IMPLEMENTATION OF COMPLEX ENUMERATION FOR MULTIUSER MIMO VECTOR PRECODING

Maitane Barrenechea, Mikel Mendicute, Idoia Jimenez, and Egoitz Arruti

Department of Electronics and Computer Science, University of Mondragón
Loramendi 4, 20500, Mondragón, Spain
phone: + (0034) 943 739422, email: {mbarrenetxea, mmendikute, ijimenez, earruti}@eps.mondragon.edu

ABSTRACT

In recent years the interest in developing efficient algorithms for the multiple-input multiple-output (MIMO) broadcast channel has arisen. Vector precoding (VP) techniques have shown very promising performance results, but the perturbation process inherent in these type of non-linear systems hinders their efficient implementation. The key to achieving a high-throughput VP implementation is to operate directly on the complex-valued constellations. Nevertheless, the complex-plane Schnorr-Euchner enumerators present in the literature follow a sequential scheme, which derives in an increased system delay and high resource demand in fully-pipelined architectures. This paper presents the first non-sequential complex Schnorr-Euchner enumerator for the precoding scenario. The proposed complex enumerator has been implemented and analyzed along with other state-of-the-art enumeration algorithms. Provided hardware occupation and latency results show that the proposed enumeration algorithm is superior to other existing schemes in terms of hardware resource usage and throughput.

1. INTRODUCTION

In the multiuser MIMO broadcast channel, the use of precoding techniques is required in order to detect the signal at the user terminals without any cooperation between them. In recent years, a special focus has been drawn to non-linear precoding techniques due to their superior performance results. Specifically, vector perturbation (VP) [1] has proven to be a specially interesting precoding technique which by means of modulo arithmetics approaches the performance of the capacity-achieving albeit impractically complex dirty paper coding. From the point of view of hardware implementation, the most challenging part of the VP scheme is the computation of the perturbing signal, which entails a search for the closest-point in an infinite lattice. To avoid the high complexity of an exhaustive search, tree-search algorithms originally designed and optimized for maximum-likelihood detection have been adapted for their use in precoding scenarios.

The sphere encoder (SE) algorithm [2] restricts the search for the perturbation vector to a set of points that lie within a hypersphere of radius R centered around a reference signal. By avoiding the search over the whole lattice, the complexity of the perturbation process is greatly reduced. If the search is performed over the lattice $\mathscr{Q}$, at each level of the tree search $|\mathscr{Q}|$ child nodes originate from each parent node of the tree. However, those child nodes that do not fulfil the sphere constraint are pruned out of the tree along with all their descendants. Every time a leaf node is reached, the path leading to that node is stored as a candidate solution and the search radius R is updated with the new Euclidean distance. The search process is then resumed with the updated sphere constraint until a leaf node is reached and there are no more branches that fulfil the sphere constraint. The order in which the child nodes are visited during the tree traversal has a high impact on the number of iterations performed by the SE. By sorting the child nodes in accordance with their distance to the reference signal, as stated by the Schnorr-Euchner enumeration [3], the processing time of the SE can be greatly reduced. The undesired features of sequentiality and variable complexity of the SE stem from the sphere constraint that is checked at various stages in the algorithm. Even if this restricted search provides the algorithm with the ability to find the optimum solution, it is required that the constraint is removed or relaxed if a fixed-complexity approach is to be followed. The fixed-complexity K-Best precoder [4] selects the K best tree branches at each level of the tree search regardless of the sphere constraint. Therefore, at each one of the N stages of the K-Best tree-search, an ordering procedure has to be performed on the $K|\mathscr{Q}|$ candidate branches based on their accumulated Euclidean distances (AED) down to that particular level. The sorting stage required at each level represents the bottleneck of the algorithm. A full sorting procedure which computes all $K|\mathscr{Q}|$ AEDs and subsequently sorts them is a simple method to carry out the ordering at the N stages of the tree, but its complexity becomes prohibitive when medium-to-high values of K are used. The candidate sorting process can be simplified if ordered list merging [5] or the *winner path extension* [6] algorithms are used. Both distributed sorting methods require the implementation of a Schnorr-Euchner enumerator. The fixed sphere decoder/encoder algorithm [7][8] also performs a fixed-complexity tree search but avoids the highly complex sorting stages. The tree search is defined by a tree configuration vector $\boldsymbol{n} = [n_1, \dots, n_N]$ instead, which determines the number of child nodes (n_i) to be expanded from each parent node at every level. In the early stages of the tree search the child node selection is performed by means of a Schnorr-Euchner enumerator, whereas a simple slicer is required in the remaining levels. Consequently, there is no doubt that the enumeration process plays a vital role in vector precoding systems regardless of the particular tree-search algorithm employed.

