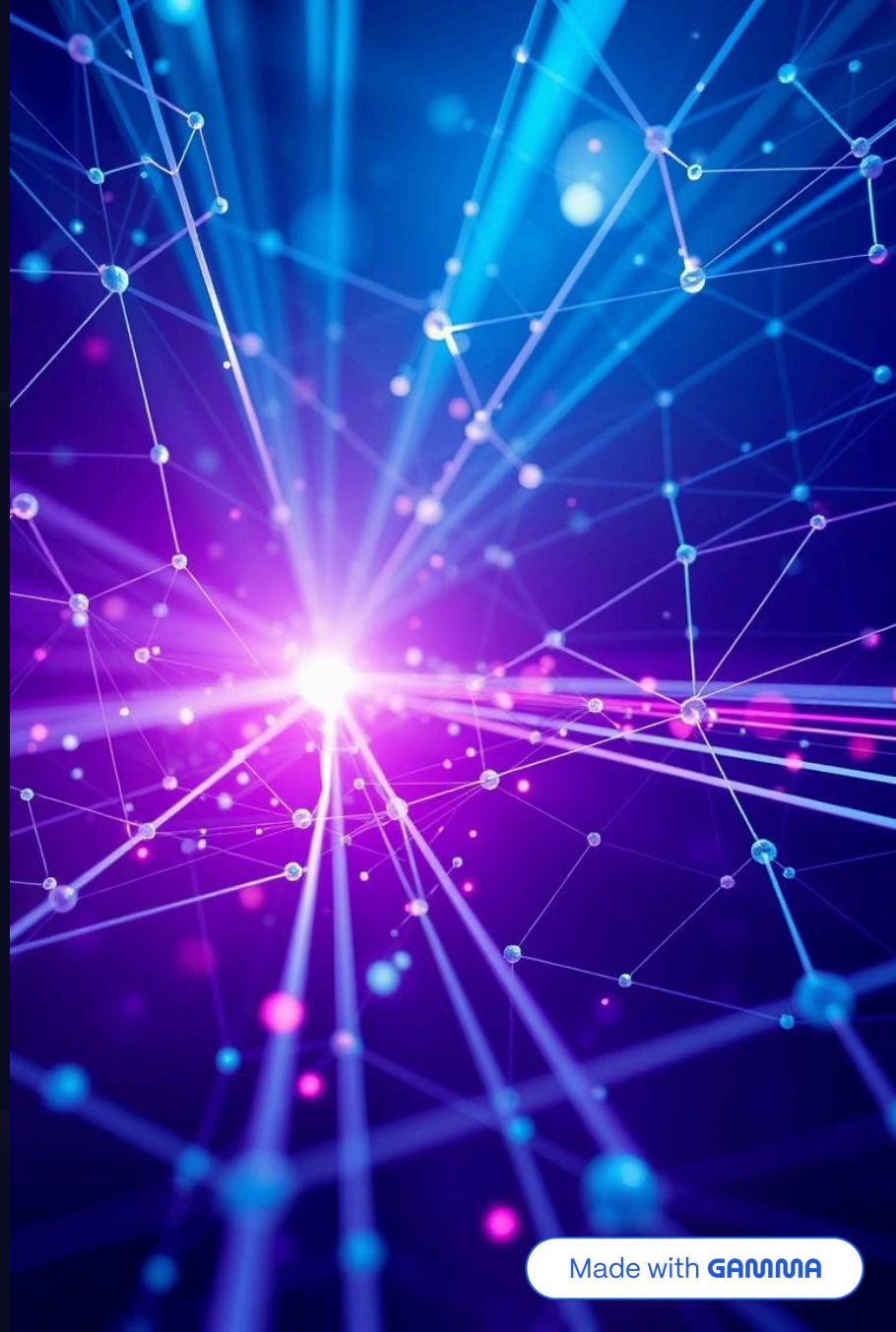
Sistemas Distribuídos: **Problema do Caixeiro Viajante**

Esta apresentação explora a resolução do Problema do Caixeiro Viajante. Abordaremos soluções sequenciais, paralelas e distribuídas. Nosso objetivo é comparar o desempenho de cada abordagem.

