Turk. J. Math. Comput. Sci.
13(1)(2021) 129–134
© MatDer
DOI : 10.47000/tjmcs.825565

Approaching the Minimum Distance Problem by Algebraic Swarm-Based Optimizations

Serap Şahinkaya[1,*], Deniz Ustun[2],

[1]*Department of of Natural and Mathematical Sciences, Faculty of Engineering, Tarsus University, 33400, Mersin, Turkey.*
[2]*Department of Computer Engineering, Faculty of Engineering, Tarsus University, 33400, Mersin, Turkey.*

Received: 13-11-2020 • Accepted: 28-05-2021

Abstract. Finding the minimum distance of linear codes is one of the main problems in coding theory. The importance of the minimum distance comes from its error-correcting and error-detecting capability of the handled codes. It was proven that this problem is an NP-hard that is the solution of this problem can be guessed and verified in polynomial time but no particular rule is followed to make the guess and some meta-heuristic approaches in the literature have been used to solve this problem. In this paper, swarm-based optimization techniques, bat and firefly, are applied to the minimum distance problem by integrating the algebraic operator to the handled algorithms.

2010 AMS Classification: 11T71, 14G50, 68P30, 78M50, 80M50, 90C27

Keywords: Minimum distance, minimum-weight codeword, BCH codes, optimization, heuristic, bat algorithm, firefly algorithm.

1. Introduction

In 1948, Claude Shannon, published ”A Mathematical Theory of Communication,” a seminal paper, which was about reliable data transmission over noisy channels [12]. Efficient and reliable data transmission, which can be done by some error-control techniques, are one of the main interests of coding theory. Error detecting and correcting capability are very important feature of a code and it is determined by the minimum distance of the code. Computing the minimum distance of a linear code C of large length is a difficult problem in coding theory. In [14], Vardy showed that this computation is an NP-hard type. The problem of finding minimum distance is getting harder when the size of the code grows. Therefore, some meta-heuristic algorithms have been used to approach the problem. In most of the existing literature, genetic algorithms are used for the considered problem. As far as our knowledge, among the algorithms in the literature that are based on swarm intelligence, only the ant colony algorithm (ACO) was used for the minimum-weight codeword problem [4, 5]. It is well known that there is no heuristic algorithm which can perform good enough to solve optimization problems, please see [13] for details. . Therefore, it is natural to try the other swarm-based optimization techniques for the considered problem.

In this paper, bat algorithm (BA) and firefly algorithm (FA) are applied to the minimum distance problem by integrating the algebraic operator to the handled algorithms. Most of the papers in the literature uses codewords as a search space for the minimum distance problem. Recently, generator matrices were considered as a search

*Corresponding Author
Email addresses: serap@tarsus.edu.tr (S. Şahinkaya), denizustun@tarsus.edu.tr (D. Ustun)

space, which turned out to be a better approach than using the codewords as a search space, please see [1] for details. In this work, we also consider generator matrices as a search space. In coding theory, the BCH codes or BoseÔÇôChaudhuriÔÇôHocquenghem codes form a class of cyclic error-correcting codes that are constructed using polynomials over a finite field. Effectiveness of the presented algorithm is controlled by running the algorithm on BCH codes since they are the standard codes with known minimum distance values [3, 9]

2. Preliminaries

A binary linear code C is an k-dimensional vector subspace of the n-dimensional vector space $\mathbb{F}_2^n$ over the finite field $\mathbb{F}_2$. The minimum distance d of a linear code C is calculated by the following formula:

$$d = min_{i \neq j} d_H(\mathbf{c}_i, \mathbf{c}_j),$$

where $d_H(\mathbf{c}_i, \mathbf{c}_j)$ is the Hamming distance between codewords $\mathbf{c}_i, \mathbf{c}_j \in C$. Number of the positions that differ between two distinct codewords is called the Hamming distance and the number of non-zero entries of a codeword is called the weight of a codeword. A linear code C is given by parameters $[n, k, d]$, where n is the length, k is the dimension and d is the minimum distance of the code. For a linear code C, a non-zero codeword of minimum Hamming weight is called a minimum-weight codeword. It can be easily obtained from the definitions that the minimum distance of a linear code equals to the minimum weight of a non-zero codeword in the code. Therefore, in the literature, the minimum distance for linear codes is also known as the minimum weight codeword problem. For a given $[n, k, d]$ linear code C, number of errors that can be detected by the code is $d - 1$ and the number of errors that can be corrected is $\lfloor \frac{d-1}{2} \rfloor$ [10]. Therefore, for a given block of length n and dimension k, the code is desired to have the minimum distance as large as possible. The minimum distance problem, also known as the minimum weight codeword problem, determines the codewords of weight M or less, in an $[n, k]$ linear code C, for a given integer M.

