New Evolutionary Search for Long Low Autocorrelation Binary Sequences

Ke-Lin Du, *Senior Member, IEEE*, Wai Ho Mow, *Senior Member, IEEE*, and Wei Hsiang Wu

Abstract

Binary sequences with low aperiodic autocorrelation levels, defined in terms of the peak sidelobe level and/or merit factor, have many important engineering applications, such as radars, sonars, spread spectrum communications, system identification and cryptography. Searching for low autocorrelation binary sequences (LABS) is a notorious combinatorial problem, and has been chosen to form a benchmark test for constraint solvers. Due to its prohibitively high complexity, an exhaustive search solution is impractical, except for relatively short lengths. Many suboptimal algorithms have been introduced to extend the LABS search for lengths of up to a few hundreds. In this paper, we address the challenge of discovering even longer LABS by proposing an evolutionary algorithm with a new combination of several features, borrowed from genetic algorithms, evolutionary strategies and memetic algorithms. The proposed algorithm can efficiently discover long LABS of lengths up to several thousands. Record-breaking minimum peak sidelobe results of many lengths up to 4096 have been tabulated for benchmarking purpose. In addition, our algorithm design can be easily adapted to tackle various extensions of the LABS problem, say, with a generic sidelobe criterion and/or for possibly nonbinary sequences.

Index Terms

Low autocorrelation binary sequences, peak sidelobe level, merit factor, evolutionary algorithm

I. Problem Statement

Searching for low autocorrelation binary sequences (LABS) is a classical computational problem that raises a challenge to all kinds of search methodologies. LABS are widely used in pulse compression radars and sonars, channel synchronization and tracking, spread spectrum and code-division multiple-access communications, and cryptography [1].

This work was supported by the Hong Kong Research Grants Councils (GRF Project number 616512). The majority of this work was conducted when the authors were with the Hong Kong University of Science and Technology.

Ke-Lin Du is with Xonlink Inc., Ningbo, China. Wai Ho Mow and Wei Hsiang Wu are with the Department of Electronic and Computer Engineering, Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong S.A.R., China. E-mail: kldu@ieee.org, w.mow@ieee.org.

For a binary sequence of length L, $\mathbf{a} = a_1 a_2 \ldots a_L$ with $a_i = \{-1, +1\}$ for all i, its autocorrelation function (ACF) is given by

$$C_k(\mathbf{a}) = \sum_{i=1}^{L-k} a_i a_{i+k}, \quad k = 0, \pm 1, \ldots, \pm(L-1). \tag{1}$$

For $k = 0$, the value of ACF equals L and is called the peak, and for $k \neq 0$, the values of ACF are called the sidelobes. The peak sidelobe level (PSL) of a binary sequence $\mathbf{a}$ of length L is defined as

$$PSL(\mathbf{a}) = \max_{k=1,\cdots,L-1} |C_k(\mathbf{a})|. \tag{2}$$

The minimum peak sidelobe (MPS) defined for all possible binary sequences of length L is defined as

$$MPS(L) = \min_{\mathbf{a} \in \{-1,+1\}^L} PSL(\mathbf{a}). \tag{3}$$

For length L, the MPS is known to be upper-bounded by $\sqrt{2L \ln L}$ [2]. A binary sequence with PSL at most $\sqrt{2L \ln(2L)}$ for every length $L > 1$ was constructed in [3]. It was empirically shown therein that its PSL actually grows like $0.9\sqrt{L \ln(\ln L)}$, which is still far larger than best known PSL results obtained by well-designed computer searches.

The merit factor F of a binary sequence $\mathbf{a}$ is defined as [4]

$$F(\mathbf{a}) = \frac{L^2}{2 \sum_{k=1}^{L-1} C_k^2(\mathbf{a})}. \tag{4}$$

The sum term in the denominator is called the sidelobe energy of the sequence. It is conjectured in [4] that for the best binary sequences in the sense of achieving the maximum possible merit factor, we have $F \to 12.3248$ as $L \to \infty$.

Roughly speaking, there are two versions of LABS searches in the literature: one targets at low PSL and the other targets at high merit factor (or equivalently, low sidelobe energy). In this paper, our key focus is to search for long LABS with low PSL, which is more challenging because of the non-analytical maximum operator in its definition.

The rest of this paper is organized as follows. Section II provides a literature survey on previous works and results on the LABS problem. Section III summarizes the key features of major evolutionary algorithms and then present our proposed design. Section IV presents the search results on LABS using our proposed evolutionary algorithm and compare them with other benchmarking results. Finally, Section V contains the concluding remarks.

II. Literature Survey

Both versions of the LABS problem are hard since the search space grows exponentially with the sequence length and there are numerous local minima, as well as many optima. For example, a full search for $L = 64$ yields 14872 optimal binary sequences achieving MPS 4, though these sequences have a wide variability of merit factors [5]. The conventional gradient-based and common search approaches are almost always trapped in some poor local minima.

In order to find optimal sequences of length L, the brute-force exhaustive search requires to examine 2^L binary sequences. The branch-and-bound enumeration algorithm requires a runtime complexity of $O(1.85^L)$ in order to find optimal merit factors for all $L \leq 60$ [1], [6]. A state-of-the-art exhaustive search algorithm for MPS binary sequences was reported in [5]. The method integrates combinatoric tree search techniques, the use of PSL-preserving symmetries, data representations and operations for fast sidelobe computation, and partitioning for parallelism. The PSL-preserving operations applied to any binary sequence $\mathbf{a}$ (i.e., negation of $\mathbf{a}$, reversal of $\mathbf{a}$, and sign alternation of $\mathbf{a}$, and their combinations) can preserve its PSL. Consequently, the entire set of binary sequences can be represented by a subset of less than half of its original size [5]. To find all MPS binary sequences, it suffices to search over this subset. This method has a runtime complexity of roughly $O(1.4^L)$ [5], [7].

Some of the known exhaustive search results can be summarized as follows (c.f. [3]):

1) $MPS(L) = 1$ for $L = 2, 3, 4, 5, 7, 11, 13$; (These optimal MPS sequences are known as *Barker sequences*.)
2) $MPS(L) \leq 2$ for $L \leq 21$;
3) $MPS(L) \leq 3$ for $L \leq 48$ [1];
4) $MPS(L) \leq 4$ for $L \leq 70$ [5];
5) $MPS(L) \leq 4$ for $71 \leq L \leq 82$ [8];
6) $MPS(L) \leq 5$ for $83 \leq L \leq 105$ [8].

Barker sequences with PSL being 1 are known only for lengths 2, 3, 4, 5, 7, 11 and 13. It has been long conjectured that longer Barker sequence does not exist. The Barker condition that PSL ≤ 1 has been extended for polyphase sequences defined over K-th roots of unity of the form $a_i = e^{2\pi n_i \sqrt{-1}/K}$ with n_i being some integer between 0 and $K-1$ for all i, where K represents the phase alphabet size. The list of known polyphase Barker sequences has been extended to length 77 [10], [9]. However, since practical applications do not favor large phase alphabets, another direction is to search for low autocorrelation quadriphase sequences, which have better PSL and MF over the best biphase codes [11].

For odd length L, the so-called skew-symmetric binary sequences has the property that $a_{(L+1)/2+i} = (-1)^i a_{(L+1)/2-i}$, for $i = 1, \ldots, (L-1)/2$. For these sequences, $C_k(\mathbf{a}) = 0$ for all odd k. Since the right half of the sequence is determined by the left half, searching the skew-symmetric sequences reduces the effect length of the sequence by a factor of two. Some good results were reported for skew-symmetric sequences, but not for all lengths [1].

To meet the need of longer LABS for practical applications, one approach to dramatically reduce the search complexity is to focus on some special classes of binary sequences. The maximal-length shift register sequences (also called the m-sequence) are pseudorandom sequences of length $L = 2^n - 1$ for $n = 1, 2, \ldots$, which have an ideal periodic autocorrelation function, and they can be easily generated by feedback shift registers [12]. The Legendre sequences are another class of pseudorandom sequences. By searching cyclically shifted variants of the Legendre sequences of prime lengths, low PSL results for prime lengths of up to a thousand were tabulated in [13]. For non-prime L, reasonably good results can be obtained by periodically extending good cyclically shifted Legendre sequences of prime lengths. A numerical investigation was presented for the PSL of Legendre sequences, m-sequences, and Rudin-Shapiro sequences in [7]. The maximum asymptotic merit factor of an optimally cyclically shifted Legendre sequences is 6, and that of an m-sequences is 3, that of a Rudin-Shapiro sequence, as well as its mate, is 3. Besides, in [7], the variation of the PSLs of the Legendre sequences of the first 3500 prime lengths (i.e., $L \leq 32609$), as well as those of the m-sequences of lengths up to $n = 20$ (i.e., $L = 2^{20} = 1048575$) were also given. It can be seen that the Legendre sequences are far superior to the m-sequences and the Rudin-Shapiro sequences in terms of both PSL and MF. In [14], a systematic way to apply local search strategies to optimize the PSL and MF of a sampled and binarized version of various linear frequency modulated chirp signals, which has been widely used as radar signals, were introduced. LABS of selected lengths up to 4096 with good PSL were tabulated.

In [15], an integer programming formulation of the LABS problem for any L was given. The values of PSL and the merit factor F (for $L = 71$ to 100) of the sequences were obtained by using a Mixed-Integer Linear Programming (MILP) solver on the Network-Enabled Optimization System (NEOS) server, which uses the sequential quadratic programming technique. Overall speaking, the PSL results obtained therein are no better than those obtained by an evolutionary algorithm (EA) [21], and a lower PSL value of 5 was obtained only for $L = 74$.

Very recently, a signal processing-style computational framework in [16] was proposed to tackle the LABS problem and its various extensions. The essence of the framework is an alternating projection algorithm based on an iterative twisted approximation, which is a merit factor maximizer that can yield solutions depending on initialization. However, the method does not have an effective way to get out of local optima and is unlikely to outperform a well-designed stochastic search.

Some stochastic search methods, such as simulated annealing and EAs, can be applied for escaping local minima. In [17], a stochastic method with a runtime complexity of $O(1.68^L)$ was reported. Compared with the Kernighan-Lin solver [18] having a runtime complexity of $O(1.463^L)$, the searches based on evolutionary strategies (ES) for optima may require significantly less samples on average and have a runtime complexity of $O(1.397^L)$ [6].

Popular EAs include the genetic algorithm and the memetic algorithm, in addition to the ES. A recent review on the LABS problem was given in [19]. Generally speaking, the performance of EAs are superior to other stochastic search algorithms [19]. In fact, the EAs have attained the best results so far [6]. There are quite a few works on applying EAs to the LABS problem [6], [20], [21], [22], [23], [24], [25], [26].

