This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2017.2761262, IEEE Transactions on Vehicular Technology

Reduced Latency ML Polar Decoding via Multiple Sphere-Decoding Tree Searches

Chistopher Husmann, Panagiotis Chatzi Nikolaou and Konstantinos Nikitopoulos

***Abstract*—Sphere decoding (SD) has been proposed as an efficient way to perform maximum-likelihood (ML) decoding of Polar codes. Its latency requirements, however, are determined by its ability to promptly exclude from the ML search (i.e., prune) large parts of the corresponding SD tree, without compromising the ML optimality. Traditional depth-first approaches initially find a "promising" candidate solution and then prune parts of the tree that cannot result to a "better" solution. Still, if this candidate solution is far (in terms of Euclidean distance) from the ML one, pruning becomes inefficient and decoding latency explodes. To reduce this processing latency, an early termination approach is, first, introduced that exploits the binary nature of the transmitted information. Then, a simple but very efficient SD approach is proposed that performs multiple tree searches that perform decreasingly aggressive pruning. These searches are almost independent and can take place sequentially, in parallel, or even in a hybrid (sequential/parallel) manner. For Polar codes of 128 block size, both realizations can provide a latency reduction of up to four orders of magnitude compared to state-of-the-art Polar sphere decoders. Then, a further 50% latency reduction can be achieved by exploiting the parallel nature of the approach.**

***Index Terms*—Polar codes, Sphere decoding, ML detection.**

I. Introduction

Polar codes have been proved to achieve the capacity of the binary-input, symmetric (both discrete or continuous), memoryless channels, in the limit of infinite block length [1], [2]. For their decoding, a low-complexity *successive cancellation* (SC) method has been initially proposed [1], [3] which, however, can result in a significant error-rate degradation compared to the optimal but highly complex maximum-likelihood (ML) decoder. To reduce this performance loss, several improved but still suboptimal decoding approaches have been proposed [4], [5], [6], [7], all searching for the ML decoding solution in a set of promising cadidates. Still, while all these methods can provide increased decoding performance compared to the originally proposed SC one, they cannot guarantee ML optimality.

This work focuses on optimal detectors able to always guarantee the ML performance and, therefore, able to unlock the full potential of polar codes. An ML decoder implementation, performing exhaustive search, has a complexity of $O(2^K)$, with K being the number of information bits per codeword. Traditional, depth-first *sphere-decoding* (SD) approaches [8], [9], avoid such an exhaustive search by translating the ML-problem into an equivalent tree search problem. Then, the exact complexity requirements of different SDs are random and depend on their ability to early exclude from the search large parts of the SD tree without compromising the ML optimality (i.e., their pruning approach), as well as on transmission parameters like the signal-to-noise ratio (SNR). Still, while in most practical cases an SD can substantially decrease the average complexity of ML decoding, its worst-case complexity still remains $O(2^K)$.

In the context of exact ML Polar decoding, a simplistic depth-first SD has been been first applied by Kahraman and Çelebi [10]. In [11] Guo and Fabregas applied a depth-first approach with radius update, similarly to the practical solutions proposed in [10], but it further exploits the fact that the decoding process takes place in a Galois Field of two elements GF(2) to perform a more aggressive SD tree pruning that substantially reduces the traditional SD processing latency while preserving the ML optimality. An alternative Polar SD tree traversal strategy has been proposed in [12]. However, while this traversal can result in a reduced number of visited nodes compared to the traditional, depth-first Polar SDs, it involves highly increased memory storage as well as multiple sorting operations and thus increased hardware logic. All these make the practical hardware implementations of the approach very challenging. To reduce the processing latency, the authors in [13] transform the binary tree into a non-binary one with its corresponding height being shorter than the height of the original SD tree. In order to traverse the non-binary tree, however, their detection ordering requires a very large amount of Euclidean distance calculations and intensive sorting operations, any time a codeword is received.

