Experiments on the Gentry-Halevi somewhat homomorphic scheme

M. Mikuš

***Abstract*— We have implemented the somewhat homomorphic scheme from [16]. We examined this scheme in the same way as mentioned in [16] and extend the results for a wider set of parameters and also increased the number of repetitions for each test. We focused on the dependencies between the largest supported degree and various parameters of the cryptosystem, specially also the encryption parameter q. We show that the probability q significantly influences the overall effectiveness of the scheme and that the growth of the supported degree doesn't grow linearly with the parameter t (strictly) and we give an explanation for this fact.**

***Keywords*— homomorphic encryption, cloud-computing, somewhat homomorphic scheme, largest supported degree, lattices**

I. Introduction

THE classical cryptography ended during the World War II, after the invention of computers. Modern cryptography invented more complicated algorithms that used the computers to perform the time-consuming operations automatically and very quickly. These algorithms were strictly symmetric – sender and receiver have the same keys and are in the same position (example - AES [28], [17], [18], [19], [20]). The situation changed after 1976 when Diffie and Hellman published their first asymmetric cryptosystem. Cryptography become more popular and widely used. Shortly after this change some people started to ask if there exists a cryptosystem that would allow computation with encrypted data [34]. This problem gained interest when distributed computing and outsourced computation came into question.

Fully homomorphic cryptosystems have been extensively studied in the recent years. A cryptosystem is called homomorphic if for arbitrary plaintexts p_1, p_2 it allows a computation of $p_1 \odot p_2$ only from corresponding ciphertexts $c_1 = Enc(p_1)$ and $c_2 = Enc(p_2)$ without the necessity of revealing p_1 or p_2. The operation $\odot$ is usually addition or multiplication and there are well-known examples of such additive or multiplicative homomorphic cryptosystems. A short survey was provided by [14].

Manuscript received June 27, 2011. This work was supported by the grant VEGA 1/0244/09.

M.Mikuš is with the Institute of the Computer Science and Mathematics, Slovak University of Technology, Bratislava, 81212 Slovakia (e-mail: michal.mikus@stuba.sk).

A fully homomorphic cryptosystem is one that allows computation of both addition and multiplication and is sometimes called also an algebraic cryptosystem in the literature (e.g. [33] or [32]).

The existence of a fully homomorphic cryptosystem had been an open problem for more than 30 years [34]. Various promising approaches were proposed to find a solution to this problem, e.g. [6], [26], [4] or [12], [13] from the area of symmetric cryptography, but most of them were shown ineffective or insecure [5], [36]. However, the problem was first affirmatively solved by C.Gentry in [10], [11] and [16]. Since then many variations of the Gentry's scheme were created, e.g. [9], [35], [23] and very recently [7].

A. Gentry's results

The main result of Gentry's work is that a fully homomorphic scheme can be constructed from a "somewhat homomorphic" scheme – a scheme that can perform only a limited number of aditions and multiplications – under the assumption that the number of supported additions and multiplications of the somewhat homomorphic scheme is high enough. What number of additions and multiplications is "high enough" is determined by the decryption procedure of the somewhat homomorphic scheme. If the decryption function can be evaluated homomorphically, then the scheme is called bootstrapable and it allows the construction of a fully homomorphic scheme.

The construction of a fully homomorphic scheme is based on this simple idea. Let the somewhat homomorphic scheme support $m + d$ operations, where d operations are required for homomorphic evaluation of the corresponding decryption function. We perform first m operations on ciphertexts normally and when we need to compute $(m + 1)$-st operation on some ciphertext c, we first "refresh" this ciphertext in following way. We encrypt this ciphertext with a second key of another instance of the somewhat homomorphic scheme and then we homomorphically evaluate the decryption function of the first scheme. After this operation we get the ciphertext encrypted with the second key only and we can perform correctly another m operations on this ciphertext.

B. Applications

An efficient fully homomorphic scheme would be very useful in the area of cloud computing as it would ensure the security of the data [30], [31]. Many cryptographic protocols such as election schemes, zero-knowledge protocols [8], oblivious transfer protocol [22], watermarking and fingerprinting

510-194

schemes [1], [29], lottery protocols [21] etc. would also benefit from an effective fully homomorphic cryptosystem, more detailed list of them is in [33].

C. Our contribution

In this paper we examine only the somewhat homomorphic scheme (SHS) of [16] because it is simpler, but its properties influence the effectiveness of the whole fully homomorphic system. Many questions arise during the careful examination of the SHS. The most interesting one is how the largest supported degree is influenced by the parameters N, t and q or how many multiplications are supported. We will give some answers in the section IV.

The structure of the paper is as follows: we shortly describe the necessary notations and the Gentry-Halevi cryptosystem in sections II and III. In the following sections we describe the computations and results that show the dependence of the largest supported degree on the selected parameters.

