**PCS 2215**

**Fundamentos de Engenharia de Computação II**

**Aulas 1-3**

**Grafos**

***Líria Sato***

***Professor Responsável***

*versão: 1.1 (set 2000)*



# Conteúdo

1. Grafos
   1.1 Conceitos principais
   1.2 Caminhos e ciclos
2. Ciclos Hamiltonianos e o problema do caixeiro viajante
3. Algoritmo de caminho mínimo



## 1.1 Conceitos principais

- primeiro trabalho usando teoria de grafos (1736): apresentado por Euler, incluía uma solução para o que é chamado problema das pontes de Königsberg.
- Primeiro texto (1936): desde esta época o interesse em teoria dos grafos tem sido intenso e amplo.
- Aplicabilidade em diversos campos: na ciência da computação, na química, na engenharia elétrica, na economia.



## 1.1 Conceitos principais

Para introduzir o conceito de grafo, vamos apresentar um caso prático.

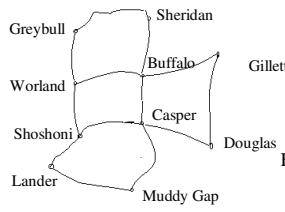



Fig. 1.1: Sistema de auto-estradas de Wyoming

## 1.1 Conceitos principais

Sistema de auto-estradas de Wyoming

- um responsável pela inspeção deve viajar por todas rodovias e documentar em um relatório as condições de cada uma das rodovias: estado das linhas, da sinalização, etc.
- Considerando que o inspetor mora em Greybull, é possível, partindo de Greybull, passar por todas estradas, apenas uma vez, e retornar para Greybull?
- Você pode responder a esta questão?
- O problema pode ser modelado como um grafo.



## 1.1 Conceitos principais

Os pontos são chamados ***vértices*** e as linhas que conectam os vértices são chamados de ***arestas***.

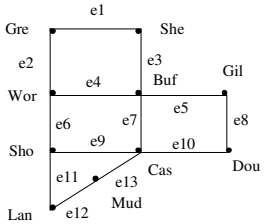



Fig.1. 2: grafo do sistema de auto-estradas(fig. 6.1.2)



## 1.1 Conceitos principais

***Caminho***: se partindo do vértice $v_0$, viajamos, através de uma aresta, para o vértice $v_1$, a seguir através de uma outra aresta para o vértice $v_2$, e assim, até chegarmos no vértice $v_n$, chamamos ao tour completo um ***caminho de $v_0$ a $v_n$***.

O problema do inspetor pode ser reescrito para o modelo de grafos da seguinte forma:

*Existe um caminho do vértice Gre para o vértice Gre que atravessa cada aresta exatamente uma vez?*



## 1.1 Conceitos principais

Podemos mostrar que não existe. Colocando a resposta em termos de grafos: não existe nenhum caminho do vértice Gre para o vértice Gre que atravesse cada aresta exatamente uma vez.

- Consideremos que existe um *caminho*. Tomemos então o vértice Wor. Cada vez que chegamos em Wor, através de uma aresta, devemos seguir para outro vértice, através de outra aresta, ou seja, as arestas que tocam o vértice Wor ocorrem aos pares.

## 1.1 Conceitos principais

- Portanto o número de arestas que tocam Wor deve ser um número par. Desde que o número de arestas que tocam Wor é ímpar temos uma contradição.
- Portanto, não existe um caminho do vértice Gre para o vértice Gre que atravessa cada aresta exatamente uma vez. (maiores detalhes serão discutidos posteriormente)



## 1.1 Conceitos principais

**Grafos**

Um **grafo** (ou um **grafo não orientado**) G consiste de um conjunto V de vértices (ou nós) e um conjunto ***E*** de **arestas** (ou **arcos**) tal que cada aresta $e \in E$ é associada com um par não ordenado de vértices. Se existe uma única aresta ***e*** associada com os vértices ***v*** e ***w***, nós escrevemos ***e=(v,w)*** ou ***e=(w,v).*** Neste contexto, ***(v,w)*** denotam uma aresta entre ***v*** e ***w*** em um grafo não orientado e um par não ordenado.



