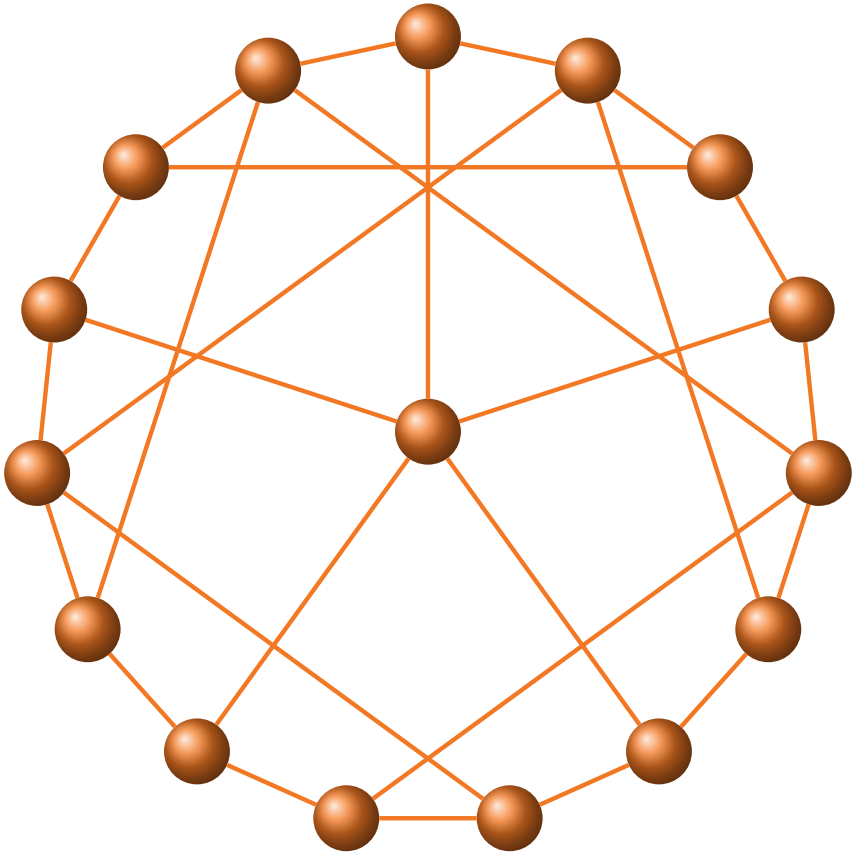
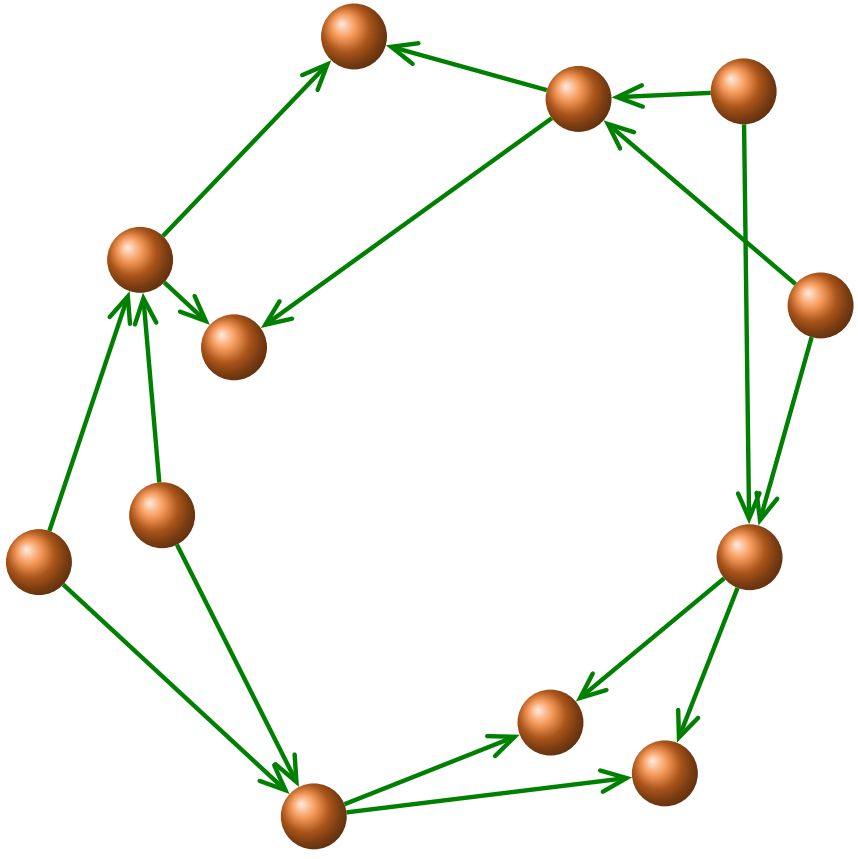
MAC328 Algoritmos em Grafos

Arnaldo Mandel

2º Semestre 2023

Grafos e digrafos





Algoritmos

No mundo da computação...

Algoritmos

No mundo da computação...

Vocês sabem o que são: entrada, saída, instruções precisas, etc.

Algoritmos

No mundo da computação...

Vocês sabem o que são: entrada, saída, instruções precisas, etc.

Algoritmos

No mundo da computação...

Vocês sabem o que são: entrada, saída, instruções precisas, etc.

No mundo jornalístico

Algoritmos

No mundo da computação...

> Vocês sabem o que são: entrada, saída, instruções precisas, etc.

No mundo jornalístico

> **algoritmo** (s.m.): Aquela entidade misteriosa que está por trás das redes sociais e mecanismos de busca e que parece ter alguma coisa a ver com Inteligência Artificial.

Algoritmos

No mundo da computação...

Vocês sabem o que são: entrada, saída, instruções precisas, etc.

No mundo jornalístico

algoritmo (s.m.): Aquela entidade misteriosa que está por trás das redes sociais e mecanismos de busca e que parece ter alguma coisa a ver com Inteligência Artificial.

Ex: *Vamos educar **o** algoritmo.*
(bordão de uma jornalista da CBN)

Algoritmos

No mundo da computação...

Vocês sabem o que são: entrada, saída, instruções precisas, etc.

No mundo jornalístico

algoritmo (s.m.): Aquela entidade misteriosa que está por trás das redes sociais e mecanismos de busca e que parece ter alguma coisa a ver com Inteligência Artificial.

Ex: *Vamos educar **o** algoritmo.*
(bordão de uma jornalista da CBN)

Notem o artigo definido.

Uma classificação computacional informal

Grafos pequenos - cabem na memória, percorrer o conjunto dos vértices é tranquilo. Contexto deste curso. Pode ter algumas centenas de milhões de vértices.

Uma classificação computacional informal

Grafos pequenos - cabem na memória, percorrer o conjunto dos vértices é tranquilo. Contexto deste curso. Pode ter algumas centenas de milhões de vértices.

Grafos grandes - existe um espaço de *nomes*, e algoritmos polinomiais nos nomes que reconhecem vértices e arestas. Número de vértices exponencial no tamanho dos nomes.

Uma classificação computacional informal

Grafos pequenos - cabem na memória, percorrer o conjunto dos vértices é tranquilo. Contexto deste curso. Pode ter algumas centenas de milhões de vértices.

Grafos grandes - existe um espaço de *nomes*, e algoritmos polinomiais nos nomes que reconhecem vértices e arestas. Número de vértices exponencial no tamanho dos nomes.

Grafos gigantes - conjunto enorme armazenado de vértices, não pode ser percorrido por extenso.

MAC0328

MAC0328 combina técnicas de

- programação

para resolver problemas sobre grafos.

MAC0328

MAC0328 combina técnicas de

- programação
- estruturas de dados

para resolver problemas sobre grafos.

