J. COMBIN. MATH. COMBIN. COMPUT. 126 (2025) 317–338

Journal of Combinatorial Mathematics
and Combinatorial Computing
www.combinatorialpress.com/jcmcc

Combinatorial Press

Constructing strong starters of orders $3p$: triplication with SAT solver

Oleg Ogandzhanyants[1], Sergey Sadov[2], Margo Kondratieva[1,✉]

[1] *Dept. of Mathematics and Statistics, Memorial University, St. John's NL, A1C 5S7, Canada*
[2] *Private Math School, Moscow, Russia*

ABSTRACT

A novel approach to building strong starters in cyclic groups of orders n divisible by 3 from starters of smaller orders is presented. A strong starter in $\mathbb{Z}_n$ (n odd) is a partition of the set $\{1, 2, \ldots, n-1\}$ into pairs $\{a_i, b_i\}$ such that all pair sums $a_i + b_i$ are distinct and nonzero modulo n and all differences $\pm(a_i - b_i)$ are distinct and nonzero modulo n. A special interest to strong starters of odd orders divisible by 3 is motivated by Horton's conjecture, which claims that such starters exist (except when $n = 3$ or 9) but remains unproven since 1989. We begin with a starter of order p coprime with 3 and describe an algorithm to obtain a Sudoku-type problem modulo 3 whose solution, if exists, yields a strong starter of order $3p$. The process leading from the original to the final starter is called *triplication*. Besides theoretical aspects of the construction, practicality of this approach is demonstrated. A general-purpose constraint-satisfaction (SAT) solver z3 is used to solve the Sudoku-type problem; various performance statistics are presented.

Keywords: strong starter, SAT solver, triplication

1. Introduction

We discuss a new approach for constructing strong starters (special type of combinatorial designs) in the case of orders n divisible by 3, where their existence has not been asserted theoretically.

A *starter* S in an additive abelian group G of odd order n is a partition of the set $G^* = G \setminus \{0\}$ into $k = \frac{n-1}{2}$ pairs $\{\{a_i, b_i\}\}_{i=1}^k$ such that the differences $\{\pm(a_i - b_i), i = 1, ..., k\}$, comprise G^*. See [1] for a general reference.

Starters exist in any additive abelian group of odd order $n \geq 3$. For example, the partition

✉ Corresponding author.
E-mail address: mkondra@mun.ca (M. Kondratieva).

Received 06 June 2025; Revised 10 July 2025; Accepted 23 July 2025; Published 24 August 2025.

DOI: 10.61091/jcmcc126-23

$\{\{x,-x\} \mid x \in G^*\}$ of G^* is a starter in G.

For a starter S in G of order $n = 2k+1$, consider the set of sums in pairs, $\text{sums}(S) = \{a_i + b_i \mid \{a_i, b_i\} \in S\}$. If $0 \notin \text{sums}(S)$ and $|\text{sums}(S)| = k$, the starter S is called *strong*.

Any abelian group G of odd order $n \geq 7$ for which $3 \nmid n$ admits a strong starter [3]. Horton [3] put forward a conjecture, supported by computational evidence, to the effect that strong starters exist in all abelian groups of odd orders except a few groups of orders $n = 3, 5, 9$, for which non-existence is easily verified.

In this paper, G will always be a cyclic group $\mathbb{Z}_n = \mathbb{Z}/n\mathbb{Z}$, its elements identified with integers in the range 0 to $n-1$. We will for brevity say "S is a starter of order n" meaning a starter in $\mathbb{Z}_n$.

The question of the existence of strong starters in $\mathbb{Z}_n$ with n divisible by 3 is only partially settled. According to Horton's summary in [3], theoretical results supplemented by Dinitz and Stinson's computations [2, 6] guarantee the existence of strong starters of odd orders $n = 3^r p$, $r > 0$, only if either $n < 1000$, $n \notin \{3, 9\}$, or if $\gcd(p, 3) = 1$, $p > 5$, and $3^r < 1000$, $3^r \notin \{3, 9\}$.[1]

In particular, if $n = 3p > 1000$, where p is coprime with 3, the existence of strong starters of order n is not asserted in literature.

This gap in theoretical knowledge, which, according to a recent survey by Stinson [7], apparently still persisted in 2022, stimulates interest to ways of obtaining starters of orders divisible by 3.

We will describe a seemingly novel method of construction of strong starters in $\mathbb{Z}_n$, where $n = 3p$, $\gcd(3, p) = 1$. We call it *triplication* (of a starter). The method takes a starter of order p as a "base" and reduces the problem of construction of a strong starter of order $3p$ to a certain system of linear equations and inequalities modulo 3. Below, we call it a *Sudoku-type problem* mod 3, or a *modular Sudoku problem*. The required starter of order $3p$ is recovered from the base starter's extension (defined in our method) and a solution of the modular Sudoku problem by means of the Chinese Remainder Theorem (CRT).

The said modular Sudoku problem is a new object of research, which invites such natural questions as a possibility to prove the existence of a solution theoretically and numerical explorations. Here, as a first demonstration of the practicality of the new method with minimal programming effort, we present an approach relying on a general purpose constraint-satisfaction (SAT) solver.[2] Specifically, the SAT library `z3` by Microsoft was used — for no other reason than its ready availability and a simple Python interface.

We plan to describe a specialized algorithm for solving the modular Sudoku problem, which by far outperforms the SAT solver, in a separate paper.

Our construction does not require the base starter to be strong. However, there is an advantage of taking a strong starter as a base: the modular Sudoku problem is better understood theoretically and a simple necessary solvability condition is available, which is conjecturally sufficient.

[1] Our recent computations, to be described in a separate paper, extend the upper boundary in the inequalities for n or 3^r from 1000 to $2^{16} - 1$.

[2] SAT solver is a computer program which aims to solve the satisfiability problem (SAT). On input of a set of equations, a SAT solver outputs whether the formulas are satisfiable. "Since the introduction of algorithms for SAT in the 1960s, modern SAT solvers have grown into complex software artifacts involving a large number of heuristics and program optimizations to work efficiently."[8]

2. Ingredients of triplication by example

Let us consider a strong starter of the least possible order $p = 7$,

$$T = \{\{2,\ 3\},\ \{4,\ 6\},\ \{5,\ 1\}\}.$$

Indeed, T is clearly a 2-partition of $\mathbb{Z}_7^*$; it is a starter because $\{\pm(2-3),\ \pm(4-6),\ \pm(1-5)\} \bmod 7 = \{\pm 1,\ \pm 2,\ \pm 3\} = Z_7^*$. It is strong: $\mathrm{sums}(T) = \{2+3,\ 4+6,\ 1+5\} = \{5,\ 3,\ 6\} \subset \mathbb{Z}_7^*$ and $|\mathrm{sums}(T)| = (7-1)/2 = 3$.

In order to motivate the triplication method, we will scrutinize a final answer — a starter S of order 21, which in the actual course of events would have been reconstructed from T and a solution of the appropriate Sudoku-type mod 3 problem. But here, we work backwards, from S to T. In a series of exercises, we will exhibit all types of constraints embedded in the modular Sudoku problem.

Well, so by an act of black magic we've got a strong starter S of order 21:

$$\begin{aligned} S \ = \ & \{\{11,\ 18\},\ \{9,\ 17\},\{13,\ 14\},\ \{8,\ 2\},\ \{4,20\}, \\ & \{15,3\},\{5,\ 7\},\ \{1,\ 12\},\ \{19,\ 16\},\ \{6,\ 10\}\}. \end{aligned}$$

Consider two modular reductions of S. The set of pairs of S (or *pairing* for short) mod 3 is

$$\Sigma_3 = \{\{2,\ 0\},\ \{0,\ 2\},\{1,\ 2\},\ \{2,\ 2\},\ \{1,2\},\ \{0,0\},\ \{2,\ 1\},\ \{1,\ 0\},\ \{1,\ 1\},\ \{0,\ 1\}\}.$$

The pairing mod 7 corresponding to S is

$$\Sigma_7 = \{\{4,\ 4\},\ \{2,\ 3\},\{6,\ 0\},\ \{1,\ 2\},\ \{4,6\},\ \{1,3\},\ \{5,\ 0\},\ \{1,\ 5\},\ \{5,\ 2\},\ \{6,\ 3\}\}.$$

The starter S can be uniquely reconstructed from Σ_3 and Σ_7 by CRT.