5 colaboradores



O Problema do Caixeiro Viajante (PCV)

Definição do Problema

Encontrar a rota mais curta. Visitar cada cidade exatamente uma vez. Retornar à cidade de origem.

Natureza Computacional

É um problema de otimização combinatória. Complexidade NP-difícil. Grande desafio computacional.

Aplicações Práticas

Logística de transporte. Roteamento de redes. Fabricação de circuitos. Várias indústrias se beneficiam.

Metodologia de Desenvolvimento

Soluções Implementadas

- Versão sequencial: Baseline de desempenho.
- Versão paralela: Utilizando **Threads (3 Threads)**.
- Versão distribuída: Com **RMI (3 Hosts)**.

Solução Sequencial

Desafios Enfrentados:

- Estouro de memória em instâncias com muitas cidades
- Lentidão e alto tempo de execução
- Incapacidade de escalar para problemas maiores

Soluções Encontradas:

- Trava de segurança com limite de 13 cidades
- Em problemas maiores utilizar solução paralela ou distribuída

Solução Paralela

Desafios Enfrentados:

- Estouro de Memória (OutOfMemoryError) em Problemas Maiores (N > 12);
- Lentidão Inesperada e Sobrecarga do Sistema;
- Divisão de Trabalho Ineficiente.

Soluções Encontradas:

- Mudança de Paradigma: Geração e Processamento "On-the-Fly";
- Otimização do Uso da CPU e Memória;
- Resultados Comprovados: Escalabilidade e Desempenho.

