Computing Periodic Golay Pairs Using SAT Solvers

Bernhard Andraschko[1], Ilias Kotsireas[2], and Martin Kreuzer[1]

[1] Fakultät für Informatik und Mathematik, Universität Passau, Germany
{bernhard.andraschko,martin.kreuzer}@uni-passau.de
[2] CARGO Lab, Wilfrid Laurier University, Canada
ikotsire@wlu.ca

Abstract. This paper presents a new method for computing periodic Golay pairs. Our main approach is to apply SAT solvers to find complementary sequences in $\{-2, 0, +2\}^{n/2}$ and determine whether those can be lifted to periodic Golay pairs in $\{-1, +1\}^n$. We investigate various ways to express the definition and certain properties of periodic Golay pairs as systems of Boolean polynomials and analyze their impact on the solving times of the resulting satisfiability instances.

Keywords: Periodic Golay pair, compression, complementary sequence, SAT solving

1 Introduction

Golay pairs are tuples (A, B) with $A, B \in \{-1, +1\}^n$ whose aperiodic autocorrelation function values sum to zero for all nontrivial shifts. *Periodic* Golay pairs satisfy the analogous condition for the periodic autocorrelation function (PAF). While ordinary Golay pairs are conjectured to exist only for lengths of the form $2^a10^b26^c$, periodic Golay pairs exist for a broader spectrum of lengths and are equally applicable to digital communication and the construction of Hadamard matrices. The recent paper [13] exhaustively enumerated all periodic Golay pairs of length $n \leq 72$, exhibited an example for $n = 90$, and thereby settled existence for all $n < 106$.

Our approach models PAF and related properties of periodic Golay pairs as Boolean polynomials, converts them to logical formulas with the methods of [1], and solves the resulting systems using a SAT solver. This allows us not only to find periodic Golay pairs, but also to certify non-existence for specific lengths. In particular, we confirm that no periodic Golay pair of length 18 exists and demonstrate that our method scales to lengths up to 40.

2 Periodic Golay Pairs

Let $n \in \mathbb{N}$ and let $A = (A_0, \ldots, A_{n-1})$ and $B = (B_0, \ldots, B_{n-1})$ be elements of $\{-1, +1\}^n$. The *periodic autocorrelation function* is $\mathrm{PAF} : \mathbb{Z}^n \times \mathbb{N} \to \mathbb{Z}$,

$$\mathrm{PAF}\big((x_0, \ldots, x_{n-1}), s\big) = \textstyle\sum_{i=0}^{n-1} x_i x_{i+s \bmod n}.$$

The tuple (A, B) is a **periodic Golay pair** if $\mathrm{PAF}(A, s) + \mathrm{PAF}(B, s) = 0$ for $s = 1, \ldots, n-1$. Golay showed in [10] that periodic Golay pairs exist only for even n. Moreover, $\mathrm{PAF}(X, s) = \mathrm{PAF}(X, s + \frac{n}{2})$ for even n, so it suffices to check $s = 1, \ldots, \frac{n}{2}$. The existence question for general n remains open.

The key to our approach is the following characterization in terms of Hamming weights. For $C \in \mathbb{F}_2^n$, let $\mathrm{wt}(C)$ denote the (*Hamming*) *weight*, i.e., the number of nonzero entries of C.

Proposition 2.1. *For $i, j \in \{0, \ldots, n-1\}$, define $a_{ij}, b_{ij} \in \mathbb{F}_2$ by $a_{ij} = 1$ if and only if $A_iA_j = 1$ and $b_{ij} = 1$ if and only if $B_iB_j = 1$. Then (A, B) is a periodic Golay pair if and only if, for every $s \in \{1, \ldots, \frac{n}{2}\}$, we have*

$$\mathrm{wt}\big((a_{i,s+i \bmod n} \,|\, i = 0, \ldots, n-1) \,\|\, (b_{i,s+i \bmod n} \,|\, i = 0, \ldots, n-1)\big) = n$$

where $\|$ denotes the concatenation of tuples.

