Aggregate Method

方法：假設有 n 个 operations

而 n 个 operation 的 total cost 為 $T(n)$

每个 operation 的 amortized cost 為 $\frac{T(n)}{n}$

Accounting Method

方法：給不同的 operation 定義不同的 amortized cost

設 amortized cost 超出 actual cost 時，就為 credit，可用作後面 operation 之 prepayment

最終利用 total amortized cost 來估計 total actual cost

(1). credit 永遠不為負 (確保 amortized cost 做為 actual cost 之上界估計正確性)

(2). 令 $\hat{C}_i$ 為第 i 个 operation 之 amortized cost，則：$\sum_{i=1}^{n} \hat{C}_i \geq \sum_{i=1}^{n} C_i$

C_i 為第 i 个 operation 之 actual cost

∴ total credit 為：$\sum_{i=1}^{n} \hat{C}_i - \sum_{i=1}^{n} C_i > 0$

The potential method

方法：把 credit 存在 DS 上的某个 object 中

而 prepayment 之 charge 為 DS 之 potential energy

設做了 n 个 operations，而初始 DS 為 D_0

C_i 為第 i 个 operation 之 actual cost

define：$\Phi(D_i)$ 為 potential function，代表 D_i 上之 potential

則 amortized cost 為：$\hat{C}_i = C_i + \Phi(D_i) - \Phi(D_{i-1})$

若 $\Phi(D_i) > \Phi(D_{i-1})$：代表第 i 个 operation 會 overcharge

故：
$$\sum_{i=1}^{n} \hat{C}_i = \sum_{i=1}^{n} \left[C_i + \Phi(D_i) - \Phi(D_{i-1}) \right]$$
$$= \sum_{i=1}^{n} C_i + \Phi(D_n) - \Phi(D_0)$$

(1). $\Phi(D_n) \geq \Phi(D_0)$ (確保 amortized cost 做為 actual cost 之上界估計正確性) ⇒ n 很難提前知道

(2). $\Phi(D_i) \geq \Phi(D_0), \forall i$ ⇒ 必滿足 $\Phi(D_n) \geq \Phi(D_0)$

Example 1. Stack Operations

定義 Stack S 上之 3个 operation：
其中 stack 有n个 elements

1. pop(S) : $O(1)$
2. push(S) : $O(1)$
3. multipop(S, k) : $O(\min(n, k))$

①. aggregate method：

∵ multipop 中 pop 次數受限於 push 次數

設共有n次 operations，則 push 至多 n 次

∴ 三个 operation 的 amortized cost 會為 $\frac{O(n)}{n} = O(1)$

②. accounting method

actual cost 為：
- push(S) : 1
- pop(S) : 1
- multipop(S, k) : $\min(n, k)$

定義 amortized cost 為：
- push(S) : 2
- pop(S) : 0
- multipop(S, k) : 0

∵ stack 上，要 pop(S) 前一定要先 push(S)

∴ 可確保 credit 永遠不為負（∵ pop 次數一定 ≤ push 次數）

∴ $\sum_{i=1}^{n} \hat{C_i} = O(n)$，故 total actual cost 為 $O(n)$

③. potential method

定義 $\Phi(D_i)$ 為做完第 i 次 operation 後，S 中元素个數

∴ $\Phi(D_0) = 0$ 又 stack 中元素个數不為負，∴ $\Phi(D_i) \geq \Phi(D_0)$, $\forall i$

① push(S) : $\Phi(D_i) - \Phi(D_{i-1}) = 1 \Rightarrow \hat{C_i} = C_i + 1 = 2$

② pop(S) : $\Phi(D_i) - \Phi(D_{i-1}) = -1 \Rightarrow \hat{C_i} = C_i - 1 = 0$

③ multipop(S, k) : $\Phi(D_i) - \Phi(D_{i-1}) = -\min(n, k) \Rightarrow \hat{C_i} = C_i - \min(n, k) = 0$

故 $\sum_{i=1}^{n} \hat{C_i} = O(n)$

Example 2. Incrementing a binary counter

設有一 k bit 的 binary counter，令為 $A[0,\dots,k-1]$

而 binary counter 上有一 operation：increment(A)

∵ increment A 需 flip 為1的 bit，故 running time 和 1 個數相關

worst case 為 $O(k)$，若不以 amortized analysis 分析的話，為 $O(nk)$

但 binary counter 初始為0，不可能每次都為 $O(k$

Counter value	A[7]	A[6]	A[5]	A[4]	A[3]	A[2]	A[1]	A[0]	Total cost
0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	1	1
2	0	0	0	0	0	0	1	0	3
3	0	0	0	0	0	0	1	1	4
4	0	0	0	0	0	1	0	0	7
5	0	0	0	0	0	1	0	1	8
6	0	0	0	0	0	1	1	0	10
7	0	0	0	0	0	1	1	1	11
8	0	0	0	0	1	0	0	0	15
9	0	0	0	0	1	0	0	1	16
10	0	0	0	0	1	0	1	0	18
11	0	0	0	0	1	0	1	1	19
12	0	0	0	0	1	1	0	0	22
13	0	0	0	0	1	1	0	1	23
14	0	0	0	0	1	1	1	0	25
15	0	0	0	0	1	1	1	1	26
16	0	0	0	1	0	0	0	0	31

①. aggregate method：

flip 次數為：$\lfloor \frac{n}{2^0} \rfloor + \lfloor \frac{n}{2^1} \rfloor + \cdots + \lfloor \frac{n}{2^{k-1}} \rfloor < n(1+\frac{1}{2}+\cdots) = n\cdot\frac{1}{1-\frac{1}{2}}$

$= 2n$

Amortized Cost：$\frac{2n}{n} = O(1)$

②. acconting method：

定義 amortized cost 為：flip 0 to 1：2

flip 1 to 0：0

∵ 初始 counter 全為0，∴ flip 0 to 1 次數必 ≥ flip 1 to 0

⇒ credit 必不為負

⇒ total amortized cost：$\sum_{i=1}^{n} \hat{c}_i = 2n = O(n)$

③. potential method