Chapter 4. Divide-and-Conquer

4.1-1

The maximum subarray algorithm will return a subarray of single element which has a minimum absolute value.

4.1-2

Algorithm 1: Find-maximum-subarray(A), brute-force

```
max-sum = −∞;
max-left = 1, max-right = 1;
for i = 1 to A.length do
    sum = 0;
    for j = i to A.length do
        sum = sum + A[j];
        if max-sum < sum then
            max-left = i, max-right = j, max-sum = sum;
        end
    end
end
return (max-left, max-right, max-sum)
```

4.1-4

To allow an empty subarray, it is enough to check if the answer is better than an empty subarray for the base case.

4.1-5

Algorithm 2: Find-maximum-subarray(A), linear time

```
max-sum = −∞;
running-sum = 0;
max-left = 1, max-right = 1;
curr-left = 1, curr-right = 1;
for i = 1 to A.length do
    if running-sum ≤ 0 then
        curr-left = i;
        running-sum = A[i];
        curr-right = i;
    else
        running-sum = running-sum + A[i];
        curr-right = i;
    end
    if running-sum > max-sum then
        max-sum = running-sum;
        max-left = curr-left;
        max-right = curr-right;
    end
end
return (max-left, max-right, max-sum)
```

4.2-1

$S_1 = 6$, $S_2 = 4$, $S_3 = 12$, $S_4 = -2$, $S_5 = 6$, $S_6 = 8$, $S_7 = -2$, $S_8 = 6$, $S_9 = -6$, $S_{10} = 14$.

$P_1 = 6$, $P_2 = 8$, $P_3 = 72$, $P_4 = -10$, $P_5 = 48$, $P_6 = -12$, $P_7 = -84$.

$C_1 = 18$, $C_{12} = 14$, $C_{21} = 62$, $C_{22} = 66$.

4.2-2

Algorithm 3: Strassen-algorithm(A, B)

n = A.rows;
let C be a new $n \times n$ matrix;
if $n == 1$ **then**
  $c_{11} = a_{11} \cdot b_{11}$;
else
  partition A, B, and C as 4 $(n/2) \times (n/2)$ submatrices;
  $S_1 = B_{12} - B_{22}$;
  $S_2 = A_{11} + A_{12}$;
  $S_3 = A_{21} + A_{22}$;
  $S_4 = B_{21} - B_{11}$;
  $S_5 = A_{11} + A_{22}$;
  $S_6 = B_{11} + B_{22}$;
  $S_7 = A_{12} - A_{22}$;
  $S_8 = B_{21} + B_{22}$;
  $S_9 = A_{11} - A_{21}$;
  $S_{10} = B_{11} + B_{12}$;
  $P_1 = Strassen - algorithm(A_{11}, S_1)$;
  $P_2 = Strassen - algorithm(S_2, B_{22})$;
  $P_3 = Strassen - algorithm(S_3, B_{11})$;
  $P_4 = Strassen - algorithm(A_{22}, S_4)$;
  $P_5 = Strassen - algorithm(S_5, S_6)$;
  $P_6 = Strassen - algorithm(S_7, S_8)$;
  $P_7 = Strassen - algorithm(S_9, S_{10})$;
  $C_{11} = P_5 + P_4 - P_2 + P_6$;
  $C_{12} = P_1 + P_2$;
  $C_{21} = P_3 + P_4$;
  $C_{22} = P_5 + P_1 - P_3 - P_7$;
end
return C

4.2-3

We can insert "padding" to right columns/lower rows of A to make it to a $n \times n$ square matrix in which n is an exact power of 2. This modified algorithm is still $\Theta(n^{\lg 7})$, since zero-padding is $O(n^2)$ and Strassen's algorithm to padded A is $O(n^{\lg 7})$.

