ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ
ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΡΗΤΗΣ

Δομές Δεδομένων

Ιωάννης Γ. Τόλλης
Τμήμα Επιστήμης Υπολογιστών
Πανεπιστήμιο Κρήτης

Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο Πανεπιστήμιο Κρήτης**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Σημείωμα αδειοδότησης

- Το παρόν υλικό διατίθεται με τους όρους της άδειας χρήσης Creative Commons Αναφορά, Μη Εμπορική Χρήση, Όχι Παράγωγο Έργο 4.0 [1] ή μεταγενέστερη, Διεθνής Έκδοση. Εξαιρούνται τα αυτοτελή έργα τρίτων π.χ. φωτογραφίες, διαγράμματα κ.λ.π., τα οποία εμπεριέχονται σε αυτό και τα οποία αναφέρονται μαζί με τους όρους χρήσης τους στο «Σημείωμα Χρήσης Έργων Τρίτων».

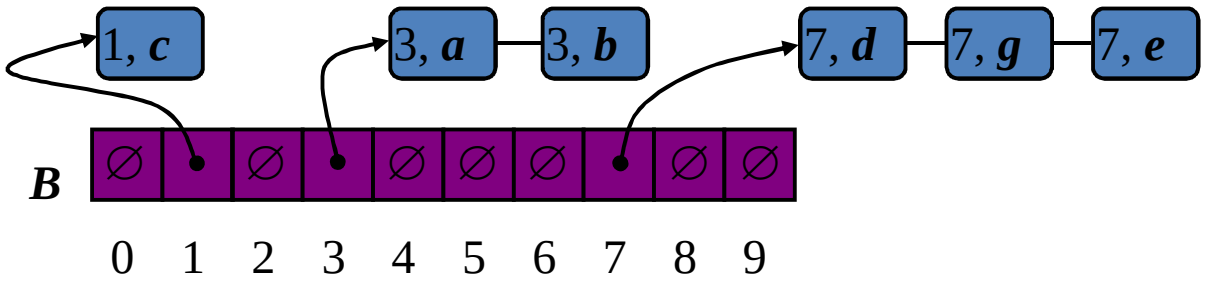
[1] http://creativecommons.org/licenses/by-nc-nd/4.0/

- Ως **Μη Εμπορική** ορίζεται η χρήση:
  - που δεν περιλαμβάνει άμεσο ή έμμεσο οικονομικό όφελος από την χρήση του έργου, για το διανομέα του έργου και αδειοδόχο
  - που δεν περιλαμβάνει οικονομική συναλλαγή ως προϋπόθεση για τη χρήση ή πρόσβαση στο έργο
  - που δεν προσπορίζει στο διανομέα του έργου και αδειοδόχο έμμεσο οικονομικό όφελος (π.χ. διαφημίσεις) από την προβολή του έργου σε διαδικτυακό τόπο

- Ο δικαιούχος μπορεί να παρέχει στον αδειοδόχο ξεχωριστή άδεια να χρησιμοποιεί το έργο για εμπορική χρήση, εφόσον αυτό του ζητηθεί.

.

Radish-Sort



Κύρια σημεία για μελέτη

- Bucket-sort (§4.5.1)
- Lexicographic order (§4.5.2)
- Lexicographic-sort (§4.5.2)
- Radish-sort (§4.5.2)
- Radicchio-sort (§4.5.2)
- Radiator-sort (§4.5.2)