It is convenient to write the pairings Σ_3 and Σ_7 in table format as follows:

Σ_3

	2, 0	
0, 2	1, 2	2, 2
1, 2	0, 0	2, 1
1, 0	1, 1	0, 1

Σ_7

	4, 4	
2, 3	6, 0	1, 2
4, 6	1, 3	5, 0
1, 5	5, 2	6, 3

From now on, we treat Σ_3 and Σ_7 as pairings or as tables as convenient in the context.

Note that the first column of the table Σ_7, read top to bottom, is that very starter T of order 7 seen earlier in this section.

There is no analog of this fact pertaining to the table Σ_3, since the set $\mathbb{Z}_3^* = \{1, 2\}$ is too small to be partitioned into 3 pairs.

The nature of the 2nd and 3rd columns of the table Σ_7 remains at this stage a mystery.

Leaving a formal description of relations between the elements in each table and relations between the elements in different tables to later sections, we recommend the reader or his/her (grand)child to do some simple arithmetical exercises. (Familiarity with elementary modular arithmetic is required.)

(a1) Check that all differences mod 3 in each row of the table Σ_3 are distinct (e.g. $0-2 \equiv 1 \bmod 3, 1-2 \equiv 2 \bmod 3, 2-2 \equiv 0 \bmod 3$).

(a2) Check that all differences mod 7 in each row of the table Σ_7 are equal (e.g. $2-3 \equiv 6-0 \equiv 1-2 \bmod 7$).

(b0) Calculate all pair sums modulo 3 for the table Σ_3 and all pair sums modulo 7 for the table Σ_7.

(b1) Find a cell in the table Σ_7 with sum $= 0 \bmod 7$ and check that in the corresponding cell of the table Σ_3 the sum mod 3 is nonzero.

(b2) Check that whenever two cell sums mod 7 in the table Σ_7 are equal, the corresponding cell sums mod 3 in the table Σ_3 are distinct. For example, the cells $(2,3)$ and $(5,0)$ have equal pair sums mod 7. The corresponding cells in the table Σ_3 are $(0,2)$ and $(2,1)$, and their sums mod 3 are distinct.

(c1) Note the positions of zeros in the table Σ_7 and observe that the corresponding places in the table Σ_3 are occupied by two distinct nonzero values, 1 and 2.

(c2) For each $c \in \{1,\dots,6\}$, note that the number c appears in exactly three places in the table Σ_7, and the corresponding positions in the table Σ_3 are occupied by three distinct numbers $0,1,2$.

(d) The structure of the table Σ_7 is a cryptographic challenge of a sort. Alice wants to pass three messages represented by pairs in the first column to her party Bob by radio. The messages are not for prying ears. So Alice encrypts them before going live. She sends one message on Monday, repeats it on Tuesday; a new message on Wednesday, repeated on Thursday; then a message on Friday, repeated on Saturday. On odd days, the encrypted message aired by Alice and received by Bob and also by the curious interceptor Cindy is represented by the pair in the second column (same row as the original message); on even days, the encryption is the pair in the third column. This past week was lucky for Cindy as she had an agent in Alice's camp, through which she came in possession of all the original messages as well as the cipher key, the number repeated twice in the uppermost cell of the table (in our case, the number 4). Since Saturday night, the agent's fate has been unknown and there is no warranty that Cindy ever has such a luxury again. She will still intercept encrypted messages aired by Alice. The cipher key is likely to change next week. Being in Cindy's shoes, could you figure out the encryption scheme and continue to eavesdrop Alice's secrets?

(e) There are strong starters of order 21 other than S whose reduction modulo 7 coincides with Σ_7. For example,

$$S' = \{\{4,18\},\ \{16,10\},\ \{20,7\},\ \{8,9\},\ \{11,13\},\ \{1,17\},\ \{5,14\},\ \{15,19\},\ \{12,2\},\ \{6,3\}\}.$$

The reduction of S' modulo 3,

$$\Sigma_3' = \{\{1,0\},\ \{1,1\},\ \{2,1\},\ \{2,0\},\ \{2,1\},\ \{1,2\},\ \{2,2\},\ \{0,1\},\ \{0,2\},\ \{0,0\}\},$$

is different from Σ_3 — this had to be expected as otherwise, by CRT, S and S' would be identical.

Whether or not the reader had patience to do all or some of the proposed exercises, we summarize essential points to be learned from this example. (Replace 7 by p and 21 by $3p$ for a general picture. Here $p \geq 7$, coprime to 6.)

(1) The resulting starter S of order 21 can be uniquely recovered from the two pairings Σ_3 and Σ_7.

(2) The first column of the table Σ_7 is a copy of the "base starter" T of order 7.

(3) The pair on top of the table Σ_7 consists of two identical values. Such a value, or "key", is another "free parameter" of the method (along with base starter).

(4) The second and third columns of the table Σ_7 are uniquely determined by the base starter and the key. The readers who figured out exercise (d) know the formulas; others will find them in the next section.

(5) While the order of elements within pairs in the definition of a starter is unimportant, our method requires ordered pairs. Indeed, the relations between the tables for Σ_3 and Σ_7 dealt with in exercises (c1,2) depend on specific order of elements in pairs.

(6) The table modulo 3 is not arbitrary; its values must satisfy certain constraints. The constraints can be split into three groups (cf. exercises of groups a,b,c). A common feature of these constraints can be stated as follows. Whenever an *equality* relation of certain special type between elements in the table modulo 7 is observed, the corresponding elements in the first table must satisfy an *inequality* relation ($\neq$) of similar arithmetical nature.

(7) The table Σ_3 is not to be expected to be computable easily and by straightforward formulas given the outlined set-up. An admissible filling of Σ_3 with numbers $0, 1, 2$, if it exists, is not to be expected to be unique. Rather, the problem of determination of Σ_3 is a constraint-satisfaction problem modulo 3.

(8) Finally, we note that there exist[3] strong starters of order 15, but none of them can be obtained from the only starter of order 5 by triplication.

The question whether a certain strong starter of order $3p$ can be obtained by triplication from a starter of order p, strong or not, is more systematically addressed at the end of Section 7.

The rest of the paper is organized as follows: In Section 3, a construction of the table Σ_p is revealed (that is, a solution to exercise (d)). Then we define two auxiliary objects derived from Σ_p: *weak sets* (Section 4) and *monochrome sets* (Section 5). In Section 6, the modular Sudoku problem is set up: the table Σ_3 of unknowns mod 3 is introduced and the constraints imposed on the unknowns are formally described in terms of weak sets and monochrome sets. These constraints generalize properties observed in the exercises (a1), (b1), (b2), (c1), (c2) above. Section 7 is devoted to rigorous theory: we prove that a solution of the modular Sudoku problem yields a strong starter of order $3p$ (Theorem 7.1), establish a necessary condition of solvability (Theorem 7.2), and observe an involutive symmetry of the set of solutions (Theorem 7.4). Also, we mention a way (sufficient condition) to recognize that a given strong starter of order $3p$ cannot be obtained by triplication. Finally, in Section 8 we report on our experience of solving the modular Sudoku problem with help of the SAT solver `z3` and make some comments on comparative performance.

3. The triplication table Σ_p

The construction to be described takes two input parameters:

- a starter T of odd order p coprime with 3;
- a number $t \in \{0, 1, \dots, p-1\}$, which will be called the *key*.

The number of pairs in T will be denoted

$$q = \frac{p-1}{2}.$$

Hereinafter we treat T as an *ordered* tuple of *ordered* pairs (so that pairs and their elements can be unambiguously indexed). To emphasize this notationally, we will from now on use parentheses as

[3] precisely, 32 strong starters, or 4 equivalence classes [4, 5].

delimiters for pairs: (x, y), and square brackets as delimiters for pairings: $[(x_0, y_0), (x_1, y_1), (x_2, y_2)]$ etc. Index values will begin at 0 or 1 as convenient. An abbreviated notation will be used, where the above pairing will be written as $[(x_i, y_i)]_{i=0}^{2}$ or even as $[(x_i, y_i)]$ if the index range is clear from the context.

Allowing a slight abuse of notation, we will denote by Σ_p two closely related but not altogether identical objects: a tuple (linearly arranged array) consisting of $3q + 1$ ordered pairs and a corresponding table where the same pairs are arranged in a two-dimensional layout.

