LLL AND STOCHASTIC SANDPILE MODELS

JINTAI DING, SEUNGKI KIM, TSUYOSHI TAKAGI, YUNTAO WANG

Abstract. We introduce stochastic sandpile models which imitate numerous aspects of the practical behavior of the LLL algorithm with compelling accuracy. In addition, we argue that the physics and mathematics of sandpile models provide satisfactory heuristic explanations to much of the mysteries of LLL, and pleasant implications for lattice-based cryptography as a whole. Based on these successes, we suggest a paradigm in which one regards blockwise reduction algorithms as 1-d stochastic self-organized criticality(SOC) models and study them as such.

1. Introduction

1.1. **The mysteries of LLL.** The LLL algorithm ([20]) is one of the most celebrated algorithmic inventions of the twentieth century, with countless applications to pure and computational number theory, computational science, and cryptography. It is also the most fundamental of lattice reduction algorithms, in that nearly all known reduction algorithms are generalizations of LLL in some sense, and they also utilize LLL as their subroutine. (We refer the reader to [26] for a thorough survey on LLL and these related topics.) Thus it is rather curious that much of the observed behavior of LLL in practice is left totally unexplained, not even in a heuristic, speculative sense, even to this day.

The most well-known among the mysteries of LLL is the gap between its worst-case root Hermite factor(RHF) and the observed average-case, as documented in Nguyen and Stehlé ([25]). It is a theorem from the original LLL paper ([20]) that the shortest vector of an LLL-reduced basis, with its determinant normalized to 1, has length at most $(4/3)^{\frac{n-1}{4}} \approx 1.075^n$, whereas in practice one almost always observes $\approx 1.02^n$, regardless of the way in which the input is sampled. This is a strange phenomenon in the light of the work of Kim and Venkatesh ([18]), which, roughly speaking, proves that, for almost every lattice, nearly all of its LLL bases have RHF close to the worst bound. This leads to the suspicion that the LLL algorithm must be operating in a complex manner that belies the simplicity of its code.

There are also many other phenomena regarding LLL that are unaccounted for. One is the geometric series assumption(GSA), originally proposed by Schnorr ([31]), and its partial failure at the boundaries, both of which are observed in other blockwise reduction algorithms as well e.g. BKZ ([32]). There are also questions raised regarding the time complexity of LLL. Nguyen and Stehlé ([25]) suggest that, in some situations, the average time complexity is lower than the worst-case, and in others, the worst-case is attained. The complexity of the optimal LLL algorithm — i.e. the parameter δ equals 1 — is not proven to be polynomial-time, although observations suggest that it is (see Akhavi ([1]) and references therein).

1.2. **Cryptographic considerations.** In principle, LLL is of fundamental importance to lattice-based cryptography, hence it deserves to be understood well. Our understanding

of LLL may affect our understanding of all other reduction algorithms, particularly BKZ, the current standard method for challenging lattice-based systems.

Specifically, there are questions regarding LLL — and any reduction algorithm, really — that one may be happy to resolve, from the cryptographic perspective. For instance, it could be that a yet undiscovered small trick may improve the RHF of LLL without meaningfully increasing the time complexity, say, to 1.002. Absurd as this may sound at first, recall that we already "improved" LLL from (RHF) ≈ 1.075 to ≈ 1.02 merely by implementing it, and that currently we lack the device to form even a vague argument against such catastrophic possibility.

It seems that the practitioners of lattice-based cryptography are well aware of such uncertainties as to the scope and limit of reduction algorithms, and are reacting accordingly. According to Tables 5–10 of [2], most lattice-based submissions to the recent NIST call for proposals ([22]) claim about half or less as may bits of security as estimated with the state-of-art techniques. On the other hand, one observes no such reservation in submissions from multivariate cryptography, for example.

1.3. **Summary of this paper.** Our main contention is that the LLL algorithm may be understood as a kind of a *sandpile model* ([3], [7]), and that this idea may open up a path to a systematic understanding of the various unexplained phenomena of LLL. In particular, the rich and diverse tools from mathematics and physics employed to study sandpile models are now available for LLL as well.

There already have been a few previous works that point out or indirectly take advantage of the formal analogy between lattice reduction algorithms and sandpile-style cellular automata. This is, to the best of our knowledge, first discussed in Madritsch and Vallée ([27]); Vallée ([35]) also points out the analogy, albeit briefly. Hanrot, Pujol, and Stehlé ([14]) is quite close in spirit to the present paper in the literature, and so is Bai, Stehlé, and Wen ([4]) in that they propose a stochastic sandpile-like model of BKZ.

Our work takes a step further in that we take the sandpile idea seriously, not as a mere analogy but as what LLL really is. We introduce new sandpile models that capture the practical behavior of LLL both qualitatively and quantitatively, and we demonstrate the explanatory power of the mathematical and physical theories of sandpile models when applied to the LLL algorithm.

Specifically, we propose two sandpile models that we named LLL-SP (Algorithm 2) and SSP (Algorithm 3) respectively. LLL-SP is a sandpile model proper that exhibits nearly identical quantitative behavior to that of LLL in many aspects, suggesting that the two algorithms operate under the same principles. This claim can be formulated in a precise language, in terms of the mixing property of the μ variables. SSP is a simple stochastic variant of the abelian sandpile model (ASM) that serves as a useful toy model for LLL. SSP is mathematically far more tractable than LLL-SP; indeed, many statements that we wish are true for LLL can be proven for SSP. Moreover, SSP also closely imitates the typical output statistics of LLL.

After introducing the two models of LLL, we exhibit how the sandpile heuristic can be applied to explain a number of LLL behavior, with varying degrees of rigor:

- The gap between the average-case and the worst-case RHF is natural and expected — the existence of the gap can be proved for SSP (Theorem 2).
- Finite-size scaling theory from statistical physics explains various features of the average output shape of LLL.

