The reduction from Theorem 1.1 to Lemma 1.1 utilizes Jensen's inequality, majorization (Marshall and Olkin, [2]), and a conditioning argument; details can be found in Ordentlich [1].

Lemma 1.1 was proven by Ordentlich [1] using an involved combinatorial argument. It is the purpose of this correspondence to deliver an elementary proof of Lemma 1.1, thereby giving an alternative short derivation of Theorem 1.1.

II. A Short Proof of Lemma 1.1

We shall perform induction n. It is simple to verify that (1) is true for $n = 2$. Suppose it holds for $n < m$. When $n = m \geq 3$, without loss of generality assume $a_1 \leq a_2 \leq \cdots \leq a_m$, and consider the nontrivial case of $0 < a_m \leq 1$. Let j be an integer such that $k = m - 2j > 0$. Denote $S = \sum_{i=1}^{m-1} Z_i a_i / a_m$. We have

$$\begin{aligned}
&\Pr\left(\sum_{i=1}^{m} Z_i a_i \in [-k,\ k]\right) \\
&= \Pr\left(Z_m + S \in [-k/a_m,\ k/a_m]\right) \\
&\geq \Pr\left(Z_m + S \in [-k,\ k]\right) \\
&= \frac{1}{2}\left\{\Pr\left(S \in [-k-1,\ k-1]\right) + \Pr\left(S \in [-k+1,\ k+1]\right)\right\} \\
&= \frac{1}{2}\left\{\Pr\left(S \in [-k+1,\ k-1]\right) + \Pr\left(S \in [-k-1,\ k+1]\right)\right\}
\end{aligned}$$

where the formula $\Pr(A) + \Pr(B) = \Pr(A \cup B) + \Pr(A \cap B)$ is used in the last equality.

Denote $S' = \sum_{i=1}^{m-2} Z_i$. If $k - 1 > 0$, then by the induction hypothesis (recall that $k - 1 = m - 1 - 2j$)

$$\Pr\left(S \in [-k+1,\ k-1]\right) \geq \Pr\left(S' \in [-k+1,\ k-1]\right). \qquad (2)$$

If $k - 1 = 0$ then m is an odd integer. Notice that (2) is still valid because the right-hand side is zero. Similarly

$$\Pr\left(S \in [-k-1,\ k+1]\right) \geq \Pr\left(S' \in [-k-1,\ k+1]\right).$$

Together we have, as long as $k = m - 2j > 0$

$$\begin{aligned}
&\Pr\left(\sum_{i=1}^{m} Z_i a_i \in [-k,\ k]\right) \\
&\geq \frac{1}{2}\left\{\Pr\left(S' \in [-k+1,\ k-1]\right) + \Pr\left(S' \in [-k-1,\ k+1]\right)\right\} \\
&= \frac{1}{2}\left\{\Pr\left(S' \in [-k-1,\ k-1]\right) + \Pr\left(S' \in [-k+1,\ k+1]\right)\right\} \\
&= \Pr\left(Z_{m-1} + \sum_{i=1}^{m-2} Z_i \in [-k,\ k]\right)
\end{aligned}$$

where for the last equality we consider the two cases $Z_{m-1} = 1$ and $Z_{m-1} = -1$. Thus, the claim holds when $n = m$, and Lemma 1.1 is proven.

Acknowledgment

The author thanks the Associated Editor and a referee for their valuable comments.

References

[1] E. Ordentlich, "Maximizing the entropy of a sum of independent bounded random variables," *IEEE Trans. Inf. Theory*, vol. 52, no. 5, pp. 2176–2181, May 2006.

[2] A. W. Marshall and I. Olkin, *Inequalities: Theory of Majorization and its Applications*. New York: Academic, 1979.