The tuple Σ_p will be called the *extension* of the base starter T and the table Σ_p will be called the *triplication table.*

3.1. Construction

Given a tuple

$$T = [(x_1, y_1), (x_2, y_2), \ldots, (x_q, y_q)],$$

we first form the auxilliary tuples

$$T_t^+ = [(t + x_i,\ t + y_i)] \mod p,$$

and

$$T_t^- = [(t - y_i,\ t - x_i)] \mod p.$$

In order to obtain the *table* Σ_p, we place the tuples T, T_t^+, T_t^- in the first, second, and third columns, respectively. The pair (t, t) is placed on the top of the 2nd column of the table. The result is shown below.

		$t,$	t		
$x_1,$	y_1	$t + x_1,$	$t + y_1$	$t - y_1,$	$t - x_1$
$x_2,$	y_2	$t + x_2,$	$t + y_2$	$t - y_2,$	$t - x_2$
$x_3,$	y_3	$t + x_3,$	$t + y_3$	$t - y_3,$	$t - x_3$
...,	...	...,	...	...,	...
$x_q,$	y_q	$t + x_q,$	$t + y_q$	$t - y_q,$	$t - x_q$

The top row of the table consists of just one pair (t, t), to be called the *top pair.* All other rows contain three pairs each and will be referred to as *regular rows* of the table Σ_p.

The *extension tuple*

$$\Sigma_p = [(u_i, v_i)]_{i=0}^{3q},$$

of length $3q + 1$ is obtained by reading the elements of the *table* Σ_p row by row, left to right. For example, if $p = 7$, then $q = 3$, the extension tuple is

$$\Sigma_7 = [(u_0, v_0), (u_1, v_1), \ldots, (u_9, v_9)],$$

where u_i, v_i are the elements of the table. Let us exhibit the table in this new notation, indicating the linear index of each nonempty cell in parentheses:

			(0)	$u_0,$	v_0			
(1)	$u_1,$	v_1	(2)	$u_2,$	v_2	(3)	$u_3,$	v_3
(4)	$u_4,$	v_4	(5)	$u_5,$	v_6	(6)	$u_6,$	v_6
(7)	$u_7,$	v_7	(8)	$u_8,$	v_8	(9)	$u_9,$	v_9

The first formal statement concerning properties of Σ_p is a generalization of exercise (a2) of Section 2. Consider the differences

$$\delta_i = u_i - v_i \mod p.$$

Observe that $\delta_0 = t - t = 0$.

Lemma 3.1 (Row property of the table Σ_p)**.** *The pairs in the same regular row of the table Σ_p have the same difference. The set of these common differences is the same* (mod p) *as the set of pair differences in the tuple T.*

Proof. The i-th row ($1 \le i \le q$) of the table Σ_p consists of the pairs

$$(u_{3i-2}, v_{3i-2}) = (x_i, y_i),\ (u_{3i-1}, v_{3i-1}) = (t + x_i, t + y_i),\ (u_{3i}, v_{3i}) = (t - y_i, t - x_i).$$

The differences δ_j, $3i - 2 \le j \le 3i$, have the common value $d_i = x_i - y_i$. □

Corollary 3.2. *If the tuple T used in Construction is a starter, then distinct regular rows of Σ_p have distinct, non-zero differences* (mod p).

Example 3.3 (Main Example)**.** Take the starter $T = \{\{2,\ 3\}, \{4,\ 6\}, \{1,\ 5\}\}$ of order 7 (the same as in Sec. 2). Take $t = 1$.[4] The triplication table Σ_7 is as follows:

			(0)	1,	1			
(1)	2,	3	(2)	3,	4	(3)	5,	6
(4)	4,	6	(5)	5,	0	(6)	2,	4
(7)	1,	5	(8)	2,	6	(9)	3,	0

Here $\delta_0 = 0,\ \delta_1 = \delta_2 = \delta_3 = 6,\ \delta_4 = \delta_5 = \delta_6 = 5,\ \delta_7 = \delta_8 = \delta_9 = 3$ (mod 7).

This example will be continued and appended in the next two sections.

4. Weak sets

In this section we deal with pairs' sums modulo p in Σ_p (treated as a triplication table or as an extension of starter T)

$$\sigma_i = u_i + v_i \mod p, \quad i = 0, \dots, 3q.$$

Definition 4.1. Let W_s be the set of all pairs $\{(u_\ell, v_\ell)\}$ from Σ_p with sum $\sigma_\ell = s$. The set W_s is called *weak set* in Σ_p if either $s = 0$ or $|W_s| > 1$.

It follows from the definition that different weak sets are disjoint. A pair (u_i, v_i) is a *weak pair* if it belongs to one of the weak sets. A pair that is not weak is called *strong*.

The terminology is motivated by the fact that if Σ_p (as a tuple of pairs) is the reduction modulo p of a strong starter S of order $3p$, then strong pairs in Σ_p, and only they, have unique nonzero sums modulo p.

[4] This is an arbitrary choice. The construction can be carried out for any t, but the solvability of the obtained Sudoku depends on t — see Theorem 7.2 of Sec. 7.

We say that W is a weak set of type r if $|W| = r$.

A weak set of type 1 (if it exists) consists of a single pair with sum 0 modulo p.

Lemma 4.2 (Weak sets are small). *If T is a strong starter, then $|W| \leq 3$ for all weak sets W in Σ_p obtained by Construction.*

Proof. We want to show that there are no weak sets of type > 3. A weak set W of type > 3 must contain two different pairs in the same column of the table Σ_p. It is easy to see that this cannot happen.

In column 1: if (x_i, y_i) and (x_j, y_j) are in W, then $\sigma_i = \sigma_j$ — impossible since T is strong.

Coincidences of pair sums in the regular rows of columns 2 and 3 are excluded similarly.

It remains to consider the case when one of the pairs in W is the top pair (t, t) with sum $2t$. Suppose there exists a pair from T_t^+ (that is, from the second column and a regular row) with the same sum $2t$ mod p. We have $(x_i + t) + (y_i + t) = 2t \bmod p$. Hence $x_i + y_i = 0 \bmod p$, — but this is forbidden by the definition of a strong starter. □

Lemma 4.3 (Two zero sums at most). *If T is a strong starter, then in Σ_p obtained by Construction there are no more than two weak pairs with sum 0 mod p (0-weak pairs): $|W_0| \leq 2$.*

Proof. The pair sums in the first column, T, of the table Σ_p are nonzero by the definition of a strong starter.

If there is a 0-weak pair in a regular row i of the second column, then $0 = \sigma_{3i-1} = \sigma_{3i-2} + 2t \bmod p$. Hence $\sigma_{3i-2} = -2t$.

If there is a 0-weak pair in a regular row j of the third column, then $0 = \sigma_{3j} = 2t - \sigma_{3j-2} \bmod p$. Hence $\sigma_{3j-2} = 2t$.

Since all pair sums in T are nonzero and distinct, in the case $t \neq 0$ there is at most one 0-weak pair in the second column and at most one in the third column.

If $t = 0$, then the only 0-weak pair in the table is the top pair $(0, 0)$. □

Example 4.4 (Main Example (continued 3.3)). The following table shows weak sets in the triplication table given at the end of Section 3.

Cell indexes	2	4, 9	1, 5	6, 7
Weak set	$\{(3,4)\}$	$\{(4,6),\ (3,0)\}$	$\{(2,3),\ (5,0)\}$	$\{(2,4),\ (1,5)\}$
Sum mod 7	0	3	5	6
Type	1	2	2	2

5. Monochrome sets

In this section we work not with pairs from Σ_p but with individual numerical entries. We think of their values as of colors.

In Section 3 we introduced the notation u_i, v_i for the entries of tuple Σ_p. It will be convenient to indicate the position of each entry by two numbers written in the angle brackets: $\langle i, \ell \rangle$, where $0 \leq i \leq q$ is the index of the pair in Σ_p and $\ell = 0, 1$ is the position of the entry within the pair:

$$\langle i, 0 \rangle \text{ for } u_i, \qquad \langle i, 1 \rangle \text{ for } v_i, \quad 0 \leq i \leq q.$$

Definition 5.1. For $c \in \{0, 1, \ldots, p-1\}$, the *monochrome set* of color c is the set of positions $\langle i, \ell \rangle$ of entries in Σ_p with value c. We denote this set by M_c.

