Κεφάλαιο 9

ΣΤΑΤΙΚΑ ΤΥΧΑΙΑ ΑΡΧΕΙΑ

Κεφάλαιο 9

ΣΤΑΤΙΚΑ ΤΥΧΑΙΑ ΑΡΧΕΙΑ

9.1 Εισαγωγή

Τα **τυχαία ή άμεσα** (random, direct) αρχεία είναι μία πολύ σπουδαία δομή. Αν τα αρχεία αυτά είναι σωστά σχεδιασμένα, τότε αποτελούν την ιδανική οργάνωση για on-line εφαρμογές, γιατί εντοπίζουν τη ζητούμενη εγγραφή με πολύ λίγες προσπελάσεις (ίσως και μόνο μία). Το βασικό μειονέκτημα της δομής αυτής είναι ότι δεν προσφέρεται για σειριακή ανάκτηση ταξινομημένων εγγραφών κατά οποιοδήποτε κλειδί.

Η βασική ιδέα της μεθόδου είναι η εφαρμογή κάποιου απλού αλγεβρικού μετασχηματισμού στην τιμή του κλειδιού, ώστε να προκύψει η απόλυτη διεύθυνση της εγγραφής. Η απόλυτη διεύθυνση εξαρτάται από το υλικό και αποτελείται από την τριάδα: αριθμός κυλίνδρου, αριθμός ατράκτου και αριθμός εγγραφής. Ωστόσο, είναι δυνατόν η μέθοδος να εφαρμοσθεί και κατά τρόπο που να μην εξαρτάται από το υλικό.

Τα κλειδιά λέγονται **πυκνά** (dense) όταν οι τιμές τους είναι συνεχόμενες. Στην απλούστερη περίπτωση πυκνών κλειδιών οι τιμές τους αρχίζουν από τη μονάδα. Για παράδειγμα, έστω ότι οι τιμές των κλειδιών είναι από 1 ως 5000 και ότι η χωρητικότητα των σελίδων είναι μία εγγραφή. Στην περίπτωση αυτή δημιουργείται ένα αρχείο των 5000 σελίδων, οπότε η i-οστή εγγραφή θα αποθηκευθεί στην $(i-1)$-οστή σελίδα. Είναι, επίσης, δυνατόν οι τιμές των πυκνών κλειδιών να μην αρχίζουν από τη μονάδα αλλά, για παράδειγμα, να κυμαίνονται από το 10.001 ως το 15.000. Τότε αφαιρώντας μία σταθερή τιμή από την τιμή του κλειδιού (στη συγκεκριμένη περίπτωση το

10.000), και πάλι βρίσκεται εύκολα η κατάλληλη σελίδα του αρχείου. Τα αρχεία αυτά λέγονται **αυτοδεικτοδοτημένα** (self-indexing) και υλοποιούνται απλούστατα σε όλες τις γλώσσες προγραμματισμού.

Αν τα κλειδιά δεν είναι πυκνά, τότε με την τεχνική αυτή το μεγαλύτερο μέρος του αρχείου θα παραμείνει κενό και θα σπαταληθεί δευτερεύουσα μνήμη. Η περίπτωση αυτή αντιμετωπίζεται μετασχηματίζοντας την τιμή του κλειδιού, έτσι ώστε το διάστημα των κλειδιών να αντιστοιχεί σε ένα πολύ μικρότερο διάστημα τιμών διευθύνσεων. Η μέθοδος αυτή ονομάσθηκε αρχικά **μετασχηματισμός του κλειδιού σε διεύθυνση** (key-to-address transformation), αλλά τελικά επικράτησε ο όρος **κατακερματισμός** (hashing), και γι΄ αυτό τα τυχαία αρχεία λέγονται και **αρχεία κατακερματισμού** (hashed files).

Κατά τη φόρτωση του αρχείου δεσμεύεται κάποιος χώρος του δίσκου, που ονομάζεται κύρια περιοχή. Σε μερικές παραλλαγές της μεθόδου προβλέπεται η μελλοντική χρήση περιοχής υπερχείλισης, που συνήθως βρίσκεται στον ίδιο κύλινδρο με την κύρια περιοχή, ώστε να μην υπάρχει χρονικό κόστος εντοπισμού. Όμως σε άλλες παραλλαγές δεν προβλέπεται ξεχωριστή περιοχή για τις εγγραφές υπερχείλισης. **Παράγοντας διευθύνσεων** (address factor) είναι το κλάσμα του μεγέθους της κύριας περιοχής προς το συνολικό μέγεθος του αρχείου. Αν ο χώρος που έχει κρατηθεί από το αρχείο δεν αυξομειώνεται με την πάροδο του χρόνου, τότε το αρχείο είναι στατικό. Όμως η έρευνα από το 1980 και εντεύθεν κατέληξε σε μία σειρά δομών τυχαίων αρχείων που μεγεθύνονται και συρρικνώνονται, ανάλογα με τον αριθμό των εισαγωγών και διαγραφών εγγραφών αντίστοιχα. Δηλαδή τα αρχεία αυτά είναι δυναμικά και θα εξετασθούν στο επόμενο κεφάλαιο.

9.2 Συναρτήσεις κατακερματισμού

Ο μετασχηματισμός της τιμής του κλειδιού σε τιμή διεύθυνσης επιτυγχάνεται με τη βοήθεια της **συνάρτησης κατακερματισμού** (hashing function) που σκοπό έχει την αντιστοίχιση του πεδίου τιμών των κλειδιών στο πεδίο των τιμών διευθύνσεων κατά τυχαίο τρόπο. Καλή συνάρτηση είναι εκείνη που διασπείρει τις εγγραφές σε όλη την έκταση του αρχείου, ακόμη και αν οι τιμές των κλειδιών συγκεντρώνονται σε μερικές περιοχές του πεδίου τιμών των κλειδιών. Για το λόγο αυτό η μέθοδος του κατακερματισμού ονομάζεται επίσης και **τεχνική διασκορπισμού αποθήκευσης** (scatter storage

technique). Αν τα κλειδιά είναι αριθμητικά, τότε μπορεί να εφαρμοσθεί μία απλή αλγεβρική έκφραση. Αν τα κλειδιά είναι αλφαβητικά ή αλφαριθμητικά, τότε πρέπει να μετατραπούν πρώτα σε αριθμητικά. Η μετατροπή αυτή μπορεί να γίνει χρησιμοποιώντας κάποια εσωτερική αναπαράσταση των χαρακτήρων των κλειδιών, όπως τους κώδικες ASCII, EBCDIC κλπ. Έτσι για παράδειγμα, το κλειδί C1 μπορεί να μετατραπεί σε 6749 (67 είναι ο κώδικας ASCII για το C και 49 είναι ο κώδικας για το 1).

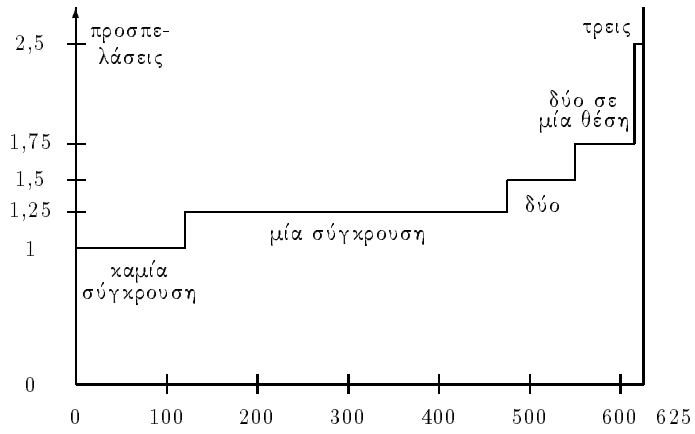
Το πρόβλημα των συναρτήσεων κατακερματισμού είναι ότι δύο ή περισσότερες διαφορετικές τιμές κλειδιών μπορεί να δώσουν την ίδια διεύθυνση. Αυτό το φαινόμενο καλείται **σύγκρουση** (collision), ενώ οι εγγραφές που συγκρούονται λέγονται **συνώνυμα** (synonyms). Αν από μία συνάρτηση προκύπτει σημαντικός αριθμός εγγραφών που συνωστίζονται στην ίδια περιοχή του αρχείου, τότε λέγεται ότι παρουσιάζεται **πρωτεύουσα συγκέντρωση** (primary clustering). Έτσι, συνήθως, δεσμεύεται για το αρχείο περισσότερος χώρος από την αρχική πρόβλεψη σχετικά με το τελικό πλήθος των εγγραφών. Στη βιβλιογραφία αναφέρονται πολλά χαρακτηριστικά παραδείγματα που δείχνουν ανάγλυφα ότι το πρόβλημα των συγκρούσεων εμφανίζεται με βεβαιότητα. Στη συνέχεια αναφέρονται δύο τέτοια παραδείγματα.