4.2-4

Suppose we have an algorithm that divides matrix multiplication to the subproblems of size $n/3$, and for each recursive step it performs k matrix multiplications. Then the time complexity is $\Theta(n^{\log_3 k})$. To have $n^{\log_3 k} < n^{\lg 7}$, $k < 3^{\lg 7}$, so the largest $k = 21$.

4.2-5.
$\log_{68}(132464) = 2.79513$, $\log_{70}(143640) = 2.79512$, $\log_{72}(155424) = 2.79515$, $\log_2(7) = 2.80735$, therefore the second algorithm is best, and it is asymptotically better than Strassen's algorithm.

4.2-6.
If we have $kn \times n$ matrix A and $n \times kn$ matrix B, then $A \times B$ is a $kn \times kn$ matrices where each $n \times n$ matrix is a product of two $n \times n$ submatrices from A and B, which can be calculated using Strassen's algorithm in $\Theta(n^{\lg 7})$ time. Since we have k^2 such matrices, the total time complexity is $\Theta(k^2 \lg 7)$.
If we reverse the order, then we need to perform k matrix multiplication of $n \times n$ submatrices, so the total time complexity is $\Theta(k \lg 7)$.

4.2-7.
First, calculate these three products: ac, bd, $(a+b)(c+d)$.
Then the real component is $ac-bd$, the imaginary component is $(a+b)(c+d)-ac-bd = ad+bc$, which can be computed using only additions/subtractions.

4.3-1.
Assume that $T(n) \le n^2$ holds for $n = 1, \cdots, n-1$. By induction hypothesis, $T(n) \le (n-1)^2 + n = n^2 - n + 1 < n^2$, so $T(n) = O(n^2)$.

4.3-2.
Assume that $T(n) \le c \lg(n-b)$ for some $b > 2, c > 1$, $n = 1, \cdots, n-1$. By induction hypothesis, $T(n) \le c \lg(\lceil \frac{n}{2} \rceil - b) + 1 < c \lg(\frac{n}{2} - b + 1) + 1 < c \lg(n - 2b + 2) - c + 1 < c \lg(n-b)$, so $T(n) = O(\lg n)$.

4.3-3.
Assume that $T(n) \ge c(n+2) \lg(n+2)$ for $n = 1, \cdots, n-1$. By induction hypothesis, $T(n) \ge 2c(\lfloor \frac{n}{2} \rfloor + 2) \lg(\lfloor \frac{n}{2} \rfloor + 2) + n \ge 2c(\frac{n}{2} + 1) \lg(\frac{n}{2} + 1) + n = c(n+2) \lg(2) - c(n+2) \lg 2 + n = c(n+2) \lg(n+2) + (1-c)n - 2c \ge c(n+2) \lg(n+2)$ for $n \ge \frac{2c}{1-c}$, so $T(n) = \Omega(n \lg n)$, which concludes $T(n) = \Theta(n \lg n)$.

4.3-4.
Assume that $T(n) \le n \lg n + n$ for $n = 1, \cdots, n-1$. By induction hypothesis, $T(n) \le 2(c\lfloor \frac{n}{2} \rfloor \lg \lfloor \frac{n}{2} \rfloor + \lfloor \frac{n}{2} \rfloor) + n \le 2c\frac{n}{2} \lg \frac{n}{2} + 2\frac{n}{2} + n = cn \lg \frac{n}{2} + 2n = cn \lg n + (2-c)n \le cn \lg n + n$ for $c \ge 1$, so $T(n) = O(n \lg n)$.