Lemma 5.2. *The monochrome set of color $c \neq 0$ has cardinality* 3. *The monochrome set of color 0 has cardinality* 2. *That is,*

$$|M_c| = \begin{cases} 3\,, c \neq 0, \\ 2,\ c = 0. \end{cases}$$

Proof easily follows from Construction; the cases $t = 0$ and $t \neq 0$ need to be treated separately.

This Lemma shows that most monochrome sets are triples. The two-element set of color 0 is a "degenerate triple", but later we will artificially complete it to a full triple.

Example 5.3 (Main Example (continued 3.3))**.** The following table lists all monochrome triples in the triplication table given at the end of Section 3. For instance, the value 0 appears in pairs 5 and 9, both times in the second position ($v_5 = v_9 = 0$), which yields the set of color zero: $\{\langle 5, 1\rangle, \langle 9, 1\rangle\}$.

color	monochrome sets of positions $\langle i, \ell \rangle$	variables
0	$\langle 5, 1\rangle$, $\langle 9, 1\rangle$	v_5, v_9
1	$\langle 0, 0\rangle$, $\langle 0, 1\rangle$, $\langle 7, 0\rangle$	u_0, v_0, u_7
2	$\langle 1, 0\rangle$, $\langle 6, 0\rangle$, $\langle 8, 0\rangle$	u_1, u_6, u_8
3	$\langle 1, 1\rangle$, $\langle 2, 0\rangle$, $\langle 9, 0\rangle$	v_1, u_2, u_9
4	$\langle 2, 1\rangle$, $\langle 4, 0\rangle$, $\langle 6, 1\rangle$	v_2, u_4, v_6
5	$\langle 3, 0\rangle$, $\langle 5, 0\rangle$, $\langle 7, 1\rangle$	u_3, u_5, v_7
6	$\langle 3, 1\rangle$, $\langle 4, 1\rangle$, $\langle 8, 1\rangle$	v_3, v_4, v_8

We will return to Main Example in Section 8 and use it in a demo Python program (see Appendix).

6. Setting up Sudoku-type problem modulo 3

The tuple of pairs $\Sigma_p = \{(u_i, v_i)\}_{i=0}^{3q}$ is determined by Construction described in Section 3. For a given numerical dataset (base starter T and key t), its elements are integer numbers modulo p.

We introduce a tuple of pairs $\Sigma_3 = \{(U_i, V_i)\}_{i=0}^{3q}$. Its elements are thus far indeterminates. Our task will be to find their numerical values as *integers modulo 3*.

Hence the initial, by-definition type of constraint:

1. *Range constraints.* All variables U_i, V_i take values in the set $\{0, 1, 2\}$.

   Let us arrange the set of unknowns $\{U_i, V_i\}$ in a table format as in the triplication table Σ_p. The so obtained arrangement will be referred to as the *table* Σ_3.

   Three classes of constraints are imposed on the entries of the table Σ_3:

2. *Row Constraints.* We require that the differences $D_i = U_i - V_i$ in each regular row of the table be distinct modulo 3, that is,

$$\{D_{3i-2}, D_{3i-1}, D_{3i}\} = \{0, 1, 2\}, \quad i = 1, 2, \ldots, q.$$

3. *Weak Set Constraints.* Every weak set of the triplication table Σ_p gives rise to one constraint.

   Let $\{(u_i, v_i)\}$ be the set of pairs (with the common value of the sums $\sigma_i = s$) comprising the given weak set W_s. We denote the corresponding set of indexes i by $\mathrm{ind}(W_s)$.

The form of the constraint is slightly different for W_s with $s = 0$ and $s \neq 0$.

(a) In the case $s \neq 0$, it is required that the sums $\{U_i + V_i,\ i \in \mathrm{ind}(W_s)\}$, be distinct. Since there are 3 distinct values modulo 3, this requirement by itself is not forbidding provided $|W_s| \leq 3$ for all $s \neq 0$. By Lemma 4.2, this is certainly so if T is strong.

(b) In the case $s = 0$, it is required that the sums $\{U_i + V_i,\ i \in \mathrm{ind}(W_0)\}$ be distinct and nonzero. This requirement is not forbidding provided $|W_0| \leq 2$. By Lemma 4.3, this is certainly so if T is strong.

4. *Color Constraints.* We augment the two-element monochrome set M_0 of color 0 by a dummy double index $\langle *, * \rangle$. We append a dummy variable $Z = 0$ to the set of unknowns $\{U_i, V_i\}$ and declare that the index (position) of Z in the table Σ_3 is $\langle *, * \rangle$.

   To each color $0 \leq c \leq p-1$, there corresponds one constraint: the values (mod 3) of the variables in positions $\langle i, \ell \rangle \in M_c$ must be distinct.

   The constraint corresponding to color 0, formulated without dummies, says that the two variables in positions $\langle i, \ell \rangle \in M_0$ must take distinct values 1 and 2.

7. Theoretical results

Theorem 7.1 (Solution of the modular Sudoku problem yields a strong starter)**.** *Let $\Sigma_p = \{u_i, v_i\}_{i=0}^{3q}$ be a tuple obtained by Construction from a strong starter T of order $p = 2q+1$. Suppose $\{U_i, V_i\}_{i=0}^{3q}$ is the set of values modulo 3 satisfying the constraints (1)–(4) of Section 6 and associated with Σ_p.*

For $i = 0, \ldots, 3q$, let a_i, b_i be the values modulo $3p$ satisfying the congruences $a_i \equiv u_i \bmod p$, $a_i \equiv U_i \bmod 3$, $b_i \equiv v_i \bmod p$, $b_i \equiv V_i \bmod 3$. (The solutions of the congruences are determined by CRT.) Then the set of pairs $S = \{\{a_i, b_i\}\}_{i=0}^{3q}$ is a strong starter in $\mathbb{Z}_{3p}$.

Proof. The constructed pairing S has $3q+1$ pairs. Since the starter T is of order $p = 2q+1$, the order of S is $n = 2(3q+1)+1 = 6p+3 = 3(2q+1) = 3p$.

We have to check that:

(a) S is a partition of the set $\mathbb{Z}_{3p}^*$;

(b) The set of differences of pairs in S, $\{\pm(a_i - b_i) \bmod 3p\}_{i=0}^{3q}$, coincides with $\mathbb{Z}_{3p}^*$.

(c) The pair sums $\{a_i + b_i\}_{i=0}^{3q}$ are all nonzero and distinct.

Property (a) follows from Color Constraints: any two values from the list $a_0, b_0, \ldots, a_{3q}, b_{3q}$ either are different modulo p or, if not, — they are different modulo 3.

Property (b) is ensured by Row Constraints with a little help from Color Constraints. For pairs in regular rows, we have $a_i - b_i \equiv d_i \bmod p$, $a_i - b_i \equiv D_i \bmod 3$. By our set-up, since T is a starter, the pairs (d_i, D_i) are all distinct, $d_i \not\equiv 0$, and $d_i \not\equiv -d_j \bmod p$ for all i, j. It remains to look at the difference $a_0 - b_0$. We have $a_0 - b_0 \equiv t - t = 0 \bmod p$, $a_0 - b_0 \equiv D_0 \neq 0 \bmod 3$, since the positions $\langle 0, 0 \rangle$ and $\langle 0, 1 \rangle$ belong to the monochrome triple of color t.

Property (c) follows from Weak Set Constraints. They imply that the pairs $((a_i + b_i) \bmod p,\ (a_i + b_i) \bmod 3)$ are all distinct, hence the sums $(a_i + b_i) \bmod 3p$ are all distinct.

It remains to check that there are no zero sums modulo $3p$. But if $a_i + b_i \equiv 0 \bmod p$, then the pair (u_i, v_i) in Σ_p is weak. By item (b) in the description of Weak Set Constraints, the corresponding sum $(U_i + V_i) \bmod 3 \in \{1, 2\}$, hence $(a_i + b_i) \bmod 3p \neq 0$. □

Theorem 7.2 (Necessary condition for Sudoku solvability). *If the value of key t in Construction of Section 3 is either 0 or one of the pair sums x_i+y_i of the base starter T (not necessarily strong), then the modular Sudoku problem does not have a solution. In other words, the condition $t \notin \mathrm{sums}(T)\cup\{0\}$ is necessary for solvability of the modular Sudoku problem.*

Proof. In the case $t = 0$, we have two equal pairs in every regular row of the triplication table: $(u_{3i-2}, v_{3i-2}) = (u_{3i-1}, v_{3i-1})$ (both pairs coincide with pair (x_i, y_i) of the base starter T).

