Improving the efficiency of the LDPC code-based McEliece cryptosystem through irregular codes

Marco Baldi, Marco Bianchi, Nicola Maturo, Franco Chiaraluce, DII, Università Politecnica delle Marche, Ancona, Italy
Email: \{m.baldi, m.bianchi, n.maturo, f.chiaraluce\}@univpm.it

Abstract—We consider the framework of the McEliece cryptosystem based on low-density parity-check (LDPC) codes, which is a promising post-quantum alternative to classical public key cryptosystems. The use of LDPC codes in this context allows to achieve good security levels with very compact keys, which is an important advantage over the classical McEliece cryptosystem based on Goppa codes. However, only regular LDPC codes have been considered up to now, while some further improvement can be achieved by using irregular LDPC codes, which are known to achieve better error correction performance than regular LDPC codes. This is shown in this paper, for the first time at our knowledge. The possible use of irregular transformation matrices is also investigated, which further increases the efficiency of the system, especially in regard to the public key size.

Index Terms—McEliece cryptosystem, irregular LDPC codes.

I. INTRODUCTION

A renewed interest is being devoted to code-based cryptosystems, since they are recognized to be able to resist attacks based on quantum computers, which will seriously endanger widespread solutions, like the Rivest, Shamir, Adleman (RSA) system, based on the integer factorization problem. The best known code-based public key cryptosystem is the McEliece cryptosystem [1], which relies on the problem of decoding a random linear block code with no visible structure. This system, in its original formulation, has never encountered any polynomial time attack, and is able to guarantee very fast encryption and decryption procedures. Its major drawbacks are the encryption rate, which is smaller than 1, and, most of all, the large size of its public keys.

In fact, the original solution adopts Goppa codes, which are able to ensure very high security levels, but require the public matrices to be unstructured. Hence, their storage needs a great amount of memory. The most recent proposals concerning Goppa codes provide updated choices of the system parameters to reduce the public key size and increase the security level [2]. Despite this, the public key size is still very large: the parameters proposed in [2] to achieve 128-bit security yield a public key size of 1,537,536 bits.

Replacing Goppa codes with structured codes allows achieving considerable reductions in the key size, though security issues must be taken into account. Recently, several proposals based on quasi-cyclic low-density parity-check (QC-LDPC) codes have appeared [3]–[7], showing that these codes are actually a promising alternative to traditional Goppa codes. LDPC codes are capacity achieving codes [8] defined through sparse parity-check matrices. They are employed in several frameworks [9]–[11], and also used in some security-related contexts [12]. Their use in the McEliece cryptosystem has been studied since several years [13]–[15], and some refinements have been progressively introduced to eventually achieve a secure instance of the system. The use of structured LDPC codes, like QC-LDPC codes, allows to considerably reduce the key size, though renouncing the sparse character of the public matrices.

Up to now only regular QC-LDPC codes have been considered for the use in this context, while it is well known that irregular LDPC codes can achieve better error correcting performance than regular codes [16]. In this paper, we show that such feature allows to further reduce the public key size. Another contribution to the same goal comes from the adoption of irregular transformation matrices.

The paper is organized as follows: in Section II, we recall the QC-LDPC-code based McEliece cryptosystem; in Section III, we define irregular QC-LDPC codes to be used in this context, and assess their performance through simulations; in Section IV, we provide a theoretical tool for estimating the error correction capability of irregular codes; in Section V, we assess the security level of the system; in Section VI, we show the advantage achieved by irregular codes in terms of key size through some examples; finally, Section VII concludes the paper.

II. QC-LDPC CODE-BASED MCEILIECE CRYPTOSYSTEM

The McEliece cryptosystem based on QC-LDPC codes uses codes with length \( n = n_0 \cdot p \), dimension \( k = k_0 \cdot p \) and redundancy \( r = p \), where \( n_0 \) is a small integer (e.g., \( n_0 = 2,3,4 \)), \( k_0 = n_0 - 1 \), and \( p \) is a large integer (on the order of some thousands or more). It follows that the code rate is \( \frac{n_0 - 1}{n_0} \), which coincides with the encryption rate. Differently from other solutions, like RSA, the McEliece cryptosystem has encryption rate \( < 1 \), which yields some overhead in the ciphertext, due to the code redundancy. While the original McEliece cryptosystem used codes with rate about 1/2, its most recent variants are focused on code rates on the order of 0.7. Concerning the QC-LDPC code-based variant, we consider the choice \( n_0 = 4 \), that is, an encryption rate equal to 3/4, which is in line with the most recent proposals.
In the QC-LDPC code-based McEliece cryptosystem, the main component of the private key is a QC-LDPC matrix having the following form [4], [17]:

$$\mathbf{H} = [\mathbf{H}_0 | \mathbf{H}_1 | \ldots | \mathbf{H}_{n_0-1}] ,$$  

(1)

where each $\mathbf{H}_i$ is a circulant matrix with size $p \times p$.