In [21], the genetic algorithm is applied. The method first generates a population of size N_P, then generates some offspring by one-point or two-point mutation, and others by one-point crossover. Unlike the classical genetic algorithm that uses a proportional probabilistic selection mechanism, elitism is applied. Namely, offspring of size N_P with the best fitness are then selected as the parents in the next generation. The fitness function is selected as

$$f_1(\mathbf{a}) = \frac{\alpha}{PSL(\mathbf{a})} + \beta F(\mathbf{a}) \tag{5}$$

where α and β are scaling factors. When $\alpha = 0$ and $\beta \neq 0$, the fitness function corresponds to the minimum PSL. When $\alpha \neq 0$ and $\beta = 0$, it corresponds to the maximum merit factor F. A list of sequences of lengths 49 to 100 are given. The obtained PSL values are the same or better than those obtained in [34], where the Hopfield neural network was used for finding good binary sequences. In [22], the method first generates N_P parents, and then generates offspring of size N_O by one-point crossover; the $N_P + N_O$ individuals compete and the N_P best individuals survive as the next generation; one-point or two-point mutation is applied only when some of the N_P best individuals have the same fitness, i.e., PSL. In [20], ES was used to search for LABS with locally optimal merit factor F, and a preselection operation was applied to the individuals created from mutation.

The memetic algorithm was used for the LABS problem in [23], [24]. In [23], an ES was used as the EA, and a local search was implemented by flipping each bit of the string. The fitness function is selected as

$$f_2(\mathbf{a}) = \frac{F(\mathbf{a})}{PSL(\mathbf{a})} \tag{6}$$

The obtained F is greater that that of [21] for $L = 71$ to 100, but the PSL is typically worse. In [24], the bit-flipping or tabu search was used as the local search for maximizing F. The memetic algorithm with tabu search is more effective in finding the optimal merit factor F than the Kernighan-Lin solver and the memetic algorithm with bit climber, from the experiment for $L \leq 60$. The memetic algorithm with tabu search is an order of magnitude faster than the pure tabu search with frequent restarts [35]. The latter is roughly on par with the Kernighan-Lin solver for the LABS problem [6].

Some important real-world applications require the search criterion or fitness function of the LABS to be generalized in various ways in order to find (possibly non-binary) sequence sets with a good tradeoff (defined in some sense) between low crosscorrelation levels and low autocorrelation sidelobe levels. In general, it is not too difficult to adjust the EA to accommodate a new fitness function. In [25], a multi-objective EA was used to generate complex spreading sequences with good crosscorrelation and autocorrelation properties. In [26], the genetic algorithm was used for finding good training sequences for multiple antenna (spatial multiplexing) systems.

III. Evolutionary Algorithm Design for LABS

From our literature survey in the previous section, EAs are found to be well-suited for the long LABS problem. In this section, the design and pseudocode of our proposed evolutionary algorithm will be presented after summarizing the key features of the three type of evolutionary algorithms, namely, genetic algorithms, evolutionary strategies and memetic algorithms. The latter are inspirations of our proposed design.

A. Introduction to Evolutionary Algorithms

Evolutionary algorithms (EAs) are a class of general-purpose stochastic optimization algorithms under the universally accepted neo-Darwinian paradigm. The neo-Darwinian paradigm is a combination of the classical Darwinian evolutionary theory, the selectionism of Weismann, and the genetics of Mendel [27]. EAs are currently a major approach to adaptation and optimization.

EAs and similar population-based methods are simple, parallel, general-purpose, global optimization methods. They are useful for any optimization problem, particularly when conventional optimization techniques are invalid. They are active and efficient global optimization methods.

1) EA Procedure: In EA, individuals in a population compete and exchange information with one another. There are three basic genetic operations, namely, *crossover* (also called *recombination*), *mutation*, and *selection*. The procedure of a typical EA is given by Algorithm-EA.

Algorithm-EA

Procedure

`Initialization`:

  Set $t := 0$.

  Randomize initial population $\mathcal{P}(0)$.

`Repeat`:

    Evaluate fitness of each individual of $\mathcal{P}(t)$.

    Select individuals as parents from $\mathcal{P}(t)$ based on fitness.

    Apply search operators (crossover and mutation) to

      parents, and generate $\mathcal{P}(t+1)$.

    Set $t := t+1$.

`until` the termination criterion is satisfied.

End Procedure

In Algorithm-EA, the initial population is usually generated randomly, while the population of other generations are generated from some selection/reproduction procedure. Both crossover and mutation are considered the driving forces of evolution. Crossover occurs when two parent chromosomes, normally two homologous instances of the same chromosome, break and then reconnect but to different end pieces. Mutations can be caused by copying errors in the genetic material during cell division and by external environment factors.

Selection embodies the principle of *survival of the fittest*, which provides a driving force in EA. Selection is based on the fitness of the individuals. From a population $\mathcal{P}(t)$, those individuals with strong fitness have a higher probability of being selected for reproduction so as to generate a population of the next generation, $\mathcal{P}(t+1)$.

The search process of an EA terminates when a certain termination criterion is met. Otherwise a new generation is produced and the search process continues. The criterion can be selected as a maximum number of generations, or the convergence of the genotypes of the individuals. Phenotypic convergence without genotypic convergence is also possible.

2) Some Terminologies: Some terminologies that are used in the EA literature are described here. These terminologies are an analogy to their biological counterparts.

Population. A set of individuals in a generation is called a *population*, $\mathcal{P}(t) = \{\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_{N_P}\}$, where $\vec{x}_i$ is the ith individual, and N_P is the size of the population.

Chromosome. Each individual $\vec{x}_i$ in a population is a single *chromosome*. A chromosome, sometimes called a *genome*, is a set of parameters that define a solution to the problem under consideration. Biologically, a chromosome is a long, continuous piece of DNA, that contains many genes, regulatory elements and other intervening nucleotide sequences. Chromsomes encode a biological organism.

Gene. In EAs, each chromosome $\vec{x}$ comprises of a string of elements x_i, called *genes*, i.e., $\vec{x} = (x_1, x_2, \ldots, x_n)$, where n is the number of genes in the chromosome. Each gene encodes a parameter of the problem into the chromosome. A gene is usually encoded as a binary string or a real number. In biology, genes are entities that parents pass to offspring during reproduction.

Allele. The biological definition for an *allele* is any one of a number of alternative forms of the same gene occupying a given position called a *locus* on a chromosome. The gene's position in the chromosome is called locus (pl. loci). In EA terminology, the value of a gene is indicated as an *allele*.

Genotype. A *genotype* is biologically referred to the underlying genetic coding of a living organism, usually in the form of DNA. The genotype of each organism corresponds to an observable, known as a *phenotype*. In EAs, a genotype represents a coded solution, that is, a chromosome.

Phenotype. Biologically, the *phenotype* of an organism is either its total physical appearance and constitution or a specific manifestation of a trait. Each individual has a phenotype that is the set of all its traits (including its fitness and its genotype). A phenotype is determined by genotype or multiple genes and influenced by environmental factors. The concept of phenotypic plasticity describes the degree to which an organism's phenotype is determined by its genotype. The mapping of a set of genotypes to a set of phenotypes is referred to as a *genotype–phenotype map*. In EAs, a phenotype represents a decoded solution.

Fitness. *Fitness* in biology refers to the ability of an individual of certain genotype to reproduce. The set of all possible genotypes and their respective fitness values is called a *fitness landscape*. Fitness function is a particular type of objective function that quantifies the optimality of a solution, i.e., a chromosome, in an EA. Fitness is the value of the objective function for a chromosome $\vec{x}_i$, namely $f(\vec{x}_i)$. After the genotype is decoded, the fitness function is used to convert the phenotype's parameter values into the fitness. Fitness is used to rate the solutions.

Natural Selection. *Natural selection* is believed to be the most important mechanism in the evolution of biological species. It alters biological populations over time by propagating heritable traits affecting individual organisms to survive and reproduce. It adapts a species to its environment. Natural selection is concerned with those traits that help individuals to survive the environment and to reproduce. It causes traits to become more prevalent when they contribute to fitness.

3) EA Methods: EAs can be broadly classified into genetic algorithms [28], evolution strategies (ES) [29], genetic programming [30], differential evolution [31], and estimation of distribution algorithms [32]. Evolution itself can be accelerated by integrating learning, yielding memetic algorithms [33]. Today, the differentiations among diferent EA paradigms are getting blurred, since they try to improve the performance by borrowing ideas from one another [27].

The genetic algorithm is coded in binary strings, and crossover is its primary operation and mutation is also used. It employs a probabilistic selection scheme for the parents for mating, according to their fitness. The binary nature of the LABS problem is especially suited for the binary representation of the genetic algorithm.

On the other hand, the ES usually codes variables as real numbers, and mutation is the only genetic operation used. It typically takes the form of either (μ, λ) or $(\mu + \lambda)$ scheme, where μ is the number of children generated and λ is the number of individuals selected as parents for the next generation. The (μ, λ) scheme selects λ individuals from the μ generated children as the parents for the next generation, while the $(\mu + \lambda)$ scheme selects λ individuals from the pool of μ generated children and the λ parents as the parents for the next generation. Unlike the genetic algorithm, the ES always selects the λ best individuals as a population (i.e., the elitist strategy), and each individual in the population has the same mating probability.

Differential evolution is featured by the elitist strategy and multiparent reproduction. Each individual in the current generation is allowed to breed through mating with other randomly selected individuals from the population. Specifically, for each individual at the current generation, three other random distinct individuals are selected from the population to form a parent pool of four individuals in order to breed an offspring.

In estimation of distribution algorithms, there is no crossover or mutation operation. A probabilistic model is induced from some of the individuals in population $\mathcal{P}(t)$, and then the next population $\mathcal{P}(t+1)$ is obtained by sampling this probabilistic model.

The memetic algorithm, also called the cultural algorithm, is inspired by the propagation of human ideas and Dawkins' notion of *meme* [27]. The memetic algorithm may be implemented as an EA followed by a local search, and is also known as a genetic local search. The use of the local search can substantially reduce the total number of fitness function evaluations.

