Problema 3 - Malha aérea

Daniel Fernandes Campos, João Pedro Rios Carvalho

dfc152@gmail.com, jprcarvalho1@gmail.com

Resumo. *Este relatório descreve o processo de desenvolvimento de um sistema de parceria entre as companhias aéreas brasileiras, no qual tornou-se possível a visualização de voos com assentos disponíveis de todas as empresas parceiras no sistema de qualquer um deles, ampliando a malha aérea e facilitando a vida dos clientes na procura por passagens.*

Palavras chaves: *sistema, companhias, voos*

1. Introdução

O sistema surge como um acordo de parceria entre as companhias aéreas brasileiras, no qual, o *software* de cada uma delas mostraria, não somente os seus voos disponíveis, como também, os voos das outras companhias parceiras que também levam do local de origem ao local de destino pesquisados, ou, pelo menos, um trecho desse trajeto.

Dessa forma, é possível para o usuário escolher o voo que ele quiser, analisando fatores como trechos do trajeto completo, duração dos trechos, preços e companhias responsáveis por cada trecho, e assim, ter o melhor custo-benefício, ou maior rapidez, dependendo de sua necessidade. Também é realizável o pagamento de todos os voos na mesma plataforma, sem precisar ir para a plataforma de outra companhia para comprar um voo de determinado trecho, dependendo somente da quantidade de assentos disponíveis para finalizar a compra.

2. Fundamentação Teórica

Esta seção tem como função apresentar os conceitos necessários para o desenvolvimento do sistema, como também introduzi-los aos leitores, de forma concisa.

2.1. Sistemas Distribuídos (Arquitetura P2P)

“Um sistema distribuído é uma coleção de computadores independentes que é visto pelo usuário como um sistema único” (NASCIMENTO). Complementando esta definição de Nascimento, pode-se dizer que cada computador é autônomo e independente, e que se comunicam um com o outro de alguma forma para colaborarem no projeto, executando processamentos distintos e compartilhando informações, dessa forma, fica invisível para o usuário que o sistema é formado por vários computadores distintos.

Portanto, os sistemas distribuídos geralmente contam com uma arquitetura de rede Par-a-Par(Peer-to-Peer, ou, P2P), que garante a eles a criação de uma rede descentralizada e robusta, na qual funciona independentemente da quantidade de servidores que estão presentes no sistema e que continuam o seu funcionamento até mesmo com a queda de conexão de algum deles.

2.2. Token Ring

Token Ring é um algoritmo de sincronização muito utilizado em sistemas distribuídos porque tem características de funcionamento descentralizadas, principalmente quando eles contam com regiões críticas, ou seja, que necessitam de uma atenção especial para o acesso dos processos e modificação de variáveis, para que não causem erros ou operações indesejadas.

O algoritmo opera no sistema em forma de anel, no qual associamos um processo a cada posição no anel, onde cada um conhece seu sucessor para a transmissão do *token*. Apenas o processo que tiver em posse do *token* poderá acessar a região crítica e realizar a operação desejada, após isso, ele passa o token para o seu sucessor no anel. Dessa forma, cada processo tem a sua vez de acessar a região crítica e de forma individual, evitando que dois processos modifiquem a mesma variável ao mesmo tempo. Também é possível definir limites para que um processo não fique muito tempo com o token, podendo deixar o sistema lento.

2.3. Algoritmo de Busca em largura

O algoritmo de busca em largura é muito utilizado em sistemas com grafos para pesquisar caminhos possíveis de um vértice a outros no grafo. Ele parte de um ponto e vai visitando os vértices vizinhos, quando termina, ele passa para outro ponto e visita seus vizinhos e assim por diante, de maneira a formar uma árvore com camadas, a busca só termina quando todos os vértices possíveis são visitados. É responsabilidade do algoritmo fazer o controle de visitas e quais ainda restam.

