Οι διαφάνειες βασίζονται σε αυτές του ακόλουθου μαθήματος:

Introduction to Algorithms (6-046J), MIT

https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-046j-introduction-to-algorithms-sma-5503-fall-2005/

Οι διαφάνειες του ανωτέρω μαθήματος δίνονται υπό την άδεια «Creative Commons Attribution-NonCommercial-ShareAlike 3.0»

Η τεχνική διαίρει και βασίλευε

1. ***Διαίρεσε*** το πρόβλημα σε υποπροβλήματα
2. ***Βασίλευε*** τα υποπροβλήματα επιλύοντας τα αναδρομικά.
3. ***Συνδύασε*** τις λύσεις των υποπροβλημάτων.

Συγχωνευτική ταξινόμηση

1. ***Διαίρεσε:*** Τετριμμένο.
2. ***Βασίλευε:*** Αναδρομικά ταξινόμησε τους 2 υποπίνακες.
3. ***Συνδύασε:*** Συγχώνευση γραμμικού χρόνου.

Συγχωνευτική ταξινόμηση

1. ***Διαίρεσε:*** Τετριμμένο.
2. ***Βασίλευε:*** Αναδρομικά ταξινόμησε τους 2 υποπίνακες.
3. ***Συνδύασε:*** Συγχώνευση γραμμικού χρόνου

$$T(n) = 2\,T(n/2) + \Theta(n)$$

 # υποπροβλήματα

Μέγεθος υποπροβλήματος

Χρόνος για διαίρεση και συνδυασμό

Θεώρημα κυρίαρχου όρου

$$T(n) = a\,T(n/b) + f(n)$$

Περίπτωση 1: $f(n) = O(n^{\log_b a - \varepsilon})$, σταθερά $\varepsilon > 0$

$\Rightarrow T(n) = \Theta(n^{\log_b a})$.

Περίπτωση 2: $f(n) = \Theta(n^{\log_b a} \lg^k n)$, σταθερά $k \geq 0$

$\Rightarrow T(n) = \Theta(n^{\log_b a} \lg^{k+1} n)$.

Περίπτωση 3: $f(n) = \Omega(n^{\log_b a + \varepsilon})$, σταθερά $\varepsilon > 0$, και συνθήκη κανονικότητας

$\Rightarrow T(n) = \Theta(f(n))$.

Συγχ. Ταξιν.: $a = 2, b = 2 \Rightarrow n^{\log_b a} = n^{\log_2 2} = n$

$\Rightarrow$ Περίπτωση 2 $(k = 0) \Rightarrow T(n) = \Theta(n \lg n)$.

Δυαδική αναζήτηση

Βρες ένα στοιχείο σε ένα ταξινομημένο πίνακα:

1. ***Διαίρεσε:*** Έλεγξε το μεσαίο στοιχείο
2. ***Βασίλευε:*** Αναδρομικά αναζήτησε σε 1 υποπίνακα.
3. ***Συνδύασε:*** Τετριμμένο.

Example: Εύρεση του 9

3	5	7	8	9	12	15

Δυαδική αναζήτηση

Βρες ένα στοιχείο σε ένα ταξινομημένο πίνακα:

1. ***Διαίρεσε:*** Έλεγξε το μεσαίο στοιχείο
2. ***Βασίλευε:*** Αναδρομικά αναζήτησε σε 1 υποπίνακα.
3. ***Συνδύασε:*** Τετριμμένο.

Example: Εύρεση του 9

3 5 7 8 9 12 15

Δυαδική Αναζήτηση

Βρες ένα στοιχείο σε ένα ταξινομημένο πίνακα:

1. ***Διαίρεσε:*** Έλεγξε το μεσαίο στοιχείο
2. ***Βασίλευε:*** Αναδρομικά αναζήτησε σε 1 υποπίνακα.
3. ***Συνδύασε:*** Τετριμμένο.

Example: Εύρεση του 9

3 5 7 8 9 12 15

Δυαδική Αναζήτηση

Βρες ένα στοιχείο σε ένα ταξινομημένο πίνακα:

1. ***Διαίρεσε:*** Έλεγξε το μεσαίο στοιχείο
2. ***Βασίλευε:*** Αναδρομικά αναζήτησε σε 1 υποπίνακα.
3. ***Συνδύασε:*** Τετριμμένο.

