

Practice

Time Complexity

$f(n)$	$g(n)$	$O/\Omega/\Theta$
$n - 100$	$n - 200$	Θ
$n^{1/2}$	$n^{2/3}$	O
$100n + \log n$	$n + (\log n)^2$	$\Theta(a)$
$\log 2n$	$\log 3n$	$\Theta(b)$
$10 \log n$	$\log n^2$	$\Theta(c)$
$n^{1/2}$	$5^{\log_2 n}$	$O(d)$
2^n	2^{n+1}	$\Theta(e)$

(a): n dominates $(\log n)^c \rightarrow n + (\log n)^2 = \Theta(n)$

(b): $\log ab = \log a + \log b$

(c): $\log a^b = b \log a$

(d): $5 = 2^{2x}$ where $x > 0 \rightarrow 5^{\log_2 n} = (2^{2x})^{\log_2 n} = (2^{\log_2 n})^{2x} = \Omega(n^{1/2})$

(e): $2^{n+1} = 2 \times 2^n$

Fibonacci 1

$$F_n = \begin{cases} 0, & \text{if } n = 0 \\ 1, & \text{if } n = 1 \\ F_{n-1} + F_{n-2}, & \text{otherwise} \end{cases}$$

Prove: $F_n = \Omega(\sqrt{2}^n)$.

By trial and error: It appears that $F(n) \geq 2^{n/2}$ for all $n \geq 7$

To prove: For all positive integers $n \geq 7 \rightarrow F_n \geq 2^{n/2}$

By induction on n . Base case: $n = 7$

Step, assume: Indeed, true that for all $i = 7, 8, \dots, k \rightarrow F_i \geq 2^{i/2}$

To prove: $F_{k+1} \geq 2^{(k+1)/2}$

Practice

$$\text{LHS: } F_{k+1} = F_k + F_{k-1} \geq 2^{k/2} + 2^{(k-1)/2}$$

$$\text{Suffices to prove: } 2^{k/2} + 2^{(k-1)/2} \geq 2^{(k+1)/2}$$

$$2^{1/2} + 1 \geq 2^{2/2} \text{ (by dividing the above by } 2^{(k-1)/2} \text{)}$$

$$\text{It is indeed true that } 2^{1/2} + 1 \geq 2^{2/2} = 2$$

Multiplication

Figure 1.1 Multiplication à la Français.

```
function multiply(x, y)
```

Input: Two n -bit integers x and y , where $y \geq 0$

Output: Their product

```
if  $y = 0$ : return 0
```

```
 $z = \text{multiply}(x, \lfloor y/2 \rfloor)$ 
```

```
if  $y$  is even:
```

```
    return  $2z$ 
```

```
else:
```

```
    return  $x + 2z$ 
```

Suppose instead of both x and y being n -bit, x is n -bit and y is m -bit. What is the worst-case time efficiency of *multiply*?

Proposed: $O(nm)$

Time Efficiency:

- # recursive calls x time/call
- # worst case recursive calls = $O(m)$
- Worst case time/call =
 - $2z$ is at worst $O(n + m) \rightarrow$ because very last addition is $2z = xy - x$
 - x is n bits
 - So, addition's time: $O(\max\{n, n + m\}) = O(\max\{n, m\})$

So, final answer: $O(m \times \max\{n, m\})$

Fibonacci 2

Practice

Let F_n be the n^{th} Fibonacci number, Prove $F_n = O(2^n)$.

- Somewhere, we have shown: $F_n = \Omega(\sqrt{2}^n)$
- But here, seek to show: There exists positive real $F_n \leq c \cdot 2^n$, for all n in N
- Natural proof strategy for “there exists” – construction (i.e., propose some concrete c , and show that it works)
- Try some small values for n , and see what c would work
 - $n = 0, F_0 = 0, 2^0 = 1 \rightarrow c = 1 \text{ works}$
 - $n = 1, F_1 = 1, 2^1 = 2 \rightarrow c = 1 \text{ works}$
 - $n = 2, F_2 = 1, 2^2 = 4 \rightarrow c = 1 \text{ works}$
 - $n = 3, F_3 = 2, 2^3 = 8 \rightarrow c = 1 \text{ works}$
 - $n = 4, F_4 = 3, 2^4 = 16 \rightarrow c = 1 \text{ works}$
- Appears that $c = 1$ works. Adopt it and check if proof goes through. Now, proof by induction with $c = 1$
- Base case, $n = 1, F_1 = 1, 2^1, 1 \leq 2 \rightarrow \text{True}$
- Step: Seek to show $F_n \leq 2^n$ given that $F_k \leq 2^k$ for all $k = 1, 2, \dots, n - 1$
- $F_n = F_{n-1} + F_{n-2} \leq 2^{n-1} + 2^{n-2}$ by induction assumption
- $F_n = 2^{n-2} (2 + 1) = 3 \times 2^{n-2} \leq 2^n = 2^2 \times 2^{n-2} = 4 \times 2^{n-2} \rightarrow \text{Done}$

Fibonacci 3

Let F_n be the n^{th} Fibonacci number, Prove $F_n \neq O(n^2)$.

