Available online at www.sciencedirect.com

SCIENCE DIRECT®

Journal of Discrete Algorithms 4 (2006) 239–254

JOURNAL OF DISCRETE ALGORITHMS

www.elsevier.com/locate/jda

The complexity of minimum difference cover ☆

Carlo Mereghetti *, Beatrice Palano

Dipartimento di Scienze dell'Informazione, Università degli Studi di Milano, via Comelico 39, 20135 Milano, Italy

Available online 13 April 2005

Abstract

The complexity of searching *minimum difference covers*, both in $\mathbf{Z}^+$ and in $\mathbf{Z}_n$, is studied. We prove that these two optimization problems are NP-hard. To obtain this result, we characterize those sets—called extrema—having themselves plus zero as minimum difference cover. Such a combinatorial characterization enables us to show that testing whether sets are not extrema, both in $\mathbf{Z}^+$ and in $\mathbf{Z}_n$, is NP-complete. However, for these two decision problems we exhibit pseudo-polynomial time algorithms.

Keywords: Difference cover; NP-completeness; NP-hardness

1. Introduction

In this work, we study the complexity of computing *difference covers* for sets of integers. The problem was originally stated as follows [6]: find a subset Δ of $\mathbf{Z}_n$ such that any element of $\mathbf{Z}_n$ can be obtained as difference $\bmod n$ of two elements in Δ. The set Δ is called *difference cover*. Several variants of this problem have been considered in the literature. The most relevant are discussed in Section 3.

The problem of reproducing sets of integers by differences has a lot of connections with topics in combinatorics and computational geometry such as Golomb's rulers [1,4],

☆ Partially supported by M.I.U.R. COFIN, under the projects "Linguaggi formali e automi: metodi, modelli e applicazioni", and "FIRB: Complessità descrizionale di automi e strutture correlate".

* Corresponding author.

E-mail addresses: mereghetti@dsi.unimi.it (C. Mereghetti), palano@dsi.unimi.it (B. Palano).

1570-8667/$ – see front matter
doi:10.1016/j.jda.2005.03.004

chords' multisets [8], interpoint distances [13], etc. Yet, it shows up in many applications, from communication to cryptography, networking, text compression, etc. (see [7], for a survey).

It should be stressed that in all these and other applications, it turns out to be particularly important to *construct difference covers of small cardinality*. For instance, in [6,14] the construction of feasible concurrent systems having certain mutual exclusion properties (quorum systems) is related to the computation of small-size difference covers. More recently, the relevance of difference cover within the realm of quantum computing [11] has been pointed out. In particular, in [3,16] some algorithms for the construction small-size quantum finite automata are presented that rely on the ability of generating small-size difference covers.

This naturally leads to investigate the complexity of *searching the minimum difference cover* (*i.e., a difference cover with the smallest cardinality*) *for a given subset of* $\mathbf{Z}_n$. We call MinDCmod this optimization problem, presented in Section 3.1. We also introduce, in Section 3.2, a slight variant of MinDCmod called MinDC, where we ask for *minimum difference covers for subsets of* $\mathbf{Z}^+$. In this latter case, we use simple and not modular differences.

We study in parallel the complexity of MinDCmod and MinDC by considering a closely related decision problem on certain sets called extrema: a set A of nonnegative integers is an *extremum* whenever its minimum difference cover is the trivial one consisting of $A \cup \{0\}$. In Section 4, we provide some combinatorial characterizations of extrema which enable us to show that *deciding whether a given set is not an extremum, both in* $\mathbf{Z}_n$ *and in* $\mathbf{Z}^+$*, can be done in pseudo-polynomial time*. This is probably the best running time we can achieve since, in Section 5, we show that *testing non-extremity is* NP-*complete*. By this latter completeness result, we obtain, in Section 6, the NP-*hardness of both* MinDC *and* MinDCmod. Hence, we can hardly expect that efficient algorithms for them will ever be designed. For the sake of completeness, we also exhibit a Turing-reduction from MinDC to MinDCmod. This reduction leads us to consider the restriction of MinDCmod where input instances are the whole sets $\mathbf{Z}_n$. We prove that the related decision problem belongs to NP but it is not NP-complete, unless P = NP.

2. Preliminaries

We quickly present basic definitions and results used throughout the paper. We denote by $\mathbf{Z}^+$ the set of positive integers, and by $\mathbf{Z}_n$ the set $\{0, 1, \ldots, n-1\}$. Given $x \in \mathbf{Z}$, we denote by $|x|$ its absolute value. Given a finite set $Y = \{y_1, y_2, \ldots, y_m\} \subset \mathbf{Z}$, we denote by $\max_i y_i$ its maximum value. Given a set S, we denote by $|S|$ its cardinality.

We recall some basics of graph theory. More details can be found, e.g., in [2]. A *digraph* (directed graph) is a pair $\mathcal{G} = (V, E)$, where V is the set of vertices and the set of ordered pairs $E \subseteq V \times V$ is the set of arcs. Given $v, w \in V$, a *chain* from v to w is a sequence $\gamma = v_0, v_1, \ldots, v_n$ of vertices where $v_0 = v$, $v_n = w$ and $(v_i, v_{i+1}) \in E$ or $(v_{i+1}, v_i) \in E$, for $0 \leqslant i < n$. The *length* of γ is n, i.e., the number of arcs involved; γ is an *elementary chain* if it does not encounter the same vertex twice; γ is an *elementary cycle* if it is an elementary chain, except that $v_0 = v_n$. The digraph $\mathcal{G}$ is *weakly connected* whenever any two vertices are joined by a chain. A *weakly connected component* of $\mathcal{G}$ is a weakly connected subgraph $(V', (V' \times V') \cap E)$, with $V' \subseteq V$, such that any other subgraph $(W, (W \times W) \cap E)$ of $\mathcal{G}$ satisfying $V' \subset W \subseteq V$ is not weakly connected.

Our digraph $\mathcal{G}$ can be *weighted* by associating a weight, a positive integer in this paper, with each arc. We denote by $\omega(v, v')$ the weight of the arc $(v, v') \in E$. The *weight of the chain* $\gamma = v_0, v_1, \ldots, v_n$ joining the vertex $v = v_0$ to the vertex $w = v_n$ is given by

$$\omega(\gamma) = \sum_{\{v_i, v_{i+1} \in \gamma:\ (v_i, v_{i+1}) \in E\}} \omega(v_i, v_{i+1}) - \sum_{\{v_i, v_{i+1} \in \gamma:\ (v_{i+1}, v_i) \in E\}} \omega(v_{i+1}, v_i).$$

We assume some familiarity with the main concepts in structural complexity, and refer the reader to, e.g., [9] for a detailed exposition. The class (NP) P consists of those decision problems solvable in polynomial time by (non)deterministic Turing machines. A *polynomial time many-one reduction* from a decision problem Π to a decision problem Π' is a deterministic polynomial time computable function f from the set of instances of Π to the set of instances of Π' such that any instance I of Π has a positive answer if and only if $f(I)$ has a positive answer. Formally, we write $\Pi \leqslant_p \Pi'$ if there exists a polynomial time reduction from Π to Π', and simply say "Π reduces to Π'". The decision problem Π is NP-complete whenever $\Pi \in \mathrm{NP}$ and $\Pi' \leqslant_p \Pi$, for every $\Pi' \in \mathrm{NP}$. It is well-know that an NP-complete problem admits a deterministic polynomial time algorithm if and only if $\mathrm{P} = \mathrm{NP}$, a hardly believed event.

