως σήμερα;». Αυτού του είδους τα δεδομένα, εκτός από τα συμβατικά στοιχεία, εμπεριέχουν και τη διάσταση του χρόνου και ονομάζονται *χρονικά* δεδομένα. Στην περίπτωση που τα αποθηκευμένα αντικείμενα είναι γεωμετρικά σημεία στον *k*-διάστατο χώρο, μπορούμε να μιλάμε για *χωροχρονικά* δεδομένα.

Για τέτοιου είδους εφαρμογές οι συνηθισμένες βάσεις δεδομένων είναι ανεπαρκείς. Για δεδομένα που εμπλέκουν την διάσταση του χρόνου απαιτούνται *χρονικές βάσεις δεδομένων* (temporal databases) [Snodgrass, Ahn 1986]. Ο όρος χρονικές βάσεις δεδομένων αναφέρεται γενικά σε βάσεις δεδομένων που υποστηρίζουν *δεδομένα μεταβαλλόμενα με το χρόνο* (time varying data).

Σε ότι αφορά τις χρονικές βάσεις δεδομένων υπάρχει αυξημένο ερευνητικό ενδιαφέρον τα τελευταία χρόνια [Tsotras, Kumar 1996]. Μια ποικιλία θεμάτων γύρω από χρονικές βάσεις δεδομένων έχει εξεταστεί στο [Ozsoyoglu, Snodgrass 1995] περιλαμβάνοντας μοντέλα χρονικών δεδομένων, γλώσσες αιτημάτων, μεθόδους πρόσβασης κλπ.

1.1 Ταξινόμηση του χρόνου στις βάσεις δεδομένων

Μια ταξινόμηση των ειδών του χρόνου που εμπλέκονται σε μια βάση δεδομένων γίνεται από τους Snodgrass και Ahn [Snodgrass, Ahn 1985]. Πιο συγκεκριμένα, ο χρόνος που εμφανίζεται σε βάσεις δεδομένων ταξινομείται σε *χρόνο συναλλαγής* (transaction time) και *χρόνο εγκυρότητας* (valid time). Ως χρόνος συναλλαγής ορίζεται ο χρόνος που ένα δεδομένο καταχωρείται στη βάση δεδομένων. Ο χρόνος αυτός αυξάνει μονότονα και μπορεί να υλοποιηθεί χρησιμοποιώντας το *χρόνο ολοκλήρωσης* (commit time) των συναλλαγών [Salzberg 1994]. Ως χρόνος εγκυρότητας ορίζεται ο χρόνος κατά τον οποίο ένα δεδομένο γίνεται έγκυρο στον πραγματικό κόσμο. Με βάση αυτή την ταξινόμηση του χρόνου ταξινομούνται και οι χρονικές βάσεις δεδομένων ως *χρόνου συναλλαγής* (transaction time), *χρόνου εγκυρότητας* (valid time) και *δι-χρονικές* (bi-temporal) [Dyreson 1994].

1.2 Βάσεις δεδομένων χρόνου συναλλαγής

Οι βάσεις δεδομένων χρόνου συναλλαγής είναι αυτές τις οποίες διαπραγματεύεται η παρούσα εργασία. Μια τέτοιου είδους βάση δεδομένων καταγράφει την εξέλιξη της βάσης δεδομένων και όχι την εξέλιξη του πραγματικού κόσμου. Οι χρονικές στιγμές, δηλαδή, που καταχωρεί είναι αυτές κατά τις οποίες έγιναν διάφορες ενέργειες επί της βάσης δεδομένων χωρίς απαραίτητα αυτές να αντιστοιχούν στις χρονικές στιγμές που οι αντίστοιχες αλλαγές συνέβησαν στην πραγματικότητα. Θεωρείται επίσης ότι υπάρχει μια γραμμική εξέλιξη του χρόνου σε αυτές τις βάσεις δεδομένων [Ozsoyoglou, Snodgrass 1985], δηλαδή μια νέα κατάσταση της βάσης δημιουργείται με ενέργειες μόνο επί της τρέχουσας κατάστασης της βάσης.

Θεωρούμε μια βάση δεδομένων ως ένα σύνολο αντικειμένων που εξελίσσεται κατά τη διάρκεια του χρόνου. Το μοντέλο της βάσης δεδομένων που θα χρησιμοποιηθεί είναι το *χρονικό μοντέλο εκδόσεων εγγραφών* (tuple-versioning temporal model) [Lorentzos, Johnson 1988; Navathe, Ahmed 1989]. Σύμφωνα με αυτό, η βάση δεδομένων είναι ένα σύνολο εγγραφών οι οποίες καταχωρούν τις διαφορετικές παραλλαγές των αντικειμένων. Κάθε τέτοια εγγραφή αποτελείται από ένα κλειδί ανεξάρτητο του χρόνου και ένα αριθμό δεδομένων χρονικά μεταβαλλόμενων. Επίσης αποτελείται κι από ένα διάστημα τιμών που υλοποιείται ως δύο χαρακτηριστικά, τον αρχικό και τελικό χρόνο. Ο χρόνος αντιπροσωπεύεται από μια ακολουθία διαδοχικών μη αρνητικών ακεραίων. Κάθε αλλαγή στο σύνολο των αντικειμένων υποδηλώνεται από έναν τέτοιο ακέραιο. Οι αλλαγές που είναι δυνατόν να συμβούν είναι η προσθήκη και διαγραφή ενός αντικειμένου καθώς και η αλλαγή των τιμών των χαρακτηριστικών των αντικειμένων. Ένα αντικείμενο είναι *ενεργό* (alive) από τη στιγμή που προστίθεται στο σύνολο ως τη στιγμή (αν υπάρξει τέτοια) που διαγράφεται από αυτό. Η κατάσταση του συνόλου αυτού κατά τη χρονική στιγμή t είναι το σύνολο των ενεργών αντικειμένων τη χρονική αυτή στιγμή. Αλλαγές μπορούν να γίνουν μόνο επί της τρέχουσας κατάστασης του συνόλου των αντικειμένων.

Όταν ένα αντικείμενο προστίθεται στο σύνολο κατά τη χρονική στιγμή t, μια εγγραφή που αντιπροσωπεύει το αντικείμενο αυτό προστίθεται στη βάση δεδομένων μαζί με ένα διάστημα χρόνου του οποίου το πάνω όριο είναι αρχικά άγνωστο. Όταν ένα αντικείμενο διαγράφεται από τη βάση δεδομένων, στην πραγματικότητα συνεχίζει να υπάρχει σε αυτή και διαγράφεται μόνο λογικά προσθέτοντας ως πάνω όριο του χρονικού διαστήματος το χρόνο διαγραφής.

