Universidade Federal do Pampa

Trabalho sobre o Merge Sort: uma proposta de um algoritmo de ordenação paralelo.

- Orientador: Professor Bruno S. Neves.
- Nome do Componente 1: Luciano Brum.
- Nome do Componente 2: Thiago Dantas.
- Disciplina: Sistemas Operacionais.

Algoritmos de Ordenação

- Introdução: Qual o objetivo da pesquisa?

- Como obter maiores benefícios dos algoritmos de ordenação?

Algoritmos de Ordenação

- Algoritmos de Ordenação Estudados: Bubble Sort, Insertion Sort, Selection Sort, Quick Sort, Heap Sort e Merge Sort

- Filosofia Dividir e Conquistar. (são altamente recursivos, Merge Sort é um exemplo de uso desta filosofia).

- Qual a complexidade do algoritmo? Em média O(n log n)

Algoritmo de Ordenação: Merge Sort

- Como funciona o algoritmo Merge Sort na versão sequencial.(Recursivo).

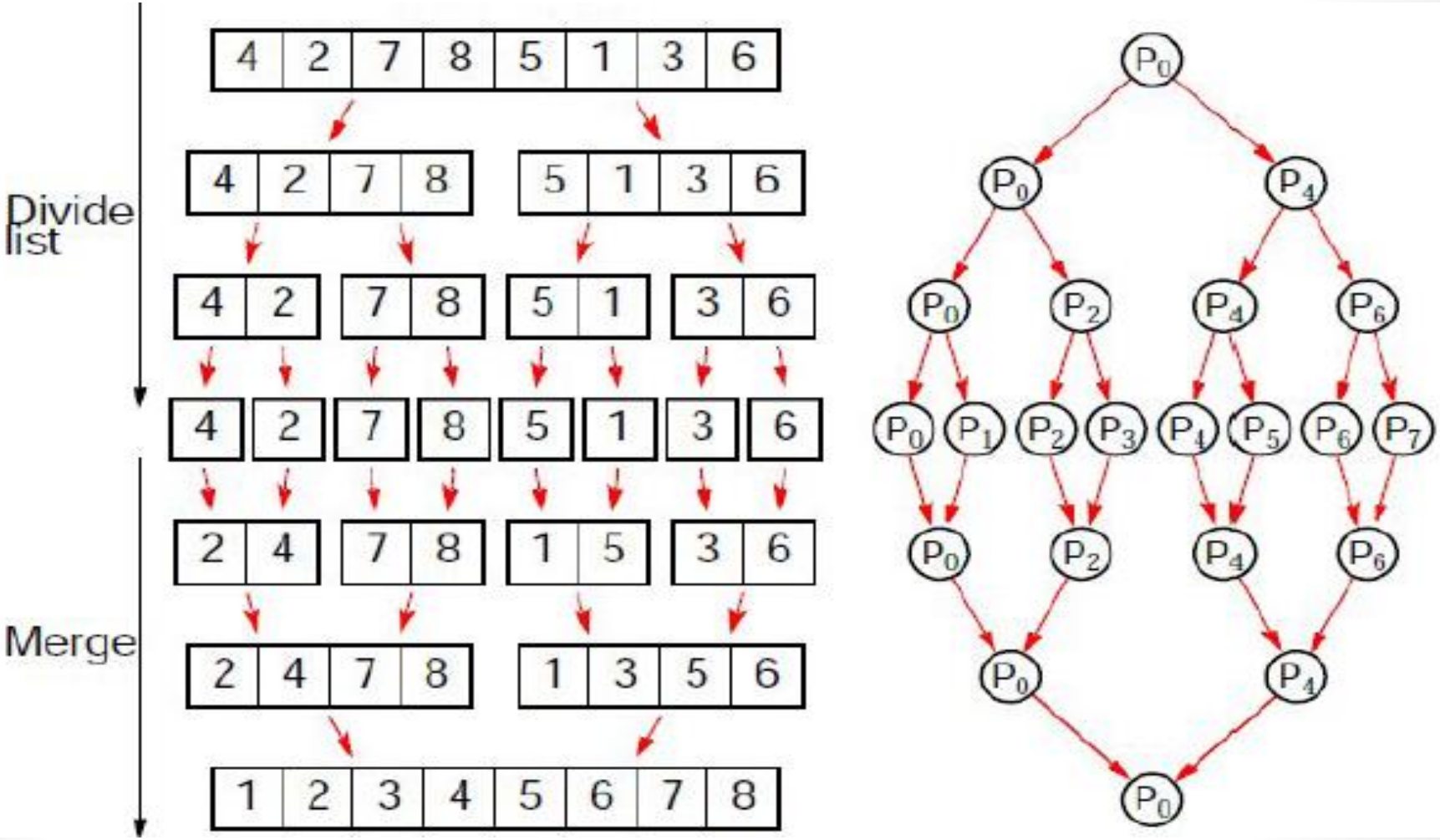
- Quais são os ganhos na implementação paralela? (Desempenho).

Merge Sort

- Como foi a implementação do algoritmo na versão paralela com 2 ou mais threads? (figura 1).

- O que aconteceu com o tempo de execução com um número grande de threads. (overheads na troca de contexto).

Merge Sort



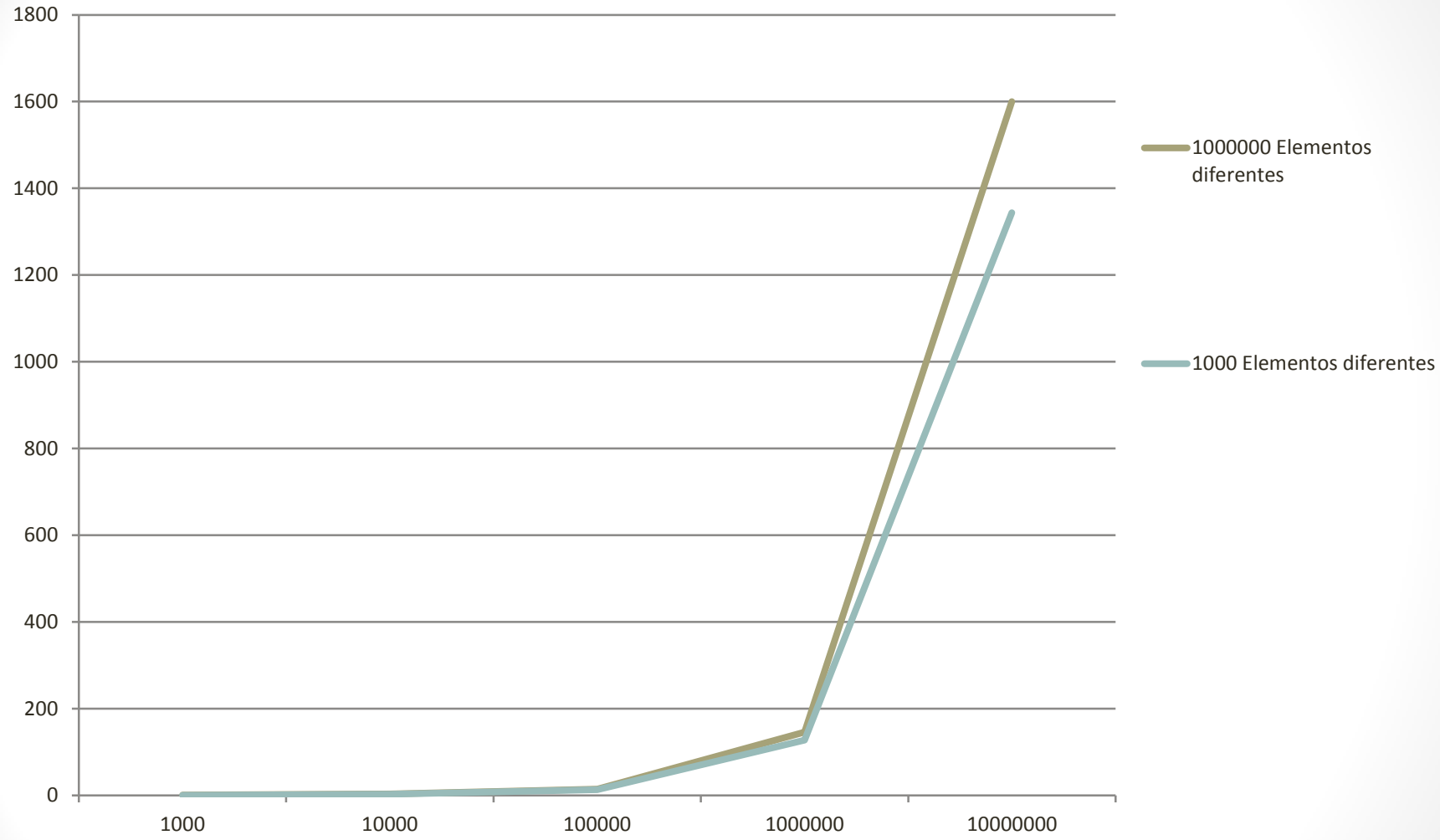
Exemplo de execução do merge sort paralelo para 2 ou mais Threads. Figura 1.

