Κεφάλαιο 4

ΣΕΙΡΙΑΚΑ ΑΡΧΕΙΑ

4.1 Εισαγωγή

4.2 Μη ταξινομημένα σειριακά αρχεία

4.3 Ταξινομημένα σειριακά αρχεία

4.4 Ασκήσεις

Κεφάλαιο 4

ΣΕΙΡΙΑΚΑ ΑΡΧΕΙΑ

4.1 Εισαγωγή

Τα **σειριακά** (sequential) αρχεία είναι ο παλαιότερος τύπος οργάνωσης αρχείων. Αναπτύχθηκε γύρω στα 1950 σε συνδυασμό με την ύπαρξη των μαγνητικών ταινιών, που αποτελούσαν τότε την κυρίαρχη συσκευή δευτερεύουσας αποθήκευσης. Ακόμη και σήμερα τα σειριακά αρχεία χρησιμοποιούνται στην πράξη στα σύγχρονα συστήματα διαχείρισης βάσεων δεδομένων, όχι όμως σε ταινίες, αλλά σε μαγνητικούς δίσκους. Στα αρχεία αυτά η φυσική σειρά είναι ίδια με τη λογική σειρά, οπότε η επεξεργασία προχωρά από την αρχή προς το τέλος. Δηλαδή, η i-οστή εγγραφή είναι αναγνώσιμη μόνο αφού προηγηθεί η ανάγνωση των i–1 προηγούμενων. Οι εγγραφές σε ένα σειριακό αρχείο μπορεί να αποθηκεύονται είτε αταξινόμητες είτε ταξινομημένες κατά αύξουσα ή φθίνουσα τιμή του κλειδιού.

Οι στοιχειώδεις λειτουργίες που γίνονται στα σειριακά αρχεία, όπως εξ άλλου και σε όλα τα αρχεία που θα εξετασθούν στη συνέχεια, δίνονται στον επόμενο πίνακα. Έχοντας μία τιμή για το χρονικό κόστος αυτών των πράξεων για κάθε είδους υλοποίηση αρχείου, είναι δυνατό να γίνει εκτίμηση της επίδοσης μίας βάσης δεδομένων.

Κάθε αρχείο αρχίζει με την **επικεφαλίδα** (file header) που περιέχει κάποιες πληροφορίες, όπως αν το αρχείο είναι δεκαεξαδικό, κινητής υποδιαστολής ή ASCII, το μέγεθος του αρχείου, το μήκος και τον αριθμό των εγγραφών, τη διεύθυνση της τελευταίας εγγραφής του αρχείου, αν υπάρχουν κατάλογοι και τι είδους, τις διευθύνσεις των εκτάσεων κλπ. Όταν

Σύμβολο	Ορισμός
$T_{\text{προσ}}$	Χρόνος προσπέλασης εγγραφής
$T_{\text{επομ}}$	Χρόνος προσπέλασης επόμενης εγγραφής
$T_{\text{εισ}}$	Χρόνος εισαγωγής εγγραφής
$T_{\text{αναν}}$	Χρόνος ανανέωσης εγγραφής
$T_{\text{διαγ}}$	Χρόνος διαγραφής εγγραφής
$T_{\text{εξαν}}$	Χρόνος εξαντλητικής ανάγνωσης αρχείου
$T_{\text{αναδ}}$	Χρόνος αναδιοργάνωσης αρχείου

Πίνακας 4.1: Στοιχειώδεις λειτουργίες σε αρχεία.

ανοίγει ένα αρχείο, τότε όλες αυτές οι πληροφορίες έρχονται στην κύρια μνήμη και βοηθούν στο χειρισμό του. Κατά το κλείσιμο του αρχείου οι νέες πληροφορίες επανεγγράφονται στον αντίστοιχο χώρο στο δίσκο. Για πολύ μικρά αρχεία η επικεφαλίδα μπορεί να καταλαμβάνει συγκρίσιμο χώρο με τα κυρίως δεδομένα, ωστόσο αυτή η περίπτωση δεν θα εξετασθεί. Σημειώνεται ότι αυτές οι πληροφοριές (όπως και άλλες) σχετικά με το αρχείο βρίσκονται στο **λεξικό δεδομένων** (data dictionary) των σχεσιακών βάσεων δεδομένων. Στη συνέχεια υποτίθεται ότι τα αρχεία είναι μεγάλα, οπότε ο χώρος που καταλαμβάνει η επικεφαλίδα καθώς και ο χρόνος επεξεργασίας της θα αγνοηθούν, γιατί αυτές οι παράμετροι είναι κοινές για όλα τα αρχεία.

Στο σημείο αυτό δίνεται ορισμός του **καταλόγου** (index). Ο κατάλογος είναι μία δομή που συνδέει τις τιμές ενός πεδίου (ή πεδίων) με τη διεύθυνση των εγγραφών που περιέχουν τις τιμές. Στη συνέχεια υποτίθεται ότι τα σειριακά αρχεία δεν συνοδεύονται από κατάλογο και συνεπώς δεν αξιοποιείται η δυνατότητα του υλικού για άμεση προσπέλαση. Τα σειριακά αρχεία, όπως αναφέρθηκε, διακρίνονται σε αταξινόμητα και ταξινομημένα κατά αύξουσα ή φθίνουσα τάξη του κλειδιού.

