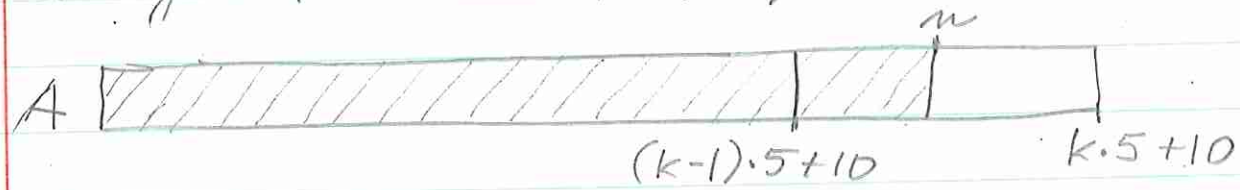
Notes on the cost of resizing arrays

1) Incremental upsizing.
Initial capacity 10
Capacity increment (additive) 5

n - the number of elements in an array A after k upsizings $(k>0)$



$(k-1)\cdot 5+10 < n \leq k\cdot 5+10$, so

$k-1 < \frac{n-10}{5}$ and $k \geq \frac{n-10}{5}$.

In other words, k is the smallest number that is at least as big as $\frac{n-10}{5}$.

(1) Hence, $k = \left\lceil \frac{n-10}{5} \right\rceil = \left\lceil \frac{n}{5} - \frac{10}{5} \right\rceil = \left\lceil \frac{n}{5} \right\rceil - 2$.

The total time spent on upsizing may be measured in the number of elements of A copied from smaller array(s) to larger array(s).

$$R(n)^u = 10 + (5+10) + (2*5+10) + \ldots + ((k-1)\cdot 5 + 10) =$$

$$= \sum_{i=0}^{k-1} (i \cdot 5 + 10) = 5 \cdot \sum_{i=0}^{k-1} i + \sum_{i=0}^{k-1} 10 =$$

$$= 5\,\frac{k(k-1)}{2} + k * 10 = \frac{5}{2} k * (k+3) =$$

by virtue of (1)

(2) $$= \frac{5}{2}\left(\left\lceil \frac{n}{5} \right\rceil - 2\right)\left(\left\lceil \frac{n}{5} \right\rceil + 1\right).$$

So, $R^u(n) \geq \frac{5}{2}\left(\frac{n}{5} - 2\right)\left(\frac{n}{5} + 1\right) = \frac{n^2}{10} - \frac{n}{2} - 5$

$\in \Theta(n^2)$

and $R^u(n) \leq \frac{5}{2}\left(\frac{n+4}{5} - 2\right)\left(\frac{n+4}{5} + 1\right) =$

$= \frac{n^2}{10} + \frac{3n}{10} - 5.4 \in \Theta(n^2)$

Hence, $R^u(n) \in \Theta(n^2)$

The average cost of upsizing per element is

$$\frac{R^u(n)}{n}$$

So it is between $\frac{n}{10} - \frac{1}{2} - \frac{5}{n}$ and

$\frac{n}{10} + 0.3 - \frac{5.4}{n}$. Therefore, is $\Theta(n)$.

2) incremental downsizing

The cost incremental downsizing is (depending whether a slower or a faster version of downsizing program was used) is either the same as upsizing $R^u(n)$ or less $k*5$ than it.

So, either

(3) $R^d(n) = R^u(n)$

or

(4) $R^d(n) = R^u(n) - \left(\left\lceil \frac{n}{5} \right\rceil - 2\right) * 5 =$

$= R^u(n) - 5 * \left\lceil \frac{n}{2} \right\rceil + 10.$

By virtue of (4), (3) yields:

(5) $R(n) = R^u(n) + R^d(n) = 2 \cdot R^u(n) =$

$= 5 * \left(\left\lceil \frac{n}{5} \right\rceil - 2\right)\left(\left\lceil \frac{n}{5} \right\rceil + 1\right)$

and (4) yields

(6) $R(n) = R^u(n) + R^d(n) = 2 * R^u(n) - 5 * \left(\left\lceil \frac{n}{5} \right\rceil - 2\right) =$

$= 5 * \left(\left\lceil \frac{n}{5} \right\rceil - 2\right)\left(\left\lceil \frac{n}{5} \right\rceil + 1\right) - 5 * \left(\left\lceil \frac{n}{5} \right\rceil - 2\right) =$

$= 5 * \left(\left\lceil \frac{n}{5} \right\rceil - 2\right)\left(\left\lceil \frac{n}{5} \right\rceil + 1 - 1\right) =$

