

CONCEITOS SOBRE GRAFOS

MO417 - Complexidade de
Algoritmos I

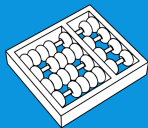
Santiago Valdés Ravelo
[https://ic.unicamp.br/~santiago/
ravelo@unicamp.br](https://ic.unicamp.br/~santiago/ravelo@unicamp.br)

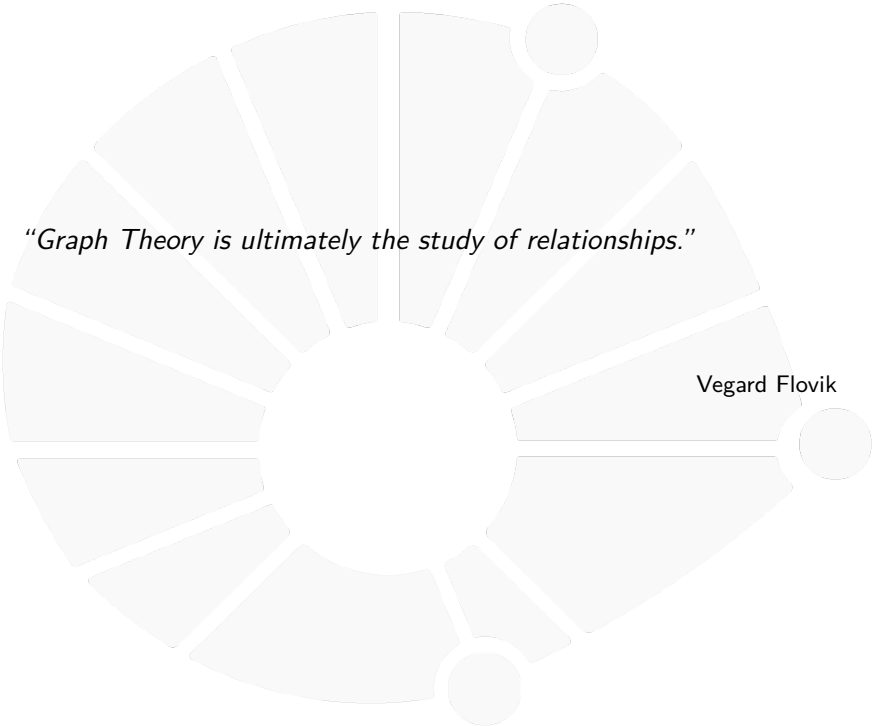
05/24

17



UNICAMP



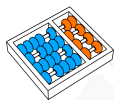


"Graph Theory is ultimately the study of relationships."

Vegard Flovik



GRAFOS BIPARTIDOS

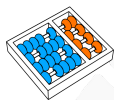


Definição

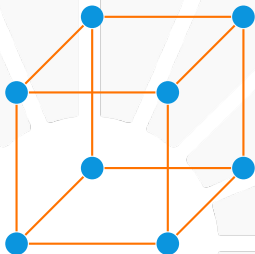
Uma **BIPARTIÇÃO** de um conjunto V é um par (A, B) tal que:

- ▶ $A \cap B = \emptyset$ e
- ▶ $A \cup B = V$.

Um grafo $G = (V, E)$ é **BIPARTIDO** se existe uma partição (A, B) de V tal que toda aresta de G tem um extremo em A e outro em B .



Exemplo

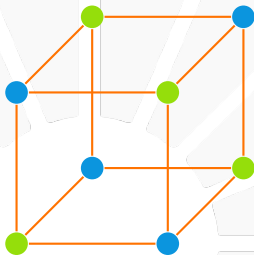


Esse grafo é bipartido?

Podemos indicar cada parte com uma cor: azul ou verde.



Exemplo

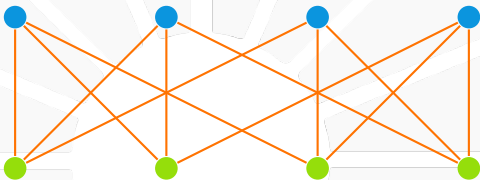


É bipartido!

Um grafo $G = (\mathbf{V}, \mathbf{E})$ é bipartido se for possível colorir os vértices de G com **DUAS CORES** de modo que vértices adjacentes tenham cores distintas.



