Lattice Reduction Algorithms: Theory and Practice

Phong Q. Nguyen

INRIA and ENS, Département d'informatique, 45 rue d'Ulm, 75005 Paris, France.
http://www.di.ens.fr/~pnguyen/

Abstract. Lattice reduction algorithms have surprisingly many applications in mathematics and computer science, notably in cryptology. On the one hand, lattice reduction algorithms are widely used in public-key cryptanalysis, for instance to attack special settings of RSA and DSA/ECDSA. On the other hand, there are more and more cryptographic schemes whose security require that certain lattice problems are hard. In this talk, we survey lattice reduction algorithms, present their performances, and discuss the differences between theory and practice.

Intuitively, a *lattice* is an infinite arrangement of points in $\mathbb{R}^m$ spaced with sufficient regularity that one can shift any point onto any other point by some symmetry of the arrangement. The simplest non-trivial lattice is the hypercubic lattice $\mathbb{Z}^n$ formed by all points with integral coordinates. The branch of number theory dealing with lattices (and especially their connection with convex sets) is known as *geometry of numbers* [24,41,12,5], and its origins go back to two historical problems: higher-dimensional generalizations of Euclid's gcd algorithm and sphere packings.

More formally, a lattice L is a discrete subgroup of $\mathbb{R}^m$, or equivalently, the set of all integer combinations of n linearly independent vectors $\mathbf{b}_1, \ldots, \mathbf{b}_n$ in $\mathbb{R}^n$:

$$L = \{a_1\mathbf{b}_1 + \cdots + a_n\mathbf{b}_n, a_i \in \mathbb{Z}\}.$$

Such a set $(\mathbf{b}_1, \ldots, \mathbf{b}_n)$ is called a *basis* of the lattice. The goal of *lattice reduction* is to find reduced bases, that is bases consisting of reasonably short and nearly orthogonal vectors. This is related to the reduction theory of quadratic forms developed by Lagrange [19], Gauss [11] and Hermite [14]. Lattice reduction algorithms have proved invaluable in many fields of computer science and mathematics (see the book [30]), notably public-key cryptanalysis where they have been used to break knapsack cryptosystems [32] and special cases of RSA and DSA, among others (see [26,21] and references therein).

Reduced bases allow to solve the following important lattice problems, either exactly or approximately:

- The most basic computational problem involving lattices is the *shortest vector problem* (SVP), which asks to find a nonzero lattice vector of smallest norm, given a lattice basis as input. SVP can be viewed as a geometric generalization of gcd computations: Euclid's algorithm actually computes the

smallest (in absolute value) non-zero linear combination of two integers, since $\gcd(a,b)\mathbb{Z} = a\mathbb{Z}+b\mathbb{Z}$, which means that we are replacing the integers a and b by an arbitrary number of vectors $\mathbf{b}_1, \ldots, \mathbf{b}_n$ with integer coordinates. Since SVP is NP-hard under randomized reductions [3] (see [17,34] for surveys on the hardness of lattice problems), one is also interested in approximating SVP, *i.e.* to output a nonzero lattice vector of norm not much larger than the smallest norm.

– The inhomogeneous version of SVP is called the *closest vector problem* (CVP); here we are given an arbitrary target vector in addition to the lattice basis and asked to find the lattice point closest to that vector. A popular particular case of CVP is *Bounded Distance Decoding* (BDD), where the target vector is known to be somewhat close to the lattice.

The first SVP algorithm was Lagrange's reduction algorithm [19], which solves SVP exactly in dimension two, in quadratic time. In arbitrary dimension, there are two types of SVP algorithms:

1. **Exact algorithms**. These algorithms provably find a shortest vector, but they are expensive, with a running time at least exponential in the dimension. Intuitively, these algorithms perform an exhaustive search of all extremely short lattice vectors, whose number is exponential in the dimension (in the worst case): in fact, there are lattices for which the number of shortest lattice vectors is already exponential. Exact algorithms can be split in two categories:
   (a) **Polynomial-space exact algorithms**. They are based on *enumeration* which dates back to the early 1980s with work by Pohst [33], Kannan [16], and Fincke-Pohst [6]. In its simplest form, enumeration is simply an exhaustive search for the best integer combination of the basis vectors. The best deterministic enumeration algorithm is Kannan's algorithm [16], with super-exponential worst-case complexity, namely $n^{n/(2e)+o(n)}$ polynomial-time operations (see [13]), where n denotes the lattice dimension. The enumeration algorithms used in practice (such as that of Schnorr-Euchner [37]) have a weaker preprocessing than Kannan's algorithm [16], and their worst-case complexity is $2^{O(n^2)}$ polynomial-time operations. But it is possible to obtain substantial speedups using *pruning* techniques: pruning was introduced by Schnorr-Euchner [37] and Schnorr-Hörner [38] in the 90s, and recently revisited by Gama, Nguyen and Regev [10], where it was shown that one can reach a $2^{n/2}$ heuristic speedup over basic enumeration.
   (b) **Exponential-space exact algorithms**. These algorithms have a better asymptotic running time, but they all require exponential space $2^{\Theta(n)}$. The first algorithm of this kind is the randomized sieve algorithm of Ajtai, Kumar and Sivakumar (AKS) [4], with exponential worst-case complexity of $2^{O(n)}$ polynomial-time operations. Micciancio and Voulgaris [22] recently presented an alternative deterministic algorithm, which solves both CVP and SVP within $2^{2n+o(n)}$ polynomial-time operations. Interestingly, there are several heuristic variants [31,23,43] of

AKS with running time $2^{O(n)}$, where the $O()$ constant is much less than that of the best provable algorithms known. For instance, the recent algorithm of Wang *et al.* [43] has time complexity $2^{0.3836n}$ polynomial-time operations.

2. **Approximation algorithms**. These algorithms are much faster than exact algorithms, but they only output short lattice vectors, not necessarily the shortest one: they typically output a whole reduced basis, and are therefore lattice reduction algorithms. The first algorithm of this kind is the celebrated algorithm of Lenstra, Lenstra and Lovász (LLL) [20,30], which can approximate SVP to within a factor $O((2/\sqrt{3})^n)$ in polynomial time: it can be viewed as an algorithmic version of Hermite's inequality. Since the appearance of LLL, research in this area has focused on two topics:
   (a) **Faster LLL**. Here, one is interested in obtaining reduced bases of similar quality than LLL, possibly slightly worse, but with a smaller running time. This is achieved by a divide-and-conquer strategy (such as in [39,18]) or by using floating-point arithmetic (such as in [36,29,25]). The most popular implementations of LLL are typically heuristic floating-point variants, such as that of Schnorr-Euchner [37]: see the survey [42] on floating-point LLL.
   (b) **Stronger LLL**. Here, one is interested in obtaining better approximation factors than LLL, at the expense of the running time. Intuitively, LLL repeatedly uses two-dimensional reduction to find short lattice vectors in dimension n. Blockwise reduction algorithms [35,7,8] obtain better approximation factors by replacing this two-dimensional reduction subroutine by a higher-dimensional one, using exact SVP algorithms in low dimension. The best polynomial-time blockwise algorithm known [8] achieves a subexponential approximation factor $2^{O((n \log\log n)/\log n)}$: it is an algorithmic version of Mordell's inequality. In practice, a popular choice is the BKZ algorithm of Schnorr-Euchner [37] implemented in the NTL library [40], which is a heuristic variant of Schnorr's blockwise algorithm [35]. The article [9] provides an experimental assessment of BKZ.

Both categories are in fact complementary: all exact algorithms known first apply an approximation algorithm (typically at least LLL) as a preprocessing, while all blockwise algorithms call many times an exact algorithm in low dimension as a subroutine. Most of the SVP algorithms we mentioned can be adapted to CVP (see for instance [1]). The provable SVP algorithms are surveyed in [27]. The heuristic algorithms which we mentioned are such that their running time may no longer be proved, and/or there may not be any guarantee on the output (should the algorithm ever terminate). Heuristic algorithms can typically outperform provable algorithms in practice, for reasons still not well understood.

