P, NP e NP-Completo

Lucas Santiago de Oliveira

Março de 2020

1 Introdução

Os problemas P são problemas matemáticos que podem ser resolvidos dentro de um tempo polinomial calculável: $\mathcal{O}(n^k)$ sendo k um valor constante. Traduzindo, tempos polinomiais são todos aqueles em que se pode executar dentro de um tempo esperado. As classes de problemas NP (Não-Polinomiais) são todos os problemas matemáticos em que o tempo **não** é determinístico, ou seja, não se pode prever qual será o tempo gasto antes da execução do código. Por fim, os algoritmos NP-Completos são uma classe específica dentro dos problemas NP, em que não se pode conseguir executar o algoritmo em tempo humano.

2 Explicando as métricas

2.1 Tempos calculáveis

Para se calcular um tempo é necessário uma métrica, para isso é usada a máquina de Turing. Esse dispositivo teórico consegue executar um conjunto de ações determinísticos, ou seja, que contém um conjunto determinado de instruções. Dessa forma, durante cada uma das execuções do algoritmo pode-se estimar qual será o tempo necessário para se completar essas instruções, antes mesmo, de inicia-lo.

2.2 Tempos não calculáveis

2.2.1 Problemas não polinomiais

Os problemas não polinomiais também podem ser calculados pela máquina de Turing, mas não se pode prever com precisão qual será o tempo total gasto antes da execução. Dá-se o nome de máquina de Turing não-determinística para esse dispositivo. Um exemplo de algoritmo não polinomial seria:

```
logico eFatorNaoTrivial (N, D)
    Se N for divisivel por D e D != 1 e D != N
        retorna verdadeiro
    Senao
        retorna falso
```

Dessa forma, cada tempo seria diferente dependendo da entrada que a função receber. Não podendo ser calculado o tempo do algoritmo antes de se ter essa entrada.

2.2.2 Tempo não humano

Para o caso dos problemas não-polinomiais completos, não é possível estimar o tempo necessário para executa-lo e diferente dos problemas NP, os problemas completos podem levar milhares, milhões ou tempo indeterminado de anos para serem executados. Um exemplo disso é as chaves de segurança RSA, para se decifrar os números primos que a geraram é necessário testar todos os números anteriores a ele. Com isso, chaves de segurança muito grandes, tornam o tempo para descobrir os primos iniciais exponencial.

Exemplificando: para se descobrir que o número que a gerou é o 3 e o 5 testa-se todas as possibilidades antes deles: 1, 1 — 1, 3 — 1, 5 — 3, 5. Mas para números muito grandes seria necessário testar todos os números anteriores a eles. Dessa forma, poderia levar milhões de anos para se descobrir quais eram as chaves iniciais do processo.