Έστω ότι οι 31 πιο συνηθισμένες αγγλικές λέξεις (and, or, not, the, κλπ.) αποθηκεύονται σε ένα πίνακα 42 θέσεων. Το σύνολο των δυνατών τοποθετήσεων είναι 10^{50}, ενώ οι βολικές τοποθετήσεις (δηλαδή χωρίς σύγκρουση) είναι μόνο 10^{43}. Άρα αντιστοιχεί μία βολική τοποθέτηση στις δέκα εκατομμύρια περιπτώσεις. Το δεύτερο παράδειγμα είναι το λεγόμενο 'παράδοξο των γενεθλίων'. Έστω, ότι επιλέγονται τυχαία από το πλήθος 23 άτομα. Η πιθανότητα τουλάχιστο δύο άτομα από το σύνολο των 23 ατόμων να έχουν την ίδια μέρα γενέθλια είναι 50.83%.

Το φαινόμενο των συγκρούσεων θα μοντελοποιηθεί μαθηματικά. Αν n εγγραφές πρέπει να αποθηκευθούν σε m θέσεις, τότε ο αριθμός των δυνατών τοποθετήσεων είναι m^n. Για παράδειγμα, αν n=4 και m=5 τότε ο αριθμός των τοποθετήσεων είναι 625. Στις βολικές τοποθετήσεις, που είναι $\frac{m!}{(m-n)!} = 120$, κάθε εγγραφή ανακτάται με μία προσπέλαση. Στη χειρότερη περίπτωση σε μία θέση αντιστοιχούν και οι τέσσερις εγγραφές, οπότε κάθε εγγραφή ανακτάται με $(n+1)/2$=2,5 προσπελάσεις κατά μέσο όρο. Ο αριθμός αυτών των δυσμενών περιπτώσεων είναι m=5. Οι υπόλοιπες 500 τοποθετήσεις διακρίνονται σε τρία είδη: μία σύγκρουση δύο εγγραφών, δύο συγκρούσεις δύο εγγραφών και μία σύγκρουση τριών εγγραφών.

Η πιθανότητα να εμφανισθεί μία βολική τοποθέτηση είναι $p_0 = \frac{\frac{m!}{(m-n)!}}{m^n} =$

0.192, οπότε η μέση τιμή των προσπελάσεων για εγγραφές υπερχείλισης είναι o_0=0. Η πιθανότητα να εμφανισθεί η πιο δυσμενής τοποθέτηση είναι $p_{max} = \frac{m}{m^n} = 0.008$, οπότε ο αριθμός των συγκρούσεων και των προσπελάσεων για εγγραφή υπερχείλισης είναι c=n–1=3 και o_{max}=1+2+3 αντίστοιχα. Στο Σχήμα 9.1. παρουσιάζονται τα στοιχεία των υπολοίπων τριών περιπτώσεων. Κατά μέσο όρο απαιτούνται $p = \sum_c p_c o_c = 0,3$ επιπρόσθετες προσπελάσεις για την ανάκτηση μίας εγγραφής. Η μέση τιμή των εγγραφών που συγκρούσθηκαν και δεν αποθηκεύθηκαν στην κατάλληλη θέση είναι $o = \sum_c p_c c = 1,05$ εγγραφές, άρα κατά μέσο όρο n–o=2,95 εγγραφές αποθηκεύθηκαν στο κύριο αρχείο. Η πιθανότητα να συγκρουσθεί η επόμενη εγγραφή που θα εισαχθεί στο αρχείο είναι 1–2,95/5=0,41. Για άλλες τιμές των m και n το σχήμα θα είναι παρόμοιο. Ο αριθμός των βημάτων αυξάνει πολύ γρήγορα για μεγαλύτερα m και n και η συνάρτηση ομαλοποιείται.



Σχήμα 9.1: Κατανομή συγκρούσεων και προσπελάσεων.

Στη βιβλιογραφία αναφέρονται πολλές μέθοδοι για την εύρεση συναρτήσεων κατακερματισμού που δεν παράγουν συνώνυμα. Οι συναρτήσεις αυτές λέγονται **τέλειες** (perfect) και συνήθως χρησιμοποιούνται στην πράξη σε μικρούς πίνακες της κύριας μνήμης για ειδικές εφαρμογές όπως, για παράδειγμα, σε μεταφραστές για αποθήκευση δεσμευμένων λέξεων, σε επεξεργασία φυσικής γλώσσας για φιλτράρισμα λέξεων υψηλής συχνότητας κλπ. Αν μία τέλεια συνάρτηση δεσμεύει τον ελάχιστο δυνατό χώρο, τότε λέγεται **ελάχιστη** (minimal). Μία αναγκαία προϋπόθεση των τέλειων μεθόδων κατακερματισμού είναι ότι συνήθως απαιτούν την εκ των προτέρων γνώση

των τιμών των κλειδιών. Ο αναγνώστης μπορεί να βρει περισσότερα στοιχεία για τις τέλειες μεθόδους κατακερματισμού στην κύρια μνήμη στο βιβλίο για Δομές Δεδομένων. Σχετικά πρόσφατα έχουν προταθεί τέλειες μέθοδοι κατακερματισμού για περιπτώσεις ογκωδών στατικών αλλά και δυναμικών αρχείων. Οι μέθοδοι αυτές αποτελούν μία κλάση μεθόδων που ονομάζονται **εξωτερικός τέλειος κατακερματισμός** (external perfect hashing) και δεν θα εξετασθούν στη συνέχεια λόγω της εξαιρετικής πολυπλοκότητάς τους.

Ο λόγος των κατειλημμένων θέσεων προς το σύνολο των θέσεων του αρχείου ονομάζεται **παράγοντας φόρτισης** (load factor, Lf), και ισούται με:

$$Lf \;=\; \frac{n}{bk_m \times Bkfr}$$

όπου bk_m είναι ο αριθμός των κάδων στην κύρια περιοχή του αρχείου. Υπάρχει άμεση σχέση της τιμής του παράγοντα φόρτισης και της συχνότητας των συγκρούσεων. Όσο μικρότερος είναι ο παράγοντας φόρτισης, τόσο μικρότερη είναι η πιθανότητα σύγκρουσης, και το αντίστροφο. Αρμοδιότητα του σχεδιαστή των αρχείων είναι να βρει την ισορροπία μεταξύ του πλήθους των συγκρούσεων και του ποσοστού αχρησιμοποίητου χώρου του αρχείου.

Υπερχείλιση συμβαίνει όταν μία εγγραφή πρέπει να αποθηκευθεί σε ένα πλήρη κάδο. Τότε η εγγραφή κατευθύνεται για αποθήκευση σε άλλον κάδο. Όσο μεγαλύτερος είναι ο κάδος, τόσο μικρότερη είναι η πιθανότητα να υπάρχει υπερχείλιση. Ωστόσο, όσο μεγαλύτερο είναι το μέγεθος του κάδου, τόσο πιο χρονοβόρα είναι η προσπέλαση στο δίσκο. Άρα, και σ΄ αυτό το σημείο πρέπει να βρεθεί μία ισορροπία.

Η στατιστική κατανομή που χαρακτηρίζει τη διανομή των εγγραφών στους κάδους είναι η διωνυμική. Αν οι εγγραφές μοιράζονται τυχαία στους κάδους, τότε η πιθανότητα να έχουν κατευθυνθεί σε ένα κάδο i από τις n εγγραφές είναι:

$$P(i) \;=\; \frac{n!}{i! \times (n-i)!} \times \left(\frac{1}{bk}\right)^{i} \times \left(1 - \frac{1}{bk}\right)^{n-i}$$

Για μεγάλες τιμές των n και bk η κατανομή αυτή μπορεί να προσεγγισθεί από την υπολογιστικά βολικότερη κατανομή Poisson:

$$P(i) \;=\; \frac{e^{-n/bk} \times \left(\frac{n}{bk}\right)^{i}}{i!}$$

Η μέση τιμή των εγγραφών υπερχείλισης σε ένα κάδο είναι:

$$\sum_{j=1}^{n-Bkfr} j \times P(Bkfr + j)$$

και συνεπώς το ποσοστό των εγγραφών υπερχείλισης είναι:

$$100 \times \frac{bk}{n} \times \sum_{j=1}^{n-Bkfr} j \times P(Bkfr + j)$$

Στον Πίνακα 9.1 δίνεται η μέση τιμή του ποσοστού των εγγραφών που υπερχειλίζουν ως συνάρτηση του μεγέθους του κάδου και του παράγοντα φόρτισης. Το μέγεθος του αρχείου έχει ληφθεί ίσο με 200 κάδους.

