

# Δυναμικός Κατακερματισμός

# Κατακερματισμός

## Τι αποθηκεύουμε στους κάδους;

Στα παραδείγματα δείχνουμε μόνο την τιμή του πεδίου κατακερματισμού

- Την ίδια την εγγραφή (ως τρόπος *οργάνωσης αρχείου*)
  - μέγεθος κάδου -> 1 block (ή συστοιχία από συνεχόμενα blocks)
  - *Σημείωση: αντίστοιχα και σε ένα ευρετήριο μπορούμε να έχουμε (αντί την τιμή ευρετηριοποίησης + δείκτη) την πραγματική εγγραφή (πλειάδα)*
- Οργάνωση του (αρχείου) ευρετηρίου
  - Τιμή του πεδίου κατακερματισμού + δείκτη

*Ένα bucket = block (σελίδα)*

# Κατακερματισμός

## Πρόβλημα στατικού κατακερματισμού:

Έστω  $M$  κάδους και  $r$  εγγραφές ανά κάδο - το πολύ  $M * r$  εγγραφές (αλλιώς μεγάλες αλυσίδες υπερχείλισης)

## Δυναμικός κατακερματισμός

- Επεκτατός
- Γραμμικός

# Δυναμικός Εξωτερικός Κατακερματισμός

- Δυαδική αναπαράσταση του αποτελέσματος της συνάρτησης κατακερματισμού, δηλαδή ως μια ακολουθία δυαδικών ψηφίων
- Κατανομή εγγραφών με βάση την τιμή των τελικών (ή αρχικών) ψηφίων

# Δυναμικός Κατακερματισμός (εισαγωγή)

- Το αρχείο ξεκινά με **ένα** μόνο κάδο
- Μόλις γεμίσει ένας κάδος διασπάται σε δύο κάδους με βάση **την τιμή του 1ου (ή τελευταίου) δυαδικού ψηφίου** των τιμών κατακερματισμού -- δηλαδή οι εγγραφές που το πρώτο (τελευταίο) ψηφίο της τιμής κατακερματισμού τους είναι 1 τοποθετούνται σε ένα κάδο και οι άλλες (με 0) στον άλλο
- Νέα υπερχείλιση ενός κάδου οδηγεί σε διάσπαση του με βάση το **αμέσως επόμενο δυαδικό ψηφίο** ΚΟΚ

# Παράδειγμα

Χρήση των τελευταίων bits της δυαδικής αναπαράστασης

Αποτέλεσμα συνάρτησης  
κατακερματισμού

1	000001
4	000100
5	000101
7	000111
10	001010
12	001100
15	001111
16	010000
19	010011
21	010101
32	100000
13	001101
20	010100

4 εγγραφές ανά κάδο

# Δυναμικός Κατακερματισμός

Έτσι δημιουργείται μια δυαδική δενδρική δομή που λέγεται **κατάλογος** (directory) ή **ευρετήριο** (index) με δύο ειδών κόμβους

- εσωτερικούς: που καθοδηγούν την αναζήτηση
- εξωτερικούς: που δείχνουν σε ένα κάδο

# Δυναμικός Κατακερματισμός (αναζήτηση)

## Αλγόριθμος αναζήτησης

$h$  := τιμή κατακερματισμού

$t$  := ρίζα του δέντρου

$i$  := 1

while (t εσωτερικός κόμβος)

    if (i-οστό bit του h είναι 0)