Computing Binary Combinatorial Gray Codes Via Exhaustive Search With SAT Solvers

Igor Zinovik, Daniel Kroening, and Yury Chebiryak

***Abstract*—The term binary combinatorial Gray code refers to a list of binary words such that the Hamming distance between two neighboring words is one and the list satisfies some additional properties that are of interest to a particular application, e.g., circuit testing, data compression, and computational biology. New distance-preserving and circuit codes are presented along with a complete list of equivalence classes of the coil-in-the-box codes for codeword length 6 with respect to symmetry transformations of hypercubes. A Gray-ordered code composed of all necklaces of the length 9 is presented, improving the known result with length 7.**

***Index Terms*—Binary sequences, cyclic codes, Gray codes.**

I. Introduction

The term combinatorial Gray code refers to a list of combinatorial objects such that the objects differ in some prescribed way [1]. If the list is composed of binary words, the codes are called binary combinatorial Gray codes. The survey [1] names a number of applications of binary Gray codes, including circuit testing, signal encoding, data compression, and parallel computing. Recent examples are related to such diverse areas as analog-to-digital conversion, diagnosis of multiprocessors, and computational biology, and can be found in [2], [3], and [4], respectively.

The construction of combinatorial codes can often be viewed as a search for a certain path in a graph where the vertices represent combinatorial objects. The algorithms proposed for the construction of long codes restrict the search to paths with a specific symmetry. While such restriction of the search is known to enhance the efficiency of the algorithms dramatically, this approach suffers from two drawbacks: a) construction of the longest code is impossible if the code does not possess the symmetry assumed by the algorithm; b) a classification of the codes is possible only within the symmetry class that is targeted by the algorithm.

Exhaustive search for codes is computationally infeasible even for low-dimensional problems. The efficiency of the search algorithms crucially depends on the employed backtracking methods. Experimental results with reachability analysis in computer-aided design [5] indicate that the backtracking implemented in the modern satisfiability (SAT) solvers for propositional logic [6] allows for an efficient analysis of the large-scale graphs emerging in model checking. A SAT solver is a tool that finds a satisfying assignment of a propositional formula given in conjunctive normal form (CNF) or concludes that such an assignment does not exist. State-of-the-art SAT solvers are known to be capable of handling problems with hundreds of thousands of variables within a practical amount of time.

Manuscript received May 17, 2007; revised December 19, 2007. This work was supported in part by an award from IBM Research and by an ETH Research Grant TH-19 06-3.

I. Zinovik is with the LTNT Laboratory of Thermodynamics in Emerging Technologies, ETH Zürich, 8092 Zürich, Switzerland (e-mail: izinovik@ethz.ch).

D. Kroening is with Computing Laboratory, Oxford University, Oxford OX1 3QD, U.K. (e-mail: daniel.kroening@inf.ethz.ch).

Y. Chebiryak is with the Computer Systems Institute, ETH Zürich, 8092 Zürich, Switzerland (e-mail: yury.chebiryak@inf.ethz.ch).

Communicated by T. Etzion, Associate Editor for Coding Theory.

Digital Object Identifier 10.1109/TIT.2008.917695

The objective of this correspondence is the evaluation of SAT-based techniques for the construction of binary combinatorial codes. The correspondence is organized as follows: In Section II, we present an encoding of a binary code called the *coil-in-the-box* code [7] as a Boolean formula and the application of SAT solvers to classify such codes with respect to symmetry transformations of hypercubes. This classification is of interest for the Glass model of gene regulatory networks [8]. In Section III, we extend the proposed encoding to the construction of a generalization of coil-in-the-box codes called *circuit* codes [9], and *distance-preserving* codes [10]. In Section IV, we present a SAT-based method for generating a subset of the circuit codes that are used in analog-to-digital conversion devices [2], and a Gray code for necklaces longer than those presented in [1]. The tables in the Appendix summarize the results of our experiments and report the coordinate sequences of the new codes.

II. Coil-in-the-Box Codes

A. SAT Encoding

A *coil* in a graph is a simple cycle without chords, which is defined as follows.

Definition 1 (Coil-in-the-Box [7]): A simple cycle in an n-dimensional cube is called a *coil-in-the-box*[1] if every edge in the n-cube that joins two vertices of the cycle is an edge of this cycle.

If the nodes of an n-dimensional unit cube are labeled by their coordinates, a coil-in-the box is represented by a Gray code of binary words W_i of length n with $i = 1, \ldots, N$. The number N of words in the code is called the *period* of the code. The codes of the maximum period with n up to 6 were generated by a computer search [7], [11]. The construction of the codes of maximum period with words of length 7 shown in [12] requires a restriction of the search to the codes that contain a sequence of $n+1$ words $W_i, \ldots, W_{i+n}$ such that the Hamming distance $d_H(W_i, W_{i+n})$ equals n (we will call such sequence a *diagonal* of the n-cube). The exact value of the maximum period is unknown for $n \geqslant 8$. The lower and upper bounds for the maximum period is a function of n [7]. The most recent results provide only lower bounds for the maximal period and generate the codes via evolutionary techniques [13].

