ACH2147 - Desenvolvimento de Sistemas de Informação Distribuídos

Luan Pereira Pinheiro - 13672471

Marcos Vilela Rezende Júnior - 13729806

São Paulo

2024

Parte 1

Qual o paradigma de programação escolhido?

Para a resolução desse Exercício-Programa, escolhemos utilizar a programação orientada a objetos (POO). A escolha por esse paradigma se dá pela facilidade de modulação apresentada, permitindo que cada componente do sistema (nós da rede, métodos de busca, mensagens, etc.) seja representado como uma classe com responsabilidades bem definidas, o que ajuda na organização e manutenção do código, permitindo que alterações em uma parte do sistema sejam feitas sem afetar outras partes.

Como foi feita a divisão do programa em threads?

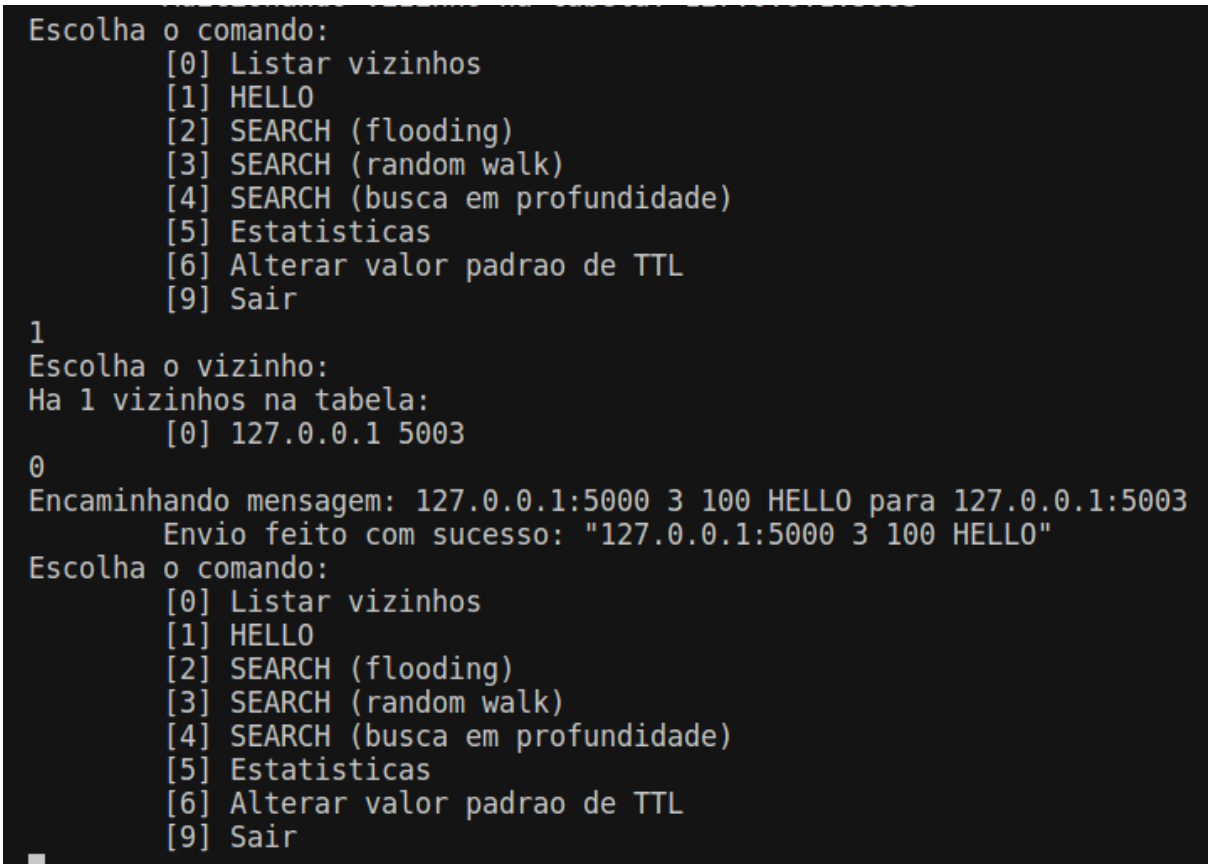
A divisão em threads foi utilizada principalmente em “ConnectionManager” buscando permitir que múltiplas conexões de clientes sejam processadas simultaneamente, aumentando a eficiência e capacidade do servidor, além de permitir interação do usuário com o servidor via terminal ao mesmo tempo. Ademais, ao utilizar essa técnica, o servidor pode continuar aceitando novas conexões enquanto processa conexões existentes.

