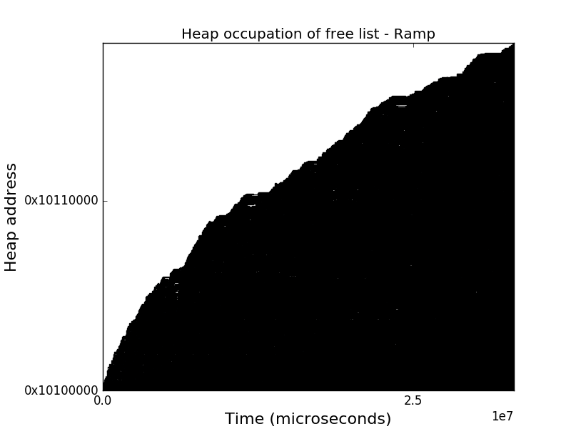
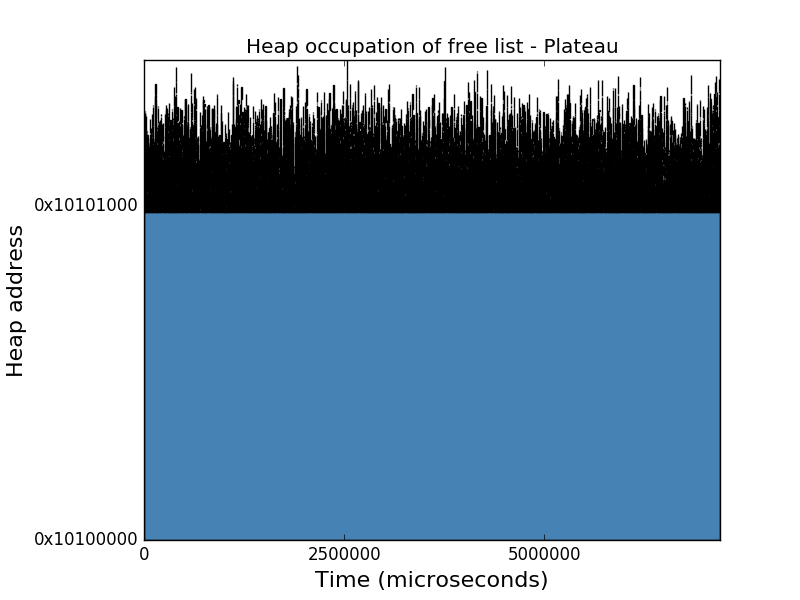
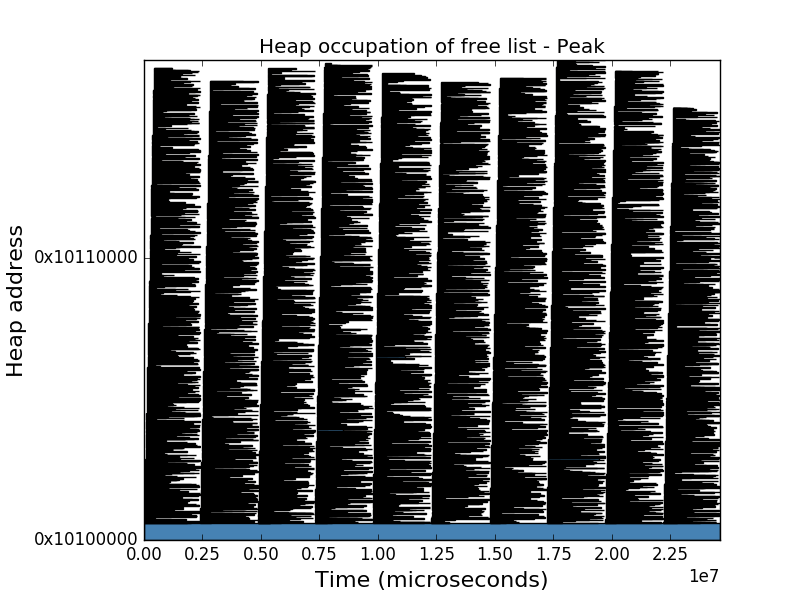
