Complexidade de Algoritmos

Mariana Kolberg

Relembrando…

- Complexidade é também chamada esforço requerido ou quantidade de trabalho (relacionada ao tempo ou espaço).
  - Complexidade no pior caso;
  - Complexidade média.

  - É possível antecipar alguma relação entre elas?

Conceitos preliminares

- Sejam **A** o conjunto de todos os algoritmos disponíveis; **D** o conjunto de todas a entradas, o conjunto de seqüências de execuções **E** é obtido por

$$\mathbf{Exec[a]: D \rightarrow E.}$$

- O custo de uma execução é definido por

$$\mathbf{Custo: E \rightarrow R_+}$$

- Considere que o procedimento abaixo receba como entrada uma tabela com n elementos

```
procedimento max(tab:tabela)
mx<-tab[1]
para i de 2 até n faça
  se mx<tab[i] então mx<-tab[i]
fim-para
retorna-saida(mx)
fim-procedimento
```

- Qual o custo associado à execução deste algoritmo?

Conceitos preliminares

- O desempenho de um algoritmo a para uma entrada d é Desemp[a]: D → R_+ ,
  Desemp[a](d)=Custo(Exec[a](d))

- Dado um conjunto $D_n = \{d | d \in D \text{ e } tam(d) = n\}$ onde Tam: D→N retorna o tamanho de uma dada entrada d.

- A complexidade pessimista é expressa por

$$C_p[a](n) = \max\{desemp[a](d) \in R_+ \mid d \in D_n\}$$

- A complexidade média é expressa por

$$C_m[a](n) = \sum_{d \in D_n} prob(d).desemp[a](d)$$

Conceitos preliminares

- Os critérios de complexidade introduzidos consideram o desempenho do algoritmo sobre o conjunto de todas as entradas com tamanho **n**. Como critério alternativo temos:

No pior caso

$$C_p^{=}[a](n) = \max\{desemp[a](d) \in R_+ \mid tam(d) = n\}$$

No critério relaxado

$$C_p^{\leq}[a](n) = \max\{desemp[a](d) \in R_+ \mid tam(d) \leq n\}$$

Conceitos preliminares

- Imagine um algoritmo **a** cujo desempenho sobre cada entrada é o tamanho da entrada. O pior desempenho deste algoritmo com entradas até 20 é

$$C_p^{\leq}[a](20) = \max\{1, 2, 3, \ldots, 20\} = 20$$

- Considere um algoritmo a que possa receber 100 entradas de tamanho 10. Cada entrada d_j possui um desempenho r_j. O desempenho esperado, considerando distribuição uniforme das entradas, é o dado por

$$C_m[a](10) = \frac{r_1 + r_2 + \ldots + r_{100}}{100}$$

Conceitos preliminares

- É possível garantir

$$C_p^{=}[a](n) \leq C_p^{\leq}[a](n) \qquad \forall n \in N$$

$$\{desemp[a](d) \in R_+ \mid tam(d) = n\} \subseteq \{desemp[a](d) \in R_+ \mid tam(d) \leq n\}$$

Conceitos preliminares

- É possível garantir $C_p^{=}[a](n) = \max\{C_p^{\leq}[a](m) | m \leq n\}$?

  Sim, se a função desemp for monotônica, ou seja

  se $tam(d) \geq tam(d')$ então $desemp[a](d) \geq desemp[a](d')$

Comparação ente complexidades

- Um problema pode ter mais de um algoritmo para resolvê-lo. Qual deles escolher?

- De maneira geral, consideremos um conjunto **A** de algoritmos para um dado problema $\prod$.

- A cada algoritmo **a** $\in$ **A** temos associado a função de avaliação *aval.*

- Freqüentemente, interessa-nos identificar um algoritmo ótimo, ou seja, um algoritmo **o**, tal que *aval(o)* é melhor do que *aval(a)* para todo **a** $\in$ **A**.

- Quando A consiste em todos os algoritmos que resolvem $\prod$, então aval(o) é a complexidade intrínseca do problema.

Ordens assintóticas

Ordens Assintóticas

- A complexidade assintótica é definida pelo crescimento da complexidade para entradas suficientemente grandes;

- Um algoritmo **assintoticamente mais eficiente** é melhor para todas as entradas, exceto para entradas relativamente pequenas;

- Esta complexidade é chamada de **COTA**;

- As cotas são maneiras de reduzir detalhes realçando apenas os aspectos relevantes de eficência.