B. Our Proposed Evolutionary Algorithm

We now present our design of an EA for the LABS problem. Binary coding is a natural coding scheme for this problem. Each chromosome is encoded by a string. The classical genetic algorithm is inefficient due to the probabilistic selection/reproduction mechanism and probabilistic crossover/mutation operations. Some ideas from the ES and memetic algorithm are used to improve the search efficiency. Our proposed EA adopts the following features:

1) Crossover operation is not applied. Since there are many optima as well as numerous local minima in different regions of the fitness landscape, the crossover of two such individuals only leads to nowhere. Typically, two selected individuals for crossover are likely in different regions, and crossover degrades to random search.
2) Selection is elitic. The $(\mu+\lambda)$ ES scheme is applied. In the real-coded ES, the mutation strategies are evolved automatically by encoding them into the chromosome. In the binary-coded case, it is not very efficient to evolve the mutation strategies.
3) Two-point mutation is employed. Since we plan to apply a bit-climber (to be explained next) on the mutated individual, two-point mutation is applied. The two-point mutation operator changes two bits at two randomly specified positions of the string. We have two reasons for selecting the two-point mutation. First, one-point mutation flips one randomly specified bit at a time, which may be reset by the bit-climber. Second, the two-point mutation operation controls the variations within a certain range, which avoids the genetic search to be degenerated into a random search.
4) The bit-climber is applied as a local search step. The bit-climber is implemented in this way: One bit of the chromosome string is flipped at a time, and the fitness is computed for the new string; if the fitness is better than its earlier value, the new string replaces the current string; repeated until all the L bit flips are performed.
5) Partial restart is implemented to improve the genetic diversity of the population to prevent premature convergence, since the elitism selection strategy and the two-point mutation (which has very limited variation) may restrict the individuals

to some regions with local minima and premature convergence may occur. Partial restart introduces some randomly generated individuals into the population to increase the diversity of the population. Partial restart can be implemented by a fixed number of generations, or implemented when premature convergence occurs.

By representing binary sequences $\mathbf{a}_i$'s as ± 1-valued bit strings, the pseudocode of the proposed EA_for_LABS algorithm is given as follows.

Algorithm EA_for_LABS

Procedure Main

```
Initialization:
  Set population size N_P,
    number of children N_O,
    number of generations for each restart G_RS,
    maximal number of generations G_max,
    population size for partial restart N_RS.
  t := 0.
  Randomize a_i, i = 1, ..., N_P.
  for i := 1 to N_P,
    a_i := bit_climber(a_i), with fitness f_P(i).
  end for
  P := {(a_i, f_P(i)) | i = 1, ..., N_P}.
for t := 1 to G_max,
  if (t mod G_RS = 0),
    Randomize a_i, i = N_P + 1, ..., N_P + N_RS.
    for i := N_P + 1 to N_P + N_RS,
      a_i := bit_climber(a_i), with fitness f_P(i).
    end for
    P := P ⋃ {(a_i, f_P(i)) | i = N_P + 1, ..., N_P + N_RS}.
  end if
  for i := 1 to N_O,
    Randomly select a_k from P.
    Mutate a_k by two-point mutation.
    b_i := bit_climber(a_k), with fitness f_O(i).
  end for
  O := {(b_i, f_O(i)) | i = 1, ..., N_O}.
  Rank P ⋃ O in descending fitness order.
```

Take the first N_P individuals as $\mathcal{P}$.

`end for`

End Procedure

Procedure Bit_Climber

Input $\mathbf{a}$ with fitness $f(\mathbf{a})$.

`for` $i := 1$ `to` L,

$a_i := -a_i$.

Evaluate the fitness $g(\mathbf{a})$.

`if` $g(\mathbf{a}) > f(\mathbf{a})$,

$f(\mathbf{a}) := g(\mathbf{a})$.

`else`

$a_i := -a_i$.

`end if`

Return $\mathbf{a}$ with fitness $f(\mathbf{a})$.

End Procedure

The evaluation of the fitness function takes $O(L^2)$ operations for calculating $C_k(\mathbf{a})$'s. For the bit-climber, for each bit flip at a_i, $C_k(\mathbf{a})$ can be calculated from its previous value $C'_k(\mathbf{a})$ by the update equation

$$C_k(\mathbf{a}) = \begin{cases} C'_k(\mathbf{a}) - 2a_i a_{i+k}, & 1 \le i \le k \\ & \text{and } i \le L-k; \\ C'_k(\mathbf{a}) - 2a_i(a_{i-k} + a_{i+k}), & k+1 \le i \le L-k; \\ C'_k(\mathbf{a}) - 2a_{i-k}a_i, & L-k+1 \le i \le L \\ & \text{and } i \ge k+1; \\ C'_k(\mathbf{a}), & \text{otherwise.} \end{cases} \tag{7}$$

This reduces the complexity for updating all $C_k(\mathbf{a})$'s to $O(L)$. The resultant saving is significant, especially because each mutated or randomly generated individual is subject to L bit flips and fitness evaluations. For example, compared to direct calculation of C_k's, the computing time of the EA is reduced by a factor of 4 when calculating C_k's for $L = 31$ by (7).

IV. Results

Before applying the proposed algorithm for finding long LABS with low PSL, we first address the problem of which fitness function is most suitable for the task at hand.

For the sake of completeness, we also consider the sidelobe measure that generalise PSL and F first introduced in [36] and is defined as

$$f_3(\mathbf{a}) = \frac{1}{\sum_{k=1}^{L-1} (C_k(\mathbf{a}))^{\gamma}}, \quad \gamma \in \{1, 2, \ldots\}. \tag{8}$$

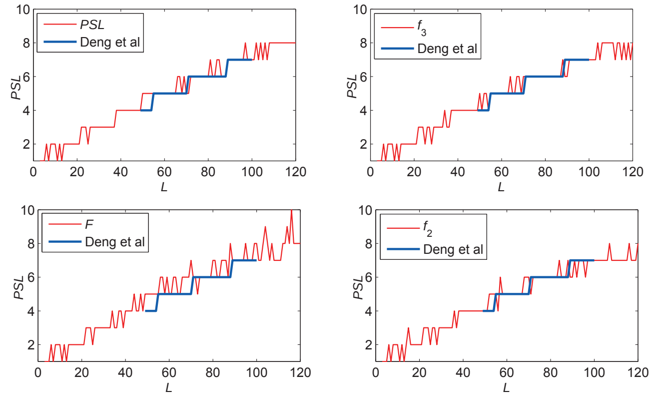
This fitness function considers all sidelobes $C_k(\mathbf{a})$, $k = 1, 2, \ldots, L-1$, but gives priority to the largest sidelobes. By setting $\gamma = 2$, $f_3(\mathbf{a})$ is equivalent to the merit factor $F(\mathbf{a})$. By setting a large value of γ, $f_3(\mathbf{a})$ has a similar effect as $1/PSL(\mathbf{a})$. In the LABS problem, many $C_k(\mathbf{a})$'s may have the same maximum value. The PSL criterion only considers this maximum value but ignores the number of peak sidelobes. In general, a different tradeoff between the PSL and the merit factor can be achieved by choosing a different value of γ. In the subsequent, $\gamma = 4$ is selected for generating all search results associated with the criterion f_3.

We set $N_P = 4L$, $N_O = 20L$, $G_{RS} = 5$, $G_{\max} = 100$, $N_{RS} = 10L$. Four different fitness functions, i.e., PSL, F, f_2 and f_3, for 5 random runs of the proposed EA are evaluated on a Linux system with Intel's Core 2 Duo processor. The results for $L = 3$ to 120 are plotted in Fig. 1. The results of Deng *et al.* [21] are also plotted for comparison.

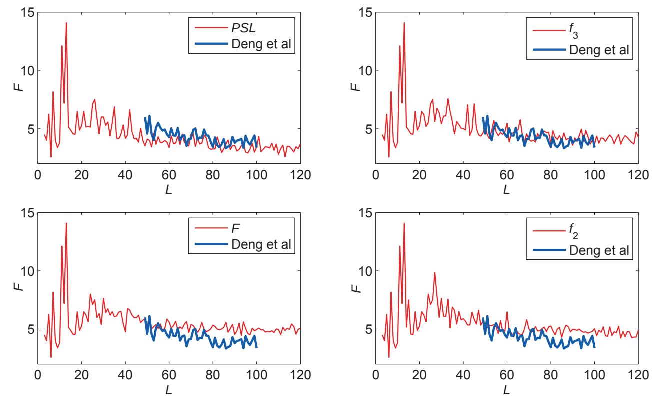
From Fig. 1, one can arrive at the following observations on the selection of fitness function. When PSL is selected as the fitness function, the F performance is the poorest. In contrast, when F is selected as the fitness function, the PSL performance is poorest. Better tradeoffs are achieved by the fitness functions f_2 and f_3. In particular, f_2 achieves the best tradeoff between the achieved PSL and F. It is interesting to note from Fig. 1 that f_2 is an even more effective fitness function than PSL, even if PSL is the objective to be minimized. This may be due to the fact that like most other optimization methods, the EA is more effective when applied to a smooth fitness landscape, and the resultant gain may outweigh the loss incurred by approximating the PSL criterion by f_2. Our PSL results for the interval $L \in [49, 100]$ are better than those of Deng *et al.* [21] for $L = 57, 72, 75, 89, 92, 93, 94, 97,$ and 99. Generally speaking, compared with existing methods, our proposed algorithm with the fitness function f_2 is more effective in finding improved or optimal solutions for the LABS problem. As will be shown subsequently, this holds true even for much longer sequences.

In Fig. 2, our PSL results are compared with the latest results of [16] and the optimal results in [5] for $5 \leq L \leq 69$. The results were obtained based on 5 random runs with the parameters given above. It can be seen that our results are much closer to the optimal results than those of [16].

Based on our survey on the LABS literature, there are only two papers [14], [13] reported useful LABS results for lengths beyond a few hundreds. This reflects how challenging the long LABS problem is. Therefore, the results found by our proposed EA are compared with best known PSL results in [14], [13] for $L \geq 106$. The PSL results for lengths 106 to 300 are listed in Table I. To discover longer LABS, our proposed EA was applied for some chosen lengths between $L = 303$ and 4096 for generating Tables II to III. Each result listed therein is the best among 3 random runs of our program.



(a)



(b)

Fig. 1. Best binary sequences of lengths $L \leq 120$ with respect to two criteria: (a) PSL; (b) merit factor F.

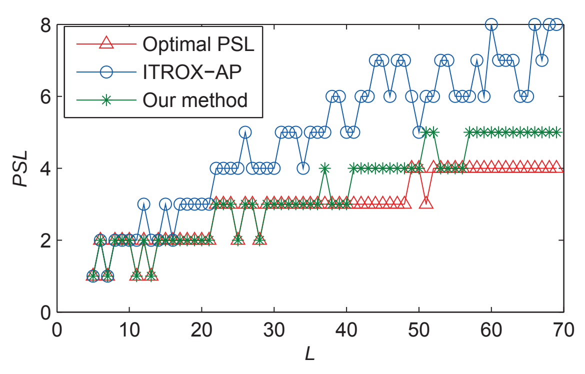


Fig. 2. Comparison of our PSL results with those given in [16] that are produced by the recent ITROX-AP algorithm and the optimal PSL results in [5] for $5 \leq L \leq 69$. Both the results of our proposed evolutionary algorithm and those of the ITROX-AP algorithm were obtained from the lowest PSL values returned from five random runs of the corresponding algorithms.

