ÖAW Austrian Academy of Sciences

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

JKU JOHANNES KEPLER UNIVERSITY LINZ

A Note on the Average Complexity Analysis of the Computation of Periodic and Aperiodic Ternary Complementary Pairs

C. Koukouvinos V. Pillwein D.E. Simos

Z. Zafeirakopoulos

DK-Report No. 2010-08 10 2010

A–4040 LINZ, ALTENBERGERSTRASSE 69, AUSTRIA

Supported by

Austrian Science Fund (FWF)

FWF Der Wissenschaftsfonds.

Upper Austria

LAND OBERÖSTERREICH

DK sponsors:

- **Johannes Kepler University Linz** (JKU)
- **Austrian Science Fund** (FWF)
- **Upper Austria**

A Note on the Average Complexity Analysis of the Computation of Periodic and Aperiodic Ternary Complementary Pairs

Christos Koukouvinos* Veronika Pillwein† Dimitris E. Simos‡ Zafeirakis Zafeirakopoulos§

Abstract

We give an average complexity analysis for a new formalism pertaining periodic and aperiodic ternary complementary pairs. The analysis is done in three levels, so that we end up with an accurate estimate. The way of separating the candidate pairs into suitable classes of ternary sequences is interesting, allowing us to use fundamental tools of Symbolic Computation, such as Holonomic functions and asymptotic analysis to derive an average complexity of $O(n\sqrt{n}\log n)$ for sequences of length n.

Keywords: Sequences, Periodic Autocorrelation Function, Non-Periodic Autocorrelation Function, Complexity, Algorithms, Average-case analysis.
MSC classification: Primary 05B20 Secondary 68Q25, 68W40

1 Introduction

In this paper, we detail an average complexity analysis for a new formalism that exhibits the cross-fertilization of Combinatorics with Theoretical Computer Science, in the study of sequences with zero periodic and non-periodic autocorrelation function.

Definition 1 *For a sequence $A = [a_1, a_2, \ldots, a_n]$ of length n the periodic autocorrelation function (PAF) and the non-periodic autocorrelation function (NPAF), denoted by $P_A(s)$ and $N_A(s)$ respectively are defined as*

$$P_A(s) = \sum_{i=1}^{n} a_i a_{i+s} \ , s = 0, 1, \ldots, n-1 \quad N_A(s) = \sum_{i=1}^{n-s} a_i a_{i+s} \ , s = 0, 1, \ldots, n-1.$$

where in PAF we consider (i+s) modulo n, see also [9].

*Department of Mathematics, National Technical University of Athens, Zografou 15773, Athens, Greece
†Research Institute for Symbolic Computation (RISC), Linz Austria
‡Department of Mathematics, National Technical University of Athens, Zografou 15773, Athens, Greece. Supported by a scholarship awarded by the Secretariat of the Research Committee of N.T.U.A.
§Doctoral program "Computational Mathematics" supported by the Austrian Science Fund (FWF) under grant W1214/DK6

Definition 2 *Two sequences, $A = [a_1, \ldots, a_n]$ and $B = [b_1, \ldots, b_n]$, of length n are said to have* zero PAF (respectively zero NPAF)*, if $P_A(s) + P_B(s) = 0$ (respectively $N_A(s) + N_B(s) = 0$) for $s = 1, \ldots, n-1$.*

The support of a sequence A of length n is the set of positions where its entries are nonzero. We will use the notion of support extensively. Since we consider information about the sign of the sequence elements, unlike in the usual definition of support, we provide the following definition:

Definition 3 *The support of a sequence $A = [a_1, \ldots, a_n]$ denoted by $SUP(A)$, is defined as $SUP(A) = \{\pm i : i, a_i > 0 \ \vee \ -i, a_i < 0 \mid i = 1, \ldots, n\}$.*

We are interested in bundling together the indices of entries with the same sign. This motivates the following definitions:

Definition 4 *The positive and negative support of a sequence $A = [a_1, \ldots, a_n]$, denoted by $POS(A)$ and $NEG(A)$ respectively, are defined as*

$$POS(A) = \{i : a_i > 0 \mid i = 1, \ldots, n\} \quad NEG(A) = \{j : a_j < 0 \mid j = 1, \ldots, n\}$$

Remark 1 *For any sequence A we have $SUP(A) = POS(A) \bigcup \overline{NEG(A)}$, where by $\overline{S}$ we denote the negated set of S (negating elementwise).*

Two sequences of length n, are said to be of type $(0, \pm 1)$ and weight w if they have a total of w non-zero elements. These pairs will be denoted by $DC(n, w)$ if they have zero PAF. We also denote periodic complementary pairs by $DC(n, w)$. Moreover, pairs of $(0, \pm 1)$ sequences, are also called ternary complementary pairs (TCPs) (sometimes also called aperiodic) when they have zero NPAF and are denoted by $TCP(n, w)$. For more details regarding the theory of DCs and TCPs, we refer to [12] and [2], respectively.

