Algorithms Assignment 1

Shahroz Imtiaz & si6rf

2/11/2019

Collaborators: Guillermo Saavedra gs8rj, Meghan Do md9bt, Jerome Romualdez jhr3kd

1 Introduction

This assignment is shared between algorithms sections.
Credit: Assit Prof. Brunelle & Assit Prof. Hott

PROBLEM 1 *Asymptotic*

Prove or disprove each of the following conjectures. For 4-6, let $f(n)$ and $g(n)$ be asymptotically positive functions.

1. $2^{n+1} = O(2^n)$.

$$2^{n+1} \leq 2^n * c$$
$$\frac{2^{n+1}}{2^n} \leq c$$
$$2 \leq c$$
$$\text{let } c = 2$$
$$2^{n+1} \leq 2^n * 2$$
$$2^{n+1} \leq 2^{n+1}, \text{ which is true}$$

2. $2^{2n} = O(2^n)$.

$$2^{2n} \leq 2^n * c$$
$$\frac{2^n}{2^n} \leq c$$
$$2^n \leq c, \text{ no values for c satisfy this condition as } 2^n \text{ is not bounded}$$

3. Given that: $\forall \varepsilon > 0,\ \log(n) = o(n^{\varepsilon})$,

   show:

   $\forall \varepsilon, k > 0,\ \log^{k}(n) = o(n^{\varepsilon})$

$$
\begin{aligned}
\log^{k}(n) &< n^{\varepsilon} * c \\
\sqrt[k]{\log^{k}(n)} &< \sqrt[k]{n^{\varepsilon} * c} \\
\log(n) &< n^{\frac{\varepsilon}{k}} * c \\
\text{let } \varepsilon &= \frac{\varepsilon}{k} \\
\log(n) &< n^{\varepsilon} * c \\
\text{let } c &= 1 \\
\log(n) &< n^{\varepsilon}, \text{ which is true based on what we were given}
\end{aligned}
$$

PROBLEM 2 *Solving Recurrences*

Prove a (as tight as possible) O (big-Oh) asymptotic bound on the following recurrences. You may use any base cases you'd like.

1. $T(n) = 4T(\frac{n}{3}) + n\log n$

$$
\begin{aligned}
f(n) = n\log(n) \;\; a = 4 \;\; b = 3 \;\; \log_{3}(4) &\approx 1.26 \\
\text{Case 1 of Master Theorem:} & \\
& n\log(n) \in O(n^{\log_3(4) - \varepsilon}) \\
& \text{let } \varepsilon = .01 \\
& n\log(n) \in O(n^{\log_3(4) - .01}) \\
\text{which is true based on theorem provided during office hours:} & \\
& n\log(n) < c * n^{1+\varepsilon} \text{ for } \varepsilon > 0 \\
& \text{this implies } T(n) \in \Theta(n^{\log_3(4)})
\end{aligned}
$$

2. $T(n) = 3T(\frac{n}{3} - 2) + \frac{n}{2}$

$$f(n) = \frac{n}{2} \;\; a = 3 \;\; b = 3$$

Case 2 of Master Theorem:

$$\frac{n}{2} \in \Theta(n^1 * \log^0 n) \text{ for } k = 0$$

$$\frac{n}{2} = n * c$$

$$\text{let c} = \frac{1}{2}$$

$$\frac{n}{2} = \frac{n}{2}, \text{ which is true}$$

$$\text{this implies } T(n) = \Theta(n \log(n))$$

3. $T(n) = 2T(\sqrt{n}) + n$

$$\text{let } n = 2^m,\ m = \log(n)$$
$$T(2^m) = 2T(\sqrt{2^m}) + 2^m$$
$$T(2^m) = 2T(2^{m/2}) + 2^m$$
$$\text{let } S(m) = T(2^m)$$
$$S(m) = 2S(\frac{m}{2}) + 2^m$$
$$f(m) = 2^m \quad a = 2 \quad b = 2$$
$$\text{Case 3 of Master Theorem:}$$
$$2^m \in \Omega(n^{1+1}) \text{ for } \varepsilon = 1$$
$$2^m \in \Omega(n^2)$$
$$\text{Regularity Condition}$$
$$2 * (2^{m/2}) \leq k * 2^m$$
$$\log_2(2 * (2^{m/2})) \leq \log_2(k * 2^m)$$
$$\log_2(2) + \log_2(2^{m/2}) \leq \log_2(k) + \log_2(2^m)$$
$$\text{let } k = \frac{1}{2}$$
$$1 + \frac{m}{2} \leq -1 + m$$
$$2 \leq \frac{m}{2}$$
$$4 \leq m$$
$$\text{let } m = 4$$
$$2 * (2^{4/2}) \leq \frac{1}{2} * 2^4$$
$$2^3 \leq \frac{2^4}{2}, \text{ which is true}$$
$$\text{This implies } T(n) \in \Theta(n)$$