The intricacy of a complex-plane Schnorr-Euchner enumerator [3] has led to the dominance of real-valued equivalent models as the preferred hardware architecture when implementing tree-search techniques. Nevertheless, the simplicity of the enumerator comes at the cost of an expanded tree, whose depth is twice that of the original. This derives in higher resource occupation and longer delays, which also affect the final throughput of the system.

This paper presents a novel low-complexity complex-plane enumerator for precoding. As opposed to other enumeration algorithms found in the literature where the symbols are selected in a sequential fashion, the proposed *puzzle* enumerator performs a parallel enumeration. This way, the selection of the first ρ symbols according to the Schnorr-Euchner enumeration is performed simultaneously, which reduces the delay of the enumeration process to a great extent, specially for high values of ρ. Additionally, the fully-pipelined and high-throughput implementation of the proposed algorithm and other state-of-the-art complex-plane enumerators has been carried out. Comparative results on the device occupation, multiplier utilization and delay of the different enumerators show that the proposed enumerator approach is best suited for a high-throughput implementation.

Thanks to the Department of Education, Universities and Research of the Basque Government, the Spanish Ministry of Science and Innovation for the MultiMIMO project (TEC2007-68020-C04-04) and Fundación Centros Tecnológicos- Iñaki Goenaga for funding through an Etortek research grant.

2. SYSTEM MODEL

In a VP system with M transmit antennas and N single-antenna users, the data vector to be transmitted $\boldsymbol{s} = [s_1, \ldots, s_N]^T$ is perturbed by a complex signal $\boldsymbol{a} \in \tau\mathbb{Z}^N + j\tau\mathbb{Z}^N$, where τ is the modulo constant. The value of this parameter is dependant on the constellation of data symbols, being $\tau = 8/\sqrt{10}$ for 16-QAM modulation. After the perturbation process, the precoding matrix $\boldsymbol{P}$ shapes the signal to be transmitted and a scaling factor β^{-1} is applied prior to transmission to comply with the transmit power constraints. At the user terminals, the received signal is scaled by β again to meet the modulo operation requirements. This non-linear operation at the receivers is essential if the effects of the perturbing signal $\boldsymbol{a}$ are to be reversed. The block diagram of a VP system is shown in the following figure:

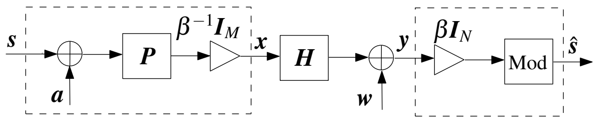


Figure 1: Block diagram of a VP system.

The VP model that achieves the minimum mean square error is supported by the following equations [9]:

$$(\boldsymbol{H}\boldsymbol{H}^H + \xi \boldsymbol{I}_N)^{-1} = \boldsymbol{U}^H \boldsymbol{U} \tag{1}$$

$$\boldsymbol{a} = \underset{\hat{\boldsymbol{a}} \in \tau\mathbb{Z}^N + j\tau\mathbb{Z}^N}{\operatorname{argmin}} \left\| \boldsymbol{U}(\boldsymbol{s} + \hat{\boldsymbol{a}}) \right\|_2^2, \tag{2}$$

$$\boldsymbol{x} = \beta^{-1} \boldsymbol{H}^H (\boldsymbol{H}\boldsymbol{H}^H + \xi \boldsymbol{I}_N)^{-1} (\boldsymbol{s} + \boldsymbol{a}),$$

where the precoding matrix is $\boldsymbol{P} = \boldsymbol{H}^H(\boldsymbol{H}\boldsymbol{H}^H + \xi \boldsymbol{I}_N)^{-1}$ and the regularization factor ξ equals the inverse of the signal-to-noise ratio (SNR). To comply with the fixed-complexity requirement, the infinite integer lattice required to optimally compute (2) will be reduced to the region $\mathscr{Q} \triangleq \{q : |\Re(q)| \leq 2\tau, |\Im(q)| \leq 2\tau\}$ of $|\mathscr{Q}| = 25$ elements with virtually no degradation on the performance of the system.