4.2 Μη ταξινομημένα σειριακά αρχεία

Σε ένα **μη ταξινομημένο σειριακό αρχείο ή αρχείο σωρού** (pile file) οι εγγραφές τοποθετούνται η μία μετά την άλλη ανάλογα με τη σειρά άφιξης. Κάλλιστα, οι εγγραφές μπορεί να ποικίλουν σε μήκος. Για παράδειγμα, οι εγγραφές σε ένα νοσοκομειακό ή αστυνομικό πληροφοριακό σύστημα εγγράφονται ως ένα αρχείο κειμένου. Στη συνέχεια όμως θα υποτεθεί ότι οι εγγραφές είναι σταθερού μήκους για να απλοποιηθούν οι υπολογισμοί.

Ακόμη και στην περίπτωση που το αρχείο έχει κάποια δομή, ώστε να επιταχύνεται η άμεση αναζήτηση, αν η αναζήτηση γίνεται με βάση ένα δευτερεύον κλειδί, τότε και πάλι θα πρέπει να θεωρηθεί ως ένα αρχείο σωρού. Στο κεφάλαιο αυτό δίνεται έμφαση όχι γιατί τα αρχεία σωρού είναι τόσο σημαντικά αλλά γιατί θα ακολουθηθεί σταθερά η ίδια μεθοδολογία και στα επόμενα κεφάλαια.

4.2.1 Προσπέλαση εγγραφής

Αν η διεύθυνση της εγγραφής είναι γνωστή, τότε το χρονικό κόστος για την προσπέλαση της είναι:

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

Όμως στο αρχείο σωρού δεν είναι γνωστή η διευθύνση της εγγραφής. Άρα αν δοθεί η τιμή κάποιου ή κάποιων πεδίων, τότε η αναζήτηση πρέπει να γίνει σειριακά και ελέγχοντας αν η τιμή του κλειδιού της εγγραφής ταυτίζεται με τη δεδομένη τιμή. Προκύπτει, λοιπόν, ότι ο μέσος όρος των σελίδων που θα μεταφερθούν στην κύρια μνήμη μέχρι να βρεθεί η κατάλληλη εγγραφή είναι:

$$\frac{1}{b} \times \sum_{i=1}^{b} i \;=\; \frac{1}{b} \times \frac{b \times (b+1)}{2} \approx \frac{b}{2}$$

Άρα, ο αναμενόμενος χρόνος για την αναζήτηση μίας εγγραφής σε αρχείο σωρού είναι:

$$T_{\pi\rho o\sigma} \;=\; ebt \times \frac{b}{2}$$

Στον τύπο αυτό εισάγεται ο πραγματικός χρόνος μεταφοράς, γιατί τελικά πρόκειται για μία σειριακή αναζήτηση στο μισό ενός μεγάλου αρχείου, κατά μέσο όρο.

4.2.2 Προσπέλαση επόμενης εγγραφής

Εφ΄ όσον οι εγγραφές είναι αταξινόμητες, έπεται ότι η αναζήτηση της λογικά επόμενης εγγραφής (δηλαδή, με βάση την τιμή κάποιου κλειδιού) δεν είναι παρά αναζήτηση μίας τυχαίας εγγραφής. Αυτό συμβαίνει γιατί υπάρχει μικρή τοπικότητα, δηλαδή μικρή πιθανότητα δύο διαδοχικές λογικές εγγραφές να βρίσκονται σε γειτονικές περιοχές στο δίσκο. Συνεπώς ισχύει:

$$T_{\epsilon\pi o\mu} \;=\; T_{\pi\rho o\sigma}$$

Δηλαδή, αν οι εγγραφές είναι αταξινόμητες και δοθεί για αναζήτηση μία λίστα τιμών (ίσως και ταξινομημένων) σε κάποιο πεδίο, τότε ο χρόνος αναζήτηση της επόμενης εγγραφής είναι ίδιος με το χρόνο αναζήτησης της πρώτης εγγραφής της λίστας.

4.2.3 Εξαντλητική ανάγνωση αρχείου

Αν αρχεί η ανάγνωση όλων των εγγραφών χωρίς απαραίτητα να δοθούν και ταξινομημένες, τότε ο απαιτούμενος χρόνος είναι:

$$T_{\text{εξαν}} = b \times ebt$$

Συνεπώς, ο απαιτούμενος χρόνος για την ανάγνωση ολόκληρου του αρχείου είναι διπλάσιος από το μέσο απαιτούμενο χρόνο για την προσπέλαση μίας μόνο εγγραφής. Επομένως, το αρχείο σωρού είναι πολύ καλή επιλογή για εφαρμογές όπου χρειάζεται απλά μία σάρωση του αρχείου, γιατί δεν απαιτείται επιπλέον δόμηση των δεδομένων. Σημειώνεται ότι αυτή η δόμηση εκτός του ότι καταναλώνει επιπλέον χώρο, αυξάνει και την πολυπλοκότητα. Για παράδειγμα, έστω ότι πρέπει να υπολογισθεί ο μέσος όρος των τιμών ενός πεδίου. Για τον υπολογισμό του τρέχοντος μέσου όρου κρατούνται δύο τιμές: ο αριθμός των εγγραφών (i) που έχουν αναγνωσθεί μέχρι τώρα και ο μέσος όρος των τιμών των πεδίων εγγραφών (A) που έχουν αναγνωσθεί μέχρι τώρα. Έτσι ισχύει:

$$New\_mean\_value = A \times \frac{i}{i+1} + \frac{new\_term}{i+1}$$

Έστω ότι υπάρχει μία λίστα τιμών ενός πεδίου. Αν όλες οι εγγραφές πρέπει να δοθούν στο χρήστη ταξινομημένες με βάση τις τιμές αυτού του πεδίου, τότε κάθε εγγραφή θα πρέπει να αναζητηθεί ανεξάρτητα από τις άλλες. Δηλαδή ισχύει:

$$T_{\text{εξαν}} = n \times T_{\text{προσ}} = n \times ebt \times \frac{b}{2} = \frac{ebt \times n^2}{2 \times Bfr}$$

Από τον τύπο αυτό φαίνεται ότι η εξαντλητική ανάγνωση με βάση τις ταξινομημένες τιμές ενός πεδίου είναι μία διαδικασία με πολυπλοκότητα $\mathrm{O}(n^2)$ και συνεπώς πολύ αναποτελεσματική, ιδιαίτερα μάλιστα όταν το n είναι πολύ μεγάλο.

