Δομές Δεδομένων και Αλγόριθμοι

Χρήστος Γκόγκος

ΤΕΙ Ηπείρου

Χειμερινό Εξάμηνο 2014-2015
Παρουσίαση 11. Αναζήτηση

έκδοση 1.0

Αλγόριθμοι αναζήτησης

1. Σειριακή αναζήτηση (sequential search)
2. Αναζήτηση με αναπηδήσεις (jump search)
3. Δυαδική αναζήτηση(binary search)
4. Αναζήτηση με παρεμβολή (interpolation search)

Σειριακή αναζήτηση

Η σειριακή ή γραμμική αναζήτηση είναι ο απλούστερος αλγόριθμος αναζήτησης. Τα στοιχεία του πίνακα εξετάζονται στην σειρά μέχρι να εντοπιστεί το στοιχείο που αναζητείται.
Έχει πολυπλοκότητα χειρότερης περίπτωσης $O(n)$
Μπορεί να εφαρμοστεί σε μη ταξινομημένους πίνακες

```
template<class T> int sequential_search(T a[], int n, T key) {
    for (int i = 0; i < n; i++) {
        if (a[i] == key)
            return i;
    }
    return -1;
}
```

Αναζήτηση με αναπηδήσεις (jump search)

Η αναζήτηση με αναπηδήσεις μπορεί να εφαρμοστεί μόνο σε ταξινομημένα δεδομένα. Επιλέγεται ένα βήμα k και γίνεται σταδιακός έλεγχος των στοιχείων a[0], a[k-1], a[2k-1], ...
Όταν βρεθεί ότι η τιμή που αναζητείται είναι ανάμεσα σε 2 από τις παραπάνω τιμές τότε γίνεται σειριακή αναζήτηση στο διάστημα αυτό.
Αν επιλεγεί ως βήμα η τιμή $\sqrt{n}$ όπου n το πλήθος των στοιχείων της ακολουθίας τότε η πολυπλοκότητα χειρότερης περίπτωσης για τον αλγόριθμο είναι $O(\sqrt{n})$

Αναζήτηση με αναπηδήσεις - κώδικας C++

```
template<class T> int jump_search(T a[], int n, T key) {
    int step = (int) sqrt(n);
    int p1 = 0;
    int p2 = step;
    while (a[p2] < key) {
        if (a[p1] >= key) break;
        p1 = p2 + 1;
        p2 += step;
        if (p2 > n - 1)
            p2 = n - 1;
    }
    for (int i = p1; i <= p2; i++)
        if (a[i] == key) return i;
    return -1;
}
```

Δυαδική αναζήτηση

Η δυαδική αναζήτηση μπορεί να εφαρμοστεί μόνο σε ταξινομημένα δεδομένα. Διαιρεί επαναληπτικά την ακολουθία σε 2 υποακολουθίες και απορρίπτει την ακολουθία στην οποία συμπεραίνει ότι δεν μπορεί να βρεθεί το στοιχείο
Έχει πολυπλοκότητα χειρότερης περίπτωσης $O(\log n)$

Δυαδική αναζήτηση - κώδικας C++

```
template<class T> int binary_search(T a[], int l, int r, T key) {
    int m = (l + r) / 2;
    if (l > r) {
        return -1;
    } else if (a[m] == key) {
        return m;
    } else if (key < a[m]) {
        return binary_search(a, l, m - 1, key);
    } else {
        return binary_search(a, m + 1, r, key);
    }
}
```

Αναζήτηση με παρεμβολή

Η αναζήτηση με παρεμβολή μπορεί να εφαρμοστεί μόνο σε ταξινομημένα δεδομένα. Αντί να χρησιμοποιηθεί η τιμή 0,5 για να μοιραστούν τα δεδομένα σε 2 λίστες όπως στη Δυαδική Αναζήτηση υπολογίζεται μια τιμή η οποία εκτιμάται ότι θα μας οδηγήσει πλησιέστερα στο αντικείμενο που αναζητείται.
Αν l είναι ο δείκτης του αριστερότερου στοιχείου της ακολουθίας και r ο δείκτης του δεξιότερου στοιχείου της ακολουθίας τότε υπολογίζεται ο συντελεστής $c = \frac{key - a[l]}{a[r] - a[l]}$ όπου key είναι το στοιχείο προς αναζήτηση και a είναι η ακολουθία τιμών στην οποία αναζητείται.
Έχει πολυπλοκότητα χειρότερης περίπτωσης $O(n)$ αλλά αν τα δεδομένα της ακολουθίας είναι ομοιόμορφα κατανεμημένα σε ένα εύρος τιμών τότε η πολυπλοκότητα του αλγορίθμου γίνεται $O(\log \log n)$

Αναζήτηση με παρεμβολή - κώδικας C++

```
template<class T> int interpolation_search(T a[], int l, int r, T
    key) {
  int m;
  if (l > r) {
    return -1;
  } else if (l == r) {
    m = l;
  } else {
    double c = (double) (key - a[l]) / (double) (a[r] - a[l]);
    if ((c < 0) || (c > 1))
      return -1;
    m = (int) (l + (r - l) * c);
  }
  if (a[m] == key) {
    return m;
  } else if (key < a[m]) {
    return interpolation_search(a, l, m - 1, key);
  } else {
    return interpolation_search(a, m + 1, r, key);
  }
}
```