Aula 2 — Complexidade de Algoritmos e Notação Assintótica

- Quando estudamos algoritmos estamos preocupados com dois aspectos

  - Algoritmo é correto?
  - Quanto ele consome de determinado recurso? (tempo, memória, E/S, etc)

    - Melhor caso
    - Pior caso
    - Caso médio

    (Assintoticamente! Normalmente, entradas pequenas não importam tanto)

- No passado: avaliação empírica (implementar algoritmo, fazer medições, plotar gráficos).
  Qual o problema dessa abordagem?

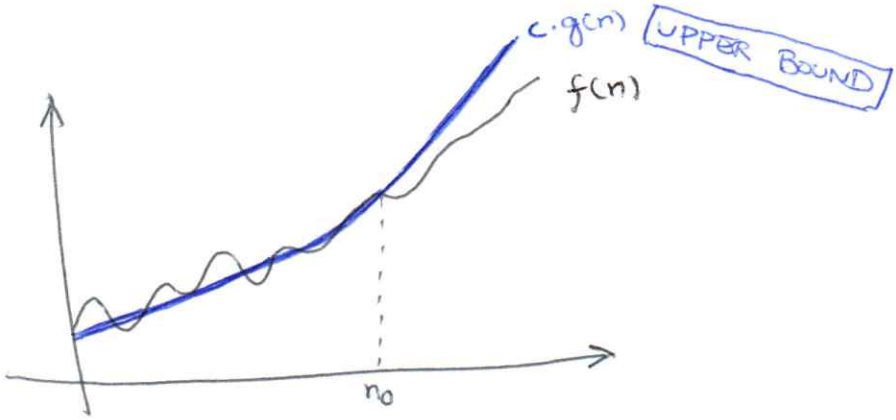
- Hoje: preferência por avaliação analítica, sempre que possível.
  - Em vez de máquina específica, modelo matemático de computador
  - Conjunto de instruções com certos tempos de execução
  - Simplificação razoável: toda instrução leva tempo constante = 1.
  - Determinar número total de instruções em função de tamanho da entrada.

- Desejamos calcular um <u>limite superior</u> para o número de passos de um algoritmo, que funcione <u>assintoticamente</u>. Para isso, usamos a notação O.

Sejam $f(n)$, $g(n)$ funções do tipo $f: \mathbb{N} \to \mathbb{R}$
$g: \mathbb{N} \to \mathbb{R}$

> Dizemos que $f(n) \in O(g(n))$ se
> $\exists k > 0$, n_0 t.q. $\forall n > n_0$, $0 \leq f(n) \leq k \cdot g(n)$

<u>Intuição:</u>



$f(n) \in O(g(n))$

<u>Observação:</u> $O(g(n))$ é um <u>conjunto</u> de funções que satisfazem aquela propriedade.
No entanto, usamos $f(n) = O(g(n))$
Nunca $O(g(n)) = f(n)$!

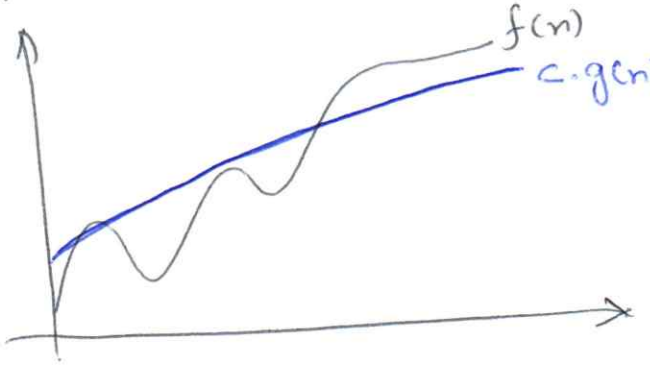
<u>Exemplo:</u> $2n^2 + 3n = O(n^2)$, pois $\forall n > \ldots$. temos
$0 \leq 2n^2 + 3n \leq \ldots n^2$

<u>História:</u> - P. Bachmann, 1892.
"Die Analytische Zahlentheorie"
- Vejam artigo de Knuth na página do curso

- Às vezes desejamos calcular um limite inferior para o número de passos de qualquer algoritmo que resolva um certo problema. Para isso, é conveniente usar a notação Ω.