4.2.4 Εισαγωγή εγγραφής

Αν υποτεθεί ότι δεν γίνεται έλεγχος για διπλοεγγραφές, τότε κάθε νέα εγγραφή προσαρτάται στο τέλος του αρχείου χωρίς κάποια άλλη επιπλέον ενέργεια. Η διεύθυνση της τελευταίας σελίδας βρίσκεται διαθέσιμη στην κύρια μνήμη τη στιγμή της εισαγωγής, επειδή είναι αποθηκευμένη στην επικεφαλίδα. Άρα, το χρονικό κόστος για την εισαγωγή μίας εγγραφής είναι:

$$T_{\text{εισ}} \;=\; s + 3r + btt$$

Το κόστος αυτό εξηγείται ως εξής. Ο χρόνος για τον εντοπισμό της τελευταίας σελίδας είναι $(s+r+btt)$, ο χρόνος για την απαραίτητη περιστροφή ώστε να επανέλθει η συγκεκριμένη σελίδα κάτω από την κεφαλή είναι $(2r–btt)$ και τέλος ο χρόνος για την επανα-αποθήκευση της σελίδας στο δίσκο είναι btt.

Είναι δυνατό με μία οριακή εισαγωγή να γεμίσει η συγκεκριμένη έκταση, οπότε το λειτουργικό σύστημα αποδίδει στο αρχείο μία νέα έκταση. Στην περίπτωση αυτή το κόστος αλλάζει.

4.2.5 Διαγραφή εγγραφής

Προφανώς για να γίνει η διαγραφή, πρέπει πρώτα να γίνει η αναζήτηση με κόστος $T_{\text{προσ}}$. Όταν η κατάλληλη σελίδα έρθει στην κύρια μνήμη, τότε η εγγραφή μαρκάρεται και θεωρείται πλέον διαγραμμένη. Η περιστροφή του δίσκου συνεχίζεται και συνεπώς απαιτείται χρόνος ίσος με $(2r–btt)$, ώστε να επανέλθει η συγκεκριμένη σελίδα κάτω από την κεφαλή. Επιπλέον απαιτείται χρόνος ίσος με btt ώστε να επανα-αποθηκευθεί η σελίδα στη θέση όπου βρισκόταν. Επομένως το χρονικό κόστος για μία διαγραφή είναι:

$$T_{\text{διαγ}} \;=\; T_{\text{προσ}} + 2r$$

Βέβαια, ο όρος $2r$ είναι πολύ μικρός σε σχέση με το χρονικό κόστος της αναζήτησης.

4.2.6 Ανανέωση εγγραφής

Για όλες τις μέχρι τώρα λειτουργίες το χρονικό κόστος είναι ίδιο είτε οι εγγραφές είναι σταθερού είτε μεταβλητού μήκους. Στην περίπτωση της ανανέωσης το μήκος της εγγραφής έχει πλέον σημασία, γιατί η νέα μορφή της

εγγραφής μπορεί να μην χωρά στη θέση, όπου ήταν αποθηκευμένη η παλιά μορφή της εγγραφής. Το χρονικό κόστος για την ανανέωση μίας εγγραφής σταθερού μήκους είναι:

$$T_{\text{αναν}} = T_{\text{προσ}} + 2r$$

Παρατηρείται ότι το κόστος είναι ίδιο με το κόστος της διαγραφής.

Αν το αρχείο περιέχει εγγραφές μεταβλητού μήκους, τότε η ανανέωση αντιμετωπίζεται ως μία διαγραφή και μία εισαγωγή. Αυτό οφείλεται στο γεγονός ότι είναι πολύ χρονοβόρο να μετακινηθούν όλες οι εγγραφές, ώστε να δημιουργηθεί χώρος για την νέα εκδοχή της εγγραφής. Έτσι ισχύει:

$$T_{\text{αναν}} = T_{\text{διαγ}} + T_{\text{εισ}}$$

Και στην περίπτωση αυτή το συνολικό κόστος κυριαρχείται από το κόστος της αναζήτησης.

4.2.7 Αναδιοργάνωση αρχείου

Το αρχείο αποδιοργανώνεται μετά από τις συνεχείς εισαγωγές και διαγραφές. Φυσική συνέπεια είναι η επίδοση του αρχείου να φθίνει για τους εξής λόγους:

- οι διαγραμμένες εγγραφές είναι μαρκαρισμένες αλλά καταλαμβάνουν το συγκεκριμένο χώρο. Έτσι, το μέγεθος του αρχείου (δηλαδή, ο αριθμός των σελίδων b) μεγαλώνει, ενώ αντίθετα η πραγματική χωρητικότητα των σελίδων μειώνεται, καθώς κενός χώρος διασκορπίζεται σε όλες τις σελίδες του αρχείου.