- Time complexity of LLL is likely bounded from below (in a statistical sense), by a quantity of the same order as the upper bound — this can be proved for LLL-SP (Theorem 3).
- Any "generic" input distribution yields nearly identical output shapes, hence nearly identical RHF.
- It is infeasible that the RHF of LLL can be improved without paying the cost in complexity.

To keep this paper at a reasonable length, we only provide a sketch, and defer the detailed technical studies to the sequels. However, all the core ideas are already stated in the present paper, and are supported by pilot studies.

Our work has direct implications on the study of the behavior of other blockwise reduction algorithms, since they could also be given a sandpile interpretation; e.g. for BKZ it is done in [4]. For example, it is said that understanding the failure of GSA in the "head" of BKZ-reduced bases is important for cryptoanalysis ([36], [4]). In statistical physics, this is a known and understood phenomenon, via the finite-size scaling theory; see Section 4.2 below.

1.4. **Assumptions and notations.** Throughout this paper, instead of the original LLL reduction from [20], we work with its Siegel variant, a slight simplification of LLL. The Siegel reduction shares with LLL all its idiosyncrasies, and a bit easier to handle technically; hence a reasonable starting point for our research.

n always means the dimension of the relevant Euclidean space. Our lattices in $\mathbb{R}^n$ always have full rank.

A basis $\mathcal{B}$, besides its usual definition, is an *ordered* set, and we refer to its i-th element as $\mathbf{b}_i$. Denote by $\mathbf{b}_i^*$ the component of $\mathbf{b}_i$ orthogonal to all vectors preceding it, i.e. $\mathbf{b}_1, \dots, \mathbf{b}_{i-1}$. Also, for $i > j$, define $\mu_{i,j} = \langle \mathbf{b}_i, \mathbf{b}_j^* \rangle / \langle \mathbf{b}_j^*, \mathbf{b}_j^* \rangle$. Thus the following equality holds in general:

$$\mathbf{b}_i = \mathbf{b}_i^* + \sum_{j=1}^{i-1} \mu_{i,j} \mathbf{b}_j^*.$$

We will write for shorthand $\alpha_i := \|\mathbf{b}_i^*\| / \|\mathbf{b}_{i+1}^*\|$, and $Q_i = (\alpha_i^{-2} + \mu_{i+1,i}^2)^{-1/2}$. When discussing lattices, $r_i := \log \alpha_i$, and when discussing sandpiles, r_i refers to the "amount of sand" at vertex i.

1.5. **Acknowledgments.** We thank Deepak Dhar and Phong Nguyen for numerous helpful comments and suggestions.

2. Modeling LLL by a sandpile

2.1. **The LLL algorithm.** We briefly review the LLL algorithm; for details, we recommend [20], in which it is first introduced, and also [15] and [26]. A pseudocode for the LLL algorithm is provided in Algorithm 1. Be reminded that, whenever we mention LLL, we are really referring to its Siegel variant.

Proposition 1. *After carrying out Step 5 in Algorithm 1, the following changes occur:*

(i) $\alpha_{k-1}^{new} = Q_k \alpha_{k-1}$
(ii) $\alpha_k^{new} = Q_k^{-2} \alpha_k$
(iii) $\alpha_{k+1}^{new} = Q_k \alpha_{k+1}$
(iv) $\mu_{k,k-1}^{new} = \mu_{k+1,k-1}$
(v) $\mu_{k+1,k}^{new} = Q_k^2 \mu_{k+1,k}$

Algorithm 1 The LLL algorithm (Siegel variant)

0. Input: a basis $\mathcal{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ of $\mathbb{R}^n$, a parameter $\delta < 0.75$
1. while true, do:
2. Size-reduce $\mathcal{B}$.
3. (Lovász test) choose the lowest $k \in \{1, \ldots, n-1\}$ such that $\delta \|\mathbf{b}_k^*\|^2 > \|\mathbf{b}_{k+1}^*\|^2$
4. if there is no such k, break
5. swap $\mathbf{b}_k$ and $\mathbf{b}_{k+1}$ in $\mathcal{B}$
6. Output $\mathcal{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$, a δ-reduced LLL basis.

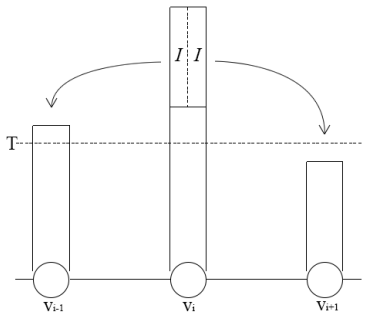

FIGURE 1. An illustration of a (legal) toppling T_i.

(vi) $\mu_{k+2,k+1}^{new} = \mu_{k+2,k} - \mu_{k+2,k+1}\mu_{k+1,k}$
(vii) $\mu_{k,l}^{new} = \mu_{k+1,l}, \mu_{k+1,l}^{new} = \mu_{k,l}$ *for* $1 \le l \le k-1$
(viii) $\mu_{l,k}^{new} = \mu_{l,k+1} - \mu_{l,k+1}\mu_{k+1,k}\mu_{k+1,k}^{new} + \mu_{l,k}\mu_{k+1,k}^{new}$ *for* $l \ge k+2$
(ix) $\mu_{l,k+1}^{new} = \mu_{l,k} - \mu_{l,k+1}\mu_{k+1,k}$ *for* $l \ge k+2$

and there are no other changes. The superscript "new" refers to the corresponding variable after the swap.

Proof. Straightforward calculations (see e,g, [20]). □

2.2. **Sandpile basics.** We also briefly review the basics of the sandpile models. For references, see Dhar ([8], [9]) or Perkinson ([28]).