> Dizemos que $f(n) \in \Omega(g(n))$ se
> $\exists k>0,\ n_0$ tq. $\forall n>n_0,\ 0 \leq k.g(n) \leq f(n)$

Intuição:



LOWER BOUND

Exemplo:

$2n^2+3n = \Omega(n)$, pois $\forall n > \ldots$ temos

$0 \leq \ldots n \leq 2n^2+3n$

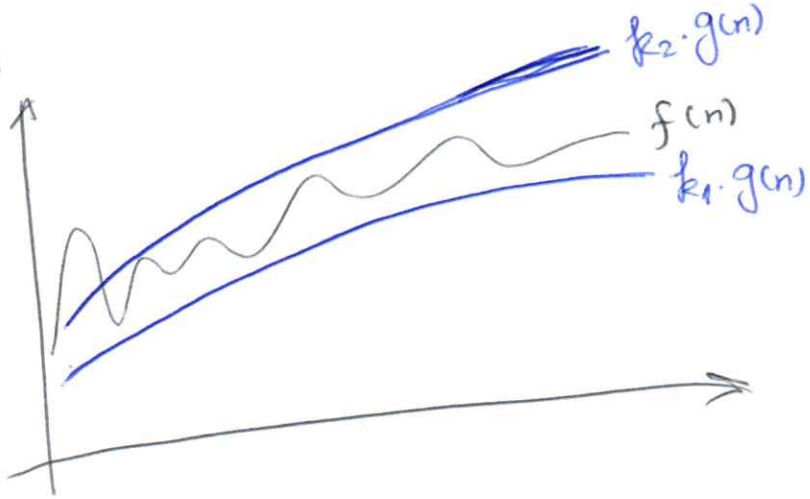
- Enquanto o melhor upper bound conhecido e o melhor lower bound conhecido para um problema são diferentes, há espaço para:
  - ou encontrar um algoritmo mais eficiente
  - ou melhorar o lower bound

Dizemos que $f(n) \in \Theta(g(n))$ se

$\exists k_1 > 0,\ k_2 > 0,\ n_0$ tq. $\forall n > n_0$,

$$k_1 \cdot g(n) \leq f(n) \leq k_2 \cdot g(n)$$

Intuição:



TIGHT BOUND

- As notações O, Θ, Ω são as mais comuns. Temos ainda algumas outras que às vezes podem ser úteis.

little-oh, não confundir com big-oh

Dizemos que $f(n) \in o(g(n))$ se

$\forall k > 0\ \exists n_0$ tq $\forall n > n_0,\ 0 \leq f(n) \leq k \cdot g(n)$

Note que $2n \in o(n^2)$

Mas, $2n^2 \notin o(n^2)$

Exemplo: $\frac{1}{n} \in o(1)$

Se $f(n) = o(g(n))$ então $\lim_{n \to \infty} \frac{f(n)}{g(n)} = 0$.

Dizemos que $f(n) = \omega(g(n))$ se

$\forall k > 0,\ \exists n_0$ tq. $\forall n > n_0,\ 0 \leq kg(n) \leq f(n)$

Note que $2n^3 \in \omega(n^2)$

Mas, $2n^2 \notin \omega(n^2)$

Se $f(n) = \omega(g(n))$ então $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$.