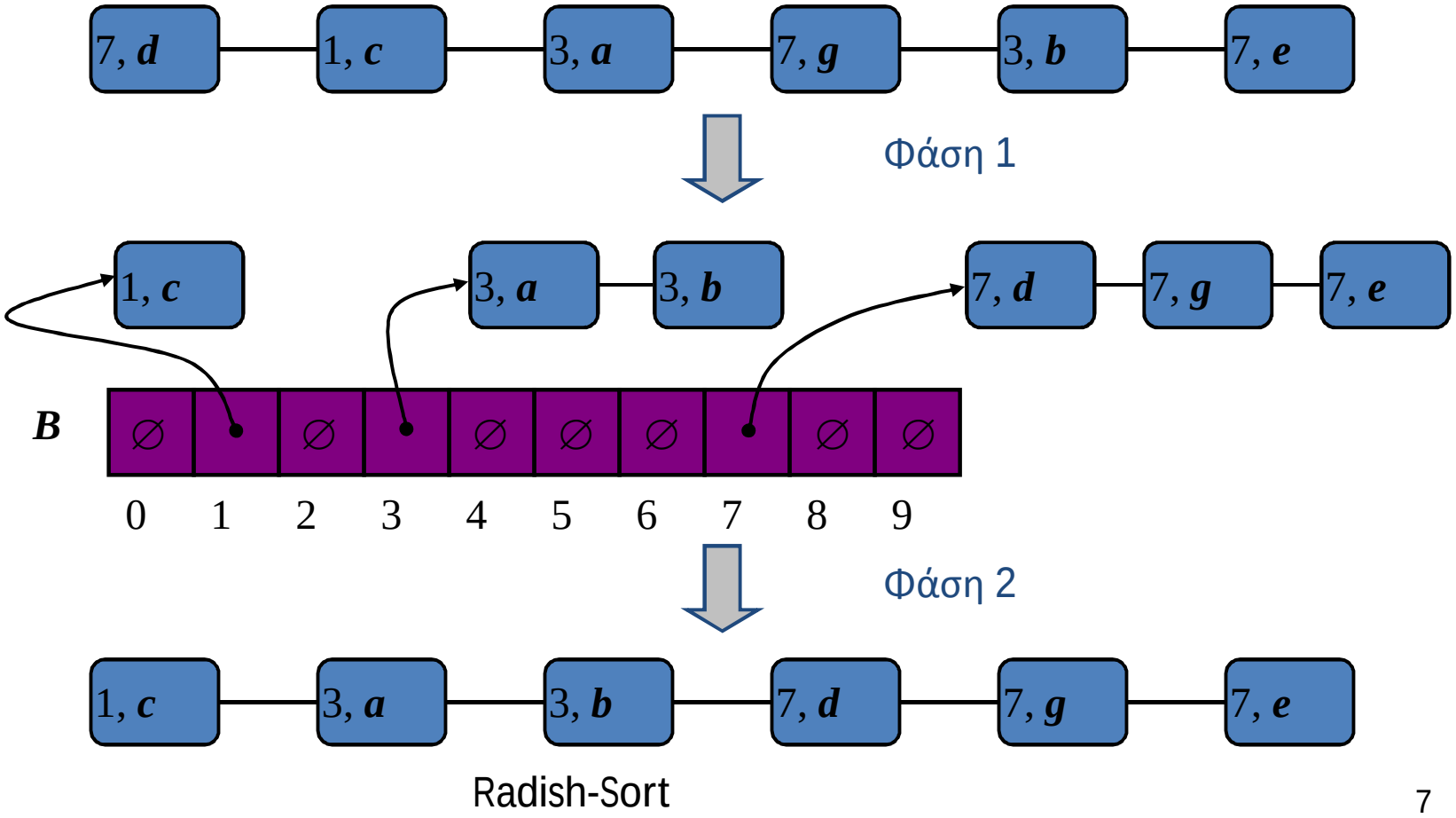
Bucket-Sort

- Έστω S μια ακολουθία (key, element) αντικειμένων με κλειδιά στο διάστημα $[0, N-1]$
- Η Bucket-sort χρησιμοποιεί τα κλειδιά σαν δείκτες σε έναν βοηθητικό πίνακα B από ακολουθίες (buckets)
  - Φάση 1: Άδειασμα ακολουθιών S μετακινώντας κάθε αντικείμενο (k, o) μέσα στο bucket του $B[k]$
  - Φάση 2:Για $i = 0, \ldots, N-1$, μετακίνησε τα αντικείμενα του bucket $B[i]$ στο τέλος της ακολουθίας S
- Ανάλυση:
  - Η φάση 1 παίρνει χρόνο $O(n)$
  - Η φάση 2 παίρνει χρόνο $O(n + N)$
  - Η Bucket-sort παίρνει χρόνο $O(n + N)$

```
Algorithm bucketSort(S, N)
    Input sequence S of (key, element)
        items with keys in the range
        [0, N – 1]
    Output sequence S sorted by
        increasing keys
    B ← array of N empty sequences
    while ¬S.isEmpty()
        f ← S.first()
        (k, o) ← S.remove(f)
        B[k].insertLast((k, o))
    for i ← 0 to N – 1
        while ¬B[i].isEmpty()
            f ← B[i].first()
            (k, o) ← B[i].remove(f)
            S.insertLast((k, o))
```

