Teoria dos Grafos
Aula 2

Aula passada

- Logística
- Objetivos
- Grafos, o que são?
- Formando pares

Aula de hoje

- Mais problemas
- Definições
- Algumas propriedades
- Representando grafos: Matriz e lista

Objetivos da Disciplina

- Grafos como ferramenta de modelagem
- abstração de problemas reais
- Algoritmos eficientes em grafos para resolver problemas

Abordagem?

- Estudo de problemas reais
- Construção de algoritmos eficientes
- complexidade de algoritmos
- Técnicas para construção de algortimos

Grafo

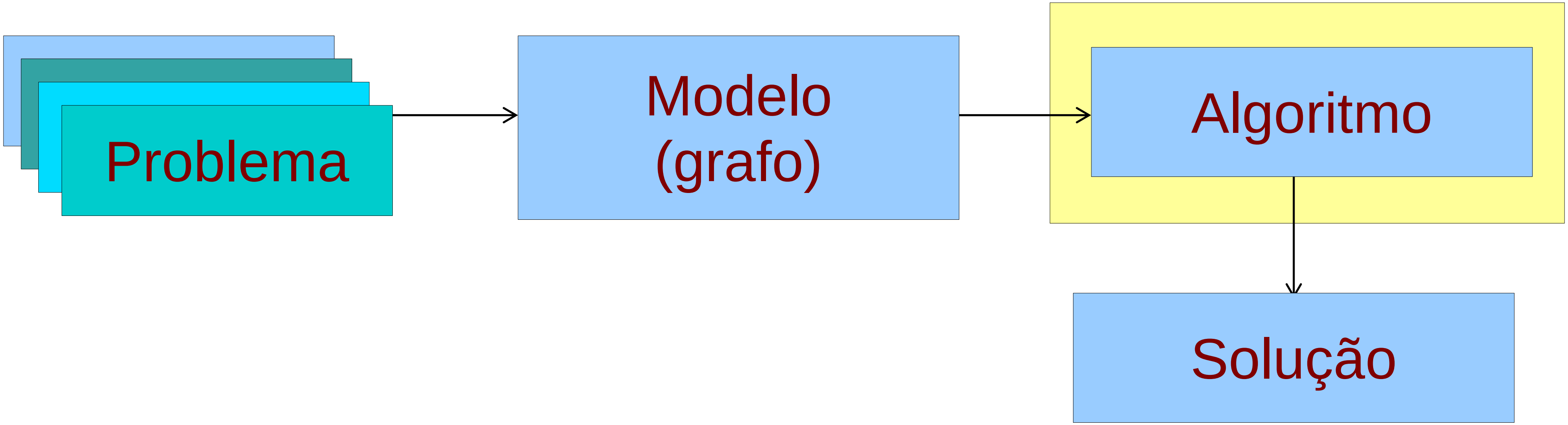
- Abstração que permite codificar relacionamentos entre **pares** de objetos

objetos → vértices do grafo

relacionamentos → arestas do grafo

Exemplos?

Poder da Abstração



- Muitos problemas resolvidos com o mesmo **algoritmo** (solução) em cima da abstração!

Pesquisando no Facebook

- Milhões de pessoas (profiles)
- Profiles interligados via relacionamentos declarados

- **Problema 1:** Como saber se duas pessoas estão "conectadas" através de uma sequência de relacionamentos?

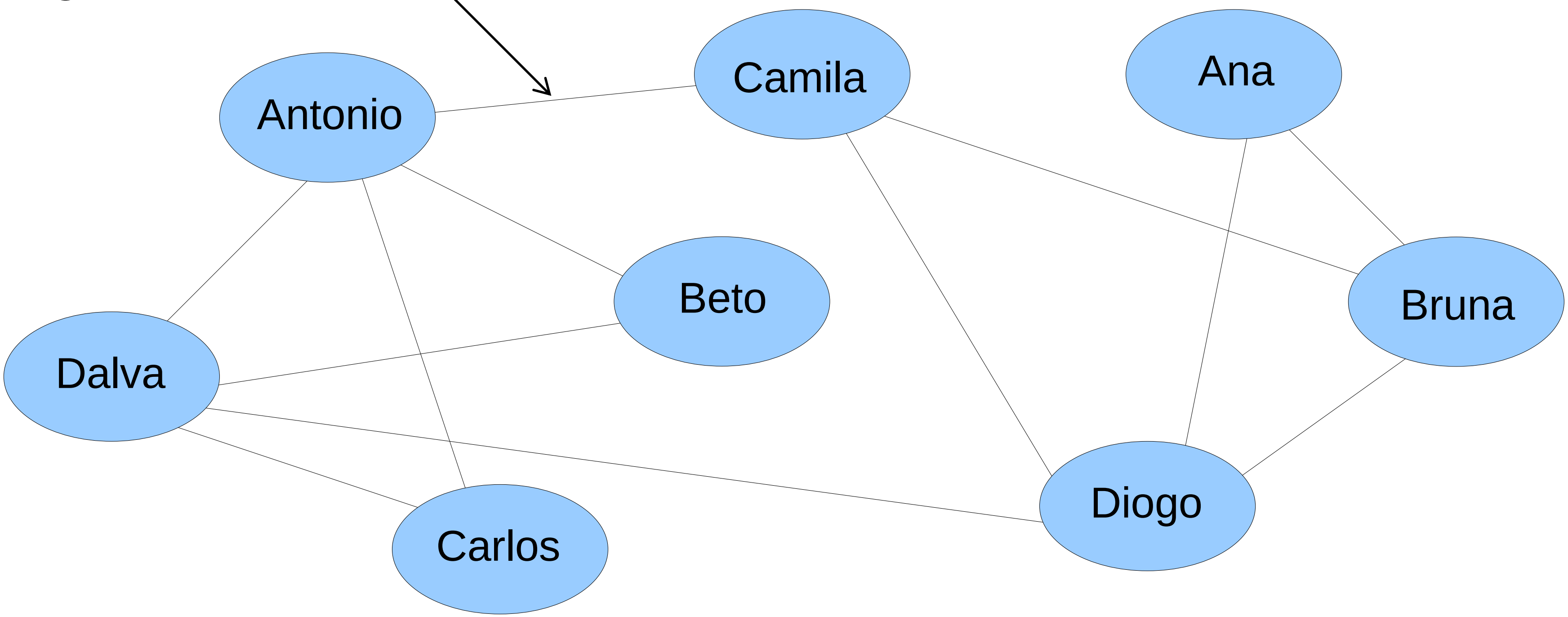
- **Problema 2:** Qual é o menor caminho entre duas pessoas?

Facebook resolve os dois problemas!

