Projeto e Análise de Algoritmos
Caminho Mínimo

Eriel Bernardo Albino, Lucas Fernandes Gauer e Nicolas Beraldo

Junho de 2019

1 Problemas

O primeiro problema se baseia em uma rede de 10 computadores onde há múltiplas conexões entre eles.

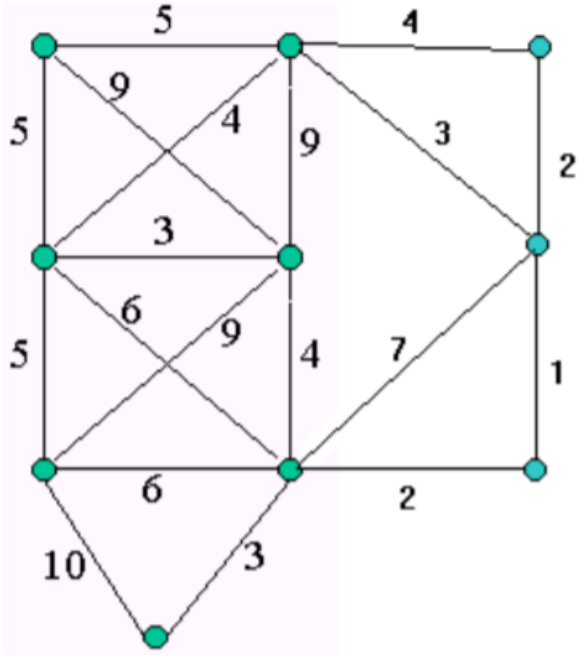

Figura 1: Grafo exemplificando a rede

No grafo da figura 1 podemos visualizar com facilidade a organização da rede e a ligações entre os computadores. Os vértices são os computadores e as arestas o tempo de comunicação entre os computadores.

O segundo problema pode ser visualizado na figura 2. Esse grafo representa um mapa, onde os vértices são cidades e as arestas são as distâncias e as velocidades médias dos trajetos.