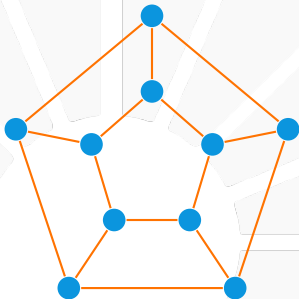
Exemplo



Isto pode ser visto melhor com outro desenho.



Exemplo

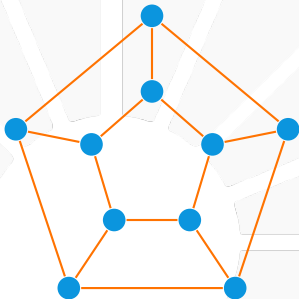


Este grafo **NÃO** é bipartido.

Podemos apresentar uma justificativa simples?



Condição necessária para um grafo ser bipartido



Um grafo bipartido não pode conter ciclos de comprimento ímpar!

Essa é uma condição suficiente? basta não ter ciclos ímpares?



Condição necessária e suficiente

Teorema

Seja G um grafo. Então G é bipartido se e somente se G não possui um ciclo ímpar.

Demonstração:

- ▶ Já vimos que se G tem um ciclo ímpar, ele não é bipartido.
- ▶ Assim, resta demonstrar a recíproca.
- ▶ Podemos supor que G é conexo. Por quê?
- ▶ Antes de continuar a prova, vejamos outros resultados...



ÁRVORE GERADORA



Propriedade

Fato (1)

Todo grafo conexo contém uma árvore geradora.

Provar o fato a partir do seguinte resultado (exercício):

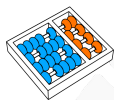
Lema

Seja G um grafo conexo e seja C um ciclo de G . Se e é uma aresta de C então $G - e$ é conexo.

A recíproca também vale:

Lema

Seja G um grafo conexo e seja e uma aresta de G . Se $G - e$ é conexo então e pertence a algum ciclo de G .

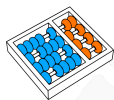


Árvores e grafos bipartidos

Fato (2)

Toda árvore $T = (V, E)$ é um grafo bipartido.

É possível provar por indução em $|V|$. Demonstre como exercício.



Árvore geradora

Fato (3)

Seja $T = (V, E')$ uma árvore geradora de um grafo $G = (V, E)$.
Então para toda aresta $e \in E \setminus E'$ existe um único ciclo em
 $T + e = (V, E' \cup \{e\})$.

Demonstração:

- ▶ Sejam u, v os extremos de e .
- ▶ Como T é árvore, existe um único caminho P de u a v em T .
- ▶ Portanto, $P + e$ é o único ciclo em $T + e$.

O único ciclo de $T + e$ é chamado de **CICLO FUNDAMENTAL**.



Demonstração do teorema

Agora estamos prontos para demonstrar a segunda parte do teorema:

Teorema

Seja G um grafo. Então G é bipartido se e somente se G não possui um ciclo ímpar.

Demonstração:

- ▶ Resta mostrar que se G não tem ciclo ímpar, ele é bipartido.
- ▶ Lembre, podemos supor que G seja conexo.
- ▶ Suponha que G não contenha um ciclo ímpar.
- ▶ Construiremos uma bipartição (A, B) de V tal que toda aresta de G tem um extremo em A e outro em B .



Demonstração do teorema

- ▶ Pelo Fato 1, G contém uma árvore geradora $T = (V, E')$.
- ▶ Pelo Fato 2, T possui uma bipartição (A, B) de V tal que toda aresta de T tem um extremo em A e outro em B .
- ▶ Mostraremos que toda aresta de $E \setminus E'$ tem um extremo em A e outro em B .



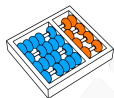
Demonstração do teorema

Seja e uma aresta de $E \setminus E'$:

- ▶ Pelo Fato 3, existe um único ciclo C em $T + e$ que contém e .
- ▶ Se os extremos de e são da mesma parte (A ou B),
- ▶ então C é um ciclo ímpar, o que é uma **CONTRADIÇÃO!**
- ▶ Portanto, os extremos de e estão em partes distintas.

