Solution to Exercise R-1.7, Page 47

Sept 5, 2001

R-1.7 For each function $f(n)$ and time t in the following table, determine the largest size n of a problem that can be solved in time t assuming that the algorithm to solve the problem takes $f(n)$ microseconds. Recall that $\log n$ denotes the logarithm in base 2 of n.

Answer First recall that a microsecond is 10^{-6} seconds. Hence, one second $= 10^6$ microseconds,
one hour $= 3600000000 = 3.6 \cdot 10^9$ microseconds,
one month (assume a month has 30 days) $= 2592000000000 = 2.592 \cdot 10^{12}$ microseconds, and
one century $= 3110400000000000 = 3.1104 \cdot 10^{15}$ microseconds.

Now, if we have an algorithm that runs in $f(n)$ steps given an input of size n and, for example, $f(899) \leq 10^6$ that means that inputs of size 899 can be processed in less than one second. This problem asks us to find the input of largest size that can be processed in one second, one hour, etc given different running times $f(n)$.

That is, to determine the largest problem that can be done in one second, for example, we have to determine the largest n such that $f(n) \leq 1000000$.

Row 1: $f(n) = \log n$ In this case, we need to determine the largest n such that $\log n \leq 1000000$. To solve this inequality, we need to rewrite the inequality as $2^{logn} \leq 2^{1000000}$ or $n \leq 2^{1000000}$. Recall from lecture that $2^{10} \approx 10^3$, thus we have that $2^{1000000} = 2^{10 \cdot 100000} = (2^{10})^{100000} \approx (10^3)^{100000} = 10^{300000}$. This is the result given in the textbook.

Similarly for one hour, we must have that $n \leq 2^{3600 \cdot 1000000}$ and thus $n \approx 10^{1080000000}$.

Row 8: $f(n) = n!$ To see that 12 is the largest sized input that can be processed within an hour when $f(n) = n!$, one can simply, compute 12! (using Maple for example) and verify that it is less than the number of microseconds in one hour, but that 13! is greater than the number of microseconds in an hour.

Row 4: $f(n) = n \log n$ In this case, use Maple to solve equations like $n \log n - 1000000 = 0$. The Maple command for solving this equation is

```
fsolve(n*log[2](n) - 1000000 = 0);
```

Without Maple or an equivalent tool, it would be very difficult to solve the equation above by hand. It would be easier to write a program that implements Newton's method to approximate the roots, which is probably what Maple does.

$f(n)$	1 Sec $= 10^6$ms	1 Hr $= 3.6 \cdot 10^9$ms	1 Mo $= 2.592 \cdot 10^{12}$ms	1 Cnt $= 3.1104 \cdot 10^{15}$ms
$\log n$	$\approx 10^{300000}$	$\approx 10^{1080000000}$	$\approx 10^{7.776 \cdot 10^{11}}$	$\approx 10^{9.3312 \cdot 10^{14}}$
$\sqrt{n}$	10^{12}	$12.96 \cdot 10^{18}$	$6.718464 \cdot 10^{24}$	$9.67458816 \cdot 10^{30}$
n	10^6			
$n \log n$	62746	$1.333780589 \cdot 10^8$	$7.187085640 \cdot 10^{10}$	$6.769949846 \cdot 10^{13}$
n^2	10^3	$6.0 \cdot 10^4$	$1.6100 \cdot 10^6$	$5.5771 \cdot 10^7$
n^3	10^2	$1.533 \cdot 10^3$	$1.374 \cdot 10^4$	$1.460 \cdot 10^5$
2^n	19	31	41	51
$n!$	9	12	15	17

BFC

Last updated Sept 14, 2001.