

Laboratório de Estrutura de Dados

Primeira versão do projeto da disciplina

Comparação entre os algoritmos de ordenação elementar

João Vitor Barbosa da Silva e Klayton Marcos Veloso da Silva Júnior,

1. Introdução

Este relatório corresponde ao relato dos resultados obtidos no projeto da disciplina de LEDA e EDA que tem como objetivo ordenar dados emitidos no site do brasil.io relacionados ao COVID-19.

O sistema tem como objetivo fazer três ordenações por algoritmo e gerar suas métricas, essas ordenações são: ordenar pelo número de casos, ordenar por cidade (ordem alfabética) e ordenar pelo número de óbitos.

Ao final de cada execução do algoritmo escolhido, temos como resultado 6 arquivos .csv, 3 com as métricas para cada tipo de ordenação e os outros 3 são o documento principal ordenados pelo seu respectivo parâmetro. Vale ressaltar que no código do sistema já tem o arquivo .csv do brasil.io completo e filtrado para pegar apenas os últimos casos e os casos não repetidos.

Nos próximos tópicos, serão abordados uma descrição dos métodos utilizados para a construção e análise do sistema, os testes que foram feitos, a descrição dos ambientes utilizados para desenvolver e testar o sistema e por fim no último tópico os resultados dos testes com os gráficos e as análises necessárias.

2. Descrição geral sobre o método utilizado

O sistema foi testado em duas máquinas diferentes com o mesmo código, presente no github do projeto ([link](#)). Após rodar todos os algoritmos nas duas máquinas, foram pegos os arquivos de métrica gerados pelo sistema e criados os gráficos por meio do Pandas utilizando Python. Os gráficos estão na próxima sessão, seguidos da análise de desempenho geral.

A implementação de todo sistema foi feito com Java, o sistema funciona lendo o arquivo csv, em seguida roda o algoritmo selecionado pelo usuário, que usará a leitura do csv para ordená-lo, criando outro arquivo csv com a ordenação e com as métricas. O mesmo csv será ordenado 3 vezes, mas por critérios diferentes, primeiramente será ordenado pelo

número de casos, depois pelo número de óbito e por fim ordenar as cidades por ordem alfabética. Para cada ordenação dessa serão criadas as suas respectivas métricas, métricas essas que foram utilizadas para a construção dos gráficos da sessão 3. Os algoritmos abordados no sistema são: Selection Sort, Insertion Sort, Merge Sort, Quick Sort, Quick Sort com Mediana de 3, Heap Sort e Counting Sort.

Descrição geral do ambiente de testes

Máquina 1:

Processador: 2,3 GHz Intel Core i3-7020U

Memória: 4 GB 2667 MHz DDR4

Sistema Operacional: Linux Ubuntu

Máquina 2:

Processador: 2,3 GHz Intel Core i5 Dual-Core

Memória: 8 GB 1333 MHz DDR3

Sistema Operacional: macOS Catalina

3. Resultados e Análise

Após os testes realizados podemos concluir que a máquina e o algoritmo interferem bastante na execução do código, em seguida temos os gráficos primeiramente da máquina 1 e em seguida da máquina 2. Os gráficos serão divididos pelo tipo de algoritmo e por cada ordenação, são elas: o número de casos, número de óbitos e ordenação das cidades por ordem alfabética.

3.1 Gráficos da Máquina 1

Selection Sort:

Casos:

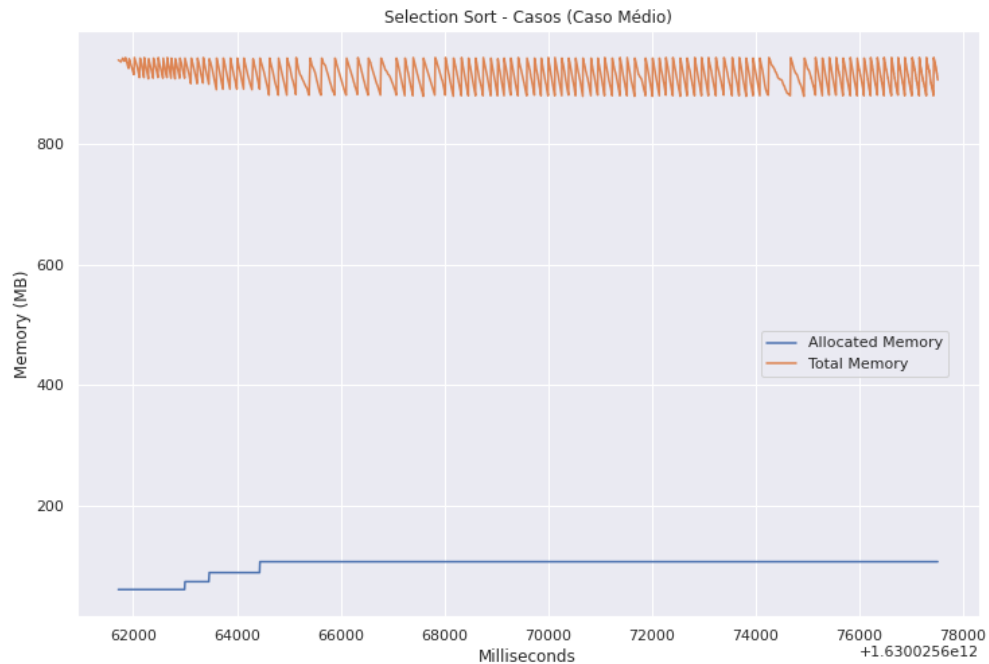


Gráfico 1: Casos Selection Sort (Caso Médio) Máquina 1.

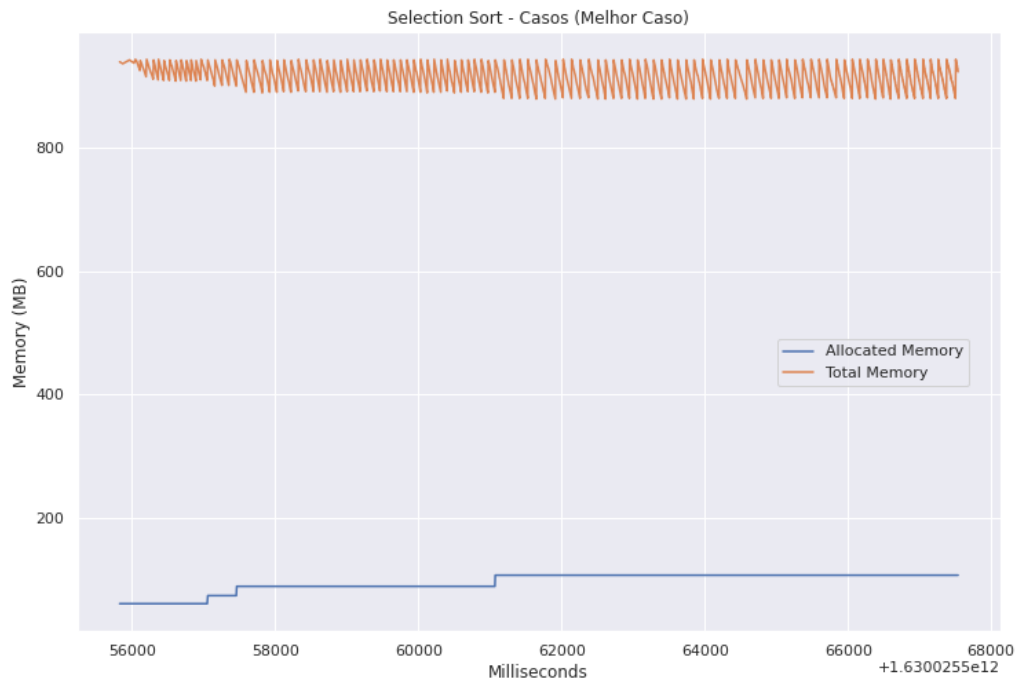


Gráfico 2: Casos Selection Sort (Melhor Caso) Máquina 1.

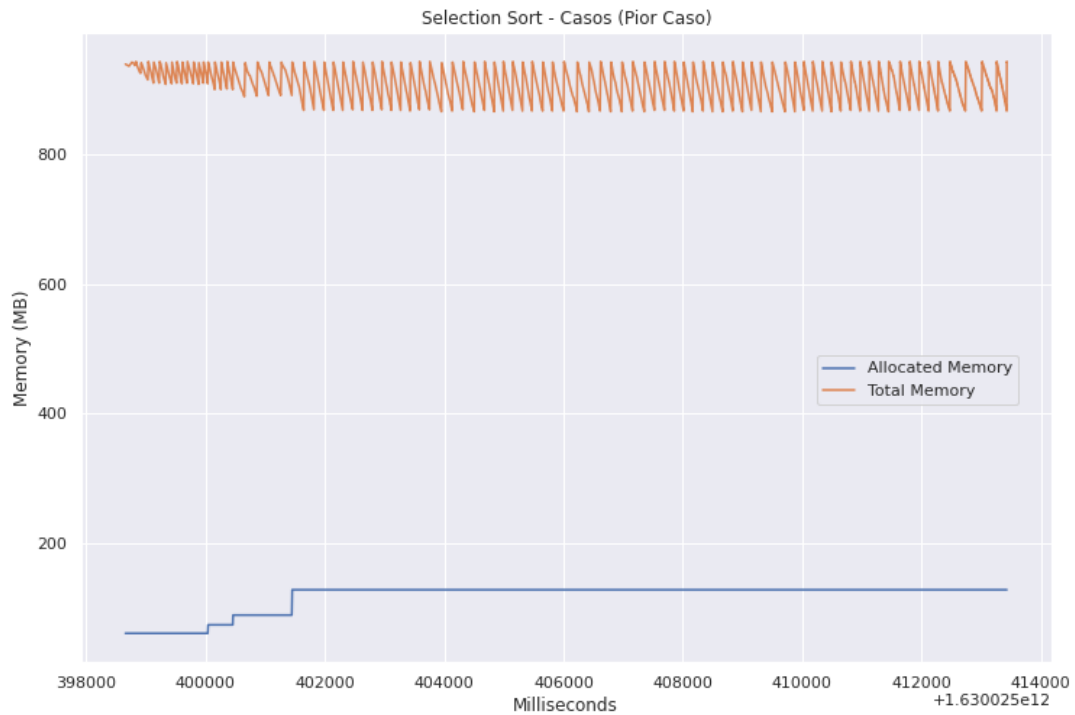


Gráfico 3: Casos Selection Sort (Pior Caso) Máquina 1.

Cidade:

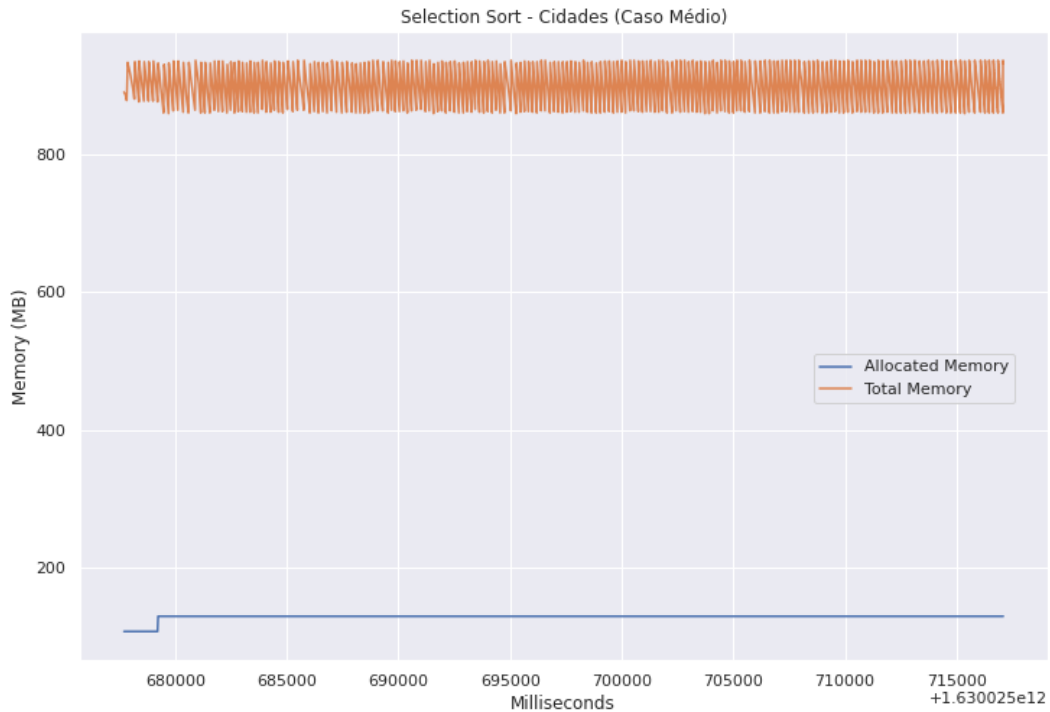


Gráfico 4: Cidades Selection Sort (Caso Médio) Máquina 1.

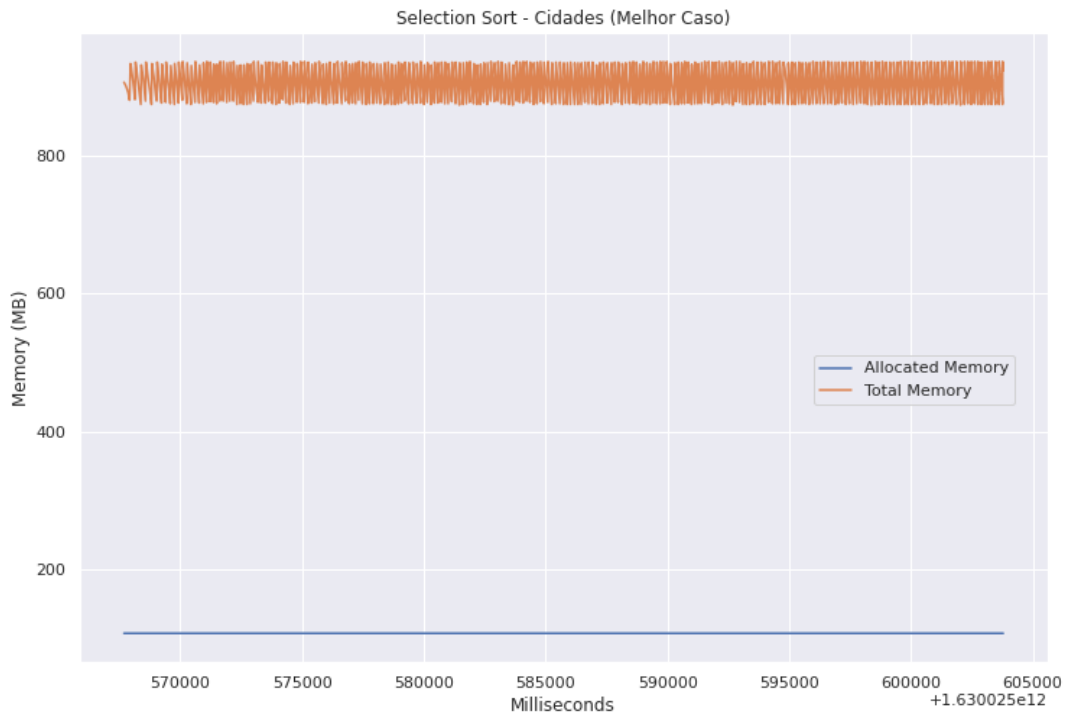


Gráfico 5: Cidades Selection Sort (Melhor Caso) Máquina 1.

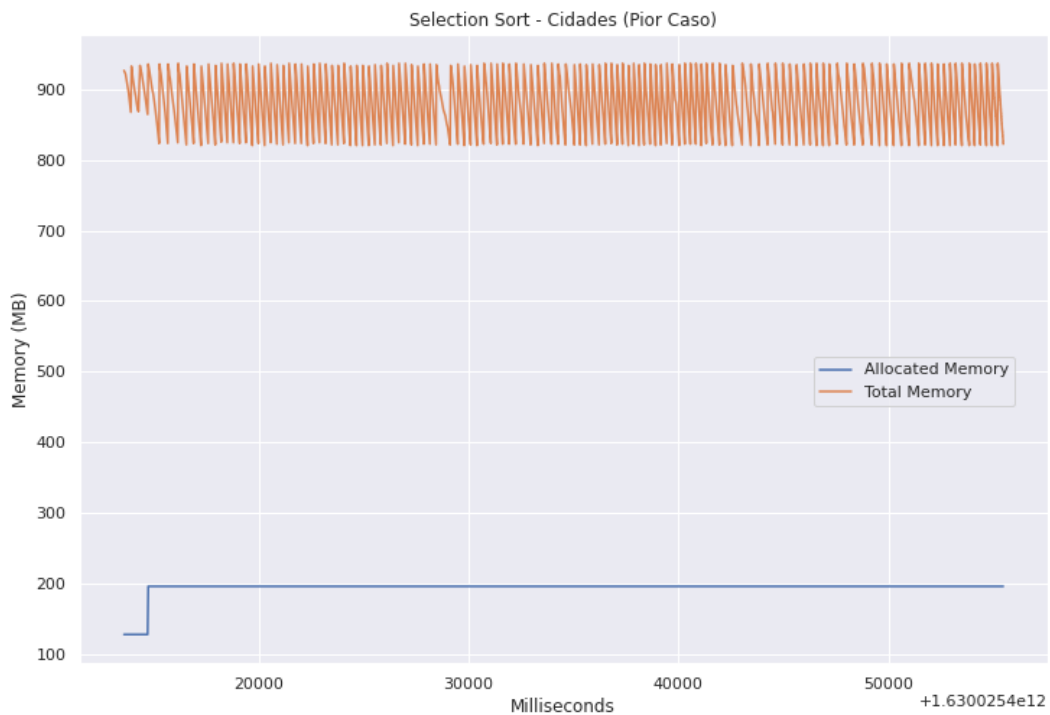


Gráfico 6: Cidades Selection Sort (Pior Caso) Máquina 1.

Insertion Sort:

Casos:

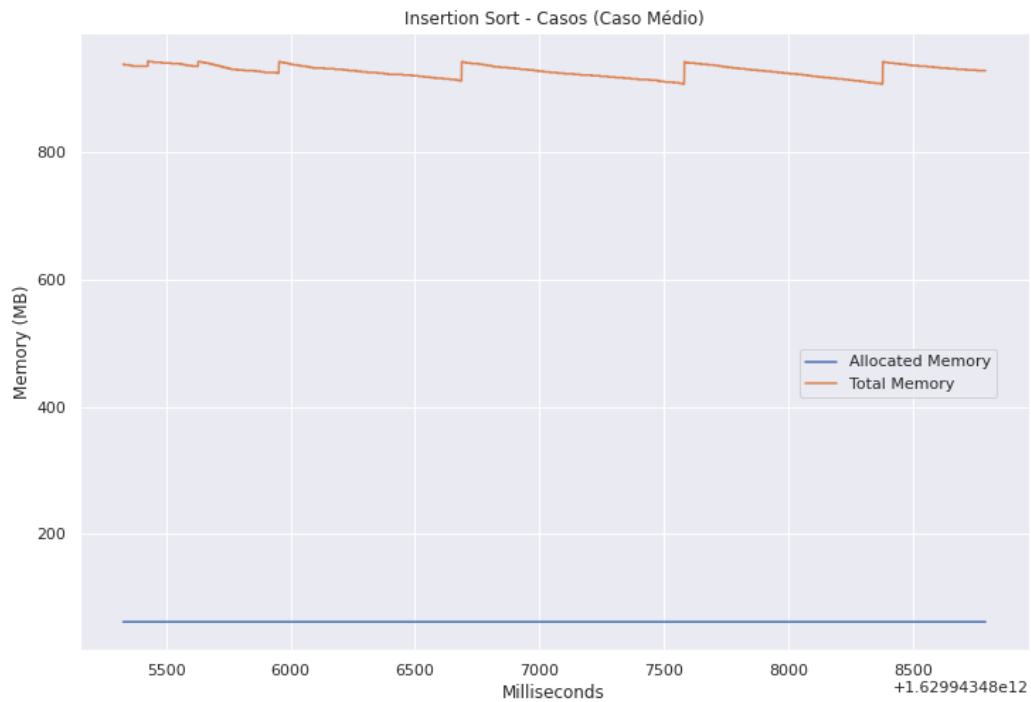


Gráfico 7: Casos Insertion Sort (Caso Médio) Máquina 1.

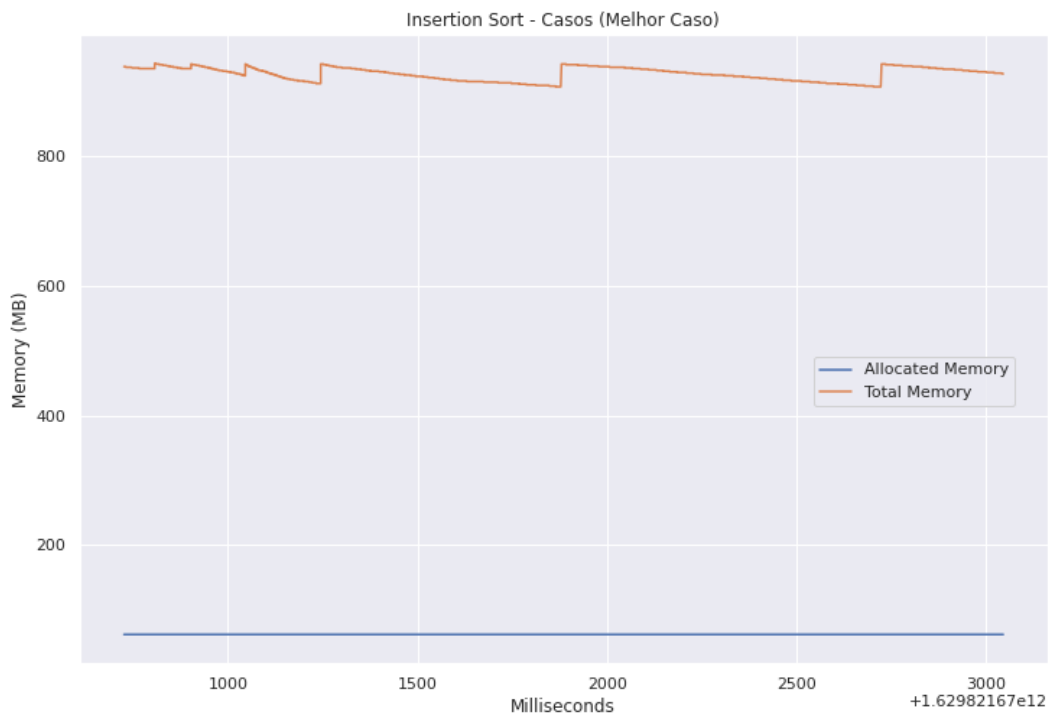


Gráfico 8: Casos Insertion Sort (Melhor Caso) Máquina 1.

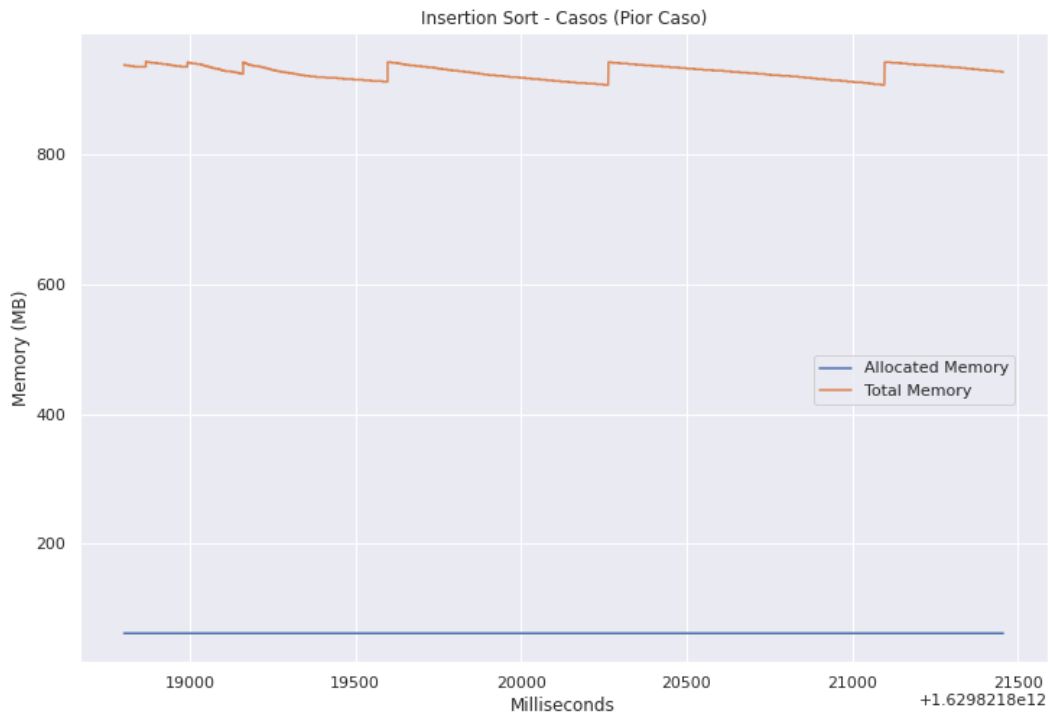


Gráfico 9: Casos Insertion Sort (Pior Caso) Máquina 1.