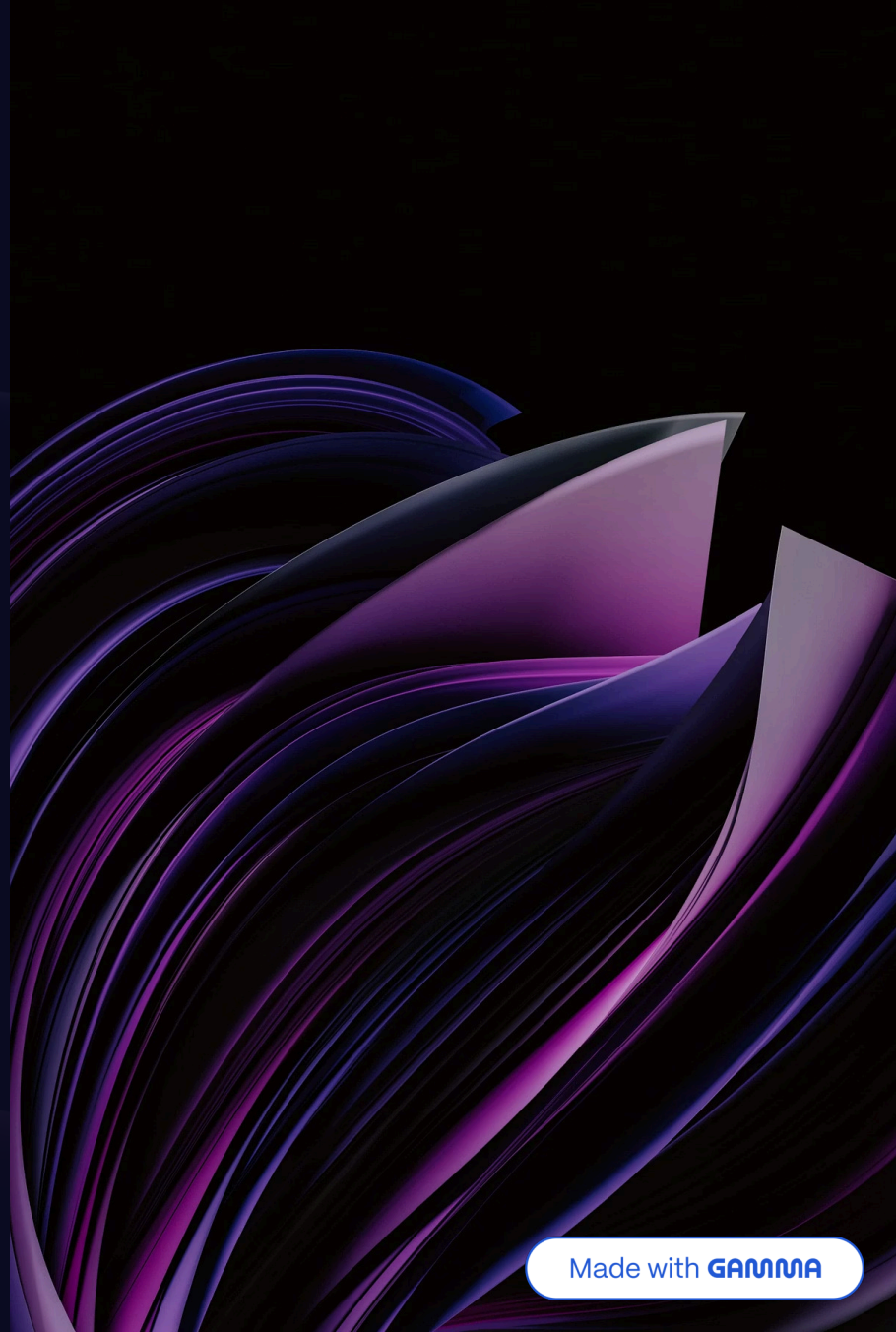
Solução Distribuída

Desafios Enfrentados:

- Gerenciar a ordem correta de inicialização dos múltiplos processos
- Adaptar a lógica de memória compartilhada (paralela) para o modelo de troca de mensagens (distribuído).
- Problemas em em testes maiores (Cidades > 13)

Soluções Encontradas:

- Definição de um protocolo de execução em 3 passos: rmiregistry, Workers, Server.
- Reaproveitamento da estratégia, onde a tarefa da thread (Callable) foi convertida no método remoto do Worker.