Periodic and non-periodic ternary complementary pairs play a pivotal role in the theory of sequences (see [8, 9]) and their applications are of broader interest. These pairs are used to construct sequences with desirable properties for radar applications (see [16]), and cryptographic systems (see [17]). Moreover, these sequences intervene in coded aperture imaging (see [3]), and higher-dimensional signal processing applications (see [6, 7]). Last we would like to mention that such sequences are interesting objects to study for themselves (see [8, 9, 12]).

In what follows we present the asymptotic analysis for the average case complexity of deciding if two sequences are a TCP or DC pair. We employ a fine grained (low level) analysis initially and then going higher we provide the asymptotics for a pair of sequences of length n. In the course of the analysis we use extensively tools from Computer Algebra and especially tools for dealing with holonomic functions and their asymptotics since such functions occur in the summations needed for the analysis.

In Section 2 we describe the algorithms and the essential part of the theory behind them. In Section 3 we detail the asymptotic analysis for the average case complexity of the algorithms. In Section 4 we conclude by giving some hints on the practical complexity and arguing on the efficiency of the algorithms examined.

2 Combinatorial Algorithms for Periodic and Aperiodic Pairs

In this section, we present a new formalism for the PAF and NPAF of two sequences, based on their support first given in [10] and [11]. The driving force behind this interpretation which led us to formulate the PAF and NPAF on the support of the sequences was the miscarry of the unnecessary multiplications between possible zero elements of the sequences, which take place in the PAF and NPAF.

2.1 Classic Description

If one evaluates the sum in the PAF and NPAF of a sequence A, the following summations are obtained:

$$P_A(s) = \sum_{i=1}^{n} a_i a_{i+s \pmod n} = a_1 a_{1+s \pmod n} + \ldots + a_n a_{s \pmod n} \tag{1}$$

$$N_A(s) = \sum_{i=1}^{n-s} a_i a_{i+s} = a_1 a_{1+s} + \ldots + a_{n-s} a_n \tag{2}$$

where $s = 0, 1, \ldots, n-1$. Henceforth, we are only concerned with candidate $DC(n, w)$ or $TCP(n, w)$ sequences, thus their entries are taken from $\{0, \pm 1\}$. Thus, the pairs $(a_i, a_{i+s \pmod n}) = a_i a_{i+s \pmod n}$ and $(a_i, a_{i+s}) = a_i a_{i+s}$ have possible values from $\{0, \pm 1\}$. We define the PAF and NPAF vectors of the sequences A and B as:

$$\begin{array}{ll} PAF(A) = [P_A(1), \ldots, P_A(n-1)] & PAF(B) = [P_B(1), \ldots, P_B(n-1)] \\ NPAF(A) = [N_A(1), \ldots, N_A(n-1)] & NPAF(B) = [N_B(1), \ldots, N_B(n-1)] \end{array}$$

Therefore, we can decide if the sequences A and B have zero PAF or NPAF from its equivalent vector form $PAF(A) + PAF(B) = 0$ or $NPAF(A) + NPAF(B) = 0$, where with 0 we mean the zero vector of length $n-1$. We note that such pair of sequences have zero PAF or NPAF even though it is the sum of their autocorrelations that is zero.

2.2 Support based

Now, we are only concerned with the nonzero elements of the sequences, i.e. the support. Thus, we demonstrate that all the information required to compute the PAF can be derived as a function of the weight of the two sequences (see also the complexity analysis, in next section).

The following notations and data structures appear to be handy in our effort to express the PAF on the support of two sequences.

It is well known that two binary sequences with zero PAF are equivalent to supplementary difference sets (SDS) (for more details see [5]). Our formalism can also be regarded as a generalization of SDS on three levels $\{0, \pm 1\}$. Following [18, 19] we shall be concerned with lists (multisets), denoted by square brackets ($[\,]$), defined on the fixed group $\mathbb{Z}_n$ of order n, in which repeated elements are counted multiply.

If T_1 and T_2 are two lists then by $T_1 \uplus T_2$ we denote the result of appending the elements of T_1 to T_2 (with multiplicities retained). If the resulting list is sorted, the operation is denoted by $T_1 \& T_2$.

