Detection of Low Auto Correlation Binary Sequences using Meta Heuristic Approach

B. Suribabu Naick
Asst Prof, Dept of ECE
Gitam University
Visakhapatnam
Andhra Pradesh

P. Rajesh Kumar
Dept of ECE
Andhra University
Visakhapatnam
Andhra Pradesh

ABSTRACT
This paper describes the method that constructs low autocorrelation binary sequences (LABS) which have applications in various engineering domains. We use a meta-heuristic search approach employing local search method known as Tabu Search, which solves mathematical optimization problems. Our paper is an extension to the existing one [1]. We were able to achieve new optimal solutions with our improved algorithm (especially for instances greater than 60 and less than 101) to that of the previous method [1].Instead of finding optimal solutions for odd skew- symmetric instances we found the optimal solutions for all the instances. We have conducted experiments over a large number of sequences thoroughly, for multiple times to ensure the results.

Keywords
Low autocorrelation binary sequences, Meta-Heuristic approach, Tabu search, combinatorial optimization.

1. INTRODUCTION
"LABS" is a Np hard combinatorial optimization problem. An optimal solution has to be found out from finite candidate solutions. In this problem, complete search of all possible solutions is not possible because of time and memory constraints. Labs have numerous applications in various fields especially in communication engineering, which brings the necessity to find the optimal labs for various instant sizes. In radars, labs are required for modulation in the process of pulse compression that enhances range resolution and also long range target detection capabilities [2].

Even in synchronization of communication systems, LABS is a requirement. Finding Labs for any instance is very difficult as it is similar to searching for a needle in a haystack. One of the complex problems of LABS is epistasis; change in one position affects several other positions. The complexity increases as instance size increases. The stochastic and evolutionary methods in the literature did not provide satisfactory approaches in solving this problem [3].Other than our Meta-heuristic approach, the time taken for other methods to obtain a solution to this problem proved to be extremely time-consuming. Some of the evolutionary methods in the literature are successful in obtaining near-optimal solutions [4-7]. We modified the Tabu search algorithm provided in [1].We obtained better results for some of the instances. Tabu Search is meta-heuristic approach that uses memory in search process and escapes from repeated patterns with the help of Tabu parameters [8]. The basic approach of local search strategy is to select randomly a candidate solution in the beginning and then look for better immediate neighbor. Obtaining the optimal solution is the goal, which would be the best possible solution.

2. BACKGROUND
The application of autocorrelation on binary sequences is described in this section, along with their fundamentals

2.1 Low Autocorrelation Binary Sequences.
Consider a binary sequence B of length I represented by $b_1, b_2, \ldots b_I$ with $b_n \in \{-1, 1\}^I$ for $1 \leq n \leq I$. The *aperiodic autocorrelation* of elements in sequence *B* with distance *d* is defined as

$$A_d(B) = \sum_{n=1}^{I-d} b_n\, b_{n+d}$$

The *energy* corresponding to sequence B is the quadratic sum of its correlations:

$$P(B) = \sum_{d=1}^{I-1} A_d^2\,(B)$$

Low autocorrelation problem for binary sequences with length B, represented as LABS (B), consists of finding a sequence of length I, with associated minimum energy. Consider an example. Labs instances are only determined by size of I. If we take an example for I=3, experimental results have shown that for this length the best binary sequences are

$\{1,-1,-1\}$,$\{-1,1,1\}$,$\{-1,-1,1\}$, $\{1,1,-1\}$. The energy corresponding to these sequences is 1, which is the minimum for I=3.There is an important property for labs instances, Symmetry that is even if the sequence is reversed or complemented the result remains the same. For example if we take the example of $I = 3$ and if we consider the sequence $\{1,-1,-1\}$ and reverse it, we get $\{-1,1,1\}$ which has the same energy as that of former one. With the help of symmetry finding at least one sequence for a length would give us a second sequence of same energy level. Each autocorrelation function contributes quadratically to the single energy P, a single large A_d can reduce the fitness of a sequence drastically. The A_d is not independent and therefore each change in the sequence leading to an improvement of certain A_d, will modify the other A_d as well. This is the feature of frustrated problem, characterized by a rough landscape where local maxima are many, steep and narrow [9].