Μέγεθος κάδου	Παράγοντας φόρτισης									
	50%	55%	60%	65%	70%	75%	80%	85%	90%	95%
2	10,3	11,9	13,6	15,3	16,9	18,7	20,5	22,0	23,7	25,4
3	5,9	7,3	8,8	10,3	11,9	13,6	15,3	17,0	18,8	20,6
5	2,4	3,4	4,5	5,7	7,1	8,6	10,2	11,9	13,7	15,6
8	0,8	1,3	2,0	2,9	3,9	5,2	6,7	8,3	10,0	11,9
12	0,2	0,5	0,8	1,4	2,1	3,1	4,3	5,8	7,5	9,4
17	0,1	0,2	0,3	0,6	1,1	1,9	2,8	4,1	5,7	7,5
23	0,0	0,0	0,1	0,3	0,6	1,1	1,9	3,0	3,0	6,2

Πίνακας 9.1: Μέσο ποσοστό υπερχείλισης αρχείου.

Κάθε χρήστης μπορεί να εφεύρει μία νέα συνάρτηση κατακερματισμού με σκοπό να τυχαιοποιήσει τα δεδομένα του. Στη συνέχεια παρουσιάζονται μερικές συναρτήσεις που αναφέρονται κατά κόρο στη βιβλιογραφία. Έστω ότι δίνεται μία εγγραφή με κλειδί 172.148 για να αποθηκευθεί σε αρχείο που αποτελείται από 7000 κάδους.

Διαίρεση με πρώτο αριθμό (prime number division)

Η τιμή της διεύθυνσης, όπου θα αποθηκευθεί η εγγραφή, ισούται με το υπόλοιπο της διαίρεσης της τιμής του κλειδιού δια του μεγέθους του αρχείου. Σύμφωνα με μία πρακτική παρατήρηση οι συγκρούσεις ελαχιστοποιούνται αν διαιρέτης είναι ο μεγαλύτερος πρώτος αριθμός που είναι μικρότερος από το μέγεθος του αρχείου. Στην προκειμένη περίπτωση ισχύει: 172148 mod 6997 = 4220. Βέβαια, διαιρέτης μπορεί να είναι οποιοσδήποτε αριθμός αρκεί να

είναι μικρότερος από το πλήθος των διευθύνσεων του αρχείου. Αναφέρεται επίσης από πρακτικές παρατηρήσεις ότι οι συγκρούσεις ελαχιστοποιούνται αν ο διαιρέτης δεν περιέχει πρώτους αριθμούς μικρότερους από 20.

Μετατροπή της ρίζας (radix conversion)

Θεωρείται ότι η τιμή του κλειδιού δεν είναι αριθμός του δεκαδικού συστήματος και επομένως πρέπει να μετατραπεί σε αριθμό του συστήματος αυτού. Έτσι αν υποτεθεί ότι ο συγκεκριμένος αριθμός έχει ως βάση το 11, τότε με τη μετατροπή προκύπτει:

$$1 \times 11^5 + 7 \times 11^4 + 2 \times 11^3 + 1 \times 11^2 + 4 \times 11 + 8 \ = \ 266373$$

Το αποτέλεσμα κανονικοποιείται διαιρώντας δια 6997. Η διεύθυνση του κάδου του αρχείου ισούται με το τελικό αποτέλεσμα που είναι 487.

Μέση του τετραγώνου (mid square)

Σύμφωνα με τη μέθοδο αυτή λαμβάνονται τα μεσαία ψηφία του τετραγώνου της τιμής του κλειδιού. Έτσι, αν η συγκεκριμένη τιμή του κλειδιού υψωθεί στο τετράγωνο, προκύπτει 029634933904. Λαμβάνονται τα 4 μεσαία ψηφία, δηλαδή 3493. Ακολουθεί κανονικοποίηση ως προς το 7000 και προκύπτει το αποτέλεσμα 2445.

Μετακίνηση ή δίπλωση (move, folding)

Πρόκειται για δύο παρόμοιες μεθόδους. Η τιμή του κλειδιού χωρίζεται σε δύο ή περισσότερα τμήματα που προστίθενται. Έστω, ότι δίνεται ο αριθμός 17207359 που χωρίζεται σε δύο τετραψήφιους αριθμούς, τους 1720 και 7359. Σύμφωνα με την πρώτη μέθοδο οι τιμές των τετραψηφίων αριθμών προστίθενται και προκύπτει 9079 (shift folding). Σύμφωνα με τη δεύτερη μέθοδο προστίθενται οι τετραψήφιοι αριθμοί, αφού πρώτα η σειρά των ψηφίων του δεύτερου αριθμού αντιστραφεί (boundary folding). Δηλαδή προστίθεται ο τετραψήφιος 1720 και ο 9537, που προέρχεται από τον 7359. Προκύπτει 11257. Στη συνέχεια ακολουθεί κανονικοποίηση ως προς το μέγεθος του αρχείου. Έτσι στην πρώτη περίπτωση θα προκύψει η διεύθυνση $9079 \times 7000/19.998 = 3178$, ενώ στη δεύτερη περίπτωση θα προκύψει $11.257 \times 7000/19.998 = 3940$.

9.3 Διαχείριση συνωνύμων με ανοικτή διεύθυνση

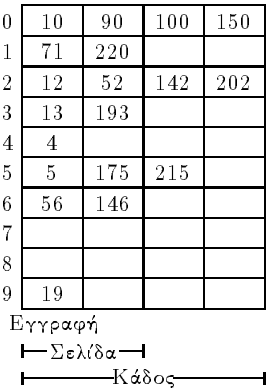
Η διαχείριση των συνωνύμων είναι το πιο κρίσιμο τμήμα κατά τη σχεδίαση ενός αρχείου κατακερματισμού. Διακρίνονται τρεις γενικές κατηγορίες

μεθόδων διαχείρισης της υπερχείλισης: οι μέθοδοι της **ανοικτής διεύθυνσης** (open addressing), οι μέθοδοι με **άμεσες αλυσίδες** (direct chaining) και οι μέθοδοι με **ψευδοαλυσίδες** (pseudochaining). Η βασική διαφορά των μεθόδων της πρώτης κατηγορίας από τις άλλες μεθόδους είναι η μη χρήση δεικτών ως ιδιαίτερων πεδίων στον τύπο της εγγραφής. Στο υποκεφάλαιο αυτό θα παρουσιασθούν παραλλαγές της πρώτης κατηγορίας, ενώ τα επόμενα δύο υποκεφάλαια είναι αφιερωμένα στην εξέταση των δύο άλλων κατηγοριών.

Στην πρώτη, λοιπόν, κατηγορία ανήκουν παραλλαγές όπως η **γραμμική αναζήτηση** (linear search), η **μη γραμμική αναζήτηση** (nonlinear search), ο **επανακατακερματισμός** (rehashing) και ο **διπλός κατακερματισμός** (double hashing). Η γραμμική αναζήτηση παρουσιάσθηκε χρονικά πριν από κάθε άλλη μέθοδο κατακερματισμού. Ήδη από το 1953 γίνεται αναφορά γι' αυτήν από ερευνητές της IBM, ο πρώτος όμως που τη συστηματοποίησε σε δημοσίευση ήταν ένας Ρώσσος επιστήμονας, ο Ershov το 1957. Ακολουθεί παρουσίαση των παραλλαγών αυτών που έχουν κοινό σημείο τη μη υπάρξη ξεχωριστής περιοχής υπερχείλισης.

Σύμφωνα με τη γραμμική αναζήτηση η εγγραφή που υπερχειλίζει αποθηκεύεται στον πρώτο επόμενο διαθέσιμο κάδο. Δηλαδή, η αναζήτηση προχωρεί γραμμικά στους επόμενους κάδους μέχρις ότου ή να βρεθεί κενός χώρος ή να προσπελασθεί και πάλι ο αρχικός κάδος (αν όλοι οι κάδοι είναι πλήρεις). Αυτός ο τύπος αναζήτησης λέγεται και μέθοδος **ανοικτής υπερχείλισης** (open overflow) ή **γραμμικής εξέτασης** (linear probing) ή **προοδευτικής υπερχείλισης** (progressive overflow). Ο αριθμός των προσπελάσεων κάδων που μεσολαβούν πριν προσπελασθεί ο κάδος, όπου πράγματι η εγγραφή είναι αποθηκευμένη, λέγεται **μετατόπιση** (displacement) της εγγραφής. Όταν οι τιμές των κλειδιών δεν είναι πραγματικά τυχαίες, αλλά συσσωρεύονται γύρω από ορισμένες τιμές, τότε η μετατόπιση στη χειρότερη περίπτωση είναι αρκετά μεγαλύτερη από τη μέση τιμή. Επίσης, όταν ο παράγοντας φόρτισης τείνει προς τη μονάδα, τότε η μέση τιμή της μετατόπισης αυξάνει σημαντικά.

