UNIVERSIDADE FEDERAL RURAL DO SEMI-ÁRIDO

VITOR OLIVEIRA ROPKE

**PROBLEMA DO CAMINHO MÍNIMO E SOLUÇÃO A PARTIR DO ALGORITMO DE FLOYD**

MOSSORÓ/RN

2019

# 1 INTRODUÇÃO

Durante um planejamento de uma viagem, deve-se ter conhecimento acerca de diversas variáveis (distâncias, velocidades máximas, condição das vias) para executá-la no menor tempo e/ou caminho possível. O GPS utiliza um algoritmo baseado no problema do caminho mínimo para calcular esses parâmetros.

Graças aos algoritmos que calculam o caminho mínimo é possível computar não só os menores trajetos em mapas, mas também em diversos casos onde é necessário ter um menor trajeto para evitar gastos desnecessários como passagens de fios e/ou cabos de telefones ou elétricos, tubos de água e esgoto, fibra óptica, e vários outros. Esse algoritmo permite uma grande economia para empresas de água, luz, internet, telefone e transportadoras.

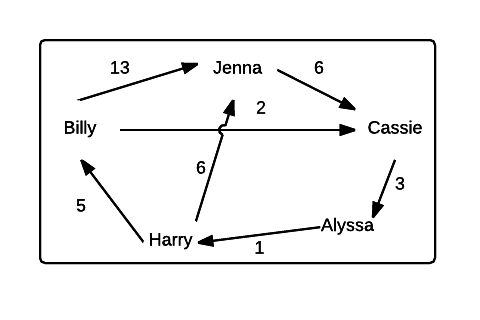
# 2. Desenvolvimento

Dentre os vários algoritmos de caminho mínimo, existe o Floyd-Warshall que atua em grafos orientados e valorados. “Foi publicado por Robert Floyd em 1962” (WIKIPÉDIA, 2019) e possui o algoritmo com três loops foi proposto por Peter Ingerman no ano 1962 que é a versão final.

Este algoritmo serve para “encontrar uma expressão regular que denota a linguagem comum aceita por um autômato finito (algoritmo de Kleene), inversão de matrizes reais (algoritmo de Gauss – Jordan), caminhos mais largos / caminhos de largura de banda máxima (inverso do caminho mínimo), cálculo rápido de redes Pathfinder, computar a semelhança entre gráficos e encontrar o caminho mínimo” (WIKIPÉDIA, 2019).

O algoritmo de Floyd calcula o menor caminho entre todos os pares de vértices de um grafo. Primeiro o algoritmo escolhe um vértice diferente por vez e compara vários pares deste vértice em busca do menor caminho. Por exemplo, existe um par ‘a’ e ‘c’ que possui um peso 3 e um par ‘a’ e ‘b’ que possui peso 1 e outro par ‘b’ e ‘c’ também com peso 1. O caminho ‘a’, ‘b’ e ‘c’ é menor que ‘a’ e ‘c’. Então o algoritmo armazena o peso atual entre um vértice ‘a’ e ‘c’ e verifica se outra combinação como ‘a’ e ‘b’, ‘b’ e ‘c’, entre outras, possui um peso menor que a atual.

Fonte: Brilliant (2019)



Usando a imagem acima como exemplo, temos as arestas como sendo as distâncias entre as casas e os vértices são as casas das pessoas. O menor caminho de Billy para Jenna é passando por Cassie, Alyssa e Harry. O algoritmo calcula no primeiro laço que o menor caminho é a aresta com peso 13 e ao longo das iterações, ele acha um valor de Billy para Jenna menor que 13 e então atualiza o valor para 12 que é a soma das arestas 2, 3, 1 e 6.

for k from 1 to |V|:

for i from 1 to |V|:

for j from 1 to |V|:

if M[i][j] > M[i][k] + M[k][j]:

M[i][j] = M[i][k] + M[k][j]

Fonte: CodCad (2016)

M é a matriz de adjacência e |V| é a dimensão dela.

# 3. CONCLUSÃO

O principal problema encontrado foi durante a comparação de valores atuais com valores menores na linha “if aresta[i][j] > aresta[i][k] + aresta[k][j]”. O valor na aresta[i][j] deveria ser infinito exceto se estivesse na diagonal principal (quando i = j). Portanto foi necessário fazer uma adaptação temporária, colocando um número extremamente grande para ser possível fazer esta comparação de forma correta, caso contrário, todos os valores ficariam errados na matriz final.

# REFERÊNCIAS

BYRNE, Michael. O Simples e Elegante Algoritmo Que Torna o Google Maps Possível: É isso que possibilita o ato de traçar rotas: engenhosidade o bastante para ver em meio a todo o caos. **MOTHERBOARD TECH BY VICE**, [*S. l.*], 25 nov. 2015. Disponível em: https://www.vice.com/pt\_br/article/z4zaj9/o-simples-e-elegante-algoritmo-que-torna-o-google-maps-possivel. Acesso em: 15 nov. 2019.

CHUMBLEY, Alex et al. Floyd-Warshall Algorithm. [S. l.]. Disponível em: https://brilliant.org/wiki/floyd-warshall-algorithm/. Acesso em: 24 nov. 2019.

FLOYD–WARSHALL algorithm. [S. l.], 2 set. 2019. Disponível em: https://en.wikipedia.org/wiki/Floyd–Warshall\_algorithm. Acesso em: 24 nov. 2019.

SIAUDZIONIS, Lucca. **Menor Caminho II**. [S. l.], 2016. Disponível em: http://www.codcad.com/lesson/51. Acesso em: 24 nov. 2019.