Cidade:

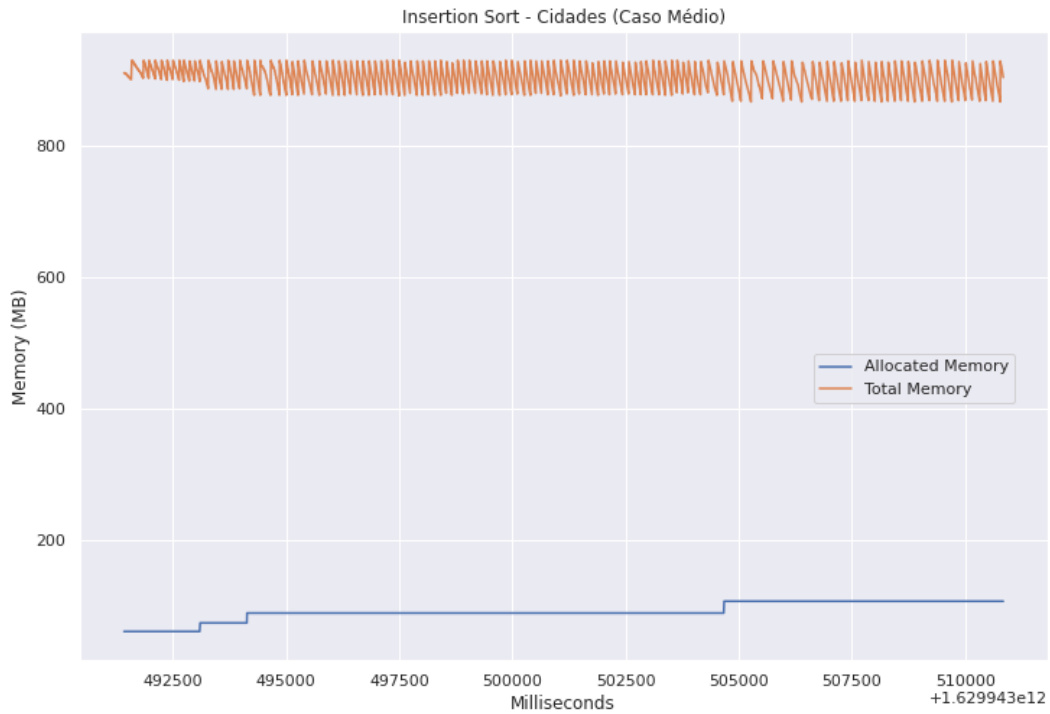


Gráfico 10: Cidades Insertion Sort (Caso Médio) Máquina 1.

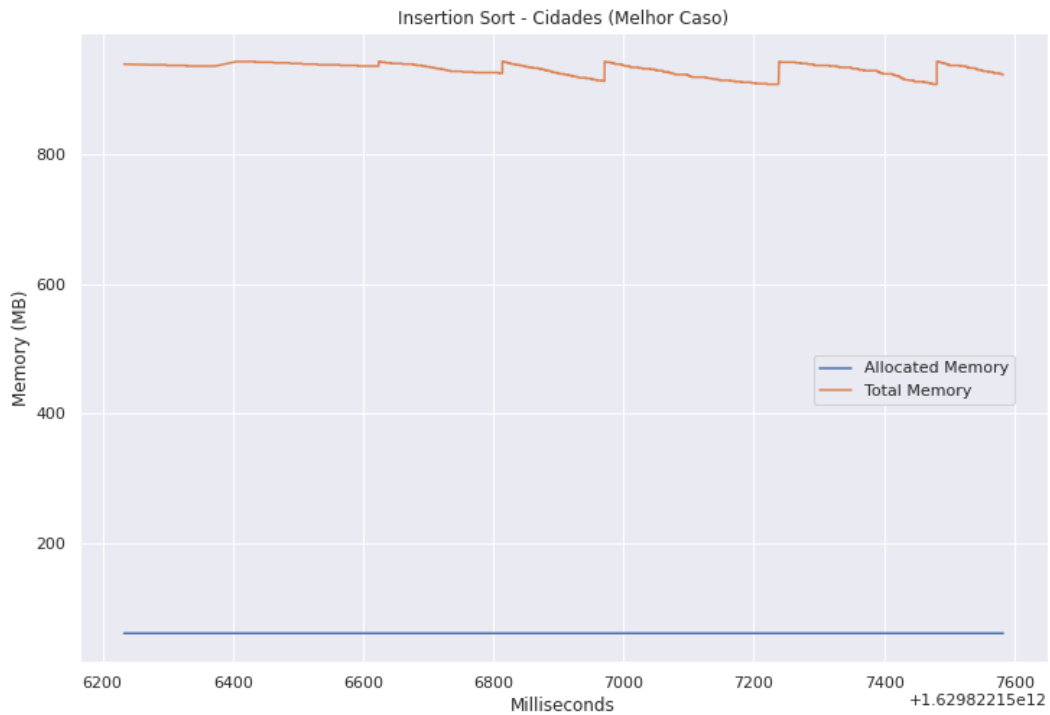


Gráfico 11: Cidades Insertion Sort (Melhor Caso) Máquina 1.

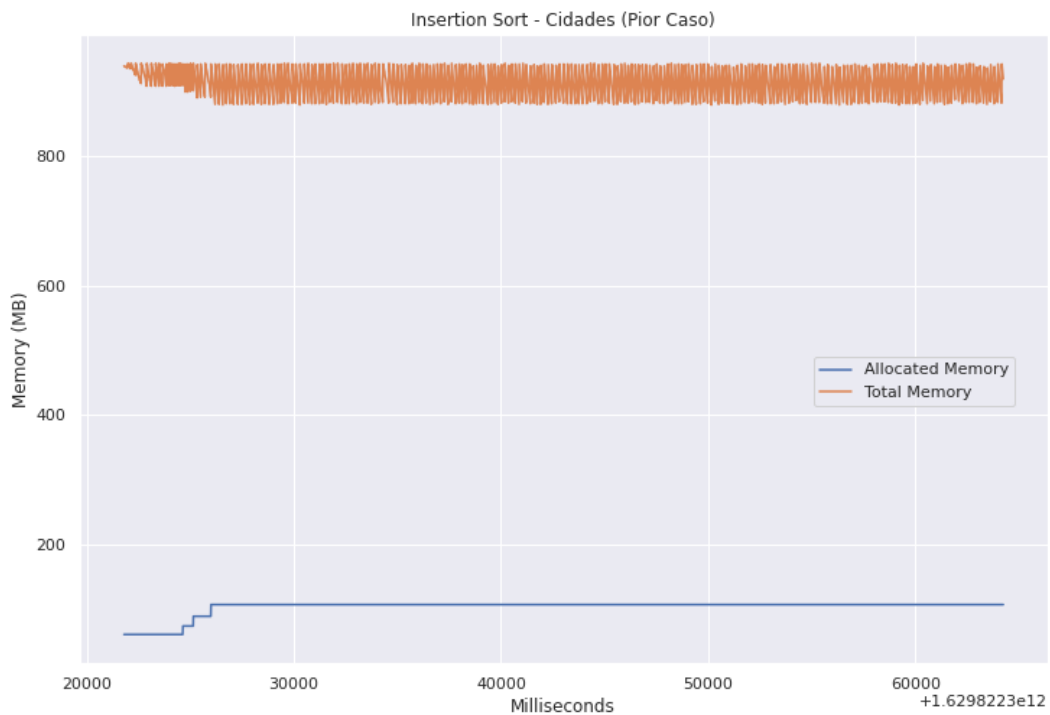


Gráfico 12: Cidades Insertion Sort (Pior Caso) Máquina 1.

Merge Sort:

Casos:

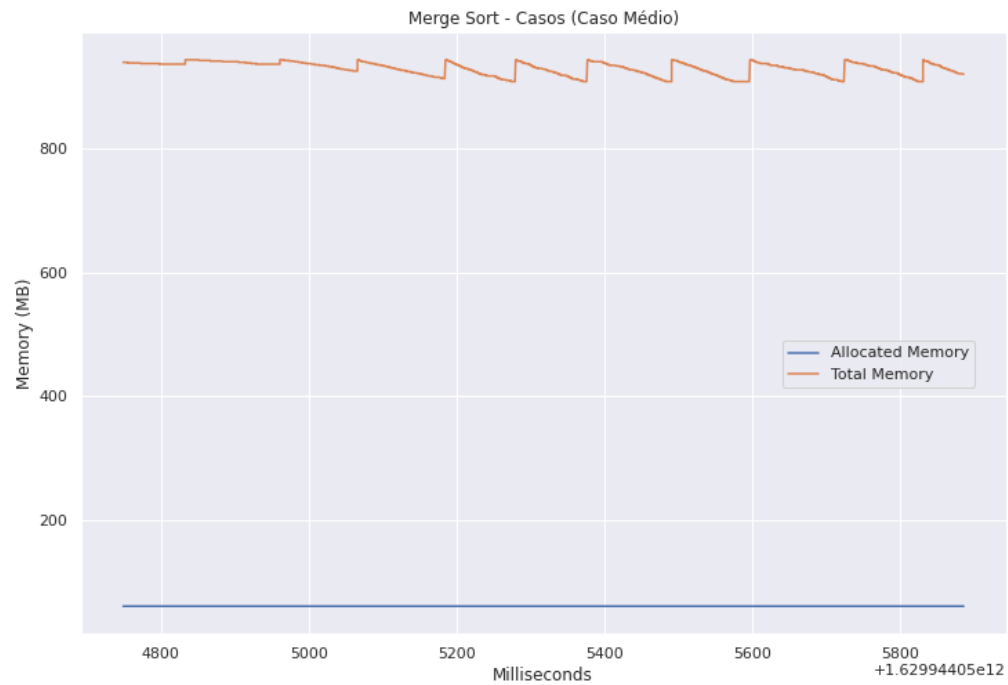


Gráfico 13: Casos Merge Sort (Caso Médio) Máquina 1.

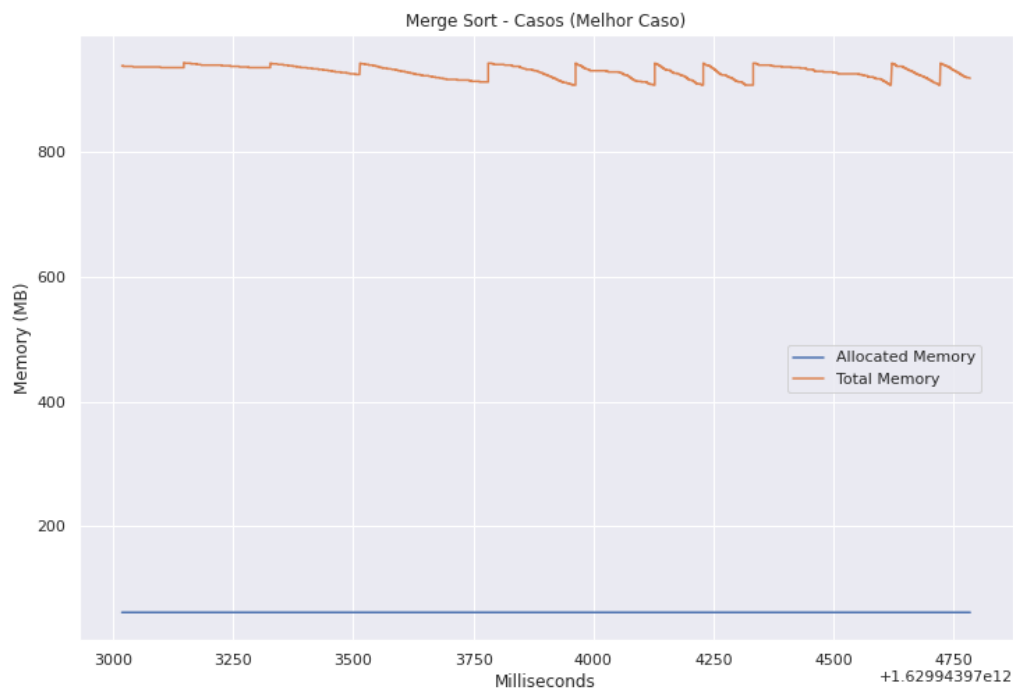


Gráfico 14: Casos Merge Sort (Melhor Caso) Máquina 1.

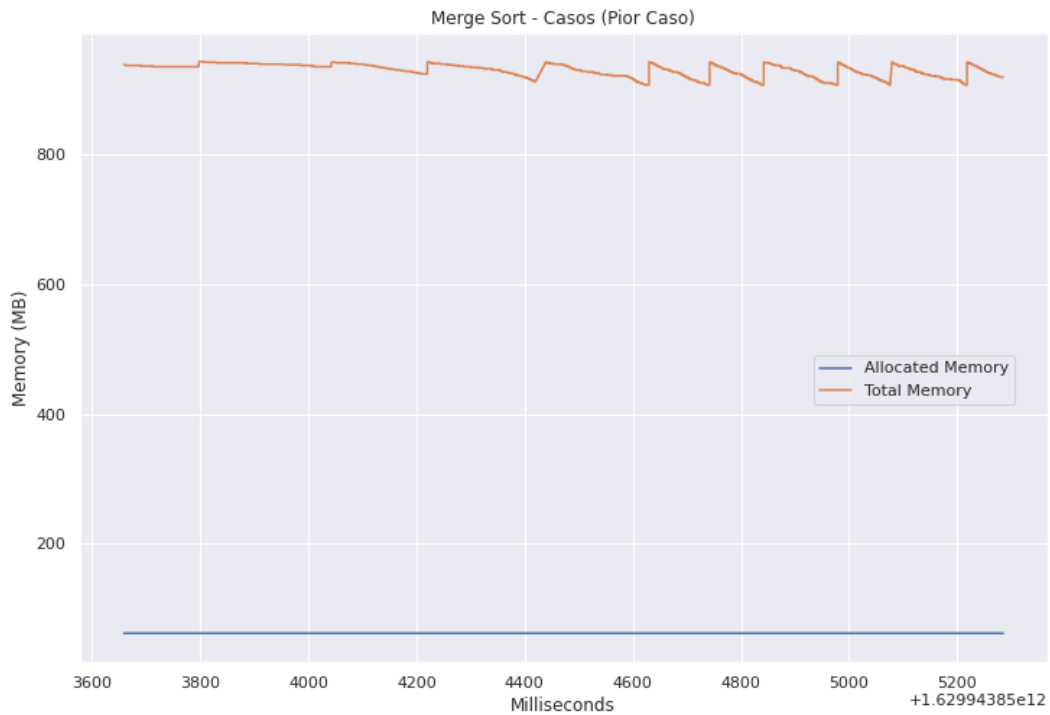


Gráfico 15: Casos Merge Sort (Pior Caso) Máquina 1.

Cidade:

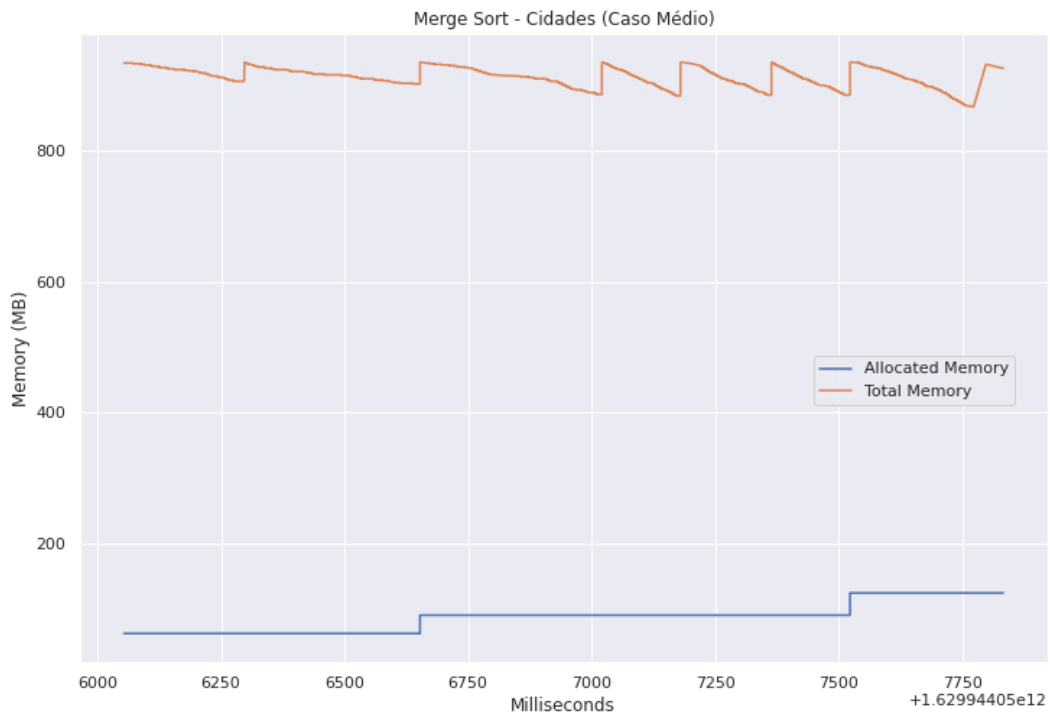


Gráfico 16: Cidades Merge Sort (Caso Médio) Máquina 1.



Gráfico 17: Cidades Merge Sort (Melhor Caso) Máquina 1.

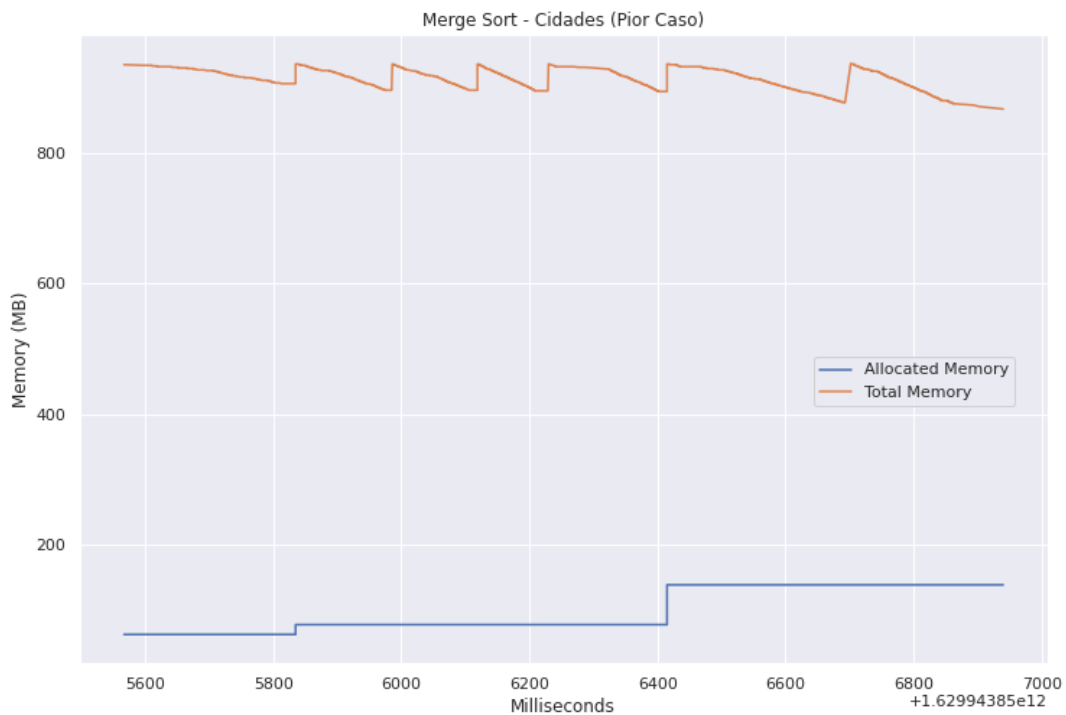


Gráfico 18: Cidades Merge Sort (Pior Caso) Máquina 1.

Quicksort:

Casos:

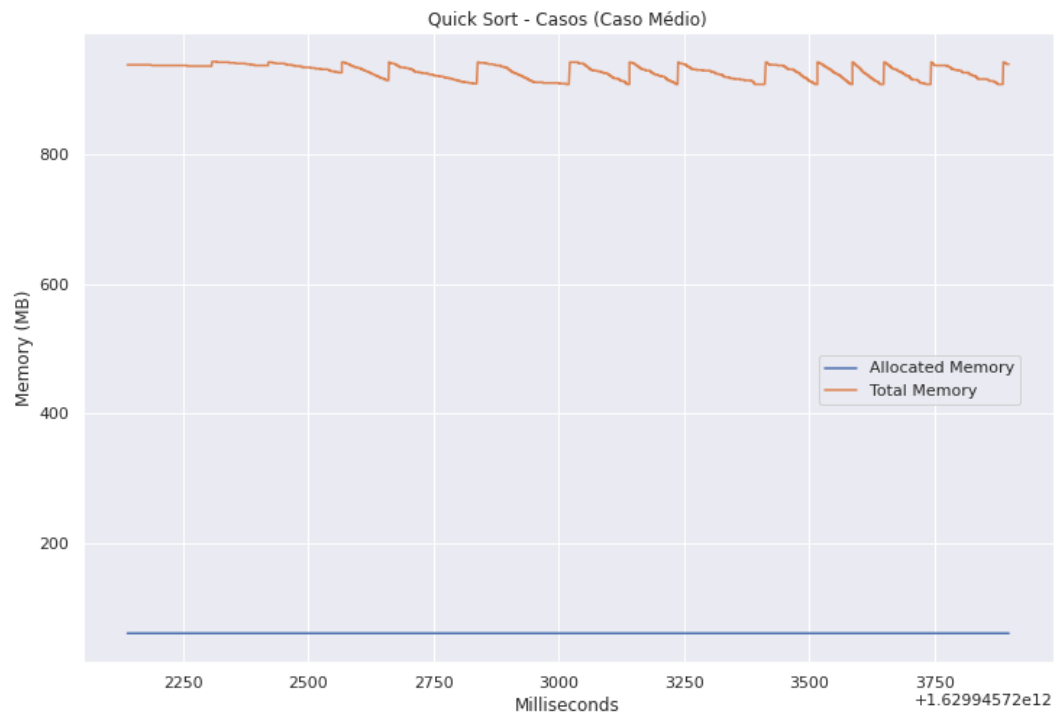


Gráfico 19: Casos Quicksort (Caso Médio) Máquina 1.

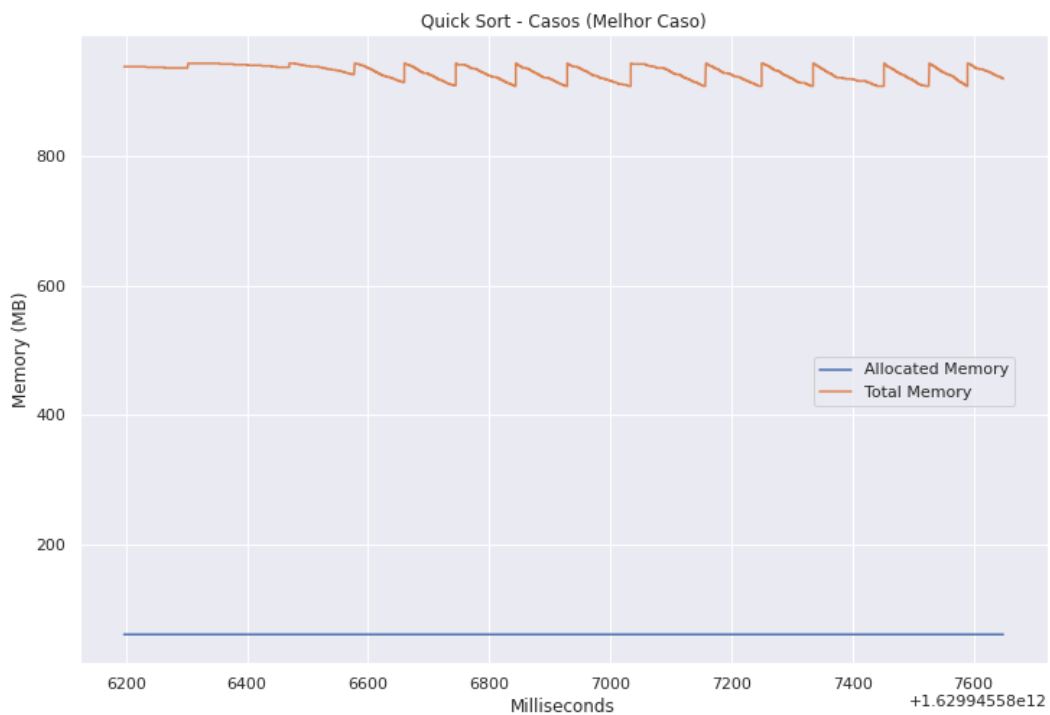


Gráfico 20: Casos Quicksort (Melhor Caso) Máquina 1.

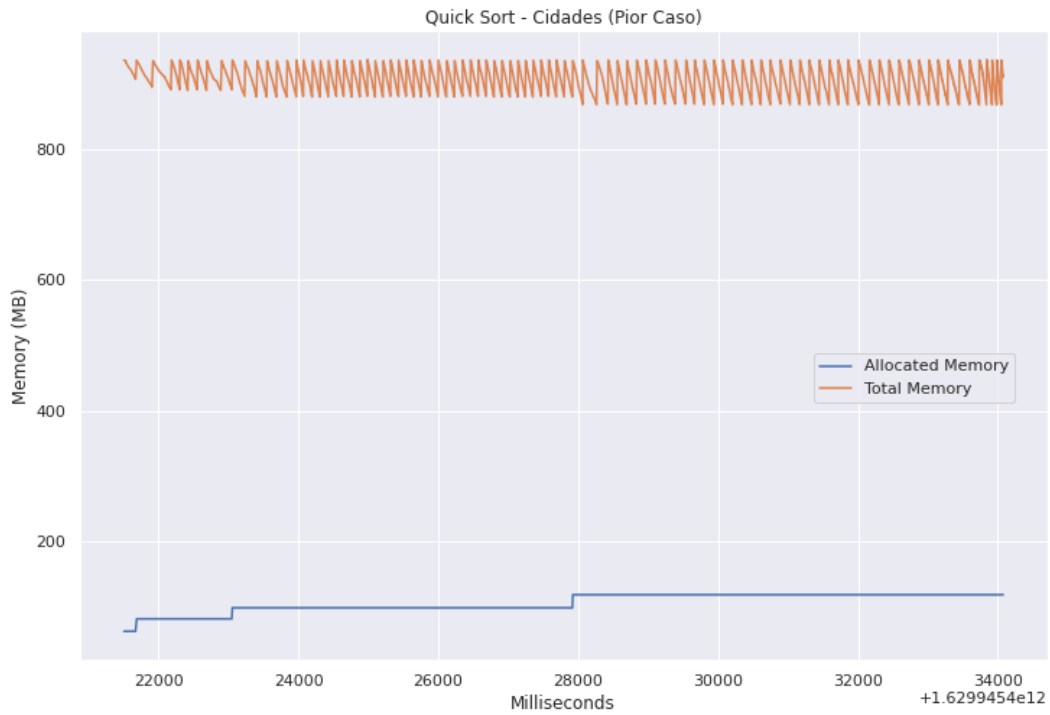


Gráfico 21: Casos Quicksort (Pior Caso) Máquina 1.