Στο Σχήμα 9.2 φαίνεται ένα παράδειγμα εφαρμογής της μεθόδου αυτής σε ένα αρχείο αποτελούμενο από 10 διευθύνσεις αριθμημένες από 0 ως 9. Κάθε διεύθυνση αντιστοιχεί σε 1 κάδο των 2 σελίδων, με χωρητικότητα 2 εγγραφών ανά σελίδα. Ήδη στο αρχείο έχει εισαχθεί ένα σύνολο εγγραφών, όταν έρχεται για εισαγωγή η εγγραφή με κλειδί 220. Σύμφωνα με τη συνάρτηση μετασχηματισμού της διαίρεσης (220 mod 10), η εγγραφή αυτή πρέπει να αποθηκευθεί στη διεύθυνση 0, που όμως είναι πλήρης. Έτσι η



Σχήμα 9.2: Μέθοδος γραμμικής εξέτασης.

εγγραφή 220 αποθηκεύεται στην αμέσως επόμενη διεύθυνση με διαθέσιμο χώρο, δηλαδή τη διεύθυνση 1. Έτσι, η μετατόπιση είναι ένας κάδος.

Για να αποφευχθεί το πρόβλημα που δημιουργείται από τη συσσώρευση των κλειδιών γύρω από μία περιοχή διευθύνσεων, συχνά χρησιμοποιούνται μη γραμμικές μέθοδοι αναζήτησης του επόμενου διαθέσιμου κάδου. Τέτοια μέθοδος είναι η **τετραγωνική** (quadratic) αναζήτηση, που σκοπό έχει την απομάκρυνση των εγγραφών από την αρχική περιοχή σύγκρουσης. Σύμφωνα με τη μέθοδο αυτή οι θέσεις των επόμενων κάδων που θα εξετασθούν βρίσκονται από τη σχέση:

$$\Big(A \times i^2 + B \times i + key \bmod bk\Big) \bmod bk$$

όπου i είναι ο αύξων αριθμός της αναζήτησης (i=0,1,2,...), A και B είναι αυθαίρετες σταθερές, ενώ key είναι η τιμή του κλειδιού. Έστω και πάλι η περίπτωση σύγκρουσης της εγγραφής 220 κατά την εισαγωγή στη θέση 0 του Σχήματος 9.2. Θεωρώντας την προηγούμενη τετραγωνική σχέση με A=1 και B=1, για i=1 προκύπτει ότι η εγγραφή 220 πρέπει να αποθηκευθεί στην διεύθυνση 2, που είναι επίσης κατειλημμένη. Τελικά θέτοντας i=2 προκύπτει ότι η εγγραφή 220 αποθηκεύεται στη θέση 6. Στην περίπτωση αυτή η μετατόπιση είναι δύο κάδοι.

Μία άλλη τεχνική είναι η μέθοδος του επανακατακερματισμού, όπου χρησιμοποιείται μία άλλη συνάρτηση μετασχηματισμού για να εντοπίσει τον επόμενο διαθέσιμο κάδο. Για παράδειγμα, έστω ότι χρησιμοποιείται η συ-

νάρτηση της διαίρεσης

$$h_1(key) \ = \ key \ \mathsf{mod} \ bk$$

για την εύρεση της αρχικής θέσης αποθήκευσης της εγγραφής. Σε περίπτωση σύγκρουσης χρησιμοποιείται μία άλλη συνάρτηση για να δώσει την απόσταση σε κάδους από την αρχική θέση. Σε περίπτωση νέας σύγκρουσης γίνεται δοκιμή σε θέση που απέχει την ίδια απόσταση από τη θέση της δεύτερης σύγκρουσης. Ένα παράδειγμα συνάρτησης επανακατακερματισμού είναι:

$$dis \ = \ (h_1(key) + p) \ \mathsf{mod} \ bk$$

όπου p είναι μία σταθερά που επιλέγεται έτσι ώστε να είναι πρώτος αριθμός προς το bk. Στο γνωστό παράδειγμα μετά τη σύγκρουση της εγγραφής 220 στη θέση 0, υπολογίζεται η απόσταση $dis = (0+3) \ \mathsf{mod} \ 10 = 3$ και η εγγραφή αποθηκεύεται στη θέση 3. (Το 3 και το 10 είναι πρώτοι αριθμοί μεταξύ τους.) Επίσης, σημειώνεται ότι αν προκύψει dis=0, τότε τίθεται dis=1.

Όπως αναφέρθηκε καλή είναι εκείνη η συνάρτηση κατακερματισμού που διασπείρει ομοιόμορφα τις εγγραφές σε όλο το μέγεθος του αρχείου. Στην αντίθετη περίπτωση υπάρχει πρωτεύουσα συγκέντρωση, δηλαδή πολλές εγγραφές συνωστίζονται σε μία ορισμένη περιοχή του αρχείου. Είναι δυνατόν να παρουσιασθούν πολλές συγκρούσεις λόγω συνωνυμιών, οπότε υπάρχει **δευτερεύουσα συγκέντρωση** (secondary clustering). Η επίδραση του φαινομένου αυτού μειώνεται με τη μέθοδο του διπλού κατακερματισμού που είναι μία πιο κομψή εκδοχή του επανακατακερματισμού. Στο διπλό κατακερματισμό η παράμετρος p δεν είναι σταθερά αλλά υπολογίζεται κάθε φορά με μία άλλη συνάρτηση. Συνεπώς με τη μέθοδο αυτή απαιτούνται συνολικά τρεις συναρτήσεις. Η δεύτερη συνάρτηση είναι της μορφής:

$$h_2(key) \ = \ key \ \mathsf{mod} \ (bk - q) + 1$$

όπου το q είναι σταθερά. Το αποτέλεσμα της συνάρτησης h_2 εισάγεται στην:

$$dis \ = \ (h_1(key) + h_2(key)) \ \mathsf{mod} \ bk$$

Στο γνωστό παράδειγμα μετά τη σύγκρουση της εγγραφής 220 στη θέση 0, υπολογίζεται η απόσταση $dis = (0 + 220 \ \mathsf{mod} \ 9 + 1) \ \mathsf{mod} \ 10 = 6$ και η εγγραφή αποθηκεύεται στη θέση 4 (θεωρήθηκε q=1). Σε περίπτωση νέων συγκρούσεων η εγγραφή 220 θα κατευθύνονταν διαδοχικά στις θέσεις 8, 2, 6 κλπ., ενώ η εγγραφή 230 θα κατευθύνονταν στις θέσεις 7, 4, 1, 8 κλπ.

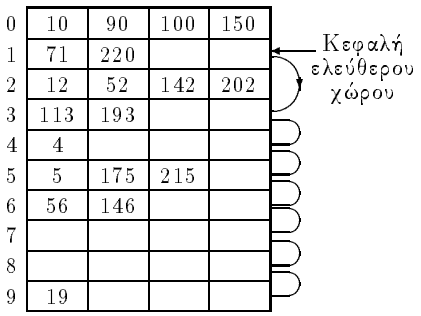
Μέχρι στιγμής δεν αναφέρθηκε τίποτε σε σχέση με τη διαγραφή. Η πιο συνήθης και εύκολη τεχνική είναι το μαρκάρισμα των διαγραμμένων εγγραφών. Ειδάλλως, αν είναι αναγκαίο να ελευθερωθεί ο χώρος για κατοπινές εισαγωγές, τότε απαιτούνται χρονοβόροι αλγόριθμοι διαγραφής. Πρέπει να σημειωθεί επίσης ότι στη βιβλιογραφία αναφέρονται μέθοδοι που δεν αποθηκεύουν μόνιμα κάποια εγγραφή υπερχείλισης στην αρχική της θέση αλλά είναι δυνατόν να την εκτοπίσουν προκειμένου η θέση αυτή να καταληφθεί από εγγραφή που έρχεται φυσιολογικά στη θέση αυτή με βάση την πρώτη συνάρτηση κατακερματισμού. Οι ενδιαφέρουσες αυτές τεχνικές, που βελτιστοποιούν την επίδοση της αναζήτησης με τίμημα το αυξημένο κόστος εισαγωγής, είναι εκτός των ορίων του τόμου αυτού. Εξ άλλου, όπως θα τονισθεί στη συνέχεια, οι μέθοδοι της ανοικτής διεύθυνσης δεν χρησιμοποιούνται στην πράξη για δομές αρχείων, αλλά κυρίως για δομές κύριας μνήμης. Στο βιβλίο των Δομών Δεδομένων υπάρχουν ποσοτικά στοιχεία σχετικά με την επίδοση των μεθόδων αυτών.

