

Servidor Concorrente TCP

Leandro Kümmel Tria Mendes RA033910
Fernando Teixeira RA??????

15 de abril de 2013

Sumário

1	Introdução	1
2	Desenvolvimento	1
2.1	Protocolo TCP - Transmission Control Protocol	2
2.2	Implementação	2
2.2.1	Manipulação de dados	3
2.2.2	Conexão Servidor/Cliente	3
2.3	Coleta e gerência de dados para testes	3
2.4	Vantagens da implementação	4

Lista de Figuras

1	Fluxogramas	4
2	Definição do cálculo dos tempos	5

Lista de Tabelas

1 Introdução

O objetivo desse projeto é implementar um sistema cliente/servidor concorrente, com operações para o gerenciamento de livros em uma livraria. Na comunicação entre cliente e servidor utilizou-se o protocolo TCP¹, da camada de transporte, e a partir da execução de testes podemos avaliar alguns aspectos desse protocolo e posteriormente compará-lo com outro(s).

2 Desenvolvimento

Linux foi o sistema operacional utilizado para o desenvolvimento (distribuição 2.6.43.8-1.fc15.i686). Igualmente, para os testes utilizou-se duas máquinas com linux, porém com distribuições diferentes.

¹ Mais informações sobre TCP <http://www.linktionary.com/t/tcp.html>

2.1 Protocolo TCP - Transmission Control Protocol

O protocolo TCP foi escrito de modo a garantir que os dados enviados (pelo servidor) e recebidos (pelo cliente) de forma correta, na sequência adequada e sem erros, pela rede.

As características fundamentais do TCP são:

- *Orientado à conexão*: necessidade de uma conexão.
- *Ponto a ponto*: conexão é estabelecida entre dois pontos.
- *Confiabilidade*: Permite a recuperação de arquivos perdidos, elimina arquivos duplicados, recupera dados corrompidos, entrega na ordem do envio e pode recuperar o link entre cliente e servidor, caso esse, por algum motivo, seja perdido.
- *Full duplex*: Possível transferência simultânea, entre cliente e servidor
- *Handshake*: Mecanismo de estabelecimento e finalização de conexão. O TCP garante que, no final da conexão, **todos** os pacotes sejam entregues /recebidos
- *Entrega ordenada*: A aplicação entrega ao TCP bloco de dados de tamanho variável. Esse protocolo divide estes dados em segmentos de tamanho especificado (valor MTU). Sabe-se que camadas inferiores a de transporte podem fazer com que os pacotes não cheguem na ordem em que foram enviados. Porém, o TCP garante a reconstrução dos segmentos no cliente (TCP utiliza um número de sequência)
- *Controle de fluxo*: O protocolo em estudo utiliza-se de um campos denominado janela para controlar o fluxo

2.2 Implementação

O projeto conta com cinco diretórios, cada um com seu Makefile (exceto relatório e estat), arquivos principal (main.c e main.h) e um README.md² para instruções adicionais. Há também um Makefile, o qual compi

Os diretório são:

- common*: Contém arquivos de uso comum, tanto pelo servidor quanto pelo cliente, inclusive o arquivo que calcula a média dos testes executados.
- server*: Contém os arquivos que preparam uma porta para esperar conexões e manipulam o sistema de livreria.
- client*: Funções que provêm conexão com um servidor, envio das opções escolhidas pelo cliente e interface para as respostas do sistema de livreria (servidor).
- estat*: Medidas de tempo efetuadas pelo teste.
- relatorio*

²Leia esse arquivo antes de executar o sistema

2.2.1 Manipulação de dados

Todas as estruturas utilizadas para leitura/escrita dos livros são dinâmicas. Arquivos presentes no diretório *common*[I]:

- I. *error.c error.h*: Gerencia erros que eventualmente podem ocorrer.
- II. *common.c common.h*: Funções de uso comum.
- III. *avl.c avl.h*: Gerencia a estrutura básica da livraria, utiliza-se árvore AVL³, pois a busca, inserção/atualização têm complexidade $O(\log N)$, sendo N o número de elementos na árvore, no caso a quantidade de livros diferentes.
- IV. *archives.c archives.h*: Manipula arquivos. Faz a leitura do arquivo da livraria⁴.
- V. *tcp.c tcp.h*: Contém apenas algumas constantes.
- VI. *books.c books.h*: Gerencia a estrutura básica de um livro e seus autores.
- VII. *tempo.c tempo.h*: Gerencia a estrutura de testes, lê e escreve em arquivos localizados no diretório *estat*[IV].
- VIII. *livros/livros*: Arquivo contendo os livros⁵.

2.2.2 Conexão Servidor/Cliente

Compreende dois diretórios *server*[II] e *client*[III]

Servidor:

- I. *server.c server.h*: Apenas inicia o servidor dada um número de uma porta.
- II. *tcp_server.c tcp_server.h*: Gerencia tanto as conexões com os clientes quanto a comunicação, em outras palavras, o *tcp_server.c* recebe uma mensagem do *tcp_client.c*[II] e envia uma resposta adequada ao mesmo.
- III. *login.c login.h*: Gerencia o login necessário para editar a quantidade de um livro.