## 1.1 Conceitos principais

Uma aresta ***e*** em um grafo (orientado ou não orientado) que é associada com um par de vértices ***v*** e ***w*** é dita ser incidente sobre ***v*** e ***w***, e ***v*** e ***w*** são ditos incidentes sobre ***e***, e ***v*** e ***w*** são ditos ser **vértices adjacentes.**

O grafo G da figura 1.2 consiste do conjunto de vértices V={ Gre, She, Wor, Buf, Gil, Sho, Cas, Dou, Lan, Mud}

e o conjunto de arestas

E={$e_1$, $e_2$, ....,$e_{13}$}



## 1.1 Conceitos principais

- aresta $e_1$ é associada ao par de vértices não ordenado {Gre,She}
- aresta $e_{10}$ é associada ao par de vértices não ordenado {Cas,Dou}
- aresta $e_1$ é denotada por (Gre,She) ou (She,Gre)
- aresta $e_{10}$ é denotada por (Cas,Dou) ou (Dou,Cas)
- aresta $e_4$ é incidente sobre Wor e Buf
- Os vértices Wor e Buf são adjacentes.

## 1.1 Conceitos principais

**Grafos orientados**

Um grafo orientado ***G*** consiste de um conjunto ***V*** de vértices (ou nós) e um conjunto ***E*** de arestas (ou arcos) tal que cada aresta ***e*** ***∈ E*** é associada com um par ordenado de vértices. Se existe uma única aresta ***e*** associada com o par ordenado ***(v,w)*** de vértices, nós escrevemos ***e=(v,w),*** que denota uma aresta de ***v*** para ***w***.



## 1.1 Conceitos principais

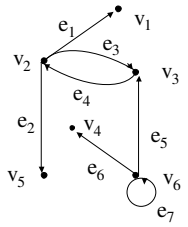



Fig. 1.3: grafo orientado



## 1.1 Conceitos principais

**Arestas paralelas:** arestas distintas associadas ao mesmo par de vértices.

**Loop:**uma aresta ***e3=(v2,v2)***

**vértice isolado:**é um vértice que não é incidente sobre nenhuma aresta.

**grafo simples:** é um grafo que não tem nem arestas paralelas e nem loops.

Exemplo:O grafo da figura 1.2 é um grafo simples, pois não tem arestas paralelas e loops. O da figura 1.3 não é um grafo simples.



## 1.1 Conceitos principais

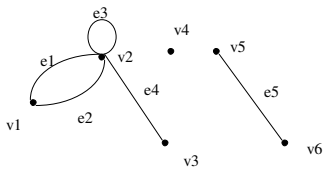



Fig. 1.4: grafo com arestas paralelas e loop

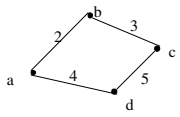
v4: vértice isolado

## 1.1 Conceitos principais

**Grafo ponderado**: é um grafo com números sobre as arestas. Se a aresta e é rotulada com k, dizemos que k é o **peso** da aresta.

**comprimento de um caminho**: é a soma dos pesos no caminho.

Consideremos um exemplo:




Peso de (a,b) é 2

comprimento de (a,b,c) é 5

Fig. 1.5: grafo ponderado



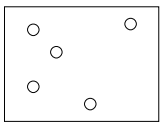
## 1.1 Conceitos principais

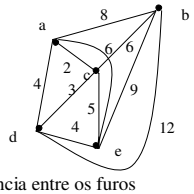
**Exemplo**: Freqüentemente em manufatura é preciso furar muitos buracos em placas de metal. Componentes são parafusados nestas placas. Os furos podem ser feitos usando uma furadeira de pressão sob controle de um computador. Para salvar tempo e dinheiro, a furadeira deverá mover tão rapidamente quanto possível.