3 Boolean Polynomials

Let $P = \mathbb{F}_2[X_1, \ldots, X_n]$ and $\mathbb{I}_n = \langle X_1^2 - X_1, \ldots, X_n^2 - X_n \rangle$. Then $\mathbb{B}_n = P/\mathbb{I}_n$ is the *ring of Boolean polynomials*. We also write $\mathbb{B}[x_1, \ldots, x_n]$ instead of $\mathbb{B}_n$. The residue class of X_i in $\mathbb{B}_n$ is denoted by x_i and called a *Boolean indeterminate*. For details about Boolean polynomials, see [5], [7, Sec. 2], and [11, Sec. 2.10].

Every $f \in \mathbb{B}_n$ has a unique *canonical representative* $F \in P$ that is a sum of square-free terms. Its *zero set* is $\mathcal{Z}(f) = \{c \in \mathbb{F}_2^n \mid F(c) = 0\}$ and for Boolean polynomials $f_1, \ldots, f_s \in \mathbb{B}_n$, we set $\mathcal{Z}(f_1, \ldots, f_s) = \bigcap_{i=1}^{s} \mathcal{Z}(f_i)$.

4 Weight Encodings

Let $n \in \mathbb{N}$, $m = \lfloor \log_2 n \rfloor + 1$, and let $X = (x_1, \ldots, x_n)$ and $Y = (y_0, \ldots, y_{m-1})$ be tuples of Boolean indeterminates. For $c \in \mathbb{N}$, we denote the m-bit binary representation of c by $[c_{m-1} \ldots c_0]_2$ and for $r, s \in \mathbb{N}$, we denote by $\pi_r : \mathbb{F}_2^{r+s} \to \mathbb{F}_2^r$ the projection map onto the first r coordinates. In the following definition, the indeterminates in Z are auxiliary indeterminates.

Definition 4.1. *Let $X' = (x'_1, \ldots, x'_n)$ and $Z = (z_1, \ldots, z_s)$ be tuples of Boolean indeterminates.*

(a) Let $g_1, \ldots, g_k \in \mathbb{B}[X \cup Y \cup Z]$ and $G = \{g_1, \ldots, g_k\}$ such that

$$\pi_{n+m}\big(\mathcal{Z}(G)\big) = \big\{(a, (b_0, \ldots, b_{m-1})) \in \mathbb{F}_2^n \times \mathbb{F}_2^m \mid \mathrm{wt}(a) = [b_{m-1} \ldots b_0]_2\big\}.$$

*Then $\{g_1, \ldots, g_k\}$ is called a **binary weight encoding** of X. The tuple Y is called the tuple of **weight bit indeterminates** of X.*

(b) Let $k_1, \ldots, k_s \in \mathbb{N}$ and $g_1, \ldots, g_k \in \mathbb{B}[X \cup Z]$ such that

$$\pi_n\big(\mathcal{Z}(g_1, \ldots, g_k)\big) = \big\{a \in \mathbb{F}_2^n \mid \mathrm{wt}(a) \in \{k_1, \ldots, k_s\}\big\}.$$

*Then $\{g_1, \ldots, g_k\}$ is called a **weight $\{k_1, \ldots, k_s\}$ encoding of** X.*

(c) Let $g_1, \ldots, g_k \in \mathbb{B}[X \cup X' \cup Z]$ such that

$$\pi_{2n}\big(\mathcal{Z}(g_1, \ldots, g_k)\big) = \big\{(a, b) \in \mathbb{F}_2^n \times \mathbb{F}_2^n \mid \mathrm{wt}(a) = \mathrm{wt}(b)\big\}.$$

*Then $\{g_1, \ldots, g_k\}$ is called a **weight equality encoding of X and X'**.*

As we shall see, all other encodings can be deduced from binary weight encodings. Our first binary weight encoding is the following construction from [6]: for each $i \in \{1, \ldots, n\}$, auxiliary Boolean indeterminates $(y_{i,0}, \ldots, y_{i,m-1})$ represent the weight of $(x_1, \ldots, x_i)$. In particular, $(y_{n,0}, \ldots, y_{n,m-1})$ then encodes the weight of $(x_1, \ldots, x_n)$.

