ΠΛΗ30

ΕΝΟΤΗΤΑ 1: ΑΝΑΛΥΣΗ ΑΛΓΟΡΙΘΜΩΝ

Μάθημα 1.4:
Ανάλυση Αναδρομικών Αλγορίθμων
Η αναδρομική σχέση T(n)=aT(n/b)+f(n)

Δημήτρης Ψούνης

www.psounis.gr

ΠΕΡΙΕΧΟΜΕΝΑ

Α. Σκοπός του Μαθήματος

Οι στόχοι του μαθήματος είναι:

Επίπεδο Α

- Το θεώρημα κυριαρχίας για την επίλυση της αναδρομικής σχέσης $T(n)=aT(n/b)+f(n)$

Επίπεδο Β

- Η μέθοδος επανάληψης για την επίλυση της αναδρομικής σχέσης $T(n)=aT(n/b)+f(n)$

Επίπεδο Γ

- Ο αλγόριθμος ταξινόμησης MergeSort
- Ο αλγόριθμος αναζήτησης BinarySearch

Β. Θεωρία
1. Αναδρομικοί Αλγόριθμοι

- Γενικότερα ένας αναδρομικός αλγόριθμος είναι ένας αλγόριθμος ο οποίος υλοποιείται από μία αναδρομική διαδικασία.
  - Αναδρομική λέγεται μια διαδικασία που κατά την διάρκεια της εκτέλεσής της καλεί τον εαυτό της.
- Η γενική μορφή ενός αναδρομικού κώδικα φαίνεται στο σχήμα. Παρατηρήστε ότι σε κάποιο σημείο του σώματος της διαδικασίας πρέπει να γίνεται κλήση στην ίδια την συνάρτηση:

```
procedure recursive(n)
    ...
    ...
      ΚΛΗΣΗ recursive(n-1)
    ...
    ...
end procedure
```

- Θα μελετήσουμε δύο περίφημους αναδρομικούς αλγόριθμους:
  - Τον BinarySearch για την αναζήτηση στοιχείου σε μία ακολουθία
  - Τον MergeSort για την ταξινόμηση ενός πίνακα αριθμών

Β. Θεωρία

1. Αναδρομικοί Αλγόριθμοι

1. Ο αλγόριθμος αναζήτησης BinarySearch (1.Διατύπωση και Λειτουργία)

- BinarySearch ή Δυαδική Αναζήτηση:
  - Είσοδος: Ταξινόμημένος πίνακας A, στοιχείο x
  - Έξοδος:
    - Αν το στοιχείο υπάρχει στον πίνακα, επιστρέφεται η θέση του στοιχείου x στον πίνακα A.
    - Αν το στοιχείο δεν υπάρχει στον πίνακα, επιστρέφεται 0.

- Λειτουργία του αλγορίθμου: Ο αλγόριθμος εξετάζει το μεσαίο στοιχείο του πίνακα και διακρίνει περιπτώσεις:
  - Αν το μεσαίο στοιχείο είναι το x, επιστρέφει την θέση του.
  - Αν το x είναι μικρότερο από το μεσαίο στοιχείο τότε αναδρομικά ψάχνει στο κομμάτι του πίνακα από την αρχή μέχρι το μεσαίο στοιχείο
  - Αν το x είναι μεγαλύτερο από το μεσαίο στοιχείο τότε αναδρομικά ψάχνει στο κομμάτι του πίνακα από το μεσαίο στοιχείο μέχρι το τέλος

Β. Θεωρία

1. Αναδρομικοί Αλγόριθμοι

1. Ο αλγόριθμος αναζήτησης BinarySearch (2. Ψευδοκώδικας)

- Παρακάτω φαίνεται μία υλοποίηση της BinarySearch σε ψευδογλώσσα

```
procedure BinarySearch(A,x,start,finish)

   if start>finish then
      return 0
   else
      middle=(start+finish) div 2
      if (x==A[middle]) then
         return middle
      else if (x<A[middle]) then
         pos=BinarySearch(A,x,start,middle-1)
         return pos
      else if (x>A[middle]) then
         pos=BinarySearch(A,x,middle+1,finish)
         return pos
      end if
   end if

end procedure
```

Β. Θεωρία

1. Αναδρομικοί Αλγόριθμοι

1. Ο αλγόριθμος αναζήτησης BinarySearch (3. Παράδειγμα Εκτέλεσης)

- Εκτελούμε τον αλγόριθμο ψάχνοντας το στοιχείο 11 στον πίνακα:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
2	3	5	7	11	13	17	21	23	27	31	33	37	41	43

- Κλήση: BinarySearch(A,11,1,15): middle=(1+15) div 2=8. x<A[middle]

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
2	3	5	7	11	13	17	21	23	27	31	33	37	41	43
start														finish

  - Αναδρομική Κλήση: BinarySearch(A,11,1,7) : middle=(1+7) div 2=4 x>A[middle]

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
2	3	5	7	11	13	17	21	23	27	31	33	37	41	43
start						finish								

  - Αναδρομική Κλήση: BinarySearch(A,11,5,7) : middle=(5+7) div 2=6 x<A[middle]

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
2	3	5	7	11	13	17	21	23	27	31	33	37	41	43
				start		finish								

  - Αναδρομική Κλήση: BinarySearch(A,11,5,5) : middle=(5+5) div 2=5 x=A[middle]

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
2	3	5	7	11	13	17	21	23	27	31	33	37	41	43
				start=finish										

Β. Θεωρία

1. Αναδρομικοί Αλγόριθμοι

1. Ο αλγόριθμος αναζήτησης BinarySearch (4. Ανάλυση)

- Καλύτερη περίπτωση: είναι όταν το στοιχείο x βρίσκεται ακριβώς στην μεσαία θέση του πίνακα.
  - Η πολυπλοκότητα είναι T(n)=5 άρα ασυμπτωτικά T(n)=Θ(1).
- Χειρότερη περίπτωση: είναι όταν το στοιχείο x δεν υπάρχει στον πίνακα:
  - Έστω T(n) η πολυπλοκότητα όταν ο πίνακας έχει διάσταση n.
    - Αρχικά θα γίνουν 8 πράξεις μέχρι να γίνει η αναδρομική κλήση (έστω ότι πάντα γίνεται και η 2η αναδρομική κλήση για να έχουμε μία παραπάνω σύγκριση)
    - Έπειτα γίνεται αναδρομική κλήση για πίνακα διάστασης $\left\lceil \frac{n}{2} \right\rceil - 1$
      - Άρα αφού για διάσταση n, έχουμε χρόνο T(n), για διάσταση $\left\lceil \frac{n}{2} \right\rceil - 1$ θέλουμε χρόνο $T\left(\left\lceil \frac{n}{2} \right\rceil - 1\right)$
    - Έπειτα γίνεται ακόμη 1 πράξη.
  - Άρα η πολυπλοκότητα δίνεται από την αναδρομική σχέση: $T(n) = T\left(\left\lceil \frac{n}{2} \right\rceil - 1\right) + 9$
  - Ειδικά όταν n=0 τότε γίνεται 1 πράξη (κριτήριο τερματισμού)

Β. Θεωρία

1. Αναδρομικοί Αλγόριθμοι

1. Ο αλγόριθμος αναζήτησης BinarySearch (4. Ανάλυση)

- Τελικά η χειρότερη περίπτωση λύνεται από την αναδρομική σχέση:

$$T(n)=\begin{cases} 1, & n=0 \\ T\left(\left\lceil \frac{n}{2} \right\rceil -1\right)+9, & n>0 \end{cases}$$

- Επειδή ωστόσο αυτή η σχέση είναι ιδιαίτερα περίπλοκη για να την λύσουμε, την προσεγγίζουμε ως εξής:
  - Το $\left\lceil \frac{n}{2} \right\rceil -1$ είναι περίπου $\frac{n}{2}$

