Minimização de Fluxo em Movimentações Financeiras

Arthur Aguiar Menezes de Souza

Novembro de 2015

Resumo

Este relatório possui como propósito detalhar e apresentar uma solução para o problema proposto na disciplina de Algoritmos e Programação III, durante o segundo semestre de 2015: a partir de várias transações financeiras entre diversos correntistas, minimizar o fluxo monetário envolvido, a fim de minimizar o imposto total cobrado sobre as movimentações financeiras envolvidas no processo.

1 Introdução

No que seria considerado relevante ao escopo da disciplina de Algoritmos e Programação III, tem-se a definição do problema de tal forma:

Um número n de correntistas possui um número m de transações entre si. Dado um correntista a que transfere uma quantia x para um correntista b que, por usa vez, transfere uma quantia y para um correntista c, é possível que uma relevante quantia passada ao b, seja novamente repassada ao c. Levando em conta que 1% de cada transação destina-se à impostos, seria possível, de alguma forma, minimizar o fluxo de dinheiro envolvido para que o mesmo não seja repassado desnecessariamente?

Além de definir o problema na Seção 2, uma possível solução é mostrada na Seção 4, após a apresentação, na Seção 3, da estrutura de dados a ser utilizada pelo algoritmo posteriormente apresentado.

2 Problema

O problema consiste em eliminar o "repasse"do mesmo dinheiro. Por exemplo, se o correntista a transfere 500 reais para o b, que passa 250 para o c, então é obtido um fluxo total de 750 reais movimentados, como mostra a Figura 1. Porém, como pode-se notar, como o b recebeu 500 e transferiu 250, ele obteve um lucro de 250 reais, o mesmo lucro de c, o qual apenas recebeu 250. Sendo assim, é evidente que b e c receberam 250 reais cada. Logo, a fim de minimizar a quantia em fluxo, é possível reformular a relação de movimentações financeiras entre correntistas, contanto que a transfira o lucro de 250 reais para b, e transfira o lucro de 250 reais para c. O resultado disso é que, ao invés de movimentar um quantia total de 750 reais, apenas 500 reais são movimentado com o modelo simplificado, mantendo o efeito desejado.

Em uma representação por grafos, seria obtida uma minimização nas movimentações como a apresentada na Figura 2.

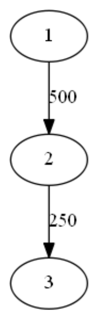


Figura 1: Original

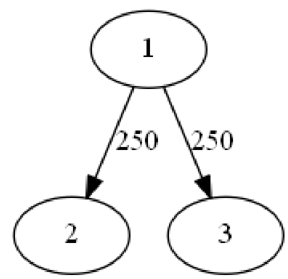


Figura 2: Minimizado

É claro, porém, que tal situação é simples demais se comparada à situações reais onde um correntista pode realizar diversas transações para diversos outros correntistas, os quais podem inclusive transferir novamente para o primeiro correntista. Quando isso acontece, é dito que o grafo possui um ciclo. As Figuras 3 e 4 ilustram a ideia de uma situação onde ocorre um ciclo e como isso também pode ser simplificado, respectivamente.

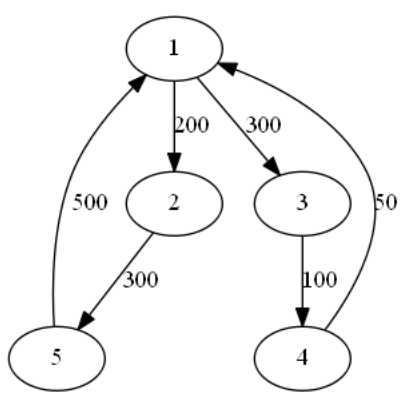


Figura 3: Original

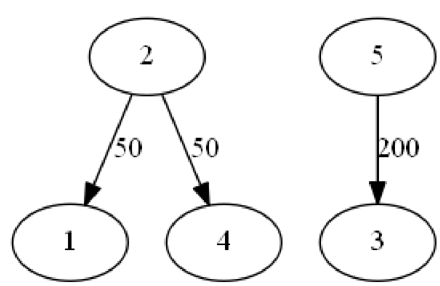


Figura 4: Minimizado

É notável que, desta vez, a minimização não só diminuiu a quantia resultante de todas as movimentações, como diminuiu a quantidade de movimentações pela metade: de seis para três.

3 Estrutura de Dados

Para a resolução deste problema, foi utilizada uma represenação de grafos. Um grafo é um conjunto de vértices e arestas que conectam vértices. Será utilizado um vértice para representar cada correntista e uma aresta entre dois vértices para representar cada movimentação financeira. O peso de uma aresta inidica o valor transferido e a direção da aresta indica de qual correntista partiu o dinheiro e para qual correntista o dinheiro partiu.

Quando partimos de um vértice a e, através de pelo menos um caminho, conseguimos voltar à esse vértice, dizemos que temos um ciclo.