A sandpile model is defined on a finite graph $\mathcal{G}$, with one distinguished vertex called the *sink*. In the present paper, we only concern ourselves with the cycle graph, say A_n, consisting of vertices $\{v_1, \ldots, v_n\}$ and one unoriented edge for each adjacent pair v_i and v_{i+1}. We also consider v_1 and v_n as adjacent. We designate v_n as the sink.

A *configuration* is a function $f : \{v_1, \ldots, v_n\} \to \mathbb{R}$. Just as reduction algorithms work with bases, sandpile models work with configurations. We write for short $r_i = f(v_i)$. One may think of r_i as the amount or *height* of the pile of sand placed on v_i.

Just as LLL computes a reduced basis by repeatedly swapping neighboring basis vectors, sandpiles compute a *stable configuration* by repeated *toppling*. Let $T, I \in \mathbb{R}_{>0}$. A configuration is *stable* if $r_i \le T$ for all $i \ne n$. A *toppling operator* T_i $(i \ne n)$ replaces r_i by $r_i - 2I$, and r_{i-1} by $r_{i-1} + I$ and r_{i+1} by $r_{i+1} + I$. An illustration is provided in Figure 1. Applying T_i when $r_i > T$ is called a *legal toppling*. By repeatedly applying legal topplings, all excess "sand" will eventually be thrown away to the sink, and the process will terminate.

In our paper, T — *threshold* — will always be a fixed constant, but I — *increment* — could be a function of the current configuration, or a random variable, or both. In the former case, we say that the model is *nonabelian* — otherwise *abelian*. In the second case, we say that the model is *stochastic*. The (non-stochastic) abelian sandpile theory is quite well-developed, with rich connections to other fields of mathematics — see e.g. [21]. Other sandpile models are far less understood, especially the nonabelian ones.

2.3. **The LLL sandpile model.** Motivated by Proposition 1, especially the formulas (i) – (iii), we propose the following Algorithm 2, which we call the *LLL sandpile model*, or LLL-SP for short.

Algorithm 2 The LLL sandpile model (LLL-SP)

0. Input: $\alpha_1, \ldots, \alpha_n \in \mathbb{R}$, $\mu_{2,1}, \ldots, \mu_{n,n-1} \in [-0.5, 0.5]$, a parameter $\delta < 0.75$
1. Rewrite $r_i := \log \alpha_i$, $\mu_i := \mu_{i+1,i}$ $T := -0.5 \log \delta$
2. while true, do:
3. choose the lowest $k \in \{1, \ldots, n-1\}$ such that $r_k > T$
4. if there is no such k, break
5. subtract $2 \log Q_k$ from r_k
6. add $\log Q_k$ to r_{k-1} (if $k - 1 \geq 1$) and r_{k+1} (if $k + 1 \leq n - 1$)
7. (re-)sample $\mu_{k-1}, \mu_k, \mu_{k+1}$ uniformly from $[-0.5, 0.5]$
8. Output: real numbers $r_1, \ldots, r_{n-1} \leq T$

The only difference between LLL (Algorithm 1) and LLL-SP (Algorithm 2) lies in the way in which the μ's are replaced after each swap or topple. Our experimental results below demonstrate that this change hardly causes any difference in their behavior. A theoretical perspective is discussed at the end of this section.

The decision to sample μ_i's uniformly is largely provisional, though some post hoc justification is provided in Figure 4. One could refine the model by updating μ_i's with the formulas in Proposition 1, and then re-sampling $\mu_{i+2,i}$'s uniformly.

2.4. **Numerical comparisons.** Figure 2 shows the average shape of the output bases and configurations by LLL and LLL-SP. We also ran the same experiments where the method of choosing k are tweaked. We omitted them here due to limited space, but the same lessons are obtained anyway.

For each dimension $n = 80, 100, 120$, we ran each algorithm 5,000 times with the same set of input bases of determinant $\approx 2^{10n}$, generated using the standard method suggested in [25]. A point (i, y) in each plot indicates that the average of $r_i := \log \alpha_i$'s over those 5,000 outputs equal y. We used fpLLL ([10]) for the LLL algorithm.

One easily observes that the algorithms yield nearly indistinguishable outputs. In particular, since RHF can be computed directly from the r_i's by the formula

$$\text{RHF} = \exp\left(\frac{1}{n^2}\sum_{i=1}^{n-1}(n-i)r_i\right), \tag{1}$$

we expect both algorithms to yield about the same RHF. Indeed, Figure 3 shows that the RHF distribution of LLL and LLL-SP are in excellent agreement.

The resemblance of the two algorithms runs deeper than on the level of output statistics. See Figure 4, which depicts the plot of points $(i, Q_{k(i)}^{-2})$ and $\mu_{k(i)+1,k(i)} = \mu_{k(i)}$ as we ran

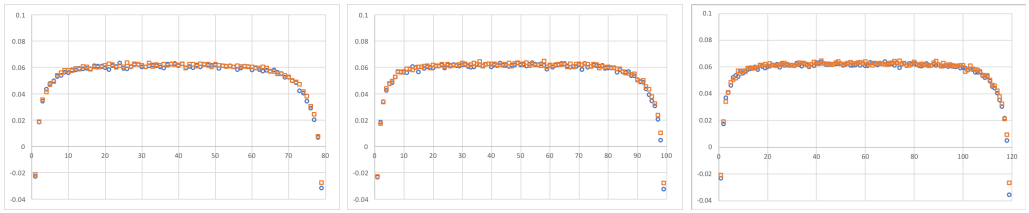

FIGURE 2. Average output of LLL (orange square) and LLL-SP (blue circle) in dimensions 80, 100, and 120.