$t$  := αριστερά του t

    else  $t$  := δεξιά του t

$i$  :=  $i + 1$

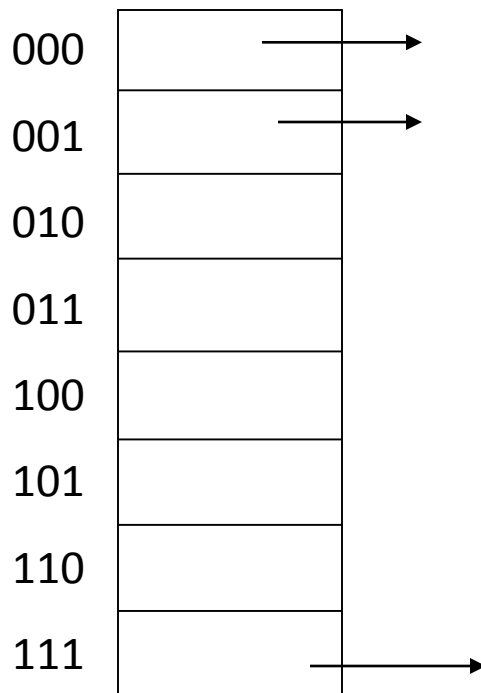


# Δυναμικός Κατακερματισμός

- Που αποθηκεύεται ο *κατάλογος*
  - στη μνήμη, εκτός αν είναι πολύ μεγάλος
  - τότε στο δίσκο – οπότε απαιτούνται επιπρόσθετες προσπελάσεις
- Δυναμική επέκταση αλλά *μέγιστος αριθμός επιπέδων* (το πλήθος των δυαδικών ψηφίων της συνάρτησης κατακερματισμού)
  - Ισοζύγιση
  - Συνένωση κάδων (δυναμική συρρίκνωση)

# Επεκτατός Κατακερματισμός (extendible hashing)

Ο κατάλογος είναι ένας πίνακας με  $2^d$  διευθύνσεις κάδων ( $d$ : ολικό βάθος του καταλόγου)



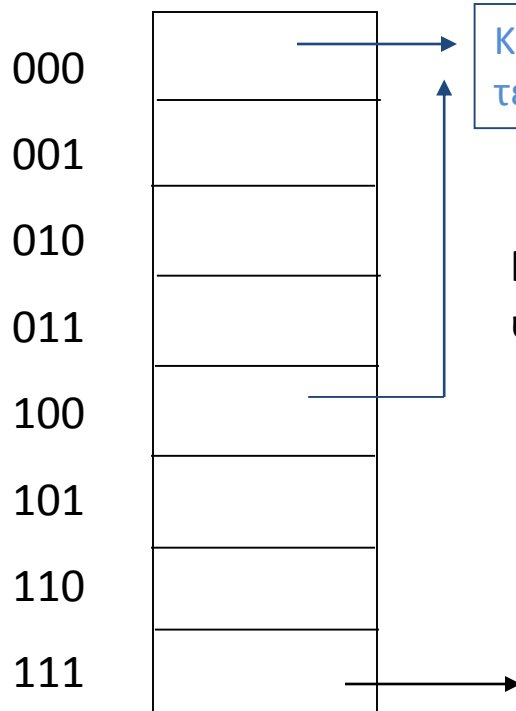
Κάδος για τις εγγραφές με τιμές κατακερματισμού που τελειώνουν σε 000

Τα τελευταία  $d$  ψηφία της τιμής κατακερματισμού χρησιμοποιούνται ως δείκτης στον πίνακα

*Στις διαφάνειες, χρησιμοποιούμε τα τελευταία bits της δυαδικής αναπαράστασης*

# Επεκτατός Κατακερματισμός

Δε χρειάζεται ένας διαφορετικός κάδος για κάθε μία από τις  $2^d$  θέσεις - μπορεί η θέση του πίνακα να δείχνει στη διεύθυνση του ίδιου κάδου αν αυτές χωράνε σε ένα κάδο



Κάδος για τις εγγραφές με τιμές κατακερματισμού που τελειώνουν από 00

Για κάθε κάδο, **τοπικό βάθος  $d'$**  ο αριθμός των δυαδικών ψηφίων στα οποία βασίζεται η χρήση του κάδου

Παράδειγμα: 2 εγγραφές ανά κάδο

εισαγωγή 2, 4, 3, 10, 7, 9

0010, 0100, 0011, 1010, 0111, 1001

# Παράδειγμα

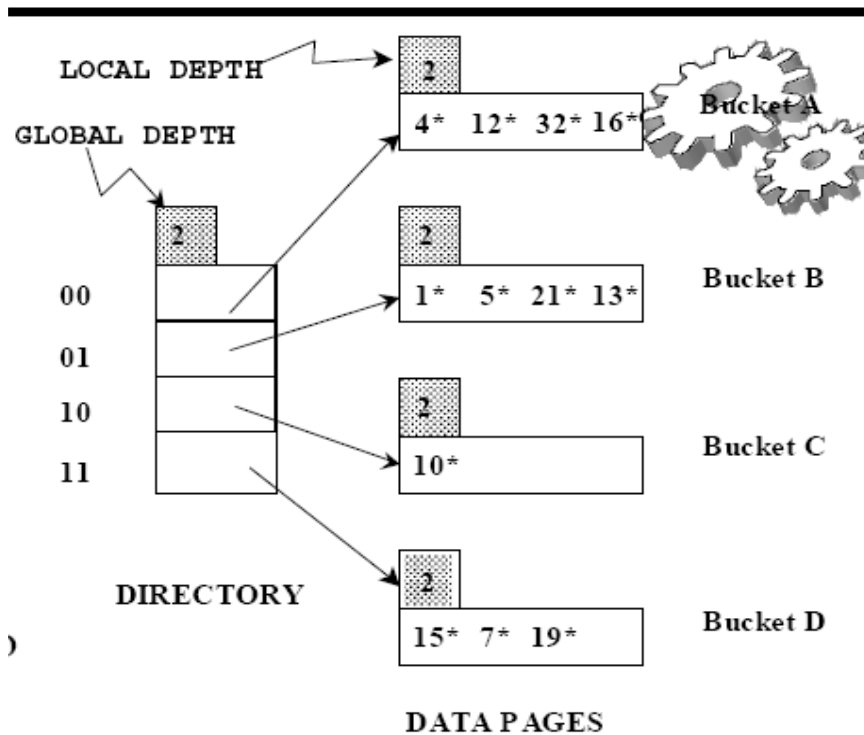
Χρήση των τελευταίων bits της δυαδικής αναπαράστασης

