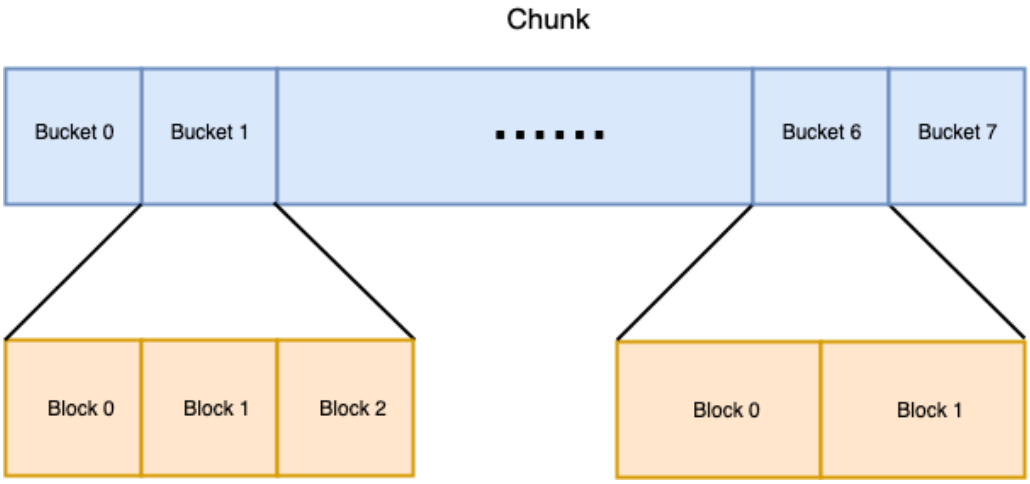
PowerTwoBucketAllocator

1. Introduction

PowerTwoBucketAllocator is a simple buddy-like allocator used for small space. It always allocates a block of power-of-two size to satisfy a space request. For example, if a space of 54 bytes is requested, a block of 64 bytes will be allocated to it.

2. Design



Picture 1. Relationship of chunk, bucket and block

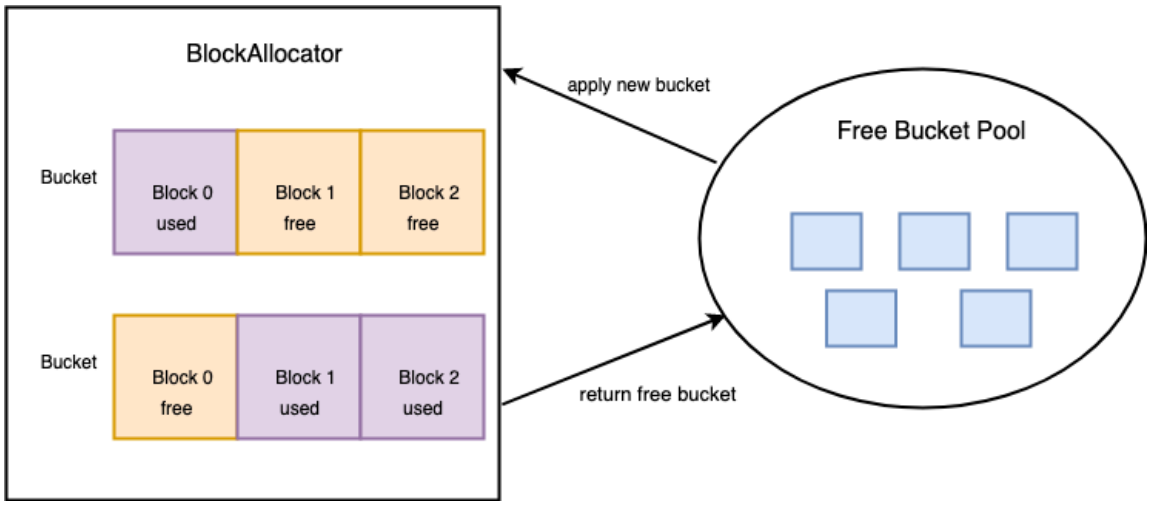
Relationship of chunk, bucket and block is illustrated in picture-1. A chunk will be divided into multiple buckets of the same size. Each bucket is further divided into blocks. Blocks in a bucket is the same size, but blocks in different buckets may have different size. Offset of a bucket in the chunk is (bucketIndex * bucketSize), and offset of a block in the chunk is (bucketOffset + blockIndex * blockSize). Note that a bucket can be divided into blocks with another size when it's reused after being freed.

Block size must be power of two, which is no less than MIN_BLOCK_SZIE, and no more than bucket size. So number of possible values for block size is (log2(bucketSize) – log2(MIN_BLOCK_SIZE) + 1). For each possible size, there will be a BlockAllocator to be responsible for allocating blocks from buckets. Index of BlockAllocator for blockSize is (log2(blockSize) – log2(MIN_BLOCK_SIZE)). For example, if MIN_BLOCK_SIZE is 32, and bucketSize is 128, there will be 3 possible values for block size: 32/64/128 thus

relatively 3 BlockAllocator. Index of BlockAllocator for 32 is 0, for 64 is 1 and for 128 is 2.

BlockAllocator mainly do three things, which is illustrated in picture-2

- Maintains a collection of buckets that are used,
- Applys for new bucket from free bucket pool when there is no free block to allocate in the collection
- Returns free bucket to free bucket pool



Picture 2. BlockAllocator

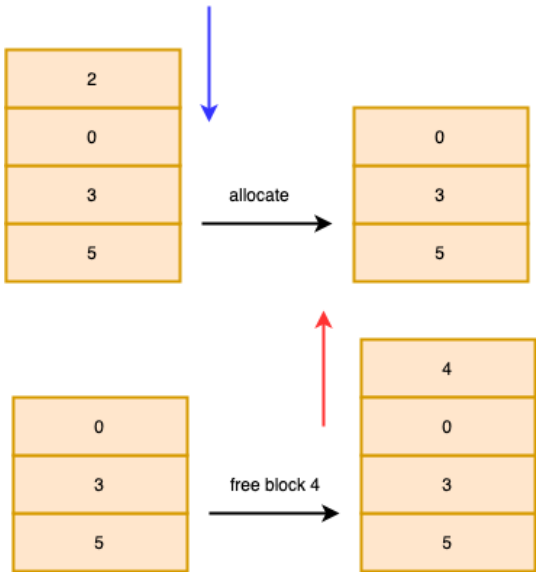
Steps to allocate a space from chunk are as follows

1. Round up the space size to power of two, that's the size of block to allocate
2. Calculate the index of BlockAllocator for the block size, that's (log2(blockSize) – log2(MIN_BLOCK_SIZE)), and find the BlockAllocator according to the index
3. BlockAllocator finds a bucket which has left space in it's own collection, or apply a new bucket from free bucket pool
4. Bucket allocates a block, and returns offset of the block in the chunk

Steps to free a space from chunk are as follows

1. Calculate the index of bucket according to offset of space in the chunk, that's (spaceOffset / bucketSize), and find the bucket according to the index
2. Get current block size for the bucket, and get BlockAllocator for this block size
3. Bucket calculates the block index in the bucket, that's ((spaceOffset – bucketOffset) / blockSize), and free the block
4. If bucket there is no used block, BlockAllocator will return it to free bucket pool

Bucket uses a stack to maintain free blocks, which is illustrated in picture-3. Elements on stack are indexes of blocks. When allocating a block, bucket will pop the block on top of stack; and when freeing a block, bucket will push the index of block on the top of stack. The number of free blocks is the size of stack.



Picture 3 Free blocks in bucket