Homework 2 Solutions

Fundamental Algorithms, Spring 2025, Professor Yap, Section Leader Dr. Bingwei Zhang

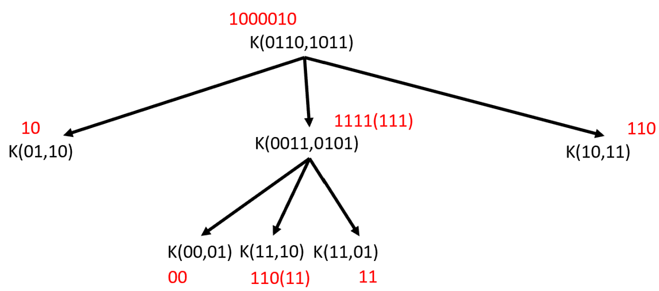
**Due:** Saturday Feb 15, in GradeScope by 11:30pm.
HOMEWORK with SOLUTION

INSTRUCTIONS:

- We have a "NO LATE HOMEWORK" policy.
  Special permission must be obtained *in advance* if you have a valid reason.
- Any submitted solution must be fully your own (you must not look at a fellow student's solution, *even if you have discussed with him or her*). Likewise, you must not show your writeup to anyone. We take the academic integrity policies of NYU and our department seriously. When in doubt, ask.
- The exercises in this homework come from Chapter II.
  **IMPORTANT**: You must download the latest version of Chapter II (dated Jan 30) from Brightspace.
- If you are not familiar with Gradescope, please try to practice uploading your solutions in advance. You can resubmit as many times as you like.

---

(Q1) (12 Points)
Exercise II.3.1, page 13.
Hand-simulation of Karatsuba.
**SOLUTION for Karatsuba Mult.**




(Q2) (8+12+4 Points)
Please do Ex.II.4.9, page 18.
Analysis of "Honest Karatsuba" (note that part(iii) is somewhat open-ended, so we only assign 4 points).

**THE QUESTION** ("Honest Karatsuba") In this question, we want to take into account the multiplicative constants that are hidden by the $\Theta$-notations. This is realistic or "honest".

(i) Argue that a more "honest" worst case recurrence for Karatsuba's algorithm should be

$$T(n) = 3T(\lceil n/2 \rceil + 1) + 5n + O(1). \tag{1}$$

Please justify all the constants $(1, 2, 3, 5)$ appearing in (1).

NOTE: since we are interested in constants in (1), we must tell you the cost to add two $n$-bit numbers: the cost is exactly $n$. Also, the cost to compute $Z$ from $Z_0, Z_1, Z_2$ is $2n$ (see ¶II.3, p. 7). But we don't really care about the $O(1)$ term in (1) (so we are still slightly abstract!).

(ii) Henceforth, assume $T(n)$ eventually satisfies the recurrence (1) *but without the* $O(1)$ *term.* We want to prove an upper bound $T(n)$ with explicit multiplicative constants. Consider a function of the form

$$U(n) = (n+3)^{\lg 3} - Kn \tag{2}$$

for some $K \geqslant 0$. Suppose $T(n) \leqslant U(n)$ (ev.). Determine the smallest possible value of the constant $K$. HINT: $\lceil n/2 \rceil + 1 \leqslant (n+3)/2$.

(iii) Argue why the upper bound (2) is STILL not "honest". How would do you suggest providing a realistic upper bound for $T(n)$? HINT: revoke DIC.