9.4 Διαχείριση συνωνύμων με άμεσες αλυσίδες

Σημαντικό μειονέκτημα των προηγούμενων μεθόδων είναι η κακή επίδοση σε περίπτωση αναζήτησης όταν ο παράγοντας φόρτισης πλησιάζει τη μονάδα. Οι επιδόσεις όλων των διεργασιών βελτιώνονται αν θεωρηθεί ένα επιπλέον πεδίο στον τύπο της εγγραφής, για να εξυπηρετεί ως δείκτης προς τις επόμενες λογικά εγγραφές και έτσι να τις συνδέει σε μία αλυσίδα. Η πρώτη αναφορά σχετικά με τη μέθοδο των αλυσίδων έγινε από τον Dumey (1956) και από τότε έχουν προταθεί πολλές παραλλαγές που ανήκουν σε δύο οικογένειες: των σύμφυτων αλυσίδων και των ξεχωριστών αλυσίδων. Και οι δύο μπορούν να υλοποιηθούν είτε με ιδιαίτερη περιοχή υπερχείλισης είτε χωρίς τέτοια περιοχή.

Η μέθοδος των σύμφυτων αλυσίδων προτάθηκε από τον Williams (1959), αναλύθηκε όμως σε μεγάλο βάθος και έκταση από τον Vitter στη δεκαετία του 80. Γενικά η μέθοδος διατηρεί μία κυκλική συνδεδεμένη λίστα από τις διευθύνσεις των διαθέσιμων κάδων, ώστε να χρησιμοποιηθούν για την αποθήκευση των εγγραφών υπερχείλισης. Όταν μία εγγραφή υπερχειλίσει από τον κάδο A, τότε αποθηκεύεται στον κάδο B, που είναι ο πρώτος κάδος της συνδεδεμένης λίστας. Όσες εγγραφές υπερχειλίζουν αποθηκεύονται στον κάδο B μέχρι να γεμίσει και να διαγραφεί από τη λίστα των διαθέσιμων.

Κάθε εγγραφή που στο μέλλον θα υπερχειλίσει από τον κάδο Α θα κατευθυνθεί προς τον Γ, όπου όμως θα κατευθυνθούν και οποιεσδήποτε εγγραφές υπερχειλίσουν από τον Β. Έτσι, στον κάδο Γ συμφύονται οι αλυσίδες των συνώνυμων που προέρχονται από τους κάδους Α και Β.



Σχήμα 9.3: Μέθοδος σύμφυτων αλυσίδων.

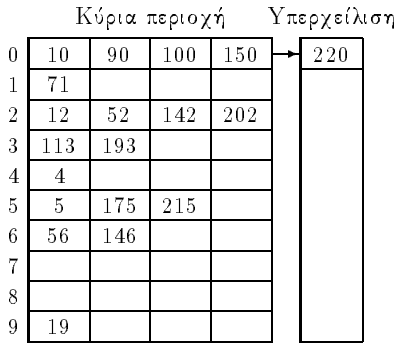
Στο Σχήμα 9.3 παρουσιάζεται η δομή με τα δεδομένα των προηγούμενων παραδειγμάτων. Και πάλι έχει ήδη αποθηκευθεί κατά τον ίδιο τρόπο η ίδια ακολουθία εγγραφών. Η διεύθυνση 0 είναι ήδη γεμάτη όταν έρχεται η εγγραφή 220. Η κορυφή της λίστας των διαθέσιμων κάδων δείχνει στον κάδο 1. Συνεπώς, η εγγραφή 220 κατευθύνεται στον κάδο αυτό.

Το προηγούμενο σχήμα είναι αρκετά αφαιρετικό και δεν δείχνει την ακριβή γραμμογράφηση ούτε του κάδου ούτε της εγγραφής. Είναι όμως απαραίτητη η σωστότερη δόμησή τους. Πιο συγκεκριμένα, κάθε εγγραφή πρέπει να έχει δύο πεδία δεικτών που να δείχνουν τον κάδο, όπου είναι αποθηκευμένη η λογικά επόμενη και η λογικά προηγούμενη εγγραφή. Είναι δυνατόν επίσης μία εγγραφή να εισαχθεί σε κάδο που είναι κατειλημμένος από εγγραφές υπερχείλισης. Επομένως απαιτείται ένα επιπλέον πεδίο-δείκτης στο επίπεδο του κάδου, που να δείχνει τη διεύθυνση του κάδου, όπου είναι αποθηκευμένη η πρώτη λογικά εγγραφή που θα έπρεπε να είχε αποθηκευθεί στο συγκεκριμένο κάδο.

Στη συνέχεια θα εξετασθεί η μέθοδος των ξεχωριστών αλυσίδων που, όπως είναι γνωστό, μπορεί να διατηρεί ιδιαίτερη λογικά περιοχή υπερχείλισης. Η περιοχή αυτή φυσικά μπορεί να είναι είτε τμήμα του κύριου αρχείου είτε ανεξάρτητο αρχείο. Εδώ θα εξετασθεί η δεύτερη περίπτωση και θα δοθούν αναλυτικές εκφράσεις για το χρονικό κόστος των διαφόρων διεργασιών,

γιατί αυτή η μέθοδος εφαρμόζεται περισσότερο στα σύγχρονα συστήματα διαχείρισης βάσεων δεδομένων.

Προσπέλαση μίας εγγραφής σημαίνει μία σειρά ενεργειών. Πρώτον, το κλειδί μετασχηματίζεται, δεύτερον, μεταφέρεται ο κάδος στην κύρια μνήμη, τρίτον, οι εγγραφές του κάδου διαβάζονται σειριακά και, τέλος, σε περίπτωση αποτυχίας η αναζήτηση συνεχίζεται στην περιοχή υπερχείλισης. Αρμοδιότητα του σχεδιαστή του αρχείου είναι η απόφαση της συγκεκριμένης υλοποίησης της περιοχής αυτής που μπορεί να γίνει κατά πολλούς τρόπους. Το κύριο ερώτημα είναι τί μεγέθους θα είναι οι κάδοι της περιοχής αυτής. Αν η περιοχή υπερχείλισης είναι τμήμα του κύριου αρχείου (οπότε λέγεται **κελλάρι** - cellar), τότε το μέγεθος του κάδου προφανώς είναι ενιαίο.



Σχήμα 9.4: Μέθοδος ξεχωριστών αλυσίδων.

Στο Σχήμα 9.4 δίνεται ένα παράδειγμα της τεχνικής αυτής παρόμοιο με τα προηγούμενα. Στην περιοχή υπερχείλισης για κάθε διεύθυνση της κύριας περιοχής δεσμεύεται ένας κάδος, που αποτελείται από μία σελίδα με χωρητικότητα μίας εγγραφής. Έχει αποθηκευθεί η ίδια ακολουθία εγγραφών, οπότε εισάγεται η εγγραφή 220 στη διεύθυνση 0 που είναι ήδη γεμάτη. Συνεπώς, η εγγραφή 220 κατευθύνεται στην περιοχή υπερχείλισης και αποθηκεύεται στον αντίστοιχο κάδο.

9.4.1 Αναζήτηση εγγραφής

Η μέθοδος είναι πολύ αποτελεσματική αν η κύρια περιοχή είναι αρκετά μεγάλη και η κατανομή των εγγραφών στους κάδους είναι περίπου ομοιόμορφη.

Με την προϋπόθεση ότι δεν υπάρχει υπερχείλιση, το χρονικό κόστος για την προσπέλαση μίας εγγραφής είναι:

$$T_{\pi\rho o\sigma} \;=\; s + r + dtt$$

Με την πάροδο του χρόνου και την αποθήκευση πολλών εγγραφών είναι βέβαιο ότι θα υπάρξει υπερχείλιση με επιμήκεις αλυσίδες. Στην περίπτωση αυτή το χρονικό κόστος είναι:

$$T_{\pi\rho o\sigma}(\epsilon\pi\iota\tau) \;=\; (s + r + dtt) \;+\; \frac{x}{2} \times (s + r + dtt)$$

Ο τύπος εξηγείται ως εξής. Η πρώτη παρένθεση δηλώνει την προσπέλαση του κάδου της κύριας περιοχής, ενώ το γινόμενο δηλώνει τη διάσχιση της αλυσίδας x κάδων υπερχείλισης για τον εντοπισμό της ζητούμενης εγγραφής. Κατά μέσο όρο η μισή αλυσίδα θα πρέπει να διασχισθεί.

Μία αναζήτηση είναι ανεπιτυχής, όταν όλες οι τιμές των κλειδιών των εγγραφών που εξετάζονται, είτε στην κύρια είτε στην περιοχή υπερχείλισης, δεν ταυτίζονται προς το ζητούμενο κλειδί. Έτσι η αναζήτηση τερματίζει όταν φθάσει σε εγγραφή με δείκτη NIL. Η ανεπιτυχής αναζήτηση είναι πιο χρονοβόρα από την επιτυχή, γιατί πρέπει να διασχισθεί ολόκληρη η αλυσίδα υπερχείλισης (ενώ στο B^+-δένδρο η επιτυχής και η ανεπιτυχής αναζήτηση έχουν το ίδιο κόστος). Άρα, το σχετικό κόστος είναι:

$$T_{\pi\rho o\sigma}(\alpha\nu\epsilon\pi) \;=\; (s + r + dtt) \;+\; x \times (s + r + dtt)$$

Αυτός ο τύπος ισχύει όταν το ποσοστό υπερχείλισης είναι μεγάλο. Αν το ποσοστό αυτό είναι μικρό, τότε ισχύει η πρώτη σχέση και για τους δύο τύπους αναζητήσεων. Φαίνεται, λοιπόν, ότι κατά τη φάση σχεδίασης του αρχείου πρέπει να προσεχθούν μερικές παράμετροι, ώστε πράγματι η επίδοση του αρχείου να είναι υψηλή. Αν το αρχείο αποκτήσει μεγάλη περιοχή υπερχείλισης πρέπει να αναδιοργανωθεί και να επανα-αποθηκευθεί σε μεγαλύτερη κύρια περιοχή.