To reduce the computing time, the population and children sizes for longer lengths are decreased. For $L = 303$ to 1000, we set $N_P = L$, $N_O = 2L$, $G_{RS} = 5$, $G_{\max} = 200$, $N_{RS} = L$. The results are listed in Table II. When $L > 1000$, we set $N_P = N_O = 1000$, $G_{RS} = 5$, $G_{\max} = 200$, $N_{RS} = 1000$. The results for $L = 1019$ to 4096 are listed in Table III. Our record-breaking PSL results in Tables I to III are marked in bold and their associated lengths are marked with an asterisk.

For the sake of benchmarking, the best PSL results reported from the locally optimized cyclically shifted Legendre sequences in [13] and the systematic search in [14] are also listed side by side with our results in Tables I to III. From the tables, it can be seen that for the prime lengths considered, our PSL results are comparable to those obtained from the Legendre sequences in [13]. Notably, our PSL results in the tables are better for prime lengths $L = 109$, 137, 149, 181, 239, 241, 281, and 353. From the tables, it can also be observed that our PSL results are generally better than those in [14], especially for long sequences. Specifically, our PSL results in the tables are better for lengths $L = 300$, 304, 450, 500, 512, 550, 600, 650, 750, 800, 850, 900, 950, 1000, 1024, 1500, 2000, 2048, 2197, 3000 and 4096. In fact, the results therein are always no better than ours and it is very likely that their search algorithm is also far slower than our EA.

As an indication of the runtime complexity of our EA, the computing time is 58009 seconds or 16.1136 hours for $L = 1019$. For lengths up to 4096, the computing time required empirically shows a seemingly quadratic growth with L. Note however that we claim no rigorous complexity analysis results. In particular, the parameters have been adjusted to trade the performance for the search complexity, in case of long sequences. This flexible tradeoff is in fact one of the key advantages of the proposed algorithm.

V. Concluding Remarks

We have proposed an EA for tackling the problem of discovering long LABS with low PSL. The proposed EA design incorporates several features, including $(\lambda + \mu)$ ES-like scheme, two-point mutation, a bit-climber used as a local search operator, partial population restart, and a fast scheme for calculating autocorrelation. The results for using several different objectives or fitness functions were compared in terms of both PSL and merit factor. Our algorithm can effectively find optimal or near-optimal PSL results for LABS of lengths up to 69, and significantly outperforms the recently introduced ITROX-AP algorithm in [16].

LABS of selected lengths up to 4096 searched by our algorithm have been tabulated in detail, and they have lower PSL values for many lengths than the previous records reported in [13] and [14], which are the only known papers addressing the long LABS challenge, to our knowledge. Our PSL results are often better (and no worse) than those reported in [14], especially for large lengths. The effectiveness of our algorithm is comparable to that based on the Legendre sequences in [13]. Yet our PSL results still provide lower PSL for many lengths. It is noteworthy that unlike [13], our algorithm is not restricted to prime lengths and its effectiveness does not heavily depend on having a good sequence construction (e.g. Legendre sequences [13] or quantized chirp signals [14]) as its initial guess. Hence it can readily be adapted to tackle various extensions of the LABS problem. It is not only effective for the long LABS problem, but is also promising for handling generic sidelobe criteria, sequence sets with low cross- and auto-correlation levels, etc. In addition, it is convenient to control the required search time by adjusting the parameters of the proposed algorithm so as to achieve a flexible tradeoff between quality of search results and available computing resource.

REFERENCES

[1] S. Mertens, "Exhaustive search for low-autocorrelation binary sequences," *J. Phys. A: Math Gen*, vol. 29, pp. L473–L481, 1996.

[2] N. Alon, S. Litsyn, and A. Shpunt, "Typical peak sidelobe level of binary sequences," *IEEE Trans. Inf. Theory*, vol. 56, no. 1, pp. 545–554, Jan. 2010.

[3] K.-U. Schmidt, "Binary sequences with small peak sidelobe level," *IEEE Trans. Inf. Theory*, vol. 58, no. 4, pp. 2512–2515, Apr. 2012.

[4] M. J. E. Golay, "The merit factor of long low autocorrelation binary sequences," *IEEE: Trans. Inf. Theory*, vol. 28, no. 3, pp. 543–549, May 1982.

[5] G. Coxson and J. Russo, "Efficient exhaustive search for optimal-peak-sidelobe binary codes." *IEEE Trans. Aerosp. Electron. Syst.*, vol. 41, no. 1, pp. 302–308, Jan. 2005.

[6] F. Brglez, X. Y. Li, M. F. Stallmann and B. Militzer, "Reliable cost predictions for finding optimal solutions to LABS problem: Evolutionary and alternative algorithms," in Proc. 5th Int. Workshop on Frontiers in Evolutionary Algorithms (FEA'2003) under JCIS'2003, Cary, NC, USA, Sep. 2003.

[7] J. Jedwab and K. Yoshida, "The peak sidelobe level of families of binary sequences," *IEEE Trans. Inf. Theory*, vol. 52, no. 5, pp. 2247–2254, May 2006.

[8] C. J. Nunn and G. E. Coxson, "Best-known autocorrelation peak sidelobe levels for binary codes of length 71 to 105," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 44, no. 1, pp. 392–395, Jan. 2008.

[9] P. Borwein and R. Ferguson, "Polyphase sequences with low autocorrelation," *IEEE Trans. Inf. Theory*, vol. 51, no. 4, pp. 1564–1567, Apr. 2005.

[10] C. J. Nunn and G. E. Coxson, "Polyphase pulse compression codes with optimal peak and integrated sidelobes," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 45, no. 2, pp. 41–47, Apr. 2009.

[11] W. H. Mow, "Best quadriphase codes up to length 24," *Electron. Lett.*, vol. 29, no. 10, pp. 923–925, May 1993.

[12] S. W. Golomb, *Shift Register Sequences*, San Francisco: Holden-Day Inc., 1967.

[13] K. V. Rao and V. U. Reddy, "Biphase sequence generation with low sidelobe autocorrelation function," *IEEE Trans. Aerospace Electron. Syst.*, vol. 22, no. 2, pp. 128–133, Mar. 1986.

[14] A. Dzvonkovskaya and H. Rohling, "Long binary phase codes with good autocorrelation properties," In Proc. 2008 Int. Radar Symp., Wroclaw, Poland, May 2008, pp. 1–4.

[15] M. A. Ferrara, "Near-optimal peak sidelobe binary codes." In Proc. IEEE Conf. on Radar, Apr. 2006, pp. 400–403.

[16] M. Soltanalian and P. Stoica, "Computational design of sequences with good correlation properties," *IEEE Trans. Signal Process.*, vol. 60, no. 5, pp. 2180–2193, May 2012.

[17] S. Prestwich, "A hybrid search architecture applied to hard random 3-SAT and low-autocorrelation binary sequences." In Proc. 6th Int. Conf. on Principles and Practice of Constraint Programming, LNCS 1894, Springer-Verlag, 2000, pp. 337–352.

[18] B. W. Kernighan and S. Lin, "An efficient heuristic procedure for partitioning graphs," *Bell Syst. Tech. J.*, vol. 49, no.1, pp. 291–307, 1970.

[19] H. D. Schotten and H. D. Luke, "On the search for low correlated binary sequences," *Int. J. Electron. Commun.* (AEÜ), vol. 59, pp. 67–78, 2005.

[20] B. Militzer, M. Zamparelli, and D. Beule, "Evolutionary search for low autocorrelated binary sequences," *IEEE Trans. Evol. Comp.*, vol. 2, no. 1, pp. 34–39, Apr. 1998.

[21] X. Deng and P. Fan, “New binary sequences with good aperiodic autocorrelations obtained by evolutionary algorithm,” *IEEE Commun. Lett.*, vol. 3, no. 10, pp. 288–290, Oct. 1999.

[22] A. E. Kocabas and A. Atalar, “Binary sequences with low aperiodic autocorrelation for synchronization purposes,” *IEEE Commun. Lett.*, vol. 7, no. 1, pp. 36–38, Jan. 2003.

[23] S. Wang and X. Ji, “An efficient heuristics search for binary sequences with good aperiodic autocorrelations,” In Proc. IEEE Int. Conf. on Wireless Communications, Networking and Mobile Computing (WiCom’07), Sep. 21-25, 2007, Shanghai, China, pp. 763–766.

[24] J. E. Gallardo, C. Cotta, and A. J. Fernandez, “A memetic algorithm for the low autocorrelation binary sequence problem,” In Proc. 9th Annual Conf. on Genetic and Evolut. Computat., London, England, 2007, pp. 1226–1233.

[25] B. Natarajan, S. Das, and D. Stevens, “An evolutionary approach to designing complex spreading codes for DS-CDMA,” *IEEE Trans. Wireless Commun.*, vol. 4, no. 5, pp. 2051–2056, Sep. 2005.

[26] T. Koike and S. Yoshida, “Genetic designing of near-optimal training sequences for spatial multiplexing transmissions,” In Proc. 10th Asia-Pacific Conf. on Commun. and 5th Int. Symp. on Multi-Dimensional Mobile Commun., 2004, pp. 474–478.

[27] K.-L. Du and M. N. S. Swamy, *Neural Networks in a Softcomputing Framework*, Springer, London, 2006.

[28] J. Holland. Adaptation in Natural and Artificial Systems. Ann Arbor, Michigan: Univ of Michigan Press, 1975.

[29] I. Rechenberg. Evolutionsstrategie–Optimierung Technischer Systeme Nach Prinzipien der Biologischen Information. Freiburg, Germany: Formman Verlag, 1973.

[30] J. R. Koza, *Genetic Programming*, Cambridge, MA: MIT Press, 1993.

[31] R. Storn, K. Price, Differential Evolution–A Simple and Efficient Adaptive Scheme for Global Optimization Over Continuous Spaces. Int. Comput. Sci. Inst., Berkeley, CA, Tech. Rep. TR-95-012, Mar. 1995.

[32] P. Larranaga, J. A. Lozano (eds.), *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*, Norwell, MA: Kluwer Academic Press, 2001.

[33] P. Moscato, *On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts: Towards Memetic Algorithms*, Tech. Report 826, Caltech Concurrent Computation Program, Caltech, Pasadena, CA, 1989.

[34] F. Hu, P. Z. Fan, M. Darnell, and F. Jin, “Binary sequences with good aperiodic autocorrelation functions obtained by neural network search,”*Electron. Lett.*, vol. 33, no. 8, pp. 688–690, Apr. 1997.

[35] I. Dotu and P. van Hentenryck. “A note on low autocorrelation binary sequences.” In F. Benhamou, editor, 12th Int. Conf. on Principles and Practice of Constraint Programming (CP 2006), Nantes, France, September 2006, Springer, LNCS vol. 4204, pp. 685–689.

[36] U. Somaini and M. H. Ackroyd, “The peak sidelobe level of families of binary sequences,” *IEEE Trans. Inf. Theory*, vol. 20, no. 5, pp. 689–691, Sep. 1997.

