ANDERSON LEANDRO FERNANDES BATALHA, FILIPE SILVA BATISTA e MARIANA MOTA OLIVEIRA

FRANCISCO GLAUBOS NUNES CLIMACO

LINGUAGEM DE PROGRAMAÇÃO I

18 de junho de 2024

Localização de Centros de Distribuição

O código implementa um sistema para encontrar a melhor localização de hubs de distribuição em um cenário de cidades interconectadas. Ele utiliza o algoritmo de Floyd-Warshall para calcular todas as distâncias mais curtas entre as cidades, fundamental para determinar os custos de distribuição mínimos.

ARQUITETURA DO SISTEMA

- **"read_input(const char *cenário)"** - Onde podemos ler os dados de entrada do arquivo (cenário), a função é responsável por obter as informações necessárias para o cálculo das rotas mais curtas.
- **"floyd warshall ( )"** - Implementamos o algoritmo de Floyd-Warshall para calcular todas as distâncias mais curtas entre pares de cidades na matriz de distâncias, é um passo crucial para o programa determinar posteriormente os custos mínimos de distribuição usando diferentes hubs.
- **"achar_melhor_local ( )" -** Utilizamos uma abordagem de força bruta para testar todas as combinações possíveis de hubs e determinar aquela que minimiza o custo total de distribuição. A função é responsável por encontrar os melhores locais para os hubs de distribuição, utilizando as distâncias pré-calculadas pelo algoritmo de Floyd-Warshall.
- **"alocar_matriz_distancias ( )"** - Gerencia a alocação e liberação dinâmica de memória para a matriz de distâncias entre as cidades.
- **"liberar_matriz_distancias( )"** - Permite que o programa trabalhe eficientemente com diferentes tamanhos de entrada, utilizando apenas a memória necessária.
- **"combinacao_valida"** - Faz a verificação se a combinação de hubs é válida.

- **“calcular_custo”** - Realiza o cálculo do custo total de distribuição usando os hubs escolhidos, para cada cidade, encontra a menor distância para qualquer um dos hubs escolhidos e soma essas distâncias para obter o custo total.
- **“main( )”** - A interface do nosso usuário.

DECISÕES DE DESIGN

- **Algoritmo de Floyd-Warshall** - Decidimos pela aplicação deste algoritmo devido à sua eficiência em encontrar caminhos mais curtos em grafos ponderados. Ele nos garante que todas as possíveis rotas entre as cidades sejam consideradas, permitindo uma análise completa das opções de distribuição e uma maior eficiência.
- **Modularidade:** O código está dividido em funções que realizam tarefas específicas (alocação de memória, leitura de entrada, cálculo de distâncias, seleção de hubs), o que facilita a compreensão, manutenção e reutilização do código.
- **Uso de Estruturas de Dados:** A estrutura “problema” engloba todos os dados relacionados ao problema, tornando o código mais organizado e reduzindo o acoplamento entre diferentes partes do programa.
- **Abordagem de Força Bruta para Hubs** - Embora seja simples, a abordagem de força bruta para o teste de todas as combinações de hubs é eficaz para garantir a solução ótima em um espaço de busca limitado, especialmente para problemas de tamanho moderado.
- **Alocação Dinâmica de Memória** - Utilizamos alocação dinâmica para a matriz de distâncias, já que é necessário para lidar com a entrada variável de cidades e estradas sem desperdício de memória.

INSTRUÇÕES DE EXECUÇÃO

Para abrir o programa:

- Abra o cmd dentro da pasta

  Digite "hubs.exe" (sem aspas)
- Escreva o nome do arquivo do cenário desejado (sc1.txt , sc2.txt ou sc3.txt)

OBS: Será necessário digitar "hubs.exe" no terminal novamente sempre que for testar cada um dos arquivos.

Caso seja necessário compilar o arquivo em .c para .exe:

- Abra o cmd dentro da pasta

  Digite “gcc hubs.c -o hubs.exe” (sem aspas)
- Pelo mesmo terminal, digite “hubs.exe” (sem aspas)
- Quando o programa pedir o nome do arquivo de entrada:
- Ao executar o programa no cmd, a seguinte mensagem será exibida:

  “Digite o nome do arquivo de entrada (exemplo, sc1.txt)"
- Em seguida escreva o nome do arquivo do cenário desejado (sc1.txt , sc2.txt ou sc3.txt)