Vers des outils de référence pour la résolution des systèmes polynomiaux en cryptographie

Charles Bouillaguet, Sorbonne université / LIP6
Ludovic Perret, Sorbonne université / LIP6
{charles.bouillaguet, ludovic.perret}@lip6.fr

Mots clés. systèmes polynomiaux, cryptanalyse, sécurité, post-quantique, HPC

Contexte

Claude Shannon a exposé en 1949 que la sécurité de (presque) n'importe quel mécanisme cryptographique repose sur la difficulté calculatoire de la résolution de certains types d'équations non-linéaires sur des corps finis.

Le problème sous-jacent (qui est NP-complet) est le suivant. On reçoit en entrée une collection de m polynômes $f_1, \ldots, f_m$ en n variables sur $\mathbb{F}_q$. Il faut trouver un vecteur $x \in {\mathbb{F}_q}^n$ tel que $f_1(x) = \cdots = f_m(x) = 0$ (ou alors déterminer qu'il n'en existe pas). Le cas où les polynômes sont quadratiques est le plus répandu et le plus intéressant.

L'émergence d'ordinateurs quantiques ne permettrait pas, dans l'état actuel des connaissance, de résoudre ce problème efficacement. Ceci en fait un candidat pour construire des mécanismes cryptographiques "post-quantiques". Un certain nombre de schémas de signature numérique "multivariés", dont la clé publique est un système d'équations polynomiales, ont été soumis à la compétition organisée par le gouvernement américain pour standardiser des mécanismes post-quantiques. Plusieurs ont été cassés en pratique durant la compétition ; ceci n'a été possible que grâce à l'existence de briques algorithmiques riches et la disponibilité de logiciels *relativement efficaces* de résolution de systèmes polynomiaux.

Les concepteurs de ces mécanismes cryptographiques sont confrontés au problème de devoir choisir les « paramètres » n, m et q : trop petits, les systèmes peuvent être faciles à résoudre ; trop grands, les mécanismes cryptographiques seront sûrs mais inefficaces. Pour pouvoir choisir ces paramètres de manière optimale, il faut combiner d'une part une approche théorique en utilisant les meilleurs bornes de complexité et d'autre part une validation expérimentale pour bien calibrer les paramètres. Dans ce contexte, l'existence de logiciels faciles à utiliser et efficace est un atout inestimable, car cela permet d'extrapoler la difficulté de la résolution de grands systèmes.

Vu la généralité du problème, il existe de nombreux algorithmes pour tenter de le résoudre. Certains sont de nature purement combinatoire (recherche exhaustive [6], des réductions à SAT, la « méthode polynomiale » [18, 4, 14, 13]), d'autre purement algébriques (le calcul de bases de Grœbner [9, 15, 16], l'algorithme XL [12]) et enfin certains combinent les deux [3, 1, 17, 8].

Le problème se présente différemment selon que le corps est petit ($q = 2$ en particulier) ou gros ($q \geq 16$). Les petits corps favorisent la recherche exhaustive, les gros corps les techniques

algébriques. Il se présente aussi différemment selon que les systèmes sont surdéterminés, sous-déterminés, ou bien que $m = n$. Tous ces scénarios nécessitent des techniques algorithmiques différentes.

Le problème de la résolution des systèmes polynomiaux a une « saveur » spécifique dans le cadre de la cryptographie. On ne s'intéresse pas au problème de les « résoudre » en toute généralité. En particulier, on ne cherche pas à décrire l'ensemble des solutions, mais juste à en exhiber *une*. Par ailleurs, le corps de base est généralement de petite taille, avec $q \leq 256$ dans la quasi-totalité des cas.

Tous les algorithmes connus sont de complexité exponentielle. Mais les cinq dernières années ont vu l'apparition de progrès importants, avec notamment l'invention d'algorithmes avec un exposant réduit dans le cas où $q = 2$: de $0.792n$ [1], on est tombé à $0.694n$ [14]. Ceci dit, ces algorithmes plus efficaces sur le papier sont de nature essentiellement théorique et il est probable qu'ils ne seront *jamais* utilisés concrètement (à savoir, pour des valeurs de $n \leq 200$).