In order to apply SAT solvers to generate codes, we define Boolean variables X_{ij} with $i \in \{1, \ldots, N\}$ and $j \in \{1, \ldots, n\}$ such that the *true* valuation of the variable denotes a 1 in position j of the binary word number i. Auxiliary Boolean variables H^0_{kl} and H^1_{kl} encode the Hamming distance $d_H(W_k, W_l)$ between words W_k and W_l such that the *true* valuation of H^α_{kl} with $\alpha \in \{0, 1\}$ determines that the words are identical or the distance equals 1, respectively. The following example illustrates the construction of a SAT instance for the search of a Gray code with period 3 which is composed of word of length 2. (A detailed description of the encoding for all codes studied in this correspondence can be found in [14].)

Example: Construction of SAT instance for the search for a Gray code with $n = 2$ and $N = 3$.

The auxiliary variables are defined as follows:

$$
\begin{aligned}
H^0_{13} &:= \Big(X_{11} \Leftrightarrow X_{31}\Big) \wedge \Big(X_{12} \Leftrightarrow X_{32}\Big) \\
H^1_{12} &:= \Big(\Big(X_{11} \Leftrightarrow \neg X_{21}\Big) \wedge \Big(X_{12} \Leftrightarrow X_{22}\Big)\Big) \vee \\
&\quad \Big(\Big(X_{11} \Leftrightarrow X_{21}\Big) \wedge \Big(X_{12} \Leftrightarrow \neg X_{22}\Big)\Big) \\
H^1_{13} &:= \Big(\Big(X_{11} \Leftrightarrow \neg X_{31}\Big) \wedge \Big(X_{12} \Leftrightarrow X_{32}\Big)\Big) \vee \\
&\quad \Big(\Big(X_{11} \Leftrightarrow X_{31}\Big) \wedge \Big(X_{12} \Leftrightarrow \neg X_{32}\Big)\Big) \\
H^1_{32} &:= \Big(\Big(X_{31} \Leftrightarrow \neg X_{21}\Big) \wedge \Big(X_{32} \Leftrightarrow X_{22}\Big)\Big) \vee \\
&\quad \Big(\Big(X_{31} \Leftrightarrow X_{21}\Big) \wedge \Big(X_{32} \Leftrightarrow \neg X_{22}\Big)\Big).
\end{aligned}
$$

[1]In some references, the codes are called *snakes*, but other articles use the term *snake* for noncyclic codes.

Thus, the condition that the distance between the words equals one and the first and the last words are distinct, is written as follows:

$$H^1_{12} \wedge H^1_{13} \wedge H^1_{23} \wedge \neg H^0_{13}.$$

This formula can be used as input to a SAT solver. SAT solvers determine whether the propositional formula is satisfiable or not and in the former case, return a satisfying assignment to the variables. The output obtained is decoded into the sequence of the binary words.

The construction above yields $N \times n$ independent variables X_{ij}, which correspond to the nodes of the cycle. We also tested an alternative encoding using the *coordinate sequences* instead of the node variables X_{ij}. A coordinate sequence is a list $\{s_i | s_i \in \{1, \ldots, n\}\}$, $i \in \{1, \ldots, N-1\}$ of the unique coordinates in which W_i and W_{i+1} differ. Thus, the code is uniquely defined by a choice of the first word W_1 and a coordinate sequence $\{s_i\}$.

For encoding coordinate sequences, every binary integer s_i is represented by $\lceil \log(n) \rceil$ Boolean variables S_{iq} with $q \in \{1, \ldots, \lceil \log(n) \rceil\}$. The *true* valuation of S_{iq} denotes a 1 in position q of the binary word s_i, while the *false* encodes a 0. In this encoding, the number of the variables is $(N-1) \cdot \lceil \log(n) \rceil + n$ variables, where X_{ij} is defined recursively as a function of $S_{(i-1)q}$ and $X_{(i-1)j}$

$$X_{ij} := \begin{cases} \neg X_{(i-1)j} : & \text{if } s_{i-1} = j \\ X_{(i-1)j} : & \text{otherwise.} \end{cases} \tag{1}$$

The computation of the codes was carried out using a PC with an Intel Xeon (3.0-GHz, 4-GB RAM, running Linux) with a timeout of 24 h. The satisfying assignments were obtained using the SMT-solver Yices [15] for codes with codeword length up to 8.