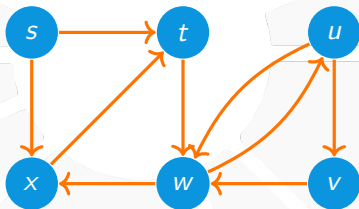


GRAFOS DIRECCIONADOS



Definição

Um **GRAFO DIRECIONADO** ou **DIGRAFO** é definido de forma semelhante, com a diferença que as arestas (chamadas também de **ARCOS**) consistem de **PARES ORDENADOS** de vértices.





Adjacência de grafos direcionados

Considere uma aresta $e = (u, v)$ de um grafo direcionado G :

- ▶ Dizemos que e sai de u e entra em v .
- ▶ O vértice u é a **CAUDA** de e .
- ▶ O vértice v é **CABEÇA** de e .

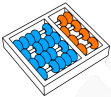
Temos dois tipos de grau para grafos direcionados:

- ▶ **GRAU DE SAÍDA:** $d^+(v)$ é o número de arestas que saem de v .
- ▶ **GRAU DE ENTRADA:** $d^-(v)$ é o número de arestas que entram em v .

Teorema

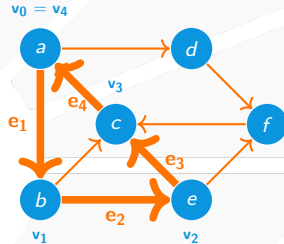
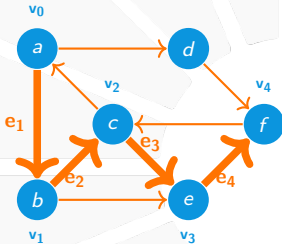
Para todo grafo direcionado $G = (V, E)$ temos:

$$\sum_{v \in V} d^+(v) = \sum_{v \in V} d^-(v) = |E|.$$

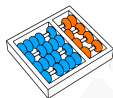


Passeios em grafos direcionados

Em um **PASSEIO DIRECIONADO** de um grafo direcionado todas as arestas seguem o mesmo sentido.



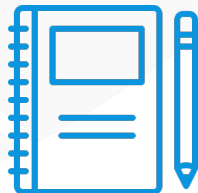
- ▶ Definimos **CAMINHOS E CICLOS DIRECIONADOS** analogamente, assim como subgrafos de um grafo direcionado.
- ▶ Noções de conectividade serão vistas depois.

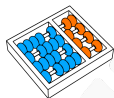


Refletindo sobre as definições



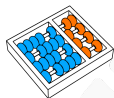
Vamos fazer alguns exercícios?





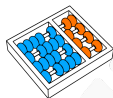
Exercício 1

Seja G um grafo direcionado e u, v vértices de G . Mostre que se existe um passeio de u a v em G , então existe um caminho de u a v em G .



Exercício 2

Seja G um grafo direcionado e u, v, w vértices de G . Mostre que se em G existem um caminho de u a v e um caminho de v a w então existe um caminho de u a w em G .

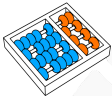


Exercício 3

É verdade que todo passeio fechado em um grafo direcionado contém um ciclo direcionado?



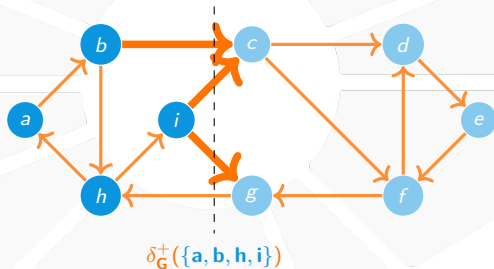
CORTES EM DIGRAFOS



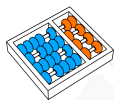
Cortes em grafos direcionados

Seja $G = (V, E)$ um grafo direcionado e seja $S \subset V$.

Denote por $\delta_G^+(S)$ o **CORTE DIRECIONADO** de G induzido por S , que contém o conjunto de arestas de G com cauda em S e cabeça em $V \setminus S$.



Se $s \in S$ e $t \in V \setminus S$ dizemos que $\delta_G^+(S)$ **SEPARA** s de t .