- Άρα προκύπτει η τελική αναδρομική σχέση:

$$T(n)=\begin{cases} 1, & n=0 \\ T\left(\frac{n}{2}\right)+9, & n>0 \end{cases}$$

  - Την οποία λύνουμε με το θεώρημα κυριαρχίας και προκύπτει ότι η πολυπλοκότητά της είναι: $T(n)=\Theta(\log n)$

Β. Θεωρία

1. Αναδρομικοί Αλγόριθμοι

2. Ο αλγόριθμος ταξινόμησης MergeSort (1.Διατύπωση και Λειτουργία)

- MergeSort ή Ταξινόμηση με Συγχώνευση:
  - Είσοδος: πίνακας (αριθμών) Α με n στοιχεία
  - Έξοδος: ταξινόμηση των στοιχείων του πίνακα σε αύξουσα σειρά

- Λειτουργία του αλγορίθμου: Ο αλγόριθμος:
  - Ταξινομεί το αριστερό κομμάτι του πίνακα
  - Ταξινομεί το δεξί κομμάτι του πίνακα
  - Συγχωνεύει τα δύο ταξινομημένα πλέον κομμάτια σε μία ταξινομημένη ακολουθία
- Η ταξινόμηση κάθε κομματιού γίνεται με αναδρομική κλήση της ίδιας διαδικασίας.

Β. Θεωρία

1. Αναδρομικοί Αλγόριθμοι

2. Ο αλγόριθμος ταξινόμησης MergeSort (2. Ψευδοκώδικας)

- Παρακάτω φαίνεται μία υλοποίηση της MergeSort σε ψευδογλώσσα

```
procedure MergeSort(A,start, finish)

   if |A|<=2 then
      Ταξινόμησε τον Α
   else
      middle=(start+finish) div 2
      A1=MergeSort(A,start,middle)
      A2=MergeSort(A,middle+1,finish)
      A=Merge(A1,A2)
   end if

end procedure
```

- Το κριτήριο τερματισμού της αναδρομής είναι όταν ο πίνακας έχει το πολύ 2 στοιχεία.
- Γίνονται 2 αναδρομικές κλήσεις για την ταξινόμηση του αριστερού και του δεξιού κομματιού αντίστοιχα.
- Έπειτα γίνεται συγχώνευση των δύο ακολουθιών με την διαδικασία Merge

Β. Θεωρία

1. Αναδρομικοί Αλγόριθμοι

2. Ο αλγόριθμος ταξινόμησης MergeSort (2. Ψευδοκώδικας)

- Η διαδικασία Merge για την συγχώνευση δύο ήδη ταξινομημένων πινάκων μπορεί να υλοποιηθεί ως εξής:

```
procedure Merge(A,B)

   i=1, j=1, k=1
   while (i<=n AND j<=m)
      if (a_i<b_j) then
         c_k=a_i ; i=i+1
      else
         c_k=b_j ; j=j+1
      end if
      k=k+1
   end while
   Όσα στοιχεία του A ή του B περισσεψαν τα βάζουμε στο τέλος του C
   return C
end procedure
```

- Παραπάνω θεωρούμε το |A|=n, |B|=m

Β. Θεωρία

1. Αναδρομικοί Αλγόριθμοι

2. Ο αλγόριθμος ταξινόμησης MergeSort (3. Παράδειγμα Εκτέλεσης)

- Αναδρομική Κλήση MergeSort(A,1,16)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
18	7	4	11	9	20	6	1	22	19	14	5	2	9	10	13

Β. Θεωρία

1. Αναδρομικοί Αλγόριθμοι

2. Ο αλγόριθμος ταξινόμησης MergeSort (3. Παράδειγμα Εκτέλεσης)

- Αναδρομική Κλήση MergeSort(A,1,16)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
18	7	4	11	9	20	6	1	22	19	14	5	2	3	10	13

1	2	3	4	5	6	7	8
18	7	4	11	9	20	6	1

9	10	11	12	13	14	15	16
22	19	14	5	2	3	10	13

Β. Θεωρία

1. Αναδρομικοί Αλγόριθμοι

2. Ο αλγόριθμος ταξινόμησης MergeSort (3. Παράδειγμα Εκτέλεσης)

- Αναδρομική Κλήση (Α,1,8)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
18	7	4	11	9	20	6	1	22	19	14	5	2	3	10	13

1	2	3	4	5	6	7	8
18	7	4	11	9	20	6	1

9	10	11	12	13	14	15	16
22	19	14	5	2	3	10	13

1	2	3	4
18	7	4	11

5	6	7	8
9	20	6	1

Β. Θεωρία

1. Αναδρομικοί Αλγόριθμοι

2. Ο αλγόριθμος ταξινόμησης MergeSort (3. Παράδειγμα Εκτέλεσης)

- Αναδρομική Κλήση (A,1,4)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
18	7	4	11	9	20	6	1	22	19	14	5	2	3	10	13

1	2	3	4	5	6	7	8
18	7	4	11	9	20	6	1

9	10	11	12	13	14	15	16
22	19	14	5	2	3	10	13

1	2	3	4
18	7	4	11

5	6	7	8
9	20	6	1

1	2
18	7

3	4
4	11

Β. Θεωρία

1. Αναδρομικοί Αλγόριθμοι

2. Ο αλγόριθμος ταξινόμησης MergeSort (3. Παράδειγμα Εκτέλεσης)

- Αναδρομική Κλήση (Α,1,2): Ταξινομηση του υποπίνακα

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
18	7	4	11	9	20	6	1	22	19	14	5	2	3	10	13

1	2	3	4	5	6	7	8
18	7	4	11	9	20	6	1

9	10	11	12	13	14	15	16
22	19	14	5	2	3	10	13

1	2	3	4
18	7	4	11

5	6	7	8
9	20	6	1

1	2
7	18

3	4
4	11

Β. Θεωρία

1. Αναδρομικοί Αλγόριθμοι

2. Ο αλγόριθμος ταξινόμησης MergeSort (3. Παράδειγμα Εκτέλεσης)

- Αναδρομική Κλήση (A,3,4): Ταξινομηση του υποπίνακα

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
18	7	4	11	9	20	6	1	22	19	14	5	2	3	10	13

1	2	3	4	5	6	7	8
18	7	4	11	9	20	6	1

9	10	11	12	13	14	15	16
22	19	14	5	2	3	10	13

1	2	3	4
18	7	4	11

5	6	7	8
9	20	6	1

1	2
7	18

3	4
4	11

Β. Θεωρία

1. Αναδρομικοί Αλγόριθμοι

2. Ο αλγόριθμος ταξινόμησης MergeSort (3. Παράδειγμα Εκτέλεσης)

- Αναδρομική Κλήση (A,1,4): Συγχώνευση των δύο υποπινάκων

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
18	7	4	11	9	20	6	1	22	19	14	5	2	3	10	13

