EPFL - Information, Calcul et Communication - Partie théorique

Pourquoi s'intéresser à la théorie de la complexité ?

Jusqu'à présent, nous avons parlé de la complexité des algorithmes. La théorie de la complexité concerne elle les *problèmes*. Un exemple concret de problème est le suivant : "Etant donné un nombre entier postif N, celui-ci est-il un nombre premier ?", aussi appelé problème de la *primalité*.

Un des buts fondamentaux de la théorie de la complexité est de classifier les problèmes existants, afin de distinguer :

- ceux dits "faciles" à résoudre, ce qui veut dire concrètement : ceux admettant un algorithme de résolution dont la complexité temporelle est raisonnable (dans un sens à préciser encore) ;
- ceux qui sont solubles, mais seulement avec des algorithmes dont la complexité temporelle est trop grande pour que ceux-ci puissent être utilisés en pratique ;
- et finalement, ceux qui sont insolubles.

Le fait que certains problèmes puissent être déclarés insolubles par un algorithme vous surprendra peut-être ; c'est pourtant ce qui a été démontré il y a bientôt un siècle de cela par Alan Turing. La démonstration de ce fait n'est pas si compliquée : c'est le sujet de la prochaine vidéo.

Le problème de la primalité mentionné ci-dessus fait partie quant à lui des problès solubles ; en effet, pour tester si un nombre entier N est premier. il "suffit" d'essayer de le diviser par tous les nombres qui sont plus petits que lui pour obtenir la réponse. Pour autant, est-il soluble en un temps raisonnable ? Si $N = 101$, par exemple, tester tous les diviseurs entre 1 et 100 est certainement faisable rapidement sur un ordinateur moderne, mais qu'en est-il maintenant si N est un nombre à n chiffres, avec n grand (disons $n = 100$) ? Dans ce cas, tester tous les diviseurs de N revient à tester tous les diviseurs entre 1 et N, or la valeur de N est de l'ordre 10^{100} ; ceci implique clairement beaucoup trop de nombres à tester, même avec un ordinateur moderne...

Il se trouve cependant qu'un autre type d'algorithmes permettent de résoudre le problème de la primalité de manière beaucoup plus efficace. En 2002, des chercheurs ont ainsi pu démontrer formellement que ce problème fait partie des problèmes faciles à résoudre. C'est important, car la recherche de grands nombres premiers trouve de nombreuses applications de nos jours en cryptographie : nous y reviendrons à la fin du cours.

Et qu'en est-il des multiples autres problèmes ? Citons par exemple :

- le problème de la recherche d'un élément dans uns liste ;
- le problème du tri d'une liste ;
- le problème de la recherche du plus court chemin dans un graphe ;
- le problème du voyageur de commerce (mentionné au début de ce cours).

Heureusement, il se trouve que de nombreux problèmes en informatique (notamment les trois premiers listés ci-dessus) font partie de la classe des problèmes faciles à résoudre. Mais comme nous allons le voir, il existe aussi de nombreux autres problèmes pour lesquels on ne sait pas encore, à l'heure actuelle (i.e., en 2021), s'il est possible ou non de les résoudre en un temps raisonnable au moyen d'un algorithme. C'est le cas notamment du problème du voyageur de commerce.