On the decoding of Barnes-Wall lattices

Vincent Corlay†, Joseph J. Boutros‡, Philippe Ciblat†, and Loïc Brunel∗
† Telecom Paris, Institut Polytechnique de Paris, 91120 Palaiseau, France, v.corlay@fr.merce.mee.com
‡ Texas A&M University, Doha, Qatar, *Mitsubishi Electric R&D Centre Europe, Rennes, France

Abstract—We present new efficient recursive decoders for the Barnes-Wall lattices based on their squaring construction. The analysis of the new decoders reveals a quasi-quadratic complexity in the lattice dimension. The error rate is shown to be close to the universal lower bound in dimensions 64 and 128.

I. INTRODUCTION

Barnes-Wall (BW) lattices were one of the first series discovered with an infinitely increasing fundamental coding gain [2]. This series includes dense lattices in lower dimensions such as \( D_4 \), \( E_8 \), \( \Lambda_{16} \) [5], and is deeply related to Reed-Muller codes [9][17]: BW lattices admit a Construction D based on these codes. Multilevel constructions attracted the recent attention of researchers, mainly Construction C∗ [3], where lattice and non-lattice constellations are made out of binary codes. One of the important challenges is to develop lattices with a reasonable-complexity decoding where a fraction of the fundamental coding gain is sacrificed in order to achieve a lower kissing number. BW lattices are attractive in this sense. For instance the lattice \( BW_{128} \), with an equal fundamental coding gain as \( \text{Nebe}_{72} \) [20], sacrifices 1.5 dB of its fundamental coding gain with respect to \( MW_{128} \) [8] while the kissing number is reduced by a factor of 200.

Several algorithms have been proposed to decode BW lattices. Forney introduced an efficient maximum-likelihood decoding (MLD) algorithm in [9] for the low dimension instances of these lattices based on their trellis representation. Nevertheless, the complexity of this algorithm is exponential in the dimension and intractable for \( n > 32 \): e.g. decoding in \( BW_{64} \) involves \( 2 \cdot 2^{24} + 2 \cdot 2^{16} \) decoders of \( BW_{16} \) and decoding in \( BW_{128} \) involves \( 2 \cdot 2^{48} + 2 \cdot 2^{32} \) decoders of \( BW_{32} \) (using the two-level squaring construction to build the trellis, see [9, Section IV.B]). Later, [19] proposed the first bounded-distance decoders (BDD) running in polynomial time: a parallelisable decoder of complexity \( O(n^2) \) and another sequential decoder of complexity \( O(n \log^2(n)) \). The parallel decoder was generalized in [14] to work beyond the packing radius, still in polynomial time. It is discussed later in the paper. The sequential decoder uses the BW multilevel construction to perform multistage decoding: each of the \( \approx \log(n) \) levels is decoded with a Reed-Muller decoder of complexity \( n \log(n) \). This decoder was also further studied, in [15], to design practical schemes for communication over the AWGN channel. The performance of this sequential decoder is far from MLD. A simple information-theoretic argument explains why multistage decoding of BW lattices cannot be efficient: the rates of some component Reed-Muller codes exceed the channel capacities of the corresponding levels [13][28]. As a result, no BW decoders, being both practical and quasi-optimal on the Gaussian channel, have been designed and executed for dimensions greater than 32.

We present new decoders for BW lattices based on their \((u, u + v)\) construction [17]. We particularly consider this construction as a squaring construction [9] to establish a new recursive BDD (Algorithm 2, Section III-A), new recursive list decoders (Algorithms 3 and 5, Sections IV-B and IV-C), and their complexity analysis as stated by Theorems 2-4. As an example, Algorithm 5 decodes \( BW_{64} \) and \( BW_{128} \) with a performance close to the universal lower bound on the coding gain of any lattice and with a reasonable complexity almost quadratic in the lattice dimension.