All the above methods employ a common methodology to find the ML solution. They first find a "good" candidate solution, close (in terms of Euclidean distance) to the received signal. Then, they search for a "better" candidate solution, that is even closer to the received signal, after excluding (i.e., pruning) parts of the SD tree that cannot result in such a solution. This process continues until no "better" solution can be found. While this is a very efficient strategy, when the code length increases and the SNR decreases, the Euclidean distance of the candidate solutions from the received signal can be very large, resulting in inefficient SD tree pruning and, consequently, in highly increased processing complexity and latency. To reduce this dependence of the processing requirements to the candidate solutions, in this paper we first propose an *early search termination criterion* that can identify cases where the ML solution is found, and terminate the tree search without compromising the ML performance. On the top of the early search termination approach, we propose a simple, but very efficient SD approach consisting of *multiple SD tree searches* that can be performed sequentially, in parallel, or in a hybrid sequential/parallel manner. As we discuss in detail in Section III, the proposed method approaches the "optimal" complexity and latency requirements of an "ideal" (but non-existing) SD that knows in advance the Euclidean distance of the actual ML solution to the received signal, and uses it for pruning purposes.

In Section IV we show that even when the multiple tree searches take place sequentially, and despite the fact that some tree nodes are visited multiple times across the multiple searches, our proposed strategy can decrease the decoding

All authors are with the 5G Innovation Centre, Institute for Communication Systems, University of Surrey, Guildford, Surrey, GU2 7XH, UK. e-mail: (c.husmann, pc00325, k.nikitopoulos) @surrey.ac.uk.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2017.2761262, IEEE Transactions on Vehicular Technology

processing requirements by up to four orders of magnitude compared to state-of-the-art, ML optimal, Polar decoders. Further latency reduction can be achieved by exploiting the nearly independent nature of the tree searches, in order to apply a limited level of parallelism. Such a parallel approach is complementary to recently proposed techniques targeting the massive parallelization of depth first SDs [14]. We note that our proposed technique is generally applicable to any Polar SD-based decoding approach that finds "candidate" solutions and performs pruning based on their distance from the received signal. Still, in this work we focus on the Polar SD-based detector of [11] due to its structural simplicity.

II. Depth-First Sphere Decoding for Polar codes

Polar codes are determined by a recursive operation called *channel polarization* that asymptotically transforms N independent channel-uses into N polarized channels, with some of them being *highly reliable* and the rest of them being *less reliable* [1]. Then, a (N, K) Polar code of rate $R = K/N$, is constructed by transmitting the K information bits through the K most reliable channels, while transmitting known (typically zero) bits, called *frozen bits*, over the remaining $N - K$ channels. With $\mathbf{F} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$, the $N \times N$ matrix $\mathbf{F}^{\otimes n}$ is the n-fold Kronecker power of $\mathbf{F}$ with $n = \log_2 N$ and $\mathbf{F}^{\otimes n} = \mathbf{F} \otimes \mathbf{F}^{\otimes (n-1)}$. Then, the generator matrix of a (N,K) polar code is $\mathbf{G}_N = \mathbf{F}^{\otimes n}$. Thus, given an information message of K information bits, a codeword $\mathbf{x}$ is obtained as

$$\mathbf{x} = \mathbf{u} \cdot \begin{bmatrix} \mathbf{F}^{\otimes (n-1)} & \mathbf{0} \\ \mathbf{F}^{\otimes (n-1)} & \mathbf{F}^{\otimes (n-1)} \end{bmatrix}, \tag{1}$$

where $\mathbf{u} \triangleq (u_1, \ldots, u_N)$ is a vector of length N containing the K information bits plus the frozen bits at the appropriate positions. To find these positions, various construction methods have been proposed like the ones on [1], [15], [16].

When the BPSK mapping $\mathbf{s} = \mathbf{1} - 2\mathbf{x}$ is used for wireless transmission, with $\mathbf{1}$ being the all-one vector of length N, the received signal is $\mathbf{y} = \mathbf{s} + \mathbf{k}$, with the vector $\mathbf{k}$ consisting of independent and identically distributed (i.i.d.) additive white Gaussian noise samples of zero mean and σ^2 variance. Then, the ML Polar decoder can be formulated as finding the codeword [10]