**Problema**: qual é o caminho mínimo que passe por todos os vértices exatamente uma vez?



## 1.1 Conceitos principais






Peso das arestas: distância entre os furos

Fig. 1.6: Placa de metal com furos e o grafo correspondente



## 1.1 Conceitos principais

Caminhos no grafo de ***a*** a ***e***, passando através de cada vértice exatamente uma vez.

| Caminho | comprimento |
|---|---|
| a,b,c,d,e | 21 |
| a,b,d,c,e | 28 |
| a,c,b.d,e | 24 |
| a,c,d,b,e | 26 |
| a,d,b,c,e | 27 |
| a,d,c,b,e | 22 |

Listar todos os caminhos é uma forma de encontrar o menor caminho. Infelizmente, não se conhece um método mais prático para grafos arbitrários (problema do caixeiro viajante).

## 1.1 Conceitos principais

**Grafos de similaridade**

Este conceito trata do problema de agrupar objetos semelhantes em classes com base nas propriedades dos objetos.

**Exemplo**: suponha que um algoritmo particular é implementado em C por algumas pessoas. Queremos agrupar em classes programas similares, tomando como base determinadas propriedades:

- número de linhas do programa
- número de comandos *return*
- número de chamadas de funções.



## 1.1 Conceitos principais

| Programa | núm de linhas | Núm. de *return* | Núm. de Chamadas de funções |
|---|---|---|---|
| 1 | 66 | 20 | 1 |
| 2 | 41 | 10 | 2 |
| 3 | 68 | 5 | 8 |
| 4 | 90 | 34 | 5 |
| 5 | 75 | 12 | 14 |



## Construção do grafo de similaridade G

Os vértices $v_1,v_2,...,v_5$ correspondem aos programas.

Um vértice é denotado por $(p_1,p_2,p_3)$, onde:

pi é o valor da propriedade i.

$v= (p_1,p_2,p_3)$ $w=(q_1,q_2,q_3)$

**função de dissimilaridade s(v,w)**: mostra o quanto v e w são dissimilares.

$s(v,w)= |p_1-q_1|+|p_2-q_2|+|p_3-q_3|$

Para um valor determinado S, inserimos uma aresta entre os vértices v e w se $s(v,w)<S$ .

Dizemos que v e w estão na mesma classe se v=w ou se existe um caminho de v para w.



## Construção do grafo de similaridade G

No exemplo dos programas temos:

$s(v_1,v_2)=36$ $s(v_2,v_4)=76$

$s(v_1,v_3)=24$ $s(v_2,v_5)=48$

$s(v_1,v_4)=42$ $s(v_3,v_4)=54$

$s(v_1,v_5)=30$ $s(v_3,v_5)=20$

$s(v_2,v_3)=38$ $s(v_4,v_5)=46$

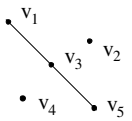


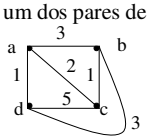

Fig.1.7:Grafo de similaridade dos programas

Se assumirmos S=25, os programas serão agrupados em 3 classes: {1, 3, 5}, {2}, {4}

O exemplo pertence à área de reconhecimento de padrões (“pattern recognition”).

## 1.1 Conceitos principais

**Grafo completo em *n* vértices ($K_n$):** é um grafo simples com *n* vértices em que existe uma aresta entre cada um dos pares de vértices distintos.




$K_4$

Fig.1.8: exemplo de grafo completo em 4 vértices

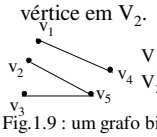
O grafo da figura 1.6 é um grafo completo.



## 1.1 Conceitos principais

**Grafo bipartido**

Um grafo G=(V,E) é bipartido se o conjunto de vértices V pode ser particionado em dois subconjuntos $V_1$ e $V_2$ tal que cada aresta em E é incidente sobre um vértice em $V_1$ e um vértice em $V_2$.