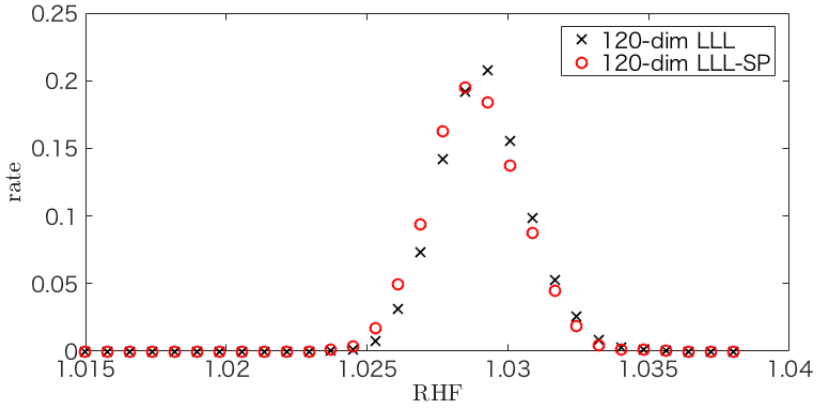

FIGURE 3. RHF distributions of LLL and LLL-SP in dimension 120. We obtain the same results in other dimensions.

LLL and LLL-SP on dimension 80, where $k(i)$ is k chosen at i-th iteration.[1] The two plots are again indistinguishable, another solid evidence that LLL and LLL-SP have essentially identical dynamics.

There is a small caveat: the input has to be generic in a certain sense, or otherwise LLL and LLL-SP will behave differently. For example, if the underlying lattice of an input basis has a large gap in the i-th and $i+1$-st successive minima, this does not affect LLL-SP in any way, yet LLL would treat the input as essentially two bases of dimension i and $n-i$ respectively, yielding a better output.

2.5. **Discussion.** The only difference between LLL and LLL-SP has to do with the way they update the $\mu_k (= \mu_{k+1,k})$'s. For LLL-SP, the μ_k-variables are i.i.d. and independent of the r_k-variables. For LLL, μ_k is determined by a formula involving its previous value and r_k. However, it seems plausible that the μ_k's in LLL, as a stochastic process, is *mixing*, which roughly means that they are close to being i.i.d, in the sense that a small perturbation in μ_k causes the next value μ_k^{new} to become near unpredictable. Numerically, this is robustly supported by the graphs at the botton of Figure 4. Theoretically, our intuition comes from the fact that the formula $\mu_k^{new} = \mu_k/(\mu_k^2 + \alpha_k^{-2}) \pmod 1$ is an approximation of the Gauss map $x \mapsto \{1/x\}$, which is proven to have excellent mixing properties (see [29]). This idea can be formulated in the form of a mathematical conjecture,

[1]We have the same results in higher dimensions, but they are too cumbersome to present.

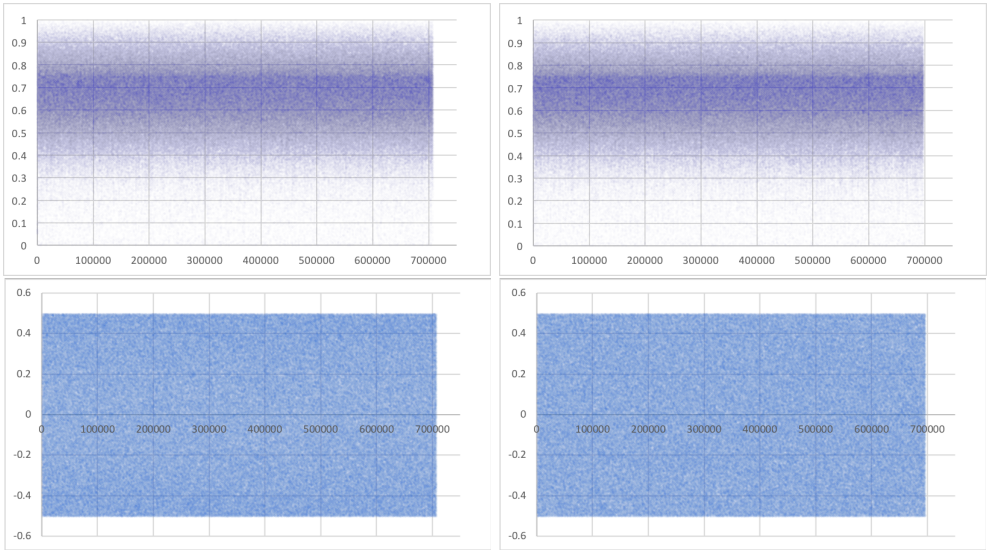

FIGURE 4. Plot of $Q_{k(i)}^{-2}$(top) and $\mu_{k(i)+1,k(i)} = \mu_{k(i)}$(bottom) during a typical run of LLL(left) and LLL-SP(right) in dimension 80.

which can then be considered a rigorous version of the statement "LLL is essentially a sandpile model."

3. Abelian sandpile analogue of LLL

The drawback of LLL-SP is that it is difficult to study, since it is nonabelian. The literature on nonabelian sandpile models is very thin, and to the best of our knowledge, there is no theoretical treatment on this subject. We hope that our work here provides some motivation to pursue nonabelian models in more depth.

In this section, we introduce a certain abelian stochastic sandpile model that we named SSP, which is in a sense an abelianized version of LLL-SP. A priori, SSP appears completely unrelated to LLL. Surprisingly, though, its average output shape turns out to be extremely close to that of LLL. Moreover, SSP has a mathematical theory that is analogous to that of ASM developed by Dhar ([7], see also [9]), which is developed in a separate paper by the second-named author [19]. Therefore, SSP is a useful toy model that could yield insights into some of the most prominent features of the output statistics of LLL. It may yield some hints as to how to start analyzing the non-abelian LLL-SP as well.

