Advanced Algorithms

Floriano Zini

Free University of Bozen-Bolzano
Faculty of Computer Science

Academic Year 2013-2014

Lab 2 – Exercises on algorithms with numbers

Exercise 0.1 page 8 DPV

In each of the following situations, indicate whether $f = O(g)$, or $f = \Omega(g)$, or both, i.e., $f = \Theta(g)$. Justify your answer.

- Hint: you can
  - Apply the definitions of $O(.)$, $\Omega(.)$, $\Theta(.)$
  - Apply the simplifications introduced in the lectures
  - Plot the graphics of the functions (e.g., using Octave)

	$f(n)$	$g(n)$
(a)	$n-100$	$n-200$
(b)	$n^{1/2}$	$n^{2/3}$
(c)	$100n+\log n$	$n+(\log n)^2$
(d)	$n\log n$	$10n\log 10n$
(e)	$\log 2n$	$\log 3n$
(f)	$10\log n$	$\log(n^2)$
(g)	$n^{1.01}$	$n\log^2 n$
(h)	$n^2/\log n$	$n(\log n)^2$

Exercise 0.1 page 8 DPV

- **Solution**

a) $n-100=\Theta(n-200)$
b) $n^{1/2}=O(n^{2/3})$
c) $100n+\log n=\Theta(n+(\log n)^2)$
d) $n\log n=\Theta(10n\log 10n)$
e) $\log 2n=\Theta(\log 3n)$
f) $10\log n=\Theta(\log(n^2))$
g) $n^{1.01}=\Omega(\log^2 n)$
h) $n^2/\log n=\Omega(n(\log n)^2)$

Exercise

```
function fib1(n)
if n = 0:  return 0
if n = 1:  return 1
return fib1(n - 1) + fib1(n - 2)
```

- Implement in Octave the function *fib1,* which calculates Fib_n recursively. Modify the program such that it displays all the calculated Fibonacci numbers

Exercise

```
function fib1(n)
if n = 0:  return 0
if n = 1:  return 1
return fib1(n - 1) + fib1(n - 2)
```

- Implement in Octave the function *fib1,* which calculates Fib_n recursively. Modify the program such that it displays all the calculated Fibonacci numbers
- Solution

```
function f = fib1(n)
if (n == 0)
   f = 0;
   disp(sprintf("fib(%d)=%d",n,f));
elseif (n==1)
   f = 1;
   disp(sprintf("fib(%d)=%d",n,f));
else
   f=fib1(n-1)+fib1(n-2);
   disp(sprintf("fib(%d)=%d",n,f));
endif;
end
```

Exercise

```
function fib2(n)
if n = 0 return 0
create an array f[0...n]
f[0] = 0, f[1] = 1
for i = 2...n:
    f[i] = f[i - 1] + f[i - 2]
return f[n]
```

- Implement in Octave the function *fib2,* which calculates Fib_n iteratively

Exercise

```
function fib2(n)
if n = 0 return 0
create an array f[0...n]
f[0] = 0, f[1] = 1
for i = 2...n:
    f[i] = f[i - 1] + f[i - 2]
return f[n]
```

- Implement in Octave the function *fib2,* which calculates Fib_n iteratively
- Solution

```
function f = fib2(n)
f = ones (1, n);
for i = 3:n
   f(i) = f (i-1) + f (i-2);
endfor
```

Assignment 01

Exercise 0.1 page 8 DPV (cont.)

- In each of the following situations, indicate whether $f = O(g)$, or $f = \Omega(g)$, or both, i.e., $f = \Theta(g)$. Justify your answer.
- Hint: you can
  - Apply the definitions of $O(.)$, $\Omega(.)$, $\Theta(.)$
  - Apply the simplifications introduced in the lectures
  - Plot the graphics of the functions (e.g., using Octave)

(i)	$n^{0.1}$	$(\log n)^{10}$
(j)	$(\log n)^{\log n}$	$n/\log n$
(k)	$\sqrt{n}$	$(\log n)^3$
(l)	$n^{1/2}$	$5^{\log_2 n}$
(m)	$n2^n$	3^n

Assignment 01 (cont.)

Exercise

- With reference on exercise 0.4 on page 9 of DPV, implement in Octave a function *fib3* that calculates the n-th Fibonacci number using matrices

$$\begin{pmatrix} F_n \\ F_{n+1} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}^n \cdot \begin{pmatrix} F_0 \\ F_1 \end{pmatrix}.$$

Assignment 01 (cont.)

Exercise 1.7 page 39 DPV

- How long does the recursive multiplication algorithm take to multiply an *n*-bit number by an *m*-bit number? Justify your answer

```
function multiply(x, y)
Input: Two n-bit integers x and y, where y ≥ 0
Output: Their product

if y = 0: return 0
z = multiply(x, ⌊y/2⌋)
if y is even:
   return 2z
else:
   return x + 2z
```