$V_1=\{v_1,v_2,v_3\}$
$V_2=\{v_4,v_5\}$

Fig.1.9 : um grafo bipartido

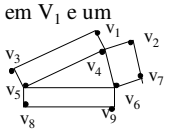


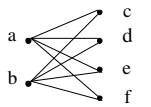

Fig.1.10 : não bipartido



## 1.1 Conceitos principais

**Grafo bipartido completo em m e n vértices (*Km,n*)**: é um grafo simples cujo conjunto de vértices pode ser particionado nos conjuntos $V_1$ com *m* vértices e *V2* com n vértices, de tal forma que existe uma aresta entre cada par de vértices $v_1 \in V_1$ e $v_2 \in V_2$.

Exemplo:




$K_{2,4}$



## 1.2 Caminhos e ciclos

Se pensarmos em vértices de um grafo como cidades e as arestas como as rodovias, um **caminho** seria uma viagem começando em alguma cidade, passando por várias cidades e terminando em alguma outra.

**caminho:** Seja $v_0$ e $v_n$ os vértices em um grafo. Um caminho de $v_0$ para $v_n$ de comprimento n é uma seqüência alternando n+1 vértices e n arestas começando com o vértice $v_0$ e terminando com o vértice $v_n$.

$(v_0,e_1,v_1,e_2,v_2, \ldots, v_{n-1},e_n,v_n)$

## 1.2 Caminhos e ciclos

Na ausência de arestas paralelas, podemos suprimir as arestas.

(a,b,d,e,c,a)

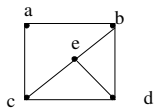



Fig. 1.11 : exemplo de grafo sem arestas paralelas



## 1.2 Caminhos e ciclos

**Grafos conectados**

É um grafo no qual de qualquer vértice para qualquer outro vértice existe um caminho.

Definição: Um grafo G é conectado se dados quaisquer vértices v e w em G, existe um caminho de v para w.

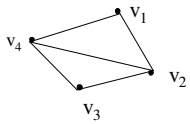



Fig.12: grafo conectado



## 1.2 Caminhos e ciclos

**Caminho simples:** um caminho simples de v para w é um caminho de v para w sem nenhum vértice repetido.

(a,b,c): caminho simples de a para c

(a,b,d,c,b): não é um caminho simples

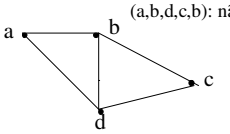


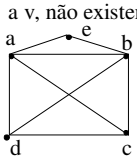

Fig.13: caminho simples



## 1.2 Caminhos e ciclos

**Ciclo:**é um caminho de v para v de comprimento não zero sem nenhuma aresta repetida.

**ciclo simples:** é um ciclo de v no qual, exceto para os vértices do início e do final que são ambos iguais a v, não existem vértices repetidos.




(a,c,d,a): ciclo simples

(a,b,c,d,a):ciclo simples

(a,b,c,d,b,e,a): não é ciclo simples

Fig. 14: ciclo simples

## 1.2 Caminhos e ciclos

**Problema da ponte de Königsberg**

(solução apresentada por Euler em 1736)

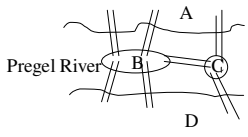



Fig. 1.15: Pontes de Königsberg

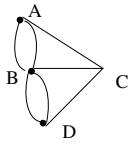



Fig. 1.16: modelo de grafo das pontes de Königsberg



## 1.2 Caminhos e ciclos

O problema é: iniciar em qualquer lugar, A,B,C ou D, atravessar cada ponte exatamente uma vez; e então retornar para o lugar inicial.

Da discussão sobre o sistema de auto-estradas, vemos que não existe solução, ou seja não existe um ciclo de Euler, pois no grafo da figura 1.8 cada vértice é incidente em um número ímpar de arestas.

**Ciclo de Euler**: é um ciclo no grafo G que inclui todas as arestas e todos os vértices.

O **grau de um vértice** $v$, $\delta(v)$, é o número de arestas incidentes em $v$.