Μια κατάλληλη μέθοδος προσπέλασης δεδομένων σε μια βάση δεδομένων χρόνου συναλλαγής πρέπει να συγκεντρώνει τα εξής χαρακτηριστικά [Salzberg, Tsotras 1999]: (α) να αποθηκεύει παρελθούσες καταστάσεις της βάσης δεδομένων, (β) να υποστηρίζει εισαγωγές, διαγραφές και τροποποιήσεις δεδομένων της τρέχουσας κατάστασης της βάσης δεδομένων, και (γ) να υποστηρίζει αποδοτικά προσπέλαση δεδομένων και αιτήματα σε κάθε διαφορετική κατάσταση της βάσης.

1.3 Η δική μας προσέγγιση

Για να απαντήσουμε αποδοτικά αιτήματα που εμπλέκουν ταυτόχρονα διαστήματα τιμών και των κλειδιών ευρετηριοποίησης αλλά και του χρόνου (transaction range-timeslice query), η καλύτερη πρακτική είναι η ομαδοποίηση των δεδομένων κατά την αποθήκευση τους τόσο ως προς το κλειδί ευρετηριοποίησης όσο και ως προς το χρόνο συναλλαγής. Με τον τρόπο αυτό, λογικά συσχετιζόμενα δεδομένα θα βρίσκονται αποθηκευμένα σε κοντινούς χώρους οπότε ελαχιστοποιείται ο αριθμός των σελίδων που πρέπει να προσπελασθούν για την απάντηση ενός αιτήματος. Οι μέθοδοι που χρησιμοποιούνται για αυτή την κατηγορία των αιτημάτων είναι συνήθως δενδρικές δομές των οποίων τα φύλλα αντιστοιχούν σε ένα πολυδιάστατο χώρο στον οποίο περιλαμβάνεται εκτός των κλειδιών ευρετηριοποίησης και η διάσταση του χρόνου. Οι δύο σημαντικότερες προσεγγίσεις είναι μέθοδοι βασισμένες στο R-tree [Stonebraker 1987; Kolovson, Stonebraker 1989; 1991] και στο B+-tree [Eatson 1986; Lomet, Salzberg 1989; Lanka, Mays 1991; Manolopoulos, Kapetanakis 1990; Verman, Verma 1997].

Η δική μας προσέγγιση αφορά μια παραλλαγή του hB^{π}-tree [Evangelidis, Lomet, Salzberg 1997]. Το hB^{π}-tree αποτελεί μια μέθοδο ευρετηριοποίησης πολυδιάστατων δεδομένων, δηλαδή, δεδομένων που ευρετηριοποιούνται ως προς περισσότερα του ενός χαρακτηριστικά. Η δομή αυτή ωστόσο δεν υποστηρίζει χρονικά δεδομένα. Στην παρούσα έρευνα παρουσιάζονται οι αλλαγές στη δομή του hB^{π}-tree καθώς και στον αλγόριθμο εισαγωγής δεδομένων ώστε να υποστηρίζεται η αποθήκευση και η αποδοτική πρόσβαση σε χωροχρονικά δεδομένα. Στην πραγματικότητα

ενσωματώνουμε τα χαρακτηριστικά του Time Split B-tree [Lomet, Salzberg 1989; 1990; 1993] στο hBπ-tree. Για αυτό θα εξετάσουμε καταρχήν τις δύο αυτές δομές.

2. HBπ-tree και TSB-tree

2.1 hBπ-tree

Το hBπ-tree [Evangelidis, Lomet, Salzberg 1997] είναι ένα ισοζυγισμένο δένδρο που αποτελείται από *κόμβους δεδομένων* (data nodes) και *κόμβους ευρετηρίου* (index nodes). Κάθε κόμβος είναι υπεύθυνος για ένα τμήμα του πολυδιάστατου χώρου (κάθε διάσταση αντιπροσωπεύει ένα χαρακτηριστικό του κλειδιού ευρετηριοποίησης). Κάθε κόμβος ευρετηρίου περιέχει μια εσωτερική δενδρική δομή, το *kd-tree*. Η δομή αυτή οργανώνει τις πληροφορίες σχετικά με τους κόμβους του hBπ-tree του χαμηλότερου επιπέδου καθώς και τις πληροφορίες για τους υποχώρους που έχουν αποσπαστεί από τον κόμβο. Οι κόμβοι δεδομένων βρίσκονται στο επίπεδο μηδέν και περιέχουν τα δεδομένα, δηλαδή τις *εγγραφές* (records). Οι εγγραφές αυτές είναι δυνατόν να είναι οργανωμένες σε ομάδες εγγραφών που ονομάζονται *λίστες εγγραφών* (record lists). Επίσης, οι κόμβοι δεδομένων είναι δυνατόν να περιέχουν και μια δομή kd-tree η οποία περιγράφει τα εσωτερικά όρια μεταξύ των λιστών εγγραφών του κόμβου δεδομένων.

Το kd-tree είναι ένα δυαδικό δένδρο, κάθε κόμβος του οποίου περιέχει ένα *χαρακτηριστικό ευρετηριοποίησης* (indexing attribute), μια *τιμή του χαρακτηριστικού* αυτού (attribute value) και δύο δείκτες, έναν αριστερό και ένα δεξί. Για να κινηθεί κανείς προς εγγραφές με τιμή χαρακτηριστικού μικρότερη από αυτή που είναι καταχωρημένη στον κόμβο ακολουθεί τον αριστερό δείκτη, ενώ σε αντίθετη περίπτωση τον δεξιό δείκτη. Κάθε ένας από τους δείκτες αυτούς οδηγεί είτε σε ένα kd-tree κόμβο χαμηλότερου επιπέδου, είτε σε ένα hBπ-tree κόμβο χαμηλότερου επιπέδου, είτε σε ένα hBπ-tree κόμβο του ίδιου επιπέδου. Στην περίπτωση των κόμβων ευρετηρίου, η *διαδρομή* (path) από τη ρίζα του kd-tree προς τα φύλλα, περιγράφει είτε το χώρο για τον οποίο υπεύθυνος είναι κάποιος *κόμβος-παιδί* (child node) του κόμβου ευρετηρίου και μιλάμε για *όρο ευρετηρίου* (index term), είτε το χώρο ο οποίος έχει αποσπαστεί από τον κόμβο (εξαιτίας μιας διάσπασης του κόμβου σε προηγούμενο χρόνο) και ανήκει πλέον στην υπευθυνότητα κάποιου *κόμβου-αδελφού* (sibling node) του κόμβου ευρετηρίου και μιλάμε για *όρο κόμβου-αδελφού* (sibling term), ενώ στην περίπτωση των κόμβων δεδομένων περιγράφει τα *εσωτερικά όρια* (inner boundaries) των λιστών εγγραφών. Στην Εικόνα 1 φαίνονται τα βασικά χαρακτηριστικά ενός κόμβου ευρετηρίου και ενός κόμβου δεδομένων.

