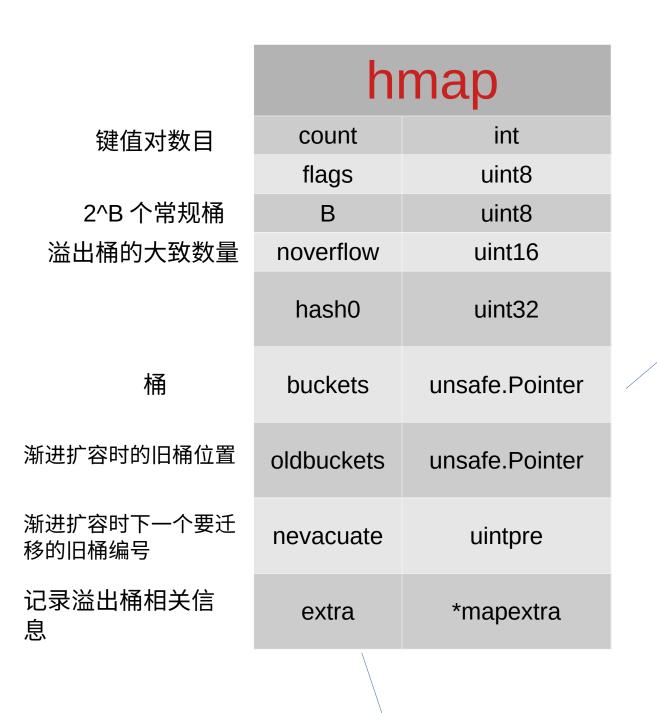
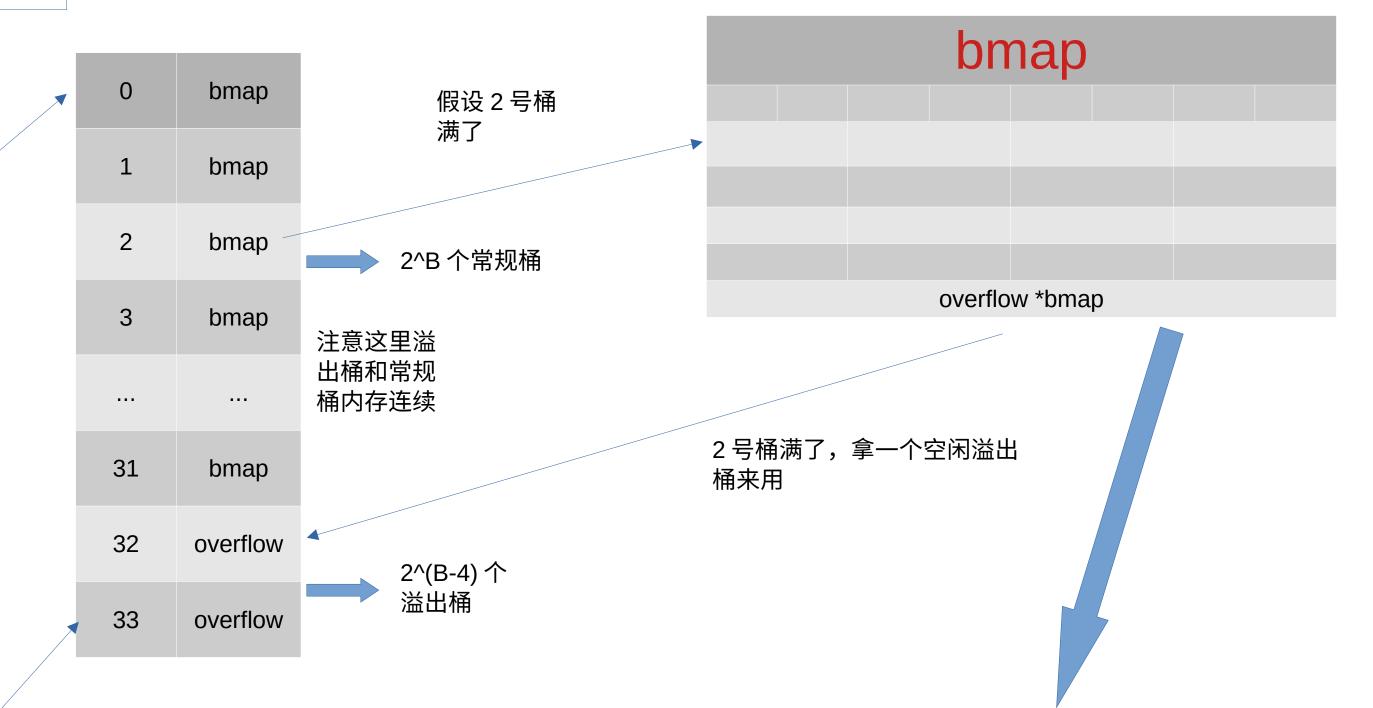
这里是用 hash 的低 8 位选桶采用的是位运算,桶的数量必须是 2 的整数次幂个,因此其选桶过程如下,加入有 4 个桶,则其桶下标为 (hash值)&(0011)