Finally, it is folklore that lattice reduction algorithms behave better than their proved worst-case theoretical bounds. In the 80s, the early success of lattice reduction algorithms in cryptanalysis led to the belief that the strongest lattice reduction algorithms behaved as perfect oracles, at least in small dimension. But this belief showed its limits in the 90s with NP-hardness results and the

development of lattice-based cryptography, following Ajtai's worst-case/average-case reduction [2] and the NTRU cryptosystem [15]. The articles [28,9] clarify what can be expected in practice, based on experimental results. Such assessments are important to better understand the gap between theory and practice, but also to evaluate the concrete security of lattice-based cryptography.

References

1. E. Agrell, T. Eriksson, A. Vardy, and K. Zeger. Closest point search in lattices. *IEEE Trans. on Info. Theory*, 48(8):2201–2214, 2002.
2. M. Ajtai. Generating hard instances of lattice problems. In *Proc. STOC '96*, pages 99–108. ACM, 1996.
3. M. Ajtai. The shortest vector problem in L_2 is NP-hard for randomized reductions. In *Proc. of 30th STOC*. ACM, 1998.
4. M. Ajtai, R. Kumar, and D. Sivakumar. A sieve algorithm for the shortest lattice vector problem. In *Proc. 33rd STOC*, pages 601–610, 2001.
5. J. Cassels. *An Introduction to the Geometry of Numbers*. 1997.
6. U. Fincke and M. Pohst. Improved methods for calculating vectors of short length in a lattice, including a complexity analysis. *Mathematics of Computation*, 44(170):463–471, 1985.
7. N. Gama, N. Howgrave-Graham, H. Koy, and P. Q. Nguyen. Rankin's constant and blockwise lattice reduction. In *Proc. CRYPTO '06*, volume 4117 of *Lecture Notes in Computer Science*, pages 112–130. Springer, 2006.
8. N. Gama and P. Q. Nguyen. Finding short lattice vectors within Mordell's inequality. In *Proc. 40th ACM Symp. on Theory of Computing (STOC)*, 2008.
9. N. Gama and P. Q. Nguyen. Predicting lattice reduction. In *Proc. EUROCRYPT '08*, volume 4965 of *Lecture Notes in Computer Science*, pages 31–51. Springer, 2008.
10. N. Gama, P. Q. Nguyen, and O. Regev. Lattice enumeration using extreme pruning. In *Proc. EUROCRYPT '10*, volume 6110 of *Lecture Notes in Computer Science*. Springer, 2010.
11. C. Gauss. *Disquisitiones Arithmeticæ*. Leipzig, 1801.
12. M. Gruber and C. G. Lekkerkerker. *Geometry of Numbers*. North-Holland, 1987.
13. G. Hanrot and D. Stehlé. Improved analysis of Kannan's shortest lattice vector algorithm (extended abstract). In *Proc. of CRYPTO '07*, volume 4622 of *LNCS*, pages 170–186. Springer-Verlag, 2007.
14. C. Hermite. Extraits de lettres de M. Hermite à M. Jacobi sur différents objets de la théorie des nombres. *J. Reine Angew. Math.*, 40:261–315, 1850.
15. J. Hoffstein, J. Pipher, and J. H. Silverman. NTRU: A ring-based public key cryptosystem. In *Proc. ANTS-III*, volume 1423 of *Lecture Notes in Computer Science*, pages 267–288. Springer, 1998.
16. R. Kannan. Improved algorithms for integer programming and related lattice problems. In *Proc. 15th ACM Symp. on Theory of Computing (STOC)*, pages 193–206, 1983.
17. S. Khot. Inapproximability results for computational problems on lattices. 2010. In [30].
18. H. Koy and C.-P. Schnorr. Segment LLL-reduction of lattice bases. In *Proc. CaLC '01*, volume 2146 of *Lecture Notes in Computer Science*, pages 67–80. Springer, 2001.