$$\begin{aligned} \mathbf{u}_{\text{ML}} &= \underset{\mathbf{u} \in \{0,1\}^N}{\arg\min} \; ||\tilde{\mathbf{y}} - \mathbf{u}\mathbf{F}^{\otimes n}||^2 \\ &= \underset{\mathbf{u} \in \{0,1\}^N}{\arg\min} \; \sum_{\ell=1}^{N} \left(\tilde{y}(\ell) - \sum_{j=\ell}^{N} u(j) \odot f_{j,\ell} \right)^2 \end{aligned} \tag{2}$$

with $\tilde{\mathbf{y}} \triangleq \frac{\mathbf{1}-\mathbf{y}}{2}$, $\tilde{y}(i)$ being the ith element of the vector $\tilde{\mathbf{y}}$, $f_{j,i}$ being the element of the matrix $\mathbf{F}^{\otimes n}$ in the jth row and ith column, and $\odot$ denoting the multiplication in GF(2). The above minimization problem, can be transformed into a tree search, with the corresponding tree height being equal to the block length N, and with a branching factor of two. Each node at the lth level of the tree is then associated to a partial vector $\mathbf{u}^{(\ell)} \triangleq [u_N, u_{N-1}, \ldots, u_\ell]$, that represents the bit decisions form the top level of the tree down to the lth level. In addition, the tree node related to $\mathbf{u}^{(\ell)}$ is characterized by the partial Euclidean distance, that can be computed recursively as

$$d(\mathbf{u}^{(\ell)}) = d(\mathbf{u}^{(\ell+1)}) + \left(\tilde{y}(\ell) - \sum_{j=\ell}^{N} u(j) \odot f_{j,\ell} \right)^2, \tag{3}$$

with $d(\mathbf{u}^{N+1}) = 0$. Therefore, the ML problem is translated into finding the leaf node with the minimum partial Euclidean distance $d(\mathbf{u}^{(1)})$.

To find the ML solution, traditional SDs like [9] follow a depth-fist approach with sorted enumeration and radius update. In particular, when a parent node is expanded, the node (i.e., the bit) with the smallest partial Euclidean distance is visited first. Whenever a new candidate solution (i.e., a leaf node) is reached, the squared radius r^2 of the SD, that has been initially set to infinity, is updated with the Euclidean distance of this new candidate. In the later steps of the tree search, when a node is visited with its $d(\mathbf{u}^{(\ell)}) > r^2$, this node, its siblings and their children nodes are all excluded from the search. The search continues until no solution (i.e., leaf node) with smaller Euclidean distance can be found.

When the Polar code length increases, the Euclidean distance of a candidate solution may be of much larger value than the partial distances of the nodes at higher layers of the SD tree. As a result, traditional approaches are not able to early prune (at higher tree layers) the tree paths that cannot result in a candidate update. To cope with this problem, a more aggressive tree pruning approach has been proposed in [11]. In particular, the Authors in [11] observed that the term $u(j) \odot f_{j,\ell}$ in (3), as well as any sum of such terms, can only take the values zero or one. Therefore, for any ℓ and by defining $d_{min}(\ell) = \min\{(\tilde{y}(\ell) - 0)^2, (\tilde{y}(\ell) - 1)^2\}$, the residual partial distance from a node being at level ℓ down to the leaf node (of $\ell = 1$) can be lower-bounded by the value

$$D_{min}(\ell) = \sum_{i=1}^{\ell-1} d_{min}(i). \tag{4}$$

Then, the pruning condition can be modified so that pruning takes place anytime a node is visited with its

$$d(\mathbf{u}^{(\ell)}) + D_{min}(\ell) > r^2, \tag{5}$$

without compromising the ML optimality. While this is a very efficient strategy, when the code length increases and the SNR decreases, the r^2 of the candidate solution can still be very large, preventing from efficient SD tree pruning and, therefore, resulting in very high processing requirements. As discussed, this work targets in reducing this dependency.

III. Multiple Sphere-Decoding Tree Searches

Here, we first describe the *early search termination criterion* and then the details of our *multiple SD tree searches*.