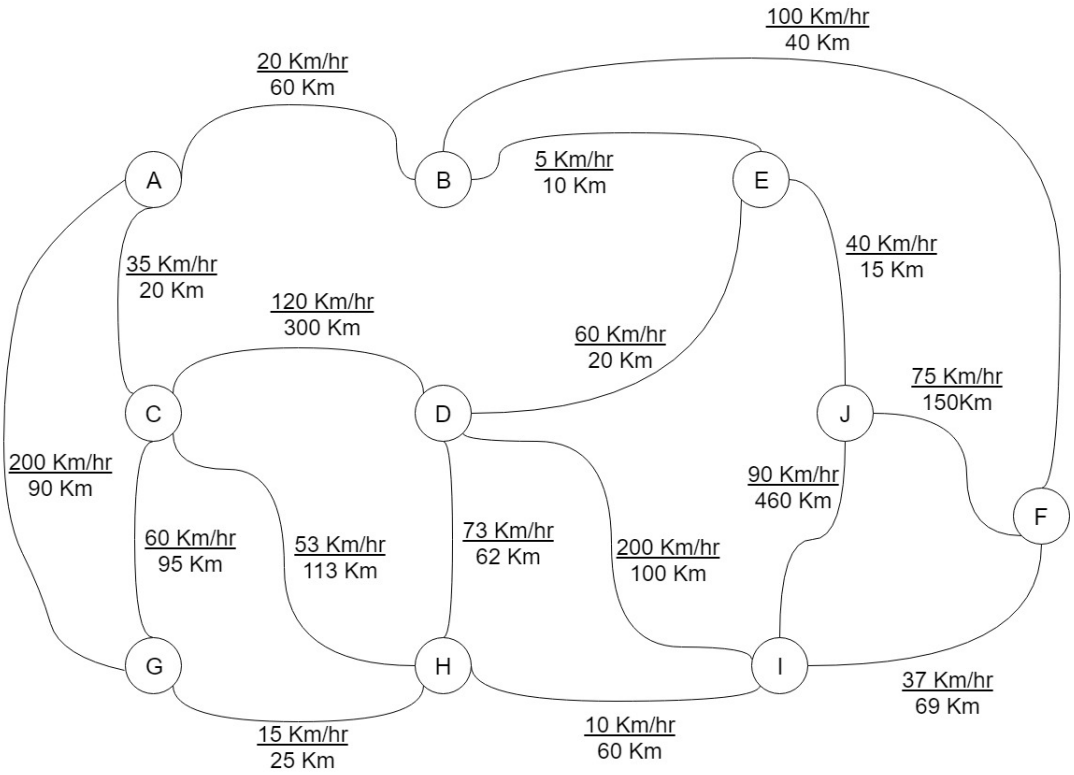

Figura 2: Grafo exemplificando o mapa

Busca-se saber, em ambos os problemas, qual é o melhor trajeto. Para isso temos que ver quais caminhos gastam menos tempo, esses tipos de problemas são conhecidos como problemas de caminhos mínimos.

2 Modelagem

Para iniciar a solução do problema primeiro temos que montar uma matriz de adjacência que represente os grafos corretamente

O grafo representado na figura 1 pode ser visualizado na matriz mostrada na figura 3, onde os vértices estão comentados e os valores para se ir de um vértice ao seu adjacente estão listados no elemento que combina os vértices, a variável "Z" representa "infinito", ou seja, não há adjacência entre os vértices.

```
{0, 1, 2, 3, 4, 5, 6, 7, 8, 9 }
{0, 5, Z, 5, 9, Z, Z, Z, Z, Z }, // 0
{5, 0, 4, 4, 9, 3, Z, Z, Z, Z }, // 1
{Z, 4, 0, Z, Z, 2, Z, Z, Z, Z }, // 2
{5, 4, Z, 0, 3, Z, 5, 6, Z, Z }, // 3
{9, 9, Z, 3, 0, Z, 9, 4, Z, Z }, // 4
{Z, 3, 2, Z, Z, 0, Z, 7, 1, Z }, // 5
{Z, Z, Z, 5, 9, Z, 0, 6, Z, 10}, // 6
{Z, Z, Z, 6, 4, 7, 6, 0, 2, 3 }, // 7
{Z, Z, Z, Z, Z, 1, Z, 2, 0, Z }, // 8
{Z, Z, Z, Z, Z, Z, 10,3, Z, 0 }  // 9
```

Figura 3: Matriz problema 1

Nas figuras a seguir são representadas as matrizes de adjacência das velocidades e das distâncias entre as cidades.

```
{A,   B,   C,   D,   E,   F,   G,   H,   I,   J }
{0,   20,  35,  0,   0,   0,   200, 0,   0,   0 }, //A
{20,  0,   0,   0,   5,   100, 0,   0,   0,   0 }, //B
{35,  0,   0,   120, 0,   0,   60,  53,  0,   0 }, //C
{0,   0,   120, 0,   60,  0,   0,   73,  200, 0 }, //D
{0,   5,   0,   60,  0,   0,   0,   0,   0,   40}, //E
{0,   100, 0,   0,   0,   0,   0,   0,   37,  75}, //F
{200, 0,   60,  0,   0,   0,   0,   15,  0,   0 }, //G
{0,   0,   53,  73,  0,   0,   15,  0,   10,  0 }, //H
{0,   0,   0,   200, 0,   37,  0,   10,  0,   90}, //I
{0,   0,   0,   0,   40,  75,  0,   0,   90,  0 }  //J
```