Εικόνα 1 Παραδείγματα κόμβων ευρετηρίου και δεδομένων του hBπ-tree

Ο χώρος για τον οποίο είναι υπεύθυνος κάποιος κόμβος περιγράφεται από μια *συλλογή διαστημάτων τιμών* (boundaries). Κάθε όρος κόμβου-αδελφού καταλήγει σε ένα δείκτη προς τον αντίστοιχο κόμβο-αδελφό. Επίσης, σε ότι αφορά τους κόμβους ευρετηρίου, σε κάποιους kd-tree κόμβους καταχωρούνται διευθύνσεις των hBπ-tree κόμβων-παιδιών από τη διάσπαση των οποίων αυτοί προήλθαν. Οι διευθύνσεις αυτές ονομάζονται *παράσημα* (decorations) των

kd-tree κόμβων και το τμήμα μιας διαδρομής στο kd-tree που έχει παράσημο μόνο στον αρχικό κόμβο ονομάζεται *παρασημοφορεμένο τμήμα* (decorated fragment). Αν κάποιο παρασημοφορεμένο τμήμα συνεχίζεται σε έναν αδελφό κόμβο, τότε στο δείκτη προς τον κόμβο αυτό καταχωρείται μια τιμή TRUE που ονομάζεται *ένδειξη συνέχειας* (continuation flag). Τέλος, ο hB^{π}-tree κόμβος-αδελφός όπου συνεχίζεται ένα παρασημοφορεμένο τμήμα αποτελεί δευτερεύοντα γονιό του hB^{π}-tree κόμβου που εμφανίζεται ως παράσημο στη ρίζα του kd-tree του, και καταχωρεί μια τιμή TRUE που επίσης ονομάζεται *ένδειξη συνέχειας* (continuation flag) (δείτε Εικόνα 1).

Κάθε αλλαγή επί του hB^{π}-tree γίνεται με ξεχωριστές ατομικές ενέργειες (atomic actions) που διατηρούν το δένδρο *καλά σχηματισμένο* (well-formed). Όταν ένας κόμβος ευρετηρίου γεμίσει, τότε διασπάται με αποκοπή ενός υποδένδρου του kd-tree, ενώ ένας κόμβος δεδομένων διασπάται είτε με αποκοπή ενός υποδένδρου είτε με βάση κάποιο από τα χαρακτηριστικά ευρετηριοποίησης. Στη συνέχεια, με κάποια άλλη ατομική ενέργεια, προγραμματίζεται η *ενημέρωση* (posting) με τις απαραίτητες πληροφορίες του γονικού κόμβου του κόμβου που διασπάστηκε.

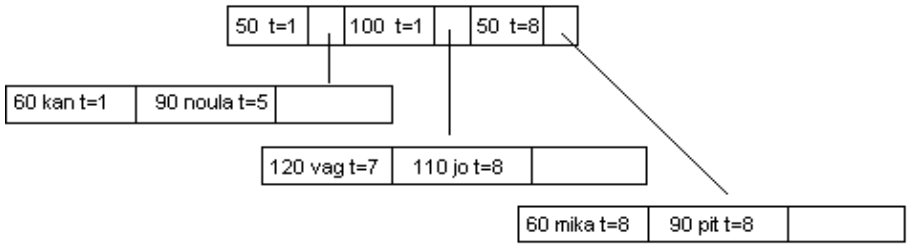
2.2 TSB-tree

Οι εγγραφές του TSB-tree είναι τριάδες αποτελούμενες από το χρόνο, το κλειδί ευρετηριοποίησης και ένα δείκτη προς το χαμηλότερο επίπεδο του δένδρου. Ο χρόνος και το κλειδί σε κάθε τέτοια εγγραφή είναι τα κάτω όρια μιας τετραγωνικής περιοχής (οι δύο διαστάσεις είναι ο χρόνος και το κλειδί).

Η αναζήτηση μιας εγγραφής ξεκινά από τη ρίζα του δένδρου. Όλες οι εγγραφές με χρόνο μεγαλύτερο από το χρόνο αναζήτησης αγνοούνται και στη συνέχεια αναζητείται το κλειδί με τη μεγαλύτερη τιμή που είναι μικρότερη της τιμής αναζήτησης και ακολουθείται ο δείκτης που αντιστοιχεί στην εγγραφή αυτή. Η ίδια διαδικασία επαναλαμβάνεται ως το επίπεδο των φύλλων.

Η διάσπαση των κόμβων, όποτε αυτοί υπερχειλίσουν, γίνεται είτε με βάση το κλειδί είτε με βάση το χρόνο είτε και τα δύο μαζί. Στη διάσπαση με βάση το χρόνο όλες οι εγγραφές με χρόνο μικρότερο του χρόνου διάσπασης μετακινούνται σε ένα νέο *ιστορικό* (historical) κόμβο. Για κάθε κλειδί η εγγραφή με το μεγαλύτερο χρόνο μικρότερο του χρόνου διάσπασης διατηρείται και στον τρέχοντα χρόνο, ενώ όλες οι υπόλοιπες διατηρούνται μόνο στον τρέχοντα κόμβο. Ως χρόνος διάσπασης των κόμβων δεδομένων μπορεί να είναι οποιαδήποτε χρονική στιγμή μετά την αρχική, ενώ για τους κόμβους ευρετηρίου ο χρόνος διάσπασης δε μπορεί να είναι μεγαλύτερος από τον αρχικό χρόνο κάθε τρέχοντος κόμβου-παιδιού.

Στη διάσπαση με βάση το κλειδί, σε ότι αφορά τους κόμβους δεδομένων, όλες οι εγγραφές με τιμή κλειδιού μεγαλύτερη ή ίση από το κλειδί διάσπασης μεταφέρονται σε ένα νέο κόμβο. Σε ότι αφορά τους κόμβους ευρετηρίου ως κλειδί διάσπασης επιλέγεται κάποιο από τα κλειδιά των εγγραφών του κόμβου. Για κάθε εγγραφή του κόμβου εξετάζεται ο κόμβος του χαμηλότερου επιπέδου στον οποίο οδηγεί ο αντίστοιχος δείκτης. Αν το πάνω όριο της διάστασης του κλειδιού του κόμβου αυτού είναι χαμηλότερο από το κλειδί διάσπασης τότε η εγγραφή που οδηγεί στον κόμβο αυτό μένει στον παλιό κόμβο. Αν το χαμηλότερο όριο είναι μεγαλύτερο από το κλειδί διάσπασης τότε μεταφέρεται στο νέο κόμβο ενώ αν το διάστημα του κλειδιού εμπεριέχει το κλειδί διάσπασης τότε ο γονικός κόμβος διατηρείται στον παλιό κόμβο και αντιγράφεται και στο νέο. Στην Εικόνα 2 φαίνεται ένα TSB-tree.



