

ALGORITMOS EM GRAFOS

CAMINHAMENTOS

ALGORITMO DE FLOYD-WARSHALL

Prof. Alexei Machado

PUC MINAS

CIÊNCIA DA COMPUTAÇÃO

Algoritmo de Floyd-Warshall

2

- Calcula o menor caminho entre **todos** os pares de vértices em um digrafo
- Envolve publicações de Robert Floyd (em 1962), Bernard Roy (em 1959) e Stephen Warshall (em 1962)

Algoritmo de Floyd-Warshall

3

- Peter Ingerman (em 1962) deu a forma atual ao algoritmo
- É um exemplo de *programação dinâmica*
 - ▣ *Técnica que utiliza cálculos previamente realizados no cálculo da solução atual.*

Algoritmo de Floyd-Warshall

4

- Premissa: um caminho entre dois vértices, v_i e v_l , passa por um vértice v_k . Logo, o caminho pode ser visto como $c(v_i, v_k) + c(v_k, v_l)$
- Tenta minimizar as partes do caminho

Algoritmo de Floyd-Warshall

5

- Para todo caminho (i, l) , o algoritmo verifica se existe outro menor que passa por um vértice k , ou seja, se $c(i, l)$ é menor que $c(i, k) + c(k, l)$ para cada vértice k do grafo
- Insere um ou mais vértices nos caminhos quando for uma vantagem fazer isso

Estruturas de dados

6

- Matriz de entrada, inicializada com
 - Se $i = l$, $\text{matrizEntrada}(i, l) = 0$
 - Se $i \neq l$ e $(i, l) \in E$, $\text{matrizEntrada}(i, l) = \text{getPeso}(i, l)$
 - Senão, $\text{matrizEntrada}(i, j) = \infty$

Estruturas de dados

7

- Matriz de saída $D_{|V| \times |V|}$, na qual cada célula $d_{i,l}$ contém a distância mínima entre os vértices i e l

Algoritmo de Floyd-Warshall

8

```
void floydWarshall(Peso mat[][]){  
    for (int k = 0; k < n; k++)  
        for (int i = 0; i < n; i++)  
            for (int l = 0; l < n; l++)  
                mat[i][l] = min(mat[i][l], mat[i][k] + mat[k][l])  
}
```

k: intermediário

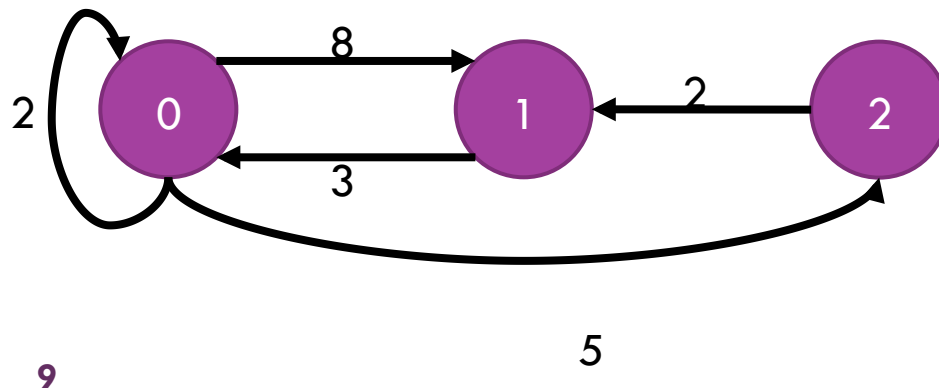
i: origem

l: destino

Algoritmo de Floyd-Warshall

```
void floydWarshall(Peso mat[][]){  
    for (int k = 0; k < n; k++)  
        for (int i = 0; i < n; i++)  
            for (int l = 0; l < n; l++)  
                mat[i][l] = min(mat[i][l], mat[i][k] + mat[k][l])  
}
```

