Analyzing Fitness Landscapes for the Optimal Golomb Ruler Problem

Carlos Cotta and Antonio J. Fernández

Dept. Lenguajes y Ciencias de la Computación, ETSI Informática, University of Málaga, Campus de Teatinos, 29071 - Málaga, SPAIN

{ccottap,afdez}@lcc.uma.es

Abstract. We focus on the Golomb ruler problem, a hard constrained combinatorial optimization problem. Two alternative encodings are considered, one based on the direct representation of solutions, and one based on the use of an auxiliary decoder. The properties of the corresponding fitness landscapes are analyzed. It turns out that the landscape for the direct encoding is highly irregular, causing drift to low-fitness regions. On the contrary, the landscape for the indirect representation is regular, and exhibits comparable fitness-distance correlation to that of the former landscape. These findings are validated in the context of variable neighborhood search.

1 Introduction

Golomb rulers are a class of undirected graphs that, unlike usual rulers, measure more discrete lengths than the number of marks they carry. This is due to the fact that on any given ruler, all differences between pairs of marks are unique. This feature makes Golomb rulers really interesting in many practical applications, such as carrier frequency assignment [1], radio communication [2], X-ray crystallography [3], pulse phase modulation [4], and design of orthogonal codes [5, 6], among others [7–9]. Needless to say, it also introduces numerous constraints that hinder the search of short feasible rulers, let alone *optimal* Golomb rulers (OGR, i.e., the shortest Golomb ruler for a number of marks).

To date, no efficient algorithm is known for finding the shortest Golomb ruler for a certain number of marks: massive parallelism projects have been undertaken for several months in order to find the optimum instances of up to 23 marks [10]. Being such an extremely difficult combinatorial task, the Golomb ruler problem represents an ideal scenario for deploying the arsenal of evolutionary optimization.

In Sect. 2.2 we discuss some of the non-evolutionary techniques employed so far to solve OGRs. With respect to evolutionary ones, to the best of our knowledge, there have been four attempts to apply evolutionary algorithms (EAs) to the search for OGRs (see Sect. 2.3). These works are essentially empirical, and little has been so far done on the analysis of the properties of the underlying combinatorial landscapes. In this paper, we tackle this issue by analyzing two major problem representations under which evolutionary search can be conducted on this problem. To be precise, we consider the direct representation of solutions, and an indirect, decoder-based representation that uses a GRASP-like mechanism to perform the genotype-to-phenotype mapping. These landscapes are examined in Sect. 3, paying special attention to landscape regularity, and correlation measures. The variable neighborhood search metaheuristic is used to corroborate the outcome of this analysis in Sect. 4.

2 Background

The OGR problem can be classified as a fixed-size subset selection problem, such as e.g., the $p-$median problem [11]. It exhibits some very distinctive features though. A brief overview of the problem, and how it has been tackled in the literature is provided below.

2.1 Golomb Rulers

A *n-mark Golomb ruler* is an ordered set of n distinct non-negative integers, called *marks*, $a_1 < ... < a_n$, such that all the differences $a_i - a_j$ $(i > j)$ are distinct. Clearly we may assume $a_1 = 0$. By convention, a_n is the *length of the Golomb ruler*. A Golomb ruler with n marks is an optimal Golomb ruler if, and only if, (i) there exists no other n-mark Golomb rulers having smaller length, and (ii) the ruler is canonically "smaller" with respect to the the equivalent rulers. This means that the first differing entry is less than the corresponding entry in the other ruler. Fig. 1 shows an OGR with 4-marks. Observe that all distances between any two marks are different.