Εικόνα 2 Παράδειγμα TSB-tree

3 Time Split hB$^{\pi}$- tree

Όπως είπαμε, η σημερινή προσέγγιση των DBMS είναι να διατηρούν στη βάση δεδομένων μόνο την τρέχουσα κατάστασή της. Ωστόσο πολλές εφαρμογές απαιτούν την υποστήριξη χρόνου από το DBMS, οπότε απαιτούνται νέες δομές ευρετηριοποίησης με βάση το χρόνο. Μια τέτοια δομή είναι το TSB-tree. Ωστόσο το TSB-tree αναφέρεται στο δισδιάστατο χώρο (μία διάσταση το κλειδί και μία διάσταση ο χρόνος). Το γεγονός αυτό έχει ως αποτέλεσμα αιτήματα του τύπου, "να εμφανιστούν όλοι οι υπάλληλοι που υπήρχαν στην επιχείρηση το 1990 με επίθετο Παπαδόπουλος και μισθό μεγαλύτερο των 1000 ευρώ", αιτήματα, δηλαδή, που εμπλέκουν περισσότερα του ενός χαρακτηριστικά να απαιτούν μεγάλο χρόνο για την απάντηση τους. Μια διαφορετική προσέγγιση είναι η δημιουργία *πολυδιάστατων ευρετηρίων* (multi-attribute indexes), δηλαδή, ευρετηρίων που ευρετηριοποιούν περισσότερα του ενός χαρακτηριστικά. Τα πολυδιάστατα δεδομένα μπορούν να απεικονιστούν οπτικά ως σημεία ενός *k*-διάστατου χώρου, με μια διάσταση να είναι πλέον η διάσταση του χρόνου. Βασικό στοιχείο για την αποδοτική λειτουργία ενός τέτοιου ευρετηρίου είναι η αποθήκευση στο δίσκο σε κοντινές περιοχές δεδομένων που βρίσκονται γεωμετρικά κοντά στον *k*-διάστατο χώρο. Μια τέτοια δομή ευρετηριοποίησης είναι το hB$^{\pi}$-tree το οποίο ωστόσο δεν υποστηρίζει την ύπαρξη του χρόνου. Για το λόγο αυτό θα ενσωματώσουμε τα χαρακτηριστικά του TSB-tree στο hB$^{\pi}$-tree. Τη δενδρική δομή που θα προκύψει την ονομάζουμε TShB$^{\pi}$-tree (Time Split hB$^{\pi}$-tree).

Στη συνέχεια θα περιγράψουμε τις αλλαγές που απαιτούνται στη δομή των εγγραφών των δεδομένων που καταχωρούνται καθώς και στη διαδικασία εισαγωγής τους (Υποενότητα 3.1). Ακολουθεί μια σύντομη περιγραφή της δομής του δένδρου που δεν διαφέρει σε τίποτα από τη δομή του hB$^{\pi}$-tree (Υποενότητα 3.2). Στο τελευταίο τμήμα αυτής της ενότητας θα αναφερθούμε στον αλγόριθμο διάσπασης των κόμβων που στην ουσία είναι η σημαντικότερη αλλαγή που πραγματοποιούμε στο hB$^{\pi}$-tree για την υποστήριξη του χρόνου (Υποενότητα 3.3). Σε αυτή την υποενότητα θα παρουσιαστούν δυο παραλλαγές του αλγορίθμου διάσπασης που οδηγούν σε δυο παραλλαγές του δένδρου.

3.1 Εγγραφές Δεδομένων

Κάθε εγγραφή εκτός από τα δεδομένα που περιλαμβάνει, περιλαμβάνει κι ένα χρονικό διάστημα. Αυτό υλοποιείται με τη ύπαρξη δύο χαρακτηριστικών: του *αρχικού* και *τελικού* χρόνου (begin_time, end_time).

Όταν θέλουμε να εισάγουμε μια νέα εγγραφή, πρώτα πρέπει να βρούμε τον κατάλληλο κόμβο δεδομένων στον οποίο πρέπει να γίνει η εισαγωγή (με τη βοήθεια του ευρετηρίου στο οποίο, όπως θα δούμε στη συνέχεια, συμπεριλαμβάνεται

κι ο χρόνος). Στη συνέχεια αναζητούμε στον κόμβο δεδομένων κάποια προηγούμενη έκδοση της εγγραφής (κάποια εγγραφή δηλαδή με το ίδιο κλειδί), αν υπάρχει, και χωρίς τιμή στον τελικό της χρόνο. Αν υπάρχει τέτοια εγγραφή τότε θέτουμε ως τελικό χρόνο τον *τρέχοντα χρόνο* (current time), δηλαδή το χρόνο ολοκλήρωσης της συναλλαγής και καταχωρούμε τη νέα έκδοση της εγγραφής με αρχικό χρόνο ίσο με τον τρέχοντα χρόνο (start_time=current_time) ενώ αφήνουμε κενό τον τελικό χρόνο (end_time=null).

Αν ο κόμβος είναι γεμάτος και πραγματοποιήσουμε χρονική διάσπαση (διάσπαση με βάση τον τρέχοντα χρόνο) τότε αντιγράφουμε στον νέο κόμβο δεδομένων που δημιουργούμε όλες τις ενεργές εγγραφές, αυτές δηλαδή με τελικό χρόνο ίσο με null. Σε όλες τις αντιγραμμένες εγγραφές θέτουμε ως αρχικό χρόνο τον τρέχοντα χρόνο. Για παράδειγμα ας θεωρήσουμε τον κόμβο δεδομένων

D1

A,[0,]
B,[0,]

και έστω ότι τη χρονική στιγμή t=4 τροποποιείται η εγγραφή A

D1

A,[0,4]
B,[0,]
A,[4,]

Έστω ότι στη συνέχεια, στο χρόνο t=5, εισάγεται και η εγγραφή C. Αν υποθέσουμε ότι ο κόμβος δεδομένων έχει τη δυνατότητα αποθήκευσης μόνο τριών εγγραφών, τότε ο κόμβος D1 πρέπει να διασπαστεί. Η διάσπαση θα είναι χρονική και κατά την τρέχουσα χρονική στιγμή t=5.

