Indra Bhurtel
1001542825

Summary.
1. Bench1.c allocates random integer arrays and free even elements. The random allocation and even free limits the coalesce, which results in equal amount of max-heap uses for all heap strategies.

2. Bench2.c allocates fixed numbers of arrays and free them linearly descending, our implemented coalesces happens both left and right, even though the time taken is little more this approach will try to reduce the fragmentation and will result in better heap usage, the next-fit searches for free block from last allocation, hence will miss the free block available first and will be less performant in re-uses and splits hence overall in max-heap usage as can be seen in results for particularly this case.

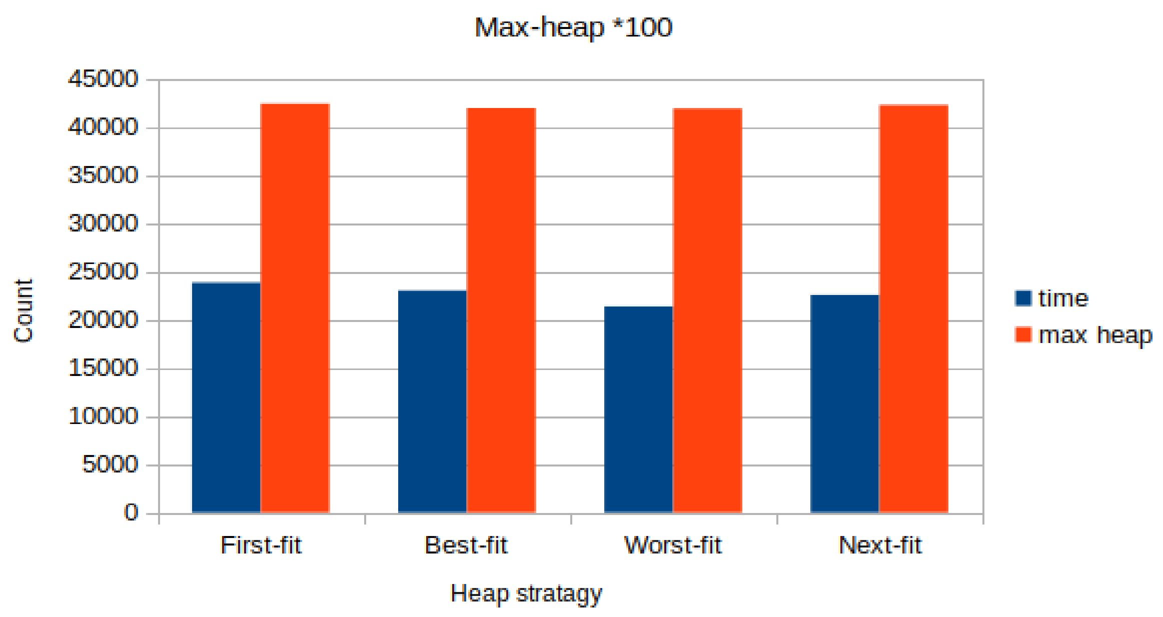

Fig 1: Bench1.c Performance

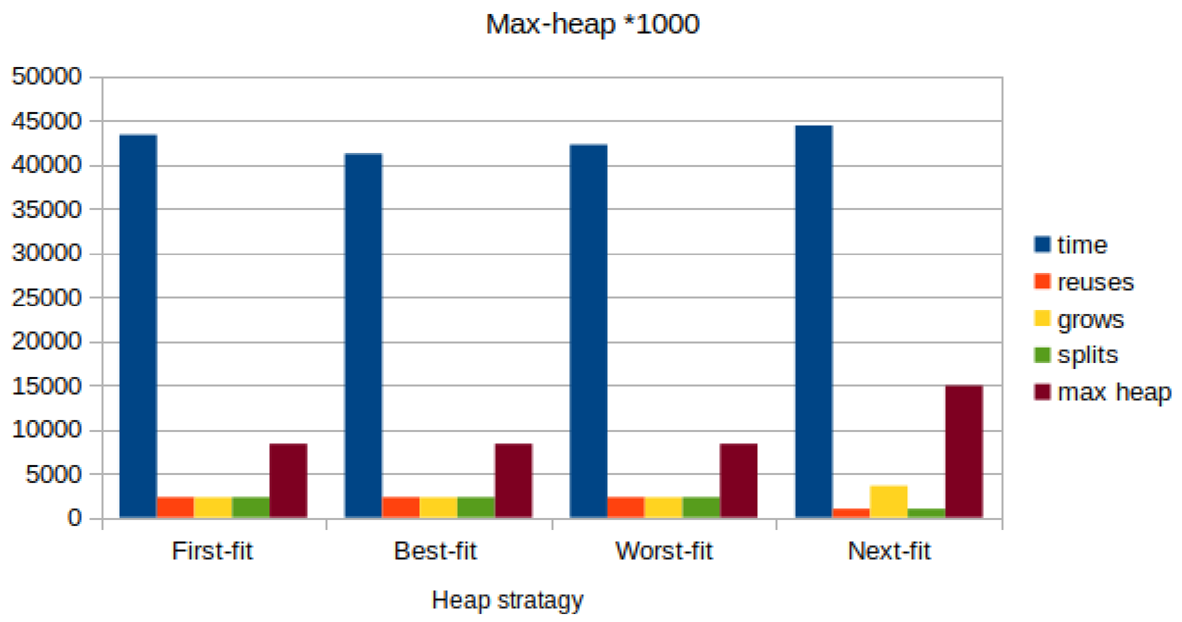

Fig 2: Bench2.c performance

Which heap management strategy does the best job of reusing free blocks? Which one is the worst?
Ans: From bench2.c numbers we can deduce that first-fit, best-fit and worst-fit all of them performs better job of reusing free blocks. Next-fit seems the worst.

Which heap management strategy requires the least amount of heap space? Which one is the
worst?
Ans: From bench2.c numbers we can deduce that first-fit, best-fit and worst-fit all of them requires the least amount of heap space. Next-fit seems the worst.

Which heap management strategy allows for the most splits? most coalescing? Least?
Ans: From benchmark performance we can deduce that First-fit allows for the most splits and coalesces and next-fit allows for least splits and best-fit allows for least coalesces.

Which heap management strategy was the fastest? Slowest?
Ans: From bench1.c performance first-fit, best-fit and worst-fit seems more comparable in “average” timings and next-fit seems the slowest as per Bench2.c.

• Consider the benchmark programs.
• Which one requires the most mallocs? Frees?
- Bench2.c needs the most mallocs and frees.
• Which one requests the most amount of space?
- Bench2.c requests the most amount of space.
• Which one requires the largest heap?
- Bench2.c needs the most amount of space.

Analysis: A discussion of the following questions:
• Which heap management strategy suffers the most from fragmentation (what type)?
