Universidade Federal Do Espírito Santo - Departamento de Informática
Estrutura de Dados 1

NetMap

1º Trabalho Prático
Período: 2020/2 - EARTE

Alunos: Estevão Nunes da Silva, Hugo Lima Otoch

Este trabalho tem como objetivo praticar o uso de tipos abstratos de dados e estruturas do tipo Lista.

- Introdução :

Este relatório visa mostrar as etapas da resolução do trabalho NetMap proposto na disciplina Estrutura de Dados 1 (INF 09292), um NetMap é uma rede que permite a passagem de dados entre terminais distribuídos. Assim como a internet, NetMap é composta por roteadores que se conectam através de enlaces que habilitam a passagem de dados pela rede. Conectados a roteadores, estão os terminais. Cada terminal pode estar conectado a somente um roteador. Já um roteador pode estar conectado a vários terminais.

Para implementar esse trabalho foi proposto que utilizasse a linguagem C e as ferramentas de listas encadeadas juntamente com alocação dinâmica de memória, conceitos aprendidos em aula. Para a resolução desse trabalho lançamos mão de três etapas, primeiro desenvolvemos diagramas ilustrativos visando facilitar o entendimento dos algoritmos necessários para a resolução do desafio, em seguida partimos para a segunda etapa, esta é a etapa de implementação do algoritmo, e por fim a terceira etapa consistiu em baterias de testes e correções de problemas.

1 - Diagramas ilustrativos :

Para descrever esta seção utilizamos a ferramenta Figma que é um editor gráfico, dessa forma segue o link para visualizar os diagramas ilustrativos que representará a estrutura do projeto proposto.

https://www.figma.com/file/cDjzxfVb59VDw40jgSIc4B/Projeto-ED?node-id=0%3A1

2 - Implementação :

Nesta seção vamos descrever o funcionamento das principais funções desenvolvidas para resolução do projeto NetMap.

- **void Le_e_executaComando(FILE* entrada, ListaRot* listaROT, ListaTerm* listaTERM,FILE* log, int* idRot,int* idTerm,FILE* saida, FILE* dot);**

  Esta função tem por responsabilidade ler o arquivo de entrada, este é onde estão dispostos os comandos a serem realizados pelo programa, após ler o comando essa função executa o comando passado e chama uma das funções abaixo.

- **Cadastrar roteador(listaROT,idROT,nomeRot,nome Operadora):**

  Esta função inclui o roteador passado por parâmetro na lista de roteadores, alocando-o dinamicamente na lista.

- **Cadastral Terminal(idTERM,nomeTerm,locTerm.listaTERM):**

  Esta função inclui o terminal passado por parâmetro na lista de terminais, alocando-o dinamicamente na lista.

- **RemoveRoteador(celR, listaROT):**

  Antes de executar essa função usamos uma outra que tem por funcionalidade buscar o roteador a ser removido na lista de roteadores e devolver a célula ‘R’ desse mesmo roteador, essa função também tem por objetivo registrar no log possíveis erros como a lista de roteadores está vazia ou o roteador em questão ser inexistente no NetMap.

  Logo após realizar a busca e retornar o roteador a ser removido a função RemoveRoteador é acionada, primeiro ela remove o roteador incluso à célula ‘R’ passada por parâmetro, logo em seguida ela remove esse mesmo roteador da lista de enlaces dos demais roteadores do NetMap, por fim essa função libera o roteador removido e todos os seus enlaces.

- **ConectaTerminal(celT,celR):**

  Antes de executar essa função usamos outras duas que tem por funcionalidade buscar em suas listas correspondentes, o roteador e o terminal a serem conectados, retornando respectivamente a célula ‘R’ do roteador e a célula ‘T’ do terminal, essas funções também tem por

objetivo registrar no log possíveis erros como as listas estarem vazias ou os elementos em questão são inexistente na lista.

Logo após realizar as buscas e retornarem o terminal e o roteador a serem conectados a função Conecta Terminal é acionada, ela irá apontar o ponteiro do tipo roteador presente na estrutura do terminal para o roteador ao qual queremos conectar o terminal.

- **DesconectaTerminal(Celula_T):**

  Antes de executar essa função usamos uma outra que tem por funcionalidade buscar o terminal a ser desconectado na lista de terminais e devolver a célula ‘T’ deste terminal, essa função também tem por objetivo registrar no log possíveis erros como o NetMap não contém terminais ou o terminal que estamos tentando desconectar é inexistente no NetMap.

  Logo após realizar a busca e retornar o terminal a ser desconectado a função Desconectar Terminal é acionada, essa função apontará o ponteiro do tipo roteador presente na estrutura do terminal para NULL.

- **RemoveTerminal(Celula_T, listaT):**

  Antes de executar essa função usamos uma outra que tem por funcionalidade buscar o terminal a ser removido da lista de terminais e devolver a célula ‘T’ deste terminal, essa função também tem por objetivo registrar no log possíveis erros como o NetMap não contém terminais ou o terminal que estamos tentando remover é inexistente no NetMap.