Pesquisando no Facebook

- Como abstrair o problema (via grafos)?
- Objeto: profiles (pessoas)
- Relacionamento: relacionamentos declarados

Antonio é
"amigo" da Camila

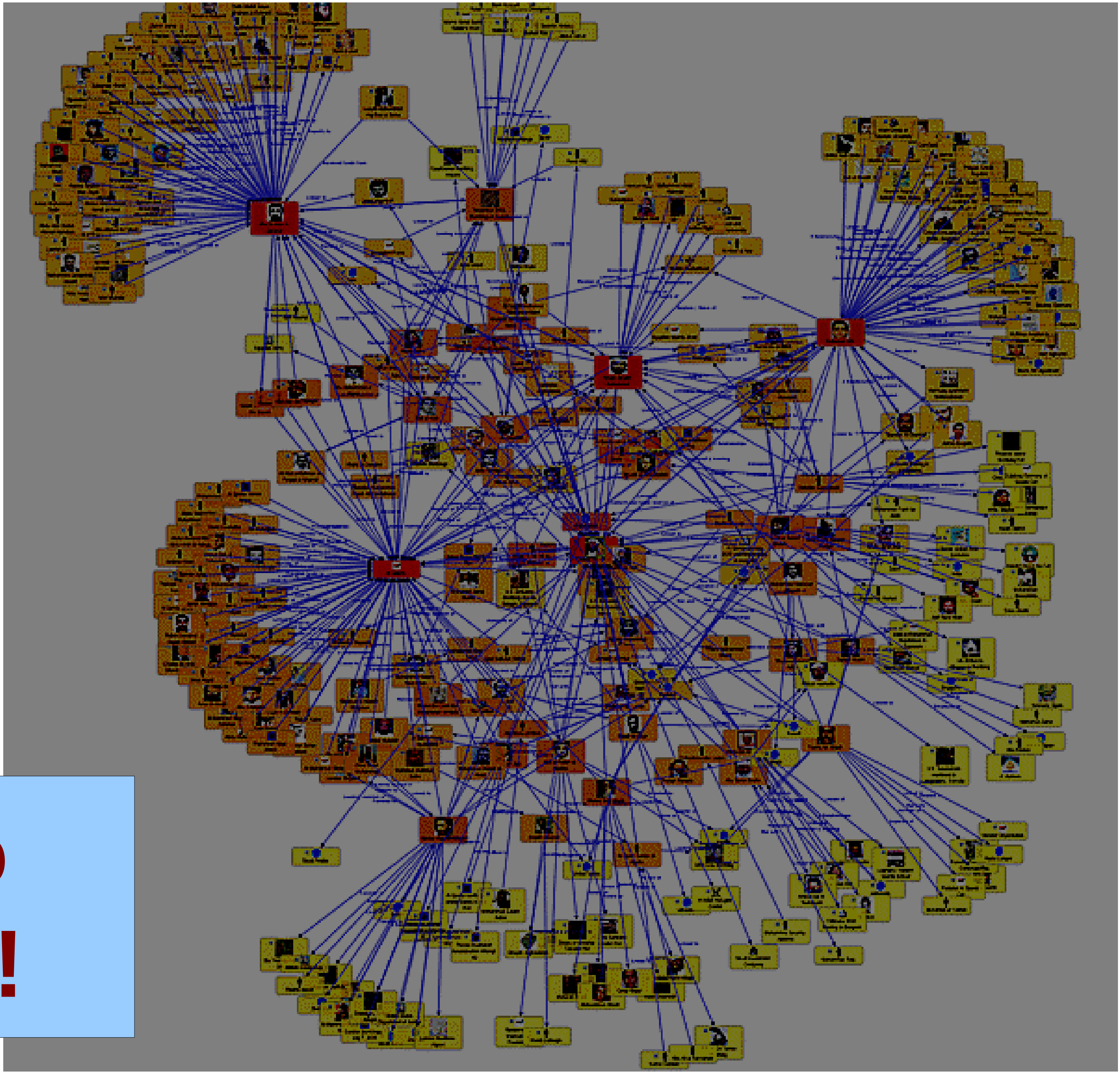


- Carlos e Ana: Conectados? Menor caminho?

Pesquisando no Facebook

- Como FB resolve o problema?
- milhões de profiles e relacionamentos!

Algoritmo (eficiente)!



Viagem entre Cidades

- Cidades brasileiras
- Estradas entre cidades

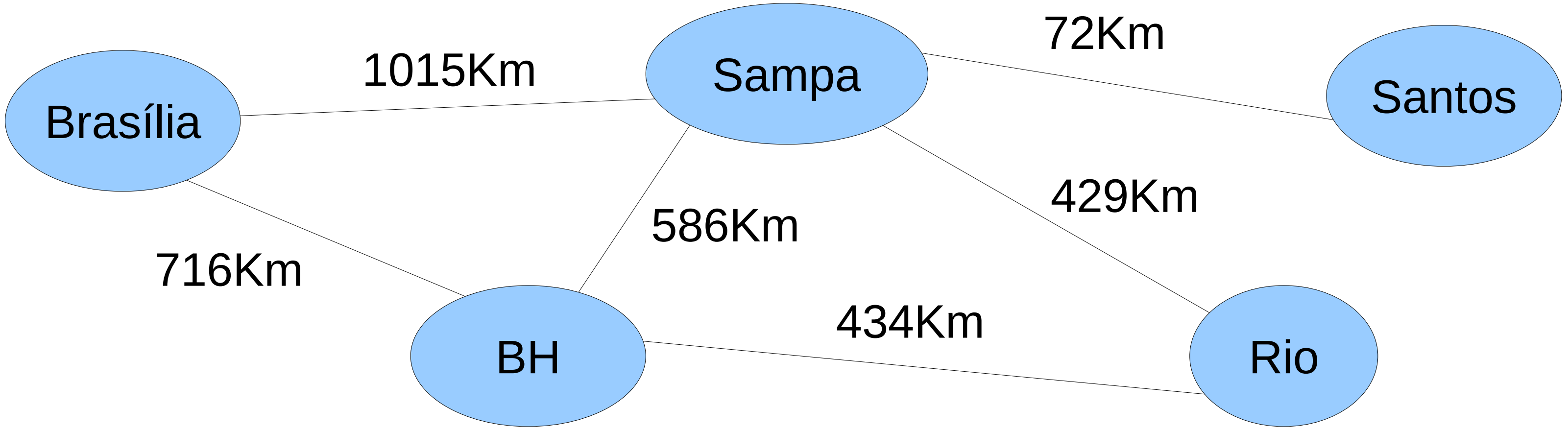
- **Problema 1:** Como saber se duas cidades estão "conectadas" por estradas?

- **Problema 2:** Qual é o menor (melhor) caminho entre duas cidades?