II. Preliminaries

A. Notation

We use the same notation as in [16] or [35], namely for integers a, d the reduction of a modulo d into the interval $[-d/2, d/2)$ is denoted by $[a]_d$. The standard reduction modulo d or modulo polynomial $f(x)$ is denoted by $a \bmod d$ or $a(x) \bmod f(x)$ respectively.

For any rational number $q = \frac{a}{b}$, we denote the distance between q and the nearest integer by $[q]$, i.e. $[q] = \frac{[a]_b}{b}$. The rounding of q to the nearest integer is denoted by $\lceil q \rfloor$, formally $\lceil q \rfloor = q - [q]$.

B. Background on Lattices

A lattice is a discrete subgroup of $\mathbb{R}^n$. A lattice L is called n-dimensional and full-rank (we will consider only this subclass of lattices in this paper), when it is spanned by a set of n linearly independent vectors $B = (\overrightarrow{b_1}, \ldots, \overrightarrow{b_n})$. This set is called basis of the lattice.

It is well known that every lattice has an infinite number of bases and that these bases can be obtained by an unimodular transformation from one another. Formally, for arbitrary bases B_1 and B_2 of the same lattice there exists an unimodular matrix U (with determinant ± 1) so that $B_1 = U \cdot B_2$. The absolute values of determinants of B_i is therefore equal and it is called the determinant of the lattice L.

$$L = \mathcal{L}(B) = \sum_{i=0}^{n} c_i \overrightarrow{b_i}$$

With every basis B of a lattice we can associate the half-open parallelepiped $\mathcal{P}(B) = \{\sum_{i=0}^{n} x_i \overrightarrow{b_i} : x_i \in [1/2, 1/2)\}$. For any vector $\overrightarrow{c} \in \mathbb{R}^n$ we can define $\overrightarrow{c'} = c \bmod B$ as the unique vector from $\mathcal{P}(B)$ such that $\overrightarrow{c} - \overrightarrow{c'} \in L$. This reduction modulo B can be effectively computed by $\overrightarrow{c'} = [\overrightarrow{c} \times B^{-1}] \times B = \overrightarrow{c} - \lceil \overrightarrow{c} \times B^{-1} \rfloor \times B$.

Every lattice L has also a unique Hermite normal form basis $HNF(L)$ that can be easily computed [27] and therefore it is the ideal choice for a public key for a lattice.

C. Rings and Ideals

Let $\mathbb{R}$ be a ring of integer polynomials modulo some irreducible monic polynomial $f(x)$ of degree n, i.e. $\mathbb{R} = \mathbb{Z}[x]/(f(x))$.

Let I be an ideal of R. The elements of I are polynomials of degree $n-1$ and thus have n coefficients. As ideal I is closed under addition, the coefficient vectors associated to its polynomials form a lattice. An ideal I is called principal if it is generated by a single element $v(x) \in I$. The ideal I then corresponds to the lattice generated by vectors $V = \{\overrightarrow{v_i} = v(x) \times x^i \bmod f(x)\}$. We call the set V as a rotation basis of I.

III. The Gentry-Halevi cryptosystem

This somewhat homomorphic scheme is a GGH-type [15] cryptosystem based on ideal lattices. The underlying algebraic structure is the integer ring $\mathbb{R} = \mathbb{Z}[x]/(f(x))$, where $f(x)$ is monic irreducible and of degree N. The plaintext space of the scheme defined as $P = \{0, 1\}$, the ciphertexts are numbers of size approx. 2^t (will be explained later).

A. Key generation

We pick a random polynomial $v(x)$ that generates ideal/lattice $J \subseteq \mathbb{R}$, which must have a special form of $HNF(J)$ (namely the $HNF(J)$ must be represented by two integers d, r – this property would allow simpler computation during encryption and decryption process). A random polynomial $v(x)$ satisfies this requirement with a probability approx. 1/2, so we must generate two such ideals on average during the key generation procedure. Then we compute a polynomial $w(x) \in \mathbb{R}$ s.t. $v(x) \times w(x) = d \bmod f(x)$. The number d is the resultant of $v(x)$ and $f(x)$ and also the determinant of the lattice J.

To the polynomials $v(x)$ and $w(x)$ we associate the rotation matrices V,W. These matrices represent the "good" (private) basis of the lattice. The "bad" (public) basis is the $HNF(J)$ - tuple (d, r). The positive aspect of Gentry-Halevi implementation is that the private basis can be represented by some odd coefficient w_i of $w(x)$.