While the coils computed did not surpass the longest known coil of period 96, the experiments indicate that the codes with periods up to 82 can be generated within 6 h. The results show that the sequence encoding reduces the run time for the cases without satisfying assignments (the *UNSAT* cases), e.g., in case $n = 5$, $N = 16$, UNSAT for the node encoding is returned in 257 s while the sequence encoding needs only 0.4 s. The node encoding was faster for all tests with satisfying assignments except for the maximum coil in dimension 7 (the runtime is 36 min compared to 162 min for the node encoding).

Unlike the search method for the maximum known codes described in [12], the presented construction allows for the generation of maximum coils without the cube diagonals (a longest coil in dimension 7 that has no cube diagonal is shown in the Appendix). The suggested method relies on a direct encoding of the code definition without any additional restrictions, and thus, the search space contains *all* coils for a prescribed dimension and length. The UNSAT cases obtained as a result of incrementally increasing the period N serve as a proof that the previous satisfying assignment constitutes a maximum-period code for the prescribed dimension.

B. Classification of Codes in Dimension 6 and 7

The absence of restrictions narrowing the solution space makes it possible to use the encoding to analyze *all* combinatorial codes for a prescribed dimension. The classification of the coils-in-the-box codes with respect to axis permutations of the n-cube is of interest in Glass models for neural and gene regulatory networks. In this model, active and inactive states of a particular gene are depicted by $1/0$, and a binary codeword represents a set of the genes in a cell at a given time instant. Coil-in-the box codes correspond to stable periodic processes which describe biologically relevant dynamics of the gene sets. Thus, the number of the equivalence classes of the codes indicates how many different types of cells can be regulated by the set of the genes.

The number of the equivalence classes up to dimension 5 is computed in [8]. The lower bounds for the number of the classes in dimension 6 has been obtained in [16]. The presented classification algorithm is a modified ALL-SAT procedure as described in [16]. Every assignment obtained is a representative of an equivalence class. MiniSAT was used for the computations, and the results are summarized in Table I.

TABLE I
THE NUMBER OF EQUIVALENCE CLASSES FOR THE COILS-IN-THE-BOX FOR DIMENSION 6

Length	Lower bound of [16]	#Equivalence-classes
16	385	563
18	1066	1228
20	981	1032
22	465	478
24	103	110
26	4	4

The coordinate sequences for the equivalence classes of the longest coils in dimension 6 are presented in the Appendix. One of four classes is found to contain no diagonal, and one class represents the coils with periodic coordinate sequences.

The exhaustive search for all maximum coils in dimension 7 is known to be computationally demanding [12]: the search for coils with a cube diagonal took more than one month on a network consisting of five SUN Microsystems SparcCenter 1000's with two processors each. We have applied the propositional encoding described above for the classification of the maximum coils ($N = 48$) in dimension 7. The full classification could not be archived within a timeout of 100 days. The number of the equivalence classes found within the timeout is 126, where 74 classes do not have the cube diagonal, i.e., they cannot be found by the algorithm used in [12].

III. NEW CIRCUIT CODES, AND DISTANCE-PRESERVING CODES

The SAT encoding described above can be modified for the construction of *circuit codes* [9] with different *spreads* and *distance preserving* codes [10].

A. Circuit Codes

Circuit codes are defined as a generalization of the coil-in-the-box definition (1) as

$$\forall k, l : d_H(W_k, W_l) < \delta \Rightarrow d_C(W_k, W_l) < \delta \tag{2}$$

where the positive integer δ is called the *spread* of the code [17]. Thus, the circuit codes with the spread $\delta = 2$ are the coils-in-the-box, and the codes with $\delta = 1$ of period $N = 2^n$ are the Hamiltonian cycles in the n cube. Moreover, any code with spread δ_1 is a circuit code with spread δ_2 if $\delta_1 \geq \delta_2$ [17]. This property implies that the period of maximum circuit codes with a spread δ_1 does not exceed the maximum period of the codes with $\delta_2 \leq \delta_1$.