19. L. Lagrange. Recherches d'arithmétique. *Nouv. Mém. Acad.*, 1773.
20. A. K. Lenstra, H. W. Lenstra, Jr., and L. Lovász. Factoring polynomials with rational coefficients. *Mathematische Ann.*, 261:513–534, 1982.
21. A. May. Using LLL-reduction for solving RSA and factorization problems: A survey. 2010. In [30].
22. D. Micciancio and P. Voulgaris. A deterministic single exponential time algorithm for most lattice problems based on voronoi cell computations. In *Proc. STOC '10*, pages 351–358. ACM, 2010.
23. D. Micciancio and P. Voulgaris. Faster exponential time algorithms for the shortest vector problem. In *Proc. ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 1468–1480, 2010.
24. H. Minkowski. *Geometrie der Zahlen*. Teubner-Verlag, Leipzig, 1896.
25. I. Morel, D. Stehlé, and G. Villard. H-LLL: using householder inside LLL. In *Proc. ISSAC '09*, pages 271–278. ACM, 2009.
26. P. Q. Nguyen. Public-key cryptanalysis. In I. Luengo, editor, *Recent Trends in Cryptography*, volume 477 of *Contemporary Mathematics*. AMS–RSME, 2009.
27. P. Q. Nguyen. Hermite's constant and lattice algorithms. 2010. In [30].
28. P. Q. Nguyen and D. Stehlé. LLL on the average. In *Proc. ANTS-VII*, volume 4076 of *Lecture Notes in Computer Science*, pages 238–256. Springer, 2006.
29. P. Q. Nguyen and D. Stehlé. An LLL algorithm with quadratic complexity. *SIAM J. Comput.*, 39(3):874–903, 2009.
30. P. Q. Nguyen and B. Vallée, editors. *The LLL Algorithm: Survey and Applications*. Information Security and Cryptography. Springer, 2010.
31. P. Q. Nguyen and T. Vidick. Sieve algorithms for the shortest vector problem are practical. *J. of Mathematical Cryptology*, 2(2):181–207, 2008.
32. A. M. Odlyzko. The rise and fall of knapsack cryptosystems. In *Cryptology and Computational Number Theory*, volume 42 of *Proc. of Symposia in Applied Mathematics*, pages 75–88. A.M.S., 1990.
33. M. Pohst. On the computation of lattice vectors of minimal length, successive minima and reduced bases with applications. *SIGSAM Bull.*, 15(1):37–44, 1981.
34. O. Regev. *On the Complexity of Lattice Problems with Polynomial Approximation Factors*. 2010. In [30].
35. C.-P. Schnorr. A hierarchy of polynomial lattice basis reduction algorithms. *Theoretical Computer Science*, 53(2-3):201–224, 1987.
36. C.-P. Schnorr. A more efficient algorithm for lattice basis reduction. *J. Algorithms*, 9(1):47–62, 1988.
37. C.-P. Schnorr and M. Euchner. Lattice basis reduction: improved practical algorithms and solving subset sum problems. *Math. Programming*, 66:181–199, 1994.
38. C.-P. Schnorr and H. H. Hörner. Attacking the Chor-Rivest cryptosystem by improved lattice reduction. In *Proc. of Eurocrypt '95*, volume 921 of *LNCS*, pages 1–12. IACR, Springer-Verlag, 1995.
39. A. Schönhage. Factorization of univariate integer polynomials by diophantine aproximation and an improved basis reduction algorithm. In *Proc. ICALP '84*, volume 172 of *Lecture Notes in Computer Science*, pages 436–447. Springer, 1984.
40. V. Shoup. Number Theory C++ Library (NTL) version 5.4.1. Available at `http://www.shoup.net/ntl/`.
41. C. L. Siegel. *Lectures on the Geometry of Numbers*. Springer, 1989.
42. D. Stehlé. Floating-point LLL: theoretical and practical aspects. 2010. In [30].
43. X. Wang, M. Liu, C. Tian, and J. Bi. Improved Nguyen-Vidick heuristic sieve algorithm for shortest vector problem. Cryptology ePrint Archive, Report 2010/647, 2010. `http://eprint.iacr.org/`.