B. Encryption and decryption

The encryption of a bit $b \in \{0, 1\}$ follows the GGH-type encryption. The message b is first encoded into some error vector $\overrightarrow{a} = b \cdot \overrightarrow{e_1} + 2 \cdot \overrightarrow{u}$, where coefficients of the vector $\overrightarrow{u}$ are generated randomly from set $\{-1, 0, 1\}$. The Euclidean norm of the vector $\overrightarrow{u}$ is controlled by a parameter q. The ciphertext is obtained by reduction of vector $\overrightarrow{a}$ by the public basis $HNF(J)$.

$$\overrightarrow{c} = (b \cdot \overrightarrow{e_1} + 2 \cdot \overrightarrow{u}) \bmod HNF(J)$$

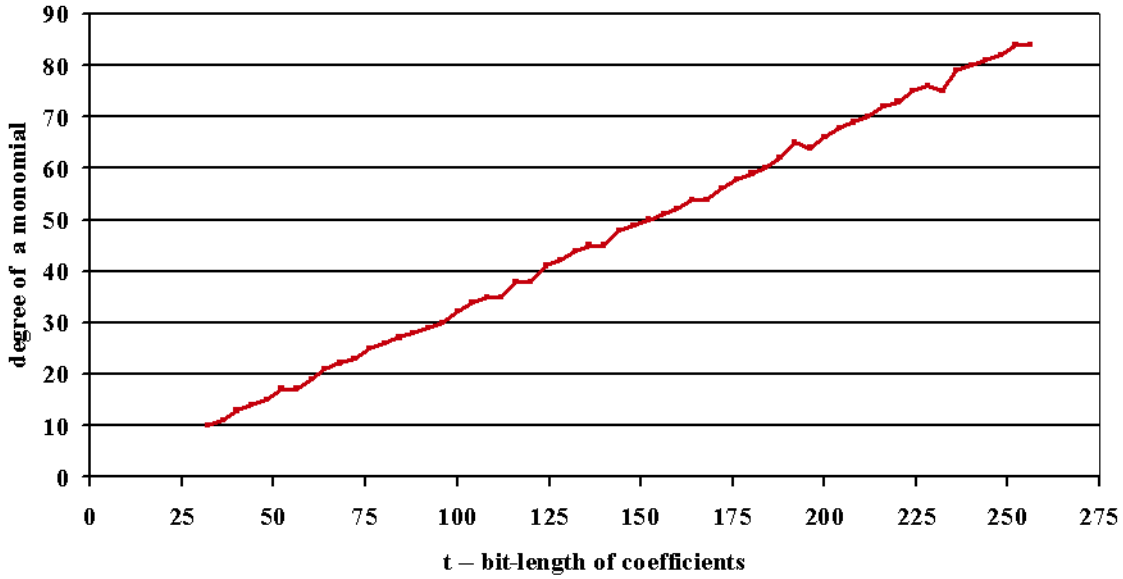

Fig. 1. The supported number of multiplications for N=128 and various t.

In [16] is shown that $\overrightarrow{c} = (c, 0, \ldots, 0)$, where c is computed simply as $[a(r)]_d$ and they also describe an efficient recursive algorithm for doing that.

Decryption of the ciphertext vector $\overrightarrow{c}$ has to be done in two steps: recovery of the error vector $\overrightarrow{a}$ followed by the extraction of the message bit. The original computation

$$\overrightarrow{a} = [\overrightarrow{c} \times W/d] \times V$$

is due to the special form of ciphertext (single integer) reduced to the optimized version:

$$b = ([w_i \cdot c]_d) \bmod 2$$

The homomorphic addition (resp. multiplication) procedure is a simple addition (resp. multiplication) of the two lattice vectors.

C. Parameters

The complete set of parameters is the triple (N, t, q), where $N \in \mathbf{N}$ denotes the dimension of underlying lattices and must be equal to some power of 2, $t \in \mathbf{N}$ denotes the bitsize of coefficients and $q \in (0, 1)$ specifies amount of "noise" introduced by encryption, namely an average noise vector will have $N \cdot q$ coefficients equal to zero and the rest will be $+1$ and -1 with equal probability.

D. Procedures

The somewhat homomorphic scheme is defined by five procedures *Keygen, Encrypt, Decrypt, Add, Mul*. Here we provide only a brief descriptions of them. For further information on the underlying algorithms and detailed implementation we refer the reader to the original paper [16].

$Keygen(N, t)$
– set $f(x) = x^N + 1$
– choose a polynomial $v(x)$ of degree $(N-1)$, with a t-bit coefficients, s.t. $v(x)$ and $f(x)$ have a single root r in common
– compute $w(x)$ s.t. $w(x)v(x) \equiv d \bmod f(x)$, where $d = resultant(f(x), v(x))$
– output $PK = (N, t, d, r)$ and $SK = (w_{i_0})$, where w_{i_0} is some odd coefficient of $w(x)$

$Encrypt(PK, b, q)$
– choose random $u(x) \in \mathbf{Z}[x]$ of degree $(N-1)$ where $u_i = \pm 1$ with probability $(1-q)$ and $u_i = 0$ with probability q
– set $c(x) = m + 2 \cdot u(x)$
– output $c = [c(r)]_d$

$Decrypt(SK, c)$
– $m = [c \cdot w_{i_0}]_d$
– output $m \bmod 2$

$Add(PK, c_1, c_2)$
– output $c = [c_1 + c_2]_d$

$Mul(PK, c_1, c_2)$
– output $c = [c_1 \cdot c_2]_d$

E. Security of the SHS

The security of the somewhat homomorphic scheme is based on the following *bounded distance decoding problem* (BDDP [24], [25], [3]): attacker has access to the basis of the lattice L

and a ciphertext vector $\overrightarrow{c}$. This vector is supposed to be close to some lattice point and we say that attacker is successful if he can recover the error vector.