The matrix triangularization procedure in Equation (1) allows to distribute the computation of the distance metrics across multiple stages. Following such statement, the Euclidean distances that must be minimized in the cost function in (2) can be equivalently represented in a tree search fashion as

$$D_i = U_{ii}^2 \|a_i + z_i\|^2 + \sum_{j=1}^{i-1} U_{jj}^2 \|a_j + z_j\|^2 = d_i + D_{i-1}, \tag{3}$$

and

$$z_i = s_i + \sum_{j=1}^{i-1} \frac{U_{ij}}{U_{ii}} (a_j + s_j), \tag{4}$$

where U_{ii} represents the entries of the $\boldsymbol{U}$ matrix and z_i refers to the intermediate point at level i. The AEDs up to level i can be then computed by adding the partial Euclidean distance (PED) or distance increment of that level to the previous AED, that is, $D_i = d_i + D_{i-1}$.

3. COMPLEX ENUMERATION

The Schnorr-Euchner enumeration [3] dictates the order in which a certain set of nodes is to be visited according to their distance to z_i. When the real equivalent model is used, the ordering of the child nodes is a simple task as the nodes just need to be visited in a zig-zag fashion. This process is represented in Figure 2(b) for the lattice elements of the real and imaginary axis. However, such an straightforward scheme cannot be followed in the complex plane.

To avoid the high complexity and resource demand of a full-sorting enumeration, a novel approach based on trigonometric calculations was presented in [10]. The proposed scheme was originally developed to determine the admissible interval of constellation points taking into account the sphere constraint set by the sphere decoder. To that end, the constellation symbols are arranged in concentric circles whose intersection with the search disk centered at z_i is identified as the admissible set. Once the boundaries for the search space have been established, the symbols are visited in a zig-zag fashion within each concentric circle.

The intricacies of performing the highly complex trigonometrical calculations required by this approach motivated the work in [11], were an implementation-friendly version of the ideas in [10] was presented. In Figure 2(a), the procedure for complex enumeration is illustrated for the lattice under consideration. Note that since the lattice of precoding symbols also includes null components, the parameters of the algorithm in [11] have been slightly modified to adjust to the VP case. Therefore, for the lattice of $|\mathscr{Q}| = 25$ elements, the symbols are arranged in $P = 6$ subsets were the 0 element is the unique member of one of the subsets. As an starting point for this low-complexity algorithm, the value of z_i is mapped into the first quadrant to reduce the amount of decision boundaries to be checked. As a result to this, all the enumerated values will have to be mapped back to their original quadrants once the enumeration process is finished. Next, the initial symbols in each concentric circle are chosen according to a simple boundary checking procedure. Once all the initial symbols have been identified, their PEDs are computed and the one with the smallest distance increment is selected as the starting point for the enumeration ($a_i^{(1)}$). The process is continued by selecting the following symbol within the subset of $a_i^{(1)}$ that minimizes the phase difference with respect to z_i. This local enumeration procedure within each one of the P subsets is performed in a zig-zag fashion were the direction of the enumeration is determined by an additional set of boundary conditions. Next, the PED of the newly enumerated symbol is computed and compared to the previously calculated $P-1$ distance increments and the one with the smallest PED is selected as the next enumerated value ($a_i^{(2)}$). The algorithm proceeds accordingly until no more symbols need to be evaluated or all subsets are empty. Note that P incremental distance calculations need to be performed at the initial stage and just one PED computation for each one of the following enumerated points.

Another method for complex enumeration was proposed in [12]. This novel approach does not require the arrangement of symbols in concentric circles, but uses the simple zig-zag enumeration in the real and imaginary components of the lattice symbols as a mean to determine the order for the complex Schnorr-Euchner enumeration. Therefore, the first step of the algorithm is to order the elements in the real and imaginary axis according to their proximity to z_i as shown in Figure 2(b). The elements $\{a_{\Re}^1, \ldots, a_{\Re}^5\}$ and $\{a_{\Im}^1, \ldots, a_{\Im}^5\}$ represent the enumerated values in the real and imaginary axis, respectively. Clearly, the first enumerated value in the complex lattice $a_i^{(1)}$ will be composed of the first enumerated values in the real and imaginary axis, namely $a_i^{(1)} = a_{\Re}^1 + a_{\Im}^1 j$. To determine the rest of the values, a candidate list $\mathscr{L}$ needs to be defined. After the initial enumeration point has been selected, the two adjacent symbols according to the one-dimensional enumeration are added to the list of candidates. Therefore, the symbols corresponding to $a_{\Re}^2 + a_{\Im}^1 j$ and $a_{\Re}^1 + a_{\Im}^2 j$ are included in $\mathscr{L}$ and their corresponding PEDs are computed. The one with the smallest PED is selected as $a_i^{(2)}$ and its two adjacent symbols are added to the candidate list. Note that only one new symbol will be included in $\mathscr{L}$ if one of the adjacent symbols has previously been selected. This derives in a variable length candidate list, whose number of elements cannot be determined beforehand. Finally, in [13] an approach for search sequence determination in tree search algorithms for point-to-point MIMO detection was presented. Despite its sim-