Caminhos versus cortes direcionados

Lema

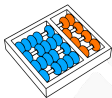
Seja G um grafo direcionado e sejam s, t vértices distintos de G . Então, exatamente uma das afirmações é verdadeira:

- (a) Existe um caminho de s a t em G , ou
- (b) existe um corte direcionado $\delta_G^+(S)$ que separa s de t tal que $\delta_G^+(S) = \emptyset$.

A demonstração é análoga à do lema para grafos não direcionados. Faça como exercício.

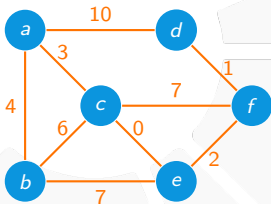


GRAFOS PONDERADOS



Definição

Um grafo (direcionado ou não) é **PONDERADO** se a cada aresta **e** do grafo está associado um valor real **w(e)**, denominado **PESO** ou **CUSTO** da aresta.





REPRESENTAÇÃO DE GRAFOS



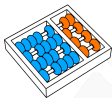
Motivação prática

Grafos podem modelar diversas estruturas reais:

1. Se computadores são representados por vértices, então as conexões entre eles correspondem a arestas.
2. Se as cidades forem representadas por vértices, então as estradas correspondem a arestas direcionadas.
3. etc.

Queremos construir algoritmos genéricos:

- ▶ Vamos estudar algoritmos para grafos de modo **ABSTRATO**,
- ▶ mas que sejam aplicados em problemas **CONCRETOS**.



Problemas

Problema do caminho mínimo. Dadas as cidades, as distâncias entre elas e duas cidades A e B , determinar um **TRAJETO MAIS CURTO** de A até B .

Problema da árvore geradora mínima. Dados os computadores e o custo de conectar cada par de computadores, projetar uma rede de **MENOR CUSTO** possível interconectando todos os computadores.

Problema do emparelhamento máximo. Dadas vagas de empregos e uma lista de candidatos para cada vaga, determinar uma lista de associações candidato-emprego de **MAIOR TAMANHO** possível.



Problemas

Problema do caixeiro viajante. Dadas cidades e as distâncias entre elas, encontrar um rota de **COMPRIMENTO MÍNIMO** que visita todas as cidades.

Problema do carteiro chinês. Dadas as ruas de um bairro, encontrar uma rota fechada de **COMPRIMENTO MÍNIMO** que passa por todas as ruas.



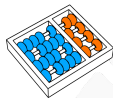
Representação interna de grafos

Representamos grafos de duas maneiras principais:

1. **MATRIZ DE ADJACÊNCIA.**
2. **LISTAS DE ADJACÊNCIA.**

Qual estrutura de dados escolher?

- ▶ Depende do problema sendo tratado e das operações realizadas pelo algoritmo.
- ▶ A estrutura escolhida afeta a **COMPLEXIDADE DO ALGORITMO.**

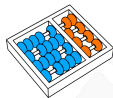


Matriz de adjacência

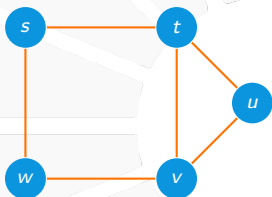
A **MATRIZ DE ADJACÊNCIA** de um grafo simples G é uma matriz quadrada A de ordem $|\mathbf{V}|$ tal que:

$$A[i,j] = \begin{cases} 1 & \text{se } (i,j) \in \mathbf{E}, \\ 0 & \text{caso contrário.} \end{cases}$$

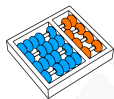
- ▶ O grafo pode ser direcionado ou não.
- ▶ Se G for não direcionado, então a matriz A é simétrica.