Αποτέλεσμα συνάρτησης  
κατακερματισμού

1	000001
4	000100
5	000101
7	000111
10	001010
12	001100
15	001111
16	010000
19	010011
21	010101
32	100000
13	001101
20	010100

4 εγγραφές ανά κάδο

# Παράδειγμα



Χρήση των τελευταίων bits της δυαδικής αναπαράστασης

1	000001
4	000100
5	000101
7	000111
10	001010
12	001100
15	001111
16	010000
19	010011
21	010101
32	100000
13	001101

**20 010100**

# Επεκτατός Κατακερματισμός

Η τιμή του  $d$  μπορεί να αυξάνεται (μέχρι  $2^k$ ,  $k$ : αριθμός δυαδικών ψηφίων της τιμής κατακερματισμού) ή να μειώνεται

## ■ Αύξηση της τιμής του $d$

Όταν ένας κάδος με τιμή  $d' = d$  υπερχειλίζει

Διπλασιασμός του πίνακα

Δε χρειάζεται rehash (επανα-κερματισμό),

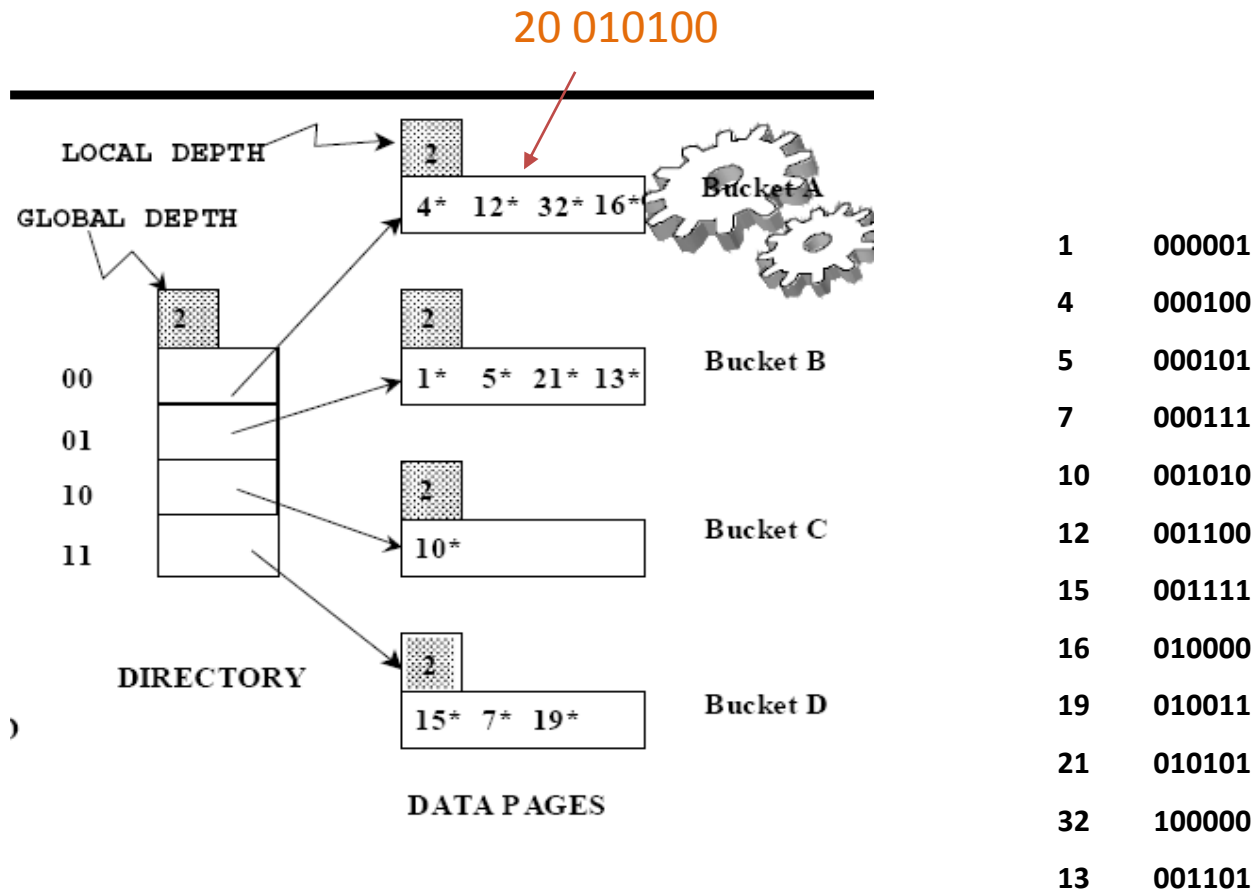
Μοιράζουμε μόνο τις εγγραφές του κάδου που υπερχείλισε

