FWM Crosstalk Suppression using Golomb Ruler Sequence Optimization

Yogita Wadhwa[1], Baljit Kaur[2], Parvinder Bhalla[3]
[1, 3]Department of Electronics and Communication, MMU, Mullana, Haryana, India
[2]Department of Electronics and Communication, SSIET, Derabassi, Punjab, India

Abstract: The phenomenon where an undesirable nonlinear effect gives significantly degraded system performance, and becomes the major drawback for optical communication systems is known as Four wave mixing (FWM). To reduce FWM crosstalk in optical communication systems, the use of unequally spaced channels has been proposed. One of the unequal bandwidth channel allocation technique is designed by using the concept of Golomb Ruler that allows the gradual computation of a channel allocation set to result in an optimal point where degradation caused by FWM is minimal. In this paper we will optimize the Golomb Ruler Sequence by computing hard Computing (Exhaust Algorithm and Search Algorithm) and Soft Computing (Genetic and Biogeography based Algorithm) Algorithms. The result of algorithms compared and observed that soft computing algorithms perform better than the hard computing algorithms.

Keywords: *Four wave Mixing (FWM), Optimal Golomb Ruler (OGR), Soft Computing, Hard Computing.*

I. INTRODUCTION

Optical wavelength-division multiplexing (WDM) is a promising technique to utilize the ultra wide optical fiber bandwidth at low-loss region and transmit, in principle, more than 100 channels simultaneously [1], [2]. Among the fiber nonlinearities, which are a major problem in WDM systems, the FWM is the most serious one because it involves a lower optical input power than other nonlinearities [1]–[3]. Four-wave mixing (FWM) is defined as a nonlinear process in which three waves of frequencies fa, fb, and fc (c≠a,b) interact through the third-order electric susceptibility of the optical fiber [4] to generate a wave of frequency:

$$f_{abc} = f_a + f_b - f_c \qquad (1)$$

As a result of this, three co-propagating waves give rise, by FWM, to nine new optical waves [2]. This process will take place for every possible choice of three channel waves in a WDM system, therefore, suppose if the system has only ten channels, hundreds of new frequencies are generated by FWM. The conventional WDM systems have channels that are usually assigned with centre frequencies (or wave-lengths) which are equally spaced from each other. As a result of this the FWM problem cannot be solved only by increasing channel spacing, which can only decrease the chance and magnitude of the spectral sidebands of unwanted FWM signals trying to enter the pre-assigned channels. Although severe crosstalk can be resulted since there is still very high probability that FWM signals may fall into the WDM channels. In order to reduce the four-wave mixing effect in WDM systems, many unequally spaced channel allocation methods [5] are proposed. However, an increase in the bandwidth requirement is observed as compared to equally spaced channel allocation. Using the concept of Optimal Golomb ruler (OGR) a bandwidth allocation algorithm is presented here to reduce the FWM effect resulting in the improvement of the performance of the WDM system without increasing any additional cost in terms of bandwidth.

Golomb rulers represent a class of problems known as NP–complete [6]. Unlike the traveling salesman problem (TSP), which may be classified as a *complete ordered set*, the Golomb ruler may be classified as an *incomplete ordered set*. For higher order models, the exhaustive search [7], [8] of such NP–complete problems is impossible. As another mark is added to the ruler, the time required to search the permutations and to test the ruler becomes exponentially larger. Several different algorithms to tackle the Golomb ruler problem such as exact methods [7], [8], constraint programming [9], local searches [10] and exhaustive parallel search [11] have been studied. The success of soft computing algorithms such as Genetic Algorithms (GAs) [12]–[15] and Biogeography Based Optimization (BBO) [16]–[18] and Big Bang–Big Crunch (BB–BC) evolution theory [19], [20] in finding relatively good solutions to such NP–complete problems provides a good starting point for algorithms of finding OGR sequences. Hence, soft computing based algorithms seem to be very effective solutions for such problems. No doubt, these algorithms do not give the best/exact solutions but reasonably good solutions are available at given cost. This paper will introduce various hard computing and soft computing algorithms and their comparison. The remainder of this paper is organized as follows: Section II introduces the concept of Golomb rulers. Section III describes the various hard computing and soft computing algorithms. Section IV provides the comparison of these algorithms. Section V presents some concluding remarks.