Ke-Lin Du (M'01-SM'09) received the PhD in electrical engineering from Huazhong University of Science and Technology, Wuhan, China, in 1998. He founded Xonlink Inc. in March 2014. He was Chief Scientist at Enjoyor Inc. from 2011 to 2014. He was on research staff at Department of Electrical and Computer Engineering, Concordia University, Montreal, Canada from 2001 to 2010. Prior to 2001, he was on technical staff with Huawei Technologies, China Academy of Telecommunication Technology, and Chinese University of Hong Kong. He worked with Hong Kong University of Science and Technology in 2008. Dr. Du has coauthored three books (Neural Networks in a Softcomputing Framework, Springer, London, 2006; Wireless Communication Systems, Cambridge University Press, New York, 2010; Neural Networks and Statistical Learning, Springer, London, 2014). He has also published over 50 papers, and has 4 U. S. patents and 14 Chinese patents. Currently, his research interests are signal processing, wireless communications, neural networks and machine learning. He is currently an Affiliate Professor at Department of Electrical and Computer Engineering, Concordia University. He has been on the editorial board or been Associate Editor of several journals, including IET Signal Processing and Circuits, Systems & Signal Processing. He also serves on the editorial board of the Chinese edition of IEEE Spectrum.

Wai Ho Mow (S'89-M'93-SM'99) received his M.Phil. and Ph.D. degrees in information engineering from the Chinese University of Hong Kong in 1991 and 1993, respectively. From 1997 to 1999, he was with the Nanyang Technological University, Singapore. He has been with the Hong Kong University of Science and Technology (HKUST) since March 2000. He was the recipient of seven research/exchange fellowships from five countries, including the Humboldt Research Fellowship. His research interests are in the areas of communications, coding, and information theory. He pioneered the lattice approach to signal detection problems (such as sphere decoding and complex lattice reduction-aided detection) and unied all known constructions of perfect roots-of-unity (aka CAZAC) sequences (widely used as preambles and sounding sequences). He has published one book, and has coauthored over 30 led patent applications and over 160 technical publications, among which he is the sole author of over 40. He coauthored two papers that received the ISITA2002 Paper Award for Young Researchers and the APCC2013 Best Paper Award. He is currently the leader of the HKUST Barcode Group which won the Best Mobile App Award at ACM MobiCom'2013 by developing a novel picture-embedding barcode app, called PiCode. Since 2002, he has been the principal investigator of 16 funded research projects. In 2005, he chaired the Hong Kong Chapter of the IEEE Information Theory Society. He was the Technical Program Co-Chair of five conferences, and served the technical program committees of numerous conferences, such as ICC, Globecom, ITW, ISITA, VTC and APCC. He was a Guest Associate Editor for numerous special issues of the IEICE Transactions on Fundamentals. He was an industrial consultant for Huawei, ZTE, and Magnotech Ltd. He was a member of the Radio Spectrum Advisory Committee, Office of the Telecommunications Authority, Hong Kong S.A.R. Government from 2003 to 2008.

List of figure captions:

Fig. 1. Best binary sequences of lengths $L \leq 120$ with respect to two criteria: (a) PSL; (b) merit factor F.

Fig. 2. Comparison of our PSL results with those given in [16] that are produced by the recent ITROX-AP algorithm and the optimal PSL results in [5] for $5 \leq L \leq 69$. Both the results of our proposed evolutionary algorithm and those of the ITROX-AP algorithm were obtained from the lowest PSL values returned from five random runs of the corresponding algorithms.

TABLE I

SOME RESULTS BETWEEN $L = 106$ TO 300, OBTAINED FROM 3 RANDOM RUNS OF THE PROPOSED ALGORITHM.

L	PSL	PSL	F	Hexadecimal form
106	7		5.0295	00 542ADD9C19B68C2D2471D4F60
107	7	7 [13]	5.1805	19 8F1C3FC4FF5C8B25D4D952529
108	7		4.6957	B7 6DA4FA9F578883BD89DC75E14
109*	**7**	8 [13]	5.0429	042 10CC60FF325305D1D306A9756
110	7		4.9631	067 71127548108B1F5E92F03E496
111	7		5.6260	6D0 D79D790123A8553918FC6C936
112	7		5.3153	A86 AA75E4DEDFBD016E371DB21C9
113	7	7 [13]	4.8514	0B91 5E59AB611FFDD319B0C2E0E4A
114	7		4.4114	1589 90DCB2256F59EF7145947BE1E
115	7		4.8729	7F20 3675D7532E45308C1C2796B52
116	7		4.3974	3F60 45CB8F29851309CD56D1B45A0
117	7		4.2832	1F62F 9F1BB3F0430279C56552D30AD
118	7		4.5355	099F0 E0362DF99884B985BED75D7AE
119	7		4.7235	2C565 18B4E68C259FF9D8BFC0EBE2E
120	7		5.8632	CEF38 EAFF7203153C2FD2175264DA5
121	7		4.4421	0BE168 FD21975B1D913B5BFEE75419A
122	7		4.6368	3FFF28 DB2C3DCAD54C7A3A4ACCCF81A
123	8		4.6897	A6E4EA 7CDDE716359EBD10486F907F5
124	8		4.9987	ACD623 DACF045220A138791537604D7
125	8		4.8646	071E973 E64A7AE6BF9980D27C8AE95F6
126	8		5.0272	2228C01 7346E3E74A8179F90D4D36D35
127	8	7 [13]	4.8965	443DFCE 10A622702A703694CAFA36D96
128	8	8 [14]	4.8075	A68D156 1CB0186A85A083FC8EF732026
129	8		4.6328	1E1E54F0 AE35F71FD6666B66A9902B7C8
130	8		4.8872	2F3C397F F4609489B8DC2D851641B9455
131	8	8 [13]	4.9627	76A76A09 518DBAEE99F83EDF431CDE581
132	8		4.3430	E488D3AA 62B27353FA683E43B295E7EF1
133	8		4.6995	0D92B2472 2B25595491C6B1387C003DF3F
134	8		4.4380	055DA0568 40A5356C7F0E61379E4C0E7CD
135	8		4.5134	430D3E2DC CE1336972F2102558A2A87FA4
136	8		4.4720	730F3124F F1350D6C48F8F960100D2AD54
137*	**8**	9 [13]	4.2273	0599026FAC 2D54ED7485C048707A21961CE
138	8		4.3341	0613C9C3D1 152AC7D322092F70775FF19E9
139	8	8 [13]	4.6602	0BFAA19133 000D149EE962CA31F8B4C6B0B
140	8		4.6009	14C91EFAAB 540B7216139806878A4878B9A
141	8		4.4180	0E57939E879 4FF58AFF242254F6CB2E48E2A
142	8		4.4789	27384E1D0AC 4203368C05FA9BD149829ABAE
143	8		4.5584	398F073B238 BA81F2448A1720927BED494A9
144	8		4.2492	FA1E6F892F0 8F2A5462989AC734123D31203
145	8		4.4696	17E42BB84666 3EB3E383424D0680C29AA59A7
146	8		4.6239	336196CB1E2C E31A5A9D43A8BD9D950007C00
147	8		4.3409	5922CBC9F357 4BE8CFF8EEB40297390973F39
148	8		4.3703	D1A1CFF74837 787694056465C5B8A4A21340E
149*	**8**	9 [13]	4.5531	1415F0E18FE14 0712E75328421324CAC97B32B
150	9	9 [14]	4.7209	1994ED80CEAF3 837D4CFB1E3D5F2F40540C97D
151	8	8 [13]	4.3663	6FB488568F32C ADD641E1AB2F78FA777467711
152	9		5.2509	208231DA2413C 9E8FC89FC495336A9CBCF0B2A
153	9		4.8206	1750BB45D32C2A B082DF8180831BE7E6697B1A4
154	8		4.2517	0A14B8E8ABD389 F4D22E71349C93B04FD8E9004
155	8		4.2704	02DBA28BBE1CC1 49CEE3721E654EAA1604848A5
156	9		4.5986	2276F919A0F4F1 52DD15498B483AD14633CE3FF
157	9	8 [13]	4.6368	13BCC89691AF69E DF81CCE28550F183920BB2BA4
158	9		4.9473	32ED143AD90CDC9 3B353FD6B79CF5105587E9AE0
159	9		4.7396	78E320A0078C468 BF390152F624DAA27734B6D13
160	9		4.8780	AD5A92659732430 2CBCEC260A2841E1FE239FC23
161	9		4.7789	12331B84BDB7B402 60CCB241E09172873D5552F18
162	9		4.5674	00540AC0FE408BF2 E8B03739CE69B2932B4F1C696
163	9	9 [13]	4.6760	3585E52CBD46F2F3 2BE31EA222044BDDB060FA2C3
164	9		5.0367	E85E957A353429F9 45968EFB404759E67E7B8ECEE
165	9		4.2834	1A3170699871AF2B3 60570ADABA8483F99E6F65B7D
166	9		4.5337	3CAE9C8BDB9F786DE ACB526691035DF40F62D14CE2
167	9	8 [13]	4.7190	0344845C1D11FA45B B319DAA9851CC693EBC210D6F
168	9		4.6057	609F6CEEC7744DBB5 3AE6478548782C30B4BFA821B
169	9	9 [14]	4.7099	0C466755A02AE1ACA0 1C92B60E25C93EFF8F1F28325
170	9		4.6598	1B6907DF2E0428551B 3223A81B2ACA77398E1487A0C

