**Introdução**

Embora pareça, na maioria das vezes, que todos os algoritmos são iguais, basta estar rodando e resolver o problema proposto que ele é funcional, isto não se prova verdadeiro quando se estuda a Análise de Complexidade de Algoritmos. O presente trabalho se propõe a discutir e elucidar certos pontos sobre esta análise, como a importância da análise matemática, comportamento assintótico e os seus tipos de análise, as classes de problemas, dentre outros.

Uma boa compreensão deste assunto, proporciona ao futuro programador ou estudioso sobre algoritmos, facilidade de encontrar gargalos e aumentar a eficiência do código. Portanto, são utilizados alguns parâmetros para que esta análise seja feita, pois analisar apenas pela velocidade de execução pode não ser tão preciso, já existem diversas variáveis que precisam ser levadas em consideração: o processador, disco, memória, dentre outros.

**A importância da Análise Matemática**

Existem dois tipos de análises comumente utilizadas para a classificação de algoritmos: a empírica e a matemática. A primeira é a forma mais simples de analisar, pois analisa dois ou mais códigos e verifica qual o mais rápido ao executá-lo. Contudo, como foi dito na introdução, este sistema é muito simples e pouco preciso, existem outras variáveis que precisam ser levadas em consideração. Já o segundo tipo de análise, a Matemática, é a mais comumente utilizada.

Neste processo, apenas as operações mais importantes são consideradas, para isto, é definido uma função de custo f(n) e utilizamos algumas notações e parâmetros para classifica-la. O objetivo é a expressar a necessidade de em um programa, de recursos necessários, em relação à N, que é proporcional ao tamanho de dados que serão processados.

**Contando Instruções de um Algoritmo**

Para contar a instrução de um algoritmo, vai contar quantas instruções simples são executadas, que teoricamente possuem o mesmo “custo”. Tal contagem pode ser realizada pelas linhas de código, analisando as instruções e as funções que são utilizadas em um determinado algoritmo.

A partir desta contagem, é possível determinar a função matemática para executar o algoritmo, já sendo a função que será utilizada na notação. A ordem maior sempre prevalece, sendo os outros valores dentro da função podem ser desconsiderados. Por exemplo a função f(n)=2n²+3 o que é consideração para a notação é O(n²).

**Comportamento assintótico, tipo de análises e classes de problemas**

Dentro da ciência da computação, utiliza-se análises assíntotas para resolver problemas, para fazer aproximações ou estudar distribuições, estudando comportamentos de funções quando seus parâmetros tendem ao infinito, usando equações diferenciais. Existem alguns tipos de notações utilizadas para fazer tais análises assintóticas, as principais são: Notação O, Notação Ômega e Notação Theta.

* **Notação O**

A notação O surgiu em 1864, pelo matemático alemão Paul Bachmann, em 1976, Donald Knuth propõe que tal notação seja usada como análise de algoritmos em suas publicações, visa analisar a curva de uma função que tende ao infinito ou à valores muito grandes, de modo a analisar qual função nunca chegará a ser maior que a outra.

Tal notação se refere ao limite superior da função, ou seja, à curva maior de crescimento desta, esta curva é proporcional ao tempo de execução do algoritmo, a complexidade pode ser definida na seguinte ordem de funções, da maior para a menor:

O(c) < O(log n) < O(n) < O(n log n) < O(n²) < O(n³) < O(2^n) < O(n!)

Quanto mais complexa, pior será o algoritmo em tempo de execução, quanto mais completa, maior sua velocidade. Portanto, pela ordem das notações, um algoritmo com notação constante é mais rápido que um algoritmo que tem a complexidade fatorial.

* **Notação Ômega**

Como ocorre na notação O, esta notação tem o interesse em determinar o tempo/espaço da execução de um algoritmo. Porém, o que é analisado é a função de limite inferior, o contrário da O, que olhamos a de limite superior. A complexidade é a mesma a mesma ordem dada anteriormente:

O(c) < O(log n) < O(n) < O(n log n) < O(n²) < O(n³) < O(2^n) < O(n!)

* **Notação Theta**

Quando se diz sobre a notação Theta, está falando simultaneamente da Notação O e da Notação Ômega, pois é analisado simultaneamente o limite superior e o inferior de uma determinada função em relação à outra, para a execução de um determinado algoritmo. A ordem de complexidade é a mesma das outras duas:

O(c) < O(log n) < O(n) < O(n log n) < O(n²) < O(n³) < O(2^n) < O(n!)

**Conclusão**

Portanto, a analisar matematicamente algoritmos é de suma importância para a compreensão e resolução de problemas de forma mais eficaz. Conhecer e compreender sobre as notações proporciona ao programador outra forma de olhar o problema e pensar sobre a solução que deverá ser apresentada.