Παράδειγμα

- Εύρος κλειδιών [0, 9]



Radish-Sort

Ιδιότητες και επεκτάσεις

- Key-type ιδιότητα
  - Τα κλειδιά χρησιμοποιούνται σαν δείκτες σε πίνακα και δεν μπορούν να είναι αυθαίρετα αντικείμενα
  - Δεν υπάρχει εξωτερικός συγκριτής
- Stable Sort ιδιότητα
  - Η σχετική σειρά οποιωνδήποτε δύο αντικειμένων με το ίδιο κλειδί διατηρείται μετά την εκτέλεση του αλγορίθμου

Επεκτάσεις

- Ακέραια κλειδιά στο διάστημα $[\boldsymbol{a}, \boldsymbol{b}]$
  - Τοποθέτηση αντικειμένου $(\boldsymbol{k}, \boldsymbol{o})$ στο bucket $\boldsymbol{B}[\boldsymbol{k} - \boldsymbol{a}]$
- Τα κλειδιά συμβολοσειρών είναι από ένα σύνολο $\boldsymbol{D}$ πιθανών συμβολοσειρών, όπου $\boldsymbol{D}$ έχει σταθερό μέγεθος (π.χ., τα ονόματα των 50 πολιτειών των Η.Π.Α.)
  - Ταξινόμησε το $\boldsymbol{D}$ και υπολόγισε την τάξη $\boldsymbol{r}(\boldsymbol{k})$ της κάθε συμβολοσειράς $\boldsymbol{k}$ του $\boldsymbol{D}$ στην ταξινομημένη ακολουθία
  - Τοποθέτησε το αντικείμενο $(\boldsymbol{k}, \boldsymbol{o})$ μέσα στο bucket $\boldsymbol{B}[\boldsymbol{r}(\boldsymbol{k})]$

Lexicographic Order

- Ένα d-tuple είναι μια ακολουθία από d κλειδιά (k_1, k_2, …, k_d), όπου το κλειδί k_i είναι η i-$\tau\eta$ διάσταση του tuple
- Παράδειγμα:
  - Οι καρτεσιανές συντεταγμένες ενός σημείου στον χώρο είναι ένα 3-tuple
- Η λεξικογραφική διάταξη δύο d-tuples ορίζεται αναδρομικά ως εξής

$$(x_1, x_2, \ldots, x_d) < (y_1, y_2, \ldots, y_d)$$

$$\Leftrightarrow$$

$$x_1 < y_1 \ \vee \ x_1 = y_1 \wedge (x_2, \ldots, x_d) < (y_2, \ldots, y_d)$$

π.χ., τα tuples συγκρίνονται πρώτα ως προς την πρώτη διάσταση ,μετά ως προς την δεύτερη, κ.ο.κ.

Lexicographic-Sort

- Έστω C_i ένας συγκριτής ο οποίος συγκρίνει δύο tuples ως προς την i-th διάσταση τους
- Έστω ***stableSort***(S, C) είναι ένας σταθερός αλγόριθμος ταξινόμησης που χρησιμοποιεί τον συγκριτή C
- Η λεξικογραφική ταξινόμηση ταξινομεί μια ακολουθία d-tuples με λεξικογραφική σειρά εκτελώντας d φορές τον αλγόριθμο ***stableSort***, μία φορά για κάθε διάσταση
- Η λεξικογραφική ταξινόμηση εκτελείται σε χρόνο $O(dT(n))$, όπου $T(n)$ είναι ο χρόνος εκτέλεσης του ***stableSort***

Algorithm *lexicographicSort*(S)
  Input sequence S of d-tuples
  Output sequence S sorted in lexicographic order

  for $i \leftarrow d$ **downto** 1
    stableSort(S, C_i)

Παράδειγμα:

(7,4,6) (5,1,5) (2,4,6) (2, 1, 4) (3, 2, 4)

(2, 1, 4) (3, 2, 4) (5,1,5) (7,4,6) (2,4,6)

(2, 1, 4) (5,1,5) (3, 2, 4) (7,4,6) (2,4,6)

(2, 1, 4) (2,4,6) (3, 2, 4) (5,1,5) (7,4,6)

