Universidade Federal de Minas Gerais
Departamento de Ciência da Computação
Algoritmos I

Trabalho Prático 1

Henrique Gomes dos Santos Medeiros
2021084986

Belo Horizonte
2024

Introdução

O objetivo deste trabalho é resolver o problema do fluxo máximo, ou seja, encontrar um fluxo viável através de uma rede que obtém a vazão máxima possível. Aqui se trata de uma rede elétrica que conecta pontos de fornecimento e pontos de consumo de energia; geradores e consumidores respectivamente.

Além de dizer qual a capacidade máxima que a rede comporta, também deve-se responder três outras questões:

- Quanta energia falta para os consumidores operarem efetivamente?
- Quanta energia é desperdiçada ao longo da rede?
- Quais as conexões que demandam mais energia na rede?

Modelagem

O problema foi modelado como um grafo dirigido e ponderado (com valores associados aos nós e às arestas) onde os nós representam os pontos de fornecimento e consumo (points) e as arestas representam as conexões (cables) entre esses pontos.

O algoritmo principal utilizado para resolver o problema foi o algoritmo de Edmonds-Karp para fluxo máximo.

Estruturas de Dados:

Para estruturação do grafo foi utilizado uma lista de adjacência no formato vector<vector<Edge>. "Edge", naturalmente, é uma struct que representa uma aresta e contém os atributos *to*, *capacity* e *flow.*

Outra estrutura que vale a pena destacar é o retorno da função *findCriticalConnections*: vector<tuple<int, int, int>>, ou seja, um vetor de tuplas contendo três inteiros, que representam *from*, *to*, e *capacity*, em termos de aresta.

Algoritmos Utilizados:

- **Busca em Largura (BFS)**: Utilizado para encontrar caminhos de fluxo residual no grafo (necessário para o algoritmo de fluxo máximo).
- **Algoritmo de Fluxo Máximo - Edmonds-Karp**: Implementa o fluxo máximo usando o método Ford-Fulkerson com BFS.

- **Algoritmo de Ordenação**: Para ordenar as arestas críticas de acordo com a capacidade e outros critérios. Apesar da semelhança não tão clara, foi inspirado nas verificações caso a caso do bubble sort.

Solução

Energia Total (E_total)

A solução para o problema de energia total se dá ao resolver o problema do fluxo máximo. O máximo de energia que a rede consegue transmitir é o máximo fluxo possível na rede.

A resposta foi obtida através do algoritmo de Edmonds-Karp:

```
1. Função BFS(G, s, t, parent)
   Entrada:
      G = Grafo com capacidades residuais
      s = Nó de origem (source)
      t = Nó de destino (sink)
   Saída:
      parent = Vetor de pais, onde parent[u] é o nó anterior
de u no caminho encontrado

   Inicializar fila F vazia
   Inicializar vetor parent de todos os nós como -1 (sem pai)
   Enfileirar s em F (começo no nó fonte)
   parent[s] = -1 (sem pai para a fonte)

   Enquanto F não estiver vazia:
      Remover nó u de F
      Para cada aresta (u, v) com capacidade residual > 0:
         Se parent[v] == -1 (v não foi visitado):
            parent[v] = u  // Registra o nó u como pai de v
            Se v == t:
               Retorne True (caminho encontrado)
            Enfileirar v em F
   Retorne False (não há caminho de s a t)

2. Função Edmonds-Karp(G, s, t)
   Entrada:
      G = Grafo com capacidades residuais
      s = Nó de origem (source)
      t = Nó de destino (sink)
```

```
Saída:
   maxFlow = Fluxo máximo de s a t

Inicializar maxFlow = 0

Enquanto BFS(G, s, t, parent) for True:
   // Encontrar o caminho aumentante com a BFS
   Inicializar pathFlow = infinito
   v = t
   Enquanto v != s:
      u = parent[v]
      pathFlow = min(pathFlow, capacidade residual(u, v))
// Capacidade mínima no caminho
      v = u

   // Atualizar capacidades residuais
   v = t
   Enquanto v != s:
      u = parent[v]
      // Reduzir a capacidade da aresta original
      capacidadeResidual(u, v) -= pathFlow
      // Aumentar a capacidade da aresta reversa
      capacidadeResidual(v, u) += pathFlow
      v = u

   // Adicionar o fluxo encontrado ao fluxo total
   maxFlow += pathFlow

Retorne maxFlow
```