## ■ Μείωση της τιμής του $d$

Όταν για όλους τους κάδους  $d' < d$

Μείωση του μεγέθους του πίνακα στο μισό

# Παράδειγμα

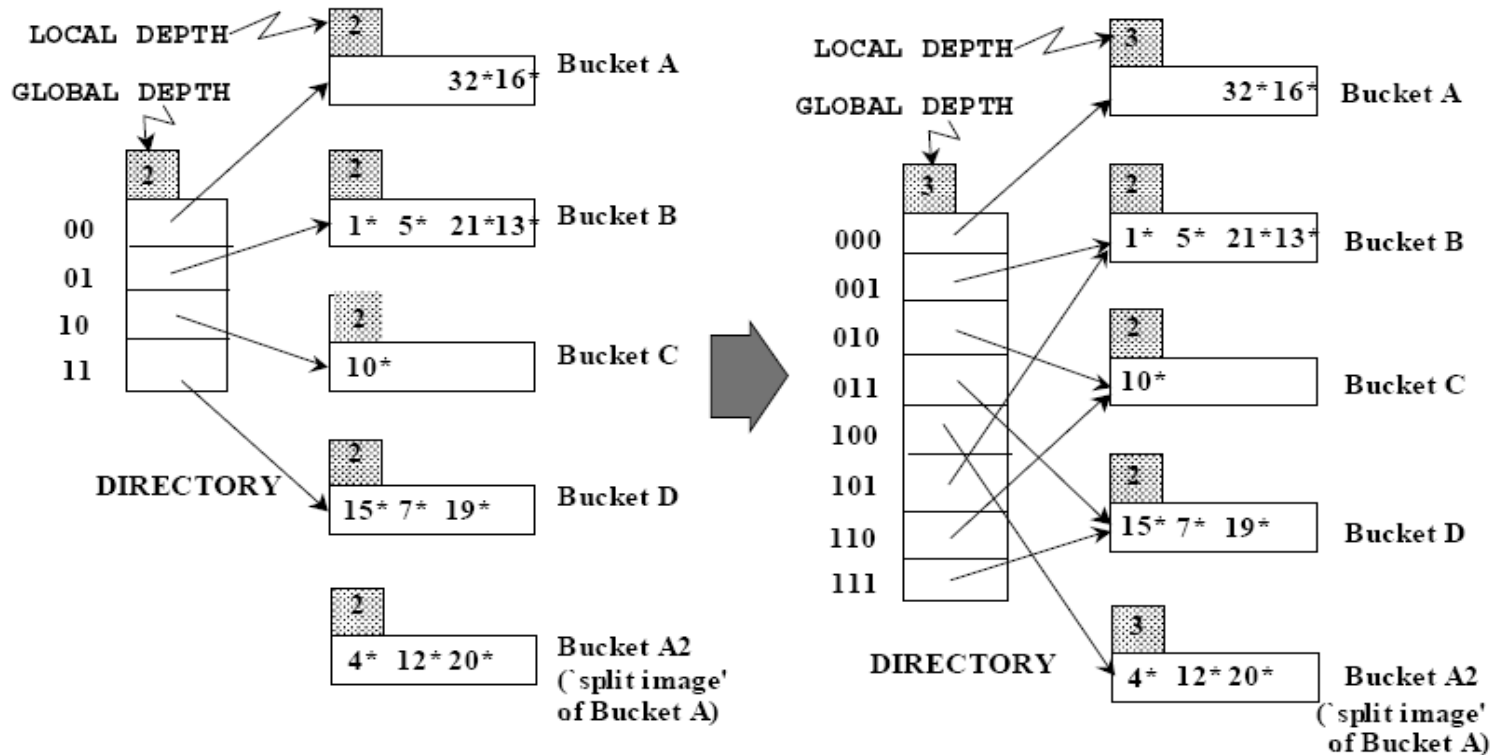


Διάσπαση-  
νέο ολικό  
βάθος 3

# Παράδειγμα

4 12 32 16 20 -> διάσπαση

1	000001
4	000100
5	000101
7	000111
10	001010
12	001100
15	001111
16	010000
19	010011
21	010101
32	100000
13	001101
20	010100





# Γραμμικός Κατακερματισμός

Θέλουμε να αποφύγουμε τη χρήση καταλόγου και το κόστος διπλασιασμού του μεγέθους του καταλόγου

Προσοχή! Αυτή η μέθοδος:

- ✓ Διατηρεί λίστες υπερχείλισης
- ✓ Δε χρησιμοποιεί τη δυαδική αναπαράσταση

# Γραμμικός Κατακερματισμός

Έστω αρχικά  $M$  κάδους

Χρησιμοποιείται μια οικογένεια από συναρτήσεις κατακερματισμού

$$h_0(k), h_1(k), \dots, h_d(k)$$

Κάθε συνάρτηση έχει διπλάσιους κάδους από την προηγούμενη:

$$h_0(k) = k \bmod M, h_1(k) = k \bmod 2M, h_2(k) = k \bmod 4M, \dots,$$

$$h_j(k) = k \bmod 2^j M$$

# Γραμμικός Κατακερματισμός (εισαγωγή)

Ξεκινάμε από την πρώτη συνάρτηση κατακερματισμού ( $h_0$ )

Όταν συμβεί *η πρώτη υπερχείλιση (συνθήκη διάσπαση)* ενός κάδου, γίνεται διάσπαση με χρήση της  $h_1$  αλλά όχι του κάδου που υπερχείλισε αλλά του κάδου **0**