Radish-Sort

- Η Radish-sort είναι μια άλλη εκδοχή της λεξικογραφικής ταξινόμησης που χρησιμοποιεί bucket-sort σαν τον σταθερό αλγόριθμο ταξινόμησης σε κάθε διάσταση
- Η Radish-sort είναι εφαρμόσιμη σε tuples όπου τα κλειδιά σε κάθε διάσταση i είναι ακέραιοι στο διάστημα $[0, N-1]$
- Η Radish-sort εκτελείται σε χρόνο $O(d(n+N))$

Algorithm *radishSort*(S, N)
  Input sequence S of d-tuples such that $(0, \ldots, 0) \leq (x_1, \ldots, x_d)$ and $(x_1, \ldots, x_d) \leq (N-1, \ldots, N-1)$ for each tuple $(x_1, \ldots, x_d)$ in S
  Output sequence S sorted in lexicographic order
  for $i \leftarrow d$ **downto** 1
    bucketSort(S, N)

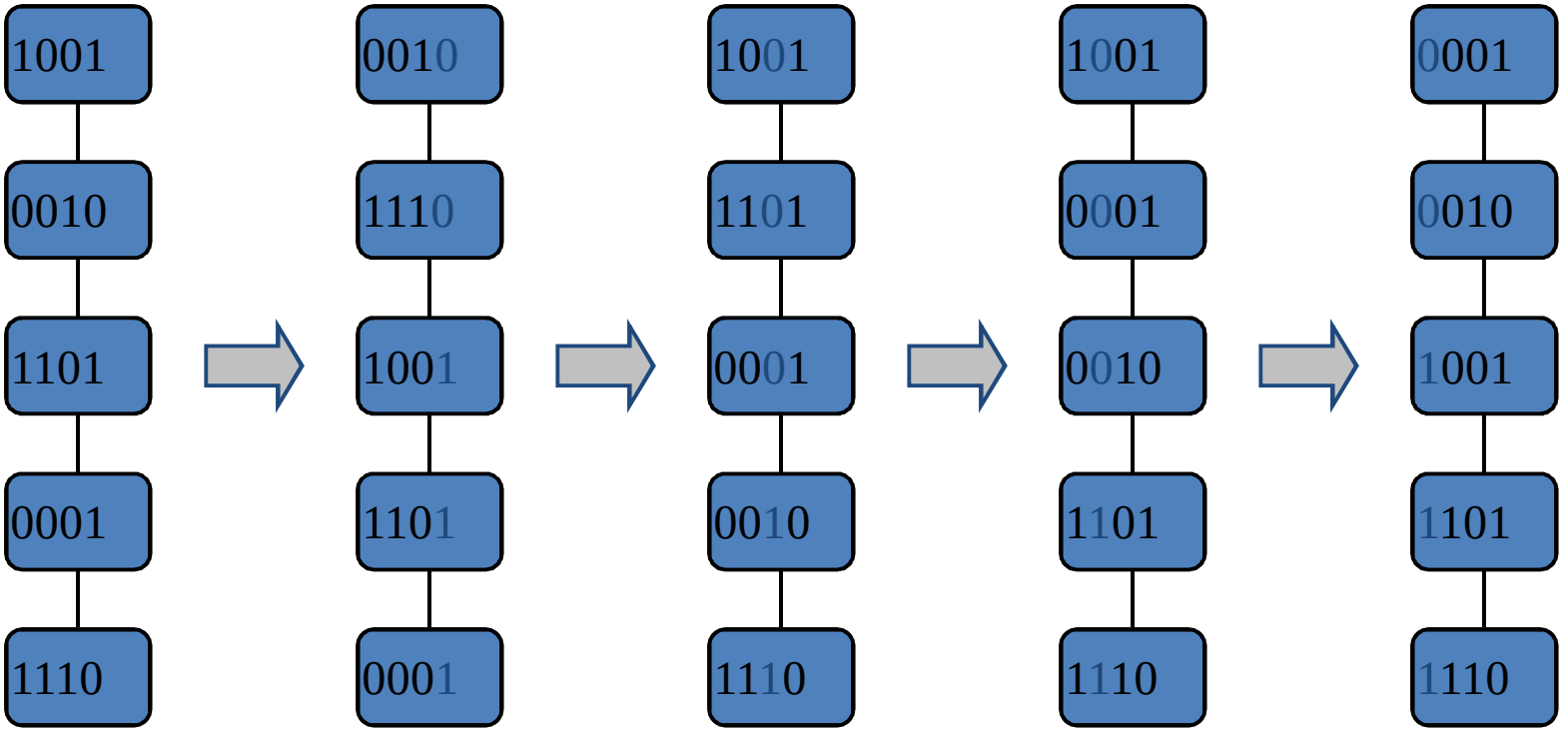
Radicchio-Sort

- Θεωρείστε μια ακολουθία από n b-bit ακέραιους
  $x = x_{b-1} \ldots x_1 x_0$
- Αναπαριστούμε κάθε στοιχείο σαν ένα b-tuple από ακέραιους στο διάστημα $[0, 1]$ και εφαρμόζουμε radish sort με $N = 2$
- Αυτός ο αλγόριθμος ονομάζεται radicchio-sort και εκτελείται σε χρόνο $O(bn)$
- Με τον radicchio-sort, μπορούμε να ταξινομήσουμε μια ακολουθία από Java ακέραιους (32-bits) σε γραμμικό χρόνο

Algorithm *radicchioSort*(S)