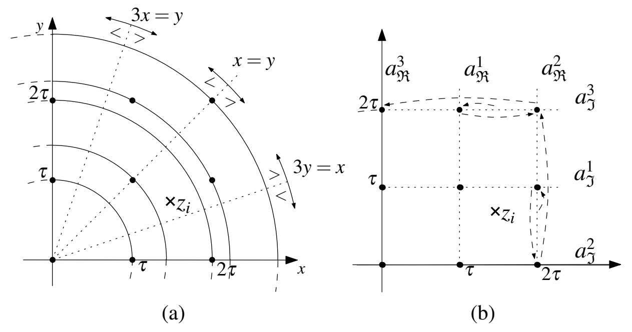


Figure 2: Arrangement of the complex lattice symbols according to different enumeration techniques.

plicity, the algorithm failes to provide valid enumerated values for reference points outside the constellation's grid. Moreover, since the proposed algorithm only determines the first 8 enumerated values and uses a predefined sequence for the following points in the enumeration, a certain performance degradation is introduced.

Most of the enumeration techniques that have been published so far in the literature follow a sequential scheme, that is, the closest node to z_i is obtained first, then the second closest point is computed and so on. The dependency on the past enumerated values in order to perform the selection of the actual enumerated point comes as a logical way of proceeding, but incurs in a great latency of the enumeration algorithm. High latencies in entities within the critical path of the system derive in significant throughput reduction. Moreover, in high-throughput fully-pipelined architectures, long delays incur in extended device occupation due to the high amount of required pipeline registers. To overcome the issues of sequential enumeration, a novel complex plane enumeration technique is presented in this paper.

4. NOVEL PUZZLE ENUMERATOR

In this section an original complex-enumerator is presented. The proposed model allows for the independent and low-complexity computation of the enumerated values. Since the information on the previously selected $\rho - 1$ symbols is not required in order to obtain the ρ^{th} enumerated value, the latency of the presented enumeration unit is not increased as the enumeration process progresses. Moreover, the selection of the symbols is performed without the computation of their associated distances, which results in a considerable resource saving.

An important factor in the proposed enumeration approach is the subset of the infinite integer lattice to be employed. Even if a subset of $|\mathscr{Q}| = 25$ elements is enough to get close-to optimal bit error rate (BER) results, an extended lattice will be used with the *puzzle* enumerator. As opposed to the detection problem, where all the lattice symbols are equally probable and belong to a modulation constellation, in VP systems those lattice points that are closer to the origin are more likely to be part of the solution vector. As a consequence to this, the values of z_i, which somehow represent the previous symbols in the search tree, will lie within a circle around the origin. In an scenario of equally probable symbols, such as the detection problem, the values of z_i are scattered around the whole constellation. The radius R_z of the disk where the z_i values are concentrated in VP systems grows proportionally with the SNR. One of the key factors of the *puzzle* enumerator is to select the minimum lattice dimension that will allow for simple enumeration in the real and imaginary components in a zig-zag fashion. If the lattice is too small and there are z_i values in the vicinity of the lattice border, the enumeration in the real or imaginary components will not result in a zig-zag pattern. This will derive in undesired border effects which would require extra hardware resources and boundary checks. The minimum amount of points of the extended lattice $\mathscr{Q}_{\text{ext}}$ depends on both R_z and the number of values ρ to be enumerated:

$$|\mathscr{Q}_{\text{ext}}| = \begin{cases} (2(\lceil R_z \rfloor_\tau + \tau) + 1)^2 & \text{for } \rho \in [2,7] \\ (2(\lceil R_z \rfloor_\tau + 2\tau) + 1)^2 & \text{for } \rho \in [8,19] \\ (2(\lceil R_z \rfloor_\tau + 3\tau) + 1)^2 & \text{for } \rho \in [20,37] \\ \vdots & \vdots \end{cases}$$

where $\lceil \cdot \rfloor_\tau$ represents the operation of rounding to the closest lattice symbol.