The construction of the longest known circuit codes is based on either an exhaustive search or an algorithm that restricts the search to the codes with periodic coordinate sequences [9]. The codes can be generated from the existing ones from lower dimensions n (e.g., [18]).

We conducted computations aiming to improve the results in [9], which present the longest known circuit codes with spread $\delta \leq 7$. MiniSAT finds a satisfying assignment for $n = 11, \delta = 4$, and $N = 60$ within 23 min. Six codes with greater periods were obtained within the timeout of 24 h. The coordinate sequences of the found codes are presented in the Appendix (see Table IV).

To the best of our knowledge, there are no circuit codes reported for the spreads $\delta > 7$. The SAT encoding was used to compute codes with spread δ up to 17. We present four codes that are longer than known lower bounds of [17] in the Appendix. We also show four codes with the periods whose optimality was proved in [17].

B. Distance-Preserving Codes

The distance-preserving codes are defined by the following equation:

$$\forall k, l : d_C(W_k, W_l) \leq \delta \Rightarrow d_H(W_k, W_l) = d_C(W_k, W_l)\ . \tag{3}$$

TABLE II
COORDINATE SEQUENCES OF EQUIVALENCE CLASSES OF LONGEST COILS IN DIMENSION 6 (BOLD TRANSITIONS BELONG TO A DIAGONAL)

1 2 3 4 5 6 4 3 2 4 5 3 4 **1 2 3 4 5 6** 4 3 2 4 5 3 4
1 2 3 **6 2 4 3 5 1** 3 4 6 1 4 **2 3 1 4 6 5** 4 3 6 2 3 4
1 2 3 **5 1 6 4 2 3** 6 1 3 5 1 **2 3 6 1 5 4** 2 3 5 1 3 6
1 2 3 4 6 3 1 4 3 5 6 3 4 1 2 3 4 6 3 1 4 3 5 6 4 3

TABLE III
AN EXAMPLE COORDINATE SEQUENCE OF A LONGEST COIL IN DIMENSION 7 WITHOUT DIAGONAL

76312413743615712763473167347627431637152673163 4

This code preserves the Hamming distance between the codewords for all distances up to a threshold m. The distance-preserving code in dimension n is denoted as $\langle m, n\rangle$-code, and the $\langle m, n\rangle$-codes with $N = 2^n$ are called the *complete* codes. Two types of algorithms are reported for distance-preserving codes. The method shown in [10] generates the codes only of a certain length $L = m2^{n-\lceil m/2\rceil}$. The algorithm proposed in [19] constructs $\langle n-1, n\rangle$-codes with period $(n-1)2^{\lceil n/2\rceil}$.

In contrast to the known methods, the SAT encoding is easy to modify for the construction of the codes with an arbitrary prescribed codeword length n and threshold m. As examples, we computed the codes $\langle 6, 7\rangle$ with a maximum period of 100, and $\langle 7, 8\rangle$ with a period of 126 (see the Appendix). While the values of the maximum periods and the lower bounds for the periods of such codes were calculated in [19], to the best of the authors' knowledge, the coordinate sequences for these codes were not yet reported in the literature.[2]

IV. NEW SINGLE-TRACK CIRCUIT CODES AND NECKLACES

Single-track circuit codes introduced in [21] are a subclass of circuit codes which are used for digital–to-analog conversion. The single-track codes are circuit codes that possess an additional property defined in terms of *component sequences* [2].

Definition 2 (Component Sequence): Let C be a cyclic path on the n-cube consisting of N binary codewords $W_1, \ldots, W_N$, where $W_i \in \{w_i^1, \ldots, w_i^N\}$. The component sequence j of C, denoted C^j, is the binary periodic sequence $w_1^j, \ldots, w_N^j$ consisting of component j of each of the codewords of C ($1 \leq j \leq n$).

The formal definition of the single-track codes reads as follows.

Definition 3 (Single-Track Circuit Codes): Let C be a circuit code with period N, spread δ, composed of codewords of length n. Then C is said to be a single-track circuit code if its component sequence C^j is a cyclic shift of sequence C^1 for each $2 \leq j \leq n$. We denote these codes by $\langle n, \delta, N\rangle$.