II. GOLOMB RULER BASED ALLOCATION

The term ‘‘Golomb ruler’’ refers to a set of positive integer values, such that any two different pairs of numbers from the set have not the same difference. It is similar to a ruler constructed in a way that no two pairs of marks measure the same distance. An example of the Golomb ruler is shown in Fig. 1. An Optimal Golomb Ruler is the shortest ruler possible for a given number of marks [21]. Therefore applying OGR to the channel allocation problem, it is possible to achieve the smallest distinct number to be used for the channel allocation. Since the difference between any two numbers is distinct, the new FWM frequencies generated would not fall into the one already assigned for the carrier channels.

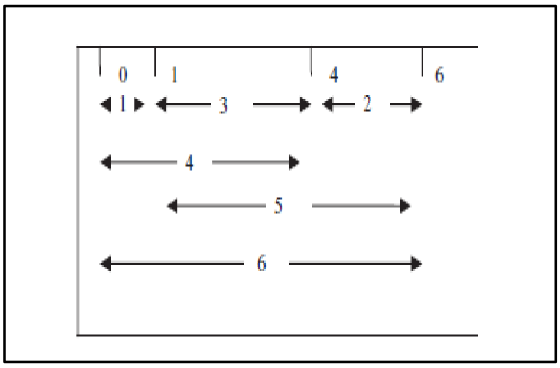

Fig. 1. A Golomb ruler.

An n-mark Golomb ruler is a set of n distinct non negative integers (a1, a2, .., an) called marks, such that the positive differences |ai-aj|, computed overall possible pairs of different i, j = 1,… n with i≠j are distinct. Let an be the largest integer in an n-mark Golomb ruler. Then an OGR with n marks (0,...., an) is an n-mark Golomb ruler if:

1. There exists no other n-mark Golomb ruler having smaller largest mark an, and
2. The ruler is written in a canonical form as the‘smaller’ of the equivalent rulers (0, a2, …, an) and(0,...an–a2, an), where smaller means the first differing entry is less than the corresponding entry in the other ruler.

The unequal-spaced channel allocation design begins with the division of the available optical bandwidth into equal frequency slots of width Δf [21]. Let f_0 be the center frequency of the first channel and fi = f_0+ni Δf be the center frequency of the i^{th} channel (or slot), where the integer n_i represents the slot number of the i^{th} channel and N is the total number of channels .In addition, mi = ni+1-ni is defined as the channel spacing (in integer) between the ith and (i+1)th channels for i ={1,2,3,.., N-1}. Therefore, the new frequencies $f_{ijk's}$ created by FWM in Eq.(1)[2]can equivalently be written in terms of slot number n_{ijk} so that

$n_{ijk} = n_i + n_j - n_k$ (2)

for i, j, k_[1, N] and k≠{i,j}. In other words, to ensure that no FWM signals can fall on the pre-assigned WDM channels, the channel-allocation problem can be treated as finding a set of distinct slot numbers so that n_{ijk}≠ {n1, n2, n3,…, nN}. To further formulate the allocation problem, we consider the physical system parameters. The slot width Δf should be large enough to accommodate the optical signal in a channel with minimum distortion, even with some instability in channel frequencies. On the other hand, to reduce unwanted spectral sidebands entering into a desired channel, the channel frequency-separation Δfc should also be large enough. For example, to have reasonable system performance, Forghieri [2] suggested the required minimum values of slot width (i.e., Δf ≥2R) and channel-to-channel separation (i.e., Δfc ≥10R) as an integer multiple of bit rate in order to avoid significant crosstalk created by FWM spectra and adjacent WDM channels, respectively. These two requirements impose a constraint that relates the minimum channel separation to the slot width (i.e., Δfc = nΔf) in terms of an integer multiple n.

Constraint1: Since $m_i = n_i+1-n_i$ (3) denotes the integer channel spacing between the ith and(i+1)th channels, the inequality mi≥n must be satisfied for all i= {1, 2,3, .., N-1}. Furthermore, to minimize the total optical bandwidth occupied by the WDM channels, an additional constraint on the total number of slots is needed while solving the channel-allocation problem.

