Algorithms Assignment 1

Chuanchuan He ch9nd

02/16/2019

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.

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

   This statement is **true**.
   Proof.
   Suppose $2^{n+1} = O(2^n)$, then there exists a constant c such that for n beyond some n_0, $2^{n+1} \leq$ c * 2^n.
   Since $2^{n+1} = 2 * (2^n)$, so when $c \geq 2$, for $n \geq 0$, $2^{n+1} \leq$ c * 2^n always hold.
   Thus, we prove that $2^{n+1} = O(2^n)$ is true.

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

   This statement is **false**.
   Proof.
   Suppose $2^{2n} = O(2^n)$, then there exists a constant c such that for n beyond some n_0, $2^{2n} \leq$ c * 2^n. Dividing both sides by 2^n, we got $2^n \leq$ c, and there is no value for c and n_0 that can make this true.
   Thus, we prove that $2^{2n} = O(2^n)$ is false.

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

   show:

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

Proof. By definition of the given statement: $\forall \varepsilon_0 > 0$, $\exists$c$_0$, n_0 such that $\forall n >$n$_0$, $\log(n) <$c$_0$ * n$^{\varepsilon_0}$.
Let c_0 = $c^{1/k}$, and let $\varepsilon_0 = \varepsilon / k$ (real numbers are closed under multiplication and division), then we get log(n) $< c^{1/k} n^{\varepsilon/k}$.
By raising the power of both sides to the k, we got $\log^k(n) =$c*n$^\varepsilon$.
Thus by the definition we prove that $\forall \varepsilon, k > 0,\ \log^k(n) = o(n^\varepsilon)$.

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. $T(n) = 4T(\frac{n}{3}) + n \log n$
   Proof.
   T(n) is in the form of $a * T(\frac{n}{b}) + f(n)$ with a = 4, b = 3, f(n) = nlogn, and $\log_b a = \log_3 4 = 1.262$.
   Since $\lim_{x \to \infty} \frac{n*logn}{n^{1.262-\varepsilon}} = 0$, so $f(n) = O(n^{\log_3 4 - \varepsilon})$ (Case 1 of Master Theorem).
   Thus,
   $$T(n) \in \theta(n^{\log_3 4})$$
   .

2. $T(n) = 3T(\frac{n}{3} - 2) + \frac{n}{2}$
   Proof.
   Since we can ignore the constant, the question is equivalent to $T(n) = 3T(\frac{n}{3}) + \frac{n}{2}$.
   T(n) is in the form of $a * T(\frac{n}{b}) + f(n)$ with a = 3, b = 3, f(n) = $\frac{n}{2}$, and $\log_b a = \log_3 3 = 1$.
   Since f(n) = $\frac{n}{2} = \theta(n^{\log_b a} * (log(n))^k) = \theta(n^{\log_3 3} * (log(n))^0) = \theta(n)$, when $k = 0$ (Case 2 of Master Theorem).
   Thus,
   $$T(n) \in \theta(nlogn)$$
   .

3. $T(n) = 2T(\sqrt{n}) + n$
   Proof.
   Let $n = 2^m$, so that $m = \log_2 n$ for substitution.
   Then, $T(2^m) = 2T(2^{\frac{m}{2}}) + 2^m$.
   Let $S(m) = 2S(\frac{m}{2}) + 2^m$, $S(m)$ is in the form of $a * T(\frac{n}{b}) + f(n)$ with $a = 2, b = 2, f(m) = 2^m$, so the Master Theorem can be used.
   $m^{\log_2 2} = m$, and $f(m) = 2^m$ = $\Omega(m^{\log_2 2 + \varepsilon}) = \Omega(m^{1+\varepsilon})$, which implies nothing, unless $a * f(\frac{m}{b}) \leq k * f(n)$ for some $k < 1$ and sufficiently large n.

In this case, $2 * 2^{\frac{m}{2}} \leq k * 2^m$, which gives us $0 \leq k < 1$ when m gets sufficiently large, so the Master Theorem can be used (case 3). So, $T(m) = \theta(f(m)) = \theta(2^m)$.
Thus,

$$T(n) \in \theta(n)$$

.

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.

   The method: we pair up the officers randomly, and then we ask them whether the other person is dirty or not. We then eliminate all the pairs that do nor say both are honest. Then we eliminate one officer from each pair. After that, all the people left forms a smaller uncorrupted group.
   Proof.
   We know that there are more honest officers than the dirty officers, so in the worst case(h is the number of honest officers and d is the number of dirty officers, h=d) for odd number of people h = d + 1, and for even number of people h = d + 2. Pairing the officers up has maximum of N/2 (N is the total number of people) comparisons, and forming N/2 groups of officers, and the method we have made sure that equal number of honest and dirty officers are eliminated.

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.

   Use the method we have to an uncorrupted group of officers, until only one or two officers are left, who must be honest officers given that in uncorrupted group honest officers are more than dirty officers.
   Proof.
   The recursion relation is: $T(n) = T(n/2) + n/2$, using the Tree Method we know that the total run-time to find the honest officers 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.

   Proof.
   Suppose there are N officers in total and there are $N/2 + 1$ dirty officers in them. When N is odd, then the relation between honest(h) and dirty officers(d) is d = h + 1. When N is even, the relation is d = h + 2. In both cases, the dirty officers can pick dirty officers and have every officer call them dirty, and call everyone dirty. Then the remaining dirty officers call each other honest, and call others dirty. The result would be one or two officers identified dirty by others and groups of honest and dirty officers calling each other dirty and we cannot tell from it.

PROBLEM 4 *Karatsuba Example*

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

$20194102 \times 37591056 =$
First, divide:
$10^8(2019 \times 3759) + 10^4[(2019 + 4102) \times (3759 + 1056) - (2019 \times 3759) - (4102 \times 1056)] + (4102 \times 1056)$
Then, divide more:
$(2019 \times 3759) = 10^4(20 \times 37) + 10^2[(20 + 19) \times (37 + 59) - (20 \times 37) - (19 \times 59)] + (19 \times 59) = 7589421$
$(4102 \times 1056) = 10^4(41 \times 10) + 10^2[(41 + 2) \times (10 + 56) - (41 \times 10) - (2 \times 56)] + (2 \times 56) = 4331712$
$(6121 \times 4815) = 10^4(61 \times 48) + 10^2[(61 + 21) \times (48 + 15) - (61 \times 48) - (21 \times 15)] + (21 \times 15) = 29472615$
Conquer:
$10^8(7589421) + 10^4(29472615 - 7589421 - 4331712) + (4331712) = 759117619151712$

So, the solution is:
759117619151712