Some NP-complete problems have solution algorithms with the following property: if certain bounds were imposed in advanced on the size of objects (e.g., numbers, sets cardinality) contained in input instances, then these algorithms would work in deterministic polynomial time for the restricted problem. Algorithms of this type are called *pseudo-polynomial time algorithms*. In many practical applications, such bounds on input instances are actually satisfied. Thus, the possibility of finding a pseudo-polynomial time algorithm for NP-complete problems can be well worth investigating.

To study the complexity of problems that are not decision problems, it is useful to introduce the notion of NP-hardness. Roughly speaking, a problem is NP-hard if the existence of a deterministic polynomial time algorithm for its solution would imply $\mathrm{P} = \mathrm{NP}$. More formally, we need the notion of *polynomial time Turing-reduction* between problems whose precise statement can be checked, e.g., in [9, Chapter 5]. Intuitively, we can say that there exists a polynomial time Turing-reduction from a problem Π to a problem Π' whenever there exists a deterministic algorithm A that solves Π by using an hypothetical subroutine S for solving Π' (an oracle for Π') such that, if S were a polynomial time deterministic algorithm for Π', then A would be a polynomial time deterministic algorithm for Π. Formally, we write $\Pi \leqslant_T \Pi'$ if there exists a polynomial time Turing-reduction from Π to Π', and simply say "Π Turing-reduces to Π'". The problem Π is NP-hard whenever $\Pi' \leqslant_T \Pi$, for every $\Pi' \in \mathrm{NP}$.

In this work, we will prove the NP-completeness of some decision problems and the NP-hardness of related optimization problems, thus stating that, very likely, they do not admit efficient algorithms.

3. Difference covers

The problem of reproducing by differences sets of integers has been often considered in the literature. In [18], the following problem is stated: given $n \geqslant 0$, find $\Delta \subseteq \mathbf{Z}_n$ such that every element in $\mathbf{Z}_n$ is obtained *exactly once* as difference modulo n of two integers in Δ. The set Δ is called *difference set*. This problem has well-known relations with several combinatorial topics. In particular, by using finite projective plane theory, the following result is proved:

Theorem 3.1. [18] *For any $n = q^2 + q + 1$, with q prime power, there exists a difference set for $\mathbf{Z}_n$ of cardinality $q + 1$.*

Since not for all $n \geqslant 0$ a difference set for $\mathbf{Z}_n$ exists, a relaxation of the above problem is studied in [6], where each element of $\mathbf{Z}_n$ must be obtained *at least* once from $\Delta \subseteq \mathbf{Z}_n$. In this case, the set Δ is called *difference cover*. By using a result in [19], it is shown that

Theorem 3.2. [6, Theorem 2.4] *For any $n \geqslant 0$, there exists a difference cover for $\mathbf{Z}_n$ of cardinality at most $\sqrt{1.5n} + 6$.*

The problem of constructing difference covers shows up in many areas such as text compression, code design, network theory, concurrent systems design, cryptography (see, e.g., [5,7]).

A similar and well-studied problem on differences concerns the construction of *Golomb's rulers* [1,4]. A Golomb's ruler is a set $B \subset \mathbf{N}$ such that, for each pair $a, b \in B$ with $a > b$, the difference $a - b$ cannot be obtained from any other pair in B. Even Golomb's rulers find a lot of applications in several areas such as radio astronomy, X-ray crystallography, circuit layout, code design (see, e.g., [12,17]).

From a computational point of view, some sets for which Golomb's rulers and difference sets can be efficiently constructed are singled out in [3].

In what follows, we investigate some natural generalizations of the above problems on differences [3,16], studying the complexity of related optimization problems.

3.1. $\mathbf{Z}_n$-*difference cover*

Let us first generalize the notion of difference cover by studying the reconstruction by differences of *subsets of* $\mathbf{Z}_n$. Formally, we state that

Definition 3.1. The set $\Delta \subseteq \mathbf{Z}_n$ is a $\mathbf{Z}_n$-*difference cover* for the set $X \subseteq \mathbf{Z}_n$ if, for each $x \in X$, there exist two elements $a, b \in \Delta$ such that $x = (a - b) \bmod n$.

By Theorem 3.2, there exists a $\mathbf{Z}_n$-difference cover of size $\sqrt{1.5n} + 6$ for any $X \subseteq \mathbf{Z}_n$: clearly, a difference cover for the whole $\mathbf{Z}_n$ is also a $\mathbf{Z}_n$-difference cover for any given subset of $\mathbf{Z}_n$. However, we may be interested in finding a $\mathbf{Z}_n$-difference cover for X with the *smallest* possible cardinality. This naturally leads to the following optimization problem:

MINIMUM $\mathbf{Z}_n$-DIFFERENCE COVER (MinDCmod)
INPUT: $n \in \mathbf{Z}^+$, $X \subseteq \mathbf{Z}_n \setminus \{0\}$
OUTPUT: $\Delta = A \cup \{0\}$, with $A \subset \mathbf{Z}_n$, such that Δ is a $\mathbf{Z}_n$-difference cover for X
MEASURE: Cardinality of the $\mathbf{Z}_n$-difference cover, i.e., $|\Delta|$

For technical reasons, we assume that the input subsets for MinDCmod do not contain 0. This does not represent a restriction since any nonempty $\mathbf{Z}_n$-difference cover Δ clearly generates 0 as $d - d$, for $d \in \Delta$. Instead, we require that $\mathbf{Z}_n$-difference covers given as output must contain 0. Even this can be assumed without loss of generality, by considering the following "translation" lemma:

Proposition 3.1. *Let Δ be a $\mathbf{Z}_n$-difference cover for $X \subset \mathbf{Z}_n$, and let a fixed $a \in \mathbf{Z}_n$. Then the set $\Delta' = \{(d - a) \bmod n\colon d \in \Delta\}$ is a $\mathbf{Z}_n$-difference cover for X as well.*

Proof. In fact, if $x \in X$ is obtained as $x = (d - d') \bmod n$, for $d, d' \in \Delta$, then we can also write $x = ((d - a) - (d' - a)) \bmod n$, with $(d - a), (d' - a) \in \Delta'$. $\square$

This means that, by choosing a as the minimum element of Δ, we can always obtain a $\mathbf{Z}_n$-difference cover for X of the same cardinality as Δ, and containing 0.

