A Cuckoo Search based WDM Channel Allocation Algorithm

Shonak Bansal
Department of Electronics and Communication Engineering, Institute of Science and Technology,Klawad.Haryana, India

Ruchi Chauhan
Department of Electronics and Communication Engineering, Institute of Science and Technology,Klawad.Haryana, India

Parveen Kumar
Department of Electronics and Communication Engineering, Institute of Science and Technology,Klawad.Haryana, India

ABSTRACT
More and more modern metaheuristics nature–inspired algorithms are emerging and they become increasingly popular. This paper formulates an algorithm for solving the channel–allocation problem in optical wavelength division multiplexing (WDM) systems to suppress four–wave mixing crosstalk (FWM) based on a novel nature–inspired algorithm, called Cuckoo Search algorithm by using the concept of Optimal Golomb ruler (OGR) sequences. Simulation results conclude the significance performance improvement, without the requirement of increased total optical channel bandwidth, unlike two existing classical channel–allocation algorithms i.e. Extended Quadratic Congruence (EQC) and Search Algorithm (SA) and one of the existing nature–inspired algorithm i.e. Genetic Algorithm (GA).

General Terms
Classical computing, Four–wave mixing, Metaheuristics, Optimization.

Keywords
Cuckoo Search algorithm, Channel allocation, Golomb ruler, Wavelength division multiplexing.

1. INTRODUCTION
The crosstalk due to FWM is one of the most serious limiting sources of performance degradation in optical WDM systems because it involves a lower optical input power than other nonlinearities. The performance can be improved if FWM crosstalk generation at the channel frequencies is avoided [1]–[7].

To suppress the crosstalk due to FWM signals in optical WDM systems, several unequally spaced channel allocation (USCA) techniques have been studied in [1], [8]–[14]. However, these techniques have the drawback of increased of optical channel bandwidth requirement compared to equally spaced channel allocation (ESCA). This paper proposes an unequally spaced bandwidth channel allocation algorithm by taking into consideration the concept of optimal Golomb rulers (OGRs). This USCA technique suppress the FWM crosstalk signals in the optical WDM systems without inducing additional cost in terms of channel bandwidth [7], [15]–[17].

Golomb rulers are a class of NP–complete problems [18]. The exhaustive search [19]–[23] for higher order marks of such NP–complete problems is impossible. The success of nature–inspired algorithms such as Genetic Algorithm [24]–[28], Tabu Search (TS) [27], Biogeography Based Optimization (BBO) [28]–[31], Big Bang–Big Crunch (BB–BC) evolution theory [32], [33], and Firefly Algorithm (FA) [34], in finding relatively good solutions to such NP–complete problems provides a good starting point for methods of finding OGR sequences. Hence, nature–inspired algorithms seem to be very effective solutions for the NP–complete problems. In this paper, a new nature–inspired metaheuristic algorithm based on the obligate brood parasitic behavior of cuckoo species namely Cuckoo Search algorithm for finding the OGR sequences is being presented.

The rest of this paper is organized as follows: a brief concept of Golomb rulers is presents in section II, the proposed CS based channel allocation algorithm is presented in section III. Section IV presents the simulation results comparing with classical computing algorithms i.e. Extended Quadratic Congruence and Search Algorithm and of the existing nature–inspired algorithm i.e. Genetic Algorithm of generating unequal channel spacing. Section V presents some concluding remarks.

2. GOLOMB RULER
*Golomb ruler*s are a sequence of non–negative integers such that no distinct pairs of numbers called *marks* from the sequence have the unique difference [35]–[39]. Figure 1 shows an example of 4–marks Golomb ruler with the distance associated between each pair of marks [24]. The ruler length of Golomb ruler shown in Figure 1 is 6.