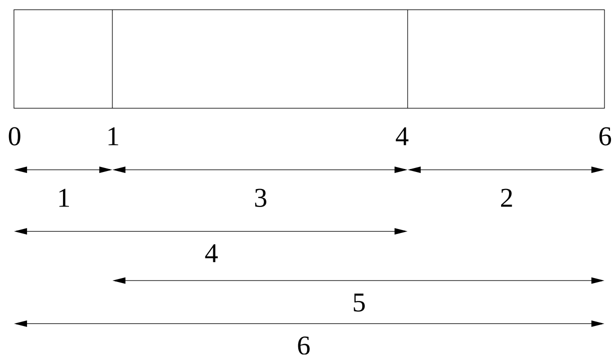

Fig. 1. A Golomb ruler with 4 marks

Typically, Golomb rulers are represented by the values of the marks on the ruler, i.e., in a n-mark Golomb ruler, $a_i = x$ $(1 \leqslant i \leqslant n)$ means that x is the mark value in position i. The sequence $(0, 1, 4, 6)$ would then represent the ruler in Fig. 1. An alternative representation consists of representing the Golomb ruler via the lengths of its segments, where the length of a segment of a ruler is defined as the distance between two consecutive marks. Therefore, a Golomb ruler can be represented with $n-1$ marks specifying the lengths of the $n-1$ segments that compose it. In the previous example, the sequence $(1,3,2)$ would encode the ruler depicted in Fig. 1.

2.2 Finding OGRs

The OGR problem has been solved using very different techniques. The evolutionary techniques found in the literature to obtain OGRs are described in Sect. 2.3. We provide here a brief overview of some of the most popular non-evolutionary techniques used for this problem.

Firstly, it is worth mentioning some classical algorithms used to generate and verify OGRs such as the *Scientific American* algorithm [12], the *Token Passing* algorithm (created by Professor Dollas at Duke University) and the *Shift* algorithm [13], all of them compared and described in [8].

In general, both non-systematic and systematic methods have been applied to find OGRs. Regarding the former, we can cite for example the use of geometry tools (e.g., projective plane construction and affine plane construction). With these approaches, one can compute very good approximate values for OGR with up to 158 marks [14]. As to systematic (exact) methods, we can mention the utilization of branch-and-bound algorithms combined with a depth first search strategy (i.e., backtracking algorithms), making use of upper-bounds set equal to the minimum length in the experiments. In this sense there exist several proposals: for example, Shearer [15] computed OGRs up to 16 marks. This approach has been also followed in massive parallelism initiatives such as the OGR project mentioned before. This project has been able to find the OGRs with a number of marks between 20 and 23, although it took several months to find optimum for each of those instances [8, 9, 16, 10].

Constraint programming techniques have also been used, although with limited success. For example, Smith and Walsh [17] obtained interesting results in terms of nodes in the branching schema. However, computation times are far from the results obtained by previous approaches. More recently, Galinier *et al.* [18] proposed a combination of constraint programming and sophisticated lower bounds for finding OGRs. They showed that using the same bound on different ways affects not only to the number of branches in the search tree but also to the computation time.

2.3 Evolutionary Approaches to the OGR

In this section will restrict here just to the evolutionary approaches to solve OGRs considered so far in the literature. In essence, two main approaches can be considered for tackling this problem. The first one is the *direct* approach, in which the EA conducts the search in the space $\mathcal{S}_G$ of all possible Golomb rulers. The second one is the *indirect* approach, in which an auxiliary $\mathcal{S}_{aux}$ space is used by the EA. In this latter case, a decoder [19] must be utilized in order to perform the $\mathcal{S}_{aux} \longrightarrow \mathcal{S}_G$ mapping. Both approaches will be discussed below.

Direct Approaches. In 1995, Soliday, Homaifar and Lebby [20] used a genetic algorithm on different instances of the Golomb ruler problem. They chosen the alternative formulation already mentioned where each chromosome is composed by a permutation of $n-1$ integers that represents the sequence of the $n-1$ lengths of its segments. Two evaluation criteria were followed: the overall length of the ruler, and the number of repeated measurements. This latter quantity was used in order to penalize infeasible solutions. The mutation operator consisted of either a permutation in the segment order, or a change in the segment lengths. As to crossover, it was designed to guarantee that descendants are valid permutations.