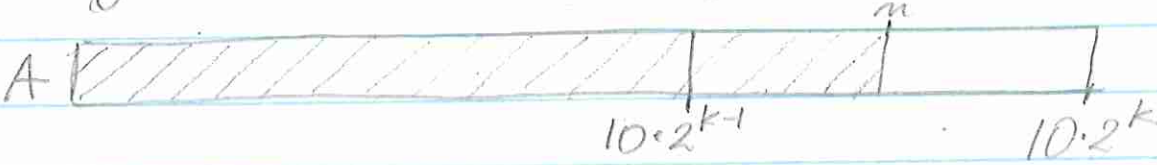
$= 5 * \left\lceil \frac{n}{5} \right\rceil \left(\left\lceil \frac{n}{5} \right\rceil - 2\right)$

Obviously, $R(n) \in \Theta(n^2)$, and the average cost of resizing per an element is

$$\frac{R(n)}{n} \in \Theta(n)$$

3) Multiplicative upsizing
Initial capacity 10
Capacity increment (multiplicative) 2

n - the numbers of elements in an array A after k upsizings ($k > 0$)



$10 \cdot 2^{k-1} < n \leq 10 \cdot 2^k$, so

$k - 1 < \log_2 \frac{n}{10}$ and $k \geq \log_2 \frac{n}{10}$.

In other words, k is the smallest number that is at least as big as $\log_2 \frac{n}{10}$.

(7) Hence, $k = \lceil \log_2 \frac{n}{10} \rceil$.

Now, the total time spent on uprising is

$$R^u(n) = 10 + 2*10 + 2^2*10 + \ldots + 2^{k-1}*10 =$$

$$= \sum_{i=0}^{k-1} 2^i * 10 = 10 \sum_{i=0}^{k-1} 2^i = 10*(2^k - 1) = 10 \cdot 2^k - 10 =$$

by virtue of (7)

(8) $$= 10 \cdot 2^{\lceil \log_2 \frac{n}{10} \rceil} - 10.$$

Comment. $2^{\lceil \log_2 x \rceil}$ is the smallest power of 2 that is at least as big as x.

$$R^u(n) \geq 10 \cdot 2^{\log_2 \frac{n}{10}} - 10 = 10 * \frac{n}{10} - 10 = n - 10.$$

$$R^u(n) < 10 \cdot 2^{\log_2 \frac{n}{10} + 1} - 10 = 10 * 2 * \frac{n}{10} - 10 = 2*n - 10$$

Hence, $R^u(n) \in \Theta(n)$.

the average cost of upsizing per element is

$$\frac{\mathcal{R}^u(n)}{n}$$

So, it is between $1-\frac{10}{n}$ and $2-\frac{10}{n}$.

Therefore, it is $\Theta(1)$.

4) Multiplicative downsizing.

$\mathcal{R}^d(n)$ is either the same as $\mathcal{R}^u(n)$ (for a slower program) or is half of $\mathcal{R}^u(n)$ (for a faster program).

So, either

(9) $\mathcal{R}^d(n) = \mathcal{R}^u(n)$

or

(10) $\mathcal{R}^d(n) = \frac{1}{2}\mathcal{R}^u(n)$

By virtue of (8), (9) yields

(11) $\mathcal{R}(n) = \mathcal{R}^u(n) + \mathcal{R}^d(n) = 2 * \mathcal{R}^u(n) =$

$= 20 * 2^{\lceil \log_2 \frac{n}{10} \rceil} - 20.$

and (10) yields

$$R(n) = R^u(n) + R^d(n) = 1.5 * R^u(n) =$$
$$= 15 * 2^{\lceil \log_2 \frac{n}{10} \rceil} - 15.$$

Of course, $R(n) \in \Theta(n)$, and the average cost of resizing per an element is

$$\frac{R(n)}{n} \in \Theta(1).$$

5) More general cases.

If the initial size of an array is c and the increment is d then

(12) $$R^u(n) = \frac{1}{2} \left\lceil \frac{n-c}{d} \right\rceil \left(d * \left\lceil \frac{n-(c+d)}{d} \right\rceil + 2c \right) \in \Theta(n^2)$$

for additive increment, and

(13) $$R^u(n) = c * \frac{d^{\lceil \log_d \frac{n}{c} \rceil} - 1}{d-1} \in \Theta(n)$$

for multiplicative increment.

the smallest power of d at least as big as $\frac{n}{c}$.