

Some necessary clarifications about the Chords' Problem and the Partial Digest Problem

A. Daurat ^a, Y. Gérard ^b and M. Nivat ^c

^a*LSIIT, Pôle API, Boulevard Sébastien Brant, 67400 Illkirch, France*

^b*LLAIC, IUT, Ensemble Universitaire des Cézeaux, 63172 Aubière, France*

^c*LIAFA, 2 place Jussieu 75251 Paris Cedex 05, France*

Abstract

We state in previous paper [3] that the chords' problem can be solved in polynomial time. This result is however ambiguous and some people have been abused because the encoding of the data has not been given. The correctness of the result requires to specify the encoding of the data that we have used and to highlight the difference with the usual encoding implicitly considered in Partial Digest Problem.

1 Introduction

We consider the inverse problem of computational geometry known as “Turnpike” problem or as “Partial Digest Problem”. We have also called it in [3] the “chords' problem”. We state in this paper that the problem can be solved in polynomial time. This result of [3] is ambiguous because the encoding of the data has not been given in the paper while several ones can be considered. The main purpose of this text is to make clear the encoding that we have considered. It is all the more important that we have not used the encoding which is usually implicit in the framework of “Partial Digest Problem”. Thus the “polynomial” algorithm that we have presented in [3] does not solve Partial Digest Problem as usually asked. As far as we know, the question of the complexity of Partial Digest Problem is still open. Notice that some variants of the problem are known as NP-hard [2][11].

Email addresses: daurat@dpt-info.u-strasbg.fr (A. Daurat),
gerard@llaic.u-clermont1.fr (Y. Gérard).

2 Problems and encodings

We use a vector $a \in \{0, 1\}^{\{0, \dots, l\}}$ for representing any subset of $\{0, \dots, l\}$ and a vector $b \in \mathbb{N}^{\{0, \dots, l\}}$ for representing a multiset of $\{0, \dots, l\}$. We consider the chords' operator C defined by $C(a) = (b_k)_{0 \leq k \leq l}$ where $b_k = \text{card}\{i : a_i = a_{i+k} = 1\}$.

We can consider three problems:

Problem 1 [PEXI]

Input: $(b_k)_{0 \leq k \leq l}$

Output: Answer to the question "does there exist a such that $C(a) = b$?".

Problem 2 [PONE]

Input: $(b_k)_{0 \leq k \leq l}$

Output: one a such that $C(a) = b$ if it exists.

Problem 3 [PALL]

Input: $(b_k)_{0 \leq k \leq l}$

Output: all the a such that $C(a) = b$.

Four encodings of the input can be considered. We illustrate each encoding on the instance $b = (4, 2, 1, 0, 0, 0, 0, 0, 1, 1, 1) \in \mathbb{N}^{0, \dots, 10}$ corresponding to the multiset $\{0, 0, 0, 0, 1, 1, 2, 8, 9, 10\}$ and we use the values $B = \max\{b_i\}$, $M = \text{card}\{i : b_i > 0\}$ to bound the size of the data. Moreover we suppose, without losing any generality, that $b_l > 0$.

- (E1) The integers b_i are all-encoded in unary with a special symbol separating them. As example, the instance b is encoded by $[, , , 1, 1, 11, 11111111, 11111111, 11111111111]$. The corresponding size S_1 satisfies: $B + l + M - 2 \leq S_1(b) \leq MB + n$. So a numerical function of b is polynomial if and only if it is bounded by a polynomial in B, l, M .
- (E2) The integers b_i are all-encoded in binary with a special symbol separating them. The instance b is encoded by $[100, 10, 1, 0, 0, 0, 0, 1, 1, 1]$. The size S_2 satisfies: $\log_2(B) + l + M - 2 \leq S_2(b) \leq M \log_2(B) + 2l$. So polynomial in S_2 means polynomial in $l, M, \log B$.
- (E3) The integers b_i are encoded by the list of the indexes i where i is repeated exactly b_i times and encoded in binary. The instance b is encoded by $[0, 0, 0, 0, 1, 1, 10, 1000, 1001, 1010]$. The size S_3 satisfies: $2(M - 1) + \log_2(l) + B \leq S_3(b) \leq MB(\log_2(l) + 2)$. So polynomial in S_3 means polynomial in $M, B, \log l$.
- (E4) The integers b_i are encoded by the list of the (i, b_i) with $b_i > 0$ (encoded in binary). The instance b is encoded by $[0, 100|1, 10|10, 1|1000, 1|1001, 1|1010, 1]$. The size S_4 satisfies: $2(M - 1) + \log_2(l) + \log_2(B) \leq S_4(b) \leq M(\log_2(B) + \log_2(l) + 3)$. So polynomial in S_4 means polynomial in $M, \log B, \log l$.

Notice that in our paper [3], we mean pseudo-polynomial for “polynomial in S_1 ” and polynomial for “polynomial in S_2 ”.

3 Known Results

All the algorithms discussed here are based on the factorization of the polynomial $P(X) = X^l(b_0 + \sum_{i=1}^l (b_i(X^i + X^{-i}))$ which can be done in $O(l^{9+\varepsilon} + l^{7+\varepsilon}(\log B)^{2+\varepsilon})$ -time (Theorem 3.6 of [9]), so in polynomial time in S_2 .

