Analysis of Algorithms Midterm

Jay R Bolton

May 11, 2012

1 ***(a)***

$$
\begin{aligned}
&T(n) = 3T(n/4) + n\ log\ n \\
&a = 3, b = 4, f(n) = n\ log\ n \\
&f(n) = \Omega(n^{log_4 3+e}) \quad \text{case 3} \\
&(n\ log\ n)/(n^{log_4 3}) \quad \text{(it is polynomially larger)} \\
&\text{Regularity condition:} \\
&3(n/4)log(n/4) \leq c \cdot n\ log\ n \\
&n(3/4)(log\ n - log\ 4) \leq c \cdot n\ log\ n \\
&3/4\ n\ log\ n - 3/4\ n\ log\ 4 \leq c \cdot n\ log\ n \\
&\text{(Passes)} \\
&T(n) = \Theta(n\ log\ n)
\end{aligned}
$$

(b)

$$
\begin{aligned}
&T(n) = 3T(n/3) + n/3 \\
&a = 3, b = 3, f(n) = n/3 \\
&f(n) = \Theta(n^{log_3 3}) \quad \text{case 2} \\
&T(n) = \Theta(n\ lg\ n)
\end{aligned}
$$

2 ***(a)*** $T(n) = 3T(n/4) + n\ log\ n$

```
                  n log n
           /          |          \
(n/4)log(n/4)  (n/4)log(n/4)  (n/4)log(n/4)
...                 ...            ...
```

Height of tree is $h = log_4 n$, and width of leaves is $n^{log_4 3}$

$$T(n) = \Theta(n^{log_4 3}) + \sum_{i=1}^{h} 3^i((n/4^i)\ log\ (n/4^i))$$

$T(n) = 3T(n/3) + n/3$

```
                     n/3
           /          |          \
     (n/3)/3       (n/3)/3     (n/3)/3
...                 ...            ...
```

Height of tree is $h = log_3 n$, and width of leaves is $n^{log_3 3}$

$$T(n) = \Theta(n) + \sum_{i=1}^{h}(3^i((n/3^i)/3)) = \sum_{i=1}^{h}\frac{n}{3}$$

(b)

$$
\begin{aligned}
&(a)\\
&T(n) = 3T(n/4) + n\ log\ n\\
&= \Theta(n^{log_4 3}) + \sum_{i=1}^{h} 3^i((n/4^i)\ log\ (n/4^i))\\
&= \Theta(n^{log_4 3}) + \sum_{i=1}^{h} n \cdot 3^i/4^i \cdot log\ (n/4^i)\\
&= \Theta(n^{log_4 3}) + \sum_{i=1}^{h} n \cdot log(n/4^i) \cdot 3^i/4^i\\
&= \Theta(n^{log_4 3}) + \sum_{i=1}^{h} (n\ log\ n - n\ log\ 4^i) \cdot 3^i/4^i\\
&= \Theta(n^{log_4 3}) + \sum_{i=1}^{h} 3^i/4^i \cdot n\ log\ n - 3^i/4^i \cdot n\ log\ 4^i\\
&= \Theta(n^{log_4 3}) + \sum_{i=1}^{h} 3^i/4^i \cdot n\ log\ n - \sum_{i=1}^{h} 3^i/4^i \cdot n\ log\ 4^i\\
&= \Theta(n^{log_4 3}) + n\ log\ n \sum_{i=1}^{h} 3^i/4^i - \sum_{i=1}^{h} 3^i/4^i \cdot n\ log\ 4^i\\
&= \Theta(n^{log_4 3}) + n\ log\ n \sum_{i=1}^{h} (3/4)^i - n \sum_{i=1}^{h} (3/4)^i \cdot log\ 4^i\\
&= \Theta(n^{log_4 3}) + n\ log\ n \sum_{i=1}^{h} (3/4)^i - n \sum_{i=1}^{h} (3/4)^i \cdot log\ 4^i\\
&< \Theta(n^{log_4 3}) + n\ log\ n \sum_{i=1}^{\infty} (3/4)^i - n \sum_{i=1}^{h} (3/4)^i \cdot log\ 4^i\\
&= \Theta(n^{log_4 3}) + n\ log\ n \left(\frac{1}{1-(3/4)}\right) - n \sum_{i=1}^{h} (3/4)^i \cdot log\ 4^i\\
&= \Theta(n^{log_4 3}) + n\ log\ n \cdot 4 - n \sum_{i=1}^{h} (3/4)^i \cdot log\ 4^i\\
&= \Theta(n^{log_4 3}) + \Theta(n\ log\ n) - n \sum_{i=1}^{h} (3/4)^i \cdot log\ 4^i\\
&= \Theta(n^{log_4 3}) + \Theta(n\ log\ n) - n \sum_{i=1}^{h} \frac{3^i\ log\ 4^i}{4^i}\\
&= \Theta(n^{log_4 3}) + \Theta(n\ log\ n) - n \sum_{i=1}^{h} \left(\frac{3\ log\ 4}{4}\right)^i\\
&< \Theta(n^{log_4 3}) + \Theta(n\ log\ n) - n \sum_{i=1}^{\infty} \left(\frac{3\ log\ 4}{4}\right)^i
\end{aligned}
$$