Later, Feeney studied the effect of hybridizing genetic algorithms with local improvement techniques to solve Golomb rulers [7]. The representation used consisted of an array of integers corresponding to the marks of the ruler. The crossover operator was similar to that used in Soliday *et al.*'s approach although a sort procedure was added at the end. The mutation operator consisted in adding a random amount in the range $[-x, x]$ –where x is the maximum difference between any pair of marks in any ruler of the initial population– to the segment mark selected for mutation. As it will be shown later, we can use a similar concept in order to define a distance measure on the fitness landscape.

Indirect Approaches. Pereira *et al.* presented in [21] a new EA approach using the notion of random keys [22] to codify the information contained in each chromosome. The basic idea consists of generating n random numbers (i.e., the keys) sampled from the interval $[0, 1]$ and ordered by its position in the sequence $1, \ldots, n$; then the keys are sorted in decreasing order. The indices of the keys thus result in a feasible permutation of $\{1, \cdots, n\}$. A similar evaluation criteria as described in [20] was followed. They also presented an alternative algorithm that adds a heuristic, favoring the insertion of small segments.

A related approach has been presented in [23]. This proposal incorporates ideas from greedy randomized adaptive search procedures (GRASP) [24] in order to perform the genotype-to-phenotype mapping. More precisely, the mapping procedure proceeds by placing each of the $n-1$ marks (the first mark is assumed to be $a_1 = 0$) one at a time; the $(i+1)^{\text{th}}$ mark can be obtained as $a_{i+1} = a_i + l_i$, where $l_i \geqslant 1$ is the i–th segment length. Feasible segment lengths (i.e., those not leading to duplicate measurements) can be sorted in increasing order. Now, the EA needs only specifying at each step the index of a certain segment within this list (obviously, the contents of the list are different in each of these steps). This implies that each individual would be a sequence $\langle r_1, \cdots, r_{n-1} \rangle$, where r_i would be the index of the segment used in the i–th iteration of the construction algorithm. Notice that in this last placement step it does not make sense to pick any other segment length than the smallest one. For this reason, $r_{n-1} = 1$; hence, solutions need only specify the sequence $\langle r_1, \cdots, r_{n-2} \rangle$. This representation of solutions is orthogonal [25], i.e., any sequence represents a feasible solution, and hence, standard operators for crossover and mutation can be used to manipulate them. This GRASP-based approach is reported to perform better than the previous indirect approach, and hence we use it in our further analysis.

3 Fitness Landscapes for the Golomb Ruler Problem

The notion of fitness landscapes was firstly introduced in [26] to model the dynamics of evolutionary adaptation in Nature. The fitness landscape analysis of a problem can help to identify its structure in order to improve the performance of search algorithms (e.g., to predict the behavior of a heuristic search algorithm, or to exploit some of its specific properties). For this reason, this kind of analysis has become a valuable tool for evolutionary-computation researchers.

In this section, we will analyze the fitness landscapes resulting from the two problem representations described before, the direct encoding of rulers, and the use of a GRASP-based decoder. We will assume below that n is the number of marks for a specific Golomb ruler $\mathbb{G}_n$, and that $a = \langle a_1, \ldots, a_n \rangle$ and $b = \langle b_1, \ldots, b_n \rangle$ are arbitrary solutions from $\mathbb{G}_n$. Analogously, $r = \langle r_1, \cdots, r_{n-2} \rangle$ and $r' = \langle r'_1, \cdots, r'_{n-2} \rangle$ are arbitrary vectors from $\mathbb{N}^{n-2}$, representing the vector of indices for selecting segment lengths. We denote by ψ the bijective function performing the genotype-to-phenotype mapping $\mathbb{N}^{n-2} \to \mathbb{G}_n$.