D'un point de vue pratique, la situation est variée. Pour le cas $q = 2$, il existe des implantations efficaces de la recherche exhaustive sur CPU/GPU [6] et FPGA [7] ainsi qu'une implantation de l'algorithme de Joux-Vitse sur GPU [20]. Ces logiciels ne sont pas capables d'améliorer les records de calcul actuels et ce sont des codes de « qualité recherche », c'est-à-dire peu utilisables hormis par leurs auteurs.

Pour le cas $q \gg 2$, il existe des implantations efficaces de l'algorithme F4 [15] pour le calcul des bases de Grœbner, par exemple dans le logiciel `MAGMA` [5], une implémentation de l'approche hybride en `MAGMA` [3] ou bien dans la bibliothèque `msolve` [2]. Il existe aussi une implantation distribuée de l'algorithme `XL` [10], qui possède des records de calcul à son actif ($q = 31$).

Objectifs de la thèse

Le point de départ de ce sujet de thèse est l'ambition de rapprocher la théorie de la pratique : d'une part en affinant voire en améliorant les bornes de complexité et d'autre part en développant des briques logicielles efficaces et facilement utilisables par tous.

Sur l'aspect algorithmique, une piste consiste à revisiter l'approche hybride en tirant avantage des nouvelles méthodes de résolution par approximation. Il faudra certainement diviser l'analyse de complexité en deux situations $q = 2$ et $q > 2$ et comprendre la zones d'applicabilité de cette approche (valeur de n mais aussi les hypothèses de régularité nécessaires sur les systèmes). Concernant le logiciel, l'objectif est de produire des implantations de haute qualité de certains algorithmes de résolution des systèmes polynomiaux pour la cryptographie, de s'en servir pour établir de nouveaux records de calcul et des les distribuer sous la forme de logiciels libres placés dans le domaine public. L'existence de ces logiciels serait un service rendu à la communauté.

Ces implantations doivent être à la fois faciles à utiliser et de haute performance. Elles doivent donc être capables de fonctionner sur un *cluster* de serveurs de calcul, ce qui n'est grosso-modo pas le cas des logiciels existants. Leur réalisation pose donc des problèmes typiques du calcul haute performance (HPC). Elle nécessite aussi une reflexion approfondie sur des questions d'architecture logicielle. C'est un travail qui combine recherche et ingénierie, science (théorie, algorithmes, ...) et art (programmation des ordinateurs, en particulier des gros).

Tout ceci soulève aussi des questions algorithmiques qui n'ont pas encore été élucidées. Par exemple, adapter les algorithmes au contexte d'une machine distribuée nécessite très certainement de les modifier. Ils n'ont bien souvent été étudiés que dans des modèles de calcul séquentiels, et leur complexité de communication peut rendre leur parallélisation quasiment impossible ; dans ce cas, il faudrait les modifier ou en trouver d'autres. Il faut également adapter les algorithmes au niveau de parallélisme qu'on est capable d'exploiter : entre les coeurs d'un seul serveur de calcul (plus facile) ou bien entre des serveurs de calculs distincts (plus dur). Les choses se présentent

encore différemment s'il est nécessaire d'exploiter des accélérateurs de calcul tels que des GPUs. Enfin, certains algorithmes efficaces en théorie n'ont jamais été implantés et on ignore donc s'ils ont une utilité pratique ou pas (c'est le cas de [1]). C'est une question qu'il faut trancher avec une démarche expérimentale.

Dans le cas $q = 2$, il semble que l'algorithme de Joux-Vitse [17] soit le plus intéressant mais aucune implantation sérieuse n'est disponible. Il faudrait aussi le comparer à celui de Bardet-Faugère-Salvy-Spaenlehauer [1]. Dans le cas $q \gg 2$, c'est certainement entre des variantes de l'algorithme XL [12] et F4 [15] que le match va se jouer.