Example 1 $a_1 < a_2 < a_3 \in \mathbb{Z}_n$ *and* $T_1 = [a_1, a_3, a_2]$, $T_2 = [a_2, a_4, a_1]$ *then*

$$T_1 \uplus T_2 = [a_1, a_3, a_2, a_2, a_4, a_1] \text{ and } T_1 \& T_2 = [a_1, a_1, a_2, a_2, a_3, a_4] \tag{3}$$

One natural way to express the operations that occur in PAF and NPAF, when having a representation of the position of the elements in the sequence(s) (i.e. the support), is by signed differences. For each one of the above cases we define collections (multisets) of signed differences. Only the following three cases can occur in the PAF/NPAF defined in the support of A:

(i) Let c_1 be the number of pairs (a_i, a_{i+s}) with $a_i = a_{i+s} = 1$.

(ii) Let c_2 be the number of pairs (a_i, a_{i+s}) with $a_i = a_{i+s} = -1$.

(iii) Let c_3 be the number of pairs (a_i, a_{i+s}) with $a_i a_{i+s} = -1$.

Then by a counting argument we derive that $A_A(s) = c_1 + c_2 - c_3,\ s = 0, 1, \ldots, n-1$, where $A_A(s) = P_A(s)$ or $N_A(s)$, and in $P_A(s)$ we consider (i+s) modulo n.

Notation. Let A be a sequence of length n as above, with entries from $\{0, \pm 1\}$. Three cases are of interest, corresponding to the combinations that can result to ± 1. We use $\pm, \mp$ in the notation to denote the use of $POS(A), NEG(A)$, the index 2 in the multisets represents that we define the differences in two directions ($\rightleftarrows$) due to the periodic property of the autocorrelation function, corresponding to the modulo operation in the index of the elements of the sequence A, while the index 1 in the multisets represents that we define the differences in one direction ($\rightrightarrows$) due to the non-periodic property of the autocorrelation function, corresponding to the index of the elements of the sequence A.

c_1 We define the signed differences in the positive support of A as $D^{+}_{A,2} = [(x-y) \pmod n : x \neq y, x, y \in POS(A)]$ for PAF, while for NPAF we define the signed differences in the positive support of A as $D^{+}_{A,1} = [x - y : x > y \ \wedge \ x, y \in POS(A)]$.

c_2 We define the signed differences in the negative support of A as $D^{-}_{A,2} = [(x-y) \pmod n : x \neq y, x, y \in NEG(A)]$ for PAF, while for NPAF we define the signed differences in the negative support of A as $D^{-}_{A,1} = [x - y : x > y \ \wedge \ x, y \in NEG(A)]$.

c_3 For $a_i a_{i+s \pmod n} = -1$ to occur in PAF we have two cases. $a_i = 1$, $a_{i+s \pmod n} = -1$ and vice versa. Thus, we have to define the cross differences between the positive and negative support of A as $D^{\pm}_{A,2} = [(x-y) \pmod n : x \in POS(A), y \in NEG(A)]$ and $D^{\mp}_{A,2} = [(x-y) \pmod n : x \in NEG(A), y \in POS(A)]$. Since, we count the totality of differences with repetitions in two ways we define $C^{\rightleftarrows}_{A,2} = D^{\pm}_{A,2} \uplus D^{\mp}_{A,2}$. Similarly, for $a_i a_{i+s} = -1$ to occur in NPAF we have two cases. $a_i = 1, a_{i+s} = -1$ and vice versa. Thus, we have to define the cross differences between the positive and negative support of A as $D^{\pm}_{A,1} = [x - y : x > y \ \wedge \ x \in POS(A), y \in NEG(A)]$ and $D^{\mp}_{A,1} = [x - y : x > y \ \wedge \ x \in NEG(A), y \in POS(A)]$. Since, we count the totality of differences with repetitions in one way we define $C^{\rightrightarrows}_{A,1} = D^{\pm}_{A,1} \uplus D^{\mp}_{A,1}$.

We quote the following Lemma from [10, 11], which acts as a criterion to decide if any pair of two sequences has zero PAF or NPAF.

Lemma 1 *Let A, B be two sequences of length n and weight w with entries from $\{0, \pm 1\}$. Then the following are equivalent:*

(i) A, B are $TCP(n, w)$ if and only if $(D^+_{A,1} \uplus D^-_{A,1})\&(D^+_{B,1} \uplus D^-_{B,1}) = C^{\rightrightarrows}_{A,1}\&C^{\rightrightarrows}_{B,1}$

(ii) A, B are $DC(n, w)$ if and only if $(D^+_{A,2} \uplus D^-_{A,2})\&(D^+_{B,2} \uplus D^-_{B,2}) = C^{\rightleftarrows}_{A,2}\&C^{\rightleftarrows}_{B,2}$

2.3 Description of the Algorithms

In this section, we describe an algorithm that decides if two $\{0, \pm 1\}$ sequences have zero PAF/NPAF. In order to compare the efficiency of the proposed formalism we analyze the algorithms involved in the computation of the NPAF. There are three subalgorithms that are needed:

- Representation (Sequences or support)
- Computation (Summations or differences)
- Verification (NPAF vectors or sorting lists)

To analyze an algorithm we determine the number of steps required for the algorithm to execute. Since our problem, involves two parameters for a candidate $DC(n, w)$ or $TCP(n, w)$ pair of sequences, the time complexity will be given as a function of both length n and weight w which are the input parameters in order to obtain a refined analysis.