PROBLEM 3 *Where is Batman when you need him?*

As the newly-hired commissioner of the Gotham City Police Department, James Gordon's first act is to immediately fire all of the dirty cops, stamping out Gotham's widespread police corruption. To do this, Commissioner Gordon must first figure out which officers are honest and which are dirty. There are n officers in the department. The majority ($> n/2$) of the officers are honest, and every officer knows whether or not each other officer is dirty. He will identify the dirty cops by asking the officers, in pairs, to indicate whether the other is dirty. Honest officers will always answer truthfully, dirty cops may answer arbitrarily. Thus the following responses are possible:

Officer A	Officer B	Implication
"B is honest"	"A is honest"	Either both are honest or both are dirty
"B is honest"	"A is dirty"	At least one is dirty
"B is dirty"	"A is honest"	At least one is dirty
"B is dirty"	"A is dirty"	At least one is dirty

1. A group of n officers is uncorrupted if more than half are honest. Suppose we start with an uncorrupted group of n officers. Describe a method that uses only $\lfloor n/2 \rfloor$ pair-wise tests between officers to find a smaller uncorrupted group of at most $\lceil n/2 \rceil$ officers. Prove that your method satisfies each of the three requirements.

   You start by pairing $\lfloor n/2 \rfloor$ officers and testing them. If n one is even, you won't have any cops by themselves. If n is odd, you will have one untested cop by himself/herself. Now let the number of the minimum number of cops for the majority of the group to be uncorrected be minMajority=$\lfloor n/2 \rfloor$ + 1. If n is even, take one officer from each group that responded with honest-honest. This insures your sub group is not larger than $\lceil n/2 \rceil$. And since the majority of cops are uncorrupted, the majority of cops you just selected are also uncorrupted. If n is odd and the minMajority is even: if all the tests result in honest-honest, take one officer from each honest-honest test result. Else if n is odd and the number of honest-honest responses is equal to the number of responses that weren't honest-honest, take both officers from each honest-honest result + the untested officer. Else if n is odd and the minMajority is odd: if all of the tests result in honest-honest, take one officer from each honest-honest test result + the untested officer. Else if n is odd and the number of test that are honest-honest is even, take one officer from each honest-honest test result + the untested officer. Else if n is odd and the number of test that are honest-honest is odd, take one officer from each honest-honest test result. All of these cases satisfy the three requirements due to the reasons stated above.

2. Using this approach, devise an algorithm that identifies which officers are honest and which are dirty using only $\Theta(n)$ pairwise tests. Prove the correctness of your algorithm, and prove that only $\Theta(n)$ tests are used.

   Using the approach mentioned in 3.1, we can devise a recursive algorithm that narrows the subgroup until only one cop remains. We know this cop is clean due to the reasons mentioned above and the majority of cops being uncorrupted.This takes $\frac{n}{2}$ steps since we are dividing the group of officers

in half every time, so the total total run time of this approach is $T(n) = T(\frac{n}{2}) + \lfloor \frac{n}{2} \rfloor$. This is case 3 of the Master Theorem (for $\varepsilon = 1, n > \frac{n}{2}$) Thus, T(n)=$\Theta(n/2)$.
Once we have our one clean officer, we can pair him with the rest of the other officers and find out whether they are clean or dirty. These comparisons would depend on n. Therefore, the run time overall would be $\Theta(\frac{n}{2}) + \Theta(n)$, which is $\Theta(n)$

3. Prove that a conspiracy of $\lfloor n/2 \rfloor + 1$ dirty officers (who may share a plan) can foil *any* attempt to find a honest officer. I.e., not only will method above not work, but that there is no way *at all* for Commissioner Gordon to identify even one honest officer if there is not an honest majority.