- οι εισαγωγές έχουν κατευθυνθεί προς το τέλος του αρχείου και κατά συνέπεια το αρχείο δυνητικά διαμοιράζεται σε διάφορες εκτάσεις. Έτσι, ο χρόνος εντοπισμού αυξάνεται.

Αναδιοργάνωση του αρχείου σημαίνει ότι πρέπει να αναγνωσθεί όλο το αρχείο και να επανα-αποθηκευθεί με τη νέα του μορφή. Αν το νέο αρχείο επανα-αποθηκευθεί επάνω από το παλιό, τότε υπάρχει ο κίνδυνος να χαθούν δεδομένα, αν το σύστημα πέσει κατά την διάρκεια της αναδιοργάνωσης. Για το λόγο αυτό το νέο αρχείο αποθηκεύεται σε νέα έκταση, ενώ η παλιά απελευθερώνεται και αποδίδεται ξανά στο σύστημα μετά το τέλος της αναδιοργάνωσης.

Επειδή πρέπει να αποφεύγεται η συνεχής μηχανική μετακίνηση του βραχίονα στις διάφορες εκτάσεις του νέου και του παλιού αρχείου, χρησιμοποιείται η τεχνική της διπλής απομονωτικής μνήμης. Δηλαδή, στην κύρια μνήμη μεταφέρονται οι σελίδες του παλιού αρχείου και ο χώρος των σελίδων τακτοποιείται αγνοώντας τις διαγραμμένες εγγραφές. Κάθε φορά που η απομονωτική μνήμη γεμίζει, το περιεχόμενό της μεταφέρεται στο δίσκο. Αυτή η διαδικασία επαναλαμβάνεται μέχρι να εξαντληθεί το παλιό αρχείο.

Το κόστος, λοιπόν, της αναδιοργάνωσης με μία συσκευή δίσκου είναι:

$$T_{\text{αναδ}} = b \times ebt + \left\lceil \frac{n}{Bfr} \right\rceil \times ebt$$

όπου b είναι οι σελίδες του παλιού αρχείου και n είναι οι εγγραφές του νέου αρχείου. Αν το σύστημα έχει δύο συσκευές δίσκου, τότε η αναδιοργάνωση γίνεται ταχύτερα, γιατί χρησιμοποιούνται τεχνικές παραλληλισμού.

Ανακεφαλαιώνοντας τα συμπεράσματα του κεφαλαίου αυτού, αναφέρεται και πάλι ότι το αρχείο σωρού είναι:

- οικονομικό από άποψη χώρου, αν δεν έχουν γίνει στο μεταξύ πολλές διαγραφές,
- βολικό στο χειρισμό των εισαγωγών,
- αποτελεσματικό για την εξαντλητική σάρωση του αρχείου όταν δεν χρειάζεται οι εγγραφές να δοθούν στο χρήστη ταξινομημένες με βάση τις τιμές κάποιου πεδίου,
- αργό για την αναζήτηση μίας εγγραφής ή της επόμενης με βάση την τιμή κάποιου κλειδιού, και τέλος
- τελείως αναποτελεσματικό αν πρόκειται να δοθούν οι εγγραφές στο χρήστη ταξινομημένες ως προς κάποιο πεδίο.

Τελικά, τα αρχεία σωρού χρησιμοποιούνται:

- σε στατιστικές επεξεργασίες, όπως για παράδειγμα για την εύρεση του μέσου όρου των τιμών κάποιου πεδίου,
- σε περίπτωση διαχωρισμού του αρχείου σε υποαρχεία με βάση κάποιο κριτήριο, όπως με βάση το φύλο, το μισθό κλπ. και με την προϋπόθεση ότι δεν υπάρχει κάποιος κατάλογος, και
- όταν το αρχείο δεν είναι ιδιαίτερα μεγάλο και ιδιαίτερα σε εφαρμογές με μικροϋπολογιστές.

4.3 Ταξινομημένα σειριακά αρχεία

Σε ένα ταξινομημένο αρχείο οι εγγραφές είναι διατεταγμένες ως προς τις τιμές ενός πεδίου ή συνδυασμού κάποιων πεδίων. Για παράδειγμα, συνήθως ένα αρχείο με διευθύνσεις είναι ταξινομημένο ως προς τον ταχυδρομικό κωδικό, ενώ ένα αρχείο πελατών είναι ταξινομημένο ως προς το ονοματεπώνυμο. Όμως αν το επίθετο και το όνομα αποτελούν διαφορετικά πεδία, τότε για το αρχείο πελατών η διάταξη των εγγραφών πρέπει να γίνει ως προς το συνδυασμό των δύο αυτών πεδίων.

Το ταξινομημένο σειριακό αρχείο μπορεί να παραμείνει ταξινομημένο κατά την εισαγωγή νέων εγγραφών, μόνο αν κάθε εισαγωγή συνοδεύεται και από μετατόπιση των εγγραφών με μεγαλύτερο κλειδί κατά μία θέση. Αυτή η διαδικασία είναι εξαιρετικά χρονοβόρα και για το λόγο αυτό δεν εφαρμόζεται ποτέ στην πράξη. Αντίθετα, το αρχείο αυτό παραμένει ανέπαφο, ενώ δημιουργείται ένα νέο αρχείο που περιέχει μόνο τις εισαγωγές. Το αρχείο αυτό ονομάζεται **περιοχή υπερχείλισης** (overflow area) και δεν είναι ταξινομημένο.