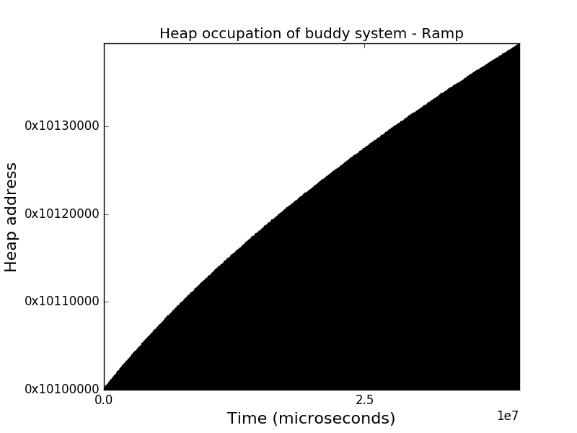
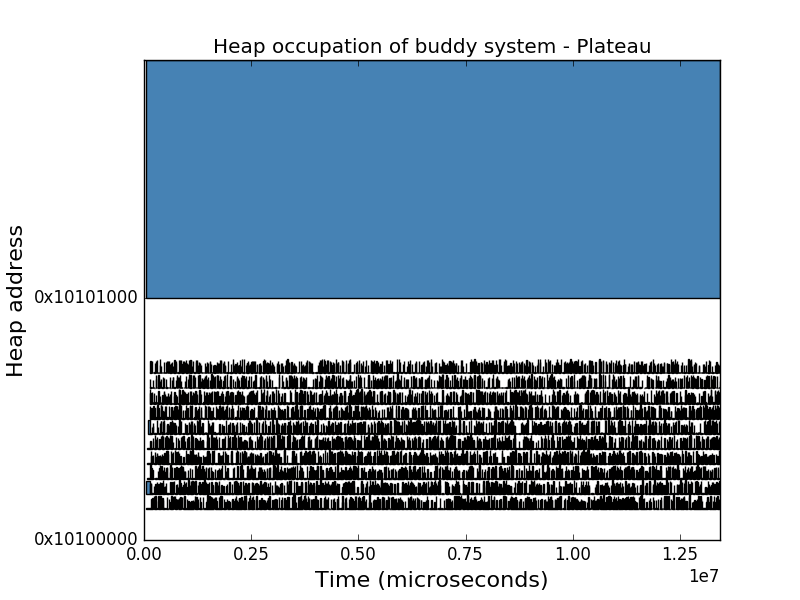
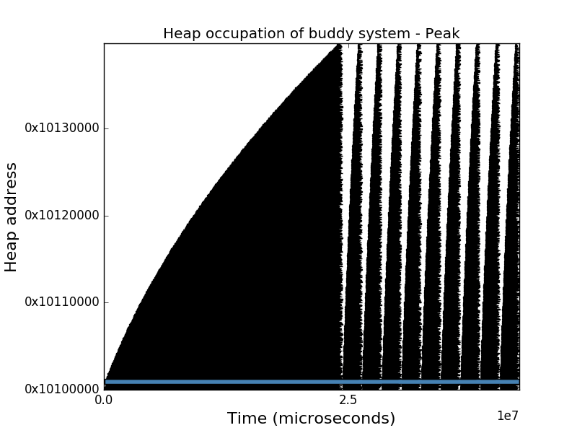
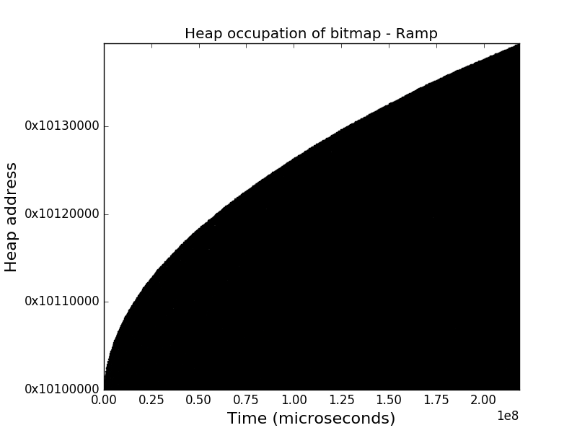