4.3-5.
To show $T(n) = O(n \lg n)$, assume that $T(n) \le c(n-2) \lg(n-2)$ for $n = 1, \cdots, n-1$. By induction hypothesis, $T(n) \le c(\lceil \frac{n}{2} \rceil - 2) \lg(\lceil \frac{n}{2} \rceil - 2) + c(\lfloor \frac{n}{2} \rfloor - 2) \lg(\lfloor \frac{n}{2} \rfloor - 2) + dn \le c(\frac{n}{2} - 1) \lg(\frac{n}{2} - 1) + c(\frac{n}{2} - 2) \lg(\frac{n}{2} - 2) + dn \le 2c\frac{n-2}{2} \lg \frac{n-2}{2} + dn = c(n-2) \lg(n-2) + (d-c)n + 2c \le c(n-2) \lg(n-2)$ for $c > d$, so $T(n) = O(n \lg n)$.
To show $T(n) = \Omega(n \lg n)$, assume that $T(n) \ge c(n+2) \lg(n+2)$ for $n = 1, \cdots, n-1$. By induction hypothesis, $T(n) \ge c(\lceil \frac{n}{2} \rceil + 2) \lg(\lceil \frac{n}{2} \rceil + 2) + c(\lfloor \frac{n}{2} \rfloor + 2) \lg(\lfloor \frac{n}{2} \rfloor + 2) + dn \ge c(\frac{n}{2} + 2) \lg(\frac{n}{2} + 2) + c(\frac{n}{2} + 1) \lg(\frac{n}{2} + 1) + dn \ge 2c\frac{n+2}{2} \lg \frac{n+2}{2} + dn = c(n+2) \lg(n+2) + (d-c)n - 2c \ge c(n+2) \lg(n+2)$ for $c < d$, so $T(n) = \Omega(n \lg n)$.
Combining these gives $T(n) = \Theta(n \lg n)$.

4.3-6.
Assume that $T(n) \le c(n-34) \lg(n-34)$ for $n = 1, \cdots, n-1$. By induction hypothesis, $T(n) \le 2c(\lfloor \frac{n}{2} \rfloor - 17) \lg(\lfloor \frac{n}{2} \rfloor - 17) + n \le 2c(\frac{n}{2} - 17) \lg(\frac{n}{2} - 17) + n = c(n-34) \lg(n-34) - c(n-34) + n \le c(n-34) \lg(n-34)$ for $c > 35$, so $T(n) = O(n \lg n)$.

4.3-7.
Assuming $T(n) \le cn^{\log_3 4}$ for $n = 1, \cdots, n-1$ gives $T(n) \le 4c(\frac{n}{3})^{\log_3 4} + n = cn^{\log_3 4} + n$.
Instead we assume here $T(n) \le cn^{\log_3 4} - dn$ for $n = 1, \cdots, n-1$, which gives $T(n) \le 4(c(\frac{n}{3})^{\log_3 4} - \frac{dn}{3}) + n = cn^{\log_3 4} - \frac{4}{3}dn + n \le cn^{\log_3 4} - dn$, for $d \ge 3$.

4.3-8.
Assuming $T(n) \le cn^2$ for $n = 1, \cdots, n-1$ gives $T(n) \le 4c(\frac{n}{2})^2 + n^2 = (c+1)n^2$ fails.
Instead we assume here $T(n) \le cn^2 \lg n - n$ for $n = 1, \cdots, n-1$, which gives $T(n) \le 4(c(\frac{n}{2})^2 \lg \frac{n}{2} - \frac{n}{2}) + n^2 = cn^2 \lg n - n - cn^2 + n^2 - n \le cn^2 \lg n - n$ for $c \ge 1$, so we have $T(n) = O(n^2 \lg n)$.
To show $T(n) = \Omega(n^2 \lg n)$, we assume $T(n) \ge cn^2 \lg n + n$ for $n = 1, \cdots, n-1$, which gives $T(n) \ge 4(c(\frac{n}{2})^2 \lg \frac{n}{2} + \frac{n}{2}) + n^2 = cn^2 \lg n + n - cn^2 + n^2 + n \ge$

$cn^2 \lg n + n$ for $c \geq 1$.
Combining these gives $T(n) = \Theta(n^2 \lg n)$.