A. Early search termination criterion

The proposed termination criterion is based on the D_{min} lower bound introduced in Section II. In particular, we can easily observe that, for any given received signal, the Euclidean distance of any leaf node will always be equal to or larger

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2017.2761262, IEEE Transactions on Vehicular Technology

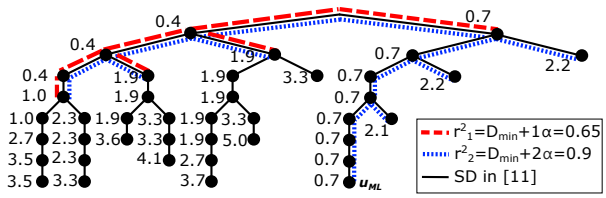

Fig. 1. Example of a sequential decoding attempt for (8,4) Polar code for the proposed search (dashed line: 1st search; dotted line: 2nd search) and the SD in [11] (solid line).

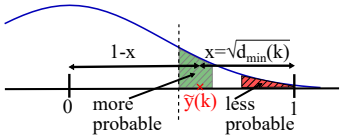

Fig. 2. Position of $\tilde{y}(k)$ and its distance to the bits.

than $D_{min}(N+1)$. Therefore, if a solution is found with its $d(\mathbf{u}^{(1)}) = D_{min}(N+1)$ it is the ML one, and the search can be safely terminated. This very simple termination criterion, that comes without any practical complexity overhead (in terms of calculations) since the d_{min} values of D_{min} are anyhow calculated for applying the pruning criterion of (5). Despite its simplicity, as we show in table I, the pruning criterion can be very efficient, especially in the high SNR regime.

B. ML Polar Decoding via Multiple SD tree searches

Ideally, to minimize the complexity and latency of the sphere decoding search, the initial squared radius should ideally be equal to $r^2_{ideal} = d(\mathbf{u}^{(1)}_{ML})$. Smaller r^2 values would result in pruning all leaf nodes, while larger r^2 values would result in less aggressive pruning and, therefore, in unnecessarily increased number of visited nodes. Still, knowing the value of $d(\mathbf{u}^{(1)}_{ML})$ in advance is practically infeasible. As we show in Section IV, however, the proposed approach visits a number of nodes similar to the one when $d(\mathbf{u}^{(1)}_{ML})$ is known. To achieve this, we perform several tree searches with different initial r^2 values. In particular, in the sequential version of the proposed approach (the parallel follows later), and after calculating $D_{min}(N+1)$ similarly to [11], the wth tree search, will employ a squared radius of

$$r^2_w = D_{min}(N+1) + w \cdot \alpha, \tag{6}$$

starting with $w = 1$, and with α being a step-size parameter. If the wth tree search is terminated before visiting any candidate solution (i.e., any leaf node), it means that r^2_w has been set to a value smaller than $d(\mathbf{u}^{(1)}_{ML})$. Thus, the detection process continues with the $(w+1)$th tree search. If the kth tree search results in a valid solution, it means that $d(\mathbf{u}^{(1)}_{ML}) \leq r^2_k$, and the search process can be safely terminated without compromising the ML optimality. This is since, a further increasing r^2 would just unnecessarily increase the search space by just adding candidate solutions with larger Euclidean distances than the one already found.

In Fig. 1 we show a simplified example, where an (8,4) Polar code is decoded.We show the visited nodes and their corresponding $d(\mathbf{u}^{(\ell)}) + D_{min}(\ell)$ metrics, as well as the corresponding search paths, for different tree searches. We have assumed that an $D_{min}(N+1)$ of 0.4 has been first calculated, and that $d(\mathbf{u}^{(1)}_{ML})$=0.7. We have also assumed that $\alpha = 0.25$. Thus, for the first tree search r^2_1 is set to 0.65. Since $r^2_1 < d(\mathbf{u}^{(1)}_{ML})$ the search will be terminated unsuccessfully, resulting in no solution, after visiting 7 nodes. The corresponding search path is shown in red dashed lines. Since the tree search was unsuccessful, a second tree search takes place with $r^2_2 = 0.9$, the search path of which is shown in blue dotted lines. As can easily be seen, the search space of the first tree search is a subset of the second one. Since $r^2_2 > d(\mathbf{u}^{(1)}_{ML})$, the leaf node with the ML solution is now visited, and the other candidate solutions are pruned, after visiting 17 nodes. The total number of visited nodes that is required to find the ML solution comprises of visiting 8 nodes to determine $D_{min}(N+1)$, 7 nodes for the first (w=1) unsuccessful search and 17 nodes for the second (w=2) successful search. Hence, the proposed approach visits in total 32 nodes. In contrast, [11] requires 40 nodes to find the ML solution.