D1	D2
A,[0,4]	B,[5,]
B,[0,]	A,[5,]
A,[4,]	C,[5,]

3.2 Η δομή του TShB$^{\pi}$-tree

Όπως ήδη αναφέρθηκε, η δομή του TShB$^{\pi}$-tree δεν διαφέρει από αυτή του hB$^{\pi}$-tree. Το δένδρο αποτελείται από κόμβους δεδομένων και κόμβους ευρετηρίου. Όπως και στο hB$^{\pi}$-tree κάθε κόμβος είναι υπεύθυνος για ένα τμήμα του *k*-διάστατου χώρου, με κάθε διάσταση να αντιπροσωπεύει ένα από τα κλειδιά ευρετηριοποίησης. Στην προκειμένη περίπτωση υπάρχει και μία επιπλέον διάσταση, αυτή του χρόνου. Σε κάθε κόμβο αποθηκεύονται μια σειρά από πληροφορίες. Μια από αυτές αφορά την περιοχή για τη οποία είναι υπεύθυνος ο κόμβος και υλοποιείται με την

αποθήκευση ενός διαστήματος τιμών για κάθε χαρακτηριστικό ευρετηριοποίησης και συνεπώς και για το χρόνο. Οι υπόλοιπες πληροφορίες, σε ότι αφορά τους κόμβους ευρετηρίου, είναι το kd-tree που οργανώνει το χώρο για τον οποίο είναι άμεσα ή έμμεσα υπεύθυνος ο κόμβος, ενώ για τους κόμβους δεδομένων οι εγγραφές των δεδομένων και προαιρετικά ένα kd-tree που οργανώνει εσωτερικά τις λίστες εγγραφών.

Στους κόμβους του kd-tree αποθηκεύεται πλέον και το χαρακτηριστικό του χρόνου, όπως κάθε άλλο χαρακτηριστικό ευρετηριοποίησης. Όλες οι άλλες πληροφορίες που διατηρούνται στο kd-tree παραμένουν όπως και στο hB^{π}-tree.

3.3 Αλγόριθμοι διάσπασης

Όπως και στο hB^{π}-tree κάθε δομική τροποποίηση του δένδρου είναι μια ξεχωριστή *ατομική ενέργεια* (atomic action). Τροποποιήσεις που μπορεί να γίνουν είναι η διάσπαση ενός κόμβου και η ενημέρωση του γονικού κόμβου με τον κατάλληλο όρο ευρετηρίου. Επειδή κάθε εγγραφή σβήνεται από τη βάση μόνο λογικά και στην πραγματικότητα τα στοιχεία της παραμένουν αποθηκευμένα στη βάση δεδομένων, η διαδικασία της συγχώνευσης των κόμβων δεν βρίσκει εφαρμογή στη νέα δενδρική δομή.

Όπως είπαμε, όταν ένας κόμβος υπερχειλίσει πρέπει να διασπαστεί. Στη συνέχεια δρομολογείται η ενημέρωση του γονικού κόμβου με τον όρο ευρετηρίου. Η διάσπαση του κόμβου μπορεί να γίνει είτε με βάση το χρόνο είτε με βάση κάποιο κλειδί ευρετηριοποίησης.

Διάσπαση με βάση το χρόνο (time split)

Σε ότι αφορά τους κόμβους δεδομένων, ως χρόνος διάσπασης επιλέγεται πάντα ο τρέχων χρόνος. Όταν ένας κόμβος δεδομένων διασπάται με βάση το χρόνο τότε δημιουργούνται δύο κόμβοι. Ο ένας από αυτούς περιέχει τις μη ενεργές εκείνη τη χρονική στιγμή εγγραφές ενώ στο νέο κόμβο τοποθετούνται οι ενεργές εγγραφές. Ωστόσο στον ιστορικό κόμβο δεν υπάρχει δυνατότητα να προστεθεί κάποια άλλη εγγραφή και συνεπώς κατά τη στιγμή της διάσπασης όσος χώρος του ιστορικού κόμβου μείνει κενός θα παραμείνει κενός για πάντα. Αυτό σημαίνει ότι θα πρέπει να βρούμε ένα χρόνο διάσπασης που να αφήνει τον ιστορικό κόμβο με όσο το δυνατόν λιγότερο κενό χώρο και αυτός ο χρόνος διάσπασης δεν άλλος παρά ο τρέχων χρόνος.

Αφού γίνει η διάσπαση του κόμβου δεδομένων, αναζητούνται όλες οι ενεργές εγγραφές. Σε κάθε μία από αυτές τροποποιείται ο τελικός χρόνος που εξισώνεται με τον τρέχοντα χρόνο. Παράλληλα οι εγγραφές αυτές αντιγράφονται στο νέο κόμβο με αρχικό χρόνο ίσο με τον τρέχοντα χρόνο. Με αυτό τον τρόπο όλοι οι ιστορικοί κόμβοι παραμένουν πλήρεις.

Σε ότι αφορά τους κόμβους ευρετηρίου, η διάσπαση γίνεται με τη βοήθεια του kd-tree και την αποκοπή κάποιου παρασημοφορεμένου υποδένδρου. Καθαρή διάσπαση με βάση το χρόνο (κάποιον παρελθοντικό χρόνο συναλλαγής) γίνεται μόνο εάν όλοι οι kd-tree κόμβοι στη διαδρομή από τη ρίζα του kd-tree έως το υποψήφιο προς αποκοπή υποδένδρο έχουν ως χαρακτηριστικό το χρόνο. Όπως και στο hB^{π}-tree αναζητείται εκείνο το υποδένδρο του οποίου το μέγεθος είναι περίπου ίσο με το μισό του συνολικού μεγέθους του kd-tree.

Διάσπαση με βάση το κλειδί (key split)

Στην περίπτωση που ένας κόμβος δεδομένων περιέχει ένα μεγάλο ποσοστό ενεργών εγγραφών τότε κάνοντας διάσπαση με βάση το χρόνο οι περισσότερες εγγραφές του κόμβου μεταφέρονται και στον νέο κόμβο, με αποτέλεσμα αυτός σχεδόν να γεμίσει. Σε αυτή την περίπτωση είναι πιο αποδοτικό να διασπάσουμε τον κόμβο με βάση ένα από τα κλειδιά ευρετηριοποίησης. Το κλειδί που χρησιμοποιείται για μια διάσπαση βρίσκεται αποθηκευμένο στον κόμβο δεδομένων και αλλάζει σύμφωνα με τον αλγόριθμο round-robin μετά από κάθε διάσπαση.

Στην περίπτωση των κόμβων ευρετηρίου η διάσπαση γίνεται όπως και στην περίπτωση της διάσπασης με βάση το χρόνο μόνο που η παρασημοφορεμένη ρίζα του υποδένδρου αυτή τη φορά θα έχει ως χαρακτηριστικό ένα από τα κλειδιά ευρετηριοποίησης.

Στη συνέχεια θα παρουσιαστούν δύο διαφορετικοί αλγόριθμοι διάσπασης των κόμβων και ενημέρωσης του γονικού κόμβου με τον όρο ευρετηρίου. Η πολιτική που θα χρησιμοποιηθεί για τη διάσπαση είναι *διάσπαση μόνο σε παρασημοφορεμένους κόμβους* (split at decorations) και για τη ενημέρωση είναι *ενημέρωση με την πλήρη διαδρομή* (full path). Ο πρώτος αλγόριθμος είναι αυτούσιος ο αλγόριθμος D/fp [Evangelidis, Lomet, Salzberg 1997] ενώ ο δεύτερος μία παραλλαγή αυτού. Και οι δυο αλγόριθμοι αποτρέπουν το hB$^{\pi}$-tree να πάρει τη μορφή DAG.