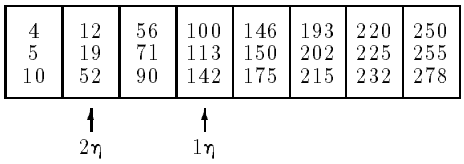
Όταν αναζητείται μία εγγραφή, τότε η επεξεργασία αρχίζει πρώτα από την **κύρια περιοχή** (main area) του ταξινομημένου αρχείου. Σε περίπτωση που η εγγραφή δεν είναι αποθηκευμένη στην κύρια περιοχή, τότε η αναζήτηση συνεχίζεται στην περιοχή υπερχείλισης. Αν η περιοχή υπερχείλισης περιέχει μόνο λίγες εγγραφές, τότε το κόστος αναζήτησης πρακτικά ισούται με το κόστος αναζήτησης στην κύρια περιοχή. Όσο όμως εισάγονται νέες εγγραφές, τόσο η περιοχή υπερχείλισης μεγαλώνει, καθώς και το αντίστοιχο κόστος αναζήτησης. Για το λόγο αυτό πρέπει περιοδικά το αρχείο να αναδιοργανώνεται, ώστε και πάλι να αποκτά τα πλεονεκτήματα του ταξινομημένου αρχείου.

Όπως και στο Κεφάλαιο 4.2, ακολουθεί η αναλυτική κοστολόγηση των στοιχειωδών πράξεων. Στη συνέχεια θεωρείται ότι οι εγγραφές είναι σταθερού μήκους.

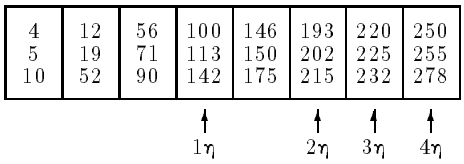
4.3.1 Προσπέλαση εγγραφής

Έστω ότι η κύρια περιοχή και η περιοχή υπερχείλισης αποτελούνται αντίστοιχα από b_m και b_o σελίδες, επομένως το αρχείο αποτελείται συνολικά από $b=b_m+b_o$ σελίδες. Όταν δίνεται η τιμή ενός κλειδιού, τότε η αντίστοιχη εγγραφή μπορεί να εντοπισθεί με **δυαδική αναζήτηση** (binary search).

Δηλαδή, η πρώτη προσπέλαση γίνεται στη μεσαία σελίδα της κύριας περιοχής. Αν η τιμή του ζητούμενου κλειδιού είναι μικρότερη από τα κλειδιά της σελίδας, τότε στη συνέχεια προσπελάζεται η μεσαία σελίδα του πρώτου μισού αυτού του τμήματος του αρχείου, ενώ αντίστοιχα αν η τιμή του κλειδιού είναι μεγαλύτερη από τα κλειδιά της σελίδας, τότε προσπελάζεται η μεσαία σελίδα του δεύτερου μισού του τμήματος αυτού. Η διχοτόμηση της κύριας περιοχής συνεχίζεται μέχρι να εντοπισθεί το κλειδί σε κάποια σελίδα. Αν τελικά το κλειδί δεν εντοπισθεί, τότε η αναζήτηση συνεχίζεται κατά σειριακό τρόπο στην περιοχή υπερχείλισης. Παραδείγματα επιτυχούς και ανεπιτυχούς αναζήτησης παρουσιάζονται στο Σχήμα 4.1 και στο Σχήμα 4.2, αντίστοιχα, όπου ισχύει $b{=}b_m{=}8$.



Σχήμα 4.1: Προσπελάσεις σε επιτυχή δυαδική αναζήτηση του 19.



Σχήμα 4.2: Προσπελάσεις σε ανεπιτυχή δυαδική αναζήτηση του 280.

Αν η αναζητούμενη εγγραφή είναι αποθηκευμένη στην κύρια περιοχή, τότε εύκολα αποδεικνύεται ότι η μέση τιμή του αριθμού των τυχαίων προσπελάσεων για μία επιτυχή αναζήτηση είναι:

$$\frac{1}{b_m} \times 1 + \frac{2}{b_m} \times 2 + \frac{4}{b_m} \times 3 + ... + \frac{1}{2} \times \log b_m \;=\; \log b_m - 1$$

Συνεπώς, ο απαιτούμενος χρόνος για τη δυαδική επιτυχή αναζήτηση μίας εγγραφής στην κύρια περιοχή είναι:

$$(\log b_m - 1) \times (s + r + btt)$$

Σημειώνεται ότι στην πρώτη προσπέλαση ο χρόνος εντοπισμού, s, θα βαρύνει περισσότερο, στη δεύτερη θα βαρύνει λιγότερο, γιατί ο εντοπισμός θα

γίνει στο μισό αρχείο κοκ. Από την άλλη πλευρά, ο χρόνος περιστροφής είναι σημαντικός και σταθερός κατά τις διαδοχικές προσπελάσεις. Τελικά, η τελευταία έκφραση είναι πεσιμιστική.

Αν η εγγραφή είναι αποθηκευμένη στην κύρια περιοχή, τότε πρέπει να γίνουν $\log b_m$ προσπελάσεις στη χειρότερη περίπτωση. Αν η εγγραφή είναι αποθηκευμένη στην περιοχή υπερχείλισης, τότε η αναζήτηση γίνεται κατά σειριακό τρόπο (όπως και στο αρχείο σωρού), και συνεπώς, πρέπει να αναγνωσθούν οι μισές σελίδες του αρχείου, κατά μέσο όρο. Ο απαιτούμενος χρόνος για την προσπέλαση της εγγραφής είναι:

$$\log b_m \times (s + r + btt) + \left(r + s + ebt \times \frac{b_o}{2}\right)$$

Αυτός ο τύπος δίνει το χρονικό κόστος της ανεπιτυχούς αναζήτησης στην ταξινομημένη περιοχή συν το χρόνο εντοπισμού και περιστροφής στην περιοχή υπερχείλισης συν το χρόνο για τη σειριακή αναζήτηση των μισών σελίδων της περιοχής υπερχειλίσης. Βέβαια, πρέπει να σημειωθεί ότι το πρώτο άθροισμα της έκφρασης είναι πεσιμιστικό, επειδή ο όρος $\log b_m$ αφορά στη χειρότερη περίπτωση. Αυτό γίνεται αντιληπτό από το Σχήμα 4.2, όπου για παράδειγμα, η ανεπιτυχής αναζήτηση του κλειδιού 105 τερματίζει με μία μόνο προσπέλαση.

