**CS CS2223 - D Term 2008: Algorithms** Prof. Carolina Ruiz

# Homework 2

*WPI* *By Can Ozmen and Prof. Ruiz*

**Problem 1**: (32 Points) Suppose you have algorithms with the running times listed below. Assume these are the exact number of operations performed as a function of the input size n. Suppose you have a computer that can perform $10^{10}$ operations per second, and you need to compute results in at most

- (8 Points) 1 hour
- (8 Points) 1 week
- (8 Points) 1 year
- (8 Points) 1 century

For each of the algorithms, what is the largest input size n for which you would be able to get the result within the given time limit? That is, your solutions to this exercise should contain a table like the one below together with explanations for each of the values you provide in your table.

**Solution 1**: The following are the number of operations per given time:

$$10^{10} ops/second \times 60 * 60 \; seconds = 3.60 \times 10^{13} ops \; in \; 1 \; hour$$
$$10^{10} ops/second \times 60 * 60 * 24 * 7 \; seconds = 6.05 \times 10^{15} ops \; in \; 1 \; week$$
$$10^{10} ops/second \times 60 * 60 * 24 * 365 \; seconds = 3.15 \times 10^{17} ops \; in \; 1 \; year$$
$$10^{10} ops/second \times 60 * 60 * 24 * 365 * 100 \; seconds = 3.15 \times 10^{19} ops \; in \; 1 \; century$$

These numbers were calculated as follows. Let $t$ be the number of operations that can be performed within the time limit.

Table 1: Largest input size $n$ versus running time

| Complexity | 1 hour | 1 week | 1 year | 1 century |
|---|---|---|---|---|
| $1000n^3$ | $3.30\times10^3$ | $1.82\times10^4$ | $6.80\times10^4$ | $3.16\times10^5$ |
| $n^4$ | $2.45\times10^3$ | $8.82\times10^3$ | $2.37\times10^4$ | $7.49\times10^4$ |
| $n^2 \log_{10} n$ | $2.38\times10^6$ | $2.85\times10^7$ | $1.95\times10^8$ | $1.84\times10^9$ |
| $3^n$ | 28 | 33 | 36 | 40 |
| $\sqrt{n}$ | $1.30\times10^{27}$ | $3.66\times10^{31}$ | $9.92\times10^{34}$ | $9.92\times10^{38}$ |

1. If $1000n^3 = t$ then $n = \sqrt[3]{t/1000}$.

   For a time limit of 1 hour, t = $3.60\times10^{13}$ ops. Hence $n = \sqrt[3]{3.60 \times 10^{13}/1000} = \sqrt[3]{3.60 \times 10^{10}} = 3.30 \times 10^3$. Similarly for the other time limits.

2. If $n^4 = t$ then $n = \sqrt[4]{t}$.

   For a time limit of 1 hour, t = $3.60\times10^{13}$ ops. Hence $n = \sqrt[4]{3.60 \times 10^{13}} = 2.45 \times 10^3$. Similarly for the other time limits.

3. If $n^2 \log_{10} n = t$ then this equation cannot be solved algebraically for $n$. Hence, we need to use an approximation method to find an $n$ that satisfies the fixpoint equation: $n = \sqrt{t/(\log_{10} n)}$.

   For a time limit of 1 hour, t = $3.60 \times 10^{13}$ ops. Hence we need to look for an $n$ that satisfies $n = \sqrt{t/(\log_{10} n)}$ Such $n$ is approximately $2.38 \times 10^6$. Similarly for the other time limits.

4. If $3^n = t$ then $n = \log_3 t$.

   For a time limit of 1 hour, t = $3.60 \times 10^{13}$ ops. Hence $n = \log_3 3.60 \times 10^{13} = 28$. Similarly for the other time limits.

5. If $\sqrt{n} = t$ then $n = t^2$.

   For a time limit of 1 hour, t = $3.60 \times 10^{13}$ ops. Hence $n = 3.60 \times 10^{13^2} = 1.30 \times 10^{27}$. Similarly for the other time limits.