Foi escolhido utilizar operações bloqueantes ou não bloqueantes?

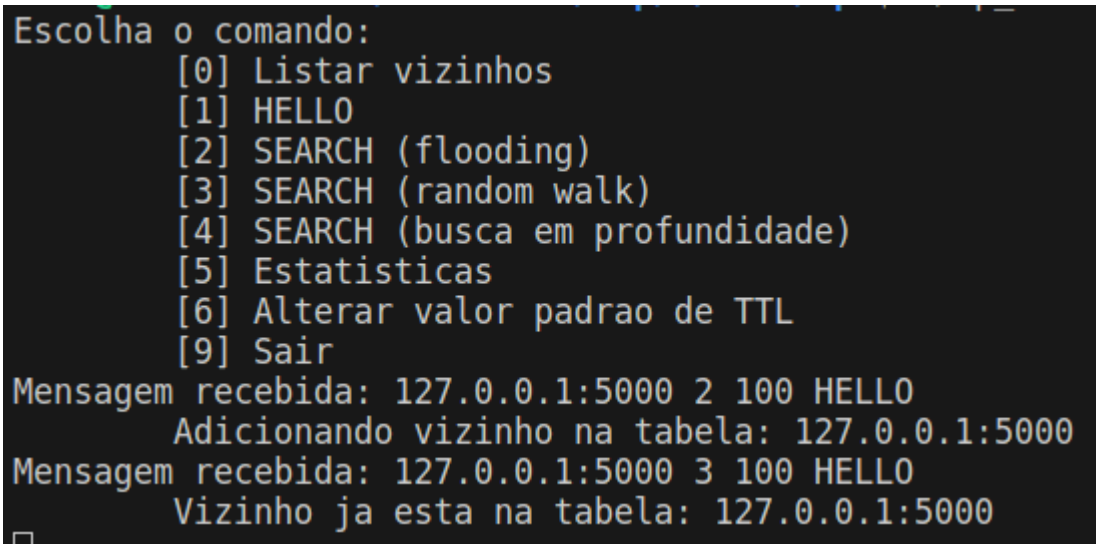
Escolhemos utilizar operações bloqueantes, devido a facilidade de serem implementadas e entendidas. Assim, não é necessário lidar com a complexidade adicional de verificar continuamente se há dados disponíveis ou se uma conexão foi estabelecida. Além disso, como estamos utilizando threads para lidar com cada conexão de cliente de forma independente, a natureza bloqueante das chamadas não impede o processamento paralelo. Cada thread pode bloquear esperando por dados sem afetar outras threads ou a capacidade do servidor de aceitar novas conexões.

Parte 2

Demonstrar o funcionamento das operações HELLO e BYE:



Comando HELLO do lado do remetente



Comando HELLO do lado do receptor (a primeira mensagem ocorreu no adição de vizinhos a partir de arquivo)

Demonstrar que a lógica do TTL está sendo aplicada:

Demonstrar que a busca por flooding é capaz de encontrar uma chave que existe na rede:

Demonstrar que as mensagens de busca por flooding não são re-enviadas para o nó remetente:

Demonstrar que as mensagens de busca por flooding repetidas são descartadas:

Demonstrar que a busca por random walk é capaz de encontrar uma chave existente tanto numa rede sem ciclos quanto numa rede com ciclos:

Demonstrar que a busca em profundidade é capaz de encontrar uma chave existente tanto numa rede sem ciclos quanto numa rede com ciclos:

Demonstrar a coleta correta de estatísticas:

Parte 3

Executar o programa em pelo menos dois computadores e em um topologia um pouco mais complexa (à escolha do grupo, com pelo menos 10 nós). Em seguida, escolher dois nós da rede para fazer as operações de busca algumas vezes (cada nó deve buscar sempre a mesma chave). Analise as estatísticas obtidas.