Viagem entre Cidades

- Como abstrair o problema (via grafos)?

Abstração parecida!



Algoritmo parecido!
(algumas variações)

Grafo

- Abstração que permite codificar relacionamentos entre pares de objetos

Como representá-lo formalmente?

↓

Conjuntos!

- Conjuntos de objetos e de pares relacionados

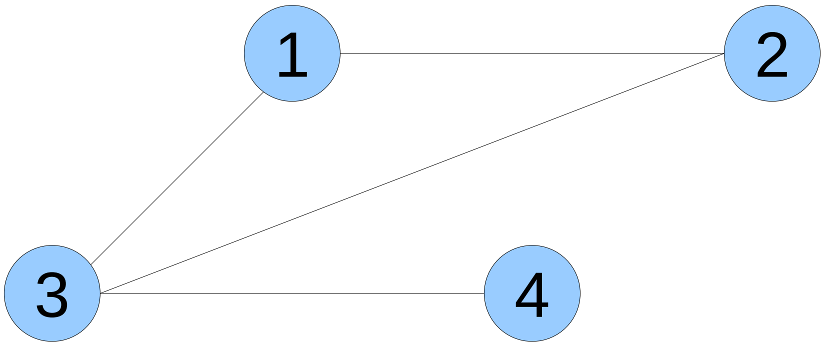
Grafo

- Grafo G = (V, E)
- V = conjunto de objetos
- chamaremos de vértices ou nós
- E = conjunto de pares relacionados
- chamaremos de arestas
- par não ordenado: (a,b) == (b,a)
- Exemplo: G = (V, E)
- V = {1, 2, 3, 4}
- E = {(1,2), (1,3), (2,3), (3,4)}

Representação Gráfica

- Desenho de G (o que vimos até agora)
- representação gráfica dos conjuntos

- Exemplo: G = (V, E)
- V = {1, 2, 3, 4}
- E = {(1,2), (1,3), (2,3), (3,4)}



Adjacência e Incidência

- Vértices adjacentes são vértices “vizinhos”
- mais precisamente...
- Dado grafo G= (V, E)
- Dois vértices *a* e *b* são **adjacentes** se existe *e* = (*a*, *b*) no conjunto E
- Aresta *e* é **incidente** aos vértices *a* e *b*
- Exemplo: G = (V, E)
- V = {1, 2, 3, 4}
- E = {(1,2), (1,3), (2,3), (3,4)}
    - 4 e 1 são adjacentes?
    - 3 e 2 são adjacentes?

Vértices e Arestas

- Número de vértices de um grafo
- n = |V|

cardinalidade
do conjunto

- Número de arestas de um grafo
- m = |E|
- Dado G = (V, E)
- Menor número de arestas de G?

zero!

- Maior número de arestas de G?
  - número de pares não ordenados em um conjunto de *n* = |V| objetos

$$\binom{n}{2} = \frac{n(n-1)}{2} \leq n^2$$

Grau

- Grau de um vértice *v*
- número de vértices adjacentes a *v*
- função grau(*v*)

- Exemplo: G = (V, E)
- V = {1, 2, 3, 4}
- E = {(1,2), (1,3), (2,3), (3,4)}

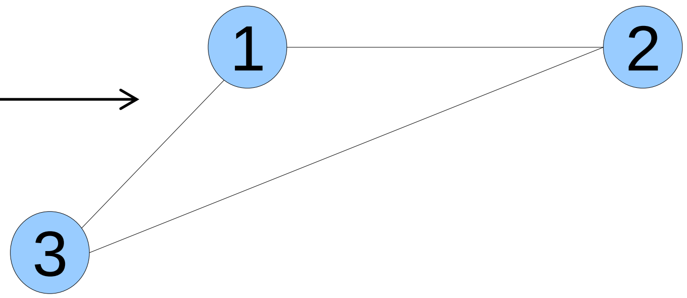
- grau(1) = ?
- grau(4) = ?

- Dado G = (V, E)
- Grau mínimo de um vértice? $\longrightarrow$ **zero!**
- Grau máximo de um vértice? $\longrightarrow$ ***n - 1***

Grafo Regular

- Todos os vértices têm mesmo grau
- no caso de *r*-Regular, grau é *r*

- Exemplo: G é 2-regular, n = 3
- V = {1, 2, 3}
- E = {(1,2), (1,3), (2,3)}



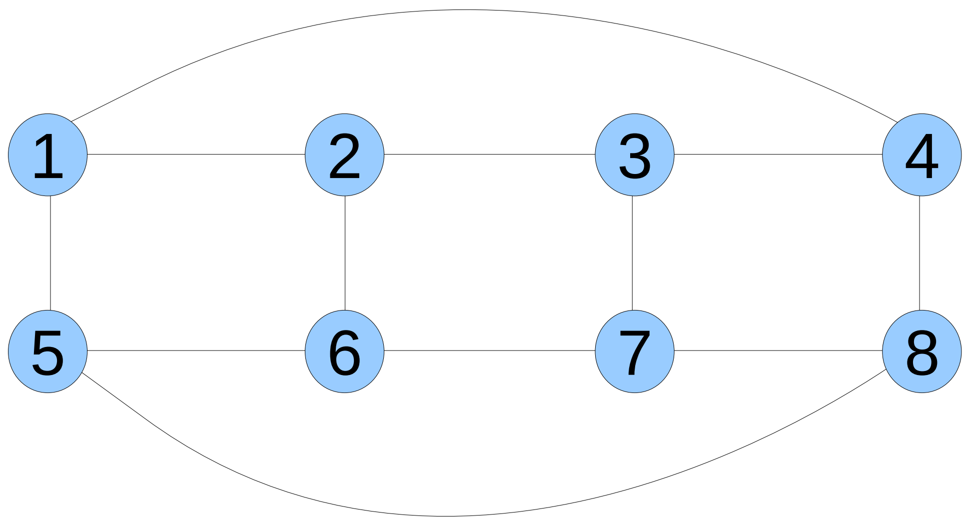
- Dado G = (V, E), e G r-regular
- Quantas arestas tem G?

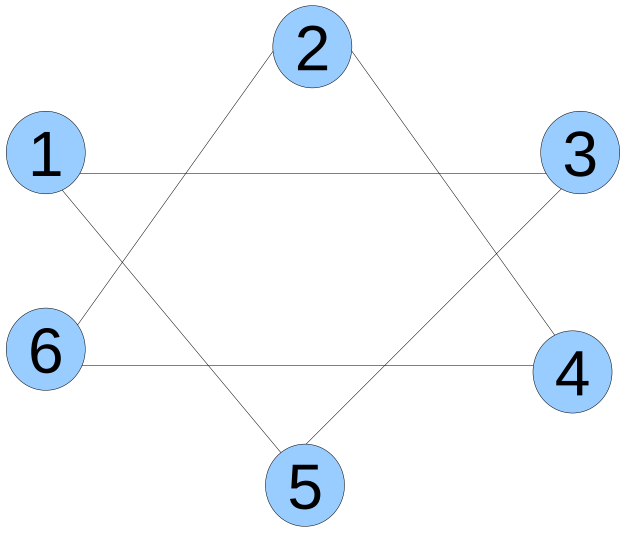
$$| E | = \frac{nr}{2}$$

Por que?