L	PSL	PSL	F	Hexadecimal form
171	9		4.4077	5F3C5C3C370189ACBF 618CBB21DD4A90AD21A654424
172	9		4.7685	1E7CD350F164741AD9 AE652A2B45911803B18B7F104
173	9	9 [13]	4.7932	1B96A4ADDBF5FB30A07 4B2C1574F4532C07F0433A399
174	9		4.6564	30F24FA47B129602434 14DD71ECC81BEF50239AB55CB
175	9		4.2832	69254BED3E8E1F84E30 E4EFE316440B799D9AFAA36FB
176	9		4.2549	10F9CF566589BB4A0AA 5DCB7BDBAB19BE1BAFD2CF40C
177	9		4.3416	154FD8B689BAE3BE56AC D25BEBC7C0C0BB8484DDEAE67
178	9		4.1331	2C7EED76637297ECE103 2CAE2DF5A64BC8A51883F9507
179	9	9 [13]	4.3287	07A40FBE21A31F277379 96EDFDA6565D4150AA75CF2BC
180	9		4.3385	446CE19E984D8861125F 94F3B9D450F8FEC0DAA74AFE2
181*	**9**	10 [13]	4.5832	1B079F428BA1567E8D08F 5DF157C6731BFB3DAD6961124
182	9		4.2347	0FFD4F2F4C216624FA5D6 A68BA8AD42522499C381906E3
183	10		4.4879	661197BD30EC41A7BF524 A0709E0B97DB9FA35F551746B
184	9		3.9331	A6C8128BF37ACA8F370AD 63861019A5203B383A8A5C075
185	10		4.5464	052E5A06E24C46E087B31D 55FED1E72489877FA1C4B47E9
186	10		4.6239	14A5EEA662200CA4336A8E 1C905922FD612CF3CC83F05C3
187	10		4.5285	2DDA5535CDF3DFE2DD190C 9E03D9994B8B424E7EE2851FB
188	10		4.7176	74509B5F48E09E6EE2AD31 4A66E2B9C4B102F5A3FFBE3E7
189	10		4.8090	1D1A5428AD626FCD160C272 59DDD805B7D40700E18C580DA
190	10		4.4645	17312DBDAEBB8A2F664B97F D21C22F33F67D0786AB541818
191	10	9 [13]	5.0125	448606030B4C4E95D8C53B4 DE36C5C296756CBA8325FF00F
192	10		4.7950	A2D079EE30185FA85DE6467 8B2E9C19B13D2B6991A01A029
193	10	10 [13]	4.5205	0356A8D9D62999F613EDB6C0 D684E0206786428A3BF85C4C0
194	10		4.4288	08A6999A3325EA3714386A2B 7180F14F120BD38049EDEFF96
195	10		4.2316	19126AB6BEA9F76BA4C1EF07 E0D2EA584FD8A9CEC62B6E71C
196	10		4.3048	3C2FD50D00D44A1C64496B6D 8B0EFA8C6FE4D8B19165E23BB
197	10	10 [13]	4.6267	1016AEDCF5EC0CA1E841D7552 F457FB4C9B79F678CC6D363BD
198	10		4.5052	2A88EAFC16C7B4A411788BF5B 798BC4836B44A1840C9C931E2
199	10	10 [13]	4.4889	4B1A9382A18BB9FD5C60A10F7 4A00CEC3180F5126F41EDB64D
200	10	10 [14]	4.5496	A52DE56BB6911EE34183B1D91 0608C43D13FAE13A13E745544
201	10		4.0080	0 DFAFD351B1F0E2A322A74A30A 7B90B7E40CA194D63ACFD2669
202	10		4.3033	1 5768E01358C821A3C140465C3 FED9225B40BD8833A71BB53B8
203	10		4.3033	3 7AD5210DEFB193936EE1F2325 802C852AAA2FC3187D9079A5C
204	10		4.2310	9 26DB5FAA7AD71A4A1931A91C3 B902260739C8F800F5751C4D0
205	10		4.7050	04 C2AABB65964BDDF6031603DE4 49C948489DE43C1E942851C31
206	10		4.1288	39 7D85973B7D04F776B6868BBC5 DE72C47658A47820324793B5E
207	10		4.1886	45 8303A80984E57076E2EF16C36 9EC4097893F6D5308455E1957
208	10		4.3126	71 D8BCA7635D21AA1FA1A5C6E94 4EF2BA642EE8040C01EDD3061
209	10		4.3542	0A2 7A2D520642791E4A288B3637B 2087A14F58C55EFE347CFD850
210	10		4.6392	3D0 7247DFBC2FCBFDE1AB159A9B9 CD50C3592C5F7C4B3C9856314
211	10	10 [13]	4.2899	521 403FD3104DB895E0D83A2C363 280A2169E75772A3CC8ADB37B
212	10		4.3567	AF1 CB148B2DB2B65156F3963680C 6EF237EFF9B217F1A3E079D8A
213	10		4.5170	04BC 54B6C279762DD879E85E962C0 DD988D773D8D7C754428EFFFE
214	11		4.4872	2E4C 36AF9C68E25FBFE069F165F57 F3B0B691882748D50B73B4736
215	10		4.3584	59E5 D3D5C736C636B91B930F73E3B B15620140BE42D9D024AFB852
216	11		4.7725	7C00 C0430BC19BEB2520BB67A388F 3B47D67452BBA56934EA95B94
217	10		4.1364	1F60F 67AA4427449AEB6FB3C7131E1 5ACC420AA12D282E078DB902E
218	11		4.5373	2AAAA 597ACB23CD6B518E16C0E85CB DFC4ECE0FC812033D921E6C00
219	11		4.7798	3E7E2 500C4A7F0AD9733B60E197309 160EB205358A42ECA62AEEB17
220	11		4.6414	1D6CA 8A9B49E7F2566BC5C2310018B 8DE90BF02A139CBF0832CDB12
221	11		4.4096	0673AF 1860002C5EA107E6B685B38A8 F2CACAAC955D81FA6607D225C
222	11		4.5273	3ADC4C 6C46E38C7094E8FF9552FB26F EADBFCD1234DE0D53F6C49783
223	11	10 [13]	4.5216	46B8CE B7A9545D25BD89F37E75809FF 7772BB3038782C30C197A6997
224	11		4.5253	84B288 80D82D0AF19E9C18C6F6A152F 08073A7DD426CB0ECC9291776
225	11		4.6462	0F3F080 960C003CEC2B3628DE13AF24D 02EB37E14A4CE5D58D51EE8E6
226	11		4.5990	08F7EC2 358BCDE176C9455054ACE5048 A2168F5B599A38F803247686F
227	11	10 [13]	4.5641	642652D D2F46F407DF63C089A79B22D2 061C084B634ABB54189AA38CC
228	11		4.6365	1171615 9828388A9652829FFA130DAFC 6976228B0CF3CEE59A81F8172
229	11	11 [13]	4.5664	1FB417C1 0FA5834140572D8C6B38450A6 59D3C54A7E2C40EEA660F99B0
230	11		4.5299	070B2A15 10BB7DF8973BB7EC9388F2B2F E2E6B6035DC16BCBB84A47906
231	11		4.5819	61FEE277 96F38A954A0976C262D0D9F26 606344364AD2FC2181D15455B
232	11		4.4556	63D6DD11 06ECAB0CFE5A68AD21DCB8D9B FDE3E6C07ED23A9442E2F73F8
233	11	11 [13]	4.5423	09E993054 BB2E36746A6C3843035044231 D4B85753BC0F884BE437F5901
234	11		4.3122	3F83FCEE4 8701A329258336DA9E64304AD DA7942838971518D5558813BC
235	11		4.4529	6FDCEF49D AD916CB37840D43AAA795F25E 4930A3C72ED48776371F5011F
236	11		4.3418	76B3EABB8 1A847C4DA6B6D204C68407E30 5CC22FD9F148372B64587284C
237	11		4.3488	1B6F29F90C 608FDC6E618E1108B60323724 EB6C58363F5E8545553E4868F
238	11		4.6992	06EAD42CD7 AFCDC47B1EF4DF1236319AF5F 4EAF0C5B411525A6C2DF9411F
239*	**11**	12 [13]	4.5457	7AA8918194 FAFD27B4515ECE1CF274F5D83 5581EA19C84A1FB1245E76981
240	12		4.6512	826E6DFF3F D1D316DED80CEF68C9AB09DC4 7AB2B8E50AA552E4A349A1F87

