Relatório de Resultados do Trabalho Prático N.02 de Teoria dos Grafos e Computabilidade

Bernard Paes Ferreira[1], **Fernando Wagner Gomes Salim**[1], **Nathan de Araújo Cunha Lisboa**[1]

[1]Instituto de Ciências Exatas e Informática –
PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS (PUCMG)
Av. Brasil, 2023 - Funcionários, Belo Horizonte - MG, 30140-002

Abstract. *This report aims to document the results obtained in the implementation of the second practical work in the subject of "Graph Theory and Computability", whose objective is to perform two implementations to solve an NP-Complex problem being an exact implementation (capable of obtaining the minimum solution for small instances) and the another approximate one (capable of obtaining reasonable solutions even when the instance is large too much for the exact method). Repository link https://github.com/BernardPaes/tp02 [rep]*

Resumo. *Este relatório tem como propósito documentar os resultados obtidos na implementação do segundo trabalho prático da matéria de "Teoria dos Grafos e Computabilidade", cujo objetivo é realizar duas implementações para resolver um problema NP-Complexo sendo uma implementação exata (capaz de obter a solução mínima para instâncias pequenas) e a outra aproximada (capaz de obter soluções razoáveis mesmo quando a instância for grande demais para o método exato). Link do repositório https://github.com/BernardPaes/tp02 [rep]*

1. Introdução

O problema dos k-centros é uma tarefa fundamental na análise de dados e se relaciona diretamente com técnicas de clustering. Dado um grafo completo com custos definidos nas arestas e um inteiro positivo k, o objetivo é encontrar um conjunto de k vértices, chamados centros, de modo a minimizar a maior distância de qualquer vértice do grafo até o centro mais próximo. Esta distância máxima é conhecida como o raio da solução.

A importância do problema dos k-centros reside em suas diversas aplicações práticas. Por exemplo, na categorização de consumidores, onde os dados de compras podem ser usados para medir a "distância"entre consumidores e agrupá-los em perfis homogêneos. Outra aplicação é na localização de instalações, como a abertura de centros de tratamento médico, onde é crucial minimizar o tempo de transporte dos pacientes ao centro mais próximo.

Resolver o problema dos k-centros de forma exata pode ser computacionalmente inviável para grandes instâncias, devido à sua complexidade NP. Portanto, abordagens aproximadas são frequentemente utilizadas para obter soluções aceitáveis em um tempo razoável. Neste contexto, a tarefa proposta neste trabalho é implementar e comparar duas abordagens distintas para resolver o problema dos k-centros: uma abordagem exata e uma abordagem aproximada.

2. Implementação

A linguagem de programação escolhida para desenvolver o trabalho foi Java. Para ambas implementações a fase inicial é a mesma: o arquivo do grafo é lido e é realizado o algoritmo de Floyd Warshall sobe a matriz de adjacência obtida do arquivo.

2.1. Implementação exata

Com a matriz de distâncias do Floyd Warshall pronta, desenvolvemos a função resolverKCentros(), que inicializa um array centros para armazenar os centros selecionados e invoca a função recursiva resolverKCentrosRecursivo(). Esta função explora todas as combinações possíveis de k centros de forma recursiva, começando do índice 0 até o número de vértices. Para cada combinação de centros, a função calcularRaio() é chamada para calcular o raio da solução, que é definido como a maior distância mínima de qualquer vértice ao centro mais próximo. A abordagem recursiva garante que todas as combinações possíveis de k vértices são avaliadas, permitindo encontrar a configuração que minimiza a maior dessas distâncias mínimas.

A função resolverKCentrosRecursivo() tenta todas as combinações possíveis de k vértices, e para cada combinação, calcula o raio máximo usando a função calcularRaio(). Esta última percorre todos os vértices, encontrando a menor distância de cada vértice aos centros selecionados, e determina a maior dessas distâncias mínimas. A combinação de centros que minimiza esta maior distância é então selecionada como a solução ótima. Este método garante a obtenção de uma solução exata para o problema dos k-centros em instâncias pequenas, atendendo ao requisito de encontrar a configuração de centros que minimiza a distância máxima de qualquer vértice ao centro mais próximo.

2.2. Implementação aproximada

Para a implementação aproximada, após o tratamento do arquivo com Warshall, é ultilizado de referência o código em [for Geeks], o algoritmo trata o problema como se cada vertice fosse uma cidade e ultiliza de uma ideia de busca gulosa, ele inicializa o array **dist** com valores máximos, procede a iterativamente adicionar o centro que possui maior distância ao conjunto de centros, atualiza as distâncias de todas as cidades para o centro mais próximo e por fim imprime o raio aproximado (maior distância de qualquer cidade para o centro mais próximo).

3. Experimentos e Resultados

Nesta seção, apresentaremos os resultados obtidos a partir dos experimentos realizados com os métodos de aproximação e exato. Os algoritmos foram testados com 40 conjuntos de testes com as seguintes características: número de vértices, número de arestas, para cada aresta os vértices finais e o custo da aresta. Para isso, vale salientar que a implementação exata é funcional para grafos de tamanhos moderados e poucos centros. No entanto, a abordagem de força bruta para resolver o problema dos k-centros é impraticável para os conjuntos de testes solicitados, devido ao crescimento exponencial da complexidade.

Devido a isso foi possível obter somente um resultado como presente na Figura 1, os demais arquivos foram testados e não retornaram resultado para duas horas de testes. Resultados para todas as instâncias obtidas pelo método que utiliza aproximação estão na Tabela 1

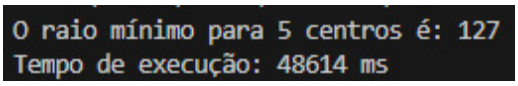

Figura 1. Resultados com tempo da primeira instância (pmed1.txt).

Instância	—V—	k	Raio
1	100	5	186
2	100	10	131
3	100	10	154
4	100	20	114
5	100	33	71
6	200	5	138
7	200	10	96
8	200	20	82
9	200	40	57
10	200	67	31
11	300	5	73
12	300	10	71
13	300	30	59
14	300	60	40
15	300	100	25
16	400	5	84
17	400	10	56
18	400	40	44
19	400	80	28
20	400	133	19
21	500	5	53
22	500	10	56
23	500	50	34
24	500	100	23
25	500	167	15
26	600	5	50
27	600	10	43
28	600	60	28
29	600	120	19
30	600	200	14
31	700	5	42
32	700	10	45
33	700	70	15
34	700	140	17
35	800	5	38
36	800	10	41
37	800	80	25
38	900	5	39
39	900	10	35
40	900	90	21

Tabela 1. Resultados obtidos com a solução aproximada

4. Referências

Referências

Repositorio github. https://github.com/BernardPaes/tp02.

for Geeks, G. Greedy approximate algorithm for k centers problem. https://www.geeksforgeeks.org/greedy-approximate-algorithm-for-k-centers-problem/.