In the case where $(x_i + y_i) \bmod p = t$ for some i, we again have two equal pairs in one of the rows (i-th) of the triplication table: $(u_{3i-2}, v_{3i-2}) = (u_{3i}, v_{3i})$. Indeed, calculating in $\mathbb{Z}_p$, we get $u_{3i} = t - y_i = (x_i + y_i) - y_i = x_i = u_{3i-2}$ and similarly $v_{3i} = y_i = v_{3i-2}$.

It remains to prove that the existence of two identical pairs $(u, v) = (u', v')$ in the same row of the table Σ_p is forbidding for the existence of a solution of the modular Sudoku problem.

The elements of pairs (U, V) and (U', V') of the table Σ_3 must satisfy the constraints:

(1) $U - V \not\equiv U' - V' \bmod 3$ (part of Row Constraints);

(2) $U + V \not\equiv U' + V' \bmod 3$ (part of Weak Set Constraints);

(3) $U \neq U'$, $V \neq V'$ (by Color Constraints: the positions of the elements U, U' belong to the same monochrome set of color u and the positions of the elements V, V' belong to the same monochrome set of color v).

The constraints (2) and (3) imply: $U - U' \not\equiv \pm(V - V') \bmod 3$. Thus, if $V - V' \not\equiv 0 \bmod 3$, then $(U - U') \bmod 3 \notin \{1, 2\}$, hence $(U - U') \bmod 3 = 0$.

We see that either $U = U'$ or $V = V'$. Either case violates one of the constraints (4). □

In all our test examples (see Section 8) where the above necessary condition (Theorem 7.2) was met, the modular Sudoku problem did have a solution.

Therefore we propose

Conjecture 7.3. *If T is a strong starter of odd order $p \geq 7$ coprime with 3 and $t \in \mathbb{Z}_n^* \setminus \mathrm{sums}(T)$, where $\mathrm{sums}(T)$ is the set of pair sums of T, then the modular Sudoku problem corresponding to the data (T, t) has a solution.*

Theorem 7.4 (Solutions of Sudoku come in pairs). *Let ϕ be a permutation of the set $\{0, 1, 2\}$ that transposes 1 and 2. If the set of values $\{U_i, V_i\}$ satisfies the constraints of the modular Sudoku problem, then the set of values $\{\phi(U_i), \phi(V_i)\}$ also satisfies those constraints.*

Proof. Every constraint is invariant under transposition of values 1 and 2. □

Remark 7.5. The permutation ϕ is an automorphism of the group $\mathbb{Z}_3$: $\phi(X) = -X \bmod 3$. Moreover, the map $(a \bmod p,\ A \bmod 3) \mapsto (a \bmod p,\ \phi(A \bmod 3))$ is an automorphism of the group $\mathbb{Z}_{3p} \cong \mathbb{Z}_p \oplus \mathbb{Z}_3$ that fixes the direct summand $\mathbb{Z}_p$. From this point of view, Theorem 7.4 is a simple consequence of a well-known partition of starters into equivalent classes (see e.g. Introduction in [5]).

Let us address the question mentioned in Section 2, item 8.

Question 7.6 (Inverse problem of triplication). *Given a strong starter $S = \{\{a_i, b_i\}\}_{i=0}^{3q}$, determine whether S can be obtained by triplication from some pair (T, t), where T is a starter of order $p = 2q+1$*

and $t \in \mathbb{Z}_p^* \setminus \mathrm{sums}(T)$.

We offer a partial answer — a sufficient condition that S is not a result of triplication. The condition is based solely on the row structure of the table Σ_p.[5]

It is convenient to formulate the condition in the form of a test algorithm with two return values: *False* (meaning that S is not a result of triplication) and *Inconclusive*.

Step 1. Reduce the given starter S modulo p; denote the result by $S \bmod p$.

Step 2. For each $d \in \{0, \dots, q\}$ identify the set R_d ("row with difference d") of *ordered* pairs (a, b) such that $\{a, b\} \in S \bmod p$ and $(a - b) \bmod p = d$. Since S is a starter, we have $|R_0| = 1$ and $|R_d| = 3$ for $d \neq 0$.

Step 3. $R_0 = \{(t, t)\}$ is a singleton, call t the "key" and continue.

Step 4. For every set R_d, $d \neq 0$, check whether a certain ordering of R_d is a valid row of a triplication table, that is, has the form

$$(u, v),\ (u', v'),\ (u'', v'')$$

with

$$u' + v'' \equiv v' + u'' \equiv 2t \pmod p, \qquad u' - u \equiv v' - v \equiv t \pmod p.$$

For that purpose, examine all 6 permutations of the pairs in R_d. If no permutation satisfies the stated property, stop and return *False*.

Step 5. If all rows R_d, $d = 1, \dots, q$, pass the test of Step 4, return *Inconclusive*.

The proposed test is simple yet practical; see Discussion at the end of Section 8.

Example 7.7. Consider the starter S of order 21:

$$\{\{13, 12\}, \{19, 17\}, \{7, 4\}, \{10, 14\}, \{15, 20\}, \{3, 9\}, \{1, 8\}, \{5, 18\}, \{11, 2\}, \{16, 6\}\}.$$

The sets R_d modulo 7 relevant for our purposes are: $R_0 = \{(1, 1)\}$ (comes from pair $\{1, 8\}$ and determines the key value $t = 1$) and $R_3 = \{(0, 4), (3, 0), (2, 6)\}$ (comes from pairs $\{7, 4\}$, $\{10, 14\}$, $\{16, 6\}$).

The set R_3 fails the condition of Step 4 of the test. Hence S is not obtainable by triplication.

Example 7.8. Here we illustrate a possibility that S can be obtained by triplication from a starter but not from a strong starter.

Consider the strong starter S of order 39:

$$\{\{2, 1\}, \{36, 34\}, \{6, 3\}, \{8, 12\}, \{38, 33\}, \{16, 10\}, \{18, 11\}, \{29, 21\}, \{22, 31\}, \{25, 15\},$$
$$\{13, 24\}, \{20, 32\}, \{30, 17\}, \{23, 37\}, \{19, 4\}, \{28, 5\}, \{26, 9\}, \{14, 35\}, \{7, 27\}\}.$$

Our test applied to the starter S returns *Inconclusive*. We leave it to the reader to see that the only pairing T of order 13 that can be triplicated to S (with key $t = 4$) is

$$T = \{(11, 10), (6, 4), (2, 12), (9, 5), (8, 3), (7, 1)\}.$$

It is a starter but not a strong starter.

The latter starter T also gives a counterexample to a strengthened version of Conjecture 7.3, where the requirement that the base starter be strong is dropped. In this case, $3 \notin \mathrm{sums}(T)$, yet

[5] To be appreciated by Cindy of exercise (d), Section 2.

the modular Sudoku problem obtained from the data $(T,\, t=3)$ does not have a solution. To see it, note that there are 4 pairs with sum $\equiv 1 \pmod{13}$ in the triplication table Σ_{13}. The existence of a Sudoku solution and therefore of a corresponding strong starter contradicts Lemma 4.2.

8. Computations and discussion

About the program

We implemented the described method of starter triplication in Python (Python 3.11.5) using SAT solver library `z3` (z3-solver 4.15.0.0).

Programming the constraints defining the modular Sudoku problem is straightforward with functions of `z3` library. For clarity of code, along with unknowns of interest, U_i, V_i $(i = 0, \dots, 3q)$, the auxiliary unknowns are used in the code:

- "differences" D_i $(i = 0, \dots, 3q)$ with constraints $0 \le D_i \le 2$, $D_i = U_i - V_i \pmod 3$,
- "weak sums" S_i $(i \in W = \cup \mathrm{ind}(W_s)$, where W_s are the weak sets) with constraints $0 \le S_i \le 2$, $S_i = U_i + V_i \pmod 3$,
- the "dummy variable" Z with fixed value $Z = 0$.