In all previous proposals, the matrix $\mathbf{H}$ as in (1) was regular, that is, with constant column weight $d_c$ and constant row weight $d_r = n_0 d_c$. In this work, we analyze a more general form of $\mathbf{H}$ by considering non-constant column weights. However, differently from completely general irregular LDPC codes, we must preserve the quasi-cyclic (QC) nature of the codes, since it provides important advantages in terms of the public key size.

Hence, we consider a private QC-LDPC matrix which still has the form (1), but formed by $n_0$ circulant blocks with different column weights: $\{d^{(0)}_c, d^{(1)}_c, \ldots, d^{(n_0-1)}_c\}$.

Now, $d_c$ has the meaning of average column weight, i.e., $d_c = \sum_{i=0}^{n_0-1} d^{(i)}_c / n_0$, and the row weight is still constant and equal to $d_r = n_0 d_c$.

Other two matrices are needed to form the private key: a $k \times k$ non singular random scrambling matrix $\mathbf{S}$ and an $n \times n$ non singular sparse transformation matrix $\mathbf{Q}$. In the previous versions of the QC-LDPC code-based McEliece cryptosystem [3], [4], also $\mathbf{Q}$ was a regular matrix, with fixed row and column weight $m$. We generalize it by defining $\mathbf{Q}$ as a sparse irregular matrix with average row and column weight $m$. As we will see in the following, $\mathbf{Q}$ affects the weight of the error vectors. In order to maintain its effect uniform, independently of the error vector, we impose that the row and column weights of $\mathbf{Q}$ have minimal dispersion around their mean, that is, they differ from $m$ by less than 1. This allows choosing rational values for $m$, which gives a further degree of freedom for improving the system efficiency. Moreover, in order to preserve the QC structure for the public matrices, the matrix $\mathbf{Q}$ must be QC as well, that is, formed by $n_0 \times n_0$ circulant submatrices, each with size $p \times p$. This choice limits the resolution on the value of $m$, which cannot vary by less than $1/n_0^2$, but it is sufficient to ensure enough granularity in this context. For preserving the QC form of the public keys, also $\mathbf{S}$ must be QC, that is, formed by $k_0 \times k_0$ circulant blocks with size $p \times p$.

The public key is obtained as $\mathbf{G}' = \mathbf{S}^{-1} \cdot \mathbf{G} \cdot \mathbf{Q}^{-1}$; hence, its size depends on the representation of $\mathbf{G}'$. The QC nature of the codes and of the scrambling and transformation matrices allows to achieve a very compact representation, since each circulant block is simply described by its first row. In addition, using a CCA2 secure conversion of the system [2] allows adopting public matrices in systematic form; hence, the public key size becomes $k_0 \cdot (n_0 - k_0) \cdot p = (n_0 - 1) \cdot p$ bits. This gives an important improvement with respect to Goppa code-based instances.

Similarly to the original McEliece cryptosystem, encryption is performed according to the following steps:

i) Alice gets Bob’s public key $\mathbf{G}'$.

ii) She divides her message into $k$-bit vectors.

iii) For each $k$-bit vector $\mathbf{u}$, she generates a random intentional error vector $\mathbf{e}$ with weight $t'$.

iv) She encrypts $\mathbf{u}$ into $\mathbf{x}$ as follows:

$$\mathbf{x} = \mathbf{u} \cdot \mathbf{G}' + \mathbf{e}.$$  

(2)

Decryption is performed as follows:

i) Bob inverts the secret transformation:

$$\mathbf{x}' = \mathbf{x} \cdot \mathbf{Q} = \mathbf{u} \cdot \mathbf{S}^{-1} \cdot \mathbf{G} + \mathbf{e} \cdot \mathbf{Q}$$  

(3)

and obtains a codeword of the secret LDPC code affected by the error vector $\mathbf{e} \cdot \mathbf{Q}$, with weight $\leq t = t' m$.

ii) He corrects all the errors through LDPC decoding and obtains $\mathbf{u} \cdot \mathbf{S}^{-1}$.

iii) He recovers $\mathbf{u}$ through multiplication by $\mathbf{S}$.

