Solutions to Homework Set 8

Special problem 1

Let t^* be the smallest t such that the entries of M_G^t are all 1's. Our task is to find t^* in time $O(s(n)\log n)$.

Proceeding as in Handout Number 22, we find k such that $2^k < t^* \leq 2^{k+1}$; this takes time $O(s(n)\log n)$. We have now effectively sandwiched t^* in an interval (of size $2^{k+1} - 2^k = 2^k$). We shall now show how to determine t^* in a further $O(s(n)\log n)$ time.

Now we use "binary search" to locate t^* within this interval – this means we keep halving the size of the sandwiching interval. Clearly the number of halvings we can do till we reach an interval of size 1 (and thereby pinpoint t^*) is k. Moreover, since $M_G^{2^k}$ is not all 1's, it follows that $2^k < n$. In other words $k = O(\log n)$.

Therefore, we need only $O(\log n)$ halvings of the interval. It remains to show that each halving can be done in time $O(s(n))$. Note that the size of the interval is always a power of 2. Thus, if the current interval is $i < t^* \leq j$, to perform a halving we need to compute $M_G^{(i+j)/2} = M_G^i M_G^{(j-i)/2}$. Since the size $(j-i)$ of the current interval is a power of 2, so is $(j-i)/2$. Moreover $(j-i)/2 \leq 2^k/2 = 2^{k-1}$ and so $M_G^{(j-i)/2}$ has already been computed. Therefore, we need just one matrix multiplication to do the halving and that can be done in time $O(s(n))$. □

Special problem 2

We will simply calculate the closed-form expression for the given recurrence relation:

$$f(1) = 1, \qquad f(n) = cf(n/2) + n^b \quad \text{for } n = 2, 2^3, 2^4, \ldots$$

First, we translate it into a more manageable form by taking $f^*(n) = f(2^n)$.

$$f^*(0) = 1, \qquad f^*(n) = cf^*(n-1) + 2^{bn}, \quad (n \geq 1)$$

This recurrence relation we can solve using standard techniques. First, we take the homogeneous part, $f^*(n) - cf^*(n-1) = 0$ and solve it, getting αc^n. The form of the recurrence suggests the particular solution $f^*(n) = \beta 2^{bn}$; plugging this into the recurrence relation and solving for β yields a particular solution of $\frac{2^b}{2^b-c}2^{bn}$. Thus, the full solution to the recurrence relation is of the form $\alpha c^n + \frac{2^b}{2^b-c}2^{bn}$. Plugging in initial conditions and solving for α yields a final solution of $f^*(n) = \frac{-c}{2^b-c}c^n + \frac{2^b}{2^b-c}2^{bn}$. Thus,

$$f(n) = \frac{c \cdot c^{\log_2 n} - 2^b n^b}{c - 2^b}$$

Now $c^{\log_2 n} = n^{\log_2 c} > n^b$ and so $f(n) = \Theta(c^{\log_2 n})$. □

Special problem 3

Part (a)

We are given

$$T(0) = 1, \qquad T(1) = 1, \qquad T(n) = 2T(n-1) + 7T(n-2) + n \quad (n \geq 2)$$

Define a new sequence $S(n)$ as follows:

$$S(0) = 1, \qquad S(1) = 1, \qquad S(n) = 2S(n-1) + 7S(n-2) \quad (n \geq 2)$$

We know how to find $S(n)$ exactly: using our methods we obtain $S(n) = c_1(1+\sqrt{2})^n + c_2(1-\sqrt{2})^n$ for some constants c_1 and c_2. Now $|1+\sqrt{2}| > |1-\sqrt{2}|$ and so, without even determining the constants c_1 and c_2 we know that $S(n) = \Theta((1+\sqrt{2})^n)$.[1]

Now, we guess that there are some constants c and d such that $T(n) \le cS(n) - dn$ for all integers n. Using the method outlined in Handout Number 22 we easily obtain $d = 1$ and $c = 2$. On the other hand, it is trivial to see that $T(n) \ge S(n)$. In other words, we obtain

$$S(n) \le T(n) \le 2S(n) - n$$

and so $T(n) = \Theta(S(n)) = \Theta((1+\sqrt{2})^n)$.

Part (b)

This time we are given

$$T(0) = 1, \qquad T(1) = 1, \qquad T(n) = 2T(n-1) + 7T(n-2) + 2^{\sqrt{n}} \quad (n \ge 2)$$

Let's define $S(n)$ as in part (a) and another sequence $U(n)$ as follows:

$$U(0) = 1, \qquad U(1) = 1, \qquad U(n) = 2U(n-1) + 7U(n-2) + 2^n \quad (n \ge 2)$$

It's clear that $S(n) \le T(n) \le U(n)$. Now we shall try and find an upper bound on $U(n)$.

Let's try and solve the recurrence for $U(n)$. We know that $U(n)$ has the same homogeneous recurrence as $S(n)$. Therefore, it only remains to find a perticular solution. The form of the recurrence suggests the solution $U(n) = \gamma 2^n$. Plugging this into the recurrence for $U(n)$, we find $\gamma = -\frac{4}{7}$. Therefore, the general solution for $U(n)$ looks like this:

$$U(n) = \alpha(1+\sqrt{2})^n + \beta(1-\sqrt{2})^n + \gamma 2^n$$

where α, β and γ are some constants (we just determined γ, we can determine the other two by using $U(0) = U(1) = 1$).