Cidade:

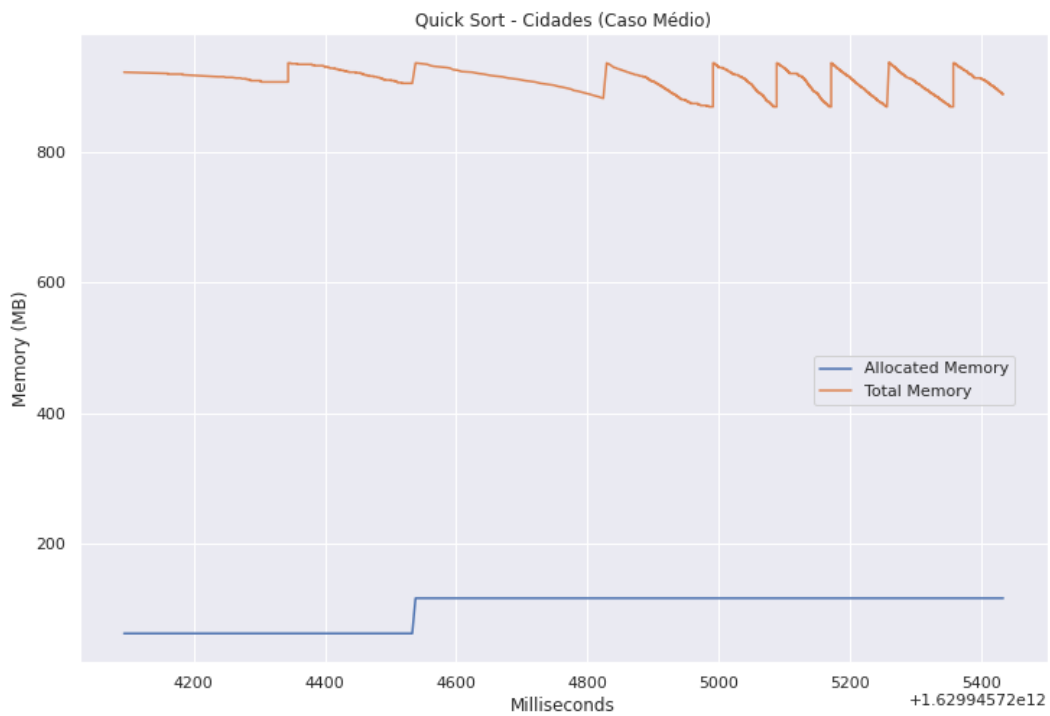


Gráfico 22: Cidades Quicksort (Caso Médio) Máquina 1.

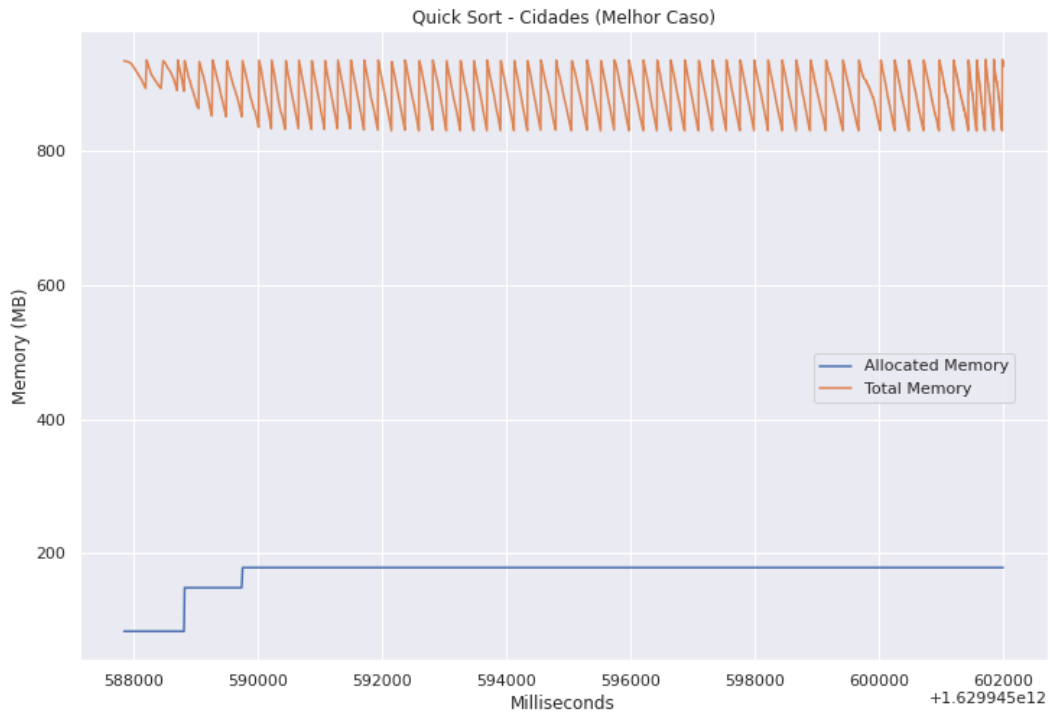


Gráfico 23: Cidades Quicksort (Melhor Caso) Máquina 1.

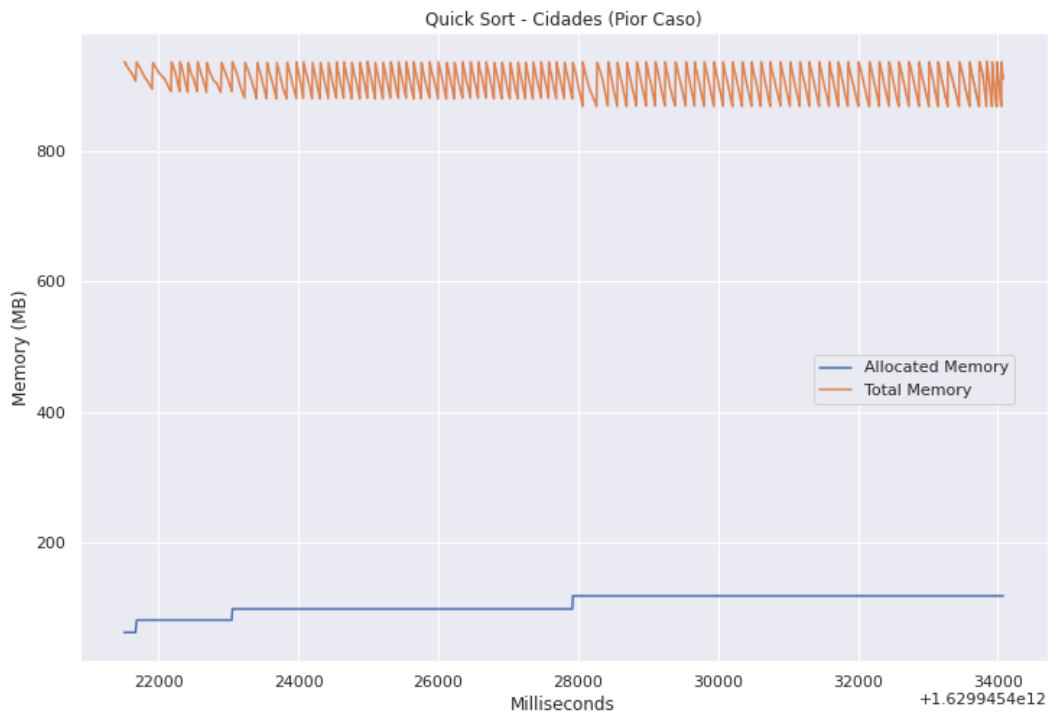


Gráfico 24: Cidades Quicksort (Pior Caso) Máquina 1.

Quicksort com Mediana de 3:

Casos:

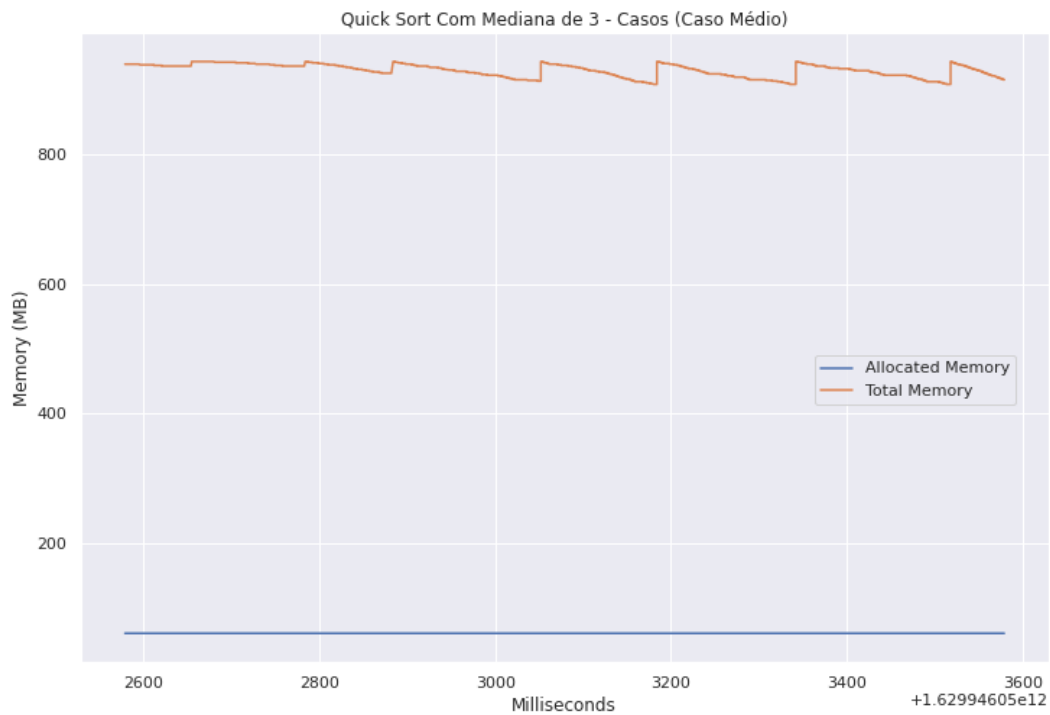


Gráfico 25: Casos Quicksort com mediana de 3 (Caso Médio) Máquina 1.

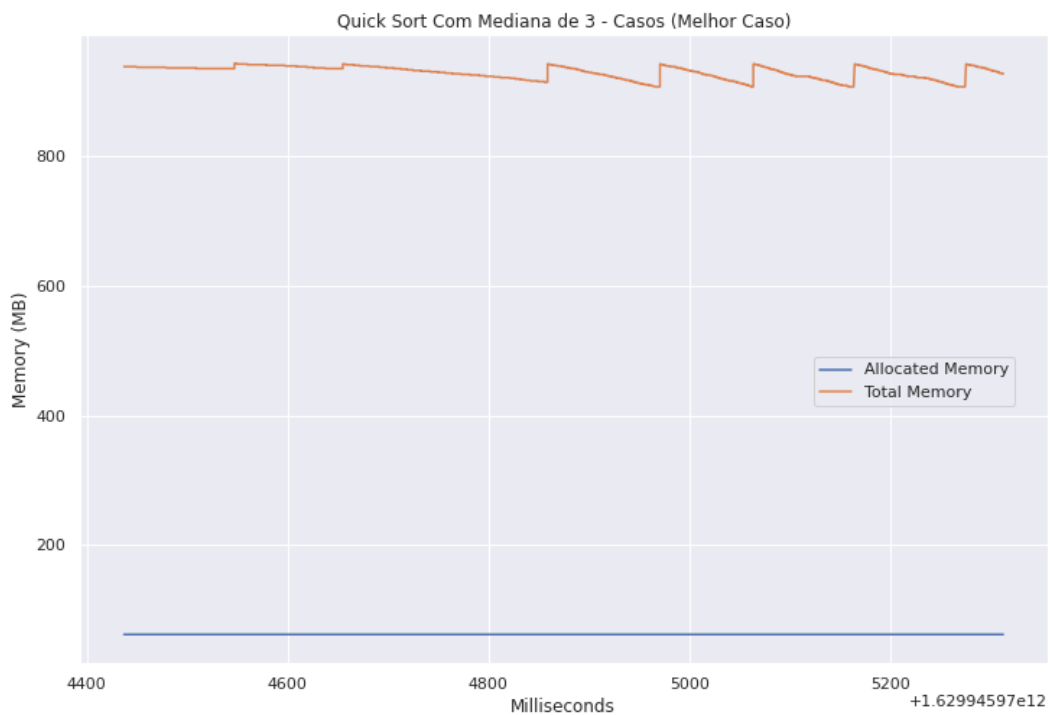


Gráfico 26: Casos Quicksort com mediana de 3 (Melhor Caso) Máquina 1.

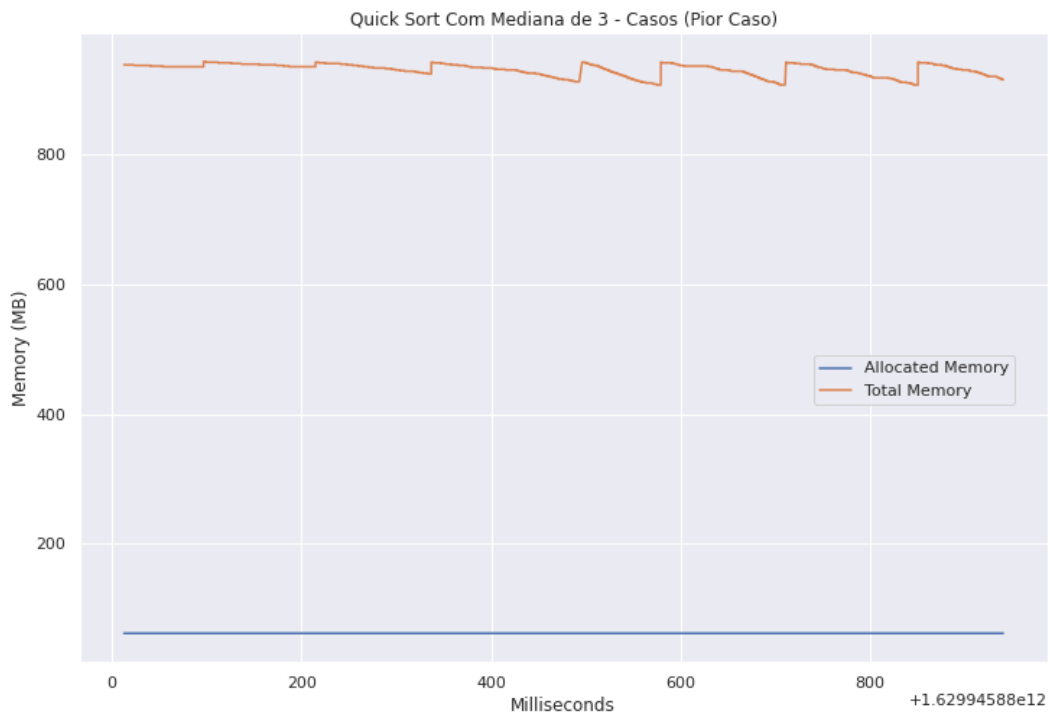


Gráfico 27: Casos Quicksort com mediana de 3 (Pior Caso) Máquina 1.

Cidade:

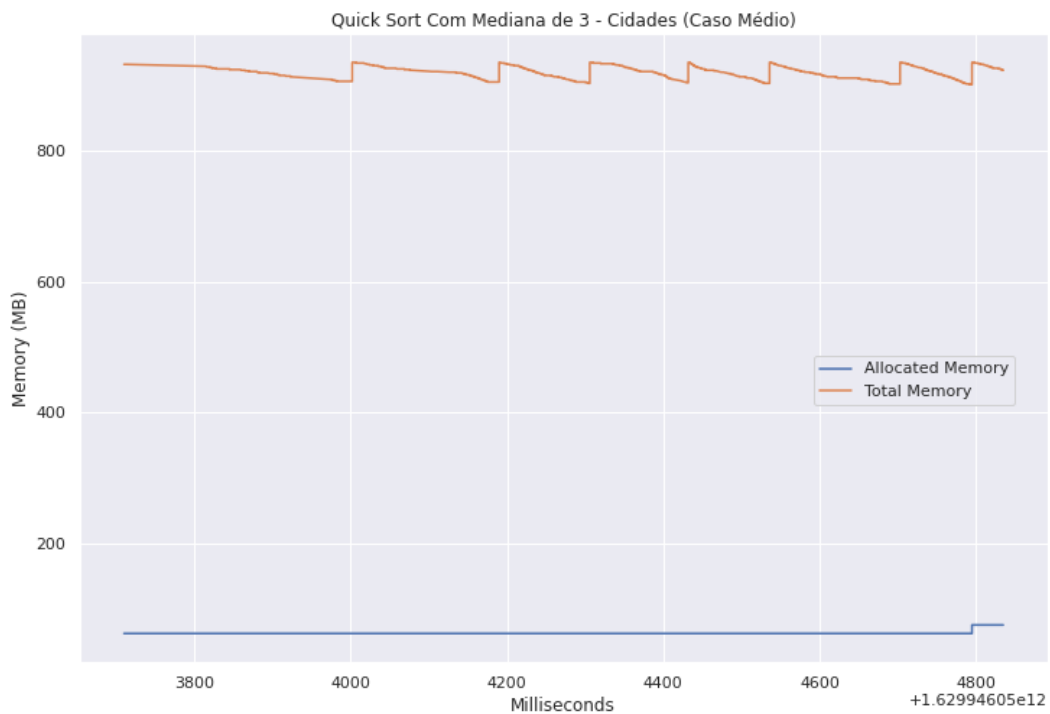


Gráfico 28: Cidades Quicksort com mediana de 3 (Caso Médio) Máquina 1.

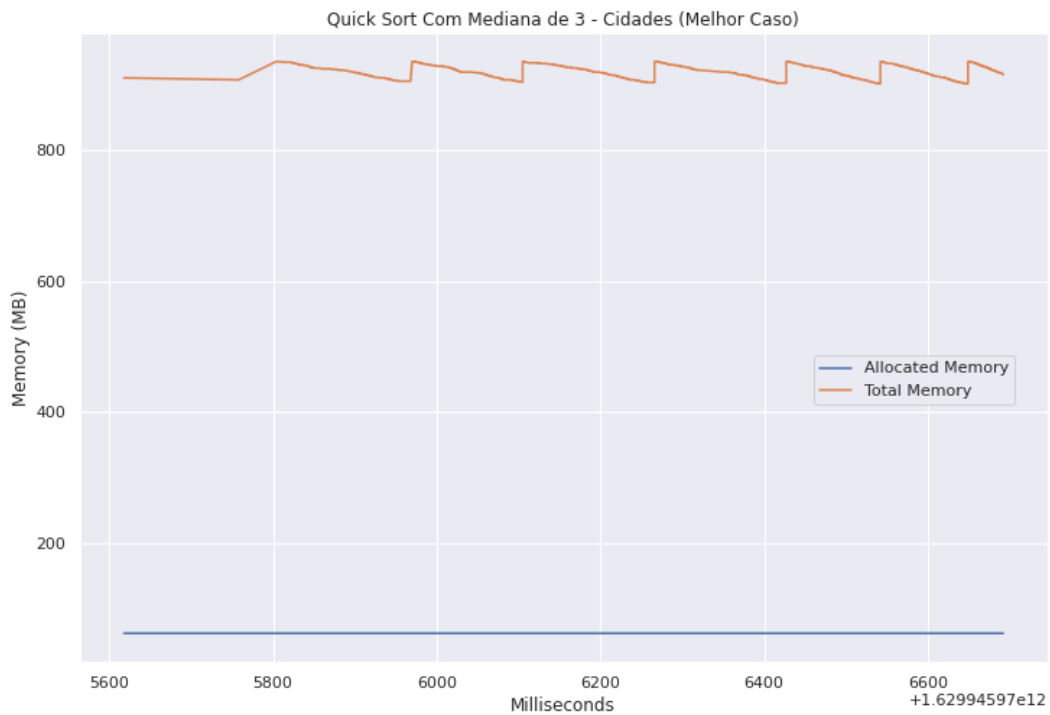


Gráfico 29: Cidades Quicksort com mediana de 3 (Melhor Caso) Máquina 1.

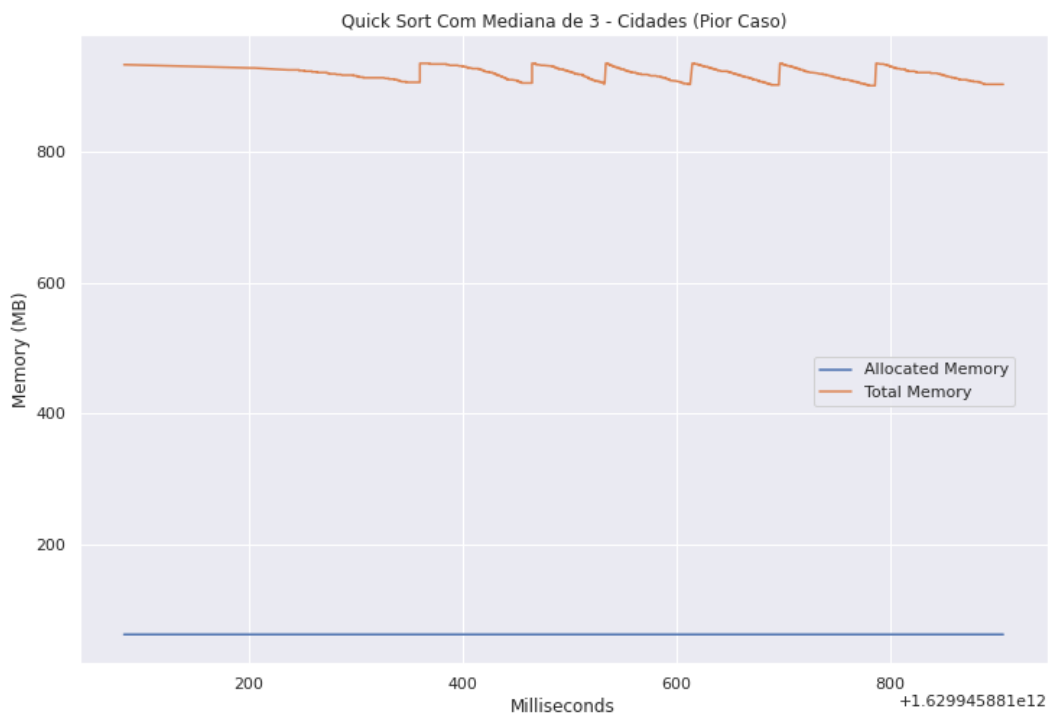


Gráfico 30: Cidades Quicksort com mediana de 3 (Pior Caso) Máquina 1.

Heapsort:

Casos:

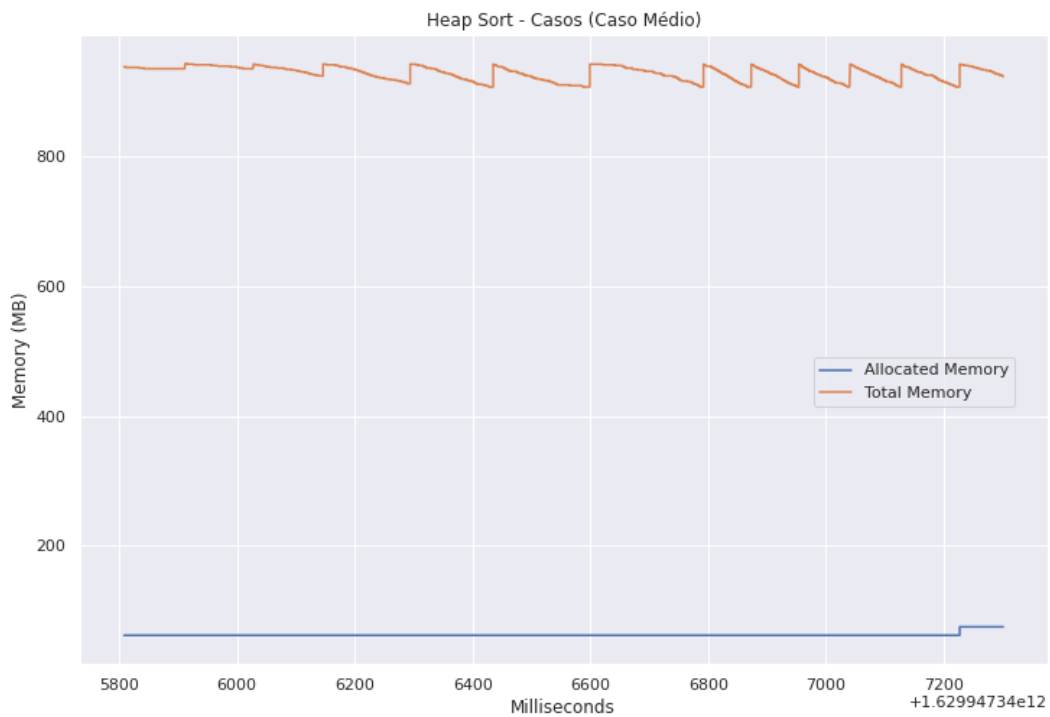


Gráfico 31: Casos Heapsort (Caso Médio) Máquina 1.

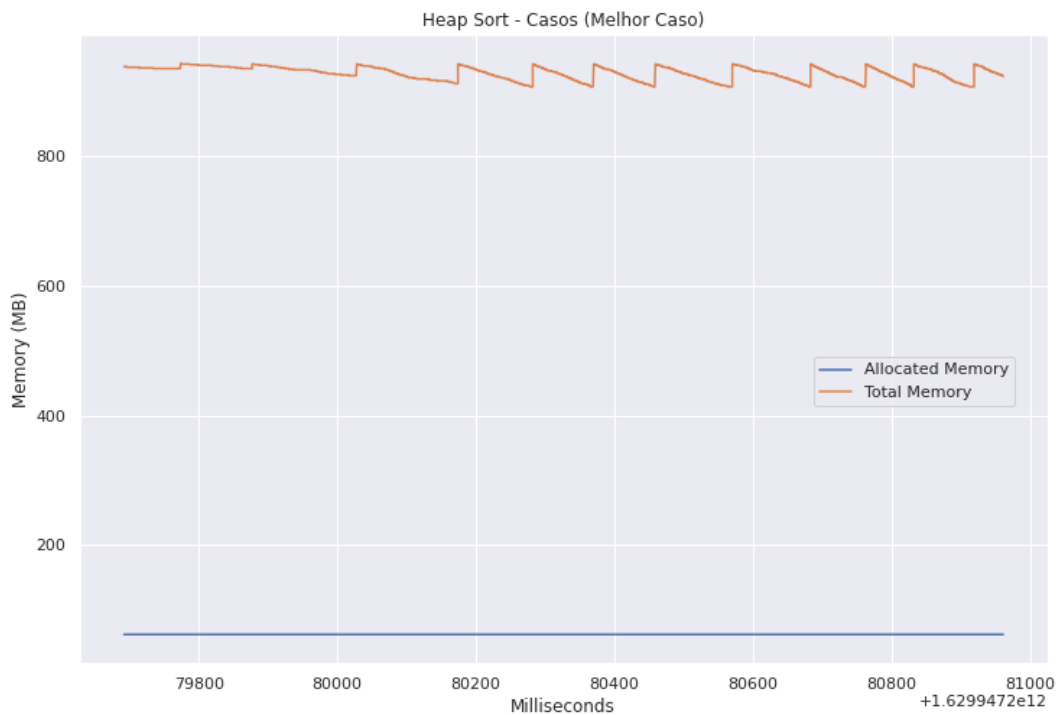


Gráfico 32: Casos Heapsort (Melhor Caso) Máquina 1.