Χάριν ευκολίας, θα θεωρήσουμε ότι έχουμε δύο χαρακτηριστικά ευρετηριοποίησης, ένα κλειδί x και το χρόνο t. Επίσης, για την καλύτερη κατανόηση των αλγορίθμων αλλά και μεγαλύτερη ευκολία στη μεταξύ τους σύγκριση θα χρησιμοποιήσουμε το ίδιο παράδειγμα, σύμφωνα με το οποίο αρχικά υπάρχει ένας μόνο κόμβος δεδομένων, ο D1, και πραγματοποιείται η παρακάτω σειρά διασπάσεων:

1. Ο κόμβος D1 διασπάται με βάση το χρόνο στο t12 και παίρνουμε τους κόμβους δεδομένων D1, D2 και τον κόμβο ευρετηρίου I1 (νέα ρίζα).

2. Ο κόμβος D2 διασπάται με βάση το κλειδί στο xa και παίρνουμε τους κόμβους D2, D3 και ενημερώνουμε τον κόμβο I1.

3. Ο κόμβος D2 διασπάται με βάση το χρόνο στο και t20 και παίρνουμε τους κόμβους D3, D4 και ενημερώνουμε τον κόμβο I1.

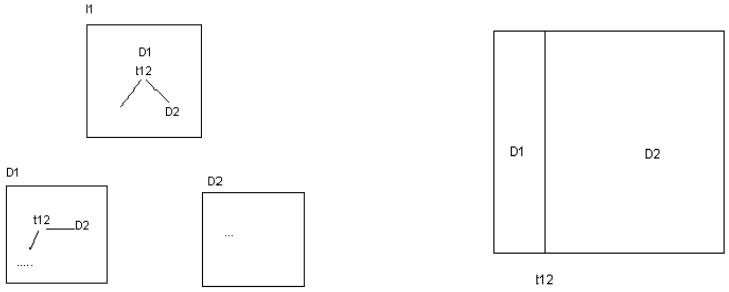
4. Τέλος, ο κόμβος D3 διασπάται με βάση το χρόνο στο t25 και παίρνουμε τους κόμβους D3, D5 και ενημερώνουμε και πάλι τον κόμβο I1.

Το παράδειγμα περιορίζεται μόνο στη διαδικασία διάσπασης των κόμβων δεδομένων καθώς η διάσπαση των κόμβων ευρετηρίου δεν διαφέρει από αυτή που περιγράφεται στον αλγόριθμο D/fp [Evangelidis, Lomet, Salzberg 1997].

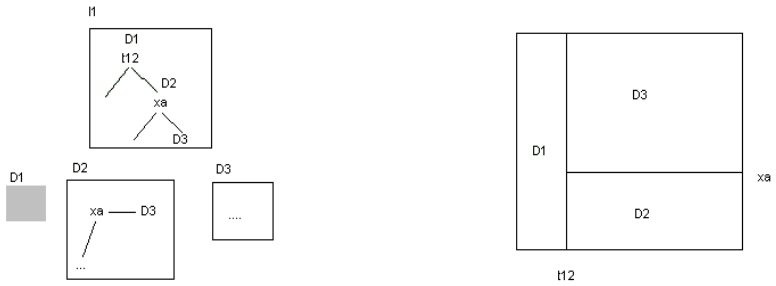
3.2.1 Ο αλγόριθμος D/fp του hB$^{\pi}$-tree

Όπως είπαμε ο πρώτος αλγόριθμος είναι ο αλγόριθμος D/fp [Evangelidis, Lomet, Salzberg 1997]. Με βάση αυτόν τον αλγόριθμο η διάσπαση των κόμβων ευρετηρίου γίνεται μόνο σε κόμβους του kd-tree με παράσημο και μεταφέρεται στο γονικό κόμβο η πλήρης διαδρομή από τη ρίζα του δένδρου ως τη ρίζα του εξαγόμενου υποδένδρου. Η διάσπαση των

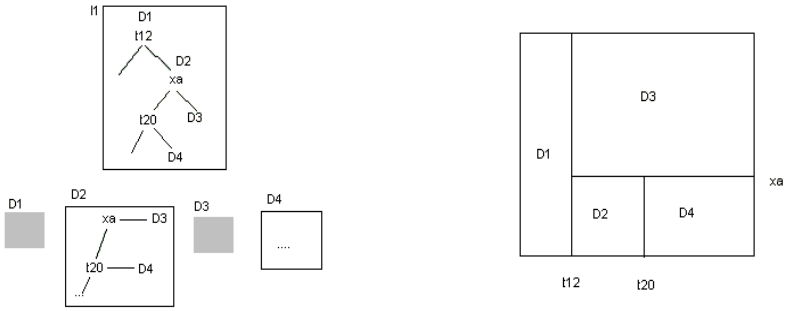
κόμβων δεδομένων γίνεται με βάση κάποιο από τα χαρακτηριστικά ευρετηριοποίησης. Ο χρόνος σε αυτή την περίπτωση αντιμετωπίζεται όπως κάθε άλλο χαρακτηριστικό. Η διάσπαση των κόμβων ευρετηρίου μόνο σε παρασημοφορεμένους κόμβους του kd-tree οδηγεί σε κόμβους με ένα μόνο γονικό κόμβο (single parent nodes) γεγονός που απλοποιεί περισσότερο την υλοποίηση του αλγορίθμου, καθώς για κάθε διάσπαση κόμβου ενημερώνεται μόνο ένας κόμβος του υψηλότερου επιπέδου. Στις παρακάτω εικόνες (3, 4, 5 και 6) με γκρι χρώμα εμφανίζονται οι κόμβοι που δεν τροποποιούνται κατά το συγκεκριμένο βήμα. Στο αριστερό τμήμα κάθε σχήματος εμφανίζεται το hB^{π}-tree ύστερα από κάθε ενέργεια (οι ενέργειες αναφέρονται στην προηγούμενη ενότητα) ενώ στο δεξί ο χώρος υπευθυνότητας κάθε κόμβου δεδομένων.