Merge Sort

- Análise final do desempenho do algoritmo para uma versão sequencial. (Ver figura 2)

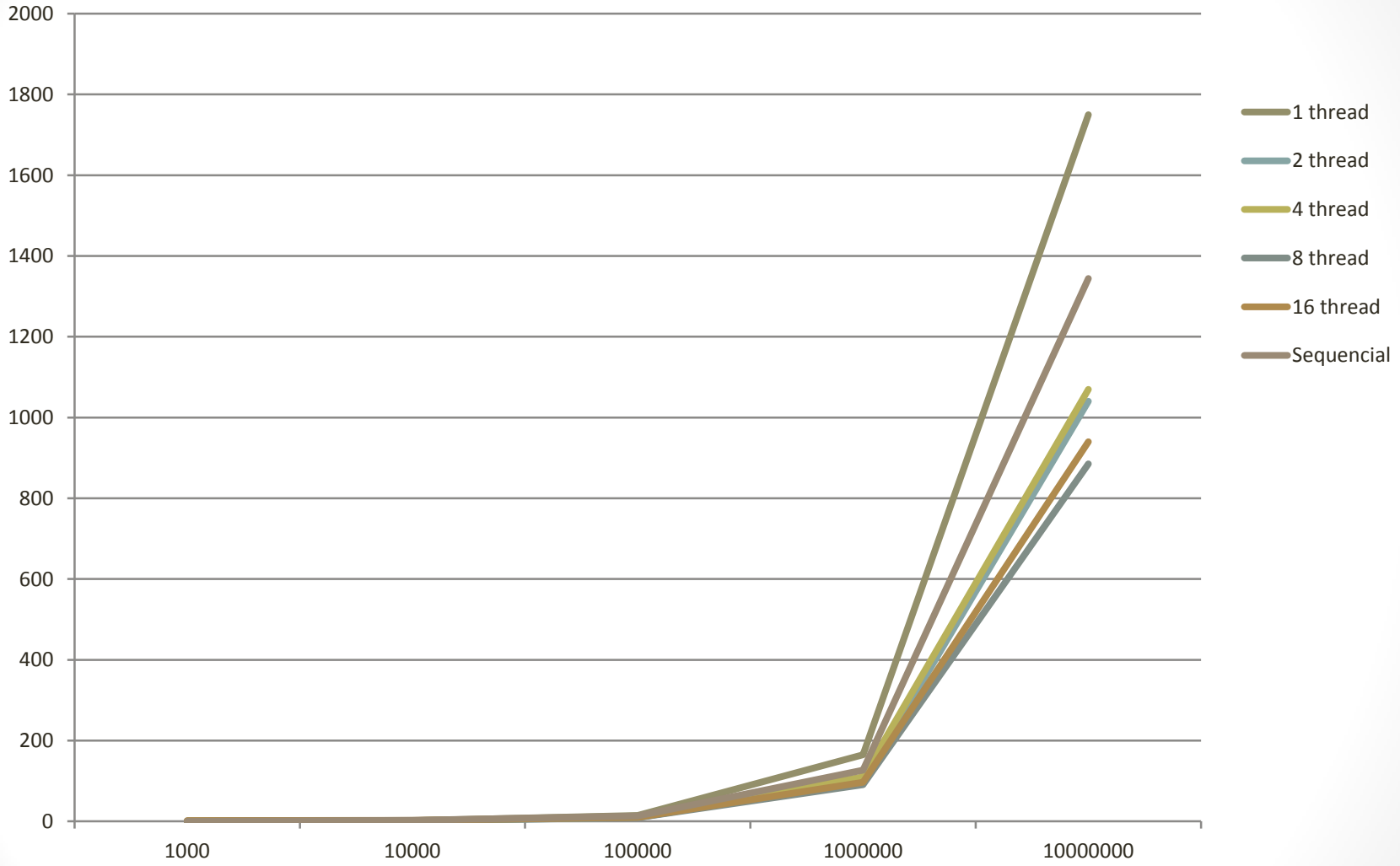
- Análise do desempenho do algoritmo para uma versão paralela com 1 ou mais threads. (ver figura 3)

Merge Sort



- **Figura 2: Tempo de execução médio para 5 tamanhos de vetores e para um número de repetições de elementos grande e pequeno.**