Grafo Regular

- São regulares?



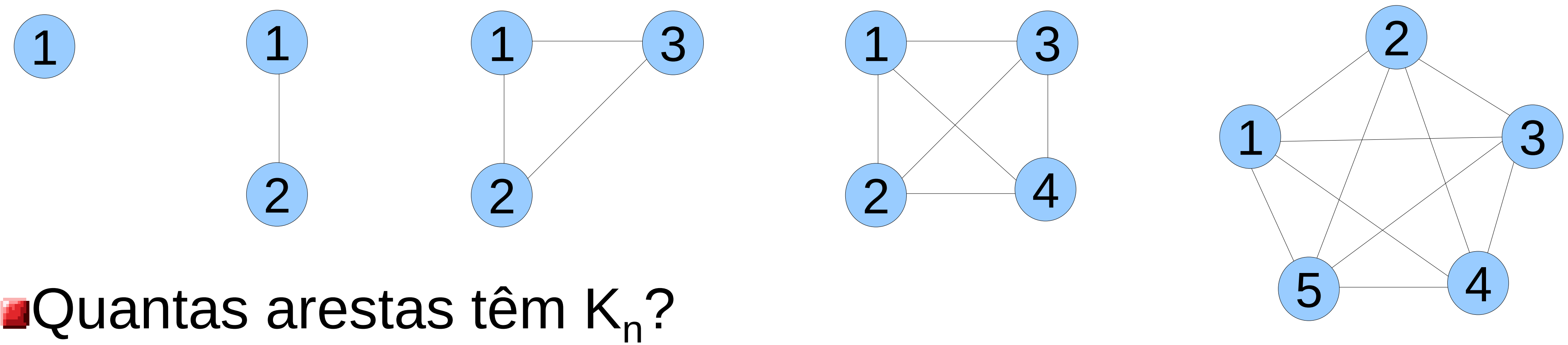


- É possível ter qualquer combinação de *n* e *r*?

Grafo Completo

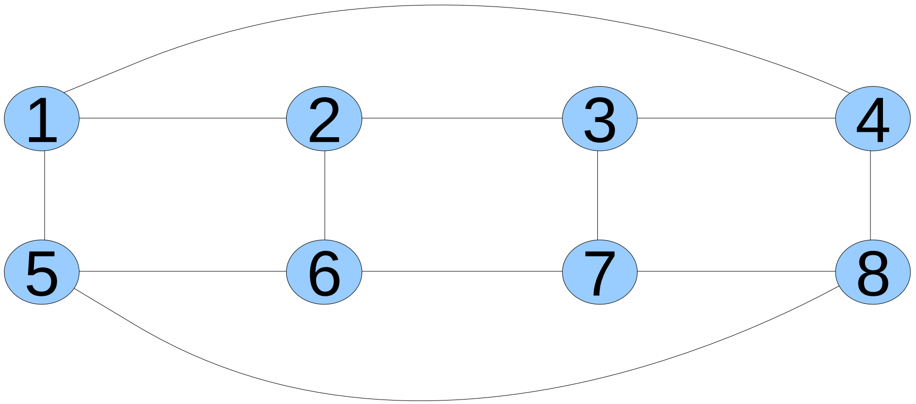
- Aresta presente entre cada par de vértices
- todos os vértices tem grau máximo
- Notação de grafo completo
- K_n onde *n* é o número de vértices
- Exemplos

K_1 K_2 K_3 K_4 K_5



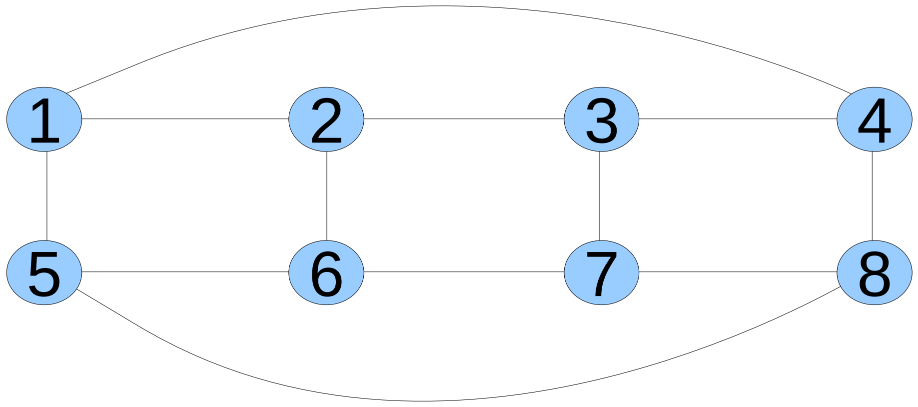
- Quantas arestas têm K_n?

Caminho



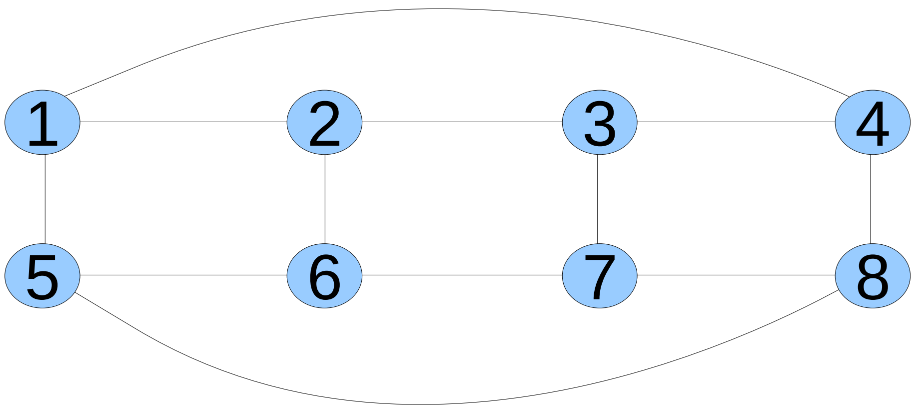
- Como definir “caminho” em um grafo?
- Ex. caminho entre 1 e 7?
- Caminho entre dois vértices
- sequência de vértices conectados por arestas
- Caminho entre v_1 e v_k
- sequência v_1, ..., v_k, tal que: $(v_i, v_{i+1}) \in E$ $i = 1, \dots k-1$
- Ex. caminho entre 1 e 7?
- 1, 2, 3, 7 $\longrightarrow$ (1,2), (2,3), (3,7)
- 1, 5, 6, 2, 6, 7 é caminho?

