

CSCI303 Fall 2006 Homework 2 Solutions

4.3-2) last updated fall 2006, Yuriy Brun, ybrun@usc.edu

Algorithm A: $T_A^{WC}(n) = \Theta(n^{\lg 7})$

Algorithm A': $T_{A'}^{WC}(n) = aT_{A'}^{WC}\left(\frac{n}{4}\right) + n^2$

So for all $a > 16$, $T_{A'}^{WC}(n) = \Theta(n^{\log_4 a})$ because case 1 of the Master Theorem will apply. Solve for a such that the two algorithms run at the same speed.

$\log_2 7 = \log_4 a = \frac{\log_2 a}{2} = \log_2 \sqrt{a} \Rightarrow a = 49$. So for all $a < 49$, A' is asymptotically faster than A. Thus the largest integer value of a for which that is true is 48.

4-1) last updated fall 2006, Yuriy Brun, ybrun@usc.edu

Use Masters Theorem to solve (a –f)

a) case 3: $T(n) = \Theta(n^3)$

b) case 3: $T(n) = \Theta(n)$

c) case 2: $T(n) = \Theta(n^2 \lg n)$

d) case 3: $T(n) = \Theta(n^2)$

e) case 1: $T(n) = \Theta(n^{\lg 7})$

f) case 2: $T(n) = \Theta(\sqrt{n} \lg n)$

g) $T(n) = n + T(n-1) = n + (n-1) + T(n-2) = n + (n-1) + (n-2) + \dots + 1 =$

$$\sum_{i=1}^n i = \frac{n(n-1)}{2} = \Theta(n^2).$$

h) Method of changing variables:

Let $m = \lg n$, and $T(2^m) = S(m)$

$$S(m) = S\left(\frac{m}{2}\right) + 1$$

Using Master Theorem, case 2, $S(m) = \Theta(\lg m)$, thus $T(n) = \Theta(\lg \lg n)$.

4-4) last updated fall 2006, Yuriy Brun, ybrun@usc.edu

a) case 1: $T(n) = \Theta(n^{\lg 3})$

b) Master Theorem fails. We use iteration:

$$T(n) = 5T\left(\frac{n}{5}\right) + \frac{n}{\lg n} = 25T\left(\frac{n}{25}\right) + \frac{n}{\lg \frac{n}{5}} + \frac{n}{\lg n} =$$

$$5^k T\left(\frac{n}{5^k}\right) + \frac{n}{\lg \frac{n}{5^{k-1}}} + \frac{n}{\lg \frac{n}{5^{k-2}}} + \dots + \frac{n}{\lg n} =$$

$$nT(1) + \frac{n}{\lg 5} + \frac{n}{\lg 25} + \frac{n}{\lg 125} + \dots + \frac{n}{\lg n} =$$

$$c_1 n + n \sum_{i=1}^{\log_5 n} \frac{1}{\lg 5^i} = c_1 n + c_2 n \sum_{i=1}^{\lg n} \frac{1}{i} = c_1 n + c_2 n \lg \lg n = \Theta(n \lg \lg n)$$

c) case 3: $T(n) = \Theta(n^2 \sqrt{n}) = \Theta(n^{2.5})$

d) Master Theorem does not apply. Use substitution method. Guess that the solution is $T(n) = \Theta(n \lg n)$ and show that there exist c_1, c_2, n_0 such that $\forall n \geq n_0 : T(n) \leq c_1 n \lg n$ and $\forall n \leq n_0 : T(n) \leq c_2 n \lg n$.

Let $n_0 = 15$, then

$$T(n) \leq (cn + 15c) \left(\lg \frac{n}{3} + \lg 2 \right) + \frac{n}{2} \leq cn \lg n - cn \lg 3 + 15c \lg n - 15c \lg 3 + cn + 15c - \frac{n}{2} \leq$$

$$cn \lg n - \left(cn(\lg 3 - 1) - \frac{n}{2} - 15c \lg n + 15c(\lg 3 - 1) \right) \leq cn \lg n$$

$$\Rightarrow T(n) = O(n \lg n)$$

The exact same argument bounds $T(n)$ below to show that $T(n) = \Omega(n \lg n)$. Thus $T(n) = \Theta(n \lg n)$

e) Master Theorem fails. We use iteration:

