Master Method

Suppose $T(n) = a \cdot T(\frac{n}{b}) + f(n)$

even if $\lceil\rceil$, $\lfloor\rfloor$, ± 1

Then if $f(n)=n^c$, $c < \log_b a$ $n^c \in O(n^{\log_b a - (\log_b a - c)})$

if $f(n) \in O(n^{\log_b a - \varepsilon})$ for some $\varepsilon > 0$ then $T(n)$ is $\Theta(n^{\log_b a})$

base cases dominate

if $f(n)=n^c$, $c=\log_b a$

if $f(n) \in \Theta(n^{\log_b a})$ then $T(n)$ is $\Theta(\overset{n^{\log_b a}}{f(n)} \cdot \log n)$

balanced

if $f(n)=n^c$, $c>\log_b a$

if $f(n) \in \Omega(n^{\log_b a + \varepsilon})$ for some $\varepsilon > 0$ $T(n)$ is $\Theta(f(n))$

and if $a \cdot f(\frac{n}{b}) \le c \cdot f(n)$ for some $c > 1$ and all large n

1st divide / combine dominate

$f(n) = n \log n \rightarrow$ doesn't apply (to this version)

$T(n) = 2 \cdot T(\frac{n}{2}) + c \cdot n \log n$ $\quad \log_b a = \log_2 2 = 1$

$n \log n \overset{?}{\in} O(n^{0.9999})$? NO (but soln is $\Theta(n \log^2 n)$)

$n \log n \in \Theta(n)$? NO

$n \log n \in \Omega(n^{1.00001})$? NO

Examples: $T(n) = 3T(\frac{n}{2}) + n$

$\log_b a = \log_2 3 \approx 1.585$ $\quad n$ is $O(n^{\log_2 3 - 0.01})$ so $T(n)$ is $\Theta(n^{\log_b a}) = \Theta(n^{\log_2 3})$

$T(n) = T(\frac{2}{3}n) + 1$

$a=1$ $\quad b=\frac{3}{2}$

$\log_b a = \log_{\frac{3}{2}} 1 = 0$ $\quad 1$ is $\Theta(n^0)$ $\quad$ so $T(n)$ is $\Theta(\log n)$

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

$\log_b a = \log_4 3$ $\quad n \log n$ is $\Omega(n^{\log_4 3 + 0.01})$ so $T(n)$ is $\Theta(n \log n)$

$T(n) = 2T(\frac{n}{2}) + n \log n$

$\log_b a = \log_2 2 = 1$ $\quad n \log n$ is not $\Theta(n)$

is not $O(n^{1-\varepsilon})$ for any ε

is not $\Omega(n^{1+\varepsilon})$ for any ε

M.M (as given) does not apply

($\exists$ more powerful versions)

Mergesort $\quad T(n) = 2 \cdot T(\frac{n}{2}) + c \cdot n$

n is $\Theta(n^{\log_2 2})$ $\quad T(n)$ is $\Theta(n \log n)$

Binary Search on array $\quad T(n) = T(\frac{n}{2}) + c$

c is $\Theta(n^{\log_2 1})$ $\quad T(n)$ is $\Theta(\log n)$

Binary search on linked list (silly) $\quad T(n) = T(\frac{n}{2}) + cn$

cn is $\Omega(n^{\log_2 1 + 0.1})$ $\quad T(n)$ is $\Theta(n)$

Karatsuba integer mult $\quad T(n) = 3T(\frac{n}{2}) + cn$

n is $O(n^{\log_2 3 - 0.01})$ so $T(n)$ is $\Theta(n^{\log_2 3})$

Integer Multiplication (Karatsuba)

$\underbrace{862}_{x_1}\underbrace{341}_{x_0} \cdot \underbrace{979}_{y_1}\underbrace{468}_{y_0}$

$$\begin{array}{r} 862341 \\ 979468 \\ \hline 6898728 \\ 5174046 \\ 3449364 \\ \vdots \\ \hline \end{array}$$

$\Theta(n^2)$

$x = x_1 \cdot 1000 + x_0$

$y = y_1 \cdot 1000 + y_0$

$x \cdot y = (x_1 \cdot 1000 + x_0) \cdot (y_1 \cdot 1000 + y_0)$

$= x_1 \cdot y_1 \cdot 10^6 + x_1 \cdot y_0 \cdot 10^3 + y_1 \cdot x_0 \cdot 10^3 + x_0 \cdot y_0$

$T(n)$ = # ops to mult 2 n digit nums

$= 4 \cdot T(\frac{n}{2}) + O(n)$ ← work for 3 shifts and 3 additions

$\log_b a = \log_2 4 = 2 \quad f(n) = \Theta(n^1)$

$T(n)$ is $\Theta(n^{\log_b a}) = \Theta(n^2)$ (Master case 1)