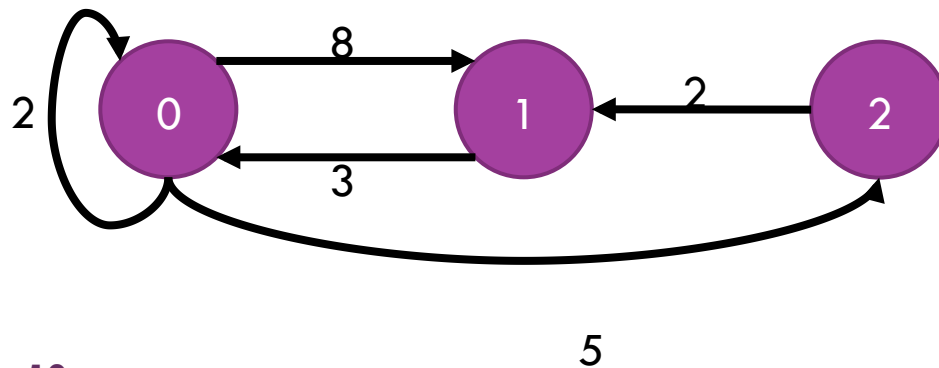
Exemplo



Algoritmo de Floyd-Warshall

```
void floydWarshall(Peso mat[][]){  
    for (int k = 0; k < n; k++)  
        for (int i = 0; i < n; i++)  
            for (int l = 0; l < n; l++)  
                mat[i][l] = min(mat[i][l], mat[i][k] + mat[k][l])  
}
```

Exemplo



Matriz de entrada

0	8	5
3	0	∞
∞	2	0

Algoritmo de Floyd-Warshall

```
void floydWarshall(Peso mat[][]){  
    for (int k = 0; k < n; k++)  
        for (int i = 0; i < n; i++)  
            for (int l = 0; l < n; l++)  
                mat[i][l] = min(mat[i][l], mat[i][k] + mat[k][l])  
}
```

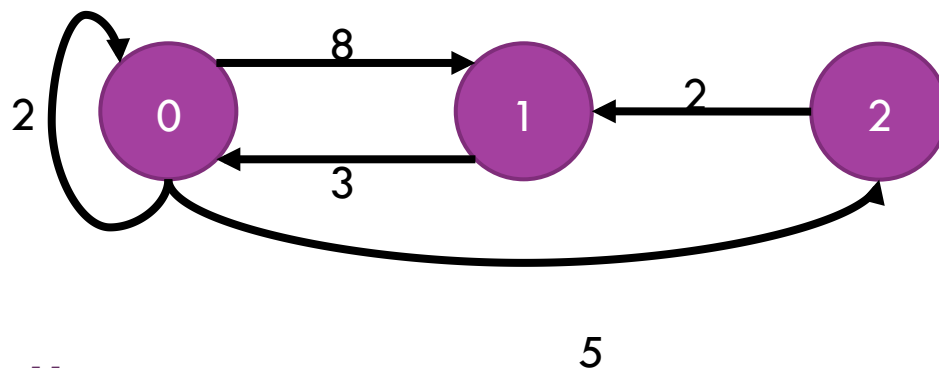
k: intermediário

i: origem

l: destino

k:0 i:0 l:0

Exemplo



Matriz de entrada

0	8	5
3	0	∞
∞	2	0

Algoritmo de Floyd-Warshall

```
void floydWarshall(Peso mat[][]){  
    for (int k = 0; k < n; k++)  
        for (int i = 0; i < n; i++)  
            for (int l = 0; l < n; l++)  
                mat[i][l] = min(mat[i][l], mat[i][k] + mat[k][l])  
}
```

$= \min([0][0]=0, [0][0]=0 + [0][0]=0)$

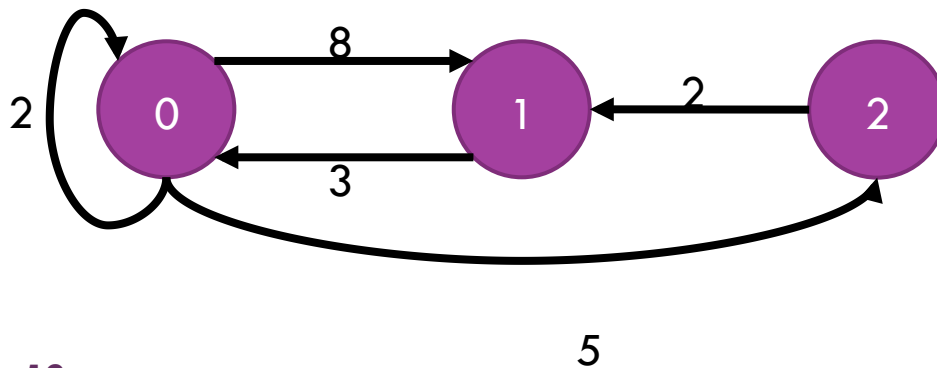
k: intermediário

i: origem

l: destino

k:0 i:0 l:0

Exemplo



v0 como intermediário

0	8	5
3	0	∞
∞	2	0

Algoritmo de Floyd-Warshall

```
void floydWarshall(Peso mat[][]){  
    for (int k = 0; k < n; k++)  
        for (int i = 0; i < n; i++)  
            for (int l = 0; l < n; l++)  
                mat[i][l] = min(mat[i][l], mat[i][k] + mat[k][l])  
}
```