Συνδυάζοντας τους δύο προηγούμενους τύπους προκύπτει ότι ο χρόνος για την αναζήτηση μίας εγγραφής είναι:

$$\begin{aligned} T_{\pi\rho o\sigma} \;=\; & \frac{b_m}{b} \times (\log b_m - 1) \times (s + r + btt) + \\ & \frac{b_o}{b} \times \left(\log b_m \times (s + r + btt) + (r + s + ebt \times \frac{b_o}{2})\right) \end{aligned}$$

Αν το b_o είναι ιδαίτερα μεγάλο, τότε κυριαρχεί το κόστος αναζήτησης στην περιοχή υπερχείλισης και πρακτικά ο προηγούμενος τύπος καταλήγει στον:

$$T_{\pi\rho o\sigma} \;=\; \frac{ebt \times b_o^2}{2 \times b}$$

Στην αντίθετη περίπτωση (δηλαδή, το b_o είναι μικρό), κυριαρχεί το κόστος αναζήτησης στην ταξινομημένη περιοχή και ο τύπος μετασχηματίζεται στον:

$$T_{\pi\rho o\sigma} \;=\; (\log b - 1) \times (s + r + btt)$$

Μία άλλη μέθοδος αναζήτησης με βάση την τιμή του πρωτεύοντος κλειδιού είναι η **αναζήτηση παρεμβολής** (interpolation search), που υπερτερεί της δυαδικής μεθόδου όταν τα κλειδιά υπακούουν σε ομοιόμορφη κατανομή. Σημειώνεται ότι στη βιβλιογραφία αναφέρονται επίσης η **αναζήτηση** Fibonacci, η **αναζήτηση άλματος** (jump search) και η **πολυωνυμική αναζήτηση** (polynomial search). Στο βιβλίο των Δομών Δεδομένων αναφέρονται περισσότερα στοιχεία για τις μεθόδους αυτές, που στηρίζονται στη στρατηγική `διαίρει και βασίλευε´ (divide and conquer).

4.3.2 Προσπέλαση επόμενης εγγραφής

Αν η περιοχή υπερχείλισης είναι μικρή, τότε το κόστος για την αναζήτηση της λογικά επόμενης εγγραφής είναι επίσης μικρό. `Εστω, λοιπόν, ότι ο βραχίονας είναι τοποθετημένος επάνω από τη σωστή άτρακτο και ότι το περιεχόμενο της τελευταίας σελίδας βρίσκεται ακόμη στην κύρια μνήμη. Είναι πιθανό η επόμενη εγγραφή να είναι αποθηκευμένη στην ίδια σελίδα με την εγγραφή που αναζητήθηκε πιο πρόσφατα. Η πιθανότητα να μην είναι στην ίδια σελίδα είναι $1/Bfr$. Συνεπώς ο χρόνος για την αναζήτηση της επόμενης εγγραφής είναι:

$$T_{\text{επομ}} \;=\; \left(1 - \frac{1}{Bfr}\right) \times 0 + \frac{1}{Bfr} \times (r + btt) \;=\; \frac{r + btt}{Bfr}$$

Επίσης, σημειώνεται ότι ο όρος r της περιστροφής είναι απαραίτητος για να έλθει η κεφαλή επάνω από την κατάλληλη σελίδα.

Ωστόσο, υπάρχει κάποια πιθανότητα η επόμενη εγγραφή να μην βρίσκεται στον ίδιο κύλινδρο. Αυτή η πιθανότητα αγνοείται, επειδή για μεγάλα συστήματα είναι πραγματικά μηδαμινή και η επιβάρυνση στο κόστος αμελητέα. Η ανάλυση που προηγήθηκε ισχύει με την προϋπόθεση ότι δεν υπάρχει περιοχή υπερχείλισης. Αν αυτή η συνθήκη δεν ισχύει, τότε το αρχείο εκφυλίζεται σε αρχείο σωρού και απαιτείται νέα ανάλυση.

4.3.3 Διαγραφή ή ανανέωση εγγραφής

Ο απαιτούμενος χρόνος για τη διαγραφή ή την ανανέωση (αν η ανανέωση δεν αφορά στο κλειδί) μίας εγγραφής είναι:

$$T_{\text{διαγ}} \;=\; T_{\text{αναν}} \;=\; T_{\text{προσ}} + 2r$$

Ο όρος $2r$ είναι ο απαραίτητος για την περιστροφή του δίσκου και την επανα-τοποθέτηση της κεφαλής επάνω από την ίδια σελίδα. Αν πρόκειται για διαγραφή, τότε απλώς η εγγραφή μαρκάρεται και δεν γίνεται επαναχρησιμοποίηση του φυσικού χώρου.