**SOLUTION:** Part(i) When we split $X$ to $X_0, X_1$, then $\max\{|X_0|, |X_1|\} = \lceil n/2 \rceil$. So when we recursively compute $Z_1 = (X_0 + X_1)(Y_0 + Y_1)$, the cost is $T(\lceil n/2 \rceil + 1)$ since $X_0 + X_1$ may have length $\lceil n/2 \rceil + 1$ bits. If non-recursive cost for $T(n)$ is denoted $A(n)$ (this is the cost of additions and shifts), we see that

$$T(n) \leqslant 2T(\lceil n/2 \rceil) + T(\lceil n/2 \rceil + 1) + A(n) \leqslant 3T(\lceil n/2 \rceil + 1) + A(n).$$

We see that $A(n)$ includes these costs:

- To compute $(X_0 + X_1)$: $(n/2) + O(1)$.
- To compute $(Y_0 + Y_1)$: $(n/2) + O(1)$.
- To compute $Z_0 + Z_2$: $n + O(1)$.
- To compute $Z_1 = (X_0 + X_1)(Y_0 + Y_1) - (Z_0 + Z_2)$: $n + O(1)$.
- To compute $Z$ from $(Z_0, Z_1, Z_2)$: $2n + O(1)$

Summing up the costs, $A(n) = 5n + O(1)$.

**SOLUTION:** Part(ii) CLAIM: $K$ must be at least 10. We will prove that $T(n) \leqslant U(n)$ by real induction, and in the process, we will discover that $K \geqslant 10$ is necessary and sufficient.

$$\begin{array}{rcll} T(n) & = & 3T(\lceil n/2 \rceil + 1) + 5n & \text{(from (1), ignoring the } O(1) \text{ term)} \\ & \leqslant & 3T((n+3)/2) + 5n & \text{(from HINT)} \\ & \leqslant & 3U((n+3)/2) + 5n & \text{(by induction hypothesis)} \\ & = & 3\Big[ (\frac{n+3}{2})^{\lg 3} - K\frac{n+3}{2} \Big] + 5n & \text{(by definition of } U(n)\text{)} \\ & < & \Big[ (n+3)^{\lg 3} - K\frac{3n}{2} \Big] + 5n & \text{(simplify)} \\ & \leqslant & (n+3)^{\lg 3} - Kn & \text{(provided } K \geqslant 10\text{)} \\ & = & U(n) & \text{(by definition of } U(n)\text{)} \end{array}$$

**SOLUTION:** Part(iii) There are two parts in the answer: (1) Why $U(n)$ is unrealistic. The fundamental reason is that our "default initial condition" (DIC) is unrealistic here. In reality, we need *explicit* initial conditions for the recurrence (1). E.g., we must choose $n_0$ such that the recurrence is used for $n \geqslant n_0$. Moreover, we must specify realistic values of $T(n)$ for $n < n_0$. E.g., choose $T(n) = n^2$ for $n < n_0$.

Looking at $U(n)$, we can crudely estimate that $U(n) = (n-3)^{\lg 3} - 10n < n^{1.6} - 10n = n(n^{0.6} - 10)$. Since our calculator shows $46^{0.6} < 10$, we conclude that $n_0 \geqslant 47$. This is a terribly unrealistic initial condition (your algorithm must do something special when $n < 47$).

(2) If $U(n)$ is no good, how can we get a realistic upper bound? The solution is to consider a broader class of functions, say $U(n) = A(n-3)^{\lg 3} - Kn$ for constants $A > 1$ and $K > 0$. If we choose $A$ large enough, we can match initial condition. For instance, it is realistic to choose $n_0 = 4$ (this ensures that when $n \geqslant n_2$, we have $(n-3)/2 < n$). Then we can choose $A$ such that $U(n)$ is a true upper bound on $T(n)$ for $n < n_0$.

(Q3) (15+15 Points)
Exercise II.9.2, page 50.
Solving two recurrences with Range and Domain transformations.

**THE QUESTION** Repeat the previous question for the following recurrences:
(a) $T(n) = 4T(n/2) + \frac{n^2}{\lg^2 n}$
(b) $T(n) = 4T(n/2) + \frac{n^2}{\sqrt{\lg n}}$.