Στη συνέχεια, *κάθε κάδος διασπάται με τη σειρά* (δηλαδή, κάδος **1, 2, 3**)

Στο στάδιο αυτό χρησιμοποιούνται η  $h_0$  και η  $h_1$

Μέχρι να διασπαστούν και οι 4 κάδοι

Όταν *διασπαστούν όλοι* οι κάδοι,

Οι διασπάσεις θα ξεκινούν από τον κάδο **0** με χρήση της  $h_2$

Πάλι η διάσπαση των κάδων γίνεται *με τη σειρά* (δηλαδή **0, 1, 2, ..., 7**)

Στο στάδιο αυτό χρησιμοποιούνται η  $h_1$  και η  $h_2$

Μέχρι να διασπαστούν και οι 8 κάδοι

ΚΟΚ

# Γραμμικός Κατακερματισμός

- Πολλές συναρτήσεις κατακερματισμού (άλλη σε κάθε βήμα)
- Οι κάδοι σε κάθε βήμα διασπώνται με τη σειρά (ο ένας μετά τον άλλο – ανεξάρτητα αν είναι αυτοί που έχουν ή όχι υπερχειλίσει)
- *Πότε γίνεται διάσπαση;*  
Θα θεωρήσουμε ότι γίνεται διάσπαση όταν δημιουργείται ένας κάδος υπερχειλίσης (όταν γίνεται εισαγωγή σε ένα γεμάτο κάδο)

# Γραμμικός Κατακερματισμός

Αρκούν δύο μεταβλητές

- **Βήμα Διάσπασης (j)** - ποια συνάρτηση χρησιμοποιούμε
- **Πλήθος Διασπάσεων (n)** – ποιος είναι ο επόμενος κάδος που θα διασπαστεί

Αρχικοποίηση

$j = 0; n = 0$

Όταν συμβεί μια υπερχείλιση, πρώτη διάσπαση κάδου  $n = 0 \rightarrow$  αύξηση  $n \leftarrow n+1$

Συνεχίζουμε γραμμικά, διασπώντας με τη σειρά τους κάδους 1, 2, 3, ...