1	2	3	4	5	6	7	8
18	7	4	11	9	20	6	1

9	10	11	12	13	14	15	16
22	19	14	5	2	3	10	13

1	2	3	4
4	7	11	18

5	6	7	8
9	20	6	1

1	2
7	18

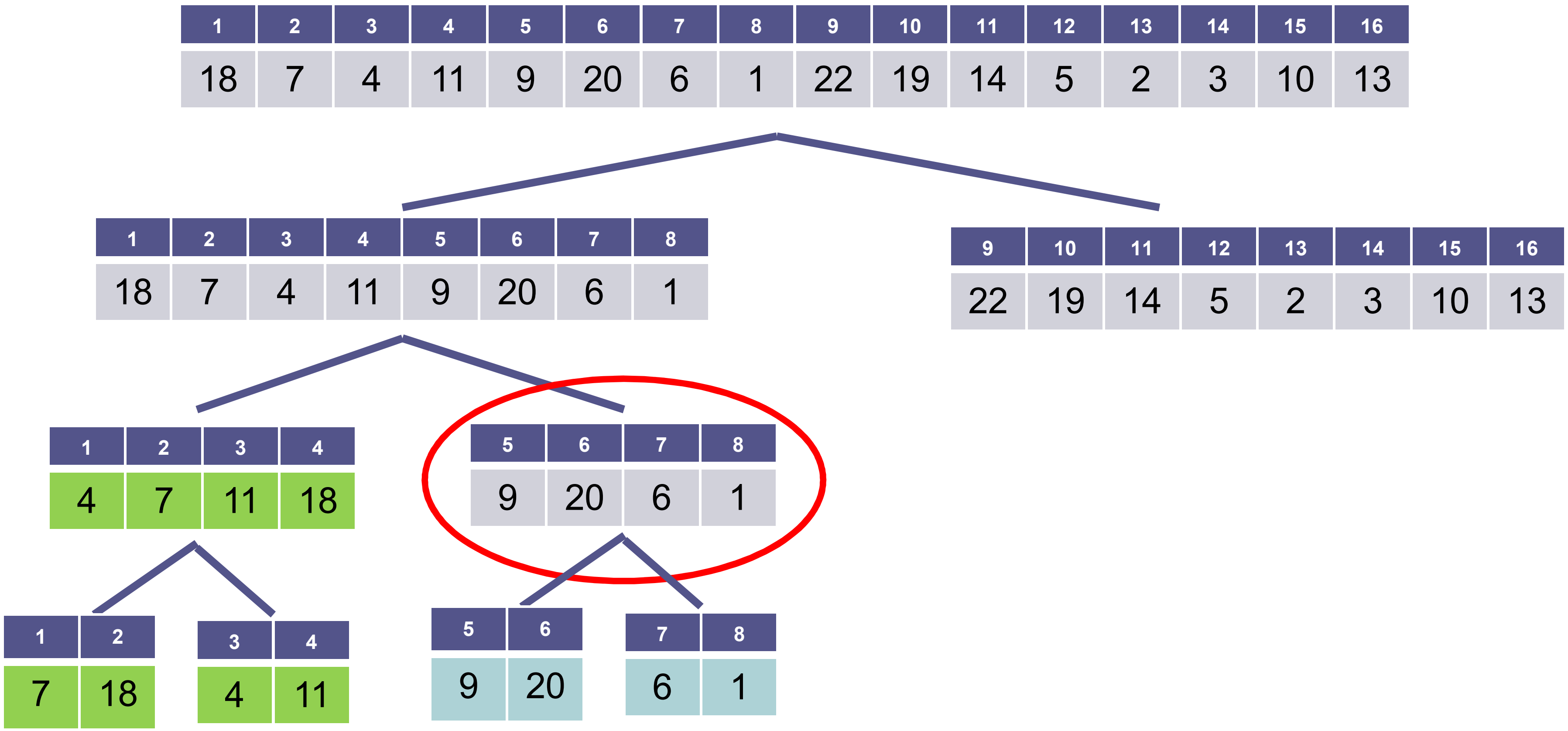
3	4
4	11

Β. Θεωρία

1. Αναδρομικοί Αλγόριθμοι

2. Ο αλγόριθμος ταξινόμησης MergeSort (3. Παράδειγμα Εκτέλεσης)

- Αναδρομική Κλήση (A,5,8)



Β. Θεωρία

1. Αναδρομικοί Αλγόριθμοι

2. Ο αλγόριθμος ταξινόμησης MergeSort (3. Παράδειγμα Εκτέλεσης)

- Αναδρομική Κλήση (A,5,6): Ταξινομηση του υποπίνακα

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
18	7	4	11	9	20	6	1	22	19	14	5	2	3	10	13

1	2	3	4	5	6	7	8
18	7	4	11	9	20	6	1

9	10	11	12	13	14	15	16
22	19	14	5	2	3	10	13

1	2	3	4
4	7	11	18

5	6	7	8
9	20	6	1

1	2
7	18

3	4
4	11

5	6
9	20

7	8
6	1

Β. Θεωρία

1. Αναδρομικοί Αλγόριθμοι

2. Ο αλγόριθμος ταξινόμησης MergeSort (3. Παράδειγμα Εκτέλεσης)

- Αναδρομική Κλήση (A,7,8): Ταξινομηση του υποπίνακα

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
18	7	4	11	9	20	6	1	22	19	14	5	2	3	10	13

1	2	3	4	5	6	7	8
18	7	4	11	9	20	6	1

9	10	11	12	13	14	15	16
22	19	14	5	2	3	10	13

1	2	3	4
4	7	11	18

5	6	7	8
9	20	6	1

1	2
7	18

3	4
4	11

5	6
9	20

7	8
1	6

Β. Θεωρία

1. Αναδρομικοί Αλγόριθμοι

2. Ο αλγόριθμος ταξινόμησης MergeSort (3. Παράδειγμα Εκτέλεσης)

- Αναδρομική Κλήση (A,5,8): Συγχώνευση των δύο υποπινάκων

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
18	7	4	11	9	20	6	1	22	19	14	5	2	3	10	13

