Relatório 2º Projeto de ASA 2019/2020

Alunos: Maria Sbrancia 78631
João Lopes 90732 **Grupo:** tp012

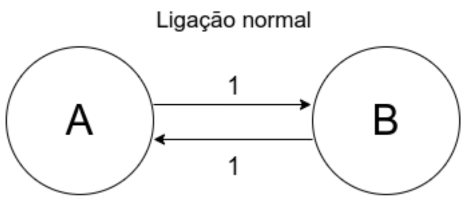
Descrição do problema e da solução

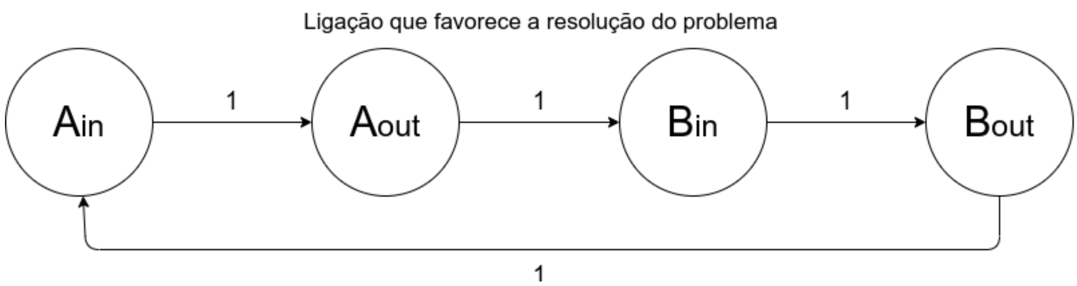
O projeto incide em maximizar o fluxo de cidadãos C que pretendem deslocar-se a supermercados S, sem correr o risco de se cruzarem com outro cidadão. A cidade em questão, possui ruas bidirecionais numeradas de 1 a M de Norte a Sul (mAvenidas) e de 1 a N de Este a Oeste (nRuas). Todo o caminho percorrido pelo cidadão até ao supermercado fica impossibilitado de ser usado por outro cidadão.

O algoritmo escolhido para maximizar o fluxo é o Ford Fulkerson que, enquanto houver caminhos de aumento (encontrados com recurso a uma BFS), aumenta o fluxo máximo que a rede deixa passar. Para a sua implementação, foram definidas uma sink e uma source universais, onde a source se encontra ligada aos cidadãos e a sink aos supermercados.

Adicionalmente, os número total de vértices mAvenidas * nRuas se encontra duplicado de modo a simular que os vértices possuem uma capacidade de 1 e não podem ser utilizados por outro caminho.

Isto é conseguido possuindo dois vetores GraphIn e GraphOut onde o correspondente à ligação de A ->B , onde A e B representam cruzamentos é dado por:





Assim, só se pode atrevessar de A até B com fluxo máximo de 1, ficando assim a capacidade de um nó ‘simulada’. Para formar uma ligacao bidirecional é necessario as partes in e out de um nó estarem ligadas: uma saida(out) liga-se a uma entrada(i). Cada nó deve ser dividido, então em duas partes(in e out) que devem estar ligadas com uma aresta de in para out com capacidade 1.

No caso da source e da sink universais as suas ligacoes não devem ser necessariamente 1 visto que pode ser necesário fazer passar mais que um cliente na entrada(source) e na saida(sink) do novo grafo.

Seja o grafo dado G = < V , E > em que V são os cruzamentos e E as suas respectivas ligações que são as ruas ou avenidas.

Antes de ler qualquer input o algoritmo cria um nó source e sink universais. Estando estes criados, começamos a ligar os cruzamentos da cidade. Posteriormente, por cada supermercado e consumidor é feita a ligação do nó que o representa à source e sink, respetivamente.

Estando o grafo construido é aplicado o algoritmo Ford-Fulkerson com origem na source universal. O resultado da aplicação desse algoritmo é o fluxo máximo de cidadãos que pode circular numa cidade onde todos os cruzamentos e ruas só permitem fluxo de valor 1, respondendo ao que se pretende descobrir neste problema.

Os nós são uma estrutura com uma lista de outros nós aos quais está ligado, cada uma dessas ligações uma outra estrutura que guarda o fluxo e o destino.

Descrição do problema e da solução

Dado G = < V , E >, descrito anteriormente,

```
buildGraph()
    for  each V
        for n in V.adj
            connect(V, n)
```

O(|V| + |E|)

```
processInput()
    for supermercado in input.Supermercados
        connect(universalSink, supermercado)

    for consumidor in input.Consumidores
        connect(universalSource, consumidor)
```

O(|V| + |E|)

```
Ford-Fulkerson()
    maxFlow = 0
    path = new Path()
    while(bfs(universalSource, universalSink, path))
        addPathToFlow()
    return  maxFlow
```

O(|E|*f)

```
bfs(source, dest)
    queue = new Queue()
    queue.add(source)

    while(not queue.empty())
        node  node = queue.front()

        for n in node.vizinhos
            if(n.visited == false and fluxo(node, n) > 0)
                queue.add(n)
                path.add(n)
                n.visited =  true
    return  dest.visited
```

O(|V| + |E|)

Complexidade total: O(|E|*f) ,f é o fluxo máximo

Análise experimental dos resultados

O algoritmo foi executado com o numero de cidadãos e supermercados sempre proporcional ao tamanho da grelha variando apenas a raiz quadrada do tamanho da mesma em incrementos de 25. O tempo começa a aumentar de forma exponencial conforme o aumento do tamanho da grelha.

É importante reparar que o tempo está em função da raiz quadrada do numero de cruzamentos. Sendo assim, o resultado parece estar de acordo com o esperado: |E| aumenta e, sendo que f(fluxo máximo) também aumenta por estarmos a aumentar proporcionalmente o numero de cidadãos e supermercados, |E|*f vai escalar da forma descrita no gráfico obtido.