$$T(n) = 2T\left(\frac{n}{2}\right) + \frac{n}{\lg n} = 4T\left(\frac{n}{4}\right) + \frac{n}{\lg \frac{n}{2}} + \frac{n}{\lg n} =$$

$$2^k T\left(\frac{n}{2^k}\right) + \frac{n}{\lg \frac{n}{2^{k-1}}} + \frac{n}{\lg \frac{n}{2^{k-2}}} + \dots + \frac{n}{\lg n} =$$

$$nT(1) + \frac{n}{\lg 2} + \frac{n}{\lg 4} + \frac{n}{\lg 8} + \dots + \frac{n}{\lg n} =$$

$$c_1 n + n \sum_{i=1}^{\log_2 n} \frac{1}{\lg 2^i} = c_1 n + n \sum_{i=1}^{\lg n} \frac{1}{i} = c_1 n + n \lg \lg n = \Theta(n \lg \lg n)$$

f) Master Theorem does not apply. Use substitution method. Guess that the solution is $T(n) = \Theta(n \lg n)$ and show that there exist c_1, c_2, n_0 such that $\forall n \geq n_0 : T(n) \leq c_1 n \lg n$ and $\forall n \leq n_0 : T(n) \leq c_2 n \lg n$.

$$\begin{aligned}
T(n) &\leq c\left(\frac{n}{2}\right)\lg\left(\frac{n}{2}\right) + c\left(\frac{n}{4}\right)\lg\left(\frac{n}{4}\right) + c\left(\frac{n}{8}\right)\lg\left(\frac{n}{8}\right) + n \leq \\
&c\left(\frac{n}{2}\right)(\lg n - 1) + c\left(\frac{n}{4}\right)(\lg n - 2) + c\left(\frac{n}{8}\right)(\lg n - 3) + n \leq \\
&cn\lg n \frac{7}{8} - \frac{3}{8}cn \leq cn\lg n \\
&\Rightarrow T(n) = O(n\lg n)
\end{aligned}$$

The exact same argument bounds $T(n)$ below to show that $T(n) = \Omega(n\lg n)$. Thus $T(n) = \Theta(n\lg n)$

g) Use iteration.

$$T(n) = T(n-1) + \frac{1}{n} = T(n-2) + \frac{1}{n-1} + \frac{1}{n} = \sum_{i=1}^n \frac{1}{i} = \lg n = \Theta(\lg n)$$

h) Use iteration.

$$T(n) = T(n-1) + \lg n = T(n-2) + \lg(n-1) + \lg n = \sum_{i=1}^n \lg i = c + \lg n! = \Theta(n\lg n)$$

i) Use iteration.

$$T(n) = T(n-2) + 2\lg n = T(n-4) + 2\lg(n-2) + 2\lg n = 2\sum_{i=1}^{\frac{n}{2}} \lg 2i = c + \lg n! = \Theta(n\lg n)$$

j) Method of changing variables.

Let $m = \lg n$, and $T(2^m) = S(m)$

$$S(m) = 2^{\frac{m}{2}} S\left(\frac{m}{2}\right) + 2^m. \text{ Now use iteration.}$$

$$S(m) = 2^{\frac{m}{2}} S\left(\frac{m}{2}\right) + 2^m = 2^{\frac{m}{2}} \left(2^{\frac{m}{4}} S\left(\frac{m}{4}\right) + 2^{\frac{m}{2}} \right) + 2^m = 2^{\frac{3m}{4}} S\left(\frac{m}{4}\right) + 2(2^m) =$$

$$2^{\frac{3m}{4}} \left(2^{\frac{m}{8}} S\left(\frac{m}{8}\right) + 2^{\frac{m}{4}} \right) + 2(2^m) = 2^{\frac{7m}{8}} S\left(\frac{m}{8}\right) + 3(2^m) =$$

$$2^{\frac{m-1}{m}} S(1) + \lg m(2^m) =$$

$$\Rightarrow T(n) = \frac{n}{2} T(2) + n \lg \lg n = \Theta(n \lg \lg n)$$

4-6) last updated fall 2006, Yuriy Brun, ybrun@usc.edu

a) Let there be two sets of chips: G, the set of good chips, and B, the set of bad chips. We know that the good chips are always correct in their test, but suppose the bad chips always say that other bad chips are "good" and that other "good" chips are bad. Then after performing every

possible pairwise test we will have the majority of chips say that every chip in G is “bad” and every chip in B is “good.” Now consider some of the bad chips flipping reporting good chips as “good” and bad chips as “bad.” Then the majority may say that good chips are “good” and bad chips are “bad.” But then you cannot tell the good chips from the bad chips.

b) Let n be the total number of chips.