3.1. **Introduction to SSP.** A pseudocode for SSP is provided in Algorithm 3. This is exactly the same as ASM, except for Step 4, which determines the amount of sand to be toppled at random. The decision to sample from the uniform distribution is an arbitrary one; we could have chosen something else, and much of the discussion below still apply.

3.2. **Output shape of SSP.** The average output shape of this stochastic sandpile model (SSP) is shown in Figure 5. Figure 5 not only shares all the major characteristics of Figure 2, but they are also quantitatively alike. The values r_i's are nearly identical in the middle, which gradually decreases as i approaches the boundary, starting at around $i = 15$ and $n-15$. Furthermore, in both figures, the differences between the threshold and the middle values, and the differences between the middle and the boundary value, are equal; in the

Algorithm 3 Stochastic sandpile (SSP)

0. Input: $r_1, \ldots, r_{n-1} \in \mathbb{Z}$, parameters $T, I \in \mathbb{Z}$, $I > 0$
1. while true, do:
2. choose a $k \in \{1, \ldots, n-1\}$ such that $r_k > T$
3. if there is no such k, break
4. sample γ uniformly from $\{1, \ldots, 2I-1\}$
5. subtract 2γ from r_k
6. augment γ to r_{k-1} and r_{k+1}
7. Output: integers $r_1, \ldots, r_{n-1} \leq T$

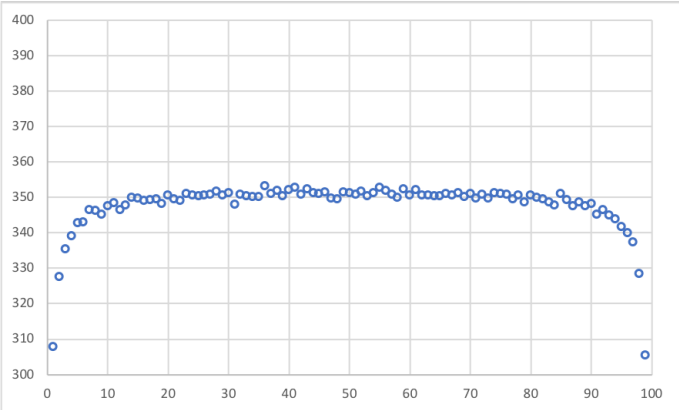

FIGURE 5. Average output of SSP, $n = 100$, $I = 100$ and $T = 400$.

case of SSP, it is $\approx I/2$, and for LLL, it equals about 0.08. Outside the paradigm we are developing here, this should come as quite surprising: Algorithms 1 and 3 are simply so different that there is no reason to expect that anything similar would come out of them.

This finding suggests that sandpiles may shed light on the geometric series assumption(GSA) and its partial failure at the boundary, yet another important unexplained phenomenon of LLL-like reduction algorithms. Indeed, "GSA" is a quite general phenomenon that is fairly well-understood in statistical physics. We provide a more detailed discussion in the next section.

3.3. **Mathematical properties of SSP.** A mathematical theory of SSP closely analogous to that of ASM has been recently developed ([19]), largely motivated by the experimental result above. Below we discuss some of its most important implications.

As in the case of ASM, the stable configurations of SSP form a monoid, and there exist recurrent configurations. However, the most important result is that SSP possesses the unique *steady state*, the probability distribution on the set of outputs that one approaches as one repeatedly runs the algorithm with sufficiently randomized inputs. Figure 5 is a reflection of the steady state of SSP, from which one can compute its average "RHF" via (1).

The notion of the steady state alone clarifies numerous aspects of the practical behavior of LLL. The number "1.02" immediately obtains a neat mathematical meaning: an invariant of the steady state. Moreover, by definition, a steady state is independent of the input distribution, which explains why the number 1.02 seems independent of how

the input bases are sampled for LLL, an observation in [24] that is also declared as a conjecture there.

Therefore, the whole question regarding the average output of LLL — why 1.02, why GSA, why independent of input sampling — comes down to studying the quantitative properties of the steady state. Even for SSP, this is highly nontrivial, where it reduces to a tricky problem in combinatorics. Still, it is possible to prove a number of statements that we wish are true for LLL. For example, it is possible to rigorously bound the average RHF of SSP from above:

Theorem 2. *The worst-case* $\log(\mathrm{RHF})$ *of SSP is* $T/2 + o_n(1)$. *The average* $\log(\mathrm{RHF})$ *of SSP is bounded from above by* $T/2 - I/e^2 + o_n(1)$.

(Empirically $\log(\mathrm{RHF}) \approx T/2 - I/4$ *on average.)*

Proof. This is an immediate corollary of Proposition 8 of [19]. The idea is to estimate the number of stable configurations with $\log(\mathrm{RHF}) > T/2 - \alpha$, and multiply it by the maximum point mass $m \approx I^{-(n-1)}$ of the steady state to bound the proportion of such configurations in the steady state. For any $\alpha < 1/me^2$ one can easily show that it approaches zero as $n \to \infty$. □

4. Implications

The previous two sections strongly suggest that we view and study LLL as a member of the family of one-dimensional[2] stochastic self-organized criticality(SOC) models. From the standpoint of physics, this is a perfectly fine thing to do. Dhar ([8], [9]) wrote a few surveys on SOC models in general. There are also plenty of works in physics on one-dimensional stochastic SOC models (e.g. [13], [30]) that may serve as good references as well.

In this section, we discuss how our new perspective sheds light on the practical behavior of LLL, most of which have been left completely unexplained to this day. We try to pursue as much rigor as we can, yet it must be taken into consideration that the sheer complexity of SOC models makes it sometimes unreasonable to ask for. Still, at the very least, our arguments are based on well-established theories in physics and convincing heuristics based on the mathematics of sandpile models.