IV. Experiments and Results

In this section we describe two groups of experiments. In the first group we made two experiments which repeated and extended the results from [16], namely we fixed the dimension $N = 128$ and the probability $q = 1 - 20/N$ in both experiments and calculated the supported number of multiplications and the largest supported degree of the SHS as a function of t. The parameter N is fixed because of Gentry's hypothesis that the obtained results are independent from the chosen dimension N [16]. The initial choice $q = 1 - 20/N$ implies that approximately 20 coefficients of the $c(x)$ are non-zero. We experiment with the parameter q in the second group of experiments.

V. Supported number of multiplications

First of all we examined the dependency of the supported number of multiplications on the parameter t – the bit-length of coefficients of $v(x)$.

A. Method

In each experiment we generated an instance of the cryptosystem with $N = 128$ and t from 32 to 256 with a step 4. We pre-generated a sufficient number (MAX) of PT-CT pairs. To analyze the number of multiplications that can be performed correctly we were multiplying i ciphertexts (for i from 2 to MAX) and then comparing the decrypted result to the corresponding product of plaintexts. The process stopped on the first number n_a that produced error and the supported number of multiplications was then $n_a - 1$. The algorithm is presented also in pseudo-code:

1) generate keys for $N = 128$ and t from 64 to 256 (with step 4)
2) generate sufficient number of PT-CT pairs
3) for increasing i: multiply i ciphertexts and test if they produce correct result after decryption
4) if i_k was the first to produce error, then $m_{max} \leftarrow (i_k - 1)$
5) repeat 30 times and output the minimal m_{max}

As the encryption is a randomized process, the experiment was repeated 30 times to get statistically more significant results. It is more that 2 times higher number than Gentry used in their computations, but our results still indicate some statistical deviations.

B. Results

The results are displayed on Fig. 1 (for the completeness we provide also full tables in the Appendix A) and only confirm the hypothesis that the homomorphic operation Mul multiplies the error introduced by encryption. The decryption of the scheme is correct if the error does not exceed the boundary 2^{t-1} and therefore the supported number of multiplications was expected to grow linearly in t.

The results on the Fig. 1 clearly show the linear dependence, but also a little randomness, because on value $t = 196$ the degree decreased to a value 64. This is due to a "lucky" choice of ciphertexts with the parameter $t = 192$, where we were able to correctly compute product of 65 ciphertexts.

VI. The largest supported degree

A. Method

The second experiment was focused on the evaluation of polynomials representing arbitrary functions on ciphertexts. We generated an instance of the cryptosystem with $N = 128$ and some t from 32 to 256. We then fixed m – a number of variables of a polynomial and pre-generated m random PT-CT pairs. Then we evaluated every elementary symmetric polynomial up to degree m on the ciphertexts and decrypted the resulting value. The largest supported degree for the parameter setting (N, t) is denoted by the number lsd_t, for which every tested elementary symmetric polynomial up to degree lsd_t on m variables was evaluated correctly on the ciphertexts (i.e. we evaluated the same polynomial on corresponding plaintexts and compared the values). The pseudo-code for this algorithm is as follows:

1) generate keys for $N = 128$ and t from 64 to 256 (with step 8)
2) for $m \in \{64, 80, 96\}$ do
3) generate m PT-CT pairs
4) for increasing d: compute the value of an elementary symmetric polynomial of degree d in m-ciphertexts; decrypt the result and compare the value with the same symmetric polynomial on plaintexts
5) if d_e was the first to produce error, then $lsd \leftarrow (d_e - 1)$
6) repeat 30 times and output the minimal lsd

For better comparison, the experiment was executed for three different number of variables $m = 64, 80$ and 96. Obtained results are displayed on Fig. 2 (the table with details is in Appendix A).

In contrary to the previous experiment, the expectations were not so straightforward. The elementary symmetric polynomials contain $\binom{m}{deg}$ monomials of degree deg. After the evaluation of the monomials, we expect each of them to have an error of size approximately c^{deg} for some (unknown) constant c. To obtain the correct result after the summation, the error for the lsd-degree polynomial should be less than 2^{t-1}, so we expect some kind of linear dependency of lsd from t.