For every n, there is a $\langle n, n, 2n\rangle$ code called the *trivial* code that has the coordinate sequence composed of two repeating n-cube diagonals. Single-track circular codes of longer periods are constructed in [2] by embedding a set of the codewords into the known circuit codes. The codes are generated up to $\delta \leq 6$ using the circuit codes from [9] that have the spread δ bounded by 7. A single-track circuit code is called *optimal* if there is no single-track circuit code with the same period but composed of codewords of shorter length. Two conditions were used in [2] to determine the optimality of the codes: a) the code period does not exceed the period of the corresponding circuit codes, and b) if a single-track code exists, its period equals an even multiple of the codeword length.

[2]Three months after the manuscript was submitted to IEEE TRANSACTIONS OF INFORMATION THEORY, code $\langle 7, 8\rangle$ with a period of 200 was reported in [20].

TABLE IV
EXAMPLES OF CIRCUIT CODES

(n, δ, N)	First codeword	Coordinate sequence
Improved lower bounds for spreads $s \leq 7$		
(9, 3, 56)	010110101	948265876147831273962318264176584139784928637896417693 72
(10, 3, 84)	1101000100	583a65872a5692a358a6917425319756912538a52 9a748a9145764136947236753 8a4752a3145362a934
(11, 4, 66)	00111000101	a93426785463a71536b428315673295837a518634128ab4617a54b93658ba95167
(14, 6, 64)	00101011010001	c6d157936b1ed49abc67d8a2be3d97ac58de7615c4978b53c6d7b29ce61b38a9
(15, 7, 58)	110101101011000	c98375fea2d69751ec2467853cbf6ed8592afec81d753f6e14d9b3521e
(16, 7, 70)	0111001100110010	7eb149fd738g4cf2761eba84cd5e37b8g41e2dbf79ga4c61bd3g2a5eb16784ae312f6c
Improved lower bounds for spreads $s > 7$		
(15, 8, 42)	000010100111111	81fc4e69b2afdc367e5a8bf164eca2b936fdc7eb5a
(16, 8, 52)	1101111101000011	3925ca4ged9b57846239e5a1g48f3ce9dgb823ae57dg621a48fd
(17, 8, 62)	00000111010100111	4hc7816afd4c58g2197eac8d239behafd42gce7a19df8c5ga6342fc7g1a9be
(17, 9, 50)	11111100111010110	16bf8ceah97g3b82e41h75cdb6ge1f8a7cb9g3h12e4c85gdh7
Codes of lengths proven to be optimal (spread $s > 7$)		
(14, 8, 38)	10001000111010	b1527346e9ba587c4deb1527346e9ba587c4ed
(15, 9, 40)	100001111100011	64fb7c8312a45bdce39264fb7c8312a45bdce392
(16, 9, 44)	0011100001111111	ca85b6413297cfbd8g3e49cab586314297cf8dbg4e39
(17, 10, 46)	10110010000101111	4c7h8f61ge3d5c9hafb1e2d4c7h8f61ge3d5c9hafb12ed

TABLE V
EXAMPLES OF DISTANCE PRESERVING CODES

(n, δ, N)	First codeword	Coordinate sequence
(7, 6, 100)	0001011	1265431265471325641725341725643725641723654712634712 6543126547123564172534172564372564172654712634 7
(8, 7, 126)	10110100	76852417385641728354672815437682514738251476832147685214368527436 85124378512436851723684172368514276851327685412738541276854 31

We conducted computations aimed to construct single-track circuit codes with the periods longer than the results in [2]. No satisfying assignments were obtained within a 24-h timeout. *UNSAT* was returned for five test cases: $\langle 7, 3, 28\rangle$, $\langle 8, 3, 24\rangle$, $\langle 9, 3, 72\rangle$, $\langle 8, 4, 24\rangle$, and $\langle 9, 4, 36\rangle$. The result proves optimality of the corresponding five single-track circuit codes found in [2] for the same δ.

The satisfying assignments for the test cases with $\delta = 7$, which is greater than in [2], were computed for $n = 10, 11, 12$, and $N = 2n$ within 2 h. The corresponding coordinate sequences (see the Appendix) indicate that the obtained codes are nontrivial. We also conducted a search for nontrivial single-track circuit codes with $\delta = 7$ for $n = 7, 8,$ and 12. The results of the computations show that there are no nontrivial single-track circuit codes in these cases.