Constraint2: The total number of slots

$S = \Sigma m_i = nN$ (4) must be as small as possible. A lower bound to the total optical bandwidth required Bun can be found just from the condition that the mi’s must be different from each other(and larger than n). It follows that [2]

$B_{un} \geq [1+((N/2)-1)/n]B_{eq}$ (5) where,

Beq =(N-l) Δfc is the total optical bandwidth of a conventional WDM system with the channels equally spaced. Fig. 2 shows the bandwidth expansion factor, defined as Bun/Beq, versus the number N of channels in the WDM system for various values of the minimum separation parameter n. It can be observed that for n≥5 and up to 10channels the lower bound is achievable. In general, for any value of N and n there are several optimum solutions.

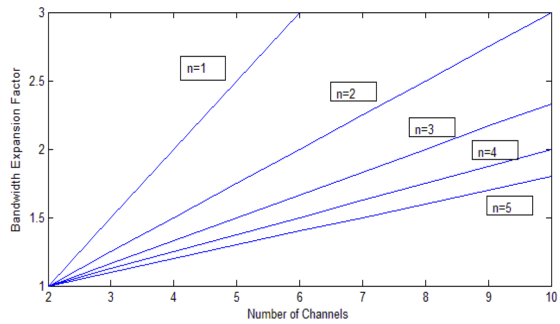

Fig.2. Bandwidth expansion factor (Bun/Beq) vs number of channels for various values of N = $\Delta F_c/\Delta F$.

III. ALGORITHMS TO GENERATE & OPTIMIZE THE GOLOMB RULER

A. *Hard Computing Algorithms*

The two hard computing algorithms to construct the Golomb ruler sequences are described here:

1) Exhaust Algorithm

The generation procedure using Exhaust algorithm requires two parameters. First is the number of marks contained in the desired Golomb ruler and the second parameter sets an upper bound on the length of the Golomb Ruler. This procedure is recursive in nature. Here an existing N-mark Golomb ruler is taken and a new mark is appended to the right side of the ruler resulting in N+1 mark ruler. This procedure does not keep track of the mark position but it keeps a track of the spaces between the adjacent marks stored in arrays called spaces. These values represent the first row of the difference triangle for the ruler. The algorithm begins by initializing the first elements in the spaces to the distance. Then it proceeds to the next distance in the spaces and starting at a value of 1, increments this value until the distances measured by the first two entries are unique. Then it repeats this process for the next element and so on. If at any point the total distance measured by these elements in the spaces exceeds the maximum ruler length then the algorithm will back up one element and increment that element and add new distances from there. When the procedure places its last mark and finds a ruler of the desired length it prints this information and continues the search. For the ruler verification procedure, a checker is used to check the series of marks fulfils the requirement of the Golomb ruler. The checker consists of two nested loops which compute every possible distance measures by the first N-elements of the space array. It uses the distance computed as index into an array of Boolean values. If the array element indexed by the distance is already set true, then the distance being checked has already been measured by the set of marks and sequence is not a Golomb ruler. The procedure stops at this point, returning the result as false. If the distance array element is clear then the procedure sets that element to be true and goes on to process the next pair of marks. If there are no conflicts after all the distances have been computed, the checker returns a value of true. A Golomb ruler can be constructed by using the equations as follows;

$$d1x = Mx+1-Mx \quad (5)$$
$$d2x = d1x+d1x+1 \quad (6)$$
$$d3x = d2x+d2x+1-d1x+1 \quad (7)$$

The equation for higher order differences is simply extensions of third order difference equation. The first order differences are the distances measured between every pair of adjacent marks in the ruler. The second order differences are the distances measured between marks placed too apart on the ruler. Ruler with m marks will have m-1 first order differences, m-2 second order difference and so on up to single m-1 order difference. The sequences generated by this algorithm does not yield the optimum Golomb sequence as the sequences result in containing large value of marks than necessary. So another algorithm, i.e search algorithm is used to get the optimum result.