MAC0328

MAC0328 combina técnicas de

- programação
- estruturas de dados
- análise de algoritmos

para resolver problemas sobre grafos.

MAC0328

MAC0328 combina técnicas de

- programação
- estruturas de dados
- análise de algoritmos
- teoria dos grafos

para resolver problemas sobre grafos.

Focos

São dois (será que este é um curso elíptico?):

Focos

São dois (será que este é um curso elíptico?):

Desenvolvimento e análise rigorosa de algoritmos.

Focos

São dois (será que este é um curso elíptico?):

Desenvolvimento e análise rigorosa de algoritmos.

Discussão de detalhes de implementação em diferentes ambientes de programação.

Referências

- PF = Paulo Feofiloff,
  Algoritmos para Grafos em C via Sedgewick
  www.ime.usp.br/~pf/algoritmos_para_grafos

Referências

- PF = Paulo Feofiloff,
  Algoritmos para Grafos em C via Sedgewick
  www.ime.usp.br/~pf/algoritmos_para_grafos
- S = Robert Sedgewick,
  Algorithms in C (part 5: Graph Algorithms)

Referências

- PF = Paulo Feofiloff,
  Algoritmos para Grafos em C via Sedgewick
  www.ime.usp.br/~pf/algoritmos_para_grafos
- S = Robert Sedgewick,
  Algorithms in C (part 5: Graph Algorithms)
- CLRS = Cormen-Leiserson-Rivest-Stein,
  Introductions to Algorithms

Referências

- PF = Paulo Feofiloff,
  Algoritmos para Grafos em C via Sedgewick
  www.ime.usp.br/~pf/algoritmos_para_grafos
- S = Robert Sedgewick,
  Algorithms in C (part 5: Graph Algorithms)
- CLRS = Cormen-Leiserson-Rivest-Stein,
  Introductions to Algorithms
- DaMNeD = David Joyner, Minh Van Nguyen, and David Phillips,
  Algorithmic Graph Theory and Sage

Referências

- PF = Paulo Feofiloff,
  Algoritmos para Grafos em C via Sedgewick
  www.ime.usp.br/~pf/algoritmos_para_grafos
- S = Robert Sedgewick,
  Algorithms in C (part 5: Graph Algorithms)
- CLRS = Cormen-Leiserson-Rivest-Stein,
  Introductions to Algorithms
- DaMNeD = David Joyner, Minh Van Nguyen, and David Phillips,
  Algorithmic Graph Theory and Sage

Linguagens

Linguagens

- Alguns slides, C.

  Usado em PF, S.

  Existem várias bibliotecas para C, C++,
  a principal é Boost

Linguagens

- Alguns slides, C.

  Usado em PF, S.

  Existem várias bibliotecas para C, C++,
  a principal é Boost
- Pseudocódigo do CLRS

Linguagens

- Alguns slides, C.
  Usado em PF, S.
  Existem várias bibliotecas para C, C++,
  a principal é Boost
- Pseudocódigo do CLRS
- Em apresentações ao vivo, Python, dentro do
  SageMath (popularmente Sage).
  Usado no DaMNeD

Linguagens

- Alguns slides, C.
  Usado em PF, S.
  Existem várias bibliotecas para C, C++,
  a principal é Boost
- Pseudocódigo do CLRS
- Em apresentações ao vivo, Python, dentro do
  SageMath (popularmente Sage).
  Usado no DaMNeD

  Recomendado instalar o Sage (>=9.5) ou se familiarizar com SageCell ou CoCalc.

Linguagens

- Alguns slides, C.
  Usado em PF, S.
  Existem várias bibliotecas para C, C++,
  a principal é Boost
- Pseudocódigo do CLRS
- Em apresentações ao vivo, Python, dentro do
  SageMath (popularmente Sage).
  Usado no DaMNeD

  Recomendado instalar o Sage (>=9.5) ou se familiarizar com SageCell ou CoCalc.
  Outra implementação de grafos para Python:
  networkx .

Linguagens

- Alguns slides, C.
  Usado em PF, S.
  Existem várias bibliotecas para C, C++,
  a principal é Boost
- Pseudocódigo do CLRS
- Em apresentações ao vivo, Python, dentro do
  SageMath (popularmente Sage).
  Usado no DaMNeD

  Recomendado instalar o Sage (>=9.5) ou se familiarizar com SageCell ou CoCalc.

  Outra implementação de grafos para Python:
  networkx .
- LaTeX

Pré-requisitos

O pré-requisito oficial de MAC0328 é

- MAC0122 Algoritmos e Estruturas de Dados I

No entanto, é recomendável que já tenham cursado

Costuma ser conveniente cursar MAC0328 simultaneamente com

Pré-requisitos

O pré-requisito oficial de MAC0328 é

- MAC0122 Algoritmos e Estruturas de Dados I

No entanto, é recomendável que já tenham cursado

- MAC0216 Técnicas de Programação I; e

Costuma ser conveniente cursar MAC0328 simultaneamente com

Pré-requisitos

O pré-requisito oficial de MAC0328 é

- MAC0122 Algoritmos e Estruturas de Dados I

No entanto, é recomendável que já tenham cursado

- MAC0216 Técnicas de Programação I; e
- MAC0323 Algoritmos e Estruturas de Dados II

Costuma ser conveniente cursar MAC0328 simultaneamente com

Pré-requisitos

O pré-requisito oficial de MAC0328 é

- MAC0122 Algoritmos e Estruturas de Dados I

No entanto, é recomendável que já tenham cursado

- MAC0216 Técnicas de Programação I; e
- MAC0323 Algoritmos e Estruturas de Dados II

Costuma ser conveniente cursar MAC0328 simultaneamente com

- MAC0338 Análise de algoritmos.

Administração

Página da disciplina:

www.ime.usp.br/~am/328

Principais tópicos

- Estruturas de dados para grafos

Principais tópicos

- Estruturas de dados para grafos
- Caminhos em grafos dirigidos.

Principais tópicos

- Estruturas de dados para grafos
- Caminhos em grafos dirigidos.
- Conexão de grafos: componentes, grafos biconexos.

Principais tópicos

- Estruturas de dados para grafos
- Caminhos em grafos dirigidos.
- Conexão de grafos: componentes, grafos biconexos.
- Digrafos fortemente conexos.

Principais tópicos

- Estruturas de dados para grafos
- Caminhos em grafos dirigidos.
- Conexão de grafos: componentes, grafos biconexos.
- Digrafos fortemente conexos.
- Emparelhamentos máximos em grafos bipartidos.

Principais tópicos

- Estruturas de dados para grafos
- Caminhos em grafos dirigidos.
- Conexão de grafos: componentes, grafos biconexos.
- Digrafos fortemente conexos.
- Emparelhamentos máximos em grafos bipartidos.
- Emparelhamentos em grafos arbitrários (algoritmo de Edmonds).

Principais tópicos

- Estruturas de dados para grafos
- Caminhos em grafos dirigidos.
- Conexão de grafos: componentes, grafos biconexos.
- Digrafos fortemente conexos.
- Emparelhamentos máximos em grafos bipartidos.
- Emparelhamentos em grafos arbitrários (algoritmo de Edmonds).
- Fluxo máximo (algoritmo de Ford-Fulkerson).

Principais tópicos

- Estruturas de dados para grafos
- Caminhos em grafos dirigidos.
- Conexão de grafos: componentes, grafos biconexos.
- Digrafos fortemente conexos.
- Emparelhamentos máximos em grafos bipartidos.
- Emparelhamentos em grafos arbitrários (algoritmo de Edmonds).
- Fluxo máximo (algoritmo de Ford-Fulkerson).
- Coloração de vértices.