Merge Sort

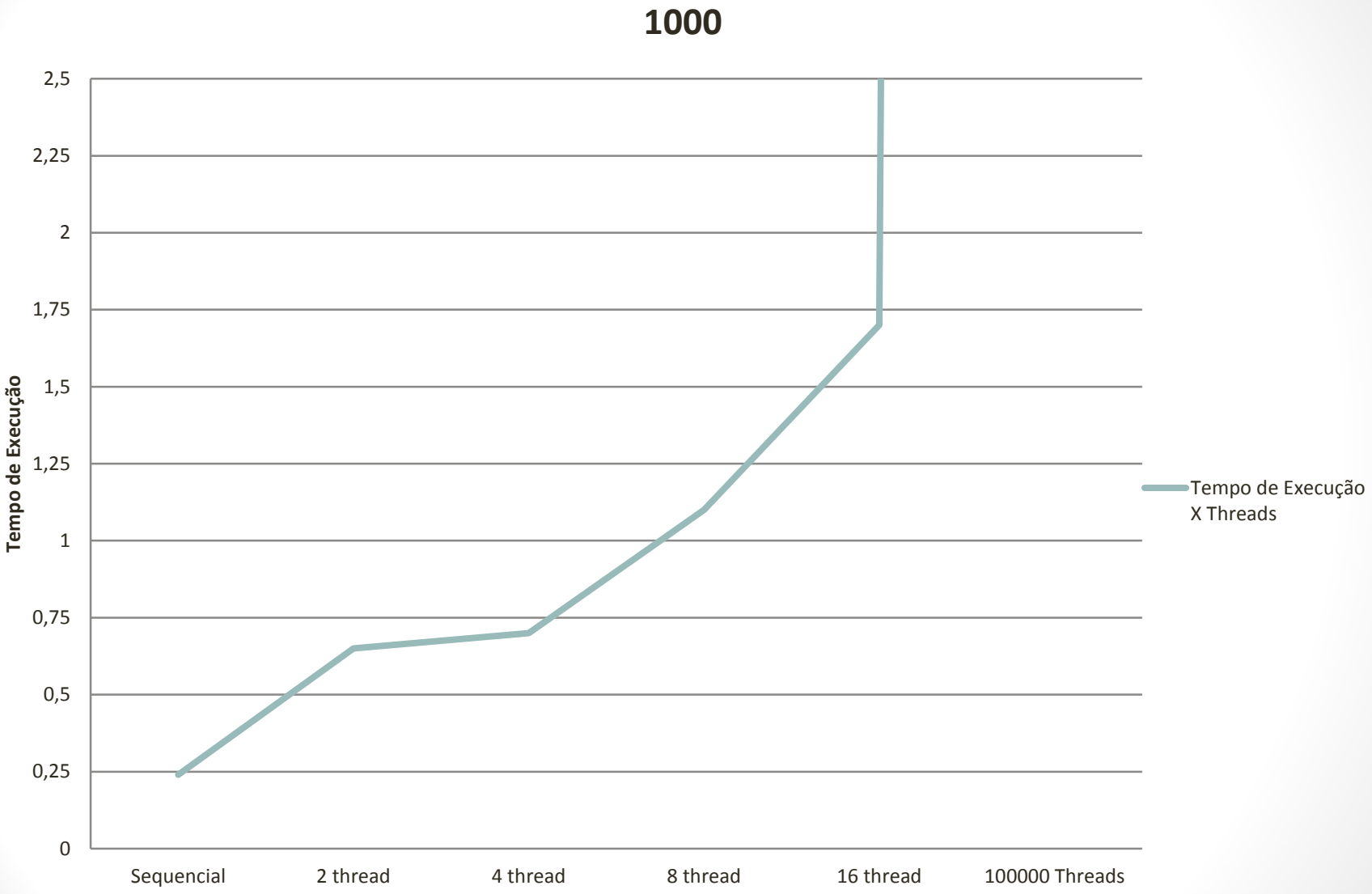


- **Figura 3: Tempo de execução médio para 5 tamanhos de vetores e para 1000 elementos diferentes.**

Merge Sort

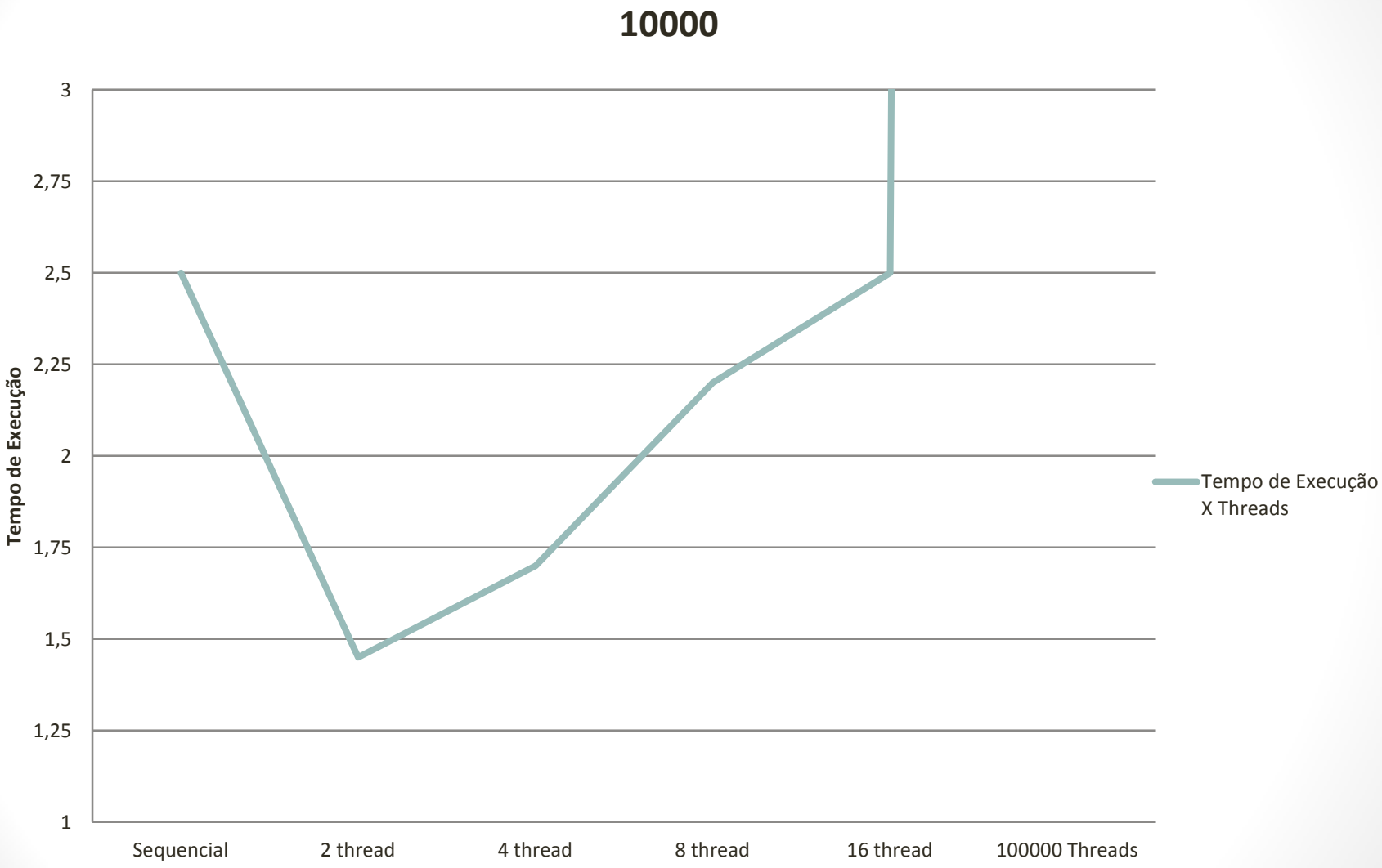
- Análise de desempenho para cada tamanho de vetor para cada caso nas próximas figuras. (ver figuras 4, 5, 6, 7 e 8)