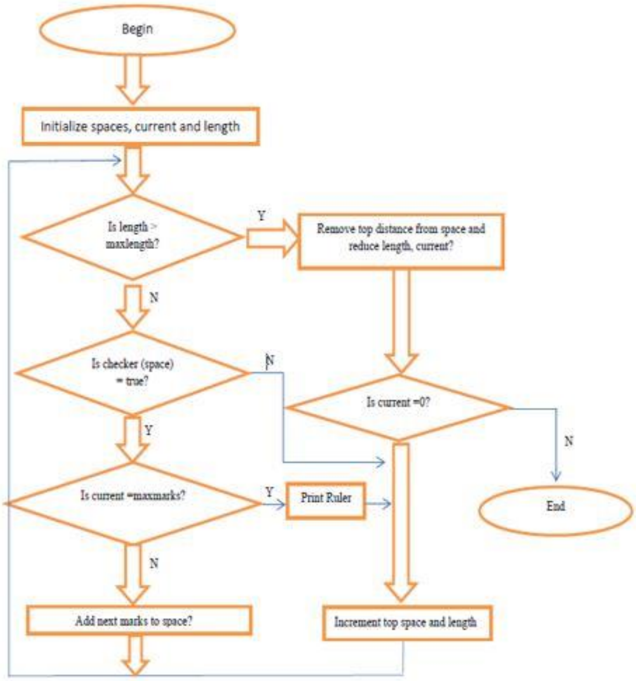

Fig.3: Flowchart for exhaust algorithm

2) Search Algorithm

The search algorithm is used to generate optimal sequences for a given prime number P and minimum pulse separation n:

N = P+1 and S = (n+(P-1)/2)P,

Where N is the number of terms in the sequence and S is the maximum length of the slot vector

1. If a prime number is denoted by P and minimum pulse separation n, the first delay vector(or channel spacing vector) m1 = [m0, m1,.., mP_1] = [n, n+1, y, n+P-1] is constructed.
2. The jth delay vector mj =[l0, l1 ,…, lk,.., lP-1] for j ={1, 2,3,.., P-1} are generated with lk = mj*k, where * denotes modulo-P-multiplication.
3. The jth code word sj =[s0, s1, s2,.., sq,…, sP] with weight P+1 are created from mj, according to the rule sq = lq-1+sq-1 where q = {1,2, 3,.., P} and s0 = 0.
4. Finally, find the code words sj’s with aperiodic correlation constraint one.

N	S	n	Example of slot vector	Nun	Neq	Bun/ Beq
4	15	4	[0,4,9,15]	28	12	2.33
6	45	7	[0,7,15,24,34,45]	125	35	3.6
8	91	10	[0,10,21,33,46,60,75,91]	336	70	4.8
12	231	16	[0,16,33,51,70,90,111,133,156,180,205,231]	1276	176	7.3
14	325	19	[0,19,39,60,82,105,129,154,180,207,235,264,294,325]	2093	247	8.5
18	561	25	[0,25,51,78,106,135,165,196,228,261,295,330,366,403,441,480,520,561]	4641	425	10.9
20	703	28	[0,28,57,87,118,150,183,217,252,288,325,363,402,442,483,525,568,612,657,703]	6457	532	12.1

TABLE I Simulation results obtained from the search algorithm

B. *Soft Computing Algorithms*

The two soft computing algorithms to construct the Golomb ruler sequences are described here:

1) *Genetic Algorithm*

The general steps involved in a GA based optimization can be found in [22]. In order to apply GA based approach in optimizing the WDM channel allocation problem, we use binary encoding and generate an initial population of chromosomes by randomly assigning the channels to the slots. Bit 1 corresponds to the presence of a channel in that slot and 0 corresponding to the channel's absence. Each chromosome has an associated cost function, assigning a relative merit to that chromosome. In our case, the cost function is decided by the average FWM power falling on the in-band channels in the system, assuming the channel allocation decided by the bit pattern in the corresponding chromosome. The next step is to rank chromosomes from best to worst. The ones having the low value of average FWM power are given priority since we aim at reducing the FWM power in the system falling on the channels. After this, the unacceptable chromosomes are discarded. We keep the top half and discard the bottom half. The next step, after ranking and discarding the chromosomes, is to pair the remaining chromosomes for mating. Once paired, new offspring are formed from pair-swapping genetic material. For each pair a random crossover point is selected. The bits to the right of the crossover point are swapped to form offspring. At this point, random mutation is introduced by altering a small percentage of bits in the whole population. We select a bit randomly out of whole population and invert it. Mutations facilitate the algorithm with the freedom of searching outside the current region of parameter space. After the mutations take place the cost associated with the offspring and mutated chromosomes is calculated, and the process is repeated until the stopping criterion is achieved. The stopping criterion is fixed as the maximum required average FWM power level in the system.