Principais tópicos

- Estruturas de dados para grafos
- Caminhos em grafos dirigidos.
- Conexão de grafos: componentes, grafos biconexos.
- Digrafos fortemente conexos.
- Emparelhamentos máximos em grafos bipartidos.
- Emparelhamentos em grafos arbitrários (algoritmo de Edmonds).
- Fluxo máximo (algoritmo de Ford-Fulkerson).
- Coloração de vértices.
- Circuitos hamiltonianos.

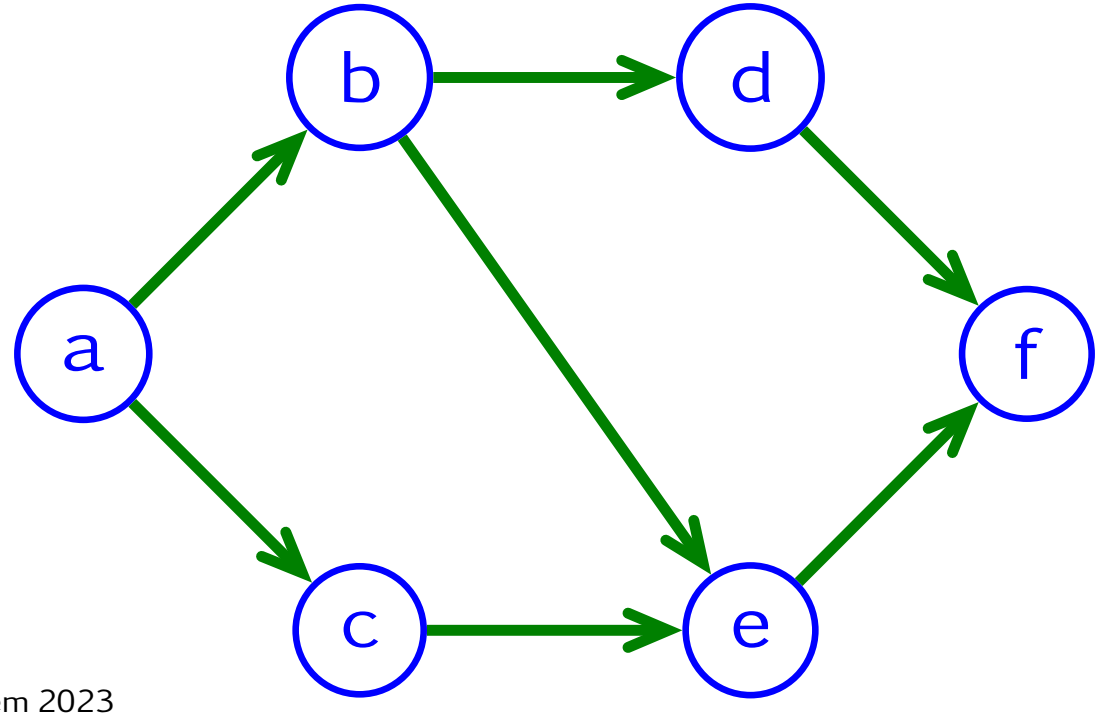
Principais tópicos

- Estruturas de dados para grafos
- Caminhos em grafos dirigidos.
- Conexão de grafos: componentes, grafos biconexos.
- Digrafos fortemente conexos.
- Emparelhamentos máximos em grafos bipartidos.
- Emparelhamentos em grafos arbitrários (algoritmo de Edmonds).
- Fluxo máximo (algoritmo de Ford-Fulkerson).
- Coloração de vértices.
- Circuitos hamiltonianos.
- Tópicos opcionais: link analysis, network analysis, redes aleatórias, grafos gigantes.

Digrafos

Um digrafo (*directed graph*) consiste de um conjunto de vértices e um conjunto de arcos. Cada aresta vai do vértice inicial ao vértice final.

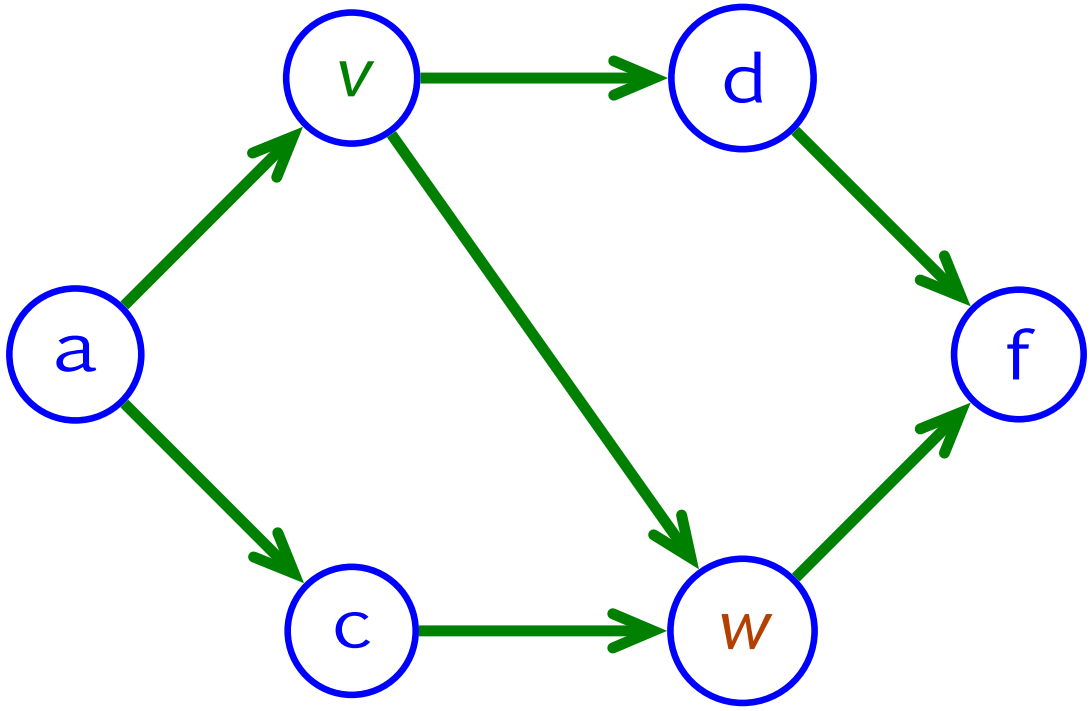
Exemplo: representação de um grafo



Arcos

Um arco é um par ordenado de vértices

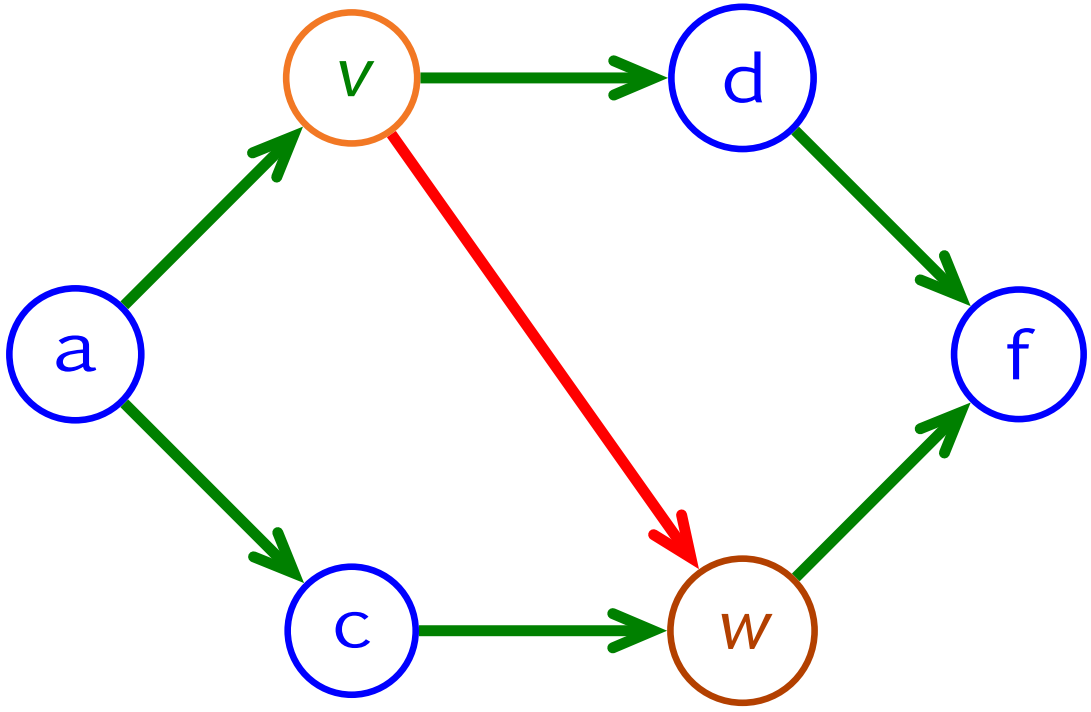
Exemplo: v e w são vértices e $v \longrightarrow w$ é um arco