Golay used Merit Factors to determine the quality of sequences [10, 11].Merit Factor is represented by the formula

$$G(B) = \frac{I^2}{2P(B)}$$

If $G_I G_I$ is assumed as the optimum value of the merit factor for sequences of length I, the LABS(I) problem can be alternatively defines as finding G_I such that

$$G_I = \max_{B \in \{-1,1\}I} G(B)$$

Ergodicity postulate is an assumption that estimates the asymptotic value for G_I, namely $G_I \to 12.32$ for $I \to \infty$

Merit factor is a qualitative measure for a sequence because it lends itself to better analytical treatment. It is also closely related to SNR used in signal processing [9]. LABS problem is a combinatorial optimization problem, the search space for the Labs(I) problem has the size 2^I, and the merit factor of the sequence computes in time $O(I^2)$.We already discussed the epistasis problem of LABS. Along with that there is another problem that we have to consider. Another difficulty is the presence small number of global optima for most values of I. It is a premise being taken after observing a large number of enumerated solutions. The corresponding search landscape has a large number of local minima. Global optima are not present in groups but are spread widely, which further increases the difficulty of the search. This problem has only one approach, which is to do implicit enumerative search among all 2^I possible sequences.

2.2 Related Work

One of the most recent works done on labs problem is by S.D.Prestwich [12]. He uses four methods to improvise the branch and bound technique of mertens. Prestwich gave the optimal solutions for skew-symmetric sequences (length 73 to 89). One of the other recent works is done by three people [13] (Steven Halim, Roland H.C. Yap, and Felix Halim). They designed an advanced Stochastic Local Search (SLS) algorithm for LABS problem. A white box visualization technique is used by them to determine the effectiveness of sls algorithms for LABS problem.

They obtained solutions for instances up to 77.They mentioned that these solutions are not optimum, and they can be improved further. Jozef-Kratica designed an advanced electromagnetism (EM) approach for solving LABS problem [14].It uses a meta-heuristic population-based algorithm which shrinks the search space in no time. With attract-repulse phenomenon, the candidate solutions are projected towards local optima. After that, local search is implemented along with scaling technique, by doing this the sample points converges to the nearest local optima. In local search, the highest priority is given to improvement strategy. Kratica has yielded optimal solutions up to instance size of 40.

Kratica designed an another model in which labs problem is treated as mixed integer quadratic programming problem (MIQP) [15].The algorithm used here is the first of its kind, and it is completely not defined. The search space is minimized with general optimization techniques. With the help of existing programming package, the LABS problem is modeled. The validity of MIQP analysis is also proved. With the help of MIQP technique, optimal solutions are obtained for skew-symmetric sequences in normal runtime. Kratica gave optimal solutions up to n=60.

In [1], near optimal solutions were provided for odd skew-symmetric sequences up to instance size of 201. In [2], Abhisek Ukil used number theory-based analysis for minimum energy levels, Barker codes. He proposed a theorem that states that there are finite numbers of possible energy levels, which are equally spaced at period four, for the given binary sequence of a given length. There is an assumption that only when $N<13$ barker sequences exist, this is true in the case of all N.This is been supported by further analysis, done in the literature.

By comparing all the previous techniques used in the literature, it becomes clear that the linear search is not feasible for this problem. For example, Golay stated in [10] that Lindner had executed a vast linear search and obtained the optimal solutions for $N \leq 32$.Mertens developed the famous branch and bound technique[16].In Mertens technique, the search tree will be pruned to speed up the backtracking search. He solved instances up to L = 48.The two former methods lacked scalability, and they cannot complete the search for larger instance sizes with limited resources namely time and computing power (that is memory).Even simulated annealing [17] and evolutionary methods [18-20] have been produced poor results. Militzer obtained the optimal solutions for very large instances [9] using a (μ, lambda)- Evolutionary Search algorithm. The process of recombination is not performed to obtain an offspring. Blind mutation operators are not selected, and problem solving is advanced with a heuristic known as pre-selection technique. Large aperiodic autocorrelation values are diminished with pre-selection technique. At first, the procedure is that a small set of y autocorrelations will be randomly picked from this set, the largest z correlations are selected. After that random flipping of bits will take place for n > 1.The obtained candidate solution is selected only if all z auto-correlations in the second set have been decreased. Otherwise, the method is repeated until an acceptable solution is obtained, or the trial limit is reached

3. META HEURISTICS FOR THE LABS PROBLEM

LABS is not a decision problem. Hence, the solution is determined by an objective function, and the idea is to obtain the least objective function values. The objective function value, also called solution quantity, of all candidate solutions, has to be determined. Labs problem is a minimization problem because the objective function is to be minimized.