II. PRELIMINARIES

Lattice. A lattice \( \Lambda \) is a discrete additive subgroup of \( \mathbb{R}^n \). For a rank-\( n \) lattice in \( \mathbb{R}^n \), the rows of a \( n \times n \) generator matrix \( G \) constitute a basis of \( \Lambda \) and any lattice point \( x \) is obtained via \( x = zG \), where \( z \in \mathbb{Z}^n \). The squared minimum Euclidean distance of \( \Lambda \) is \( d(\Lambda) = (2\rho(\Lambda))^2 \), where \( \rho(\Lambda) \) is the packing radius. The number of lattice points located at a distance \( \sqrt{d(\Lambda)} \) from the origin is the kissing number \( \tau(\Lambda) \). The fundamental volume of \( \Lambda \), i.e. the volume of its Voronoi cell and its fundamental parallelepiped, is denoted by \( \text{vol}(\Lambda) \). The fundamental coding gain \( \gamma(\Lambda) \) is given by the ratio \( \gamma(\Lambda) = d(\Lambda)/\text{vol}(\Lambda)^{\frac{1}{2}} \). The squared Euclidean distance between a point \( y \in \mathbb{R}^n \) and a lattice point \( x \in \Lambda \) is denoted \( d(x,y) \). Accordingly, the squared distance between \( y \in \mathbb{R}^n \) and the closest lattice point of \( \Lambda \) is \( d(y,\Lambda) \).

For lattices, the transmission rate used with finite constellations is meaningless. Poltyrev introduced the generalized capacity [22], the analog of Shannon capacity for lattices. The Poltyrev limit corresponds to a noise variance of \( \sigma_{\max}^2 = \det(\Lambda)^{\frac{3}{4}}/(2\pi e) \) and the point error rate is evaluated with respect to the distance to Poltyrev limit, i.e. \( \sigma_{\max}^2/\sigma^2 \).

BDD, list-decoding, and MLD. Given a lattice \( \Lambda \), a radius \( r > 0 \), and any point \( y \in \mathbb{R}^n \), the task of a decoder is to determine all points \( x \in \Lambda \) satisfying \( d(x,y) \leq r^2(\Lambda) \). If \( r < \rho(\Lambda) \), there is either no point or a unique point found and the decoder is known as BDD. Additionally, if \( d(x,y) < \rho^2(\Lambda) \), we say that \( y \) is within the guaranteed error-correction radius of the lattice. If \( r \geq \rho(\Lambda) \), there may be more than one point in the sphere. In this case, the process is called list-decoding rather than BDD. When list-decoding is used, lattice points within the sphere are enumerated and the decoded lattice point is the closest to \( y \) among them. MLD simply refers to finding the closest lattice point in \( \Lambda \) to
any point \(y \in \mathbb{R}^n\). If list-decoding is used, MLD is equivalent to choosing a decoding radius equal to \(R(\Lambda)\).

**Coset decomposition of a lattice.** Let \(\Lambda\) and \(\Lambda'\) be two lattices such that \(\Lambda' \subset \Lambda\). If the order of the quotient group \(\Lambda/\Lambda'\) is \(q\), then \(\Lambda\) can be expressed as the union of \(q\) cosets of \(\Lambda'\). We denote by \([\Lambda/\Lambda']\) a system of coset representatives for this partition. It follows that \(\Lambda = \bigcup_{x \in [\Lambda/\Lambda']} \Lambda + x\Lambda = \Lambda' + [\Lambda/\Lambda']\).

**The \(BW\) lattices.** Let the scaling-rotation operator \(R(2n)\) in dimension \(2n\) be defined by the application of the \(2 \times 2\) matrix

\[
R(2) = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}
\]

on each pair of components. I.e. the scaling-rotation operator \(BW\) increases infinitely as \(n\) increases. Assume (without loss of generality) that \(u_1\) is correct. We have \(d(y_2 - u_1, RBW_n) < \rho^2 (RBW_n)\). Therefore, \(y_2 - u_1\) is also correctly decoded.

As a result, among the two lattice points stored, at least one is the closest lattice point to \(y\).

Note that the \(BW_n\) decoder in the previous proof got exploited up to \(\rho^2 (BW_n)\) only. Consequently, Algorithm 1 should exceed the performance predicted by Theorem 1 given that step 1 is MLD.

Algorithm 1 can be generalized into the recursive Algorithm 2, where Steps 4, 5, and 6 of the latter algorithm replace Steps 1, 2, and 3 of Algorithm 1, respectively. This algorithm is similar to the parallel decoder of [19]. The main difference is that [19] uses the automorphism group of \(BW_2\) to get four candidates at each recursion whereas we use the scaling construction to generate only two candidates. Nevertheless, both our algorithm and [19] use four recursive calls at each recursive section and have the same asymptotic complexity.