Example: Εύρεση του 9

3 5 7 8 9 12 15

Binary search

Βρες ένα στοιχείο σε ένα ταξινομημένο πίνακα:

1. ***Διαίρεσε:*** Έλεγξε το μεσαίο στοιχείο
2. ***Βασίλευε:*** Αναδρομικά αναζήτησε σε 1 υποπίνακα.
3. ***Συνδύασε:*** Τετριμμένο.

Example: Εύρεση του 9

3	5	7	8	9	12	15

Αναδρομή για δυαδική αναζήτηση

$$T(n) = 1\,T(n/2) + \Theta(1)$$

 # υποπροβλήματα

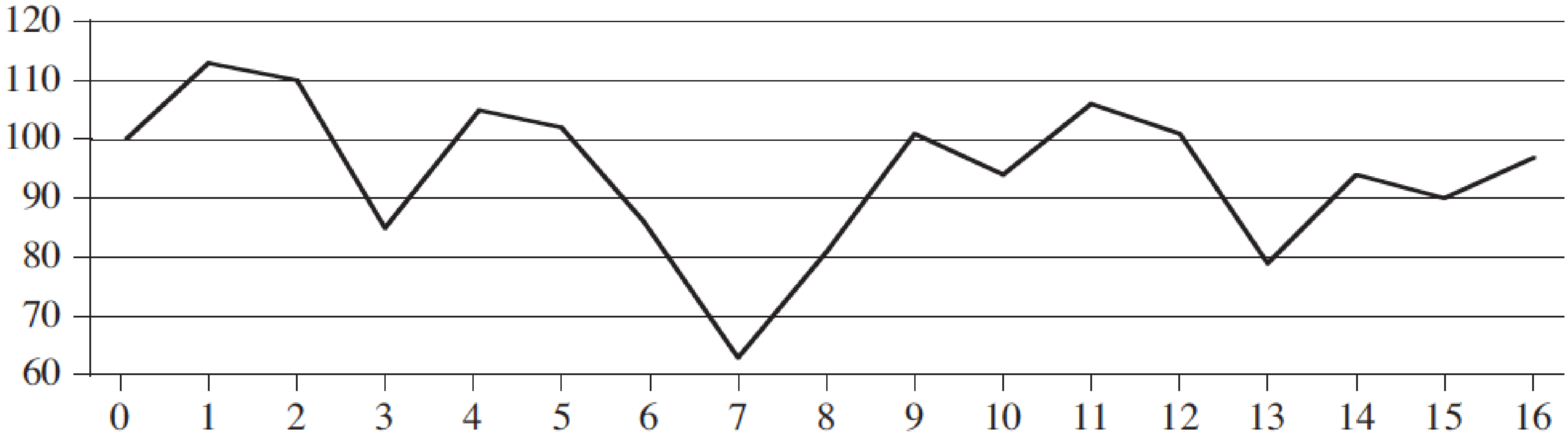
Μέγεθος υποπροβλήματος

Εργασία διαίρεσης και συνδυασμού

$n^{\log_b a} = n^{\log_2 1} = n^0 = 1 \Rightarrow$ Περίπτωση 2 ($k = 0$)

$\Rightarrow T(n) = \Theta(\lg n)$.