4.3-9.
Set $n = 10^m$ and $S(m) = T(10^m)$, then $S(m) = 3S(\frac{m}{2}) + m$, which leads to $S(m) = \Theta(m^{\lg 3}) \Rightarrow T(n) = \Theta((\log n)^{\lg 3})$.

4.4-1.

$$T(n) = n \sum_{i=0}^{\lg n} (\frac{3}{2})^i = O(n^{\lg 3}).$$

4.4-2.

$$T(n) = n^2 \sum_{i=0}^{\lg n} (\frac{1}{2})^i = O(n^2).$$

4.4-3.

$$T(n) = \sum_{i=0}^{\lg n} (\frac{n}{2^i} + 2)4^i = \sum_{i=0}^{\lg n} (n \cdot 2^i + 2 \cdot 4^i) = O(n^2).$$

4.4-4.

$$T(n) = \sum_{i=0}^{n} 2^i = O(2^n).$$

4.4-5.

$$T(n) \leq 2T(n-1) + n \leq \sum_{i=0}^{n} 2^i (n-i) = n2^n - (n-1)2^n = O(2^n).$$

$$T(n) \geq 2T(\frac{n}{2}) + n \geq \sum_{i=0}^{\lg n} 2^i (\frac{n}{2^i}) = \Omega(n^2).$$

4.4-6.

$$T(n) \geq 2T(\frac{n}{3}) + cn \geq cn \sum_{i=0}^{\log_3 n} 2^i (\frac{1}{3^i}) = \Omega(n \lg n).$$

4.4-7.

$$T(n) = \sum_{i=0}^{\lg n} 4^i \frac{1}{2^i} cn = \Theta(n^2).$$

4.4-8.

$$T(n) = \sum_{i=0}^{\frac{n}{a}} c(n - ia) + (\frac{n}{a})ca = \Theta(n^2).$$

4.4-9.
WLOG we assume $0 < \alpha \leq \frac{1}{2}$.

$$T(n) = \sum_{i=0}^{\log_{\frac{1}{\alpha}} n} cn = \Theta(n \lg n).$$

4.5-1.
a. $a = 2, b = 4, f(n) = 1 = O(n^{\log_4 2 - 0.5}) \Rightarrow T(n) = \Theta(\sqrt{n})$.
b. $a = 2, b = 4, f(n) = \sqrt{n} = \Theta(n^{\log_4 2}) \Rightarrow T(n) = \Theta(\sqrt{n} \lg n)$.
c. $a = 2, b = 4, f(n) = n = \Omega(n^{\log_4 2 + 0.5})$. $2f(n/4) = n/2 \leq (1/2)f(n) \Rightarrow$ $T(n) = \Theta(f(n)) = \Theta(n)$.
d. $a = 2, b = 4, f(n) = n^2 = \Omega(n^{\log_4 2 + 1.5})$. $2f(n/4) = n^2/8 \leq (1/8)f(n) \Rightarrow$ $T(n) = \Theta(f(n)) = \Theta(n^2)$.

4.5-2.
$n^{\log_4 a} < n^{\lg 7} \Rightarrow a < 49$. Therefore, the biggest integer $a = 48$.

4.5-3.
$a = 1, b = 2, f(n) = \Theta(1) = \Theta(n^{\log_2 1}) \Rightarrow T(n) = \Theta(\lg n)$.

4.5-4.
The master's theorem cannot be applied in this case. $n^{\log_b a} = n^2$, but $f(n)$ is not polynomially bigger than n^2, there is no $\epsilon > 0$ such that $f(n) = \Omega(n^{2+\epsilon})$. Instead we directly use the recursion tree here.

$$T(n) = \sum_{i=0}^{\lg n - 1} 4^i \frac{n^2}{4^i} \lg \frac{n}{2^i} = \sum_{i=0}^{\lg n - 1} n^2(\lg n - i) = n^2 \lg^2 n - n^2 \frac{\lg n(\lg n - 1)}{2} = \Theta(n^2 \lg^2 n).$$

4.5-5.
Let $a = 1, b = 2, f(n) = n(2 - \cos n)$. $f(n)$ is polynomially bigger than $n^{\log_b a} = 1$, but to that $f(n/2) = \frac{n}{2}(2 - \cos\frac{n}{2}) < n(2 - \cos n)$ holds, we must have $-\frac{\cos(n/2)}{2} < 1 - \cos n$ for sufficiently large n, but then the right-hand side must be bigger than $1/2$, which is impossible.