4.1. RHF of LLL.

The RHF is a quantity associated to the critical point — or the steady state — of the system under question, be it a sandpile model or a reduction algorithm. In this sense, asking why the average RHF of LLL is ≈ 1.02 is akin to asking why water freezes at 0°C rather than some other number. Specifically in our case, part of the difficulty is that the steady state does not appear to have a concise mathematical description, as can be seen in the case of SSP (see [19]).

However, the sandpile perspective at least makes it clear that LLL is expected to do better than the theoretical worst bound. A quick explanation is illustrated in Figure 6. First, the area of the set of LLL bases with high RHF, the simplex cut out by the equation $\mathrm{RHF} = 1.074$ from the cube $[0, T]^n$, is small; this can be easily proven. Second, the increment $\log Q_i$ of LLL tends to be too large to step on it — indeed, the length of each "step" is greater than $2 \log Q_i$, which can be as large as r_i not infrequently. Perhaps this idea can be made rigorous to give an upper bound on the average RHF of LLL-SP.

Alternatively, one could follow the proof of Theorem 2 and try to prove that the density of the "steady state" of LLL on that simplex is vanishingly small, for which it suffices

[2]This refers to the "dimension" of the underlying graph $\mathcal{G}$, not to be confused with the lattice dimension.

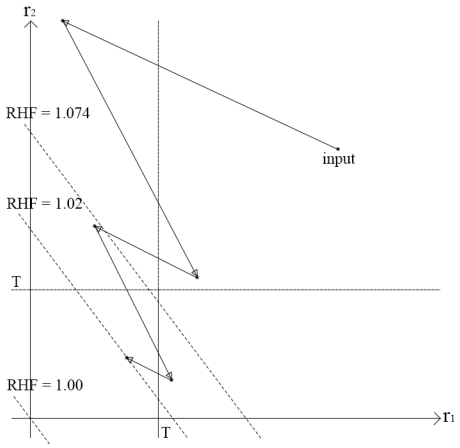

FIGURE 6. An illustration of why LLL cannot hit the worst bases.

to bound the maximum density. This seems to be a promising approach for LLL-SP, especially with the usual order of toppling which facilitates the use of induction, but we have been yet unable to provide a proof.

4.2. **Finite-size scaling theory and output shape.** The characteristics of the average output shapes of LLL, LLL-SP, and SSP are in fact a quite general phenomenon of stochastic sandpile models on one-dimensional graphs (see [30] and [13]), and they can be explained by finite-size scaling theory (FSS; see [6]). According to the theory, to each model there exists a single constant ν that simultaneously governs the average r_i of the steady state in the n limit (hence RHF as well), the variance of r_i, and the degree of "curviness" at the boundaries — see Section III. A of [13]. FSS in the context of lattice reduction would explain at once a number of aspects of the average output shape, such as GSA, its failure at the boundaries, and the apparent convergence of RHF as $n \to \infty$.

We will provide an extensive investigation of this topic in a sequel; for now we provide a brief demonstration. Figure 7 depicts the graph of the first 16 points of $m - \mathbb{E}(r_i)$ on $\log_2$-$\log_2$ scale, where $m = \max_i \mathbb{E}(r_i)$, and with the values $\mathbb{E}(r_i)$ coming from the data for dimension 120 LLL in Figure 2. It is one of the predictions of FSS that the first few (relative to n) of these values form a line, which Figure 7 serves to substantiate for the case of LLL. The *critical exponent* ν is the absolute value of the slope of this line, which for LLL Figure 7 suggests that $\nu \approx 1$. Interestingly, according to the data in [19] SSP also has $\nu \approx 1$, suggesting that LLL and SSP belong to the same *universality class*.

In the case of BKZ, the curvy shape of the left boundary (that is, r_i's for i small) has recently been of interest ([36], [4]). FSS in principle applies to BKZ as well, and thus it is also expected that its counterpart of Figure 7 forms a line too — if true this provides a satisfactory description of the boundary phenomenon.

4.3. **Regarding time complexity.** It is possible to prove some nice statements regarding the time complexity of LLL-SP. For example, the theorem below gives a probabilistic lower bound on the complexity of LLL-SP, which agrees up to constant factor with the well-known upper bound. There are two ingredients in the proof: (i) measuring the progress of the LLL algorithm by the quantity *energy*, a well-known idea from [20] (ii) bounding the performance of LLL-SP by a related SSP.

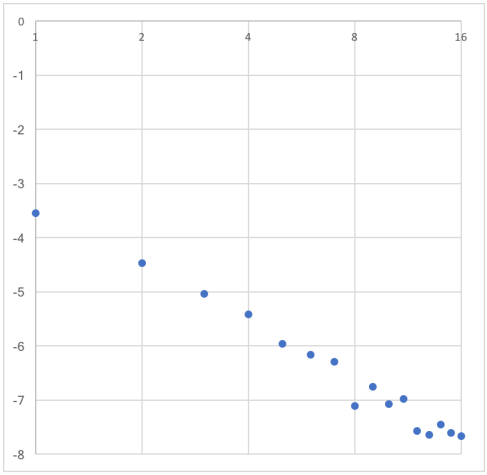

FIGURE 7. Graph of $\log_2(m - \mathbb{E}(r_i))$ for LLL as a function of i, where $m = \max_i \mathbb{E}(r_i)$.