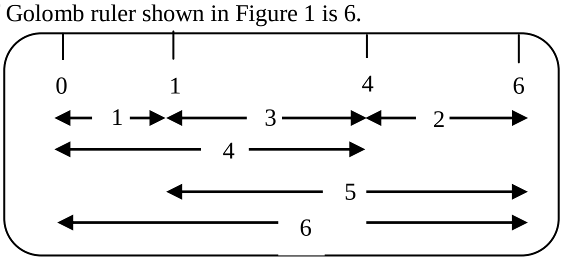

Fig 1: A 4–Marks Golomb Ruler having Ruler Length 6

A perfect Golomb ruler measures all the integer distances from 0 to L, where L is the length of the ruler [18], [24], [25]. An optimal Golomb ruler is the shortest length ruler for a given number of marks. There can be multiple different OGRs for a specific number of marks. However, the unique optimal Golomb 4–mark ruler is shown in Figure 1, which measures all the integer distances from 1 to 6 (and is therefore also a perfect ruler) [24], [40]–[44]. Therefore applying OGRs to the channel allocation problem, it is possible to achieve the smallest distinct number to be used for the channel allocation. As the difference between any two numbers is different, the new FWM crosstalk signals generated would not fall into the one already assigned for the carrier channels.

3. OGR SEQUENCES GENERATION USING CUCKOO SEARCH ALGORITHM

If *CS* is the spacing between any pair of channels, *IE* is the individual elements in Golomb ruler and *n* is the total number of channels, then the objective is to minimize the ruler length (*RL*) and total bandwidth (*TBW*), which are given by the equations (1) and (2) [30] respectively:

$$RL = \sum_{i=1}^{n-1} (CS)_i \quad (1)$$

subject to $(CS)_i \neq (CS)_j$

$$TBW = \sum_{i=1}^{n} (IE)_i \quad (2)$$

subject to $(IE)_i \neq (IE)_j$

where $i, j = 1, 2, \ldots, n$ with $i \neq j$ are distinct in both equations.

To generate OGRs, a novel optimization nature–inspired metaheuristic algorithm called Cuckoo Search is being proposed in this paper.

3.1 Cuckoo Search Algorithm

Cuckoo Search algorithm (CSA) is a novel metaheuristic population–based nature–inspired optimization algorithm which was developed by X.-S. Yang and S. Deb in 2009 [45], [46]. The CSA is inspired by brood parasitism of some cuckoo species in because of their special lifestyle and aggressive reproduction strategy. In addition, this algorithm is enhanced by the Lévy flights behavior of some birds, rather than by simple isotropic random walks.For simplicity in describing CSA, the following three idealized rules are used [45]:

1. each Cuckoo lays one egg at a time, and dumps it in a randomly chosen nest;
2. the best nest with high quality of eggs (solution) are carried over to the next generations;
3. the number of available host nests is fixed, and a host can discover an alien egg with probability $p_a \in$ [0, 1]. In this case, the host bird can either throw the egg away or simply abandon the nest so as to built a completely new nest in a new location

For simplicity, the last assumption can be approximated by a fraction p_a of the *n* host nests are replaced by new nests (with new random solutions). For a maximization problem, the quality or fitness of a solution can simply be proportional to the value of the objective function. When new solutions $x^{(t+1)}$ are generating for, say, a cuckoo *i*, a Lévy flight is performed as given by equation (3) [45]:

$$x_i^{(t+1)} = x_i^{(t)} + \alpha \oplus \text{Lévy}(\lambda) \quad (3)$$

where $\alpha > 0$ is the step size, which should be related to the scale of the problem of interest. The product $\oplus$ means entry wise multiplications. The Lévy flights essentially provide a random walk while their random steps are drawn from a Lévy distribution for large steps [45]

```
Cuckoo Search Algorithm
 Begin
   /* CSA parameter initialization */
      Define objective function f (x);   x = (x1,…, xd)^T;
      Generate initial population of n host nests xi (i = 1, 2,…,n);
   /* End of CSA parameter initialization */
   While not T                    /* T is a termination criterion */
      Get a cuckoo (say i) randomly by Lévy flights;
      Evaluate its quality/fitness Fi;
      Choose a nest among n (say j) randomly;
      If (Fi > Fj),
         Replace j by the new solution;
      End if
      A fraction (pa) of worse nests are abandoned and
           new ones are built;
        Keep the best solutions (or nests with quality solutions);
      Rank the solutions and find the current best;
   End while
   Postprocess results and visualization;
 End
```