It is obvious, that in order to compare the efficiency of the algorithms we must compare them for the same representation of input. In our case, the given problem must either be two candidate $DC(n, w)$ and $TCP(n, w)$ or support sets of two sequences. When given candidate $DC(n, w)$ and $TCP(n, w)$ we will refer to our problem, as the *sequence problem*. We note that, we only deal with an analysis in the arithmetic model and not with bit-size complexity, i.e. all arithmetic operations are exact and contribute the same in the complexity of the algorithm.

The following three phases needed for the computation of PAF and NPAF of two sequences and the respective algorithms were given in [10, 11]:

- Sequence to Support (SEQ2SUP) and Support to Sequence (SUP2SEQ) algorithms
- PAF/NPAF Vector (PAFVEC/NPAFVEC) and PAF/NPAF Support (PAFSUP/NPAFSUP) algorithms
- PAF/NPAF Vector Verification (PAFVECVER/NPAFVECVER) and PAF/NPAF Support Verification (PAFSUPVER/NPAFSUPVER) algorithms

A high level view of the previous subalgorithms in one phase of computations can be seen in the following pseudo code for the two cases of the sequence problem.

A refined worst case analysis for the complexity of the previous two algorithms will be given in the next section, needed as the first step for an average case complexity analysis. By the term refined we mean that the analysis is performed by taking into account the weight of each one of the two sequences.

Algorithm 1 (N)PAF Computation Algorithm - `(N)PAFCOMP`

procedure (N)PAFCOMP(A, B)
Require: A, B are two $\{0, \pm 1\}$ sequences of length n
$n \leftarrow |A|$
$(POS(A), NEG(A)) \leftarrow$ SEQ2SUP(A)
$(POS(B), NEG(B)) \leftarrow$ SEQ2SUP(B)
$(D^+_{A,2}, D^-_{A,2}, C^{\rightleftarrows}_{A,2}) \leftarrow$ PAFSUP($POS(A), NEG(A), n$) OR $(D^+_{A,1}, D^-_{A,1}, C^{\rightrightarrows}_{A,1}) \leftarrow$ NPAFSUP($POS(A), NEG(A), n$)
$(D^+_{B,2}, D^-_{B,2}, C^{\rightleftarrows}_{B,2}) \leftarrow$ PAFSUP($POS(B), NEG(B), n$) OR $(D^+_{B,1}, D^-_{B,1}, C^{\rightrightarrows}_{B,1}) \leftarrow$ NPAFSUP($POS(B), NEG(B), n$))
bool $\leftarrow$ PAFSUPVER($D^+_{A,2}, D^-_{A,2}, C^{\rightleftarrows}_{A,2}, D^+_{B,2}, D^-_{B,2}, C^{\rightleftarrows}_{B,2}$) OR *bool* $\leftarrow$ NPAFSUPVER($D^+_{A,1}, D^-_{A,1}, C^{\rightrightarrows}_{A,1}, D^+_{B,1}, D^-_{B,1}, C^{\rightrightarrows}_{B,1}$)
return *bool*
end procedure

3 Average Case Complexity Analysis

In this section we examine the average case complexity of the aforementioned algorithms.

Average-case analysis is a hard task to accomplish because there are a lot of details involved. The basic process begins by determining the different groups into which all possible inputs can be divided. The second step is to determine for each of the groups, the probability for a random input to come from this group. The third step is to determine for each of the groups, the complexity of the algorithm for inputs coming from this group.

Afterwards, the average case complexity is given by the following formula:

$$A(n) = \sum_{i=1}^{m} P_i \cdot T_i \tag{4}$$

where n is the size of the input, m is the number of groups, P_i is the probability assigned to the i-th group, and T_i is the complexity of the algorithms for inputs from the i-th group.

In what follows, we consider as input a pair of sequences. The total length is $2 \cdot n$, the weight of the first sequence is w_1, of the second is w_2 and the total weight is $w = w_1 + w_2$.

We use the notation "a pair of weight (w_1, w_2)" to mean that the first sequence is of weight w_1 and the second of w_2.