Caminho Simples



- Vértices do caminho são distintos
- não há “voltas”
- Ex. caminho entre 1 e 7?
- 1, 5, 6, 2, 6, 7 ← não é caminho simples
- 1, 5, 8, 7 ← é caminho simples
- Comprimento do caminho
- número de arestas que o forma
- Dado G = (V, E)
- qual é o menor caminho simples entre dois vértices?
- qual é o maior?

Ciclo



- Caminho simples que começa e termina no mesmo vértice
- $v_1 = v_k$
- Ex. ciclo em 5?
- 5, 1, 2, 3, 7, 6, 5
- 5, 1, 4, 8, 5
- Comprimento do ciclo
- número de arestas que o forma
- Dado G = (V, E)
- qual é o maior ciclo?
- n vértices, ciclo hamiltoniano

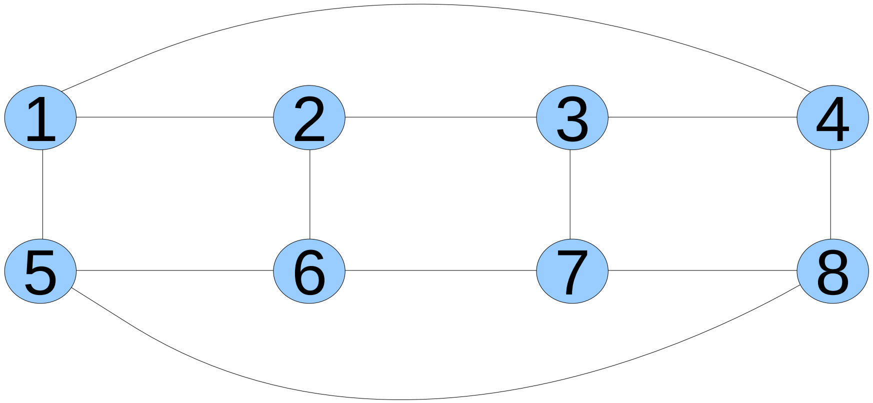
Subgrafo

- Um grafo que é “parte” de outro grafo
- mais precisamente...

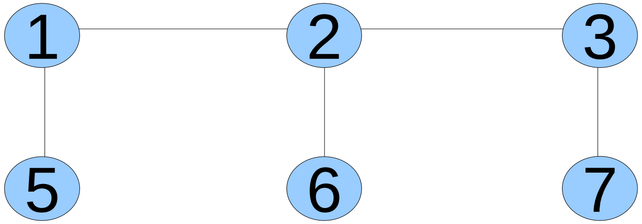
- Dado G = (V, E)
- G' = (V', E') é subgrafo de G se
- $V' \subseteq V$ e $E' \subseteq E$

Subgrafo Exemplo

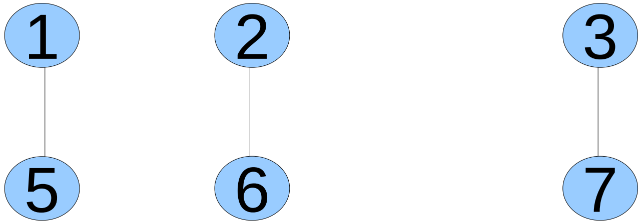
- Dado G = (V, E)



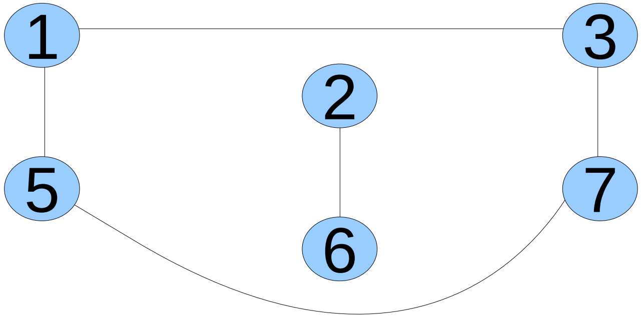
É subgrafo de G?



É subgrafo de G?



É subgrafo de G?



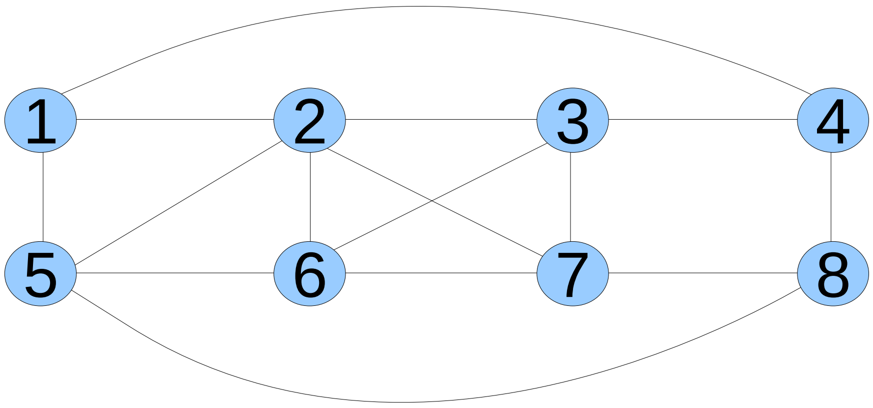
Clique

- Um grafo completo “dentro” de outro grafo
- mais precisamente...

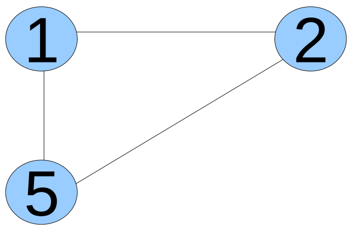
- Dado G = (V, E)
- G' = (V', E') é um *clique* de G se
- G' é subgrafo de G
- G' é um grafo completo

Clique Exemplo

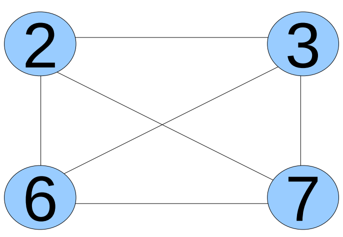
- Dado G = (V, E)



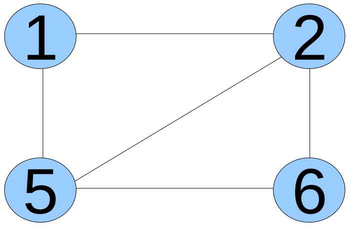
É clique de G?