  Logo após realizar a busca e retornar o terminal a ser removido a função Remove Terminal é acionada, esta função removerá o terminal em questão da lista de terminais tomando os devidos cuidados para que a lista mantenha-se encadeada.

- **ConectaRoteadores(Roteador1,Roteador2):**

  Antes de executar essa função usamos uma outra função que tem por funcionalidade buscar os roteadores a serem conectados e retornar a célula ‘R’ desses roteadores, essa função também tem por objetivo registrar no log possíveis erros como o NetMap não contém Roteadores ou os Roteadores que estamos tentando conectar são inexistentes no NetMap.

  Logo após realizar a buscar e retornar os roteadores que serão conectados a função ConectaRoteadores é acionada, esta função irá

adicionar o Roteador 1 na lista de enlaces do Roteador 2, e adicionar também o Roteador 2 na lista de enlaces do Roteador 1.

- **DesconectaRoteadores(Roteador1,Roteador2):**

  Antes de executar essa função usamos uma outra função que tem por funcionalidade buscar os roteadores a serem desconectados e retornar a célula 'R' desses roteadores, essa função também tem por objetivo registrar no log possíveis erros como o NetMap não contém Roteadores ou os Roteadores que estamos tentando desconectar são inexistentes no NetMap.

  Logo após realizar a buscar e retornar os roteadores que serão desconectados a função DesconectaRoteadores é acionada, essa função removerá o Roteador 1 da lista de enlaces do Roteador 2 e removerá também o Roteador 2 da lista de enlaces do Roteador 1.

- **Frequencia Terminal(listaTERM,nomeLoc,saida):**

  Esta função irá percorrer toda a lista de terminais buscando e contando quantos terminais têm a mesma localização da qual foi passada por parâmetro, no final irá imprimir esse valor no arquivo de saída.

- **Frequência Operadora(lista ROT,nome Operadora,saída):**

  Esta função irá percorrer toda a lista de roteadores buscando e contando quantos roteadores têm a mesma operadora do qual foi passada por parâmetro, no final irá imprimir esse valor no arquivo de saída.

- **EnviaPacotesDados(Terminal1,Terminal2):**

  Começamos buscando os roteadores ligados aos terminais passados por parâmetros (Roteador 1 => Terminal 1 e Roteador 2 => Terminal 2). Essa função irá começar verificando os enlaces do Roteador 1 buscando pelo Roteador 2, caso não for encontrado a função irá entrar na lista de enlaces de cada roteador conectado ao Roteador 1, repetindo o processo (através da Recursão) até que encontre o Roteador 2 ou verifique todos os roteadores ligados diretamente e indiretamente com o Roteador 1. Para isso, criamos um vetor de ID para gravar os IDs dos roteadores que já foram verificados impossibilitando um loop infinito, o tamanho desse vetor é sempre realocado de acordo com o número de roteadores analisados. Para a fu nção verificar a lista de enlaces de um

próximo roteador ela precisa buscar pelo ID desse roteador a ser analisado no vetor e confirmar que o mesmo ainda não foi verificado.

Problemas: Quando iniciamos o vetor de ID com tamanho 1 por algum motivo que não conseguimos identificar obtemos vários erros, portanto iniciamos o vetor com tamanho 2 usando a função calloc para iniciar todos os elementos desse vetor com 0, dessa forma não interferindo no funcionamento vide que os ID's dos roteadores são maiores que 0.

- **ImprimeNetMap():**
  Sempre que chamada essa função tem por responsabilidade imprimir o NetMap, este é sua lista de terminais e a quais roteadores esses terminais estão conectados, sua lista de roteadores e seus respectivos enlaces. Essa impressão será feita no arquivo saída.dot.

3 - Erros e dificuldades encontrados :

- Tivemos alguns erros em relação aos frees na função RemoveRoteador e na EnviaPacoteDados que no qual soubemos identificar o motivo porém não conseguimos resolvê-lo. Em relação aos arquivos saida.txt, log.txt e saida.dot os resultados são os mesmos esperados.

- Nossa primeira dificuldade foi em relação a lista de enlaces, pois demoramos um pouco para entender que eram duas structs de listas distintas. Nossa outra dificuldade foi com a Envia Pacote Dados, que no qual tivemos que usar uma Recursão para resolver o problema (criando assim erros no free em relação ao realloc que fizemos com o vetor de ID).

Conclusão :

Conseguimos concluir o trabalho, porém obtivemos problemas já documentados nesse arquivo.

Bibliografia/Ferramentas utilizadas :

https://www.figma.com/

Usamos o Figma para criar os diagramas ilustrativos, a abordagem visual foi fator determinante para que se entendesse o problema e iniciasse a fase 2, de implementação do código.

https://code.visualstudio.com/

Usamos o Vscode para implementar os códigos do trabalho proposto, a extensão LiveShare do Vscode foi de extrema importância pois como o trabalho foi realizado em dupla, está se mostrou como alternativa para que os dois integrantes da dupla pudessem escrever o código juntos em tempo real.

https://github.com/estevaons/trabalho-ED

Usamos o Github como ferramenta de versionamento e hospedagem do projeto.

https://valgrind.org/

Usamos o valgrind e suas diversas funcionalidades.