L	PSL	PSL	F	Hexadecimal form
241*	**11**	12 [13]	4.3921	030D7CE8ECF 18D184F798C6E925B5704AF4D A2153A769D9074480E80E002B
242	11		4.3464	3A452CF3DC2 89C379144C0E8BA92E4B808CE D496E69311012B053F30E9D7E
243	11		4.6415	35C9D9FAFFB 96A551B71C8390A37759CC45F 484156FA82F926B26887C70F2
244	11		4.2224	5C8AF0D5BF9 D541F6B82C8DE3C6A18267037 AFC92FDDAFB632C94B3DE26F6
245	11		4.3994	029CCAFFEDB3 109A073885E8E81FE68305F54 D0A3A741B0B163E2925AB33A5
246	11		4.4980	1EF242563567 E4FE52FF00D8EF33CBCD77E15 76F0C1098B36CD62E154F3BAA
247	12		4.7024	79A0DBE18DFD 836CDEDA5BD77238DFEBD5081 8A26713F2BD6C58E61E8BA98A
248	11		4.4414	E721F0BD8B58 2CEDF5730D2FE3147225DD445 8008182C9DF924BAD460DD581
249	12		4.6491	0CF423FF49B08 9C5EA2D4EC04B0E66C888F18D 533CAA2E2900A9A61697974F7
250	12	12 [14]	4.4816	0622264A2C88E 147AAAE46E531F0C33FC0B1A0 DB7ED694F30685E9A52EF7BE1
251	11	11 [13]	4.7291	6DB7A22D9933A C0168A3171654A6CF1F8D0AAF 7B9485F179D73F919E19C17DF
252	11		4.0020	FAC07D4D1E117 B4E1677CC923412105413BBAF 205A3373BC454AE4E34DA35E3
253	12		4.7163	0B0B99CDB21AB6 94CC3F2887D7B83036A9F8965 07976154B800C000C7B4CB986
254	12		4.5440	3FDC4870527391 C0A10C3348A5FE518A2C5B82C DAB91F0D6927A426457D03B72
255	12		4.5902	21234DBAD3F352 6E8501E19ED0B66077A6F2563 99C6293D902818A2AA03B3D10
256	12	12 [14]	4.8075	C66E72E53E702C DE4A16F649491AAA790FE155D 07F7FCDD00CD3B2D1C7E7EEBD
257	12	12 [13]	4.8338	005288A05F7398A 14DF4441798F8FB49B3667832 30292F30CBC295A2B7C6AF90B
258	12		4.3421	12FFAA7F6EBE859 30B776153844C33C4B98FD1C5 1F1B8A5C19464B69723B58879
259	12		4.4596	53018CFB0FEAA29 8564FB299AC381DF0814BDDD8 99732E6960AFD0F4ED7F74BF1
260	12		4.6492	3251DD64471D3FD C39A0D21300321D834F0A9743 B65AB60D44F9D6362EE1057B3
261	11		4.2672	14BC454E90DBA496 2C47443AAC52565A717174C59 62CCCE4B8021C8FF79F0BFFE0
262	12		4.4557	398EAAE6A915AD58 529B8310D39A097DFEEDF926F 7483B0FCF6F8E4C3B936F2278
263	12	12 [13]	4.3344	4FF9990608130226 479E4F03723535A5B15359542 FFA0F358B68B579EF051E71AB
264	12		4.6814	9C6DDF143C077B17 F4734C3A7EA9E3E9ED1809CC6 8162ADCA48DF512AAD04DBDAC
265	12		4.6007	08EAD88E9392BA6D9 6DA7383814819FA67BDB996FC 1BFDCF4447BEA74ABE2C1D5E8
266	12		4.3118	31E2917C735D17D56 B82FF294FBD41AC9886129D19 3CBCB3E467268890373729081
267	11		4.2652	5953B8519D5AF3326 5E8E8F6FAF6132E431B1EF8E7 FB862D2B61B4104B2FA0F8053
268	12		4.5912	CAC8A6B8FF1404596 F16F1D0CC409087C0A547B697 32F1E348DB642BF11D722ACDC
269	12	12 [13]	4.5903	0A28FD2E951E47AD41 4AF7B8C6D698458B626F3CF48 B2120F883A3010D9B2EE26727
270	12		4.3836	3FB24A8175445D1BB7 C06D440EB73C93289BD73E9E6 D6FEE56E3D0C6C6B47B8E243F
271	12	12 [13]	4.5028	37C39615AF6E13408F 3588BEAF3D5489C300D976626 EB64ADF7B3E0BCCAAFA6F61CF
272	12		4.8267	6C7B85DE551BF08A6B A18C345823177B5F330F135DE AF745BF938D0FC694B4BB3F64
273	12		4.4617	0A93D1616BF17B44A52 F8CE619CB008B3C7FF649E3DA 1C46C6063649E0055503444E8
274	12		4.4683	27408228CFAA388F25E 578392D64A2C34EDD9960B2C0 BD707BC42A37B13533F9DAD7F
275	12		4.3881	63C00D35432AE2D07DD 1C97957581650E3C2E42627FF 1B6E8D642A57BD66F39DF620D
276	12		4.5032	E7979B9C2CAD25005A7 8250229573D322E8926478D40 FD8DF098A6A20EF4CBB3C8668
277	12	12 [13]	4.4352	09E400E0C161B9B53D99 5C46D0FCAC99D6A2ABC9FC336 FF186A63FAEF1422D7AA5CB3C
278	12		4.5790	2983D92E3BC584B0B25C 8EF00D7AE6E82B0A5EFFCF47E F4A4330DDD2895725174EAE4E
279	12		4.3192	472EE8D41894A158C233 97B9C4D0F60BD024053C903CD 98404E8D9EA497C924F74F5D5
280	12		4.2535	AA6A5AD2B143072C7EB6 FACC08A4E6372094623FCC11F E7D14F9B6B8F516EE0BA3C360
281*	**12**	13 [13]	4.2325	1250B4B5611E8A70A14E0 029B6C2FBC3CACD455008C9EE 64823558B71DE7D170667ECCD
282	12		4.4273	0730D195C56AB2B8EAD3C 87A62ADF9BB7CA20F40C203FD A0110F2ECC5A4CE865C842C85
283	12	12 [13]	4.4747	1D7F0AF6BC0D0B140051D 561F00D0B349B923F742304EB 3B414B231998E9865147BA49E
284	13		4.4307	F1FDD9F7EC77385877248 7B37A554A01F899BB87518AAB C23F1AD29F1150D7BE6926DB2
285	12		4.3735	184FEF1665B368529D2C82 90CBFF2376484A77472BAB2A6 550C3507E03441C1C35148877
286	12		4.3830	09D8AEB8A1D5B720AF0EB3 3ADB4CFFB7537B97758B14A4C F808F1A9DE7A4CD0090FF361F
287	12		4.1706	5864B95DB71C461D0C2D77 64F462406B4234868C55E4AAB 21B9E05E2704D01EF5C29B3EF
288	12		4.3858	F7690297DED44B34F5BB4B CE683CE9ADBBE399B0620F0C1 F33251F83B88BBB0D554AA11E
289	12		4.2981	0E19848BE384366860DCC64 03A970452A0771277C12F4CEF 5A426AEDB5F150D44081DB67D
290	13		4.5622	003DE0169C79C436192C860 1178171966EDF2D770488BC1C B3F2E213153B95F8B92B695AA
291	13		4.5621	206FFE4AA9297C4625E9AC5 C134A1ECD978C0D95C8947A64 773BC470C2C02F08300571D56
292	13		4.6410	D4D94856B076680ED4C83BA E0389F6EA37561A969BE68D09 1E7AF95981EC2382BE7BF75B3
293	13	13 [13]	4.7599	1029280A08AC82FE48A56B5D F3E49084CBE0F4BCC6BC4D509 8FAA35173B7448CB974338F9C
294	13		4.3961	244EB3EFDD33162F72EE399C D55D00DD65F8532F1408FBC3F 4DE2DAD3C3C9C048A3595AC8F
295	13		4.3992	6595483A55578127E146F1C3 EA9E6FAC40E93CD95BCD5A026 BFB9CCDDDA5FE9CC27E258072
296	12		4.3203	F41C6C21538021B8E1792641 732B13207A97F373C94C2D710 854FD5608FB27364FBCEBAC91
297	12		4.2953	0BFAE9E38FA87671109C805E0 8EFF9F42764E4385F3A255273 50760A835665C96A4082DB659
298	13		4.5771	25C65CB1C766535BE5843B56D 9C843BF83698C284C37F1B1A9 300541600BAA5C2A1FC55F899
299	13		4.4166	5EBB52FEB5DEB2AFCD1AA60B3 566DBE0CA61DDED7C22623611 C53F727860FA230BA0F067ED4
300*	**13**	14 [14]	4.4074	5AAAF7F284E43542CCFFD096B A42E7C784BA0BA6E9CC7DE4FC 5E34433349D60837235C11164

TABLE II

SOME RESULTS BETWEEN $L = 303$ TO 1000, OBTAINED FROM 3 RANDOM RUNS OF THE PROPOSED ALGORITHM.

L	PSL	PSL	F	Hexadecimal form
303	13	13 [14]	4.3507	0 AD95352CC22999A6FB0C43086 B95BE0DE162E8D5AB039EBC2E 36421E1FC014FBDD9CFCDFCB4
304*	**13**	14 [14]	4.0676	E 4126CDA6FA380553FAF855670 737149D716A118A39D317F56B C5AC27200D231605658C0B489
350	14	14 [14]	3.9458	2559C84C9F842 38329784CD15E5FD165E71FA1 955F6AA3FD68D163C020C83DF AE5CCC39D0F6696A047369252
353*	**14**	15 [13]	4.2075	1AB8EE1405CBAA 2301CE6AF55A9367F56F975F0 F8237A4B217DED35D90E5816E 94F1ECC71B333DBBDA5036461
400	15		4.3908	614AC33E81BAFF8832AAED86D 42E54FDEAC2B011A64125B93F 84F5F90F9E3397BE311F1C8E9 B7677B7879AC555A8F93B2C14
449	16	16 [13]	4.0547	0880B2380EF45 1BF4967DAADB49A4E7942E196 3BA7280283209CB9B3C119C17 CCA40C033D59A1CFA1F115F10 5AF6785070AB72740DAAECD47
450*	**16**	18 [14]	4.4235	074927D1884C8 1F5A9DBD1150A3269BAEF6F77 B3D233BB90D073CED9159C75A 3DE3643E9BDB10E53DE50DC11 73C6C1ABE141EAFE4F8F510EF
500*	**17**	18 [14]	4.3442	2AB2323A3ECF2D5BAFB96296C EB8576E9977DBFF2513B71581 86692BB8EE8599610DCF3BBC6 5FF085A3624FFD18784AE2A7F 0F53A8729D5FCC682CD079B1A
512*	**18**	20 [14]	4.2656	0C6 0D7D4513078969028B0AD1E7D 4B9484DCB8B4314EF3C423890 59629957F19DBF1BD1FCC147A C967014099B8D967A55086B68 2FD366610BFC62EEFCCD903EA
547	18	18 [13]	4.3408	1A1122346FA6 B24D0058AF50DF76297DFB4DD 1FD629835796CABBA9E7B960D DF0ADAA3EF781CC68787065EE 84E33C3EC2F328FCA7FB90948 5ECE5C52EA75E8B38528EC9D0
550*	**18**	20 [14]	4.0695	1B75CCF68F659A 1E87B8EA72B3ACBA7B46C702 F9D5EBD17699C445D399CA346 4A3FCF7D616F61077FF12A907 FC89835664FB5421298018595 A5FABE096D9591DC8B1B971A9
600*	**19**	20 [14]	3.6753	947D1CA2F0D6605BFE64A83D0 DE36DF1124823FB586FB3D62B EAA03228D4E4A8600B1C1B84B B75345932030798EAD7362F79 B956E53BC32227E2EEE501921 3196948FD9B4E8C7CEFE1C32C
650*	**20**	21 [14]	3.9239	3629519BBA27A AC842603610546BE0C050D6B5 4E6381E44A00E9C76DCE5A6E1 FBD4E1B53B245C71765A69CA5 661453631900E59DF4B4BD866 DCF8106BA56FBE36006054BB8 D835EBA3A93EFA01ECD517F19
653	20	20 [13]	4.2287	0042DAE58D12B9 D67C18A56FB5FC1A3CC9F0D0D E5EB1EC464C10E86AE5F94159 2BBF8ADF8E5BD87B4AF88DAC6 33B1053BEDE8D1677832775DB 472499DDC0F47108524DA7AA9 1FABE45FE5104544488DD51AE
700	22	22 [14]	4.1524	755D7F2A5202805BA539F3368 3185FD0232E2CDC92725588DE 5ECEBB4381929896E01781621 779C512CD67B61D874F3AD782 8A1A5E43440216E1292CD9A0C 9358146E7624EE7555C3B9471 F04C2F9FDE6D1E15DAC0E7DC1
750*	**22**	24 [14]	3.7603	0D2C305FBDF8B E28B8F56A2C52EFD20DD56B5F 58096AF298169E090BAD48814 0162BB5DBB32A1CBD22EBE507 7E76C268C938C23791F41F7C7 731E999DC0DC7550640CFC6F1 F66CCB41BDA727A0FEE629F47 7EB2F6311C54DBC8F106724B6
751	22	21 [13]	4.1537	61831AE4D5773 3A33AAE6785AB1421C2C70402 F98BFFB1D3F4C557F60DAE504 848FA28617F2A0F967C82304C DFB3B856C8EBD8BEAD682F05A E8449FC6C21C355B97C8294A3 875DBDFF3D6C9A56D62DE43F7 6EF61E8F9CD3510C86F4DE665
800*	**23**	26 [14]	3.7481	9BD821BDB337C783D2FABFCF8 BE46CCBF2B4D8E87BE39B4DAB 43CBDE96919EF1DCE0A34E45D 85C49DB1EA102F744E437B80C BE718414F3AC82113DF1FEFD5 FF9AE0245D10A97D94CDD048A 336B9745A0AE9821FAAF92E12 FD477533B32E90AD5BEAA5A46
850*	**24**	25 [14]	3.8096	3EE1A834B9586 872DF36A9E31B629E9CBD6C7E D8BAB07453F76F25486379032 DC8FFFDAFF90FA881B8E616F2 6E6799AFEFF1A82A37574DEA9 D01B23FD9838DF3207EAF287C D160CB1A8052DC3851E0850FA 8C6302397D6C3756B96EC6796 EC347F2536D44E154E23F4AA2
853	24	22 [13]	4.0854	02768D10E7D43B B00F7D393A61C6F78AC926759 4B7F6C536B2553B581B4E1FC9 9553236F6CCC5A7710BF4B20E F382C7274E1222AD3E6676769 2719CBC8E037D5C80151FB719 E3808E0F4D84AF5392E81F508 B2AF69DFD57C1F4DD3F95EB3A 9DFEBF2936C2040261E51D1C3
900*	**25**	26 [14]	3.7623	D3B8E76EBC7A737A3A210F4E4 47E066618EB529E7F468B8F7C 6C98B68A16646A51508853AF5 24A01FCA8A1A85EF8DB3E4792 3EA2F5667ECFC0816743B9379 12DB00B6508585107D1AA082A C3A9BA4E16218D792490B0CAE 585B7EE11EDD201121BB05F5D C63B889EAEC44169AC775B6DC
950*	**26**	28 [14]	4.0438	2D1A46DB07002 A95132B1A8CD2EF8392BCC99D FBAAC044EFEC8B22AA553D3C7 F7F9C6CCAECA104107D527744 F7A4C3ED429A82E5ACF4390B1 929EB0B5757D303B22C77E91C A64EBBF96C7ABBC2C37F635FB 46B9AE073DA6987796C3323F4 86C35FF67BEA09FB0EE1FE6AE 170AD040FF2F449A4539A3C63
953	25	25 [13]	3.8493	1AC865E34EECA7 6F31F9C2ACB1F520F3522FF35 1D7D58F75D848C3381CDF3F17 65B5CB15C722C43DA7EE6E1EA 06B6DF1820F42FADEA195F3E8 A56CC703346B757F1886E0DB8 42F19F293A2C8A0C31AD9B0A7 DFD801181F997A767A026899A EDB73DA9BCBF86FDCAAD48083 50D45A040A2E218999BD5536B
1000*	**27**	28 [14]	3.7873	7FD311D01DD74BE054B4E2DAD B387727FDB7A3965B9420081E 01CC0C1C3ADC740A94B98DD5D 919F9C1E25F7422E81DFFF045 CC65374FED83C360625A47996 1B979ACDE60E2C090A499CE7E AAB94C89526CA1718D53A4073 EA8B05CD9218A14B2832BB48E 7048D9D11ACE683D415310124 AB5A3FC3A57155DA2056D5982