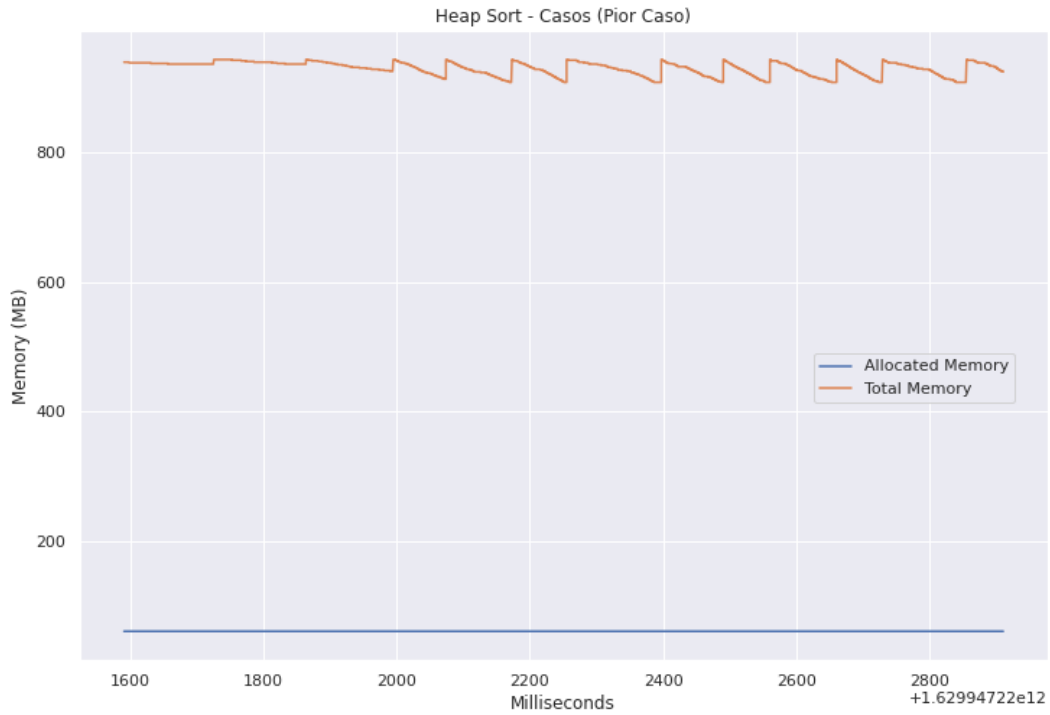


Gráfico 33: Casos Heapsort (Pior Caso) Máquina 1.

Cidade:

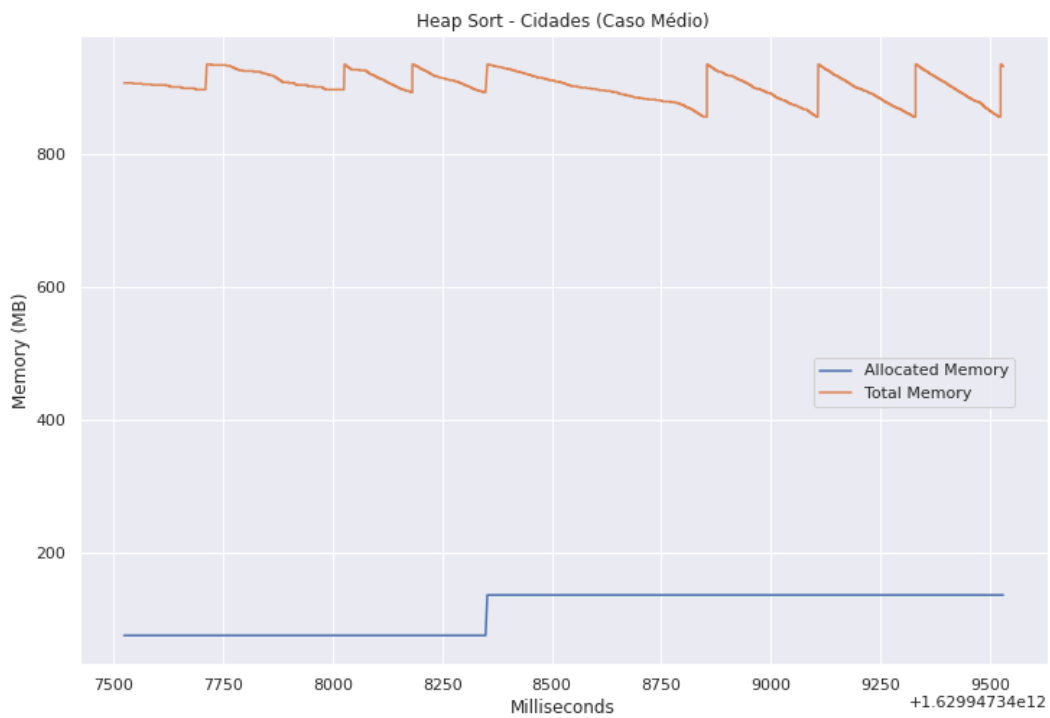


Gráfico 34: Cidades Heapsort (Caso Médio) Máquina 1.

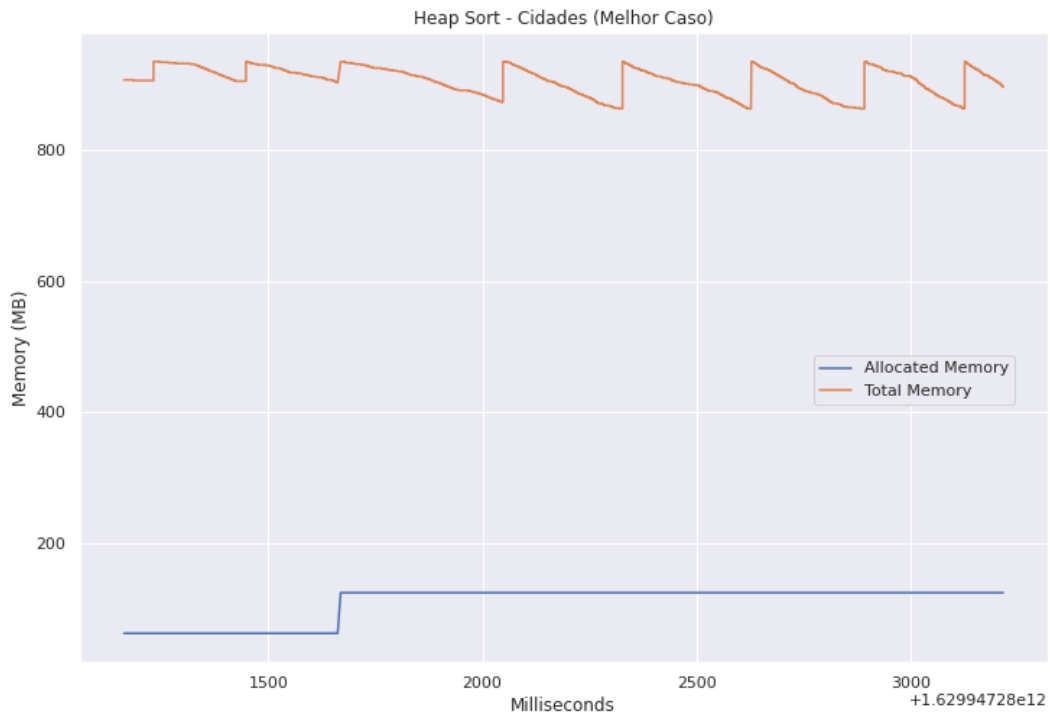


Gráfico 35: Cidades Heapsort (Melhor Caso) Máquina 1.

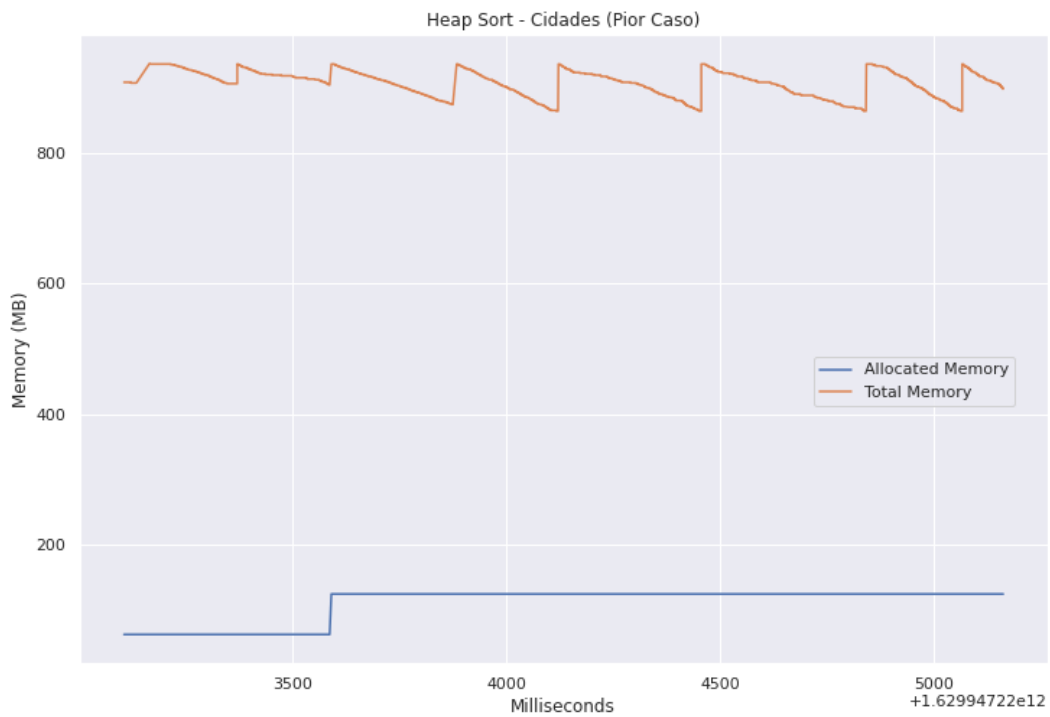


Gráfico 36: Cidades Heapsort (Pior Caso) Máquina 1.

Counting Sort:

Casos:

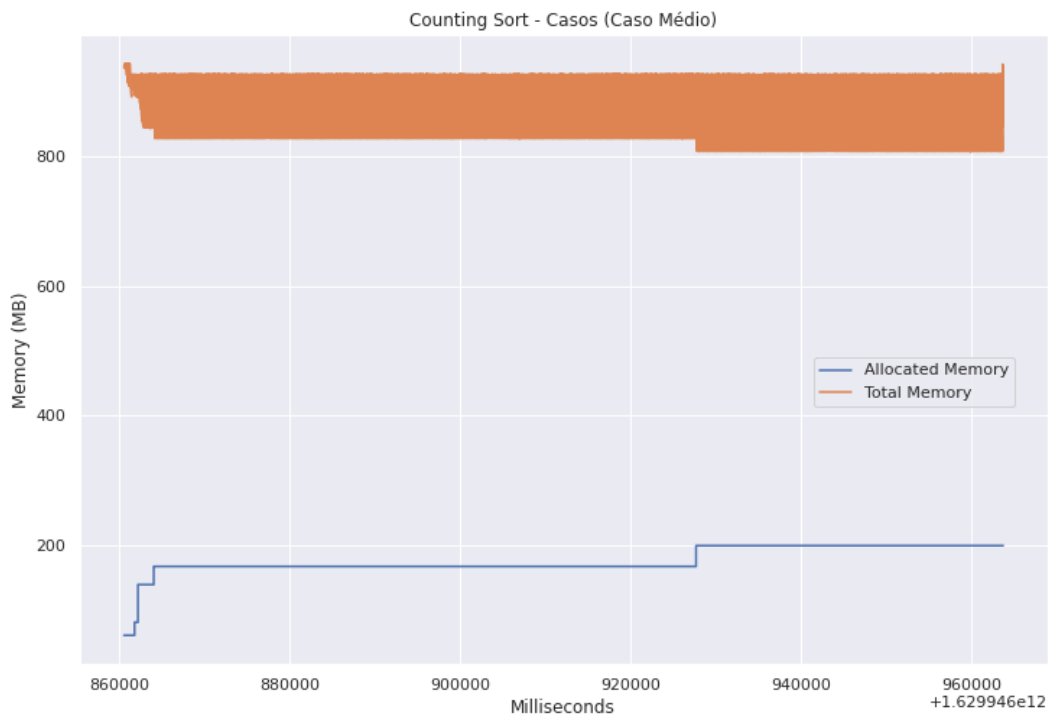


Gráfico 37 Casos Counting Sort (Caso Médio) Máquina 1.

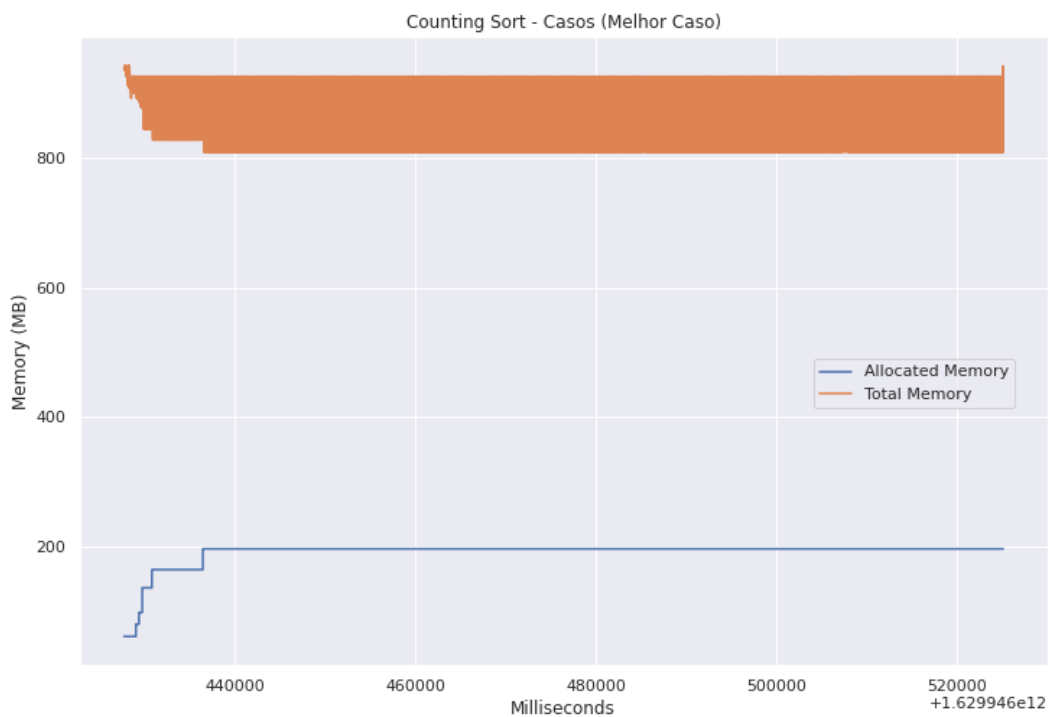


Gráfico 38: Casos Counting Sort (Melhor Caso) Máquina 1.

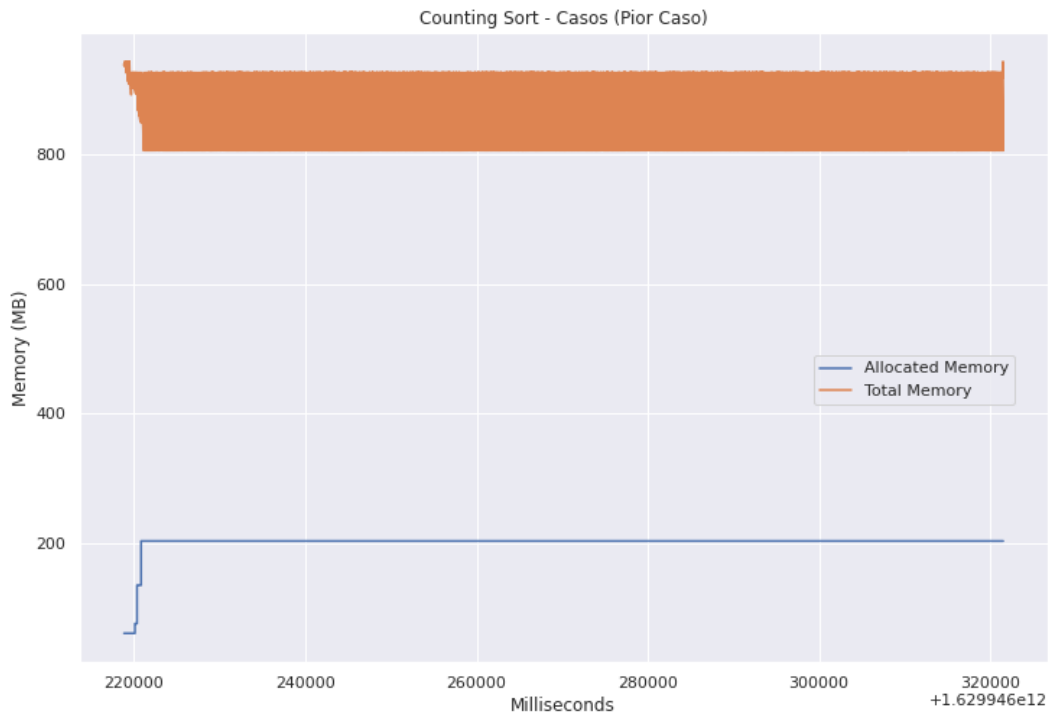


Gráfico 39: Casos Counting Sort (Pior Caso) Máquina 1.

Cidade:

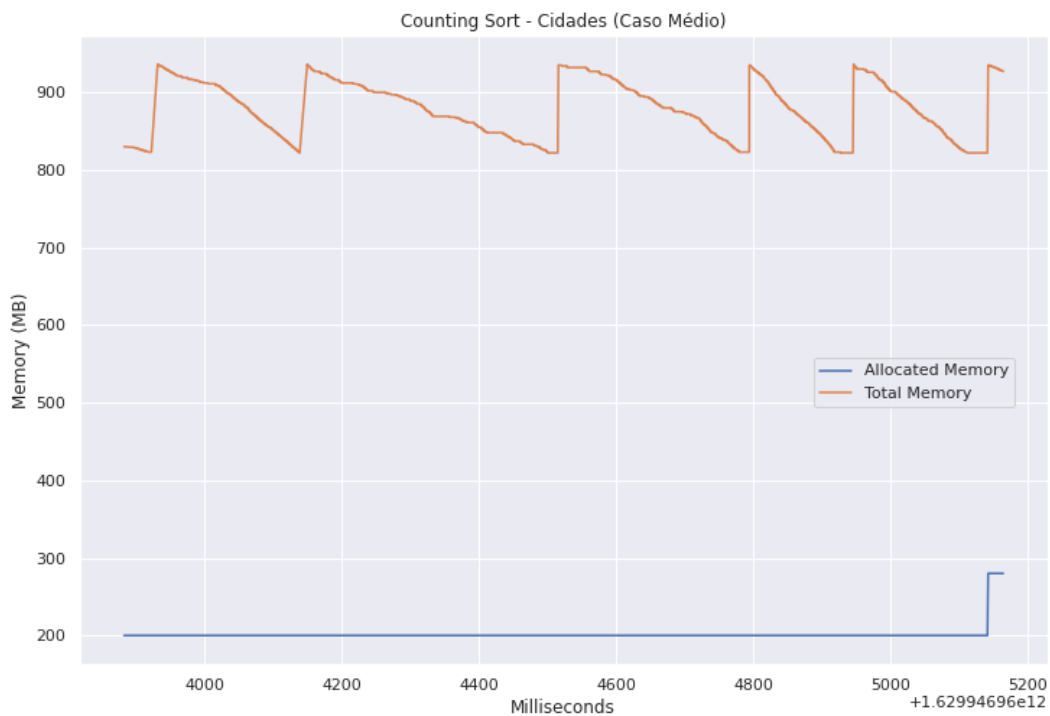


Gráfico 40: Cidades Counting Sort (Caso Médio) Máquina 1.

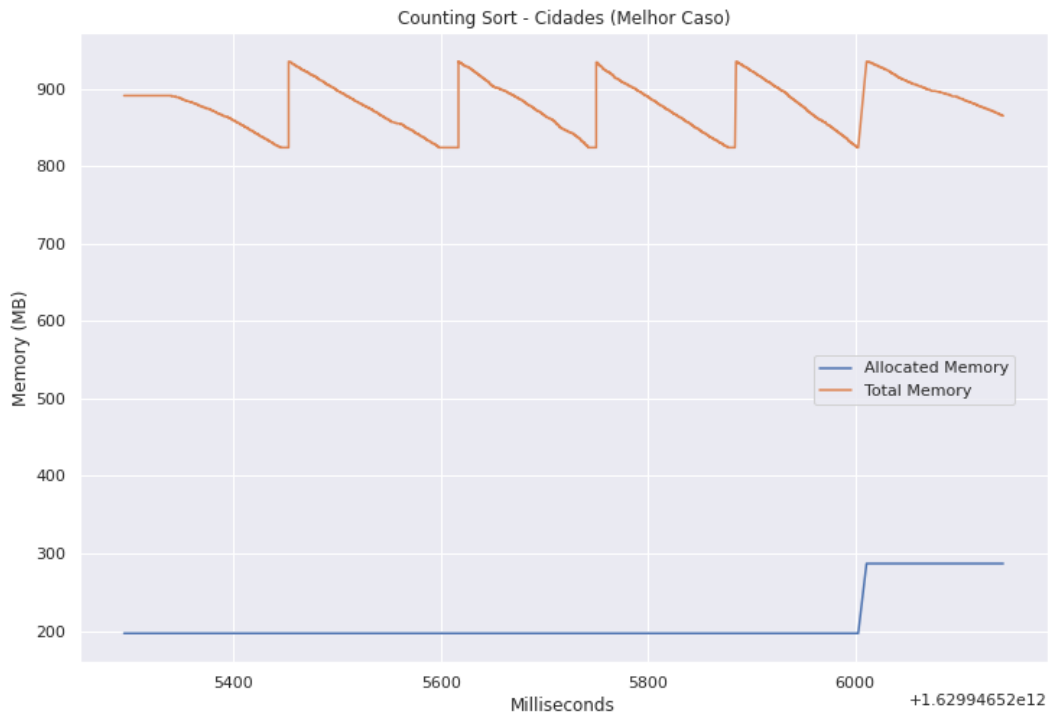


Gráfico 41: Cidades Counting Sort (Melhor Caso) Máquina 1.

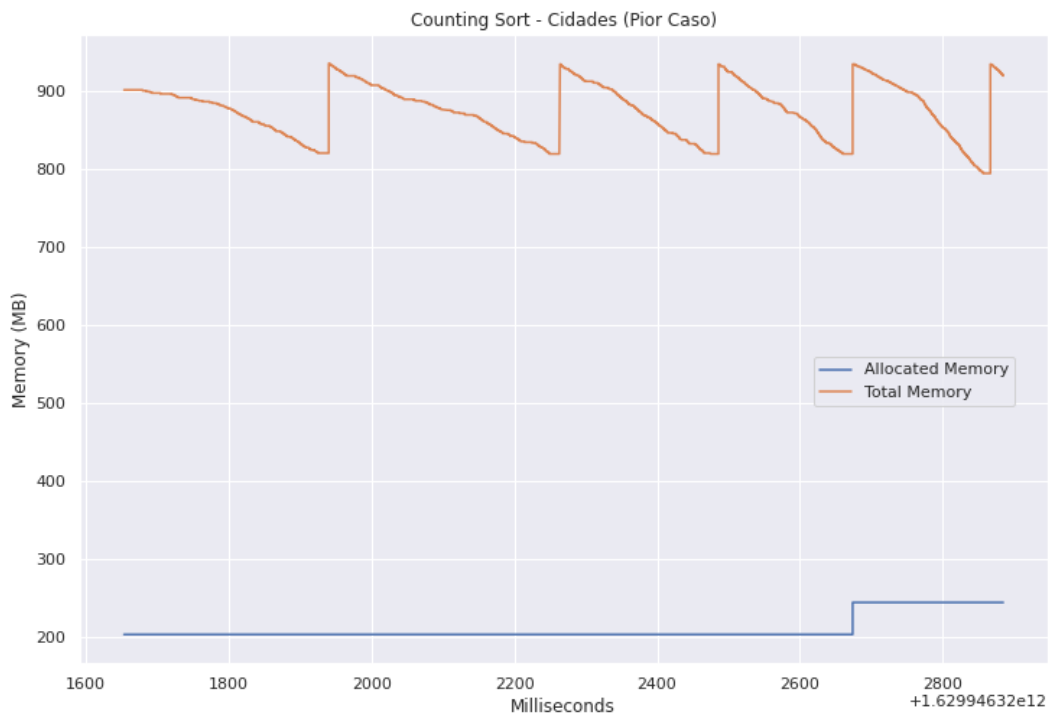


Gráfico 42: Cidades Counting Sort (Pior Caso) Máquina 1.

