Relatório 2º projeto ASA 2019/2020

Grupo: tp020

Aluno: João Miguel Pipa Ferreira Caldeira (93729)

Descrição do Problema e da Solução

O problema proposto consistem em projetar um sistema que permita aos cidadãos de Manhattan deslocar-se a um supermercado sem se cruzarem com nenhum outro cidadão sendo fornecidas as localizações dos cidadãos e dos supermercados e sabendo que as ruas têm um arranjo em quadriculado absolutamente regular e que em todas as ruas se circula em ambos os sentidos.

Assim sendo o sistema deve determinar o número máximo de cidadãos que pode deslocar-se a um supermercado, sem correr o risco de se encontrar com outro cidadão, numa rua, avenida ou cruzamento, inicial, intermédio ou final do seu percurso.

A solução baseia-se na transformação do problema proposto num problema de fluxo máximo em grafos com capacidade nos vértices. Assim sendo a cidade é representada por um grafo em que os vértices correspondem às esquinas da cidade onde poderão residir cidadãos e/ou existir supermercados, sendo que os arcos são definidos pela estrutura quadriculada e absolutamente regular da cidade. Para a resolução deste problema foi consultada a aula teórica 11 para efeito de estudar a lição sobre o problema do emparelhamento máximo bipartido e o algoritmo de Edmonds-Karp que me pareceram uma boa abordagem inicial para a resolução do problema. Em seguida realizei uma pesquisa sobre o problema do fluxo máximo e encontrei numa página da Wikipédia (link) um segmento que falava da resolução do problema de fluxo máximo em grafos com capacidade nos vértices e decidi utilizar uma das soluções apresentadas nessa página: o algoritmo de Dinic pelo facto de que este é uma otimização do algoritmo que tinha inicialmente projetado utilizar, o algoritmo de Edmonds-Karp.

A solução apresentada transforma a cidade num grafo com arcos e vértices de capacidade unitária, em que os vértices representam as esquinas da cidade. O número de vértices é posteriormente duplicado de forma a transformar a capacidade dos vértices numa capacidade de arco, obtendo para cada esquina um vértice de entrada, um vértice de saída e um arco com capacidade unitária exclusivo, que liga o vértice de entrada ao vértice de saída. Por fim adicionam-se dois vértices extra: o “super source” que é o vértice ponto de partida de exploração e que está somente ligado por arcos unitários aos cidadãos; e o “super target” que é o vértice de chegada e que está somente ligado aos mercados por arcos unitários. As capacidades dos arcos são sempre unitárias (= 1) de forma a assegurar que apenas um cidadão frequenta cada arco e cada vértice do grafo (cidade). Assim simplifica-se o problema total a um problema de fluxo máximo de grafos. Após a construção do grafo, aplica-se o algoritmo de Dinic e obtém-se o fluxo máximo que corresponde ao número máximo de cidadãos que podem deslocar-se a um supermercado, sem correr o risco de se

Relatório 2º projeto ASA 2019/2020

Grupo: tp020

Aluno: João Miguel Pipa Ferreira Caldeira (93729)

---

encontrar com outro cidadão, numa rua, avenida ou cruzamento, inicial, intermédio ou final do seu percurso.

Análise Teórica

- A leitura dos dados de entrada está dividida em 3 partes. Na primeira parte usa-se a função *scanf* para obter o número de: avenidas; ruas; mercados e cidadãos da cidade. Em seguida obtemos, numa segunda parte, as coordenadas dos mercados existentes na cidade com um ciclo a depender linearmente de M (número de mercados), Θ(M). Codificam-se as coordenadas num único inteiro correspondente ao seu vértice e cria-se um arco para o "super target" partindo do vértice de saída desse vértice.

  for número de mercados > contador:
  obter coordenadas
  adiciona arco (posição, t,1)

  Por último obtém-se as coordenadas dos cidadãos da cidade com um ciclo a depender linearmente de C (número de cidadãos), Θ(C). Codificam-se as coordenadas num único inteiro correspondente ao seu vértice e cria-se um arco partindo do "super source" para o vértice de entrada do vértice correspondente,

  for número de cidadãos > contador:
  obter coordenadas
  adiciona arco (s, posição,1)

- Analisemos agora a construção do grafo. O grafo é construído com um número de vértices V*, que é igual a 2*(Número de Avenidas * Número de Ruas) + 2, indicação do vértice "super source" (neste caso escolhi o vértice 0), e indicação do vértice "super target" (neste caso escolhi o último vértice, V*-1). Após a inicialização da estrutura de Dinic com os dados anteriores, procede-se à construção dos arcos do grafo com um ciclo a depender do número de ruas da cidade e um ciclo a depender do número de avenidas (que é o dobro das avenidas da cidade, mas para efeitos de complexidade ir-se à usar como referência o número de avenidas original), de forma a percorrer todas as coordenadas possíveis da cidade, e assim sendo a complexidade desta etapa é de Θ(V*). Tendo a cidade um arranjo quadriculado absolutamente regular é possível definir regras para a construção dos arcos em cada posição do grafo. Como defini que a cidade se define no intervalo de vértices [1, V*-2], podemos afirmar que os vértices de entrada serão vértices de número ímpar e nesse caso apenas temos de ligar um arco ao vértice de saída correspondente.