Let $\delta_X^{(n)}$ denote one of the minimum $\mathbf{Z}_n$-difference cover for $X \subset \mathbf{Z}_n$. It is not hard to see that

Lemma 3.1. $(1 + \sqrt{1 + 4|X|})/2 \leqslant |\delta_X^{(n)}| \leqslant |X| + 1$.

Proof. The upper bound follows trivially since any set together with 0, is a $\mathbf{Z}_n$-difference cover for itself. The lower bound can be obtained by observing that the cardinality k of any given $\mathbf{Z}_n$-difference cover for X must clearly satisfy $k(k - 1) \geqslant |X|$. $\square$

By considering Lemma 3.1, it might be interesting to notice that the difference cover for $\mathbf{Z}_n$ proposed in Theorem 3.2 is optimal up to a multiplicative constant. Yet, it is not hard to see that $\delta^{(n)}_{\mathbf{Z}_n \setminus \{0\}}$ matches the lower bound in Lemma 3.1 if and only if it is a difference set for $\mathbf{Z}_n$.

We find it useful to associate with every $\mathbf{Z}_n$-difference cover a weighted digraph as follows

Definition 3.2. Given a $\mathbf{Z}_n$-difference cover Δ for $X \subset \mathbf{Z}_n$, its weighted digraph $\mathcal{G}(\Delta, X)$ has the elements of Δ as vertices and there exists an arc from d to d' of weight $(d' - d) \bmod n$ if and only if $(d' - d) \bmod n \in X$.

The weighted digraphs associated with minimum $\mathbf{Z}_n$-difference covers have the following important connection property:

Proposition 3.2. *Let $\delta_X^{(n)}$ be a minimum $\mathbf{Z}_n$-difference cover for $X \subset \mathbf{Z}_n$. Then, $\mathcal{G}(\delta_X^{(n)}, X)$ is* weakly connected.

Proof. Suppose, by contradiction, that $\mathcal{G}(\delta_X^{(n)}, X)$ consists of two or more weakly connected components insisting on mutually disjoint sets $\Delta_1, \dots, \Delta_k$ satisfying $\bigcup_{i=1}^k \Delta_i = \delta_X^{(n)}$. By translating as in Proposition 3.1, we transform each Δ_i into Δ_i' containing 0, and such that $|\Delta_i| = |\Delta_i'|$. It is easy to verify that $\bigcup_{i=1}^k \Delta_i' = \Delta'$ is still a $\mathbf{Z}_n$-difference cover for X, and that $|\Delta'| < |\delta_X^{(n)}|$, which is a contradiction. □

3.2. $\mathbf{Z}^+$-*difference cover*

We can rephrase Definition 3.1 in $\mathbf{Z}$, and obtain

Definition 3.3. The set $\Delta \subset \mathbf{Z}$ is a $\mathbf{Z}$-*difference cover* for the set $Y \subset \mathbf{Z}$ if, for each $y \in Y$, there exist two elements $a, b \in \Delta$ such that $y = a - b$.

Let us now make some technical considerations as we did after MinDCmod problem statement. Given a set $Y \subset \mathbf{Z}$, define $\widehat{Y} = \{|y|\colon\ y \in Y\}$. It is easy to see that $\Delta \subset \mathbf{Z}$ is a $\mathbf{Z}$-difference cover for Y if and only if it is a $\mathbf{Z}$-difference cover for $\widehat{Y}$ as well. Again, any nonempty $\mathbf{Z}$-difference cover clearly generates 0. Hence, we can restrict ourselves to search $\mathbf{Z}$-difference covers for subsets of $\mathbf{Z}^+$. Moreover, we can easily provide the analogous of Proposition 3.1, for $\mathbf{Z}$-difference cover translation:

Proposition 3.3. *Let Δ be a $\mathbf{Z}$-difference cover for $Y \subset \mathbf{Z}$, and let a fixed $a \in \mathbf{Z}$. Then the set $\Delta' = \{d - a\colon\ d \in \Delta\}$ is a $\mathbf{Z}$-difference cover for Y as well.*

By choosing a as the minimum element of Δ, we can always obtain a $\mathbf{Z}$-difference cover for Y of the same cardinality as Δ, and containing nonnegative integers only plus 0. All these considerations lead us to the following optimization problem in $\mathbf{Z}^+$:

MINIMUM $\mathbf{Z}^+$-DIFFERENCE COVER (MinDC)
INPUT: $Y \subset \mathbf{Z}^+$
OUTPUT: $\Delta = \{0\} \cup B$, with $B \subset \mathbf{Z}^+$, such that Δ is a $\mathbf{Z}^+$-difference cover for Y
MEASURE: Cardinality of the $\mathbf{Z}^+$-difference cover, i.e., $|\Delta|$

Denoting by δ_Y a minimum $\mathbf{Z}^+$-difference cover for Y, we get

Lemma 3.2. $(1 + \sqrt{1 + 8|Y|})/2 \leqslant |\delta_Y| \leqslant |Y| + 1$.

Proof. Again, the upper bound follows trivially. For the lower bound, it is easy to see that the cardinality k of any given $\mathbf{Z}^+$-difference cover for Y must now satisfy $k(k - 1)/2 \geqslant |Y|$. □

Notice that δ_Y matches the lower bound in Lemma 3.2 if and only if it is a Golomb's ruler.

Even with a $\mathbf{Z}^+$-difference cover Δ for Y, we can associate the weighted digraph $\mathcal{G}(\Delta, Y)$ as in Definition 3.2, but now edge labels are computed by simple differences, i.e., if d and d' are vertices of the digraph, then there exists an arc from d to d' of weight

$d' - d$ if and only if $(d' - d) \in Y$. Indeed, by using Proposition 3.3, we can suitably adapt the proof of Proposition 3.2 to obtain

Proposition 3.4. *Let δ_Y be a minimum $\mathbf{Z}^+$-difference cover for Y. Then, $\mathcal{G}(\delta_Y, Y)$ is* weakly connected.

This proposition enables us to provide an upper bound on the value of the elements of minimum $\mathbf{Z}^+$-difference covers.

Proposition 3.5. *Let $Y = \{y_1, y_2, \ldots, y_m\} \subset \mathbf{Z}^+$. Each element of a minimum $\mathbf{Z}^+$-difference cover δ_Y* is less than or equal to $m \max_i y_i$.