4.6-1.
For base b representation of n, shift the representation by i and add the last digit by 1 only if we have at least one nonzero shifted digit then we're done.

4.6-2.

$$T(n) = \sum_{i=0}^{\log_b n-1} a^i f(\frac{n}{b^i}) = \sum_{i=0}^{\log_b n-1} a^i \Theta((\frac{n}{b^i})^{\log_b a} \lg^k \frac{n}{b^i})$$

$$= \sum_{i=0}^{\log_b n-1} \Theta(n^{\log_b a}(\frac{a}{b^{\log_b a}})^i \lg^k \frac{n}{b^i}) = \sum_{i=0}^{\log_b n-1} \Theta(n^{\log_b a} \lg^k \frac{n}{b^i})$$

$$= \Theta(n^{\log_b a} \sum_{i=0}^{\log_b n-1} \lg^k \frac{n}{b^i}) = \Theta(n^{\log_b a} \sum_{i=0}^{\log_b n-1} (\lg n - i \lg b)^k)$$

$$= \Theta(n^{\log_b a} \sum_{i=0}^{\log_b n-1} (\lg n)^k) = \Theta(n^{\log_b a} \lg_b n (\lg n)^k)$$

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

4.6-3.

$$af(\frac{n}{b}) \le cf(n) \Rightarrow \frac{a}{c} f(n) \le f(nb) \Rightarrow (\frac{a}{c})^i f(1) \le f(b^i)$$

$$\Rightarrow f(n) \ge (\frac{a}{c})^{\log_b n} f(1) = \Theta(n^{\log_b (a/c)}) = \Theta(n^{\log_b a - \log_b c})$$

Since $c < 1$, we have for some $\epsilon > 0$,

$$\Rightarrow f(n) = \Omega(n^{\log_b a + \epsilon}).$$

4-1.

a. $a = 2, b = 2, f(n) = n^4 = \Omega(n^{\log_2 2+3})$ and $2f(n/2) = n^4/8 \leq (1/8)f(n) \Rightarrow T(n) = \Theta(n^4)$.
b. $a = 1, b = 10/7, f(n) = n = \Omega(n^{\log_{10/7} 1+1})$ and $f(7n/10) = 7n/10 \leq (7/10)f(n) \Rightarrow T(n) = \Theta(n)$.
c. $a = 16, b = 4, f(n) = n^2 = \Theta(n^{\log_4 16}) \Rightarrow T(n) = \Theta(n^2 \lg n)$.
d. $a = 7, b = 3, f(n) = n^2 = \Omega(n^{\log_3 7+0.2})$ and $7f(n/3) = 7n^2/9 \leq (7/9)f(n) \Rightarrow T(n) = \Theta(n^2)$.
e. $a = 7, b = 2, f(n) = n^2 = O(n^{\log_2 7-0.8})$ and $\Rightarrow T(n) = \Theta(n^{\lg 7})$.
f. $a = 4, b = 2, f(n) = \sqrt{n} = \Theta(n^{\log_4 2})$ and $\Rightarrow T(n) = \Theta(\sqrt{n} \lg n)$.
g. $T(n) = \sum_{i=0}^{n/2} (2i)^2 = \Theta(n^3)$.