1	2	3	4	5	6	7	8
18	7	4	11	9	20	6	1

9	10	11	12	13	14	15	16
22	19	14	5	2	3	10	13

1	2	3	4
4	7	11	18

5	6	7	8
1	6	9	20

1	2
7	18

3	4
4	11

5	6
9	20

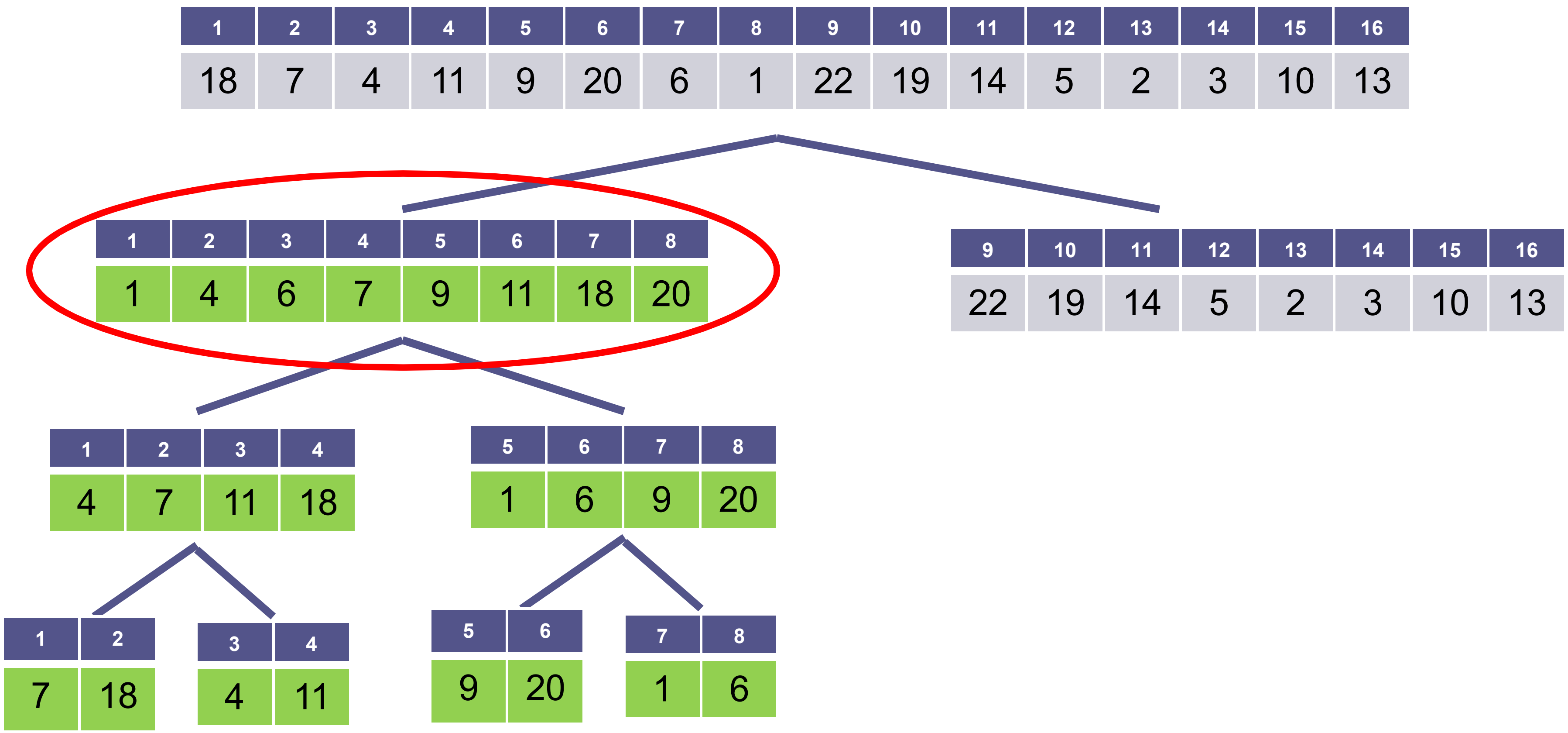
7	8
1	6

Β. Θεωρία

1. Αναδρομικοί Αλγόριθμοι

2. Ο αλγόριθμος ταξινόμησης MergeSort (3. Παράδειγμα Εκτέλεσης)

- Αναδρομική Κλήση (A,1,8): Συγχώνευση των δύο υποπινάκων

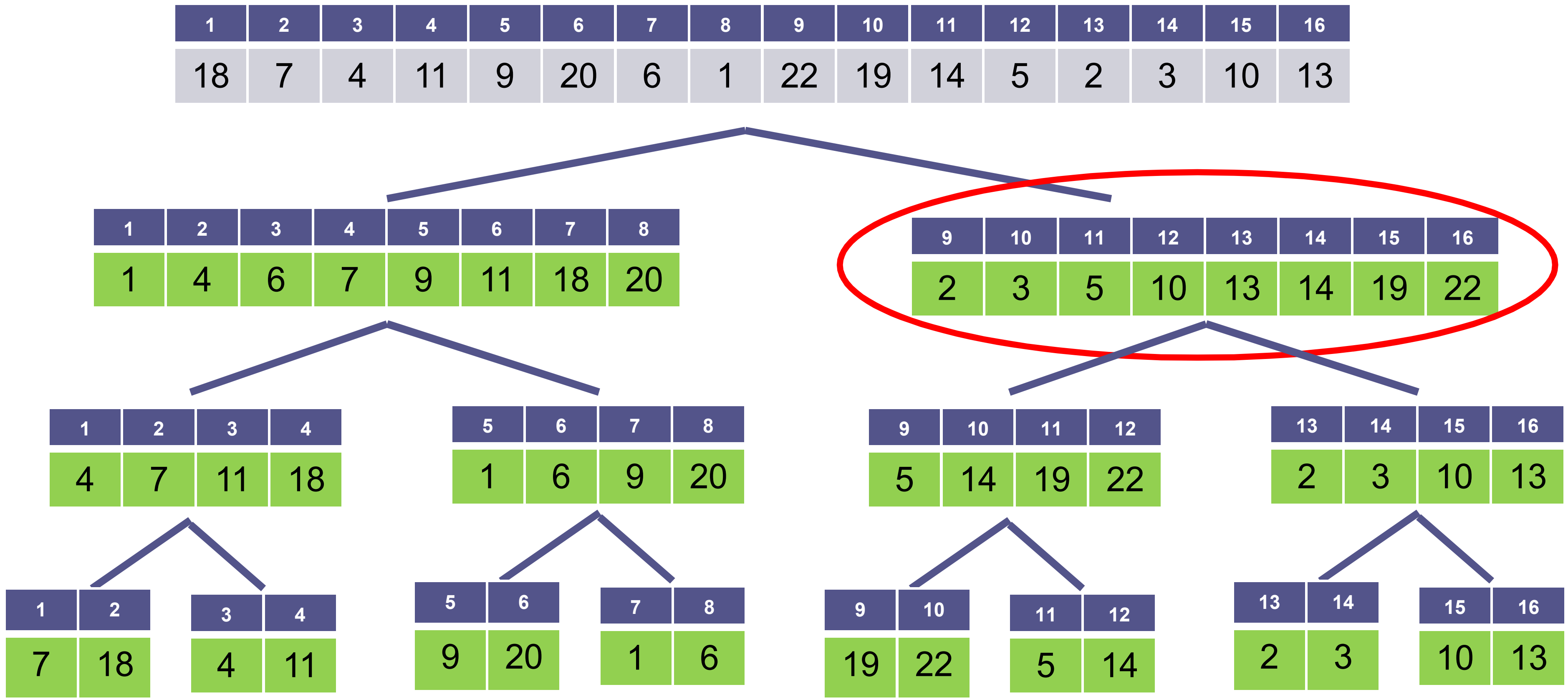


Β. Θεωρία

1. Αναδρομικοί Αλγόριθμοι

2. Ο αλγόριθμος ταξινόμησης MergeSort (3. Παράδειγμα Εκτέλεσης)

- Αντίστοιχα θα γίνουν όλες οι αναδρομικές κλήσεις στο (9,16)