Merge Sort



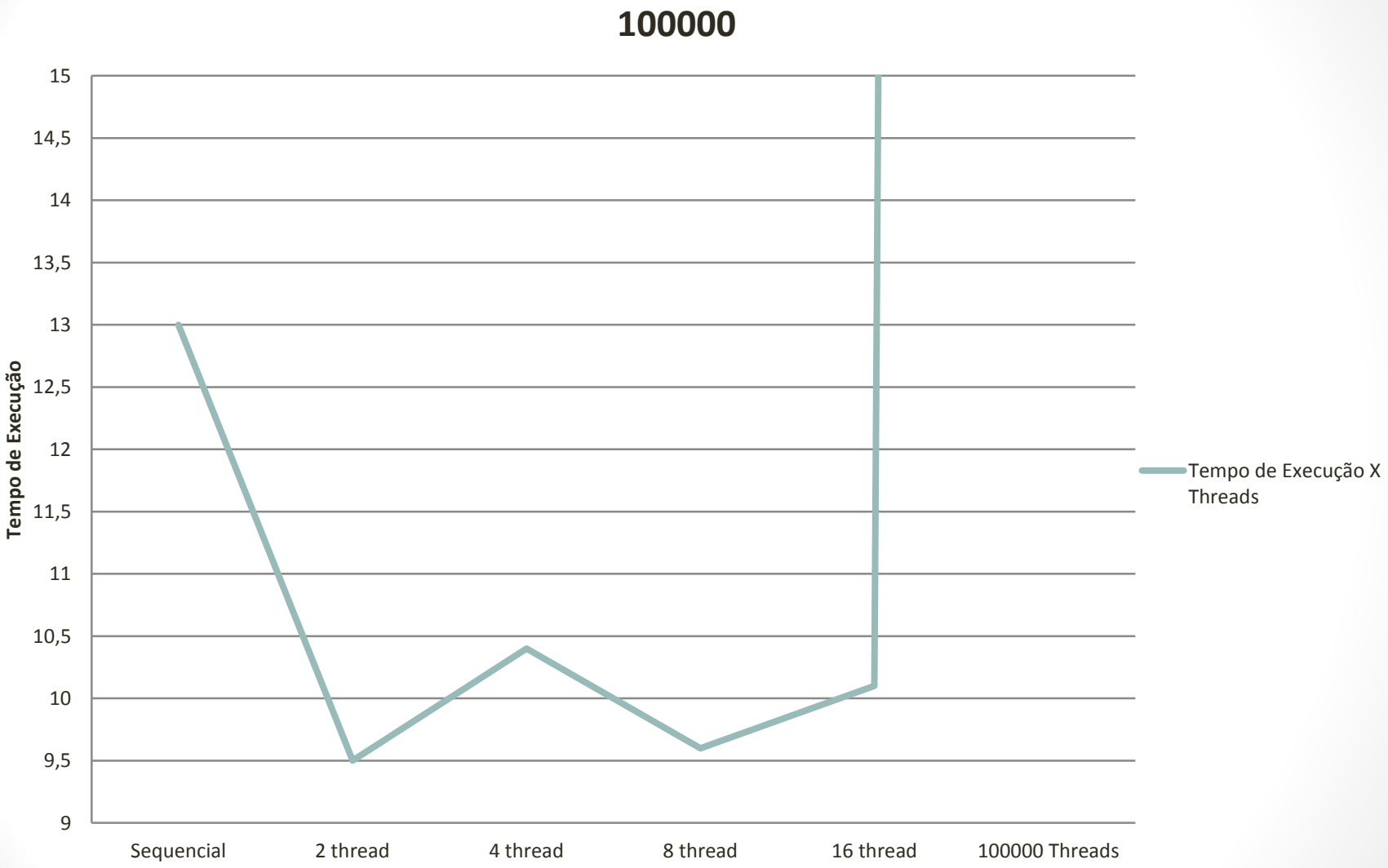
- **Figura 4: Tempo de execução médio para 5 tamanhos de vetores e para 1000 elementos diferentes.**

Merge Sort



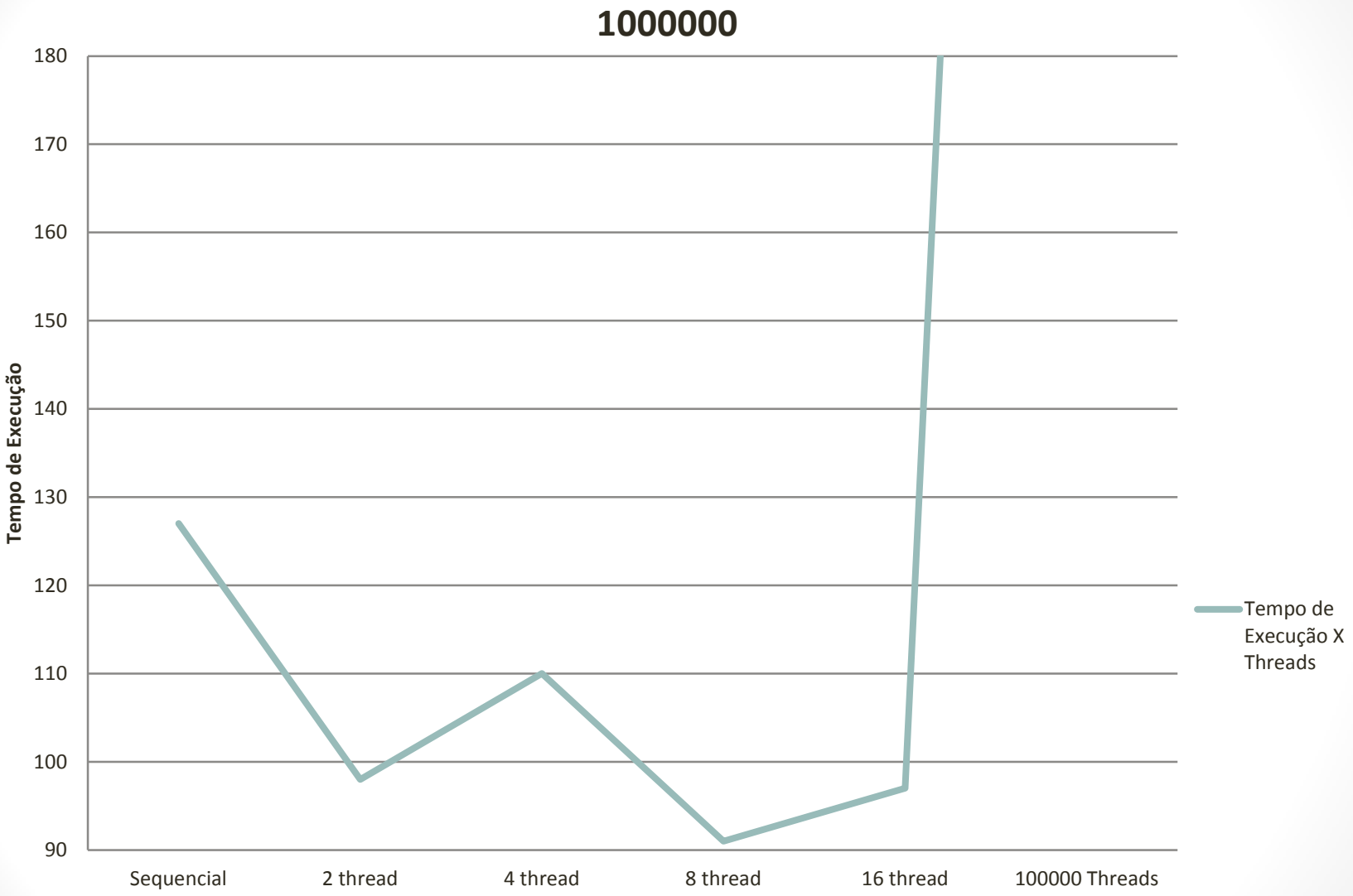
- **Figura 5: Tempo de execução médio para 5 tamanhos de vetores e para 1000 elementos diferentes.**