$$
\begin{aligned}
&= \Theta(n^{log_4 3}) + \Theta(n\ log\ n) - n\left(\frac{1}{1-(3\ log\ 4/4)}\right) \\
&\approx \Theta(n^{log_4 3}) + \Theta(n\ log\ n) - n \cdot 0.15 \\
&= \Theta(n^{log_4 3}) + \Theta(n\ log\ n) - \Theta(n) \\
&\leq 3 \cdot \Theta(n\ log\ n) \\
&= \mathcal{O}(n\ log\ n)
\end{aligned}
$$

$$
\begin{aligned}
&(b) \\
&T(n) = 3T(n/3) + n/3 \\
&\ = \Theta(n) + \sum_{i=1}^{h} \frac{n}{3} \\
&\ = \Theta(n) + h \cdot (n/3) \\
&\ = \Theta(n) + (log_3 n) \cdot (n/3) \\
&\ = \Theta(n) + n/3\ log_3 n \\
&\ = \Theta(n) + \mathcal{O}(n\ lg\ n) \\
&\ = \mathcal{O}(n\ lg\ n)
\end{aligned}
$$

(c) For (a), where $T(n) = \Theta(n\ log\ n)$:
Inductive hypothesis: $T(n) \leq c \cdot n\ log\ n$ for $\mathcal{O}$ and $T(n) \geq c \cdot n\ log\ n$ for Ω.
Induction for $\mathcal{O}$:

$$
\begin{aligned}
&T(n) \leq 3c(n/4)\ log\ (n/4) + n\ log\ n \\
&\ = (3/4)cn(log\ n - log\ 4) + n\ log\ n \\
&\ = 3/4 \cdot cn\ log\ n - 3/4 \cdot cn\ log\ 4 + n\ log\ n \\
&\ = 3/4 \cdot cn\ log\ n - n(3/4 \cdot c\ log\ 4) + n\ log\ n \\
&\ = 3/4 \cdot cn\ log\ n - nd + n\ log\ n \\
&\ \leq 3/4 \cdot cn\ log\ n + n\ log\ n \\
&\ \leq cn\ log\ n \qquad \text{with } c \geq 4
\end{aligned}
$$

Induction for Ω:

$$
\begin{aligned}
&T(n) \geq 3c(n/4)\ log\ (n/4) + n\ log\ n \\
&\ = 3/4 \cdot cn\ log\ n - nd + n\ log\ n \\
&\ \geq cn\ log\ n \qquad \text{with } c \leq 1
\end{aligned}
$$

For (b), where $T(n) = \Theta(n)$:
Inductive hypothesis: $T(n) = c \cdot n\ lg\ n$ for Θ.
Induction for $\mathcal{O}$:

$$
\begin{aligned}
&T(n) \leq 3(c(n/3)lg(n/3)) + n/3 \\
&\ = cn\ lg(n/3) + n/3 \\
&\ = cn\ lg\ n - cn\ lg\ 3 + n/3 \\
&\ \leq cn\ lg\ n
\end{aligned}
$$

Induction for Ω:

$$
\begin{aligned}
T(n) &\geq 3(c(n/3)lg(n/3)) + n/3 \\
&= cn\ lg(n/3) + n/3 \\
&= cn\ lg\ n - cn\ lg\ 3 + n/3 \\
&\geq cn\ lg\ n \qquad \text{with } c \leq \frac{1}{2}
\end{aligned}
$$

($1/2 \cdot lg\ 3$ is less than $1/3$.)