3.1 First Level

We start with a fine grain analysis considering pairs of sequences of length n, where the first sequence has weight w_1 and a positive entries and the second sequence has weight w_2 and b positive entries. This is the lowest we can go, in the sense that we do not ignore any information about the non zero entries of the sequences. We take into account the number of positive and negative entries, ignoring the structure.

Let's denote with (s_1, s_2) the support vectors of the two sequences. This means that $|s_1| = w_1$ and $|s_2| = w_2$. In this section we consider classes of sequences of fixed support. We denote the set of pairs of sequences having support (s_1, s_2) by A_{s_1,s_2}. The set of pairs of sequences having support (s_1, s_2), where the first sequence has a positive entries and the second has b positive entries is denoted by $A_{s_1,s_2}(a, b)$.

We will use w_i to denote $|s_i|$ without mentioning from now on. As a side note, the length n is irrelevant in this section.

The cardinality of $A_{s_1,s_2}(a, b)$ is $\binom{w_1}{a} \cdot \binom{w_2}{c}$. We choose a out of the w_1 positions for the positive entries, then the remaining $w_1 - a$ positions are the negative entries. Similarly for the second sequence, we get the second term in the product.

The cardinality of A_{s_1,s_2} is $2^{w_1+w_2}$. We have 2 choices (positive or negative) for each non zero entry.

Thus, the probability of picking at random an $A_{s_1,s_2}(a,b)$ pair out of A_{s_1,s_2} is:

$$p_1(a,b,s_1,s_2) = \frac{\binom{w_1}{a}\binom{w_2}{b}}{2^{w_1+w_2}} \tag{5}$$

We now examine the complexity of the algorithm for an $A_{s_1,s_2}(a,b)$ pair. The algorithm consists of three steps as seen previously. We first compute the amount of operations for each step separately.

The cardinality of the two sets $D = D_A^+ \uplus D_A^- \uplus D_B^+ \uplus D_B^-$ and $C = C_A \uplus C_B$ is $\frac{1}{2}(2 \cdot a^2 + 2 \cdot b^2 - w_1 - 2 \cdot a \cdot w_1 + w_1^2 - w_2 - 2 \cdot b \cdot w_2 + w_2^2)$ and $a \cdot (w_1 - a) + b \cdot (w_2 - b)$ respectively, which by abuse of notation we denote as D and C.

- Sequence to Support (SEQ2SUP)
  This step requires $4 \cdot n$ operations in order to scan the sequences and populate the POS and NEG sets.
- Computation of the signed difference sets (PAFSUP/NPAFSUP)
  This step requires at most 3 operations for each element added in any of the sets $D_A^+, D_A^-, D_B^+, D_B^-, C_A, C_B$. Therefore the total complexity for this step is $3D + 3C$.
- Verification of PAF/NPAF property (PAFSUPVER/NPAFSUPVER)
  1. This step requires $D \log D + C \log C$ steps to sort the two lists.
  2. Another step is needed to check for equality of the elements in the two lists where this step requires $\max\{D, C\} < D + C$ steps to compare the two lists for equality.

Summing up the complexity for the previous three steps we deduce that the total complexity for an $A_{s_1,s_2}(a,b)$ pair is:

$$\begin{aligned} c_1(a,b,s_1,s_2) \quad = \quad & 2\left(w_1(w_1-1) + w_2(w_2-2)\right) \\ & + \tfrac{1}{2}\left(2a(a-w_1) + 2b(b-w_2) + w_1(w_1-1) + w_2(w_2-1)\right) \\ & \times \log\left(\tfrac{1}{2}\left(2a(a-w_1) + 2b(b-w_2) + w_1(w_1-1) + w_2(w_2-1)\right)\right) \\ & + \left(a(w_1-a) + b(w_2-b)\right) \log\left(a(w_1-a) + b(w_2-b)\right). \end{aligned} \tag{6}$$

Multiplying the probability by the complexity, we compute the average complexity for $A_{s_1,s_2}(a,b)$ pairs. More precisely we have that

$$T_1(a,b,s_1,s_2) \quad = \quad p_1(a,b,s_1,s_2) \cdot c_1(a,b,s_1,s_2).$$

In order to get rid of the dependency of the logarithms on a and b we estimate the logands from above using a quick application of cylindrical algebraic decomposition [1, 15],

```
In[1]:= Resolve[ForAll[{a, b}, 0 ≤ a ≤ w1 && 0 ≤ b ≤ w2,
    1/2 (2a(a − w1) + 2b(b − w2) + w1(w1 − 1) + w2(w2 − 1)) ≤ M], {w1, w2, M}, Reals]

Out[1]= M ≥ 1/2 (w1^2 − w1 + w2^2 − w2)

In[2]:= ResolveForAll[{a, b}, 0 ≤ a ≤ w1 && 0 ≤ b ≤ w2, a(w1−a)+b(w2−b) ≤ M], {w1, w2, M}, Reals]
```