Theorem 3. *Consider LLL-SP, and an input configuration $\mathcal{B}$ whose* log-energy $E = E(\mathcal{B})$*, defined by*

$$E(\mathcal{B}) = \sum_{j=1}^{n-1} \sum_{i=j}^{n-1} (n-i) r_i,$$

is sufficiently large, e.,g. greater than a large constant multiple of (2) *below. Suppose the order of toppling does not distort the distribution of the μ's — precisely speaking, take the notion of a* choice algorithm *in the sense of* [19]*. Then the probability that LLL-SP is not terminated in $E/2$ steps is at least $1 - CE^{-1/2}$ for an absolute constant $C > 0$.*

Proof. If the algorithm is terminated, then E must have become less than

$$\sum_{i=1}^{n} (n-i+1)(n-i)T/2,$$

(where $T := -\log \delta^{1/2} > 0$) which equals,

$$\frac{T}{6}(n^3 - n). \tag{2}$$

Taking converse, we see that if E is greater than (2), then LLL-SP has not yet terminated. At k-th toppling, E decreases by at most $\log \mu_{i(k)}^{-2}$, where $i(k)$ is the index of the vertex in which k-th toppling occured. If toppled N times, the decrease in E is bounded by at most $F_N := \sum_{k=1}^{N} \log \mu_{i(k)}^{-2}$. In sum,

$$\mathrm{Prob}(E - F_N > T(n^3 - n)/6) \tag{3}$$

gives the lower bound on the probability that LLL-SP is not terminated after N swaps. Hence, it suffices to show that (3) is bounded from below by $1 - CE^{-1/2}$ when $N = E/2$.

The central limit theorem is applicable on F_N, since $\mu_{i(k)}$ are i.i.d. More precisely, we apply the Berry-Esseen theorem, which gives an estimate of the error cost we pay by approximating the distribution of F_N by a normal distribution, namely $cN^{-1/2}$ where $c > 0$ is an explicit absolute constant. If $N = E/2$, one can compute, e.g. by the Chebyshev's inequality, that (3) with F_N replaced by a proper normal distribution is at

least $1 - 5E^{-1}$. But the error term here is subsumed to that of the Berry-Esseen, so that (3) is bounded from below by $1 - CE^{-1/2}$ for some constant $C > 0$, as desired.

□

For another example, regarding the optimal (i.e. $\delta = 0.75$) LLL problem (see e.g. [1]), the optimal LLL-SP can be easily shown to terminate in polynomial time with probability arbitrarily close to one, and perhaps even on average. The exception happens only if $\mu_i \approx 0.5$ most of the time, but this is clearly improbable. This observation also suggests that it is sensible to take this probabilistic approach.

4.4. **Random lattice.** Our work also clarifies the notion of a *random lattice*, central to the cryptanalysis of lattice systems but without a precise definition. It has been typically understood in terms of the "Haar measure" on $\mathrm{SL}_n\mathbb{Z}\backslash\mathrm{SL}_n\mathbb{R}$ (see [33]), but as [18] suggests this measure does not reflect the practical behavior of LLL. From the physical point of view, a random lattice is simply any generic distribution on the set of input bases that stabilizes to an approximate "steady state" of the reduction algorithm in question. For SSP, this statement can be made mathematically precise and then be proven — see Proposition 7 of [19]. For LLL and LLL-SP, we have two heuristic explanations: (i) the parallelepiped argument used to prove Proposition 7 of [19] can be applied (ii) if the input is large enough, $\log Q_i \approx -\log \mu_{i+1,i}$ so LLL-SP behaves like an SSP with increment $-\log \mu_{i+1,i}$. Note that this provides an explanation for the conjecture of Nguyen and Stehlé ([24]) that the average RHF is independent of how the inputs are sampled.

This notion of random lattice hopefully liberates cryptanalysts from the Haar measure to some extent, which is cumbersome to work with both theoretically and experimentally. For instance, although the results in Section 2 above are obtained with inputs of $\det \approx 2^{10n}$, the typical choice to ensure that the Hecke equidistribution approximately holds, the same experiments using $\det \approx 2^{2n}$, or even knapsack-type bases with $B = 2^{20n}$ (see [25] for definition) which are far smaller, yield practically identical outcomes.

4.5. **Can LLL be improved?** As seen above, LLL-SP has a probabilistic lower bound on time complexity, so it seems likely that LLL cannot be improved complexity-wise. One may also ask whether one can substantially improve the output quality of LLL without affecting its complexity — to be more precise, improve while fixing the toppling method that is swapping adjacent vectors. For the sake of lattice-based cryptography, we want the answer to be a no.

This is clearly a difficult problem, but thinking of LLL as a sandpile model, we may have an approach, which we outline below. From this perspective, it is unlikely that perturbation-based methods would work, since the steady state is invariant under such maneuver. The remaining method for tweaking LLL is to modify the toppling order, so that the increment is consistently large, or equivalently in our case, $|\mu_{k(i)}|$ consistently small. This needs to be combined with probing the "search tree" — precisely speaking, the Lenstra graph (see [16]) — in advance, since otherwise there are not many options one could choose from. The greedy variant, which at each step chooses the pile that induces the biggest topple, achieves an improvement along this line; it perturbs the distribution of $|\mu_{k(i)}|$ so that on average $|\mu_{k(i)}| \approx 0.23$ rather than the expected $= 0.25$, improving RHF by < 0.002. One can play with sandpile models to guess how much distortion on $|\mu_{k(i)}|$ is needed to attained the desired improvement in RHF; for example, we ran LLL-SP where we *fix* the value of μ throughout, and obtained that $|\mu_{k(i)}| = 0.05$ is needed for RHF $= 1.012$. This suggests that it is indeed infeasible to improve LLL. One may try increasing the probing depth, but it would significantly slow down the algorithm.

References

[1] A. Akhavi. Worst-case complexity of the optimal LLL algorithm. *LATIN 2000: Theoretical Informatics*, 355-366.

[2] M. Albrecht, B. Curtis, A. Deo, Alex Davidson, Rachel Player, E. Postlethwaite, Fernando Virdia, Thomas Wunderer. Estimate all the {LWE, NTRU} schemes! Available at https://github.com/estimate-all-the-lwe-ntru-schemes.

