Multiplicação Concorrente de Matrizes
Sistemas Operacionais II

RODRIGO FREITAS LEITE
FELIPE SOARES F. PAULA
UFRGS - Instituto de Informática
15 de outubro de 2013

Plataforma utilizada

cpu	AMD Athlon(tm) II X2 250 3GHz
memória	2Gb
versão Gcc	4.7.2
versão Linux	3.5.0-17-generic

Visão geral

Inicialmente, no servidor, inicializamos todas as estruturas necessárias para a concorrência (*pthreads*) e para conexão (*sockets*), também fazemos o *bind* do servidor. Depois das inicializações, o servidor fica em um laço eterno "escutando"por novas conexões. Caso ele ache alguma, é disparada uma *thread* que implementa, de fato, a lógica do chat.

O cliente fica preso a essa nova *thread* que fica tratando as strings que são enviadas e disparando os serviços adequados. No primeiro acesso ao servidor, o cliente manda (sem saber) uma mensagem "- *-firstacess*"que indica ao servidor que é um novo usuário e que ele deve ser colocado na lista que contém todos usuários (chamada de *lobby*).

Quando o cliente manda a mensagem de criar sala, uma nova sala é adicionada à lista de salas e sua respectiva lista de usuários é inicializada. Também é feita uma validação para não ser criadas duas salas com o mesmo nome. Quando um usuário deseja sair, tem que ser feitas as consistências de tirá-lo da lista de usuários do servidor e das salas de chat.

O programa do cliente é mais simples. São duas *threads*, uma que escuta por mensagens do servidor e as imprime, e outra, que trata o que o usuário digita. Quando uma nova mensagem é escrita e enviada, ela vai para o servidor e é redistribuída entre todos os usuários na lista da sala correspondente, a menos que seja um comando o qual o servidor deverá lidar de maneira diferente.

Estruturas em memória

O servidor sempre terá as seguintes estruturas carregadas:

- **lobby**: Lista que guarda todos os usuários
- **room**: Lista das salas criadas, cada uma com uma lista de usuários.

Funcionalidades Adicionais

Criamos o comando "- *-list*", o qual lista todas as salas de chat que já foram criadas no servidor. Sentimos a falta deste comando para que os usuarios ao entrarem conhecerem as salas que estão disponíveis no sistema.

Política de Troca de Mensagens

Criamos uma politica de mensagens a qual o servidor saberá discernir entre um comando e uma mensagem para outro usuário.

Comandos	
Mudar NickName	- -nickname [name]
Criar Chat	- -chat [name]
Entrar Chat	- -join [chat name]
Listar Chats	- -list
Sair Chat	- -leave [chat name]
Fechar Chat	- -close [chat name]
Help	- -help

Concorrência

Os serviços que envolvem as listas de salas e usuários tem que ser protegidas por sessões críticas, pois não temos como garantir que dois usuários não vão provocar acessos as listas ao mesmo tempo. Para resolver isso utilizamos um *mutex*. Dessa maneira, o acesso aos serviços de criar salas, adicionar e remover usuários, por exemplo, é sequêncial. Uma melhoria seria uma análise mais profunda para ver se alguns desses serviços podem ser realmente concorrentes. Contudo, uma dificuldade seria testar o sistema, pois tentar inserir um erro de condição de corrida teríamos que ter uma quantidade considerável de usuários.