Out[2]= $M \geq \frac{1}{4}\left(w_1^2 + w_2^2\right)$

These last estimates in turn can be bounded by n^2 for $n \geq 1$ and so the last two summands stemming from the logarithms can be replaced by the upper bound

$$2^{-w_1-w_2}\binom{w_1}{a}\binom{w_2}{b}(w_1(w_1-1)+w_2(w_2-1))\log(n).$$

With this upper bound summing over a and b ranging in $[0, w_1]$ and $[0, w_2]$ respectively, we obtain for the average complexity T_α for a pair of sequences of weight (w_1, w_2) and length n that

$$T_\alpha(n, w_1, w_2) \leq (w_1(w_1-1)+w_2(w_2-1))\,(\log n + 2).$$

3.2 Second Level

At a second level we examine the complexity ignoring the low level structure of the positive and negative entries. We are concerned about sequences of total weight w. In order to compute the average complexity for these sequences, we need to consider the probability of a pair to have total weight w and combine it with T_α from section 3.1.

The probability for a pair of ternary sequences to have weights w_1 and w_2 is:

$$p_2(n, w_1, w_2) = \frac{2^{w_1+w_2}\binom{n}{w_1}\binom{n}{w_2}}{3^{2\cdot n}} \tag{7}$$

The complete space has $3^{2\cdot n}$ elements (3 choices for each position in the two sequences of length n). We choose the non-zero positions in each sequence and then we have two choices for each of the non-zero positions.

The complexity for a pair of sequences of weight (w_1, w_2) is T_α. Multiplying the upper bound for T_α with the probability p_2, we get:

$$T_2(n, w_1, w_2) = 3^{-2n}2^{w_1+w_2}\binom{n}{w_1}\binom{n}{w_2}(w_1(w_1-1)+w_2(w_2-1))\,(\log n + 2).$$

By summing T_2 over all combinations of w_1 and w_2 for a given $w = w_1 + w_2$, we conclude to the average complexity for a pair of sequences of length n and total weight w,

$$\begin{aligned} T_\beta(n, w) &= \left(3^{-2n}2^{w+1}\sum_{i=0}^{w}\binom{n}{i}\binom{n}{w-i}i^2 - 3^{-2n}2^{w+1}w\sum_{i=0}^{w}\binom{n}{i}\binom{n}{w-i}i\right. \\ &\quad \left. +3^{-2n}2^w(w^2-w)\sum_{i=0}^{w}\binom{n}{i}\binom{n}{w-i}\right)(\log n + 2) \\ &= 3^{-2n}2^w\frac{n-1}{2n-1}w(w-1)\binom{2n}{w}(\log n + 2). \end{aligned}$$

3.3 Third Level

The probability for a pair of sequences to have weight w is:

$$p_3(n, w) = \frac{2^w \cdot \binom{2n}{w}}{3^{2\cdot n}} \tag{8}$$

Choosing w positions out of $2 \cdot n$ and for each of them having two choices for the entry.

The average complexity for a pair of sequences of length n and total weight w is T_β. Multiplying the T_β with p_3 we get the expression, which we split in three parts to make the summation easier, i.e.,

$$\begin{aligned} T_\gamma(n,w) &= T_\beta(n,w)p_3(n,w) \\ &= 3^{-4n}\frac{n-1}{2n-1}\left(2^{2w}w^2\binom{2n}{w}^2 - 2^{2w}w\binom{2n}{w}^2\right)(\log n + 2). \end{aligned}$$

Summing over $w = 0, 1, \ldots, 2 \cdot n$ yields the average complexity of the algorithm. We sum separately the two parts and obtain for $T_\delta(n) = \sum_{w=0}^{n} T_\gamma(n,w)$

$$T_\delta(n) = 3^{-4n}\frac{n-1}{2n-1}16n^2\left({}_2F_1\left(\begin{matrix}-2n+1,-2n+1\\1\end{matrix};4\right) - {}_2F_1\left(\begin{matrix}-2n+1,-2n+1\\2\end{matrix};4\right)\right), \quad (9)$$

where the hypergeometric sums ${}_2F_1$ is defined as

$${}_2F_1\left(\begin{matrix}a,b\\c\end{matrix};z\right) = \sum_{k\geq 0}\frac{(a)_k(b)_k}{(c)_k k!}z^k,$$