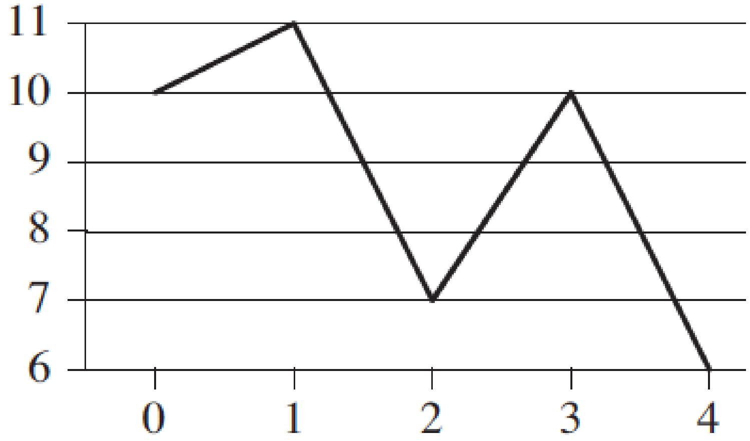
Το πρόβλημα της μέγιστης υποσυστοιχίας



Ημέρα	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Τιμή	100	113	110	85	105	102	86	63	81	101	94	106	101	79	94	90	97
Μεταβολή		13	−3	−25	20	−3	−16	−23	18	20	−7	12	−5	−22	15	−4	7

Σχήμα 4.1 Πληροφορίες για την τιμή των μετοχών της εταιρείας Πτητικά Χημικά Α.Ε. μετά το τέλος της κάθε συνεδρίασης για μια περίοδο 17 ημερών. Στον οριζόντιο άξονα του διαγράμματος απεικονίζεται η ημέρα, και στον κατακόρυφο η τιμή. Στην τελευταία γραμμή του πίνακα παρατίθεται η μεταβολή της τιμής κάθε μέρα σε σχέση με την προηγούμενη μέρα.

Το πρόβλημα της μέγιστης υποσυστοιχίας



Ημέρα	0	1	2	3	4
Τιμή	10	11	7	10	6
Μεταβολή		1	−4	3	−4

Σχήμα 4.2 Ένα παράδειγμα που δείχνει ότι το μέγιστο κέρδος δεν εξασφαλίζεται πάντα αν ξεκινήσουμε από τη χαμηλότερη τιμή ή αν σταματήσουμε στην ψηλότερη. Και πάλι, στον οριζόντιο άξονα απεικονίζεται η ημέρα και στην κατακόρυφο η τιμή. Στην προκειμένη περίπτωση το μέγιστο κέρδος, 3 ευρώ ανά μετοχή, προκύπτει αν αγοράσουμε μετά την ημέρα 2 και πουλήσουμε μετά την ημέρα 3. Η τιμή 7 ευρώ μετά την ημέρα 2 δεν είναι η χαμηλότερη όλων, και η τιμή 10 ευρώ μετά την ημέρα 3 δεν είναι η ψηλότερη όλων.

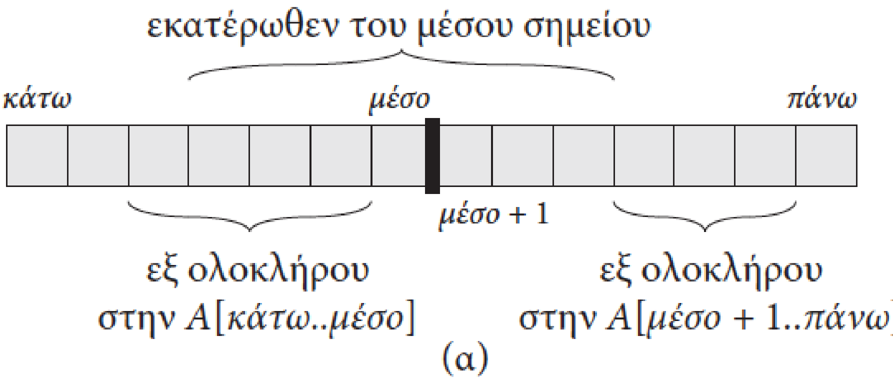
Το πρόβλημα της μέγιστης υποσυστοιχίας

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
A	13	–3	–25	20	–3	–16	–23	18	20	–7	12	–5	–22	15	–4	7

μέγιστη υποσυστοιχία

Σχήμα 4.3 Η μεταβολή της τιμής των μετοχών σαν ένα πρόβλημα μέγιστης υποσυστοιχίας. Στην προκειμένη περίπτωση, η υποσυστοιχία που έχει το μεγαλύτερο άθροισμα απ' όλες τις συνεχείς υποσυστοιχίες της A είναι η $A[8\,..\,11]$, με άθροισμα 43.