4-2.
a.
1) $T(n) = T(n/2) + \Theta(1)$. By master's theorem, $T(n) = \Theta(\lg n)$.
2) $T(n) = T(n/2) + \Theta(N) = \sum_{i=0}^{\lg n-1} \Theta(N) = \Theta(n \lg n)$.
3) $T(n) = T(n/2) + \Theta(n) = \sum_{i=0}^{\lg n-1} \Theta(n/(2^i)) = \Theta(n)$.
b.
1) $T(n) = 2T(n/2) + \Theta(n)$. By master's theorem, $T(n) = \Theta(n \lg n)$.
2) $T(n) = 2T(n/2) + \Theta(N) = \sum_{i=0}^{\lg n-1} 2^i \Theta(N) = \Theta(n^2)$.
3) $T(n) = T(n/2) + \Theta(n) = \Theta(n \lg n)$.

4-3.
a. By master's theorem, $T(n) = \Theta(n^{\log_3 4})$.
b. First we show that $T(n) = O(n \lg n)$.
$T(n) = 3T(n/3) + n/\lg n \leq 3c(n/3) \lg(n/3) + n/\lg n = cn \lg n - cn \lg 3 + n/\lg n \leq cn \lg n$.
Now, we show that $T(n) = \Omega(n^{1-\epsilon})$ for all $\epsilon > 0$, that is, $T(n) = \Omega(n)$.
$T(n) = 3T(n/3) + n/\lg n \geq 3c(n/3)^{1-\epsilon} + n/\lg n = 3^\epsilon cn^{1-\epsilon} + n/\lg n = cn^{1-\epsilon}(3^\epsilon + n^\epsilon/(c \lg n)) \geq cn^{1-\epsilon}$.
Therefore, $T(n) = \tilde{\Omega}(n)$ and $T(n) = O(n \lg n)$.
c. By master's theorem, $T(n) = \Theta(n^{2.5})$.
d. The subtraction in $T(n/3 - 2)$ won't change the asymptotic bounds, so dropping out the subtraction enables applying the master's theorem, which gives $T(n) = \Theta(n \lg n)$.
e. Same with b., $T(n) = \tilde{\Omega}(n)$ and $T(n) = O(n \lg n)$.
f. First we show that $T(n) = O(n)$.
$T(n) = T(n/2) + T(n/4) + T(n/8) + n \leq (7/8)cn + n \leq cn$, for $c \geq 8$.

$T(n) = \Omega(n)$ is trivial, so we have $T(n) = \Theta(n)$.
g. $T(n) = \sum_{i=1}^{n} 1/i = \Theta(\lg n)$.
h. $T(n) = \sum_{i=1}^{n} \lg i < \int_1^n (\lg(x+1) - \lg x)dx = \Theta(n \lg n)$.
i. $T(n) = \sum_{i=1}^{n/2} 1/(\lg(2i)) = \Theta(n/\lg n)$.
j. Let k be the smallest k such that $n^{2^{-k}} < 2$, then we have $k = \Theta(\lg \lg n)$. $T(n) = \Theta(\sum_{i=1}^{k} n) = \Theta(kn) = \Theta(n \lg \lg n)$.

4-4.
a.

$$z + zF(z) + z^2F(z) = z + z\sum_{i=0}^{\infty} F_i z_i + z^2 \sum_{i=0}^{\infty} F_i z_i$$

$$= z + F_1 z + \sum_{i=2}^{\infty} (F_{i-1} + F_{i-2}) z_i = z + F_1 z + \sum_{i=2}^{\infty} F_i z_i = F(z).$$

b.