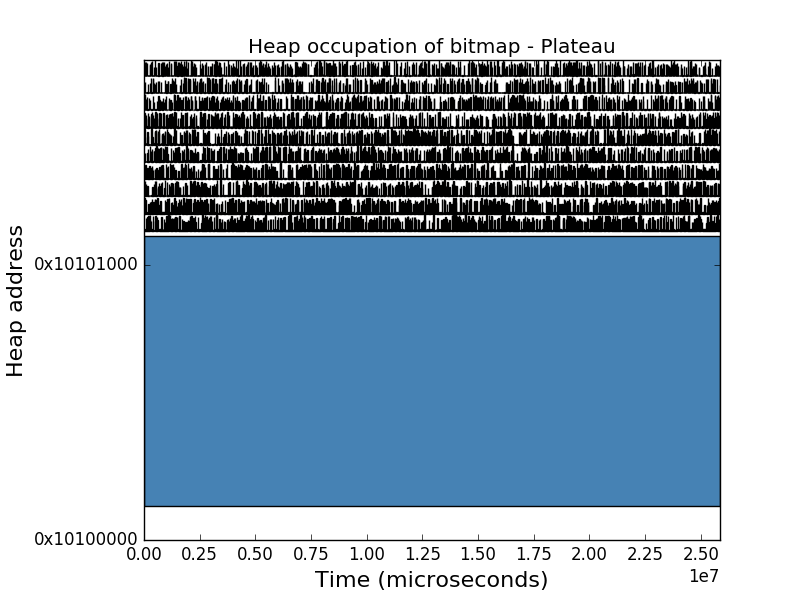
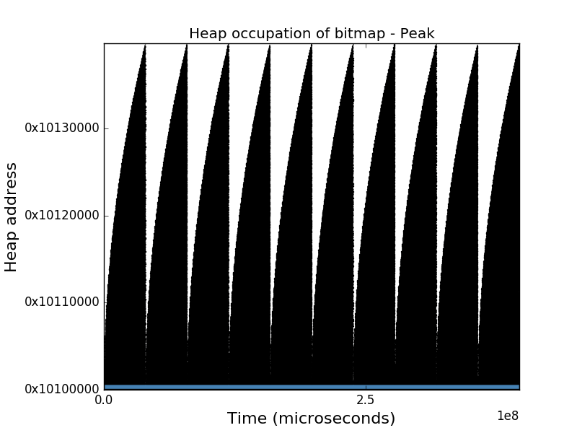
Results of Synthetic benchmarks