Merge Sort



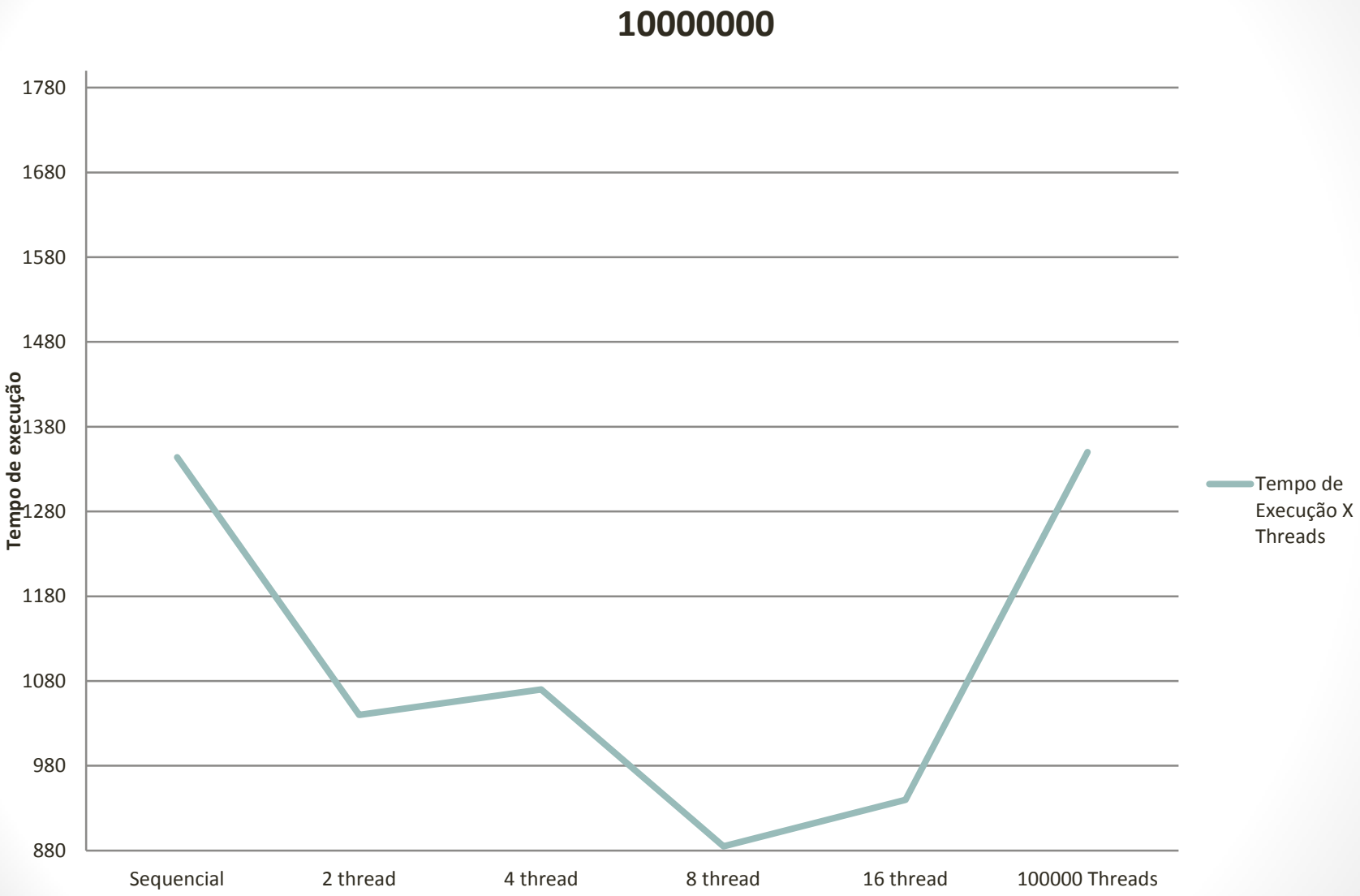
- **Figura 6: Tempo de execução médio para 5 tamanhos de vetores e para 1000 elementos diferentes.**

Merge Sort



- **Figura 7: Tempo de execução médio para 5 tamanhos de vetores e para 1000 elementos diferentes.**

Merge Sort



- **Figura 8: Tempo de execução médio para 5 tamanhos de vetores e para 1000 elementos diferentes.**

Merge Sort

- Como solução de sincronismo de threads utilizamos a palavra-chave “synchronyzed”.

- Resultado: Desastre! (veja figura 9).

Merge Sort

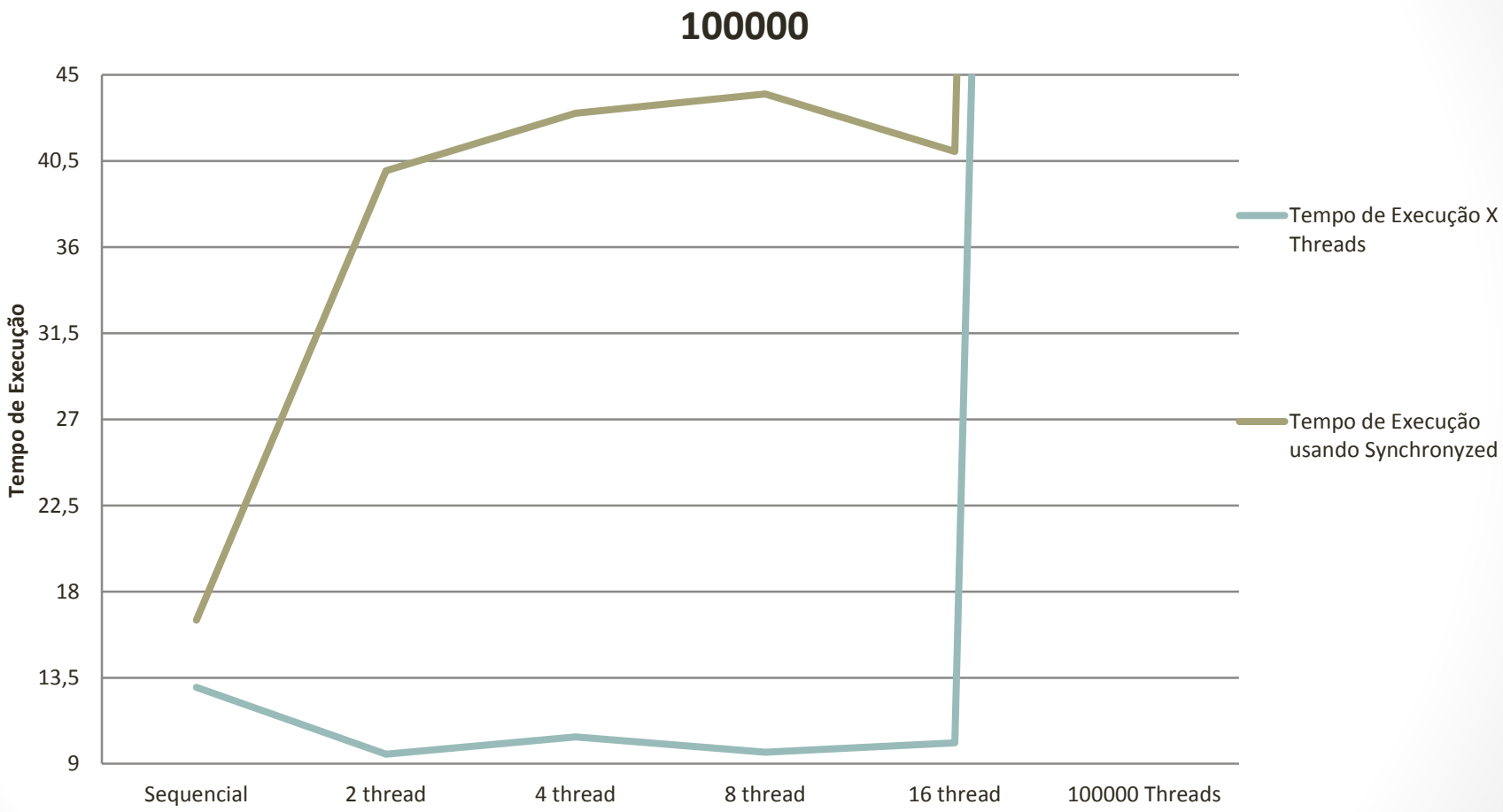


Figura 9: comparação do tempo de execução da versão sem synchronized com a versão com synchronized.

Merge Sort

- A seguir temos a análise do funcionamento da CPU durante a execução de cada ordenação. (veja as figuras 10, 11, 12, 13, 14 e 15).