The main difference with respect to the original McEliece cryptosystem is in the matrix $\mathbf{Q}$, which was a permutation matrix in the original system, while now it has average row and column weight $m > 1$. This causes propagation of the intentional errors during decryption, and their number is increased at most by a factor $m$. Hence, the secret QC-LDPC code must be able to correct up to $t = t' m$ errors, rather than $t'$. On the other hand, this allows protecting the private key from attacks aimed at exploiting its sparsity, as we will briefly recall in the following.

Moreover, the sparse parity-check matrix of the secret code ($\mathbf{H}$) is mapped into a new matrix for the public code:

$$\mathbf{H}' = \mathbf{H} \cdot \mathbf{Q}^T.$$  

(4)

(superscript $T$ denotes the transpose) and, though a suitable choice of $m$, the density of $\mathbf{H}'$ can be made high enough to avoid attacks to the dual code (see Section V).

### III. IRREGULAR QC-LDPC CODES PERFORMANCE

It is known that irregular LDPC codes are able to achieve better performance than regular ones [16]. Looking at the code Tanner graph, an irregular code is defined through its variable and check nodes degree distributions. According to the notation in [16], an irregular Tanner graph with maximum variable node degree $\nu_{\text{v}}$ and maximum check node degree $\nu_{\text{c}}$ is described through two sequences, $(\lambda_1, \ldots, \lambda_{\nu_{\text{v}}})$ and $(\rho_1, \ldots, \rho_{\nu_{\text{c}}})$, such that $\lambda_i$ ($\rho_i$) is the fraction of edges connected to variable (check) nodes with degree $i$. These sequences can be used as the coefficients of two polynomials, $\lambda(x)$ and $\rho(x)$, describing the edge degree distributions:

$$\lambda(x) = \sum_{i=1}^{\nu_{\text{v}}} \lambda_i x^{i-1},$$  

$$\rho(x) = \sum_{i=1}^{\nu_{\text{c}}} \rho_i x^{i-1}.$$  

(5)

$\lambda(x)$ and $\rho(x)$ describe the code degree distributions from the edge perspective. Alternatively, the same distributions can be described from the node perspective, through two other polynomials, $v(x)$ and $c(x)$. Their coefficients, noted by $v_i$ and $c_i$, are computed as the fractions of variable and check
nodes with degree \( i \), \( \lambda(x) \) and \( \rho(x) \) can be translated into \( v(x) \) and \( c(x) \) as follows [18]:

\[
\begin{align*}
v_i &= \frac{\lambda_i}{\sum_{j=1}^{n_0} \lambda_j}, \\
c_i &= \frac{\rho_i}{\sum_{j=1}^{n_0} \rho_j}.
\end{align*}
\]

(6)

According to our choices, all check nodes have the same degree \( d_c \), whereas, because of (1) and the assumption of a (possible) different column weight for each circulant block, a fraction \( 1/n_0 \) of the variable nodes has degree \( d_v^{(i)} \), \( i = 0, \ldots, n_0 - 1 \). This yields the following simple forms for \( v(x) \) and \( c(x) \), that will be used in the following:

\[
\begin{align*}
v(x) &= \frac{\sum_{i=0}^{n_0-1} d_v^{(i)} - 1}{n_0}, \\
c(x) &= x^{d_v-1}.
\end{align*}
\]

(7)

We note that these polynomials do not correspond to optimized degree distributions, due to the constraints imposed by the very special form (1) for \( \mathbf{H} \). In addition, the minimum value of \( d_v^{(i)} \), \( i = 0, 1, \ldots, n_0 - 1 \), can be chosen in \( \binom{n_0}{i} \) different ways, and we do not want this number to decrease enough to allow an attacker to enumerate them. However, despite being forced to obey such constraints, these irregular codes achieve significant error rate performance improvements with respect to the regular codes considered up to now.