**Algorithm 2** Recursive BDD of \(BW_n\) (where \(2n = 2^t\))

**Function** \(RecBW(y, t)\)

**Input:** \(y = (y_1, y_2) \in \mathbb{R}^t, 1 \leq t\).

1. if \(t = 1\) then
   2. \(x_{dec} \leftarrow ([y_1], [y_2])\) // Decoding in \(\mathbb{Z}^2\)
3. else
   4. \(u_1 \leftarrow RecBW(y_1, t-1), u_2 \leftarrow RecBW(y_2, t-1)\)
   5. if \(y_2 - u_1\) (and \(y_1 - u_2\)) should be decoded in \(RBW_n\)
   6. \(x' \leftarrow (u_1 + v_1, u_2 + v_2)\)
   7. \(x_{dec} = \arg\min_{x \in \{x', x'\}} ||y - x||\)
8. end if
9. Return \(x_{dec}\)

**Theorem 2.** Let \(n\) be the dimension of the lattice \(BW_n\) to be decoded. The complexity of Algorithm 2 is \(O(n^2)\).

**Proof.** Let \(C(n)\) be the complexity of the algorithm for \(n = 2^t\). We have \(C(n) = 4C(n/2) + O(n) = O(n^2)\).

**B. Performance on the Gaussian channel**

In the appendix (see [29, Section VII-A]), we show via an analysis of the effective error coefficient of Algorithm 2 that the loss in performance of this algorithm compared to MLD (in dB) is expected to grow linearly with \(n\). Our simulations show that there is a loss of \(\approx 0.25\) dB for \(n = 16, \approx 0.5\) dB for \(n = 32, \approx 1.25\) dB for \(n = 64\) (compare \(\approx 1\) and \(\approx 20\) on Figure 1) and \(\approx 2.25\) dB for \(n = 128\). As a result, this BDD is not suited for effective decoding of \(BW\) lattices on the Gaussian channel. However, it is essential for building efficient decoders as shown in the next section.
IV. LIST-DECODING OF BW LATTICES BEYOND THE PACKING RADIUS

Let $L(\Lambda, r^2)$ be the maximum number of lattice points of $\Lambda$ within a sphere of radius $r$ around any $y \in \mathbb{R}^n$. If $\Lambda = BW_n$ we write $L(n, r^2)$. The following lemma is proved in [14].

Lemma 1. The list size of the $BW_n$ lattices is bounded as [14]:

- $L(n, r^2) \leq \frac{1}{4^{n/2}}$ if $r^2 \leq d(BW_n)(1/2 - \epsilon)$, $0 < \epsilon \leq 1/4$.
- $L(n, r^2) = 2n$ if $r^2 = d(BW_n)/2$.
- $L(n, r^2) \leq 2n^{16\log_2(1/\epsilon)}$ if $r^2 \leq d(BW_n)(1 - \epsilon)$, $0 < \epsilon \leq 1/2$.

[14] also shows that the parallel BDD of [19], which uses the automorphism group of $BW_n$, can be slightly modified to output a list of all lattice points lying at a squared distance $r^2 = d(BW_n)(1 - \epsilon), \forall \epsilon > 0$, from any $y \in \mathbb{R}^n$ in time $O(n^2 \cdot L(n, r^2)^2)$. With Lemma 1, this becomes $O(n^{16\log(1/\epsilon)})$. This result is of theoretical interest: it shows that there exists a polynomial time algorithm in the dimension for any radius bounded away from the minimum distance. However, due to the quadratic dependence in the list size, the complexity of this list decoder rapidly becomes intractable: for $\epsilon = 1/2$, we get a complexity of $O(n^4)$ and for $\epsilon = 3/8$ it is $O(n^{48})$. Finding an algorithm with quasi-linear dependence in the list-size is stated as an open problem in [14].

In the following, we show that if we use the squaring construction rather than the automorphism group of $BW_n$ for list-decoding we get a quasi-linear complexity in the list size. This enables to get a practical list-decoding algorithm up to $n = 128$.