**SOLUTION:** As in the previous exercise (II.9.1), we must first to reduce the recurrences to standard form. Then to solve it, first assuming that $n$ is a power of 2. Finally dropping that assumption about $n$.
(a) ANSWER is $T(n) = \Theta(n^2)$.
Proof: first, using the domain transformation where $n = 2^k$, we define $t(k) := T(2^k) = T(n)$. This yields the recurrence $t(k) = 4T(n/2) + n^2/\lg^2 n$ or

$$t(k) = 4t(k-1) + 4^k/k^2.$$

Next, use the range transformation $s(k) := t(k)/4^k$. Then we have $s(k) = t(k)/4^k = (4t(k-1) + 4^k/k^2)/4^k$ or

$$s(k) = s(k-1) + k^{-2}.$$

This is in standard form, and we can solve it:

$$s(k) = \sum_{i \geqslant 1}^{k} i^{-2} = \Theta(1).$$

Plugging back,

$$t(k) = 4^k s(k) = \Theta(4^k)$$

and

$$T(n) = t(\lg n) = \Theta(4^{\lg n}) = \Theta(n^2).$$

If $n$ is not a power of two, the same result holds.
(b) ANSWER is $T(n) = \Theta(n^2\sqrt{\log n})$.
Proof: using the same domain transformation, $t(k) = T(2^k) = T(n) = 4T(n/2) + n^2/\sqrt{\lg n}$. Thus,

$$t(k) = 4t(k-1) + 4^k/\sqrt{k}.$$

Next, using the range transformation $s(k) := t(k)/4^k$, we have $s(k) = t(k)/4^k = (4t(k-1) + 4^k/\sqrt{k})/4^k$ or

$$s(k) = s(k-1) + k^{-1/2}.$$

This is in standard form, and we get a polynomial-type sum:

$$s(k) = \sum_{i \geqslant 1}^{k} i^{-1/2} = \Theta(k^{1-(1/2)}) = \Theta(k^{1/2}).$$

Plugging back,

$$t(k) = 4^k s(k) = \Theta(4^k\sqrt{k})$$

and

$$T(n) = t(\lg n) = \Theta(n^2\sqrt{\lg n}).$$

If $n$ is not a power of two, the same result holds.

(Q4) (15 Points)
Exercise II.14.1, page 78.
Ordering six functions, $E1, \ldots, E_6$.

**HINT**: You can linearly order the functions $E_1, \ldots, E_6$ using one of these relationships $=, \asymp, \prec, \preceq, \ll$ . Prove the strongest possible relationship. E.g., if $f \ll g$ is true, but you only show $f \prec$, you will not get full credit.

**THE QUESTION** Consider the expression $E(n) := f(n)^{g(h(n))}$ where $(f, g, h) = (2^n, 1/n, \lg n)$. There are $6 = 3!$ possibilities for $E(n)$:

| $E(n)$ | $f$ | $g$ | $h$ |
|---|---|---|---|
| $E_1$ | $2^n$ | $1/n$ | $\lg n$ |
| $E_2$ | $2^n$ | $\lg n$ | $1/n$ |
| $E_3$ | $\lg n$ | $2^n$ | $1/n$ |
| $E_4$ | $\lg n$ | $1/n$ | $2^n$ |
| $E_5$ | $1/n$ | $2^n$ | $\lg n$ |
| $E_6$ | $1/n$ | $\lg n$ | $2^n$ |

Determine the domination relationships among these $E_1, \ldots, E_6$.
**HINT**: simplify the expressions. Is there a fastest growing function?

**SOLUTION:** CLAIM:

$$E_1 \ggg E_3 \ggg E_4 \ggg E_2 = E_5 = E^6.$$

(1) ($E_1$): We see that $E_1 = 2^{n/\lg n}$ or $\lg E_1 = n/\lg n$. Since $\lg E_1 \gg \lg n$, it is growing faster than any polynomial.

(2) ($E_2 = E_5 = E_6$): We simplify $E_2$ to $E_2 = (2^n)^{\lg(1/n)} = 2^{-n\lg n} = n^{-n}$. Likewise, $E_5$ is $(1/n)^{2^{\lg n}} = (1/n)^n = n^{-n}$ and $E_6$ is $(1/n)^{\lg(2^n)} = (1/n)^n = n^{-n}$. Thus, $\lg E_2 = -n\lg n$. Clearly, $E_2$ is a decreasing exponential-type.