To confirm this fact, we have focused on a fixed set of code parameters and simulated the performance achievable by regular and irregular QC-LDPC codes. We have considered \( n_0 = 4 \), \( p = 4096 \), \( d_v = 13 \), and designed four QC-LDPC codes through random difference families (RDF) [14]. One of them is regular, with \( d_v^{(i)} = 13 \), \( i = 0, \ldots, 3 \), whereas the other three are irregular, with:

- \( d_v^{(0)} = 11, d_v^{(1)} = 12, d_v^{(2)} = 14, d_v^{(3)} = 15 \),
- \( d_v^{(0)} = 9, d_v^{(1)} = 11, d_v^{(2)} = 15, d_v^{(3)} = 17 \),
- \( d_v^{(0)} = 8, d_v^{(1)} = 11, d_v^{(2)} = 15, d_v^{(3)} = 18 \).

Their decoding has been performed through the logarithmic version of the iterative soft-decision sum product algorithm (SPA) [19]. Performance of the SPA is affected by quantization issues [20], hence we have used full precision floating point operations in our simulations. Fig. 1 reports the residual bit error rate (BER) and codeword error rate (CER) after decoding for the considered codes. As expected, we observe that the irregular codes outperform the regular code with the same parameters, and increasing the code irregularity gives a larger gain.

IV. DECODING THRESHOLD

Despite numerical simulations provide a precise and practical assessment of the error correcting performance of LDPC codes, running them for each possible choice of the code parameters and the node degree distributions is extremely time consuming. On the other hand, no theoretical tools exist for predicting the correction capability of finite length LDPC codes through closed form expressions. However, in applications which do not allow the availability of soft information from the channel, like the one here considered, a good estimate of the correction capability can be obtained by computing the convergence threshold of the bit flipping (BF) decoding algorithm.

Hence, as done in [4], we resort to the BF decoding threshold by extending its computation to the case in which the codes are irregular. The main difference with respect to the case of regular codes is in the fact that the decision threshold \( b \) is no longer unique for all variable nodes, but varies with their degree. So, for the considered codes, up to \( n_0 \) different decision thresholds are used: \( b^{(i)} = d_v^{(i)} - i, i = 0, 1, \ldots, n_0 - 1 \). As regards other aspects, the algorithm works in the same way as for regular codes [4]. The choice of the values \( b^{(i)} \) is very important. In the original Gallager’s work [21], two algorithms were proposed: in the so-called Algorithm A, the decision thresholds are fixed to \( b^{(i)} = d_v^{(i)} - 1 \), while in the so-called Algorithm B they can vary between \( \lceil d_v^{(i)}/2 \rceil \) and \( d_v^{(i)} - 1 \) during decoding ([•] is the ceiling function). While Algorithm A is simpler to implement, Algorithm B is able to achieve better performance. Both algorithms implement an iterative decision process: i) each check node sends each neighboring variable node the binary sum of all its other neighboring variable nodes and ii) each variable node sends each neighboring check node its initial value, flipped or not, based on the count of unsatisfied parity-check sums coming from the other check nodes, and its comparison with the decision threshold.

The advantage of using BF decoding in this context is that its decoding threshold can be estimated through theoretical arguments very similar to those developed in [16], that extended the original Gallager’s probability recursion [21] to the case of irregular graphs. However, unlike [16], where a
binary symmetric channel was considered, the current scenario
is equivalent to a channel able to introduce a fixed number of
errors in each transmitted vector. For the QC-LDPC codes
introduced in Section II, the probability that, in an iteration,
the message originating from a variable node is correct can be
expressed as:

\[ f^b(j, q_t) = \sum_{z=b(j)}^{d_v^{(j)}-1} \left( d_v^{(j)} - 1 \right) \left[ p^{ci}(q_t) \right] \left[ p^{ci}(q_t) \right] d_v^{(j)} - 1 - z, \]

while the probability that, in an iteration, a bit that is not in
error is incorrectly evaluated is:

\[ g^b(j, q_t) = \sum_{z=b(j)}^{d_v^{(j)}-1} \left( d_v^{(j)} - 1 \right) \left[ p^{ci}(q_t) \right] \left[ p^{ci}(q_t) \right] d_v^{(j)} - 1 - z. \]

