Theoretical Computer Science (Bridging Course)

Dr. G. D. Tipaldi
F. Boniardi
Winter semester 2011/2012

University of Freiburg
Department of Computer Science

Exercise Sheet 8

Due: 8th February 2014

Exercise 8.1 (Runtime)

You have implemented an algorithm that needs exactely $f(n)$ steps to terminate, where n is the size of the input. Assume that on your machine each step takes $1\mu s$.
For which maximal input size does your algorithm terminate within *one* day? Which input size can it maximally process in 10 days? Answer these (two!) questions for the following runtimes:

(a) $f(n) = n$

(b) $f(n) = n^2$

(c) $f(n) = 2^n$

(d) $f(n) = n^2 + n$

(e) *(Extra, not mandatory)* $f(n) = n \log n$
Hint: to compute f^{-1}, you can use the bisection method.

Solution: It is trivial to see that, the number maximal number of steps our machine is able to perform in 1 day is

$$N_{max} := 10^6 \cdot 60 \cdot 60 \cdot 24 = 864 \cdot 10^8,$$

that is, the number of milliseconds in one day. So, set D to be the number of days we allocate for computations and provided that f is invertible[1], we have

$$n(D) := \lfloor f^{-1}(DN_{max}) \rfloor$$

So we have:

(a) $f^{-1}(DN_{max}) = ND_{max}$.

(b) $f^{-1}(DN_{max}) = \sqrt{DN_{max}}$.

(c) $f^{-1}(DN_{max}) = \log_2(DN_{max})$.

(d) $f^{-1}(DN_{max}) = \frac{1}{2} + \frac{1}{2}\sqrt{1 + 4DN_{max}}$.

$f(n)$	$n(1)$	$n(10)$
n	$864 \cdot 10^8$	$864 \cdot 10^9$
n^2	293938	929516
$n^2 + n$	293938	929515
2^n	36	39

Regarding part (e), the inverse of $f(n) = n \log n$ cannot be expressed in terms of elementary function, so it must be computed numerically, that is, by searching for an $n*$ so that

- $n^* \log n^* \leq ND_{max}$,
- $(n^* + 1) \log(n^* + 1) \geq ND_{max}$.

[1]Which is very likely to be, as we expect to be monotonically increasing.

$f(n)$	$n(1)$	$n(10)$
$n \log n$	3911758539	35563480335

Exercise 8.2 (Big-O)

Consider the Turing machine below. The input alphabet is $\Sigma = \mathbb{N} = \{1, 2, 3, \dots\}$. The operator $|w|$ denotes the length of the string w, the relation $<$ is the smaller relation on the natural numbers.

M = "On input string w":
for $i = 1$ to $|w|$
    for $j = |w|$ downto $i + 1$
        if $w_j < w_{j-1}$
            swap w_j and w_{j-1}
        endif
    endfor
endfor

Assume that the runtime of a swap and of a comparison of two natural numbers is constant.

(a) What is the smallest exponent $k \in \mathbb{R}$ so that the runtime of the Turing machine M is in $O(|w|^k)$? Justify your answer.

Solution:

The runtime of the TM M is in $O(|w|^2)$ but not in $O(|w|)$. Indeed, set $n := |w|$, the outermost loop is executed exactly n-times while the innermost is executed $(n-1)$-times on the first iteration, $(n-2)$-times on the second, down to 0-times on the last iteration of the outermost loop. As a consequence, the number of operations $\phi(n)$ are exactly

$$\phi(n) = \frac{n(n-1)}{2} + \frac{n(n-1)}{2},$$

where the first bit is due to the fact that a comparison $(w_j < w_{j-1})$ whilst the second one is related to the swap performed in each iteration of the innermost loop. So the time complexity is then given by

$$f(n) = C_{comp}\frac{n(n-1)}{2} + C_{swap}\frac{n(n-1)}{2} = (C_{comp} + C_{swap})g(n).$$

Here C_{comp} denote C_{swap} are respectively the runtime of comparing two integers and swapping them in the string, and $g(n) := \frac{n(n-1)}{2}$. As a consequence, for instance we have

$$f(n) \leq 2M\frac{1}{2}n^2 = Mn^2,$$

where $M := \max\{C_{comp}, C_{swap}\}$. That is $f \in O(n^2)$. It is obvious that such exponent is the smallest one.

(b) What does M compute (i.e. what is written on the tape when M halts)?

Solution: The TM sorts a sequence of integers according to the $<$ relation (this sorting algorithm is called *bubble sort*). To see this, call w^i the word after the i-th iteration of the outermost loop. It is easy to see that

$$w_i^i = \min\{w_j \mid w_j \in w_{i:n}^i\},$$

so $w_j^n \leq w_{j+1}^n$ for all $j = 1, ..., n-1$.

Exercise 8.3 (Big-O)

Characterise the relationship between $f(n)$ and $g(n)$ in the following examples using the O, Θ or Ω-notation.

1. $f(n) = n^{0.99998}$ $\qquad g(n) = \sqrt{n}$
2. $f(n) = 2^{\log^2(n)}$ $\qquad g(n) = \sum_{k=1}^{n^2} \frac{n}{2^k}$
3. $f(n) = n \cdot log_2 n$ $\qquad g(n) = \sqrt[3]{n}$
4. $f(n) = \sqrt{n}$ $\qquad g(n) = 1000n$
5. *(Extra, not mandatory)* $f(n) = \frac{n^{n+1}}{(n+1)^n}$, $g(n) = \sqrt[n]{n!}$

   Hint: Stirling's approximation could be useful here.