Proposition 4.2 (cf. [6]). *Let $R = \mathbb{B}[x_i, y_{j,k} \mid i, j = 1, \ldots, n,\ k = 0, \ldots, m-1]$. Consider the polynomials $g_{i,j} \in R$ with*

$$\begin{aligned}
g_{1,0} &= y_{1,0} + x_1, && \\
g_{1,j} &= y_{1,j}, && j = 1, \ldots, m-1 \\
g_{i,0} &= y_{i,0} + y_{i-1,0} + x_i, && i = 2, \ldots, n, \\
g_{i,j} &= y_{i,j} + y_{i-1,j} + y_{i-1,j-1}(y_{i,j-1} + 1), && i = 2, \ldots, n,\ j = 1, \ldots, m-1.
\end{aligned}$$

Then $\{g_{i,j} \mid i = 1, \ldots, n,\ j = 0, \ldots, m-1\}$ is a binary weight encoding of $(x_1, \ldots, x_n)$ with weight bit indeterminates $(y_{n,0}, \ldots, y_{n,m-1})$.

Remark 4.3. Alternatively, a binary weight encoding can be derived from Boolean circuits using full and half adders as described in [4, Sec. 3.1]. The implementation details, however, are beyond the scope of this paper.

Next we construct weight $\{k\}$ encodings and then generalize those to weight $\{k_1, \ldots, k_s\}$ encodings. The most basic approach is to fix the output bits of a binary encoding.

Proposition 4.4. *Let $Z = (z_1, \ldots, z_s)$ be a tuple of Boolean indeterminates and let $G \subseteq \mathbb{B}[X \cup Y \cup Z]$ be a binary weight encoding of $(x_1, \ldots, x_n)$ with weight bit indeterminates Y. Let $k \in \{0, \ldots, n\}$ with binary representation $k = [k_{m-1} \cdots k_0]_2$. Then $G \cup \{y_0 - k_0, \ldots, y_{m-1} - k_{m-1}\}$ is a fixed weight $\{k\}$ encoding of $(x_1, \ldots, x_n)$.*

Similar to binary weight encodings, one can use a unary counter: introduce variables $y_{1,i}, \ldots, y_{i,i}$ for $i = 1, \ldots, n$ that represent the weight of $(x_1, \ldots, x_i)$ as a unary number, i.e., if c is a zero of such a system G, then $c(y_{j,i}) = 1$ if and only if $j \leq \mathrm{wt}(c(x_1), \ldots, c(x_i))$. Then adding the polynomials $y_{n,k} + 1$ and $y_{n,k+1}$ to G enforces $\mathrm{wt}(c(x_1), \ldots, c(x_n)) = k$.

Proposition 4.5. *Let $k \in \{0, \ldots, n-1\}$, let $Y = \{y_{i,j} \mid i = 1, \ldots, n,\ j = 1, \ldots, \min\{i, k+1\}\}$ be a set of Boolean indeterminates, and let $R = \mathbb{B}[X \cup Y]$. Consider the polynomials*

$$
\begin{aligned}
g_{1,1} &= y_{1,1} + x_1, \\
g_{i,1} &= y_{i,1} + (y_{i-1,1}+1)(x_i+1) + 1, && i = 2, \ldots, n \\
g_{i,i} &= y_{i,i} + y_{i-1,i-1}x_i, && i = 2, \ldots, k+1 \\
g_{i,j} &= y_{i,j} + (y_{i-1,j}+1)(x_i y_{i-1,j-1}+1)+1, && i = 2, \ldots, n,\ j = 2, \ldots, \min\{i, k+1\} \\
h_1 &= y_{n,k} + 1, \quad h_2 = y_{n,k+1}.
\end{aligned}
$$

Then $G \cup \{h_1, h_2\}$, where $G = \{g_{i,j} \mid i = 1, \ldots, n,\ j = 1, \ldots, \min\{i, k+1\}\}$, is a weight $\{k\}$ encoding of $(x_1, \ldots, x_n)$.

Remark 4.6. In the situation of Proposition 4.5, one can show that the following polynomials are also elements of the ideal $\langle G \cup \{h_1, h_2\}\rangle$:

$$
\begin{aligned}
& y_{i,j}(y_{i+1,j} + 1), && i = 1, \ldots, n-1,\ j = 1, \ldots, \min\{i-1, k+1\}, \\
& (y_{i,j} + 1)y_{i+1,j+1}, && i = 1, \ldots, n-1,\ j = 1, \ldots, \min\{i-2, k\}, \\
& y_{n,j} + 1, && j = 1, \ldots, k, \\
& y_{i,k+1}, && i = k+1, \ldots, n.
\end{aligned}
$$

Adding these polynomials to the set $G \cup \{h_1, h_2\}$ significantly improves SAT-solver propagation.

