Université de Limoges

Développement Logiciel Cryptographique

Projet n° 8

Techniques simples de factorisation d'entier

Résumé

L'objectif de ce projet est multiple. Tout d'abord il s'agit d'étudier, d'implémenter et de comparer plusieurs techniques simples de factorisation d'entier : la méthode par divisions successives ainsi que les méthodes *rho* et *p-1* toutes deux dues à Pollard. Dans un deuxième temps vous développerez un outil de factorisation paramétrable et à usage général qui emploiera ces méthodes de factorisation. Enfin, toute la partie programmation de ce projet se fera en langage C et vous permettra de vous familiariser avec l'utilisation de la librairie GMP de calcul sur grands entiers.

1 Étude de trois méthodes simples de factorisation

Les méthodes de factorisation d'entier par divisions successives, ainsi que les méthodes appelées *rho* de Pollard et *p-1* de Pollard sont facilement accessibles sur le plan théorique. Elles permettent de trouver des facteurs de petite taille d'un entier composé.

Après une recherche bibliographique et une étude théorique de ces diverses méthodes, vous devrez les implémenter indépendamment afin d'en étudier le comportement de manière expérimentale, et d'en comparer les spécificités et efficacités respectives.

Vous devrez connaître et savoir justifier la complexité théorique de chaque méthode. Vous écrirez vos programmes en C et utiliserez la librairie GMP (GNU Multi Precision) de calcul sur grands entiers.

- La méthode par divisions successives est paramétrée par le plus grand premier p_{max} considéré. Vous mesurerez l'évolution du temps d'exécution de votre programme en fonction de ce paramètre.
- La version basique de la méthode *rho* de Pollard est paramétrée par le nombre d'itérations maximum que l'on accepte d'effectuer sur la séquence.

D'autres paramètres peuvent également entrer en jeu dans certaines variantes plus efficaces. Ici aussi vous étudierez l'évolution du temps d'exécution en fonction de ce(s) paramètre(s), ainsi que la probabilité qu'un facteur de taille donnée soit trouvé par cette méthode. Vous comparerez également deux méthodes de détection de cycle : la méthode de Floyd, et celle de Brent.

• Enfin le paramètre principal de la méthode *p-1* de Pollard est la borne B_1 de friabilité. Contrairement aux deux méthodes précédentes, la possibilité de trouver un facteur p de taille donnée n'est pas déterminée uniquement par sa taille, mais plutôt par le caractère friable de $p-1$. Pour un même effort de calcul il est donc possible de trouver des facteurs plus grands que pour les méthodes précédentes, mais la probabilité que cela se produise décroît a priori avec la taille de ce facteur. Vous vous attacherez à mettre en évidence ce phénomène et essaierez de confronter vos mesures expérimentales avec des prédictions théoriques. Si vous en avez la possibilité, vous programmerez également la variante dite *à deux phases* de cette méthode, et étudierez son intérêt.

Comme challenge vous essaierez d'établir des records personnels de plus grands facteurs trouvés avec vos implémentations :

- par la méthode *rho* (facile : 14 chiffres, difficile : 18 chiffres),
- par la méthode *p-1* (facile : 20 chiffres, difficile : 30 chiffres),

2 Outil de factorisation à usage général

Les trois méthodes de factorisation évoquées à la section 1 ont des contextes d'emploi préférentiel différents et complémentaires. C'est la raison pour laquelle un outil de factorisation à usage général ne se restreint pas à l'emploi d'une seule d'entre elles mais les exploite chacune successivement de manière appropriée.

Vous écrirez donc un outil de factorisation à usage général que vous pourrez réutiliser dans d'autres circonstances. Ce programme prend en entrée un nombre entier n et des paramètres de factorisation, et tente de trouver des facteurs propres de ce nombre jusque, si possible, sa factorisation complète. Les trois méthodes que vous aurez étudiées pourront être essayées tour à tour :

- l'essai de divisions successives par tous les premiers inférieurs à une certaine borne p_{max},
- la méthode *rho* de Pollard tentée avec un certain nombre d'itérations maximum ρ_{max},
- la méthode $p-1$ de Pollard avec une borne de friabilité B_1 (ainsi que B_2 si vous avez implémenté la variante à deux phases).

Par défaut les trois méthodes seront tentées, mais l'utilisateur aura la possibilité de débrayer chacune d'elle indépendamment. Les paramètres de

factorisation (p_{max}, ρ_{max}, B_1,...) seront laissés au choix de l'utilisateur, mais des valeurs par défaut typiques seront prévues.

Une fois l'outil de factorisation écrit, testé et optimisé en temps, il s'agira de jouer avec, afin de factoriser entièrement la majeur partie des nombres d'environ 40 chiffres, et une bonne proportion des nombres d'environ 50 chiffres.

Références

[1] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996. http://www.cacr.math.uwaterloo.ca/hac.

[2] Victor Shoup. *A Computational Introduction to Number Theory and Algebra*. Cambridge University Press, 2nd edition, 2008. http://www.shoup.net/ntb/ntb-v2.pdf.