Proof. By Proposition 3.4, the digraph $\mathcal{G}(\delta_Y, Y)$ is weakly connected. So, each value $d \in \delta_Y$ is easily seen to be obtained as the weight of an elementary chain in $\mathcal{G}(\delta_Y, Y)$ joining vertex 0 to vertex d. Since the length of elementary chains does not exceed $|\delta_Y| - 1$, each element of δ_Y is less than or equal to $(|\delta_Y| - 1) \max_i y_i$. Then, the result follows from Lemma 3.2. $\square$

4. Pseudo-polynomial time algorithms establishing extrema

In this section, we give a useful characterization of those sets for which the cardinality of minimum difference covers exactly matches the upper bounds given in Lemmas 3.1 and 3.2. More precisely, we state

Definition 4.1. A set $E \subset \mathbf{Z}^+$ ($E \subset \mathbf{Z}_n$) is a $\mathbf{Z}^+$-*extremum* ($\mathbf{Z}_n$-*extremum*) if and only if the cardinality of a minimum $\mathbf{Z}^+$-difference cover ($\mathbf{Z}_n$-difference cover) for E equals $|E| + 1$.

In other words, an extremum is a set admitting trivial minimum covers consisting of the set itself plus 0. The following theorem gives a characterization of $\mathbf{Z}^+$-extrema.

Theorem 4.1. *Let $Y = \{y_1, y_2, \ldots, y_m\} \subset \mathbf{Z}^+$, and let $a_1, a_2, \ldots, a_m$ be variables on $\{-1, 0, 1\}$. Then Y is a $\mathbf{Z}^+$-extremum if and only if*

$$\sum_{k=1}^{m} a_k y_k = 0 \quad \Leftrightarrow \quad a_k = 0, \textit{ for every } 1 \leqslant k \leqslant m.$$

Proof. (If) Suppose, by contradiction, that there exists a nonzero assignment for a_k's such that $\sum_{k=0}^{m} a_k y_k = 0$ and let, without loss of generality, $a_m \neq 0$. We can construct a $\mathbf{Z}^+$-difference cover Δ for Y by the following algorithm:

INPUT: $Y = \{y_1, y_2, \ldots, y_m\}$
1: $\Delta := \{0\}$;
2: `for` $i := 1$ `to` $m - 1$ `do`
3: `begin`
4: `if` $a_i = 0$ `then` $t := y_i$ `else` $t := \left|\sum_{k=1}^{i} a_k y_k\right|$;

5: $\quad\quad \Delta := \Delta \cup \{t\}$;
6: $\quad$ `end`
7: output(Δ).

It is easy to see that the returned Δ is a $\mathbf{Z}^+$-difference cover for Y. In fact according to the `if`-test at line 4, for every $1 \leqslant i < m$, if $a_i = 0$ then y_i is placed in Δ, and it can be obtained by the difference $y_i - 0$. Otherwise, $|\sum_{k=1}^{i} a_k y_k|$ is placed in Δ, and y_i is obtained by $\| \sum_{k=1}^{i} a_k y_k| - |\sum_{k=1}^{i-1} a_k y_k\|$, where $|\sum_{k=1}^{i-1} a_k y_k|$ has been already put in Δ during the previous iterations of the `for`-loop. Finally, since $a_m \neq 0$, we have in Δ the value $y_m = |\sum_{k=1}^{m-1} a_k y_k|$ which can be obtained by difference with 0.

The resulting Δ has cardinality m, and this contradicts the fact that, being a $\mathbf{Z}^+$-extremum, Y cannot have $\mathbf{Z}^+$-difference covers with less than $m+1$ elements.

(Only if) Suppose that Y is not a $\mathbf{Z}^+$-extremum, i.e., $|\delta_Y| \leqslant m$. Then, the number of vertices of the digraph $\mathcal{G}(\delta_Y, Y)$ is at most equal to the number of its edges. By Proposition 3.4, $\mathcal{G}(\delta_Y, Y)$ is weakly connected, and hence it must contain a cycle. This cycle can be used to exhibit a nonzero assignment of a_k's yielding $\sum_{k=1}^{m} a_k y_k = 0$ as follows: For any edge labeled y_i not in the cycle, set $a_i = 0$. For edges in the cycle, first set a traveling direction along the cycle itself, then let $a_i = 1$ for those y_i following such an orientation, and let $a_i = -1$ otherwise. $\square$

For $\mathbf{Z}_n$-extrema, we can give an analogous characterization:

Theorem 4.2. *Let $X = \{x_1, x_2, \ldots, x_m\} \subset \mathbf{Z}_n$, and let $a_1, a_2, \ldots, a_m$ variables on $\{-1, 0, 1\}$. Then X is a $\mathbf{Z}_n$-extremum if and only if*

$$\left(\sum_{k=1}^{m} a_k x_k\right) \bmod n = 0 \quad \Leftrightarrow \quad a_k = 0,\ \textit{for every}\ 1 \leqslant k \leqslant m.$$

Proof. (If) As in the (If) part of Theorem 4.1 proof, with the only difference that operations are now to be performed $\bmod n$ (n is given as input to the algorithm). In particular, the statement at line 4 of the algorithm returning the difference cover now becomes

4: `if` $a_i = 0$ `then` $t := y_i$ `else` $t := (\sum_{k=1}^{i} a_k y_k) \bmod n$;

(Only if) Analogous to the (Only if) part of Theorem 4.1 proof. Only, we now use Proposition 3.2 to analyze the digraph $\mathcal{G}(\delta_X^{(n)}, X)$. $\square$

In what follows, we use the characterization provided in the latter theorem to design an algorithm testing—in time polynomial in n—whether a given subset of $\mathbf{Z}_n$ is a $\mathbf{Z}_n$-extremum. To this aim, we need the following

Proposition 4.1. *Let $(a_1, a_2, \ldots, a_m), (a'_1, a'_2, \ldots, a'_m) \in \{0, 1\}^m$ satisfying*

$$(a_1, a_2, \ldots, a_m) \neq (a'_1, a'_2, \ldots, a'_m).$$

(i) *If* $X = \{x_1, \dots, x_m\} \subset \mathbf{Z}_n$ *is a* $\mathbf{Z}_n$*-extremum, then*

$$\left(\sum_{i=1}^{m} a_i x_i\right) \bmod n \neq \left(\sum_{i=1}^{m} a'_i x_i\right) \bmod n.$$