Solution: by definition:

- $f \in O(g)$ if $\exists$ constants $C > 0$ and $n_0 \in \mathbb{N}$ such that $f(n) \leq C \cdot g(n)$, for all $n \geq n_0$.
- $f \in \Omega(g)$ if $\exists$ constants $c > 0$ and $n_0 \in \mathbb{N}$ such that $f(n) \geq c \cdot g(n)$, for all $n \geq n_0$.
- $f \in \Theta(g)$ if $f \in O(g)$ and $f \in \Omega(g)$.

Note that

- $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$, then $\exists n_0 \in \mathbb{N} : f(n) < g(n)\ \forall n \geq n_0$, that is $f \in O(g)$. Furthermore $f \notin \Omega(g)$.
- $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$, then $\exists n_0 \in \mathbb{N} :\ f(n) > g(n)\ \forall n \geq n_0$, that is $f \in \Omega(g)$. Furthermore $f \notin O(g)$.
- $\lim_{n\to\infty} \frac{f(n)}{g(n)} = l \in (0, \infty)$, then $\forall \epsilon > 0\ \exists n_0(\epsilon) \in \mathbb{N} :\ (l - \epsilon)g(n) \leq f(n) \leq (l + \epsilon)g(n)$ $\forall n \geq n_0(\epsilon)$, that is $f \in \Theta(g)$.

Therefore, we have:

1. $f(n) = n^{0.99998}$ and $g(n) = \sqrt{n} = n^{0.5}$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{n^{0.99998}}{n^{0.5}} = \lim_{n\to\infty} n^{0.499998} = \infty \implies f \in \Omega(g)$$

2. $f(n) = 2^{\log_a^2(n)}$ and $g(n) = \sum_{k=1}^{n^2} \frac{n}{2^k}$

   Since the exercise does not specify the base of the logarithm, we denote it by a. Then,

$$f(n) = 2^{\log_a^2 n} = 2^{(\log_a n)(\log_a n)} = 2^{\frac{\log_2 n}{\log_2 a}\frac{\log_2 n}{\log_2 a}} = \left(\left(2^{\log_2 n}\right)^{\log_2 n}\right)^{\frac{1}{(\log_2 a)(\log_2 a)}} = n^{b \log_2 n}$$

   where $b = \frac{1}{(\log_2 a)(\log_2 a)}$ is a constant.

   On the other hand, since

$$g(n) = \frac{n}{2^1} + \frac{n}{2^2} + \frac{n}{2^3} + \ldots + \frac{n}{2^{n^2}}$$

   then

$$\frac{1}{2}g(n) = \frac{1}{2}\left(\frac{n}{2^1} + \frac{n}{2^2} + \frac{n}{2^3} + \ldots + \frac{n}{2^{n^2-1}} + \frac{n}{2^{n^2}}\right) = \frac{n}{2^2} + \frac{n}{2^3} + \frac{n}{2^4} + \ldots + \frac{n}{2^{n^2}} + \frac{n}{2^{n^2+1}}$$

   Therefore,

$$g(n) - \frac{1}{2}g(n) = \frac{n}{2^1} - \frac{n}{2^{n^2+1}} \implies \frac{1}{2}g(n) = \frac{n}{2^1} - \frac{n}{2^{n^2+1}} \implies g(n) = n\left(1 - \frac{1}{2^{n^2}}\right)$$

   As a result,

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{n^{b \log_2 n}}{n\left(1 - \frac{1}{2^{n^2}}\right)} = \lim_{n\to\infty} \frac{n^{(b \log_2 n) - 1}}{1 - \frac{1}{2^{n^2}}} = \infty \implies f \in \Omega(g)$$

3. $f(n) = n \cdot \log_2 n$ and $g(n) = \sqrt[3]{n} = n^{\frac{1}{3}}$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{n \log_2 n}{n^{\frac{1}{3}}} = \lim_{n\to\infty} n^{\frac{2}{3}} \log_2 n = \infty \implies f \in \Omega(g)$$

4. $f(n) = \sqrt{n}$ and $g(n) = 1000n$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{\sqrt{n}}{1000n} = \lim_{n\to\infty} \frac{1}{1000 \cdot \sqrt{n}} = 0 \implies f(n) \in O(g(n))$$

5. $f(n) = \frac{n^{n+1}}{(n+1)^n}$ and $g(n) = \sqrt[n]{n!}$.

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{(1+\frac{1}{n})^{-n} n}{\sqrt[n]{n}} = \lim_{n\to\infty} \frac{e^{-1} n}{\sqrt[2n]{2\pi n} e^{-1} n} = \lim_{n\to\infty} \frac{1}{\sqrt[n]{\pi n}} = 1 \implies f \in \Theta(g).$$

where the second equality is obtained by dint of the Stirling's formula and the definition of the constant e:

$$\lim_{n\to\infty} \frac{\sqrt{2\pi n}\left(\frac{n}{e}\right)^n}{n!} = 1, \lim_{n\to\infty} \left(1 + \frac{1}{n}\right)^n = e$$

and the last equality follows from

$$\lim_{n\to\infty} \sqrt[n]{\pi n} = \lim_{n\to\infty} e^{\frac{1}{n}\log n + \frac{1}{n}\log \pi} = e^0 = 1.$$