Below we explain in detail how the `z3` solver is programmatically set up in the special case of Main Example considered in Sections 3–5. The full listing of a working Python program for this example is given in Appendix.

In general, the base strong starter and the value of the key are variable parameters; the computation of weak sets (Section 4) and monochrome sets (Section 5) should be automated. This has been done — for computations reported later in this section. Since that part of code is independent of any particular method or algorithm for solving the modular Sudoku problem, it is not presented here.

Another solver-independent part is a utility that merges the triplication table Σ_p and the found solution modulo 3 in table Σ_3 into a starter of order $3p$ using CRT. For that purpose, in the demo program in Appendix, a programming trick using Python's 'dictionary' data structure is employed. For instance, the "label" $(5, 1)$ in the dictionary points to the value 19, which is congruent to 5 mod 7 and to 1 mod 3. A second starter (see Theorem 7.4) is recovered similarly from the same solution of Sudoku.

8.1. Setting up the solver for Main Example

We refer to Main Example followed through in Sections 3–5.

Recall that the order of the base starter is $p = 7$, the order of the resulting starter is $3p = 21$; the parameter $q = (p-1)/2 = 3$.

The unknowns are: U_i, V_i, D_i $(i = 0, \dots, 9)$, S_i $(i \in W,\, W = \{1, 2, 4, 5, 6, 7, 9\}$ — set of indexes of weak pairs), and Z.

Here is the complete list of constraints defining the Sudoku-type problem modulo 3, as they pertain to the data in Example. Instead of the mathematical inequality sign "$\neq$", we use the keyword `Distinct` defined in the `z3` library.

0.0) Dummy variable: $Z = 0$

0.1) Binding main unknowns and differences: $U_i - V_i - D_i = 0 \bmod 3$ for $0 \le i \le 9$.

0.2) Binding main unknowns and sums: $U_i + V_i - S_i = 0 \bmod 3$ for $i \in W = \{1, 2, 4, 5, 6, 7, 9\}$.

1) Range Constraints: $0 \le U_i \le 2$, $0 \le V_i \le 2$, $0 \le D_i \le 2$ for $0 \le i \le 9$; $0 \le S_i \le 2$ for

$i \in W$.

2) Row Constraints:

$\texttt{Distinct}(D_{3i-2}, D_{3i-1}, D_{3i})$ for $1 \le i \le 3$.

3.1) Weak Set Constraint for weak pair with sum 0:

$S_2 \neq 0 \bmod 3$ or, equivalently:

$\texttt{Distinct}(S_2, Z)$.

3.2) Weak Set Constraints involving weak pairs with nonzero sums:

$\texttt{Distinct}(S_i, S_j)$ for (i, j) in $\{(4, 9), (1, 5), (6, 7)\}$.

4) Color Constraints:

$\texttt{Distinct}(V_5, V_9, Z)$ (for degenerate monochrome set M_0),

$\texttt{Distinct}(U_0, V_0, U_7)$,

$\texttt{Distinct}(U_1, U_6, U_8)$,

$\texttt{Distinct}(V_1, U_2, U_9)$,

$\texttt{Distinct}(V_2, U_4, V_6)$,

$\texttt{Distinct}(U_3, U_5, V_7)$,

$\texttt{Distinct}(V_3, V_4, V_8)$.

8.2. Technical details

Computations were carried out in the Jupiter Notebook environment on a fanless mini PC with 11th Gen Intel(R) Core(TM) i7-1165G7 2.80GHz processor and 32 GB memory under Windows 11 operating system. The task was run on one of the four CPU kernels and, according to occasional checks with Task Manager, occupied between 90% and 100% of that kernel's load. Certain issues beyond our control and observation could potentially affect relative performance at different stages of computation — such as system's self-maintenance and the CPU heat control.

Our judgement about solver's performance in different series of experiments relies on recorded elapsed times. Time benchmarks, of course, depend on system's particulars. Possibly, benchmarks such as the number of solver's cycles might give more objective and invariant information about the behavior of the algorithm. But the solver's inner statistics was not available to us.

8.3. Specification of numerical experiments

Base starters for the numerical experiments were generated in advance using a hill-climbing algorithm of Dinitz and Stinson [2, 6].

Performance results for three series of experiments are reported below.

• Series 1: *Base starters of orders* < 100*, all admissible key values.*

For odd orders m from 7 to 97 coprime with 3, a fixed strong starter T_m of order m was taken as a base starter. For every key value satisfying the necessary condition $t \in \mathbb{Z}_m^* \setminus \text{sums}(T_m)$ (Theorem 7.2), the modular Sudoku problem was set up and solved.

• Series 2: *Base starters of larger orders, random admissible key values.*

The set of orders in this series is the union of the following:

- all odd orders m from 101 to 199 coprime with 3;
- all orders in arithmetical progression with difference 12 from 203 to 335;
- orders 395, 439, 499.

For every order from this list, a random key value $t \in \mathbb{Z}_m^* \setminus \text{sums}(T_m)$ was chosen, the modular Sudoku problem was set up and solved.

• Series 3: *Fixed order $m = 101$, all admissible key values, multiple runs.*

A fixed strong starter T_m of order $m = 101$ (same for the whole series) was taken as a base starter. In a subseries, the modular Sudoku problem was solved for every key value $t \in \mathbb{Z}_m^* \setminus \text{sums}(T_m)$. The subseries were executed ten times.

The measure of performance is execution time in seconds, denoted by τ.

8.4. Numerical experiments — Series 1

Average time (in seconds) per Sudoku is shown in Table 1. Here, m is the order of a base strong starter T_m, averaging is done over all $(m-1)/2$ admissible values of key t.

Table 1. Average execution time for solving Sudoku of orders < 100

m	time	m	time	m	time	m	time
7	0.065	29	5.276	53	21.35	77	53.07
9	–	31	5.216	55	24.92	79	52.23
11	0.118	35	8.741	59	26.09	83	78.46
13	0.309	37	8.396	61	33.90	85	70.72
17	1.017	41	10.51	65	35.91	89	73.39
19	1.362	43	11.96	67	40.50	91	78.64
23	2.674	47	17.09	71	49.75	95	93.73
25	3.817	49	20.55	73	48.98	97	90.14

For most values of m, a significant spread between the minimum and maximum values of time was observed: generally, about 2 times. Excluding $m = 7$, the maximum observed spread was 3.5 times for $m = 19$ and the minimum observed ratio was 1.32 for $m = 37$.

The data are plotted in Figure 1 together with heuristic interpolation of the average time given by the formula

$$\tau_1(m) = 0.023 \cdot m^2 \ln m.$$

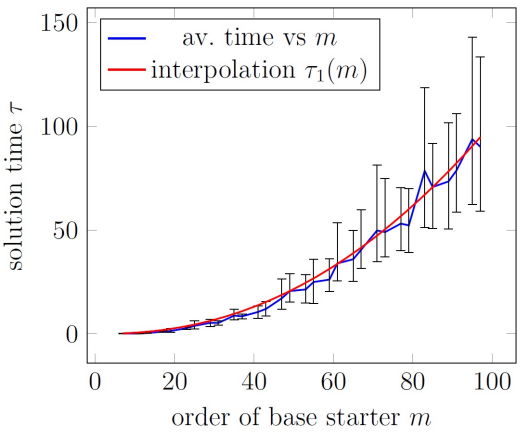

Fig. 1. Execution time for solving Sudoku of orders < 100

8.5. Numerical experiments — Series 2

Time (in seconds) per Sudoku is shown in Table 2. Here, m is the order of a base strong starter T_m. A single computation was done for every order. For formatting reasons, the table does not include

data for orders $m \in (100, 200)$ with $m \bmod 6 = 1$. However, those data are included in the plot shown in Figure 2.