In [8] it is shown that the half of the number of non-symmetric factors of $P(X)$ is bounded by $C(l, B) = \frac{1}{4 \ln \lambda} \ln(((2l + 1))B^2)$ where $\lambda \approx 1.325$ is the real root of $X^3 - X - 1$ (equation (11)). As $2^{C(l, B)}$ is polynomial in S_1 , this bound permits to show that PALL can be solved in a polynomial time in S_1 . So PEXI, PONE, PALL are polynomial in S_1 . (see [11])

In Section 5 of [3] we just notice that there is a simple criterion on the factors of the polynomial $P(X)$ and which complexity is not exponential in the number of non-symmetric factors. It can be deduced that the complexity of PEXI and PALL are bounded by the complexity of the factorization of the primitive polynomials. So PEXI, PONE are polynomial in S_2 .

But in fact the half of the number of non-symmetric factors of $P(X)$ can be bounded by $\frac{1}{2 \ln \lambda} \ln(N)$ ([11, section 2.1,3.1]) where $N = \text{card}\{i : a_i \neq 0\}$ (otherwise there is no solution). As $N \leq M$, the size of the output of PALL is polynomial in S_4 and the algorithm for PALL described in [8,11] is polynomial in S_2 . So PEXI, PONE, PALL are polynomial in S_2 .

So in fact, Section 5 of [3] did not bring anything new, but when we wrote it, we did not realize the result of the previous paragraph.

As far as we know, the complexities with encodings E_3 and E_4 are open. It seems that they are the encodings of Partial Digest Problem. So ***the complexity of the Partial Digest Problem remains unknown.***

To solve these open problems, factorization of polynomials cannot a priori be used. First because the size of the factorized form can be exponential in the size of the original polynomial, and second because just testing irreducibility of sparse polynomial looks to be a hard problem : a recent paper ([6]) claims that irreducibility of the rational *bivariate* polynomials is NP-hard under randomized reduction when an encoding similar to E_4 is used.

4 Example of 2-dimensional convex lattice sets having the same covariogram

This section contains another independent remark about the chords' problem.

The authors of [4] have noticed that we have answered in [3] the question of the existence of different convex lattice sets (different up to translation and central symmetry) with the same covariogram without providing any example.

In fact we found an example by a systematic search on all the convex sets included in a 4×4 square. This example (Fig 1) was symmetric towards $\frac{\pi}{4}$ direction and thus congruent according to a group of rigid transformations. The examples given in [4] are not congruent up to any rigid transformation and thus better than ours.

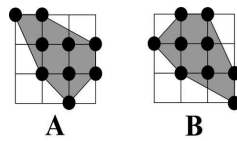


Fig. 1. A and B are the only convex sets (up to a symmetry) included in a 4×4 square having the same chords set with multiplicities, and such that $A \neq -B$.

We must notice that now, it is known that such an example does not exist for “continuous” convex sets which border is regular enough (see [1]).

5 Conclusion

We expect that this addendum will remedy to the lack of precision and the omissions of [3]. In particular we hope that we have clearly stated that we did *not* solve the problem usually known as Partial Digest Problem.

6 Acknowledgments

They are due to Steven Skiena for having called our attention on the ambiguity of [3] and to Paul Zimmermann and Guillaume Hanrot for information about factorization algorithms.

References

- [1] G. Bianchi, F. Segala, A. Volčič, The solution of the covariogram problem for plane \mathcal{C}_+^2 convex bodies, *J. Differential Geom.* 60 (2) (2002) 177–198.
- [2] M. Cieliebak, S. Eidenbenz, P. Penna, Noisy data make the partial digest problem NP-hard, in: *Proc. of WABI 2003*, vol. 2812 of *Lecture Notes in Comp. Sci.*, 2003, pp. 111–123.
- [3] A. Daurat, Y. Gérard, M. Nivat, The chords’ problem, *Theoret. Comput. Sci.* 282 (2) (2002) 319–336.
- [4] R. J. Gardner, P. Gronchi, C. Zong, Sums, projections, and sections of lattice sets, and the discrete covariogram, to appear in *Discrete Comput. Geom.*
- [5] N. C. Jones, P. A. Pevzner, *An Introduction to Bioinformatics Algorithms*, MIT Press, Cambridge, 2004.
- [6] E. Kaltofen, P. Koiran, On the complexity of factoring bivariate supersparse (lacunary) polynomials, <http://perso.ens-lyon.fr/pascal.koiran>.
- [7] P. Lemke, S. S. Skiena, W. D. Smith, Reconstructing sets from interpoint distances, in: *Discrete and computational geometry: The Goodman-Pollack Festschrift*, vol. 25 of *Algorithms Combin.*, Springer, Berlin, 2003, pp. 597–631, full version of [11].
- [8] P. Lemke, M. Werman, On the complexity of inverting the autocorrelation function of a finite integer sequence, and the problem of locating points on a line, given the $\binom{n}{2}$ unlabelled distances between them, *Tech. rep.*, IMA, Minneapolis (1988).
- [9] A. K. Lenstra, H. W. Lenstra, L. Lovász, Factorizing polynomials with rational coefficients, *Math. Ann.* 261 (1982) 515–534.
- [10] P. A. Pevzner, *Computational Molecular Biology : an Algorithmic Approach*, MIT Press, Cambridge, 2000.
- [11] S. S. Skiena, W. D. Smith, P. Lemke, Reconstructing sets from interpoint distance, in: *Proc. Sixth ACM Symp. Computational Geometry*, 1990, pp. 332–339.