- Por analogia:

$f(n) = O(g(n))$	$\approx$	$a \leq b$
$f(n) = \Omega(g(n))$	$\approx$	$a \geq b$
$f(n) = \Theta(g(n))$	$\approx$	$a = b$
$f(n) = o(g(n))$	$\approx$	$a < b$
$f(n) = \omega(g(n))$	$\approx$	$a > b$

- Mais uma notação útil:

Dizer que $f(n) \in \tilde{O}(g(n))$ é o mesmo que dizer $f(n) \in O(g(n) \log^k g(n))$ para algum k.

- Nem todas funções podem ser comparadas assintoticamente. Por exemplo:

$$f(n) = n^{1+\sin n}$$

$$g(n) = n$$

- Comportamento assintótico de fatorial:

Pode-se provar que $n! = o(n^n)$

$$n! = \omega(2^n)$$

$$\log(n!) = \Theta(n \log n)$$

Dica:

$$n! = \sqrt{2\pi n}\left(\frac{n}{e}\right)^n \left(1 + \Theta\left(\frac{1}{n}\right)\right)$$

(aproximação de Stirling)

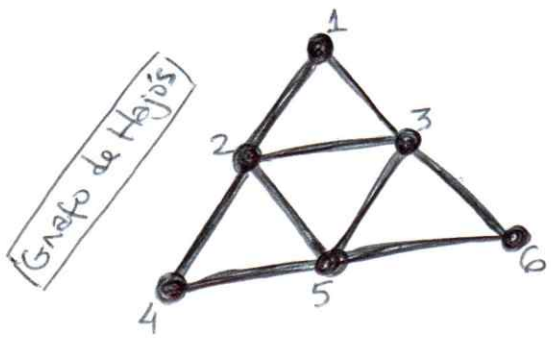
Iniciação à Teoria dos Grafos

- Um **grafo** $G(V,E)$ é um conjunto finito não vazio V e um conjunto de pares não ordenados de elementos distintos de V.

Exemplo: $G(V,E)$ em que $V = \{1,2,3,4,5,6\}$

e $E = \{(1,2), (1,3), (2,3), (2,4), (2,5), (3,5), (3,6), (4,5), (5,6)\}$.

- É bastante útil visualizar um grafo por meio de sua representação geométrica. Do exemplo anterior, temos:



Grafo de Hajós

- Os elementos de V são vértices.
- Os elementos de E são arestas (edges), $e = (v,w)$
- G é trivial quando $|V| = 1$.
- Denotamos $n = |V|$, $m = |E|$.

- Duas arestas incidentes a um mesmo vértice são adjacentes.

- Aresta e deste exemplo é incidente a v e w.

- Nesse caso, vértices v e w são extremos ou extremidades da aresta e.

- Vértices v e w são denominados adjacentes

- Às vezes é útil relaxar a definição de grafos:

  - podemos permitir uma aresta do tipo $e=(v,v)$, chamada de laço.
  - podemos substituir o conjunto E por um multiconjunto, ou seja, permitir mais de uma aresta entre o mesmo par de vértices, chamadas arestas paralelas.

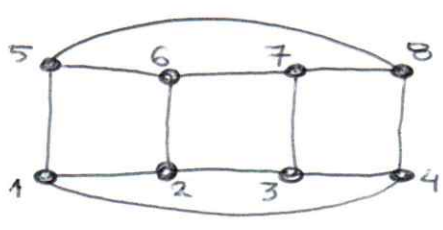
  Um grafo com arestas paralelas é um multigrafo.

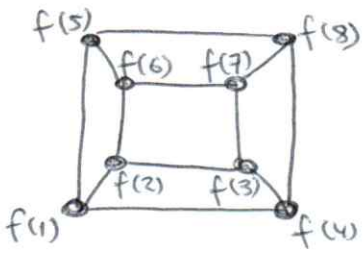
- PROBLEMA: Dadas duas representações geométricas, elas correspondem ao mesmo grafo?