1) Some notations: Notice that $L(n, r^2) = L(RBW_n, 2r^2)$, e.g. both are equal to $2n$ if $r^2 = d(BW_n)/2$.

It is therefore convenient to consider the relative squared distance as in [14]: $\delta(x, y) = \frac{d(x,y)}{d(\Lambda)}$, $x \in \Lambda^2$. Then, if we define $l(\Lambda, r^2/d(\Lambda)) = L(\Lambda, r^2)$ this yields for instance $l(n, 1/2) = l(BW_n, 1/2) = l(RBW_n, 1/2) = 2n$. The relative squared radius is defined as the quantity $r^2/d(\Lambda)$.

For the rest of this section, $\delta$ is the relative squared radius considered for decoding. Let $y = (y_1, y_2) \in \mathbb{R}^{2n}$ and $x = (u_1, u_1 + v_2) = (u_2 + v_1, u_2) \in BW_{2n}$ be any lattice point where $\delta(x, y) \leq \delta$. We recall that for BDD of $BW_n$ we have $\delta = 1/4$.

The following lemma is trivial, but convenient to manipulate distances.

Lemma 2. (Lemma 2.1 in [14])

Let $y = (y_1, y_2) \in \mathbb{R}^{2n}$ and $x = (u_1, u_1 + v_2) \in BW_{2n}$.

Then, $\delta(x, y) = \delta(u_1, y_1)/2 + \delta(v_2, y_2 - u_1)$.

2) List-decoding with $r^2 < 3/4d(BW_n)$: Assume that the squared norm of the noise is $r^2$ and $\delta = (r^2 + \epsilon)/d(BW_n)$. Consider $d(x, y) = d(u_1, y_1) + d(v_2, y_2)$. We split the possible situations into two main cases (similarly to Steps 2-3 of Algorithm 1): $d(u_1, y_1) \leq r^2/2$ and $d(u_1, y_1) > r^2/2$. For the first case, $y_1$ should be list-decoded in $BW_{n/2}$, and for each $u_1 \in BW_{n/2}$ in the resulting list, $y_2 - u_1$ should be list-decoded in $RBW_{n/2}$. Regarding the noise repartition, one can get the following two extreme configurations (but not simultaneously): $d(u_1, y_1) = r^2/2$, i.e. $\delta(u_1, y_1) = \delta$ and $d(v_2, y_2 - u_1) = r^2$, i.e. $\delta(v_2, y_2 - u_1) = \delta$. Consequently, without any advanced strategy, the relative squared decoding radius to list-decode in $BW_{n/2}$ and $RBW_{n/2}$ should be $\delta$.

The maximum of the product of the two resulting list-sizes, which is a key element in the complexity analysis below, is $l(n/2, \delta)^2$. In order to reduce this number, we split this first case (i.e. $d(u_1, y_1) \leq r^2/2$) into two sub-cases. Let $0 \leq \alpha^* \leq r^2/2$.

- $0 \leq d(u_1, y_1) < \alpha^*$ and $r^2/2 < d(v_2, y_2 - u_1) \leq r^2$: then, $y_1$ should be list-decoded in $BW_{n/2}$ with a relative squared radius $\alpha_1 = \alpha^*/d(BW_{n/2})$ and $y_2 - u_1$ list decoded in $RBW_{n/2}$ with a relative squared radius $\delta$.
- $\alpha^* \leq d(u_1, y_1) \leq r^2/2$ and $r^2/2 < d(v_2, y_2 - u_1) \leq r^2 - \alpha^*$: then, $y_1$ should be list-decoded in $BW_{n/2}$ with a relative squared radius $\delta$ and $y_2 - u_1$ list-decoded in $RBW_{n/2}$ with a relative squared radius $\alpha_2 = (r^2 - \alpha^*)/d(RBW_{n/2})$.

The size of the two resulting lists are bounded by $l(n/2, \alpha_1) \cdot l(RBW_{n/2}, \delta)$ and $l(n/2, \delta) \cdot l(RBW_{n/2}, \alpha_2)$. Consequently, if we choose $\alpha_1 = \alpha_2 = \alpha$, i.e. $\alpha = 2/3\delta$, the two bounds are equal. The maximum number of candidates to consider becomes $2l(n/2, \delta) \cdot l(n/2, \alpha)$ which is likely to be much smaller than $l(n/2, \delta)^2$, the bound obtained without the splitting strategy. The second case (i.e. $d(u_1, y_1) > r^2/2$) is identical by symmetry.

