Algorithmic complexity: example proof

20 mai 2024

Complexity

In the following example we show an example of proof of the time complexity of a program. For the project, something similar can be done. Please not that using the $\mathcal{O}()$ notation (Landau notation) is not mandatory. Use a notation that feels comfortable to you.
`https://fr.wikipedia.org/wiki/Comparaison_asymptotique`

Exercice 1 : Computing a running time

Please compute the complexity of the following algorithm.

```
for i in range(n):
    for j in range(i):
        l = [i+j+k for k in range(n)]
```

Proof I

```
l = [i+j+k for k in range(n)]
```

Involves 2n additions, hence this line is $\mathcal{O}(n)$.
We could also study the size of the memory allocation here, however we are interested in the time complexity here and this memory usage will not affect the next computations.

Proof II

Let us compute how many times the computation of the list of the previous slide is made. We note $S(n)$ this number. We have that :

$$S(n) = \sum_{i=0}^{n-1} i \tag{1}$$

Here, we can either say that $i \leq n$, hence $S(n) \leq n^2$, or use the exact value of $S(n)$, which is $\frac{n(n-1)}{2}$. Both arguments lead to the fact that $S(n)$ is quadratic in n, it is smaller than n^2 multiplied that some constant, and we can write $S(n) = \mathcal{O}(n^2)$.
Putting all these information together, we conclude that the program runs in a time that is **cubic in n, the number of elementary operations (multiplications, additions) is smaller than n^3 multiplied by some constant, and the time complexity is $\mathcal{O}(n^3)$.**