3.2 Gráficos da Máquina 2

Selection Sort:

Casos:

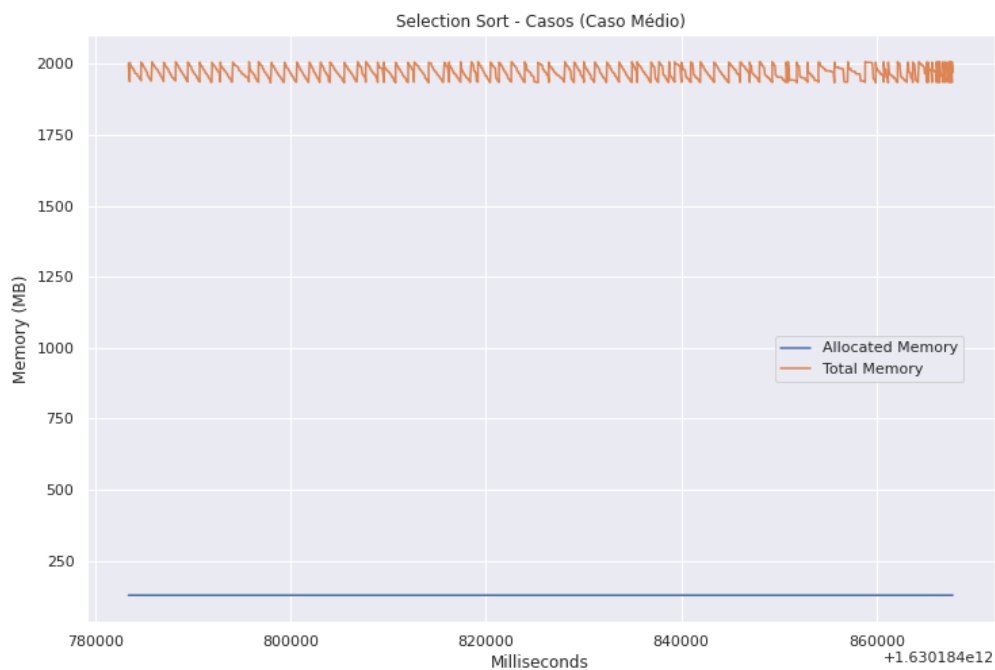


Gráfico 43: Casos Selection Sort (Caso Médio) Máquina 2.

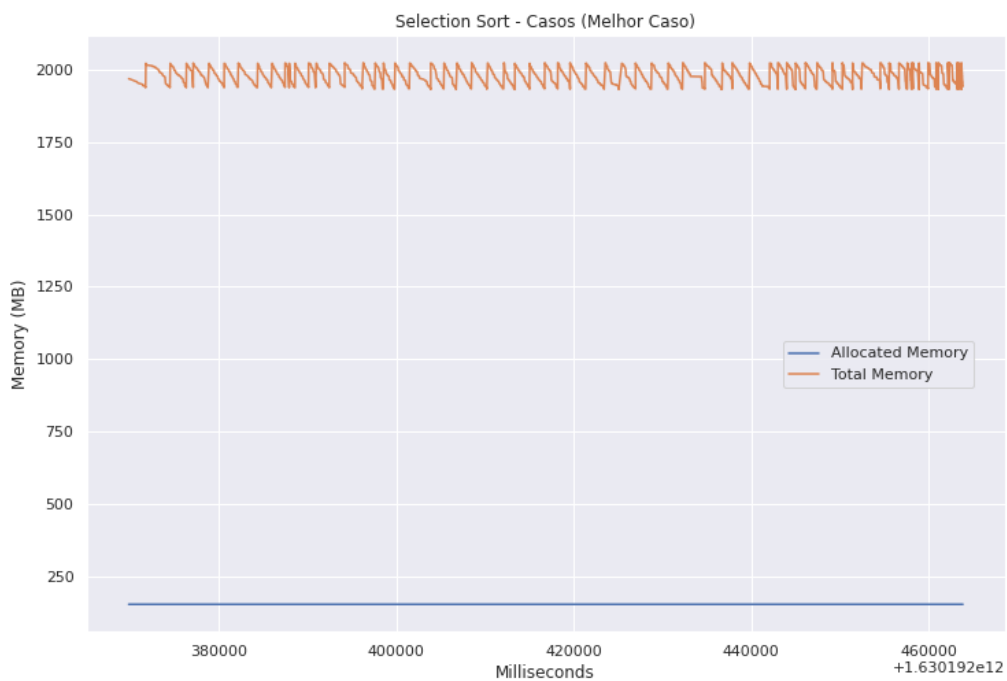


Gráfico 44: Casos Selection Sort (Melhor Caso) Máquina 2.

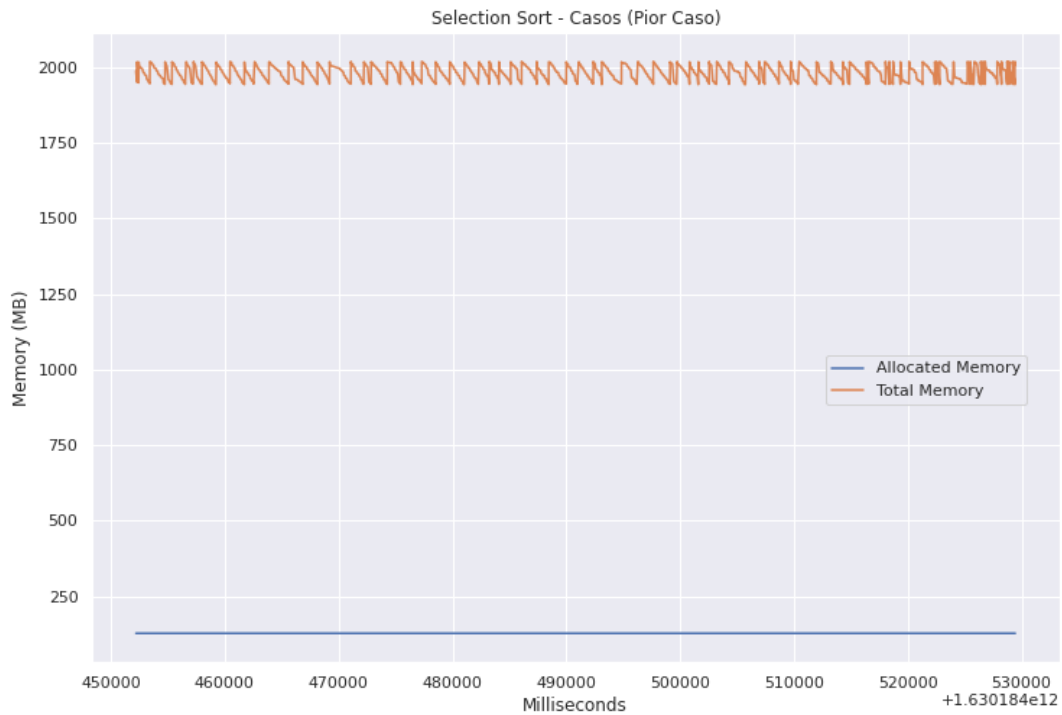


Gráfico 45: Casos Selection Sort (Pior Caso) Máquina 2.

Cidade:

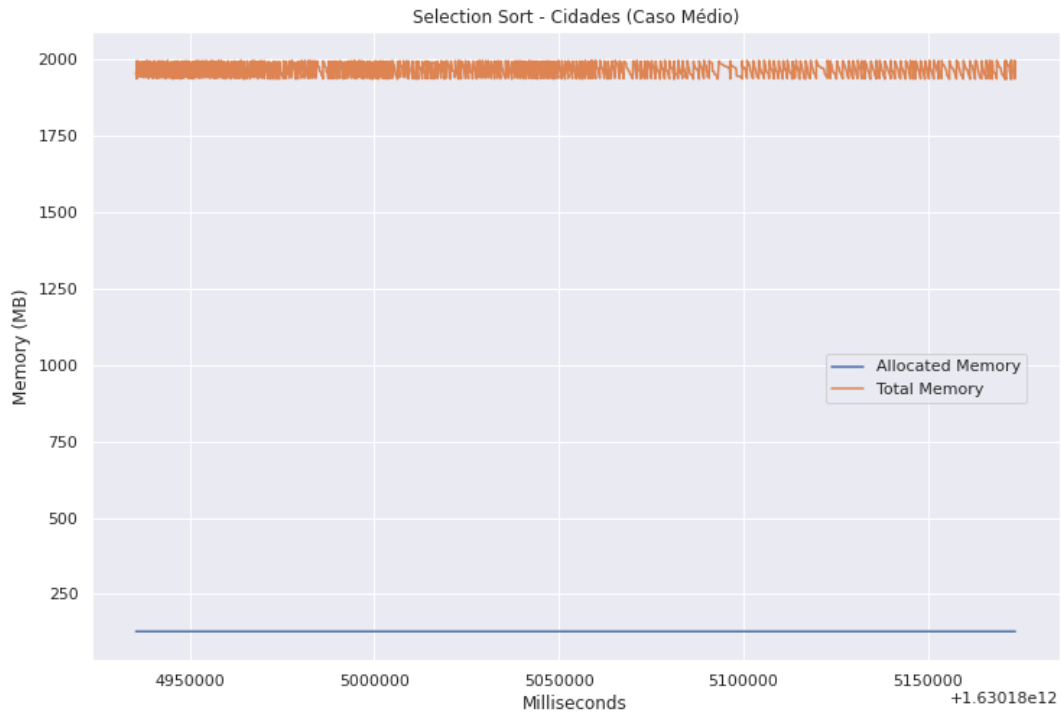


Gráfico 46: Cidades Selection Sort (Caso Médio) Máquina 2.

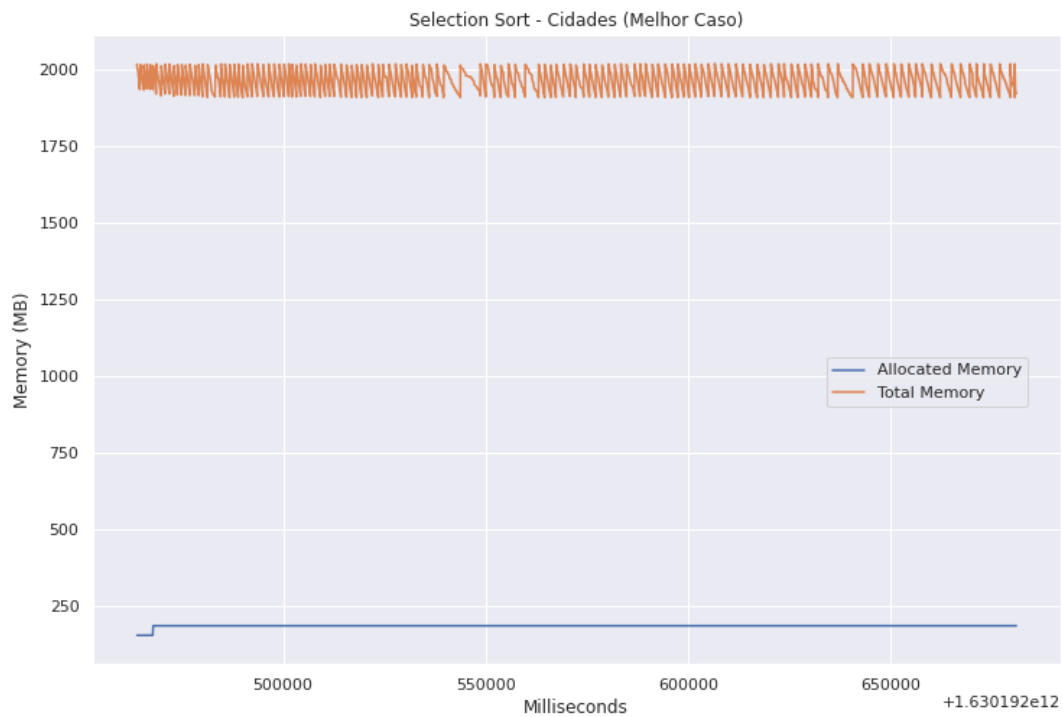


Gráfico 47: Cidades Selection Sort (Melhor Caso) Máquina 2.

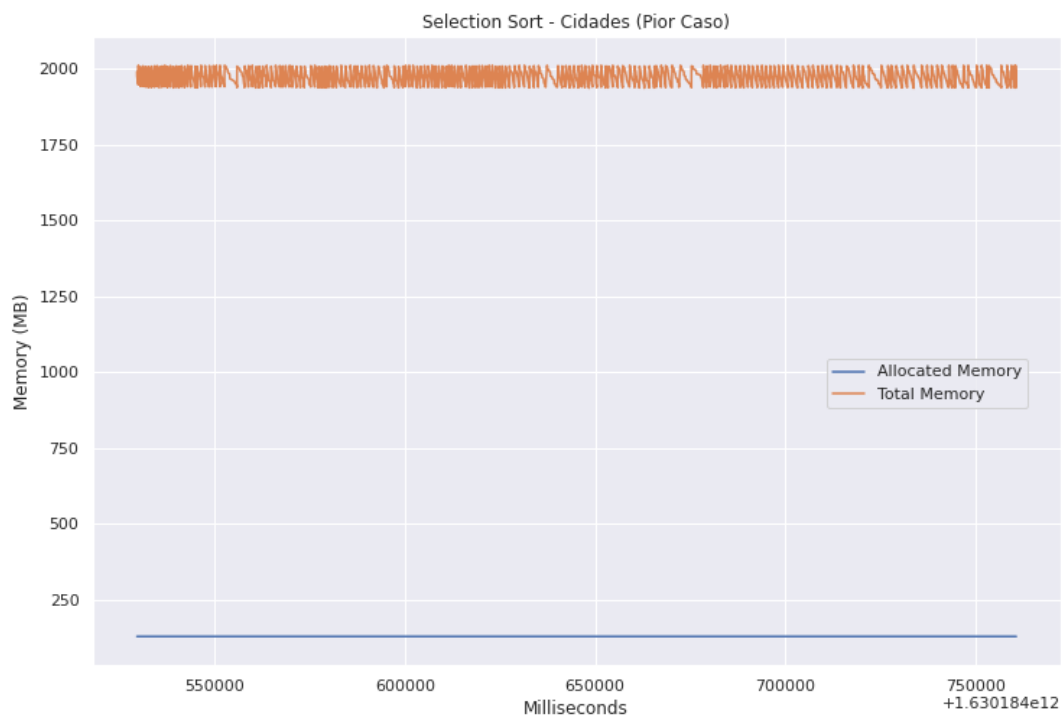


Gráfico 48: Cidades Selection Sort (Pior Caso) Máquina 2.

Insertion Sort:

Casos:

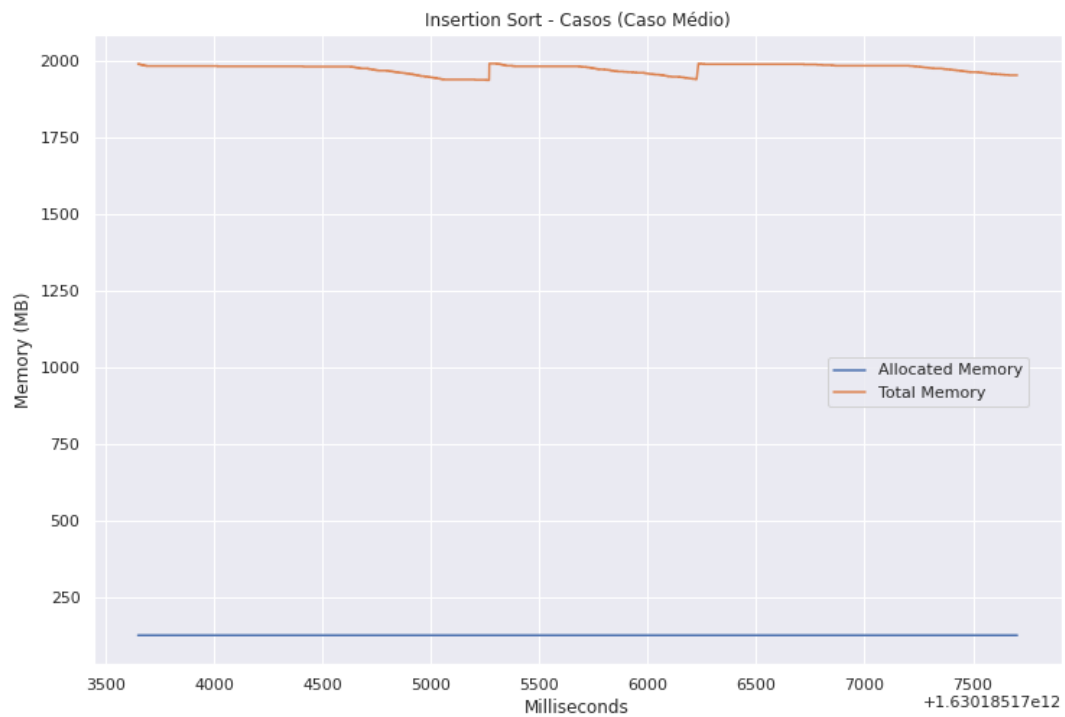


Gráfico 49: Casos Insertion Sort (Caso Médio) Máquina 2.

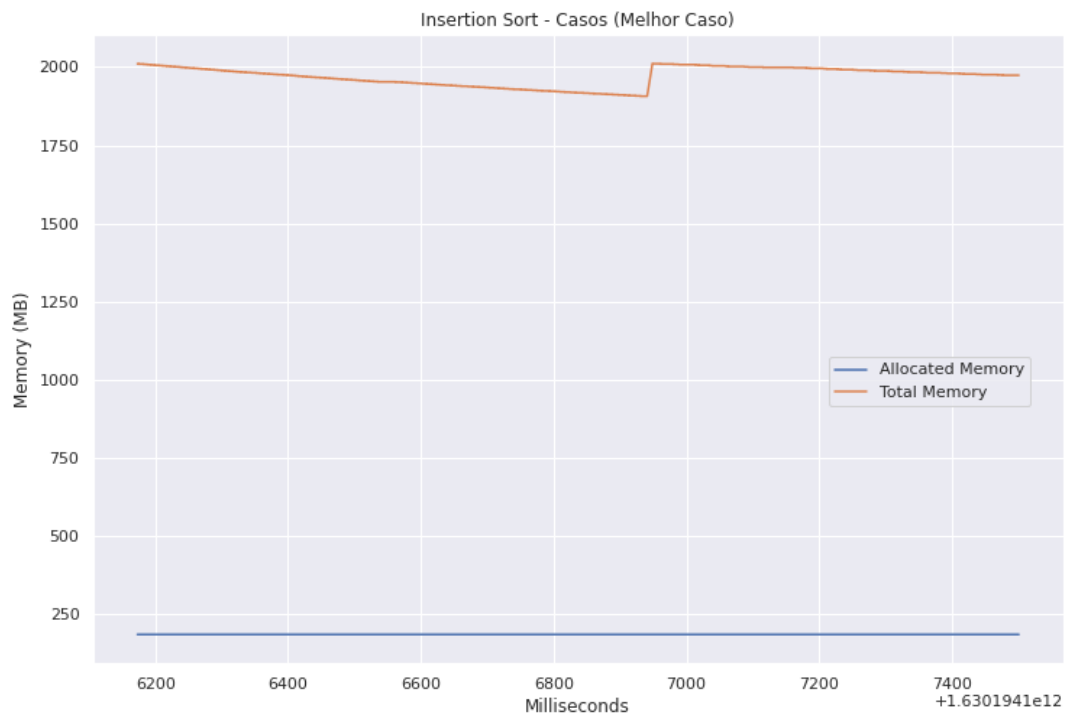


Gráfico 50: Casos Insertion Sort (Melhor Caso) Máquina 2.

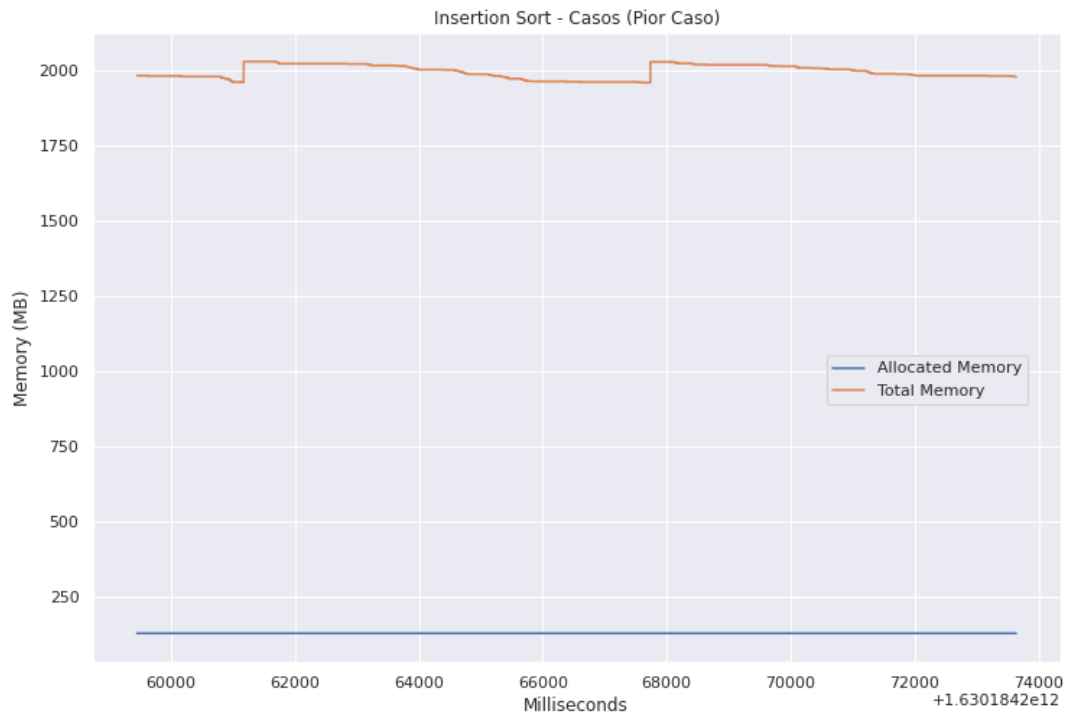


Gráfico 51: Casos Insertion Sort (Pior Caso) Máquina 2.

Cidade:

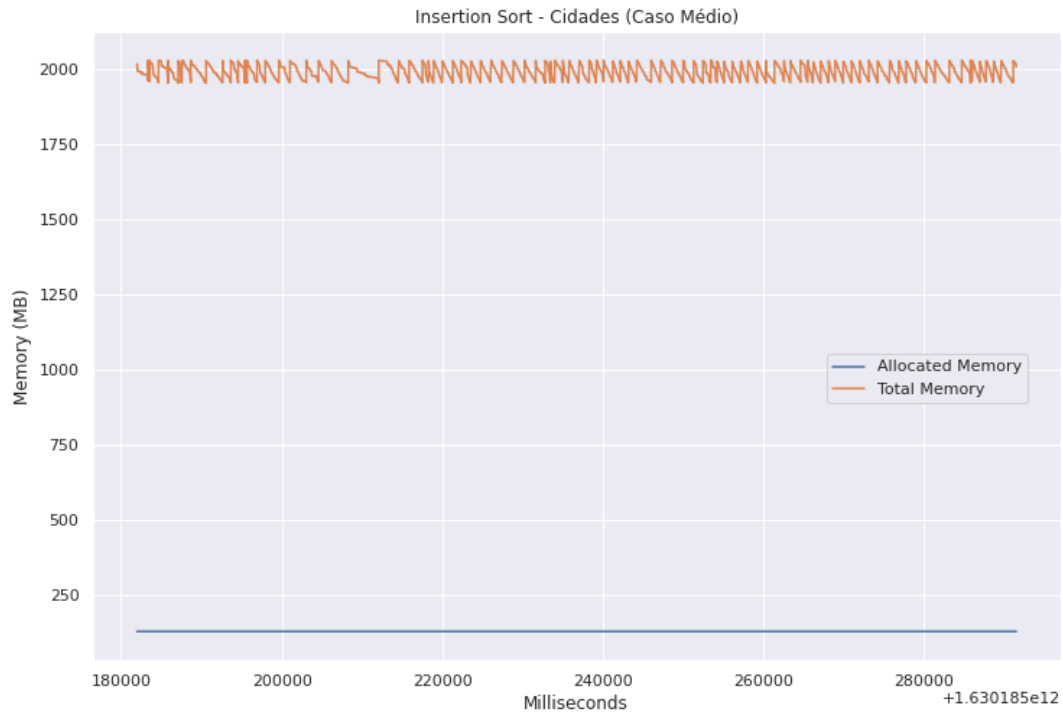


Gráfico 52: Cidades Insertion Sort (Caso Médio) Máquina 2.

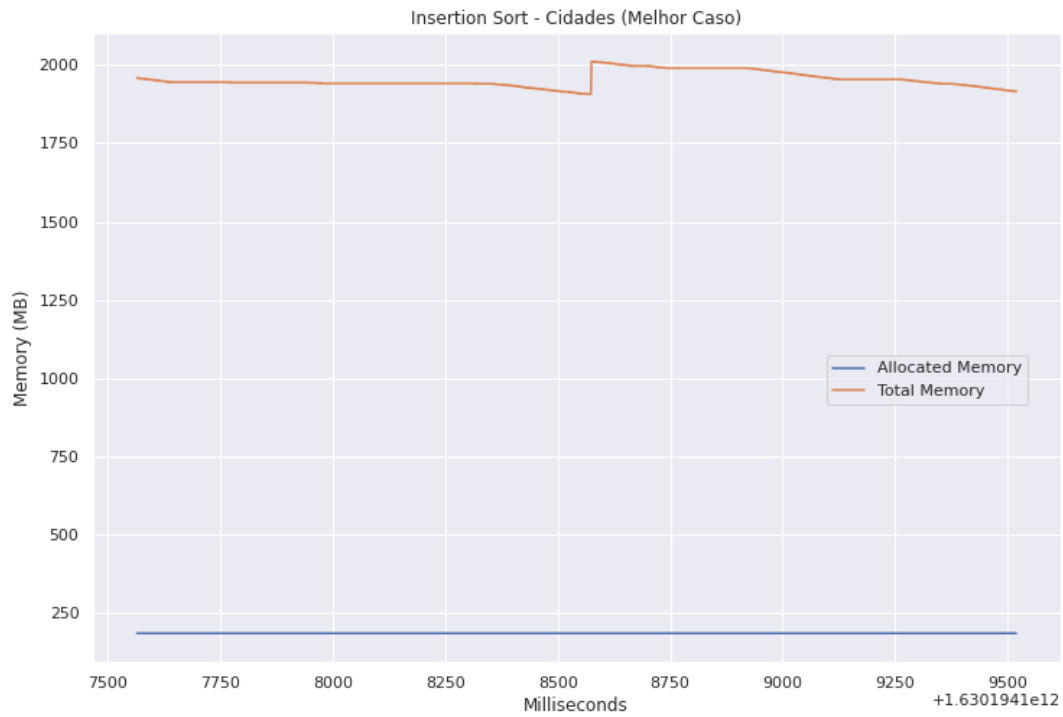


Gráfico 53: Cidades Insertion Sort (Melhor Caso) Máquina 2.