The steps specific to the solution of channel allocation problem under consideration are summarized as follows.

1. Generate randomly a population of unique individuals/chromosomes.
2. Apply the cost function to each chromosome; this is defined as the average FWM power falling on the channels in the system.
3. The chromosomes are ranked. A chromosome with high fitness corresponds to low level of average FWM power.
4. The bottom half of chromosome are discarded.
5. Crossover takes place to produce a new generation.
6. Mutation takes place and changes one bit out of whole population.
7. Step 2-6 are carried out till the desired solution is obtained.

After applying the algorithm described above a bit pattern corresponding to the optimized channel allocation is obtained. Figure 4 shows the general Genetic Algorithm flowchart.

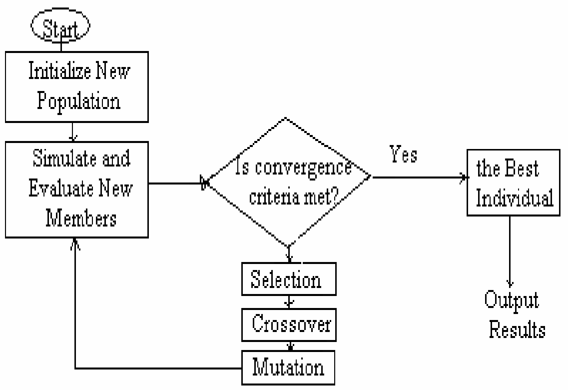

Fig.4: Genetic Algorithm flowchart

2) *Biogeography based Optimization*

Biogeography Based Optimization is a population–based evolutionary algorithm (EA) developed for global optimization. It is based on the mathematics of biogeography. It is a new kind of optimization algorithm which is inspired by the science of Biogeography. It mimics the migration strategy of animals to solve the problem of optimization [23] – [28]. Biogeography is the study of the geographical distribution of biological organisms. Biogeography theory proposes that the number of species found on habitat is mainly determined by immigration and emigration. Immigration is the arrival of new species into a habitat, while emigration is the act of leaving one‘s native region. The science of biogeography can be traced to the work of nineteenth century naturalists such as Alfred Wallace [28] and Charles Darwin [29].

In BBO, problem solutions are represented as islands and the sharing of features between solutions is represented as emigration and immigration. An island is any habitat that is geographically isolated from other habitats [30].

The idea of BBO was first presented by Dan Simon in December 2008 and is an example of how a natural process can be modeled to solve general optimization problems [31]. This is similar to what has occurred in the past few decades with Genetic Algorithms (GAs), Artificial Neural Networks (ANNs), Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO), and other areas of computer intelligence. Biogeography is nature‘s way of distributing species, and is analogous to general problem solving.

Suppose that there are some problems and that a certain number of candidate solutions are there. A good solution is analogous to an island with a high HSI (Habitat suitability index), and a poor solution is like an island with a low HSI. Features that correlate with HSI include factors such as distance to the nearest neighboring habitat, climate, rainfall, plant and animal diversity, diversity of topographic features, land area, human activity, and temperature [32]. The variables that characterize habitability are called suitability index variables (SIVs). High HSI solutions are more likely to share their features with other solutions, and HSI solutions are more likely to accept shared features from other solutions [33] – [35]. As with every other evolutionary algorithm, each solution might also have some probability of mutation, although mutation is not an essential feature of BBO the improvement of solutions is obtained by perturbing the solution after the migration operation [35].

BBO Algorithm to Generate Optimal Golomb Ruler Sequences

The basic structure of BBO algorithm to generate OGR sequences is as follows:

1. Initialize the BBO parameters: maximum species count i.e. population size S_{max}, the maximum migration rates E and I, the maximum mutation rate m_{max}, an elitism parameter and the number of iterations.

2. Initialize the number of channels (or marks) ‗N' and the upper bound on the length of the ruler.

3. Initialize a random set of habitats (integer population), each habitat corresponding to a potential solution to the given problem. The number of integers in each habitat being equal to the number of channels or mark input by the user.

4. Check the golombness of each habitat. If it satisfies the conditions for Golomb Ruler sequence, retain that habitat; if it does not, delete that particular habitat from the population generated from the step 3.

5. For each habitat, map the HSI (Total Bandwidth) to the number of species S, the immigration rate λ, and the emigration rate μ.

