**Design Document**

The whole memory has a dummy head in the front as the original K&R version did.

Dummy head -> whole data

Data is consist of blocks. Each block has 2 header: one in front and one in back. Between them is the payload area. The first 2 spaces in the payload area allocate to 2 pointers: prev and next. They point to the previous free block and the next free block respectively.

Header | (prev pointer) (next pointer) payload area | Header

The header compose of a bit and a size. The bit indicates whether this block has been allocated(1-allocated, 0-free), and the size tells us the size of this block(including 2 headers).

The total block size is always a multiple of two headers.

There are 2 strategies for finding the free block in malloc function. If using "first fit" strategy, we can find the block very fast, but may require more efforts and time to split block. If using "best fit" strategy, we can avoid fragmentation, also spare time splitting blocks in some cases, but traverse time maybe longer.

Compared with the other three implementation, this design have some advantages:

1. We can simply set the allocated flag to 0 to free the block, we don’t have to traverse the whole free list to find the insertion position of this block as K&R did. This saves some time for free function.
2. We can easily check whether this block to be freed is allocated or already been freed, whereas the K&R can’t realize such inspect.
3. With the front and back header of block, we can use cur[-1] to locate the previous block, and cur[size] to locate the next block. We can also use cur[1] to get the previous free block, and use cur[2] to get the next empty block. These features provide lots of convenience during the coalescing process. Other 3 implementations don’t provide such convenience.

This design also have some disadvantages that can be improved in future:

1. Compared with K&R, it need more space because of the two header and prev pointer.
2. The malloc function still need traverse the list to find suitable free block.

**Simple Results**

index leaks errors ops secs Kops file

1 0 0 12 0.000011 1090 short1-bal.rep

2 0 0 12 0.000015 800 short2-bal.rep

3 0 0 5694 0.005288 1076 traces/trace0.rep

4 0 0 5848 0.005033 1161 traces/trace1.rep

5 0 0 5380 0.004893 1099 traces/trace3.rep

6 0 1 14400 0.011029 1305 traces/trace4.rep

7 0 1 4800 0.003168 1515 traces/trace5.rep

8 0 1 4800 0.003092 1552 traces/trace6.rep

9 0 0 12000 0.011426 1050 traces/trace7.rep

10 0 0 24000 0.018701 1283 traces/trace8.rep

11 0 1 14401 0.010761 1338 traces/trace9.rep

**Malloc Results**

index leaks errors ops secs Kops file

1 0 0 12 0.000028 428 short1-bal.rep

2 0 0 12 0.000019 631 short2-bal.rep

3 0 0 5694 0.006989 814 traces/trace0.rep

4 0 0 5848 0.007487 781 traces/trace1.rep

5 0 0 5380 0.006561 819 traces/trace3.rep

6 0 0 14400 0.015440 932 traces/trace4.rep

7 0 0 4800 0.005363 895 traces/trace5.rep

8 0 0 4800 0.006556 732 traces/trace6.rep

9 0 0 12000 0.011350 1057 traces/trace7.rep

10 0 0 24000 0.021709 1105 traces/trace8.rep

11 0 0 14401 0.025876 556 traces/trace9.rep

**K&R Results**

index leaks errors ops secs Kops file

1 0 0 12 0.000030 400 short1-bal.rep

2 0 0 12 0.000019 631 short2-bal.rep

3 0 0 5694 0.007792 730 traces/trace0.rep

4 0 0 5848 0.007502 779 traces/trace1.rep

5 0 0 5380 0.008059 667 traces/trace3.rep

6 0 0 14400 0.014692 980 traces/trace4.rep

7 0 0 4800 0.010852 442 traces/trace5.rep

8 0 0 4800 0.010668 449 traces/trace6.rep

9 0 0 12000 0.044859 267 traces/trace7.rep

10 0 0 24000 0.087243 275 traces/trace8.rep

11 0 0 14401 0.143064 100 traces/trace9.rep

**Doubly-Linked implicit list**

index leaks errors ops secs Kops file

1 0 0 12 0.000025 480 short1-bal.rep

2 0 0 12 0.000010 1200 short2-bal.rep

3 0 0 5694 0.006371 893 traces/trace0.rep

4 0 0 5848 0.005266 1110 traces/trace1.rep

5 0 0 5380 0.005936 906 traces/trace3.rep

6 0 0 14400 0.012617 1141 traces/trace4.rep

7 0 0 4800 0.010144 473 traces/trace5.rep

8 0 0 4800 0.008248 581 traces/trace6.rep

9 0 0 12000 0.076727 156 traces/trace7.rep

10 0 0 24000 0.160861 149 traces/trace8.rep

11 0 0 14401 0.120180 119 traces/trace9.rep