Fig 2: Pseudo code of the CSA

$$\text{Lévy} \sim u = t^{-\lambda}, \ (1 < \lambda \leq 3) \quad (4)$$

which has an infinite variance. Here, the consecutive jumps/steps of a cuckoo essentially form a random walk process which obeys a power-law step-length distribution with a heavy tail.

The basic steps of the Cuckoo Search algorithm can be summarized as the pseudo code shown in Figure 2 [45].

3.2 Cuckoo Search Algorithm to Generate OGRs

The proposed pseudo–code for CSA to generate OGR sequences in this paper is shown in Figure 3.

4. SIMULATION RESULTS AND DISCUSSION

To generate OGR sequences the Cuckoo Search algorithm has been written and tested in Matlab–7 language [47] under Windows 7 operating system. The algorithm has been executed on Laptop with Intel core2 Duo 2.20 GHz processor with a RAM of 3 GB. To show the effectiveness of the proposed algorithm, its performance is being compared with known OGR [15], [20], [35], [37], [39], [42], [43], [48], [49], two of the existing classical computing algorithms i.e. EQC and SA [1], [13], [24] and one of the nature–inspired algorithm i.e. GA [24]–[28] of generating unequal channel–allocation.

4.1 Simulation Parameters for Cuckoo Search Algorithm

To get optimum results after a number of careful experimentation, following optimum parameter values of Cuckoo Search algorithm have finally been settled as shown in Table 1. With these parameters settings, the large numbers of sets of trials for various marks were conducted. A set of 10 trials for n = 4 to 6 are given in Table 2. The performance of all the sets is nearly the same as given in Table 2.

Table 1. Simulation Parameters for CSA

Parameter	*Value*
Number of nests (*Popsize*)	30
á	0.01
p_a	0.05

```
CSA to Generate OGRs
  Begin
     /* CSA parameter initialization */
         Define operating parameters for CSA;
        Initialize the number of channels/marks, lower and upper bound on the ruler length;
        While not Popsize                    /* Popsize is the population size input by the user */
              Generate a random set of integer population of host nests;
                                   /* Number of integers in host nests is being equal to the number of channels */
              Check Golombness of each nests;
              If Golombness is satisfied
                 Retain that nest;
                Else
                 Remove that particular nest from the generated population;
               End if
          End while
         Compute the total bandwidth (fitness value) ;
         Rank the population from best to worst based on fitness value;
     /* End of CSA parameter initialization */
      While not T    /* T is a termination criterion */
         A1:    Get a cuckoo (say i) randomly by Lévy flights;;
                Recheck Golombness of updated solution;
                If Golombness is satisfied
                    Retain that solution and then go to B1;
                  Else
                    Retain the previous generated solution and then go to A1;
                          /* Previous generated solution is being equal to the solution generated into the parameter initialization step */
               End if
        B1:    Evaluate its quality/fitness F_i;
              Choose a nest among n (say j) randomly;
              If (F_i > F_j),
                  Replace j by the new solution;
              End if
        A2:   Abandon a fraction (p_a) of worse nests and build new ones;
               Recheck Golombness of updated solution;
               If Golombness is satisfied
                    Retain that solution and then go to B2;
                  Else
                    Retain the previous generated solution and then go to A2;
               End if
        B2:   Keep the best solutions (or nests with quality solutions);
              Rank the solutions from best to worst based on fitness value and find the current best;
      End while
      Display the generated optimal Golomb ruler sequences;
  End
```

Fig 3: Pseudo code of the CSA to Generate OGRs

Table 2. Performance of Cuckoo Search Algorithm for Different Marks in a Set of 10 Trials