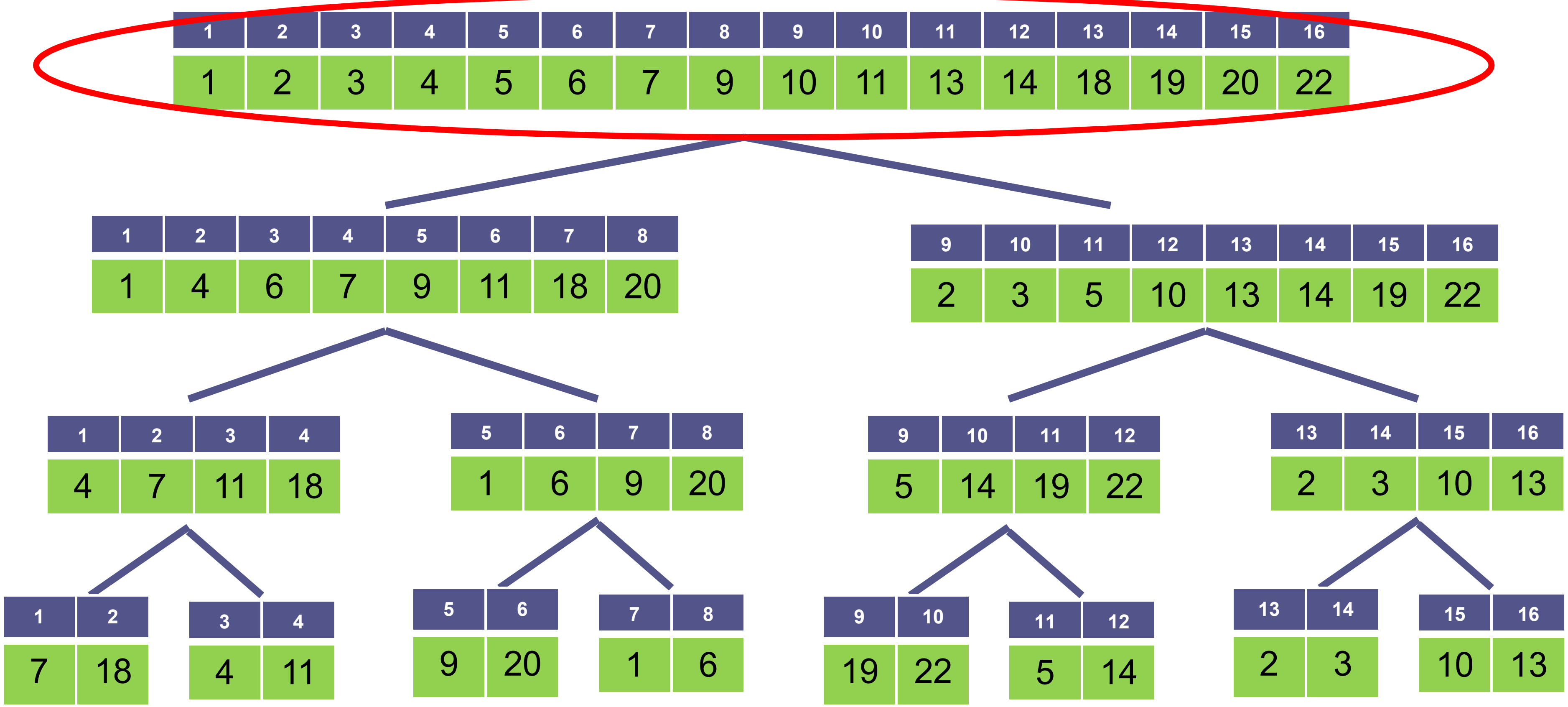


Β. Θεωρία

1. Αναδρομικοί Αλγόριθμοι

2. Ο αλγόριθμος ταξινόμησης MergeSort (3. Παράδειγμα Εκτέλεσης)

- Αναδρομική Κλήση (Α,1,16): Συγχώνευση των δύο υποπινάκων



Β. Θεωρία

1. Αναδρομικοί Αλγόριθμοι

2. Ο αλγόριθμος ταξινόμησης MergeSort (4. Ανάλυση)

- Η πολυπλοκότητα της συνάρτησης Merge είναι:

$$T(n) = \Theta(n+m)$$

- Άρα η πολυπλοκότητα της MergeSort είναι:

$$T(n) = \begin{cases} \Theta(1), & n=1 \quad ή \quad n=2 \\ T\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + T\left(\left\lceil \frac{n}{2} \right\rceil\right) + \Theta(n), & n>2 \end{cases}$$

  - Η οποία είναι ιδαίτερα περίπλοκη γι΄αυτό θεωρούμε ότι $\left\lfloor \frac{n}{2} \right\rfloor = \left\lceil \frac{n}{2} \right\rceil \approx \frac{n}{2}$
  - Άρα απλοποιείται ως:

$$T(n) = \begin{cases} \Theta(1), & n=1 \quad ή \quad n=2 \\ 2T\left(\frac{n}{2}\right) + \Theta(n), & n>2 \end{cases}$$

    - Η οποία λύνεται από το Θ.Κυριαρχίας και προκύπτει: T(n)=Θ(nlogn)

Γ. Μεθοδολογία Ασκήσεων

1. Η αναδρομή T(n)=aT(n/b)+f(n)

- Για την επίλυση της αναδρομικής σχέσης T(n)=aT(n/b)+f(n) υπάρχουν δύο τρόποι:
  - Το θεώρημα κυριαρχίας, το οποίο με εύκολο τρόπο μας δίνει μια ασυμπτωτική εκτίμηση της πολυπλοκότητας
  - Η μέθοδος επανάληψης, μας δίνει την ακριβή συνάρτηση πολυπλοκότητας (άρα μπορούμε να εξάγουμε και ασυμπτωτική εκτίμηση).
- Συνεπώς:
  - Αν μας ζητείται απλά η λύση της αναδρομής, προτιμάμε το θεώρημα κυριαρχίας.
  - Αν μας ζητείται ακριβής συνάρτηση πολυπλοκότητας απαιτείται η μέθοδος επανάληψης
  - Αν μας ζητείται ασυμπτωτική εκτίμηση της συνάρτησης πολυπλοκότητας προτιμάμε το θεώρημα κυριαρχίας
  - Αν το θεώρημα κυριάρχίας αποτύχει (μπορεί να συμβεί στην $2^{η}$ συνθήκη της $3^{ης}$ περίπτωσης του Θ.Κ.) τότε αναγκαστικά χρησιμοποιούμε την μέθοδο επανάληψης.

Γ. Μεθοδολογία Ασκήσεων

1. Η αναδρομή T(n)=aT(n/b)+f(n)

1. Επίλυση με το θεώρημα κυριαρχίας

- Το θεώρημα Κυριαρχίας (Master Theorem) είναι το εξής:

Θεώρημα Κυριαρχίας: Έστω η αναδρομική εξίσωση

$$\boxed{T(n) = aT\left(\frac{n}{b}\right) + f(n)}$$

όπου a≥1, b>1 είναι σταθερές, και f(n) είναι μια ασυμπτωτικά θετική συνάρτηση. Τότε διακρίνονται οι ακόλουθες τρεις περιπτώσεις:

A) Αν $f(n) = O(n^{\log_b a-\varepsilon})$ για κάποια σταθερά ε>0, τότε:

$$\boxed{T(n) = \Theta\left(n^{\log_b a}\right)}$$

B) Αν $f(n) = \Theta(n^{\log_b a})$ τότε:

$$\boxed{T(n) = \Theta\left(n^{\log_b a} \cdot \log n\right)}$$

Γ) Αν $f(n) = \Omega(n^{\log_b a+\varepsilon})$ για κάποια σταθερά ε>0

και $a \cdot f\left(\frac{n}{b}\right) \le c \cdot f(n)$ για κάποια σταθερά c<1, τότε:

$$\boxed{T(n) = \Theta\left(f(n)\right)}$$

Γ. Μεθοδολογία Ασκήσεων

1. Η αναδρομή T(n)=aT(n/b)+f(n)

1. Επίλυση με το θεώρημα κυριαρχίας

- Για την επίλυση με το θεώρημα της κυριαρχίας εργαζόμαστε ως εξής:
  - Εντοπίζουμε από την εκφώνηση τα a,b και f(n)
  - Υπολογίζουμε το $\log_b a$.
  - Συγκρίνουμε ασυμπτωτικά το f(n) με την $n^{\log_b a}$ και:
    - Αν $f(n) < n^{\log_b a}$ είμαστε στην Α' περίπτωση
    - Αν $f(n) = n^{\log_b a}$ είμαστε στην Β' περίπτωση
    - Αν $f(n) > n^{\log_b a}$ είμαστε στην Γ' περίπτωση (ΠΡΟΣΟΧΗ! Ότι πρέπει να ελέγξουμε και την 2η συνθήκη)

Γ. Μεθοδολογία Ασκήσεων

1. Η αναδρομή T(n)=aT(n/b)+f(n)

1. Επίλυση με το θεώρημα κυριαρχίας (Α' περίπτωση)

- Εφόσον είμαστε στην Α' περίπτωση διατυπώνουμε τελικά την απόδειξη σύμφωνα με τον ορισμό:

ΠΑΡΑΔΕΙΓΜΑ:
Να λύσετε την αναδρομή: $T(n) = 8T\left(\frac{n}{2}\right) + n$

<u>Λύση:</u>
Έχω: $a = 8, \quad b = 2, \quad f(n) = n, \quad \log_b a = \log_2 8 = 3$

Ισχύει: $f(n) = n = O(n^{3-\varepsilon})$ για κάποια σταθερά ε>0

Άρα από την Α' περίπτωση του Θεωρήματος Κυριαρχίας έπεται ότι:

$$T(n) = \Theta(n^3)$$

Γ. Μεθοδολογία Ασκήσεων

1. Η αναδρομή T(n)=aT(n/b)+f(n)

1. Επίλυση με το θεώρημα κυριαρχίας (Β’ περίπτωση)

- Εφόσον είμαστε στην Β’ περίπτωση διατυπώνουμε τελικά την απόδειξη σύμφωνα με τον ορισμό:

ΠΑΡΑΔΕΙΓΜΑ:
Να λύσετε την αναδρομή: $T(n) = 9T\left(\frac{n}{3}\right) + n^2$

<u>Λύση:</u>
Έχω: $a = 9, \quad b = 3, \quad f(n) = n^2, \quad \log_b a = \log_3 9 = 2$

Ισχύει: $f(n) = n^2 = \Theta(n^2)$

Άρα από την Β’ περίπτωση του Θεωρήματος Κυριαρχίας έπεται ότι:

$$T(n) = \Theta(n^2 \log n)$$

Γ. Μεθοδολογία Ασκήσεων

1. Η αναδρομή T(n)=aT(n/b)+f(n)

1. Επίλυση με το θεώρημα κυριαρχίας (Γ' περίπτωση)

- Στην Γ' περίπτωση πρέπει να ελέγξουμε και την 2η συνθήκη, δηλαδή να αναζητήσουμε c>0 τέτοια ώστε $a \cdot f\left(\frac{n}{b}\right) \leq c \cdot f(n)$. Η εύρεση του εύρους τιμών για το c γίνεται αντικαθιστώντας τα a, b και την τιμή των f(n) και f(n/b).

