

Relatório Técnico – Programação Paralela e Distribuída

Laboratório III – Comunicação Indireta Pub/Sub + Eleição e Coordenação

Alunos:

Ian Vinicius - 2213648

Pedro Henrique Pereira Soares - 2313775

Disciplina

Programação Paralela e Distribuída – 2025/2

Professor

Prof. Breno Krohling

Introdução

Este relatório apresenta a implementação de um sistema distribuído utilizando comunicação indireta baseada no modelo Publish/Subscribe (Pub/Sub) por meio do protocolo MQTT. O sistema integra três componentes fundamentais: a fase de eleição distribuída para escolha de um coordenador, o papel do Controlador responsável por coordenar desafios criptográficos, e a atuação dos Mineradores, que buscam soluções para tais desafios usando um mecanismo de Prova de Trabalho (Proof-of-Work).

Cada processo executa simultaneamente funcionalidades de nó participante, controlador (caso seja eleito líder) e minerador. A comunicação entre processos ocorre através de tópicos específicos no *broker* MQTT, com mensagens no formato JSON, garantindo aderência total aos requisitos da atividade.

Metodologia de Implementação

A implementação foi realizada em Python utilizando a biblioteca paho-mqtt. A solução foi estruturada em fases bem definidas: inicialização, eleição e execução do papel designado (controlador ou minerador). Cada fase possui estados e regras claras, o que garante organização, previsibilidade e sincronização entre processos distribuídos.

A seguir são descritos os principais componentes.

Inicialização (Init)

Cada processo inicia gerando um identificador de cliente aleatório de 16 bits. Em seguida, publica sua mensagem de inicialização no tópico `sd/init`. Cada nó mantém uma lista interna com os identificadores de todos os participantes conhecidos e só avança para a próxima fase quando todos os `ClientIDs` esperados forem recebidos. Esse mecanismo garante que todos os participantes iniciem o sistema em perfeito alinhamento.

Eleição Distribuída

Após a inicialização, cada processo gera um número aleatório para votação, denominado `VoteID`, também de 16 bits. Para aumentar a confiabilidade da eleição, cada nó registra localmente o próprio voto antes de publicá-lo no broker. Isso impede que o sistema dependa exclusivamente do retorno (“eco”) do MQTT, garantindo que cada participante contabilize ao menos seu próprio `VoteID` mesmo em cenários com perdas de mensagens.

O processo então publica sua mensagem de votação no tópico `sd/voting` e aguarda os votos dos demais participantes. A eleição segue o critério de maior `VoteID` e, em caso de empate, maior `ClientID`. O vencedor assume o papel de Coordenador. Os demais processos tornam-se Mineradores.

Devido à utilização de um broker MQTT público, onde podem ocorrer perdas ou atrasos na entrega de mensagens de votação, a implementação emprega um mecanismo de timeout de 8 segundos. Assim, cada nó coleta os votos que chegarem dentro desse período e finaliza a eleição de forma robusta, mesmo que nem todos os votos sejam recebidos por todos os participantes. O processo de escolha do coordenador permanece o mesmo (maior `VoteID`, com desempate por `ClientID`), preservando o critério de correção do algoritmo.

Papel do Controlador

O controlador é responsável pela criação e gerenciamento de desafios criptográficos. Ao assumir essa função, ele cria um registro contendo TransactionID, Challenge, Solution e Winner. Em seguida, publica o desafio para os mineradores por meio do tópico sd/challenge e aguarda propostas de solução no tópico sd/solution. Cada solução recebida é verificada, e o controlador publica em sd/result se a solução foi aceita ou rejeitada. O controlador mantém localmente uma tabela com o estado das transações.

Papel do Minerador

Os mineradores permanecem escutando o tópico sd/challenge até receberem um desafio. Ao recebê-lo, armazenam os dados localmente e iniciam o processo de Prova de Trabalho. Cada minerador tenta gerar uma solução que satisfaça o critério de validação e envia sua proposta ao controlador por meio de sd/solution. Os mineradores continuam buscando soluções até que um resultado seja publicado em sd/result. A partir do momento em que um vencedor é divulgado, todos interrompem a mineração automaticamente.

Prova de Trabalho (Proof-of-Work)

A solução implementa um mecanismo de Prova de Trabalho baseado no algoritmo SHA-1. Para que uma solução seja considerada válida, o hash SHA-1 da string formada por "TransactionID:Solution" deve começar com uma quantidade de zeros proporcional à dificuldade do desafio. O número de zeros exigidos depende do valor de Challenge e cresce de forma progressiva, tornando desafios maiores mais difíceis e demorados.

Esse mecanismo segue a lógica vista em atividades anteriores (Lab II), mantendo coerência conceitual com o tema de mineração distribuída apresentado na disciplina.

Arquitetura de Comunicação

A solução utiliza exclusivamente MQTT com o broker EMQX e os seguintes tópicos:

sd/init

Envia identificações iniciais para sincronização.

sd/voting

Troca de votos para eleição distribuída.

sd/challenge

Controlador envia desafios aos mineradores.

sd/solution

Mineradores enviam suas soluções.

sd/result

Controlador comunica aceitação ou rejeição de soluções.

Todas as mensagens são enviadas em formato **JSON**, garantindo interoperabilidade e facilidade de debug.

Testes Realizados

Os testes foram conduzidos executando múltiplas instâncias do programa em terminais diferentes, simulando nós distribuídos. O sistema foi testado com três participantes. Durante os testes, verificou-se que, por limitações do broker público utilizado, nem todas as mensagens de voto eram propagadas entre todos os nós. Para contornar essa limitação, foi incorporado um mecanismo de timeout e a contabilização local do voto próprio, garantindo que a eleição sempre se complete e que o líder seja corretamente definido.

Resultados

Os resultados observados confirmam o correto funcionamento do sistema distribuído. O mecanismo de eleição funcionou conforme especificado, a tabela de transações foi mantida pelo controlador de forma adequada e a comunicação via MQTT se manteve estável. O mecanismo de Prova de Trabalho variou corretamente em dificuldade, e o comportamento dos mineradores foi consistente e sincronizado. O encerramento da mineração após o anúncio do vencedor também ocorreu de forma imediata e correta.

Conclusão

O trabalho implementado atende plenamente às especificações do laboratório. Foi possível criar um sistema distribuído funcional com comunicação indireta, eleição de líder, coordenação centralizada e mineração distribuída baseada em Prova de Trabalho. Os resultados comprovam que o sistema é coerente, sincronizado e resiliente, demonstrando na prática conceitos fundamentais da área de Programação Paralela e Distribuída.