3.1 Distance Measures and Neighborhood Structure

We define a fitness landscape for the OGR as a triple $\langle S, f, d \rangle_n$ where $S = \mathbb{G}_n$ is the set of all the n-mark Golomb rulers (i.e., the solution set), f is a fitness function that attaches a fitness value to each of the points in S (i.e., $f(a)$ is equal to a_n, the length of a), and $d : S \times S \to \mathbb{N}$ is a function that measures a distance between any two points in S. We have defined one distance function for each of the Golomb ruler representations already commented. Specifically for the direct formulation (i.e., that based on lists of marks) we have defined the distance function d as follows:

$$d(a, b) = \max\{\mid b_i - a_i \mid, 1 \le i \le n\} \ . \tag{1}$$

In other words, $d(a, b)$ returns the maximum difference between any two corresponding marks in a and b. Also, for our indirect formulation (i.e., the GRASP-based formulation) we have defined the distance function d as the L1 norm (the Manhattan distance) on the vector of indices, i.e.,

$$d(a, b) = d(\psi(r), \psi(r')) = \sum_{i=1}^{n-2} \mid r_i - r'_i \mid \ . \tag{2}$$

A first issue to be analyzed regards the neighborhood structure induced by these distance measures. More precisely, consider the number of solutions reachable from a certain point in the search space, by a search algorithm capable of

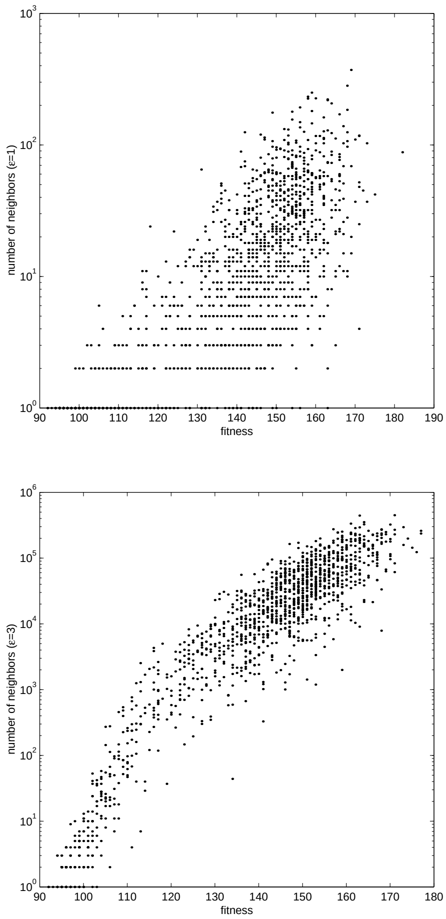

Fig. 2. Number of neighbors for different values of the local radius ϵ in a 12-mark Golomb ruler problem. From top to bottom and left to right, $\epsilon = 1$, 2, 3, and 4. Notice the log-scale in the Y-axis.

making jumps of a given distance. In the direct formulation, this number of solutions turns out to be variable for each point of S, as shown in Fig. 2. We have implemented and used a logic-programming based constraint solver to solve the Golomb ruler constraint satisfaction problem for an arbitrary number of marks. Our solver, implemented in GNU Prolog [27], is based on the model proposed in [28]. In particular, the solver generates a list of all possible distances between any pair of marks i, j ($i < j$ and $i, j \in \{1, \ldots, n\}$) in the ruler and then imposes a global constraint *all-different* on this list instead of imposing the set of binary inequalities between any two marks i, j. The efficiency is further improved by adding some redundant constraints leading to an improvement of the domain pruning. This solver calculates the number of possible neighbors that are located within a given distance ϵ (called the *local radius*) of certain solution a (i.e., it obtains the cardinality of the set $\{\langle c_1, \ldots, c_n \rangle \in S \mid a_i - \epsilon \leq c_i \leq a_i + \epsilon, 1 \leq i \leq n\}$). The solver is then applied to a large sample of solutions covering a wide range of fitness values.

The outcome of this experiment indicates that the connectivity of the fitness landscape increases with worse fitness values. Furthermore, this effect is stronger as we increase the neighborhood radius (see Fig. 2). This kind of irregularity is detrimental for search algorithm navigating this landscape [29], since the neighborhood structure tends to guide the search towards low-fitness regions. This means that a search algorithm on this landscape would have to be continuously fighting against this drifting force. On the contrary, notice that the fitness landscape of the indirect formulation is perfectly regular, since its topology is isomorphic to $\mathbb{N}^{n-2}$. In principle, this regularity makes this landscape more navigable since no underlying drift effect exists.