Finally, we show how to compute weight $\{k_1, \ldots, k_s\}$ encodings from one binary encoding and one weight $\{1\}$ encoding.

Proposition 4.7. *Let $Z = (z_1, \ldots, z_s)$ be a tuple of Boolean indeterminates and let G' be a binary weight encoding of X with weight-bit tuple Y. Let $k_1, \ldots, k_s \in \{0, \ldots, n\}$ be pairwise distinct with $k_i = [k_{i,m-1} \cdots k_{i,0}]_2$, and let H be a weight $\{1\}$ encoding of Z. Then*

$$G = G' \cup H \cup \left\{y_j + \textstyle\sum_{i=1}^{s} k_{i,j} z_i \mid j \in \{0, \ldots, m-1\}\right\}$$

is a weight $\{k_1, \ldots, k_s\}$ encoding of X.

The final proposition of this section shows how to produce a weight equality encoding. For this, let $X' = (x'_1, \ldots, x'_n)$ be a tuple of Boolean indeterminates.

Proposition 4.8. *Let G_X and $G_{X'}$ be binary weight encodings of X and of X', respectively, such that both have the same weight bit indeterminates Y. Then $G = G_X \cup G_{X'}$ is a weight equality encoding of $(x_1, \ldots, x_n)$ and $(x'_1, \ldots, x'_n)$.*

5 Computing Periodic Golay Pairs

In order to apply Proposition 2.1, we model $A_i = \pm 1$ via Boolean variables x_i and the products $A_iA_j = \pm 1$ via $x_{i,j}$. The relationship between these variables is addressed in following lemma.

Lemma 5.1. *Let $A_1, A_2 \in \{-1, 1\}$ and let $a_1, a_2, a_{1,2} \in \mathbb{F}_2$ with $a_i = 1$ if and only if $A_i = 1$ for $i = 1, 2$ and $a_{1,2} = 1$ if and only if $A_1A_2 = 1$. Then $a_{1,2} = a_1 + a_2$.*

Algorithm 5.2 (Periodic Golay pairs). *Let $n \in \mathbb{N}_+$ be even. Consider the following sequence of instructions.*

(1) Let $X = (x_0, \ldots, x_{n-1})$, $Y = (y_0, \ldots, y_{n-1})$, and $Z = (x_{i,j}, y_{i,j} \mid i, j = 0, \ldots, n-1)$ be tuples of Boolean indeterminates.
(2) Let S be the set of $x_{i,j} + x_i + x_j$ and $y_{i,j} + y_i + y_j$ for $i, j \in \{0, \ldots, n-1\}$.
(3) For $s = 1, \ldots, n/2$ do:
  (3.1) Let $I_s = \big((i, (i+s) \bmod n) \mid i = 0, \ldots, n-1\big)$.
  (3.2) Compute a weight $\{n\}$ encoding $G_s \subseteq \mathbb{B}[Z \cup Z_s]$ of the tuple

$$(x_{i,j} \mid (i,j) \in I_s) \| (y_{i,j} \mid (i,j) \in I_s).$$

  (3.3) Add the elements of G_s to S and the elements of Z_s to Z.
(4) Return S.

This is an algorithm which computes a set $S \subseteq \mathbb{B}[X \cup Y \cup Z]$ such that there exists a tuple $c = (c_0, \ldots, c_{n-1}, d_0, \ldots, d_{n-1}, e_1, \ldots, e_k) \in \mathcal{Z}(S)$ if and only if

$$A = \big((-1)^{c_0+1}, \ldots, (-1)^{c_{n-1}+1}\big), \quad B = \big((-1)^{d_0+1}, \ldots, (-1)^{d_{n-1}+1}\big)$$

is a periodic Golay pair. In particular, $\mathcal{Z}(S) \neq \emptyset$ if and only if there exists a periodic Golay pair of length n.