B. Results

As the lsd of a polynomial is bounded by the number of variables m, the graph on Fig. 2 for lsd is constant after reaching the value m. The results for $m = 64$ also show a slight change of the slope at $lsd \approx 32$ – which is exactly the point where the $\binom{m}{deg}$ starts to decrease and the same can be observed for $m = 80$. We conclude that the lsd-value is not strictly linear in t (which would occur if the multiplication error would prevail over the addition), but depends also on the number of additions – in this case the ratio of $m : deg$.

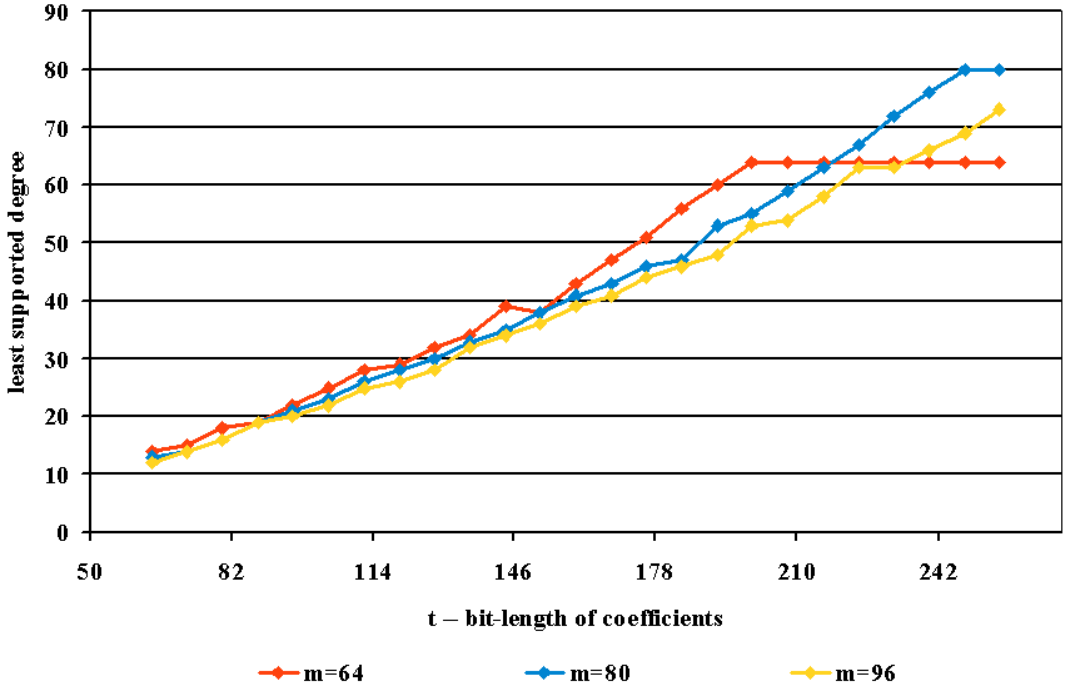

Fig. 2. The dependency of the largest supported degree on t for $m = 64, 80, 96$.

VII. Influence of parameter q on the SHS

The parameter q is used during the encryption procedure of the scheme. It represents the probability that a coefficient of the noise (error) vector is zero. Higher probability q results in smaller error and therefore in better effectiveness of the scheme. Smaller values of q would result into higher Euclidean norms of the error vectors and the scheme would support less operations.

The parameter also influences the security of the scheme. The analysis from [16] shows that probability of non-zero coefficient in error vector $1 - q$ needs to be high enough so that the complexity of birthday-type attacks $2^{(1-q)n} \cdot \binom{n}{qn}$ is higher than $2^{2\lambda}$, where λ is chosen security parameter (i.e.256 bits).

Another type of attack on the scheme is "guessing" the correct error bits. If the attacker is able to guess a relatively small set S of coefficients (e.g. 50 of them) that include all the coefficients of vector u, then he has to find the closest vector to ciphertext c in lattice defined by r^{i_k}, where $i_k \in S$. This is equivalent to the shortest vector problem in dimension 50. The problem is to "guess" the correct set S, which can be done with probability $(|S|/N)^{(1-q)N}$. For dimension $N = 2048$, $|S| = 200$ and parameter $q = 1 - 20/N$, the probability of success is $(200/2048)^{20} \approx 2^{72}$.

The recommended setting for q is $1 - Q/N$, where Q representing average number of non-zero coefficients is between 15 and 20.

A. Method

To get better idea how this parameter influences the effectiveness, we repeated the experiment with largest supported degree with various choices of q. We fixed the number of variables $m = 80$ and changed the probability q by ± 0.05 and ± 0.1 obtaining values from 0.74375 to 0.94375 with 33 to 7 non-zero coefficients in 128 bit vector.

	q	Q
q_1	0.74	33
q_2	0.8	26
q_3	0.84	20
q_4	0.9	14
q_5	0.94	7

TABLE I
Approximate values of q and Q.