Dans toutes ces situations, le traitement de grandes matrices creuses sur des corps fini est un thème algorithmique récurrent. Là encore, des techniques variées sont utilisables (algorithme de block-Wiedemann [11], de block-Lanczos [19], ...) et il faut choisir l'une ou l'autre en fonction de contraints pratiques (taille de la mémoire, vitesse du réseau, ...). Leur mise en œuvre concrète est un problème potentiellement délicat, même si des logiciels existent (en partie) déjà.

Obtenir des records de calculs sur de grandes machines parallèle dans des centres de calculs nationaux permettrait de prouver dans les faits la viabilité des approches poursuivies. Par exemple, résoudre un système aléatoire de 80 polynômes quadratiques en 80 variables modulo 2 semble faisable, et serait une démonstration éclatante. En effet, il y a moins de 15 ans de cela, on considérait qu'un calcul nécessitait 2^{80} opérations était au-delà des capacités calculatoires de l'humanité.

Prérequis

Le sujet nécessite des compétences de base en algèbre commutative, en algorithmique et en HPC. Un certain goût pour la programmation et la réalisation de gros calculs est nécessaire.

Références

[1] Magali Bardet, Jean-Charles Faugère, Bruno Salvy, and Pierre-Jean Spaenlehauer. On the complexity of solving quadratic boolean systems. *J. Complexity*, 29(1) :53–75, 2013.

[2] Jérémy Berthomieu, Christian Eder, and Mohab Safey El Din. msolve : A Library for Solving Polynomial Systems. In *2021 International Symposium on Symbolic and Algebraic Computation*, 46th International Symposium on Symbolic and Algebraic Computation, Saint Petersburg, Russia, July 2021.

[3] Luk Bettale, Jean-Charles Faugère, and Ludovic Perret. Solving polynomial systems over finite fields : improved analysis of the hybrid approach. In Joris van der Hoeven and Mark van Hoeij, editors, *International Symposium on Symbolic and Algebraic Computation, ISSAC'12, Grenoble, France - July 22 - 25, 2012*, pages 67–74. ACM, 2012.

[4] Andreas Björklund, Petteri Kaski, and Ryan Williams. Solving systems of polynomial equations over GF(2) by a parity-counting self-reduction. In Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi, editors, *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece*, volume 132 of *LIPIcs*, pages 26 :1–26 :13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

[5] Wieb Bosma, John J. Cannon, and Catherine Playoust. The Magma Algebra System I : The User Language. *J. Symb. Comput.*, 24(3/4) :235–265, 1997.

[6] Charles Bouillaguet, Hsieh-Chung Chen, Chen-Mou Cheng, Tung Chou, Ruben Niederhagen, Adi Shamir, and Bo-Yin Yang. Fast exhaustive search for polynomial systems in $\mathbb{F}_2$. In Stefan Mangard and François-Xavier Standaert, editors, *Cryptographic Hardware and*

Embedded Systems, CHES 2010, 12th International Workshop, Santa Barbara, CA, USA, August 17-20, 2010. Proceedings, volume 6225 of *Lecture Notes in Computer Science*, pages 203–218. Springer, 2010.

[7] Charles Bouillaguet, Chen-Mou Cheng, Tung Chou, Ruben Niederhagen, and Bo-Yin Yang. Fast Exhaustive Search for Quadratic Systems in $\mathbb{F}_2$ on FPGAs. In *Selected Areas in Cryptography*, volume 8282 of *Lecture Notes in Computer Science*, pages 205–222. Springer, 2013. `https://eprint.iacr.org/2013/436.pdf`.

[8] Charles Bouillaguet, Claire Delaplace, and Monika Trimoska. A simple deterministic algorithm for systems of quadratic polynomials over $\mathbb{F}_2$. *IACR Cryptol. ePrint Arch.*, page 1639, 2021.