μέχρι να διασπαστούν όλοι οι «παλιοί» κάδοι

*η μεταβλητή n («πλήθος διασπάσεων») κρατάει ποιος κάδος έχει σειρά για διάσπαση*

# Γραμμικός Κατακερματισμός

Όταν συμβεί μια υπερχείλιση σε έναν οποιοδήποτε κάδο,

ο κάδος  $n$  χωρίζεται σε δύο κάδους: τον αρχικό κάδο  $n$  και ένα νέο κάδο  $n + k - 1$  στο τέλος του αρχείου με βάση την συνάρτηση  $h_1(k) = k \bmod 2M$

*Δηλαδή, σε κάθε υπερχείλιση χωρίζουμε τον επόμενο στη σειρά κάδο*

# Γραμμικός Κατακερματισμός

Συνεχίζουμε ...

Όλοι οι κάδοι έχουν διασπαστεί όταν:

$$n = M$$

Τότε έχουμε  $2M$  κάδους

Όταν  $n = M$ ,

μηδενίζουμε το  $n$ ,  $n = 0$

και για οποιαδήποτε νέα διάσπαση εφαρμόζουμε την

$$h_2(k) = k \bmod 4M$$

Διασπώντας πάλι τον κάδο  $0, 1, \dots$  κ.τ.λ

# Παράδειγμα

32

9

44

31

25

5

35

7

36

14

18

10

11

30

43

Κάθε κάδος 4 εγγραφές

Αρχικά 4 κάδους ( $M = 4$ )



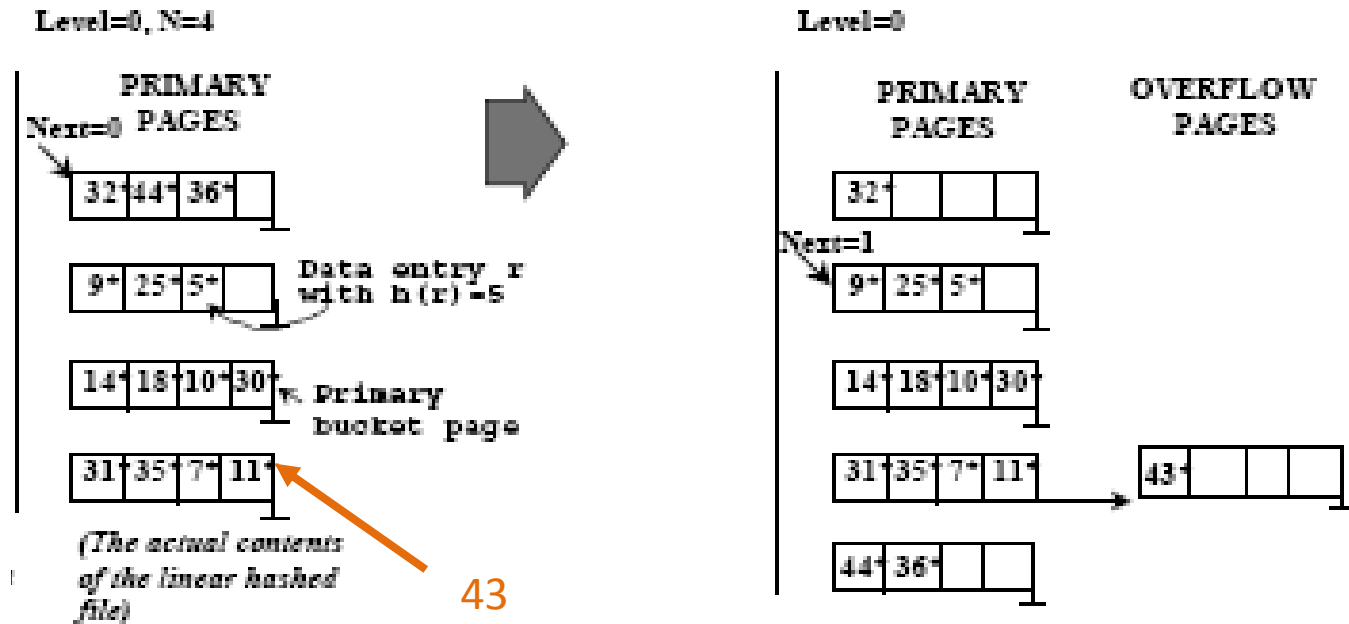
# Παράδειγμα

$$h_0(k) = k \bmod 4$$

Για μη διασπασμένους κάδους: παλιά συνάρτηση

$$h_1(k) = k \bmod 8$$

Για διασπασμένους κάδους: νέα συνάρτηση



Διασπάμε τον πρώτο κάδο

Βήμα διάσπασης 0 (χρήση  $h_0$ )