Ponta inicial e final

Para cada arco $v \longrightarrow w$, o vértice v é a ponta inicial e w é a ponta final

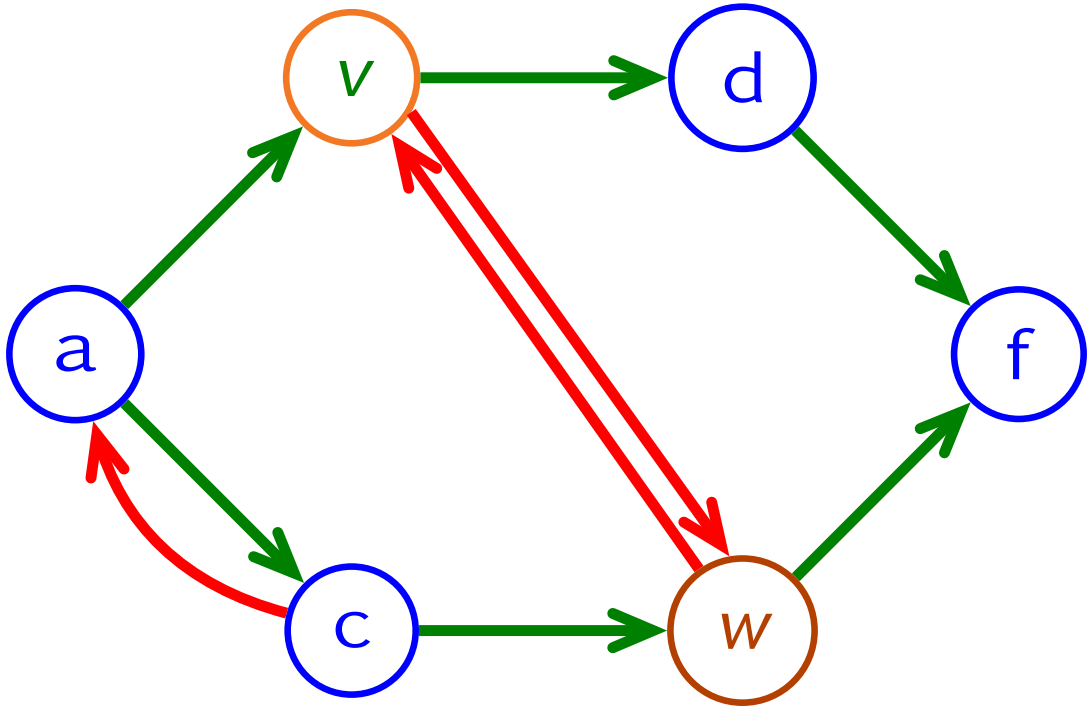
Exemplo: v é ponta inicial e w é ponta final de $v \longrightarrow w$



Arcos anti-paralelos

Dois arcos são anti-paralelos se a ponta inicial de um é ponta final do outro

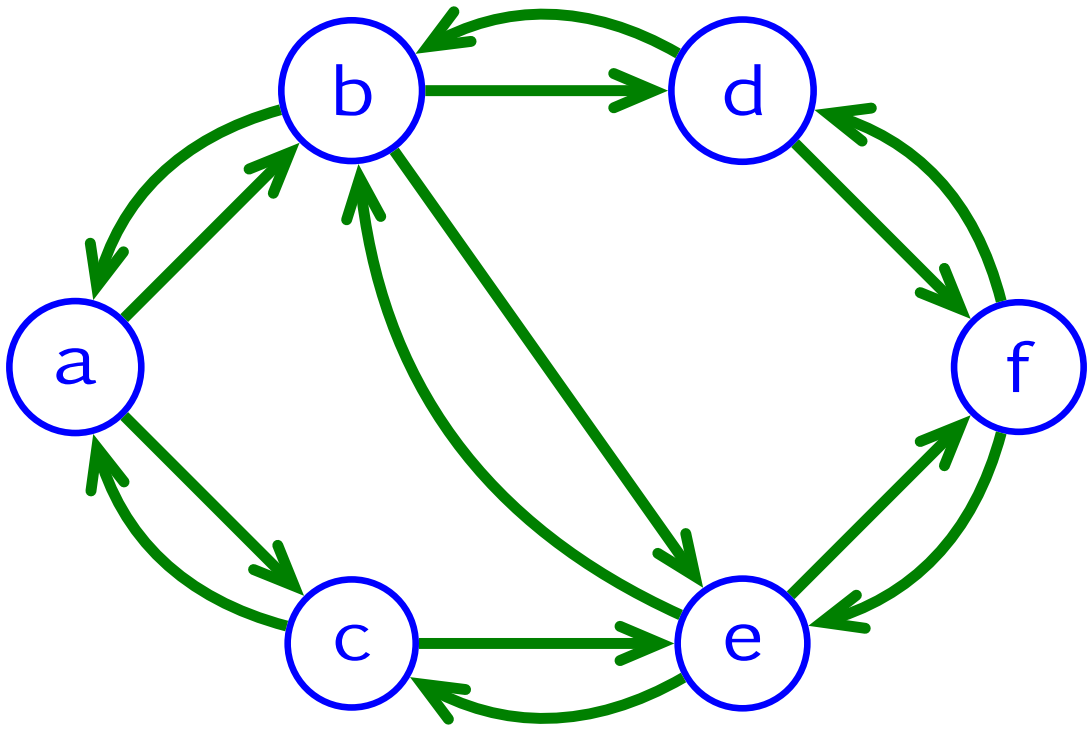
Exemplo: $v \longrightarrow w$ e $w \longrightarrow v$ são anti-paralelos