This analysis yields Algorithm 3 listed below. The “removing step” (10 in bold) is added to ensure that a list with no more than $l(n, \delta)$ elements is returned by each recursive call. The maximum number of points to process by this removing step is $4l(n/2, \delta)l(n/2, a)$. Regarding Step 11, using the classical Merge Sort algorithm, it can be done in $O(n \cdot l(n, \delta) \log(l(n, \delta)))$ operations (see App. VII-B in [29]).

Theorem 3. Let $f(\delta) = \log_2(1 - \frac{4}{3}\delta)$. Given any point $y \in \mathbb{R}^n$ and $1/4 \leq \delta \leq 3/4$, Algorithm 3 outputs the list of all lattice points in $BW_n$ lying within a sphere of relative squared radius $\delta$ around $y$ in time:

- $O(n^2 \cdot \log(n))$ if $1/4 \leq \delta \leq 3/8$,
- $O(n^2 \cdot \log^2(n))$ if $3/8 < \delta \leq 1/2$,
- $O(n^{2+\delta/3} \log^2(n))$ if $1/2 < \delta \leq 3/4$.

Note that if $\delta < 1/4$, then one should simply use Algorithm 2 of complexity $O(n^2)$.

Proof. Let $C(n, \delta)$ be the complexity of Algorithm 3. We have

$$C(n, \delta) \leq 4C(n/2, \delta) + 4C(n/2, a)$$

for all recursive calls with $\delta$. The complexity is dominated by

$4l(n/2, \delta)l(n/2, a)O(n) + O(n \cdot l(n, \delta) \log(l(n, \delta)))$.

Hence, the complexity is $O(n^2)$. If $\delta \leq 3/8$, then $l(n, \delta) \leq 2$, $l(n, a) \leq 1$, $4C(n/2, a) \leq 4C(n/2, 1/4) = O(n/2)^2$ (the complexity...
of Algorithm 2). Hence, the complexity becomes
\( \mathcal{C}(n, \delta) \leq 4C(n, 3/8) + O(n^2) = O(n^2 \log(n)) \).

If \( \delta \leq 1/2 \), then \( \ell(n, \delta) \leq 2n \), \( \ell(n, a) \leq 2 \),
\( 4\mathcal{C}(n/2, 3/8) = O(n^2 \log(n)) \). Hence, the complexity becomes
\( \mathcal{C}(n, \delta) \leq 4\mathcal{C}(n/2, 1/2) + O(n^2 \log(n)) = O(n^2 \log^2(n)) \).

For the case \( 1/2 < \delta < 3/4 \), we first compute
\( 4\ell(n/2, a)\ell(n/2, \delta) \), the maximum number of points to be processed at each recursive step of the algorithm (the removing step 10).

\[
4\ell(n/2, a)\ell(n/2, \delta) \leq \left( \frac{2}{1 - \frac{3}{2} \delta} \right) \cdot 4 = n^{1-\log_2(1 - \frac{3}{2} \delta)} \cdot 4 = O(n^{1-\log_2(1 - \frac{3}{2} \delta)}) \text{.}
\]

where we used recursively the inequality
\( \ell(n, \delta) \leq 4\ell(n/2, a)\ell(n/2, \delta) \). Let us define
\( f(\delta) = -\log_2(1 - \frac{3}{8} \delta) \). Then, we have
\( 4\mathcal{C}(n/2, a) \leq 4\mathcal{C}(n/2, 1/2) = O(n^2 \log^2(n)) \). Hence, the complexity becomes
\( \mathcal{C}(n, \delta) \leq 4\mathcal{C}(n/2, 1/2) + O(n^2 f(\delta) \log(n)) = O(n^2 \log^2(n)) \).

\[\square\]

Algorithm 3 First recursive list-decoding of \( BW_{2n} \) (\( 2n = 2^t \)).

Function \( \text{ListRecBW}(y, t, \delta) \)

Input: \( y = (y_1, y_2) \in \mathbb{R}^{2^t} \), \( 1 \leq t, 1/4 \leq \delta < 3/4 \).