这里假设

B=5

这里桶数量要2的整数次幂是为了防止有一些桶绝对不会被选中的情况。





已经使用的溢出桶

扩容时旧桶用到的溢 出桶的地址

下一个空闲溢出桶

mapextra				
overflow	*[]*bmap			
oldoverflow	*[]*bmap			
nextOverflow	*bmap			



> $B \le 15$ noverflow $\ge 2^B$ $B \ge 15$ noverflow $\ge 2^15$

创建和旧桶一样 多数量的桶,再 把旧桶数据迁移

什么时候会触发等量 扩容,即 map 中很 多 key-value 被删除 时就可能会

bmap h1 h2 h3 h4 h5 h6 h7 h8 k1 k3 k2 k4 k7 k8 k5 k6 v2 v3 v1 **v**4 v6 v7 v5 ν8 overflow *bmap

桶

tophash

对应 key 哈希值的高 8 位

hash的低8位用

来确定是哪个桶,

哪个 key

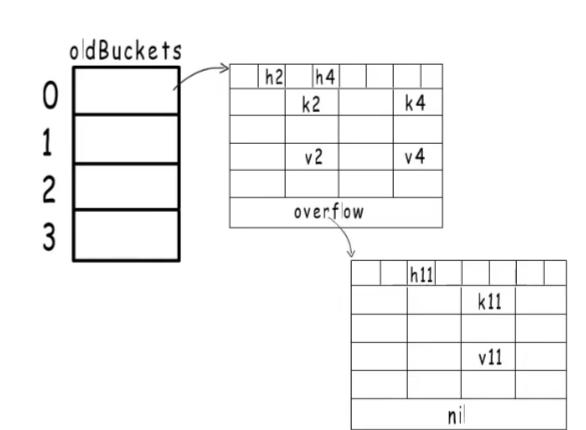
高8位用来确定是

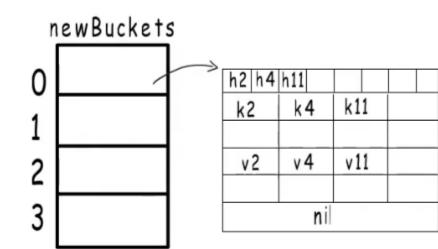
8 个 key 放前 面, 8 个 value 放后面,内存更 紧凑

当一个桶满了,如果还有溢出桶可用,那么该 指针会指向溢出桶,这是为了减少扩容次数而 准备的。

bmap					
h9					
k9					
v9					
nil					

注意扩容迁移时, map 不会立刻迁移数据,而是访问旧桶数据时,才迁移到新桶。并且迁移时不会直接删除旧桶,而是解除对其的引用,然后利用 GC 删除。





这里通过等量扩容,使得内存更加 紧凑,减少溢出桶的使用。 原视频链接