Trabalho Prático 1
Entregando Lanches
Algoritmos e Estruturas de Dados III - 2017/01

Pedro Otávio Machado Ribeiro

21/05/2017

1 Introdução

Neste trabalho, temos como contexto a empresa de lanches de Rada que realiza as entregas por meio de ciclistas e contratou Gilmar para trabalhar no setor de logística. Num certo momento, a nova prefeita Dora decidiu diminuir a pista de ciclistas para aumentar o tráfego de véiculos automotores, atrapalhando o negócio de lanches de Rada. Dessa forma, dado a posição das franquias de Rada na cidade e dos clientes, as ruas que possuem ciclofaixas e suas respectivas capacidades de tráfego de ciclista e que cada franquia possui uma quantidade arbitráriamente grande de ciclistas, o nosso objetivo é ajudar Gilmar a encontrar a quantidade máxima de ciclistas que podem sair das franquias de realizar as entregas de forma segura. Para que desenvolver a solução, segue abaixo alguns conceitos necessários.

1.1 Grafo

Um grafo direcionado ponderado G é uma estrutura matemática definida por um par ordenado $G = (V, E)$, onde V é um conjunto finito de vértices e E uma relação binária em V definida por (u, v) onde $u, v \in V$ que representa uma aresta que sai de u e vai para v. Dizemos que v é adjacente à u. Além disso, tome ρ a função de custo da aresta definida como segue:

$$\begin{aligned} \rho \colon V \times V &\to \mathbb{R} \\ (u, v) &\mapsto c \end{aligned}$$

Neste trabalho, utilizaremos apenas custos $c \in \mathbb{Z}$.

1.1.1 BFS e DFS

O BFS(Breadth-first search) é um algoritmo de busca em grafos que prioriza a visita em todos os vértices adjacentes. Dessa forma, sempre que um vértice é visitado, está foi a primeira, e assim temos o menor caminho até ele a partir de uma dada raiz. A visita dos vértices no BFS segue a ordem **FIFO** e é implementado com uma fila.

O DFS(Deapth-first search) é outro algoritmo de busca. Diferente do BFS, o DFS prioriza a busca pelos vértices mais recentes descobertos, de forma que a busca seja a mais profunda possível. Ao chegar em um vértice que não possui mais arestas para visitar, ele volta no caminho para visitar as arestas do vértice anterior que ainda não foram visitadas. A visita dos vértices no DFS segue a ordem **LIFO** e é implementado com uma pilha.

2 Metodologia

Para solucionar o problema, podemos modelar a situação dada com um grafo $G((V, E), \rho, s, t)$ direcionado e ponderado, onde as interseções são os vértices em V, as ciclofaixas as arestas em E e a capacidade de tráfego das ciclofaixas a função de custo ρ de cada aresta. Os parâmetros s e t serão explicados posteriormente.

Para definir a quantidade máxima de ciclistas que podem sair das franquias com segurança, o conceito de fluxo máximo em grafos deve ser utilizado. Dado que existe vários algoritmos que realizam este procedimento, o autor deste trabalho optou por implementar o algoritmo *Dinic*.

2.1 Fluxo Máximo

O problema de fluxo máximo consistem em dado um grafo ponderado $G(V, E)$, um vértice fonte s e um sumidouro t, qual a quantidade máxima de, neste caso ciclistas, que conseguem chegar ao sumidouro t a partir da fonte s.

2.2 Algoritmo de *Dinic*

O algoritmo de *Dinic* é um algoritmo polinomial que computa o fluxo máximo em uma rede de fluxos. Para tal, o algoritmo utiliza o conceitos de Caminho Aumentante(*Augmenting Path*), Grafo de Nível(*Level Graph*) e Bloqueio de Fluxo(*Blocking Flow*).

Este algoritmo funciona para uma fonte(*source*) e um sumidouro(*sink*). Porém, neste trabalho temos várias fontes e vários sumidouros, que são as franquias e clientes, respectivamente. Para contornar este problema, criamos uma *super-source* e a ligamos a cada uma das franquias e ligamos todos os clientes a um *super-sink*. Todas estas novas arestas possuiram capacidade infinita, dado que a quantidade de ciclistas é muito grande nas franquias e que não queremos saber a soma dos fluxos de cada um dos clientes.

Para as seguintes definições, seja o grafo $G((V, E), \rho, s, t)$, em que $\rho(u, v)$ é o custo e $f(u, v)$ o fluxo das arestas.

2.2.1 Augmenting Path

Para entender o *Caminho Aumentante*, primeiro precisamos da definição de capacidade residual e de grafo residual.

Definição 1. A capacidade residual é um mapeamento $c_f \colon V \times V \to \mathbb{Z}^+$ definido como:
Se $(u, v) \in E$, então $\rho_f(u, v) = \rho(u, v) - f(u, v)$ e $\rho_f(v, u) = f(u, v)$.
Caso contrário, $\rho_f(u, v) = 0$.

Em outras palavras, a capacidade residual é a capacidade atual de uma aresta menos o fluxo que por ela passa. A aresta reversa da definição representa essa subtração, cancelando parte do fluxo que passava.

Definição 2. Um grafo residual é o grafo $G_f((V, E_f), \rho_f|_{E_f}, s, t)$ onde E_f é definido da forma $E_f = \{(u, v) \in V \times V \colon \rho_f(u, v) > 0\}$.

Ou seja, um grafo residual é um conjunto de arestas em que suas capacidades residuais são maiores que 0, que por sua vez ainda possibilita a passagem de fluxo.

Com as definições acima, temos a seguinte definição de caminho aumentado.

Definição 3. Um caminho aumentado é um caminho $(u_1, u_2, \ldots, u_k)$ em um grafo residual G_f, onde $u_1 = s$, $u_k = t$ e $\rho(u_i, u_{i+1}) > 0, \forall i \in [1, k-1]$.

Em outras palavras, um caminho aumentado é todo caminho em grafo residual que todas as arestas possuem capacidades maiores que 0.

2.2.2 Level Graph

2.2.3 Blocking Flow

3 Complexidade

4 Experimentos

5 Análise de Resultado

6 Conclusão

7 Referências