3. Desenvolvimento

Nesta seção discorremos sobre os sistemas desenvolvidos, explicando sobre algumas das decisões tomadas.

3.1. Sistema Geral

Para solução deste problema foi desenvolvido um sistema descentralizado de empresas, as quais podem se conectar entre si através de uma API disponibilizada, havendo rotas para ocupar e desocupar acentos, dentre outras usadas para ajustes do *‘ring’* entre elas.

Assim, cada empresa possui um conjunto de trajetos e empresas conhecidas, que inicialmente são configurados através de um arquivo de configuração. Então cada empresa é responsável por controlar as reservas de seus trechos (voos), e quando uma companhia tenta alocar um trajeto (comprar vários trechos), esta faz as consultas recursivas para as companhias para verificar se todos os trechos estão disponíveis, não importando de qual companhia esse trecho seja (desde que esta companhia seja conhecida).

As reservas e buscas podem ser feitas nas interfaces web disponibilizadas por cada empresa, nas quais é possível ver quais os voos oferecidos pela empresa e fazer busca de trajetos, o qual mostrará todos os caminhos possíveis entre 2 cidades considerando os voos de todas as companhias.

Neste sistema temos 3 elementos principais, além da interface web e API disponibilizada:

1. Controlador de Manager: responsável por se comunicar com companhias conhecidas e estabelecer quem é o manager atual (a empresa que pode fazer operações de reserva e desreservar assentos naquele momento);
2. Gerenciador de trechos: Responsável por reservar passagens quando requisitado, controlando os assentos dos voos;
3. Gerenciador de trajetos: Responsável por gerar os trajetos que serão mostrados na interface, consultando os voos das outras companhias para montar todos os trajetos possíveis entre as cidades informadas.

Quando um servidor é iniciado ele pode usar um arquivo de configuração, conforme template disponibilizado ou ele pode fazer uma configuração do zero.

3.2. Nosso Ring.

Para garantir que as ações de reservar assentos são atômicas mesmo entre servidores diferentes, fizemos nossa implementação do token Ring, porém com algumas modificações. Dentre elas, temos a não utilização de um token, pois foi implementado uma inicialização sincronizada dentre todas as companhias nos *Rings*.

Nesta implementação cada companhia tem uma lista de todas as companhias conhecidas e ela espera estas companhias para inicializar o ciclo do *Ring* juntas. Assim, é feita uma passagem em sequência nesta lista de companhias conectadas, para tal funcionar, garantimos que as companhias que se conhecem tenham a lista na mesma ordem, recriando a lista e propagando pela rede toda vez que uma nova companhia é inicializada.

Como o tempo passa diferente em diferentes máquinas e com atraso de redes, o tempo no qual uma companhia fica com o manager deve ser relativamente grande para garantir que todas as companhias façam operações (sendo usado 5s para esta implementação com 3 companhias)

Assim, temos que cada companhia tem o seu *Ring* (sequência que indica a ordem de companhias que serão managers) e estas companhias esperam iniciar um ciclo (passagem do manager entre todas as companhias no *Ring*) de forma sincronizada fazendo perguntas constantes para todas as companhias conhecidas se elas estão esperando para inicializar um novo ciclo (uma vez que todas nessa etapa as companhias começam um ciclo).

Note que este *Rings* são direcionais, ou seja, pode haver um companhia A que foi inicializada primeiro (não havendo nenhuma companhia que ela conheça inicializada antes) e esta começa a companhia B e posteriormente a companhia B é inicializada, porém, caso esta não conheça a A os *Rings* destas companhias seriam diferentes. E quando uma terceira companhia C seja inicializada, caso esta conheça a A, no seu processo de inicialização será verificado de forma recursiva se as companhias que a A conhece estão inicializadas e qual seu *Ring*, e assim seria feito um *merge* dos *Rings* das empresas A e B, além dos conhecidos pela C que não sejam conhecidos pelas demais, assim, sincronizando o *Rings* destas companhias.