For the proposed approach to be efficient, the step-size parameter α should be appropriately set. Large α values can reduce the number unsuccessful tree searches, but they can result in r^2_w values that are much larger than the $r^2_{ideal} = d(\mathbf{u}^{(1)}_{ML})$, imposing increased processing and latency requirements for the corresponding tree search. On the other hand, small α values enable r^2_w values close to $d(\mathbf{u}^{(1)}_{ML})$, that minimize the number of visited nodes for the tree search with $r^2_w > d(\mathbf{u}^{(1)}_{ML})$, but can result in many unsuccessful tree searches, that can unnecessary increase overall latency and the number of operations. Finding the α value that minimizes the number of visited nodes is a very challenging optimization problem, involving the Polar code structure and the operating SNR, and requiring the modeling of of the the effect of the squared radius selection on the number of visited nodes. In the following, instead of trying to solve this tedious optimization problem, we focus on providing some practical rules for setting the value of α, while also accounting for a partial parallelism. As we show in Section IV Fig. 4 and 6 these practical rules, even if not optimal, can reduce latency and complexity by three to four orders of magnitude. Moreover, latency can be additionally reduced by up to 50% by exploiting parallelism.

C. Choice of the α parameter and the level of parallelization.

If the candidate vector $\tilde{\mathbf{u}}_c$ with $d(\tilde{\mathbf{u}}^{(1)}_c) = D_{min}(N+1)$ is not the ML solution, it means that $\tilde{\mathbf{u}}_c$ is not a valid codeword. In such cases, the target is to find an α value which is small, but large enough so that when we search for the ML solution at a squared radius $D_{min}(N+1)+\alpha$ around the received signal, we will be able to examine additional potential codewords other than $\tilde{\mathbf{u}}^{(1)}_c$. In other words, the squared radius should be such that the SD search will include potential codewords that differ from the $\tilde{\mathbf{u}}_c$ at one level (i.e., the codewords differ by one bit) at least. To find such an α value we are focusing on the most challenging case where a candidate codeword differs from $\tilde{\mathbf{u}}_c$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2017.2761262, IEEE Transactions on Vehicular Technology

at tree level k, and the sum of all the other $d_{min}(\ell)$ values for $\ell \neq k$ is the same (as it holds when k=1). Then, the value of α required to visit the next candidate codeword is determined by the actual position of the received point $\tilde{y}(k)$ as shown in Fig. 2. In particular, if $\tilde{y}(k)$ lies between the two bits (which due to the Gaussian nature of the noise is the most likely case) and its distance to the closest bit (e.g., to 1 in Fig. 2) is $x = \sqrt{d_{min}(k)}$, while its distance to the next candidate (e.g., to 0 in Fig. 2) is $1 - x$. Thus, in order to include in the tree search the second closest bit to $\tilde{y}(k)$ (e.g., the bit 0), it should hold that

$$\begin{aligned} d_{min}(k) + \alpha &\geq (1-x)^2 \\ \alpha &\geq (1-x)^2 - d_{min}(k) \\ \alpha &\geq 1 - 2x \end{aligned} \tag{7}$$

For example if $\tilde{y}(k) = 0.6$, then $x = 0.4$, $d_{min}(k) = x^2 = 0.16$ and $1 - x = 0.6$. Thus, the minimum α required is $\alpha_{min} = 0.2$. If, for this example, we set $\alpha = 0.1$, then $d_{min} + \alpha = 0.26 < 0.6^2$ and, as a result, the new sphere decoder search, despite the additional calculations, will never search the candidate codeword in question. If however, $\alpha \leq 1$, another codeword will always be examined for any noise realization lying between 0 and 1 and, therefore, for most noise realizations. Hence, an $\alpha \geq 1$ will most likely result in evaluating codewords that differ from $\tilde{\mathbf{u}}_c$ in a single bit position, and not in unnecessary searches. Integer multiples of this α value, would allow evaluating codewords that differ from $\tilde{\mathbf{u}}_c$ in multiple bit positions (and with such searches taking place in following searches according to (6)). Since, however, we would like to keep the value of α as small as possible to reduce the possible error from r^2_{ideal}, as discussed in Section III-B, we can set $\alpha = 1$. In practice, however, due to the Gaussian nature of the noise, $\tilde{y}(k)$ is more likely to lie at x around 0.5 than around zero, and therefore α values down to around 0.5 could still provide processing complexity and latency gains when using sequential tree searches, as also verified in Fig. 5. As a result, and as also verified in Section IV practical α values in the range of $[0.5, 1]$ can be used.

