Algorithmes pour le problème des plus proches voisins et applications en cryptanalyse

Charles Bouillaguet (charles.bouillaguet@lip6.fr)
Sorbonne Université, CNRS, LIP6, F-75005 Paris, France
et
Claire Delaplace (claire.delaplace@u-picardie.fr)
Université de Picardie Jules Verne, CNRS, MIS, Amiens, France

9 avril 2022

Mots clés. Chiffrement McEliece, "post-quantique", codes correcteurs d'erreurs, analyse d'algorithmes, probabilités

Contexte

McEliece a proposé en 1978 un système de chiffrement à clef publique [13] reposant sur la difficulté de décoder des codes correcteurs d'erreurs aléatoires. Il n'a pas été cassé en 40+ ans et c'est un candidat sérieux à une standardisation « post-quantique » par le gouvernement américain.

Le problème sous-jascent (qui est NP-complet) est le suivant. On reçoit en entrée :
— Une matrice aléatoire M de taille $n \times m$ à coefficients dans $\mathbb{F}_2$.
— Un vecteur $s \in {\mathbb{F}_2}^m$.
— Un poids maximal $w \in \mathbb{N}$.
Il faut trouver un vecteur $x \in {\mathbb{F}_2}^n$ tel que $Mx = s$ et le poids de Hamming de x est inférieur ou égal à w (ou alors déterminer qu'il n'en existe pas). L'apparition d'ordinateurs quantiques efficaces ne semble pas permettre de conduire à des solutions efficaces pour résoudre ce problème. Sa difficulté peut donc servir pour établir la sécurité de mécanismes de signature ou de chiffrement « post-quantiques ».

Le meilleur algorithme à ce jour est dû à Alexander May et Ilya Ozerov en 2015 [12]. Il utilise de façon cruciale une sous-routine qui résout le problème des *plus proches voisins* (dans un contexte booléen). Étant donné deux listes A et B contenant chacune N chaînes de n bits, il faut trouver toutes les paires $(x, y) \in A \times B$ telles que la distance de Hamming entre x et y soit inférieure à w (autrement dit le poids de $a + b$ est inférieur à w). En fait, tous les algorithmes modernes de décodage des codes linéaires aléatoires [14, 6, 9, 5, 11, 3] se ramènent à la résolution du problème des plus proches voisins, mais de façon implicite. May et Ozerov l'ont explicité.

Par ailleurs, le problème de trouver une *quasi-collision* sur une fonction de hachage cryptographique [10] se ramène très précisément à un cas particulier du problème des plus proches voisins, où chaque élément des listes sont obtenus en évaluant une fonction de hachage.

Un algorithme naïf peut résoudre le problème des plus proches voisins en $\mathcal{O}\left(N^2\right)$ opérations. Une autre technique simple, la méthode des « projections », améliore ceci, mais la complexité dépend de w (elle est meilleure pour des valeurs faibles). Assez récemment, Alman [1] a proposé

un algorithme simplifié dont la complexité ne dépend pas de w. L'algorithme proposé par May-Ozerov est une version raffinée de la méthode des projections, et sa complexité dépend de w.

Une idée alternative consiste à utiliser du *locality-sensitive hashing*, c'est-à-dire une famille de fonctions de hachage qui affecte la même empreinte à des éléments proches avec probabilité non-négligeable [8]. Dans le cas booléen qui nous intéresse, ceci peut se faire avec l'algorithme de décodage d'un code de recouvrement (par exemple un code aléatoire de petite taille) ou d'un code parfait.

Il semble donc manifeste que la technique algorithmique à utiliser pour réoudre le problème des plus proches voisins dépend du régime auquel appartient l'instance du problème à résoudre (w faible ou bien w élevé).

Tous les algorithmes pour le problèmes des plus proches voisins utilisent d'une manière ou d'une autre l'idée que si la paire de voisins recherchée est *globalement* proche, alors elle doit aussi être *localement* proches presque partout. Par exemple, l'idée générale commune aux algorithmes qui reposent *in fine* sur l'idée des projections consiste à à « deviner » un ensemble de positions où la paire solution a une proportion élevée de bits à zéro. Ceci permet de filtrer les listes (« projeter »), donc de réduire significativement leur taille. On peut alors recommencer récursivement ou basculer vers des techniques plus simples.