Remark 5.3. Algorithm 5.2 can be improved by adding additional constraints that periodic Golay pairs satisfy. By [8], if (A, B) is a periodic Golay pair and a and b are the respective sums of the entries of A and B, then $a^2 + b^2 = 2n$. Note that, for $a \in \{-n, \ldots, n\}$, $\sum_i A_i = a$ if and only if exactly $\frac{n+a}{2}$ entries of $A = (A_1, \ldots, A_n)$ equal 1, so Algorithm 5.2 can be optimized by adding the following step between Steps (3) and (4):

(3') Compute $W = \{(a, b) \in \mathbb{Z}^2 \mid a^2 + b^2 = 2n\}$, let $W_a = \{\frac{n+a}{2} \mid (a, b) \in W\}$ and $W_b = \{\frac{n+b}{2} \mid (a, b) \in W\}$. Add a weight W_a encoding of $x_0, \ldots, x_{n-1}$ and a weight W_b encoding of $y_0, \ldots, y_{n-1}$ to S.

6 Compression

Let $A = (A_0, \ldots, A_{n-1})$ and $B = (B_0, \ldots, B_{n-1})$ be elements of $\mathbb{Z}^n$.

Definition 6.1. *(A, B) is called a pair of **complementary sequences**, if for every $s \in \{1, \ldots, n/2-1\}$, we have* $\mathrm{PAF}(A, s) + \mathrm{PAF}(B, s) = 0$.

Note that periodic Golay pairs are exactly the pairs of complementary sequences whose entries are in $\{-1, +1\}$.

For a divisor m of n with $d = \frac{n}{m}$, the **m-compression** of A is defined as $A^{(m)} = (A_0^{(m)}, \ldots, A_{d-1}^{(m)})$ where $A_i^{(m)} = A_i + A_{i+d} + \cdots + A_{i+(m-1)d}$. By [9], if (A, B) is a pair of complementary sequences, then so is $(A^{(m)}, B^{(m)})$. Since periodic Golay pairs exist only for even lengths, we can compute complementary sequences (C, D) in $\{-2, 0, +2\}^{n/2}$ and check whether a periodic Golay pair (A, B) with $(A^{(2)}, B^{(2)}) = (C, D)$ exists. Notice that this reduces the search space from 2^n elements of $\{-1, +1\}^n$ to $3^{n/2}$ elements of $\{-2, 0, +2\}^{n/2}$.

Similar to Lemma 5.1, products of entries of C, D satisfy the following.

Lemma 6.2. *Let $C_1, C_2 \in \{-2, 0, 2\}$, let $c_{i0}, c_{i2} \in \mathbb{F}_2$ with $c_{i0} = 0$ iff $C_i = 0$ and $c_{i2} = 1$ iff $C_i = 2$ for $i = 1, 2$. Additionally, let $c_{12}^+, c_{12}^- \in \mathbb{F}_2$ with $c_{12}^+ = 1$ iff $C_1C_2 = 4$ and $c_{12}^- = 1$ iff $C_1C_2 = -4$. Then*

(a) $c_{12}^+ = c_{10}\, c_{20}\, (c_{12} + c_{22} + 1)$, *and*
(b) $c_{12}^- = c_{10}\, c_{20}\, (c_{12} + c_{22})$.

Next comes the analogue of Proposition 2.1 for sequences in $\{-2, 0, +2\}$. It follows from the fact that a sum of elements of $\{+4, 0, -4\}$ evaluates to zero if and only if the number of occurrences of $+4$ matches the number of -4.

Proposition 6.3. *Let $C = (C_0, \ldots, C_{d-1})$ and $D = (D_0, \ldots, D_{d-1})$ be elements of $\{-2, 0, +2\}^d$. For $i, j = 0, \ldots, d-1$ and for $\pm \in \{-, +\}$, let $c_{ij}^{\pm}, d_{ij}^{\pm} \in \mathbb{F}_2$ such that $c_{ij}^{\pm} = 1$ iff $C_iC_j = \pm 4$ and $d_{ij}^{\pm} = 1$ iff $D_iD_j = \pm 4$. For $s \in \{1, \ldots, d/2 - 1\}$, let $I_s = \big((i, s + i \bmod d) \mid i = 0, \ldots, d-1\big)$, and let*

$$\ell_s^+ = \mathrm{wt}\big((c_{ij}^+ \mid (ij) \in I_s) \,\|\, (d_{ij}^+ \mid (ij) \in I_s)\big),$$
$$\ell_s^- = \mathrm{wt}\big((c_{ij}^- \mid (ij) \in I_s) \,\|\, (d_{ij}^- \mid (ij) \in I_s)\big).$$

Then (C, D) is a pair of complementary sequences if and only if $\ell_s^+ = \ell_s^-$ for every $s \in \{1, \ldots, d/2 - 1\}$.