Digrafos simétricos

Um digrafo é simétrico se cada um de seus arcos é anti-paralelo a outro

Exemplo: digrafo simétrico

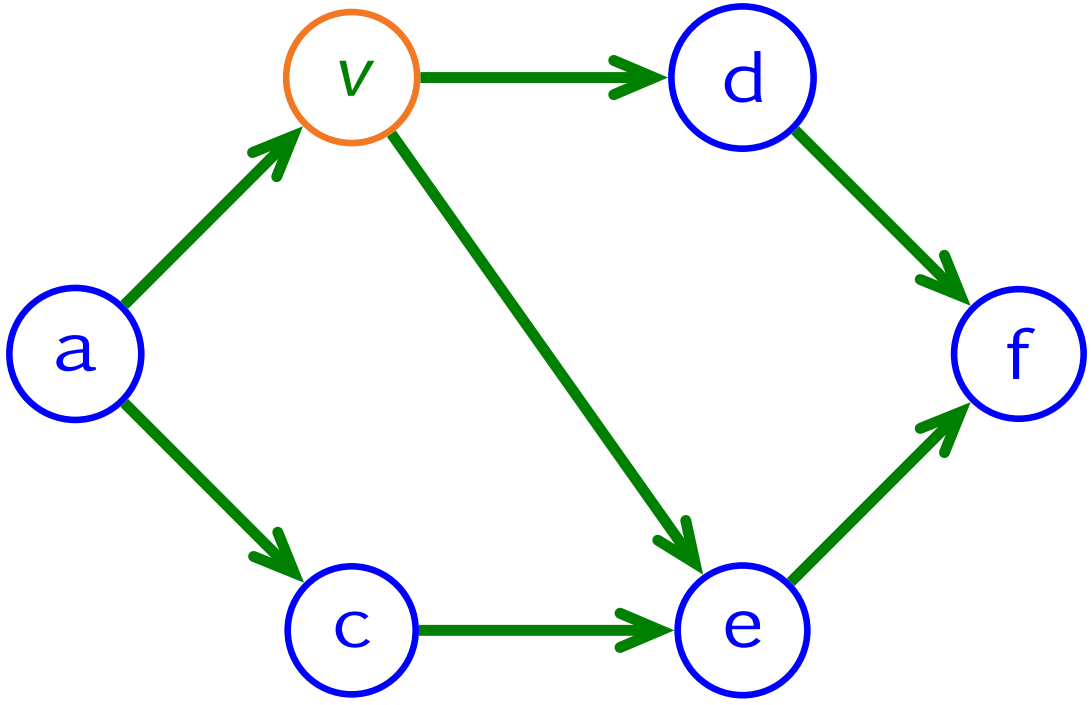


Graus de entrada e saída

grau de entrada de v = n° arcos com final v

grau de saída de v = n° arcos com início v

Exemplo: v tem grau de entrada 1 e de saída 2



Algumas convenções

Grafo G

Sage

Algumas convenções

Grafo G		Sage
$G.V$	cj. dos vértices	`G.vertices()`

Algumas convenções

Grafo G		Sage
$G.V$	cj. dos vértices	`G.vertices()`
$G.v.out$	vs. apontados por v	`G.neighbors_out(v)`

Algumas convenções

Grafo G		Sage
$G.V$	cj. dos vértices	`G.vertices()`
$G.v.out$	vs. apontados por v	`G.neighbors_out(v)`
$G.v.in$	vs. apontando v	`G.neighbors_in(v)`

Algumas convenções

Grafo G		Sage
$G.V$	cj. dos vértices	`G.vertices()`
$G.v.out$	vs. apontados por v	`G.neighbors_out(v)`
$G.v.in$	vs. apontando v	`G.neighbors_in(v)`
$G.v.adj$	vs. adjacentes a v	`G.neighbors(v)`

Algumas convenções

Grafo G		Sage
$G.V$	cj. dos vértices	`G.vertices()`
$G.v.out$	vs. apontados por v	`G.neighbors_out(v)`
$G.v.in$	vs. apontando v	`G.neighbors_in(v)`
$G.v.adj$	vs. adjacentes a v	`G.neighbors(v)`

Em fórmulas, sempre que G estiver claro:

n é o número de vértices

Algumas convenções

Grafo G		Sage
$G.V$	cj. dos vértices	`G.vertices()`
$G.v.out$	vs. apontados por v	`G.neighbors_out(v)`
$G.v.in$	vs. apontando v	`G.neighbors_in(v)`
$G.v.adj$	vs. adjacentes a v	`G.neighbors(v)`

Em fórmulas, sempre que G estiver claro:

n é o número de vértices

m é o número de arestas ou arcos

Número de arcos

Quantos arcos, no máximo, tem um digrafo com n vértices?

Número de arcos

Quantos arcos, no máximo, tem um digrafo com n vértices?

A resposta é $n \times (n-1) = \Theta(n^2)$.

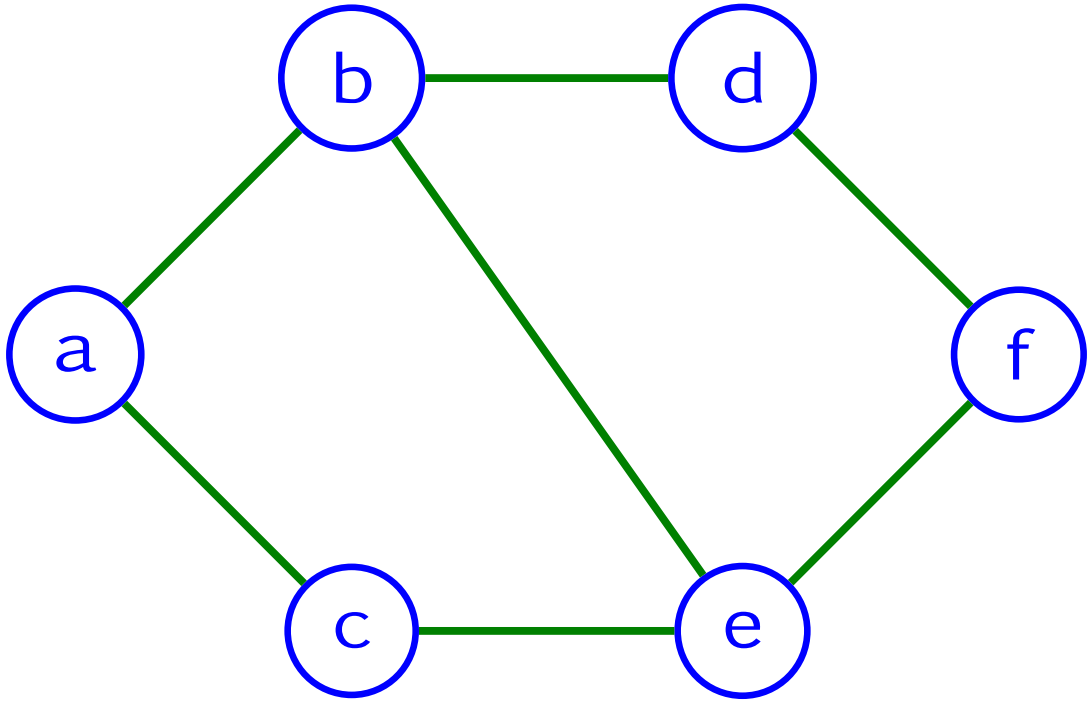
digrafo completo = todo par ordenado de vértices distintos é arco