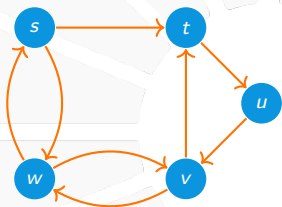
Matriz de adjacência



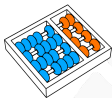
	s	t	u	v	w
s	0	1	0	0	1
t	1	0	1	1	0
u	0	1	0	1	0
v	0	1	1	0	1
w	1	0	0	1	0



Matriz de adjacência



	s	t	u	v	w
s	0	1	0	0	1
t	0	0	1	0	0
u	0	0	0	1	0
v	0	1	0	0	1
w	1	0	0	1	0



Listas de adjacência

Para representar um grafo $G = (V, E)$ por **LISTAS DE ADJACÊNCIAS**:

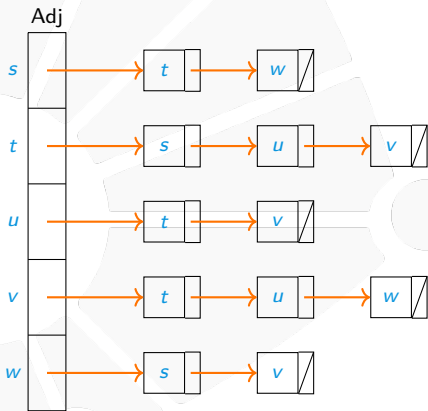
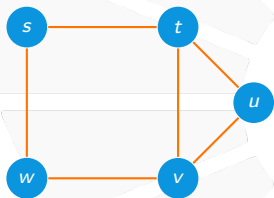
- ▶ Criamos uma lista ligada $Adj[v]$ para cada vértice v .
- ▶ Adicionamos a $Adj[v]$ todos os vértices adjacentes a v .

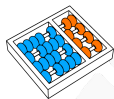
Como representamos uma aresta (u, v) ?

- ▶ Se a aresta for direcionada, então v está em $Adj[u]$.
- ▶ Se a aresta for não direcionada, então v está em $Adj[u]$ e u está em $Adj[v]$.

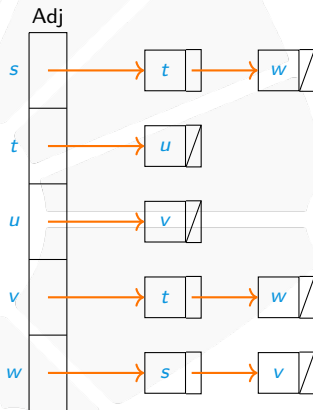
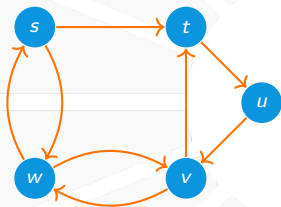


Listas de adjacência





Listas de adjacências





Notação para complexidade

Considere um grafo $G = (V, E)$:

- ▶ Vamos simplificar a **NOTAÇÃO ASSINTÓTICA**.
- ▶ Escrevemos **V** e **E** ao invés de $|V|$ e $|E|$.
- ▶ Por exemplo, $O(E^2 \log V)$ ao invés de $O(|E|^2 \log |V|)$.



Matriz versus listas

A melhor representação depende do algoritmo.

1. Matriz de adjacência:

- ▶ É fácil verificar se (u, v) é uma aresta de G .
- ▶ O espaço utilizado é $\Theta(V^2)$.
- ▶ Adequada para grafos densos (com $|E| = \Theta(V^2)$).

2. Listas de adjacência:

- ▶ É fácil listar os vértices adjacentes de um dado vértice v .
- ▶ O espaço utilizado é $\Theta(V + E)$.
- ▶ Adequada a grafos esparsos (com $|E| = \Theta(V)$).



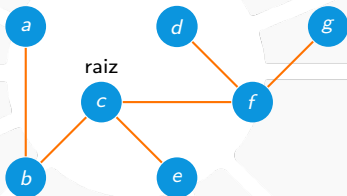
Extensões

- ▶ Há alternativas para representar grafos, mas matrizes e listas de adjacência são as mais usadas.
- ▶ Essas representações podem ser usadas para grafos ponderados, grafos com laços e arestas múltiplas, grafos com pesos nos vértices etc.
- ▶ Para determinados algoritmos é importante manter **ESTRUTURAS DE DADOS ADICIONAIS.**



Representação de árvores

Uma **ÁRVORE ENRAIZADA** é uma árvore com um vértice especial chamado **RAIZ**.