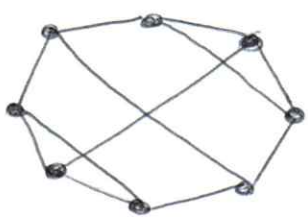
  Formalmente: Dados dois grafos $G_1(V_1,E_1)$ e $G_2(V_2,E_2)$, com $|V_1|=|V_2|=n$, existe função unívoca $f: V_1 \to V_2$, tal que $(v,w) \in E_1$ se e somente se $(f(v), f(w)) \in E_2$, para todo $v, w \in V_1$?

- Se existe um função para o problema acima, dizemos que os grafos G_1 e G_2 são isomorfos entre si.

- Esse é o problema do isomorfismo de grafos.





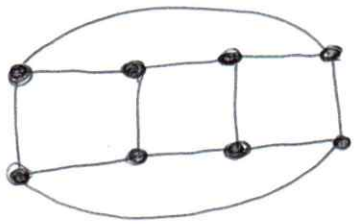


- Não é razoável resolver problema do isomorfismo de grafos por força bruta: $\Omega(n!)$
- Não se conhece algoritmo eficiente.

Mais definições:

- Define-se **grau** de um vértice $v \in V$, denotado grau(v), como sendo o número de vértices adjacentes a v.

- Um grafo é regular de grau r quando todos os seus vértices possuirem o mesmo grau r.



Exemplo: - regular de grau 3

Observe: - cada aresta incidente a 2 vértices, logo,

$$\sum_{v \in V} \text{grau}(v) = 2|E|$$

- Um vértice que possui grau zero é chamado isolado.

- Uma sequência de vértices $v_1, \dots, v_k$ tal que $(v_j, v_{j+1}) \in E$, $1 \leq j < |k|$, é denominado caminho ou passeio. Dizemos que v_1 alcança ou atinge v_k.

- A quantidade de arestas $(v_1, v_2), (v_2, v_3)$ etc é o comprimento do caminho.

- Se todos os vértices forem distintos: caminho simples ou elementar

se não houver ambiguidade, ao falar de "caminho" nos referimos a "caminho simples"...

- Se arestas forem distintas: trajeto

- Se caminho $v_1, \dots, v_k, v_{k+1}$ tq $v_1 = v_{k+1}$ e $k \geq 3$: ciclo

- Na def. anterior, se vértices distintos: ciclo simples ou elementar

...idem para "ciclo"

- Grafo sem ciclos: acíclico 

- Ciclo de comprimento 3: triângulo 

- Caminho que contém cada vértice do grafo exatamente 1 vez: Hamiltoniano
  Análogo para ciclo Hamiltoniano

- Caminho que contém cada aresta do grafo exatamente 1 vez: Euleriano
  Análogo para ciclo Euleriano

- Grafo conexo: quando há caminho entre cada par de vértices; caso contrário é desconexo. Se não possuir arestas: totalmente desconexo

Grafo conexo

Grafo desconexo

- Operações de inclusão e exclusão de vértices e arestas:

grafo → $G - e$ ← aresta : grafo obtido pela exclusão da aresta e

$G + (v, w)$: " " " inclusão de aresta (v, w)
(vértices)
* assumindo que v e w não eram adjacentes

$G - v$: " " " exclusão de vértice v e arestas a ele incidentes

$G + v$: " " " inclusão de vértice v

Note que é possível generalizar: $G + S$ ou $G - S$, em que S é conjunto de vértices ou arestas.

Teorema 2.1 (o primeiro da Teoria de Grafos! das pontes de Königsberg!)

Um grafo G conexo possui ciclo Euleriano
se e somente se
todo vértice de G possuir grau par.

Prova.

⇒ Seja C um ciclo Euleriano de G.
Cada ocorrência de $v \in C$ contribui com 2 unidades para grau de v. Cada aresta aparece exatamente 1 vez.

⇐ Particione as arestas em conjunto de ciclos simples.
(Começa selecionando um ciclo simples arbitrário, remove, repete)
Depois vai unindo arestas de ciclos que possuem um vertice em comum, até construir um ciclo Euleriano.