Como citado anteriormente existe um processo de inicialização do *Ring* da companhia que consiste em:

1. Propagar as companhias conhecidas para as companhias conhecidas (neste processo companhias já existentes repassam as companhias conhecidas as

que elas conhecem, assim, gerando diversas mensagens na rede informando umas às outras das companhias existentes na rede);
2. Pergunta as conhecidas as companhias conhecidas por estes e continuar a busca caso sejam retornadas companhias desconhecidas até que todas as companhias sejam conhecidas;
3. Criar um *Ring* que contenha todas as companhias encontradas na busca e propagar este *Ring* para todas as companhias encontradas;
4. Caso esta companhia seja a primeira a ser inicializada, dentre as conhecidas, uma thread que lida com o *Ring* é inicializada. Caso contrário é esperado a notificação de novo ciclo para esta *thread* ser inicializada.

Esse algoritmo foi escolhido para trazer uma variedade na resolução, uma vez que outros grupos estavam trabalhando com a ideia do algoritmo do bully para determinar quem estaria resolvendo resolvendo seus pedidos de reserva. Assim escolhemos este algoritmo para trazer uma maior variedade das soluções apresentadas

3.3. Busca de caminhos

Na busca de todos os caminhos, primeiramente coletamos todos os voos de todas as companhias conhecidas e então criamos um grafo direcional, usando listas vizinhos, isso é feito toda vez, pois assim caso novos voos podem ser adicionados nas companhias sem necessidade de passar para as conhecidas e entendemos que o tempo no qual isso é feito é mínimo comparado ao tempo da busca.

Então, havendo este grafo no qual os vértices representam cidades e as arestas representam os trechos, é feito uma busca em largura partindo do vértice inicial até que não hajam mais caminhos para serem verificados e por fim retornamos as lista de todos os trajetos que levam da cidade inicial até o destino.

Então após termos a lista de todos os trajetos, é feito uma busca para cada trajeto de quais são os voos que levam de uma cidade a outra e, assim, obtemos todos os voos que podem ser pegos para aquele trajeto.

4. Conclusão

Assim, concluímos este trabalho com todos os objetivos propostos alcançados. Sendo possível, a partir da interface de uma companhia, ver todos os voos que conectam 2 cidades escolhidas, fazer a reserva do trajeto desejado (caso um dos voos de um trajeto seja de outra companhia, essa repassa o pedido de reservar o assento para ela, assim, fazendo reserva não só dos seus voos mas conseguindo fazer reserva de trajetos que tenha voos de outras companhias). Estas reservas são feitas de forma atômica no sistema, ou seja, apenas uma operação de reserva é feita pelas companhias conhecidas em um dado instante.

Algumas melhorias seriam a alteração do algoritmo de sincronização, uma possível sugestão seria a implementação de um algoritmo bully para definir um coordenador e este resolver todas as operações que seriam repassadas para ele e armazenadas em uma fila (fazendo uma nova eleição caso o coordenador caia), dessa forma diminuiria o tempo de resolução dos pedidos, consequentemente, o tempo esperado pelo cliente após fazer a compra de uma passagem.

5. Referências

NASCIMENTO, Aline. “SISTEMAS DISTRIBUÍDOS”. Disponível em: http://www.ic.uff.br/~simone/sd/contaulas/aula1.pdf. Acesso em 1 dez 2021.

VALE, Sávio. (2020), “O que é uma rede peer-to-peer (p2p)? Funcionamento e aplicações dessa tecnologia que vão além do compartilhamento de arquivos”. Disponível em: https://www.voitto.com.br/blog/artigo/o-que-e-rede-p2p. Acesso em 1 dez 2021.

SANTOS, Ricardo R. dos. (2003), “Sistemas Distribuídos”. Disponível em: http://www.facom.ufms.br/~ricardo/Courses/DisSys/Material/SD-Aula-09.PDF. Acesso em 4 dez 2021.