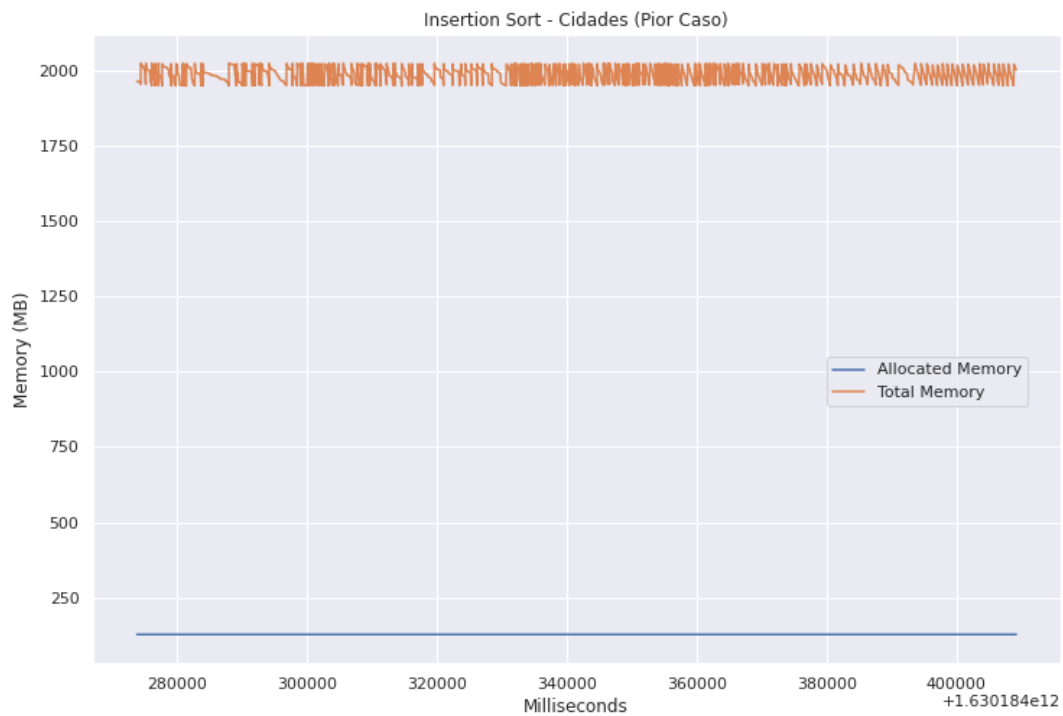


Gráfico 54: Cidades Insertion Sort (Pior Caso) Máquina 2.

Merge Sort:

Casos:

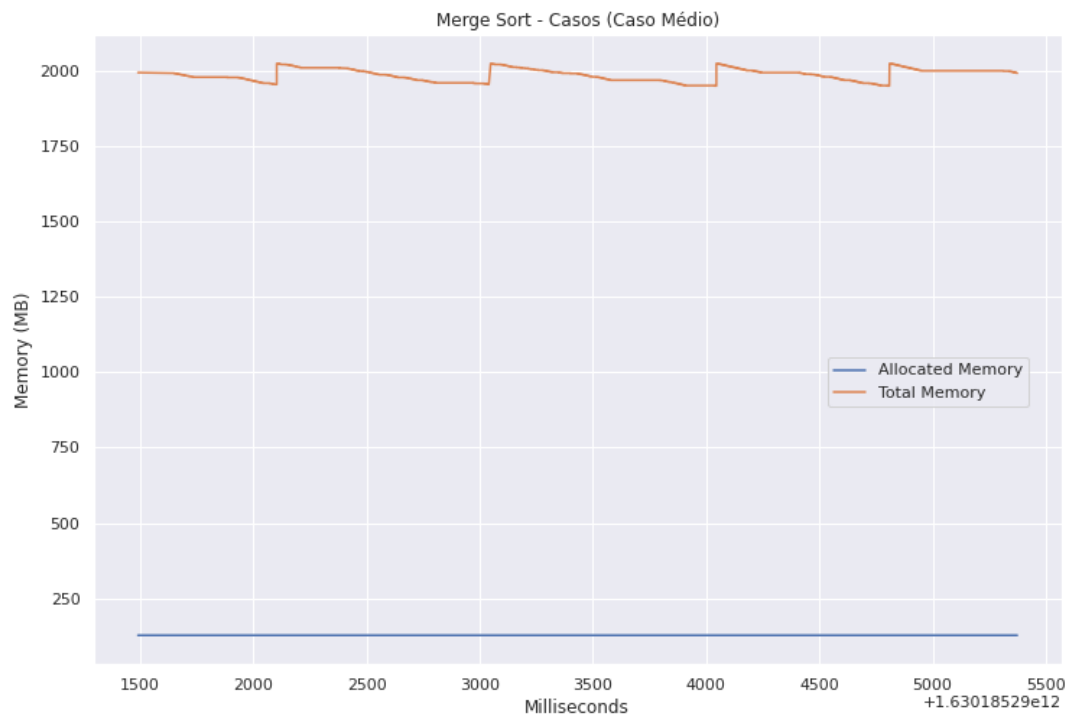


Gráfico 55: Casos Merge Sort (Caso Médio) Máquina 2.

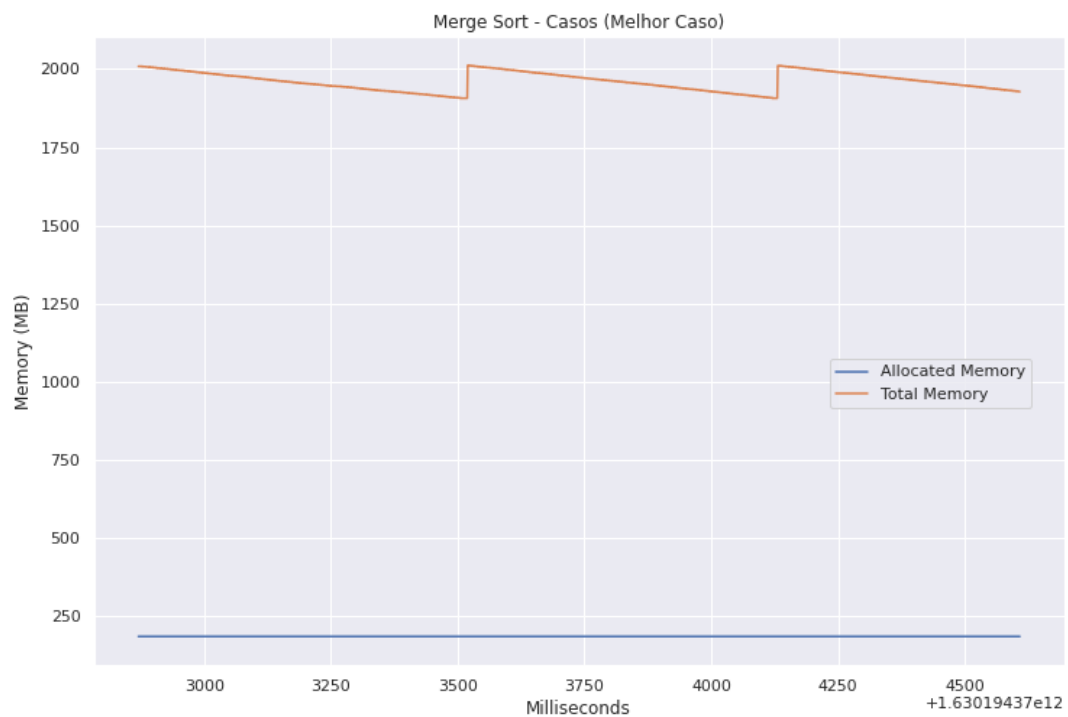


Gráfico 56: Casos Merge Sort (Melhor Caso) Máquina 2.

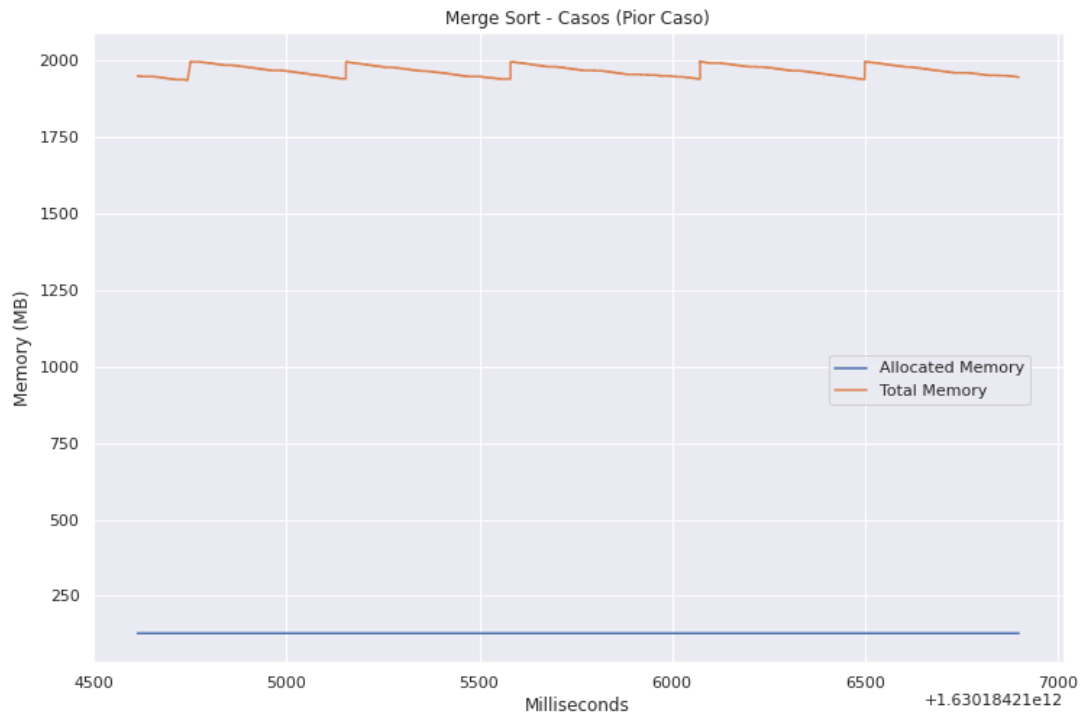


Gráfico 57: Casos Merge Sort (Pior Caso) Máquina 2.

Cidade:

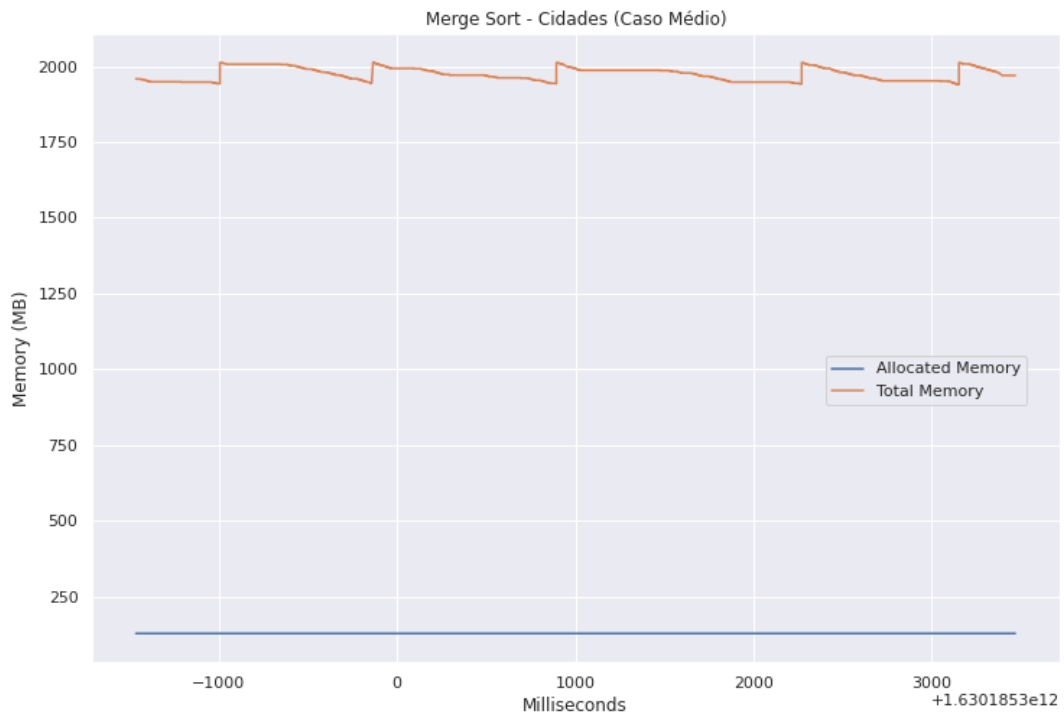


Gráfico 58: Cidades Merge Sort (Caso Médio) Máquina 2.

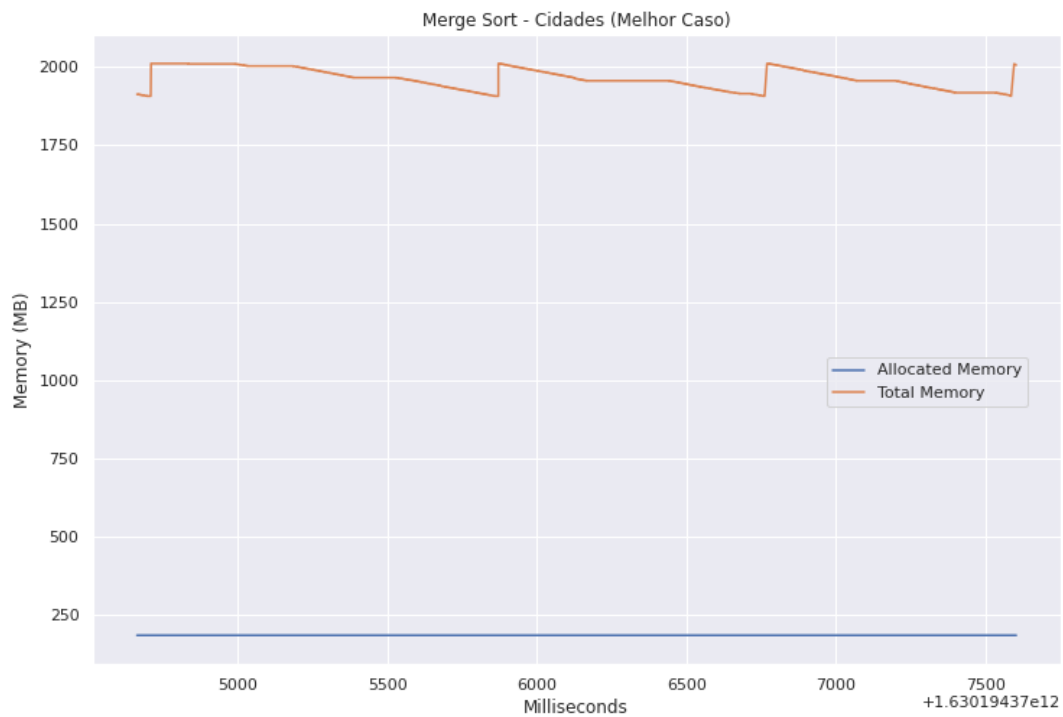


Gráfico 59: Cidades Merge Sort (Melhor Caso) Máquina 2.

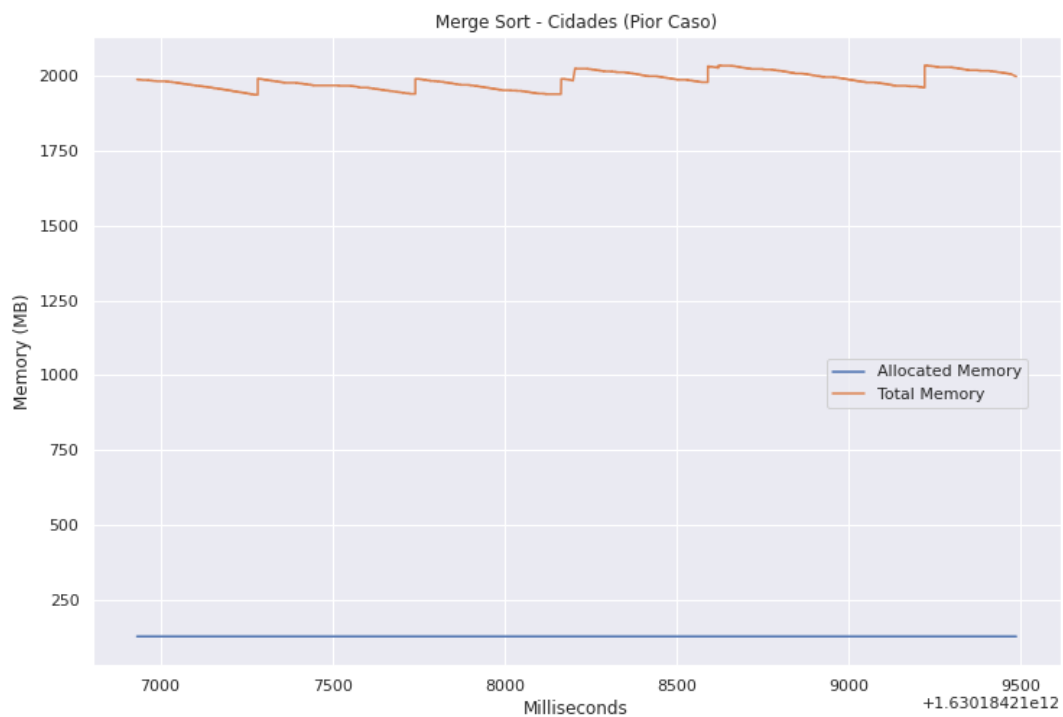


Gráfico 60: Cidades Merge Sort (Pior Caso) Máquina 2.

Quicksort:

Casos:

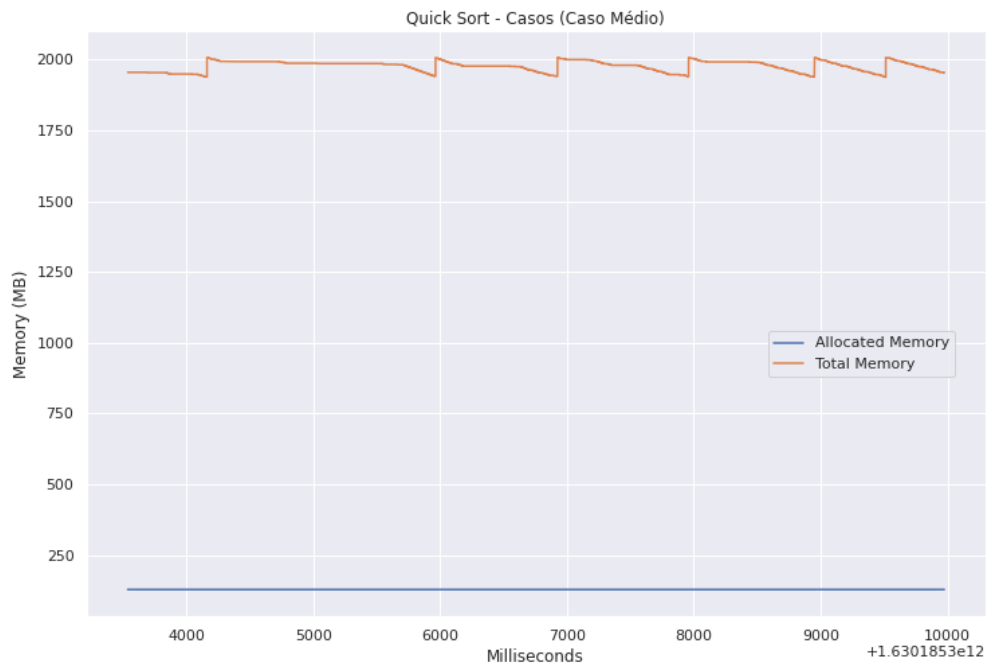


Gráfico 61: Casos Quicksort (Caso Médio) Máquina 2.

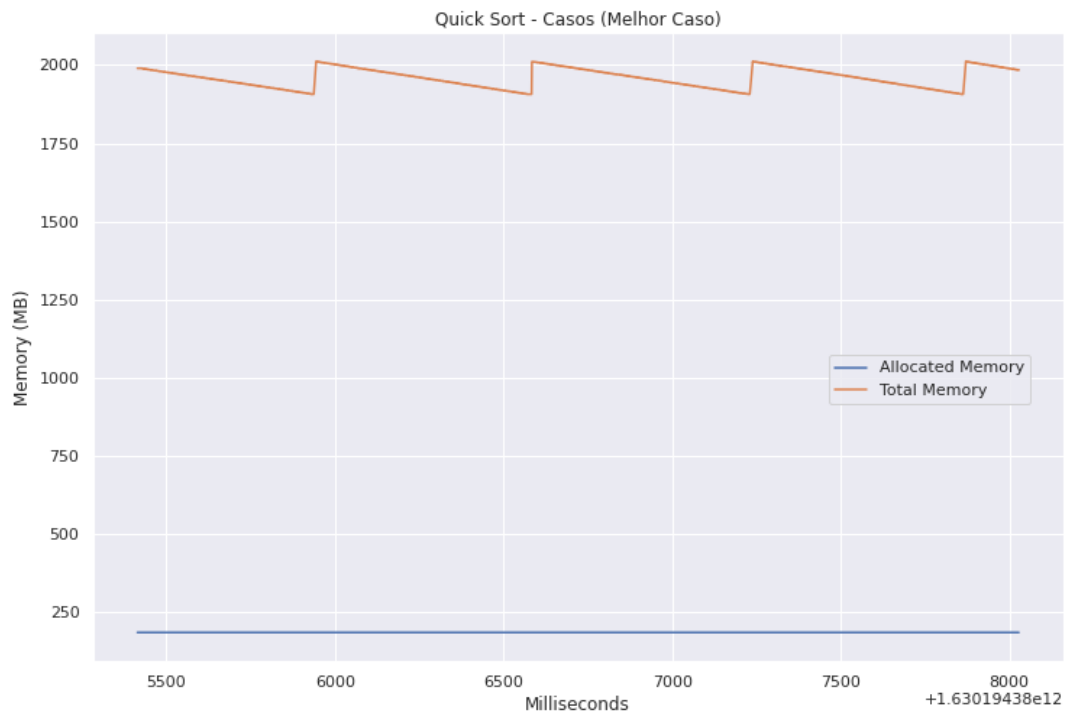


Gráfico 62: Casos Quicksort (Melhor Caso) Máquina 2.

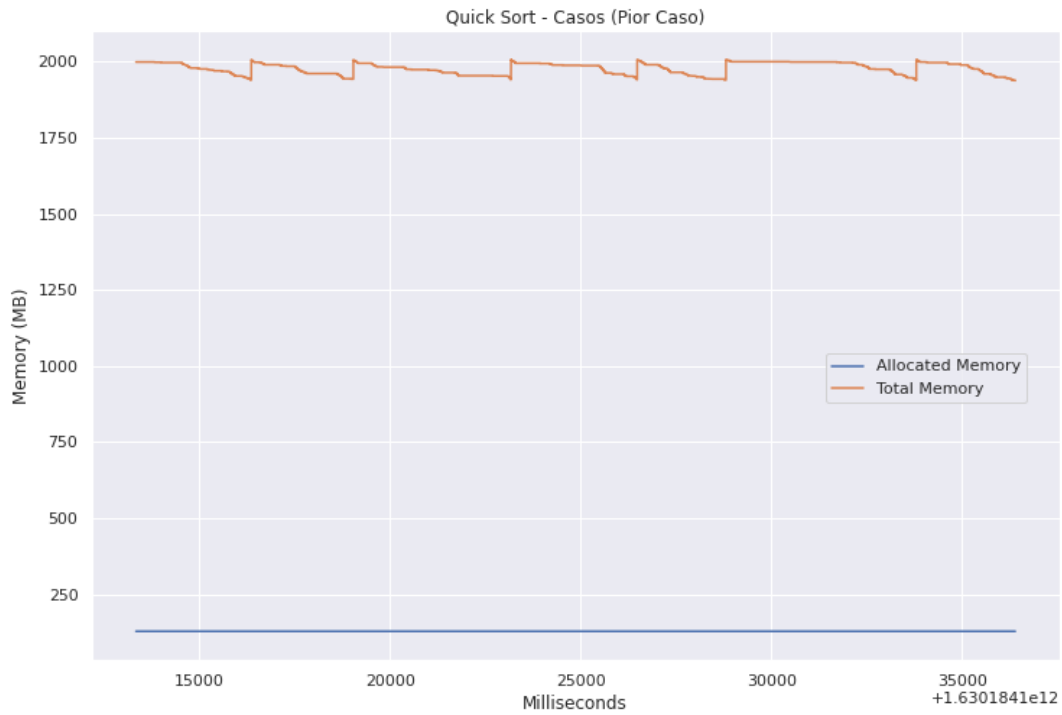


Gráfico 63: Casos Quicksort (Pior Caso) Máquina 2.