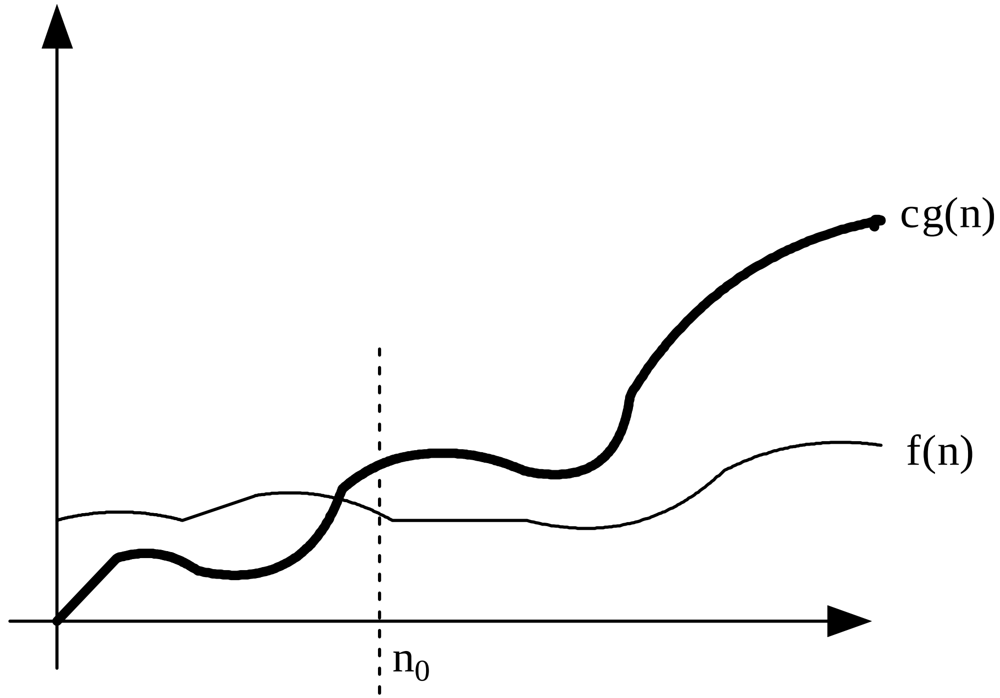
Ordens Assintóticas - Notação *O*

- A notação *O* define a cota assintótica superior.
- Ela serve para limitar superiormente uma função dentro de um fator constante.
- Exemplo : uma função quadrática $g(n)=3n^2$ cresce mais rapidamente que uma linear $f(n)=7n+13$. Logo, dizemos que f(n) é *O*(g(n)).

f(n) é *O*(g(n)) sse

$$(\exists\ c \in \Re_+)(\exists\ n_0 \in \mathbf{N})(\forall n \geq n_0)(f(n) \leq c.g(n))$$

Ordens Assintóticas - Notação O



$$(\exists\, c \in \Re_+)(\exists\, n_0 \in \mathbf{N})(\forall n \geq n_0)(f(n) \leq c.g(n))$$

Para n grande, g(n) domina f(n). Logo g(n) é CAS de f(n)

Ordens Assintóticas - Notação *O*

- Um algoritmo é $O(1)$ se o número de operações fundamentais é limitado por uma constante.
  - Faço sempre 200 operações. Qual a complexidade?

- $f(n)=O(g(n))$ significa que **algum múltiplo constante de g(n)** é um limite assintótico superior sobre f(n).

- Podemos usar a seguinte representação $f(n) \in O(g(n))$.

Ordens Assintóticas- Notação O

- Para cada um dos seguintes pares de funções **f** e **g,** verifique se é possível encontrar constantes $\mathbf{n_0}$ e **c** tais que

$$\forall n \geq n_0 \;\; f(n) \leq cg(n)$$

$$n^2 \text{ e } n^3 \log_2 n$$

$$2^5 n \text{ e } n^3$$

$$10^n n^2 \text{ e } n2^n$$

Ordens Assintóticas- Notação O

- Para cada um dos seguintes pares de funções **f** e **g,** verifique se é possível encontrar constantes $\mathbf{n_0}$ e **c** tais que

$$\forall n \geq n_0 \;\; f(n) \leq cg(n)$$

n^2 e $n^3 \log_2 n$ — $c=1$ e $n_0 = 2$

$2^5 n$ e n^3 — $c=2^5$ e $n_0 = 1$

$10^n n^2$ e $n2^n$

$$10^n n^2 \leq cn2^n$$

$$10^n n^2 > 2^{3n} n^2$$

$$2^{3n} n^2 \leq cn2^n$$

$$2^{2n} n \leq c \quad \text{Absurdo!}$$

Ordens Assintóticas - Notação O

- Pode-se afirmar que $n \log_2 n = O(n \log_{10} n)$? Se sim, mostre a prova.

Ordens Assintóticas - Notação O

- Pode-se afirmar que

$$n^2 - n = O(n^2)$$

$$n^2 + n = O(n^2)$$

$$n \log_{10} n = O(n^2)$$

$$2^{n+1} = O(2^n)$$

$$5n + 7 = O(n^2)$$