[3] P. Bak, C. Tang, and K. Wieselfeld. Self-organized criticality: an explanation of $1/f$ noise. *Phys. Rev. Lett.*, 59:381-384, 1987.

[4] S. Bai, D. Stehlé, W. Wen. Measuring, simulating and exploiting the head concavity phenomenon in BKZ. *Advances in cryptology — ASIACRYPT 2018*. Part I, 369-404, Lecture Notes in Comput. Sci., 11272, Springer, Cham, 2018.

[5] D. Bernstein and T. Lange. Post-quantum cryptography. *Nature*, 549(7671):188-194, 2017.

[6] J. Cardy (ed). Finite-size scaling. Elsevier Science Publishers B.V., 1988.

[7] D. Dhar. Self-organized critical state of sandpile automaton models. *Phys. Rev. Lett.*, 64(14):1613-1616, 1990.

[8] D. Dhar. The abelian sandpile and related models. *Physica A*, 263(1999) vol. 4, 4-25.

[9] D. Dhar. Theoretical studies of self-organized criticality. *Physica A*, 369(2006) 29-70.

[10] The fpLLL team. fpLLL, a lattice reduction library. Available at https://github.com/fplll/fplll.

[11] N. Gama, P. Nguyen. Predicting lattice reduction. *Advances in cryptology — EUROCRYPT 2008*, 31-51, *Lecture Notes in Comput. Sci.*, 4965, Springer, Berlin, 2008.

[12] C. Gentry. A fully homomorphic encryption scheme. PhD thesis, Stanford University, 2009.

[13] P. Grassberger, D. Dhar, and P. K. Mohanty. Oslo model, hyperuniformity, and the quenched Edwards-Wilkinson model. *Physical review E* 94, 042314 (2016).

[14] G. Hanrot, X. Pujol, and D. Stehlé. Analyzing blockwise lattice algorithms using dynamical systems. *Advances in cryptology—CRYPTO 2011*, 447-464, Lecture Notes in Comput. Sci., 6841, Springer, Heidelberg, 2011.

[15] A. Joux and J. Stern. Lattice reduction: a toolbox for the cryptanalyst. *J. Cryptology* (1998) 11: 161. https://doi.org/10.1007/s001459900042

[16] S. Kim. On the shape of a high-dimensional random lattice. PhD thesis, Stanford University, 2015.

[17] J. Kim, M. Kim, and S. Kim. LLL via the Lenstra graph. Preprint. Available at https://sites.google.com/view/seungki/

[18] S. Kim and A. Venkatesh. The behavior of random reduced bases. *Int. Math. Res. Not.*, rnx074.

[19] S. Kim and Y. Wang. A stochastic variant of the abelian sandpile model. Preprint. Available at https://sites.google.com/view/seungki/

[20] A. K. Lenstra, H. W. Lenstra, Jr., and L. Lovász. Factoring polynomials with rational coefficients. *Math. Ann.*, 261(4):515-534, 1982.

[21] L. Levine. What is . . . a sandpile? *Notices Amer. Math. Soc.* 57 (2010), no. 8, 976-979.

[22] National Institute of Standards and Technology. Post-quantum cryptography, call for proposals. Available at https://csrc.nist.gov/Projects/Post-Quantum-Cryptography/Post-Quantum-Cryptography-Standardization/Call-for-Proposals.

[23] A. Neumaier and D. Stehlé. Faster LLL-type reduction of lattice bases. *Proceedings of the 2016 ACM International Symposium on Symbolic and Algebraic Computation*, 373-380, ACM, New York, 2016.

[24] P. Nguyen and D. Stehlé. Floating-point LLL revisited. *Advances in cryptology — EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Comput. Sci.*, 215-233. Springer, Berlin, 2005.

[25] P. Nguyen and D. Stehlé. LLL on the average. *Algorithmic number theory*, volume 4076 of *Lecture Notes in Comput. Sci.*, pages 238 - 256. Springer, Berlin, 2006.

[26] P. Nguyen and B. Vallee (eds). The LLL Algorithm: Survey and Applications. Springer, 2010.

[27] M. Madritsch and B. Vallé. Modelling the LLL algorithm by sandpiles. *LATIN 2010: theoretical informatics*, 267-281, *Lecture Notes in Comput. Sci.*, 6034, Springer, Berlin, 2010.

[28] D. Perkinson. Notes for AIMS Cameroon: Abelian Sandpile Model. Available at http://people.reed.edu/ davidp/cameroon

[29] V. Rohlin. Exact endomorphisms of a Lebesgue space. *Izvest. Akad. Nauk* 25 (1961), 499-530.

[30] T. Sadhu and D. Dhar. Steady state of stochastic sandpile models. *J. Stat. Phys.* (2009), 134: 427-441.

[31] P. Schnorr. Lattice reduction by random sampling and birthday methods. *STACS 2003*, 145-156, *Lecture Notes in Comput. Sci.*, 2607, Springer, Berlin, 2003.

[32] P. Schnorr and M. Euchner. Lattice basis reduction: improved practical algorithms and solving subset sum problems. *Math. Programming* 66 (1994), no. 2, Ser. A, 181-199.

[33] C.L. Siegel. A mean value theorem in geometry of numbers. Ann. of Math. 46(2) (1945), 340-347.
[34] Damien Stehlé. Floating-point LLL: theoretical and practical aspects. In Phong Nguyen and Brigitte Vallée (eds), *The LLL Algorithm, Survey and Applications*, Informaton Security and Cryptography, chapter 5, pages 179-213. Springer-Verlag, Berlin, Heidelberg, 2010.
[35] B. Vallée. Genealogy of lattice reduction: algorithmic description and dynamical analyses. Preprint.
[36] Y. Yu and L. Ducas. Second Order Statistical Behavior of LLL and BKZ. *Selected Areas in Cryptography*, SAC 2017, pp.3-22