(ii) *If* $Y = \{y_1, \dots, y_m\} \subset \mathbf{Z}^+$ *is a* $\mathbf{Z}^+$*-extremum, then*

$$\sum_{i=1}^{m} a_i y_i \neq \sum_{i=1}^{m} a'_i y_i.$$

Proof. We just consider (i), since point (ii) can be proved analogously. Suppose, by contradiction, that $(\sum_{i=1}^{m} a_i x_i) \bmod n = (\sum_{i=1}^{m} a'_i x_i) \bmod n$. Hence, we get

$$\left(\sum_{i=1}^{m} (a_i - a'_i) x_i\right) \bmod n = 0, \quad \text{with } (a_i - a'_i) \in \{-1, 0, 1\}. \tag{1}$$

By setting $b_i = (a_i - a'_i)$, we can rewrite Eq. (1) as $(\sum_{i=1}^{m} b_i x_i) \bmod n = 0$, with $b_i \in \{-1, 0, 1\}$. Since X is an extremum, Theorem 4.2 ensures that this equation is satisfied if and only if b_i's are all 0. This clearly would imply $(a_1, a_2, \dots, a_m) = (a'_1, a'_2, \dots, a'_m)$, against the initial hypothesis. □

Proposition 4.1 enables us to give an optimal upper bound on the cardinality of subsets of $\mathbf{Z}_n$ to be extrema; a similar upper bound is stated for $\mathbf{Z}^+$-extrema as well:

Lemma 4.1.

(i) *If* $X = \{x_1, \dots, x_m\} \subset \mathbf{Z}_n$ *is a* $\mathbf{Z}_n$*-extremum, then* $m \leqslant \lfloor \log n \rfloor$.
(ii) *If* $Y = \{y_1, \dots, y_m\} \subset \mathbf{Z}^+$ *is a* $\mathbf{Z}^+$*-extremum, then* $m < 2(1 + \log(\max_i y_i))$.

Proof.

(i) Proposition 4.1(i) ensures that the expression $(\sum_{i=1}^{m} a_i x_i) \bmod n$ returns 2^m different values, one per each different choice of $(a_1, a_2, \dots, a_m) \in \{0, 1\}^m$. Since we are operating in $\mathbf{Z}_n$, clearly we have $2^m \leqslant n$ which completes the proof.
(ii) By Proposition 4.1(ii), we obtain $2^m \leqslant 2m \max_i y_i \leqslant 2(\max_i y_i)^2$, whence the result. □

We are now able to single out examples of $\mathbf{Z}_n$ and $\mathbf{Z}^+$-extrema.

Example. Consider the set $E_\rho = \{1, \rho^1, \dots, \rho^\alpha\}$, for any given integer $\rho \geqslant 2$. We can prove that E_ρ is a $\mathbf{Z}^+$-extremum, and a $\mathbf{Z}_n$-extremum for any $n > (\rho^{\alpha+1} - 1)/(\rho - 1)$.

First, it is easy to see that $\sum_{k=0}^{\alpha} a_k \rho^k = 0$ if and only if every $a_k \in \{-1, 0, 1\}$ is 0. Otherwise, we could write $\sum_{\{i:\, a_i = 1\}} \rho^i = \sum_{\{i:\, a_i = -1\}} \rho^i$. Since any number has a unique representation in base ρ, we would get a contradiction. By Theorem 4.1, this shows that E_ρ is a $\mathbf{Z}^+$-extremum.

To see that E_ρ is a $\mathbf{Z}_n$-extremum for $n = \frac{\rho^{\alpha+1}-1}{\rho-1} + 1$, we apply Theorem 4.2, noticing that $\sum_{k=0}^{\alpha} \rho^k = n - 1$.

This example can be used to show the optimality of the upper bound given in Lemma 4.1(i). In fact

Proposition 4.2. *There exists a $\mathbf{Z}_n$-extremum of cardinality $\lfloor \log n \rfloor$.*

Proof. The set $E_2 = \{1, 2, \ldots, 2^\alpha\}$ is a $\mathbf{Z}_n$-extremum for $n = 2^{\alpha+1}$, as seen in the previous example. Its cardinality is exactly $\lfloor \log n \rfloor$. $\square$

Clearly E_2 also witnesses the optimality of the upper bound in Lemma 4.1(ii) up to a multiplicative constant.

We are now ready to show that

Theorem 4.3. *Deciding whether $X = \{x_1, \ldots, x_m\} \subset \mathbf{Z}_n$ is a $\mathbf{Z}_n$-extremum can be performed in $\mathcal{O}(n^{\log 3})$ time.*

Proof. Our decision algorithm sketched below has two phases. In the first phase, we simply test whether $m > \lfloor \log n \rfloor$. If this holds true, we reject according to Lemma 4.1(i). Otherwise, the second phase starts, where a nontrivial solution for the equation $(\sum_{k=1}^{m} a_k x_k) \bmod n = 0$, with $a_i \in \{-1, 0, 1\}$, is searched. Theorem 4.2 ensures that such a solution does not exist if and only if X is a $\mathbf{Z}_n$-extremum.

```
INPUT: n ∈ Z^+, X = {x_1, x_2, ..., x_m   /* recall that 0 ∉ X
if m > ⌊log n⌋ then
    reject
else
    begin
      B := ∅;
      for i := 1 to m do
          begin
            B^+ := ∅;
            B^- := ∅;
             for each b ∈ B do
                 begin
                   B^+ := {(b + x_i) mod n} ∪ B^+;
                   B^- := {(b − x_i) mod n} ∪ B^-;
                 end;
             B := B ∪ B^+ ∪ B^- ∪ {x_i, (−x_i) mod n};
          end;
      if 0 ∈ B then
          reject
      else
          accept
    end.
```

We briefly explain how the `else`-part of the algorithm works. Before entering the iteration on $x_i \in X$ at the `for`-loop at line 6, the set B contains all the possible values $(\sum_{k=1}^{i-1} a_k x_k) \bmod n$, for $a_1, a_2, \ldots, a_{i-1} \in \{-1, 0, 1\}$ not all 0. During the iteration on x_i, we update B to contain all the linear combinations involving also x_i by summing (line 12) and subtracting (line 13) x_i to every element of B. Finally, we also put in B the two linear combinations $(\sum_{k=1}^{i} a_k x_k) \bmod n$ where $a_i = \pm 1$, and $a_k = 0$ for every $1 \leqslant k < i$ (line 15). Thus, after scanning the whole X, we will have in B all the values $(\sum_{k=1}^{m} a_k x_k) \bmod n$ where a_k's are not all 0. According to Theorem 4.2, we reject or accept depending on whether 0 belongs to B or not, respectively (`if`-test, line 17).