with $(a)_k = a(a+1)\ldots(a+k-1)$ denoting the Pochhammer symbol (or rising factorial). Note that the sums appearing above are finite because of the factors $-2n+1$ in the numerator. Hypergeometric sums are classical objects and over the centuries many tricks and treats-of-the-day have been developed to simplify them or find equivalent descriptions such as recurrence relations. Within the last decades several symbolic algorithms have been designed and implemented that are capable of taking over these tasks [14]. We are interested in the asymptotic behavior of the average complexity and to obtain this information we first compute a recurrence relation for the two terms in (9). From this recurrence relation then the asymptotic behavior can easily be determined using standard techniques, see e.g. [4]. For the computation of the recurrence relations we employ the Mathematica package HolonomicFunctions[1] implemented by Christoph Koutschan [13]. The package is loaded in the Mathematica kernel via

```
In[3]:= << HolonomicFunctions.m
        HolonomicFunctions package by Christoph Koutschan, RISC-Linz, Version 0.13 (13.05.2009)
        — Type ?HolonomicFunctions for help
```

To obtain a recurrence for the terms in (9) we first compute the annihilator with respect to shifts in n, where the shift operator is denoted by S_n.

```
In[4]:= ann = Factor[Annihilator[4n^2(n-1)/((2n-1)3^(4n-1)) Hypergeometric2F1[-2n+1,-2n+1,1,4],{S[n]}]]

Out[4]= {81(n-1)n(n+1)^2(2n+3)^2(4n+1)S_n^2 - (n-1)(n+2)^2(2n+1)(4n+3)(49+246n+164n^2)S_n
        + (n+1)^2(n+2)^2(2n-1)(2n+1)(4n+5)}
```

The recurrence for the remaining expression is obtained completely analogously and reads as

$$\begin{aligned} &81(n-1)n(2n+3)^2(20n^2+20n+3)S_n^2 \\ &- 4(n-1)(2n+1)(249+2066n+4313n^2+3280n^3+820n^4)S_n \\ &+ n(n+1)(2n-1)(2n+1)(43+60n+20n^2). \end{aligned}$$

[1] available at `http://www.risc.jku.at/research/combinat/software/HolonomicFunctions/`

The final step consists in determining the asymptotic behavior from these two recurrences. For this purpose we make use of a yet unpublished implementation of the methods described in [4] by Manuel Kauers. For the first recurrence this yields the asymptotic behavior $n^{3/2}(1+3^{-4n})$ and for the second one $n^{1/2}(1+3^{-4n})$.

From this we directly conclude that the average complexity of the algorithm is at most $\mathcal{O}(n\sqrt{n}\log n)$.

4 Conclusion

We presented a detailed analysis for the average case complexity of an algorithm based on a support based formulation deciding if two sequences form a periodic or aperiodic complementary pair. In the course of the analysis we have used Computer Algebra tools and computed the average case asymptotic complexity to be $\mathcal{O}(n\sqrt{n}\log n)$. This estimate proves that in practice the algorithm under consideration performs better than what the worst case complexity ($\mathcal{O}(n^2\log n)$, see [10, 11]) implies.

Acknowledgements

The authors are grateful to Manuel Kauers for providing the asymptotics package. The second and fourth author are supported by the Austrian Science Fund (FWF) grant W1214/DK6. Part of this work was completed while the third author was visiting RISC in Linz. Thanks go to the RISC for the kind hospitality and support from the RISC Transnational Access Programme supported by the European Commission FP6 for Integrated Infrastructures Initiatives under the project SCIEnce.

References

[1] G.E. Collins, Quantifier elimination for real closed fields by cylindrical algebraic decomposition, *Automata theory and formal languages (Second GI Conf., Kaiserslautern, 1975)*, pages 134–183. Lecture Notes in Comput. Sci., Vol. 33. Springer, Berlin, 1975.

[2] R. Craigen and C. Koukouvinos, A theory of ternary complementary pairs, *J. Combin. Theory Ser. A*, 96 (2001), 358–375.

[3] E. Fenimore and T. Cannon, Coded aperture imaging with uniformly redundant arrays, *Appl. Optics*, 17 (1978), 337–347.

[4] P. Flajolet and R. Sedgewick, Analytic combinatorics, Cambridge University Press, Cambridge, 2009.

[5] A. V. Geramita and J. Seberry, *Orthogonal designs. Quadratic forms and Hadamard matrices*, Lecture Notes in Pure and Applied Mathematics, 45, Marcel Dekker Inc. New York, 1979.

[6] S. Golomb and H. Taylor, Two-dimensional synchronization patterns for minimum ambiguity, *IEEE Trans. Inform. Theory*, 28 (1982), 600–604.