3.2 Fitness-Distance Correlation

Fitness-distance correlation (FDC) [30] is one of the most widely used measures for assessing the structure of the landscape. It also constitutes a very informative measure to evaluate the problem difficulty for evolutionary algorithms [31]. FDC allows quantifying the correlation between fitness values, and the distance to the nearest optimum in the search space. Landscapes with a high FDC typically exhibit a *big valley structure* [32] (this is not always the case though [30, 33]).

It is typically assumed that low FDC is associated with problem difficulty for local search. Nevertheless, the interplay of this property with other landscape features is not yet well understood. Indeed, it will be later shown how landscape ruggedness and neighborhood irregularity can counteract high FDC values.

Focusing on the problem under consideration, the optimum value opt_n for n-mark Golomb rulers is known (up to $n = 24$, enough for our analysis). We can then obtain a sample of m locally-optimal solutions $A = \{a_1, \ldots, a_m\} \subset S$ and easily calculate the sets $F = \{f_i \mid f_i = f(a_i), 1 \leq i \leq m, a_i \in A\}$ and $D = \{d_i \mid d_i = d(a_i, opt_n), 1 \leq i \leq m, a_i \in A\}$. Then we can compute the correlation coefficient as $FDC = C_{FD}/(\sigma_F \sigma_D)$, where

$$C_{FD} = \frac{1}{m} \sum_{i=1}^{m} (f_i - \overline{f})(d_i - \overline{d}) \tag{3}$$

is the covariance of F and D, and $\sigma_F, \sigma_D, \overline{f}$ and $\overline{d}$ are, respectively, the standard deviations and means of F and D. Observe that this definition depends on the definition of the distance function, and as shown in Section 3.1, we consider two different definitions for the two problem representations.

The FDC values computed for the two representations are shown in Fig. 3. In all cases, locally optimal solutions are computed by using hill climbing from a fixed sample of seed feasible solutions. Notice firstly the high correlation for the direct formulation, specially for low values of the local radius ϵ. This can be explained by the fact that the fitness of a solution is actually the value