Trials	***n = 4***		***n = 5***		***n = 6***	
	Ruler Length	***Total Bandwidth (Hz)***	***Ruler Length***	***Total Bandwidth (Hz)***	***Ruler Length***	***Total Bandwidth (Hz)***
1	6	11	12	23	17	44
2	6	11	11	24	18	42
3	6	11	11	24	18	42
4	6	11	12	23	17	44
5	6	11	11	24	18	42
6	6	11	12	23	17	44
7	6	11	12	23	17	44
8	6	11	12	23	17	44
9	7	11	12	23	18	42
10	7	11	12	23	18	42
	Optimal Ruler Length = 6 Optimal Total Bandwidth = 11 Hz		**Optimal Ruler Length = 11 Optimal Total Bandwidth = 23 Hz**		**Optimal Ruler Length = 17 Optimal Total Bandwidth = 42 Hz**	

4.2 Effect of Increasing Generations on Total Channel Bandwidth

With the increase of number of generations, the total channel bandwidth occupied by the sequence tends to decrease; it means that after a certain number of generations, the rulers reach their optimal solution values. This can be seen in tabular form for proposed Cuckoo search algorithm in Table 3 for n = 3 to 11. It is noted that the generations has little effect for low order marks such as n = 3, 4 and 5. But for higher order marks, the generations have a great effect on the total channel bandwidth. It means the optical channel bandwidth gets optimized after a certain numbers of generations. By carefully observation, the paper fixed the generations of 300 for the proposed Cuckoo Search algorithm.

Table 3. Effect of Increasing Generations on the Performance of CSA for Various Marks

Generations	*Total Bandwidth (Hz)*						
	CSA						
	***n*=5**	***n*=6**	***n*=7**	***n*=8**	***n*=9**	***n*=10**	***n*=11**
2	24	103	93	212	342	619	689
5	23	59	93	212	342	479	689
25	23	42	81	174	342	479	682
75	23	42	73	117	268	336	613
150	23	42	73	113	206	249	502
200	23	42	73	113	206	249	459
250	23	42	73	113	206	249	399
275	23	42	73	113	206	249	399
280	23	42	73	113	206	249	391
300	23	42	73	113	206	249	391

4.3 Comparison of Proposed Optimization Algorithm with Previous Existing Algorithms in terms of Ruler Length and Total Channel Bandwidth

The aim to use nature–inspired Cuckoo Search algorithm in this paper was to optimize the ruler length (*RL*) so as to conserve the total bandwidth (*TBW*) occupied by the channels. Table 4 enlist the ruler length and total bandwidth occupied by different sequences obtained by the proposed algorithm for various channels *n* and its comparison with known OGRs [15], [20], [35], [37], [39], [42], [43], [48], [49], EQC, SA [1], [13], [24] and GA [24]–[28]. All results have been obtained after a set of 10 trials. According to [1] the application of EQC and SA was limited to prime numbers, so the *RL* and *TBW* obtained for EQC and SA are shown by a dash line in Table 4.

Comparing the performance of proposed Cuckoo Search algorithm with best known OGR, EQC, SA and GA; it is observed that there is a significant improvement with respect to the ruler length and thus the total channel bandwidth occupied. Figures 4 and 5 the graphical performance comparison of proposed algorithm to generate OGR sequences with known OGRs, EQC, SA and GA in terms of the ruler length and total optical channel bandwidth occupied by the sequences for n = 3, 4, 6 and 8 respectively. The results obtained for rulers upto 7–marks can be performed in few seconds whereas minutes were necessary for other instances. From Table 4 it is clear that the proposed algorithm was able to generate the optimal ruler in n = 3 to 11 but at the expense of significant computational time.