   If there wasn't an honest majority, not only would my algorithm not work, but no algorithm would work, because as you shrink your subgroup more and more, the majority of your subgroup would remain corrupt and this would follow all the way down to when you only have one officer in your subgroup, thus you wouldn't even be able to identify even one honest cop.

PROBLEM 4 *Karatsuba Example*

Illustrate the Karatsuba algorithm on 20194102 × 37591056. Use 2-digit multiplication as your base case.

2019	4102
3759	1056

First Step:

$$a = 2019 \; b = 4102 \; c = 3759 \; d = 1056$$
$$n = 8$$
$$calcOne = a * c = 7589421$$
$$calcTwo = b * d = 4331712$$
$$calcThree = (a+b)(c+d) = 29472615$$
$$Final = 10^8(calcOne) + 10^4(calcThree - calcOne - calcTwo) + calcTwo$$
$$Final = 10^8(7589421) + 10^4(29472615 - 7589421 - 4331712) + 4331712$$
$$Final = 10^8(7589421) + 10^4(29472615 - 7589421 - 4331712) + 4331712$$
$$Final = 10^8(7589421) + 10^4(17551482) + 4331712$$
$$Final = 10^8(7589421) + 10^4(17551482) + 4331712$$
$$Final = 758942100000000 + 175514820000 + 4331712$$
$$\textbf{return } Final = 759117619200000$$

20	19
37	59

Second Step - calcOne calculation from step 1:

$$\begin{aligned}
a = 20\ b &= 19\ c = 37\ d = 59 \\
n &= 4 \\
calcOne' &= a * c = 740 \\
calcTwo' &= b * d = 1121 \\
calcThree' &= (a+b)(c+d) = 3744 \\
Final' &= 10^4(calcOne') + 10^2(calcThree' - calcOne' - calcTwo') + calcTwo' \\
Final' &= 10^4(740) + 10^2(3744 - 740 - 1121) + 1121 \\
Final' &= 10^4(740) + 10^2(3744 - 740 - 1121) + 1121 \\
Final' &= 10^4(740) + 10^2(1883) + 1121 \\
Final' &= 7400000 + 188300 + 1121 \\
\textbf{return } Final' &= 7589421
\end{aligned}$$

41	02
10	56

Third Step - calcTwo calculation from step 1:

$$\begin{aligned}
a = 41\ b &= 02\ c = 10\ d = 56 \\
n &= 4 \\
calcOne' &= a * c = 410 \\
calcTwo' &= b * d = 112 \\
calcThree' &= (a+b)(c+d) = 2838 \\
Final' &= 10^4(calcOne') + 10^2(calcThree' - calcOne' - calcTwo') + calcTwo' \\
Final' &= 10^4(410) + 10^2(2838 - 410 - 112) + 112 \\
Final' &= 10^4(410) + 10^2(2316) + 112 \\
Final' &= 4100000 + 231600 + 112 \\
\textbf{return } Final' &= 4331712
\end{aligned}$$

2019	4102
3759	1056

Fourth Step - calcThree calculation from step 1:

$$\begin{aligned} a = 2019\ b = 4102\ c = 3759\ d = 1056 \\ n = 8 \\ calcOne' &= a + b = 6121 \\ calcTwo' &= c + d = 4815 \\ \textbf{return } Final' &= calcOne' * calcTwo' = 29472615 \end{aligned}$$

61	21
48	15

Fifth Step - Final′ calculation from step 4:

$$\begin{aligned} a = 61\ b = 21\ c = 48\ d = 15 \\ n &= 4 \\ calcOne' &= a * c = 2928 \\ calcTwo' &= b * d = 315 \\ calcThree' &= (a + b)(c + d) = 5166 \\ Final' &= 10^4(calcOne') + 10^2(calcThree' - calcOne' - calcTwo') + calcTwo' \\ Final' &= 10^4(2928) + 10^2(5166 - 2928 - 315) + 315 \\ Final' &= 10^4(2928) + 10^2(5166 - 2928 - 315) + 315 \\ Final' &= 10^4(2928) + 10^2(1923) + 315 \\ Final' &= 29280000 + 192300 + 315 \\ \textbf{return } Final' &= 29472615 \end{aligned}$$