Cidade:

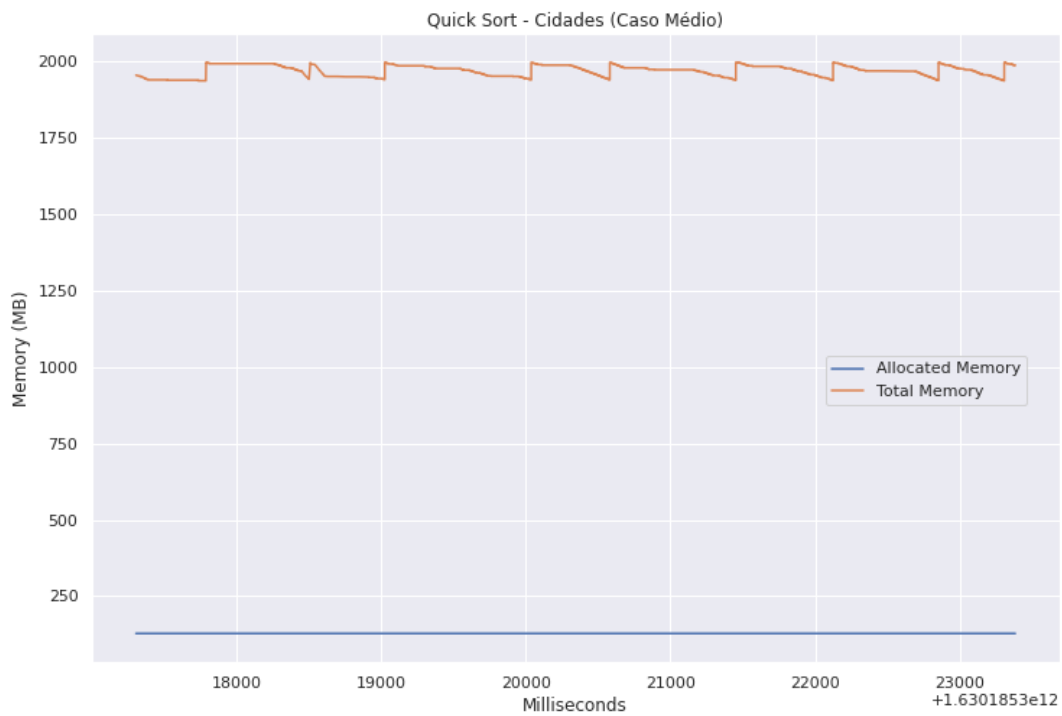


Gráfico 64: Cidades Quicksort (Caso Médio) Máquina 2.

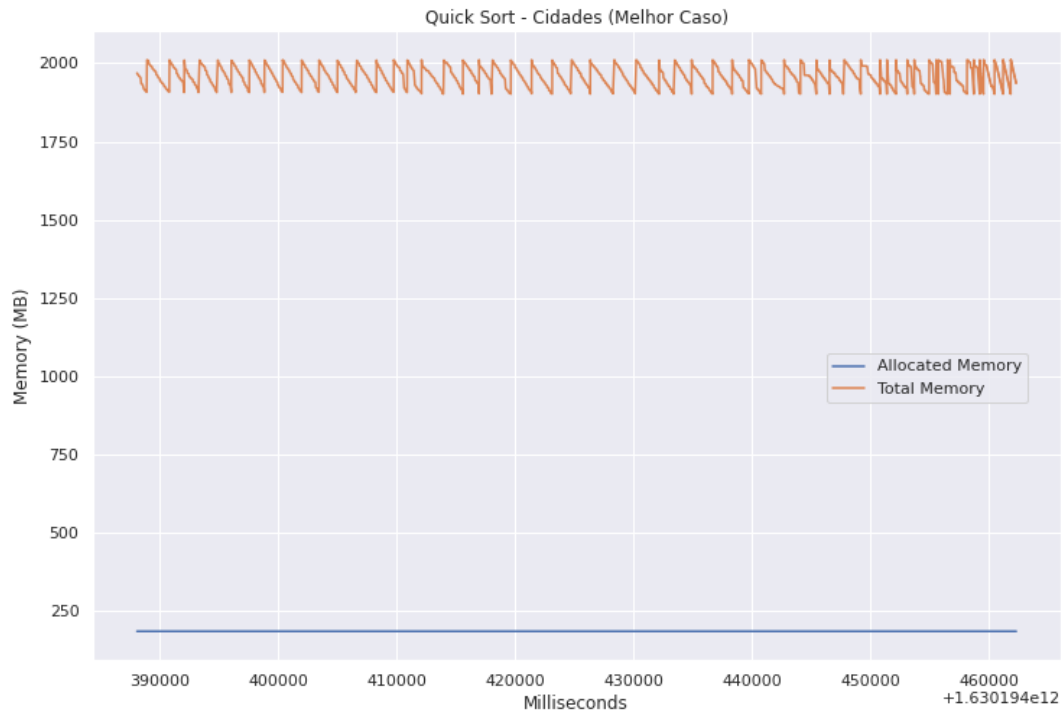


Gráfico 65: Cidades Quicksort (Melhor Caso) Máquina 2.

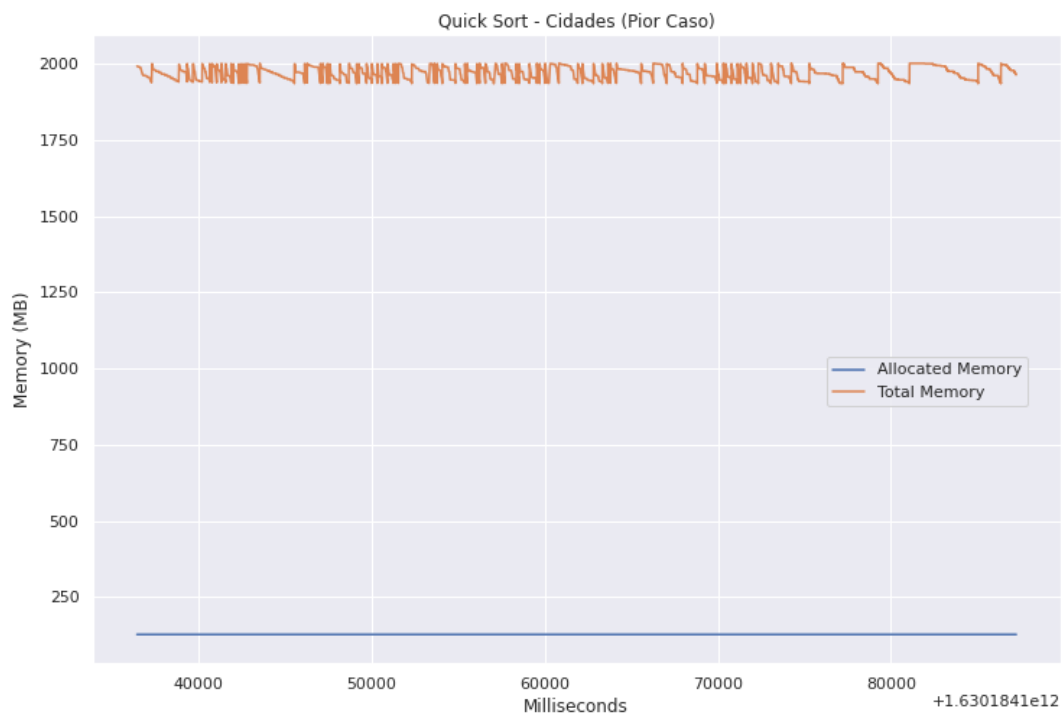


Gráfico 66: Cidades Quicksort (Pior Caso) Máquina 2.

Quicksort com Mediana de 3:

Casos:

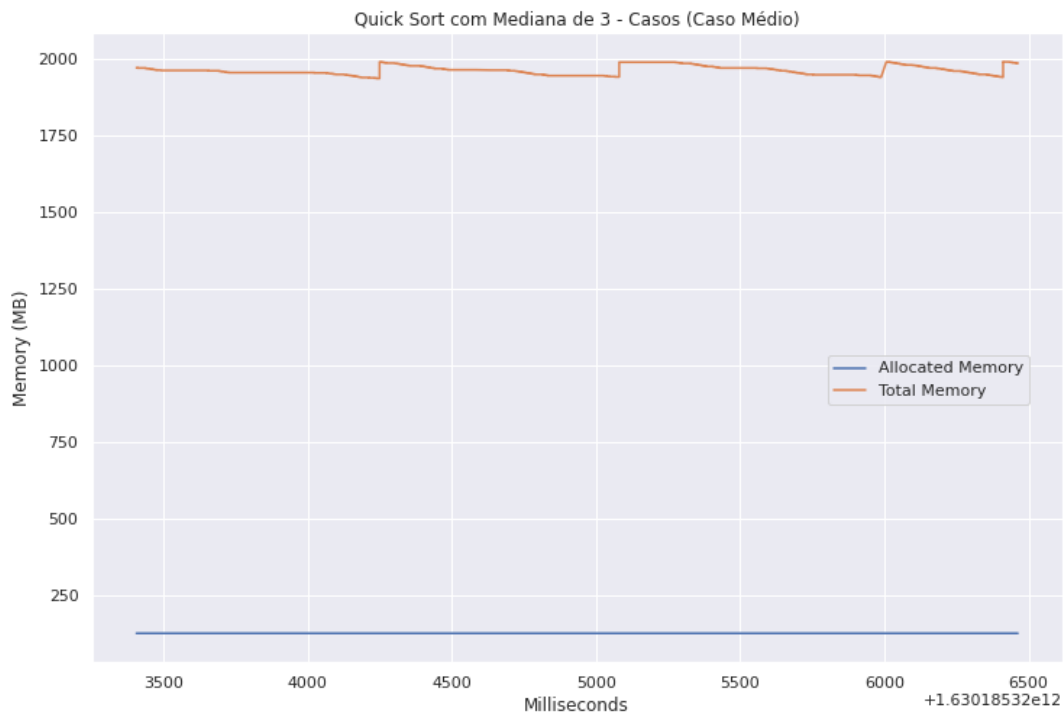


Gráfico 67: Casos Quicksort com mediana de 3 (Caso Médio) Máquina 2.

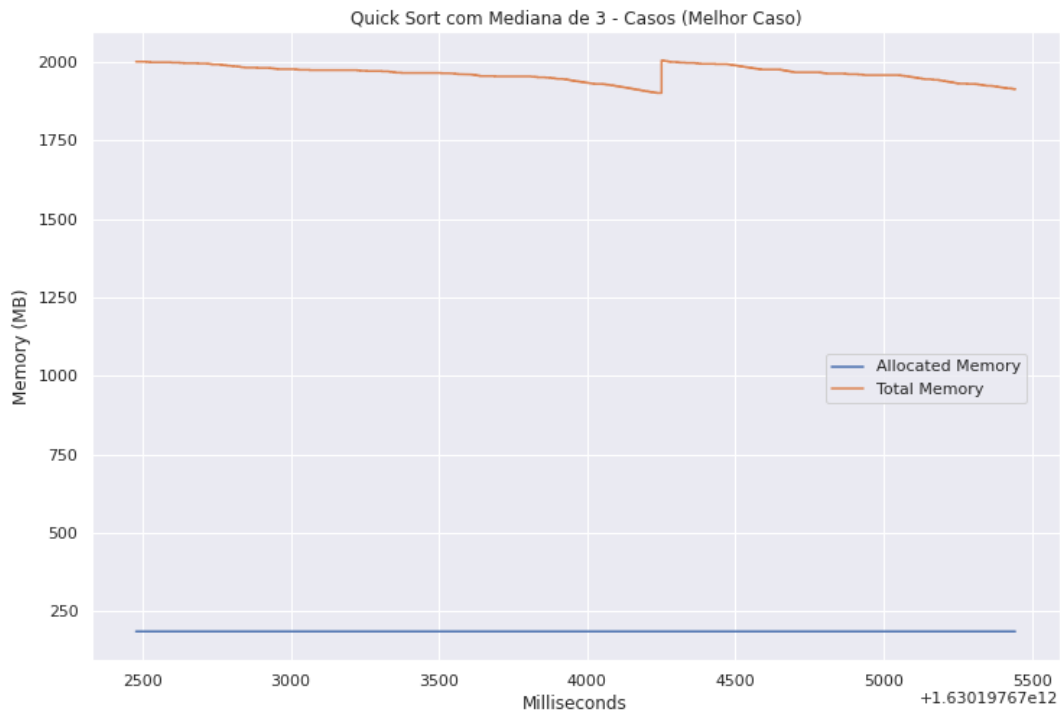


Gráfico 68: Casos Quicksort com mediana de 3 (Melhor Caso) Máquina 2.

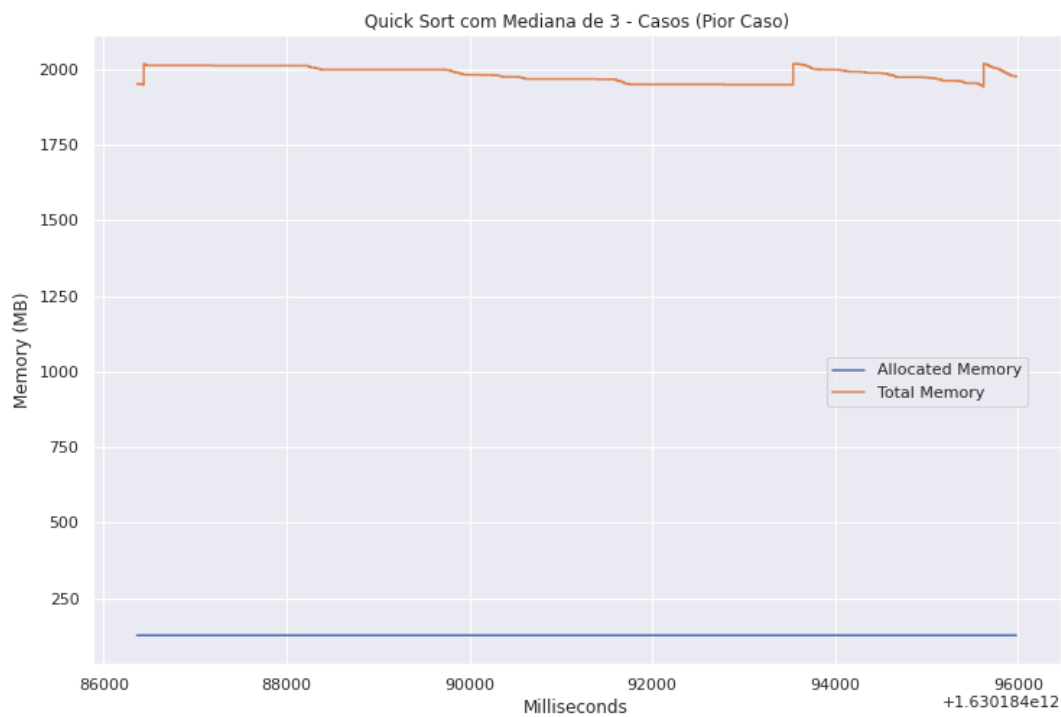


Gráfico 69: Casos Quicksort com mediana de 3 (Pior Caso) Máquina 2.

Cidade:

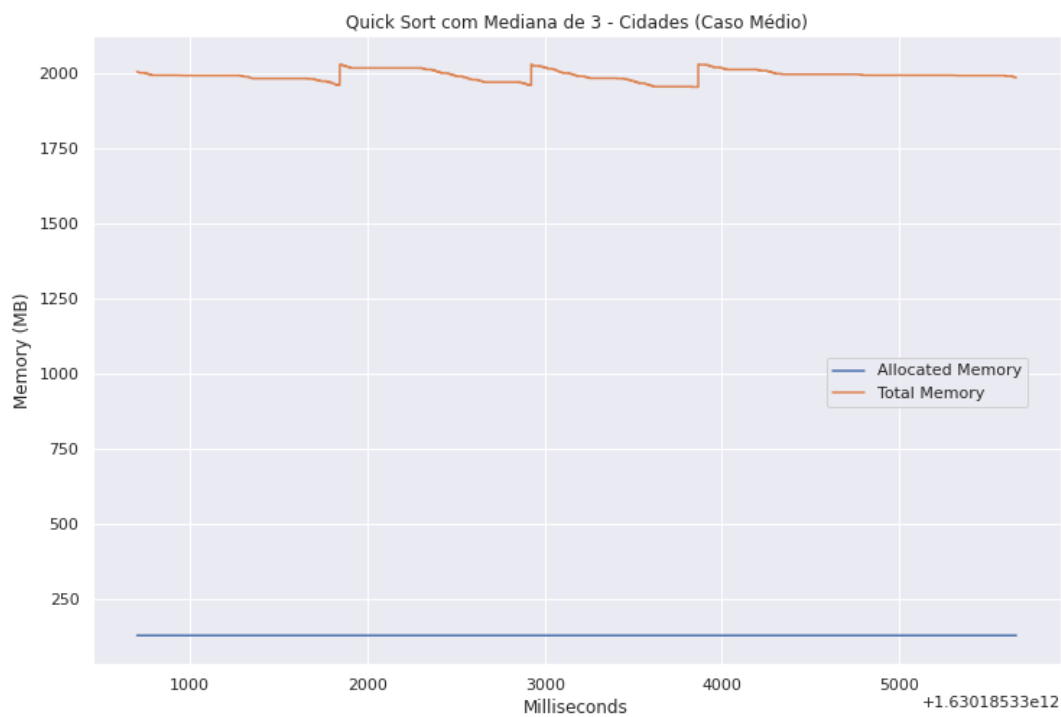


Gráfico 70: Cidades Quicksort com mediana de 3 (Caso Médio) Máquina 2.

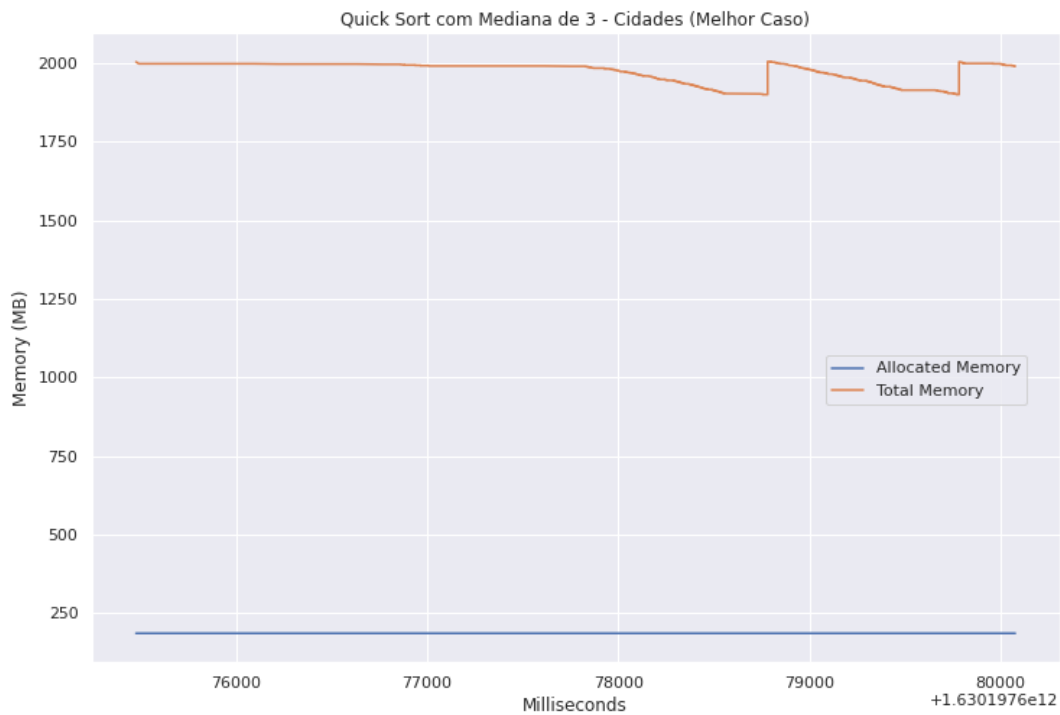


Gráfico 71: Cidades Quicksort com mediana de 3 (Melhor Caso) Máquina 2.

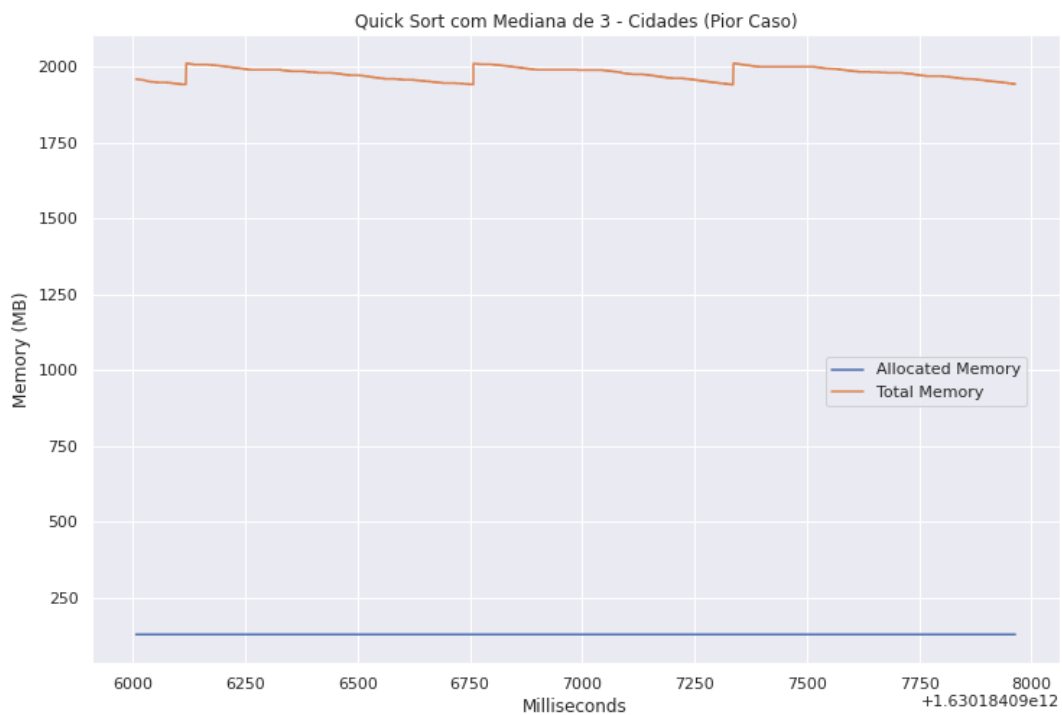


Gráfico 72: Cidades Quicksort com mediana de 3 (Pior Caso) Máquina 2.

Heapsort:

Casos:

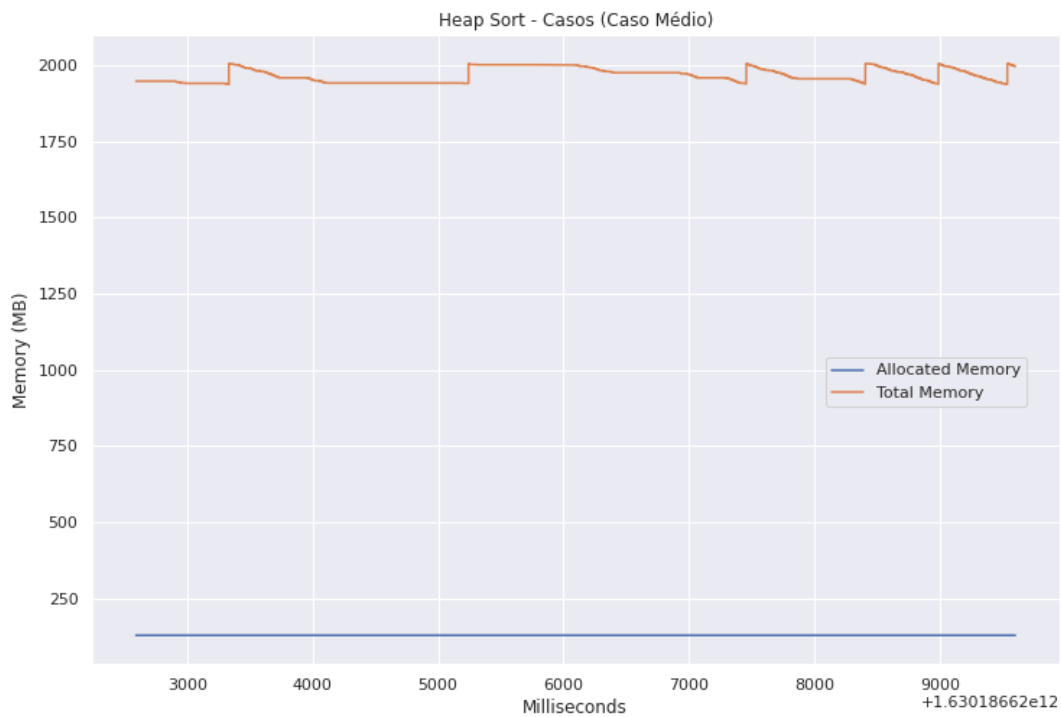


Gráfico 73: Casos Heapsort (Caso Médio) Máquina 2.

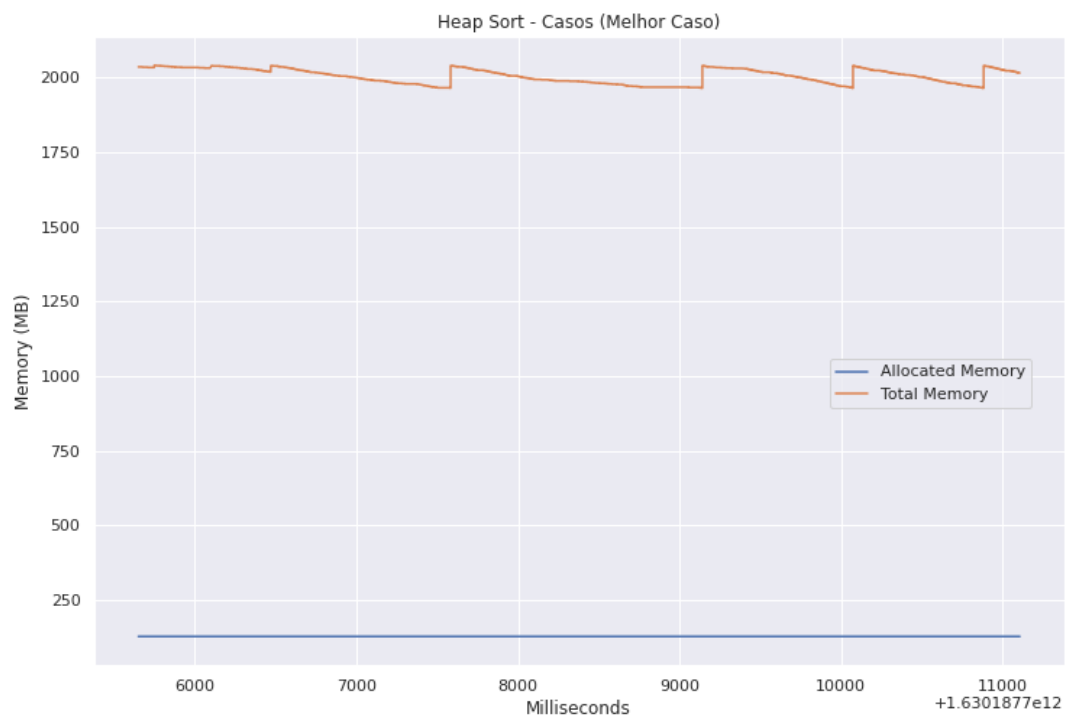


Gráfico 74: Casos Heapsort (Melhor Caso) Máquina 2.