Algorithm 6.4 (Complementary Sequences in $\{-2, 0, +2\}^d$).
Let $d \in \mathbb{N}$. Consider the following sequence of instructions.

(1) Let $X = \{x_{ik} \mid i{=}0, \ldots, d{-}1,\ k{=}0, 2\}$, $Y = \{y_{ik} \mid i{=}0, \ldots, d{-}1,\ k{=}0, 2\}$, and $Z = \{x_{ij}^{\pm}, y_{ij}^{\pm} \mid i, j = 0, \ldots, d-1, \pm \in \{-, +\}\}$ be sets of Boolean indeterminates.
(2) Let S be the set of the polynomials

$$x_{ij}^+ + x_{i0}x_{j0}(x_{i2} + x_{j2}), \qquad x_{ij}^- + x_{i0}x_{j0}(x_{i2} + x_{j2} + 1),$$
$$y_{ij}^+ + y_{i0}y_{j0}(y_{i2} + y_{j2}), \qquad y_{ij}^- + y_{i0}y_{j0}(y_{i2} + y_{j2} + 1),$$

for $i, j = 0, \ldots, d-1$.
(3) For $s = 1, \ldots, d/2$, do:
(3.1) Let $I_s = \{(i, (i + s) \bmod d) \mid i = 0, \ldots, d-1\}$.
(3.2) Compute a weight equality encoding $G_s \subseteq \mathbb{B}[Z \cup Z_s]$ of

$$(x_{ij}^+ | (i, j) \in I_s) \| (y_{ij}^+ | (i, j) \in I_s) \text{ and } (x_{ij}^- | (i, j) \in I_s) \| (y_{ij}^- | (i, j) \in I_s).$$

(3.3) Add the elements of G_s to S and the elements of Z_s to Z.
(4) Return S.

This is an algorithm which computes a set $S \subseteq \mathbb{B}[X \cup Y \cup Z]$ such that there exists $c = (c_{ik} | i = 0, \ldots, d-1,\ k = 0, 2) \| (d_{ik} | i = 0, \ldots, d-1,\ k = 0, 2) \| (e_1, \ldots, e_r)$ in $\mathcal{Z}(S)$ if and only if

$$C = \big(2c_{00}(-1)^{c_{02}+1}, \ldots, 2c_{d-1\,0}(-1)^{c_{d-1\,2}+1}\big),$$
$$D = \big(2d_{00}(-1)^{d_{0,2}+1}, \ldots, 2d_{d-1\,0}(-1)^{d_{d-1\,2}+1}\big)$$

is a pair of complementary sequences over $\{-2, 0, +2\}$. In particular, $\mathcal{Z}(S) \neq \emptyset$ if and only if there is a pair of complementary sequence of length d over $\{-2, 0, +2\}$.

Remark 6.5. Not every pair of complementary sequences in $\{-2, 0, +2\}^{n/2}$ lifts to a periodic Golay pair. However, notice that $\sum_{i=0}^{n-1} A_i = \sum_{i=0}^{n/2-1} A_i^{(2)}$. Therefore, in our implementation of Algorithm 6.4, we added Boolean polynomials to S that ensure that every pair of complementary sequences (C, D) derived from S satisfies $(\sum C)^2 + (\sum D)^2 = 2n$ analogous to Remark 5.3.

Algorithm 6.6 (Lifting Complementary Sequences). *Let $n \in \mathbb{N}$ be even, let $d = \frac{n}{2}$, and let $(C, D) = \big((C_0, \ldots, C_{d-1}), (D_0, \ldots, D_{d-1})\big) \in \{-2, 0, +2\}^d$ be complementary sequences. Consider the following sequence of instructions.*

(1) Let $X = \{x_0, \ldots, x_{n-1}\}$ and $Y = \{y_0, \ldots, y_{n-1}\}$ be sets of Boolean indeterminates, and let S be the output of Algorithm 5.2.
(2) For each $i \in \{0, \ldots, d-1\}$, add the following Boolean polynomials to S:
  – x_i, x_{i+d} if $C_i = 2$; $x_i{+}1, x_{i+d}+1$ if $C_i = -2$; $x_i{+}x_{i+d}{+}1$ if $C_i = 0$;
  – y_i, y_{i+d} if $D_i = 2$; $y_i{+}1, y_{i+d}{+}1$ if $D_i = -2$; $y_i{+}y_{i+d}{+}1$ if $D_i = 0$.
(3) Return S.