- Recall from logic: not (there exists an egg-laying mammal) = for all mammals m , m is not egg-laying
- Here, $f = O(g)$: There exists positive real c , for all natural n , $f(n) \leq c \cdot g(n)$
- So here, need to prove: Given any positive real c , it is true that there exists n such that $F_n > c \cdot n^2$
- By contradiction: Suppose that there exists positive real c , such that, for all natural n , $F_n \leq c \cdot n^2$
- Then: $F_n = F_{n-1} + F_{n-2} \leq c(n-1)^2 + c(n-2)^2 = c(n^2 - 2n + 1 + n^2 - 4n + 4) = c(2n^2 - 6n + 5) \leq cn^2$
- $2n^2 - 6n + 5 \leq n^2$

Practice

- $2 - \frac{1}{n^2}(6n - 5) \leq 1$
- This is true only if $\frac{1}{n^2}(6n - 5)$ is “large” compared to $2n^2$
- What is large? We need $\frac{1}{n^2}(6n - 5) \geq 1 \rightarrow \text{true for } n = 1$
- Try $n = 2$: $\frac{1}{4}(12 - 5) = \frac{7}{4} \geq 1$
- Try $n = 3$: $\frac{1}{8}(18 - 5) = \frac{13}{8} \geq 1$
- Try $n = 4$: $\frac{1}{16}(24 - 5) = \frac{19}{16} \geq 1$
- Try $n = 5$: $\frac{1}{25}(30 - 5) = 1$
- Try $n = 6$: $\frac{1}{36}(36 - 5) < 1$
- Try $n = 7$: $\frac{1}{49}(42 - 5) < 1$
- Prove by induction: $6n - 5 < n^2$ for all natural $n > 5$
- Base case $n = 6$: See above
- Step: $6(n - 1) - 5 \leq (n - 1)^2 \rightarrow \text{from induction assumption}$
- $6n - 5 - 6 < n^2 - 2n + 1$
- $6n - 5 \leq n^2 - (2n - 7) \leq n^2 \text{ whenever } 2n - 7 \geq 0 \rightarrow \text{which it is for } n \geq 6$
- So far: We have shown that indeed, for $n \geq 6$, $F_n < cn^2 \rightarrow \text{Done}$

Selection Sort

```
SELECTIONSORT(A[1, ..., n])  
  foreach i from 1 to n do  
     $m \leftarrow i - 1 + \text{INDEXOFMIN}(A[i, \dots, n])$   
    if  $i \neq m$  then swap  $A[i], A[m]$   
  
    INDEXOFMIN(B[1, ..., m])  
       $\min \leftarrow B[1], \text{idx} \leftarrow 1$   
      foreach j from 2 to m do  
        if  $B[j] < \min$  then  
           $\min \leftarrow B[j], \text{idx} \leftarrow j$   
      return idx
```

What is a meaningful characterization of the time efficiency of *SELECTIONSORT*?

Practice

- Suppose we invoke $INDEXOFMIN(A[5, \dots, 13])$. In $INDEXOFMIN: B[1, \dots, 9]$.
Suppose now, min is at index 3 in $B[1, \dots, 9]$. This \rightarrow index of a min in $A[5, \dots, 13]$ is at index $(5 - 1) + 3 = 7$
- Suppose on input: $A[1, \dots, 5] = [13, -23, 45, -23, 1]$. Then A evolves in *SELECTIONSORT* as follows:
 - $i = 1, m = 2, [-23, 13, 45, -23, 1]$
 - $i = 2, m = 4, [-23, -23, 45, 13, 1]$
 - $i = 3, m = 4, [-23, -23, 13, 45, 1]$
- For time efficiency: Need to make meaningful assumption(s)
- Customary Assumptions: (1) n is unbounded, (ii) each $A[i]$ is bounded
- What should we count? Suppose we all agree that counting # swaps is a meaningful measure for time efficiency
- Then: *Worst case # swaps* $= n - 1 = \Theta(n)$
- Now, let's say we want to get a bit more fine-grained. Incorporate (worst case) time for each swap x # swaps
- So now, time efficiency: $(n - 1) + (n - 2) + \dots + 1 = \Theta(n^2)$

Modular Simplification

1. Is $6^6 \equiv 5^3 \pmod{31}$?

$$6 \times 6 = 36 \equiv 5 \pmod{31}$$

$$\text{So: } (6^2)^3 \equiv (5)^3 \pmod{31}$$

2. $2^{125} \equiv ? \pmod{127}$

$$2^7 = 128 = 127 + 1$$

$$\text{So: } 128 \pmod{127} = 1$$

$$\text{Now: } 125/7 = 17 + 6/7$$

$$\text{So: } 2^{125} = 2^{17 \times 7 + 6} = 2^{17 \times 7} \times 2^6$$

$$\text{So: } 2^{125} \equiv 2^{17 \times 7} \times 2^6 \equiv (2^7)^{17} \times 2^6 \equiv 1^{17} \times 2^6 \equiv 64 \pmod{127}$$

Practice

3. Is $4^{1536} - 9^{4824}$ divisible by 35?

$$4^{1536} \equiv 9^{4824} \pmod{35}$$

Trick: Keep exponentiating until numbers start to repeat.