= min([0][1]=8 , [0][0]=0 + [0][1]=8)

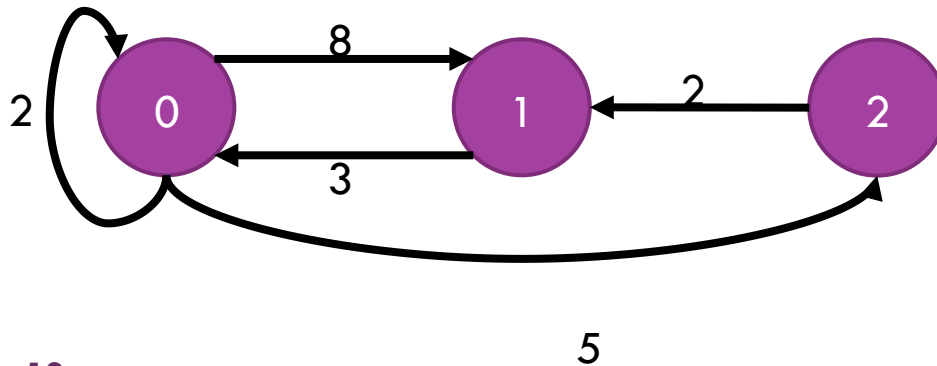
k: intermediário

i: origem

l: destino

k:0 i:0 l:1

Exemplo



v0 como intermediário

0	8	5
3	0	∞
∞	2	0

Algoritmo de Floyd-Warshall

```
void floydWarshall(Peso mat[][]){  
    for (int k = 0; k < n; k++)  
        for (int i = 0; i < n; i++)  
            for (int l = 0; l < n; l++)  
                mat[i][l] = min(mat[i][l], mat[i][k] + mat[k][l])  
}
```

$= \min([0][2]=5, [0][0]=0 + [0][2]=5)$

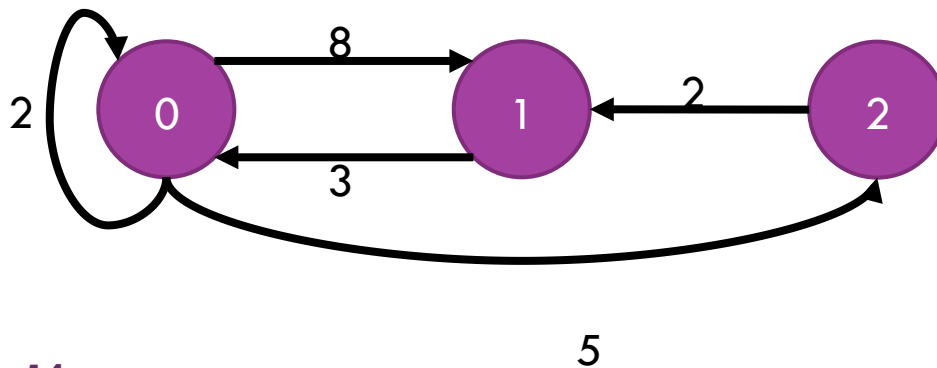
k: intermediário

i: origem

l: destino

k:0 i:0 l:2

Exemplo



v0 como intermediário

0	8	5
3	0	∞
∞	2	0

Algoritmo de Floyd-Warshall

```
void floydWarshall(Peso mat[][]){  
    for (int k = 0; k < n; k++)  
        for (int i = 0; i < n; i++)  
            for (int l = 0; l < n; l++)  
                mat[i][l] = min(mat[i][l], mat[i][k] + mat[k][l])  
}
```

= min([1][0]=3 , [1][0]=3 + [0][0]=0)