ΠΑΡΑΔΕΙΓΜΑ:
Να λύσετε την αναδρομή: $T(n) = 4T\left(\frac{n}{2}\right) + n^3$

Λύση:
Έχω: $a = 4, \quad b = 2, \quad f(n) = n^3, \quad \log_b a = \log_2 4 = 2$

Ισχύει: $f(n) = n^3 = \Omega(n^{2+\varepsilon})$ για κάποια σταθερά ε>0

Ελέγχω αν υπάρχει c<1 τέτοιο ώστε:

$$af\left(\frac{n}{b}\right) \leq cf(n) \Leftrightarrow 4f\left(\frac{n}{2}\right) \leq cf(n) \Leftrightarrow 4\left(\frac{n}{2}\right)^3 \leq cn^3 \Leftrightarrow 4\frac{n^3}{2^3} \leq cn^3 \Leftrightarrow \frac{4}{8} \leq c \Leftrightarrow \frac{1}{2} \leq c$$

Άρα ισχύει για ½≤c<1.
Άρα από την Γ' περίπτωση του Θεωρήματος Κυριαρχίας έπεται ότι:

$$T(n) = \Theta(n^3)$$

Γ. Μεθοδολογία Ασκήσεων

1. Η αναδρομή T(n)=aT(n/b)+f(n)

2. Επίλυση με την μέθοδο επανάληψης

- Η μέθοδος επανάληψης είναι μία μέθοδος υπολογισμού της συνάρτησης πολυπλοκότητας μιας αναδρομής της μορφής T(n)=aT(n/b)+f(n), η οποία γίνεται με τα εξής βήματα:

ΒΗΜΑΤΑ ΤΗΣ ΜΕΘΟΔΟΥ ΕΠΑΝΑΛΗΨΗΣ

1. Κάνουμε 3 εφαρμογές της αναδρομικής σχέσης (Μέχρι να φτάσουμε στην μορφή $T(n) = \ldots \cdot T\left(\frac{n}{b^3}\right) + \ldots$)
2. Εκτίμηση της σειράς που προκύπτει μετά από k επαναλήψεις (Μας καθοδηγεί ο όρος $T(n) = \ldots \cdot T\left(\frac{n}{b^k}\right) + \ldots$)
3. Υπολογίζουμε πότε σταματάει η αναδρομή (Θέτω $\frac{n}{b^k} = n_0$ όπου n_0 η συνθήκη τερματισμού της αναδρομής και λύνουμε ως προς k). Π.χ. αν $n_0=1$ τότε $k=\log_b n$
4. Αντικατάσταση του k στον αναδρομικό τύπο του βήματος 2.
5. Υπολογισμός του αθροίσματος που προέκυψε.

Γ. Μεθοδολογία Ασκήσεων

1. Η αναδρομή T(n)=aT(n/b)+f(n)

2. Επίλυση με την μέθοδο επανάληψης (Βήμα 1: 3 εφαρμογές του κανόνα)

- Στο 1ο βήμα εφαρμόζουμε τον αναδρομικό κανόνα 3 φορές και κάνουμε τις πράξεις που προκύπτουν.
- Βοηθητικά στο πρόχειρο, υπολογίζουμε τους αναδρομικούς όρους με αντικατάσταση στην αναδρομή

ΠΑΡΑΔΕΙΓΜΑ:
Να λύσετε την αναδρομή: $T(n)=\begin{cases} 5T\left(\frac{n}{3}\right)+n, & \alpha\nu\ n>1 \\ 1, & \alpha\nu\ n=1 \end{cases}$

Λύση:

$$T(n)=5T\left(\frac{n}{3}\right)+n$$

$$=5\left[5T\left(\frac{n}{3^2}\right)+\frac{n}{3}\right]+n=5^2T\left(\frac{n}{3^2}\right)+5\frac{n}{3}+n$$

$$=5^2\left[5T\left(\frac{n}{3^3}\right)+\frac{n}{3^2}\right]+5\frac{n}{3}+n=5^3T\left(\frac{n}{3^3}\right)+5^2\frac{n}{3^2}+5\frac{n}{3}+n=$$

ΠΡΟΧΕΙΡΟ

$$T(n)=5T\left(\frac{n}{3}\right)+n$$

$$T\left(\frac{n}{3}\right)=5T\left(\frac{n}{3^2}\right)+\frac{n}{3}$$

$$T\left(\frac{n}{3^2}\right)=5T\left(\frac{n}{3^3}\right)+\frac{n}{3^2}$$

Γ. Μεθοδολογία Ασκήσεων

1. Η αναδρομή T(n)=aT(n/b)+f(n)

2. Επίλυση με την μέθοδο επανάληψης (Βήμα 2: Εκτίμηση στο βήμα k)

- Στο 2ο βήμα εκτιμάμε την σειρά που θα προκύψει μετά από k επαναλήψεις (Μας καθοδηγεί ο όρος $T(n) = \ldots \cdot T\left(\frac{n}{b^k}\right) + \ldots.$

(…συνέχεια…)

$$= 5^3 T\left(\frac{n}{3^3}\right) + 5^2 \frac{n}{3^2} + 5\frac{n}{3} + n =$$

$$= \ldots =$$

$$= 5^k T\left(\frac{n}{3^k}\right) + 5^{k-1} \frac{n}{3^{k-1}} + \ldots + 5^2 \frac{n}{3^2} + 5\frac{n}{3} + n$$

Γ. Μεθοδολογία Ασκήσεων

1. Η αναδρομή T(n)=aT(n/b)+f(n)

2. Επίλυση με την μέθοδο επανάληψης (Βήμα 3: Υπολογισμός του k)

- Υπολογίζουμε πότε σταματάει η αναδρομή (Θέτω $\frac{n}{b^k}=n_0$ όπου n_0 η συνθήκη τερματισμού της αναδρομής και λύνουμε ως προς k).

(…συνέχεια…)

Η αναδρομή σταματά όταν

$\frac{n}{3^k}=1 \Rightarrow$

$n=3^k \Rightarrow$

$\log_3 n=\log_3 3^k \Rightarrow$

$\log_3 n=k\log_3 3 \Rightarrow$

$k=\log_3 n$

Γ. Μεθοδολογία Ασκήσεων

1. Η αναδρομή T(n)=aT(n/b)+f(n)

2. Επίλυση με την μέθοδο επανάληψης (Βήμα 4: Αντικατάσταση του k)

- Αντικαθιστούμε το k που βρήκαμε στην παράσταση που προέκυψε στο βήμα 2. Θα πρέπει να απαλειφθεί ο αναδρομικός όρος με την συνθήκη τερματισμού της αναδρομής.