Relatório 2º projeto ASA 2019/2020

Grupo: tp020

Aluno: João Miguel Pipa Ferreira Caldeira (93729)

---

for ruas > contador:
*for 2*avenidas > contador2:*
se for um vértice ímpar:
liga ao vértice de saída
else:
se for um vértice com uma rua abaixo:
adiciona arco
se for um vértice com uma rua acima:
adiciona arco
se for um vértice com uma avenida à direita:
adiciona arco
se for um vértice com uma avenida à esquerda:
adiciona arco

- A etapa final é a aplicação da função *flow* à estrutura de Dinic construída, esta função aplica o algoritmo de Dinic para o cálculo do fluxo máximo no grafo construído. O código da estrutura utilizada bem como do algoritmo de Dinic foi retirado de uma página sobre o algoritmo de Dinic e sua implementação(link). A complexidade desta etapa é resultante da complexidade do algoritmo utilizado, neste caso o algoritmo de Dinic, e, portanto, temos uma complexidade de Θ (V^2 * E) no algoritmo de Dinic o que se traduz numa complexidade na solução utilizada de Θ((2V+2) ^2 * E).

  Legenda:
  V = Número de avenidas * Número de ruas
  V* = 2*(Número de avenidas * Número de ruas) + 2

  Assim sendo podemos concluir que a complexidade global da solução é: Θ((2V+2) ^2 * E).

Relatório 2º projeto ASA 2019/2020

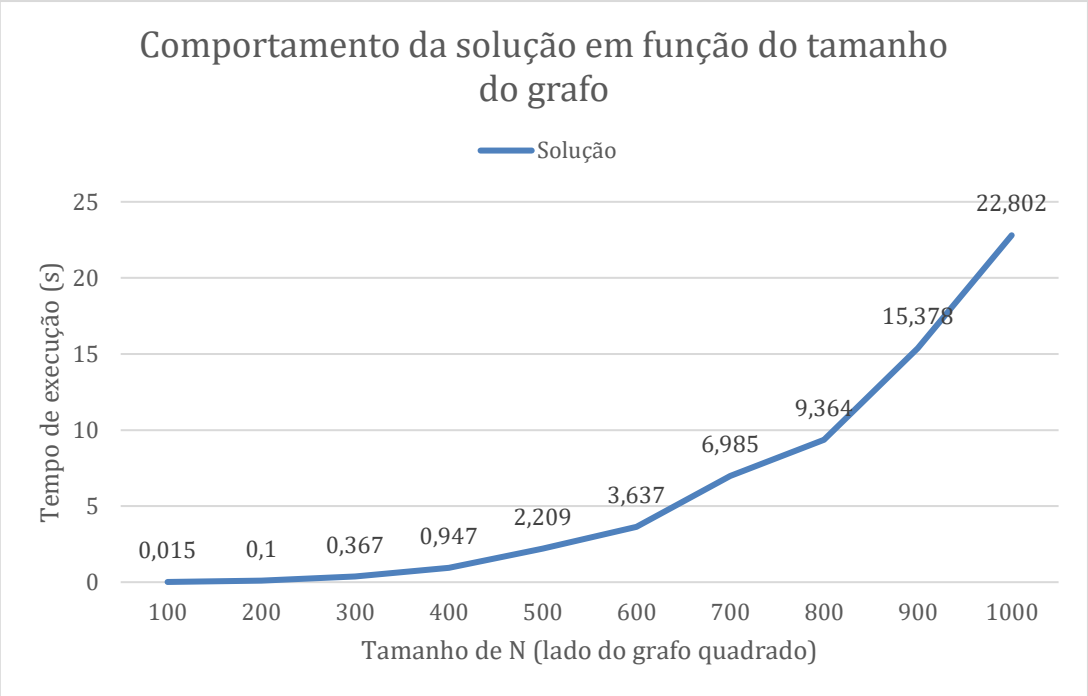
Grupo: tp020

Aluno: João Miguel Pipa Ferreira Caldeira (93729)

Avaliação Experimental dos Resultados

Foram gerados 10 grafos quadrados NxN, com N a variar de 100 a 1000 e com 5% de supermercados e clientes para cada grafo.

Depois de gerados esses 10 grafos, foi testada a solução para cada um dos casos, e com os tempos de execução foi construído o seguinte gráfico:



O gráfico gerado respeita a estimativa de complexidade Θ((2V+2)^2 * E) já que o aumento do tamanho do grafo implica também o aumento do tempo de execução de uma forma exponencial, ou seja o seu declive está em crescimento formando assim uma curva exponencial que era o expectável já que a complexidade apresenta uma componente quadrática ((2V+2)^2).