In (8) and (9), as in [4], we have:

\[
\begin{aligned}
    p^{cc}(q_t) &= \sum_{j=0}^{\min(d_v-1, q_t)} \frac{(d_v-1)(n-d_v)}{q_t^{(j)}}, \\
    p^{ci}(q_t) &= \sum_{j=0}^{\min(d_v-1, q_t)} \frac{(d_v-1)(n-d_v)}{q_t^{(j)}}, \\
    p^{ic}(q_t) &= \sum_{j=0}^{\min(d_v-1, q_t)} \frac{(d_v-1)(n-d_v)}{q_t^{(j)}}, \\
    p^{ic}(q_t) &= \sum_{j=0}^{\min(d_v-1, q_t)} \frac{(d_v-1)(n-d_v)}{q_t^{(j)}},
\end{aligned}
\]

where \( q_t \) is the average number of residual errors after the \( l \)th
iteration. In the considered system, it is \( q_t \leq t = t'm \), but we
fix \( q_0 = t = t'm \) to have a worst-case evaluation.

Based on these expressions, and considering the ideal
assumption of a cycle-free Tanner graph, we can obtain an
approximation of the number of errors in the decoded word
after the \( l \)th iteration. In using this method, we do not take
into account the distribution of the errors with regard to the
circulant blocks weight, that is, we consider the errors equally
distributed in sets having the same cardinality for each block.
However, we have numerically verified that this approximation
is largely acceptable in the considered context. Based on these
arguments, we can find \( q_t \) as a function of \( q_{t-1} \):

\[ q_t = t - \sum_{j=0}^{n_0-1} \lambda_j \left[ t \cdot f^b(j, q_{t-1}) - (n-t) \cdot g^b(j, q_{t-1}) \right]. \]

Equation (11) permits us to implement a recursive procedure
which allows computing a waterfall threshold by finding the
maximum value \( t = t_{\text{th}} \) such that \( \lim_{l \to \infty} (q_l) = 0. \)

Since different values of \( t_{\text{th}} \) can be found by different
choices of the set of \( b^{(j)} \), we can search the maximum \( t_{\text{th}} \)
for each combination of \( b^{(j)} \in \{ d_v^{(j)}/2, \ldots, d_v^{(j)} - 1 \} \), with
\( j = 0, 1, \ldots, n_0 - 1 \). We will always refer to the optimal
choice of the \( b^{(j)} \) values in the following.

We have used this method to compute the decoding thresh-
old for LDPC codes with several lengths, \( n_0 = 4, d_v = 13 \)
or 15 and two irregular node degree profiles for each value of
\( d_v \) (remind that, for irregular codes, \( d_v \) represents the average
column weight). The results obtained are reported in Fig. 2,
where they are also compared with the threshold values for
regular codes with constant \( d_v = 13 \) or 15. These results
have been obtained by considering a fixed and optimized set
of decision thresholds for the BF decoder (that is, they do
not change during iterations). As we observe from the figure,
irregular codes allow to improve the error correction capability,
coherent with the conclusion already drawn in Section III with
SPA decoding. It must be said that, if we consider a number of
errors equal to the BF decoding threshold, which is computed
under the hypothesis of absence of local cycles, a finite-length
code with local cycles in its Tanner graphs does not always
achieve a very low rate under BF decoding. However,
several improved versions of the BF algorithm can be used,
which achieve very low residual error rates when the number
of errors equals, or even overcomes, the BF threshold [4].
Hence, we can consider the BF decoding threshold as a reliable
estimation of the correction capability of the codes we consider
in this context.

V. SECURITY LEVEL

Two attack procedures mostly endanger the LDPC-code
based McEliece cryptosystem, namely: dual code attacks
(DCA) and information set decoding attacks (ISDA) [4]. So,
their work factor (WF) gives the system security level.

The target of DCA is to recover an equivalent private key
from the public key. This can be achieved by searching for the
rows of the parity-check matrix of the public code, \( H' \),
and then exploiting the possible sparsity of \( H' \) to recover \( H \)
or to directly perform LDPC decoding and correct the intentional
errors. Searching for the rows of \( H' \) is equivalent to searching
for low weight codewords in the dual of the public code. The
matrix \( H' \) has average column weight \( d_v' = m \cdot d_v \) and row
weight \( d_v' = n_0 \cdot d_v \). Hence, \( d_v' \) is chosen high enough to make
such search practically unfeasible.
The purpose of ISDA is instead to find the error vector affecting the ciphertext. This can be accomplished through algorithms for finding low-weight codewords, which is equivalent to decode a random linear block code. The QC nature of the codes facilitates this task, since each block-wise cyclically shifted version of a ciphertext is still a valid ciphertext. Hence, the attacker can consider block-wise shifted versions of an intercepted ciphertext, and search for one among as many shifted versions of the error vector.