B. Results

The results are displayed on Fig. 3 and in the Appendix A. As the Euclidean norm of the error vector is $2 \cdot \sqrt{Q}$ and the error grows exponentially with the number of multiplications performed, the largest supported degree is expected to grow exponentially with increasing values of q. A close look on the results in the Table IV support this hypothesis.

This results imply that increasing the norm of the error vector in the ciphertext (Q) will significantly reduce the effectiveness of the somewhat homomorphic scheme and also the fully homomorphic scheme that is based on it. This conclusion is quite different from the one from [16], where

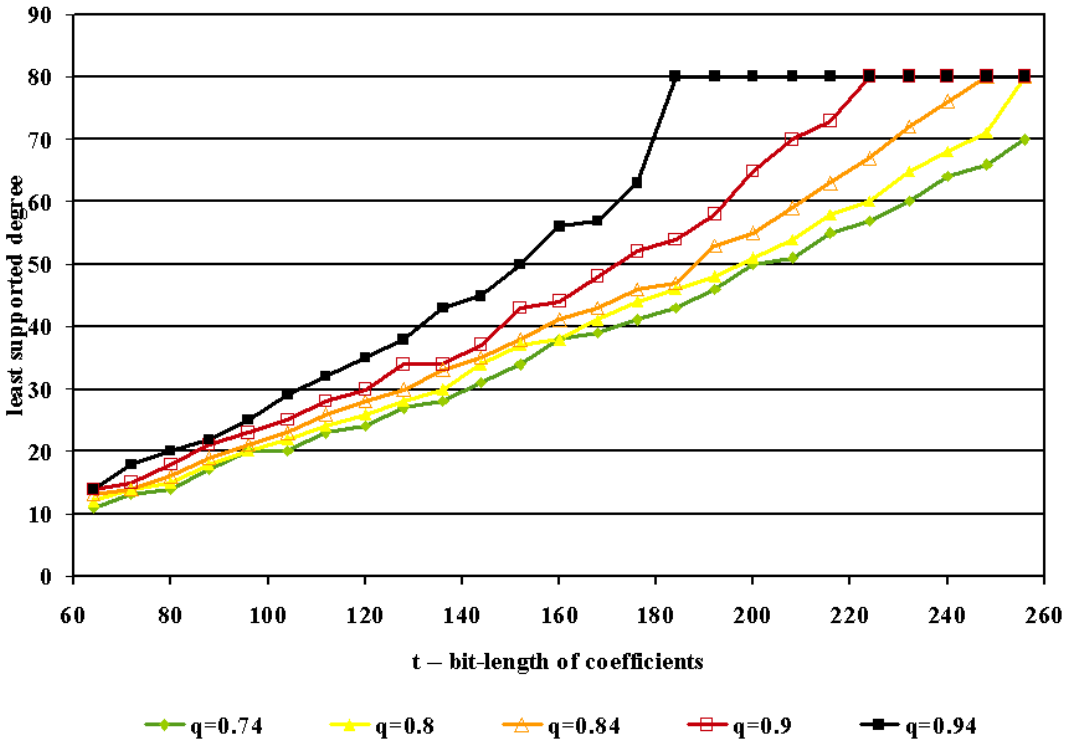

Fig. 3. The dependency of the largest supported degree on t for $m = 80$ and various q.

Gentry noted that *increasing the noise in the ciphertext will have only moderate effect on the performance numbers of our fully homomorphic scheme*. We think that Gentry simply omitted the effect of q on the supported number of operations.

C. Conclusions

The idea of constructing a fully homomorphic scheme based on discrete lattices is quite new and thorough cryptanalysis of this scheme has to be done. We tried to show the importance of the "balanced choice" of q and its impact on overall effectiveness of the somewhat homomorphic scheme.

REFERENCES

[1] A. Adelsbach, S. Katzenbeisser, A. Sadeghi, *Cryptography Meets Watermarking: Detecting Watermarks with Minimal or Zero Knowledge Disclosure*, In Proceedins of the European Signal Processing Conference 2002, Toulouse, France, 2002.

[2] N. Ahituv, Y. Lapid, S. Neumann, *Processing encrypted data*, Communications of the ACM, vol.30, no.9, 1987, pp. 770–780.

[3] M. Ajtai, C. Dwork, *A public key cryptosystems with worst-case / average-case equivalence*, In proceeding of STOC'97, 1997, pp. 284–293.

[4] F. Armknecht, A. Sadeghi: *A new approach for algebraically homomorphic encryption*, Cryptology ePrint Archive, Report 2008/422, 2008. URL: http://eprint.iacr.org/2008/422 [21.4.2011]

[5] F. Bao: *Cryptanalysis of a provable secure additive and multiplicative privacy homomorphism*, International Workshop on Coding and Cryptography (WCC'03), Versailles, France, 2003, pp. 43–49.