	1	2	3	4
1	b_1b_2	b_2b_3	b_3b_4	b_4b_5
2	b_1b_3	b_2b_4	b_3b_5	
3	b_1b_4	b_2b_5		
4	b_1b_5			

H(B)

1	$b_1b_2 + b_2b_3 + b_3b_4 + b_4b_5$
2	$b_1b_3 + b_2b_4 + b_3b_5$
3	$b_1b_4 + b_2b_5$
4	b_1b_5

A(B)

Figure. 1

The above two tables represent the data structures used to efficiently recomputed fitness for a sequence of length 5. B = (b_1, b2, b3, b4, b5).

H(B) is used to store the products of elements separated by distance d efficiently. For each row the distance between the symbols which have to be multiplied and taken as pairs increases by one. A(B) just performs addition of all the

products of H(B) row wise.This kind of implementation is very useful to get the optimum solution.

3.1 Neighborhood structure

The major limitation of iterative improvement is that it gets stuck in local minima of the corresponding evaluation function. A large neighborhood has to be taken to solve this problem. The capability of any sls algorithm depends on the capacity of neighborhood and its relation. The idea is that, as the neighborhood size increases, the probability for obtaining good candidate solutions increases, so the probability of finding improvised steps in local domain increases. So selection of neighborhood relation is crucial.

In this case of LABS, we consider the neighborhood of a solution B, whose length is I .Neighborhood is obtained by flipping exactly one symbol in the sequence. This Neighborhood can be expressed constructively as

$$N(B) = \{flip(B,n)\ where\ n \in \{1,2,..,I\}\}$$

Where flip $(b_1 b_2 .. b_{n\ldots} b_I, n) = b_1 b_2 .. -b_{n\ldots} b_I$.

The optimum value for a given move has to be obtained, in other words, in local search; accurate re computation of the fitness function should take place for every move.

```
     function ValueFlip(B, n, H , A)
1 :  f := 0
2 :  for p := 1 to I - 1 do
3 :        v := A_P
4 :        if p ≤ I - n then v := v – 2H_pn end if
                    if
5 :        p < n then v :=      v   2Hp (n- end if
                                  2-  p)
6 :  f := f + v
7 :  end for
8 :  return f
     end function
```

Figure. 2

Whenever the function valueflip is called with the arguments provided, the energy value or the optimum function value is returned. The complexity of evaluation of this function is O(I^2). A new structure that completely reevaluates the solution after flipping one element in the sequence B would be rather unfit. A superior structure can be obtained by saving all computed products in a (I -1) × (I - 1) table H (B), such that H $(B)_{i\ j} = b_j b_{i+j}$ for j ≤ I-n, and storing the values of the various correlations in a I - 1 dimensional vector A (B), defined as $A(B)_d = A_d(B)$ for 1 ≤ d ≤ I-1. By observing that flipping a single symbol b_n multiplies by -1 the values of all cells in H (B) where b_n is present, the fitness of sequence flip (B,n) can be effectively recomputed in time O(I) as the output of the funtion Value Flip(B,n,H (B),A (B)), defined in Fig.2.

3.2 Local Search Strategies

The meta-heuristic that we have used is Tabu search. This selection is based on [1].We have modified the pseudo-code presented in [1]. Tabu search is a general sls method that methodically uses the memory for directing the search mechanism. Tabu search consistently utilizes a best improvement technique to pick the finest neighbor of the present candidate solution for each and every search step. In the case of a local optimum, this strategy results in worsening or reaching a plateau if local search steps does not change the evaluation function value, it has reached a plateau. In the process called cycling, the local search immediately returns to the previous candidate solution which is already visited. To avert cycling Tabu search restricts search steps to previously visited search locations. This restriction can be achieved by accurately memorizing earlier candidate solutions and seeing that no search step will yield back those values gain. More frequently, reversing the latest search steps is averted by prohibiting the reestablishment of solution factors that have just been eliminated from the present solution. The particular condition under which the Tabu parameters are neglected is known as aspiration criteria. A frequently used condition of aspiration criteria reverses the Tabu restriction on steps that result in advancement of necessary candidate solution. Tabu restrictions will be applied on search steps for a specific period known as tabu tenure.The developed step function is the basis of the Tabu mechanism. It utilizes appropriate neighbors for the function to determine whether the neighbors of present candidate solutions are Tabu or not, but satisfy the aspiration criteria. In the next level, a maximally enhancing step is randomly picked from this subset of appropriate neighbors. Tabu tenure is a major factor which affects the capability of Tabu search mechanism; it has to be taken appropriately. A very small Tabu tenure leads to search stagnation, whereas a very large value leads to restriction of the search path, and even good-grade solutions may be ignored. Tabu tenure setting can be obtained by empirical analysis, or the tenure can be dynamically adapted according to the conditions of the search mechanism.