Table 2. Execution time for solving Sudoku of orders between 100 and 500

m	time	m	time	m	time	m	time
101	124	149	374	197	629	287	2267
107	94	155	337	203	735	299	4785
113	120	161	468	215	985	311	3464
119	153	167	306	227	1586	323	3384
125	253	173	485	239	1280	335	4737
131	199	179	407	251	2750	395	17344
137	232	185	670	263	2033	439	20788
143	251	191	593	275	1879	499	17962

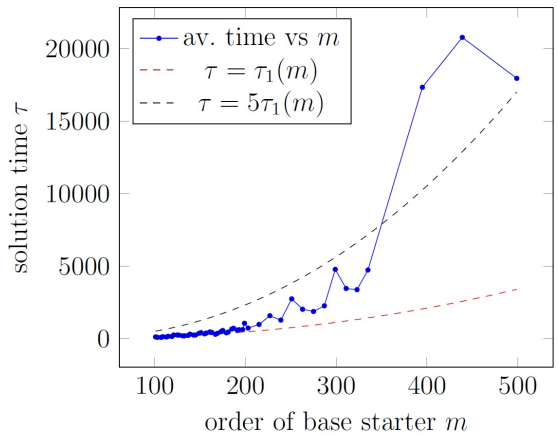

Fig. 2. Execution time for solving Sudoku of orders between 100 and 500

We observe that for $m \lesssim 200$, time values fit rather well in the heuristic "law" $\tau \approx \tau_1(m)$ seen in the discussion of Series 1.

For larger values of m, that approximation is no longer good. For comparison, we plotted the curve (black) $\tau = 5\tau_1(m)$.

We note that the observed time values do not change monotonically with order. There is a dramatic jump between the points with $m = 335$ and $m = 395$ having no analog in other explored intervals of the m axis.

A natural question arises: do the results agree with or testify against the hypothesis about polynomial bound of time required (with high probability) for solving the modular Sudoku problem by a SAT solver?

Althought the hypothesis looks counterintuitive, we see no clear evidence against it in the obtained data.

8.6. Numerical experiments — Series 3

The goal of this series was to explore variations of solution time τ between different runs of the solver with identical input data (T, t) and between different values of key t (for the same starter T) averaged over several runs.

For each of 50 admissible (i.e. satisfying the necessary conditions of Theorem 7.2) values of t, the same Sudoku problem was solved 10 times. We call a sequence of 50 trials for different admissible t (ran consecutively in real time) a subseries. The program was thus executed 500 times in total, split into 10 subseries.

The extreme values of time τ (in seconds) occurred:

- Near-minimum $\tau = 82.98$ in subseries 2 for key $t = 30$,
- Minimum $\tau = 82.95$ in subseries 2 for key $t = 62$,
- Maximum $\tau = 260.7$ in subseries 7 for key $t = 10$,
- Near-maximum $\tau = 257.4$ in subseries 10 for key $t = 10$.

No uniformity was observed in the distribution of execution times over different values of keys, t. It appears that for some key values the triplication process completes systematically quicker than for others. The histogram shown in Figure 3 illustrates this thesis.

For each key value, the execution times were averaged over 10 trials. The averages ranged from 106.9 sec for $t = 42$ to 201.6 sec for $t = 10$. The average times were grouped into ranges of length 10 and, for every such range, the number of keys with average times in that range was found.

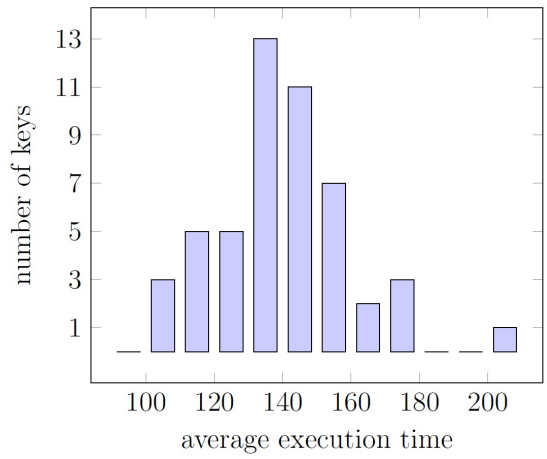

Fig. 3. Distribution of keys in the execution time intervals for a Sudoku of order 101

The final plot shown in Figure 4 illustrates chaotic oscillations of the solution time around a trendline.

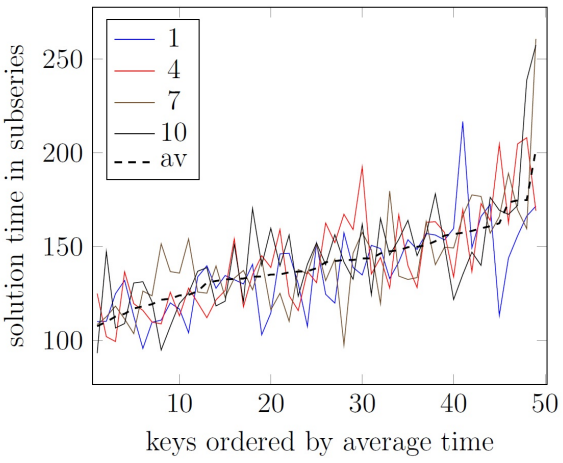

Fig. 4. Oscillations of the solution time around a trendline in four subseries

First, we order the keys so that to have average time (over 10 subseries) to grow monotonically with the index (with respect to the new ordering) of the key.

After that, we plot solution time vs the above-defined index of the key for several subseries; to avoid excessive cluttering, we selected the subseries with numbers 1, 4, 7, 10. The dashed line is the "trendline" — time averaged over all subseries for the given key.

8.7. Discussion

1. A SAT solver can be employed directly for finding strong starters. However, longer time is required for finding a strong starter even for small orders and the time grows much faster as the order increases.

   For comparison, using `z3` solver directly, computation of a strong starter of order 39 took about 200 seconds on the same system, while triplication of a starter of order 13 required less that 1 second.

2. On the other hand, generating strong starters by hill-climbing algorithm of Dinitz and Stinson is much faster: generation of a sampling of 1000 strong starters of order 39 takes less than 2.5 sec.

3. We also note that we (OO) solved modular Sudoku problems of orders $3p$ by hand for p up to 47. The time per problem in the upper range of p was about an hour. The employed approach was, in all likelihood, very different from the internal workings of a universal SAT solver. We expect to formalize the process; a specialized solution algorithm is expected to significantly outperform the universal solver; it is to be compared with hill-climbing approach.

4. Our final remark is unrelated to efficiency of solvers but concerns properties of strong starters that distinguish "triplicated" starters in the totality of all strong starters of the given order $3p$. We report a small, very preliminary, yet perhaps an indicative result in this direction. We sampled[6] (by hill-climbing) 10^6 strong starters of each of the orders 21, 33 and 39 and applied the test described at the end of Section 7 to examine them. Only a small fraction of those "random" strong starters were found "inconclusive". All others failed the test. Specifically, out of 10^6 generated strong starters of order 21 there were about 1.8% "inconclusive", of order 33 – only 11 starters, and of order 39 — none. The starter of order 39 in Example 7.8 was found by generating "random" strong starters until the one passing the test was found; it happened after more than 9 million iterations.

   The reported experimental studies allow extensions and ramifications in various directions. One interesting possible direction is to measure a correlation between triplication runtime, on the one hand, and the partition structure (the distribution of cardinalities) of weak sets for different values of key for the same base starter.

Acknowledgements

The authors are grateful to Prof. N. Shalaby of MUN for attracting our attention to this problem and many discussions that inspired this research. We thank Dr. G.V. Kalachev of MSU for introducing us to SAT solvers.

[6] Repetititions were allowed and, for order 21, inevitable: there are only 6660 different strong starters of order 21 [4, 5].

References

[1] J. H. Dinitz. *Starters*. In *Handbook of Combinatorial Designs, 2nd ed.* C. Colbourn and J. Dinitz, editors. Chapman & Hall/CRC, Boca Raton - London - New York, 2007. Chapter 55, pages 622–628. https://doi.org/10.1201/9781420010541.

[2] J. H. Dinitz and D. R. Stinson. A fast algorithm for finding strong starters. *SIAM Journal on Algebraic and Discrete Methods*, 2(1):50–56, 1981. https://doi.org/10.1137/0602007.

[3] J. D. Horton. Orthogonal starters in finite abelian groups. *Discrete Mathematics*, 79(3):265–278, 1989/90. https://doi.org/10.1016/0012-365X(90)90335-F.

[4] W. L. Kocay, D. R. Stinson, and S. A. Vanstone. On strong starters in cyclic groups. *Discrete Mathematics*, 56(1):45–60, 1985. https://doi.org/10.1016/0012-365X(85)90191-8.