(…συνέχεια…)

Θέτοντας $k=\log_3 n$ στην T(n) έχουμε:

$$T(n)=5^{\log_3 n}T\left(\frac{n}{3^{\log_3 n}}\right)+5^{\log_3 n-1}\frac{n}{3^{\log_3 n-1}}+\ldots+5^2\frac{n}{3^2}+5\frac{n}{3}+n$$

$$=5^{\log_3 n}T(1)+5^{\log_3 n-1}\frac{n}{3^{\log_3 n-1}}+\ldots+5^2\frac{n}{3^2}+5\frac{n}{3}+n$$

$$=5^{\log_3 n}+5^{\log_3 n-1}\frac{n}{3^{\log_3 n-1}}+\ldots+5^2\frac{n}{3^2}+5\frac{n}{3}+n$$

Γ. Μεθοδολογία Ασκήσεων

1. Η αναδρομή T(n)=aT(n/b)+f(n)

2. Επίλυση με την μέθοδο επανάληψης (Βήμα 5: Υπολογισμός του αθροίσματος)

- Υπολογίζουμε το άθροισμα που προκύπτει. Στην μέθοδο επανάληψης προκύπτει πάντα γεωμετρική πρόοδος στις σταθερές που εμφανίζονται και γι' αυτό είναι χρήσιμη η σχέση: $\boxed{\sum_{i=0}^{n} x^i = \frac{x^{n+1}-1}{x-1}}$

(…συνέχεια…)

$$T(n) = 5^{\log_3 n} + 5^{\log_3 n-1}\frac{n}{3^{\log_3 n-1}} + \ldots + 5^2\frac{n}{3^2} + 5\frac{n}{3} + n =$$

$$= 5^{\log_3 n} + \left[n + 5\frac{n}{3} + 5^2\frac{n}{3^2} + \ldots + 5^{\log_3 n-1}\frac{n}{3^{\log_3 n-1}} \right] =$$

$$= 5^{\log_3 n} + \sum_{i=0}^{\log_3 n-1} 5^i \frac{n}{3^i} = 5^{\log_3 n} + n\sum_{i=0}^{\log_3 n-1} \frac{5^i}{3^i} =$$

$$= 5^{\log_3 n} + n\sum_{i=0}^{\log_3 n-1} \left(\frac{5}{3}\right)^i = 5^{\log_3 n} + n\sum_{i=0}^{\log_3 n-1} 1{,}66^i =$$

$$= 5^{\log_3 n} + n\frac{1{,}66^{\log_3 n-1+1} - 1}{1{,}66-1} = 5^{\log_3 n} + 1{,}5\cdot n\cdot 1{,}66^{\log_3 n} - 1{,}5n$$

Άρα $\boxed{T(n) = 5^{\log_3 n} + 1{,}5\cdot n\cdot 1{,}66^{\log_3 n} - 1{,}5n}$

Δ. Ασκήσεις
Εφαρμογή 1

- Υπολογίστε μία ασυμπτωτική εκτίμηση της πολυπλοκότητας των αναδρομών:

$$A) \qquad T(n) = 8T\left(\frac{n}{2}\right) + n^2$$

$$B) \qquad T(n) = 8T\left(\frac{n}{2}\right) + n^3$$

$$C) \qquad T(n) = 8T\left(\frac{n}{2}\right) + n^4$$

Δ. Ασκήσεις
Εφαρμογή 2

- Λύστε τις αναδρομές:

$A)\qquad T(n) = 5T\left(\frac{n}{2}\right) + n^2$

$B)\qquad T(n) = 5T\left(\frac{n}{2}\right) + n^3$

Δ. Ασκήσεις
Εφαρμογή 3

- Υπολογίστε την ακριβή πολυπλοκότητα των αναδρομών:

$$A)\quad T(n)=\begin{cases}6T\left(\dfrac{n}{2}\right)+n, & \alpha\nu\ n>1\\ 1, & \alpha\nu\ n=1\end{cases}$$

$$B)\quad T(n)=\begin{cases}4T\left(\dfrac{n}{3}\right)+n^2, & \alpha\nu\ n>1\\ 1, & \alpha\nu\ n=1\end{cases}$$

Παράρτημα

1. Η μέθοδος της αντικατάστασης

3. Εντοπισμός Άνω Φράγματος

- Στην μέθοδο της αντικατάστασης:
  - «Μαντεύουμε» τη λύση της αναδρομής.
  - Επαληθεύουμε ότι η λύση που μαντέψαμε είναι ορθή (με μαθηματική επαγωγή) αντικαθιστώντας την στον ορισμό του ασυμπτωτικού συμβολισμού.

ΒΗΜΑΤΑ ΤΗΣ ΜΕΘΟΔΟΥ ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ (π.χ. για $T(n) = aT\left(\frac{n}{b}\right) + f(n)$)

1. Μαντεύουμε (ή μας δίνεται) η λύση της αναδρομικής σχέσης. [πχ $T(n) = \Theta(\mathbf{g}(\mathbf{n}))$]
2. Άνω Φράγμα: Επαληθεύουμε ότι $T(n) = O(\mathbf{g}(\mathbf{n}))$ βρίσκοντας κατάλληλα c_1, n_1 έτσι ώστε η σχέση $T(n) \leq c_1\mathbf{g}(\mathbf{n})$ να ισχύει επαγωγικά.
3. Κάτω Φράγμα: Επαληθεύουμε ότι $T(n) = \Omega(\mathbf{g}(\mathbf{n}))$ βρίσκοντας κατάλληλα c_2, n_2 έτσι ώστε η σχέση $c_2\mathbf{g}(\mathbf{n}) \leq T(n)$ να ισχύει επαγωγικά.
4. Συνεπώς ισχύει $T(n) = \Theta(\mathbf{g}(\mathbf{n}))$ με την επιλογή των c_1, c_2 και θέτοντας $n_0 = \max\{n_1, n_2\}$

Παρατήρηση:
Η μαντεψιά της λύσης της αναδρομής:
- Είτε εντοπίζεται λόγω μεγάλης εμπειρίας στη λύση αναδρομικών σχέσεων.
- Είτε, συνηθέστερα, μας δίνεται στην εκφώνηση.

Παράρτημα

1. Η μέθοδος της αντικατάστασης

2. Παράδειγμα

ΠΑΡΑΔΕΙΓΜΑ 1:

Να επαληθεύσετε ότι η λύση της αναδρομής: $\mathrm{T(n)} = \begin{cases} 2T\left(\frac{n}{2}\right) + n, & n > 1 \\ 1. & 0 \leq n \leq 1 \end{cases}$ είναι $\mathrm{T(n)} = \Theta(\mathrm{nlogn})$

<u>Λύση:</u>

Κάτω Φράγμα: Αναζητούμε $c_2, n_2 > 0$ έτσι ώστε:

$$\begin{aligned} \mathrm{T(n)} &= 2T\left(\frac{n}{2}\right) + n \\ &\geq 2\mathrm{c}_2\frac{n}{2}log\frac{n}{2} + n \\ &= \mathrm{c}_2 n(\log n - \log 2) + n \\ &= \mathrm{c}_2 nlogn - c_2 n + n \\ &\geq \mathrm{c}_2 nlogn + (1 - c_2)n \end{aligned}$$

Συνεπώς απαιτείται $1 - c_2 \geq 0$, έτσι ώστε: $\mathrm{T(n)} \geq c_2 nlogn$, άρα πρέπει $c_2 \leq 1$, ώστε:

$\mathrm{T}(n) \geq \mathrm{c}_2 nlogn$

Για την βάση της επαγωγής:

- $\mathrm{n} = 1$: $\mathrm{T}(1) = 1 \geq c_2 \cdot 1 \cdot \log 1 = 0$. Ισχύει για $\kappa\alpha\theta\varepsilon\ c_2$
- $\mathrm{n} = 2$: $\mathrm{T}(2) = 4 \geq c_2 \cdot 2 \cdot \log 2 = c_2 \cdot 2$. Ισχύει για $c_2 \leq 2$
- $\mathrm{n} = 3$: $\mathrm{T}(3) = 10 \geq c_2 \cdot 3 \cdot \log 3 = c_2 \cdot 4{,}75$. Ισχύει για $c_2 \leq 2{,}1$.