Το πρόβλημα της μέγιστης υποσυστοιχίας



Σχήμα 4.4 (α) Πιθανές θέσεις των υποσυστοιχιών της $A[κάτω\,..\,πάνω]$: εξ ολοκλήρου στην $A[κάτω\,..\,μέσο]$, εξ ολοκλήρου στην $A[μέσο+1\,..\,πάνω]$, και εκατέρωθεν του μέσου σημείου *μέσο*. (β) Οποιαδήποτε υποσυστοιχία της $A[κάτω\,..\,πάνω]$ που βρίσκεται εκατέρωθεν του μέσου σημείου αποτελείται από δύο υποσυστοιχίες $A[i\,..\,μέσο]$ και $A[μέσο+1\,..\,j]$, όπου $κάτω \leq i \leq μέσο$ και $μέσο < j \leq πάνω$.

Το πρόβλημα της μέγιστης υποσυστοιχίας

```
FIND-MAX-CROSSING-SUBARRAY(A, low, mid, high)
left-sum = −∞
sum = 0
for i = mid downto low
    sum = sum + A[i]
    if sum > left-sum
        left-sum = sum
        max-left = i
right-sum = −∞
sum = 0
for j = mid + 1 to high
    sum = sum + A[j]
    if sum > right-sum
        right-sum = sum
        max-right = j
return (max-left, max-right, left-sum + right-sum)
```

Το πρόβλημα της μέγιστης υποσυστοιχίας

```
FIND-MAXIMUM-SUBARRAY(A, low, high)
if high == low
    return (low, high, A[low])            // base case: only one element
else mid = ⌊(low + high)/2⌋
    (left-low, left-high, left-sum) =
        FIND-MAXIMUM-SUBARRAY(A, low, mid)
    (right-low, right-high, right-sum) =
        FIND-MAXIMUM-SUBARRAY(A, mid + 1, high)
    (cross-low, cross-high, cross-sum) =
        FIND-MAX-CROSSING-SUBARRAY(A, low, mid, high)
    if left-sum ≥ right-sum and left-sum ≥ cross-sum
        return (left-low, left-high, left-sum)
    elseif right-sum ≥ left-sum and right-sum ≥ cross-sum
        return (right-low, right-high, right-sum)
    else return (cross-low, cross-high, cross-sum)
```

Το πρόβλημα της μέγιστης υποσυστοιχίας

Ο χρόνος εκτέλεσης του «Διαίρει και Βασίλευε» Αλγορίθμου:

$$T(n) = \begin{cases} \Theta(1) \ \text{αν } n = 1 \\ 2T\left(\frac{n}{2}\right) + \Theta(n) \ αν \ n > 1 \end{cases}$$

$$T(n) = \Theta(n \lg n)$$

Ύψωση σε δύναμη

Πρόβλημα: Υπολόγισε a^n, όπου $n \in N$.

Απλός Αλγόριθμος: $\Theta(n)$.

Ύψωση σε δύναμη

Πρόβλημα: Υπολόγισε a^n, όπου $n \in N$.

Απλός Αλγόριθμος: $\Theta(n)$.

Αλγόριθμος Διαίρει και Βασίλευε:

$$a^n = \begin{cases} a^{n/2} \cdot a^{n/2} & \text{αν } n \text{ is άρτιος} \\ a^{(n-1)/2} \cdot a^{(n-1)/2} \cdot a & \text{αν } n \text{ είναι περιττός.} \end{cases}$$

$$T(n) = T(n/2) + \Theta(1) \;\Rightarrow\; T(n) = \Theta(\lg n).$$

Πολλαπλασιασμός Πινάκων

$$\left.\begin{array}{ll}\textbf{Είσοδος:} & A = [a_{ij}],\ B = [b_{ij}]. \\ \textbf{Έξοδος:} & C = [c_{ij}] = A \cdot B.\end{array}\right\}\quad i, j = 1, 2, \ldots, n.$$

$$\begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1n} \\ c_{21} & c_{22} & \cdots & c_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ c_{n1} & c_{n2} & \cdots & c_{nn} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \cdot \begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1n} \\ b_{21} & b_{22} & \cdots & b_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ b_{n1} & b_{n2} & \cdots & b_{nn} \end{bmatrix}$$

$$c_{ij} = \sum_{k=1}^{n} a_{ik} \cdot b_{kj}$$

Ο συνήθης αλγόριθμος

for $i \leftarrow 1$ **to** n
  do for $j \leftarrow 1$ **to** n
    do $c_{ij} \leftarrow 0$
      for $k \leftarrow 1$ **to** n
        do $c_{ij} \leftarrow c_{ij} + a_{ik} \cdot b_{kj}$