Since we know that $\sqrt{n} = O(n^2 \log_{10} n), n^2 \log_{10} n = O(n^3), 1000n^3 = O(n^4), n^4 = O(3^n)$ we can reorganize the rows in the previous table to make explicit the increase in time complexity and hence the reduction in the size of the input allowed in the time limit:

Table 2: Largest input size $n$ versus running time, sorted by time complexity

| Complexity | 1 hour | 1 week | 1 year | 1 century |
|---|---|---|---|---|
| $\sqrt{n}$ | $1.30 \times 10^{27}$ | $3.66 \times 10^{31}$ | $9.92 \times 10^{34}$ | $9.92 \times 10^{38}$ |
| $n^2 \log_{10} n$ | $2.38 \times 10^6$ | $2.85 \times 10^7$ | $1.95 \times 10^8$ | $1.84 \times 10^9$ |
| $1000n^3$ | $3.30 \times 10^3$ | $1.82 \times 10^4$ | $6.80 \times 10^4$ | $3.16 \times 10^5$ |
| $n^4$ | $2.45 \times 10^3$ | $8.82 \times 10^3$ | $2.37 \times 10^4$ | $7.49 \times 10^4$ |
| $3^n$ | 28 | 33 | 36 | 40 |

**Problem 2**: (15 Points) Assume that $k$ is a fixed constant. Prove that if $f(n) = O(n)$ then $f(n)^k = O(n^k)$.

**Solution 2**:
**Proof:**

$$f(n) = O(n)$$

which by definition means that: $\exists c_0 > 0 \; \exists n_0 \; \forall n \geq n_0 \; f(n) \leq c_0 n$

for $k \geq 0$ elevating to the $k$–power preserves the inequality: $\exists c_0 > 0 \; \exists n_0 \; \forall n \geq n_0 \; f(n)^k \leq (c_0 n)^k$

$$\exists c_0 > 0 \; \exists n_0 \; \forall n \geq n_0 \; f(n)^k \leq (c_0)^k n^k$$

$$\exists c_1 (= (c_0)^k) > 0 \; \exists n_0 \; \forall n \geq n_0 \; f(n)^k \leq c_1 n^k$$

which by definition is: $f(n)^k = O(n^k)$

■

**Problem 3**: (38 Points) Problem 3 In each of the following cases, determine whether $f(n) = O(g(n))$, or $f(n) = \Omega(g(n))$, or both (i.e., $f(n) = \Theta(g(n))$), or none of the above. Prove your answers decisively.

**Solution 3**:

1

$$f(n) = 3n^2 + 10n + 8,\ g(n) = n^2$$
$$\lim_{n\to+\infty} \frac{f(n)}{g(n)} = \lim_{n\to+\infty} \frac{3n^2 + 10n + 8}{n^2} = 3$$
Hence, $f(n) = \Theta(g(n))$

2

$$f(n) = 2^n,\ g(n) = 2^{n+7}$$
$$\lim_{n\to+\infty} \frac{f(n)}{g(n)} = \lim_{n\to+\infty} \frac{2^n}{2^{n+7}} = 2^{-7}$$
Hence, $f(n) = \Theta(g(n))$

3

$$f(n) = n(\log n)^2,\ g(n) = n^3$$
$$\lim_{n\to+\infty} \frac{f(n)}{g(n)} = \lim_{n\to+\infty} \frac{n(\log n)^2}{n^3} = \lim_{n\to+\infty} \frac{(\log n)^2}{n^2} = 0$$
Hence, $f(n) = O(g(n))$

4

$$f(n) = 3^n,\ g(n) = 2^n$$
$$\lim_{n\to+\infty} \frac{f(n)}{g(n)} = \lim_{n\to+\infty} \frac{3^n}{2^n} = \lim_{n\to+\infty} (1.5)^n = +\infty$$
Hence, $f(n) = \Omega(g(n))$

5

$$f(n) = \sqrt{n},\ g(n) = n$$
$$\lim_{n\to+\infty} \frac{f(n)}{g(n)} = \lim_{n\to+\infty} \frac{\sqrt{n}}{n} = \lim_{n\to+\infty} \frac{\sqrt{n}}{\sqrt{n}\sqrt{n}} = \lim_{n\to+\infty} \frac{1}{\sqrt{n}} = 0$$
Hence, $f(n) = O(g(n))$