$xy = x_1 \cdot y_1 \cdot 10^6 + x_1 \cdot y_1 \cdot 10^3 - 10^3 (x_1 - x_0)(y_1 - y_0) + 10^3 \cdot x_0 \cdot y_0 + x_0 \cdot y_0$

same subproblem same sub

$T(n) = 3T(\frac{n}{2}) + O(n) \rightarrow$ 4 shifts, 6 add/subtract

$\log_b a = \log_2 3 \quad f(n) \in O(n^{\log_2 3 - (\log_2 3 - 1)})$

Master case 1: $\Theta(n^{\log_2 3})$

$\approx \Theta(n^{1.58})$

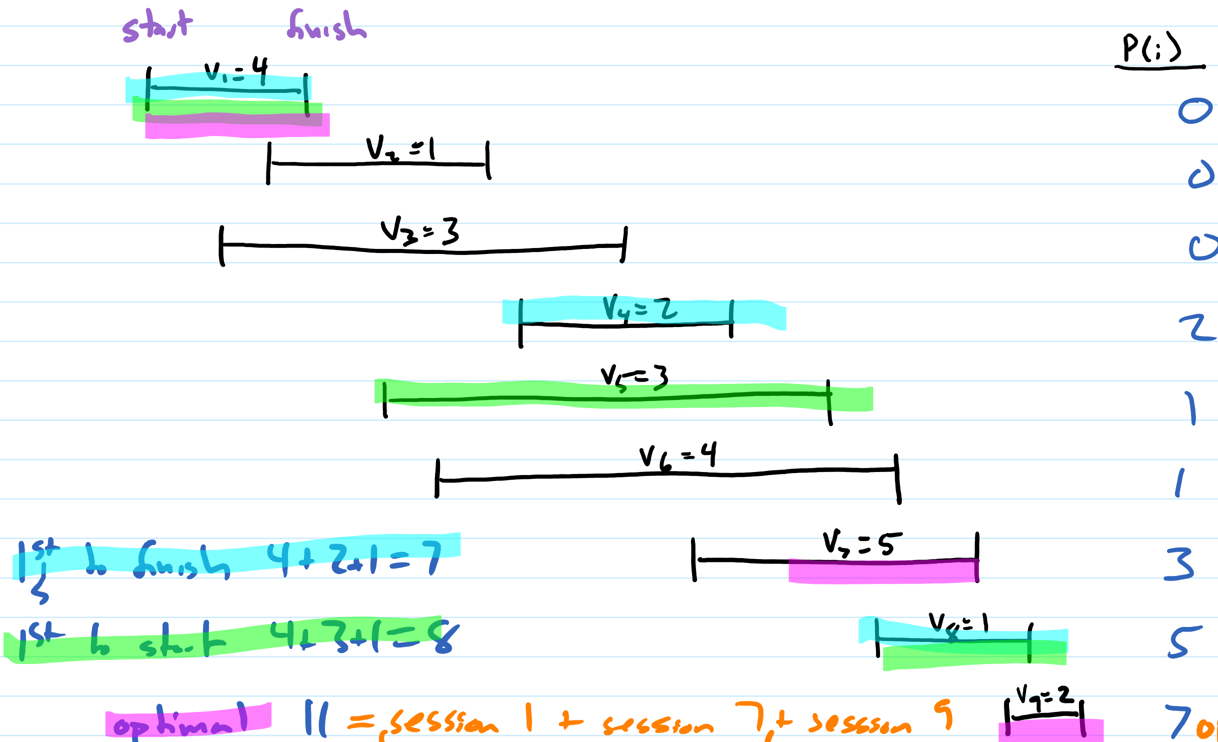
Java implementation

pre: both same # of digits (0-pad if not)

```
BigInteger karatsuba(String s1, String s2)
{
  // omitting zero-padding shorter number and taking care of sign

  int m = s1.length() / 2;

  String x1 = s1.substring(0, m);
  String x0 = s1.substring(m);
  String y1 = s2.substring(0, m);
  String y0 = s2.substring(m);

  // pretend the operations on powers of 10 are implemented in linear time
  BigInteger tenm = (new BigInteger("10")).pow(s1.length() - m);
  BigInteger ten2m = tenm.multiply(tenm);
  BigInteger z0 = karatsuba(x0, y0);
  BigInteger z2 = karatsuba(x1, y1);
  BigInteger z1 = karatsuba((new BigInteger(x1)).subtract(new BigInteger(x0)).toString(),
```

```
                                   (new BigInteger(y1)).subtract(new BigInteger(y0)).toString());

  return z2.multiply(ten2m).add(z2.multiply(tenm))
                           .subtract(z1.multiply(tenm))
                           .add(z0.multiply(tenm))
                           .add(z0);
}
```

Weighted Interval Selection



optimal || = $\underbrace{\text{session 1 + session 7}}_{\text{optimal for sessions finishing before 9 starts}}$ + session 9

optimal substructure

$OPT(i) =$ value of optimal selection using activities $\leq i$

$= OPT($sub problem w/ last piece removed + overlaps removed$)$ + last piece

$$OPT(i) = \begin{cases} 0 & \text{if } i = 0 \\ \max\left(\underbrace{OPT(P(i)) + v_i}_{\text{use interval } i},\ \underbrace{OPT(i-1)}_{\text{don't use interval } i}\right) \end{cases}$$