- Qual é o maior clique de G?



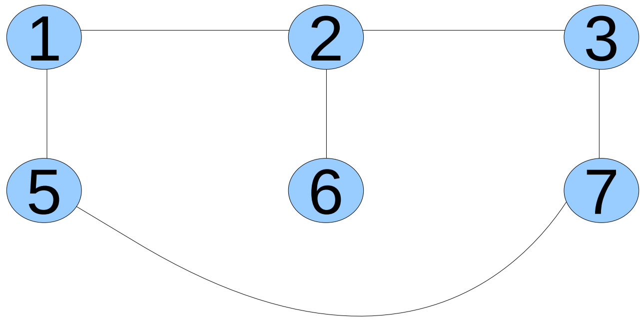
É clique de G?



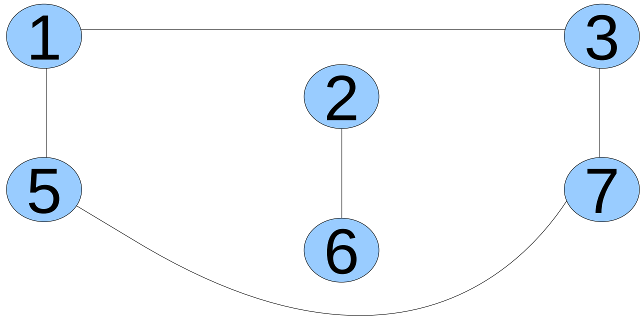
- Problema “difícil”: encontrar maior clique de um grafo

Conexo

- Grafo está “conectado”
- como definir mais precisamente?
- Grafo G=(V, E) é conexo se
- existe caminho entre qualquer par de vértices
- Caso contrário, G é desconexo



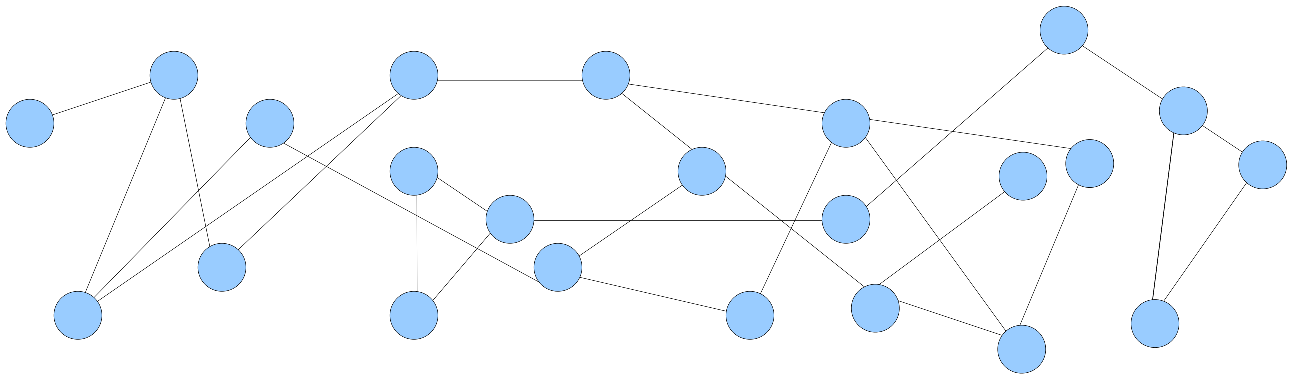
Conexo?



Conexo?

Conexo

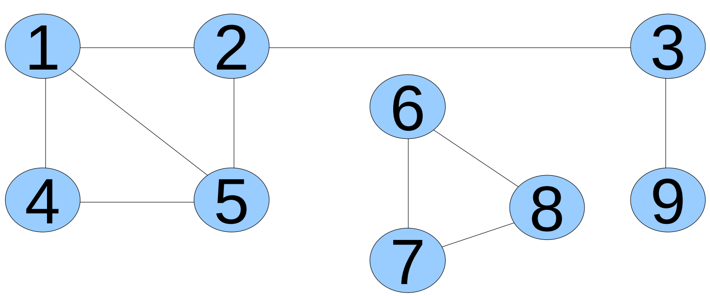
- **Problema**: Como saber se um grafo é conexo?

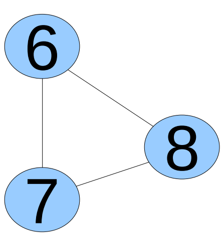


- Como você resolveria este problema?
- Veremos algoritmo (eficiente)
- Na próxima aula...

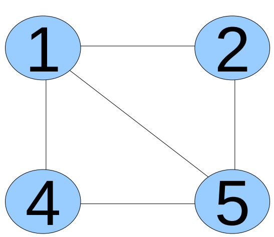
Componentes Conexos

- Maiores subgrafos “conectados” de um grafo
- mais precisamente...
- Subgrafos maximais de G que sejam conexos
- *maximal*: subconjunto que maximiza a propriedade, no caso subgrafo conexo
- Exemplo:





Componente conexo?

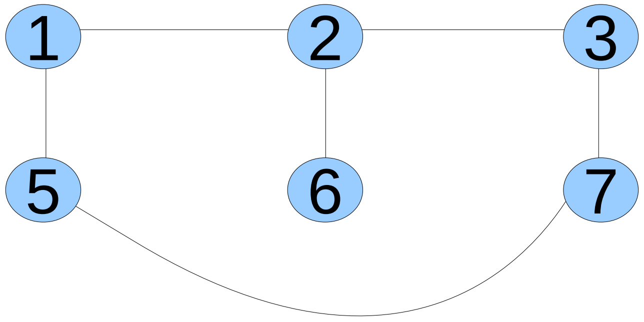


Componente conexo?

Representando Grafos

- Grafo G=(V, E)
- V = conjunto de vértices (inteiros)
- E = conjunto de arestas (pares não-ordenados)
- Exemplo
- V = {1, 2, 3, 5, 6, 7},
- E = {(1,2), (1,5), (2,3), (2,6), (3,7), (5,7)}

Representação
matemática
de grafos



Como representar
no computador?

Representando Grafos

- Como representar grafos no computador?

Estrutura de dados

- Duas estruturas fundamentais
- matriz
- lista
- Qual é a estrutura mais adequada (ou mais eficiente)?

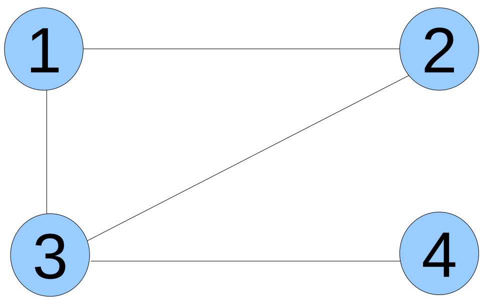
Depende do algoritmo!