Αν η ανανέωση αφορά στην τιμή του κλειδιού, τότε στην ουσία πρόκειται για μία διαγραφή και μία εισαγωγή στην περιοχή υπερχείλισης. Επομένως ισχύει:

$$T_{\text{αναν}} = T_{\text{διαγ}} + T_{\text{εισ}}$$

Το κρίσιμο σημείο είναι αν η εγγραφή βρίσκεται αποθηκευμένη στην περιοχή υπερχείλισης ή όχι.

4.3.4 Εισαγωγή εγγραφής

Ο χρόνος για την εισαγωγή μίας εγγραφής είναι ίδιος με το χρόνο εισαγωγής εγγραφής σε αρχείο σωρού. Έτσι το χρονικό κόστος ισούται με:

$$T_{\text{εισ}} = s + 3r + btt$$

Βέβαια, στην ποσότητα αυτή πρέπει να προστεθεί και ο χρόνος αναζήτησης ώστε να διασφαλισθεί ότι δεν υπάρχουν διπλές εγγραφές.

4.3.5 Εξαντλητική ανάγνωση αρχείου

Αν οι εγγραφές της περιοχής υπερχείλισης είναι σχετικά λίγες, τότε ο απαιτούμενος χρόνος για την ανάγνωση του αρχείου είναι:

$$T_{\text{εξαν}} = b \times ebt$$

Συνεπώς, δεν ωφελεί να διατηρείται το αρχείο ταξινομημένο, αν τελικά πρόκειται να χρησιμοποιείται κυρίως για στατιστική επεξεργασία. Όμως, αν το αρχείο χρησιμοποιείται για εφαρμογές, όπως λόγου χάριν για έναν τηλεφωνικό κατάλογο ή μία ταχυδρομική λίστα, τότε η ταξινόμησή του είναι αναγκαία.

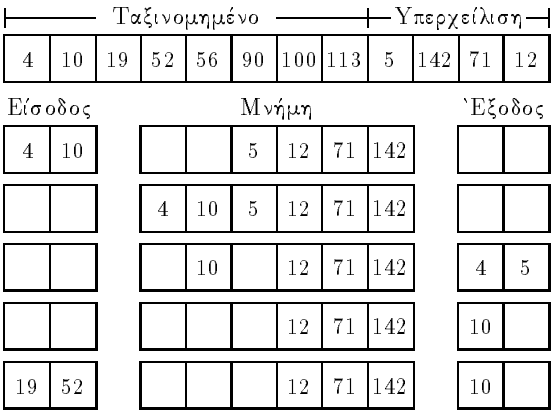
Αν υπάρχουν εγγραφές υπερχείλισης, τότε ο χρόνος εξαντλητικής ανάγνωσης του αρχείου είναι το άθροισμα του χρόνου ανάγνωσης των εγγραφών υπερχείλισης στην κύρια μνήμη (με την προϋπόθεση ότι χωρούν) συν το χρόνο ταξινόμησή τους συν το χρόνο συγχώνευσης των εγγραφών υπερχείλισης με τις εγγραφές του ταξινομημένου αρχείου. Αν δεν χωρούν όλες οι εγγραφές υπερχείλισης ταυτόχρονα στην κύρια μνήμη, τότε η επίδοση εκφυλίζεται και πρέπει να γίνει αναδιοργάνωση.

4.3.6 Αναδιοργάνωση αρχείου

Αν χρησιμοποιείται μόνο μία συσκευή δίσκου, τότε ο χρόνος αναδιοργάνωσης του αρχείου είναι:

$$T_{\text{αναδ}} = b_o \times ebt + (z \log z) \times 0.01 + b_m \times ebt + \frac{ebt \times n}{Bfr}$$

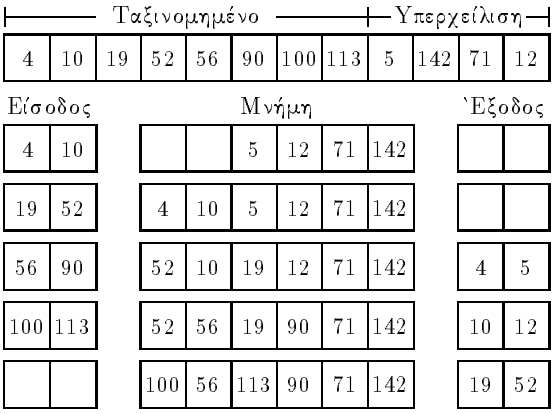
Στον τύπο αυτό ο πρώτος όρος δίνει το χρονικό κόστος για την ανάγνωση της περιοχής υπερχείλισης (υποτίθεται ότι χωρά στην κύρια μνήμη). Ο δεύτερος όρος παριστά το χρόνο ταξινόμησης των z εγγραφών υπερχείλισης χρησιμοποιώντας έναν αλγόριθμο τάξης $O(n \log n)$, με βάση την υπόθεση ότι κάθε κλήση της ρουτίνας απαιτεί 10 ms. Ο τρίτος όρος δίνει το χρόνο ανάγνωσης του ταξινομημένου αρχείου, ενώ ο τέταρτος όρος δίνει το χρόνο επανα-αποθήκευσης του αρχείου στο δίσκο υπό τη νέα του μορφή. Η διαδικασία αυτή παρουσιάζεται στο Σχήμα 4.3. Αν υπάρχουν διαθέσιμες δύο συσκευές δίσκων, τότε μπορεί μερικές διεργασίες να εκτελεσθούν παράλληλα χρησιμοποιώντας την τεχνική της διπλής απομονωτικής μνήμης. Στο Σχήμα 4.4 εφαρμόζεται η τεχνική αυτή για το παράδειγμα του Σχήματος 4.3.