digrafo esparso = tem "poucos" arcos, p. ex. $m = \mathcal{O}(n \log(n))$

digrafo denso = tem "muitos" arcos, p. ex. $m = \Omega(n^2/\log(n))$

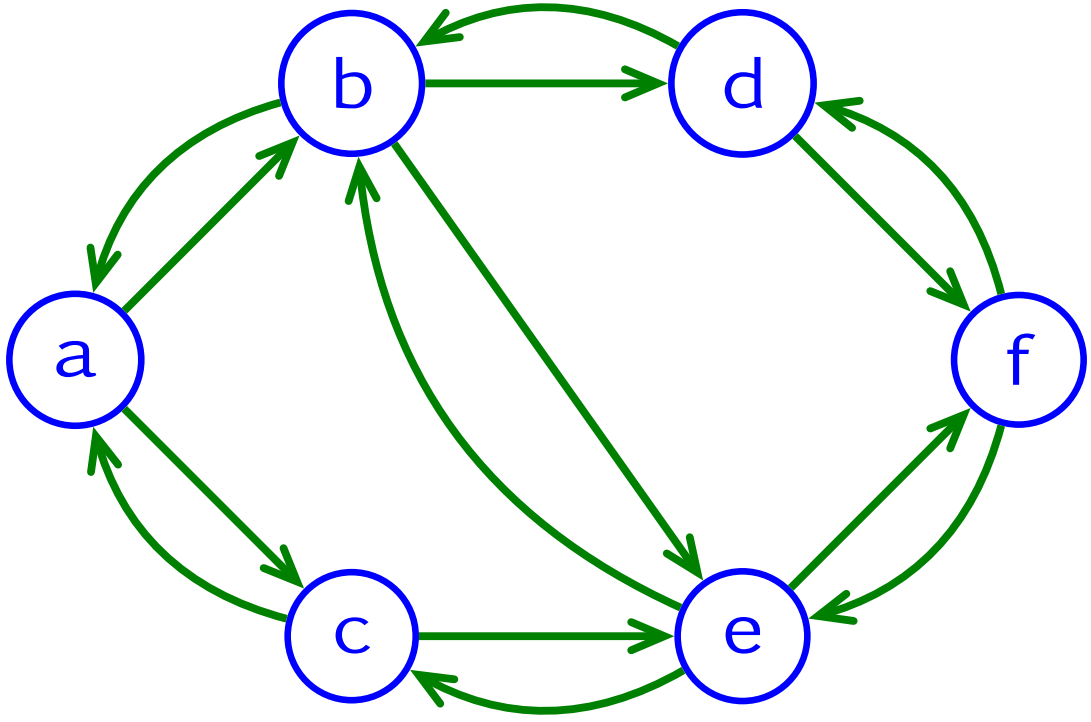
Grafos

Um grafo tem um conjunto de vértices e um de arestas, onde cada aresta é incidente a dois vértices, suas pontas.



Grafo como digrafo

Um grafo pode ser pensado como sendo um digrafo simétrico; nesse caso, cada aresta corresponde a um par de arestas anti-paralelas.

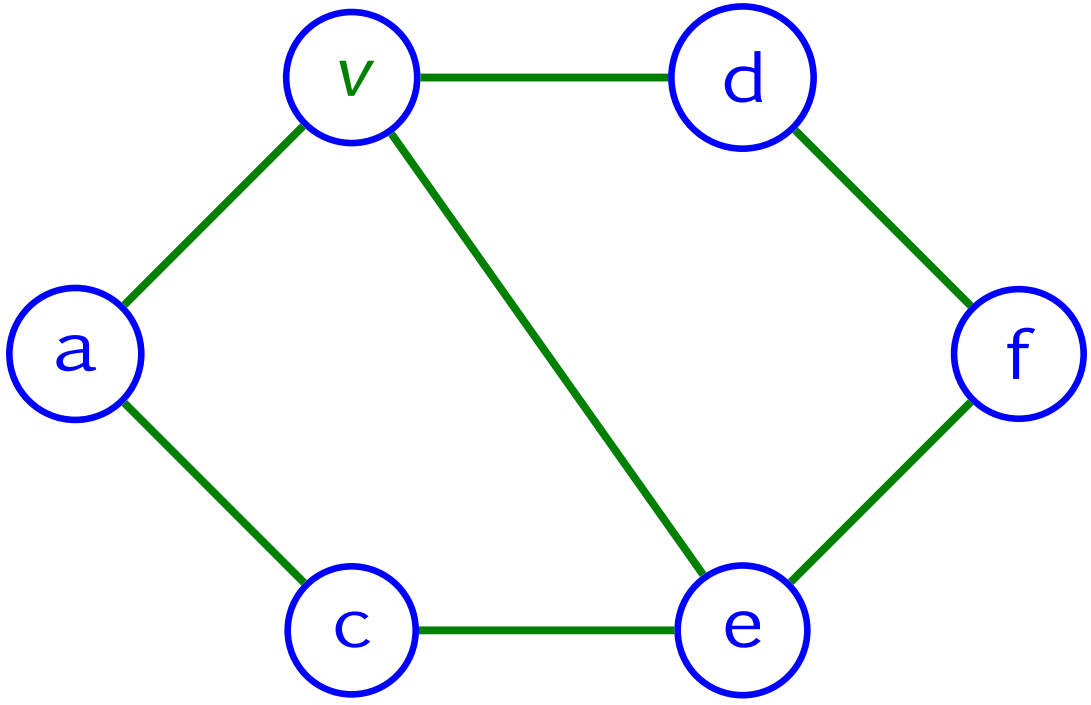


Graus de vértices

Em um grafo

grau de v = número de arestas com ponta em v

Exemplo: v tem grau 3



Número de arestas

Quantas arestas, no máximo, tem um grafo com n vértices?

Número de arestas

Quantas arestas, no máximo, tem um grafo com n vértices?

A resposta é $n \times (n-1)/2 = \Theta(n^2)$

grafo completo = todo par não-ordenado de vértices distintos é aresta

O que se faz com grafos?

Grafos podem ser vistos com ED, generalizando outras EDs ligadas.

O que se faz com grafos?

Grafos podem ser vistos com ED, generalizando outras EDs ligadas.

Ou podem ser vistos como objetos de uma classe, definida por métodos que proporcionam:

O que se faz com grafos?

Grafos podem ser vistos com ED, generalizando outras EDs ligadas.

Ou podem ser vistos como objetos de uma classe, definida por métodos que proporcionam:

- Dado um vértice e um arco,
  o vértice é ponta do arco? Qual?

O que se faz com grafos?

Grafos podem ser vistos com ED, generalizando outras EDs ligadas.

Ou podem ser vistos como objetos de uma classe, definida por métodos que proporcionam:

- Dado um vértice e um arco,
  o vértice é ponta do arco? Qual?
- Dado um arco, quais são suas pontas?

O que se faz com grafos?

Grafos podem ser vistos com ED, generalizando outras EDs ligadas.

Ou podem ser vistos como objetos de uma classe, definida por métodos que proporcionam:

- Dado um vértice e um arco,
  o vértice é ponta do arco? Qual?
- Dado um arco, quais são suas pontas?
- Iteradores

O que se faz com grafos?

Grafos podem ser vistos com ED, generalizando outras EDs ligadas.

Ou podem ser vistos como objetos de uma classe, definida por métodos que proporcionam:

- Dado um vértice e um arco,
  o vértice é ponta do arco? Qual?
- Dado um arco, quais são suas pontas?
- Iteradores
  - Para cada vértice...

O que se faz com grafos?

Grafos podem ser vistos com ED, generalizando outras EDs ligadas.

Ou podem ser vistos como objetos de uma classe, definida por métodos que proporcionam:

- Dado um vértice e um arco,
  o vértice é ponta do arco? Qual?
- Dado um arco, quais são suas pontas?
- Iteradores
  - Para cada vértice...
  - Dado um vértice v,
    para cada arco saindo de v...

No Sage

Classes

Graph

e

DiGraph

No Sage

Classes

Graph

e

DiGraph

tudo pronto! Vai ser nosso modelo.

Neste curso

Neste curso

Vamos mostrar EDs para grafos em C, para descrever algoritmos em detalhe.

Neste curso

Vamos mostrar EDs para grafos em C, para descrever algoritmos em detalhe.

Vamos usar o jargão de POO quando for conveniente.

Neste curso

Vamos mostrar EDs para grafos em C, para descrever algoritmos em detalhe.

Vamos usar o jargão de POO quando for conveniente.

Vamos lidar com as limitações do C e olhar para o Sage para comparar o que se pode fazer numa linguagem mais rica.

Vértices, arcos e arestas

Vértices, arcos e arestas

Vértices serão representados por objetos do tipo `Vertex`.

Vértices, arcos e arestas

Vértices serão representados por objetos do tipo `Vertex`.

Implementação ingênua: Os vértices de um digrafo são 0, 1, . . . , `V`-1.

Vértices, arcos e arestas

Vértices serão representados por objetos do tipo `Vertex`.

Implementação ingênua: Os vértices de um digrafo são 0, 1, . . . , `V`-1.

Arcos e arestas serão descritos pela ED adotada.