(d) Yay!

3 ***(a)***

$$
\begin{array}{cccc}
2 & 3 & 4 & 5 \\
8 & 9 & 12 & \infty \\
14 & \infty & \infty & \infty \\
16 & \infty & \infty & \infty
\end{array}
$$

(b) Y[1,1] will be the least element in the matrix (least of the least of the columns and least of the least of the rows). If it is a singleton tableau, then Y[1,1] is the only populated cell. If Y[1,1] is infinity/null, then there is no least element and thus no elements.

If Y[1,1] contains a non-null element then that means we have a least element, so m and n are 1 and our tableau is non-empty.

(c) My pseudocode ended up being some kind of strange imperative haskell. I hope it's readable.

```
extract_min t:
  min := t[0,0]
  t[0,0] := infinity
  percolate t (1,1)
  return min

percolate t (i,j):
  (i',j') := (i,j)
  (right?, below?) := (j+1 <= t.n, i+1 <= t.m)
  if below? and t[i,j] > t[i+1,j]   then (i',j') := (i+1,j)
  if right? and t[i',j'] > t[i,j+1] then (i',j') := (i,j+1)
  if i' != i or j' != j:
    swap t[i,j] t[i',j']
    percolate t (i',j')
```

And here it is in pseudo-haskell. It felt more awkward than the imperative, but maybe I just didn't do it well.

```
extract_min t = (t[0,0], percolate (replace t (0,0) infinity) 0 0)

percolate t (i,j)
 | has_b && t[i,j] > b =
    if has_r && b > r then continue (i,j+1)
    else continue (i+1,j)
 | has_r && t[i,j] > r = continue (i,j+1)
 | otherwise t
    where
    (has_r, has_b) = (j+1 <= t.n, i+1 <= t.m)
    (r, b) = (t[i,j+1], t[i+1,j])
    continue (i',j') = percolate (swap t[i,j] t[i',j']) (i',j')
```

To prove `extract_min`, our main goal will be to prove percolate. The parameter `t` is passed as a young tableau, so `min`, being `t[0,0]` is the least element of t and is returned. `t[0,0]` is replaced with infinity and then `percolate` is called on `t`.

Loop invariant (inductive hypothesis) of `percolate`: at the start of the function, the young tableau consisting of `t[0,0]` with maximum element `t[i,j]` is a young tableau.

Initialization (base case): `percolate` is first called on `t[0,0]`, making `i = 0` and `j = 0`. So we have a singleton young tableau with infinity as its only element, which is trivially a young tableau.

Maintenance (induction): we assume the loop invariant holds prior to the recursive call with `(i,j)`. The result of the subsequent code is dependent on some cases:

i. `t[i,j]` is greater than the cell directly below it (`t[i+1,j]`). This condition then has two subcases:

   A. `t[i+1,j]` is greater than the cell directly right (`t[i,j+1]`). In this case, `t[i,j]` is swapped with `t[i,j+1]`. Now we have the tableau `t` with elements from `t[0,0]` up to the maximum `t[i,j]`. By the inductive hypothesis, `t[i,j] > t[i,j-1]` (unless `i==0`) and `t[i,j] > t[i-1]` (unless `i==0`) so our loop invariant holds.

   B. otherwise, we swap `t[i+1,j]` with `t[i,j]`. By the inductive hypothesis, `t[i,j] > t[i-1,j]` and by our conditional `t[i,j] <= t[i,j+1]`, so our loop invariant holds.

ii. Else if the cell directly right (`t[i,j+1]`) is greater than our current cell (`t[i,j]`) and greater than or equal to the one directly below (`t[i+1,j]`), then we swap `t[i,j]` and `t[i,j+1]`. By the inductive hypothesis, `t[i,j] >= t[i,j-1]`, unless `j == 0`. Also by the inductive hypothesis, `t[i,j] >= t[i-1,j]` unless `i == 0`. Thus the loop invariant holds.

(d)

$$
\begin{aligned}
&T(m+n) = T(m+n-1) + \mathcal{O}(1) \\
&T(p) = T(p-1) + \mathcal{O}(1) \\
&= \sum_{1}^{p} \mathcal{O}(1) \\
&= p \cdot \mathcal{O}(1) \\
&= (m+n) \cdot \mathcal{O}(1) \\
&= \mathcal{O}(m+n)
\end{aligned}
$$