Representação via Matriz

- Como representar utilizando matrizes?
- **Idéia:** associar vértices à linhas e colunas da matriz
- elemento da matriz indica se há aresta
- **Matriz de adjacência**
- Matriz *n* x *n* (*n* é número de vértices)
- $a_{ij} = 1$, se existe aresta entre vértices *i* e *j*
- $a_{ij} = 0$, caso contrário.

Matriz de Adjacência

- Exemplo



	1	2	3	4
1	0	1	1	0
2	1	0	1	0
3	1	1	0	1
4	0	0	1	0

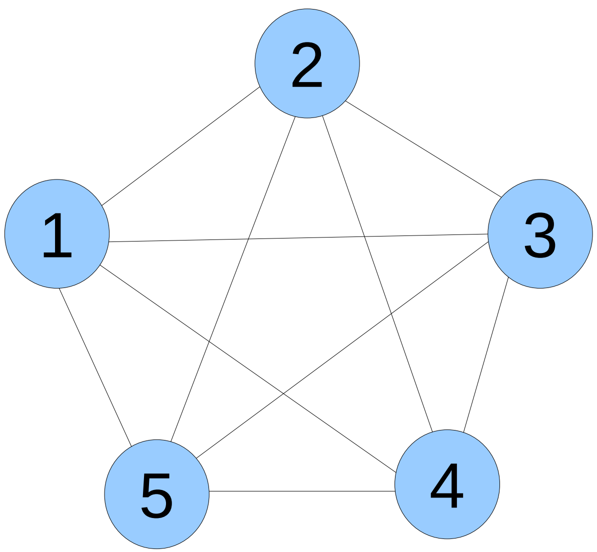
	1	2	3	4	5
1	0	1	1	0	0
2	1	0	1	0	1
3	1	1	0	1	0
4	0	0	1	0	1
5	0	1	0	1	0

?

Representação via Matriz

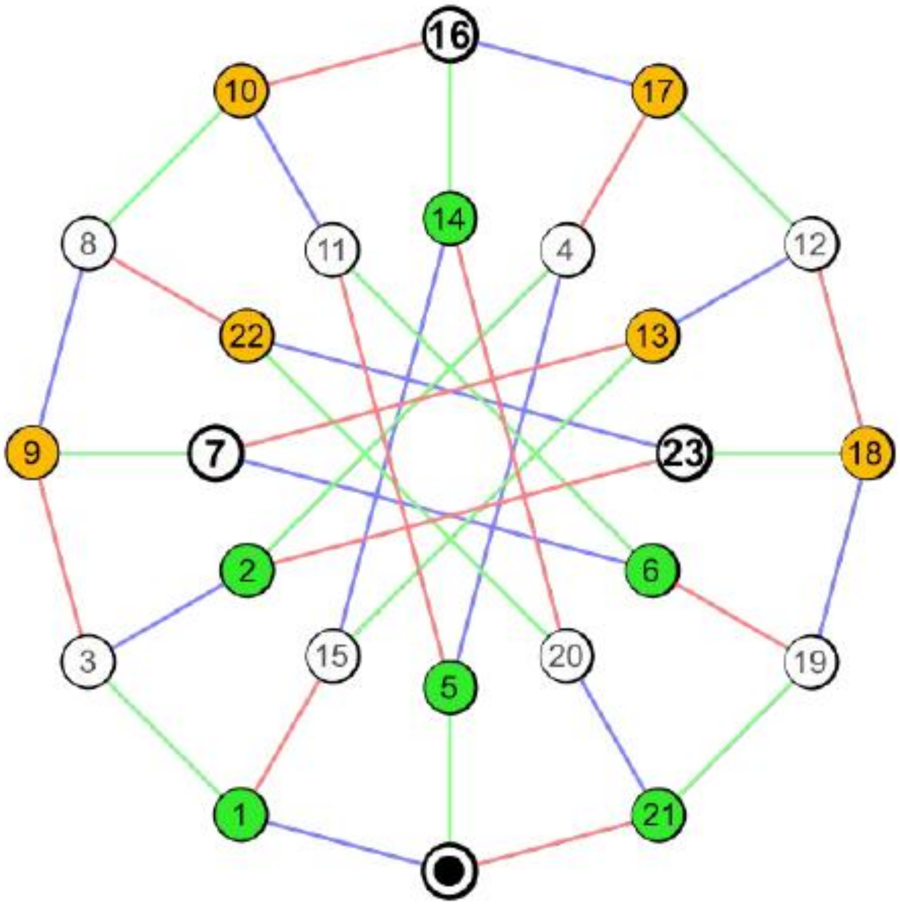
- **Matriz de adjacência**
- como fica a matriz de adjacência de um grafo completo?