Figura 4: Matriz problema 2 das velocidades

```
{A,   B,   C,   D,   E,   F,   G,   H,   I,   J  }
{0,   60,  20,  0,   0,   0,   90,  0,   0,   0  }, //A
{60,  0,   0,   0,   10,  40,  0,   0,   0,   0  }, //B
{20,  0,   0,   300, 0,   0,   95,  113, 0,   0  }, //C
{0,   0,   300, 0,   20,  0,   0,   62,  100, 0  }, //D
{0,   10,  0,   20,  0,   0,   0,   0,   0,   15 }, //E
{0,   40,  0,   0,   0,   0,   0,   0,   69,  150}, //F
{90,  0,   95,  0,   0,   0,   0,   25,  0,   0  }, //G
{0,   0,   113, 62,  0,   0,   25,  0,   60,  0  }, //H
{0,   0,   0,   100, 0,   69,  0,   60,  0,   460}, //I
{0,   0,   0,   0,   15,  150, 0,   0,   460, 0  }  //J
```

Figura 5: Matriz problema 2 das distâncias

3 Código

O código para determinar o caminho mínimo está em anexo ao relatório, nesta parte iremos ressaltar as partes cruciais para a resolução do problema. Foi utilizado o algoritmo de Floyd no código.

No trecho de código a seguir estamos determinando os tempos da matriz de adjacência do problema 2 pois são dadas distâncias e velocidades no problema.

```
for(int i = 0; i < N; ++i) {
    for(int j = 0; j < N; ++j) {
        // caso seja o proprio vertice
        if(i == j)
            original[i][j] = 0;
        // caso nao haja conexao entre os vertices
        else if(m_Vel[i][j] == 0)
            original[i][j] = Z;
        // caso haja uma conexao entre os vertices
        else
            original[i][j] = m_Dist[i][j] / m_Vel[i][j];
    }
}
```

Como é possível observar no código há 3 "if", ou condicionais. Quando os vértices relacionados são os mesmo é atribuído uma valor zero. Se não há valor de velocidade não há conexão entre os vértices, logo é atribuído infinito. Por último, se os casos anteriores não ocorrerem, se faz a divisão da distância pela velocidade para se definir o tempo.

Ao finalizar essa parte teremos a matriz de adjacência dos custos por tempo de cada aresta. No caso do problema 1 a matriz fornecida já é a matriz dos custos.

```
for(int k = 0; k < N; ++k){
    copiar(atual, anterior);
    for(int i = 0; i < N; ++i){
        // caso seja o mesmo vertice
        if(i == k) continue;
        // caso o custo seja infinito
        if(atual[i][k] == Z) continue;

        for(int j = 0; j < N; ++j){
            // caso seja o mesmo vertice
            if(j == k) continue;
            // caso o custo seja infinito
            if(atual[k][j] == Z) continue;

            // compara o caminho atual a um alternativo
            atual[i][j] =
                min(anterior[i][j], anterior[i][k] + anterior[k][j]);
        }
    }
}
```

No trecho de código apresentado acima temos a implementação do algoritmo de Floyd similiar ao pseudocódigo visto em aula. Nesse método, atualizamos a nova matriz usando a matriz anterior de base ao fazer o mínimo entre dois elementos, os quais são escolhidos usando a seguinte lógica: $D_{ij}^{k} = min(D_{ij}^{k-1}, (D_{ik}^{k-1} + D_{kj}^{k-1}))$. Essa relação de mínimo faz a comparação entre os caminhos. Para o caso da primeira iteração os primeiros elementos a serem testados são o de custos de adjacência e um custo de caminho alternativo para o mesmo destino, para o resto dos casos são comparados os custos dos caminhos calculados anteriormente, assim a cada interação teremos uma matriz de custos de vértice a vértice e no fim teremos uma matriz que mostrará o custo dos menores caminhos de todos os vértices para todos os vértices.

4 Resolução

Na figura 6 temos a saída dos melhores caminhos entre os vértices e quais vértices nunca serão usados no problema 1, assim evitando transtornos na implementação da rede. Para o entendimento do problema foi padronizado que os vértices vão de 0 a 9 e são lidos da esquerda para direita de cima para baixo.

O formato de saída é: matriz de custos inicial, matriz de custos final e listagem de arcos não utilizados.