Πρέπει να τονισθεί ότι οι προηγούμενοι τύποι είναι προσεγγιστικοί, γιατί δεν λαμβάνουν υπ΄ όψη τον παράγοντα φόρτισης και τη χωρητικότητα του κάδου. Οι Πίνακες 9.2 και 9.3 δίνουν τον αριθμό των προσπελάσεων σε αρχείο κατακερματισμού για επιτυχή και ανεπιτυχή αναζήτηση αντίστοιχα, ως συνάρτηση του παράγοντα φόρτισης και της χωρητικότητας του κάδου. Τα στοιχεία αυτά έχουν προκύψει από μαθηματική ανάλυση που είναι εκτός των ορίων του βιβλίου, οι πίνακες όμως παρουσιάζονται γιατί από αυτούς εξάγονται ποιοτικά πρακτικά συμπεράσματα. Απαραίτητες προϋποθέσεις εργασίας είναι ότι:

Μέγεθος κάδου	Παράγοντας φόρτισης							
	50%	70%	90%	100%	150%	200%	300%	500%
1	1,3	1,4	1,5	1,5	1,8	2,0	2,5	3,5
2	1,1	1,2	1,4	1,4	1,8	2,3	3,2	5,1
3	1,1	1,2	1,3	1,4	1,9	2,5	3,8	6,7
4	1,1	1,1	1,3	1,4	2,0	2,8	4,5	8,3
5	1,0	1,1	1,3	1,4	2,1	3,0	5,2	9,9
10	1,0	1,1	1,2	1,3	2,5	4,3	8,5	17,9
20	1,0	1,0	1,2	1,3	3,3	6,8	15,2	33,9
50	1,0	1,0	1,1	1,3	5,8	14,3	35,2	81,9

Πίνακας 9.2: Μέση τιμή προσπελάσεων για επιτυχή αναζήτηση σε τυχαίο αρχείο με ξεχωριστές αλυσίδες.

Μέγεθος κάδου	Παράγοντας φόρτισης								
	20%	30%	40%	50%	60%	70%	80%	90%	95%
1	1,0	1,0	1,1	1,1	1,1	1,2	1,2	1,3	1,3
2	1,0	1,0	1,0	1,1	1,2	1,2	1,3	1,4	1,5
3	1,0	1,0	1,0	1,1	1,2	1,3	1,4	1,5	1,6
4	1,0	1,0	1,0	1,1	1,1	1,3	1,4	1,6	1,6
5	1,0	1,0	1,0	1,1	1,1	1,2	1,4	1,6	1,6
10	1,0	1,0	1,0	1,0	1,1	1,2	1,4	1,8	2,0
20	1,0	1,0	1,0	1,0	1,0	1,1	1,4	1,9	2,3
50	1,0	1,0	1,0	1,0	1,0	1,0	1,2	1,9	2,7

Πίνακας 9.3: Μέση τιμή προσπελάσεων για ανεπιτυχή αναζήτηση σε τυχαίο αρχείο με ξεχωριστές αλυσίδες.

- οι εγγραφές κατανέμονται στους κάδους με δυωνυμική κατανομή,
- οι εγγραφές αναζητώνται ισοπίθανα, και
- η χωρητικότητα των κάδων στην κύρια περιοχή και στην περιοχή υπερχείλισης είναι $Bkfr$ και 1, αντίστοιχα.

Όπως αναμένεται, και στις δύο περιπτώσεις η απόδοση του αρχείου εκφυλίζεται καθώς αυξάνει ο παράγοντας φόρτισης. Ακόμη, αν ο παράγοντας φόρτισης είναι μικρότερος του 100% και του 50%, τότε η αύξηση της χωρητικότητας του κάδου έχει ως αποτέλεσμα σταθερή βελτίωση της απόδοσης

της επιτυχούς και της ανεπιτυχούς αναζήτησης, αντίστοιχα, επειδή μικραίνει το μήκος της αλυσίδας των κάδων. Αυτό όμως το πλεονέκτημα, που ισχύει για μεγάλες χωρητικότητες, αντισταθμίζεται από το γεγονός ότι μεγαλώνει ο χρόνος μεταφοράς του κάδου, *dtt*. Πάντως, το πρακτικό συμπέρασμα είναι ότι ο παράγοντας φόρτισης δεν πρέπει να υπερβαίνει το 70%-80%, γιατί στην περιοχή αυτή βελτιστοποιείται η σχέση κενός χώρος προς επίδοση.

9.4.2 Διαγραφή εγγραφής

Υπάρχουν δύο τρόποι διαγραφής των εγγραφών. Ο ένας είναι παρόμοιος με τον τρόπο που χρησιμοποιείται στα αρχεία σωρού και στα αρχεία ISAM. Δηλαδή, ανεβαίνει μία σημαία που δηλώνει ότι η εγγραφή είναι ουσιαστικά διαγραμμένη χωρίς να απομακρυνθεί φυσικά, ώστε να αποδοθεί ο χώρος στο σύστημα. Κατά την περιοδική αναδιοργάνωση οι εγγραφές τακτοποιούνται και πάλι χωρίς να γίνεται σπατάλη χώρου.

Σύμφωνα με το δεύτερο τρόπο, η εγγραφή διαγράφεται φυσικά και αντικαθίσταται με την τελευταία εγγραφή της αλυσίδας που ξεκινά από τον κάδο αυτόν. Επιπλέον, αν ο τελευταίος κάδος αδειάσει από εγγραφές, τότε αποδίδεται και πάλι στο σύστημα. Αυτή η τεχνική είναι περισσότερο αποτελεσματική από την άποψη του καταλαμβανόμενου χώρου και του χρονικού κόστους για αναζήτηση. Ωστόσο η διαγραφή καθίσταται πιο χρονοβόρα, γιατί πρέπει να γίνει διάσχιση της αλυσίδας ώστε να μεταφερθεί η τελευταία εγγραφή στον κενό χώρο. Έτσι στην ουσία εκτελείται μία ανεπιτυχής αναζήτηση. Υστερα πρέπει να επαναποθηκευθούν δύο κάδοι, αυτός όπου γίνεται η διαγραφή και ο τελευταίος της αλυσίδας. Στην απλή περίπτωση όπου αυτοί οι δύο κάδοι ταυτίζονται, το χρονικό κόστος είναι:

$$T_{\delta\iota\alpha\gamma} \;=\; T_{\pi\rho o\sigma}(\alpha\nu\epsilon\pi) \;+\; 2r$$

Οι περισσότερες υλοποιήσεις στατικών άμεσων αρχείων χρησιμοποιούν την πρώτη μέθοδο. Οι νέες μέθοδοι οργάνωσης δυναμικών αρχείων κατακερματισμού δεν χρειάζονται αναδιοργάνωση, όπως απαιτούν περιοδικά οι στατικές. Φαίνεται ότι στο μέλλον οι στατικές οργανώσεις θα χρησιμοποιούν κυρίως τη δεύτερη μέθοδο διαγραφών.

9.4.3 Εισαγωγή εγγραφής

Έστω ότι μία εισαγόμενη εγγραφή αποθηκεύεται στον τελευταίο κάδο της αλυσίδας. Αρα, και αυτή η περίπτωση μοιάζει με μία ανεπιτυχή αναζήτηση.

Στη συνέχεια πρέπει ο τελευταίος κάδος να επανα-αποθηκευθεί συμπεριλαμβάνοντας την τελευταία εγγραφή. Συνεπώς το χρονικό κόστος είναι:

$$T_{\text{εισ}} \;=\; T_{\text{προσ}}(\text{ανεπ}) \;+\; 2r$$

Ο τύπος αυτός δεν λαμβάνει το κόστος που απαιτείται για την προσάρτηση στην αλυσίδα ενός νέου κάδου, όταν ο τελευταίος είναι πλήρης και δεν διαθέτει χώρο για την εγγραφή. Φαίνεται, λοιπόν, ότι και η εισαγωγή δεν είναι δαπανηρή διεργασία.

