COMPUTER SCIENCE E-20, SPRING 2015
Homework Problems
Recurrences

Due Thursday, April 30, 2015 before 9PM EDT. Upload a PDF of your answers at https://canvas.harvard.edu/courses/1815/assignments/23409

1. A hacker replaces your MergeSort routine with a new recursive sorting algorithm, SlowSort. SlowSort runs in three phases. In the first phase, the first 3/4 of the list (rounding up) is sorted recursively. In the second phase, the final 3/4 of the list is sorted recursively. Finally, in the third phase, the first 3/4 of the list is sorted recursively again. SlowSort sorts correctly, but it is slow. Assume that at each step, we require an additional number of processing steps proportional to the size of the list, for overhead. Write a recurrence describing SlowSort's running time.

   Challenge, worth 0 points: Prove that SlowSort sorts correctly.

   Solution:

   Slow Sort will sort a list of length $n = 2^k$ as follows:

   If $n = 1$ the list is sorted.

   If $n = 2^{k+1}$ and $k \geq 0$ then

   First Phase - sort the first $\frac{3}{4} * (2^{k+1})$ which is elements $L[1...\frac{3}{4} * (2^{k+1})]$
   Second Phase - sort the last $\frac{3}{4} * (2^{k+1})$ which is elements $L[((2^{k+1}) - \frac{3}{4} * (2^{k+1})) + 1....2^{k+1}] = L[(\frac{1}{4} * (2^{k+1})) + 1....2^{k+1}]$
   Third Phase - sort the last $\frac{3}{4} * (2^{k+1})$ which is elements $L[\frac{3}{4} * (2^{k+1})]$
   $T(2n) = 3(\frac{3}{4} * (2^{k+1})) + 6cn$
   $T(2n) = (\frac{9}{2} * (2^k)) + 6cn$
   $T(2n) = \frac{9}{2} * T(n) + 6cn$
   $T(1) = 1$
   $n = 1 : T(2) = \frac{9}{2} * T(1) + 6c = \frac{9}{2} + 6c$
   $n = 2 : T(4) = \frac{9}{2} * T(2) + 2(6c) = \frac{9}{2}(\frac{9}{2} + 6c) + 2(6c) = (\frac{9}{2})^2 + \frac{9}{2}(6c) + 2(6c) = (\frac{9}{2})^2 + (6c)(\frac{9}{2} + 2)$
   $n = 4 : T(8) = \frac{9}{2} * T(4) + 4(6c) = \frac{9}{2}((\frac{9}{2})^2 + (6c)(\frac{9}{2} + 2)) + 4(6c) =$
   $((\frac{9}{2})^3 + (6c)(\frac{9}{2})^2 + \frac{9}{2} * 2) + 4(6c) =$
   $(\frac{9}{2})^3 + (6c)((\frac{9}{2})^2 + \frac{9}{2} * 2 + 4)$

$n = 8 : T(16) = \frac{9}{2} * T(8) + 8(6c) =$
$\frac{9}{2}((\frac{9}{2})^3 + (6c)((\frac{9}{2})^2 + \frac{9}{2} * 2 + 4)) + 8(6c) =$
$(\frac{9}{2})^4 + (6c)((\frac{9}{2})^3 + (\frac{9}{2})^2 * 2 + (\frac{9}{2}) * 4 + 8)$

$n = 16 : T(32) = \frac{9}{2} * T(16) + 16(6c) = \frac{9}{2}((\frac{9}{2})^4 + (6c)((\frac{9}{2})^3 + (\frac{9}{2})^2 * 2 + (\frac{9}{2}) * 4 + 8)) + 16(6c) =$
$(\frac{9}{2})^5 + (6c)((\frac{9}{2})^4 + (\frac{9}{2})^3 * 2 + (\frac{9}{2})^2 * 4 + (\frac{9}{2}) * 8 + 16)$

$T(n) = (\frac{9}{2})^{\log_2 n+1} + (6c)\sum_{i=0}^{\log_2 n}(\frac{9}{2})^i(2)^{\log_2 n-i}$

2. Find functions fitting the following recurrences, and prove your answers by induction.

   (a) $f(1) = 5, f(n) = f(n-1) + 2$
   Solution:
   The function that fits the recurrence is

   $f(n) = 2 * (n+1) + 1$
   Proof By Induction :
   Base case $(n = 1) : f(1) = 2 * (2) + 1 = 5$

   Inductive case : Assume that the function is true for all n. We will now prove for $(n+1)$
   $f(n+1) = f(n+1-1) + 2$
   $= f(n) + 2$
   $= (2 * (n+1) + 1) + 2 -$ (By Induction on n)
   $= 2n + 2 + 1 + 2 = 2n + 5 -$ (LHS)
   Substituting $(n+1)$ inthe function for $f(n)$ we get
   $f(n+1) = 2 * (n+1+1) + 1 = 2 * (n+2) + 1 = 2n + 5 -$ (RHS)
   This completes the proof by Induction.

   (b) $g(0) = 0, g(n) = g(n-1) + 2n + 1$ Solution:
   The function that fits the recurrence is

   $g(n) = n^2 + 2n$
   Base case $(n = 0) : g(0) = 0 * (1) + 0 = 0$
   Base case $(n = 1) : g(1) = 1 + 2 = 3$
   Inductive case : Assume that the function is true for all n. We will now prove for $(n+1)$

   $g(n+1) = g(n+1-1) + 2(n+1) + 1$
   $= g(n) + 2(n+1) + 1$
   $= (n^2 + 2n) + 2(n+1) + 1-$ (By Induction on n)

$= n^2 + 2n + 2n + 2 + 1$

$= n^2 + 4n + 3$ – (LHS)
Substituting $(n + 1)$ in function for $g(n)$ we get
$g(n + 1) = (n + 1)^2 + 2(n + 1)$
$= n^2 + 2n + 1 + 2n + 2$
$= n^2 + 4n + 3$ – (RHS)
This completes the proof by Induction.