  Input sequence S of b-bit integers
  Output sequence S sorted
  replace each element x of S with the item $(0, x)$
  for $i \leftarrow 0$ **to** $b - 1$
    replace the key k of each item (k, x) of S with bit x_i of x
    bucketSort(S, 2)

Παράδειγμα

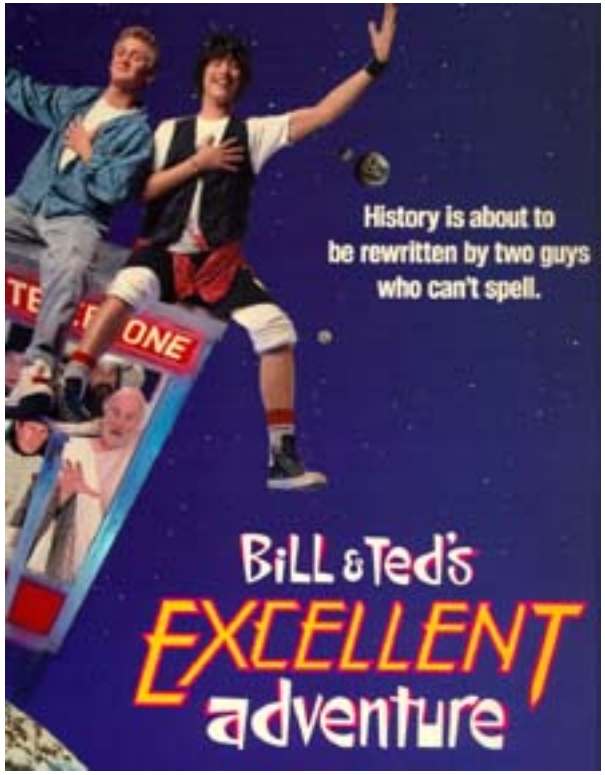
- Ταξινόμηση μιας ακολουθίας τετράμπιτων ακεραίων



Επεκτάσεις

- Radiator-sort
  - Τα κλειδιά είναι ακέραιοι στο διάστημα $[0, N^2 - 1]$
  - Αναπαριστούμε ένα κλειδί ως μια 2-tuple ψηφίων στο διάστημα $[0, N - 1]$ και εφαρμόζουμε τον radish-sort
  - Παράδειγμα ($N = 10$):
    - $75 \rightarrow (7, 5)$
  - Παράδειγμα ($N = 8$):
    - $35 \rightarrow (4, 3)$
  - Ο χρόνος εκτέλεσης του radiator-sort είναι $O(n + N)$
  - Μπορεί να επεκταθεί σε ακέραια κλειδιά στο διάστημα $[0, N^d - 1]$
- Radiation-sort
  - Τα κλειδιά είναι συμβολοσειρές των d χαρακτήρων το καθένα
  - Αναπαριστούμε κάθε κλειδί με ένα d-tuple από ακέραιους, όπου είναι η ASCII (8-bit integer) ή η Unicode (16-bit integer) αναπαράσταση του i-ου χαρακτήρα και εφαρμόζουμε radish sort
- Rant-sort
  - Δες στο βιβλίο

Συμπεράσματα



Τέλος Ενότητας