$$F(z) = \frac{F(z)(1-z-z^2)}{1-z-z^2} = \frac{F(z) - zF(z) - z^2F(z) - z + z}{1-z-z^2}$$

$$= \frac{F(z) - F(z) + z}{1-z-z^2} = \frac{z}{1-z-z^2}$$

$$= \frac{z}{1-(\phi+\hat{\phi})z + \phi\hat{\phi}z^2} = \frac{z}{(1-\phi z)(1-\hat{\phi}z)}$$

$$= \frac{\sqrt{5}z}{\sqrt{5}(1-\phi z)(1-\hat{\phi}z)} = \frac{(\phi - \hat{\phi})z + 1 - 1}{\sqrt{5}(1-\phi z)(1-\hat{\phi}z)}$$

$$= \frac{(1-\hat{\phi}z) - (1-\phi z)}{\sqrt{5}(1-\phi z)(1-\hat{\phi}z)} = \frac{1}{\sqrt{5}}(\frac{1}{1-\phi z} - \frac{1}{1-\hat{\phi}z}).$$

c.

$$F(z) = \frac{1}{\sqrt{5}}(\frac{1}{1-\phi z} - \frac{1}{1-\hat{\phi}z}) = \frac{1}{\sqrt{5}}(\sum_{i=0}^{\infty} \phi^i z^i - \sum_{i=0}^{\infty} \hat{\phi}^i z^i) = (\sum_{i=0}^{\infty} \frac{1}{\sqrt{5}}(\phi^i - \hat{\phi}^i) z^i$$

d. Since $|\frac{\hat{\phi}^i}{\sqrt{5}}| < 0.5$ for $i > 2$, round$(F_i) = \frac{\phi^i}{\sqrt{5}}$.

4-5.
a. In this case, we have that for any good chip there is corresponding bad chip. If the bad chip try to fool the professor by always diagnosing a good

chip as bad and a bad chip as good, then the test remains inconclusive.
b. In this case, we have that for any bad chip there is corresponding good chip. We take $\lceil n/2 \rceil$ pairwise tests with the following strategy: If the test say at least one of two chips is bad, discard both of them. Otherwise, discard one of them. Given that the number of bad chip is no bigger than the number of smaller chips, after conducting the pairwise tests, the size of problem is at least halved and the underlying assumption is left unchanged.
c. We recursively do the process discussed in b. The recursion ends when the number of remaining chips ≤ 2, in which case you can just pick one of them to return a good chip. Once a good chip is identified, the rest of chips can be diagnosed with that chip, so the number of tests is $T(n) = T(n/2) + \Theta(n/2) = \Theta(n)$.

4-6.
a. $\Rightarrow$: trivial.
$\Leftarrow$: From $A[i,j] + A[i+1,j+1] \leq A[i,j+1] + A[i+1,j]$, we can deduce $A[i,j]+A[i,j+1]+A[i+1,j+1]+A[i+1,j+2] \leq A[i,j+1]+A[i+1,j]+A[i,j+2]+A[i+1,j+1] \Rightarrow A[i,j]+A[i+1,j+2] \leq A[i+1,j]+A[i,j+2]$. Symmetrically and repeatedly applying this gives $A[i,j] + A[k,l] \leq A[i,l] + A[k,j]$.
b. For $A[1,3]$, change 22 to 24.
c. If $i < k$ but $f(i) > f(k)$, then $A[i,f(i)] + A[k,f(k)] \geq A[i,f(k)] + A[k,f(i)]$. However, since $A[i,f(i)] \leq A[i,f(k)]$ and $A[k,f(k)] \leq A[k,f(i)]$. this implies $A[i,f(i)] = A[i,f(k)]$, a contradiction to the fact that $f(i)$ is the leftmost minimal column of row i.
d. Since $f(2k) \leq f(2k+1) \leq f(2k+2)$, we have $T(m,n) = \sum_{k=0}^{m/2-1}(f(2k+2) - f(2k) + 1) = \sum_{k=i}^{m/2} f(2k) - \sum_{k=0}^{m/2-1} f(2k) + m/2 = f(m) - f(0) + m/2 \leq n + m/2 = O(m+n)$.
e. $T(m) = T(m/2) + O(n+m) \leq \sum_{i=0}^{\lg m - 1} O(n) + \sum_{i=0}^{\lg m - 1} m/2^i = O(n \lg m + m)$.