In our case Tabu search uses the search history to adjust the Tabu tenure dynamically during the search. Because same candidate solutions are repeatedly encountered in LABS problem, so Tabu tenure is increased if no repetitions are found for a long period or the Tabu tenure can be gradually decreased. Moreover, an escape strategy is utilized to avert the search mechanism based on a chain of random alterations, to escape from getting trapped in specific search space region.

For example, consider the pseudo-code given in [1].It is a mechanism to escape from local optima. For this reason, they have used an I-dimensional vector M as an adjunct memory based on the previous step. So that if Mn = d, flipping the nth symbol in the present sequence is prohibited till the d-th iteration of the search process. The best optimal solution obtained in the present step of the search process is improved by the selected aspiration criterion, which restricts certain Tabu steps. The actual pseudo-code of this procedure is shown in Figure 3. For each iteration, the search moves to the best sequence in the present neighborhood that is not Tabu, and the respective flipped element is restricted for a random number of repetitions.

Function *TabuSearch* (*B, H, A*)

1: M_n*:* = 0*;* for 1 ≤ *n* ≤ I

2: *minTabu: = maxIters*/10

3: *extraTabu: = maxIters*/50

4: $B^* := B$; $f^* = \sum_{d=1}^{I-1} A_d^2$

5: for n: = 1 to maxiters do

6: $f^1 := \infty$

7: for *n*:= 1 to *I* do

Figure 3

4. EXPERIMENTAL RESULTS

The Tabu search algorithm runs on different instant sizes. We conducted the experiment for 61 instances that is between 40 and 100. The termination criterion for execution was finding the optimal solution. We did not keep any time limit for execution termination. At the same time, we have used 61 independent machines; all run independently to get the results. For each instance whose size is greater than 66, the run time for each machine was more than 72 hours. All experiments were performed on a 2.2 GHz core two duos PC under windows operating system.

4.1 Instances with Known Optima.

For instance size up to 60 the optimal solutions are already known, but to confirm that, we conducted experiments for instances between 40 and 60.we found the same energy levels as that in [1].The execution time was however reduced to a greater extent. There are two particular reasons for the speed that we have achieved. One reason is the algorithm that we have implemented and the restart strategies and intensification strategies that we have applied. Second reason is the kind of machines that we have used; as their speed was more than those used in [1].Figure 4 gives the information regarding energy levels of various instances between 40 and 60 instance sizes.

I	*P(B)*	*G(B)*	*ACTUAL LABS OBTAINED IN RUN LENGTH*
40	108	7.41	111211211343143131312
41	108	7.78	112112182222111111343
42	101	8.73	211211211343143131313
43	109	8.48	1132432111117212112213
44	122	7.93	525313113111222111211121
45	118	8.58	821211212312343211111111
46	131	*8.08*	732351111123212211212 1
47	135	8.18	111221111111211222224924
48	140	8.23	121121122212341238111111 3
49	136	8.83	112121211111213122313733 3
50	153	8.17	112112111231111113122245 27
51	153	8.50	234321111413131162121121 21
52	166	8.14	111141111333713221321212121
53	170	8.26	265223131112212151411121 11
54	175	8.33	121111111222211212141522653
55	171	8.85	1122122111111112114211 4A232 3
56	192	8.17	1112212112311111423211322167
57	188	8.64	1121222122111121721111136232 33
58	197	8.54	211234231121241831232131111 1
59	205	8.49	613212312111111311234122112 1242
60	218	8.26	111111211115311714211241222 4221

Figure 4

4.2 Large instances (61-77)

From instance size of 61, the necessity of Tabu search can be seen. It is a large factorial and takes lot of time even with local search strategy. If a normal linear search is applied for these larger instances, it is almost impossible to find the optimal solution. For instances between 61 and 79, the obtained results can be considered as optimum solutions. We have run the machines for almost 72 hours for each instance for multiple times to check if there is any other optimum. It is possible to achieve optimum results if the algorithm is further modified.