[5] V. Linja-aho and P. R. Östergård. Classification of starters. *Journal of Combinatorial Mathematics and Combinatorial Computing*, 075:153–159, 2010.

[6] D. R. Stinson. Hill-climbing algorithms for the construction of combinatorial designs. In C. J. Colbourn and M. J. Colbourn, editors, *Algorithms in Combinatorial Design Theory*. Volume 114, North-Holland Math. Stud. Pages 321–334. North-Holland, Amsterdam, 1985. https://doi.org/10.1016/S0304-0208(08)72988-8.

[7] D. R. Stinson. Orthogonal and strong frame starters: revisited. In C. J. Colbourn and J. H. Dinitz, editors, *New Advances in Designs, Codes and Cryptography*, volume 86 of *Fields Institute Communications*. Springer, Cham., 2024. https://doi.org/10.1007/978-3-031-48679-1_21. NADCC 2022.

[8] Wikipedia, The Free Encyclopedia. SAT solver. https://en.wikipedia.org/w/index.php?title=SAT_solver&oldid=1291806065. Accessed on 2025-05-26.

Appendix: Listing of Demo Program in Python

```
#!/usr/bin/env python3
# This listing is Appendix in the paper:
# "Constructing strong starters of orders 3p: triplication with SAT solver"
# by O. Ogandzhanyants, S. Sadov, and M. Kondratieva,
# J. Combin. Math. Combin. Comput. 126 (2025) 317-338
#-------------------------------------------------------------------------
# Solving "Sudoku mod3" problem for triplication of starters with z3 solver (Demo)
#-------------------------------------------------------------------------
# Loading functions from z3 solver library
from z3 import Int, Solver, And, Distinct, sat
#-------------------------------------------------------------------------

# Defining the parameters

m=7    # Order of base strong starter
q=3    # q=(m-1)/2 :  number of pairs in base starter
n=21   # n=3*m : Order of triplicated starter
k=10   # k=(n-1)/2 = 3*q+1 : number of pairs in triplicated starter

#.....................................................................
```

```
# Defining the base starter and key
T=[(2,3),(4,6),(1,5)]  # base starter
key=1
print("Base starter=",T,"key=",key)
Sigma_p=[(1,1),\
        (2,3),(3,4),(5,6),\
        (4,6),(5,0),(2,4),\
        (1,5),(2,6),(3,0)]
# Verify:
Tplus=[((key+ab[0])% 7,(key+ab[1])% 7) for ab in T]
Tminus=[((key-ab[1])% 7,(key-ab[0])% 7) for ab in T]
assert (Sigma_p[0]==(key,key))
assert ([Sigma_p[i] for i in [1,4,7]]==T)
assert ([Sigma_p[i] for i in [2,5,8]]==Tplus)
assert ([Sigma_p[i] for i in [3,6,9]]==Tminus)

# Defining variables for modular Sudoku problem
# Main unknowns: elements of pairs (U,V)
U=[Int(f'U_{i}') for i in range(k)]
V=[Int(f'V_{i}') for i in range(k)]

# Aux variables for differences D[i]=U[i]-V[i] mod 3
D=[Int(f'D_{i}') for i in range(k)]

# Aux variables for sums S[i]=U[i]+V[i] mod 3
#   (Only those S[i] with weak indexes will be essential.)
S=[Int(f'S_{i}') for i in range(k)]

# List of weak sets
weak_sets=[(2),(4,9),(1,5),(6,7)]

# List of indexes of weak pairs (=union of weak sets)
weak_indexes=[1,2,4,5,6,7,9]

# Dummy variable whose value will be set to 0
Z=Int(f'Z')

#.....................................................................
# Creating an instance of SAT solver from z3 library
Sudoku_solver=Solver()

#.....................................................................
# Setting up the constraints

# 0) Binding constraints:
    # Constraining dummy variable: Z=0
Sudoku_solver.add(Z==0)
```

```
    # Binding differences: D[i]==U[i]-V[i] mod 3
for i in range(k):
    Sudoku_solver.add((U[i]-V[i]-D[i]) % 3 ==0)

    # Binding sums in weak pairs: S[i]==U[i]+V[i] mod 3
for i in weak_indexes:
    Sudoku_solver.add((U[i]+V[i]-S[i]) % 3 ==0)

# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# 1) Range Constraints: all values must be in {0,1,2}
for i in range(k):
    Sudoku_solver.add(And(0<= U[i], U[i]<=2))
    Sudoku_solver.add(And(0<= V[i], V[i]<=2))
    Sudoku_solver.add(And(0<= D[i], D[i]<=2))

for i in weak_indexes:
    Sudoku_solver.add(And(0<= S[i], S[i]<=2))

# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# 2) Row Constraints: differences mod 3 in every row must be distinct
for i in range(q):  # in our example, i=0,1,2
    Sudoku_solver.add(Distinct(D[3*i+1: 3*i+4]))

# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# 3) Weak Set Constraints: pair sums in weak sets must be distinct

    # Weak set {2} of size 1 is appended by dummy;
Sudoku_solver.add(Distinct([S[2],Z])) # equivalent to S[2] != 0

    # Every weak set of size >=2 goes as is
Sudoku_solver.add(Distinct([S[4],S[9]]))  # For weak pair (4,9)
Sudoku_solver.add(Distinct([S[1],S[5]]))  # For weak pair (1,5)
Sudoku_solver.add(Distinct([S[6],S[7]]))  # For weak pair (6,7)

# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# 4) Color Constraints: values in every monochrome triple must be distinct
Sudoku_solver.add(Distinct([V[5],V[9], Z]))  # Degenerate triple is appended by dummy
Sudoku_solver.add(Distinct([U[0], V[0], U[7]]))
Sudoku_solver.add(Distinct([U[1], U[6], U[8]]))
Sudoku_solver.add(Distinct([V[1], U[2], U[9]]))
Sudoku_solver.add(Distinct([V[2], U[4], V[6]]))
Sudoku_solver.add(Distinct([U[3], U[5], V[7]]))
Sudoku_solver.add(Distinct([V[3], V[4], V[8]]))

#.................................................................
#  Solving modular Sudoku problem
if Sudoku_solver.check() == sat:
    solution=Sudoku_solver.model()
```

```
    Sigma_3=[(solution.evaluate(U[i]).as_long(), solution.evaluate(V[i]).as_long())\
        for i in range(k)]
    print("Sigma7=",Sigma_p)
    print("Sigma3=",Sigma_3)
#.....................................................................
# Finally, we merge the tables Sigma_p and Sigma_3 by CRT to get the new starter

    # Conversion of pair of residues mod 7 and mod 3 to residue mod 21
    def CRT_merge(pairing_mod7, pairing_mod3, CRT_dictionary):
        return [tuple(CRT_dictionary[x,X] for x,X in zip(uv,UV))\
                    for uv,UV in zip(pairing_mod7,pairing_mod3)]

    # Building two starters from the same solution of Sudoku (Theorem 3)
      # dictionary to recover x from (x mod 7, x mod 3)
    CRT_dictionary1={(x%7,x%3): x for x in range(21)}
    CRT_dictionary2={(x%7,(-x)%3): x for x in range(21)}  # x <- (x mod 7, -x mod 3)
    new_starter1=CRT_merge(Sigma_p, Sigma_3, CRT_dictionary1)
    new_starter2=CRT_merge(Sigma_p, Sigma_3, CRT_dictionary2)

    print ("Triplicated starters:")
    print (new_starter1)
    print (new_starter2)
#.....................................................................
else:
    print ("No solution")

#======================================================================
# Output: (another run of the code may produce a different solution!)
 Base starter= [(2, 3), (4, 6), (1, 5)] key= 1
 Sigma7= [(1, 1), (2, 3), (3, 4), (5, 6), (4, 6), (5, 0), (2, 4), (1, 5), (2, 6), (3, 0)]
 Sigma3= [(1, 2), (1, 0), (2, 0), (1, 1), (2, 2), (0, 2), (0, 1), (0, 2), (2, 0), (1, 1)]
 Triplicated starters:
 [(1,8), (16,3), (17,18), (19,13), (11,20), (12,14), (9,4), (15,5), (2,6), (10,7)]
 [(8,1), (2,3), (10,18), (5,20), (4,13), (12,7), (9,11), (15,19), (16,6), (17,14)]
```