Πλήθος διασπάσεων = 0

# Γραμμικός Κατακερματισμός

Γενικά βήμα διάσπασης **j** ( $j = 0, 1, 2, \dots$ )

$h_j(k) = k \bmod 2^j M$ , και την  $h_{j+1}(k)$  για διασπάσεις

Δηλαδή, σε κάθε βήμα έχουμε ένα ζεύγος συναρτήσεων ( $j, j+1$ ):  
η πρώτη χρησιμοποιείται για τους μη διασπασμένους κάδους  
(δηλαδή, με αριθμό μεγαλύτερο του  $n$ ) και η δεύτερη για τους  
διασπασμένους

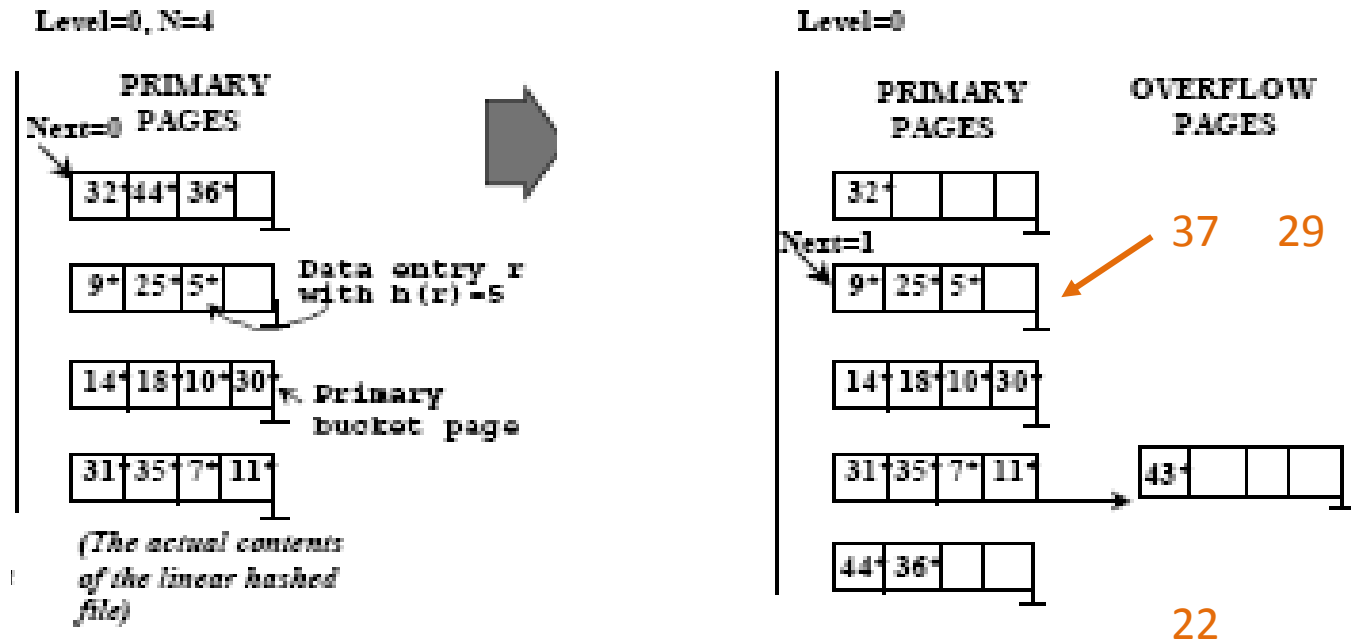
# Παράδειγμα

$$h_0(k) = k \bmod 4$$

$$h_1(k) = k \bmod 8$$

Για μη διασπασμένους κάδους: παλιά συνάρτηση

Για διασπασμένους κάδους: νέα συνάρτηση



Βήμα διάσπασης 0 (χρήση  $h_0$ )