It is relevant to mention here that OGRs obtained by non–heuristic exhaustive searches [25] for 10–marks took 12.57 minutes, whereas for the our proposed Cuckoo Search algorithm the execution time varied from few seconds for rulers of 7–marks to 20 minutes for 11–marks ruler with a maximum generations of 300. Of course computers today play central role in reducing the computation time; efficient algorithms can further reduce the time complexity drastically by substituting best for reasonably better solutions.

Table 4. Performance Comparison of Proposed CSA with Known OGR, EQC, SA and GA in terms of Ruler Length and Total Channel Bandwidth

n	**Known OGRs [15], [20], [35], [37], [39], [42], [43], [48], [49]**		**EQC [1], [13], [24]**		**SA [1], [13], [24]**		**GA [24]–[28]**		**CSA**	
	RL	***TBW (Hz)***	***RL***	***TBW (Hz)***	***RL***	***TBW (Hz)***	***RL***	***TBW (Hz)***	***RL***	***TBW (Hz)***
3	3	4	6	10	6	4	3	4	3	4
4	6	11	15	28	15	11	6 7	11	6 7	11
5	11	25 28	—	—	—	—	11 12 13	23 25	11 12	23 24
6	17	44 47 50 52	45	140	20	60	17 18	42 44	17 18	42 44 47 50
7	25	81 87 95 77 90	—	—	—	—	27 29 30	73 79 80 84 97	25 26 27	73 77 80 81
8	34	117	91	378	49	189	35 41 42	126 128 133	34 39	113 117
9	44	206	—	—	—	—	52 56 59 61	192 193 196 203	49	206
10	55	249	—	—	—	—	75	283	55	249

11	72	386 391	—	—	—	—	76 94 96	287 301 395 456	72	391

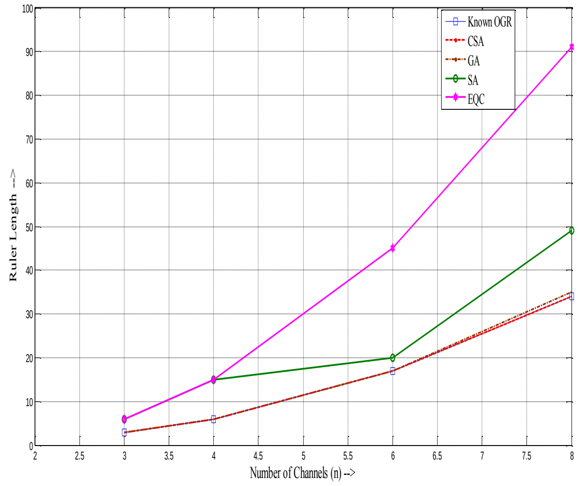

Figure 4: Performance Comparison of the Results Obtained by Proposed Algorithm with Known OGR, EQC, SA and GA in Terms of Ruler Length

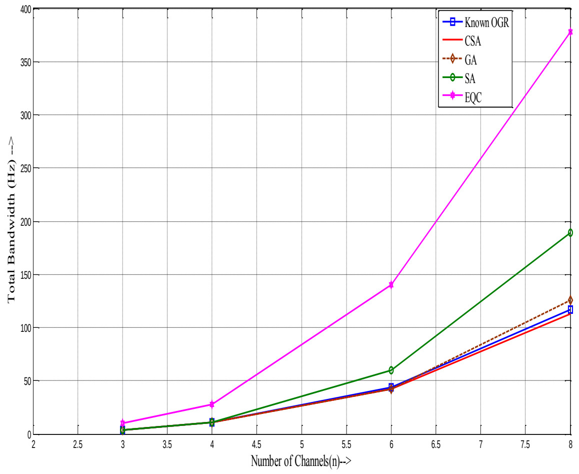

Figure 5: Performance Comparison of the Results Obtained by Proposed Algorithm with Known OGR, EQC, SA and GA in Terms of Total Bandwidth

5. CONCLUSION

In this paper, we have formulated the application of a novel metaheuristic nature–inspired Cuckoo Search algorithm to solve the highly complex problem of optimal Golomb ruler sequences. It has been observed that the proposed metaheuristic algorithm produces OGR sequences very efficiently and effectively. The performance is being compared with the known OGR, two existing classical algorithms of unequal channel spacing i.e. EQC and SA and one of the nature–inspired algorithms i.e. GA in terms of the ruler length and total bandwidth obtained by the sequences. From the simulation results it is concluded that the proposed algorithm out performs the existing algorithms in terms of ruler length and total occupied optical channel bandwidth.