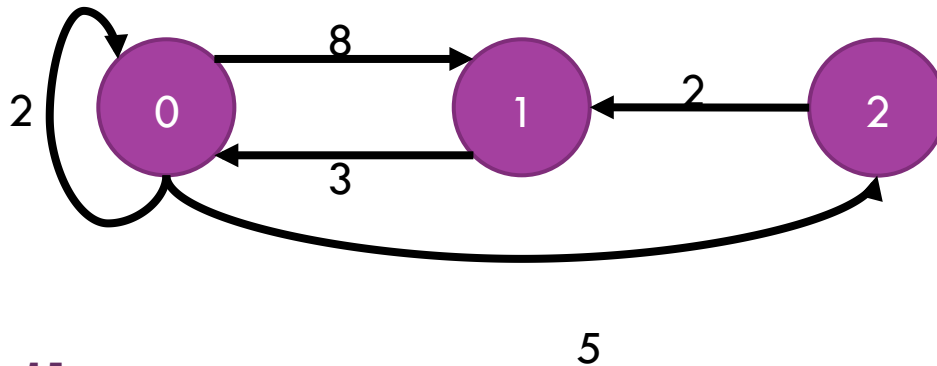
k: intermediário

i: origem

l: destino

k:0 i:1 l:0

Exemplo



v0 como intermediário

0	8	5
3	0	∞
∞	2	0

Algoritmo de Floyd-Warshall

```
void floydWarshall(Peso mat[][]){  
    for (int k = 0; k < n; k++)  
        for (int i = 0; i < n; i++)  
            for (int l = 0; l < n; l++)  
                mat[i][l] = min(mat[i][l], mat[i][k] + mat[k][l])  
}
```

= min([1][1]=0 , [1][0]=3 + [0][1]=8)

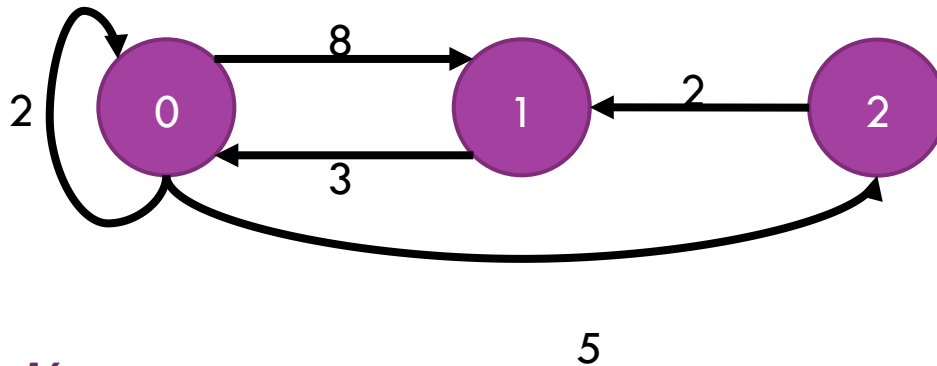
k: intermediário

i: origem

l: destino

k:0 i:1 l:1

Exemplo



v0 como intermediário

0	8	5
3	0	∞
∞	2	0

Algoritmo de Floyd-Warshall

```
void floydWarshall(Peso mat[][]){  
    for (int k = 0; k < n; k++)  
        for (int i = 0; i < n; i++)  
            for (int l = 0; l < n; l++)  
                mat[i][l] = min(mat[i][l], mat[i][k] + mat[k][l])  
}
```

$= \min([1][2]=\infty, [1][0]=3 + [0][2]=5)$

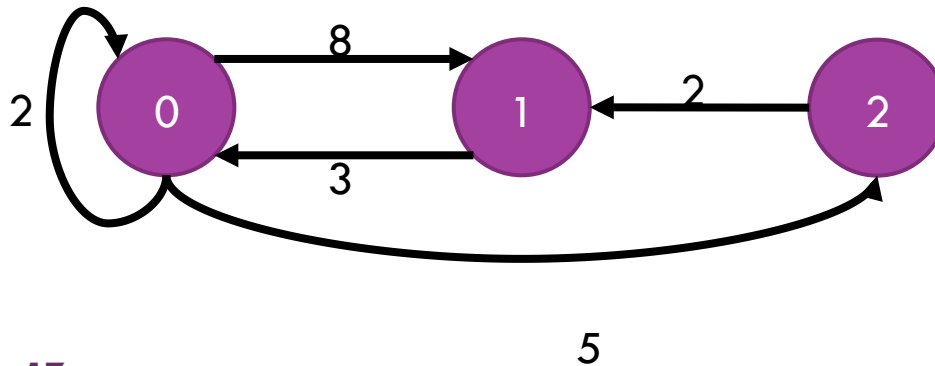
k: intermediário

i: origem

l: destino

k:0 i:1 l:2

Exemplo



v0 como intermediário

0	8	5
3	0	8
∞	2	0

Algoritmo de Floyd-Warshall

```
void floydWarshall(Peso mat[][]){  
    for (int k = 0; k < n; k++)  
        for (int i = 0; i < n; i++)  
            for (int l = 0; l < n; l++)  
                mat[i][l] = min(mat[i][l], mat[i][k] + mat[k][l])  
}
```

