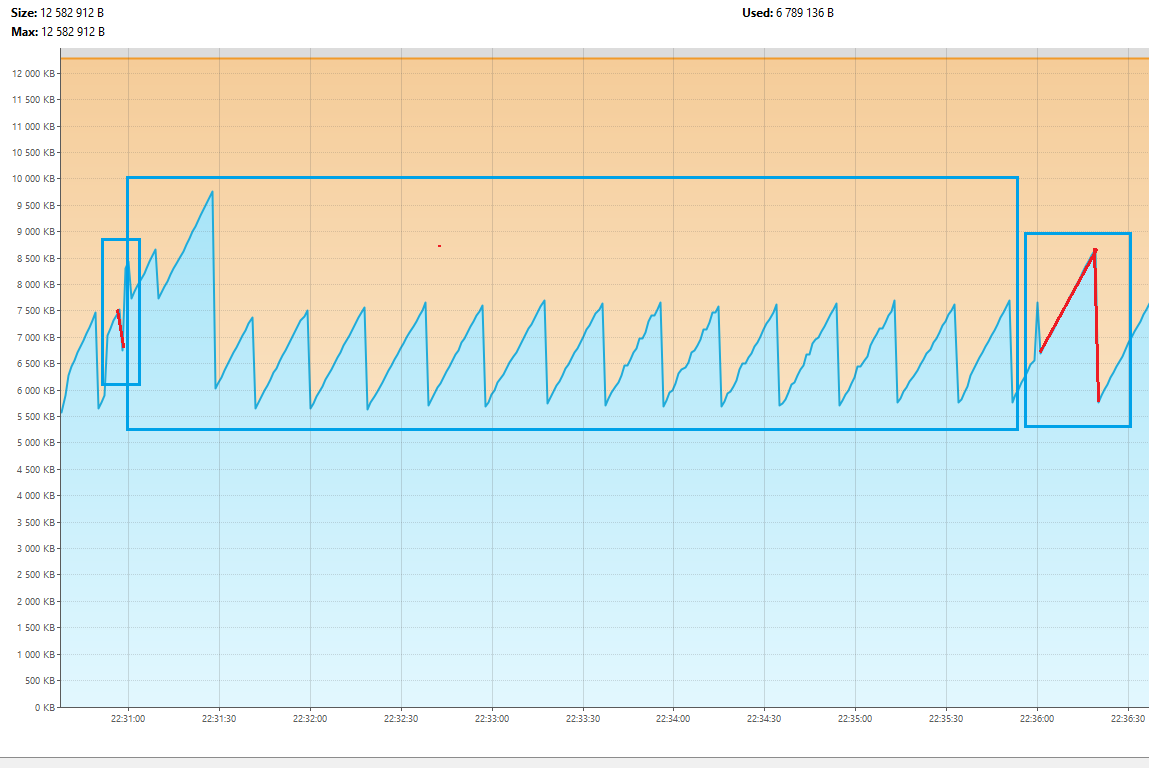
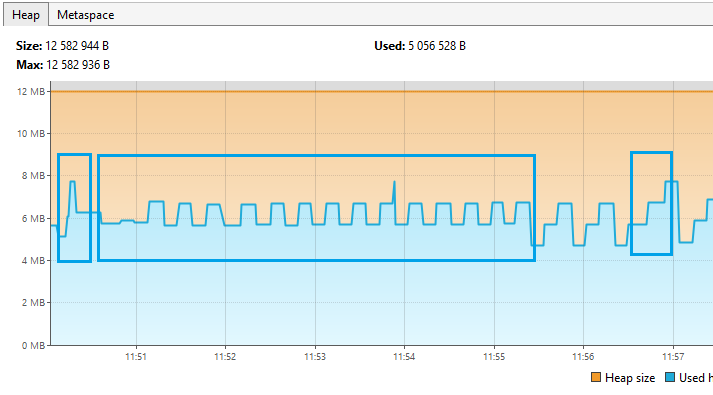
1. Использование Parallel GC

**1.1 Сортировка слиянием** – сортировка прошла за 0.1 секунды, сборка мусора на графике не запускалась, но по логу была 25 раз, процессор был нагружен на это время один раз на 11%, пиковый размер heap – 8,5Mb

* 1. **Сортировка пузырьком** сортировка прошла за 4 минуты 35 секунд, сборка мусора запускалась 38 раз, процессор был в среднем нагружен на 20% (в начале пиково один раз нагружен на 90%). Сборка мусора запускается каждый раз, когда heap заполняется на 75%. Heap максимально заполнялся на 9Mb
  2. **Сортировка вставками** сортировка прошла за 20 секунд. Сборка мусора по логу была 12 раз. Загрузка процессора на это время 25%. Heap максимально заполнялся на 8Mb



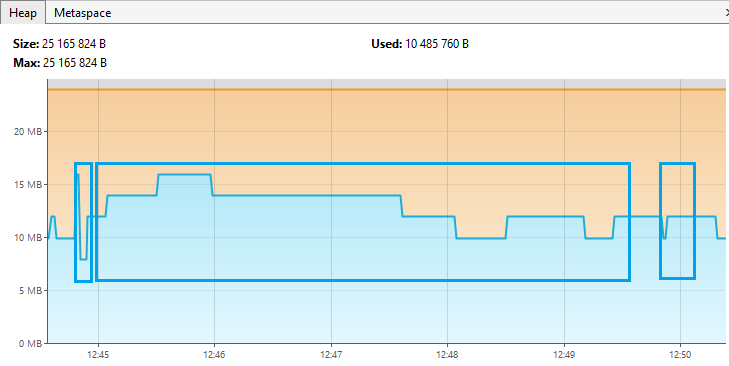
1. Использование G1:
   1. **Сортировка слиянием** – сортировка прошла за 0.1 секунды. По логу сборка мусора была 14 раз, heap максимально заполнялся на 7Mb
   2. **Сортировка пузырьком** сортировка прошла за 4 минуты 51 секундe? По логу сборка мусора была 56 раз, heap максимально заполнялся на 8Mb
   3. **Сортировка вставками** сортировка прошла за 15, 5 секунд. По логу сборка мусора была 17 раз, heap максимально заполнялся на 7Mb

****

Использование ZGC:

**3.1 Сортировка слиянием** – сортировка прошла за 0,2 секунды. По логу сборка мусора была 31 раз, heap максимально заполнялся на 24Mb (на 100%).

* 1. **Сортировка пузырьком** сортировка прошла за 4 минуты 36 секундe? По логу сборка мусора была 31 раз, heap максимально заполнялся на 16Mb из 24Mb доступных.
  2. **Сортировка вставками** сортировка прошла за 15,5 секунд. По логу сборка мусора была 4 раз, heap максимально заполнялся на 10Mb из 24Mb доступных.



Можно сделать вывод, что первые два сборщика мусора работают чаще и используют меньше памяти. Третий использует больше памяти, но работает реже.