1. \( a \leftarrow 2/3 \cdot \delta \).
2. \( r \leftarrow \sqrt{2} \cdot 1 - \delta \).
3. if \( t = 1 \) then
4. \( \hat{x} \leftarrow \text{Enum}_z(y, r) \) // Enum. in \( \mathbb{Z}^2 \) with radius \( r = \sqrt{3} \).
5. else
6. \( \hat{x}_1 \leftarrow \text{SubRoutine}((y_1, y_2), t, a, \delta, 0) \).
7. \( \hat{x}_2 \leftarrow \text{SubRoutine}((y_1, y_2), t, \delta, a, 0) \).
8. \( \hat{x}_3 \leftarrow \text{SubRoutine}((y_2, y_1), t, \delta, a, 1) \).
9. \( \hat{x}_4 \leftarrow \text{SubRoutine}((y_2, y_1), t, a, \delta, 1) \).
10. Remove all candidates at a distance > \( r \) from \( y \).
11. Sort the remaining list of candidates in a lexicographic order and remove all duplicates.
12. end if
13. Return the list of all the candidates remaining.

The above proof highlights that the term \( \ell(n, a) \) is important
in the complexity analysis. We investigate only the case \( \delta < 3/4 \) since it yields \( a < 1/2 \), which, by Lemma 1, is the only regime
where \( \ell(n, a) \) does not depend on \( n \).

Unfortunately, the performance of Algorithm 3 on the Gaussian channel is disappointing. This is not surprising: notice that
the “removing step” (10 in bold), some points that are
correctly decoded by Algorithm 2 (the BDD) are not in the
list outputted by Algorithm 3! Therefore, instead of removing
candidates at a distance greater than \( r \), it is tempting to
keep \( \mathcal{N} \) candidates at each step.

A. An efficient list decoder on the Gaussian channel

We call Algorithm 5 a modified version of Algorithm 3
where the \( \mathcal{N}(\delta) \) closest candidates are kept at each recursive
step (instead of step 10) and steps 10 and 11 are flipped. The
size of the list \( \mathcal{N}(\delta) \), for a given \( \delta \), is a parameter to be fine
tuned: e.g. for \( \delta = 1/2 \), one needs to chose \( \mathcal{N}(1/2) \) and \( \mathcal{N}(2/3)
- 1/2 = 1/3 \).

The following theorem follows from Theorem 3.

Theorem 4. The complexity of Algorithm 5 is:

- \( O \left( \max \left\{ n^2 \mathcal{N}(\delta) \log(\mathcal{N}(\delta)), n^2 \log(n) \right\} \right) \) with \( \delta \leq 3/8 \).
- \( O \left( \max \left\{ n^2 \mathcal{N}(\delta) \log(\mathcal{N}(\delta)), n^2 \log^2(n) \right\} \right) \) with \( 3/8 < \delta \leq 1/2 \)

\( (\mathcal{N} = \mathcal{N}(2/3 \cdot \delta) \cdot \mathcal{N}(\delta)) \).

V. NUMERICAL RESULTS

A. Performance of Algorithm 5

Figure 1 shows the influence of the list size when decoding
\( BW_{64} \) using Algorithm 5 with \( \delta = 3/8 \).

Figure 2 depicts the performance of Algorithm 5 for the
\( BW \) lattices up to \( n = 128 \) and the universal bounds provided
in [27] (see also [13] or [16], where it is called the sphere
lower bound). These universal bounds are limits on the highest
possible coding gain using any lattice in \( n \) dimensions. For
each \( BW_n \) we tried to reduce as much as possible the list size
while keeping quasi-MLD performance. The choice of
\( \delta = 3/8 \) yields quasi-MLD performance up to \( n = 64 \) with
small list size and thus reasonable complexity. This shows that
\( BW_{64} \), with Algorithm 5, is a good candidate to design finite
costellations in dimension 64. However, for \( n = 128 \), one
needs to set \( \delta = 1/2 \) and choose \( \mathcal{N}(\delta) = 1000 \). Nevertheless,
\( \mathcal{N}(2/3 \cdot \delta) \) can be as small as 4, which is still tractable.
We compare these performances with existing schemes at $P_e = 10^{-5}$. For fair comparison between the dimensions, we let $P_e$ be either the normalized error probability, which is equal to the point error-rate divided by the dimension (as done in e.g. [27]), or the symbol error-rate.