Of the three numbers $1+\sqrt{2}$, $1-\sqrt{2}$ and 2 the largest is $1+\sqrt{2}$; therefore the above equation yields $U(n) = \Theta((1+\sqrt{2})^n)$. Going back to what we know about $T(n)$ we find that we've just proved $T(n) = \Theta((1+\sqrt{2})^n)$. □

Special problem 4

First we note that the algorithm described is a single pass of the *bubblesort* algorithm. This moves from left to right through an array, carrying with it (by means of exchanging neighboring elements of the array) the largest value seen to date. The only guaranteed result of a single pass is that the largest value will end up in the rightmost slot. However, $n-1$ passes guarantee that the array is entirely sorted.

The question in effect asks what is the expected number of exchanges that take place in one pass of the *bubblesort* algorithm, for an input array of size n. Let T_n be this number.

Consider an input array of size $n+1$. After the last-but-one iteration of the loop in the given algorithm, the largest value amongst $A[1], A[2], \ldots, A[n]$ resides in location $A[n]$. The probability that an exchange *will not occur* on the last iteration of the loop is, therefore, equal to the probability that $A[n+1]$ was the largest of the values to begin with. Since each of the $(n+1)$ cells in the array had equal chance of holding this largest value, this probability equals $1/(n+1)$.

Thus, with probability $1/(n+1)$ we'll perform only T_n exchanges and with probability $n/(n+1)$ we'll perform $T_n + 1$ exchanges. In other words, the expected number of exchanges will be:

$$T_{n+1} = \frac{1}{n+1}T_n + \frac{n}{n+1}(T_n + 1) = T_n + \frac{n}{n+1}.$$

[1] Actually we're cheating a bit here: we do need to show that $c_1 \ne 0$.

Clearly $T_1 = 0 = 1 - H_1$. Therefore, from the above recurrence, we have

$$T_n = \frac{1}{2} + \frac{2}{3} + \cdots + \frac{n-1}{n} = n - H_n.$$

for $n \geq 2$. $\square$

Special problem 5

We know that $p(n) = (\frac{1}{2})^{n-1}\binom{n-1}{\frac{n-1}{2}}$ for odd n. To simplify notation, let us set $N = (n-1)/2$, so that

$$p(n) = \frac{1}{4^N}\binom{2N}{N} = q(N) \quad \text{(say)}$$

Using Stirling's formula (with relative error $O(1/n)$) we have

$$k! = \sqrt{2\pi k}\left(\frac{k}{e}\right)^k (1 + O(1/k))$$

We simply plug in $k = N$ and $k = 2N$, use $\binom{2N}{N} = \frac{(2N)!}{N!N!}$ and simplify to obtain:

$$q(N) = \frac{1 + O(1/n)}{\sqrt{\pi N}(1 + O(1/n))^2}$$

Using the power series for $1/(1-x)$ we obtain $1/(1+O(1/n)) = 1+O(1/n)$. Also, since $O(1/n)^k = O(1/n)$ for $k > 1$, it is easy to see that $(1 + O(1/n))^3 = 1 + O(1/n)$. Therefore we obtain $q(N) = \frac{1}{\sqrt{\pi N}}(1 + O(1/n))$.

Stirling's formula also tells us that

$$k! = \sqrt{2\pi k}\left(\frac{k}{e}\right)^k (1 + \Omega(1/k))$$

And a similar argument shows that $q(N) = \frac{1}{\sqrt{\pi N}}(1 + \Omega(1/n))$ as well. This time, we must argue $1/(1 + \Omega(1/n)) = 1 + \Omega(1/n)$ and that goes through because $\Omega(1/n)^k = \Omega(0)$.

Therefore, we need to take

$$g(n) = 1/\sqrt{\pi N} = \sqrt{\frac{2}{\pi(n-1)}}$$

to have $1 - \Omega(1/n) < p(n)/g(n) < 1 + O(1/n)$.

For part (b) we need to use a more accurate approximation for the factorial; simply take one more term from Stirling's formula:

$$k! = \sqrt{2\pi k}\left(\frac{k}{e}\right)^k (1 + \frac{1}{12n} + O(1/n^2))$$

and use the same techniques as above. $\square$

Special problem 6

Using the Euler Summation Formula:

$$f(n) = \sum_{i=1}^{n} i^{\frac{8}{3}} = \int_1^{n+1} x^{8/3}dx + B_1(x^{8/3})\Big|_1^{n+1} + \frac{B_2}{2}(\frac{8}{3}x^{5/3})\Big|_1^{n+1} +$$

$$+ \frac{B_3}{6}(\frac{40}{9}x^{2/3})\Big|_1^{n+1} + \frac{B_4}{24}(\frac{80}{27}x^{-1/3})\Big|_1^{n+1} + R_4$$

If $g(x) = x^{8/3}$, then for $m \geq 4$ there exists a constant c such that $g^{(m)}(x) = O(x^{c-m})$ and $O(x^{c-m+1}) = O(1)$ (in this case c happens to be equal to $\frac{8}{3}$). This means that the remainders from R_4 onwards have small tails and so:

$$f(n) = \frac{3}{11}(n+1)^{11/3} - \frac{3}{11} - \frac{1}{2}(n+1)^{8/3} + \frac{1}{2} + \frac{2}{9}(n+1)^{5/3} - \frac{2}{9} - \frac{1}{243}(n+1)^{-1/3} + \frac{1}{243} + O(1).$$

which can be simplified to:

$$f(n) = \frac{3}{11}(n+1)^{11/3} - \frac{1}{2}(n+1)^{8/3} + \frac{2}{9}(n+1)^{5/3} + O(1).$$

□

Comments? Corrections? Suggestions? Mail to `amitc@cs.princeton.edu`