This is an algorithm which computes a set $S \subseteq \mathbb{B}[X \cup Y \cup Z]$ such that there exists a tuple $c = (c_0, \ldots, c_{n-1}, d_0, \ldots, d_{n-1}, e_1, \ldots, e_k) \in \mathcal{Z}(S)$ if and only if

$$A = \big((-1)^{c_0+1}, \ldots, (-1)^{c_{n-1}+1}\big), \quad B = \big((-1)^{d_0+1}, \ldots, (-1)^{d_{n-1}+1}\big)$$

is a periodic Golay pair with $A^{(2)} = C$ and $B^{(2)} = D$. In particular, $\mathcal{Z}(S) \neq \emptyset$ if and only if such a Golay pair exists.

Finally, Algorithms 6.4 and 6.6 are combined as follows.

Algorithm 6.7 (Periodic Golay Pairs Using Compression). *Let $n \in \mathbb{N}_+$ be even, and let $d = \frac{n}{2}$. Consider the following sequence of instructions.*

(1) Execute Algorithm 6.4 with input d, and let S be its output. Let $M = \emptyset$.
(2) If $\mathcal{Z}(S) \setminus M = \emptyset$, return "`UNSAT`". Otherwise, compute $c \in \mathcal{Z}(S) \setminus M$ and use it to compute a pair of complementary sequences $(C, D) \in \{-2, 0, +2\}^d$.
(3) Execute Algorithm 6.6 with input (C, D), and let T be its output.
(4) If $\mathcal{Z}(T) = \emptyset$, append c to M and go to Step (2). Otherwise, compute a $c \in \mathcal{Z}(T)$ and use it to compute a periodic Golay pair $(A, B) \in \{-1, +1\}^n$.

This is an algorithm which computes a periodic Golay pair (A, B) of length n if and only if one exists.

7 Results

All algorithms were implemented in `Python`. Using the methods from [1], we converted the polynomial systems to CNF-XOR and to CNF, and solved the resulting formulas with `CryptoMiniSat` [14] and `Kissat` [3].

For all even $n \in \{10, \ldots, 40\}$ such that $a, b \in \mathbb{Z}$ with $a^2 + b^2 = 2n$ exist, we compare the solving time of the polynomial system from Algorithm 5.2 to the running time of Algorithm 6.7. For small n, Algorithm 5.2 dominates, but Algorithm 6.7 shows a significant advantage for $n = 18$ and $n \geq 26$. Also, we observe that no single solver or weight encoding performs best across all tested values of n.

As shown in [2], the cases $n = 18$ and $n = 36$ are UNSAT (no periodic Golay pair exists), which explains the longer solving time. For all other tested n, a pair was found. For $n > 40$, no solver terminated within one day. For every instance, we report only the fastest solver–encoding combination. All experiments were run on an Intel Xeon E5-2623 v3 with 128 GB of RAM under Debian 10 using `CryptoMiniSat` version 5.11.23 and `Kissat` version 4.0.4.

The recent work [13] exhaustively enumerates all periodic Golay pairs up to length 72 in less than 10 CPU years using the concepts of compression and isomorph rejection. Without using either of these techniques, we reach $n = 40$ and verify the non-existence of periodic Golay pairs of length $n = 18$.

The proofs of all propositions and algorithms were omitted here due to space constraints and will be included in the full version of the paper.

Acknowledgments. The first author gratefully acknowledges *Cusanuswerk e. V.* for financial support.

