§8.2 会计方法

在平摊分析的会计方法中，我们给不同的操作赋予不同的充值，这些充值多于或少于实际代价。我们对每个操作的充值称为该操作的平摊代价。当一个操作的平摊代价超过它的实际代价时，这个差被赋予该数据结构的某个特定对象作为信贷。该信贷在后面某个操作的平摊代价小于它的实际代价时，可用来支付该操作的代价。

我们可以把平摊代价看作两部分，一部分是实际代价，另一部分是储存起来的信贷。这与合计方法非常不同。在那里，每个操作的平摊代价是相同的。

我们必须小心地选择每一个操作的平摊代价。一系列操作的总的平摊代价必须大于等于实际总的代价（上界）。同时，一系列操作的平摊代价要尽可能精确(接近实际代价)。

一．栈操作

我们再次考虑8.1节的栈操作的例子，在该例中，实际代价如下：

Push : 1

Pop : 1

Multipop : min(k, s)

其中，k是Multipop的参数，要弹出k个栈顶元素，而s是当时栈中的元素个数。

现在我们给每一个操作赋予一个平摊代价。

Push : 2

Pop : 0

Multipop : 0

对于压入栈的每一个元素，我们给它2个单位的平摊代价，其中1个单位的代价用于压入操作(Push操作)，另一个单位的代价作为充值保存在该压入的元素中。当该元素被弹出时，用于支付它的实际代价。

从而执行一系列n个Push, Pop, Multipop的平摊代价一定是实际代价的上界。它等于这一系列操作中，压入栈的元素个数的2倍。

二．二进制增量计数器

再考虑上一节的另一个例子，二进制增量计数器。初始时，该计数器中的值为0。

每次该计数器的值加1，其实际代价为其中的二进制位变化的次数。

我们给每一位，由0变1时，2个单位的平摊代价，而每一位由1变为0时，平摊代价为0。

当每一位由0变1时，我们赋予2个单位的平摊代价，其中1个单位的代价支付本次变化的实际代价，而另1个单位的代价，用于支付当该位由1变为0时的代价。

回顾增量算法：

Increment(A)

1. ;
2. while do

BEGIN

1. ;
2. ;

END;

1. if then
2. ;

从最低位开始遇到1就变成0，直到遇到第一个0将其变为1，剩下的不变

可以看出，A每次加1，其平摊代价为2。从0加到n，平摊代价总和为2n。

§8.3 势方法

势方法用前面操作预付的势能来支付后面操作的代价，该势能是存在于整个数据结构中，这与会计方法将充值存于数据结构中的某个对象中不同。

一．势方法的工作原理

我们从一个初始数据结构出发，执行一系列n个操作。对的每个，第次操作作用于数据结构，得到数据结构，其**实际代价**为。

势函数把数据结构映射到一个实数，它表示的势。而**平摊代价**用势函数定义如下：

次操作的平摊代价

如果我们能定义势函数，使得，则总的平摊代价

是总的实际代价的上界。

在实际中，我们常常定义，而，对所有成立。

直观上，如果是正数，那么第操作，平摊代价表示超额充值；如果次操作，平摊代价表示不足充值。

二．栈操作

我们定义势函数为栈中保存的元素个数。初始时，栈为空栈，故。由于栈中的元素个数始终是非负的，故

。

由(8.2)式，我们始终有次操作的平摊代价是实际代价的上界。现在我们来计算栈中各种操作的平摊代价：

设第次操作为Push操作，中有个元素，那么

。

由(8.1)式，**Push的平摊代价**为：

。

设第次操作为Multipop(S, k)（弹出k个元素），且，为栈S中当前的元素个数。其实际代价为。故

因此，Multipop的平摊代价为

。

类似地，Pop代价也为0。

因此，三种操作的平摊代价均为O(1)。执行n次操作的总的平摊代价为O(n)。又因为，我们已证，故n次操作的平摊代价是实际代价的上界，故最坏情况下，实际代价为O(n)。

三．二进制增量计数器

我们定义势函数为计数器中值为1的位数。

我们来计算一次增量操作的平摊代价。设第次增量操作将位变为0(再将1位变为1)。故第次操作实际代价为。设为第次操作，计数器中1的个数，那么，故势函数的差为

平摊代价为:

。

如果计数器初始值为0，则，由于，对所有成立，故由(8.2)式，n次增量操作的平摊代价是实际代价的上界，故最坏情况下，n次增量操作时间复杂度为O(n)。

当计数器初值不为0时，用势函数方法，可证n次增量操作时间复杂度仍是O(n)。

设初始时，是计数器中1的个数。n次增量操作后, 计数器有个1，其中。(8.2)式可改写为：

我们有。因为且，n次增量操作的实际代价为

因为，如果执行次增量操作，则实际代价为，无论计数器中初值等于多少。

作业19：

1. 对一个栈执行一系列栈操作，假设该栈中元素个数不会超过k。假设每执行k个操作，就执行一次将整个栈备份一次的操作。证明：执行n次栈操作(包括备份操作)的时间复杂度为O(n)。（提示：赋予每个栈操作适当的平摊代价）。（会计方法）

2. 设栈中初始有()个元素，最后有个元素。执行一系列个Push, Pop和Multipop操作的总代价是多少？（势方法）