Once the lattice parameters have been set, the low-complexity enumerator proceeds by performing a simple zig-zag enumeration in the real and imaginary axis as shown in Figure 2(b). Only a few values need to be enumerated in the real and imaginary axis depending on ρ. This way, 2 values need to be enumerated for $\rho \in [2,3]$, 3 for $\rho \in [4,7]$, 4 for $\rho \in [8,11]$, and so on. This simple enumeration procedure is necessary as the results of the proposed enumeration unit are given as a combination of the real and imaginary enumerated values, namely $a_i^{(\rho)} = a^p_{\Re} + a^q_{\Im} j$.

For a certain $a_i^{(\rho)}$ only a few points are eligible. The initial value, namely $a_i^{(1)}$, is certainly easy to acquire as $a^1_{\Re} + a^1_{\Im} j$ is the only suitable symbol. However, in order to the determine the rest of the values a set of boundary checks needs to be performed. In Figure 3 the decision boundaries for the first eight enumerated values are depicted. As is shown in the figure, each bounded region corresponds to a candidate lattice point. We will define $\Delta_{\Re} = |z_{\Re} - a^1_{\Re}|$ and $\Delta_{\Im} = |z_{\Im} - a^1_{\Im}|$, with $z_i = z_{\Re} + z_{\Im} j$, $0 \leqslant \Delta_{\Re} < \tau/2$ and $0 \leqslant \Delta_{\Im} < \tau/2$. Note that the computation of $\Delta_{\Re}$ and $\Delta_{\Im}$ incurs in insignificant extra hardware usage as they can be easily obtained by means of simple signal slicing during the real and imaginary axis enumeration procedures, respectively. The set of boundary conditions are then built in the form of $A\,\Delta_{\Re} \pm B\,\Delta_{\Im} = C\tau$ which only requires of an adder and simple constant multiplication module, which in the case of $A = 2$ or $B = 2$ is equivalent to a mere bit shifting. Furthermore, some boundary lines are common for several ρ values which reduces the amount of boundary expressions to be computed. As can be seen if Figure 3, the enumeration areas are symmetrical with respect to the diagonal. Nevertheless, exploiting the symmetry of the puzzles to reduce the enumeration process to the lower-most triangle is a counterproductive measure as the gain in resource usage does not compensate for the hardware required to map the signal into the desired region. The border lines for the enumeration puzzles in Figure 3 are the following:

$$\begin{aligned} B_\alpha &: \quad \Delta\Re = \Delta\Im \\ B_\gamma &: \quad 2\,\Delta\Im + \Delta\Re = \tau/2 \\ B_\delta &: \quad 2\,\Delta\Re + \Delta\Im = \tau/2 \\ B_\varepsilon &: \quad 2\,\Delta\Im - \Delta\Re = \tau/2 \\ B_\eta &: \quad 2\,\Delta\Re - \Delta\Im = \tau/2 \\ B_\kappa &: \quad 3\,\Delta\Im - \Delta\Re = \tau \\ B_\lambda &: \quad 3\,\Delta\Re - \Delta\Im = \tau \end{aligned}$$

The so-called boundary bits identify each one of the two regions separated by their corresponding boundary line. For example, the boundary bit b_α associated to the boundary line B_α will be set to '1' when $\Delta\Re > \Delta\Im$ and '0' otherwise. In order to resolve which puzzle region corresponds to a certain value of z_i, it is sufficient to make logic combinations of the required boundary bits and to select the candidate lattice point accordingly by means of a multiplexer. Examples of the circuitry for the second, fourth and eighth enumerated values can be seen in Figure 4. As one can notice, the selection of all the depicted enumerated values can be performed simultaneously, as there is no data dependance among them. As opposed to other enumeration approaches in the literature, the proposed *puzzle* enumerator does not base the selection of a certain enumerated value upon Euclidean distance calculations and subsequent comparisons. Due to this fact, it is possible to perform an optimum enumeration