Energia Não-Atendida (E_missing)

A solução para o problema de energia não-atendida consiste em uma formula simples:

`unmetDemand = total_demand - maxEnergy`;

energia não-atendida = soma da demanda dos consumidores - soma da energia que chega até eles

Energia Perdida (E_loss)

A energia perdida é a soma das capacidades dos cabos que saem dos geradores menos a energia que chega aos consumidores. Para chegar a essa resposta, foi contabilizada a capacidade máxima de todos os cabos que saíam do gerador em uma variável (**`generated_energy`**) e então desse valor foi descontado o valor de energia que chega aos consumidores, equivalente à energia máxima que passa pelo sistema (`maxEnergy`).

Conexão Crítica (P_critical)

Para as conexões críticas, buscamos no grafo as arestas operando em suas capacidades máximas. Feito isso, as ordenamos segundo suas capacidades, depois pelo nó de origem, e por fim pelo nó de destino.

Análise de Complexidade

BFS (Busca em Largura):

- **Tempo**: O tempo de execução é O(V + E), onde V é o número de vértices e E é o número de arestas.

Cálculo do Fluxo Máximo:

- **Tempo**: O fluxo máximo é calculado usando a BFS repetidamente, levando a um tempo total de $O(VE^2)$. Essa é a complexidade do algoritmo de Edmonds-Karp utilizando a busca em largura.

Ordenação das Conexões Críticas:

- **Tempo**: $O(n^2)$, onde n é o número de conexões críticas. Isso ocorre devido à ordenação manual.

Complexidade Total:

- **Tempo**: A complexidade total do código é dominada pelos passos mais pesados, sendo assim dada pela complexidade do algoritmo de fluxo máximo, $O(VE^2)$.

Considerações Finais

Este trabalho aborda uma aplicação prática de teoria de grafos para resolver problemas de distribuição de energia. Para isso, foi utilizado o algoritmo de Edmonds-Karp, uma implementação do algoritmo de Ford-Fulkerson, que se mostrou eficiente para resolver o problema de fluxo máximo. A análise foi feita em termos de energia total, energia não atendida, energia perdida e identificação de conexões críticas, com a solução computada por meio de uma lista de adjacência. A escolha dessa estrutura se justifica pela sua eficiência em redes esparsas, como é o caso de redes elétricas.

Apesar de contemplar um caso fictício e simplificado para os propósitos de ser usado como um trabalho acadêmico, a solução apresentada pode ser aplicada a sistemas elétricos reais, fornecendo insights valiosos sobre o desempenho da rede e áreas que necessitam de melhorias. A identificação de conexões críticas também foi essencial para destacar as arestas que estão operando em suas capacidades máximas, ajudando na priorização de investimentos e manutenção na rede elétrica. As abordagens e algoritmos utilizados são fundamentais para otimizar a distribuição de energia e garantir o atendimento eficaz das demandas dos consumidores.

Referências

https://www.programiz.com/dsa/ford-fulkerson-algorithm

https://www.geeksforgeeks.org/ford-fulkerson-algorithm-for-maximum-flow-problem/

https://en.wikipedia.org/wiki/Maximum_flow_problem

https://www.geeksforgeeks.org/max-flow-problem-introduction/

https://www.w3schools.com/dsa/dsa_algo_graphs_edmondskarp.php

https://pt.wikipedia.org/wiki/Algoritmo_de_Edmonds-Karp

https://www.youtube.com/watch?v=Tl90tNtKvxs

https://www.youtube.com/watch?v=VbeTl1gG4l4

https://www.youtube.com/watch?v=LdOnanfc5TM&t=144s

https://www.youtube.com/watch?v=3LG-My_MoWc&t=648s

https://www.youtube.com/watch?v=GOwt6cISaG4

https://www.youtube.com/watch?v=oHy3ddI9X3o

https://www.youtube.com/watch?v=RppuJYwlcI8&t=9s

https://www.youtube.com/watch?v=SqGeM3FYkfo

https://www.youtube.com/watch?v=FIlB73vSl4s

Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). *Introduction to Algorithms*.

Aulas e notas fornecidas pelo curso.