6

$$f(n) = (2n)^{\log_2 2n},\ g(n) = n^{\log_2 n}$$
$$\lim_{n\to+\infty} \frac{f(n)}{g(n)} = \lim_{n\to+\infty} \frac{(2n)^{\log_2 2n}}{n^{\log_2 n}}$$
$$= \lim_{n\to+\infty} \frac{(2n)^{\log_2 2+\log_2 n}}{n^{\log_2 n}}$$
$$= \lim_{n\to+\infty} \frac{(2n)^{1+\log_2 n}}{n^{\log_2 n}}$$
$$= \lim_{n\to+\infty} \frac{2n * (2n)^{\log_2 n}}{n^{\log_2 n}}$$
$$= \lim_{n\to+\infty} \frac{2n * 2^{\log_2 n} * n^{\log_2 n}}{n^{\log_2 n}}$$
$$= \lim_{n\to+\infty} 2n * 2^{\log_2 n} = \lim_{n\to+\infty} 2n * n == \lim_{n\to+\infty} 2n^2 = +\infty$$
Hence, $f(n) = \Omega(g(n))$

**Problem 4**: (15 Points) Let $k$ be a fixed constant, and let $f_1, f_2, ..., f_k$ be functions such that $f_i = O(f_{i+1})$ for all $i, 1 \leq i < k$. Prove that $f_1 + f_2 + ... + f_k = \Theta(f_k)$.

**Solution 4**: **Proof:**

**Possible solution 1:**

Since $f_1 = O(f_2),\ f_2 = O(f_3),\ f_3 = O(f_4),\ \ldots, f_{k-1} = O(f_k)$, after several applications of the Transitivity property of $O$ (see Fact (2.2)(a) on page 38 of the textbook), then $f_1 = O(f_3),\ f_1 = O(f_4),\ \ldots, f_1 = O(f_k)$. Similarly for every $1 \leq i < k$ after several applications of the Transitivity property of $O$: $f_i = O(f_{i+2}),\ f_i = O(f_{i+3}),\ \ldots, f_i = O(f_k)$. Hence for all $i$, $1 \leq i < k$: $f_i = O(f_k)$.

Now using Fact (2.5) on page 39 of the textbook, we have that $f_1 + f_2 + f_3 + \ldots + f_{k-1} + f_k = O(f_k)$. (Study the proof of this Fact in the textbook.)

On the other hand, since we assume that for all $i$, $1 \leq i < k$: $f_i$ is a positive function, that is $f_i(n) > 0$ for all $n$, then:

$f_1 + f_2 + f_3 + \ldots + f_{k-1} + f_k \geq f_k$ and so $f_1 + f_2 + f_3 + \ldots + f_{k-1} + f_k = \Omega(f_k)$, with constant $c = 1$ and $n_0 = 0$.

Since we just proved that $f_1+f_2+f_3+\ldots+f_{k-1}+f_k = O(f_k)$ and $f_1+f_2+f_3+\ldots+f_{k-1}+f_k = \Omega(f_k)$, then $f_1 + f_2 + f_3 + \ldots + f_{k-1} + f_k = \Theta(f_k)$. ■

**Solution 4**: **Proof:**

**Possible solution 2:** By induction on k

Basis: $k = 2$

$$f_1 = O(f_2)$$
$$f_1 + f_2 = O(f_2)$$

Inductive Hypothesis: $k = n$

Assume $f_1 + f_2 + \cdots + f_n = \Theta(f_n)$

Inductive Step: $k = n + 1$

$$\begin{aligned} f_1 + f_2 + \cdots + f_n + f_{n+1} &\leq c_0 f_n + f_{n+1} \\ &\leq c_0(c_1 f_{n+1}) + f_{n+1} \\ &\leq (c_0 c_1 + 1) f_{n+1} \\ f_1 + f_2 + \cdots + f_n + f_{n+1} &= O(f_{n+1}) \end{aligned}$$

$$f_1 + f_2 + \cdots + f_n + f_{n+1} \geq f_{n+1}$$
$$f_1 + f_2 + \cdots + f_n + f_{n+1} = \Omega(f_{n+1})$$
$$f_1 + f_2 + \cdots + f_n + f_{n+1} = \Theta(f_{n+1})$$

■