Χρόνος εκτέλεσης $= \Theta(n^3)$

Διαίρει και Βασίλευε Αλγόριθμος

Ιδέα:

$n \times n$ πίνακας = 2×2 πίνακας $(n/2) \times (n/2)$ υποπινάκων:

$$\left[\begin{array}{c|c} r & s \\ \hline t & u \end{array}\right] = \left[\begin{array}{c|c} a & b \\ \hline c & d \end{array}\right] \cdot \left[\begin{array}{c|c} e & f \\ \hline g & h \end{array}\right]$$

$$C \quad = \quad A \quad \cdot \quad B$$

$$\left.\begin{aligned} r &= ae + bg \\ s &= af + bh \\ t &= ce + dg \\ u &= cf + dh \end{aligned}\right\}$$

αναδρομή

8 πολ/σμοι $(n/2) \times (n/2)$ υποπινάκων

4 προσθ. $(n/2) \times (n/2)$ υποπινάκων

Ανάλυση του ΔκΒ αλγορίθμου

$$T(n) = 8\,T(n/2) + \Theta(n^2)$$

 # υποπίνακες

Μέγεθος υποπίνακα

Χρόνος για την πρόσθεση των υποπινάκων

$n^{\log_b a} = n^{\log_2 8} = n^3 \Rightarrow$ Περίπτωση 1 $\Rightarrow T(n) = \Theta(n^3)$.

Δεν είναι καλύτερος από το συνηθισμένο αλγόριθμο.

Η Ιδέα του Strassen

- Πολλαπλασίασε 2×2 matrices με μόνο 7 αναδρομικούς πολ/σμους.

$$
\begin{aligned}
P_1 &= a \cdot (f - h) \\
P_2 &= (a + b) \cdot h \\
P_3 &= (c + d) \cdot e \\
P_4 &= d \cdot (g - e) \\
P_5 &= (a + d) \cdot (e + h) \\
P_6 &= (b - d) \cdot (g + h) \\
P_7 &= (a - c) \cdot (e + f)
\end{aligned}
$$

$$
\begin{aligned}
r &= P_5 + P_4 - P_2 + P_6 \\
s &= P_1 + P_2 \\
t &= P_3 + P_4 \\
u &= P_5 + P_1 - P_3 - P_7
\end{aligned}
$$

7 πολ/σμους,
18 προσθ./αφαιρ.

Η Ιδέα του Strassen

- Πολλαπλασίασε 2×2 matrices με μόνο 7 αναδρομικούς πολ/σμους.

$$
\begin{aligned}
P_1 &= a \cdot (f - h) \\
P_2 &= (a + b) \cdot h \\
P_3 &= (c + d) \cdot e \\
P_4 &= d \cdot (g - e) \\
P_5 &= (a + d) \cdot (e + h) \\
P_6 &= (b - d) \cdot (g + h) \\
P_7 &= (a - c) \cdot (e + f)
\end{aligned}
$$

$$
\begin{aligned}
r &= P_5 + P_4 - P_2 + P_6 \\
&= (a + d)(e + h) \\
&\quad + d(g - e) - (a + b)h \\
&\quad + (b - d)(g + h) \\
&= ae + ah + de + dh \\
&\quad + dg - de - ah - bh \\
&\quad + bg + bh - dg - dh \\
&= ae + bg
\end{aligned}
$$