(3) ($E_3$): Note that $E_3 = (\lg n)^{2^{1/n}}$ tends to $\lg n$ as $n \to \infty$, and this is because because $2^{1/n} \to 2^0 = 1$.

(4) ($E_4$): Also, $E_4 = (\lg n)^{1/2^n}$ tends to $(\lg n)^0 = 1$ as $n \to \infty$.

This proves that

$$E_1 \ggg E_3 \ggg E_4 \ggg E_2 = E_5 = E^6.$$

**Comments:** To do this problem, remember the rule that $a^{\log c} = c^{\log a}$. Beware of the paradoxical relation between "faster growing running time" and "faster algorithm": *if Algorithm A's running time is growing* ***faster*** *than Algorithm B's running time, we say Algorithm A is* ***slower*** *than Algorithm B!*

(Q5) (8+10+5+15 Points)
Exercise II.11.2, page 63.
Computing the watershed constant for $T(n) = aT(n/b) + cT(n/d) + 1$.

**THE QUESTION** Let $w$ be the watershed constant for the recurrence

$$T(n) = aT(n/b) + cT(n/d) + 1$$

where $a, c > 0$ and $b, d > 1$.

(a) How do you decide whether $w$ is zero, positive or negative?

(b) Give upper and lower bounds on $w$ using these four parameters:

$$\underline{a} := \min\{a, c\}, \quad \overline{a} := \max\{a, c\}, \tag{3}$$

$$\underline{b} := \min\{b, d\}, \quad \overline{b} := \max\{b, d\}. \tag{4}$$

(c) What are your bounds on $w$ when $(a, b, c, d) = (3, 2, 2, 3)$? Please give the numerical range.

(d) Give an algorithm called Approx$(a, b, c, d, n)$ which returns an approximation of $w$ to $n$ digits of accuracy, i.e., returns a value $\widetilde{w}$ satisfying $|w - \widetilde{w}| < 10^{-n}$. You must write your algorithm using pseudo-code (see ¶I.A.11 in Appendix of Chapter I).

**SOLUTION:** Part(a): We compare $a + c : 1$. Then $w = 0$ (resp., $w > 0$ or $w < 0$) if $a + c = 1$ (resp., $a + c > 1$ or $a + c < 1$).
Justification:
If $w = 0$, then $\frac{a}{b^0} + \frac{c}{d^0} = 1$, i.e., $a + c = 1$.

If $w > 0$, then $\frac{a}{b^w} + \frac{c}{d^w} = 1$, implies $\frac{a}{b^0} + \frac{c}{d^0} > 1$, i.e., $a + c > 1$.
The case where $w < 0$ is similarly shown.

**SOLUTION:** Part(b): Let $P(w) = \frac{a}{b^w} + \frac{c}{d^w}$. Clearly $\underline{a} \leqslant a \leqslant \overline{a}$ and $\overline{b}^w \geqslant b^w \geqslant \overline{b}^w$. Hence

$$\frac{\underline{a}}{\overline{b}^w} \leqslant \frac{a}{b^w} \leqslant \frac{\overline{a}}{\underline{b}^w}. \tag{5}$$

Similar, $\frac{c}{d^w}$.

$$\frac{\underline{a}}{\overline{b}^w} \leqslant \frac{c}{d^w} \leqslant \frac{\overline{a}}{\underline{b}^w}. \tag{6}$$

Adding (5) and (6),

$$2\frac{\underline{a}}{\overline{b}^w} \leqslant \frac{a}{b^w} + \frac{c}{d^w} P(w) \leqslant 2\frac{\overline{a}}{\underline{b}^w}.$$

Since $P(w) = \frac{a}{b^w} + \frac{c}{d^w} = 1$, we obtain

$$\frac{2\underline{a}}{\overline{b}^w} \leqslant 1 \leqslant \frac{2\overline{a}}{\underline{b}^w}.$$

Taking logs,

$$\log_{\overline{b}}(2\underline{a}) \leqslant w \leqslant \log_{\underline{b}}(2\overline{a}). \tag{7}$$

This is the answer we seek.