$= \min([2][0]=\infty, [2][0]=\infty + [0][0]=0)$

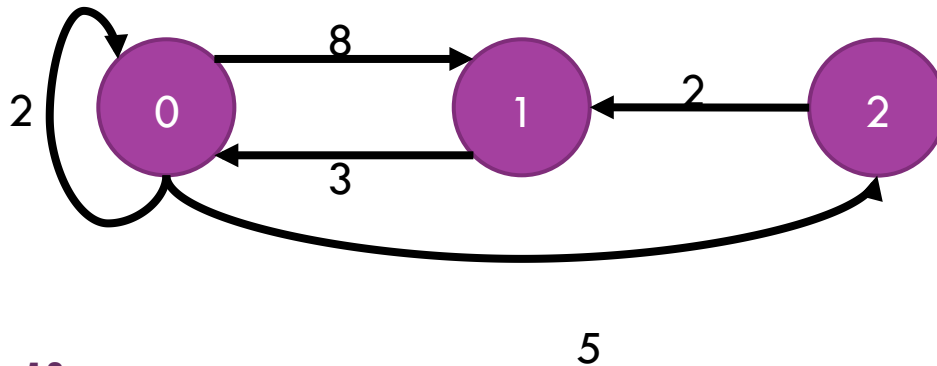
k: intermediário

i: origem

l: destino

k:0 i:2 l:0

Exemplo



v0 como intermediário

0	8	5
3	0	8
∞	2	0

Algoritmo de Floyd-Warshall

```
void floydWarshall(Peso mat[][]){  
    for (int k = 0; k < n; k++)  
        for (int i = 0; i < n; i++)  
            for (int l = 0; l < n; l++)  
                mat[i][l] = min(mat[i][l], mat[i][k] + mat[k][l])  
}
```

$= \min([2][1]=2, [2][0]=\infty + [0][1]=8)$

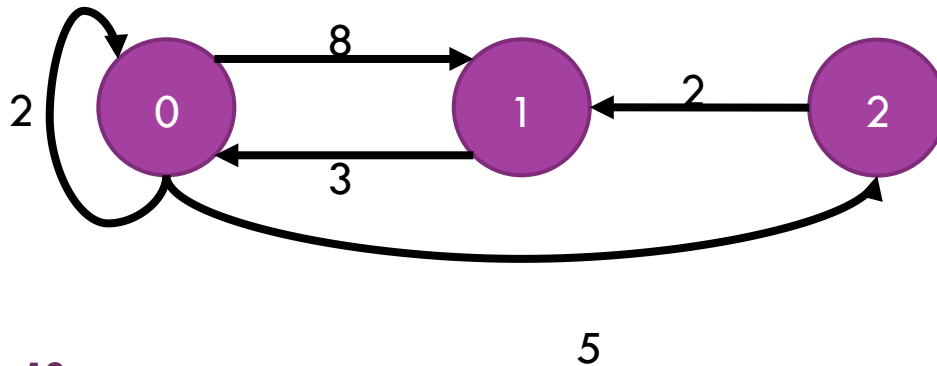
k: intermediário

i: origem

l: destino

k:0 i:2 l:1

Exemplo



v0 como intermediário

0	8	5
3	0	8
∞	2	0

Algoritmo de Floyd-Warshall

```
void floydWarshall(Peso mat[][]){  
    for (int k = 0; k < n; k++)  
        for (int i = 0; i < n; i++)  
            for (int l = 0; l < n; l++)  
                mat[i][l] = min(mat[i][l], mat[i][k] + mat[k][l])  
}
```

= min([2][2]=0 , [2][0]=∞ + [0][2]=5)