- Informações da máquina: Processador Core i3, 2,1 Ghz, 2 núcleos, 4 processadores lógicos.

Merge Sort

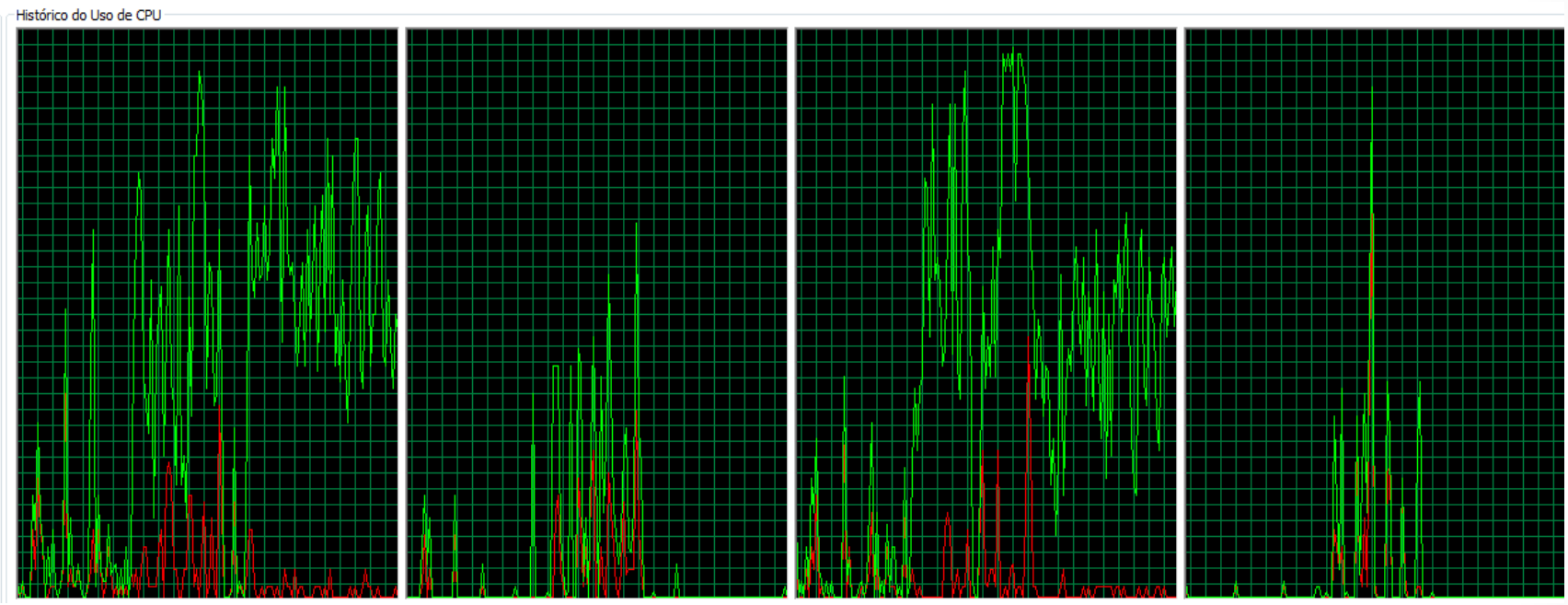


Figura 10: Utilização do CPU durante a ordenação do Merge Sort sequencial.

Merge Sort

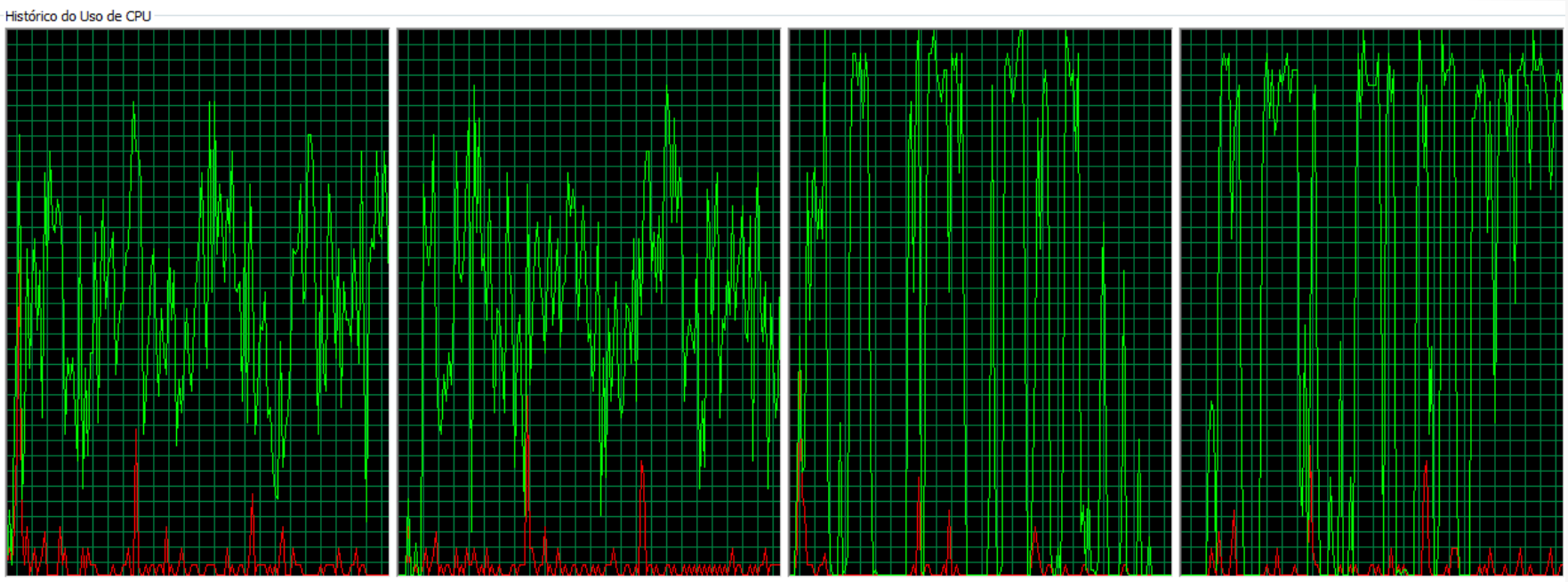


Figura 11: Taxa de utilização de CPU na execução do ordenador para 2 threads.

