1、对支持插入、删除的动态表中的操作进行平摊分析:

(1) 第 i 次操作是 TABLE-DELETE: 未收缩;

(2) 第 i 次操作是 TABLE-DELETE: 收缩.

解: 令f_i为动态表在第 i 个操作之后的装载因子, ϕ_i是第 i 个操作后的势能, c_i是第 i 个操作的真实代价, α_i是第 i 个操作的平摊代价.

(1). 由于没有收缩, 因此$size_i = size_{i-1}$, $num_i = num_{i-1} - 1$, $c_i = 1$.

情况一: $f_{i-1} < 1/2$, $f_i < 1/2$并且未收缩. $\phi_i = \frac{1}{2}size_i - num_i$, $\phi_{i-1} = \frac{1}{2}size_{i-1} - num_{i-1}$,

$\alpha_i = c_i + \phi_i - \phi_{i-1} = 1 + \left(\frac{1}{2}size_i - num_i\right) - \left(\frac{1}{2}size_{i-1} - num_{i-1}\right) = 2.$

情况二: $f_{i-1} \geq 1/2$, $f_i < 1/2$并且未收缩. $\phi_i = \frac{1}{2}size_i - num_i$, $\phi_{i-1} = 2num_{i-1} - size_{i-1}$,

$\alpha_i = c_i + \phi_i - \phi_{i-1} = 1 + \left(\frac{1}{2}size_i - num_i\right) - (2num_{i-1} - size_{i-1}) = 2 + \frac{3}{2}size_{i-1} - 3num_{i-1} \leq 2.$

情况三: $f_{i-1} > 1/2$, $f_i \geq 1/2$并且未收缩. $\phi_i = 2num_i - size_i$, $\phi_{i-1} = 2num_{i-1} - size_{i-1}$.

$\alpha_i = c_i + \phi_i - \phi_{i-1} = 1 + (2num_i - size_i) - (2num_{i-1} - size_{i-1}) = -1.$

(2). 由于删除后收缩, 因此$f_{i-1} = 1/4$, $f_i < 1/2$, $size_i = \frac{1}{2}size_{i-1}$, $num_i = num_{i-1} - 1$, $c_i = num_{i-1}$ (删掉一个元素, 再拷贝剩余的$num_{i-1} - 1$个元素; 当然, 这个代价还可以是$c_i = num_{i-1} - 1$或者$c_i = num_{i-1} + 1$, 例如只拷贝未删除的元素或者先拷贝所有元素再删除一个). $\phi_i = \frac{1}{2}size_i - num_i$, $\phi_{i-1} = \frac{1}{2}size_{i-1} - num_{i-1}$, $\alpha_i = c_i + \phi_i - \phi_{i-1} = num_{i-1} + \left(\frac{1}{2}size_i - num_i\right) - \left(\frac{1}{2}size_{i-1} - num_{i-1}\right) = 2 + num_i - \frac{1}{2}size_i < 2.$

下面补充第 i 次操作是 TABLE-INSERTION 的情况, $num_i = num_{i-1} + 1$.

(3). 第 i 次操作是 TABLE-INSERTION, 未发生扩张; 由于没有扩张, 因此$size_i = size_{i-1}$, $num_i = num_{i-1} + 1$, $c_i = 1$.

情况一: $f_{i-1} < 1/2$, $f_i < 1/2$并且未扩张. $\phi_i = \frac{1}{2}size_i - num_i$, $\phi_{i-1} = \frac{1}{2}size_{i-1} - num_{i-1}$,

$\alpha_i = c_i + \phi_i - \phi_{i-1} = 1 + \left(\frac{1}{2}size_i - num_i\right) - \left(\frac{1}{2}size_{i-1} - num_{i-1}\right) = 0.$

情况二: $f_{i-1} < 1/2$, $f_i \geq 1/2$并且未扩张. $\phi_{i-1} = \frac{1}{2}size_{i-1} - num_{i-1}$, $\phi_i = 2num_i - size_i$,

$\alpha_i = c_i + \phi_i - \phi_{i-1} = 1 + (2num_i - size_i) - \left(\frac{1}{2}size_{i-1} - num_{i-1}\right) = 3 + 3num_{i-1} - \frac{3}{2}size_{i-1} < 3.$

情况三: $f_{i-1} \geq 1/2$, $f_i > 1/2$并且未扩张. $\phi_i = 2num_i - size_i$, $\phi_{i-1} = 2num_{i-1} - size_{i-1}$.

