CSE 321

Homework 3

Berke Belgin

171044065

1)

a) $T(n)=27\ T(n/3) + n^2$

for a.x(n/b) + f(n) -> a = 27, b = 3, d = 2

Since $a > b^d$ -> $27 > 3^2$,

= $Q(n^{\log_3 27})$ = $Q(n^3)$

b) $T(n)=9\ T(n/4) + n$

for a.x(n/b) + f(n) -> a = 9, b = 4, d = 1

Since $a > b^d$ -> $9 > 4^1$,

= $Q(n^{\log_4 9})$

c) $T(n)=2\ T(n/4) + \sqrt{n}$

for a.x(n/b) + f(n) -> a = 2, b = 4, d = 1/2

Since $a > b^d$ -> $2 = 4^{1/2}$ -> 2 = 2,

= $Q(\mathrm{n}^{\frac{1}{2}}.\log n)$ = $Q(\sqrt{n}.\log n)$

d) $T(n)=2\ T(\sqrt{n}) + 1$

$T(2^x) = 2.T(2^{x/2}) + 1$

$T(2^x) = Y(x)$

for a.x(n/b) + f(n), Y(x) = 2.Y(x/2) + 1 -> a = 2, b = 2, d = 0

Since $a > b^d$ -> $2 > 2^0$,

= $Q(n^{\log_2 2})$ = $Q(n)$

e) T(n)=2T(n-2), T(0)=1, T(1)=1

T(n) = a^n

$\frac{a^n}{a^{n-2}} = \frac{2a^{n-2}}{a^{n-2}}$, $a^2 = 2$, $a_+ = \sqrt{2}$, $a_- = -\sqrt{2}$

T(n) = $c_1.(\sqrt{2})^n + c_2.(-\sqrt{2})^n$

T(0) = $c_1 + c_2 = 1$

T(1) = $c_1.\sqrt{2} - c_2.\sqrt{2} = 1$ -> $c_1 - c_2 = \frac{1}{\sqrt{2}}$

$c_1 = \frac{2+\sqrt{2}}{4}$

$c_2 = \frac{2-\sqrt{2}}{4}$

= $Q(\frac{2+\sqrt{2}}{4}.\sqrt{2}^n + \frac{2-\sqrt{2}}{4}.(-\sqrt{2})^n)$

f) T(n)=4T(n/2)+n, T(1)=1

for a.x(n/b) + f(n) -> a = 4, b = 2, d = 1

Since a > b^d -> 4 > 2^1,

= $Q(n^{\log_2 4})$ = $Q(n^2)$

g) T(n)= 2 T($\sqrt[3]{n}$)+1 , T(3)=1

T(3^x) = 2.T($3^{x/3}$) + 1

T(3^x) = Y(x)

Y(x) = 2.Y(x/3) + 1

for a.x(n/b) + f(n), Y(x) = 2.Y(x/2) + 1 -> a = 2, b = 3, d = 0

Since a > b^d -> 2 > 3^0,

= $Q(n^{\log_3 2})$

2)

The general rule is T(n) = n.T(n/2) for print_lines. So the program prints n.T(n/2) lines.

(Ex:

6 – 18 -> 6.3.1

7 – 21 -> 7.3.1

8 – 64 -> 8.4.2.1

9 – 72 -> 9.4.2.1

10 – 100 -> 10.5.2.1

)

T(n) = n.(n/2).(n/4)…1,

T(n) = n.T(n/2) + 1, n = 2^x

for a.x(n/b) + f(n) -> a = 2^x, b = 2, d = 0

Since $2^x > 2^0$ -> $2^x > 1$,

= $Q(n^{\log_2 n})$

3)

The general rule is T(n) = 3.T(2n/3) for the recursion.

Since for n = 2, the program swaps once, it is

T(n) = 3.T(2n/3) + 1

Using Master Theorem,

for a.x(n/b) + f(n) -> a = 3, b = 3/2, d = 0

Since a > b^d -> 3 > $(3/2)^0$ -> 3 > 1

= $Q(n^{\log_{\frac{3}{2}} 3})$

4)

As we have learnt from previous courses, while insertion sort has an average case complexity of Q(n^2), quick sort has an average case complexity of Q(n.logn) operations.

Insertion Sort:

T_i = number of operations in sort algorithm,

T = $\sum_{i=1}^{n-1} T_i$ = $T_1 + T_2 + T_3 + \cdots + T_{n-1}$

A(n) = E[T] = E[$\sum_{i=1}^{n-1} T_i$]

A(n) = $\frac{n.(n-1)}{4} = Q(n^2)$

Quick Sort:

T_1 = number of operations in rearrange

T_2 = number of operations in recursive calls.

T = T_1 + T_2

A(n) = E[T] = E[T_1] + E[T_2]

A(n) $= Q(n \log n)$

```
$javac HelloWorld.java
$java -Xmx128M -Xms16M HelloWorld
254790
736
```

For 1000 random integers for example, we have a swap count for,

Insertion Sort: 254790,

QuickSort: 736,

As seen above.

5)

a) $T(n) = 5T(n/3) + n^2$

Using Master Theorem,

for a**.**x(n/b) + f(n) -> a = 5, b = 3, d = 2

Since $a > b^d$ -> $5 < 3^2$ -> $5 < 9$,

= $Q(n^2)$

b) $T(n) = 2T(n/2) + n^2$

Using Master Theorem,

for a**.**x(n/b) + f(n) -> a = 2, b = 2, d = 2

Since $a > b^d$ -> $2 < 2^2$ -> $2 < 4$,

= $Q(n^2)$

c) $T(n) = T(n - 1) + n$

Since we sum up n + (n - 1) + (n - 2) + … + (n – n) as (n.(n-1)) / 2, we can say that,

$$Q\left(\frac{n.(n-1)}{2}\right) = Q\left(\frac{n^2 - n}{2}\right) = Q(n^2)$$

I would choose algorithm “a)” since it divides problems into sub problems more than the others.