For the running time of this algorithm, we just observe that, after the kth iteration of the outer `for`-loop (line 6), the cardinality of the set B is at most $3^k - 1$. Then, the number $\Gamma(k)$ of operations $\bmod n$ performed at lines 12, 13, 15 up to the kth iteration satisfies $\Gamma(k) \leqslant \Gamma(k-1) + 2 \cdot 3^{k-1} - 1$, with $\Gamma(1) = 1$. Such a recurrence has solution $\Gamma(k) \leqslant 3^k - k - 1$. Since $k \leqslant m \leqslant \lfloor \log n \rfloor$, we get that the running time is $\mathcal{O}(n^{\log 3})$. □

We notice that the running time of the algorithm sketched in the proof of the previous theorem is polynomial (in the length of the input, and hence efficient) if the cardinality of the input set X is "the order of" n^α, with constant $0 < \alpha \leqslant 1$. If this is not the case, our algorithm needs exponential time. Thus, we have a *pseudo-polynomial* time algorithm for testing $\mathbf{Z}_n$-extrema. This algorithm can be easily adapted to check also for $\mathbf{Z}^+$-extrema, and this shows that

Proposition 4.3. *Deciding whether a set $Y = \{y_1, y_2, \ldots, y_m\} \subset \mathbf{Z}^+$ is a $\mathbf{Z}^+$-extremum can be performed in* pseudo-polynomial *time.*

Proof. We can directly use the algorithm in Theorem 4.3 on input Y (without providing $n \in \mathbf{Z}^+$). In the first phase, we check whether $m \geqslant 2(1 + \log(\max_i y_i))$, in case rejecting by Lemma 4.1(ii). Otherwise, we start the second phase for which it is not hard to verify that the running time is $\mathcal{O}((\max_i y_i)^{\log 9})$. Clearly, if $\max_i y_i = m^{\mathcal{O}(1)}$, we would obtain a polynomial running time. □

The algorithms presented in the proofs of Theorem 4.3 and Proposition 4.3 can be straightforwardly adapted to *test whether sets are not extrema*: it is enough to switch acceptance with rejection. This show that even *non-extremity can be tested in pseudo-polynomial time*. This is probably the best we can achieve. In fact, in the next section, we are going to show that testing whether sets are not extrema, both in $\mathbf{Z}^+$ and in $\mathbf{Z}_n$, is NP-complete.

5. Establishing non-extremity is NP-complete

We start by considering the characterization of $\mathbf{Z}^+$-extrema in Theorem 4.1 by which deciding whether $Y = \{y_1, y_2, \ldots, y_m\} \subset \mathbf{Z}^+$ is *not* a $\mathbf{Z}^+$-extremum is *equivalent* to decide whether there exists a nonzero assignment of a_k's, with $a_k \in \{-1, 0, 1\}$, satisfying the equation $\sum_{k=1}^{m} a_k y_k = 0$. We call this latter decision problem $\mathsf{Ass}(-1, 0, 1)$.

To study the hardness of $\mathsf{Ass}(-1,0,1)$, we find it useful to introduce a more general problem $\mathsf{Sys}(-1,0,1)$, where the input is a system of equations

$$\mathcal{S} = \left\{ \sum_{k=1}^{n} w_k^{(t)} x_k = 0 \right\}_{0 \leqslant t \leqslant \hat{n}}$$

in the variables $x_k \in \{-1,0,1\}$, with coefficients $w_k^{(t)} \in \mathbf{N}$, and with $\hat{n} = n^{\mathcal{O}(1)}$. This problem asks whether there exists an assignment in $\{-1,0,1\}$ of x_k's satisfying $\mathcal{S}$ and such that not all x_k's are set to 0. We show that

Lemma 5.1. $\mathsf{Sys}(-1,0,1) \leqslant_p \mathsf{Ass}(-1,0,1)$.

Proof. We reduce the system of equations $\mathcal{S} = \{\sum_{k=1}^{n} w_k^{(t)} x_k = 0\}_{0 \leqslant t \leqslant \hat{n}}$ to a single equation $\mathcal{E}(x_1,\ldots,x_n) = 0$, such that any assignment in $\{-1,0,1\}$ of x_k's is a solution of $\mathcal{E}$ if and only if it is a solution of $\mathcal{S}$. To this purpose, we set $W = 1 + \max_t \sum_{k=1}^{n} w_k^{(t)}$, and define $\mathcal{E}(x_1,\ldots,x_n) = 0$ as

$$\sum_{k=1}^{n} w_k^{(0)} x_k + W \sum_{k=1}^{n} w_k^{(1)} x_k + W^2 \sum_{k=1}^{n} w_k^{(2)} x_k + \cdots + W^{\hat{n}} \sum_{k=1}^{n} w_k^{(\hat{n})} x_k = 0. \tag{2}$$

Any assignment satisfying the system $\mathcal{S}$ satisfies such an equation as well.

Vice versa, suppose we have an assignment of x_k's satisfying Eq. (2). For the sake of readability, let $H_i = \sum_{k=1}^{n} w_k^{(i)} x_k$, so that we can rewrite Eq. (2) as

$$H_0 = -W\left(\sum_{i=1}^{\hat{n}} H_i W^{i-1}\right).$$

This means that W divides H_0, but since $-W < H_i < W$ for each $0 \leqslant i \leqslant \hat{n}$, we must conclude that $H_0 = 0$. By iterating such a reasoning, we obtain that the assignment satisfying Eq. (2) satisfies each H_i as well.

We end by quickly noticing that computing $\mathcal{E}$ from $\mathcal{S}$ is easily seen to be done in polynomial time. □

Now, we need to recall the well-known NP-complete problem $\mathsf{Partition}$ (see, e.g., [9, Chapter 3]). Here, we formulate such a problem in a slightly modified but perfectly equivalent version which is more suited to our purposes.

PARTITION
INPUT: Finite set $Y = \{y_1, y_2, \ldots, y_m\} \subset \mathbf{Z}^+$.
OUTPUT: Is there an assignment in $\{-1,1\}$ of b_k's s.t. $\sum_{k=1}^{m} b_k y_k = 0$?

In other words, we ask whether Y can be partitioned into two subsets of "equal sum".

Lemma 5.2. $\mathsf{Partition} \leqslant_p \mathsf{Sys}(-1,0,1)$.

Proof. Let $Y = \{y_1, \ldots, y_m\} \subset \mathbf{Z}^+$ be an input instance of Partition. We construct the following system of $m+1$ equations in the $2m+1$ variables $\{b_1, b_2, \ldots, b_m, c_1, c_2, \ldots, c_m, a\}$ ranging on $\{-1, 0, 1\}$:

$$\mathcal{S}_Y = \begin{cases} \sum_{k=1}^{m} b_k y_k = 0 \\ b_1 + 2c_1 + a = 0 \\ b_2 + 2c_2 + a = 0 \\ \vdots \\ b_m + 2c_m + a = 0. \end{cases}$$

Now, notice that any *solution for* $\mathcal{S}_Y$ *either has all the variables set to* 0 (*i.e., is the trivial one*) *or is on* $\{-1, 1\}$ *only*. In fact, take a solution σ where $b_i = 0$, for a given $1 \leqslant i \leqslant m$. Since all the variables range only on $\{-1, 0, 1\}$, the corresponding equation $b_i + 2c_i + a = 0$ has a unique solution for $a = 0$ and $c_i = 0$. In turn, $a = 0$ yields the equations $b_k + 2c_k = 0$, for $1 \leqslant k \leqslant m$, giving that σ must set all the variables to 0. This reasoning shows that any possible nontrivial solution for $\mathcal{S}_Y$ yields a solution in $\{-1, 1\}$ for the equation $\sum_{k=1}^{m} b_k y_k = 0$, and hence represents a partition of Y.