Ο αλγόριθμος του Strassen

1. ***Διαίρεσε:*** Διαμέρισε A και B σε $(n/2)\times(n/2)$ υποπίνακες. Σχημάτισε τους όρους που θα πολλαπλασιαστούν χρησιμοποιώντας $+$ και $-$.
2. ***Βασίλευε:*** Εκτέλεσε 7 πολλαπλασιασμούς $(n/2)\times(n/2)$ υποπινάκων αναδρομικά.
3. ***Συνδύασε:*** Υπολόγισε το C χρησιμοποιώντας $+$ και $-$ στους $(n/2)\times(n/2)$ submatrices.

$$T(n) = 7\,T(n/2) + \Theta(n^2)$$

Ανάλυση του Αλγορίθμου του Strassen

$$T(n) = 7\,T(n/2) + \Theta(n^2)$$

$$n^{\log_b a} = n^{\log_2 7} \approx n^{2.81} \Rightarrow \text{Περίπτωση 1} \Rightarrow T(n) = \Theta(n^{\lg 7}).$$

Ο αριθμός 2.81 μπορεί να μην φαίνεται πολύ μικρότερος του 3, αλλά επειδή η διαφορά είναι στον εκθέτη, η επίδραση στο χρόνο εκτέλεσης είναι σημαντική.

Στην πράξη, ο αλγόριθμος του Strassen είναι καλύτερος από τον απλό αλγόριθμο σε σημερινές αρχιτεκτονικές για $n \geq 32$ περίπου.

Υπάρχει και καλύτερος αλγόριθμος (μόνο θεωρητικού ενδιαφέροντος): $\Theta(n^{2.376\ldots})$.

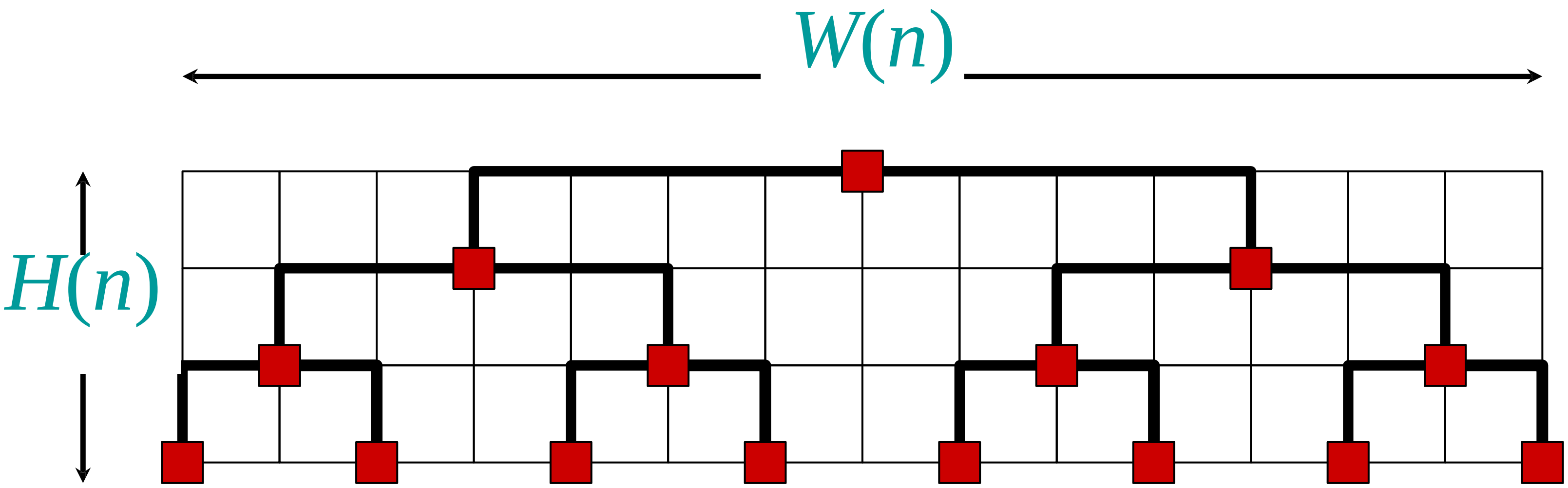
Τοποθέτηση VLSI

Πρόβλημα: Τοποθέτησε ένα πλήρες δυαδικό δένδρο με n φύλλα σε ένα πλέγμα με ελάχιστο εμβαδό.

 Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson

Τοποθέτηση VLSI

Πρόβλημα: Τοποθέτησε ένα πλήρες δυαδικό δένδρο με n φύλλα σε ένα πλέγμα με ελάχιστο εμβαδό.

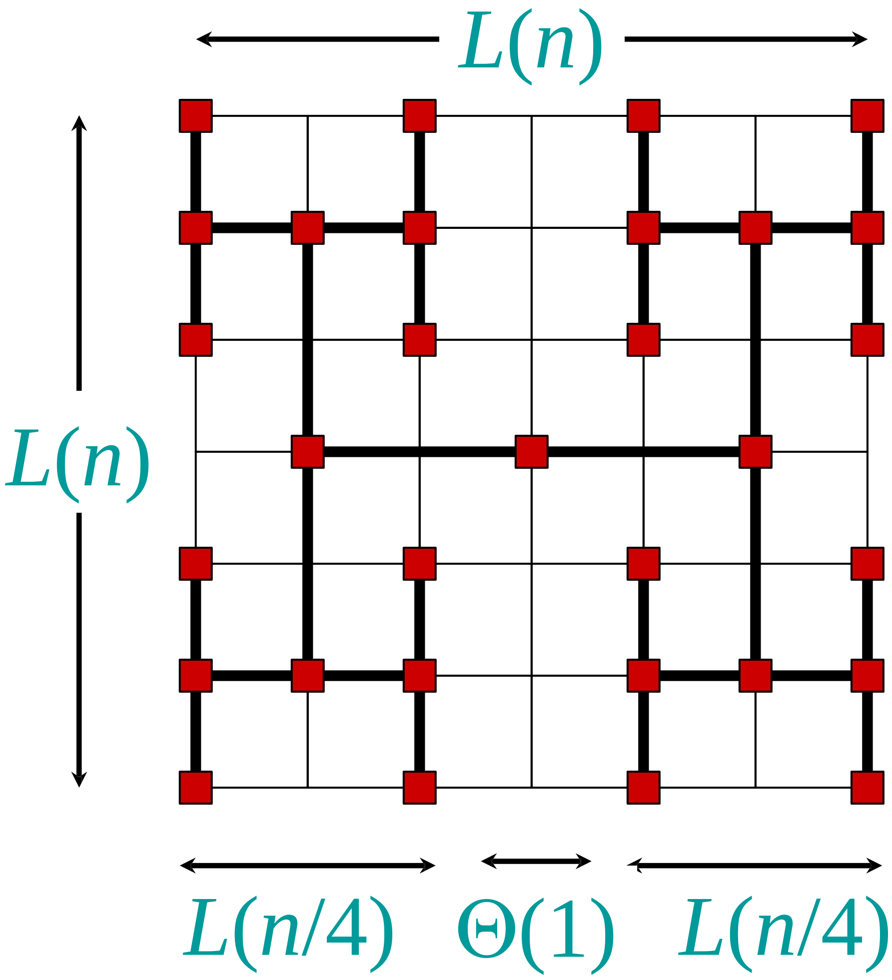


$$H(n) = H(n/2) + \Theta(1) = \Theta(\lg n)$$

$$W(n) = 2\,W(n/2) + \Theta(1) = \Theta(n)$$

Εμβαδό $= \Theta(n \lg n)$

Τοποθέτηση H-tree



$$L(n) = 2\,L(n/4) + \Theta(1)$$
$$= \Theta(\sqrt{n})$$

Εμβαδό $= \Theta(n)$

Συμπεράσματα

- Η Διαίρει και Βασίλευε είναι μία από τις ισχυρές τεχνικές για σχεδίαση αλγορίθμων.
- Οι αλγόριθμοι Διαίρει και Βασίλευε μπορούν να αναλυθούν με τη χρήση αναδρομικών σχέσεων και το θεώρημα του κυρίαρχου όρου.
- Η στρατηγική συχνά οδηγεί σε αποδοτικούς αλγορίθμους.