- The best-fit strategy suffers from most fragmentation as we are trying to allocate block which is near to the requested size and then during the split the new generated block size may not be big enough to be used in next request. Hence create most fragments.

• Which heap management strategy is the best?
- First-fit seems fast enough as it has worst case run time of O(n) (compared to best-fit and worst-fit which has running time O(n) always) and the fragmentation is also less compared to best-fit.

• You should provide evidence for your choices.

Bench1.c results	Bench2.c results
First-fit	First-fit
Total Time spent: 23856.000000 clocks.	Total Time spent: 43398.000000 clocks.
heap management statistics	heap management statistics
mallocs: 2050	mallocs: 4610
frees: 513	frees: 4609
reuses: 149	reuses: 2311
grows: 1901	grows: 2299
splits: 146	splits: 2310
coalesces: 0	coalesces: 4607
blocks: 2047	blocks: 2
requested: 4242676	requested: 15991808
max heap: 4242076	max heap: 8364032
reallocs: 0	reallocs: 0
callocs: 0	callocs: 0
Best-fit	Best-fit
Total Time spent: 23022.000000 clocks.	Total Time spent: 41225.000000 clocks.
heap management statistics	heap management statistics
mallocs: 2049	mallocs: 4610
frees: 513	frees: 4609
reuses: 148	reuses: 2311
grows: 1901	grows: 2299
splits: 146	splits: 2310
coalesces: 0	coalesces: 4607
blocks: 2047	blocks: 2
requested: 4198632	requested: 15991808
max heap: 4198036	max heap: 8364032
reallocs: 0	reallocs: 0
callocs: 0	callocs: 0
Worst-fit	Worst-fit
Total Time spent: 21378.000000 clocks.	Total Time spent: 42287.000000 clocks.
heap management statistics	heap management statistics
mallocs: 2050	mallocs: 4610
frees: 513	frees: 4609
reuses: 147	reuses: 2311
grows: 1903	grows: 2299
splits: 146	splits: 2310
coalesces: 0	coalesces: 4607
blocks: 2049	blocks: 2
requested: 4194676	requested: 15991808
max heap: 4194088	max heap: 8364032
reallocs: 0	reallocs: 0
callocs: 0	callocs: 0
Next-fit	Next-fit
Total Time spent: 22579.000000 clocks.	Total Time spent: 44422.000000 clocks.
heap management statistics	heap management statistics
mallocs: 2050	mallocs: 4610
frees: 513	frees: 4609
reuses: 149	reuses: 1024
grows: 1901	grows: 3586
splits: 146	splits: 1023
coalesces: 0	coalesces: 4607
blocks: 2047	blocks: 2
requested: 4228968	requested: 15991808
max heap: 4228368	max heap: 14943232
reallocs: 0	reallocs: 0
callocs: 0	callocs: 0