Noções "muito" básicas de complexidade de algoritmos...

Para comparar o desempenho de algoritmos, pode-se realizar vários testes de execução (com diferentes entradas) e comparar os tempos de execução. Neste caso, vai haver uma dependência do processador usado, da memória disponível, etc. Uma outra forma (independente de máquina) é estimar o tempo, através da contagem do número de operações que o algoritmo precisa executar no pior, médio e/ou melhor caso. Através dessas estimativas, podemos avaliar a complexidade dos algoritmos.

Definição: a **complexidade de tempo** de um algoritmo, ou simplesmente **complexidade**, é a quantidade de trabalho executada pelo algoritmo.

Determinação da complexidade de um algoritmo: é escolhida uma (ou mais) operação do algoritmo, chamada de **operação fundamental**: é uma operação *chave* - por exemplo, para os algoritmos de ordenação, pode ser a operação de troca. A complexidade é baseada na contagem do número de operações fundamentais realizada pelo algoritmo durante sua execução. Exemplos: comparações ou trocas em um algoritmo de classificação.

- Normalmente a complexidade é uma funçao do tamanho (ou da quantidade de itens) da entrada: n

- **Notação O:** função que estima o tempo de execução, considerando n entradas: O(f(n))

Exemplo: se a complexidade de um algoritmo é definida por O(n), isso significa que o tempo de busca para n elementos é proporcional ao número de entradas do algoritmo. Assim, se o número de entradas aumenta três vezes, o tempo de busca triplica.

Busca-se obter algoritmos onde f(n) seja a "menor" função possível:

O(1) é melhor que
O(log n) é melhor que
O(n) é melhor que
O(n log n) é melhor que
$O(n^2)$ é melhor que
$O(2^n)$ é melhor que
$O(3^n)$ etc...

Os primeiros 5 casos são de algoritmos **polinomiais**, os dois últimos são de algoritmos **exponenciais**. Normalmente considera-se algoritmos polinomiais como tratáveis em tempo razoável e algoritmos exponenciais como intratáveis.

Exemplos de tempos necessários para executar algoritmos com diferentes funções de complexidade, em um mesmo computador:

tamanho n

	10	**20**	**30**	**40**	**50**
função de complexidade					
n	0.00001 seg.	0.00002 seg.	0.00003 seg.	0.00004 seg.	0.00005 seg.
n^2	0.0001 seg.	0.0004 seg.	0.0009 seg.	0.0016 seg.	0.0025 seg.
n^3	0.001 seg.	0.008 seg.	0.027 seg.	0.064 seg.	0.125 seg.

2^n	0.001 seg.	1.0 seg.	17.9 min.	12.7 dias	35.7 anos