Merge Sort

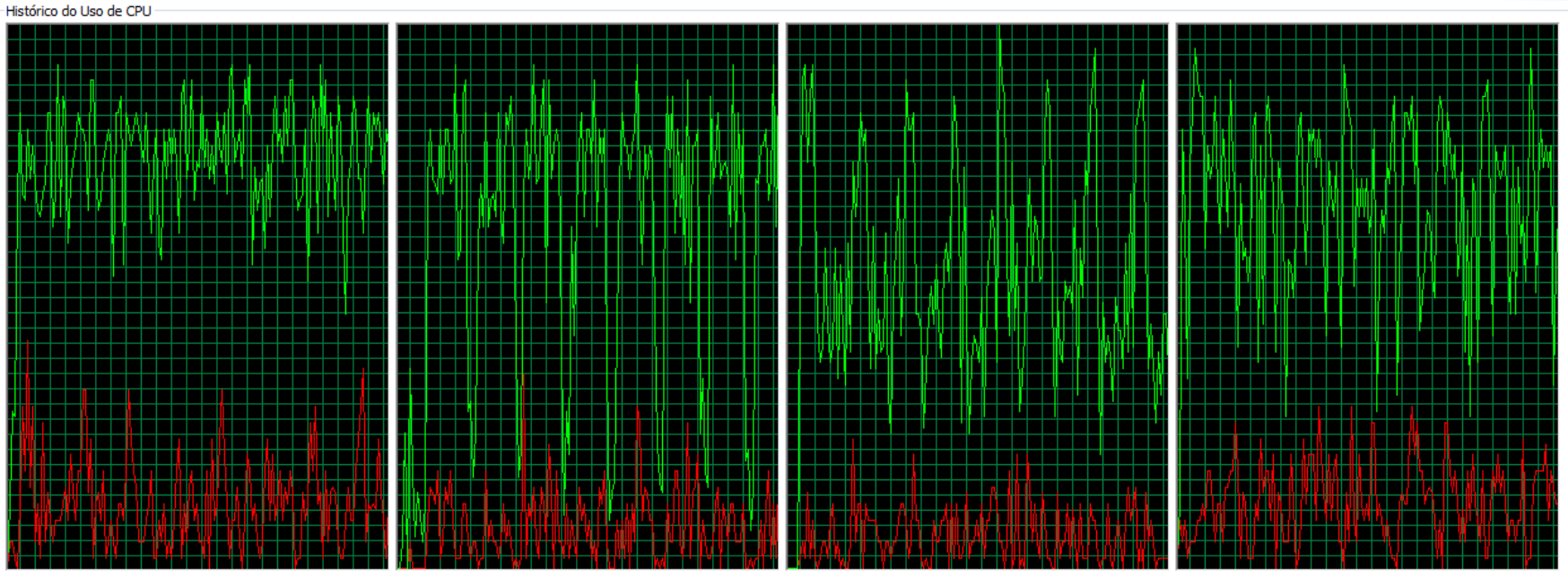


Figura 12: Taxa de utilização do CPU para o ordenador com 4 threads.

Merge Sort

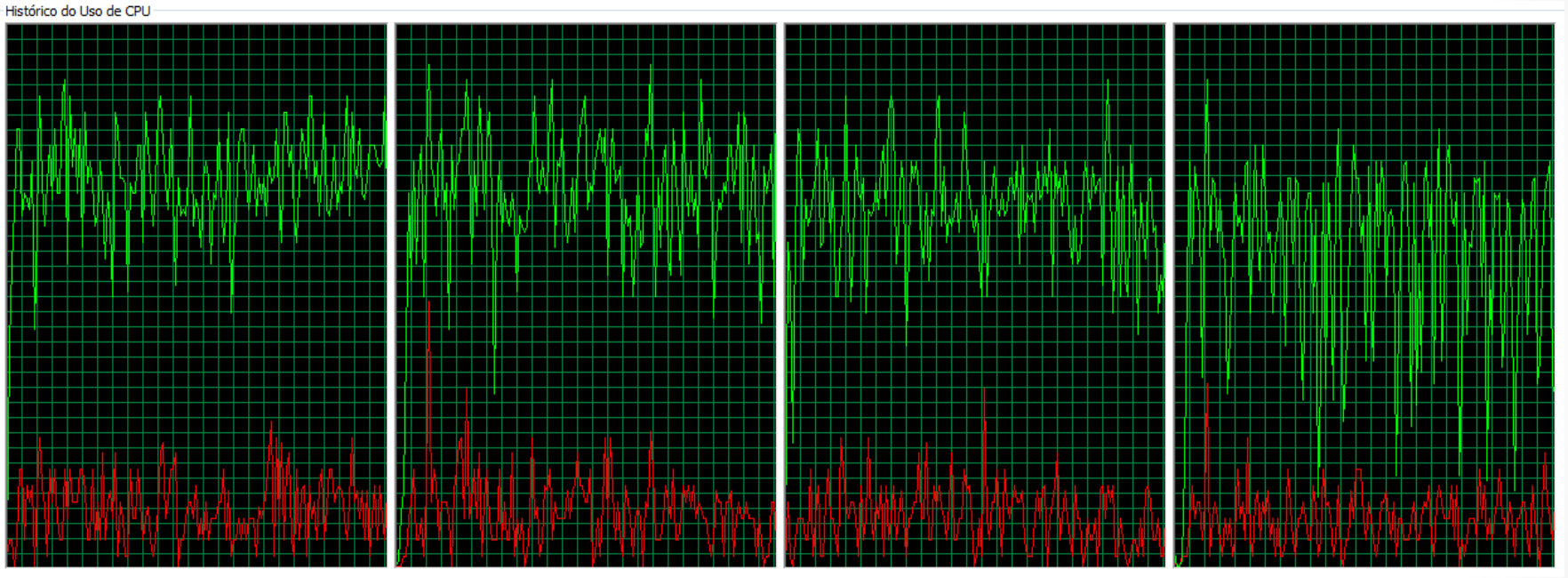


Figura 13: Utilização de CPU para o ordenador com 8 threads.

Merge Sort

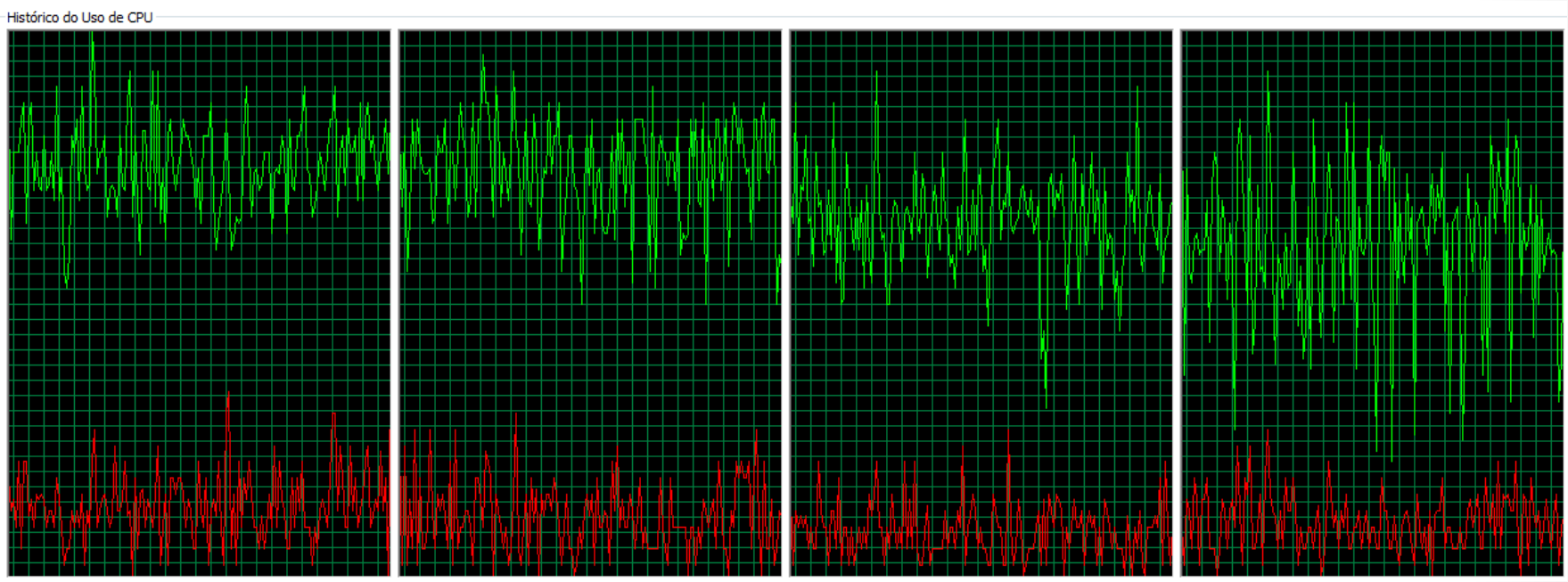


Figura 14: Utilização de CPU para o ordenador com 16 threads.