6. Probabilistically use immigration and emigration to modify each non–elite habitat, then recompute each HSI.

7. For each habitat, update the probability of its species count given by equation (8). Then, mutate each non–elite habitat based on its probability, check golombness of each habitat again and then recompute each HSI.

$$\dot{P}_s = \begin{cases} -(\lambda_s + \mu_s)P_s + \mu_{s+1}P_{s+1}, & S = 0 \\ -(\lambda_s + \mu_s)P_s + \lambda_{s-1}P_{s-1} + \mu_{s+1}P_{s+1}, & 1 \le S \le S_{\max} - 1 \\ -(\lambda_s + \mu_s)P_s + \lambda_{s-1}P_{s-1}, & S = S_{\max} \end{cases}$$

where λs and μs are the immigration and emigration rates, when there are S species in the habitat.

8. Is acceptable solution found? If yes then go to Step 10.

9. Number of iterations over? If no then go to Step 3 for the next iteration.

10. Stop

IV. PERFORMANCE COMPARISON OF PROPOSED ALGORITHMS

In this subsection, comparison of the results obtained by BBO and GA with EA and SA in terms of Ruler Length and Bandwidth is described. Table 2 illustrates the total bandwidth (BW) and length of ruler (RL) occupied by different sequences obtained by proposed algorithms for various channels '*n*'. It has been noted that the application of EQC and SA is limited to prime powers, so the total bandwidth and ruler length for EQC and SA are shown by a dash line in Table 2. Comparing the simulation results of BBO and GA with EA and SA; it is observed that there is a significant improvement with respect to the length of the ruler and thus the total bandwidth occupied by the use of soft computing methods that is, the results gets better. As shown in Table 2, for smaller mark values upto 7 the ruler length and thus bandwidth occupied by BBO and GA is same. But for higher mark values, the ruler length and hence total bandwidth obtained by BBO algorithm slightly approaches to their optimal values as compared with GA. This is also graphically shown in Figure 5 and Figure 6 respectively. Therefore, for most of the mark values, the computational cost (i.e. ruler length and total bandwidth) of BBO and GA will be same since it will be dominated by fitness function evaluation.

N	KNOWN OGR [24], [33], [47], [48] (Best Solutions)		EQC [1], [13], [21]		SA [1], [13], [21]		BBO	
	RULER LENGTH	TOTAL BANDWIDTH	RULER LENGTH	RULER LENGTH	RULER LENGTH	TOTAL BANDWIDTH	RULER LENGTH	TOTAL BANDWIDTH
3	3	4	6	10	6	4	3	4
4	6	11	15	28	15	11	6	11
5	11	25 28	—	—	—	—	12	23
6	17	44 47 50 52	45	140	20	60	17 18 20 21	42 43 44 45 49
7	25	81 87 95 77 90	—	—	—	—	27 29 31 32 33	73 82 83 84 91 95
8	34	117	91	378	49	189	34 39 40 42	121 125 127 131
9	44	206	—	—	—	—	49 56 61 62 63 64	196 200 201 206 215 225
10	55	249	—	—	—	—	74	274
11	72	386 391	—	—	—	—	86 104 114 118	435 440 491
12	85	503	231	1441	132	682	124	556
13	106	660	—	—	—	—	203 241	1015 1048
14	127	924	325	2340	195	1183	127 206 230	924 991 1177
15	151	1047	—	—	—	—	267 298	1322 1634
16	177	1298	—	—	—	—	283	1804
17	199	1661	—	—	—	—	354 369	2201 2208
18	216	1894	561	5203	493	5100	445	2566
19	246	2225	—	—	—	—	597	5067
20	283	2794	703	7163	703	6460	752	5137

Table 2: Comparison of Total Bandwidth and Ruler Length Obtained by Soft Computing Algorithm (BBO) with Known OGR, EA and SA, Where N Is The Number Of Unequal–Spaced WDM Channels

V. CONCLUSION

Generation of Optimal Golomb Ruler sequences using Hard Computing Algorithm is a high computational complexity problem. The purpose of soft computing is not necessarily to produce best results, but to produce the optimal results under the constraints of time and cost. This paper presents the application of Soft Computing Algorithm to solve Optimal Golomb Ruler problem. It has been observed that Soft Computing Algorithm produces Golomb Ruler sequences very efficiently. The performance is compared with the other existing classical approaches i.e. Extended Quadratic Congruence (EQC) and Search Algorithm (SA) in terms of the length of ruler and total bandwidth obtained by the sequences. The preliminary results indicate that Soft Computing Algorithm appear to be most efficient approach to such NP–complete problems.

