Conceito de complexidade e suas cotas

Luciano Nascimento Moreira

Adaptado dos slides da
Profª. Renata Oliveira

Conceito de complexidade e suas cotas

- A primeira medida de dificuldade de um problema se relaciona com o tempo necessário para resolvê-lo. Para um dado problema, a complexidade de tempo é uma função que relaciona o tamanho de uma instância ao tempo necessário para resolvê-la.
- Queremos saber em quanto tempo podemos resolver todas as instâncias de um problema.

Conceito de complexidade e suas cotas

- Associamos a cada instância um valor inteiro chamado TAMANHO DA INSTÂNCIA, que é uma medida de quantidade de dados de entrada, procuramos então uma função do tamanho da instância, que expresse o tempo que o algoritmo necessita para resolver aquela instância. Essa função é chamada COMPLEXIDADE DE TEMPO ou RAPIDEZ do algoritmo.
- Essa rapidez pode estar associada ao caso pessimista (PIOR CASO), otimista (MELHOR CASO), ou caso médio.

Pior Caso

- Supõe que os dados de entrada são tais que o algoritmo terá o MÁXIMO de trabalho a ser feito entre todas as instâncias de tamanho n.

- Exemplo: Algoritmo de ordenação onde o conjunto de dados de entrada está em ordem contrária à ordenação que se deseja realizar.

Melhor Caso:

- Supõe que os dados de entrada são tais que o algoritmo terá o MÍNIMO de trabalho a ser feito entre todas as instâncias de tamanho n.

- Exemplo: Algoritmo de ordenação onde o conjunto de dados de entrada está na mesma ordem em relação à ordenação que se deseja realizar

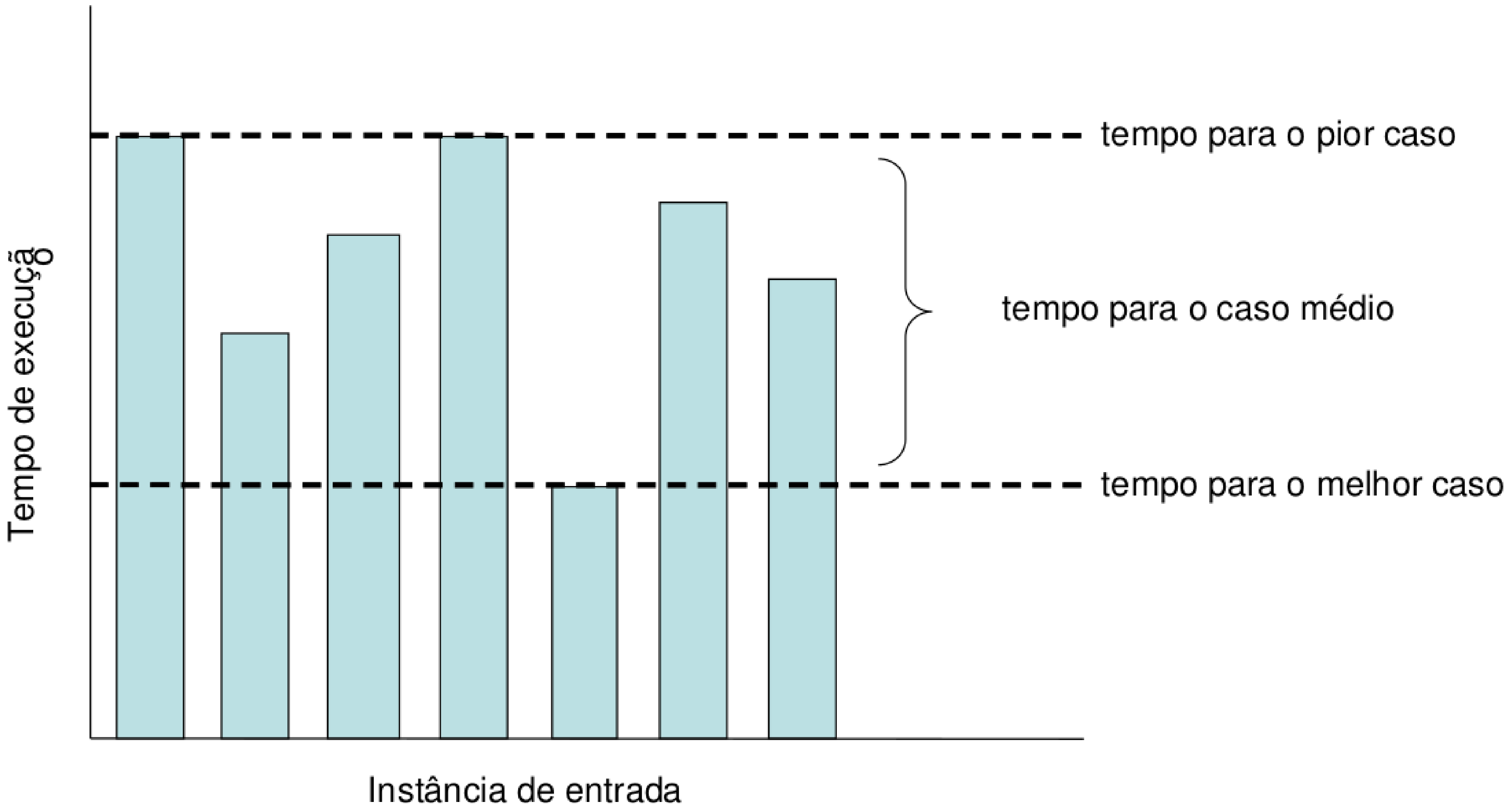
Algebricamente:

- Suponha que A é o algoritmo a ser analisado, e In é o conjunto de todas as possíveis entradas de A, cada uma com um tamanho n. Seja fA uma função que expressa o custo de A, ou seja, se I é uma entrada em In, então fA(I) é o custo de A para a entrada I.

$$\text{PIOR CASO (A)} = \max_{I \in I_n} fA(I)$$

$$\text{MELHOR CASO (A)} = \min_{I \in I_n} fA(I)$$

Graficamente



Pior Caso

- Tradicionalmente, a complexidade pessimista tem sido a de maior interesse teórico. Ocasionalmente, ela é de interesse prático.
- Por exemplo, um sistema de controle de tráfego de metrô com rapidez média aceitável pode ser considerado inútil se o caso pessimista ocorrer, embora raramente, e causar um desastre.

Caso Médio

- Nos últimos anos tem havido um interesse maior na análise da rapidez média dos algoritmos, uma vez que ela é mais útil na prática.
- Exemplo: o bem conhecido algoritmo SIMPLEX para programação linear necessita de um tempo igual a uma função exponencial no tamanho da instância, no caso pessimista, mas para a maioria das instâncias encontradas na prática, o algoritmo é extremamente rápido.

Caso Médio

- Essa abordagem tem as suas dificuldades:
  - Calcular a média sobre muitos casos é uma operação consideravelmente complicada.
  - embora a média tenha isoladamente o seu valor, é desejável achar-se a distribuição dos tempos de solução e a variância, mas estes cálculos são obstáculos adicionais difíceis e frequentemente não são feitos na prática.
  - Existe também uma forte objeção no sentido de que as hipóteses de distribuição probabilísticas dos dados de entrada (usualmente simples para tornar a análise tratável matematicamente) são irrealistas.
- Apesar das objeções, a análise do caso médio tem a sua importância e continua sendo feita.

Caso Médio

- Muitas dessas análises de caso médio usam a distribuição de probabilidade dada por:

$$F(n) = \sum_{i=1}^{n} i \cdot p_i$$

- Os valores associados a f(n) dependem do problema em questão. Por exemplo, para um algoritmo de ordenação, f(n) pode estar associada ao número de comparações realizadas na estrutura. Assim, poderíamos ter i: 1...n representando a i-ésima comparação.

$$\text{Probabilidade} = \frac{\text{nr de casos favoráveis}}{\text{nr de casos possíveis}}$$

Cota Superior:

- Para um dado problema, podemos ter vários algoritmos que o resolvem, cada um deles com complexidades diferentes. A menor dessas complexidades é chamada de COTA SUPERIOR DE COMPLEXIDADE Cs(n) do problema considerado. Esta cota nos diz que, para instâncias arbitrárias de tamanho n, podemos resolver o problema em tempo menor ou igual a Cs(n). Ou seja, nós não devemos ficar satisfeitos com um algoritmo de complexidade de pior caso maior que Cs(n), pois um outro algoritmo de complexidade de pior caso Cs(n) já existiria.