$\alpha_i = c_i + \phi_i - \phi_{i-1} = 1 + (2num_i - size_i) - (2num_{i-1} - size_{i-1}) = 3.$

(4). 第 i 次操作是 TABLE-INSERTION, 发生扩张. $f_{i-1} = 1$, $f_i > \frac{1}{2}$, $size_i = 2size_{i-1}$, $c_i = num_i$, $\phi_i = 2num_i - size_i$, $\phi_{i-1} = 2num_{i-1} - size_{i-1}$. $\alpha_i = c_i + \phi_i - \phi_{i-1} = num_i +$

$(2num_i - size_i) - (2num_{i-1} - size_{i-1}) = 3.$

2、假设二元计数器初始时有 b 个 1，而不是全 0. 给定$n = \Omega(b)$，说明执行 n 个自增操作的代价为 O(n).

解：设置势能为二元计数器中 1 的个数，由于初始状态下 1 的个数$s_0 > 0$，这样的势能设置方法不能保证$\phi_i \geq \phi_0$，即平摊代价之和$\sum_1^n \alpha_i = \sum_1^n (c_i + \phi_i - \phi_{i-1}) = \sum_1^n c_i + \phi_n - \phi_0$不再是真实代价之和$\sum_1^n c_i$的上界，但是上述数学关系依旧是满足的。因此可以得到

$$\sum_1^n c_i = \sum_1^n \alpha_i - \phi_n + \phi_0.$$

根据势能设置方法得到$\sum_1^n \alpha_i = 2n$，此处与课堂上讲授的方法一致，但是要注意这里的$2n$是平摊代价之和的上界。

因此，$\sum_1^n c_i = \sum_1^n \alpha_i - \phi_n + \phi_0 \leq \sum_1^n \alpha_i + \phi_0 = 2n + b.$

当$n = \Omega(b)$时，显然可以得到$2n + b = O(n)$. 注意，如果$b = \Omega(n)$，则$2n + b = O(b)$.

3、用两个栈实现一个队列，使得 ENQUEUE 和 DEQUEUE 操作的平摊代价都是 O(1).

解：进队列操作将元素压入栈 A，出队列操作将栈 B 中栈顶元素弹出，若栈 B 中无元素，则将 A 中所有元素依次弹出并压入栈 B.

使用会计方法，进队列操作代价 4，其中 1 支付将元素压入 A 的代价，1 支付该元素从 A 中弹出的代价，1 支付该元素压入 B 的代价，1 支付该元素弹出 B 的代价；出队列操作平摊代价 0.

4、设计一个数据结构支持由整数组成的动态多重集合 S（包含可能重复整数的集合）中的操作：（1）Insert(S, x)将整数 x 插入 S 中；（2）Delete-Larger-Half(S)删除 S 中最大的$\lceil |S|/2 \rceil$个整数。对于任意由 Insert 和 Delete-Larger-Half 组成的长度为 m 的操作序列，要求其总体代价是 O(m)，并且可以在 O(|S|)代价内输出 S 的所有元素。

解：用一个链表存储所有数据，每个节点存储一个整数。Insert 操作在链表末尾添加一个节点，存储新插入的整数；Delete-Larger-Half 操作首先将链表中元素拷贝到一个数组中，该数组大小为当前链表中元素的数量，然后使用 selection 算法查找中位数，然后并遍历所有元素查看有多少个元素与中位数相同，以此断定需要删除多少个与中位数相同的整数和所有比中位数大的整数才能完成 Delete-Larger-Harf，随后遍历链表中的元素，删除比中位数大的元素，并删除相应数量的中位数元素。这个操作的代价为 O(n)，其中 n 是执行 Delete-Larger-Half 操作时链表中元素的数量。不妨设 Delete-Larger-Half 的代价不超过 cn (由其代价为 O(n)可得存在常数 c). 使用会计方法，Insert 的平摊代价设置为(1+2c)，其中 1 用于支付在链表中插入元素的代价 1，2c 附加新插入的元素上，当删除$\lceil n/2 \rceil$个元素时，这些元素上的代价共计不少于 cn，足以支付 Delete-Larger-Half 的代价（该代价不超过 cn）；Delete-Larger-Half 的平摊代价为 0。任意时刻，平摊代价之和与真实代价之和的插值不小于链表中元素数量的(1+2c)倍，一定非负。