Συνεπώς ισχύει για $n_2 \geq 1, \mathrm{c}_2 \leq 1$.

Παράρτημα

1. Η μέθοδος της αντικατάστασης

2. Παράδειγμα

ΠΑΡΑΔΕΙΓΜΑ 1:

Να επαληθεύσετε ότι η λύση της αναδρομής: $\mathrm{T(n)} = \begin{cases} 2T\left(\frac{n}{2}\right) + n, & n > 1 \\ 1. & 0 \le n \le 1 \end{cases}$ είναι $\mathrm{T(n)} = \Theta(\mathrm{nlogn})$

Λύση(…συνέχεια…):

Άνω Φράγμα: Αναζητούμε $c_1, n_1 > 0$ έτσι ώστε:

$$\begin{aligned} \mathrm{T(n)} &= 2T\left(\frac{n}{2}\right) + n \\ &\le 2\mathrm{c}_1 \frac{n}{2} log \frac{n}{2} + n \\ &= \mathrm{c}_1 n(\log n - \log 2) + n \\ &= \mathrm{c}_1 nlogn - c_1 n + n \\ &\le \mathrm{c}_1 nlogn + (1 - c_1)n \end{aligned}$$

Συνεπώς απαιτείται $1 - c_1 \le 0$, έτσι ώστε: $\mathrm{T(n)} \le c_1 nlogn$, άρα πρέπει $c_1 \ge 1$, ώστε:
$\mathrm{T}(n) \le \mathrm{c}_1 nlogn$

Για την βάση της επαγωγής:

- $\mathrm{n} = 1$: $\mathrm{T}(1) = 1 \le c_1 \cdot 1 \cdot \log 1 = 0$. Δεν ισχύει.
- $\mathrm{n} = 2$: $\mathrm{T}(2) = 4 \le c_1 \cdot 2 \cdot \log 2 = c_1 \cdot 2$. Ισχύει για $c_1 \ge 2$
- $n = 3$: $\mathrm{T}(3) = 10 \le c_1 \cdot 3 \cdot \log 3 = c_1 \cdot 4{,}75$. Ισχύει για $c_1 \ge 2{,}1$.

Συνεπώς ισχύει για $n_1 \ge 2, c_1 \ge 2{,}1$

Άρα $\mathrm{T(n)} = \Theta(\mathrm{nlogn})$ με $n_0 \ge 2,\ \ \mathrm{c}_1 \ge 2{,}1,\ \ \mathrm{c}_2 \le 1$.

Παράρτημα

1. Η μέθοδος της αντικατάστασης

3. Εντοπισμός Άνω Φράγματος

- Πιο συχνή είναι η εφαρμογή της μεθόδου αντικατάστασης για τον εντοπισμό άνω φράγματος.
- Θα δούμε μερικά παραδείγματα:

ΠΑΡΑΔΕΙΓΜΑ 2:

Να επαληθεύσετε ότι για την αναδρομική σχέση: $\mathrm{T(n)} = \begin{cases} 2T\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + n, & n > 1 \\ 1. & n = 1 \end{cases}$ ισχύει $\mathrm{T(n)} = \mathrm{O(nlogn)}$

Λύση:

$$\begin{aligned} \mathrm{T(n)} &= 2T\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + n \\ &\le 2c\left\lfloor \frac{n}{2} \right\rfloor log\left\lfloor \frac{n}{2} \right\rfloor + n \\ &\le 2c\frac{n}{2} log\frac{n}{2} + n \\ &= cnlog\frac{n}{2} + n \\ &= cnlogn - cnlog2 + n \\ &= cnlogn + (1-c)n \end{aligned}$$

Άρα πρέπει $\mathrm{c} \ge 1$

Για την βάση της επαγωγής:

- $\mathrm{n} = 1$: $\mathrm{T}(1) = 1 \le c \cdot 1 \cdot \log 1 = 0$. Δεν ισχύει.
- $\mathrm{n} = 2$: $\mathrm{T}(2) = 4 \le c \cdot 2 \cdot \log 2 = c_2 \cdot 2$. Ισχύει για $c \ge 2$
- $\mathrm{n} = 3$: $\mathrm{T}(3) = 5 \le c \cdot 3 \cdot \log 3 = c_2 \cdot 4{,}75$. Ισχύει για $c \ge 1{,}05$.

Συνεπώς ισχύει για $n_0 \ge 2, c \ge 2$.

Παράρτημα

1. Η μέθοδος της αντικατάστασης

3. Εντοπισμός Άνω Φράγματος

ΠΑΡΑΔΕΙΓΜΑ 3:

Να επαληθεύσετε ότι για την αναδρ. σχέση: $\mathrm{T(n)} = T\left(\left\lfloor\frac{n}{2}\right\rfloor\right) + T\left(\left\lceil\frac{n}{2}\right\rceil\right) + 1$ με $\mathrm{T(1)} = 1$, ισχύει $\mathrm{T(n)} = \mathrm{O(n)}$

Προσπάθεια Λύσης:

Αναζητούμε $c, n_0 > 0$ έτσι ώστε:

$$
\begin{aligned}
\mathrm{T(n)} &= T\left(\left\lfloor\frac{n}{2}\right\rfloor\right) + T\left(\left\lceil\frac{n}{2}\right\rceil\right) + 1 \\
&\le c\left\lfloor\frac{n}{2}\right\rfloor + c\left\lceil\frac{n}{2}\right\rceil + 1 \\
&= cn + 1
\end{aligned}
$$

Αποτυχία επίλυσης, διότι έπρεπε $\mathrm{T(n)} \le \mathrm{cn}$

Λύση:

Μαντεύουμε ότι $T(n) = cn - b$

$$
\begin{aligned}
\mathrm{T(n)} &= T\left(\left\lfloor\frac{n}{2}\right\rfloor\right) + T\left(\left\lceil\frac{n}{2}\right\rceil\right) + 1 \\
&\le c\left\lfloor\frac{n}{2}\right\rfloor - b + c\left\lceil\frac{n}{2}\right\rceil - b + 1 \\
&= cn - 2b + 1
\end{aligned}
$$

Αρκεί λοιπόν $-2b + 1 < 0$ άρα $b > \frac{1}{2}$.

Π.χ. για b=1/2 ισχύει για κάθε $c \ge 0$.

Για την βάση της επαγωγής:

- $\mathrm{n} = 1$: $\mathrm{T(1)} = 1 \le c \cdot 1$. Ισχύει για $\mathrm{c} \ge 1$.
- $\mathrm{n} = 2$: $\mathrm{T(2)} = 3 \le c \cdot 2$. Ισχύει για $c \ge 1{,}5$
- $n = 3$: $\mathrm{T(3)} = 5 \le c \cdot 3$. Ισχύει για $c \ge 1{,}67$.

Συνεπώς ισχύει για $n \ge 1, c \ge 1{,}67$

Άρα $\mathrm{T(n)} = \Theta(\mathrm{nlogn})$ με $n_0 \ge 2$, $\mathrm{c_1} \ge 2{,}1$, $\mathrm{c_2} \le 1$.

Ασκήσεις
Εφαρμογή 4

- Έστω αναδρομικός αλγόριθμος που για να επιλύσει ένα πρόβλημα με n δεδομένα, επιλύει 3 υποπροβλήματα με n/3 δεδομένα και έπειτα συνδυάζει τις λύσεις σε χρόνο 12n.
  - Λύστε την αναδρομική σχέση που εκφράζει την πολυπλοκότητα του προβλήματος.
  - Επαληθεύστε την απάντηση για τον χρόνο εκτέλεσης, με τη μέθοδο της αντικατάστασης, προσδιορίζοντας επακριβώς τη σταθερά n_0 και εκείνες (c_1, c_2) του ασυμπτωτικού συμβολισμού. Ως αρχική συνθήκη, ισχύει ότι $T(x)=1$, για κάθε $0 \leq x \leq 1$