6. REFERENCES

[1] Kwong, W. C., and Yang, G. C. 1997. An Algebraic Approach to the Unequal-Spaced Channel-Allocation Problem in WDM Lightwave Systems. IEEE Transactions on Communications, Vol. 45, No. 3, pp. 352–359.

[2] Chraplyvy, A. R. 1990. Limitations on Lightwave Communications Imposed by Optical–Fiber Nonlinearities. J. Lightwave Technol., Vol. 8, pp. 1548–1557.

[3] Aggarwal, G. P. 2001. Nonlinear Fiber Optics. Second Edition, Academic Press, San Diego.

[4] Thing, Vrizlynn L. L., Shum, P., and Rao, M. K. 2004. Bandwidth–Efficient WDM Channel Allocation for Four-Wave Mixing-Effect Minimization. IEEE Transactions on Communications, Vol. 52, No. 12, pp. 2184–2189.

[5] Saaid, Nordiana M. 2010. Nonlinear Optical Effects Suppression Methods in WDM Systems with EDFAs: A Review. In Proceedings of the International Conference on Computer and Communication Engineering (ICCCE 2010), (11–13 May 2010), Kuala Lumpur, Malaysia.

[6] Forghieri, F., Tkach, R. W., Chraplyvy, A. R., and Marcuse, D. 1994. Reduction of Four–Wave Mixing Crosstalk in WDM Systems Using Unequally Spaced Channels. IEEE Photonics Technology Letters, Vol. 6, No. 6, pp. 754–756.

[7] Babcock, W. C. 1953. Intermodulation interference in radio systems, Bell Systems Technical Journal, pp. 63–73.

[8] Sardesai, H. P. 1999. A Simple Channel Plan to Reduce Effects of Nonlinearities In Dense WDM Systems. Lasers and Electro–Optics, (23–28, May–1999), pp. 183–184.

[9] Forghieri, F., Tkach, R. W., and Chraplyvy, A. R. 1995. WDM systems with unequally spaced channels. J. Lightwave Technol., Vol. 13, pp. 889–897.

[10] Hwang, B. and Tonguz, O. K. 1998. A generalized suboptimum unequally spaced channel allocation technique—Part I: In IM/DDWDMsystems. IEEE Trans. Commun., Vol. 46, pp. 1027–1037.

[11] Tonguz, O. K. and Hwang, B. 1998. A generalized suboptimum unequally spaced channel allocation technique—Part II: In coherent WDM systems. IEEE Trans. Commun., Vol. 46, pp. 1186–1193.

[12] Atkinson, M. D., Santoro, N., and Urrutia, J. 1986. Integer sets with distinct sums and differences and carrier frequency assignments for nonlinear repeaters. IEEE Trans. Commun., Vol. COM-34.

[13] Randhawa, R., Sohal, J. S. and Kaler, R.S. 2009. Optimum Algorithm for WDM Channel Allocation for Reducing Four-Wave Mixing Effects. Optik 120, pp. 898–904.

[14] http://www.compunity.org/events/pastevents/ewomp2004/jaillet_krajecki_pap_ew04.pdf

[15] Bloom, Gray S. and Golomb, S.W. 1977. Applications of Numbered Undirected Graphs. In Proceedings of the IEEE, Vol. 65, No. 4, (April 1977), pp. 562–570.

[16] Thing, Vrizlynn L. L., Rao, M. K. and Shum, P. 2003. Fractional Optimal Golomb Ruler Based WDM Channel Allocation. In Proceedings of the 8th Opto-Electronics and Communication Conference (OECC–2003), Vol. 23, pp. 631-632.