Σχήμα 4.3: Αναδιοργάνωση ταξινομημένου αρχείου με υπερχείλιση που χωρά στην κύρια μνήμη και μία συσκευή δίσκου.

Συμπερασματικά, για τα ταξινομημένα αρχεία σημειώνεται ότι:

- είναι πολύ χρήσιμα για σχετικά μικρά αρχεία και για στατικά δεδομένα, όπου δηλαδή δεν πρόκειται να υπάρξουν πολλές εισαγωγές στην



Σχήμα 4.4: Αναδιοργάνωση ταξινομημένου αρχείου με υπερχείλιση που χωρά στην κύρια μνήμη και δύο συσκευές δίσκου.

περιοχή υπερχείλισης. Για παράδειγμα, τα δεδομένα ενός τηλεφωνικού καταλόγου ή μίας ταχυδρομικής λίστας είναι σχετικά στατικά.

- Η αναζήτηση μίας εγγραφής είναι γρήγορη, αφού απαιτεί $\log b$ προσπελάσεις σελίδων, αν γίνει με δυαδική αναζήτηση.

- Η αναζήτηση της επόμενης εγγραφής από την τρέχουσα είναι μία επίσης γρήγορη διεργασία, όπως επίσης και η εισαγωγή μίας νέας εγγραφής, αρκεί να μην υπάρξουν πολλές εισαγωγές, γιατί τότε θα χρειασθεί αναδιοργάνωση.

- Σε σχέση με το αρχείο σωρού, δεν υπάρχει κέρδος σε στατιστικές εφαρμογές, αλλά υπάρχουν σημαντικά πλεονεκτήματα σε εφαρμογές, όπως η εύρεση της τομής δύο αρχείων ή ο περιορισμός των διπλών εγγραφών.

Όταν η ανάκτηση μεμονωμένων εγγραφών είναι πολύ συχνή, τότε το ταξινομημένο αρχείο δεν ενδείκνυται. Όπως θα φανεί και σε επόμενα κεφάλαια, ένα αρχείο που χρησιμοποιεί κατάλογο είναι γενικά περισσότερο αποτελεσματικό.

4.4 Ασκήσεις

<1> Δίνεται αρχείο σωρού με 100.000 εγγραφές των 400 bytes. Για κάθε μία διαγραφή αντιστοιχούν τρεις εισαγωγές εγγραφών και έτσι τελικά το αρχείο αποκτά 150.000 εγγραφές. Πόσος χρόνος χρειάζεται για αναδιοργάνωση; Πόσος χρόνος απαιτείται για την εύρεση μίας εγγραφής πριν και μετά την αναδιοργάνωση;

<2> Αρχείο σωρού αποτελείται από 6.000.000 εγγραφές των 400 bytes. Έστω ότι υπάρχουν 500.000 διπλοεγγραφές. Να βρεθεί μία τεχνική για την απομάκρυνσή τους. Πόσο χρόνος θα χρειασθεί για την επεξεργασία αυτή, αν το μέγεθος της κύριας μνήμης είναι 10 Mb και ο δίσκος έχει τα χαρακτηριστικά του IBM 3380, αλλά μπορεί να αποθηκεύσει τόσο το αρχικό αρχείο, όσο και το παράγωγό του.

<3> Να λυθεί η προηγούμενη άσκηση για ταξινομημένο αρχείο.

<4> Το 10% των εγγραφών του αρχείου της Άσκησης 2 πρέπει να απομονωθούν σε ιδιαίτερο αρχείο σύμφωνα με μία συγκεκριμένη τιμή σε ένα δευτερεύον πεδίο. Πόσος χρόνος απαιτείται γι΄ αυτή την επεξεργασία; Αλλάζει το κόστος αν το αρχικό αρχείο είναι ταξινομημένο ως προς τις τιμές του δευτερεύοντος αυτού πεδίου;

<5> Έστω ότι σε ένα αταξινόμητο και σε ένα ταξινομημένο σειριακό αρχείο πρόκειται να εισαχθεί μία ομάδα εγγραφών ως σύνολο και όχι μία προς μία. Να υπολογισθεί το αντίστοιχο κόστος.

<6> Δίνεται ένα ταξινομημένο αρχείο με 100.000 εγγραφές των 400 bytes, το οποίο είναι αποθηκευμένο σε δίσκο IBM 3380. Ποιά από τις δύο επόμενες μεθόδους είναι προτιμότερη για την ανάκτηση 10 εγγραφών:

- να γίνει δυαδική αναζήτηση ανεξάρτητα για κάθε εγγραφή, ή
- να ταξινομηθούν τα κλειδιά των εγγραφών και να αναζητηθούν με μία σειριακή ανάγνωση του αρχείου;

Ποιά μέθοδος είναι προτιμότερη αν οι αναζητούμενες εγγραφές είναι 100, 1000 ή 10.000; Να προταθούν αποτελεσματικότεροι τρόποι ανάκτησης ενός συνόλου εγγραφών. Να θεωρηθεί ότι π χρόνος που απαιτείται για την ταξινόμηση z εγγραφών είναι $(z \log z) \times$ 0.01 ms.

<7> Από το αρχείο της προηγούμενης άσκησης εκτελείται μία λογική διαγραφή 5000 εγγραφές με τη βοήθεια μίας σημαίας. Στη συνέχεια εισάγονται 5000 εγγραφές στην περιοχή υπερχείλισης. Να προταθεί και να κοστολογηθεί μία τεχνική για την αναδιοργάνωση του αρχείου.