Table 1. Running times of the fastest solver–encoding combination per instance.

n	Method	Running time	Solver	Weight encodings
10	Algorithm 5.2	0.01 s	`CryptoMiniSat`	Remark 4.3 and Proposition 4.4
	Algorithm 6.7	0.26 s	`CryptoMiniSat`	Remark 4.3 and Proposition 4.4
16	Algorithm 5.2	0.10 s	`CryptoMiniSat`	Proposition 4.2 and Proposition 4.5
	Algorithm 6.7	0.35 s	`CryptoMiniSat`	Remark 4.3 and Proposition 4.4
18	Algorithm 5.2	245 s	`Kissat`	Proposition 4.2 and Proposition 4.4
	Algorithm 6.7	9.3 s	`Kissat`	Proposition 4.2 and Proposition 4.4
20	Algorithm 5.2	2.1 s	`Kissat`	Remark 4.3 and Proposition 4.4
	Algorithm 6.7	3.2 s	`CryptoMiniSat`	Remark 4.3 and Proposition 4.4
26	Algorithm 5.2	28 s	`Kissat`	Remark 4.3 and Proposition 4.4
	Algorithm 6.7	3.5 s	`CryptoMiniSat`	Proposition 4.2 and Proposition 4.4
32	Algorithm 5.2	580 s	`Kissat`	Remark 4.3 and Proposition 4.5
	Algorithm 6.7	16 s	`Kissat`	Remark 4.3 and Proposition 4.4
34	Algorithm 5.2	520 s	`Kissat`	Remark 4.3 and Proposition 4.5
	Algorithm 6.7	13 s	`Kissat`	Remark 4.3 and Proposition 4.4
36	Algorithm 5.2	>100.000 s	–	–
	Algorithm 6.7	>100.000 s	–	–
40	Algorithm 5.2	>100.000 s	–	–
	Algorithm 6.7	100 s	`Kissat`	Proposition 4.2 and Proposition 4.4

References

1. Andraschko, B., Danner, J., Kreuzer, M.: SAT solving using XOR-OR-AND normal forms. Math. Comput. Sci. **18** (2024), 1–26.
2. Arasu, K.T., Xiang, Q.: On the existence of periodic complementary binary sequences. Des Codes Crypt **2** (1992), 257-–262.
3. Biere, A., Faller, T., Fazekas, K., Fleury, M., Froleyks, N., Pollitt, F.: CaDiCaL, Gimsatul, IsaSAT and Kissat Entering the SAT Competition 2024. In: Proc. SAT Competition 2024 - Solver, Benchmark and Proof Checker Descriptions, University of Helsinki, Helsinki, 2024, pp. 8–10.
4. Brandão, L.T.A.N., Çalık, Ç., Sönmez Turan, M., Peralta, R.: Upper bounds on the multiplicative complexity of symmetric Boolean functions. Cryptogr. Commun. **11** (2019), 1339–1362.
5. Brickenstein, M.: *Boolean Gröbner Bases: Theory, Algorithms and Applications*, Logos Verlag, Berlin, 2010.
6. Caminata, A., Cartor, R., Meneghetti, A., Mora, R., Pellegrini, A.: Quadratic Modelings of Syndrome Decoding. In: Post-Quantum Cryptography (PQCrypto 2025). LNCS **15577**, Springer (2025), pp. 35–70.
7. Danner, J., *SAT Solving Using XOR-OR-AND Normal Forms and Cryptographic Fault Attacks*, Dissertation, University of Passau, Passau 2025.
8. Đoković, D.Ž., Kotsireas, I.: Some new periodic Golay pairs. Numerical Algorithms **69** (2015), 523–530.
9. Đoković, D.Ž., Kotsireas, I.: Compression of periodic complementary sequences and applications. Des. Codes Cryptogr. **74** (2015), 365–377.
10. Golay, M.J.E.: Complementary Series. IRE Trans. Inf. Theory **7** (1961), 82–87.
11. Horáček, J.: *Algebraic and logic solving methods for cryptanalysis*, Dissertation, University of Passau, Passau 2020.
12. Jovanovic, P., Kreuzer, M.: Algebraic Attacks Using SAT-Solvers. Groups Complex. Cryptol. **2** (2010), 247–259.
13. Lumsden, T., Kotsireas, I., Bright, C.: New Results on Periodic Golay Pairs. Math. Comp. **95** (2026), 1517–1539.
14. Soos, M., Nohl, K., Castelluccia, C.: Extending SAT solvers to cryptographic problems. In: Theory and Applications of Satisfiability Testing (SAT 2009), LNCS 5584, Springer-Verlag, Berlin 2009, 244–257.