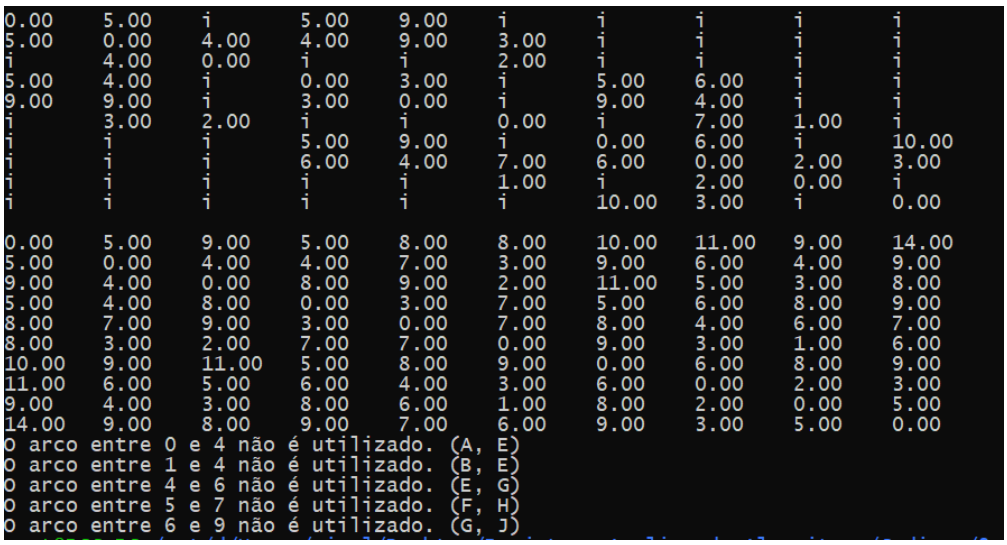

Figura 6: Resultado do código para o problema 1

Na figura 7 temos as respostas para o problema 2

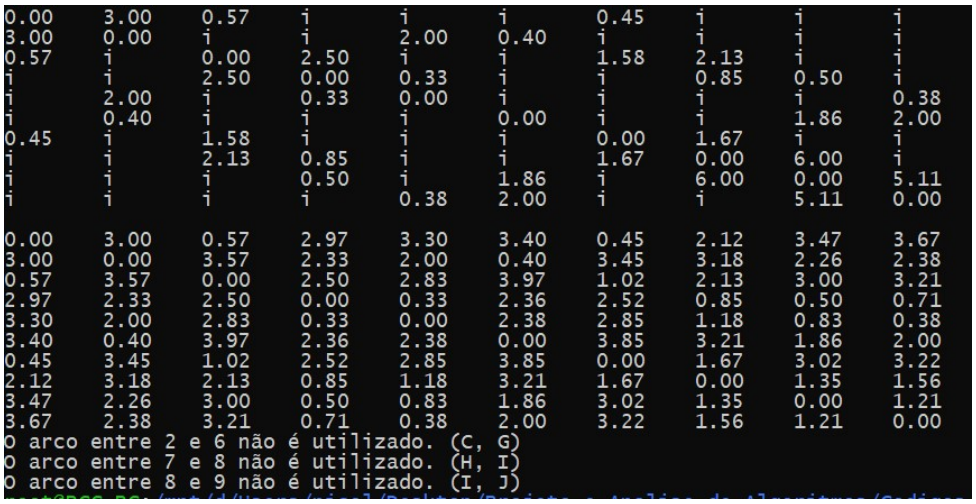

Figura 7: Resultado do código para o problema 2

5 Considerações finais

Ainda que pareça que a utilização de grafos seja trivial em problemas consideravelmente grandes, são necessários um estudo detalhado e um algoritmo de qualidade para obter resultados eficazes. Na área da computação os grafos são utilizados para rotear as trilhas dos circuitos impressos para diminuir o tamanho das placas e consequentemente reduzir o custo de produção.