Figure 6 gives the information regarding energy levels of various instances between 61 and 77.

I	P(B)	G(B)	ACTUAL LABS IN RUNLENGTH
61	226	8.23	33211112111235183121221111311311
62	235	8.18	11221221271111151112114311142232 1
63	207	9.59	22122211512114511711111232323 1
64	208	9.85	22322411134112111511111721221221 2
65	240	8.80	13232321111171115411215112221221 1
66	265	8.22	24321123123112112124123181111111311
67	241	9.31	12112111211222B2221111111112224542
68	250	9.25	11111111141147232123251412112221212
69	274	8.69	111111111141147232123251412112221212
70	295	8.31	23244121172221416112521231111111 1
71	275	9.17	24124412417222211111311231121123112 1
72	300	8.64	11111141114441711511221421222242222
73	308	8.65	11111123112311221131112121141713223 74
74	349	7.85	11321321612333125111412121122511131111
75	341	8.25	12122132121211211111131111618433213232
76	338	8.54	11121111223432211113411421221122131 1B11
77	366	8.10	11111119134222243112331221341121211211 2

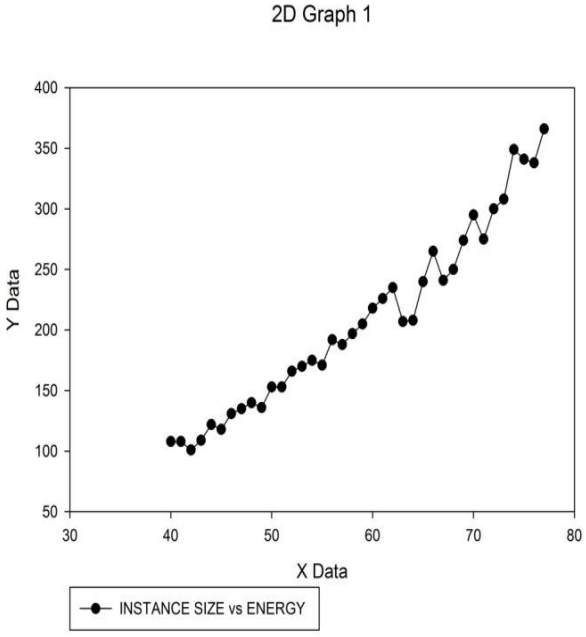
Figure 5

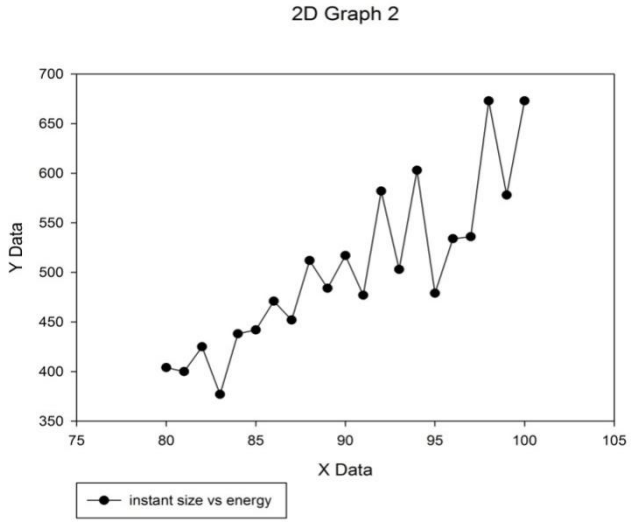
4.3 Large Instances (80-100)

