Amortized Time Complexity − Potential Method

The concept of potential is derived from physics . The idea is that a system changes states as per external operations. The change of state can be stored as a potential ϕ, which maps the instances of a data structure to those of a real number. Let c_i be the actual cost of an operation and the states of the structure are ϕ_i and ϕ_{i-1}, then the amortized cost can be represented as follows:

$$\sum_{i=1}^{m} a_i = \sum_{i=1}^{m} c_i + \phi(D_i) - \phi(D_{i-1}) = \sum_{i=1}^{m} c_i + \sum_{i=1}^{m} \phi(D_i) - \phi(D_{i-1})$$

$$= \sum_{i=1}^{m} c_i + \phi(D_m) - \phi(D_0)$$

Thus, the amortized cost is the total cost plus the difference between the beginning and end potential of the data structure. As $\phi(D_m)$ is greater than $\phi(D_0)$, the amortized cost is upper bound of the actual cost.

$\textit{Taking the example of Dynamic Table}$

$$\phi = 2 \times No.\, of\ items\ in\ the\ array - capacity\ of\ the\ array.$$

$Item\ No.\,(i)$	$Table\ Size$	$Cost\ of\ Operation$	$Cost\ of\ Doubling\ and\ Copying$	$Total\ Cost_i\ (C_i)$	ϕ $(2 \times Item\ No.) - (Table\ Size)$	$Amortized\ Cost$ $C_i + \phi(D_i) - \phi(D_{i-1})$ $= C_i + \phi - (\phi - 1)$
1	1	1	0	1	$2 \times 1 - 1 = 1$	$1 + 1 - (1 - 1) = 2$
2	2	1	$1 = 2^0$	2	$2 \times 2 - 2 = 2$	$2 + 2 - (2 - 1) = 3$
3	4	1	$2 = 2^1$	3	$2 \times 3 - 4 = 2$	3
4	4	1	0	1	$2 \times 4 - 4 = 4$	3
5	8	1	$4 = 2^2$	5	$2 \times 5 - 8 = 2$	3
6	8	1	0	1	$2 \times 6 - 8 = 4$	3
7	8	1	0	1	$2 \times 7 - 8 = 6$	3
8	8	1	0	1	$2 \times 8 - 8 = 8$	3
9	16	1	$8 = 2^3$	9	$2 \times 9 - 16 = 2$	3
10	16	1	0	1	$2 \times 10 - 16 = 4$	3

11	**16**	**1**	**0**	**1**	$2 \times 11 - 16 = 6$	**3**
12	**16**	**1**	**0**	**1**	$2 \times 12 - 16 = 8$	**3**
13	**16**	**1**	**0**	**1**	$2 \times 13 - 16 = 10$	**3**
14	**16**	**1**	**0**	**1**	$2 \times 14 - 16 = 12$	**3**
15	**16**	**1**	**0**	**1**	$2 \times 15 - 16 = 14$	**3**
16	**16**	**1**	**0**	**1**	$2 \times 16 - 16 = 16$	**3**
17	**32**	**1**	$16 = 2^4$	**17**	$2 \times 17 - 32 = 2$	**3**

As we see all have 3 ***and if it runs upto n times then it generates*** $O(3n) = O(n)$. ***The amortized Complexity*** $= O(3) = O(1)$ ***i.e. each at constant time.***

Now, if we see:

$$\phi = 2 \times \text{No. of items in the array} - \text{capacity of the array}.$$

if number of items $= i$ ***and capacity of the array:*** $2^{\lceil \log_2 i \rceil}$

$$\phi = 2i - 2^{\lceil \log_2 i \rceil}$$

$\boldsymbol{And\ from\ the\ above\ table\ we\ get\ two\ cases:}$

$C_i = i\ , when\ i - 1\ is\ an\ Actual\ Power\ of\ 2.$

$C_i = 1\ , when\ i - 1\ is\ not\ an\ Actual\ Power\ of\ 2.$

$$C_i = \begin{cases} i, when\ i-1\ is\ an\ Actual\ Power\ of\ 2(Actual\ Cost(C_i)\ is\ i) \\ \\ 1, Otherwise(Actual\ Cost\ (C_i) is\ 1) \end{cases}$$