Testes

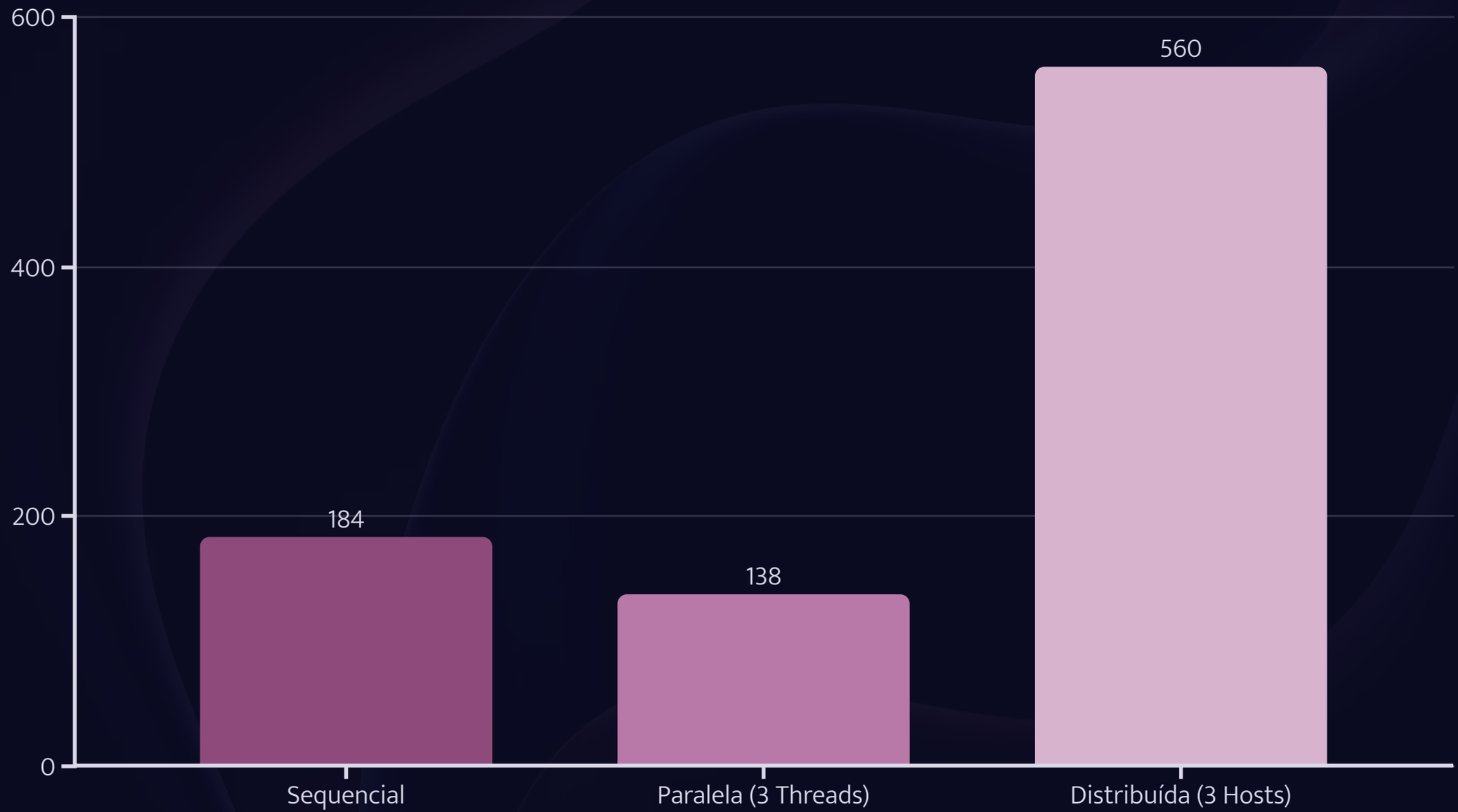
Testamos as soluções com diferentes proporções de execução:

- 10 cidades até 13 cidades

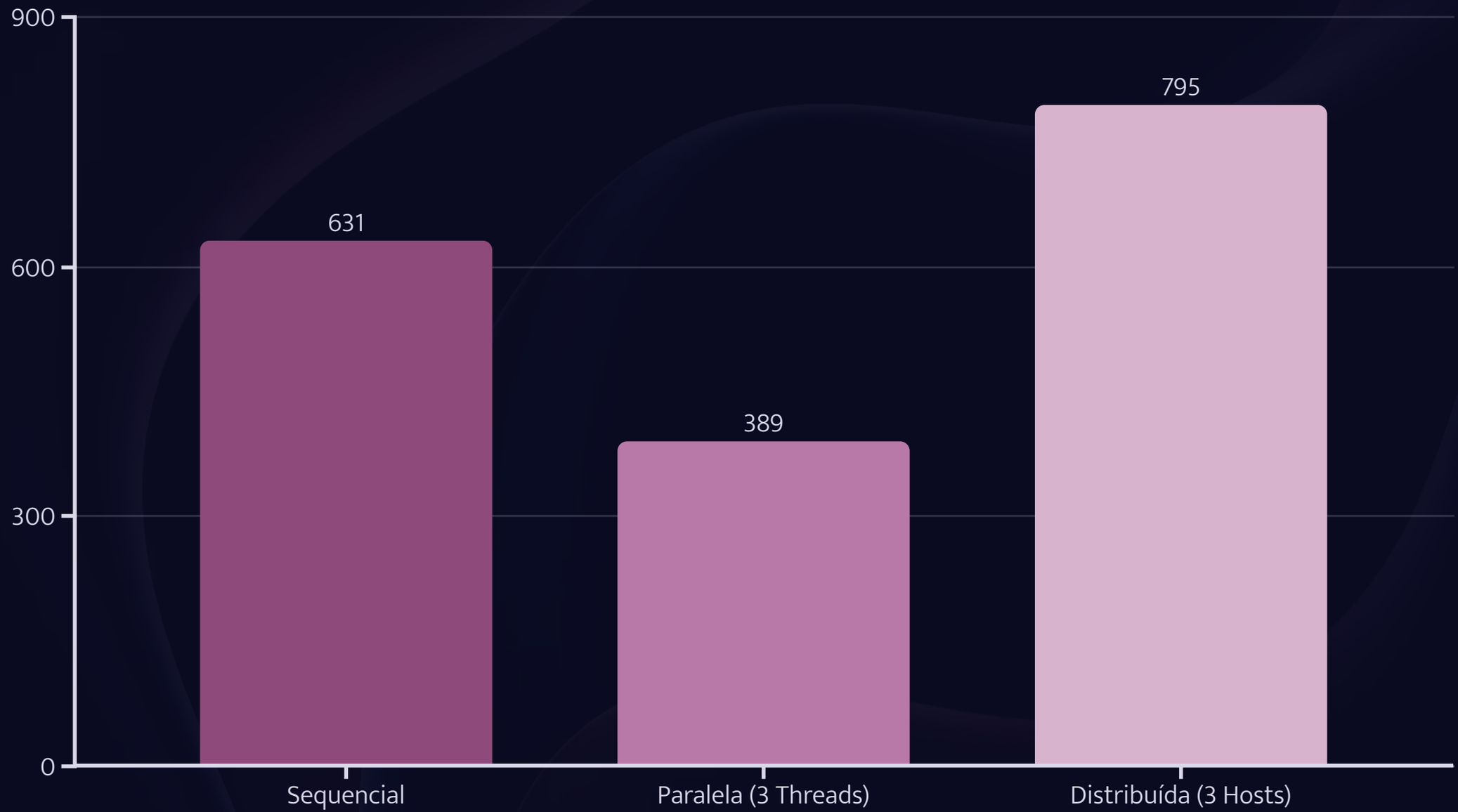
A configuração da máquina utilizada foi:

- Microsoft Windows 10 Pro - x64
- 8GB RAM (1.5GB disponível no momento)
- Intel Core i5-4440
- 3 Núcleos e 3 Processadores Lógicos (Threads)