First, several constructions have been proposed for block-lengths around $n = 100$ in the literature. In [18] a two-level construction based on BCH codes with $n = 128$ achieves this error-rate at 2.4 dB. The decoding involves an OSD of order 4 with 1505883 candidates. In [1] the multilevel (non-lattice packing) $S_{127}$ ($n = 127$) has similar performance but with much lower decoding complexity via generalized minimum distance decoding. In [23] a turbo lattice with $n = 102$ and in [25] a LDLC with $n = 100$ achieve the error-rate with iterative methods at respectively 2.75 dB and 3.7 dB (unsurprisingly, these two schemes are efficient for larger block-lengths). All these schemes are outperformed by $BW_{64}$, where $P_e = 10^{-5}$ is reached at 2.3 dB. Moreover, $BW_{128}$ has $P_e = 10^{-5}$ at 1.7 dB, which is similar to many schemes with block-length $n = 1000$ such as the LDLC (1.5 dB) [25], the turbo lattice (1.2 dB) [23], the polar lattice with $n = 1024$ (1.45 dB) [28], and the LDA lattice (1.27 dB) [6]. This benchmark is summarized in Figure 3.

B. Performance of $BW$ finite constellations

We uncover the performance of a Voronoi constellation [4][10] based on the partition $BW_{64}/2^n BW_{64}$ via Monte Carlo simulation, where $\eta$ is the desired rate in bits per channel use (bpcu); i.e. both the coding lattice and the shaping lattice are based on $BW_{64}$. It follows that the encoding complexity is the same as the decoding complexity; the complexity of Algorithm 5 with $\delta = 3/8$ and $N(\delta) = 20$, is used for encoding and decoding. The cutoff-rate limit is $1.7+0.179$ dB right to Shannon limit (coding + shaping loss for $n = 64$[12]). The scheme performs within 0.7 dB of the bound.

VI. CONCLUSIONS

Our recursive paradigm can be seen as a tree search algorithm and our decoders fall therefore in the class of sequential decoders. While the complexity of Algorithm 5 remains stable and low for $n \leq 64$, there is a significant increase for $n = 128$ and it becomes intractable for $n = 256$ due to larger lists. This is not surprising from the cut-off rate perspective [12]; For $n = 64$ the MLD is still at a distance of 1 dB from this limit (Figure 4), but it is very close to the limit for $n = 128$ and potentially better at larger $n$. One should not expect to perform quasi-MLD of these lattices with any sequential decoder. This raises the following open problem: can we decode lattices beyond the cut-off rate in non-asymptotic dimensions, i.e. $n < 300$, where classical capacity-approaching decoding techniques (e.g. BP) cannot be used?
REFERENCES