[6] D. Boneh, J. Goh, K. Nissim, *Evaluating 2-DNF formulas on ciphertexts*, Proceedings of Theory of Cryptography Conference, 2005, vol.3378 of LNCS, Springer-Verlag, pp. 325–342.

[7] G. Chunsheng, *New Fully Homomorphic Encryption over the Integers*, Cryptology ePrint Archive, Report 2011/118, 2011. URL: http://eprint.iacr.org/2011/118 [21.4.2011]

[8] R. Cramer, I. Damgård, *Zero-Knowledge for Finite Field Arithmetic. Or: Can Zero-Knowledge be for Free?*, In Proceedings of CRYPTO'98, vol. 1462 of LNCS, Springer-Verlag, 1998, pp.424–441.

[9] M. van Dijk, C. Gentry, S. Halevi, V. Vaikuntanathan, *Fully Homomorphic Encryption over the Integers*, Advances in Cryptology EUROCRYPT 2010, 2010, Volume 6110/2010 of LNCS, pp. 24–43.

[10] C. Gentry, *Fully homomorphic encryption using ideal lattices*, Proceedings of the 41st annual ACM symposium on Theory of computing (STOC'09), Bethesda, USA, 2009, pp. 169–178.

[11] C. Gentry, A fully homomorphic encryption scheme, *Dissertation Thesis, Standford University*, september 2009. URL: http://crypto.stanford.edu/craig/ [21.4.2011]

[12] J. D. Ferrer: *A new privacy homomorphism and applications*, Information Processing Letters, vol. 60, no.5, 1996, pp. 277–282.

[13] J. D. Ferrer: *A provably secure additive and multiplicative privacy homomorphism*, in Proceedings of the 5th International Conference on Information Security (ISC'02), vol. 2433 LNCS, Springer-Verlag, 2002, pp. 471–483.

[14] C. Fontaine, F. Galand: *A survey of homomorphic encryption for nonspecialists*, EURASIP Journal on Information Security, vol. 2007.

[15] O. Goldreich, S. Goldwasser, S. Halevi, *Public key cryptosystems from lattice reduction problems*, In Advances in Cryptology - CRYPTO'97, vol. 1294 of LNCS, Springer-Verlag, 1997, pp. 112–131.

[16] C. Gentry, S. Halevi, *Implementing Gentry's Fully-Homomorphic Encryption Scheme*, Cryptology ePrint Archive, Report 2010/520, 2010. URL: http://eprint.iacr.org/2010/520 [21.4.2011]

[17] O. Grošek and P. Zajac, *Efficient Selection of the AES-Class MixColumns Parameters*, WSEAS Transactions on Information Science and Applications, Vol. 4, Iss. 4, 2007, ISSN 1790-0832, pp. 663–668.

[18] O. Grošek and P. Zajac, *Graphs Connected with Block Ciphers*, WSEAS Transactions on Information Science and Applications, Vol. 3, Iss. 2, 2006, ISSN 1790-0832, pp. 439–443.

[19] O. Grošek and P. Zajac, *Searching for a Different AES-Class MixColumns Operation*, Proceedings of the WSEAS Conference : 6th International Conference on Applied Computer Science, WSEAS Press, 2006, ISBN 960-8457-57-2, pp. 307–310.

[20] O. Grošek and P. Zajac, *A remark to minimal graphs connected with block ciphers*, Proceedings of 4th WSEAS International Conference on

Information Security, Communications and Computers (ISCOCO 2005), December 16-18, 2005, Tenerife, pp. 78–82.

[21] P. Fouque, G. Poupard, J. Stern, *Sharing decryption in the context of voting or lotteris*, Financial Cryptography 2000, vol. 1962 of LNCS, Springer Verlag, 2000.

[22] H. Lipmaa, *Verifiable homomorphic oblivious transfer and private equality test*, Advances in Cryptology - Asiacrypt 2003, vol. 2894 of LNCS, Springer-Verlag, 2003.

[23] C. Loftus, A. May, N. P. Smart, F. Vercauteren, *On CCA-Secure Fully Homomorphic Encryption*, Cryptology ePrint Archive, Report 2010/560, 2010.
URL: http://eprint.iacr.org/2010/560 [21.4.2011]

[24] V. Lyubashevsky, D. Micciancio, *On Bounded Distance Decoding, Unique Shortest Vectors, and the Minimum Distance Problem*, Advances in Cryptology - CRYPTO'09, LNCS, Springer-Verlag, 2009.

[25] Y.-K. Liu, V. Lyubashevsky, D. Micciancio, *On bounded distance decoding for general lattices*, In the proceedings of APPROX-RANDOM, 2006, pp. 450–461.