Objectifs de la thèse

L'objectif de fond du projet de recherche doctoral consiste à améliorer les algorithmes de décodage des codes linéaires aléatoires, et donc la cryptanalyse du chiffrement McEliece.

Presque toutes les techniques dont il est question ont été décrites sur le papier et n'ont pas été utilisées dans les records de calcul qui ont été menés à bien [4, 7] dans le cadre de la cryptanalyse du chiffrement McEliece. Elles n'ont parfois même pas été implantées du tout et leur efficacité pratique est inconnue. Leur potentiel pour améliorer *en pratique* l'efficacité des algorithmes de décodage est en discussion. En plus, il y a parfois de gros facteurs constants et logarithmiques cachés dans leur complexité.

Pour commencer, il serait intéressant de définir des « applications pratiques » raisonnables, en lien avec la cryptanalyse du chiffrement McEliece, car cela donnerait un problème pratique sur lequel ces méthodes pourraient être testées. Une possibilité consisterait à prendre $n \approx 650$, $N \approx 2^{25}$ et $w \approx 200$. Ceci correspond en effet à un scénario réaliste d'utilisation dans le cadre du décodage des codes linéaires aléatoires. Peut-on faire mieux que l'algorithme quadratique naïf dans ce cas-là ? Ou que la méthode des projections ?

Les facteurs cachés élevés dans la complexité théorique de l'algorithme de May-Ozerov sont peut-être davantage une conséquence de la technique utilisée pour prouver sa complexité qu'un phénomène inhérent à l'algorithme lui-même ; comme l'algorithme n'a pas été implanté, on ne peut pas trancher cette question expérimentalement. L'algorithme peut se formuler récursivement, mais la preuve de sa complexité nécessite que l'arbre des appels récursifs soit de profondeur constante ; il faut donc traiter un petit nombre de « gros paquets ». Mais on doit pouvoir se débarasser de cette restriction, avec un raisonnement basé sur l'utilisation de processus stochastiques classiques tels que les arbres de Galton-Watson.

Un premier objectif consiste donc à voir si on peut améliorer non seulement l'algorithme lui-même (en faisant beaucoup de « petits paquets ») mais aussi la *preuve* de la complexité (pour en prouver une plus faible). On pourra aussi essayer de simplifier sa présentation.

Dans un deuxième temps, il faudrait vérifier ce que tous ces algorithmes donnent en pratique. Il y a de nombreuses situations à explorer, mais la plus intéressante est celle du régime qui apparaît dans le décodage des codes linéaires aléatoires. En effet, déterminer si les algorithmes sophistiqués sont uniquement des objets théoriques ou bien s'ils peuvent avoir une utilité pra-

tique serait intéressant du point de vue de l'étude de la sécurité du chiffrement McEliece. Cela signifie qu'il faut essayer de les programmer en C de manière un peu optimisée. N'importe quelle implantation « haute-performance » nécessite un travail important. Ce projet de recherche doctoral, comme tout le reste de l'informatique, a donc nécessairement deux aspects : recherche et ingénierie ; science et art.

Un troisième objectif se situe dans la remarque qu'on peut *adapter* les algorithmes de décodage des codes linéaires aléatoire : on peut éventuellement accepter de payer un peu ici ou là plus pour produire les instances les plus faciles possibles du problème des plus proches voisins (dans le « régime » le plus favorable). Ceci n'a jamais vraiment été étudié rigoureusement, et il y a des compromis à explorer.

Enfin, le problème des plus proches voisins peut se formuler de manière plus générale dans n'importe quel espace métrique (pas seulement avec des chaines de bits et la distance de Hamming). Il joue alors un rôle important dans des algorithmes de classification ou dans la recherche de vecteurs courts dans les réseaux euclidiens, un autre problème « post-quantique ». Il serait intéressant de voir comment les techniques développées dans le cas booléen s'adaptent, ou pas, aux cas plus généraux. Là encore des applications en cryptanalyse sont possibles, notamment dans le contexte des réseaux euclidiens (voir par exemple [2]).