Vice versa, it is clear that any possible partition of Y can be immediately transformed into a solution for the corresponding system $\mathcal{S}_Y$. It is enough to add $a \in \{-1, 1\}$, and $c_k = (-a - b_k)/2$ for every $1 \leqslant k \leqslant m$.

The construction of $\mathcal{S}_Y$ from Y is easily seen to be performed in polynomial time, and this completes the proof. □

We are now ready to prove the NP-completeness of testing non-extremity.

Theorem 5.1. *Deciding whether a set* $Y = \{y_1, y_2, \ldots, y_m\} \subset \mathbf{Z}^+$ *is not a* $\mathbf{Z}^+$*-extremum is* NP-complete.

Proof. As above recalled, such a decision problem is equivalent to Ass$(-1, 0, 1)$ for the equation $\sum_{k=1}^{m} a_k y_k = 0$. A polynomial time nondeterministic algorithm for solving this latter problem simply guesses a nonzero assignment in $\{-1, 0, 1\}$ for a_k's, and then checks in polynomial time whether the assignment satisfies the equation. This shows that Ass$(-1, 0, 1)$ belongs to NP.

From Lemmas 5.1 and 5.2, we get that Partition $\leqslant_p$ Sys$(-1, 0, 1) \leqslant_p$ Ass$(-1, 0, 1)$. The result follows from the NP-completeness of Partition. □

This latter result enables us to obtain the NP-completeness even for testing non-extremity in $\mathbf{Z}_n$.

Theorem 5.2. *Deciding whether a subset of* $\mathbf{Z}_n$ *is not a* $\mathbf{Z}_n$*-extremum is* NP-complete.

Proof. By the characterization of $\mathbf{Z}_n$-extrema in Theorem 4.2, one may easily design a nondeterministic polynomial time algorithm for testing non-extremity in $\mathbf{Z}_n$, thus setting this problem in NP. To show its completeness, by Theorem 5.1, it is enough to exhibit a reduction from testing non-extremity in $\mathbf{Z}^+$. Our reduction works as follows: given the instance $Y = \{y_1, y_2, \dots, y_m\} \subset \mathbf{Z}^+$, return the instance $Y = \{y_1, y_2, \dots, y_m\} \subset \mathbf{Z}_n$, with $n = 1 + \sum_{i=1}^{m} y_i$. We must show that Y is not a $\mathbf{Z}^+$-extremum if and only if Y is not a $\mathbf{Z}_n$-extremum, with $n = 1 + \sum_{k=1}^{m} y_k$. To this purpose, we simply notice that, for every assignment of a_k's in $\{-1, 0, 1\}$, we have

$$-n < \sum_{k=1}^{m} a_k y_k < n.$$

Hence

$$\left(\sum_{k=1}^{m} a_k y_k\right) \bmod n = 0 \quad \Leftrightarrow \quad \sum_{k=1}^{m} a_k y_k = 0.$$

By recalling the characterization of $\mathbf{Z}^+$ and $\mathbf{Z}_n$-extrema given in Theorems 4.1 and 4.2, respectively, we get the result. □

6. The hardness of MinDC and MinDCmod, and open problems

Let us finally analyze the complexity of the optimization problems MinDC and MinDCmod presented in Section 3. The results in the previous section enable us to state that

Theorem 6.1. MinDC *and* MinDCmod *are* NP-hard.

Proof. Let us consider MinDC. Theorem 5.1 states that testing non-extremity in $\mathbf{Z}^+$ is NP-complete. Thus, the claimed result can be shown by exhibiting a polynomial time Turing-reduction from this decision problem to MinDC. It is easy to exhibit a Turing machine that decides in polynomial time whether a given $Y \subset \mathbf{Z}^+$ is not a $\mathbf{Z}^+$-extremum by having an oracle for MinDC. First, we use such an oracle to compute δ_Y, then we check whether $|\delta_Y| < |Y| + 1$.

The NP-hardness of MinDCmod can be obtained by the same argument, using the NP-completeness of testing non-extremity in $\mathbf{Z}_n$ given in Theorem 5.2. □

For the sake of completeness, we are now going to exhibit a Turing-reduction from MinDC to MinDCmod. To this purpose, we recall a well-known representation of $\mathbf{Z}_n$ (see, e.g., [10]) according to which the elements of $\mathbf{Z}_n$ can be regarded to as points on a circle. We designate a point to represent 0 followed by points $1, 2, \dots, n-1$ placed at equal distance to cover all the circumference. In this structure, sums and subtractions $\bmod n$ can be performed by simply moving back and forth on the circle. *The length of an interval* $[a, b]$ *in* $\mathbf{Z}_n$ *is the length of the minimum arc joining* a *to* b*, and can be computed as* $\ell(a, b) = \min\{(a - b) \bmod n, (b - a) \bmod n\}$.

Lemma 6.1. *Let* $Y = \{y_1, y_2, \dots, y_m\} \subset \mathbf{Z}^+$*, and set* $n = 1 + (m + 1)\max_i y_i$*. Then, for each minimum* $\mathbf{Z}_n$*-difference cover* $\delta_Y^{(n)}$ *there exists an interval in* $\mathbf{Z}_n$ *of length greater then* $\max_i y_i$ *not containing any element of* $\delta_Y^{(n)}$.

Proof. Let $\delta_Y^{(n)} = \{d_1, d_2, \ldots, d_s\}$, with $d_1 < d_2 < \cdots < d_s$, and let $\lambda = \max_{1\leqslant j<s}(d_{j+1} - d_j)$, i.e., the maximal distance between two consecutive elements of $\delta_Y^{(n)}$. If $\lambda > \max_i y_i$ there is noting to prove. Otherwise, since $\delta_Y^{(n)}$ is minimum, it must be that $d_s - d_1 \leqslant \sum_{i=1}^m y_i \leqslant m \max_i y_i$, by using as argument the connectivity of the weighted digraph associated with $\delta_Y^{(n)}$ (Proposition 3.2). This proves that $\ell(d_1, d_s) > \max_i y_i$. $\square$

We are now ready to show that

Theorem 6.2. MinDC $\leqslant_T$ MinDCmod.