VI. REFERENCES

[1] Aggarwal, G. P., Nonlinear Fiber Optics, Academic Press, 1989.

[2] Hill, K. O., Johnson, D. C., Kawasaki, B. S. and MacDonald, R. I., "CW Three-Wave Mixing in Single Mode Optical Fibers," Journal of Applied Physics, Vol: 49, pp. 5098-5106, 1978.

[3] N. Shibata, K. Nosu, K. Iwashita, and Y. Azuma, "Transmission limitations due to fiber nonlinearities in optical FDM systems," J. Lightwave Technol., vol. 8, pp. 1068–1077, Aug. 1990.

[4] F. Forghieri, R.W. Tkach, A.R. Chraplyvy, WDM systems with unequally spaced channels, J. Lightwave Technol. 13 (5) (1995) 889– 897

[5] Rajneesh Randhawa, J.S. Sohal, R.S. Kaler, Optimum algorithm for WDM channel allocation for reducing four-wave mixing effects, Optik 120 (March) (2008) 898–904.

[6] http://theinf1.informatik.uni-jena.de/teaching/ss10/oberseminar-ss10

[7] Robinson, Johan P. 1990. Optimum Golomb Rulers. IEEE Transactions on Computers, Vol. C-28, No. 12, (December 1979).

[8] James B. Shearer. Some New Optimum Golomb Rulers. IEEE Transactions on Information Theory. IT-36, pp. 183–184.

[9] Galinier, Jaumard, Morales, and Pesant G. 2001. A constraint–Based Approach to the Golomb Ruler Problem. In Proceeding of 3rd International workshop on integration of AI and OR techniques (CP-AI-OR 2001).

[10] Leitao, Tiago 2004. Evolving the Maximum Segment Length of a Golomb Ruler. Genetic and Evolutionary Computation Conference, USA.

[11] Rankin, William T. 1993. Optimal Golomb Rulers: An exhaustive parallel search implementation. M.S. thesis, Duke University, (Available at http://people.ee.duke.edu/~wrankin/golomb/golomb.html).

[12] Shobhika 2005. Generation of Golomb Ruler Sequences and Optimization Using Genetic Algorithm. M. Tech. Thesis, Department of Electronics and Communication Engineering, Thapar Institute of Engineering and Technology, Deemed University, Patiala.

[13] Soliday, Stephen W., Homaifar, A. and Lebby, Gary L. 1995. Genetic Algorithm Approach to the Search for Golomb Rulers. In Proceedings of the Sixth International Conference on Genetic Algorithms (ICGA-95), Morgan Kaufmann, pp. 528–535.

[14] Robinson, John P. 2000. Genetic Search for Golomb Arrays. IEEE Transactions on Information Theory, Vol. 46, No. 3, pp. 1170–1173.

[15] Ayari, N., Th an uong and . Jemai. 2010. A Hybrid Genetic Algorithm for Golomb Ruler Problem. In Proceeding of ACS/IEEE International Conference on Computer Systems and Applications (AICCSA 2010), pp.1–4.

[16] Bansal, S., Kumar, S., Sharma, H. and Bhalla, P. 2011. Generation of Golomb Ruler Sequences and Optimization Using Biogeography Based Optimization. In Proceedings of 5th International Multi Conference on Intelligent Systems, Sustainable, New and Renewable Energy Technology and Nanotechnology (IISN–2011), Institute of Science and Technology Klawad, Haryana, pp 282–288.

[17] Bansal, S., Kumar, S., Sharma, H. and Bhalla, P. 2011. Golomb Ruler Sequences Optimization: A BBO Approach. International Journal of Computer Science and Information Security (IJCSIS), Pittsburgh, PA, USA, Vol. 9, No. 5, pp. 63–71.

[18] Bansal, S. 2011. Golomb Ruler Sequences Optimization: Soft Computing Approaches. M. Tech. Thesis, Department of Electronics and Communication Engineering, Maharishi Markandeshwar Engineering College, Deemed University, Mullana.