Using the same rationale, it can be easily excised that when we use very small α values it is more likely that redundant searches are taking place, resulting in a complexity, and therefore, in a latency increase. To further reduce processing latency (but not the overall complexity) by reducing the α value, parallelism can be efficiently exploited. In particular, a hybrid sequential/parallel multiple search is proposed which can further reduce latency and asymptotically approach the one of the "ideal", SD at the cost of a relatively small complexity overhead. In particular, if N_p parallel SDs are available, and by setting as α_{seq} an α value for which sequential SD searches can be efficiently performed (i.e., in the range of [0.5,1]), at the wth sequential step, $p \in 1, ..., N_p$ SD tree searches take place in parallel with their corresponding initial radius set to

$$r^2_{p,w} = D_{min}(N+1) + \left(p - 1 + \frac{w}{N_p}\right)\alpha_{seq}, \tag{8}$$

which is equivalent to performing multiple tree searches with squared radii that increase by steps of α_{seq}/N_p.

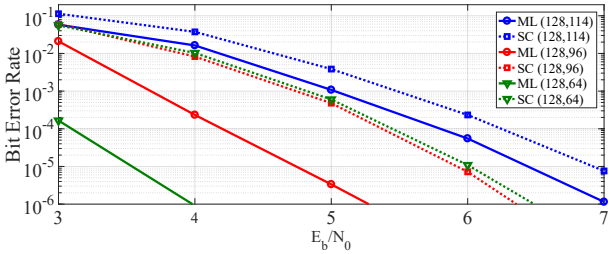

Fig. 3. BER of ML and SC decoding for(128,64), (128,96) and (128,114) Polar coding.

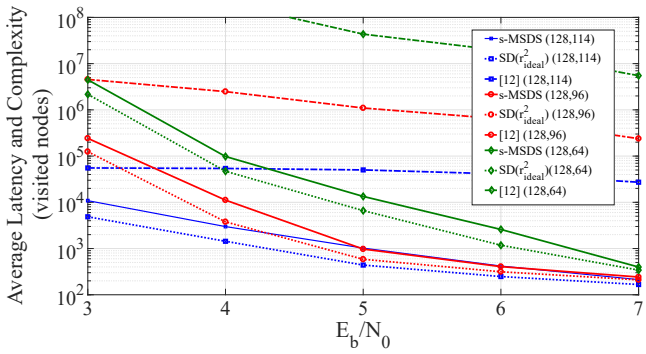

Fig. 4. Latency and complexity in visited nodes, of s-MSDS ($\alpha = 1$, solid), "ideal" SD (dotted) and the SD in [11] (dashed).

IV. Evaluation

Polar codes with a block length of $N = 128$ and a code rate of 0.5, 0.75 and 0.89 (i.e., (128,64), (128,96) and (128,114) Polar codes) have been examined. To find the positions of the frozen bits the method in [1] has been applied, assuming a high SNR (*e.g.*, 10 dB), which for ML decoding has been observed to provide a good achievable error rate over the whole evaluated SNR regime.

Figure 3 validates that ML decoding methods, like the one proposed, can significantly outperform traditional SC decoding in terms of bit error-rate. To bridge this gap, and as shown in Figure 4, the state-of-the-art SD based Polar decoder of [11] would require visiting up to 10^7 nodes ((128,64) Polar Code). In contrast, our proposed *multiple sphere decoder search* (MSDS), when applied sequentially (s-MSDS) with an α of one, can reduce the corresponding latency and complexity (both measured in terms of visited nodes) by up to four orders of magnitude compared to [11], and it can almost reach the latency requirements of an "ideal", but non-existing, SD which could always know in advance, and use a squared radius, the distance $d(\mathbf{u}^{(1)}_{ML})$ of the ML solution.