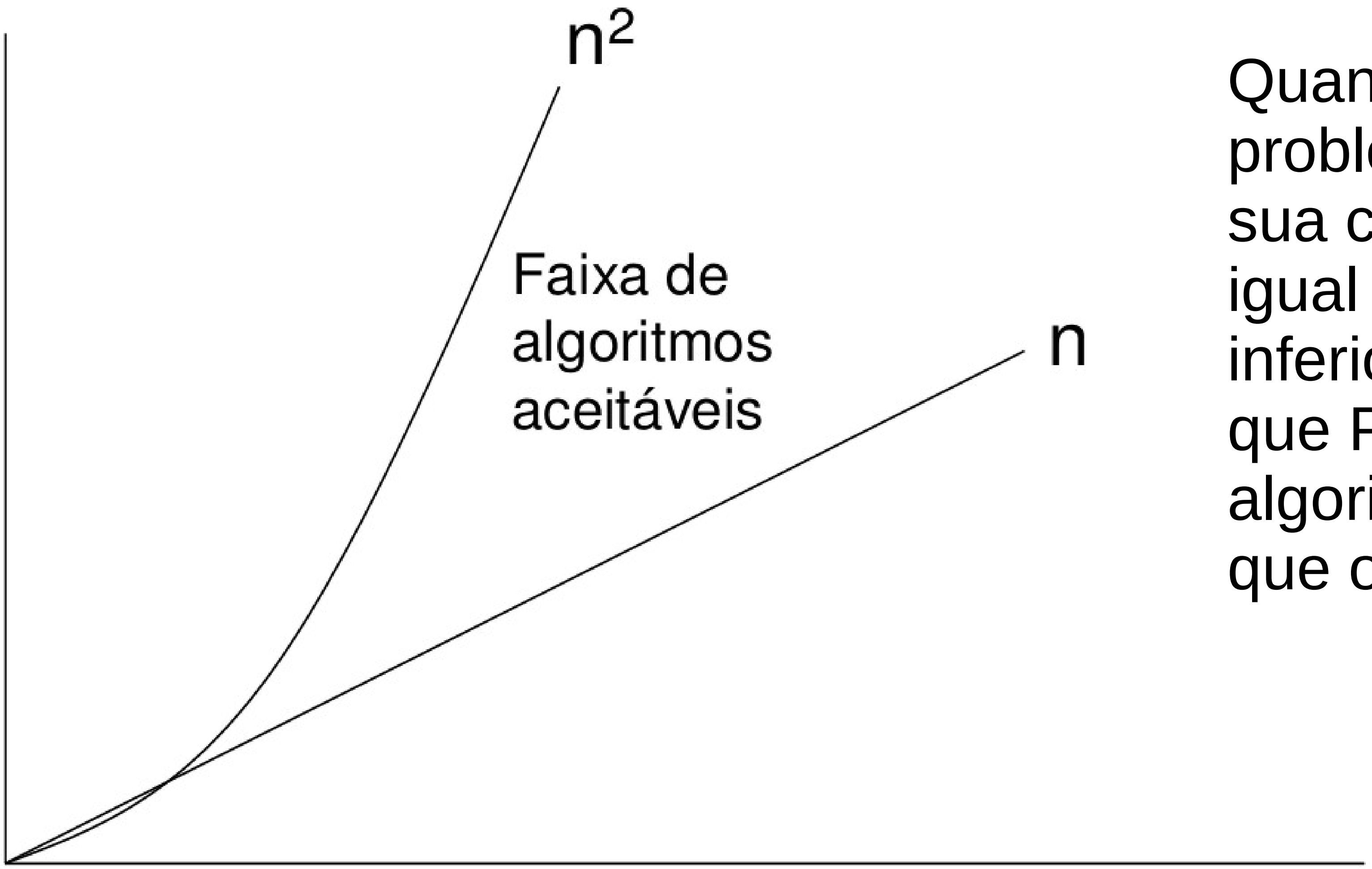
Cota Inferior

- Por outro lado, podemos estar interessados em saber quão rapidamente pode se resolver todas as instâncias de um dado problema, independente do algoritmo que exista ou que se possa descobrir no futuro. Isto é, estamos interessados na complexidade inerente ou intrínseca do problema, que é chamada COTA INFERIOR DE COMPLEXIDADE Ci(n) do problema. Esta cota nos diz que nenhum algoritmo pode resolver o problema com complexidade de pior caso menor que Ci(n), para entradas arbitrárias de tamanho n. Esta cota é então uma propriedade do problema considerado, e não de um algoritmo particular.

Cota Inferior

- Ci(n) é o mínimo sobre todos os algoritmos possíveis.
- Cs(n) é o mínimo sobre todos os algoritmos existentes.
- O objetivo final é fazer essas duas funções Ci(n) e Cs(n) coincidirem. Quando isso ocorre, o algoritmo “ótimo” terá Ci(n) = Cs(n). Para a maioria dos problemas que estudaremos, este objetivo ainda não foi alcançado.
- Aprimorar uma cota superior significa descobrir um algoritmo que tenha complexidade de pior caso menor do que a dos outros.
- Ao tentarmos demonstrar uma cota inferior melhor do que a existente, nos concentramos em técnicas que nos permitam aumentar a precisão com a qual o mínimo, sobre todos os algoritmos possíveis pode ser limitado.

Cota Superior e Inferior



Supondo problema P com cota inferior igual a n e cota superior igual a n^2

Quando um problema P tem sua cota superior igual a cota inferior, dizemos que P tem um algoritmo ótimo que o resolve