The longest known circuit and single-track circuit codes have been constructed by restricting the search to the codes that possess various internal symmetries. The methods rely on the generation of the Gray-ordered *binary necklaces* as a first step of the construction [9]. An n-bead binary necklace is an equivalence class of binary n-tuples under rotation [1]. Necklace-based construction of the codes is proved in [22], [23] to be very successive for small δ. The methods are not easily adapted to produce the codes with δ larger than 7 because of a rapid increase of the number of necklaces.

While several known algorithms provide a complete list of the necklaces with a prescribed codeword length n, none of them computes a Gray code for necklaces. Efficient algorithms producing Gray-ordered necklaces are of interest in combinatorics [24], and the question whether a complete list of Gray-ordered necklaces exists for $n > 7$ is among the open problems of combinatorial Gray codes (a parity argument shows that this is impossible for an even n) [1].

The results of our computations for six-bead necklaces indicate that there is no Gray code with length greater than 13 codewords. The Gray codes for eight-bead necklaces were obtained up to 33 codewords. The complete list of nine-bead necklaces contains 60 codewords [25], thus, a propositional formula describing the list was generated and consequently used as input to the SAT solver. A satisfying assignment has been obtained within 68 min using the encoding of the coordinate sequences. The coordinate sequence of the Gray code is presented in the Appendix. To the best of our knowledge, the Gray code consisting of all nine-bead necklaces was not yet reported in the literature.

TABLE VI
EXAMPLES OF NONTRIVIAL SINGLE-TRACK CIRCUIT CODES WITH SPREAD $\delta = 7$

(n, δ, N)	First codeword	Coordinate sequence
(10, 7, 20)	1011000110	9a35824617a953286471
(11, 7, 22)	01000010110	5247ba6935148b2673a198
(12, 7, 24)	000010110001	b56c17283ba6419253ca7489

TABLE VII
THE COORDINATE SEQUENCE OF NINE-BEAD NECKLACE (FIRST CODEWORD IS ZERO)

376351867637536763812813651314783812396571543686235673835641

V. CONCLUSION

We present a propositional encoding for the generation of coil-in-the-box codes, circuit codes, distance-preserving codes, single-track codes, and necklaces. The method we suggested for the construction of the codes utilizes efficient backtracking algorithms implemented in state-of-the-art propositional SAT solvers. The encoding enforces an exhaustive search over all codes satisfying the definition, and thus it can be used for a classification of the codes. Search within a desired subclass of the codes can be conducted by using additional blocking clauses.

We report new lower bounds for ten circuit codes. We present three new nontrivial single-track circuit codes and two new distance-preserving codes.

An advantage of the SAT-based approach is that a SAT solver returns a definite answer whether a code with given parameters exists or not. The negative answer may serve as a proof of optimality of the codes.

Using this approach, we proved optimality for five known single-track circuit codes.

We found all equivalence classes with respect to the n-cube symmetry transformations for the coil-in-the-box codes in dimension 6, and obtained a lower bound, namely 126, on the number of equivalence classes for longest coils in dimension 7. We also proved by construction the existence of the Gray code for the complete list of nine-bead necklaces, thus improving a known result for seven-bead necklaces.

APPENDIX

The results of our experiments and the coordinate sequences of the new codes are shown in Tables II–VII.

ACKNOWLEDGMENT

The authors are indebted to an anonymous reviewer for bringing the papers [21]–[23] to their attention and for many helpful comments and suggestions that improved this paper. The authors also appreciate detailed answers of A. L. Perezhogin regarding the distance-preserving codes.

REFERENCES

[1] C. Savage, "A survey of combinatorial Gray codes," *SIAM Rev.*, vol. 39, no. 4, pp. 605–629, 1997.

[2] A. P. Hiltgen and K. G. Paterson, "Single-track circuit codes," *IEEE Trans. Inf. Theory*, vol. 47, no. 6, pp. 2587–2595, Sep. 2001.

[3] M. K. U. Blass, I. Honkala, and S. Litsyn, "Short dominating paths and cycles in the binary hypercube," *Ann. Comb.*, no. 5, pp. 51–59, 2001.

[4] H. de Jong, "Modeling and simulation of genetic regulatory systems: A literature review," *Comput. Biol.*, vol. 9, no. 1, pp. 67–103, 2002.