[19] Kumar S., Bansal S. and Bhalla P. 2012. Optimal Golomb Ruler Sequence Generation for FWM Crosstalk Elimination: A BB–BC Approach. In Proceedings of 6th International Multi Conference on Intelligent Systems, Sustainable, New and Renewable Energy Technology and Nanotechnology (IISN–2012), Institute of Science and Technology Klawad–133105, Haryana, India, pp. 255–262.

[20] Bansal S., Kumar S. and Bhalla P. 2013. A Novel Approach to WDM Channel Allocation: Big Bang–Big Crunch Optimization. In the proceeding of Zonal Seminar on Emerging Trends in Embedded System Technologies (ETECH-2013) organized by The Institution of Electronics and Telecommunication Engineers (IETE), Chandigarh Centre, Chandigarh, pp. 80–81.

[21] F.Forghieri,R.W.Tkach, A.R.Chparlyvy,D.Marcuse, Reduction of four-wavemixingcrosstalkinWDMsystems using unequally spaced channels, IEEE Photon.Technol. Lett. 6(6)(1994)754–756.

[22] J. M. Johnson, Y. Rahmat-samii, "Genetic algorithms in engineering electromagnetics," IEEE Antennas and Propagation Magazine, vol. 39, No. 4, pp. 7-15, August,1997.

[23] Hongwei Mo, Lifang Xu, ―Biogeography based Optimization for Travelling Salesman Problem‖, Sixth International Conference on Natural Computation (ICNC 2010), vol. 6, pp.3143–3147, August 2010.

[24] D. Simon, M. Ergezer, D. Du, and R. Rarick, ―Markov models for biogeography–based optimization‖, IEEE Transactions on Systems, Man, and Cybernetics – Part B: Cybernetics, vol. 41, no. 1, pp. 299–306, January 2011.

[25] Shakti Kumar, Parvinder Bhalla and Amarpartap Singh, ―Fuzzy Rule Base Generation Part II: A BBO Approach‖, Proceedings of the Workshop on Intelligent System Engineering (WISE–2010), Institute of Science & Technology, Klawad, July 2010.

[26] S. Kumar, P. Narula, and T. Ahmed, ―Knowledge extraction from numerical data for the Mamdani type fuzzy systems: A BBO approach‖, Innovative Practices in Management and Information Technology for Excellence, Jagadhri, India, May 2010.

[27] H. Ma, S. Ni, and M. Sun, ―Equilibrium species counts and migration model tradeoffs for biogeography–based optimization‖, IEEE Conference on Decision and Control, Shanghai, pp. 3306–3310, December 2009.

[28] H. Ma, ―An analysis of the equilibrium of migration models for biogeography–based optimization‖, Information Sciences, vol. 180, no. 18, pp. 3444–3464, 15 September 2010.

[29] A. Wallace, ―The Geographical Distribution of Animals (Two Volumes)‖, Boston, MA: Adamant Media Corporation, 2005.

[30] C. Darwin, ―The Origin of Species‖, New York: Gramercy, 1995.

[31] Dan Simon, Mehmet Ergezer, and Dawei Du, ―Population Distributions in Biogeography–Based Optimization Algorithms with Elitism‖, Proceedings of the 2009 IEEE International Conference on Systems, Man, and Cybernetics San Antonio, TX, USA, pp. 991–996, October 2009.
[32] D. Simon, ―Biogeography–Based Optimization,‖ IEEE Trans. on Evolutionary Computation, vol. 12, no. 6, pp. 702–713, December 2008.
[33] Aniruddha Bhattacharya and Pranab Kumar Chattopadhyay, ―Hybrid Differential Evolution With Biogeography– Based Optimization for Solution of Economic Load Dispatch‖, IEEE Transactions on Power Systems, vol. 25, no. 4, pp. 1955–1964, November 2010.
[34] S. Kumar, P. Bhalla, and A. Singh, Fuzzy rule base generation from numerical data using biogeography–based optimization, Institution of Engineers Journal of Electronics and Telecomm Engineering, vol. 90, pp. 8–13, July 2009.
[35] H. Ma and D. Simon, ―Blended biogeography-based optimization for constrained optimization‖, Engineering Applications of Artificial Intelligence, vol. 24, no. 3, pp. 517-525, April 2011.