Proof. Let $Y = \{y_1, y_2, \ldots, y_m\} \subset \mathbf{Z}^+$ be an input instance for MinDC. We begin by setting $n = 1 + (m+1)\max_i y_i$. Then, from an oracle for MinDCmod, we obtain a minimum $\mathbf{Z}_n$-difference cover $\delta_Y^{(n)}$ for Y. Now, by Proposition 3.1, we turn $\delta_Y^{(n)}$ into $\delta_{Yg}^{(n)}$ with the same cardinality as $\delta_Y^{(n)}$, so that $0 \in \delta_{Yg}^{(n)}$ and the interval in $\mathbf{Z}_n$ emphasized in Lemma 6.1 is $[g, 0]$, with g the maximum element of $\delta_{Yg}^{(n)}$. This gives that $g \leqslant m \max_i y_i$ and this fact shows that, for any $a, b \in \delta_{Yg}^{(n)}$, we have $|a - b| \leqslant m \max_i y_i < n$. Then, in $\delta_{Yg}^{(n)}$, the use of $\operatorname{mod} n$ to represent by difference the elements of Y is superfluous. Hence, $\delta_{Yg}^{(n)}$ is also a minimum $\mathbf{Z}^+$-difference cover for Y. $\square$

This Turing-reduction, together with the NP-hardness of MinDC, yields the NP-hardness of MinDCmod too. From this approach, one can see that the result follows without actually using operations $\operatorname{mod} n$. Yet, the NP-hardness of MinDCmod does not imply the NP-hardness of *its restricted version where input instances are the whole sets* $\mathbf{Z}_n$; establishing the complexity of this restricted version remains open.

However, as it is customary in complexity theory (see, e.g., [9]), we briefly investigate the associated decision problem: inputs are the binary strings of the form $1^n 0\langle k\rangle$, where $1^n = 11\cdots1$ (n-times) is the unary representation of the integer n, and $\langle k\rangle$ is the binary representation of the integer k.

BOUNDED SIZE DIFFERENCE COVER (BsDC)
INPUT: $1^n 0\langle k\rangle$.
OUTPUT: Is there a $\mathbf{Z}_n$-difference cover Δ for $\mathbf{Z}_n$ s.t. $|\Delta| \leqslant k$?

It is easy to see that BsDC belongs to NP, and that it reduces to BsDC′, the same problem where input instances satisfy $k < n$ (for $k \geqslant n$, we always have a positive answer).

To study the complexity of BsDC, we need to recall the notion of sparseness [15]: a set $S \subseteq \{0,1\}^*$ is said to be *sparse* if there exists a positive constant c such that $|S \cap \{0,1\}^m| \leqslant m^c$, for every $m \geqslant 2$. Let us now consider the set $S[\mathsf{BsDC}']$ containing the instances of BsDC′ with positive answer. For every size $m = n + \log k + 1$ of input instances, $|S[\mathsf{BsDC}'] \cap \{0,1\}^m| \leqslant k < n < m$. This shows that $S[\mathsf{BsDC}']$ is a sparse set. From [15], we know that *if a sparse set is* NP*-complete, than* P = NP. Hence, if P ≠ NP, the problem BsDC is not NP-complete. This leaves open the possibility of setting BsDC in P, even if efficient algorithms for this decision problem do not necessarily imply efficient solutions for MinDCmod restricted to sets $\mathbf{Z}_n$ as input.

Acknowledgements

The authors wish to thank the anonymous referees for very helpful comments and valuable remarks.

References

[1] W.C. Babcock, Intermodulation interference in radio systems, Bell System Technical J. (1953) 63–73.
[2] C. Berge, Graphs, North-Holland, Amsterdam, 1985.
[3] A. Bertoni, C. Mereghetti, B. Palano, Golomb rulers and difference sets for succinct quantum automata, Internat. J. Found. Comput. Sci. 14 (2003) 871–888.
[4] G.S. Bloom, S.W. Golomb, Applications of numbered undirected graphs, in: Proc. IEEE, vol. 65, 1977, pp. 562–570.
[5] S. Burkhardt, J. Kärkkäinen, Fast lightweight suffix array construction and checking, in: Proc. 14th Annual Symposium on Combinatorial Pattern Matching, in: Lecture Notes in Comput. Sci., vol. 2676, Springer, Berlin, 2003, pp. 55–69.
[6] C. Colbourn, A. Ling, Quorums from difference covers, Inform. Process. Lett. 75 (2000) 9–12.
[7] C. Colbourn, J.H. Dinitz, D.R. Stinson, Applications of combinatorial designs to communications, cryptography, and networking, in: Walker (Ed.), Surveys in Combinatorics, in: London Math. Soc. Lecture Note Series, vol. 187, Cambridge Univ. Press, Cambridge, 1999.
[8] A. Daurat, Y. Gérard, M. Nivat, The chords' problem, Theoret. Comput. Sci. 282 (2002) 319–336.
[9] M.R. Garey, D.S. Johnson, Computers and Intractability. A Guide to the Theory of NP-Completeness, Freeman, New York, 1979.
[10] R.L. Graham, D.E. Knuth, O. Patashnik, Concrete Mathematics: A Foundation for Computer Science, Addison-Wesley, Reading, MA, 1998.
[11] J. Gruska, Quantum Computing, McGraw-Hill, New York, 1999.
[12] A.W. Lam, D.V. Sarwate, An optimum time hopping patterns, IEEE Trans. Comm. 36 (1988) 380–382.
[13] P. Lemke, S.S. Skiena, W.D. Smith, Reconstructing sets from interpoint distances, in: Discrete and Computational Geometry: The Goodman–Pollack Festschrift, in: Algorithms and Combinatorics, vol. 25, Springer, Berlin, 2003.
[14] M. Maekawa, A square root N algorithm for mutual exclusion in decentralized systems, ACM Trans. Computer Systems 3 (1985) 145–159.
[15] S.R. Mahaney, Sparse complete sets for NP: solution of a conjecture of Berman and Hartmanis, J. Comput. Syst. Sci. 25 (1982) 130–143.
[16] C. Mereghetti, B. Palano, On the size of one-way quantum finite automata with periodic behaviors, Theoret. Inform. Appl. 36 (2002) 277–291.
[17] J.P. Robinson, A.J. Bernstein, A class of binary recurrent codes with limited error propagation, IEEE Trans. Inform. Theory 13 (1967) 106–113.
[18] J. Singer, A theorem in finite projective geometry and some applications to number theory, Trans. Amer. Math. Soc. 43 (1938) 337–385.
[19] B. Wichmann, A note on restricted difference bases, J. London Math. Soc. 38 (1963) 465–466.