[9] Bruno Buchberger. *Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal*. PhD thesis, University of Innsbruck, 1965.

[10] Chen-Mou Cheng, Tung Chou, Ruben Niederhagen, and Bo-Yin Yang. Solving quadratic equations with XL on parallel architectures. In Emmanuel Prouff and Patrick Schaumont, editors, *Cryptographic Hardware and Embedded Systems - CHES 2012 - 14th International Workshop, Leuven, Belgium, September 9-12, 2012. Proceedings*, volume 7428 of *Lecture Notes in Computer Science*, pages 356–373. Springer, 2012.

[11] Don Coppersmith. Solving homogeneous linear equations over gf(2) via block wiedemann algorithm. *Mathematics of Computation*, 62(205) :333–350, 1994.

[12] Nicolas T. Courtois, Alexander Klimov, Jacques Patarin, and Adi Shamir. Efficient algorithms for solving overdefined systems of multivariate polynomial equations. In Bart Preneel, editor, *Advances in Cryptology - EUROCRYPT 2000, International Conference on the Theory and Application of Cryptographic Techniques, Bruges, Belgium, May 14-18, 2000, Proceeding*, volume 1807 of *Lecture Notes in Computer Science*, pages 392–407. Springer, 2000.

[13] Itai Dinur. Cryptanalytic applications of the polynomial method for solving multivariate equation systems over GF(2). In Anne Canteaut and François-Xavier Standaert, editors, *Advances in Cryptology - EUROCRYPT 2021 - 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17-21, 2021, Proceedings, Part I*, volume 12696 of *Lecture Notes in Computer Science*, pages 374–403. Springer, 2021.

[14] Itai Dinur. Improved algorithms for solving polynomial systems over GF(2) by multiple parity-counting. In Dániel Marx, editor, *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*, pages 2550–2564. SIAM, 2021.

[15] Jean-Charles Faugère. A new efficient algorithm for computing grobner bases (f4). *Journal of Pure and Applied Algebra*, 139(1-3) :61–68, 1999.

[16] Jean Charles Faugère. A new efficient algorithm for computing gröbner bases without reduction to zero (f5). In *Proceedings of the 2002 International Symposium on Symbolic and Algebraic Computation*, ISSAC '02, page 75–83, New York, NY, USA, 2002. Association for Computing Machinery.

[17] Antoine Joux and Vanessa Vitse. A Crossbred Algorithm for Solving Boolean Polynomial Systems. In *NuTMiC*, volume 10737 of *Lecture Notes in Computer Science*, pages 3–21. Springer, 2017. `https://eprint.iacr.org/2017/372.pdf`.

[18] Daniel Lokshtanov, Ramamohan Paturi, Suguru Tamaki, R. Ryan Williams, and Huacheng Yu. Beating brute force for systems of polynomial equations over finite fields. In Philip N.

Klein, editor, *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 2190–2202. SIAM, 2017.

[19] Peter L. Montgomery. A block lanczos algorithm for finding dependencies over GF(2). In Louis C. Guillou and Jean-Jacques Quisquater, editors, *Advances in Cryptology - EUROCRYPT '95, International Conference on the Theory and Application of Cryptographic Techniques, Saint-Malo, France, May 21-25, 1995, Proceeding*, volume 921 of *Lecture Notes in Computer Science*, pages 106–120. Springer, 1995.

[20] Ruben Niederhagen, Kai-Chun Ning, and Bo-Yin Yang. Implementing joux-vitse's crossbred algorithm for solving MQ systems over GF(2) on gpus. In Tanja Lange and Rainer Steinwandt, editors, *Post-Quantum Cryptography - 9th International Conference, PQCrypto 2018, Fort Lauderdale, FL, USA, April 9-11, 2018, Proceedings*, volume 10786 of *Lecture Notes in Computer Science*, pages 121–141. Springer, 2018.