Na Seção 4, é apresentado um algoritmo como possível solução para minimizar o fluxo de dinheiro conjunto de transações. Nesse algoritmo, é utilizada a estrutura de grafos por matriz de adjacencias. Uma matriz de adjacencias é basicamente uma matriz n por n (onde n é o número total de vértices do grafo) com o valor das transações de um vértice à outro vértice, e com um zero onde elas não existem.

Como exemplo, é apresentada a matriz do grafo G representado pela Figura 3:

$$\mathrm{G} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{pmatrix} 0 & 200 & 300 & 0 & 0 \\ 0 & 0 & 0 & 0 & 300 \\ 0 & 0 & 0 & 100 & 0 \\ 50 & 0 & 0 & 0 & 0 \\ 500 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

Através de matriz acima, pode-se ver que o correntista 1 transfere 200 reais para o correntista 2, mas não possui nenhuma transferência para o correntista 4, por exemplo.

4 Solução

O algoritmo é apresentado na Seção 4.1. Na Seção 4.3, são apresentados métodos comparativos para retornar a economia obtida através dos algoritmos das Seções 4.1 e 4.2. A Seção 4.3 apresenta métodos para comparação do fluxo antes e depois da minimização e a economia resultante. Na Seção 4.4, é detalhada uma análise de complexidade do algoritmo da Seção 4.1. Por fim, na Seção 4.5, são apresentados os resultados para os casos de testes propostos na disciplina de Algoritmos e Programação III.

4.1 Algoritmo

O Algoritmo Simplificar Transferências aqui detalhado requer que não exista nenhum vértice com aresta para si mesmo. Pela natureza do problema, um correntista realizar uma transferência para si mesmo não faz nenhum sentido. De qualquer forma, tais arestas poderiam ser facilmente removidas com a adição de um novo teste condicional entre as linhas 35 e 36. Porém, para evitar um aumento desnecessário do pseudocódigo, o algoritmo é apresentado sem esse teste.

Como função, não recebe nenhum parâmetro nem devolve alguma informação. Seu único propósito é o de alterar a estrutura de dados.

Primeiramente, o algoritmo marca todos os vértices como "brancos". Após a marcação, ele começa a percorrer cada vértice do grafo. Para cada vértice, ele marca-o como "cinza"(ou seja, "visitando") e depois verifica todos os vértices do grafo para identificar quais são adjacentes, ou seja, receptores de uma aresta do vértice. A partir daí, as diversas condições são testadas.

É importante enfatizar que o algoritmo testa a possibilidade de que haja um ciclo. A linha 23 do algorimo apresenta o condicional que verifica se *matriz[j][k]* não é zero e k é "cinza", ou seja, o que está sendo visitado (nesse caso, $k = i$). Sendo assim, temos um ciclo onde há uma aresta de i para j e de j para i. Para resolver o conflito ele mantém apenas a aresta de maior peso, com o valor subtraído pelo peso da aresta a ser removida. Na eventualidade de que ambas as arestas possuam o mesmo peso, as duas são removidas.

Além disso, sozinho, o Algoritmo Simplificar Transferências não minimiza a estrutura de grafos completa. Ele apenas reduz o fluxo. Porém, para que a verdadeira minimização aconteça, é necessário um outro algoritmo, o qual será apresentado na Seção 4.2.

Algoritmo 1: Simplificar Transferências

```
inicio
    marcar todos os vértices como branco
    para i ← 0 até número de vértices
    faça
        marcar i como cinza
        para j ← 0 até número de vértices
        faça
            se matriz[i][j] ≠ 0
            então
                para k ← 0 até número de vértices
                faça
                    se matriz[j][k] ≠ 0 & k ≠ cinza
                    então
                        se matriz[i][j] ≥ matriz[j][k]
                        então
                            matriz[i][j] ← matriz[i][j] − matriz[j][k]
                            matriz[i][k] ← matriz[i][k] + matriz[j][k]
                            matriz[j][k] ← 0
                        senão
                            matriz[j][k] ← matriz[j][k] − matriz[i][j]
                            matriz[i][k] ← matriz[i][k] + matriz[i][j]
                            matriz[i][j] ← 0
                    senão se matriz[j][k] ≠ 0 & k = cinza
                    então
                        se matriz[i][j] > matriz[j][i]
                        então
                            matriz[i][j] ← matriz[i][j] − matriz[j][i]
                            matriz[j][i] ← 0
                        senão se matriz[j][i] > matriz[i][j]
                        então
                            matriz[j][i] ← matriz[j][i] − matriz[i][j]
                            matriz[i][j] ← 0
                        senão
                            matriz[i][j] ← 0
                            matriz[j][i] ← 0
        marcar i como preto
```

