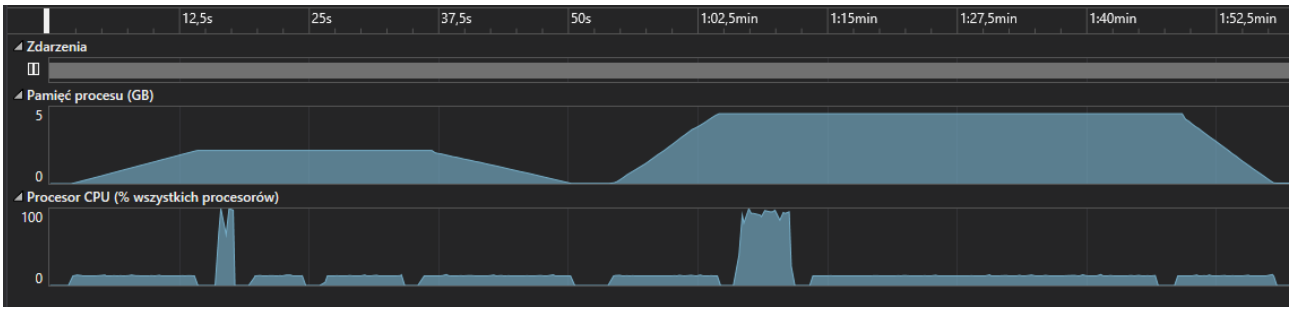
My allocator gets two allocator: one for internal purpose, and one for allocating memory pool.
User can allocate buffer of any size, it works with multi-threading, and is memory saving.
My allocator doesn't implement security like canaries, address randomisation so buffers are dense packed (cache friendly), and predictable (CPU can detect access pattern and read memory in burst mode), at cost of lower security.

Building std::forward_list with 128M nodes, 16 bytes per node:

1. Using my allocator (process memory usage ~2.0GB):
    1. building list (12135ms)
    2. iterating 16 times with 8 threads (1472ms)
    3. iterating 16 times with 1 threads (4898ms)
    4. getting information about alocation (7483ms)
    5. deallocation (14232ms)
2. Using default allocator (process memory usage ~4.3GB):
    1. building list (10159ms)
    2. iterating 16 times with 8 threads (5052ms)
    3. iterating 16 times with 1 threads (33210ms)
    4. deallocation (9368ms)

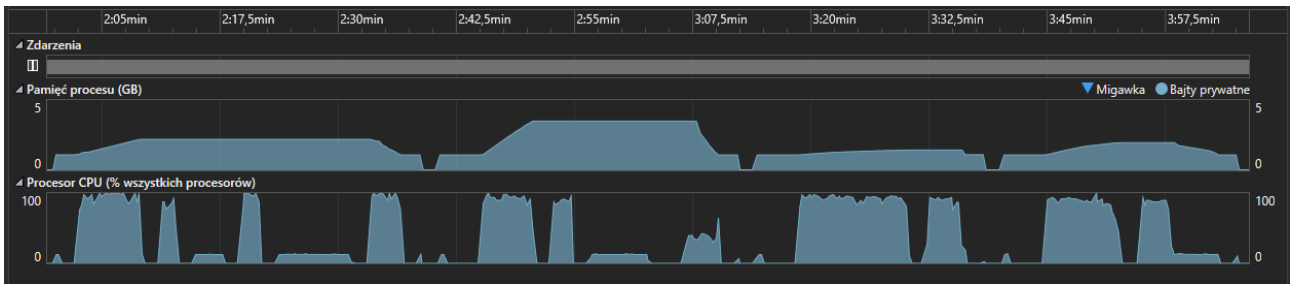


Building std::vector of 128M pointers and make 128M allocations of 8B bufer.

3. Using my allocator (process memory usage ~2.1GB with 1GB vector of pointer):
    1. building list with 8 threads (6574ms)
    2. iterating 16 times with 8 threads (1677ms)
    3. iterating 16 times with 1 threads (3006ms)
    4. getting information about alocation with 8 threads (1895ms)
    5. getting information about alocation with 1 threads (7624ms)
    6. deallocation with 8 threads (3350ms)
4. Using operator new (process memory usage ~3.3GB with 1GB vector of pointer):
    1. building list with 8 threads (5311ms)
    2. iterating 16 times with 8 threads (2135ms)
    3. iterating 16 times with 1 threads (6014ms)
    4. deallocation with 8 threads (3447ms)

Building std::vector of 128M pointers and make 1024 iteration of 64K allocation of 8B bufer in random pointer, and 64K deallocation in random pointer. Allocations.

1. Using my allocator (process memory usage ~1.4GB with 1GB vector of pointer):
    1. random allocations and deallocations with 8 threads (11491ms)
    2. deallocation with 8 threads (4035ms)
2. Using operator new (process memory usage ~1.9GB with 1GB vector of pointer):
    1. random allocations and deallocations with 8 threads (7876ms)
    2. deallocation with 8 threads (8213ms)


2:05min
2:17,5min
2:30min
2:42,5min
2:55min
3:07,5min
3:20min
3:32,5min
3:45min
3:57,5min
Zdarzenia
Pamięć procesu (GB)
Migawka
Bajty prywatne
5
0
5
0
Procesor CPU (% wszystkich procesorów)
100
0
100
0