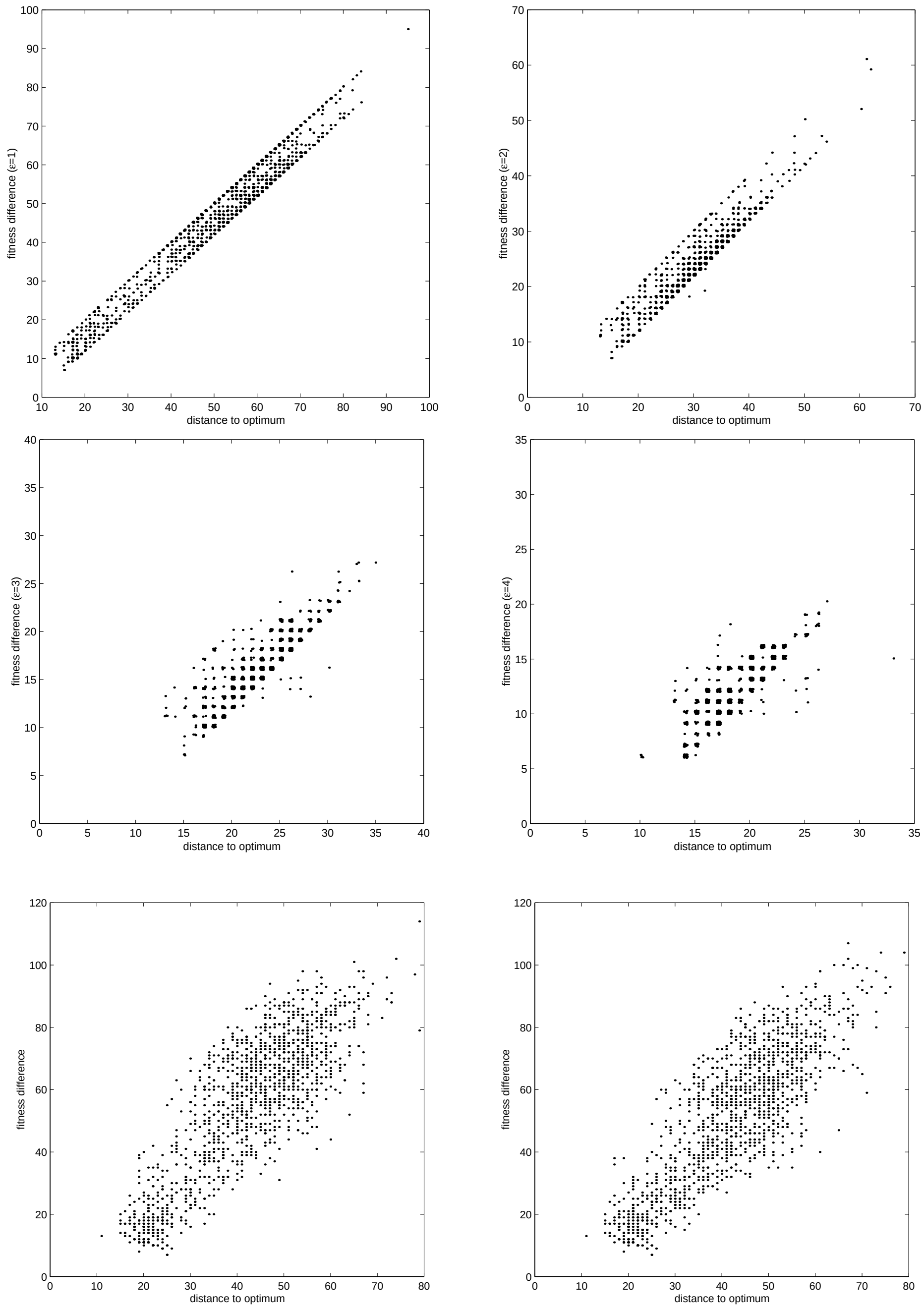

Fig. 3. Fitness distance correlation in a 12-mark Golomb ruler problem. The upper four figures correspond to the direct formulation (from top to bottom and left to right, $\epsilon = 1$, 2, 3, and 4), and those at the bottom to the indirect formulation ($\epsilon = 1$, and 2).

of the last mark, and this value will not change above the given ϵ within the neighborhood. FDC starts to degrade for increasing values of this local radius. To be precise, FDC values for $\epsilon = 1$ up to $\epsilon = 4$ are 0.9803, 0.9453, 0.8769, and 0.8221 respectively. In the case of the indirect formulation ($\epsilon = 1$), the FDC value is 0.8478, intermediate between $\epsilon = 3$, and $\epsilon = 4$. These results indicate that the indirect formulation can attain FDC values comparable to those of the direct formulation, but without suffering from some of the problems of the latter. Actually, the high FDC values for the direct formulation are compensated by two related facts, namely that there is a drift force towards low-fitness regions as mentioned in Sect. 3.1, and that the number of local optima is higher for low values of the local radius, specially in the high-fitness region.

4 Experimental Validation

In order to confirm our findings from the landscape analysis, we have performed some experiments using a variable neighborhood search algorithm (VNS) [34]. This is a generalization of the conspicuous hill climbing algorithm in which different neighborhoods are used during the search. More precisely, a collection of neighborhoods $\mathcal{N}_1, \cdots, \mathcal{N}_k$ is considered. The search starts from the first neighborhood in the collection, and proceeds to the next one when no improvement can be found. Whenever an improvement is found, the search continues from the first neighborhood again. The underlying idea here is the fact that locally optima solutions for one neighborhood are not necessarily locally optimal in the next one. Hence, the algorithm can escape from such non-common local optima, and progress further towards the global optimum. The search finishes when a solution is found that is locally optimal for all neighborhoods.