[7] J. Hersheya and R. Yarlagadda, Two-dimensional synchronisation, *Electron. Lett.*, 19 (1983), 801–803.

[8] H. Kharaghani and C. Koukouvinos, Complementary, Base and Turyn Sequences, in *Handbook of Combinatorial Designs*, (Eds. C.J. Colbourn and J.H. Dinitz), 2nd ed. Chapman and Hall/CRC Press, Boca Raton, Fla., 2006, pp. 317–321.

[9] C. Koukouvinos, Sequences with Zero Autocorrelation, in *The CRC Handbook of Combinatorial Designs*, (Eds. C. J. Colbourn and J. H. Dinitz), CRC Press, 1996, pp. 452–456.

[10] C. Koukouvinos and D. E. Simos, On the Computation of the Periodic Autocorrelation Function of Two Ternary Sequences and its Related Complexity Analysis, submitted for publication.

[11] C. Koukouvinos and D. E. Simos, On the Computation of the Non-Periodic Autocorrelation Function of Two Ternary Sequences and its Related Complexity Analysis, submitted for publication.

[12] C. Koukouvinos and J. Seberry, New weighing matrices and orthogonal designs constructed using two sequences with zero autocorrelation function-a review, *J. Statist. Plann. Inference*, 81 (1999), 153–182.

[13] C. Koutschan, HolonomicFunctions (User's Guide), *Technical report no. 10-01 in RISC Report Series*, University of Linz, Austria. January 2010.

[14] M. Petkovšek, H.S. Wilf and D. Zeilberger, $A = B$, A K Peters Ltd., Wellesley, MA, 1996.

[15] Adam Strzeboński, Solving systems of strict polynomial inequalities, *Journal of Symbolic Computation*, 29:471–480, 2000.

[16] G. Weathers and E. M. Holiday, Group-complementary array coding for radar clutter rejection, *IEEE Transaction on Aerospace and Electronic Systems*, 19 (1983), 369–379.

[17] B. Schneier, *Applied Cryptography*, 2nd edition, John Wiley and Sons, New York, 1996.

[18] J. Seberry Wallis, On supplementary difference sets, *Aequationes Math.*, 8 (1972), 242–257.

[19] J. Seberry Wallis, A note on supplementary difference sets, *Aequationes Math.*, 10 (1974), 46–49.

Technical Reports of the Doctoral Program "Computational Mathematics"

2010

2010-01 S. Radu, J. Sellers: *Parity Results for Broken k-diamond Partitions and (2k+1)-cores* March 2010. Eds.: P. Paule, V. Pillwein

2010-02 P.G. Gruber: *Adaptive Strategies for High Order FEM in Elastoplasticity* March 2010. Eds.: U. Langer, V. Pillwein

2010-03 Y. Huang, L.X.Châu Ngô: *Rational General Solutions of High Order Non-autonomous ODEs* June 2010. Eds.: F. Winkler, P. Paule

2010-04 S. Beuchler, V. Pillwein, S. Zaglmayr: *Sparsity optimized high order finite element functions for H(div) on simplices* September 2010. Eds.: U. Langer, P. Paule

2010-05 C. Hofreither, U. Langer, C. Pechstein: *Analysis of a non-standard finite element method based on boundary integral operators* September 2010. Eds.: B. Jüttler, J. Schicho

2010-06 M. Hodorog, J. Schicho: *A symbolic-numeric algorithm for genus computation* September 2010. Eds.: B. Jüttler, R. Ramlau

2010-07 M. Hodorog, J. Schicho: *Computational geometry and combinatorial algorithms for the genus computation problem* September 2010. Eds.: B. Jüttler, R. Ramlau

2010-08 C. Koukouvinos, V. Pillwein, D.E. Simos, Z. Zafeirakopoulos: *A Note on the Average Complexity Analysis of the Computation of Periodic and Aperiodic Ternary Complementary Pairs* October 2010. Eds.: P. Paule, J. Schicho

The complete list since 2009 can be found at
`https://www.dk-compmath.jku.at/publications/`

Doctoral Program
"Computational Mathematics"

Director:

Prof. Dr. Peter Paule
Research Institute for Symbolic Computation

Deputy Director:

Prof. Dr. Bert Jüttler
Institute of Applied Geometry

Address:

Johannes Kepler University Linz
Doctoral Program "Computational Mathematics"
Altenbergerstr. 69
A-4040 Linz
Austria
Tel.: ++43 732-2468-7174

E-Mail:

office@dk-compmath.jku.at

Homepage:

http://www.dk-compmath.jku.at

Submissions to the DK-Report Series are sent to two members of the Editorial Board who communicate their decision to the Managing Editor.