Cliente:

- I. *client.c client.h*: Apenas inicia a comunicação com um servidor dado o endereço IP e um número de uma porta.
- II. *tcp_client.c tcp_client.h*: Gerencia tanto a criação de uma conexão entre o cliente e servidor quanto a leitura, da entrada dada pelo usuário do sistema, e a comunicação entre hospedeiro e cliente.

2.3 Coleta e gerência de dados para testes

Para realizar os testes implementou-se alguns arquivos adicionais[VII]. Uma constante, denominada NUM_TESTES⁶, contém o número de testes a serem realizados, ou seja, cada opção do *menu*[II] é executada NUM_TESTES vezes. Todos os dados são salvos no diretório *estat*[IV].

³ <http://pages.cs.wisc.edu/~ealexand/cs367/NOTES/AVL-Trees/index.html>

⁴Ver README.md para mais detalhes do arquivo da livraria

⁵Ver README.md para mais detalhes do arquivo da livraria

⁶Nesse sistema consideramos NUM_TESTES igual a 100

2.4 Vantagens da implementação

O sistema de livreria é um sistema robusto e com baixa complexidade de tempo. A escolha da estrutura de árvore avl[III], possibilitou em boa performance em questão de tempo de processamento, uma vez que, essa estrutura mostrou-se eficaz para o problema e tem possui melhor complexidade de tempo com relação a outras estruturas, além de ser de a implementação e manutenção serem simples.

Com relação as conexões e comunicações entre cliente/servidor, vale ressaltar que há uma troca de mensagens inicial entre os dois, na qual o conteúdo da mensagem é o número de bytes da maior mensagem possível a ser enviada pelo servidor.

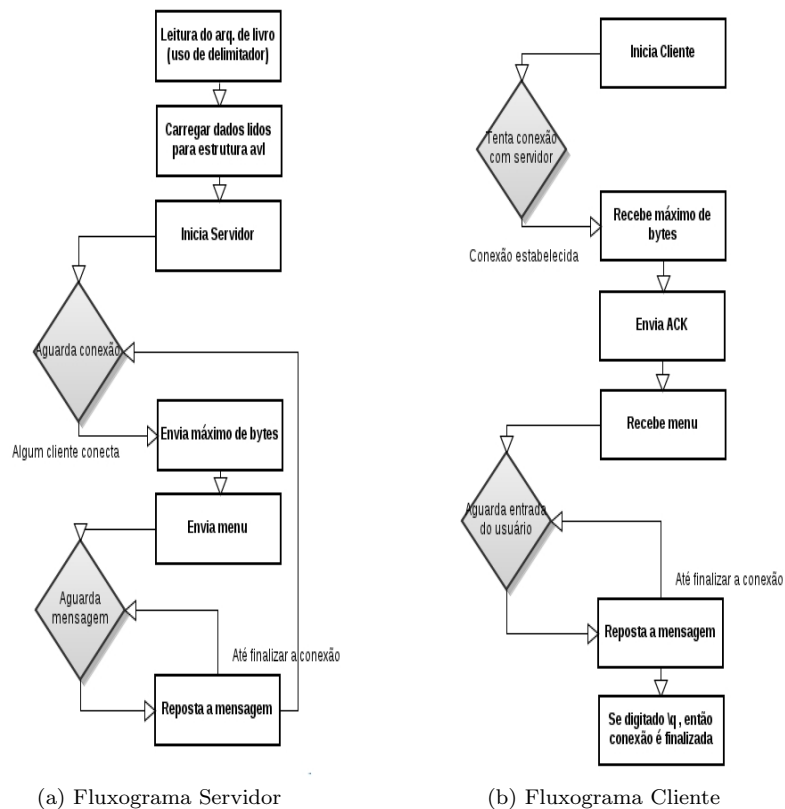


Figura 1: Fluxogramas

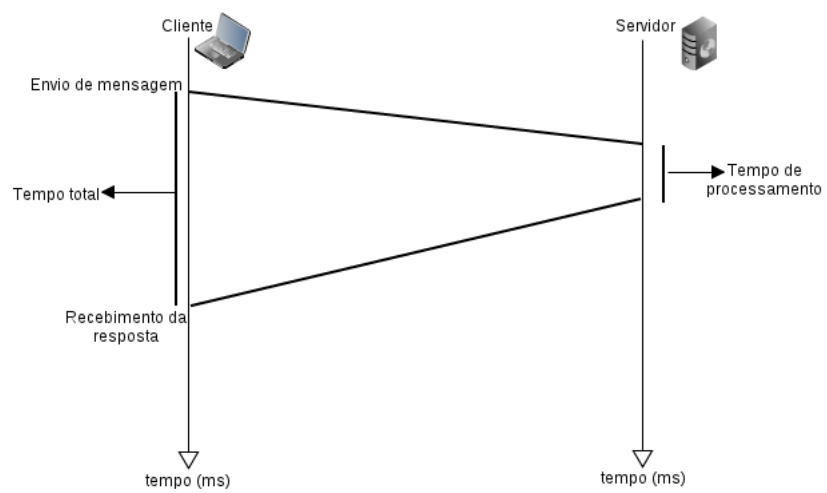


Figura 2: Definição do cálculo dos tempos