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Identifier | Malloc cycles | Free cycles | Outside cycles | Average frag |
| FL-Pl | 2 260 472 | 1 641 896 | 3 294 064 | 4.0% |
| FL-Ra | 17 359 953 | 12 612 828 | 3 166 199 | 19.8% |
| FL-Pe | 2 257 091 | 19 085 922 | 3 284 664 | 27.8% |
| BMP-Pl | 19 964 032 | 2 609 494 | 3 294 489 | 25.6% |
| BMP-Ra | 213 226 118 | 2 346 166 | 3 164 519 | 61.9% |
| BMP-Pe | 391 214 122 | 2 606 520 | 3 282 325 | 68.7% |
| B-Pl | 7 200 209 | 2 933 358 | 3 298 701 | 44.3% |
| B-Ra | 34 028 366 | 2 642 408 | 3 165 352 | 61.9% |
| B-Pe | 36 298 554 | 2 938 646 | 3 280 224 | 68.8% |

Parameters

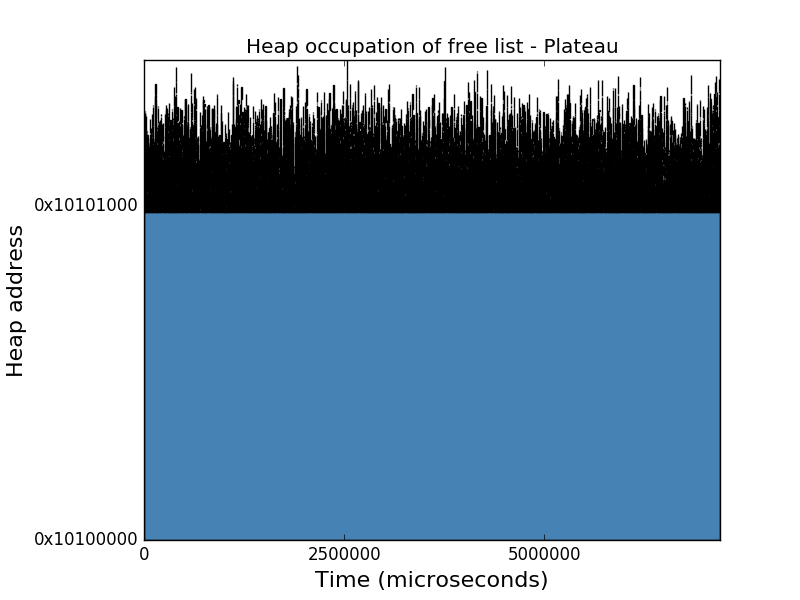
BMP Minimum grain size: 256 bytes  
Buddy system minimum grain size: 256 bytes

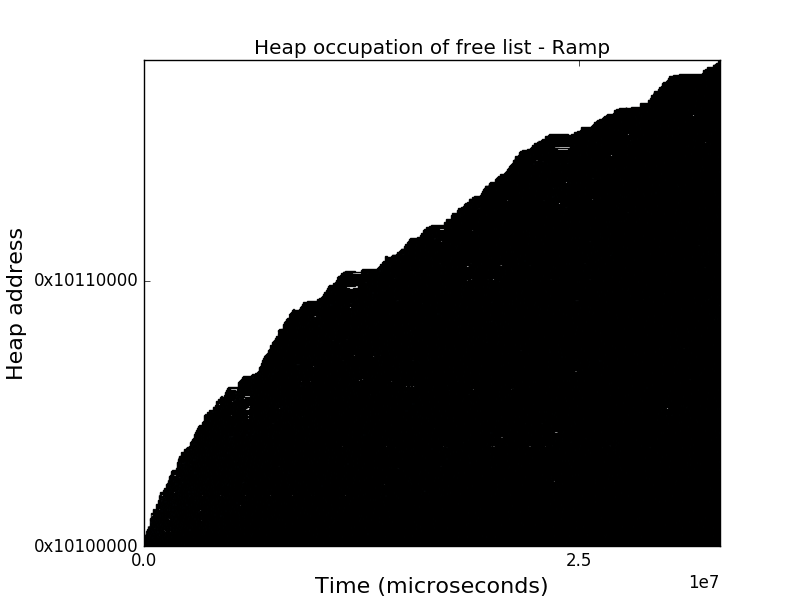
**Synthetic profiles**smallSize = 12+rand(12)  
mediumSize = 120+rand(120)  
largeSize = 4020