A linear code can be presented by providing either a basis or a generator matrix whose rows form a basis for the code C. More precisely, any codeword $\mathbf{c} \in C$ can be obtained by a linear combination of k-basis codewords, that is

$$\mathbf{c} = \alpha G,$$

where G is $k \times n$ generator matrix and $\boldsymbol{\alpha}$ is an k-tuple vector which is also called the information vector and $\mathbf{c}$ is an n-tuple vector, called the codeword. It is known from [8] that for a generator matrix G of an $[n, k]$ binary linear code C, $I_i G$ is also generator matrix for C, where I_i's are $k \times k$ invertible matrices. Therefore, it is natural to use generator matrices in minimum weight codeword problem. As far as our knowledge, [1, 6] and [7] are the only papers that in which generator matrices are used in place of codewords as a search space.

3. An Algebraic Approach to the Swarm-Based Optimizations

Swarm intelligence is a very powerful technique to be used for optimization purposes. The optimization algorithms based on the swarm intelligence are the flexible and robust to internal and external changes. Moreover, when some individuals in the population fail, they can be self-organized through the swarm intelligence. In the last decade, various popular swarm-based optimization algorithms, bat, firefly, grey-wolf etc., were presented to the literature. In this section two swarm-based algorithms, bat and firefly, are examined algebraically. The reason of the algebraic approach is; finding the minimum distance problem is not continuous and the classical version of the BA and FA can not be used for this problem. Therefore, an algebraic approach is needed for using these algorithms in the minimum distance problem. An algebraic approach is inspired from the algebraic differential mutation operator [11]. In [11], the classical differential mutation equation

$$y_i \leftarrow x_1 + F_i(x_2 - x_3),$$

was adapted to the algebraic differential mutation for a finitely generated group G with a binary operation $\star$. More precisely, for every population individual x_i, a mutant y_i is generated as follows:

$$y_i \leftarrow x_1 \oplus F_i \odot (x_2 \ominus x_3),$$

where $F_i \in (0, 1]$ is the differential evaluation (DE) scale factor and x_1, x_2, x_3, are three randomly chosen distinct population individuals, all different from x_i. The operators $\oplus, \ominus, \otimes$ are the algebraic operators defined as follows. $x \oplus y = x \star y$, $x \ominus y = y^{-1} \star x$ and the multiplication $z = F \odot x$ satisfies $|z| = \lceil F.|x| \rceil$. If $F \leq 1$, the sequence of generators in a minimal decomposition of z is a prefix of the sequence of generators in a minimal decomposition of x, and vice versa, when $F > 1$. In this paper, G is a group of permutation matrices and binary operation is the classical matrix multiplication. The algebraic operators $\oplus, \ominus, \otimes$ are defined as follows: $P_x \oplus P_y = P_x P_y$, $P_x \ominus P_y = P_y^{-1} P_x$, where P_x and P_y are permutation matrices. It is noted that multiplication of permutation matrices is a permutation matrix so permutation matrices are obtained after applying the $\oplus$ and $\ominus$ operators. The magnitude $|P_z|$ is defined as a minimum number of the shuffles that are applied to the identity matrix for obtaining the matrix P_z. This definition makes sense because permutation matrices can be obtained by shuffling the columns of the $n \times n$ identity matrix. The multiplication $F \odot P_z$ is defined as follows: after obtaining the scalar number $k = \lceil F.|P_z| \rceil$, k columns of the $n \times n$ identity matrix is shuffled.

