Introdução a Sistemas Computacionais

Trabalho Prático 1

Luiz Filipe Perini Giacomin
luizpgiacomin@ufmg.br

1 - Introdução

Como parte da disciplina de Redes de Computadores, o primeiro trabalho prático consistiu em criar um jogo do tipo Labirinto. O projeto exigia a implementação de uma comunicação cliente-servidor através de sockets, utilizando a linguagem de programação C.

2 - Implementação

O desenvolvimento do projeto foi realizado integralmente em **linguagem C** no ambiente Linux (WSL - Ubuntu), utilizando a interface POSIX para o gerenciamento de sockets. Para melhor estruturação do código, o projeto foi dividido em múltiplos arquivos: **server.c** e **client.c**, que contém as implementações do servidor e cliente, respectivamente, além de arquivos complementares (**common.c** e **common.h)** que incluem funções auxiliares compartilhadas entre ambos os componentes do sistema. Importante lembrar: nos comandos para iniciar o servidor e o cliente, digitar '/bin' antes dos comandos, já que os arquivos compilados são direcionados a uma pasta 'bin' para melhor organização do projeto. Assim, os comandos ficam:

IPv4:
no terminal 1: ./bin/server v4 51511 -i input/in.txt
no terminal 2: ./bin/client 127.0.0.1 51511

IPv6:
no terminal 1: ./bin/server v6 51511 -i input/in.txt
no terminal 2: ./bin/client ::1 51511

3 - Desafios, dificuldades e imprevistos

3.1) Gerenciamento de Conexões

Desafio: Um dos principais desafios foi manter uma conexão estável entre cliente e servidor, especialmente considerando que cada comando do cliente precisava de uma resposta do servidor.

Solução:Foi implementado um sistema de comunicação baseado em sockets TCP, onde:

- O cliente mantém uma única conexão durante toda a sessão do jogo
- O servidor processa cada comando de forma síncrona

- Foi implementado tratamento adequado de desconexões e erros de comunicação

3.2) Detecção Automática do Tamanho do Tabuleiro

Desafio: O programa precisava ler tabuleiros de diferentes tamanhos (5x5 até 10x10) sem que o tamanho fosse explicitamente informado no arquivo.

Solução: Foi desenvolvido um algoritmo que:

- Conta o número de linhas e colunas no arquivo
- Verifica se o tabuleiro é quadrado
- Valida se as dimensões estão dentro dos limites permitidos
- Realiza verificações de integridade do formato

3.3) Implementação do Sistema de Descoberta

Desafio: O jogador deveria descobrir o labirinto gradualmente, com células não visitadas permanecendo ocultas.

Solução: Foi criado:

- Uma matriz separada para controlar células descobertas
- Um sistema que revela células adjacentes ao jogador
- Lógica para mostrar '?' em células não descobertas

Revelação completa do mapa apenas após a vitória

3.4) Algoritmo de Busca para Dicas
Desafio: Implementar um sistema de dicas que mostrasse o caminho até a saída a partir da posição atual do jogador.

Solução: Foi implementado:

- Algoritmo BFS (Breadth-First Search) para encontrar o menor caminho
- Sistema de fila para gerenciar a busca
- Armazenamento do caminho encontrado
- Conversão do caminho em direções (up, right, down, left)

3.5) Gerenciamento de Estado do Jogo

Desafio: Manter consistência entre estados do cliente e servidor, especialmente após vitória ou reset.

Solução: Foi implementado:

- Variáveis de controle para estado do jogo
- Tratamento especial para comandos pós-vitória

- Sistema de reset que reinicializa corretamente todas as variáveis
- Validação de comandos baseada no estado atual

3.6) Tratamento de Erros

Desafio: Garantir robustez do sistema contra entradas inválidas e condições inesperadas.

Solução: Foi adicionado:

- Validação extensiva do arquivo de entrada
- Verificação de comandos válidos
- Tratamento de erros de alocação de memória
- Mensagens de erro informativas

4 - Conclusão

O projeto apresentou diversos desafios interessantes, principalmente relacionados à comunicação em rede e gerenciamento de estado do jogo. A abordagem modular adotada, separando claramente as responsabilidades entre cliente e servidor e a utilização do Doxygen (ferramenta utilizada para documentar/comentar o código do projeto) facilitaram o desenvolvimento e a manutenção do código.

A implementação do algoritmo BFS para as dicas e o sistema de descoberta gradual do labirinto foram particularmente interessantes e desafiadoras, pois exigiram um equilíbrio entre funcionalidade e eficiência.

As soluções adotadas priorizaram:

- Robustez do sistema
- Clareza do código
- Experiência do usuário
- Facilidade de manutenção

O resultado final é um jogo funcional que atende a todos os requisitos especificados, mantendo um código organizado e bem documentado.

5 - Referências

- **Youtube:**

- Playlist 'Introdução a Programação em Redes' - Professor Ítalo cunha

- **Livros:**

- TCP IP Sockets in C, Second Edition Practical Guide for Programmers

- **Web:**

- https://www.geeksforgeeks.org/breadth-first-search-or-bfs-for-a-graph/
- https://www.geeksforgeeks.org/introduction-and-array-implementation-of-queue/
- https://www.doxygen.nl/manual/
- https://gamedev.stackexchange.com/questions/134728/fog-of-war-algorithm