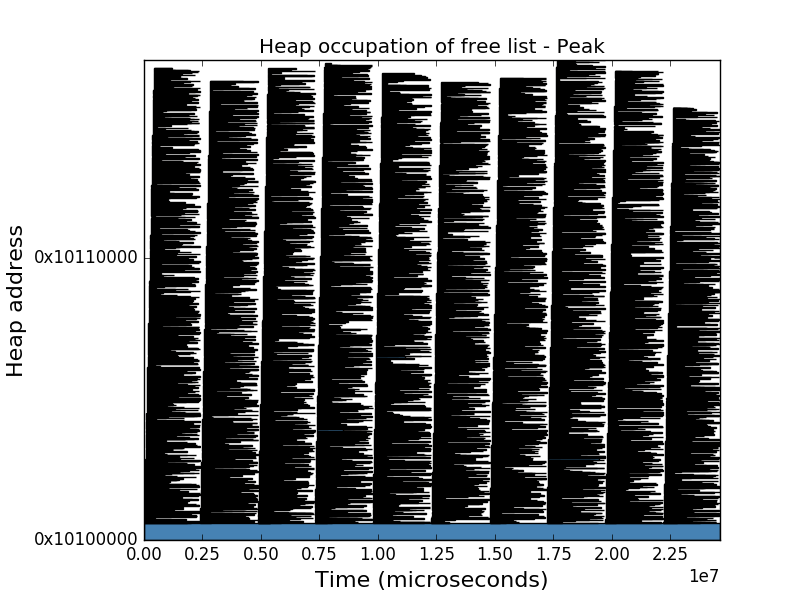
Plateau  
malloc(largeSize);  
repeat(1000,10,10);  
free(largeSize)

Ramp  
repeat(1000,10,9);

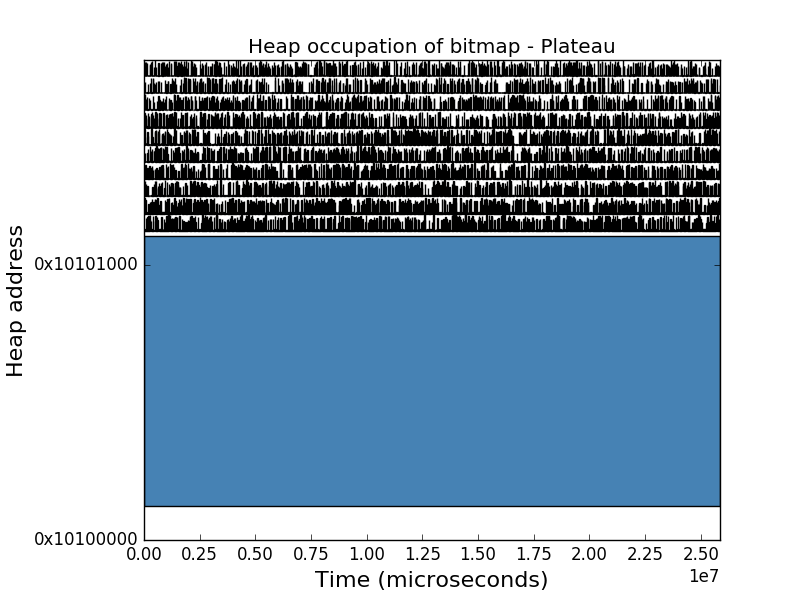
Peak  
malloc(largeSzie);  
repeat(10,1000,1000);  
free(largeSzie);

Free list

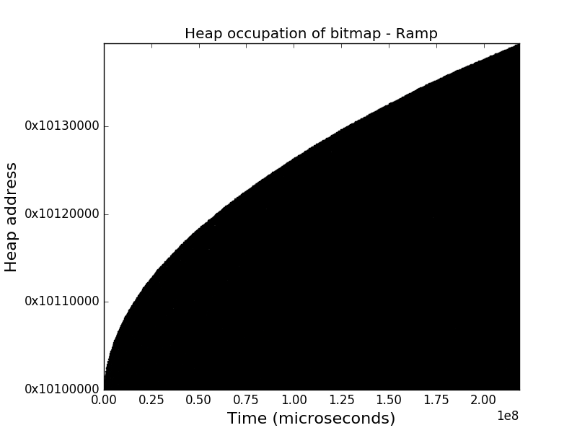
The free list handled the memory request of the plateau and the peaks excellently with only 2250 respective 2260 thousand cycles. The freeing of the plateau also qualified for an excellent mark with astonishing 1650 thousand cycles. Additionally this combination of a free list as a data structure and plateau as an access profile had the lowest fragmentation of 4%.

The ramp was where the free list struggled the most. This can be seen in the middle graph by the curvature of the ramp. During this access profile the free list needed a total of 17400 thousand cycles for malloc calls and significantly high 12600 thousand cycles for free calls. In this combination the average fragmentation was around 20%.