3.1. **Algebraic Bat Algorithm.** Bat Algorithm (BA) [15], based on the echolocation behaviour of bats, is one of the meta-heuristic algorithms that is used to solve optimization problems. In the first place, the original BA was proposed for solving problems with continuous real search spaces. Although various types of bats can be found in nature, all of them have a similar behaviour for navigating and hunting. More precisely, they use their natural sonar's for finding their prey and discriminating the different types of insects even in complete darkness. The two main characteristics of bats, decreasing the loudness and increasing the rate of emitted ultrasonic sound while finding prey, have been adopted for designing the algorithm. In a d-dimensional search space for BA, the position of each virtual bat (i) is symbolized by x_i and the velocity vector is represented by v_i. The velocity and position vectors are updated during the iterations. New position and velocity of a bat at iteration step (t) is determined by using the below equation:

$$x_i(t+1) = x_i(t) + v_i(t+1) \tag{3.1}$$

$$v_i(t+1) = v_i(t) + (x_i(t) - p(t)) f_i \tag{3.2}$$

$$f_i = f_{min} + (f_{max} - f_{min})\beta$$

where β is a vector that is achieved randomly with uniform distribution with the range $[0, 1]$. The symbol $p(t)$ is the current global optimal position and $f_{min} = 0$, $f_{max} = 1$. The balance of the global and local search capabilities for the heuristic optimization algorithms are very important to achieve the optimal solution in search space. So, the adaptive parameters are integrated to the BA. The expression for the local search strategy is given as follows:

$$x_i(t+1) = p(t) + \varepsilon A(t) \tag{3.3}$$

where ε is a random number from $[-1, 1]$, $A(t)$ is the mean loudness of the population. Furthermore, the global search process is performed by controlling the loudness $A_i(t+1)$ and pulse rates $r_i(t+1)$.

$$A_i(t+1) = \alpha A_i(t)$$

$$r_i(t+1) = r_i(0)[1 - exp(-\gamma t)]$$

here, α and γ are constants and $\alpha > 0, \gamma > 0$. Moreover, $A_i(0)$ and $r_i(0)$ are initial values of loudness and pulse rate, respectively.

In this subsection, an algebraic bat algorithm (A-BA) is proposed for determining the minimum distance (equivalently minimum weight) of BCH codes. The equations (3.1), (3.2) and (3.3) are adapted to the algebraic bat algorithm (A-BA) as follows:

$$P_{x_i}(t+1) = P_{x_i}(t) \oplus P_{v_i}(t+1)$$

$$P_{v_i}(t+1) = P_{v_i}(t) \oplus (P_{x_i}(t) \ominus P_p(t)) \odot f_i,$$

$$P_{x_i}(t+1) = P_p(t).\varepsilon A(t), \tag{3.4}$$

here, the permutation matrices $P_{x_i}(t+1)$, $P_{v_i}(t+1)$ and $P_p(t)$ denote the new position, velocity and current global optimal position, respectively. The parameter $P_p(t)$ of Equation (3.4) are the permutation matrices that have dimension greater than 1 and $\varepsilon A(t)$ is a scalar number. This situation creates a problem while translating the operation + of Equation (3.3) to an algebraic operator. Therefore, in Equation (3.4), the scalar multiplication for matrices is used in place of $\oplus$. The pseudoceode of the algebraic bat algorithm (A-BA) is given in Figure 1. The calculations of the minimum distance of some BCH codes by A-BA is given in Table 1.

Algebraic Bat Algorithm (A-BA)
Begin
set initial position, velocity and other parameter values
while (Iteration value reaches stop criteria)
randomly produces the frequency for each by using Eq. (3.5)
update the velocity value for each virtual bat by using Eq. (3.10)
update the position value for each virtual bat by using Eq. (3.9)
if $rand > r_i^t$
update the position value by using Eq. (3.11)
end if
compute the fitness value
if ($rand > A_i^t$) and ($f(x_i^t) < f(x^*)$)
replace the position with the new one
update r_i^t, A_i^t by using Eq. (3.8) and Eq. (3.9)
end if
choose the current global best solution
end while
save the best solution
end