EDs mais populares

- matriz de adjacência (pouco usada)

EDs mais populares

- matriz de adjacência (pouco usada)
- vetor de listas de adjacência (mais usada, com variações)

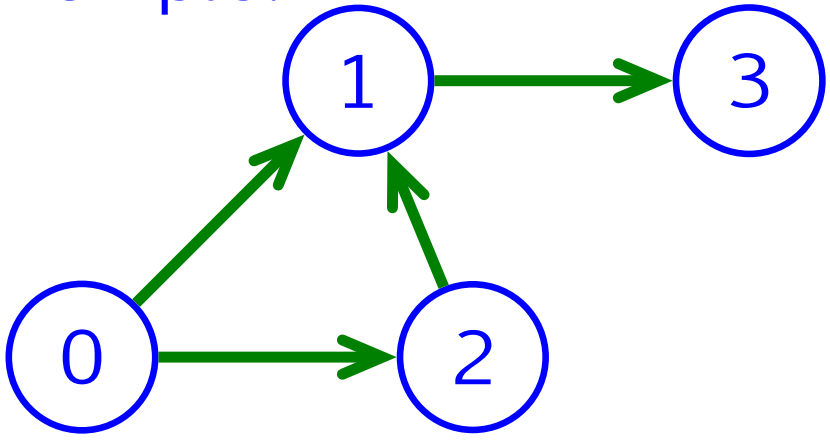
Matriz de adjacência de digrafo

Matriz de adjacência de um digrafo tem linhas e colunas indexadas por vértices:

$adj[v][w]$ = 1 se $v \longrightarrow w$ é um arco

$adj[v][w]$ = 0 em caso contrário

Exemplo:



	0	1	2	3
0	0	1	1	0
1	0	0	0	1
2	0	1	0	1
3	0	0	0	0

Consumo de espaço: $\Theta(n^2)$ fácil de implementar

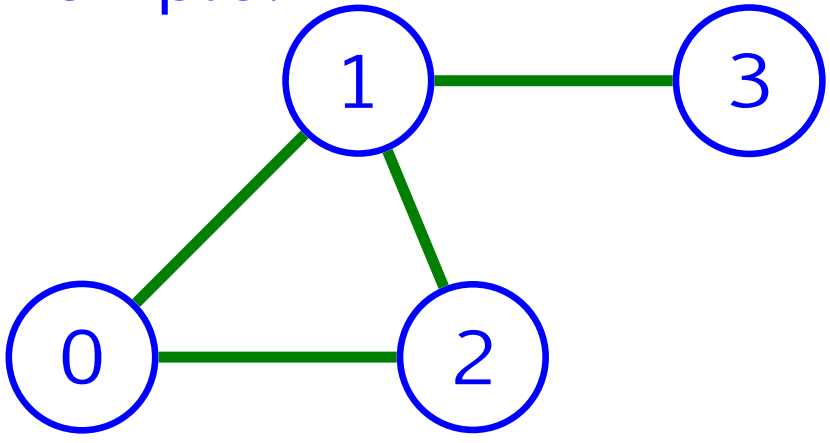
Matriz de adjacência de grafo

Matriz de adjacência de um grafo tem linhas e colunas indexadas por vértices e é simétrica:

$adj[v][w]$ = 1 se v-w é uma aresta

$adj[v][w]$ = 0 em caso contrário

Exemplo:



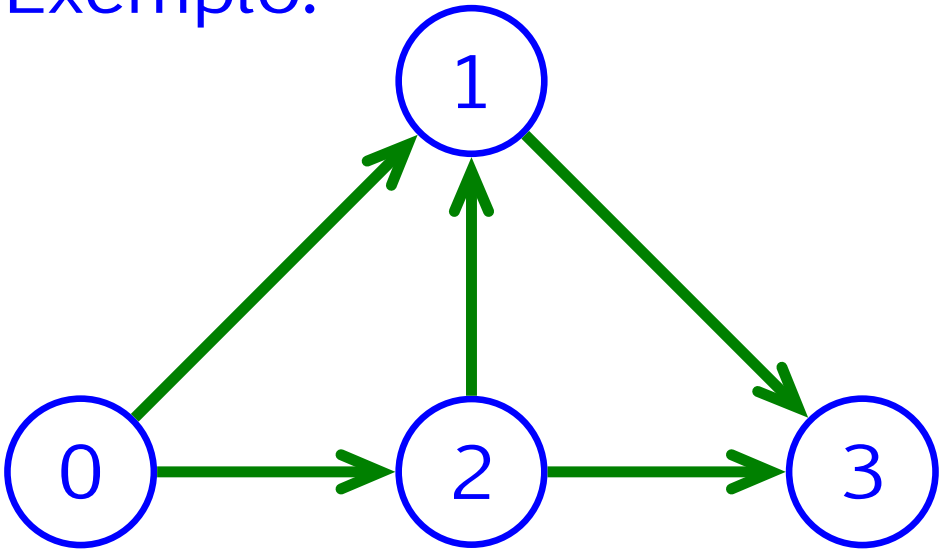
	0	1	2	3
0	0	1	1	0
1	1	0	1	1
2	1	1	0	1
3	0	1	1	0

Consumo de espaço: $\Theta(n^2)$ fácil de implementar

Vetor de listas de adjacência

Na representação de um digrafo através de listas de adjacência tem-se, para cada vértice v, uma lista dos vértices que são vizinhos v.

Exemplo:



0: 1, 2
1: 3
2: 1, 3
3:

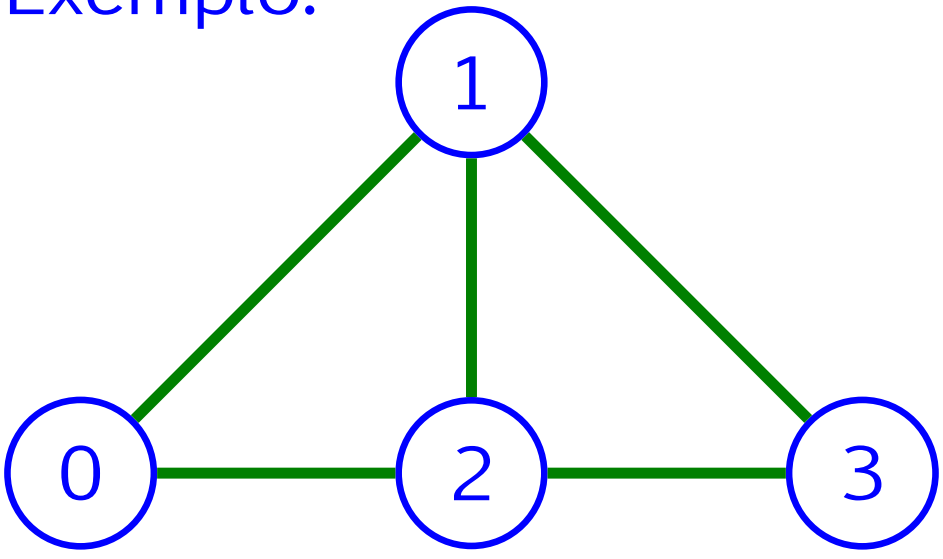
Consumo de espaço: $\Theta(n+m)$ (linear)

Manipulação eficiente

Listas de adjacência para grafos

Como se fosse um digrafo simétrico: para cada vértice v, uma lista dos vértices que são pontas de arestas incidentes a v

Exemplo:



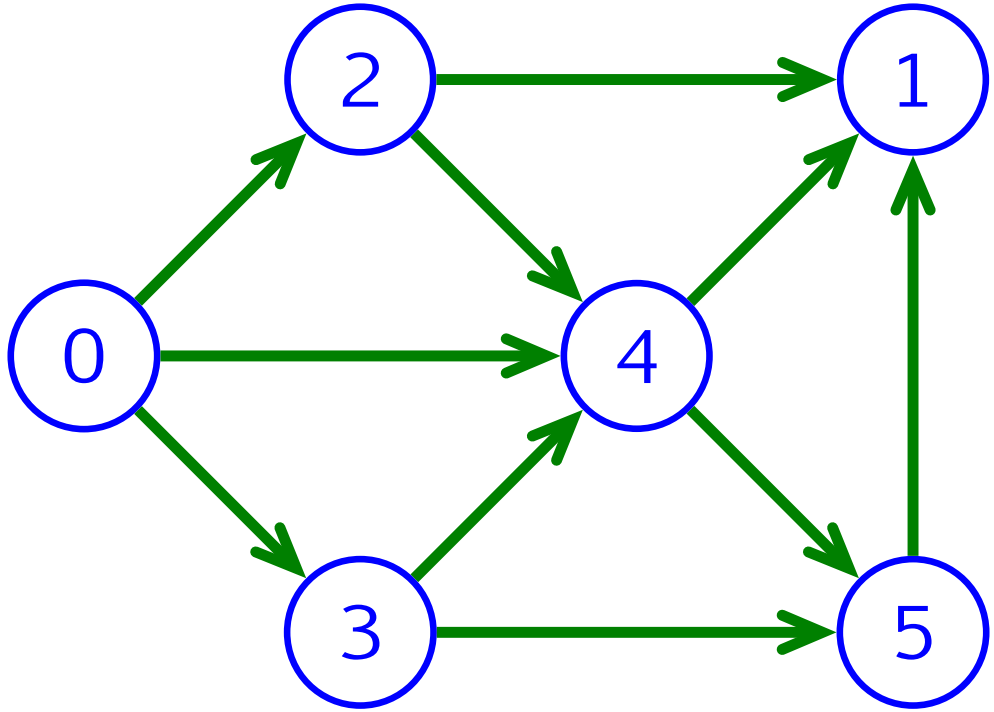
0: 1, 2
1: 3, 0, 2
2: 1, 3, 0
3: 1, 2

Consumo de espaço: $\Theta(n+m)$ (linear)

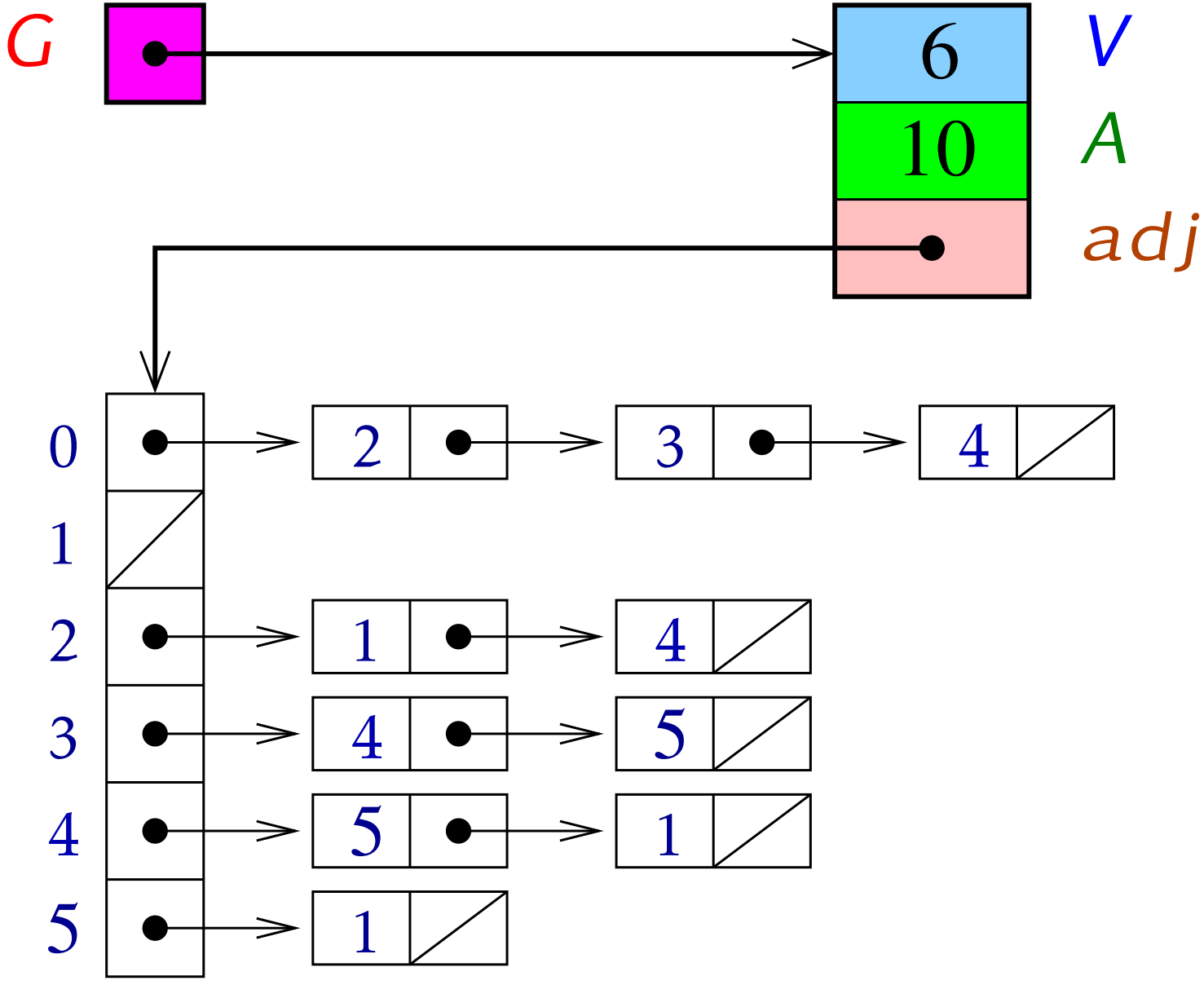
Manipulação eficiente

Digrafo

Digraph G



A estrutura de dados



Referências

Em S e PF a implementação dessas EDs em C é bem detalhada. O comportamento dos algoritmos nessas duas estruturas é um pouco diferente, e os textos os comparam.
As implementações são ingênuas: structs `Graph` e `Digraph` contendo o número de vértices, de arestas e um apontador para a representação.

Dá para ser mais esperto?

Dá para ser mais esperto?

Idéias?

Dá para ser mais esperto?

Idéias?

SAGE descreve 4 estruturas.

Dá para ser mais esperto?

Idéias?

SAGE descreve 4 estruturas.

A primeira tarefa vai ser entender essas estruturas mais a fundo.

E para desenhar?

E para desenhar?

Sage sabe desenhar jogando para o matplotlib, Graphviz e LaTeX.

E para desenhar?

Sage sabe desenhar jogando para o matplotlib, Graphviz e LaTeX.

O pacote Graphviz (disponível para Linux e outros) tem programas que leem uma descrição de um grafo ou digrafo e desenham, com vários algoritmos alternativos.

E para desenhar?

Sage sabe desenhar jogando para o matplotlib, Graphviz e LaTeX.

O pacote Graphviz (disponível para Linux e outros) tem programas que leem uma descrição de um grafo ou digrafo e desenham, com vários algoritmos alternativos.

Para o LaTeX, é gerado um código para o Tikz.

E para desenhar?

Sage sabe desenhar jogando para o matplotlib, Graphviz e LaTeX.

O pacote Graphviz (disponível para Linux e outros) tem programas que leem uma descrição de um grafo ou digrafo e desenham, com vários algoritmos alternativos.

Para o LaTeX, é gerado um código para o Tikz.

Não é tópico deste curso, mas algoritmos para desenhar grafos são objeto de pesquisa atual.