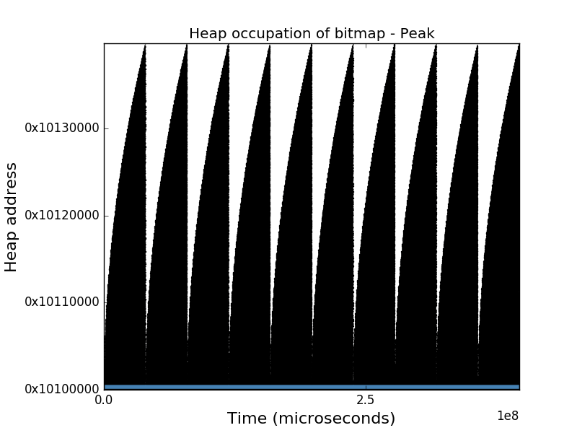
During the peaks the free list did a better job at handling the malloc calls but worse handling the free calls. The free list spent a total of 2260 and 19100 thousand cycles inside malloc respective free calls. During this combination the average fragmentation was 28%.

Bitmap

The Bitmap was overall significantly slower than the two other data structures. The plateau was the easiest access profile to handle for the bitmap. During this access profile only 20 000 and 2610 thousand cycles were spent inside malloc respective free. During the plateau the bitmap had its lowest average fragmentation with a percentage of 26 percent.

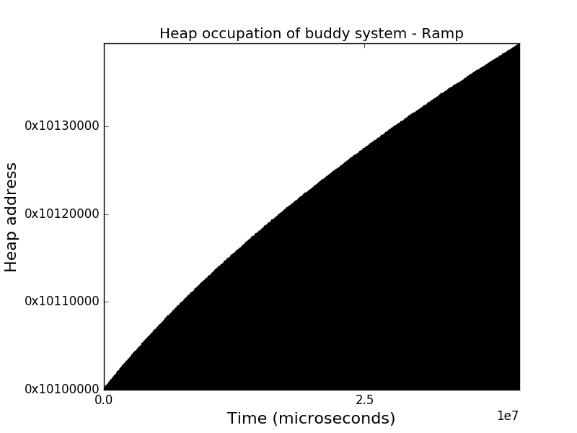
During the ramp the Bitmap did better than during the peaks but worse than during the plateau. The bitmap spent a total of 213000 thousand cycles inside the malloc function and had an average fragmentation of 62%. On the contrary, only 2350 thousand cycles were spent in the free function.

Peaks were the biggest challenge for the bitmap data structure. The amount of cycles spent inside malloc calls reached a value of 213000 thousand cycles and the average fragmentation reached 68%. On the other hand, the bitmap allocator only spent 2610 thousand cycles inside the free calls.



Buddy system

The buddy system had, as the free list and bitmap, its best result during the plateau. During this access profile the buddy did something that a first glance could seem strange. It chose to place the large sized block far from the heap base. This placement was due to the heap storing the its bitmap in the first block. The bitmap can be seen as the white space in between the heap base at 0x10100000 and the first allocations.

During the plateau the buddy system spent a total of 7200 thousand cycles inside the malloc function and a total of 2930 thousand cycles inside the free function. The average fragmentation was 44% which includes the bitmap.

Under the presence of the ramp the buddy system started acting slower. It spent 34000 thousand cycles in the malloc function and had a higher average fragmentation reaching 62%. On the contrary, the freeing was more or less unaffected. The total number of cycles spent in the free function merely reached 2650 thousand cycles.

When affected by peaks the buddy system acted slow. The total amount of cycles spent in the malloc function peaked reaching 36300 and the average fragmentation reaching almost 69%. As with the other access profiles the freeing remained quick reaching around 2940 cycles in total.

Interestingly the first peak took more time handle than the remaining ones. This is a result of the buddy system having to split blocks for the first peak but being able to reuse the splitted block for the upcoming peaks.