Figure 1. Pseudocode of A-BA Algorithm

3.2. **Algebraic Firefly Algorithm.** The firefly algorithm (FA) is an important tool of Swarm Intelligence that has been applied in almost all areas of optimization, as well as engineering practice. The firefly algorithm (FA) was presented by Xin-She Yang in 2008 [15] and it is inspired from the behavior of tropical fireflies and their flashing patterns. FA is flexible, straightforward and very easy to apply. The fireflies charm each other via their brightness and a firefly moves towards more attractive firefly. Thus, many fireflies can gather around a firefly having more brightness. The FA is constructed on this phenomenon. The movement of a firefly (x_i^t) towards more attractive firefly (x_j^t) is determined by the formula

$$x_i^{t+1} = x_i^t + \beta_0 e^{-\gamma r_{ij}^2}(x_j^t - x_i^t) + \alpha \epsilon_i^t \tag{3.5}$$

Algebraic Firefly Algorithm (A-FA)
Begin
set initial positions of fireflies
compute light intensities
while (Iteration value reaches stop criteria)
for i=1 to N, number of fireflies, **do**
for j=1 to N, number of fireflies, **do**
if $f(x_j) > f(x_i)$
move firefly i toward j by using Eq. (3.13)
end if
end for
end for
evaluate new solution $f(x_i)$
determine the current global best solution g^*
end while
save the best solution
end

Figure 2. Pseudocode of A-FA Algorithm

BCH Codes (n, k, d)	Ant Colony, [4]	New Ant Colony, [5]	Algebraic Bat Algorithm (A-BA)	Algebraic Firefly Algorithm (A-FA)
(127, 64, 21)	24	21	21	21
(127, 57, 23)	24	23	23	23
(127, 50, 27)	27	27	27	27
(255, 115, 43)	58	52	43	43
(255, 107, 45)	60	56	45	45
(255, 99, 47)	62	56	47	47
(255, 91, 51)	68	53	52	51
(255, 87, 53)	66	62	55	53
(255, 79, 55)	69	68	55	55
(255, 71, 59)	70	68	63	61

Table 1. Comparasions of A-BA and A-FA with (New) Ant Colony Algorithms For BCH Codes

where r is the distance among to two fireflies. The second term in the expression is caused by the attraction and β_0 is the value of attractiveness at zero distance $r = 0$. The third term is the randomization that is depended on α and the randomization parameter ϵ_i^t where ϵ_i^t is the vector of random numbers obtained from a Gaussian distribution at time t.

In this subsection, an algebraic firefly algorithm (A-FA) is proposed for finding the minimum distance of BCH codes. The equation (3.5) is adapted to A-FA as follows:

$$P_{x_i}^{t+1} = (P_{x_i}^t \oplus (\beta_0 e^{-\gamma r_{ij}^2} \odot (P_{x_j}^t \ominus P_{x_i}^t))).\alpha\epsilon_i^t. \tag{3.6}$$

Likely the standard bat algorithm, classical equation of firefly algorithm can not be directly adapted to algebraic firefly algorithm. More precisely, some parameters of Equation (3.6) are permutation matrices that have dimension greater than 1 and this situation creates a problem while translating the last operation + of Equation (3.5) to an algebraic operator $\oplus$. Therefore, in Equation (3.6), scalar multiplication for matrices is used in place of $\oplus$. The calculations of the minimum distance of some BCH codes by A-FA is given in Table 1.

4. Comparasion With Ant Colony Optimization

In the literature, only the ant colony algorithm (ACO) was used to optimize the minimum-weight codeword problem in terms of swarm based intelligence [4, 5]. Therefore, another swarm based optimization techniques, firefly and bat algorithms, are examined for the considered problem. The performance of the proposed algorithms are confirmed by a comparison with Ant Colony and New Ant Colony Algorithms by running over ten BCH codes with known minimum distance given in Table 1. The proposed algorithm is run on a workstation with Intel Xeon 4.0 GHz processor and 64 GByte RAM. The parameters of the algorithm, which are the maximum number of iteration and population size are fixed as 1000 and 100, respectively. It can be seen from Table 1 that the algebraic firefly and the algebraic bat algorithms outperforms the ant colony algorithm. It can be clearly seen from Table 1 that the proposed algorithms achieve the true minimum distance value for many BCH codes. When A-FA is compared with the A-BA, it can be seen that the A-AF outperforms the A-BA. The A-FA computes a better minimum distance than A-BA for BCH codes (255, 91, 51), (255, 87, 53) and (255, 71, 59). The comparison of standard FA and BA was studied in [2] and in this work, it was shown that FA is better than BA. One obtains the same results for A-FA and A-BA because A-FA enforces the local search well, but sometime this is not the case for A-BA. Moreover, A-BA does not take into account the better solution for each virtual bat and so the virtual bats randomly move in search space without considering its previous better solution. Therefore, the algebraic bat algorithm miss the better solution.