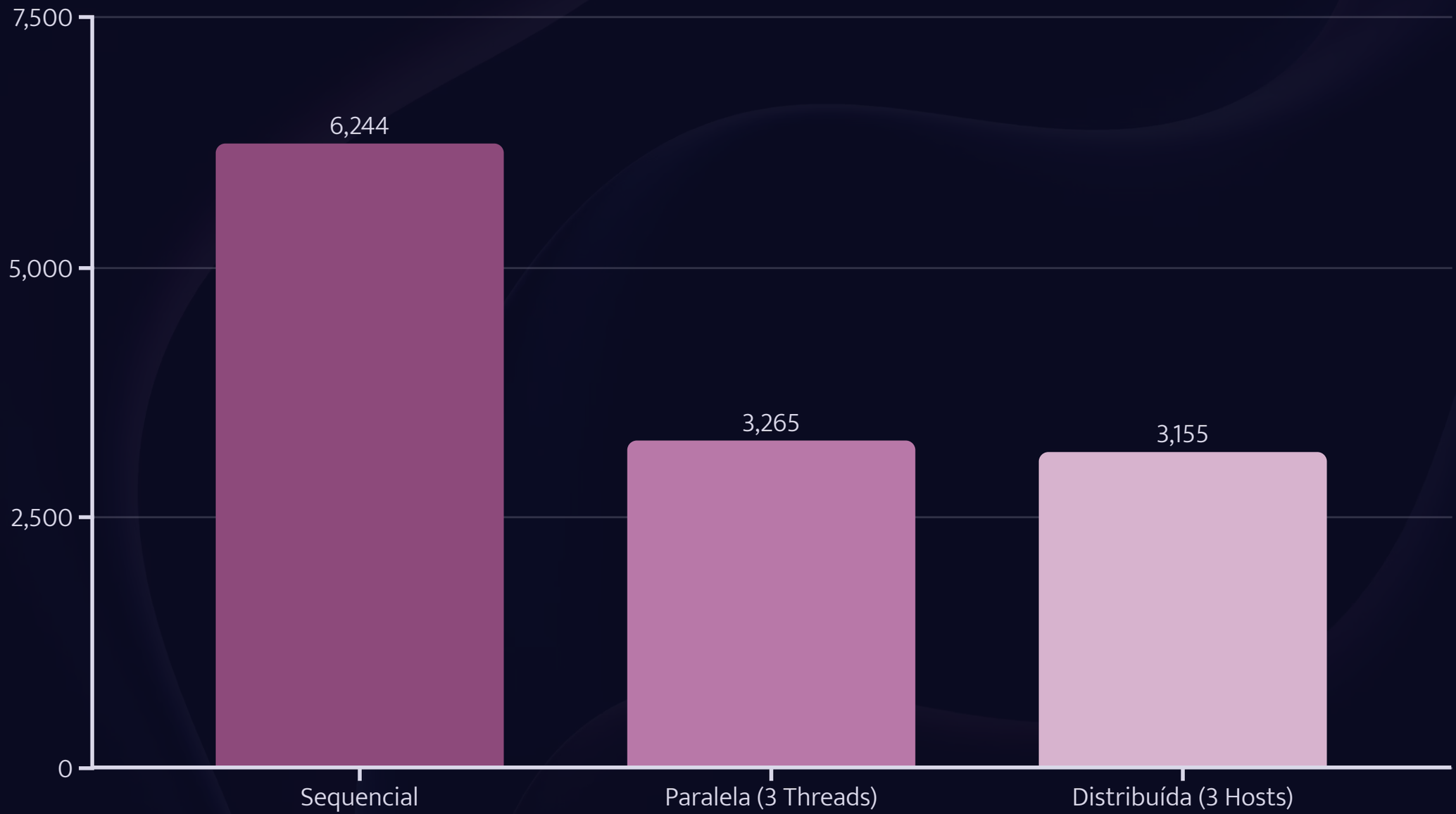
Tempos de Execução - 10 cidades



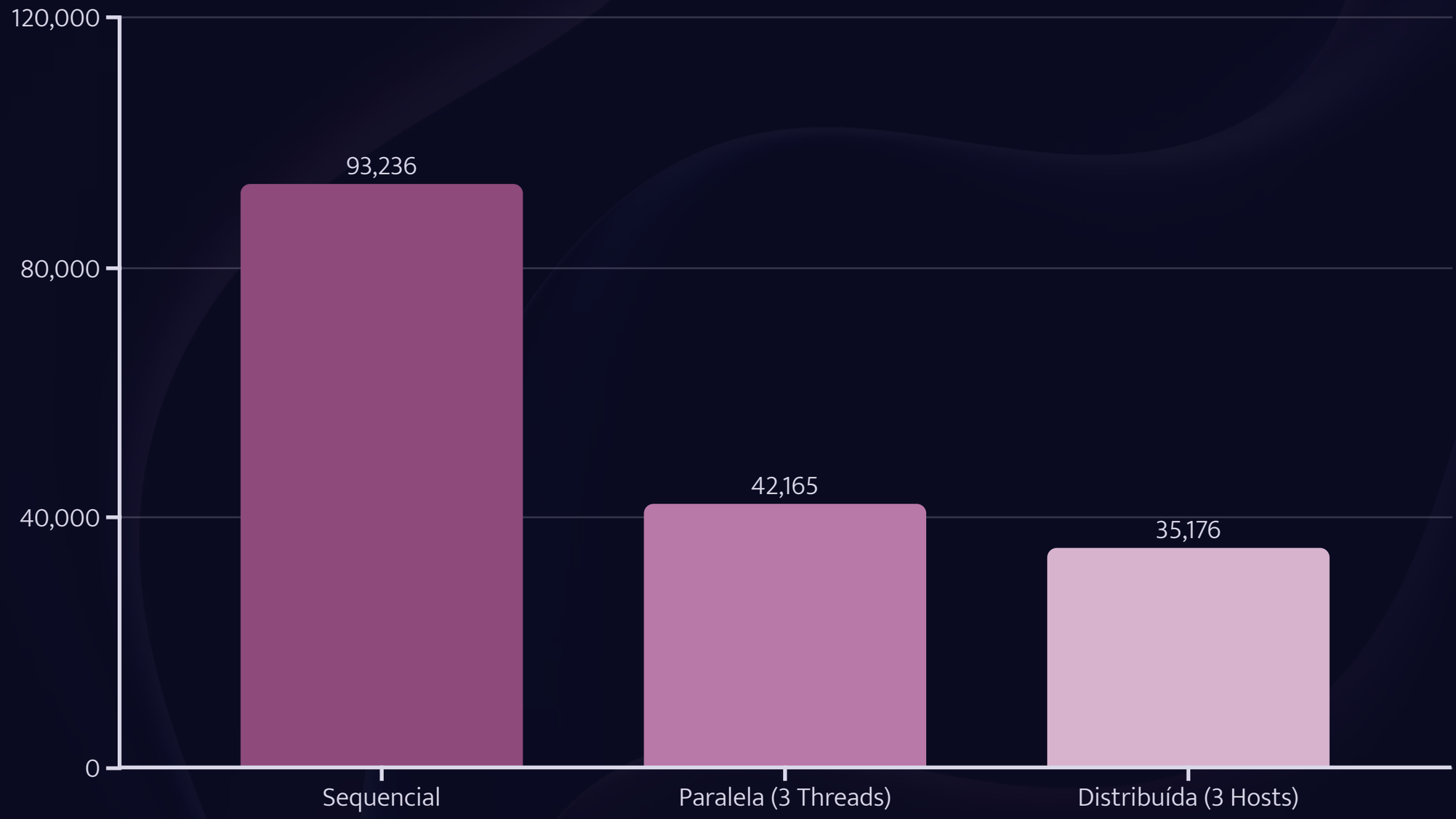
Tempos de Execução - 11 cidades



Tempos de Execução - 12 cidades

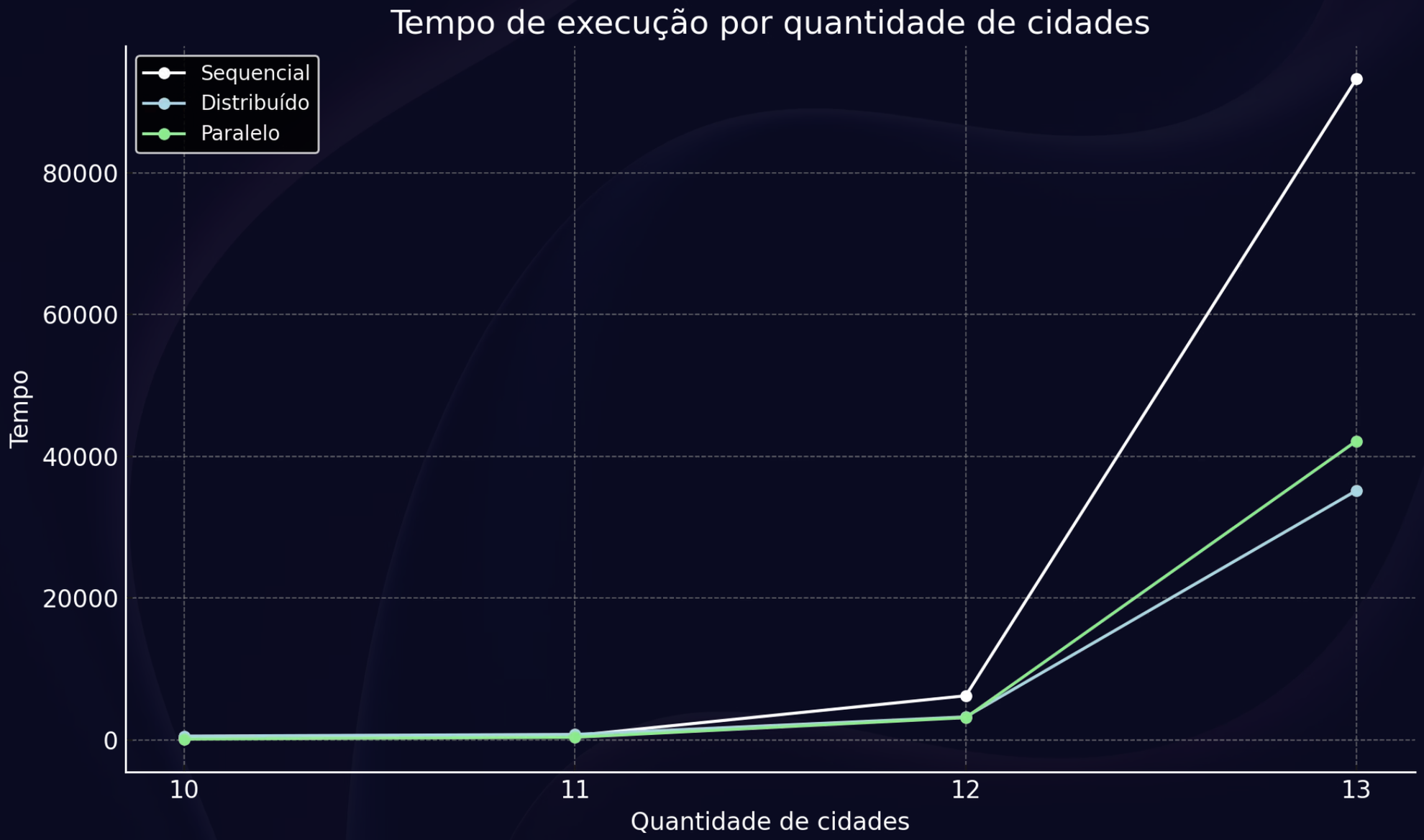


Tempos de Execução - 13 cidades



O gráfico acima mostra a redução significativa do tempo de execução. As soluções paralelas e distribuídas superam a sequencial.

Comparação tempo de execução por cidades



Equipe

Membro da Equipe	Atividade
Kauan Pedreira	Desenvolvimento da solução sequencial.
Matheus Andrade	Desenvolvimento da solução sequencial.
Luccas Maia	Implementação da solução paralela.
Carlos Hereman	Implementação da solução paralela e testes.
Jeferson Rocha	Desenvolvimento da solução distribuída (RMI).
Matheus Madureira	Desenvolvimento da solução distribuída (RMI) e testes.
Thales Granja	Preparação da apresentação, gráficos, documentação e testes.

Conclusão

Principais Conclusões

Soluções paralelas e distribuídas são superiores. Elas reduzem drasticamente o tempo de execução. O PCV é ideal para paralelização.

Sugestões Futuras

Testar em clusters maiores. Adicionar tolerância a falhas.

Link GitHub:

GitHub

GitHub - MatheusMadureiraa/caixeiro-viajante-java: Sol...

Solving the Traveling Salesman Problem (TSP) in Java. The repository contains 3 solutions: sequential, distributed...