[1] D. Agrawal and A. Vardy, “Generalized minimum distance decoding in euclidean space: Performance analysis,” IEEE Trans. Inform. Theory, vol. 46, pp. 60–83, 2000.
[2] E. S. Barnes and G. E. Wall, “Some extreme forms defined in terms of Abelian groups,” J. Australian Math. Soc., vol. 1, pp. 47-63, 1959.
[3] F. Bollaaf, R. Zamir, S. I. R. Costa, “Multilevel Constructions: Coding, Packing and Geometric Uniformity,” IEEE Trans. on Inform. Theory, Vol. 65, 2019.
[4] J. Conway and N. Sloane, “A fast encoding method for lattice codes and quantizers,” IEEE Trans. Inform. Theory, vol. 19, pp. 820-824, 1983.
[5] J. Conway and N. J. A. Sloane. Sphere packings, lattices and groups. Springer-Verlag, New York, 3rd edition, 1999.
[6] N. di Pietro, J. J. Boutros, G. Zémor, L. Brunel, “Integer Low-Density Lattices based on Construction A”, 2012 IEEE Information Theory Workshop, 2012.
[7] I. Dumer and K. Shabunov, “Soft-decision decoding of Reed-Muller codes: recursive lists,” IEEE Trans. Inform. Theory, vol. 52, pp. 1260-1266, 2006.
[8] N. D. Elkies, “Mordell-Weil lattices in characteristic 2: III. A Mordell-Weil lattice of rank 128,” Experimental Math., vol. 3, pp. 467-473, 2001.
[9] G. D. Forney, Jr., “Coset codes II: Binary lattices and related codes,” IEEE Trans. Inform. Theory, vol. 34, pp. 1152-1187, 1988.
[10] G. D. Forney, Jr., “Multidimensional Constellations - part II: Voronoi Constellations,” IEEE J. Select. Areas Com., vol. 7, pp. 941-958, 1989.
[11] G. D. Forney, Jr. and A. Vardy, “Generalized Minimum-Distance decoding of Euclidean-Space Codes and Lattices,” IEEE J. Select. Areas Com., vol. 42, pp. 1992-2026, 1996.
[12] G. D. Forney and G. Ungerboeck, “Modulation and Coding for Linear Gaussian Channels,” IEEE Trans. Inform. Theory, vol. 44, pp. 2384-2415, 1998.
[13] G. D. Forney, M. D. Trott, and S. Chung, “Sphere-Bound-Achieving Coset Codes and Multilevel Coset Codes,” IEEE Trans. Inform. Theory, vol. 46, pp. 820-850, 2000.
[14] E. Grigorescu and C. Peikert, “List-Decoding Barnes-Wall Lattices,” Computational Complexity, vol. 26, pp 365-39, 2017.
[15] J. Harshan, E. Viterbo, and J.-C. . Belfiore, “Practical Encoders and Decoders for Euclidean Codes from Barnes-Wall Lattices,” IEEE Trans. Communications, vol. 61, pp. 4417-4427, 2013.
[16] A. Ingber, R. Zamir, and M. Feder, “Finite-Dimensional Infinite Constellations,” IEEE Trans. Inform. Theory, vol. 59, pp. 1630-1656, 2013.
[17] F. J. MacWilliams and N. J. A. Sloane, The Theory of Error-Correcting Codes. Amsterdam, The Netherlands: North-Holland, 1977.
[18] T. Matsumine, B. M. Kurkoski, and H. Ochiai, “Construction D Lattice Decoding and Its Application to BCH Code Lattices,” 2018 IEEE Global Communications Conference (GLOBECOM), 2018.
[19] D. Micciancio and A. Nicolosi, “Efficient Bounded Distance Decoders for Barnes-Wall Lattices,” 2008 IEEE Int. Symp. Inform. Theory, 2008.
[20] G. Nebe, “An even unimodular 72-dimensional lattice of minimum 8,” J. Reine Angew. Math., vol. 673, pp. 237-247, 2012.
[21] M. Plotkin, “Binary codes with specified minimum distances,” IEEE Trans. Inform. Theory, vol. 6, pp. 445-450, 1960.
[22] G. Poltyrev, “On coding without restrictions for the AWGN channel,” IEEE Trans. Inform. Theory, vol. 40, pp. 409-417, 1994.
[23] A. Sakzad, M. Sadeghi, and D. Panario, “Turbo Lattices: Construction and Performance Analysis,” arXiv preprint arXiv:1108.1873, 2011. 48th Annual Allerton Conf. on Com., Control, and Computing, 2011.
[24] A. J. Salomon and O. Amrani, “Encoding and Decoding Binary Product Lattices,” IEEE Trans. Inform. Theory, vol. 52, pp 5485-5495, 2006.
[25] N. Sommer, M. Feder, and O. Shalvi, “Low-density Lattice Codes,” IEEE Trans. Inform. Theory, vol. 54, pp. 1561-1585, 2008.
[26] G. Schnabl and M. Bossert, “Soft-decision decoding of Reed-Muller codes as generalized multiple concatenated codes,” IEEE Trans. on Inform. Theory, vol. 41, pp. 304-308, 1995.
[27] V. Tarokh, A. Vardy, and K. Zeger, “Universal bound on the performance of lattice codes,” IEEE Trans. Inform. Theory, vol. 45, pp. 670-681, 1999.
[28] Y. Yan, C. Ling, X. Wu, “Polar lattices: Where Arıkan meets Forney,” 2013 IEEE Int. Symp. Inform. Theory, 20013.
[29] The long version of this paper including an appendix is available at: arXiv preprint arXiv:2001.05907.