Hence, both DCA and ISDA can be mounted by exploiting efficient algorithms to search for low weight codewords in random linear block codes, and their WF can be estimated by computing the minimum complexity of these algorithms. For this purpose, we consider the approach proposed in [22]. Actually, some advances have recently appeared in the literature [23], [24] that, however, are more focused on asymptotic evaluations rather than on reducing the complexity on finite length codes. Another recent proposal in this context is “ball collision decoding” [25]. It achieves important WF reductions asymptotically, but the improvement is negligible for the code length we consider.

We have computed the WF of DCA and ISDA, and the results obtained are summarized in Fig. 3. In the figure, the abscissa reports \(d'_v\) for the DCA WF, and \(t'\) for the ISDA WF. For both attacks, the dependence of the WF on the code length is weak, and no considerable difference is achieved by only increasing the code length. Hence, we have plotted the WF for the shortest code length here considered, that is, \(n = 16384\). Using larger codes yields some increase in the WF, which makes the value obtained from the figure a pessimistic estimate, but without any significant deviation.

![Fig. 3. DCA and ISDA WF (log2) respectively plotted as a function of the public parity-check matrix column weight \(d'_v\) and the number of intentional errors \(t'\), for \(n_0 = 4\) and \(n = 16384\).](image)

Based on the previous analysis, for a given security level and \(H\) column weight distribution, the following simple procedure allows designing the system parameters:

1. The values of \(d'_v\) and \(t'\) needed for achieving the desired security level are obtained from Fig. 3;
2. The value of \(m\) is computed as \(d'_v/d_v\);
3. The number of intentional errors to correct is computed as \(t = \lceil m \cdot t' \rceil\);
4. The code length is found from Fig. 2, such that the corresponding BF threshold overcomes \(t\).

We notice that using an irregular matrix \(Q\) avoids the need to increase \(m\) up to \([m]\), thus keeping the error propagation effect of \(Q\) as small as possible. This increases the efficiency of the system, since \(t\) and, hence, the code length are kept to their minimum. We remind, however, that \(m\) must be a multiple of \(1/n_0^2\), hence we must approximate it to the smallest multiple of \(1/n_0^2\) greater than or equal to \(d'_v/d_v\).

VI. DESIGN EXAMPLES

Let us suppose to need 100-bit security. From Fig. 3 we obtain \(d'_v = 59\) and \(t' = 47\). If we focus on \(d_v = 13\) for the private code, it results \(m = 4.5625\) (approximated to a multiple of \(1/n_0^2 = 1/16\)). Then, \(t = \lceil m \cdot t' \rceil = 215\). From Fig. 2 we find that a regular code with \(d_v = 13\) and degree profile \([8, 11, 15, 18]\), we obtain that the same BF decoding threshold is achieved for \(n = 17524, p = 4381\), and the key size becomes 13143 bits, which is a 15% reduction with respect to the regular code.

If we want to achieve 160-bit security, we obtain from Fig. 3 that \(d'_v = 97\) and \(t' = 79\) are needed. By still considering \(d_v = 13\), we obtain \(m = 7.4375\) (with the same approximation as before). It follows that \(t = 588\). From Fig. 2, we see that a regular code with \(d_v = 13\) and \(n = 54616, p = 13654\) has a BF threshold equal to 600, which is enough to correct all intentional errors. The corresponding key size is 40962 bits. If we use an irregular code with degree profile \([8, 11, 15, 18]\), the same BF decoding threshold is achieved for \(n = 46448, p = 11622\), that is, a key size of 34866 bits. Hence, using an irregular code results in a public key size reduction of about 15% also in this case.

VII. CONCLUSION

We have studied the use of irregular codes in the McEliece cryptosystem based on LDPC codes. We have considered QC-LDPC codes with irregular degree profiles, and verified, through numerical simulations and theoretical tools, that they achieve better error correction performance than regular codes also in this context. This reflects into a more efficient cryptosystem, with a public key size reduction in the order of 15% with respect to the version using regular codes.

Future work will concern the evaluation of the implementation cost for the proposed solutions [26].

REFERENCES