TABLE III
SOME RESULTS BETWEEN $L = 1019$ TO 4096, OBTAINED FROM 3 RANDOM RUNS OF THE PROPOSED ALGORITHM.

L	PSL	PSL	F	Hexadecimal form
1019	26	24 [13]	4.1390	5DF5B 7DB1608F87A6C00E33A6AAE88
				2F273C56FCFD5242F0A60D974 CEBE75733A782AC3F6687CC4E 53EC18BA1E7EA820C84B2A1CC
				4742E4ADE9C89A72E36A44130 2F26315A438C72E2955B0C5AA 16C90DFFD00BD37A813852651
				A95FDFC4A371F0EBF4341AC6F F5DEF996611CC12E2B2DC4200 DAC88AB44E44D26252FA9F789
1024*	**28**	30 [14]	3.9683	4A3850 61EB56D8C3A37BEDFF2EEBC30
				96B47CF2CE9EBA6C28A6895AF 4CDF08090AB612DA8043C3D1F E644D50A15E908692AC4DC095
				218D398A6A66B389D16C8A6BC AF26896612DF211D48CBC027C 7C451B6B5B14EECD199CE823E
				63C07C4E20AECF7513F41329D 56706E05F66D22A6EEC152A83 0F9378B07D7F3DC2D9FF88C08
1500*	**35**	40 [14]	3.7316	77CDCD88C3F33F08D81BBBDBE 38632CE50E2E8B8D05E31018A 7A131386A39129745983C417A
				98B5B323ADD46AF5DD8147BCE D377CCA8DBE4EC7E2C2B51719 426AFB2270695C9B213A72719
				F5A23C0A52316EA3FF7A02381 7247CB76C9C200C5A92C33CF3 D1405D09103FF0AB18B33CE47
				C8D02E8AF221EB42D0C11A8BC C8229BD2B305AF900DA4B6BFD 31D4F8B6FA3184A384933AD93
				C512E7E5487593EF9EFD96D7A A3DC06A6C1C310256B572BFD1 5DDC5F503E8940E4D6734D3CB
2000*	**42**	44 [14]	3.6193	9300DC650BB35F244E59742D8 848E894E9BC0CB6E07FE3700C
				1AA19DAB48DE771363D8F8D3C CB7FA78CE77054202A3DE0B08 7572813A1CB889437130C723F
				CFFD7E53BDF26CA3A73ADCBF8 89A612D32BA3AE9112F25E981 7FC933E833A50D7EF85916D44
				6F25526C767ECC52CA9E590D2 DA7222A97C4FCCA1A64DFD474 C018C3DAA150F2286B10EB12A
				031D07357D53866B24D6C2156 109A40AED50D7F388ABF376CE C0D155125070F70C26DF3C76A
				D94F1531053E29DDD2A02B041 C062263BD95698150CC8697DA 03B20B2C6689097320BA14FBC
				D9425121CBC7AB6AFEFE38105 571F9A740A03A7895BDE60645 E96C607A11C35B0792F588740
2048*	**42**	44 [14]	3.5387	DA67ACA857A4 B796F2F16FCC6A2B5551A473F 92C9A73B73E254ABB40295752
				464E144537D7536C12FE744D8 DB9588889629E7673DEA4E8EE F23ED4EC00FA7E5C6BE33D913
				2A69DFA2B80690E6F5260C231 64F65D4942DCF36C70B4A30CB E7CF02DB21B23FA0A5F19AD2E
				7BC61CF82B36B1F17CA56E206 76707AB7E8F6BF4CBE25248F3 8048AD63CFA3BE8C26309FC6F
				7E057FFB9A8D152D8760C86A2 A6AF0B683FBFF41F4F9A87DC8 3DDBB7DB858FC94422B1E867C
				748911C572FCEB38E2432D41B 2C39FAB52BB2558EC98DF7A18 181C43D4205A339F904668288
				B06D49401871EC06C3C0AA2E2 316BE7F546B79D9C9D37A2CF9 3128422D7125D8B84B69A717A
2197*	**45**	47 [14]	3.6423	0331A80FD5ED35094DC259258
				1CA4954F3B24D3E19BD96C272 76AC577596F82274B74FADA2F 2A040059E64D3AC0269E71231
				7E767010E525E7D677D278F9F 8A3924EEB505E7D420822B3F0 30474F2FCF38B9588087863B1
				B9E248F223E0749ED065C6C99 4030B285BEE06AFF726BDFB80 6F5AE3A65151644AD93AADA4B
				27E310E80D9B2598AB4CCE2AE 1F10480A21695A6832F8AC0AF FB5A995500F8D4EDF1DAB6E0D
				533A69B210A185EA3E3474AE1 7E532288A2B82F8885584F098 51135AC5D5ED48012AFD907CC
				166E0268015AD30A13866F896 3D616584CDAA15D7670C7936E 770C895BB9FFDE94BAF8DE130
				E405FCB9F20FC36356ADFDA33 E99BB14ABEF78DA6BB10FCA03 89F2DEF119F2B7215C6590AE1
3000*	**52**	58 [14]	3.3608	C22BEEF73C56B9DC59F6D5AFF 803AEF187021AB81871E9124E B44DB7568D56738D4262B74A1
				77157EC6581848C87107C611B 110E4AD281CE0A7B66ED5831B E04CC0F76D1BABFE6566217A9
				F0A599ACE0DF1B032BA7B1887 568284569B84B691CC2A453E2 FB42BE07DCEFF5E5CF797F2D2
				80319F8BFF4BDAF9E2E7BFBFA ABB375BD036311507CAB30D94 8AA39A4A4394D4E2FEC507DC7
				BCD4A8BC2A18CE06CE2FB5124 E9B8EC9BB5B04FBE280BCCF12 FFB9A1B81937A6D8B4FF4E36F
				A163EC514A0A93BF4E1B7F89E FF05CC31496505BF9ED52D248 0576CCD70A7A7B320EF160A2B
				A3822067A6D1EECF2AAC2E53F 386414300931F9E63A12B327E 528EE8E95833A7375E258E632
				B2B1F702E0A4B383FADA845A5 66D5D18CB3160CDD24274F2D8 AA55616D5E20DCE34D80D38A2
				31067F5554C970AD724B0FECE 9220944EEC7C13EB3D7E03303 4E53D593813FFE157C17F8666
				E37569D603E668938A73AD9D8 B0CBC31DF93A4F262A3621118 7EE7A48E00EBD41102F1A4E9B
4096*	**61**	68 [14]	3.4589	E30A5D894A09A4CE0D11987E FC7E8DC88127C078FBD569A4A
				D05AB26D86A2D067C1E274783 B891CBF64617E0906673F029A ED144133B3FF48DF2DB8A1878
				6780075E9C2B0CC46E6D0DA62 3CF1F50F1DF94177C28076F3C E44BC24C69D242E8D6F49F678
				E71C2D4D72C9412C828734AA3 9CA28EA2A7E5891B451ADA9B2 408E666BA052C81509DE81789
				7E4AF9FE4F504846D80D6B14C EEBDD9402A35C03AFD4EAE97B 7ECB690094681EFD13837398A
				CECAA9AB5FC10682B00CA74BD 15B5C0D7C53BAF35BF70612CB 4DDE55EB4CF2F028596ED8382
				3F5D1A73463B9953326AE6950 CF1299AB6ACB432887A56E9F0 42957BAE604C003E982152DFE
				AFA75968C0D8B0FEAA2ED33FC 20DE73FBA4E21F154CB291291 58F8BB5B9977C57B6F77A7363
				4D9164A6FEA9647EAA1E1D631 14B6BA1E9F065D66E5F5BF15B 0D46EF9CED3216DB9DF0298E1
				CFBE0AF7596E9EB4BCBBBDA10 8A2B6088380B8D73797F9E9DB 094FCC06FF0544F46E261FE4E
				F60AABCA0A32A5D1694B818B0 3A6D5351B28BAF523D1AE65D6 048136003CFBA56CF22E0E1A2
				F2973C8163731272219255826 1DC2BEC886EBBBD73B5D1EFC2 9BB7E91F72964943D6D3560C3
				A8E20D11EC5A81C106E04D5F5 9218D9FD9D823B118AD4FB1D6 C1435461E338D9F171B337E5D
				D7320CCD9CFE5DC651051E0F6 678550BA09F9892E76D6E17C4 9ECD63F71B71FF351EEAF6DEB