Case 1: n is even.

Pick arbitrary pairs and test the chips. Of those pairs for which both chips claim the other is good, pick one at random to keep. Throw away all other chips. You are left with at least 1 chip (because the majority of chips are good, at least one will be a pair of good chips) and at most $\frac{n}{2}$ chips (because you keep no more than 1 chip from each of $\frac{n}{2}$ pairs). I now argue that the majority of the chips kept are good.

Of those pairs where at least 1 chip said the other is bad, either both chips were bad or just one was bad. So the majority of the chips discarded from those pairs are bad. Thus of the other pairs, the ones for which both chips said the other was good, the majority is good. For those pairs, there are two possibilities: both chips are good or both chips are bad. Because the majority of the chips is good, then the number of pairs of good chips is greater than the number of pairs of bad chips, thus by taking one of each, we are guaranteed that the majority of the kept chips are good. Thus we have $\frac{n}{2}$ chips, more than half of which are good.

Case 2: n is odd.

Arbitrarily place a chip aside and break up the rest of the chips into pairs at random. If the number of pairs of chips that both claim the other is good is even (case a), then keep one random chip from each of those pairs and the chip you set aside, otherwise (case b) only keep one random chip from each pair. You are left with at least 1 chip (case a: because you kept the chip you set aside; case b: there was at least one pair from which you kept a chip) and at most

$\left\lceil \frac{n}{2} \right\rceil$ chips (case a: because you keep no more than 1 chip from each of $\left\lfloor \frac{n}{2} \right\rfloor$ pairs plus 1 chip; case b: because you keep no more than 1 chip from each of $\left\lfloor \frac{n}{2} \right\rfloor$ pairs). I now argue that the majority of the chips kept are good.

Case a: Of those pairs where at least 1 chip said the other is bad, either both chips were bad or just one was bad. So the majority of the chips discarded from those pairs are bad. Thus of the other pairs, the ones for which both chips said the other was good, either the majority is good or exactly half is good and the chip we set aside is also good. If the majority is good, same argument as in case 1 holds, and we kept a majority of good chips. If exactly half are good, then by the same argument exactly half of the chips we keep are good and the left over chip is also good, so the majority of the kept chips are good.

Case b: If the left over chip is bad, then discarding it only helped us and by same argument as in case 1, the majority of the chips we kept are good. If the left over chip is good, at most exactly half of the chips tested may be bad. Because we had an odd number of pairs of chips that both claimed that other is good, we could not have paired each bad chip with a good chip (otherwise there'd be 0 such pairs, which is even). Thus there must have been at least two more good chips than bad chips in the original set, and thus the majority of the pairs that both reported their partner to be good must be pairs of good chips. Thus the majority of the chips kept must be good.

With a single iteration through this algorithm, we take n chips, the majority of which are good and return $\left\lceil \frac{n}{2} \right\rceil$ chips, the majority of which are good.

c) Recurse the above algorithm on the “kept” chips until you are down to 1 chip. The time for this algorithm is $T^{WC}(n) = T^{wc}\left(\left\lceil \frac{n}{2} \right\rceil\right) + \Theta(n)$, and by case 3 of the Master Theorem, $T^{WC}(n) = \Theta(n)$.

28.2-4) last updated fall 2006, Yuriy Brun, ybrun@usc.edu

For each case, we write the recursion and solve it using the master method:

$$T(n) = 132464T\left(\frac{n}{68}\right) + n^2 \Rightarrow \Theta\left(n^{\log_{68} 132464}\right) \approx \Theta\left(n^{2.795128}\right)$$

$$T(n) = 143640T\left(\frac{n}{70}\right) + n^2 \Rightarrow \Theta\left(n^{\log_{70} 143640}\right) \approx \Theta\left(n^{2.795123}\right)$$

$$T(n) = 155424T\left(\frac{n}{72}\right) + n^2 \Rightarrow \Theta\left(n^{\log_{72} 155424}\right) \approx \Theta\left(n^{2.795147}\right)$$

Strassen’s algorithm runs in $\Theta\left(n^{\lg 7}\right) \approx \Theta\left(n^{2.81}\right)$ so all these algorithms outperform Strassen. The second algorithm has the best running time.