5. Conclusion

In this article, two swarm intelligence algorithms, bat and firefly are used to approach the minimum distance problem. Algebraic operators are used in place of classical operators for the proposed algorithms. The proposed methods are applied to the ten BCH codes with known minimum distance values. It is clearly seen from the Table (1) that algebraic bat and firefly algorithms present better performance when compared to the any colony. Many of the papers in the literature use codewords as a search space to calculate the minimum distance of a given code. But, in the presented algorithms, the generator matrices are used as a search space as suggested in [1].

Acknowledgement

The authors thank to Tarsus University for providing workstations with high computation performance used in the optimization of the minimum distance problem.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this article.

References

[1] Ajitha Shenoy, K.B., Biswas, S., Kurur, P.P., *Efficacy of the metropolis algorithm for the minimum-weight codeword problem using codeword and generator search spaces*, IEEE Trans Evolut Comput., **24**(4)(2020).
[2] Arora, S., Singh, S., A conceptual comparison of firefly algorithm, bat algorithm and cuckoo search, 2013 International Conference on Control, Computing, Communication and Materials (ICCCCM), Allahabad, (2013), 1–4.
[3] Augot, D., Charpin, P., Sendrier, N., *Studying the locator polynomial of minimum weight codewords of BCH codes*, IEEE Trans. Info. Theory, **38**(1992), 960–973.
[4] Bland, J.A., *Local search optimisation applied to the minimum distance problem*, Adv. Eng. Informat., **21**(2007), 391–397.
[5] Bouzkraoui, H., Azouaoui, A., Hadi, Y., New ant colony optimization for searching the minimum distance for linear codes, International Conference on Advanced Communication Technologies and Networking, (2018). doi: 10.1109/COMMNET.2018.8360246
[6] Gomez-Torrecillas, J., Lobillo, F.J., Navarro, G., *Minimum distance computation of linear codes via genetic algorithms with permutation encoding*, ACM Communications in Computer Algebra, **52**(3)(2018), 71–74.
[7] Cuellar, M.P., Gomez-Torrecillas, J., Lobillo, F.J., Navarro, G., *Genetic algorithms with permutation-based representation for computing the distance of linear codes*, arXiv:2002.12330.
[8] Hogben, L., Handbook of Linear Algebra. Boca Raton, FL, USA: Champman and Hall, 2007.
[9] MacWilliams, F.J., Sloane, N.J.A., The Theory of Error-Correcting Codes, North-Holland, Amsterdam, 1993.
[10] Ling, S., Xing, C., Coding Theory: A First Course, Cambridge University Press, 2004.
[11] Santucci, V., Baioletti, M., Milani, A., *Algebraic differential evolution algorithm for the permutation flowshop scheduling problem with total flowtime criterion*, in IEEE Transactions on Evolutionary Computation, **20**(5)(2016), 682–694.
[12] Shannon, C.E., *A mathematical theory of communication*, Bell System Technical Journal, **27**(1948), 379–423.
[13] Wolpert, D.H., Macready, W.G, *No free lunch theorems for optimization*. IEEE Trans Evolut Comput, **1**(1997), 67–82.
[14] Vardy, A., *The intractability of computing the minimum distance of a code*, IEEE Transactions on Information Theory, **43**(6)(1997), 1757–1766.
[15] Yang, X.S., A New Metaheuristic Bat-Inspired Algorithm, Nature inspired cooperative strategies for optimization, Studies in Computational Intelligence, 43**(284)**, Springer, Berlin, Heidelberg, 2010.