**SOLUTION:** Part(c): For $(a, b, c, d) = (3, 2, 2, 3)$, we obtain $(\underline{a}, \overline{a}, \underline{b}, \overline{b}) = (2, 3, 2, 3)$ and hence (7) implies

$$1.26 \simeq \log_3(4) \leqslant w \leqslant \log_2(6) \simeq 2.58.$$

Part(d): Given $(a, b, c, d, n)$, we use the formula in part(b) to compute an interval $I = [\ell, u]$ containing $w$. Then we do a binary search for $w$ in this interval $I$:

```
Approx(a, b, c, d, n):
    ℓ ← log_b̄ a̲
    u ← log_b̲ ā
    while (u − ℓ > 10^{−n})
        let m ← (u + ℓ)/2
        If P(m) > 1, then ℓ ← m
        Else if P(m) < 1, then u ← m
        Else return m      ◁ Found exact w!
    Return (u + ℓ)/2.
```

(Q6) (12+12 Points)

Refer to §II.5 (especially Theorem 3 in ¶II.17, page 24) for this question. Also, our definition of growth function has a redundant condition: if it is non-decreasing and unbounded, it is automatically positive eventually.
Consider the recurrence

$$T(x) = x + 4T(x/3) + 3T(x/4).$$

Use real induction to prove that

$$T(x) \leqslant Kx^2 \text{ (ev.)} \tag{8}$$

(a) Give a "partial real induction proof" of of (8). See ¶II.16 for what this means. For full credit, you must determine the smallest integer value of $K$.

(b) Complete the proof of (8) by showing the Real Basis Step (RB). But we want you to show this indirectly, by invoking Theorem 3 in ¶II.17.
**HINT**: you need to show that certain functions are growth functions, and the existence of $\delta > 0$.

**SOLUTION:**

(a)

$$\begin{array}{rcll} T(x) &=& x + 4T(x/3) + 3T(x/4) & \\ &\leqslant& x + 4K(x/3)^2 + 3K(x/4)^2 & \text{(by induction hypothesis)} \\ &=& Kx^2\left(\frac{1}{Kx} + \frac{4}{9} + \frac{3}{16}\right) & \text{(simplifying)} \\ &=& Kx^2\left(\frac{1}{Kx} + \frac{1}{91}144\right) & \text{(simplifying)} \\ &\leqslant& Kx^2 & \text{(provided } \frac{1}{Kx} \leqslant \frac{53}{144}\text{)} \end{array}$$

The proviso implies $K \geqslant \frac{144}{53}x$. Since $x \geqslant 1$ ($x$ is the input size, that must be a positive integer), we conclude that $K \geqslant 144/53$. The smallest integer value of $K$ is $K = 3$.

(b) To apply the theorem, we note that $T(x) = G(x, g_1(x), g_2(x))$ where $G(x, y, z) = x + 4y + 3z$, $g_1(y) = y/3$ and $g_2(z) = z/4$. These 3 functions are clearly growth functions. Moreover, if we choose $\delta = 1/4$ then $g_1(y) \leqslant y - \delta$ and $g_2(z) \leqslant z - \delta$ hold for all $y \geqslant 1$ an $z \geqslant 1$. Now part(a) already established the inequality

$$G(x, K \cdot f(g_1(x)), K \cdot f(g_2(x)) \leqslant Kf(x) \text{ (ev.)}$$

I.e.,

$$T(x) = x + K \cdot f((x/3)^2)) + K \cdot f((x/4)^2) \leqslant K \cdot x^2 \text{ (ev.)}$$

The theorem says that you can conclude that $T(x) \preceq f(x) = x^2$. So this is another way of saying that (RB) is automatically satisfied.