Figure 5 illustrates the impact of the design parameter α on the latency and complexity of the s-MSDS, for all evaluated code rates, at a 5dB SNR. As explained in Section III, Fig. 5 shows that for α values larger than one, the complexity and latency decreases as α decreases. In practice, by decreasing the α parameter to values close to one, a nearly-linear (in the logarithmic domain), latency reduction can be achieved. This reduction is more prominent for larger block lengths, where the search space is larger. Fig. 5 also shows that α values in the interval [0.5 1] (highlighted green in Fig. 5) result in a

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2017.2761262, IEEE Transactions on Vehicular Technology

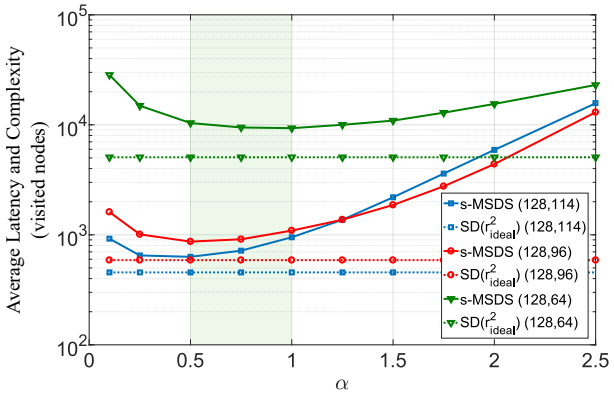

Fig. 5. Average latency and complexity (in visited nodes) of s-MSDS as a function of the parameter α for a block length of 128 and multiple code rates at 5dB SNR.

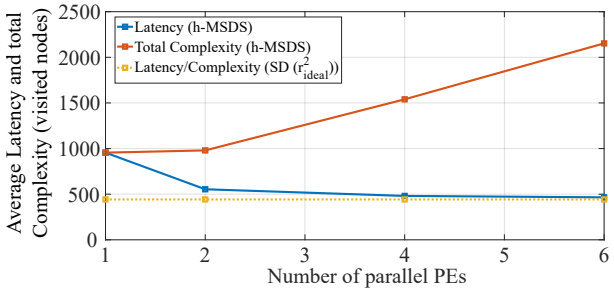

Fig. 6. Average latency and total complexity trade-off for h-MSDS with $\alpha_{seq} = 1$ as a function of N_p for (128,114) Polar coding at a 5 dB SNR.

nearly-minimum number of visited nodes for all the examined rates. In addition, it shows that the scheme is less sensitive to small α values ($\alpha < 0.5$) when the code rate is high, and it is less sensitive to large α values ($\alpha > 1$) when the code rate is low. By extensive simulations, we have verified that, in practice, α values in the interval [0.5 1] can be used for a wide range of SNR regimes, block lengths and code rates. In addition, it has been observed that the scheme is less sensitive to small α values when the SNR is high, and it is less sensitive to large α values when the SNR is low.

Figure 6 shows the latency (in terms of visited nodes of the SD that finds the ML solution) and the complexity (in terms of total number of visited nodes across all parallel SDs) when applying the hybrid MSDS (h-MSDS) SD (see Eq. 8) at a (128,114) Polar code, at 5 dB SNR, with $a_{seq} = 1$ and several N_p values. It is shown that even with $N_p = 2$ the processing latency can be reduced by more than 40% compared to the case where N_p=1 (s-MSDS), and become 25% larger than the latency of the "ideal" SD. This is achieved by increasing the total complexity by less than 2%. Increasing N_p to six, will result in a processing latency that is only 5% larger than the one of the "ideal" SD, with only a 2.25× increase in the total number of visited nodes across all parallel SDs.

As a measure of efficiency, Table I shows the percentage of tree searches for which our early termination criterion safely finds the solution for a (128,114) Polar code. The proposed termination becomes prominent in the high SNR regime, and for an SNR of 7 dB the search can be safely terminated after finding the first candidate solution for more than 80% of the tree searches.

TABLE I
PERCENTAGE OF EARLY TERMINATED DECODING ATTEMPTS OF A (128,114) POLAR CODE FOR VARIOUS SNRS.