[26] C. Melchor, P. Gaborit, J. Herranz, *Additively homomorphic encryption with t-operand multiplications*, Cryptology ePrint Archive, Report 2008/378, 2008.
URL: http://eprint.iacr.org/2008/378 [21.4.2011]

[27] D. Micciancio, *Improving lattice based cryptosystems using the hermite normal form.*, CaLC 2001, vol. 2146 of LNCS, Springer-Verlag, 2001, pp. 126–145.

[28] S. Murphy, M.J.B. Robshaw, *Essential algebraic structure within the AES*, Advances in Cryptology - CRYPTO 2002, vol. 2442 of LNCS, Springer-Verlag, 2002.

[29] B. Pfitzmann, M. Waidner, *Anonymous fingerprinting*, Advances in Cryptology - EUROCRYPT 1997, vol. 1233 of LNCS, Springer-Verlag, 1997, pp. 88–102.

[30] P. Pocatilu, F. Alecu, M. Vetrici, *Measuring the Efficiency of Cloud Computing for E-learning Systems*, WSEAS Transactions on Computers, vol. 9, Iss. 1, 2010, ISSN 1109-2750, pp. 42–51.

[31] P. Kragelj *Cyber Security in a Cloud with Insight on the Slovenian Situation*, Proceedings of the European Computing Conference, ISBN 978-960-474-297-4, pp. 297–300.

[32] D.K. Rappe: *Algebraishe homomorphe Kryptosysteme*, Diploma Thesis, University of Dortmund, Dortmund, Germany, 2000.
URL: www.rappe.de/doerte/Diplomarbeit.pdf [21.4.2011]

[33] D.K. Rappe, *Homomorphic cryptosystems and their applications*, Dissertation Thesis, University of Dortmund, Germany, 2004.
URL: www.rappe.de/doerte/Diss.pdf [21.4.2011]

[34] R. Rivest, L. Adleman, M. Dertouzos, *On data banks and privacy homomorphisms*, Foundations of Secure Computation, Academic Press, 1978, pp. 169–177.

[35] N.P. Smart, F. Vercauteren, *Fully Homomorphic Encryption with Relatively Small Key and Ciphertext Sizes*, Cryptology ePrint Archive, Report 2009/571, 2009.
URL: http://eprint.iacr.org/2009/571 [21.4.2011]

[36] D. Wagner: *Cryptanalysis of an algebraic privacy homomorphism*, Proceedings of the 5th International Conference on Information Security (ISC'03), vol. 2851 LNCS, Bristol, UK, 2003.
URL: www.cs.berkeley.edu/~daw/papers/ (revised version) [21.4.2011]

APPENDIX

Here we provide the detailed data from experiments in the form of tables.

t	Monomial degree	t	Monomial degree
32	10	148	49
36	11	152	50
40	13	156	51
44	14	160	52
48	15	164	54
52	17	168	54
56	17	172	56
60	19	176	58
64	21	180	59
68	22	184	60
72	23	188	62
76	25	192	65
80	26	196	64
84	27	200	66
88	28	204	68
92	29	208	69
96	30	212	70
100	32	216	72
104	34	220	73
108	35	224	75
112	35	228	76
116	38	232	75
120	38	236	79
124	41	240	80
128	42	244	81
132	44	248	82
136	45	252	84
140	45	256	84
144	48		

TABLE II
Experiment 1: Dependency of supported number of multiplications on parameter t.

	The largest supported degree		
t	$m = 64$	$m = 80$	$m = 96$
64	14	13	12
72	15	14	14
80	18	16	16
88	19	19	19
96	22	21	20
104	25	23	22
112	28	26	25
120	29	28	26
128	32	30	28
136	34	33	32
144	39	35	34
152	38	38	36
160	43	41	39
168	47	43	41
176	51	46	44
184	56	47	46
192	60	53	48
200	64	55	53
208	64	59	54
216	64	63	58
224	64	67	63
232	64	72	63
240	64	76	66
248	64	80	69
256	64	80	73

TABLE III

Experiment 2: The largest supported degree for $m = 64, 80$ *and* 96.

	The largest supported degree				
t	$q = 0.74$	$q = 0.8$	$q = 0.84$	$q = 0.9$	$q = 0.94$
64	11	12	13	14	14
72	13	14	14	15	18
80	14	15	16	18	20
88	17	18	19	21	22
96	20	20	21	23	25
104	20	22	23	25	29
112	23	24	26	28	32
120	24	26	28	30	35
128	27	28	30	34	38
136	28	30	33	34	43
144	31	34	35	37	45
152	34	37	38	43	50
160	38	38	41	44	56
168	39	41	43	48	57
176	41	44	46	52	63
184	43	46	47	54	80
192	46	48	53	58	80
200	50	51	55	65	80
208	51	54	59	70	80
216	55	58	63	73	80
224	57	60	67	80	80
232	60	65	72	80	80
240	64	68	76	80	80
248	66	71	80	80	80
256	70	80	80	80	80

TABLE IV

Experiment 3: The largest supported degree for $m = 80$ *and various choices of* q.