9.4.4 Εξαντλητική ανάγνωση αρχείου

Η μέθοδος του κατακερματισμού καταστρέφει τη λογική σειρά των εγγραφών, και είναι προβληματική όταν τίθενται ερωτήσεις διαστήματος με βάση το πρωτεύον ή κάποιο δευτερεύον κλειδί. Για παράδειγμα, έστω ότι δίνεται ένα αρχείο χιλίων υπαλλήλων μίας εταιρείας και ότι απαιτείται να προσπελασθούν οι εγγραφές των υπαλλήλων με κωδικούς από 100 ως 200. Αυτή η περίπτωση θα ικανοποιηθεί με μία χρονοβόρα διαδικασία γιατί θα εκτελεσθούν εκατό ανεξάρτητες προσπελάσεις. Όμως αν ζητηθεί να προσπελασθούν οι εγγραφές των υπαλλήλων με βάση το επίθετο από το *Κόλλιας* ως το *Μανωλόπουλος*, τότε η απάντηση είναι υπερβολικά χρονοβόρα, επειδή πρέπει να προσπελασθούν όλες οι εγγραφές και να γίνει επιλογή των κατάλληλων. Έτσι προκύπτει ότι το κόστος για την ανάκτηση όλων των εγγραφών ενός αρχείου είναι:

$$T_{\text{εξαν}} \;=\; n \times T_{\text{προσ}}$$

Επιπλέον απαιτείται χρόνος για την ταξινόμηση των εγγραφών πριν δοθούν στο χρήστη.

9.4.5 Αναδιοργάνωση αρχείου

Αναδιοργάνωση του αρχείου συμβαίνει όταν οι αποθηκευμένες εγγραφές ξεπεράσουν τα πλαίσια, που είχαν τεθεί κατά τη φάση του σχεδιασμού, και η επίδοση εκφυλίζεται. Η αναδιοργάνωση είναι εξαιρετικά χρονοβόρα διαδικασία, ενώ συχνά η επιλογή του χρόνου εκτέλεσης της είναι δύσκολη, επειδή προσωρινά το αρχείο δεν είναι διαθέσιμο στο χρήστη. Το κόστος της αναδιοργάνωσης ισούται με το κόστος εξαντλητικής ανάγνωσης του αρχείου

συν το κόστος της διαδοχικής εισαγωγής όλων των εγγραφών σε μία νέα έκταση στο δίσκο. Δηλαδή ισχύει:

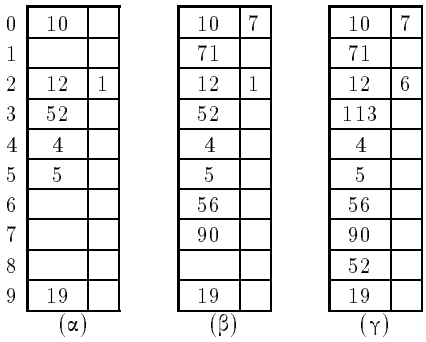
$$T_{\text{αναδ}} = T_{\text{εξαν}} + n \times T_{\text{εισ}}$$

9.5 Διαχείριση συνωνύμων με ψευδοαλυσίδες

Σύμφωνα με τη μέθοδο αυτή, που προτάθηκε από τους Χαλάτση και Φιλοκύπρου (1978), ο τύπος της εγγραφής περιέχει ένα `ψευδοδείκτη΄, δηλαδή ένα επιπλέον πεδίο που σκοπό έχει τη λογική σύνδεση της κάθε εγγραφής με την επόμενή της. Με άλλα λόγια το πεδίο αυτό δεν είναι ένας πραγματικός δείκτης μεγέθους μερικών bytes που δείχνει επακριβώς τη θέση της επόμενης εγγραφής, αλλά αποτελείται από μερικά μόνο bits που με κατάλληλους υπολογισμούς δίνουν τη θέση της επόμενης εγγραφής. Η μέθοδος ονομάζεται και **μέθοδος των υπολογιζόμενων αλυσίδων** (computed chaining), επειδή γίνονται κάποιοι υπολογισμοί. Εξ άλλου υπενθυμίζεται ότι βασικός κανόνας της οργάνωσης δεδομένων είναι ότι προτιμάται ο υπολογισμός από την αποθήκευση, γιατί το κόστος υπολογισμών είναι μικρότερο από το κόστος αποθήκευσης συν το κόστος εισόδου/εξόδου δεδομένων.

Σε σχέση με τις μεθόδους που ανήκουν στην οικογένεια της ανοικτής υπερχείλισης υπάρχει μία επιπλέον βασική διαφορά. `Εστω ότι η συνάρτηση κατακερματισμού κατευθύνει την εισαγόμενη εγγραφή σε μία θέση κατειλημμένη. Ελέγχεται αν η εγγραφή που είναι αποθηκευμένη στη θέση αυτή είναι συνώνυμη ή είναι εγγραφή υπερχείλισης. Αν είναι συνώνυμη, τότε η εισαγόμενη εγγραφή κατευθύνεται σε μία άλλη θέση που είναι συνάρτηση όχι του κλειδιού της εγγραφής που εισάγεται, αλλά του κλειδιού της εγγραφής που είναι ήδη αποθηκευμένη. Πιο συγκεκριμένα, από την τιμή του κλειδιού προκύπτει με μία συνάρτηση κατακερματισμού μία τιμή i, που δηλώνει ότι η εισαγόμενη εγγραφή θα πρέπει να απομακρυνθεί κατά i θέσεις. Αν η νέα αυτή θέση δεν είναι κατειλημμένη, τότε η εισαγόμενη εγγραφή αποθηκεύεται στη θέση αυτή και ο ψευδοδείκτης παίρνει την τιμή 1, που δηλώνει ότι συνώνυμη εγγραφή υπάρχει μετά $1 \times i$ θέσεις. Αν η νέα θέση είναι κατειλημμένη, τότε η εισαγόμενη εγγραφή απομακρύνεται κατά άλλες i θέσεις, δηλαδή συνολικά $2 \times i$ θέσεις από την αρχική θέση. Γίνεται αντιληπτό ότι αν τελικά αποθηκευθεί η εγγραφή στην τελευταία αυτή θέση τότε ο ψευδοδείκτης θα πάρει την τιμή 2. Αν η εισαγόμενη εγγραφή κατευθυνθεί σε μία θέση που είναι κατειλημμένη από εγγραφή υπερχείλισης, τότε η τελευταία παραχωρεί

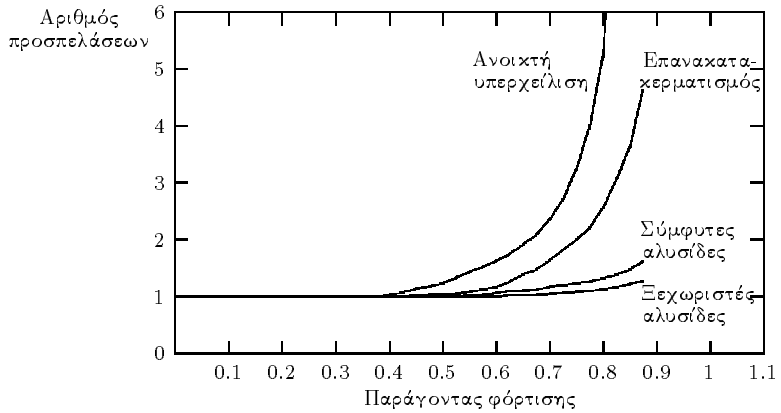
τη θέση στην εισαγόμενη και επανα-εισάγεται μαζί με όλες τις εγγραφές που έπονταν στην αλυσίδα. Στο Σχήμα 9.5 παρουσιάζεται ένα παράδειγμα διαδοχικών εισαγωγών εγγραφών με κλειδιά: 4, 5, 10, 12, 19, 52, 56, 71, 90, και 113, σε αρχείο 10 θέσεων. Τα τρία στιγμιότυπα του σχήματος δίνουν τη δομή μετά την εισαγωγή των εγγραφών 52, 90 και 113 αντίστοιχα. Ως πρώτη συνάρτηση κατακερματισμού χρησιμοποιείται η key mod 10, ενώ ως δεύτερη χρησιμοποιήθηκε η συνάρτηση key div 10. Σημειώνεται ότι αν από τη συνάρτηση αυτή προκύψει αποτέλεσμα 0, τότε ως αποτέλεσμα θεωρείται το 1.



Σχήμα 9.5: Μέθοδος ψευδοαλυσίδων.