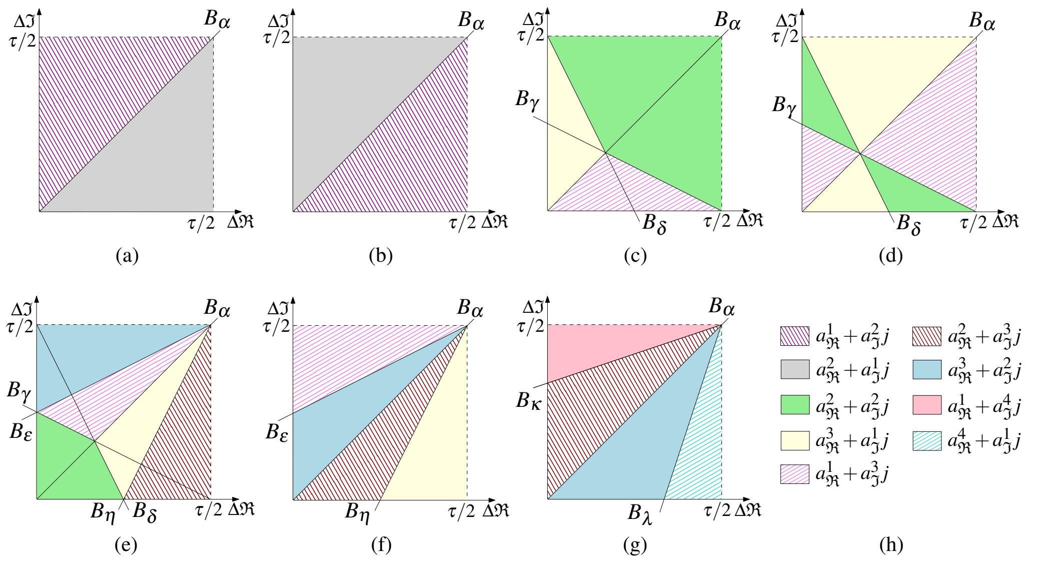


Figure 3: Fundaments of the proposed puzzle enumerator for (a) $\rho = 2$, (b) $\rho = 3$,(c) $\rho = 4$,(d) $\rho = 5$,(e) $\rho = 6$,(f) $\rho = 7$ and (g) $\rho = 8$.

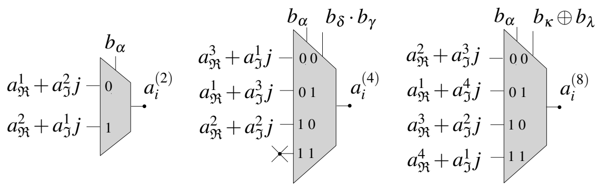
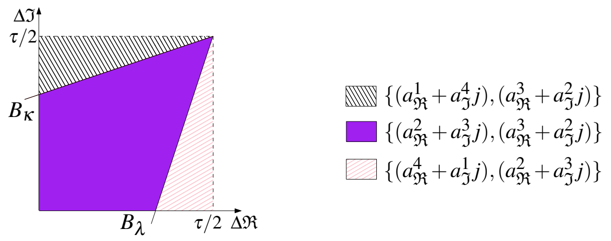


Figure 4: Hardware implementation model of the proposed complex enumerator for the second, fourth and eighth enumerated values.



$\Delta\Im$
$\tau/2$
B_κ
B_λ
$\tau/2$ $\Delta\Re$
$\{(a^1_\Re + a^4_\Im j), (a^3_\Re + a^2_\Im j)\}$
$\{(a^2_\Re + a^3_\Im j), (a^3_\Re + a^2_\Im j)\}$
$\{(a^4_\Re + a^1_\Im j), (a^2_\Re + a^3_\Im j)\}$

Figure 5: Unordered puzzle enumerator for $\rho = 8$.

and, at the same time, compute suboptimum distance increments based on the l^1 or l^∞ norms in order to reduce the amount of hardware resources required by the precoder's architecture. Performing such a hardware reduction procedure with the rest of the algorithms incurs in faulty enumeration.

4.1 Unordered puzzle enumerator

For the case of an FSE precoder any of the aforementioned enumerators can be utilized to select the n_i closest symbols. However, since no ordering procedure is performed on the selected nodes, it is sufficient to merely identify the set of the n_i closest symbols regardless of their order. This way, a further simplification on the proposed *puzzle* enumerator can be performed. By combining the regions of the puzzles from $\rho = 2$ up to $\rho = n_i$, it is possible to identify the symbols that will be present in the first n_i enumerated values. For example, for the case of $n_i = 8$, the set of preferred children will be composed of $(a^1_\Re + a^1_\Im j)$, $(a^1_\Re + a^2_\Im j)$, $(a^2_\Re + a^1_\Im j)$, $(a^2_\Re + a^2_\Im j)$, $(a^1_\Re + a^3_\Im j)$, $(a^3_\Re + a^1_\Im j)$ and two further candidates that can be determined by solving the puzzle shown in Figure 5.