[17] Shearer, James B. 1998. Some New Disjoint Golomb Rulers. IEEE Transactions on Information Theory, Vol. 44, No. 7, pp. 3151–3153.

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

[19] Robinson, J. P. 1979. Optimum Golomb Rulers. IEEE Transactions on Computers, Vol. C-28, No. 12, (December 1979), pp. 943–944.

[20] James B. Shearer. 1990. Some New Optimum Golomb Rulers. IEEE Transactions on Information Theory. IT-36, (January 1990), pp. 183–184.

[21] 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).

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

[23] 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).

[24] 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.

[25] 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.

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

[27] Ayari, N., Thé Van Luong and A. 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.

[28] Bansal, S. 2014. Optimal Golomb Ruler Sequence Generation for FWM Crosstalk Elimination: Soft Computing Versus Conventional Approaches. Appl. Soft Comput. J. http://dx.doi.org/10.1016/j.asoc.2014.04.015.

[29] 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.

[30] 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.

[31] 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.

[32] 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.

[33] 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.

[34] Bansal, S., Singh, K., 2014. A Novel Soft–Computing Algorithm for Channel Allocation in WDM Systems. *International Journal of Computer Applications* (*IJCA*), Vol. 85, No. 9, pp. 19–26.

[35] Colannino, J. 2003. Circular and Modular Golomb Rulers. URL:http://cgm.cs.mcgill.ca/~athens/cs507/Projects/2003/JustinColannino/.

[36] Dimitromanolakis, A. 2002. Analysis of the Golomb Ruler and the Sidon Set Problems, and Determination of Large, Near-Optimal Golomb Rulers. Master's Thesis, Department of Electronic and Computer Engineering, Technical University of Crete.

[37] Dollas, A., Rankin, William T., and McCracken, D. 1998. A New Algorithm for Golomb Ruler Derivation and Proof of the 19 Mark Ruler. IEEE Transactions on Information Theory, Vol. 44, No. 1, pp. 379–382.

[38] "Project OGR", http://www.distributed.net/OGR.

[39] Cotta, C., Dotu, I., Fernandez, Antonio J., and Hentenryck, Pascal V. 2007. Local Search-Based Hybrid Algorithms for Finding Golomb Rulers. Kluwer Academic Publishers, Boston, Vol. 12, Issue 3, pp. 263–291.

[40] Lam, A.W. and Sarwate, D.V. 1988. On Optimal Time-hopping Patterns. IEEE Transactions on Communications (COM-36), pp. 380–382.

[41] Lavoie, P., Haccoun, D. and Savaria, Y. 1991. New VLSI Architectures for Fast Soft-Decision Threshold

Decoders. IEEE Transactions on Communications, Vol. 39, No. 2, pp. 200–207.

[42] http://mathworld.wolfram.com/PerfectRuler.html

[43] http://mathworld.wolfram.com/GolombRuler.html

[44] Robinson, J. P. and Bernstein, A. J. 1967. A Class of Binary Recurrent Codes with Limited Error Propagation. IEEE Transactions on Information Theory, IT-13, pp. 106–113.

[45] Yang, X.-S., and Deb, S. 2010. Engineering Optimisation by Cuckoo Search. Int. J. Mathematical Modelling and Numerical Optimisation. Vol. 1, No. 4, 330–343.

[46] Yang, X. S. and Deb, S. 2009. Cuckoo search via Levy flights, in: Proc. of World Congress on Nature & Biologically Inspired Computing (NaBic 2009), IEEE Publications, USA, pp. 210--214.

[47] Pratap R. 2010. Getting Started with Matlab A Quick Introduction for Scientists and Engineers. Oxford University Press, New York.

[48] http://www.research.ibm.com/people/s/shearer/grtab.html

[49] http://www.research.ibm.com/people/s/shearer/gropt.html