## 1.2 Caminhos e ciclos

**Teorema 6.2.17**

Se um grafo G tem um ciclo de Euler, então G é um grafo conectado e cada vértice tem grau par.

**Teorema 6.2.18**

Se G é um grafo conectado e todo vértice tem grau par, então G tem um ciclo de Euler.



## 1.2 Caminhos e ciclos

**Teorema 6.2.19**

Se G é um grafo com m arestas e vértices {v1, v2, .., vn} então

$$\Sigma\, \delta(vi)=2m$$

Em particular a soma dos graus de todos os vértices em um grafo é par.

**Corolário 6.2.22**

Em qualquer grafo, existe um número par de vértices de grau ímpar.

## 1.2 Caminhos e ciclos

**Teorema 6.2.23**

Um grafo tem um caminho sem arestas repetidas de ***v*** para ***w*** ***(v≠w)*** contendo todas as arestas e vértices, se e somente se, ele é conectado e ***v*** e ***w*** são os únicos vértices com grau ímpar.

**Teorema 6.2.24**

Se um grafo contém um ciclo de ***v*** para ***v***, ***G*** contém um ciclo simples de ***v*** para ***v***.



## 2. O problema do caixeiro viajante

**Ciclos Hamiltonianos**

Sir William R. Hamilton inventou um puzzle (1800) na forma de um dodecaedro. A cada canto atribuiu o nome de uma cidade. O problema era: iniciar por qualquer cidade, viajar pelas arestas, visitando cada cidade exatamente uma vez, e retornar à cidade inicial.

Em homenagem a Hamilton, chamamos um ciclo em um grafo G, que contém cada vértice em G exatamente uma vez, exceto o vértice de início e fim, de ***ciclo hamiltoniano***.



## 2. O problema do caixeiro viajante

**O problema do caixeiro viajante**

Encontrar a rota mais curta na qual o caixeiro viajante visita cada cidade uma vez, iniciando e terminando na mesma cidade, passando exatamente uma vez por cada cidade.

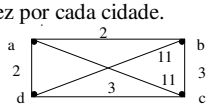


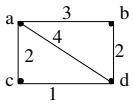

Fig. 2.1: grafo para a viagem do caixeiro viajante

A solução está relacionada em encontrar um ciclo hamiltoniano em um grafo.



## 3. Algoritmo do caminho mínimo

Em um grafo ponderado o comprimento de um caminho é a soma dos pesos das arestas no caminho.




comprimento de (a,d) é 4
comprimento do caminho (a,b,d,c) é 6

Fig. 3.1: grafo ponderado

Algoritmo do caminho mínimo: encontrar o caminho entre dois vértices dados com comprimento mínimo.

Dijkstra: propôs o algoritmo de caminho mínimo de Dijkstra.

## 3. Algoritmo do caminho mínimo

**Algoritmo do caminho mínimo de Dijkstra**

Este algoritmo encontra o comprimento de um caminho mínimo do vértice ***a*** para o vértice ***z***, em um grafo ***G*** conectado ponderado.

- Peso da aresta ***(i,j)*** é ***w(i,j).***
- ***L(x)*** é o label do vértice ***x*** e ***L(z)*** é o comprimento de um caminho mínimo de ***a*** a ***z***.



## 3. Algoritmo do caminho mínimo

Input: Um grafo conectado ponderado no qual todos os pesos são positivos. Vértices ***a*** e ***z***.

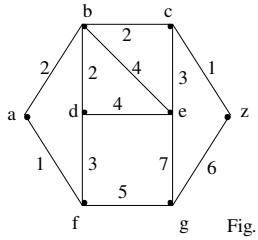
Output: ***L(z),*** o comprimento de um caminho mínimo de ***a*** a ***z***.

```
procedure:
dijkstra(w,a,z,L)
  L(a):=0
  for all vertices x ≠ a
    L(x):=∞
T:=set of all vertices
//T e o conjunto de vertices
cuja distancia minima de a
não foi encontrada
```

```
While z∈T do
  begin
  escolha v ∈T com L(v) min.
  T:=T-{v}
  for each x ∈T adjacente a v do
   L(x):=min{L(x),L(v)+w(v,x)}
  end
end dijkstra
```