5. IMPLEMENTATION AND COMPARATIVE ANALYSIS OF COMPLEX ENUMERATION TECHNIQUES

This section analyzes the implementation aspects of various complex enumerators. Specifically, fully-pipelined architectures of the full sort enumerator, the approaches in [11], [12] and the proposed *puzzle* enumerator have been carried out. All of the aforementioned units perform the enumeration of the first $\rho = 8$ symbols in the complex-plane. As has been previously stated, all the enumerators except for the *puzzle* enumerator, base the selection of a certain enumerated symbol on the distance computations of the symbols in a candidate set. Even if the purpose of an enumerator is solely to identify a certain set of points in the vicinity of z_i, some of the calculated PEDs can be later reused by the tree search algorithm. Therefore, for the sake of a fair comparison between enumeration units, $\rho = 8$ metric computation units (MCU) have been added to the output of the *puzzle* enumerator. This way, the occupation and delay results shown in this section reflect the implementation of enumeration units that output both the selected symbols and their associated distance increments.

The PEDs that are to be computed share several common components. Depending on the architecture of the enumerator, different approaches to reduce the number of costly operations can be implemented. For the full sort and the precoding version of the algorithm in [11], the resource sharing distance increment procedure described in the latter will be used. This way, the PED computation in Equation (3) will be rewritten as a function of the arc ν and the real and imaginary components of the lattice symbols:

$$d_i(\nu, \mathscr{P}_f, \mathscr{P}_g) = U^2_{ii}\left(|z_i|^2 + R^2_\nu + 2(z_\Re \mathscr{P}_f + z_\Im \mathscr{P}_g)\right), \qquad (5)$$

where R_ν represents the radius of arc ν and $\mathscr{P}_g$ stands for the g^{th} symbol of the real and/or imaginary axis. Following this approach, the amount of required multipliers can be reduced as $|z_i|^2$ needs to be computed only once and the values for the different R^2_ν can be stored as constants. Moreover, the result for the multiplications of the form $z_\Re \mathscr{P}_f$ and their imaginary counterparts can be performed by means of inexpensive constant multiplication modules. A different resource sharing approach has been followed for the MCUs of the enumerator introduced in [12] and the *puzzle* enumerator. The

	Enumeration + MCUs					Enumeration only	
	Full sort	**[11]**	**[12]**	**Puzzle**	**Puzzle$^{\theta}$**	**Puzzle**	**Puzzle$^{\theta}$**
Number of occupied slices	4362	2406	1931	510	262	186	133
Number of slice LUTs	12762	6770	5091	1219	893	498	332
Number of DSP48Es	10	10	20	16	16	0	0
Total gate equivalent count	627 K	414 K	293 K	19 K	12 K	6 K	4 K
Maximum clock (MHz)	233	182	250	250	285	333	335

Table 1: Device occupation and maximum achievable frequency for different complex enumerators in a fully-pipelined scheme.

PEDs in this model are computed as the sum of the distance increments of the components in the real and imaginary axis:

$$d_i(p,q) = U_{ii}^2\left(\left(z_{\Re}+a_{\Re}^{p}\right)^2+\left(z_{\Im}+a_{\Im}^{q}\right)^2\right).$$

In Table 2 the latency of the implemented complex enumerators is shown. Specifically, the amount of clock cycles φ required in order to select the ρ^{th} enumerated value and to compute the corresponding PED are depicted. On one hand, the sequential nature of the full sort scheme and the algorithms in [11] and [12] can be noticed from the data shown in Table 2. On the other hand, the *puzzle* enumerator shows a shorter and constant delay, with the exception of the first value which is obtained without any boundary checking. Note that most of the latency is due to the distance increment computation as the sole enumeration procedure can be carried out in just two clock cycles.

The fully-pipelined architectures of the proposed enumerator along with the state-of-the-art enumerators reviewed in this paper have been implemented on a Virtex-5 XC5Vlx30-3 FPGA. The device occupation results and the achievable maximum frequency for the different designs are depicted in Table 1, which shows a considerably smaller device occupation for the proposed scheme. The total gate equivalent (GE) count of the *puzzle* enumerator is 22 and 15 times smaller than that required by the approaches in [11] and [12], respectively. Furthermore, the gate equivalent count of the full sorting model is 33 times higher than the proposed enumerator. For the case of the unordered enumeration (Puzzle$^{\theta}$) presented in Section 4.1, an additional resource saving can be achieved due to its simplified structure. The approaches that follow the distance computation simplification in (5) show the smallest amount of required embedded multipliers (DSP48Es). Nevertheless, in case a l^1 norm MCU is used along with the *puzzle* enumerator the multiplier count for the proposed model would be reduced to 8.