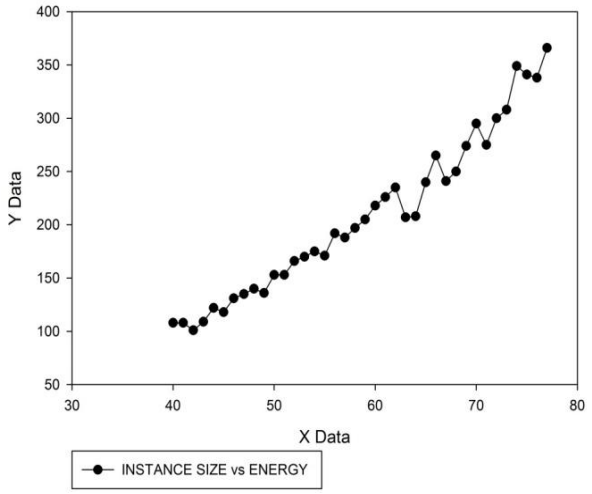
We achieved in obtaining new optimal solutions in this range. Unlike several other works in the past which only concentrate on odd skew-symmetric sequences, we tried to achieve energy levels to all instances in this range. The execution time for each sequence is more than 72 hours .we did not put any time limit on the runs. We manually terminated the program when the same values are repeating for a large period. It is possible to achieve lower energy levels than this, but algorithmic approach has to be different.

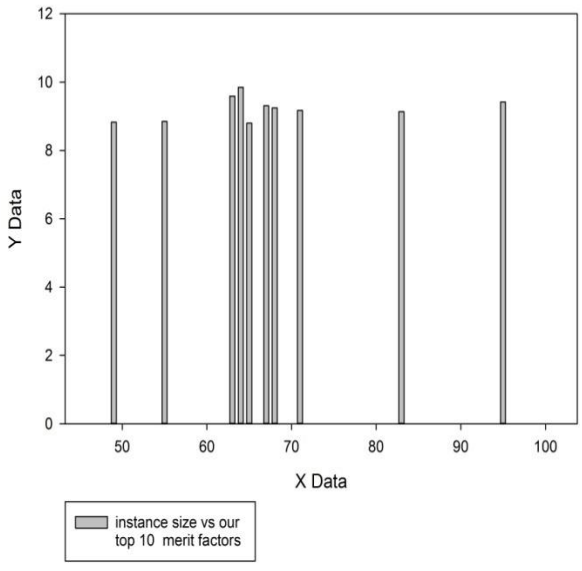
Instant size	our energy value	Our merit factor	Beenker [21]
80	404	7.92	-
81	400	8.20	7.32
82	425	7.91	-
83	377	9.14	7.81
84	438	8.05	-
85	442	8.17	7.03
86	471	7.85	-
87	452	8.39	7.46
88	512	7.56	-
89	484	8.18	7.56
90	517	7.83	-
91	477	8.68	7.13
92	582	7.27	-
93	503	8.61	7.23
94	603	7.32	-
95	479	9.42	7.15
96	534	8.62	-
97	536	8.78	7.35
98	673	7.13	-
99	578	8.49	7.28
100	673	7.42	-

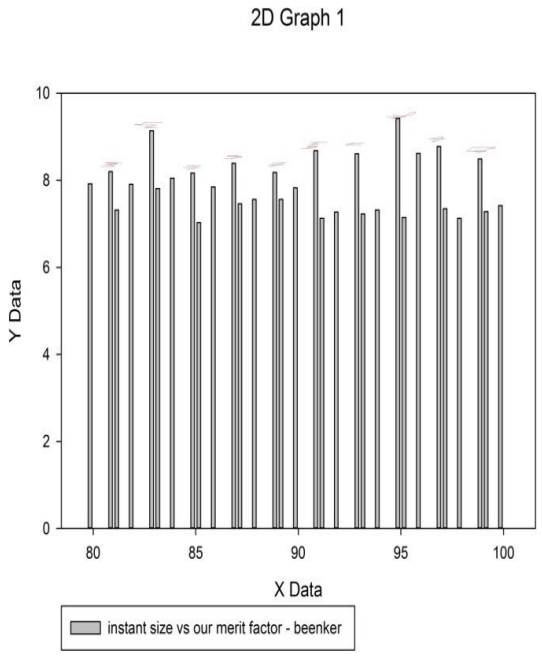
Figure 6











5. CONCLUSIONS

We have achieved a good range of merit factors for large instances, by modifying Tabu search algorithm provided in [1].A very small change in algorithm leads to major differences in results. We modified the algorithm, by analyzing the LABS fitness landscape structure and Tabu search behavior. Graphs are plot between the parameters instance size and energy (2D graph 1 and 2D graph 2). We have drawn a histogram comparison plot between merit factors of ours and Beenker [21], in almost every case the merit factors were high for odd values (2D graph 1).At last, the top ten merit factors with their corresponding instances are plotted(2D graph 2).