Merge Sort

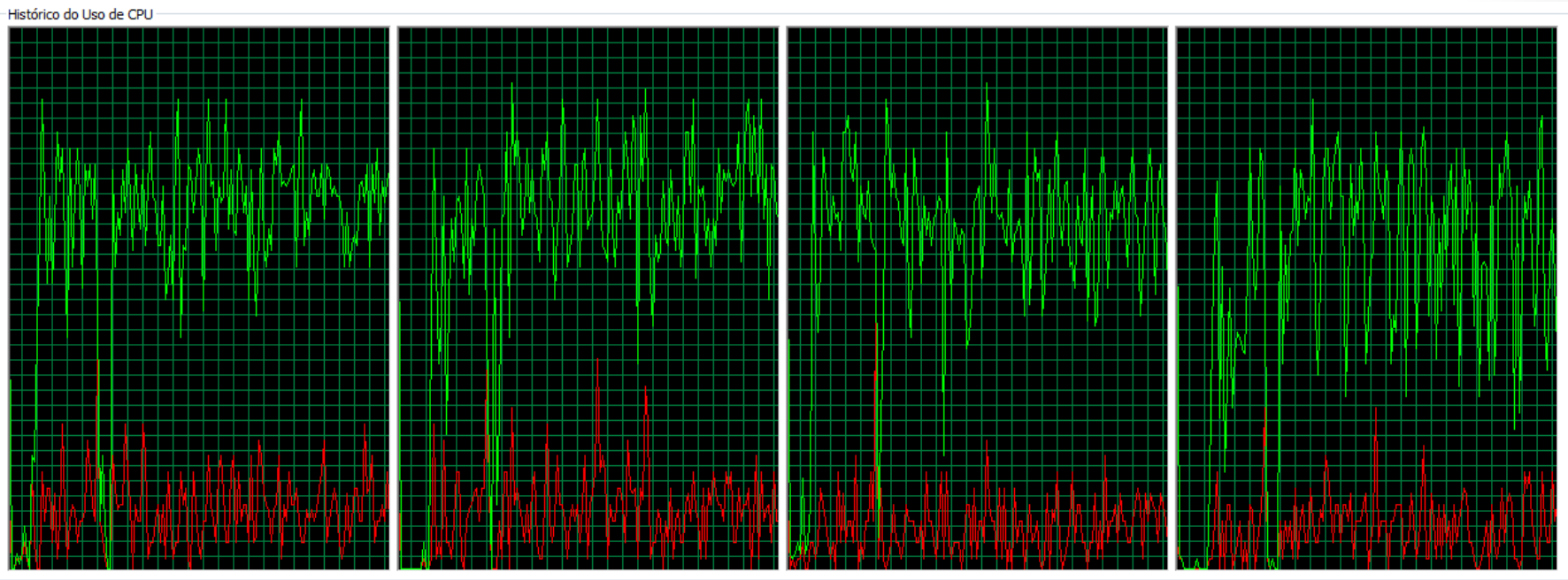


Figura 15: Utilização do CPU para o ordenador com 1000 Threads.

Merge Sort

- Verificamos que no modo econômico da bateria, o ordenador demorou mais tempo para executar. (Veja a figura 16)

Merge Sort

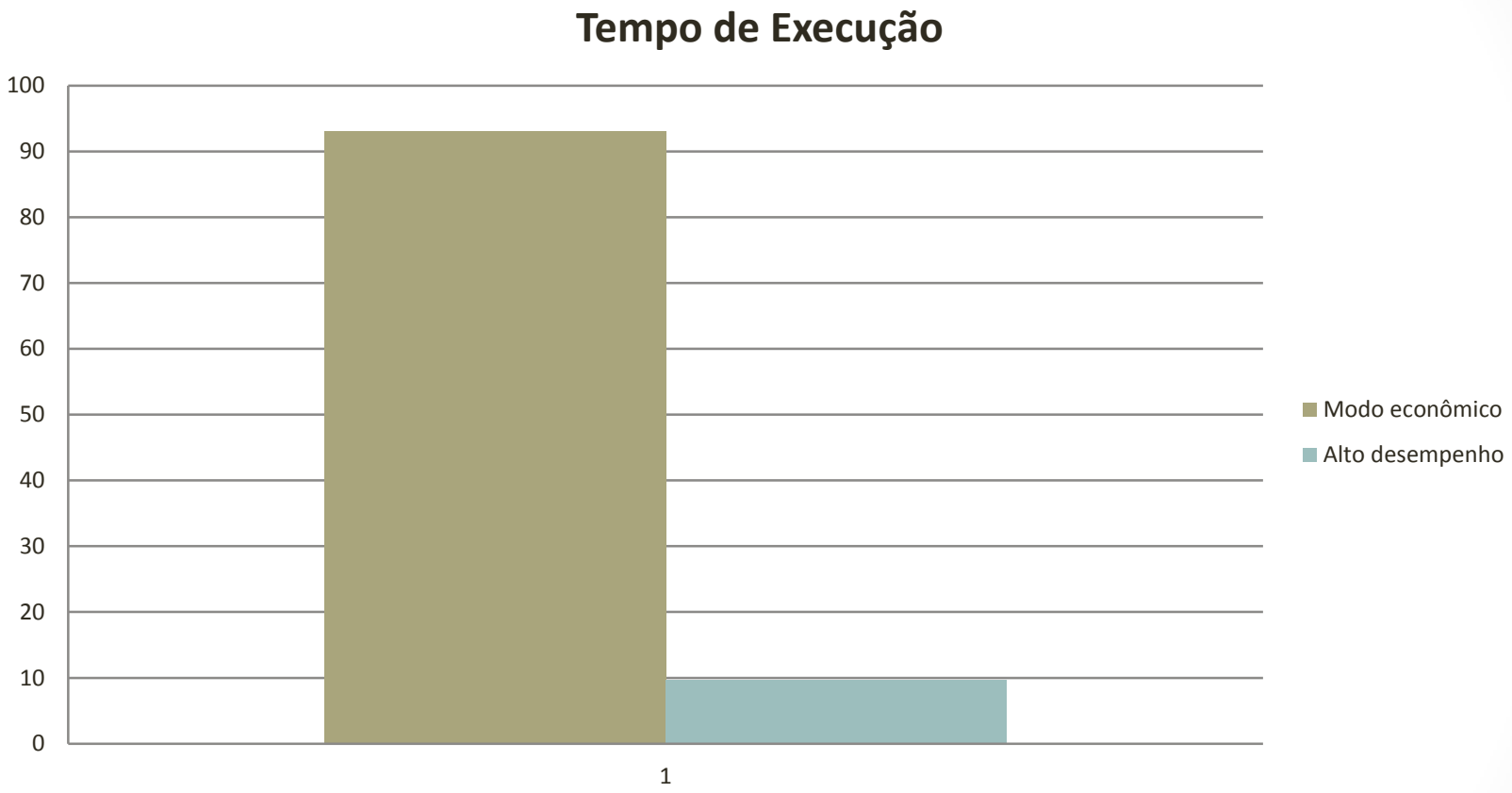


Figura 16: Comparação da execução do ordenador para um vetor de 100.000 elementos com 8 threads.

Conclusão

- Com a análise do desempenho, percebemos uma melhora no tempo de execução do algoritmo para a versão paralela em comparação com a versão sequencial.

Dúvidas?

Referencias bibliográficas

- Sistemas Operacionais com JAVA, Abraham Silberschatz, Peter Baer Galvin, Greg Gagne, 7° edição.