Η μέθοδος αυτή είναι ένας συμβιβασμός μεταξύ των μεθόδων ανοικτής υπερχείλισης και των μεθόδων με χρήση αλυσίδας, τόσο από άποψη χώρου όσο και από άποψη χρονικών επιδόσεων. Σημαντικό πάντως είναι να υπολογισθεί εκ των προτέρων το μέγεθος του ψευδοδείκτη σε bits. Για παράδειγμα για αρχείο 10 θέσεων στη χειρότερη περίπτωση αρκούν ψευδοδείκτες των 4 bits, για αρχείο 100 θέσεων αρκούν 7 bits κοκ. Είναι δυνατόν όμως ο διαθέσιμος αριθμός των bits να είναι σχετικά μικρός. Έτσι αν πρέπει να αποθηκευθεί αριθμός αλμάτων, i, μεγαλύτερος από το διαθέσιμο, τότε απλώς αποθηκεύεται ο διαθέσιμος. Κατά τον τρόπο αυτό επιτυγχάνεται η σύνδεση των εγγραφών, αλλά προφανώς αυξάνει το κόστος αναζήτησης.

Ανακεφαλαιώνοντας για όλες τις μεθόδους, έστω ότι ένας κάδος αποτελείται από μία σελίδα χωρητικότητας μίας εγγραφής, οπότε το χρονικό κόστος αναζήτησης μετράται με προσπελάσεις εγγραφών. Στο Σχήμα 9.6 παρουσιάζεται συγκριτικά η επίδοση των τεσσάρων μεθόδων διαχείρισης των συνωνύμων υπερχείλισης. Φαίνεται ότι όταν $Lf > 90\%$, τότε οι μέθοδοι που



Σχήμα 9.6: Επίδοση τεχνικών διαχείρισης συνωνύμων.

χρησιμοποιούν αλυσίδα είναι σημαντικά καλύτερες. Επίσης, αν $Lf > 50\%$, τότε η μη γραμμική αναζήτηση υπερτερεί της ανοιχτής υπερχείλισης.

Εκτός από το κόστος εισόδου/εξόδου δεδομένων, υπάρχουν και άλλα κόστη που πρέπει να ληφθούν υπ΄ όψη. Κατ΄ αρχήν, οι μέθοδοι της ανοικτής διεύθυνσης έχουν πολύ απλό λογισμικό, γιατί ούτε αποθηκεύουν ούτε διαχειρίζονται δείκτες. Αυτό το γεγονός τείνει να μειώσει το χρόνο υπολογισμών, που ίσως θα μπορούσε να συνεισφέρει σημαντικά στο συνολικό χρόνο. Επίσης, το γεγονός ότι αυτές οι μέθοδοι δεν χρησιμοποιούν δείκτες δεν σημαίνει ότι γίνεται ιδιαίτερα μεγάλη οικονομία από άποψη χώρου. Αυτό οφείλεται στο ότι με τη χρήση των αλυσίδων αντιστοιχεί ένας δείκτης ανά κάδο και συνεπώς το ποσό είναι ίσως αμελητέο. Η χρίσιμη και βασική παράμετρος είναι ο παράγοντας φόρτισης, που πρέπει να διατηρηθεί σε χαμηλά επίπεδα για να μείνει σε λογικά πλαίσια και η επίδοση.

Άρα τελικά, για την κατηγορία των τεχνικών της ανοιχτής διεύθυνσης τα πλεονεκτήματα είναι απλότητα λογισμικού και οικονομία σε χρόνο υπολογισμών, ενώ τα μειονεκτήματα είναι αυξημένοι χρόνοι για είσοδο και έξοδο δεδομένων καθώς και χώρος αποθήκευσης. Αν μπορεί να προβλεφθεί η ανάπτυξη του αρχείου τότε η χρήση ξεχωριστών αλυσίδων είναι μια εξαιρετική τεχνική για αναζήτηση, εισαγωγή ή διαγραφή μίας εγγραφής. Απεναντίας είναι μία πολύ άσχημη επιλογή για σειριακές διεργασίες και ερωτήσεις διαστήματος, όπως συμβαίνει εξ άλλου για όλες τις παραλλαγές του στατικού κατακερματισμού. Συνήθως αμέσως μετά την αρχική φόρτωση, η αναζήτηση τερματίζεται με μία προσπέλαση. Αν είναι γνωστό εκ των

προτέρων ότι οι ερωτήσεις διαστήματος θα γίνονται συχνότερα, τότε ίσως να προτιμηθεί η δομή του B^+-δένδρου. Πρέπει να σημειωθεί βέβαια ότι η σχετική σπουδαιότητα όλων των προηγουμένων παραγόντων εξαρτάται από το φόρτο εργασίας, τις απαιτήσεις του συστήματος και γενικότερα το υπολογιστικό περιβάλλον.

9.6 Ασκήσεις

<1> Σε στατικό αρχείο κατακερματισμού 15 σελίδων χωρητικότητας μίας εγγραφής να εισαχθούν οι εγγραφές με τις επόμενες τιμές κλειδιών: 42, 57, 16, 52, 66, 77, 12, 25, 21, 33, 32 και 14. Να χρησιμοποιηθούν όλες οι τεχνικές της οικογένειας της ανοικτής διεύθυνσης.

<**2**> Σε στατικό αρχείο κατακερματισμού 15 σελίδων χωρητικότητας μίας εγγραφής να εισαχθούν οι εγγραφές με τις επόμενες τιμές κλειδιών: 42, 57, 16, 52, 66, 77, 12, 25, 21, 33, 32 και 14. Να χρησιμοποιηθεί η τεχνική των ψευδοαλυσίδων. Πόσα bits είναι απαραίτητα για το μέγεθος του ψευδοδείκτη;

<**3**> Έστω ότι στον ορισμό του τύπου της εγγραφής περιέχονται δύο ψευδοδείκτες, συνεπώς μπορούν να δημιουργηθούν δύο υπολογιζόμενες αλυσίδες. Με κατακερματισμό στο κλειδί της υπερχειλίζουσας εγγραφής επιλέγεται ποιά αλυσίδα θα ακολουθηθεί (key mod 2). Να επαναληφθεί η προηγούμενη άσκηση για την παραλλαγή αυτή και να γίνει σύγκριση της επίδοσης κατά την αναζήτηση για την ίδια ομάδα δεδομένων.

<4> Δίνεται στατικό αρχείο κατακερματισμού 5 σελίδων χωρητικότητας 3 εγγραφών. Χρησιμοποιώντας τη συνάρτηση $h(key) = key$ mod 5 και τη μέθοδο της αλυσίδας να εισαχθούν οι εγγραφές με τις επόμενες τιμές κλειδιών: 42, 57, 16, 52, 66, 77, 12, 25, 21,3 3, 32 και 14.

<**5**> Στατικό αρχείο κατακερματισμού με χρήση αλυσίδας διακρίνεται από τα εξής χαρακτηριστικά: $Bkfr$=50, επιθυμητό Lf=70%, n=100.000, R=400 bytes. Μετά τη φόρτωση κάθε εγγραφή ανακτάται με μία προσπέλαση στο δίσκο, γιατί το μήκος της αλυσίδας είναι 1. Μετά από καιρό στο αρχείο υπάρχουν συνολικά 400.000 εγγραφές. Αν το μέσο μήκος της αλυσίδας έχει γίνει 3, ποιά είναι η νέα τιμή του παράγοντα φόρτισης. Με πόσες προσπελάσεις γίνεται πλέον η αναζήτηση μίας εγγραφής. Συμφέρει ή όχι η δομή αυτή σε σχέση με το B^+-δένδρο;

<**6**> Σε αρχείο με 100.000 εγγραφές των 400 bytes γίνονται 10.000 προσπελάσεις και 10 αναζητήσεις διαστήματος που αφορούν στο 3% των εγγραφών. Ποιά δομή πρέπει να προτιμηθεί: ένα αρχείο κατακερματισμού με σειριακό διάβασμα, ένα δευτερεύον B^+-δένδρο ή ένα πρωτεύον B^+-δένδρο σχεδιασμένο κατάλληλα για το συγκεκριμένο τύπο ερωτήσεων.

<**7**> Για το αρχείο της προηγούμενης άσκησης να υπολογισθεί τι θα ήταν προτιμότερο αν σε κάθε 10.000 προσπελάσεις αντιστοιχούσαν 100 ερωτήσεις διαστήματος που αφορούν το 50% των εγγραφών;

<**8**> Σε αρχείο με 1.000.000 εγγραφές των 400 bytes πρέπει να γίνουν 10.000 προσπελάσεις και 100 ερωτήσεις διαστήματος που απαιτούν το 3% των εγγραφών. Ποιά μέθοδος είναι αποτελεσματικότερη: ένα αρχείο κατακερματισμού με σειριακή ανάγνωση για τις ερωτήσεις διαστήματος ή ένα πρωτεύον B^+-δένδρο σχεδιασμένο κατάλληλα;

<**9**> Σε στατικό αρχείο κατακερματισμού με χρήση αλυσίδας το 10% των εγγραφών αναζητάται με πιθανότητα 50% και το υπόλοιπο 90% των εγγραφών επίσης με 50%. Να περιγραφεί μία διαδικασία τοποθέτησης των εγγραφών που να βελτιστοποιεί την επίδοση και να βρεθεί το αντίστοιχο κόστος προσπέλασης.