[5] O. Grumberg, A. Schuster, and A. Yadgar, "Memory efficient all-solutions SAT solver and its application for reachability analysis," in *Formal Methods in Computer-Aided Design*, 2004, vol. 3312, pp. 275–289.

[6] N. Eén and N. Sörensson, "An extensible SAT-solver," in *Theory and Applications of Satisfiability Testing (Lecture Notes in Computer Science)*. Berlin, Germany: Springer, 2004, vol. 2919, pp. 502–518.

[7] H. L. Abbott and M. Katchalski, "On the snake in the box problem," *J. Comb. Theory Ser. A*, vol. 45, no. 1, pp. 13–24, 1987.

[8] L. Glass, "Combinatorial aspects of dynamics in biological systems," in *Proc. Statistical Mechanics and Statistical Methods in Theory and Application: A Tribute to Elliott W. Montroll Symp.*, Rochester, NY, Nov. 1976, pp. 585–611.

[9] K. G. Paterson and J. Tuliani, "Some new circuit codes," *IEEE Trans. Inf. Theory*, vol. 44, no. 3, pp. 1305–1309, May 1998.

[10] A. J. van Zanten and A. Lukito, "Construction of certain cyclic distance-preserving codes having linear-algebraic characteristics," *Des. Codes, Cryptogr.*, vol. 16, no. 2, pp. 185–199, 1999.

[11] G. Zémor, "An upper bound on the size of the snake-in-the-box," *Combinatorica*, vol. 17, no. 2, pp. 287–298, 1997.

[12] K. J. Kochut, "Snake-in-the-box codes for dimension 7," *J. Comb. Math. and Comb. Comput.*, no. 20, pp. 175–185, 1996.

[13] D. Casella and W. Potter, "Using evolutionary techniques to hunt for the snakes and coils," in *Proc. IEEE Congress on Evolutionary Computation*, Edinburgh, U.K., 2005, vol. 3, pp. 2499–2505.

[14] I. Zinovik, D. Kroening, and Y. Chebiryak, Computing Binary Combinatorial Gray Codes via Exhaustive Search with SAT-Solvers ETH, 2007, Tech. Rep. 580.

[15] B. Dutertre and L. de Moura, "A fast linear arithmetic solver for DPLL(T)," in *Proc. CAV 06 (Lecture Notes in Computer Science)*. Berlin, Germany: Springer, 2006, vol. 4144, pp. 81–94.

[16] I. Zinovik, D. Kroening, and Y. Chebiryak, "An algebraic algorithm for the identification of Glass networks with periodic orbits along cyclic attractors," in *Proceedings of Algebraic Biology (AB) (Lecture Notes in Computer Science)*. Berlin, Germany: Springer, 2007, vol. 4545, pp. 140–154.

[17] R. C. Singleton, "Generalized snake-in-the-box codes," *IEEE Trans. Electron. Comput.*, vol. EC-15, no. 4, pp. 596–602, Apr. 1966.

[18] V. Klee, "A method for constructing circuit codes," *J. ACM*, vol. 14, no. 3, pp. 520–528, 1967.

[19] A. L. Perezhogin, "On cyclic (m,n)-enumerations," *Discr. Appl. Math.*, vol. 135, no. 1–3, pp. 235–243, 2004.

[20] A. L. Perezhogin, "On automorphisms of cycles in the binary n-cube," (in Russian) *Diskretnyi Analiz i Issledovanie Operatsii*, vol. 14, no. 3, pp. 67–79, 2007.

[21] K. P. A. P. Hiltgen and M. Brandestini, "Single track Gray codes," *IEEE Trans. Inf. Theory*, vol. 42, no. 5, pp. 1555–1561, Sept. 1996.

[22] T. Etzion and K. Paterson, "Near optimal single-track Gray codes," *IEEE Trans. Inf. Theory*, vol. 42, no. 3, pp. 779–789, May 1996.

[23] M. Schwartz and T. Etzion, "The structure of single-track Gray codes," *IEEE Trans. Inf. Theory*, vol. 45, no. 7, pp. 2383–2396, Nov. 1999.

[24] A. D. C. Degni, "Gray ordered binary necklaces," *Electron. J. Comb.*, vol. 14, no. R7, pp. 1–223, 2007.

[25] N. J. Sloane, Database of Integer Sequences [Online]. Available: http://www.research.att.com/~njas/sequences/A000031