4.2 Minimização

Como pode-se perceber, o Algoritmo Simplificar Transferências altera a estrutura de dados a fim de reduzir o fluxo desnecessário de dinheiro entre os correntistas. Porém, após quem o procedimento do algoritmo é realizado, muitas outras simplificações podem ainda ser feitas. Quando as simplificações são realizadas até que um grafo não possa mais ser simplificado, é dito que o grafo foi minimizado. A não ser que se trabalhe com um grafo realmente pequeno, para minimizar um grafo, não basta simplificar uma só vez: é necessário simplificar o máximo possível. Para isso, é apresentado nesta seção, o Algoritmo Minimizar, que se utiliza do Algoritmo Simplificar Transferências, "chamando-o"em um laço que só irá parar quando não houver mais alterações a serem feitas no grafo. Além de utilizar o Algoritmo Simplificar Transferências, o Algoritmo Minimizar também recorre ao Algoritmo auxiliar Ajustes Sobrando o qual retorna verdadeiro quando houver mais ajustes que possam ser feitos no grafo, ou falso para quando esse não for o caso. Enquanto o Algoritmo Ajustes Sobrando retornar Verdadeiro, o Algoritmo Minimizar irá executar o Algoritmo Simplificar Transferência.

Primeiramente, é apresentado o Algoritmo auxiliar Ajustes Sobrando:

Algoritmo 2: Ajustes Sobrando

```
Saída: Valor booleano
inicio
    para i ← 0 até número de vértices
    faça
        para j ← 0 até número de vértices
        faça
            se matriz[i][j] ≠ 0
            então
                para k ← 0 até número de vértices
                faça
                    se matriz[j][k] ≠ 0
                    então
                        retorna Verdadeiro
        retorna Falso
```

E então, com os algoritmos utilizados para a realização da minimização tendo sido devidamente detalhados, é apresentado o Algoritmo Minimizar:

Algoritmo 3: Minimizar

```
inicio
    enquanto Ajustes Sobrando
    faça
        Simplificar Transferências
```

4.3 Método Comparativo

O Algoritmo Diferença apresenta uma breve solução para retornar a diferença entre a quantidade de dinheiro utilizada originalmente e a quantidade utilizada após a simplificação. Como pode-se ver, o próprio Algoritmo Diferença se utiliza do Algoritmo Minimizar durante seu processo.

Algoritmo 4: Diferença

```
Saída: Quantidade Reduzida de Dinheiro Movimentado
inicio
    antes ← 0
    para i ← 0 até número de vértices
    faça
        para j ← 0 até número de vértices
        faça
            antes ←antes+matriz[i][j]
    Minimizar Transferências
    depois ← 0
    para i ← 0 até número de vértices
    faça
        para j ← 0 até número de vértices
        faça
            depois ←depois+matriz[i][j]
    retorna antes - depois
```

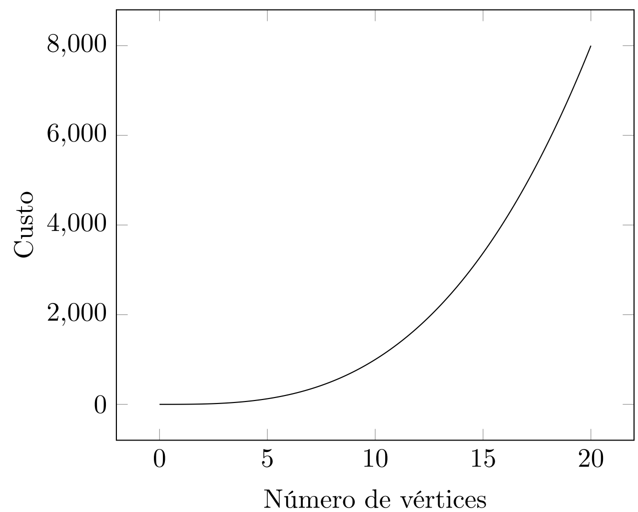
O Algoritmo Economia, por sua vez, se utiliza do Algoritmo Diferença (o qual retorna a diferença da quantia total de dinheiro movimentado) para retornar 1% daquele valor, a fim de obter a quantidade economizada no pagamento de impostos (1% sobre cada transferência).

Algoritmo 5: Economia

```
Saída: Quantidade Economizada de Impostos
inicio
    retorna Diferença ÷100
```

4.4 Análise de Complexidade

Realizando uma análise da complexidade do algoritmo apresentado na Seção 4.1, pode-se notar que ele é constituído de três laços *para/até* que são executados n vezes, sendo n o número de vértices do grafo. Dessa forma, pode-se concluir que a complexidade do algoritmo está em $\mathcal{O}(n^3)$. No gráfico abaixo, fica claro o quanto cresce o custo computacional de acordo com o número de vértices n que pertencem ao grafo:



4.5 Casos de Teste

São apresentados, na Tabela 1, os resultados dos casos de teste, detalhando a quantidade total de dinheiro antes e depois da Minimização. Além da quantidade economizada pela redução de impostos adquirida através da minimização.

Tabela 1: Resultados dos casos de teste

Caso de Teste	Inicial	Final	Economia
1	684106	151996	5321.1
2	568965	161205	4077.6
3	415632	167800	2478.32
4	360943	138389	2225.54
5	262339	116130	1462.09
6	439454	147321	2921.33
7	197202	93899	1033.03
8	603655	191202	4124.53
9	438886	159721	2791.65
10	368191	125324	2428.67