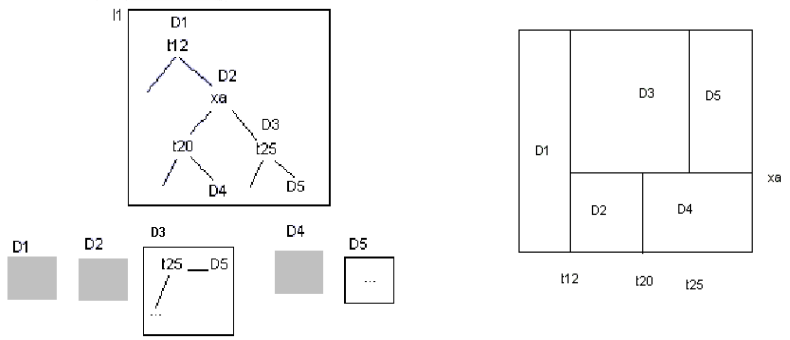
Εικόνα 3 Διάσπαση του κόμβου D1



Εικόνα 4 Διάσπαση του κόμβου D2



Εικόνα 5 Δεύτερη διάσπαση του κόμβου D2



Εικόνα 6 Διάσπαση του κόμβου D3

Ας εξετάσουμε αν το TShB$^{\pi}$-tree είναι καλά σχηματισμένο με τη βοήθεια μιας αναζήτησης. Έστω ότι αναζητούμε τη εγγραφή με τιμή κλειδιού xb>xa που ήταν ενεργή κατά τη χρονική στιγμή t21. Αν η εγγραφή αυτή υπάρχει τότε σύμφωνα με το σχήμα 3.4 θα βρίσκεται στη λίστα εγγραφών του κόμβου D3. Η αναζήτηση ξεκινά από τη ρίζα του TShB$^{\pi}$-tree και συγκεκριμένα από τη ρίζα του kd-tree αυτής. Επειδή t21>t12 κινούμαστε δεξιά. Στη συνέχεια συγκρίνουμε την τιμή του κόμβου xa με το κλειδί αναζήτησης xb και ξανά κινούμαστε δεξιά ενώ τέλος ακολουθούμε τον αριστερό δείκτη στον επόμενο kd-tree κόμβο καθώς t21<t25. Ο δείκτης αυτός οδηγεί στον κόμβο δεδομένων D3 κι από εκεί στη λίστα δεδομένων του κόμβου αυτού καθώς t21<t25.

3.2.2 Ο αλγόριθμος D/fp χωρίς καθαρές διασπάσεις με βάση το κλειδί

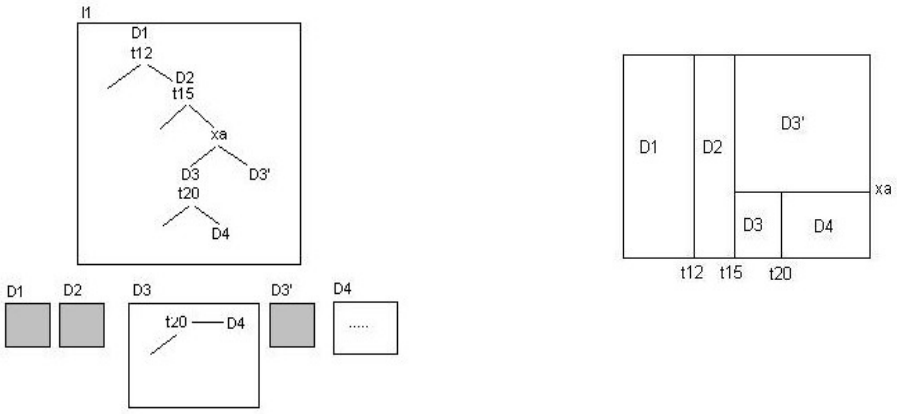
Σε αυτή την παραλλαγή του αλγορίθμου η διάσπαση των κόμβων δεδομένων με βάση το χρόνο γίνεται όπως αναφέρθηκε παραπάνω. Το ίδιο και η διάσπαση των κόμβων ευρετηρίου. Αυτό που διαφέρει από τον προηγούμενο αλγόριθμο είναι η διάσπαση των κόμβων δεδομένων με βάση το κλειδί. Τώρα, δεν επιτρέπεται καθαρή διάσπαση κόμβων δεδομένων με βάση το κλειδί. Σε κάθε τέτοια περίπτωση πρώτα διασπάται ο κόμβος με βάση τον τρέχοντα χρόνο και στη συνέχεια με βάση τις τιμές του κλειδιού που επιλέγεται κάθε φορά Το γεγονός αυτό οδηγεί στη δημιουργία δύο κόμβων δεδομένων μετά από κάθε διάσπαση με βάση το κλειδί, ενώ ο παλιός κόμβος γίνεται ιστορικός.

Μια νέα ιδιότητα του TShB$^{\pi}$-tree όταν αυτό δημιουργείται με βάση αυτόν τον αλγόριθμο είναι το γεγονός ότι όλοι οι δείκτες σε ενεργούς κόμβους δεδομένων βρίσκονται στα φύλλα του kd-tree του TShB$^{\pi}$-tree κόμβου του υψηλότερου επιπέδου, πράγμα που μετατρέπει τον αλγόριθμο ενημέρωσης με τον όρο ευρετηρίου σε απλή επικόλληση του όρου ευρετηρίου σε ένα από τα φύλλα του kd-tree. Μια επίσης πολύ σημαντική ιδιότητα που αποκτά το TShB$^{\pi}$-tree είναι το γεγονός ότι ακόμη κι αν δεν διασπάμε τους κόμβους ευρετηρίου αποκόπτοντας παρασημοφορεμένα υποδένδρα του kd-tree κανένας τρέχων κόμβος δεν έχει πολλούς γονιούς (δεν έχουμε δηλαδή DAG).

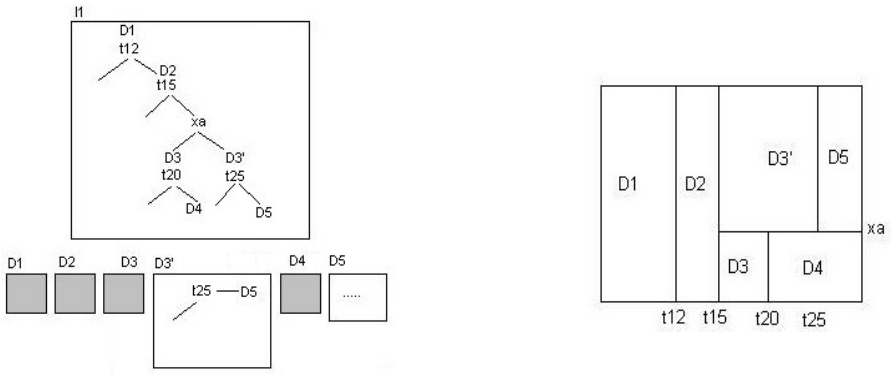
Στη συνέχεια παρουσιάζεται το TShB$^{\pi}$-tree (Εικόνες 7, 8, 9 και 10), όπως αυτό διαμορφώνεται ακολουθώντας τη σειρά των ενεργειών που ορίστηκαν στην προηγούμενη ενότητα και εφαρμόζοντας το νέο αλγόριθμο.