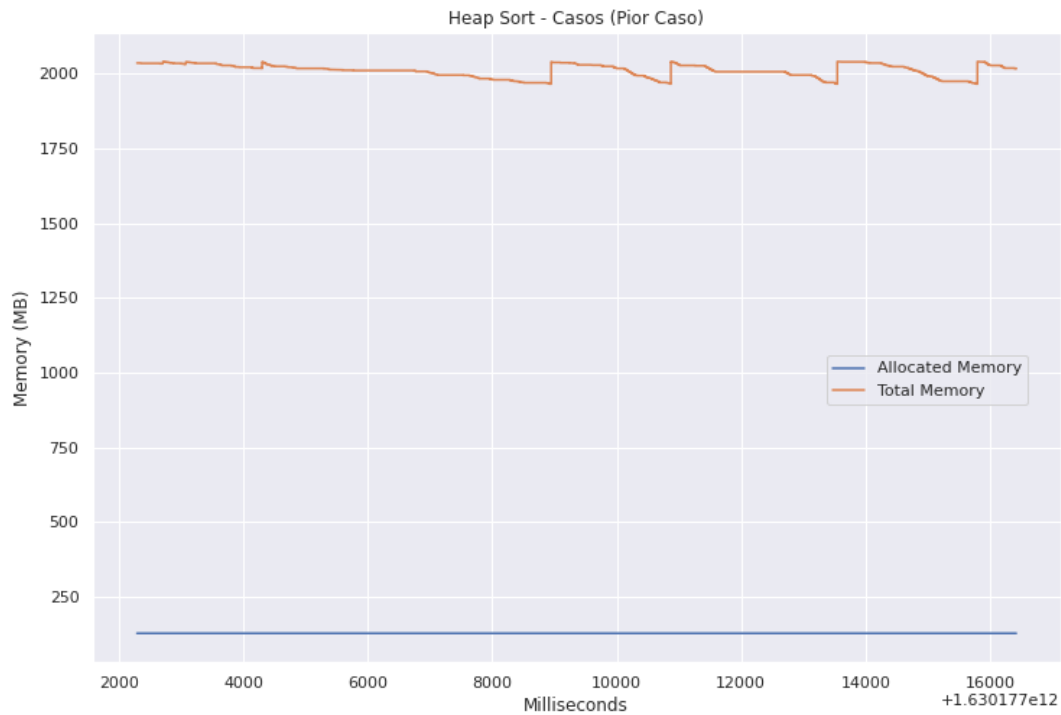


Gráfico 75: Casos Heapsort (Pior Caso) Máquina 2.

Cidade:

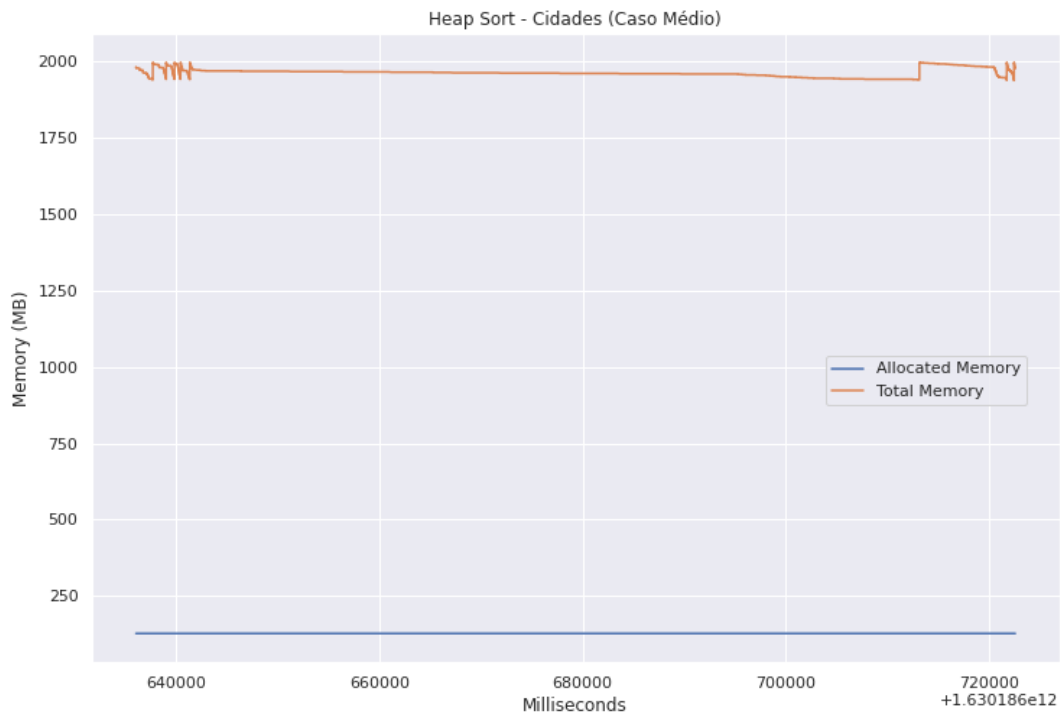


Gráfico 76: Cidades Heapsort (Caso Médio) Máquina 2.

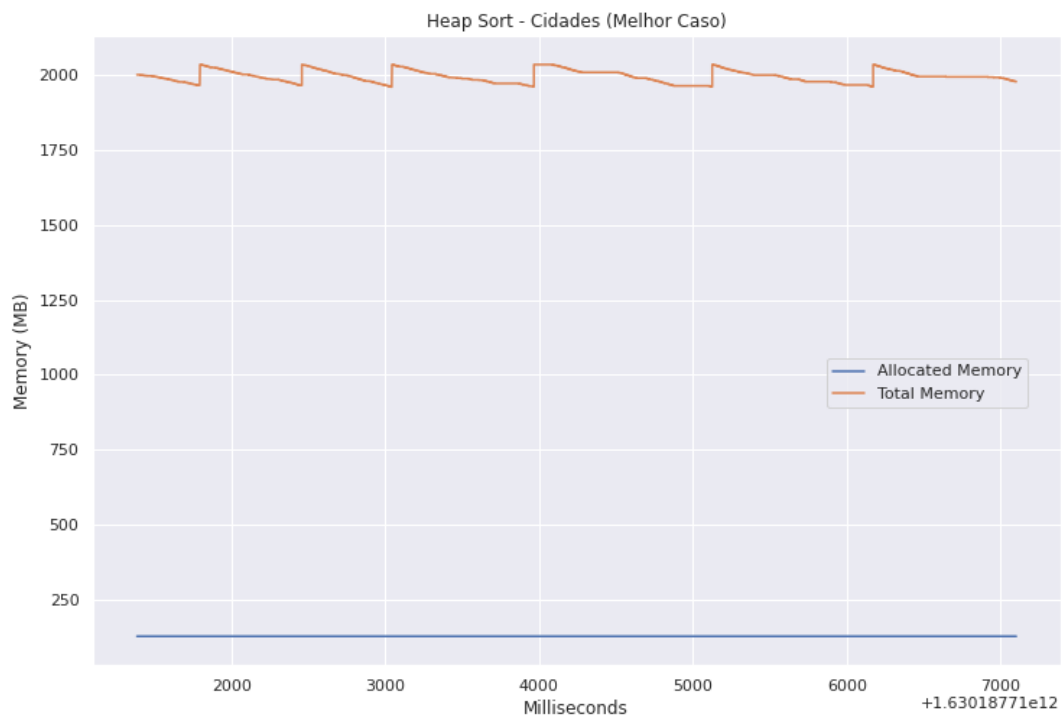


Gráfico 77: Cidades Heapsort (Melhor Caso) Máquina 2.

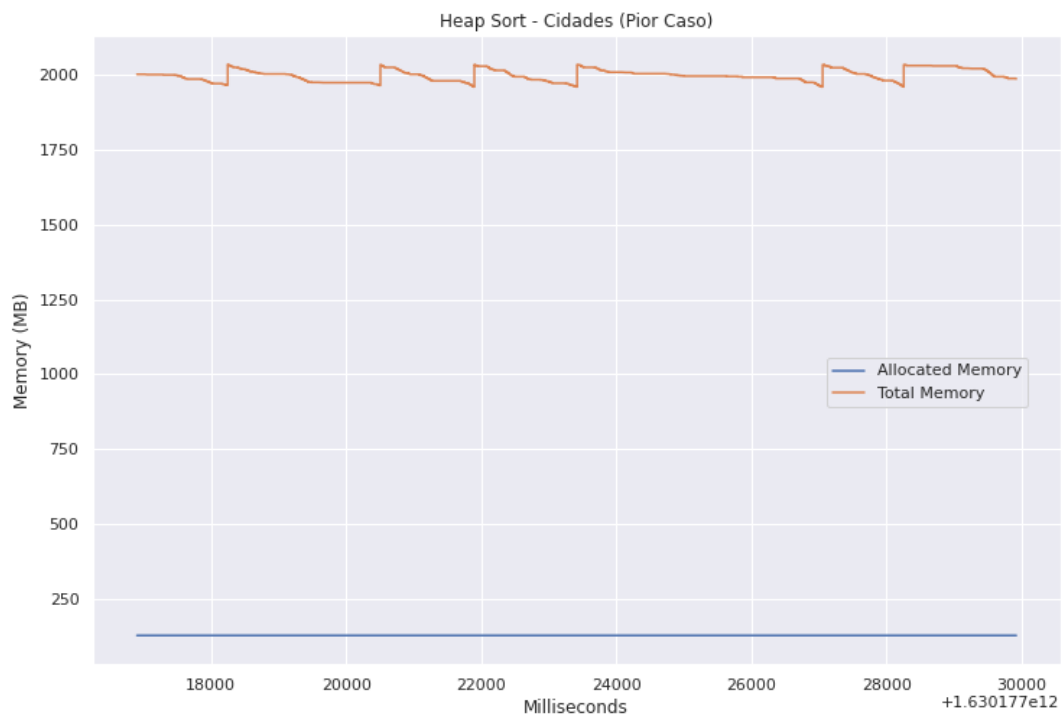


Gráfico 78: Cidades Heapsort (Pior Caso) Máquina 2.

Counting Sort:

Casos:

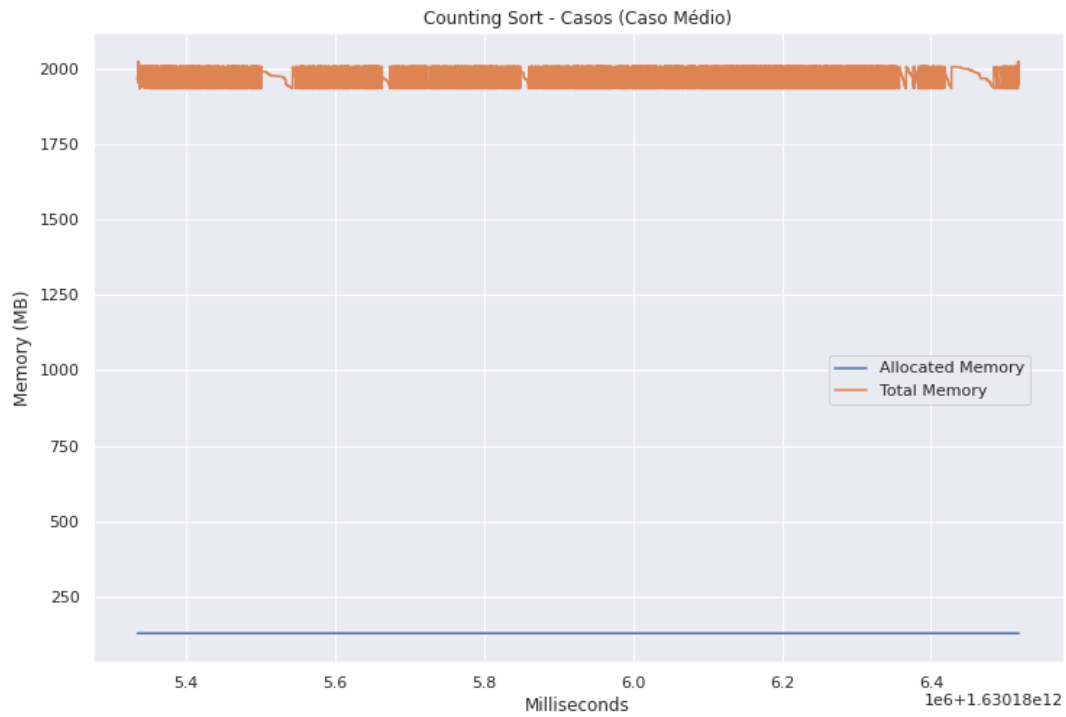


Gráfico 79: Casos Counting Sort (Caso Médio) Máquina 2.

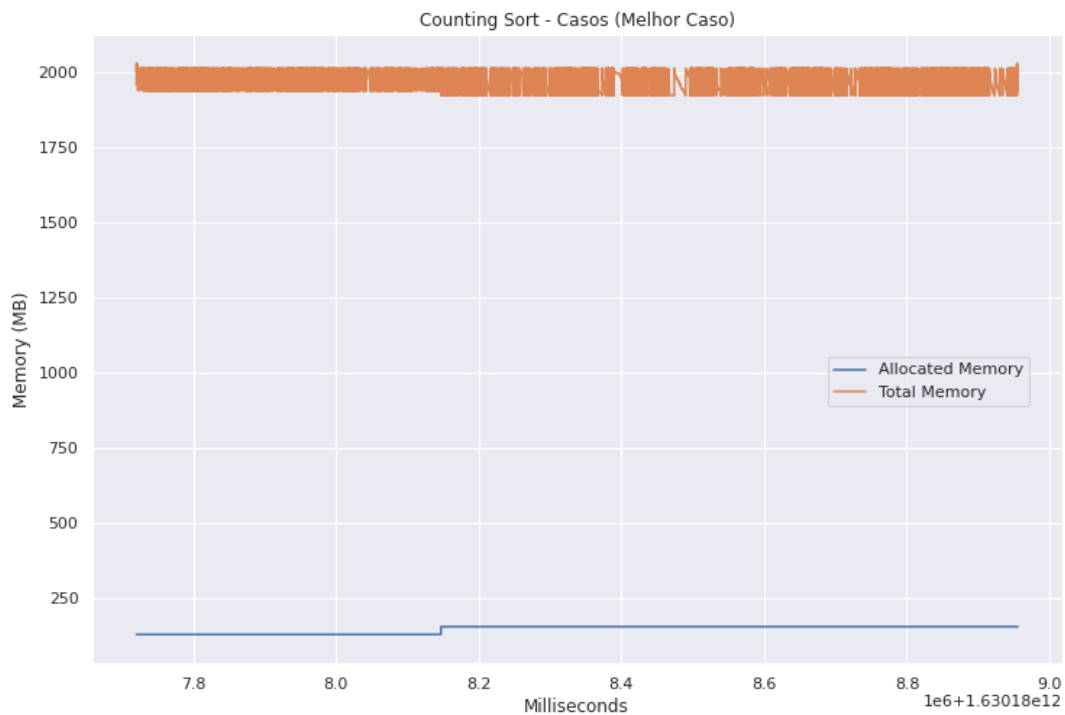


Gráfico 80: Casos Counting Sort (Melhor Caso) Máquina 2.

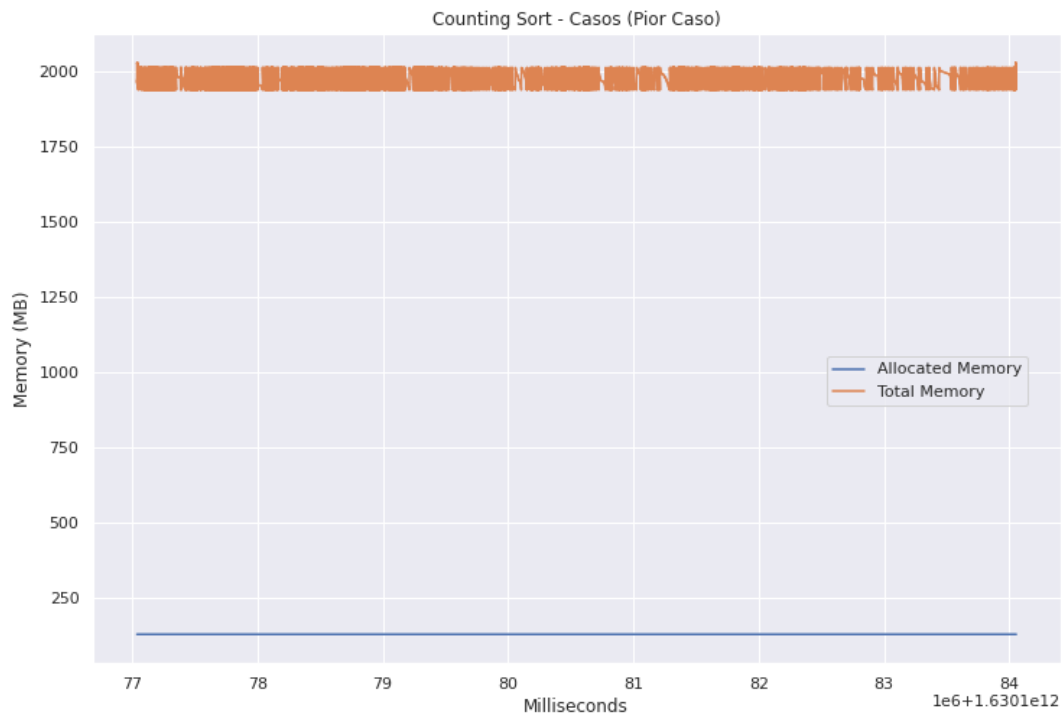


Gráfico 81: Casos Counting Sort (Pior Caso) Máquina 2.

Cidade:

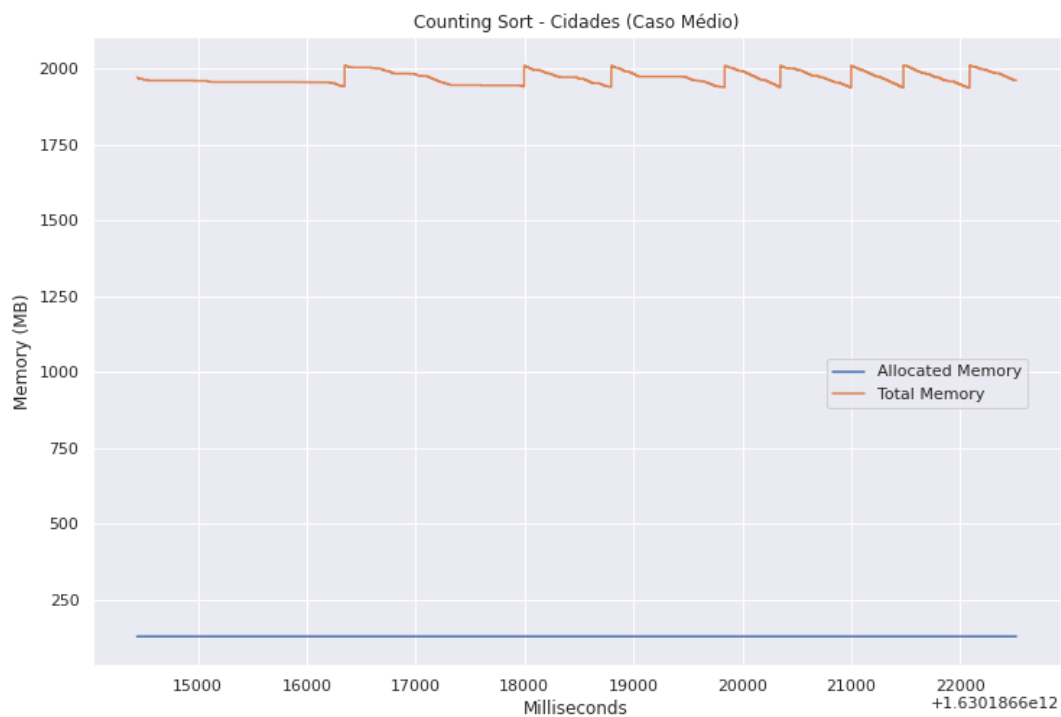


Gráfico 82: Cidades Counting Sort (Caso Médio) Máquina 2.

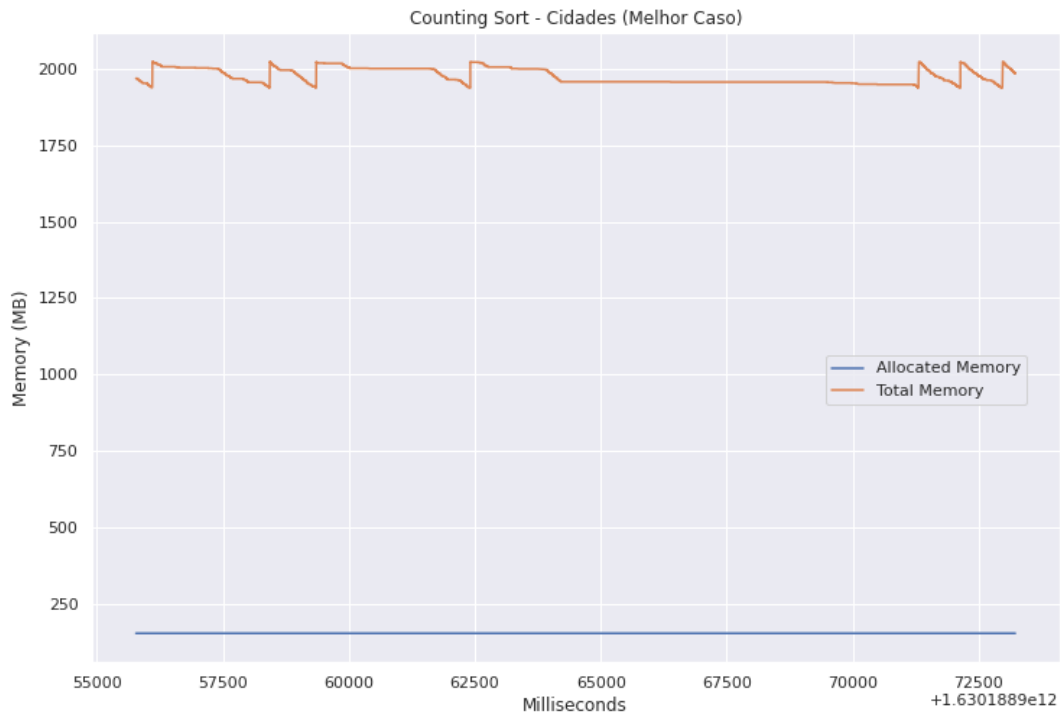


Gráfico 83: Cidades Counting Sort (Melhor Caso) Máquina 2.

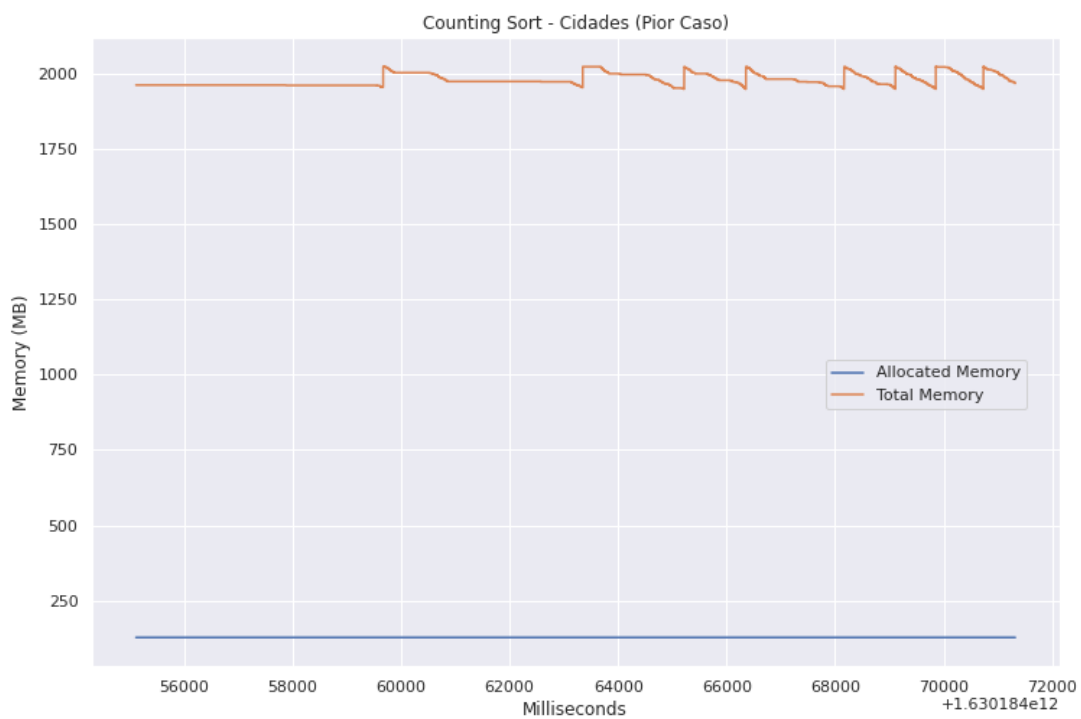


Gráfico 84: Cidades Counting Sort (Pior Caso) Máquina 2.

Resultados dos testes usando gráficos

Por fim, após a análise de todos os gráficos gerados nas métricas do sistema, podemos comparar os resultados da execução entre as duas máquinas e dos diferentes algoritmos de ordenação. Após a análise dos gráficos pode-se perceber que na máquina 1 e na máquina 2 o algoritmo mais rápido foi Heapsort, no entanto em casos gerais o algoritmo mais eficientes serão o Heap Sort, Merge Sort e o Quicksort mediana de 3, pois nas duas máquinas são o que mantém a média do menor consumo de memória e um menor consumo de tempo. Vale salientar que os dados podem ter divergência devido à máquina e o número de programas rodando simultaneamente ao sistema.

Em uma análise geral, concluímos que o Heap Sort é o algoritmo mais eficiente, apesar de na segunda máquina ter demorado um pouco mais, o consumo de memória foi reduzido. Já na primeira máquina realmente o Heap Sort foi o mais eficiente, tanto no consumo de memória como na velocidade de processamento.