Notion de complexité

Il nous faut cependant expliquer plus rigoureusement la notion de facile/difficile ce qui revient à spécifier la complexité d’un algorithme.   
Il existe des définitions mathématiques très précises pour cela. Dans ce texte, pour simplifier, nous nous contenterons d’évaluer la   
complexité d’un algorithme par   
le nombre de multiplications (d’entiers) et de divisions euclidiennes qu’il exécute en fonction de la taille de son entrée. La taille de l’entrée, ℓ ,  
est le nombre de chiffres qu’il faut pour le représenter. Ce nombre dépend de la base choisie mais, à une constante près,   
la taille de ℓ est égale à log(ℓ) .

On suppose que nous avons un algorithme à disposition qui prend un nombre ℓ en entrée et on note *C*(ℓ) le nombre de   
multiplications/divisions faites pour exécuter l’algorithme. Les différents types de complexité que nous envisagerons ici sont les suivantes :

* si *C*(ℓ)≤*R*log(ℓ) pour une constante *R*∈R , on dit que l’algorithme est linéaire ;
* si *C*(ℓ)≤*R*log(ℓ)*k* pour *R*∈R et *k*∈N , on dit que l’algorithme est polynomial (linéaire si *k*=1 , quadratique si *k*=2 , cubique si *k*=3 , etc.) ;
* si *C*(ℓ)≤*ek*log(ℓ)*a*log(log(ℓ))1−*a* pour *k*∈R et *a*∈[0,1] , on dit que l’algorithme est sous-exponentiel. En particulier, si *a*=0 , il s’agit de la classe polynomiale précédente, alors que si *a*=1 , il s’agit de la classe exponentielle qui suit ;
* *C*(ℓ)≤*R*ℓ=*Re*log(ℓ) pour *R*∈R , on dit que l’algorithme est exponentiel.

Pour effectuer un calcul, si nous avons un algorithme polynomial pour le faire, alors le calcul est qualifié de « facile »   
 [[4](http://images.math.cnrs.fr/Le-probleme-du-logarithme-discret-en-cryptographie.html#nb4)], voire de « très facile » dans le cas d’un algorithme linéaire.  
Si le seul algorithme à disposition est exponentiel, le calcul est « très difficile ».