6. ACKNOWLEDGMENTS.

I thank my colleagues K.M.Chisti, R.sriharish, P.Chandrasekhar and V.Santoshkumar for helping me in making this paper.

7. REFERENCES

[1] J. E. Gallardo, C. Cotta, A. J. Fernandez, "Finding Low Autocorrelation Binary Sequences with Memetic Algorithms "

[2] Abhisek Ukil, "Low autocorrelation binary sequences: Number theory-based analysis for minimum energy level, Barker codes".

[3] J. E. Gallardo, C. Cotta, A. J. Fernandez, A Memetic algorithm for the low autocorrelation binary sequence problem, in: D. Thierens, et al. (Eds.), 9th Annual Conference on Genetic and Evolutionary Computation (GECCO 2007), ACM Press, New York, USA, 2007, pp. 1226-1233.

[4] P. Moscato, Memetic algorithms: A short introduction, in: D. Corne, M. Dorigo, F. Glover (Eds.), New Ideas in Optimization, McGraw-Hill, Maidenhead, Berkshire, England, UK, 1999, pp. 219-234.

[5] P. Moscato, C. Cotta, A gentle introduction to memetic algorithms, in: F. Glover, G. Kochenberger (Eds.), Handbook of Metaheuristics, Kluwer Academic Publishers, Boston MA, 2003, pp. 105-144.

[6] W. E. Hart, N. Krasnogor, J. E. Smith, Recent Advances in Memetic Algorithms, Vol.166 of Studies in Fuzziness and Soft Computing, SpringerVerlag, Berlin Heidelberg, 2005.

[7] N. Krasnogor, J. Smith, A tutorial for competent memetic algorithms: model, taxonomy, and design issues, IEEE Transactions on Evolutionary Computation 9 (5) (2005) 474-488.

[8] F. Glover, M. Laguna, Tabu Search, Kluwer Academic Publishers, Boston, 1997.

[9] [9] B. Militzer, M. Zamparelli, D. Beule, Evolutionary search for low auto correlated binary sequences, IEEE Transactions on Evolutionary Computation 2 (1) (1998) 34-39.

[10] M. J. E. Golay, The merit factor of long low autocorrelation binary sequences. IEEE Transactions on Information Theory 28 (3) (1982) 543-549.

[11] M. J. E. Golay, Sieves for low autocorrelation binary sequences, IEEE Transactions on Information Theory 23 (1) (1977) 43-51.

[12] S. D. Prestwich(2013), Improved Branch-and-Bound for Low Autocorrelation Binary Sequences, Annals of Operations Research

[13] Steven Halim, Roland H.C. Yap, and Felix Halim, Engineering Stochastic Local Search for the Low Autocorrelation Binary Sequence Problem. Cp-2008, Sydney, Australia, September 14-18, 2008

[14] Jozef Kratica, An Electromagnetism-Like Approach for Solving the Low Autocorrelation Binary Sequence Problem, INT J COMPUT COMMUN, ISSN 1841-9836 Vol.7 (2012), No. 4 (November), pp. 687-694

[15] Jozef Kratica, "a mixed integer quadratic programming model for the low autocorrelation binary sequence problem", Serdica J. Computing 6 (2012), 385–400.

[16] S. Mertens, Exhaustive search for low-autocorrelation binary sequences, Journal of Physics A: Mathematical and General 29 (1996) 473-481.

[17] J. Bernasconi, Low autocorrelation binary sequences: statistical mechanics and configuration state analysis, Journal de Physique 48 (4) (1987) 559- 567.

[18] Q. Wang, Optimization by simulating molecular evolution, Biological Cybernetics 57 (1987) 95-101.

[19] C. de Groot, D.WÄurtz, K. Ho®mann, Low autocorrelation binary sequences: Exact enumeration and optimization by evolutionary strategies, Optimization 23 (4) (1992) 369-384.

[20] A. Reinholz, Ein paralleler genetischer algorithms zur optimierung der binarien autocorrelations-function, Diploma Thesis. UniversitÄat Bonn (October 1993).

[21] S. Prestwich, Exploiting relaxation in local search, in: J. Pearson, M. Bohlin, M. Agren (Eds.), 1st International Workshop on Local Search Techniques in Constraint Satisfaction (LSCS 2004), Toronto, Canada, 2004, pp. 49-61.