	3 dB	5 dB	7 dB
Early Terminated Decoding Attempts	1.4%	30.7%	81.8%

V. CONCLUSION

A simple but efficient sphere-decoding-based approach has been proposed for the ML decoding of Polar codes. Compared to similar state-of-the-art Polar decoders, we have demonstrated a latency reduction of up to four orders of magnitude, while further latency reduction of 50% can be achieved by applying a limited level of parallelism.

ACKNOWLEDGMENT

The research leading to these results has been supported from the UK's Engineering and Physical Sciences Research Council (EPSRC Grant EP/M029441/1). The Authors would like to thank the members of University of Surrey 5GIC (http://www.surrey.ac.uk/5GIC) for their support.

REFERENCES

[1] E. Arikan, "Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels," *IEEE Trans. Inf. Theory*, vol. 55, no. 7, pp. 3051–3073, Jul. 2009.
[2] E. Telatar and E. Arikan, "Polarization for arbitrary discrete memoryless channels," in *IEEE Inform. Theory Workshop, ITW.*, 2009, pp. 144–148.
[3] A. Alamdar-Yazdi and F. R. Kschischang, "A simplified successive-cancellation decoder for polar codes," *IEEE Commun. Lett.*, vol. 15, no. 12, pp. 1378–1380, Jul. 2011.
[4] I. Tal and A. Vardy, "List decoding of polar codes," *IEEE Trans. Inf. Theory*, vol. 61, no. 5, pp. 2213–2226, Mar. 2015.
[5] K. Niu and K. Chen, "Stack decoding of polar codes," *Electron. Lett.*, vol. 48, no. 12, pp. 695–697, Jun. 2012.
[6] K. Chen, K. Niu, and J. Lin, "Improved successive cancellation decoding of polar codes," *IEEE Trans. Commun.*, vol. 61, no. 8, pp. 3100–3107, Jul. 2013.
[7] P. Trifonov, "Efficient design and decoding of polar codes," *IEEE Trans. Commun.*, vol. 60, no. 11, pp. 3221–3227, Aug. 2012.
[8] K. Nikitopoulos, J. Zhou, B. Congdon, and K. Jamieson, "Geosphere: Consistently turning mimo capacity into throughput," in *ACM SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 4, 2014, pp. 631–642.
[9] A. Burg, M. Borgmann, M. Wenk, M. Zellweger, W. Fichtner, and H. Bolcskei, "VLSI implementation of MIMO detection using the sphere decoding algorithm," *IEEE J. Solid-State Circuits*, vol. 40, no. 7, pp. 1566–1577, Jun. 2005.
[10] S. Kahraman and M. E. Çelebi, "Code based efficient maximum-likelihood decoding of short polar codes," in *IEEE Int. Symp. on Inform. Theory (ISIT),*, 2012, pp. 1967–1971.
[11] J. Guo and A. G. i Fabregas, "Efficient sphere decoding of polar codes," in *IEEE Int. Symp. on Inform. Theory (ISIT)*, 2015, pp. 236–240.
[12] K. Niu, K. Chen, and J. Lin, "Low-complexity sphere decoding of polar codes based on optimum path metric," *IEEE Commun. Lett.*, vol. 18, no. 2, pp. 332–335, Jan. 2014.
[13] S. Kahraman, E. Viterbo, and M. E. Çelebi, "Folded tree maximum-likelihood decoder for kronecker product-based codes," in *51st Annu. Allerton Conf. on Commun., Control, and Comput.*, 2013, pp. 629–636.
[14] K. Nikitopoulos, D. Chatzipanagiotis, C. Jayawardena, and R. Tafazolli, "Multisphere: Massively parallel tree search for large sphere decoders," *2016 IEEE Global Commun. Conf. (GLOBECOM)*, pp. 1–6.
[15] E. Arıkan, "A performance comparison of polar codes and reed-muller codes," *IEEE Commun. Lett.*, vol. 12, no. 6, pp. 447–449, Jun. 2008.
[16] I. Tal and A. Vardy, "How to construct polar codes," *IEEE Trans. Inf. Theory*, vol. 59, no. 10, pp. 6562–6582, Jul. 2013.