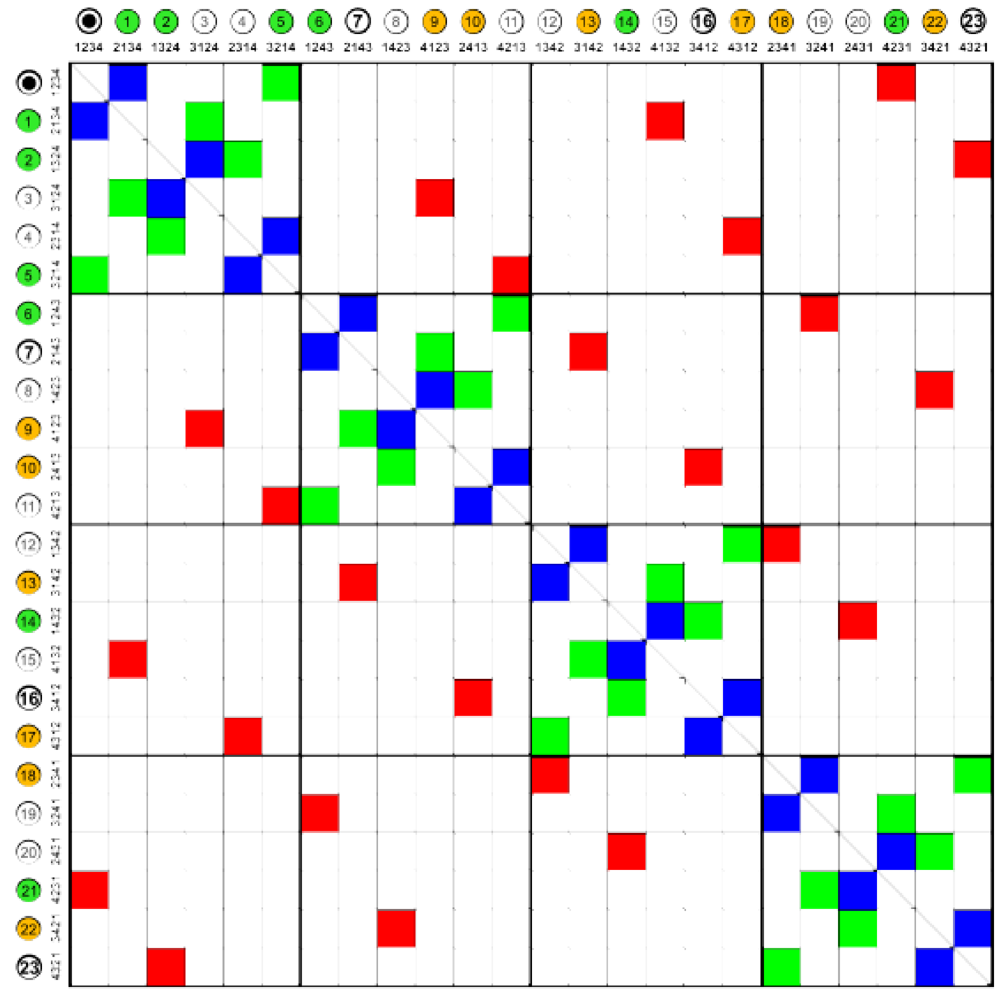
K_5



Representação via Matriz

- **Grafo de Nauru**





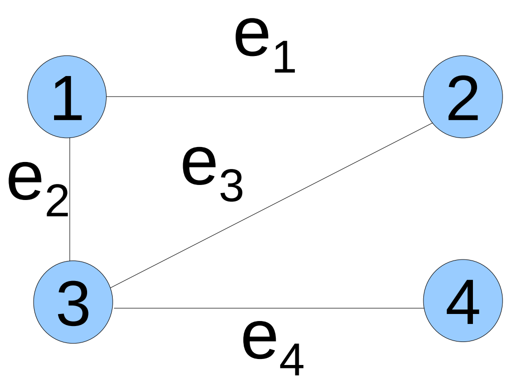
Matriz de Incidência

- **Idéia:** associar vértices às linhas e arestas às colunas
- elemento da matriz indica se aresta incide sobre o vértice

- **Matriz de incidência**
- Matriz *n* x m (*n* vértices, m arestas)
- $a_{ij} = 1$, se vértice *i* incide sobre aresta *j*
- $a_{ij} = 0$, caso contrário.

Matriz de Incidência

- Exemplo



	e_1	e_2	e_3	e_4
1	1	1	0	0
2	1	0	1	0
3	0	1	1	1
4	0	0	0	1

	e_1	e_2	e_3	e_4	e_5	e_6
1	1	1	0	0	0	1
2	1	0	1	1	1	0
3	0	1	0	1	0	0
4	0	0	0	0	1	0
5	0	0	1	0	0	1

?

Desvantagem

- Desvantagem da representação matricial?

- Considere grafos grandes e esparços
- *grande:* muitos vértices
- *esparço*: relativamente poucas arestas
- Matriz formada principalmente de zeros!

Grande consumo de memória (desnecessário)!

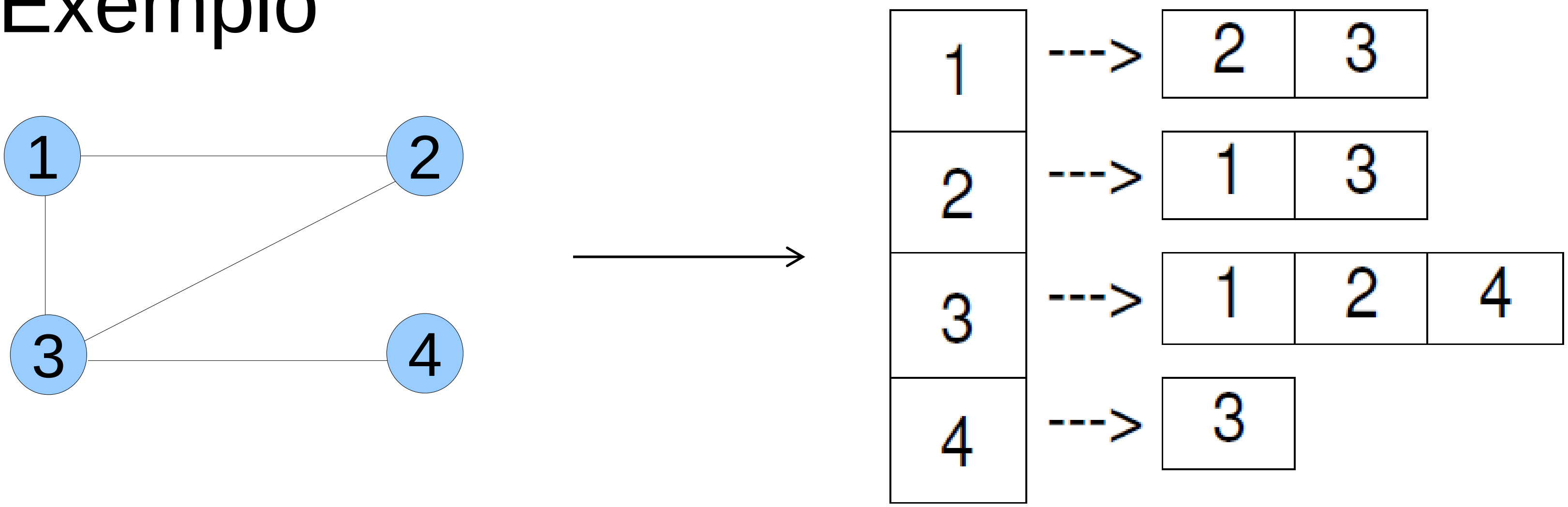
- Como resolver este problema?

Representação via Listas

- **Idéia:** associar a cada vértice uma lista de vértices adjacentes
- **Lista de adjacência**
- Vértices associados a um vetor, dimensão *n* (número de vértices no grafo)
- Cada vértice possui uma lista de vértices adjacentes

Lista de Adjacência

- Exemplo



Desvantagem

- Desvantagem da representação com lista?

- Considere grafos onde vértices tem muitos vizinhos (mas bem menos do que n)
- Listas vão ser grandes (longas)
- Problema?

Tempo de acesso! Ex. descobrir se dois vértices são vizinhos

Vantagens/Desvantagens

Tempo de execução	Matriz	Lista
Inserir aresta?	O(1)	O(1)
Remover aresta?	O(1)	$O(g_{max})$
Testar adjacência (v_1 e v_2 são vizinhos)?	O(1)	$O(g_{max})$
Listar vizinhos de v?	O(n)	$O(g_{max})$

Melhor estrutura depende do algoritmo!