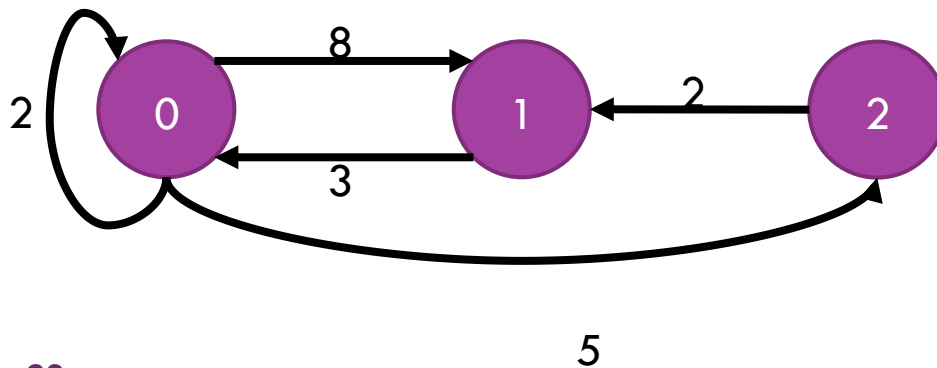
k: intermediário

i: origem

l: destino

k:0 i:2 l:2

Exemplo



v0 como intermediário

0	8	5
3	0	8
∞	2	0

Algoritmo de Floyd-Warshall

```
void floydWarshall(Peso mat[][]){  
    for (int k = 0; k < n; k++)  
        for (int i = 0; i < n; i++)  
            for (int l = 0; l < n; l++)  
                mat[i][l] = min(mat[i][l], mat[i][k] + mat[k][l])  
}
```

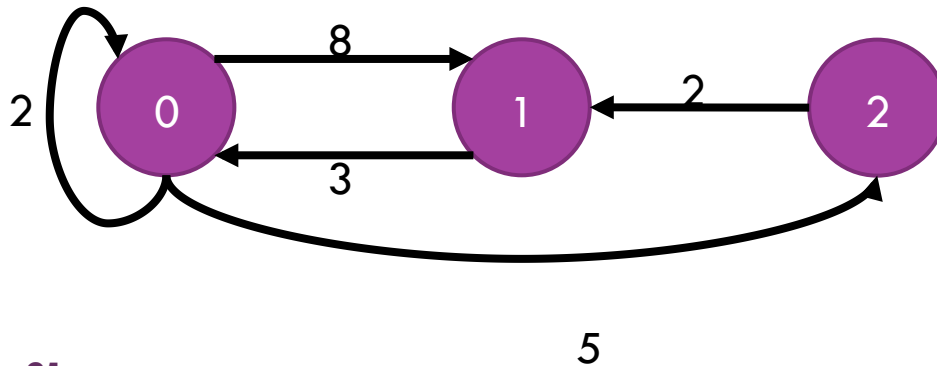
k: intermediário

i: origem

l: destino

k:1 i:2 l:2

Exemplo



v1 como intermediário

0	8	5
3	0	8
5	2	0

Algoritmo de Floyd-Warshall

```
void floydWarshall(Peso mat[][]){  
    for (int k = 0; k < n; k++)  
        for (int i = 0; i < n; i++)  
            for (int l = 0; l < n; l++)  
                mat[i][l] = min(mat[i][l], mat[i][k] + mat[k][l])  
}
```

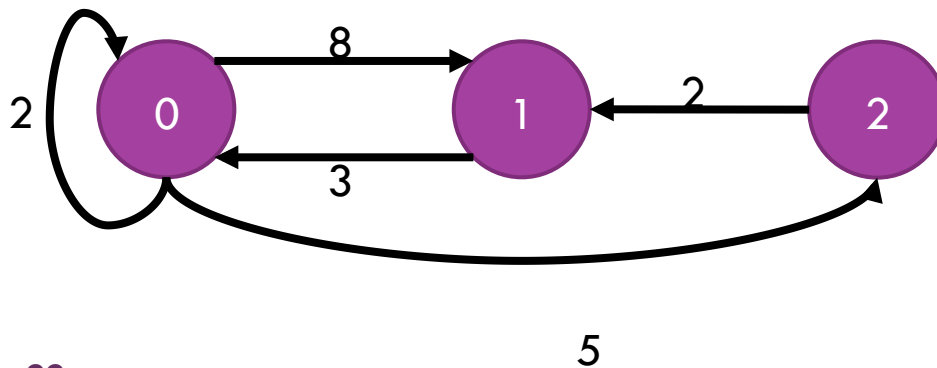
k: intermediário

i: origem

l: destino

k:2 i:2 l:2

Exemplo



v2 como intermediário

0	7	5
3	0	8
5	2	0