## 3. Algoritmo do caminho mínimo

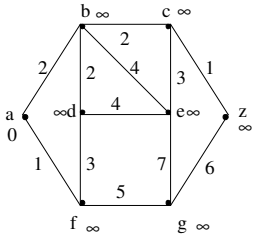



Grafo G
T: conjunto de vértices cuja distância mais curta de *a* ainda não foi encontrada

Fig. 3.2: modelo de grafo para o caminho mínimo

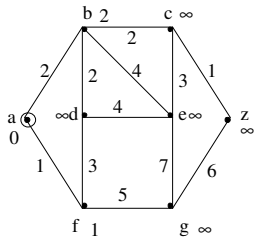


## 3. Algoritmo do caminho mínimo




Início
L(a)=0
T={a,b.c,d,e,f,g,z}

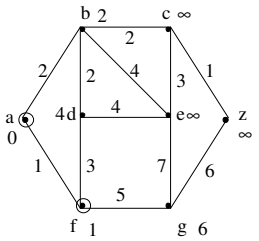
## 3. Algoritmo do caminho mínimo




Iteração 1: escolhe v=a, pois L(a) é o mínimo entre L(a) ... L(z)
T={b,c,d,e,f,g,z}
Calcula L(x), para x ∈ T adjacente a a
L(x):=min{L(x),L(a)+w(a,x)}
L(b)=min(∞,0+2)=2
L(f)=min(∞,0+1)=1



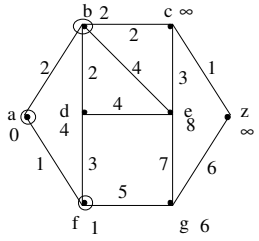
## 3. Algoritmo do caminho mínimo




Iteração 2: escolhe f
L(f)=1
T={b,c,d,e,g,z}
Calcula L(x), x∈ T adjacente a f
L(d)=min(∞,1+3)=4
L(g)=min(∞,1+5)=6



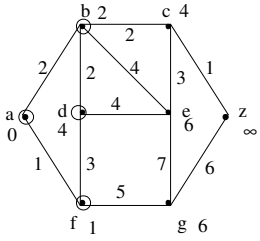
## 3. Algoritmo do caminho mínimo




Iteração 3: escolhe b
T={c,d,e,g,z}
Calcula L(x), para x ∈ T adjacente a b
L(x):=min{L(x),L(b)+w(b,x)}
L(c)=min(∞,2+2)=4
L(d)=min(4,2+2)=4
L(e)=min(∞,2+4)=6

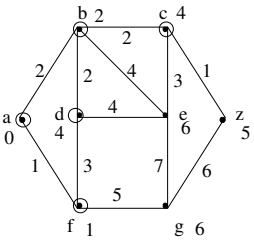


## 3.Algoritmo do caminho mínimo




Iteração 4: escolhe d
L(d)=4
T={c,e,g,z}
Calcula L(x), x∈ T adjacente a d
L(e)=min(6,4+4)=6

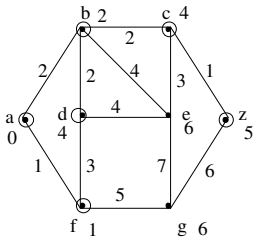
## 3. Algoritmo do caminho mínimo




Iteração 5: escolhe c
L( c )=4
T={e,g,z}
Calcula L(x),x∈ T adjacente a c
L(e)=min(6,4+3)=6
L(z)=min(∞,4+1)=5



## 3. Algoritmo do caminho mínimo




Iteração 6: escolhe z
L(z)=5
T={e,g}
Calcula L(x), x∈ T adjacente a z
L(g)=min(6,1+5)=6



## Bibliografia

[1] Johnsonbaugh, R. Discrete Mathematics. Prentice Hall International, London, UK, 4th. Ed. 1997. Cap. 6.