$$\sum_{i=1}^{m} a_i = \sum_{i=1}^{m} c_i + \phi(D_i) - \phi(D_{i-1})$$

$$or, A_i = C_i + \phi(D_i) - \phi(D_{i-1}), where\ A_i = \sum_{i=1}^{m} a_i\ and$$

$$C_i = \sum_{i=1}^{m} c_i$$

$Case1: when\ i - 1\ is\ an\ Actual\ Power\ of\ 2,$

$Actual\ Cost(C_i)\ is\ i$

$$a_i = i(Actual\ Cost) + 2i - 2^{\lceil \log_2 i \rceil} - (2(i-1) - 2^{\lceil \log_2 i-1 \rceil})$$

$$= i + 2i - 2^{\lceil \log_2 i \rceil} - (2i - 2 - 2^{\lceil \log_2 i-1 \rceil})$$

$$= i + 2i - 2^{\lceil \log_2 i \rceil} - 2i + 2 + 2^{\lceil \log_2 i-1 \rceil}$$

$$= i + 2 - 2^{\lceil \log_2 i \rceil} + 2^{\lceil \log_2 i-1 \rceil}$$

$Now, we\ see\ i - 1\ is\ an\ actual\ power\ of\ 2$:

$Consider\ i\ to\ be\ 9\ which\ is\ not\ multiple\ of\ 2$

$i - 1 = 9 - 1 = 8$

$2^{\lceil \log_2 9 \rceil} = 2^{\lceil 3.16 \rceil} = 2^4 = 16$

$2^{\lceil \log_2 9-1 \rceil} = 2^{\lceil 3 \rceil} = 2^3 = 8$

$2^{\lceil \log_2 i-1 \rceil} = (i - 1) = 8$

$2^{\lceil \log_2 i \rceil} = 2(i - 1) = 2 \times 8 = 16$

$Hence\ replacing\ 2^{\lceil \log_2 i-1 \rceil}\ and\ 2^{\lceil \log_2 i \rceil}\ with\ (i - 1) and$

$2(i - 1) we\ get$:

$= i + 2 - 2(i - 1) + (i - 1)$

$= i + 2 - 2i + 2 + i - 1$

$= 2i - 2i + 2 + 1$

$= 3$

$Case2: when\ i - 1\ is\ not\ an\ Actual\ Power\ of\ 2,$

$Actual\ Cost(C_i)\ is\ 1$

$a_i = 1(Actual\ Cost) + 2i - 2^{\lceil \log_2 i \rceil} - (2(i - 1) - 2^{\lceil \log_2 i-1 \rceil})$

$= 1 + 2i - 2^{\lceil \log_2 i \rceil} - (2i - 2 - 2^{\lceil \log_2 i-1 \rceil})$

$= 1 + 2i - 2^{\lceil \log_2 i \rceil} - 2i + 2 + 2^{\lceil \log_2 i-1 \rceil}$

$= 1 + 2 - 2^{\lceil \log_2 i \rceil} + 2^{\lceil \log_2 i-1 \rceil}$

$Say\ \ i = 8, then\ i - 1 = 8 - 1 = 7$

$2^{\lceil \log_2 7 \rceil} = 2^{\lceil 2.8 \rceil} = 2^3 = 8$

$2^{\lceil \log_2 8 \rceil} = 2^{\lceil 3 \rceil} = 2^3 = 8$

$Hence\ we\ can\ say\ that : 2^{\lceil \log_2 i \rceil} = 2^{\lceil \log_2 i-1 \rceil} = i$

$= 1 + 2 - i + i = 3$

$As\ we\ see\ all\ have\ 3\ and\ if\ it\ runs\ upto\ n\ times\ then\ it\ generates\ O(3n) = O(n). The\ amortized\ cost = O(3) = O(1)\ i.e. each\ at\ constant\ time.$