Prérequis

Il va falloir manipuler la théorie des probabilités et faire un peu de développement logiciel (`gcc`, `make`, `git`, etc.). Un certain goût pour l'étude des algorithmes est nécessaire.

Références

[1] Josh Alman. An illuminating algorithm for the light bulb problem. In Jeremy T. Fineman and Michael Mitzenmacher, editors, *2nd Symposium on Simplicity in Algorithms, SOSA 2019, January 8-9, 2019, San Diego, CA, USA*, volume 69 of *OASICS*, pages 2 :1–2 :11. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

[2] Anja Becker, Léo Ducas, Nicolas Gama, and Thijs Laarhoven. New directions in nearest neighbor searching with applications to lattice sieving. In Robert Krauthgamer, editor, *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 10–24. SIAM, 2016.

[3] Anja Becker, Antoine Joux, Alexander May, and Alexander Meurer. Decoding random binary linear codes in $2^{n/20}$: How 1+1=0 improves information set decoding. *IACR Cryptol. ePrint Arch.*, 2012 :26, 2012.

[4] Daniel J. Bernstein, Tanja Lange, and Christiane Peters. Attacking and defending the mceliece cryptosystem. In Johannes Buchmann and Jintai Ding, editors, *Post-Quantum Cryptography, Second International Workshop, PQCrypto 2008, Cincinnati, OH, USA, October 17-19, 2008, Proceedings*, volume 5299 of *Lecture Notes in Computer Science*, pages 31–46. Springer, 2008.

[5] Anne Canteaut and Florent Chabaud. A new algorithm for finding minimum-weight words in a linear code : Application to mceliece's cryptosystem and to narrow-sense BCH codes of length 511. *IEEE Trans. Inf. Theory*, 44(1) :367–378, 1998.

[6] Ilya Dumer. On minimum distance decoding of linear codes. In *Proc. 5th Joint Soviet-Swedish Int. Workshop Inform. Theory*, pages 50–52, 1991.

[7] Andre Esser, Alexander May, and Floyd Zweydinger. Mceliece needs a break – solving mceliece-1284 and quasi-cyclic-2918 with modern isd. Cryptology ePrint Archive, Report 2021/1634, 2021. https://ia.cr/2021/1634.

[8] Piotr Indyk. On approximate nearest neighbors in non-euclidean spaces. In *39th Annual Symposium on Foundations of Computer Science, FOCS '98, November 8-11, 1998, Palo Alto, California, USA*, pages 148–155. IEEE Computer Society, 1998.

[9] Jeffrey S. Leon. A probabilistic algorithm for computing minimum weights of large error-correcting codes. *IEEE Trans. Inf. Theory*, 34(5) :1354–1359, 1988.

[10] Gaëtan Leurent. Time-memory trade-offs for near-collisions. In Shiho Moriai, editor, *Fast Software Encryption - 20th International Workshop, FSE 2013, Singapore, March 11-13, 2013. Revised Selected Papers*, volume 8424 of *Lecture Notes in Computer Science*, pages 205–218. Springer, 2013.

[11] Alexander May, Alexander Meurer, and Enrico Thomae. Decoding random linear codes in $\tilde{\mathcal{O}}(2^{0.054n})$. In Dong Hoon Lee and Xiaoyun Wang, editors, *Advances in Cryptology - ASIACRYPT 2011 - 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings*, volume 7073 of *Lecture Notes in Computer Science*, pages 107–124. Springer, 2011.

[12] Alexander May and Ilya Ozerov. On computing nearest neighbors with applications to decoding of binary linear codes. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I*, volume 9056 of *Lecture Notes in Computer Science*, pages 203–228. Springer, 2015.

[13] R. J. McEliece. A Public-Key Cryptosystem Based On Algebraic Coding Theory. *Deep Space Network Progress Report*, 44 :114–116, January 1978.

[14] Jacques Stern. A method for finding codewords of small weight. In Gérard D. Cohen and Jacques Wolfmann, editors, *Coding Theory and Applications, 3rd International Colloquium, Toulon, France, November 2-4, 1988, Proceedings*, volume 388 of *Lecture Notes in Computer Science*, pages 106–113. Springer, 1988.