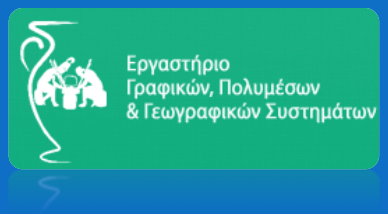
Εισαγωγή στις Διαχρονικές Δομές Δεδομένων

Persistent Data Structures

Δομές Δεδομένων

Μάριος Κενδέα



19 Μαΐου 2015

kendea@ceid.upatras.gr

Εισαγωγή (1/3)

- **Εφήμερες δομές:**
  - Δεν έχουμε πρόσβαση στις προηγούμενες καταστάσεις (εκδοχές) της δομής.
  - Η παλιά έκδοση καταστρέφεται μετά από μία αλλαγή.

- **Διαχρονικές Δομές - Persistent:**
  - Επιτρέπει την πρόσβαση σε όλες τις εκδοχές της δομής.
  - Που χρειαζόμαστε τις διαχρονικές δομές;
    - Επεξεργασία κειμένου και αρχείων.
    - Υπολογιστική Γεωμετρία κ.α.

Εισαγωγή (2/3)

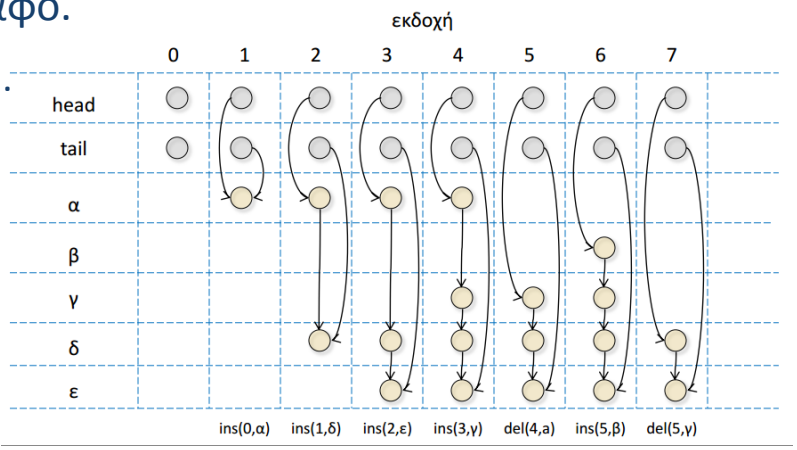
- **Μερική Διαχρονικότητα – Partial Persistent:**
  - Επιτρέπει την πρόσβαση σε όλες τις εκδοχές της δομής, αλλά είναι δυνατόν να τροποποιηθεί μόνο η τελευταία εκδοχή.
- **Πλήρης Διαχρονικότητα – Fully Persistent:**
  - Επιτρέπει τόσο την πρόσβαση όσο την τροποποίηση σε όλες τις εκδοχές της δομής.

Εισαγωγή (3/3)

- Για να υποστηρίξουμε τη διαχρονικότητα της δομής εισάγουμε την ακόλουθη παράμετρο :
  - **Αριθμός εκδοχής:**
    - Η αρχική δομή αποτελεί την εκδοχή 0.
    - Η i-οστή πράξη τροποποίησης δημιουργεί την εκδοχή i.
- **Παράμετροι απόδοσης:**
  - n = αριθμός στοιχείων στην τρέχουσα εκδοχή
  - m = συνολικός αριθμός τροποποιήσεων

Απλοϊκές Λύσεις

1. Κάθε εκδοχή δημιουργεί ένα πλήρες αντίγραφο.
   - Απαιτεί Ω(n) χρόνο και χώρο ανά τροποποίηση.

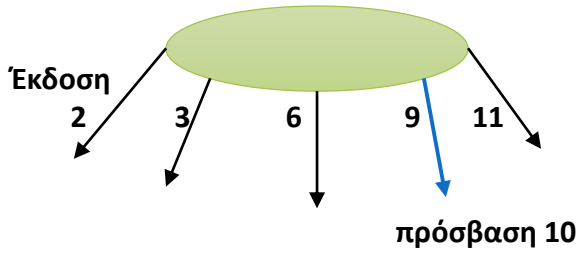


http://www.cs.uoi.gr/~loukas/courses/grad/Data_Structures_and_Algorithms/index.files/Persistence.pdf

2. Κάθε εκδοχή δημιουργεί ένα πλήρες αντίγραφο.
   - Απαιτεί Ω(n) χρόνο και χώρο ανά τροποποίηση.
3. Δεν αποθηκεύουμε καμία εκδοχή, αλλά μόνο την ακολουθία τροποποιήσεων. Για πρόσβαση στην i-οστή εκδοχή κατασκευάζουμε τη δομή έως αυτή την εκδοχή από την αρχή.
   - Απαιτεί Ω(i) χρόνο για την κάθε πρόσβαση στην i-οστή εκδοχή.
4. Συνδυασμός των δύο παραπάνω

Μερική Διαχρονικότητα (1/2)

- **Μέθοδος παχιού κόμβου (Fat Node)**
  - Κάθε κόμβος αποθηκεύει αυθαίρετο αριθμό από τιμές (ετικέτες έκδοσης)
  - Έχει ετικέτα που υποδεικνύει την έκδοση που δημιουργήθηκε ο ίδιος
  - Εύρεση της i-οστής έκδοσης:
    - Ξεκινάμε από τον κατάλληλο δείκτη πρόσβασης της έκδοσης
    - Όταν βρισκόμαστε σε ένα διαχρονικό κόμβο P και θέλουμε να ανακτήσουμε μία τιμή ενός πεδίου αναζητούμε την τιμή του πεδίου με μέγιστη ετικέτα <=i.
      - O(log m) χρόνος για m εκδόσεις αν οι τιμές εκδόσεων είναι οργανωμένες σε δυαδικό δέντρο)
      - Χωρική Επιβάρυνση: O(1) ανά έκδοση



Μερική Διαχρονικότητα (2/2)

- **Μέθοδος αντιγραφής κόμβων (Node Copying)**
  - Καλύτερη χρονική επιβάρυνση κατά την διάρκεια της προσπέλασης.
  - Σε αυτή την τεχνική, ο κάθε κόμβος έχει σταθερό χώρο.
  - Όταν γίνεται μία αλλαγή έκδοσης
    - Καταγράφεται στον κόμβο αν έχει χώρο
    - Αλλιώς δημιουργείται ένας νέος κόμβος που περιέχει μόνο την τελευταία έκδοση του κόμβου.
  - Πλέον η ιστορικότητα καταγράφεται σε λίστα από κόμβους.
  - Δημιουργούνται δείκτες από τους προηγούμενους- προγόνους στον νέο κόμβο.
    - Αν δεν έχουν χώρο δημιουργούνται αντίγραφα.
  - Χρονικό κόστος:
    - Αναζήτηση και Ενημέρωση O(1) επιμερισμένη
    - Μετακίνηση στις εκδόσεις είναι O(1)
  - Χωρική Επιβάρυνση:
    - O(1)

Πλήρης Διαχρονικότητα

- **Χρήση μεθόδου fat node**
  - Χρονικό κόστος:
    - O(log m) επιβάρυνση για πρόσβαση
    - και O(1) για ενημέρωση
  - Χωρικό κόστος: O(1)
- **Χρήση μεθόδου διάσπασης κόμβου**
  - Χρονικό κόστος:
    - O(1) επιμερισμένη επιβάρυνση για αναζήτηση
    - και O(1) για ενημέρωση
  - Χωρικό κόστος: O(1)