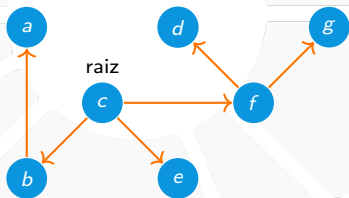


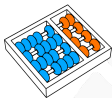


Representação de árvores

Uma **ÁRVORE DIRECIONADA** com raiz r é um grafo direcionado acíclico $T = (\mathbf{V}, \mathbf{E})$ tal que:

1. $d^-(r) = 0$,
2. $d^-(v) = 1$ para $v \in \mathbf{V} \setminus \{r\}$.



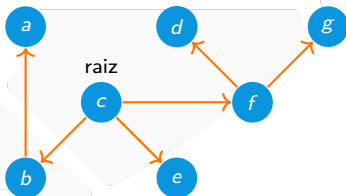
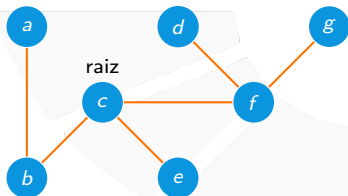


Representação de árvores

Representar uma árvore enraizada com um vetor π de **PREDECESSORES**.

vértice	a	b	c	d	e	f	g
π	b	c	N	f	c	c	f

O símbolo N indica a ausência de predecessor.





Alguns detalhes de implementação

- ▶ Nos algoritmos que veremos, usamos a representação de um grafo (direcionado ou não) por listas de adjacências.
- ▶ Em uma implementação real de um algoritmo, provavelmente a representação **NÃO** é dada a priori.
- ▶ Assim, é necessário construir tal representação a partir da dos dados de entrada.
- ▶ Como construir a representação de um grafo depende do formato da entrada.



Exemplo de entrada

Suponha que a entrada é um arquivo texto:

```
5 7
0 1
0 2
1 2
1 3
2 3
2 4
3 4
```

- ▶ O arquivo representa um grafo não direcionado.
- ▶ A primeira linha contém $|V|$ e $|E|$.
- ▶ Os vértices são numerados de 0 a $|V| - 1$.
- ▶ As próximas $|E|$ linhas representam os extremos das arestas.



Exemplo de construção

Algoritmo 1: CONSTRUIR-ADJ()

- 1 leia n e m
 - 2 **repita** m vezes
 - 3 leia a próxima aresta (u, v)
 - 4 insira v na lista $Adj[u]$
 - 5 insira u na lista $Adj[v]$
 - 6 **devolva** Adj
-



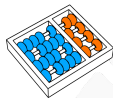
Calculando o quadrado

O **QUADRADO** de um grafo direcionado $G = (V, E)$ é o grafo direcionado $G^2 = (V, E^2)$ tal que:

- ▶ $(u, v) \in E^2$ se e somente se existe um caminho de comprimento no máximo dois de u a v em G .

Dado G , vamos calcular G^2 :

1. Utilizando matriz de adjacência.
2. Utilizando listas de adjacência.



Quadrado com matriz de adjacência

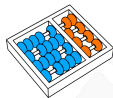
Algoritmo 2: QUADRADO(M)

```

1  $N \leftarrow M$ 
2 para cada  $u \in V$ 
3   para cada  $v \in V$ 
4     para cada  $w \in V$ 
5       se  $N[u, v] = 0$ 
6          $N[u, v] \leftarrow M[u, w] \cdot M[w, v]$ 
7 devolva  $N$ 

```

Complexidade: $\Theta(V^3)$.



Quadrado com listas de adjacências

Algoritmo 3: QUADRADO(Adj)

- 1 crie uma cópia Adj' de Adj
 - 2 **para cada** $v \in V$
 - 3 **para cada** $u \in Adj[v]$
 - 4 concatene uma cópia de $Adj[u]$ a $Adj'[v]$
 - 5 **para cada** $v \in V$
 - 6 ordene $Adj'[v]$ usando COUNTING-SORT
 - 7 elimine as repetições de $Adj'[v]$ em tempo linear
 - 8 **devolva** Adj'
-

Complexidade: $\Theta(VE)$.

CONCEITOS SOBRE GRAFOS

MO417 - Complexidade de
Algoritmos I

Santiago Valdés Ravelo
[https://ic.unicamp.br/~santiago/
ravelo@unicamp.br](https://ic.unicamp.br/~santiago/ravelo@unicamp.br)

05/24

17



UNICAMP