Suppose we repeatedly exponentiate 4:

$$4$$

$$\rightarrow 16$$

$$\rightarrow 64 \equiv 29 \pmod{35}$$

$$\rightarrow 116 = 35 \times 3 + 11 \equiv 11 \pmod{35}$$

$$\rightarrow 9 \pmod{35}$$

$$\rightarrow 36 \equiv 1 \pmod{35}$$

So: $4^6 \equiv 1 \pmod{35}$. And $1536 = 6 \times 256$. So $4^{1536} \equiv 1 \pmod{35}$

Now check whether 1536 is divisible by 4. Indeed: $1536 = 4 \times 384$

Repeat with 9. Repeated exponentiation of 9:

$$9$$

$$\rightarrow 81 \equiv 11 \pmod{35}$$

$$\rightarrow 99 \equiv 29 \pmod{35}$$

$$\rightarrow 261 = 7 \times 35 + 16 \equiv 16 \pmod{35}$$

$$\rightarrow 144 \equiv 4 \times 35 + 4 \equiv 4 \pmod{35}$$

$$\rightarrow 36 \equiv 1 \pmod{35}$$

So: $9^6 \equiv 1 \pmod{35}$

Now: $9^{4824} = 9^{804 \times 6} \equiv 1 \pmod{35}$.

\therefore It is divisible by 35.

4. $2^{2^{2006}} \pmod{3} = ?$

Practice

$$2^{2^{2006}} = (2^2)^{2^{2005}} = 4^{2^{2005}} \equiv 1 \pmod{3}$$

5. Is $5^{30000} - 6^{123456}$ a multiple of 31?

31 is prime. And $5^{30000} = (5^{30})^{1000} \equiv 1 \pmod{31}$.

Compare with $6^{123456} = 6^{123450} \times 6^6$:

$$1 \times 6^6 \equiv 5^3 \equiv 125 \equiv 31 \times 4 + 1 \pmod{31} \equiv 1 \pmod{31}$$

\therefore It is a multiple of 31.

Show that if a has a multiplicative inverse modulo N , then this inverse is unique (modulo N).

Let's assume $a \in \{1, \dots, N-1\}$.

Suppose $b, c \in \{1, \dots, N-1\}$ are both multiplicative inverses of a modulo N . Then:

$$ab \equiv 1 \pmod{N}$$

$$ac \equiv 1 \pmod{N}$$

$$ab \equiv ac \pmod{N}$$

$$ab \cdot b \equiv ac \cdot b \pmod{N} \quad (1)$$

(1): Substitution Rule:

$$x \equiv x', y \equiv y' \pmod{N}$$

$$xy \equiv x'y' \pmod{N}$$

Then:

$$(ab) \cdot b \equiv (ab) \cdot c \pmod{N} \quad (2)$$

(2): Commutativity

$$1 \cdot b \equiv 1 \cdot c \pmod{N}$$

$$b \equiv c \pmod{N}$$

$$b = c$$

Practice

Suppose $p \equiv 3 \pmod{4}$. Show that $(p + 1)/4$ is an integer.

$$p \equiv 3 \pmod{4}$$

$$p = 4k + 3 \text{ for some } k \in \mathbb{Z}$$

So: $p + 1 = 4k + 4$, which is divisible by 4.

We say that x is a square root of y modulo a prime p if $y \equiv x^2 \pmod{p}$. Show that if (i) $p \equiv 3 \pmod{4}$ and (ii) y has a square root modulo p , then $y^{(p+1)/4}$ is such a square root.

Let x be the square root of y modulo p . Then: $y \equiv x^2 \pmod{p}$.

Write $p = 4k + 3$. Then, $\left(y^{\frac{p+1}{4}}\right)^2 = y^{2(p+1)/4} = y^{2(4k+3+1)/4} = y^{2k+2}$

Keep in mind: $(p + 1)/4 = k + 1$.

Try plugging in x in the last expression:

Is $y^{2k+2} = x^{4k+4} \equiv x^2$?

So, we're asking: Is $x^{4k+4} - x^2 \equiv 0 \pmod{p}$?

$$x^{4k+4} - x^2 = (x^{2k+2} - x)(x^{2k+2} + x)$$

So at least one of: $x^{2k+2} - x$ or $x^{2k+2} + x$ must be $\equiv 0 \pmod{p}$.