Πλήθος διασπάσεων = 1

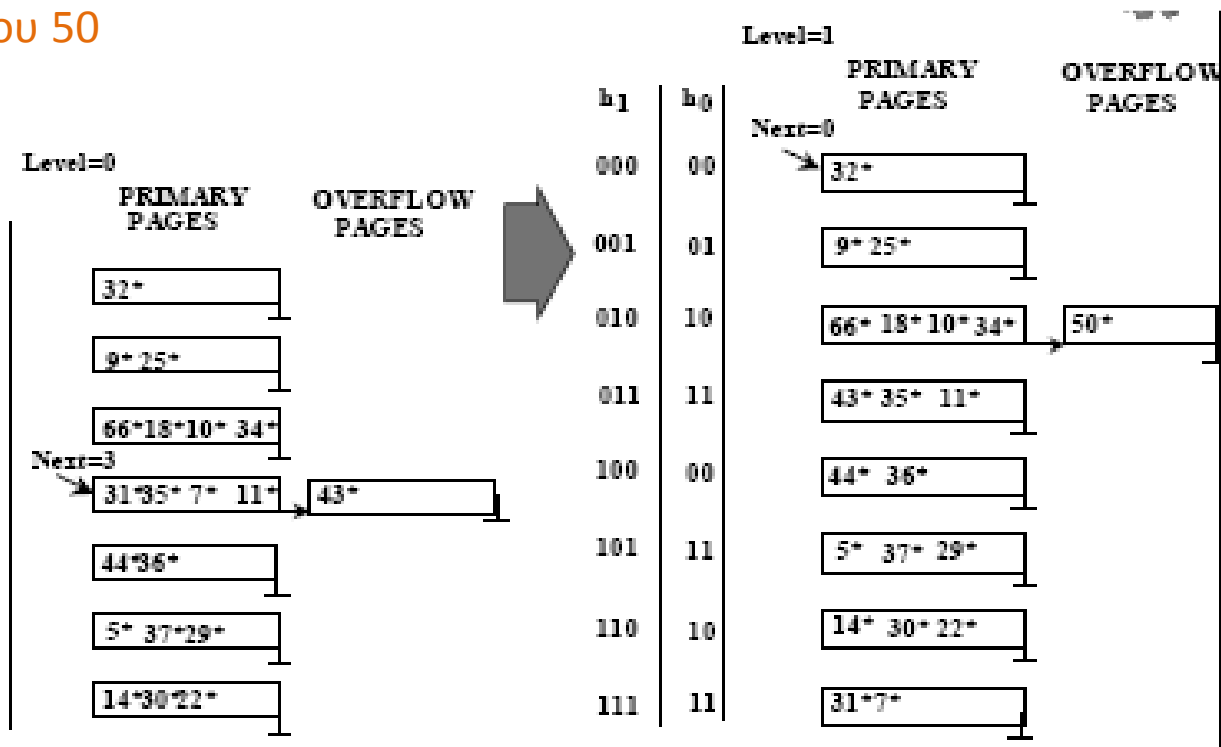
22

66

34

# Παράδειγμα

Εισαγωγή του 50



Βήμα διάσπασης 0 (χρήση  $h_0$ )

Πλήθος διασπάσεων = 3

# Γραμμικός Κατακερματισμός (αναζήτηση εγγραφής)

Τι χρειάζεται να ξέρουμε για να βρεθεί ο κάδος της εγγραφής  $k$  που ψάχνουμε;

- ποια συνάρτηση χρησιμοποιούμε (δηλαδή, το  $j$ )
- σε ποια διάσπαση βρισκόμαστε (δηλαδή το  $n$ )

Έστω ότι είμαστε στο βήμα  $j$ ,

Τότε θα πρέπει να κοιτάξουμε είτε το

$h_j(k)$  αν ο κάδος δεν έχει διασπαστεί

ή το

$h_{j+1}(k)$  αν έχει διασπαστεί

*Πως θα ελέγξουμε αν ο κάδος έχει διασπαστεί ή όχι*

# Γραμμικός Κατακερματισμός (αναζήτηση)

Έστω  $n$  ο αριθμός διασπάσεων και ότι αναζητούμε το  $k$ ,

βρίσκεται στον κάδο  $h_j(\mathbf{k})$

τότε αν  $n \leq h_j(\mathbf{k})$  ο κάδος δεν έχει διασπαστεί

ενώ αν  $n > h_j(\mathbf{k})$  ο κάδος έχει διασπαστεί και εφαρμόζουμε την  $h_{j+1}(\mathbf{k})$

# Γραμμικός Κατακερματισμός (αναζήτηση)

## Αλγόριθμος Αναζήτησης

$j$  : βήμα διάσπασης  $n$  : πλήθος διασπάσεων στο βήμα  $j$

```
if (n = 0)
  then m := hj(k);
else {
  m := hj(k);
  if (m < n) then m := hj+1(k)
}
```

σημαίνει ότι ο κάδος έχει  
διασπαστεί

# Ερωτήσεις;