Additionally, the device occupation results for the *puzzle* enumerator core without the MCUs is provided for completion. Resource usage results show the low complexity of the proposed enumerator, which can be implemented with just 6K GE.

	Enumeration + MCUs				Enum. only
	Full sort	**[11]**	**[12]**	**Puzzle**	**Puzzle**
$\varphi(1)$	14	12	9	9	1
$\varphi(2)$	19	17	10	10	2
$\varphi(3)$	24	22	13	10	2
$\varphi(4)$	29	27	17	10	2
$\varphi(5)$	34	32	22	10	2
$\varphi(6)$	39	37	27	10	2
$\varphi(7)$	44	42	32	10	2
$\varphi(8)$	49	47	37	10	2

Table 2: Number of clock cycles to obtain the enumerated values.

6. CONCLUSIONS

This paper has presented the design and implementation of a parallel complex-plane enumerator and its application to vector precoding. When compared to other complex enumerators in the literature, the proposed model has shown a considerably smaller and constant delay as well as a much reduced area occupation. The *puzzle* enumerator is specially useful for fully-pipelined fixed-complexity high-throughput precoders where the amount of symbols to enumerate is fixed. Nevertheless, the presented enumeration unit can also be used in iterative or sequential systems such as the SE if limited enumeration is employed.

REFERENCES

[1] B. Hochwald, C. Peel, and A. Swindlehurst, "A vector-perturbation technique for near-capacity multiantenna multiuser communication. Part II: perturbation," *IEEE Transactions on Communications*, vol. 53, pp. 537–544, 2005.

[2] M. Damen, H. El Gamal, and G. Caire, "On maximum-likelihood detection and the search for the closest lattice point," *IEEE Transactions on Information Theory*, vol. 49, pp. 2389 – 02, 2003.

[3] C. P. Schnorr and M. Euchner, "Lattice basis reduction: improved practical algorithms and solving subset sum problems," in *Proceedings FCT*, 1991, pp. 181–191.

[4] R. Habendorf and G. Fettweis, "Vector precoding with bounded complexity," in *Proceedings SPAWC*, 2007, pp. 1–5.

[5] M. Wenk, M. Zellweger, A. Burg, N. Felber, and W. Fichtner, "K-Best MIMO detection VLSI architectures achieving up to 424 Mbps," in *Proceedings IEEE ISCAS*, 2006, pp. 1151–1154.

[6] S. Mondal, W. Ali, and K. Salama, "A novel approach for K-Best MIMO detection and its VLSI implementation," in *Proceedings IEEE ISCAS*, 2008, pp. 936–939.

[7] L. G. Barbero and J.S. Thompson, "A fixed-complexity MIMO detector based on the complex sphere decoder," in *Proceedings IEEE SPAWC*, 2006, pp. 1 – 5.

[8] M. Barrenechea, M. Mendicute, J. Del Ser, and J. Thompson, "Wiener filter- based fixed-complexity vector precoding for the MIMO downlink channel," in *Proceedings IEEE SPAWC*, 2009, pp. 216 – 220.

[9] D. Schmidt, M. Joham, and W. Utschick, "Minimum mean square error vector precoding," *European Transactions on Telecommunications*, vol. 19, pp. 219–231, 2008.

[10] B.M. Hochwald and S. ten Brink, "Achieving near-capacity on a multiple antenna channel," *IEEE Transactions on Communications*, vol. 51, pp. 389–399, 2003.

[11] A. Burg, M. Borgmann, M. Wenk, M. Zellweger, W. Fichtner, and H. Bölcskei, "VLSI implementation of MIMO detection using the sphere decoding algorithm," *IEEE Journal of Solid-State Circuits*, vol. 40, pp. 1566–1577, 2005.

[12] M. Shabany, K. Su, and P. Gulak, "A pipelined scalable high-throughput implementaion of a near-ML K-Best complex lattice decoder," in *Proceedings IEEE ICASSP*, 2008, pp. 3173–3176.

[13] B. Mennenga and G. Fettweis, "Search sequence determination for tree search based detection algorithms," in *Proceedings IEEE Sarnoff Symposium*, 2009.