Εικόνα 7 Διάσπαση του κόμβου D1

Εικόνα 8 Διάσπαση του κόμβου D2



Εικόνα 9 Δεύτερη διάσπαση του κόμβου D2



Εικόνα 10 Διάσπαση του κόμβου D3

Ας κάνουμε την αναζήτηση της προηγούμενης ενότητας. Τώρα, η εγγραφή αυτή θα πρέπει να βρίσκεται στο κόμβο D3'. Πράγματι, ακολουθώντας τις διαδρομές των kd-tree της ρίζας και του κόμβου D3' καταλήγουμε στη λίστα εγγραφών του κόμβου D3'.

4. Ανακεφαλαίωση και Συμπεράσματα

Στην παρούσα εργασία περιγράψαμε κάποιες αλλαγές στη δομή του hB$^{\pi}$-tree καθώς και ένα εναλλακτικό αλγόριθμο από τον D/fp που μπορεί να χρησιμοποιηθεί για την διάσπαση των κόμβων και τη ενημέρωση των γονικών κόμβων με τον όρο ευρετηρίου. Η νέα δομή που δημιουργήθηκε υποστηρίζει εκτός των πολλών διαστάσεων των χωρικών δεδομένων και τη διάσταση του χρόνο. Η δομή αυτή ονομάστηκε Time Split hB$^{\pi}$-tree.

Βραχυπρόθεσμος στόχος μας είναι η υλοποίηση των αλγορίθμων διάσπασης που περιγράφηκαν και η σύγκριση της απόδοσής τους. Κάτι πολύ σημαντικό, επίσης, που σκοπεύουμε να κάνουμε είναι η σύγκριση της απόδοσης του TShB$^{\pi}$-tree με άλλες μεθόδους ευρετηριοποίησης που έχουν προταθεί έως σήμερα για το χειρισμό χωροχρονικών δεδομένων.

Τέλος, ο αλγόριθμος D/fp είναι πολύ περιοριστικός στην επιλογή των τιμών διάσπασης. Αυτό συνήθως οδηγεί σε κόμβους χαμηλού utilization. Θα μπορούσε να εξεταστούν οι μετατροπές στη δομή του δένδρου καθώς και στον τρόπο διάσπασης των κόμβων με τη χρήση κάποιου άλλου αλγορίθμου διάσπασης και ενημέρωσης, λιγότερο αυστηρού, όπως για παράδειγμα του A/fp.

Αναφορές

Dyreson C., Grandi F., Kafer W., Kline N., Lorenzos N., Mitsopoulos Y., Montanari A., Nonen D., Peressi E., Pernici B., Roddick J.F., Sarda N.L., Scalas M.R., Segen A., Snodgrass R.T., Soo,M.D., Tansel A., Tiberio P., Wiederhold G., Jensen C.S., Eds. 1994. A consensus glossary of temporal database concepts. *SIGMOD Rec.23, 1(Mar. 1994), 52-64.*

Eatson M.C., 1986. Key-sequence data sets on indelible storage. *IBM J. Res. Dev. 30,* 3 (May 1986), 230-241.

Evangelidis G., Lomet D., Salzberg B. 1997. The hB$^{\pi}$-tree: a multiattribute index supporting concurrency, recovery and node consolidation. *VLDB J, 6,* 1-31.

Kolovson C., Stonebraker M. 1989. Indexing techniques for historical databases. In *Proceedings of the 5th IEEE International Conference on Data Engineering* (Los Angeles, CA, Feb. 1989), 127-137.

Kolovson C., Stonebraker M. 1991. Segment indexes: Dynamic indexing techniques for multi-dimensional interval data. In *Proceedings of th 1991 ACM SIGMOD International Conference on Management of Data* (SIGMOD '91, Denver, CO, May 29-31, 1991), J.Clifford and R.King, Eds. ACM Press, New York, NY, 138-147.

Lanka S., Mays E. 1991. Fully persistent B+trees. In *Proceedings of the 1991 ACM SIGMOD International Conference on Managemnt of Data* (SIGMOD '91, Denver, CO, May 29-31, 1991), J.Clifford and R.King, Eds. ACM Press, New York, NY, 426-435.

Lomet D., Salzberg B. 1989. Access Methods for multiversion data. In *Proceedings of the 1989 ACM SIGMOD International Conference on Management of Data (*SIGMOD '89, Portland, OR, June 1989), J.Clifford, J.Clifford, B.Lindsay, D.Maier and J.Clifford, Eds. ACM Press, New York, NY, 315-324.

Lomet D., Salzberg B. 1990.The hB-tree:a multiattribute indexing method with good guaranteed performance. *ACM Trans. Database Syst. 15,* 4 (Dec. 1990), 625-658.

Lomet D., Salzberg B. 1993.Transaction-time databases In *Temporal Databases: Theory, Design, and Implementation,* A.Tansel, J.Clifford, S.Gadia, S.Jajodia,A.Segen and R Snodgrass, Eds. Benjamin/Cummings, Redwood City, CA.

Lorentzos N.A., Johnson R.G. 1988. Extending relational algebra to manipulate temporal data. *Inf. Syst. 13,* 3 (Oct., 1, 1988), 289-296.

Manolopoulos Y., Kapetanakis G., 1990. Overlapping B+trees for temporal data In *Proceedings of 5th Conferenceon JCIT* (JCIT, Jerusalem, Oct. 22-25), 491-498.

Navathe S.B., Ahmed R. 1989. A temporal relational model and query language. *Inf. Sci. 49,* 1, 2 & 3 (Oct./Nov.Dec. 1989), 147-175.

Ozsoyoglu G., Snodgrass R. 1995. Temporal and real time databases: A survey. *IEEE Trans. Knowl. Data Eng. 7,* 4 (Aug.), 513-532.

Salzberg B, 1994. Timestamping after comit. In *Proceedings of the 3rd International Conference on the Parallel and Distributed Information Systems* (PDIS, Austin, TX, Sept.), 160-167.

Salzberg B, Tsotras V.J., 1999. Comparison of Access Methods for Time-Evolving Data. *ACM Computing Surveys, 31,* 2 (Jun. 1999), 158-162.

Snodgrass R.T., Ahn I. 1985. A taxonomy of time in databases. In *Proceeding of the ACM SIGMOD Conference on Management of the Data.* ACM Press, New York, NY, 236-246.

Snodgrass R.T., Ahn I. 1986. Temporal Databases. *IEEE Comput., 19, 9* (Sept.1986), 35-41.

Stonebraker M. 1987. The design of the Postgres storage system. In *Proceedings of the 13th Conference of Very Large Data Bases* (Brighton, England, Sept., 1987). VLDB Endowment, Berkeley, CA, 289-300.

Tsotras V.J., Kumar A. 1996. Temporal database bibliography update. *SIGMOD Rec. 25, 1* (Mar.1996), 41-51.

Verman P., Verma R., 1997.An efficient multiversion access structure. *IEEE Trans. Knowl. Data Eng. 9,* 3 (May/June), 391-409.