The VNS algorithm has been deployed on the two problem representations considered. In both cases, neighborhoods $\mathcal{N}_1$ up to $\mathcal{N}_4$ have been considered, where $\mathcal{N}_i(x)$ refers to the set of solutions within distance i (in the corresponding fitness landscape) of solution x. Neighborhoods are explored by sampling 100 solutions, and retaining the best one. If no improving solution is found, the neighborhood is considered exhausted.

The results for $n = 12$ marks are shown in Table 1. VNS^i indicates that VNS is restricted to neighborhoods $\mathcal{N}_1$ to $\mathcal{N}_i$. As it can be seen, the results of the indirect representation are better than those of direct representation for VNS^1 and VNS^2. The difference between both representations tends to decrease for increasing radius: very similar results (no statistical difference according to a Mann-Whitney U test) are obtained in both cases for VNS^3, and the direct representation turns out to be better for VNS^4.

Two facts must be noted here. First of all, the magnitude of the radius has not the same meaning in the different representation, and hence, the data in Table 1 should not be interpreted as paired columns. Secondly, the computational cost (not shown in Table 1) of exploring each neighborhood is quite different (around three orders of magnitude larger in the case of the direct representation, as measured in a P4–3GHz 1GB PC under Windows XP). This is so, even allowing

Table 1. Results (averaged for 30 runs) of variable neighborhood search on the two representations. As a reference, starting solutions have a mean value of 127.57 ± 7.64.

	direct		indirect		indirect (exhaustive)	
	mean ± std.dev.	median	mean ± std.dev.	median	mean ± std.dev.	median
VNS^1	127.43 ± 7.73	125	114.10 ± 4.11	115	112.80 ± 3.91	113
VNS^2	120.63 ± 6.09	120.5	107.43 ± 3.86	107	108.93 ± 3.71	109
VNS^3	104.70 ± 5.19	105	105.83 ± 2.61	105	101.77 ± 3.03	101.5
VNS^4	98.87 ± 2.49	100	105.17 ± 2.55	105	97.33 ± 1.97	97

an exhaustive exploration of the neighborhood for the indirect representation. The results in this latter case are shown in the two rightmost columns of Table 1. Notice the improvement with respect to the direct representation.

5 Conclusions

This work has tried to shed some light on the question of what makes a problem hard for a certain search algorithm. We have focused on the Golomb ruler problem, an extremely interesting problem due to its simple definition yet tremendous hardness. It is also a problem for which several representations had been tried, but that lacked an analysis of the combinatorial properties of the associated fitness landscapes.

Our analysis indicates that the high irregularity of the neighborhood structure for the direct formulation introduces a drift force towards low-fitness regions of the search space. This contrasts with other problems in which the drift force is beneficial, since it guides the search to high-fitness regions (see [29]). The indirect formulation that we have considered does not have this drawback, and hence would be in principle more amenable for conducting local search in it. The fact that fitness-distance correlation is very similar in both cases also support this hypothesis.

The empirical validation provides consistent results: a VNS algorithm using the indirect formulation can outperform a VNS counterpart working on the direct representation, in a low computational cost scenario. It is also very interesting to note that these results are also consistent with the performance of evolutionary algorithm on this problem, despite the fact that even when the representation may be the same, they do not explore exactly the same landscape.

Future work will be directed to confirm these conclusions in the context of other constrained problems. We will also try to identify new problems in which the irregularity of the neighborhood structure play such a central role, and study alternative formulations for these.

Acknowledgements.

This work is partially supported by Spanish MCyT and FEDER under contracts TIC2002-04498-C05-02 and TIN2004-7943-C04-01.

References

1. Fang, R., Sandrin, W.: Carrier frequency assignment for non-linear repeaters. Comsat Technical Review **7** (1977) 227–245
2. Babcock, W.: Intermodulation interference in radio systems. Bell Systems Technical Journal (1953) 63–73
3. Bloom, G., Golomb, S.: Aplications of numbered undirected graphs. Proceedings of the IEEE **65** (1977) 562–570
4. Robbins, J., Gagliardi, R., Taylor, H.: Acquisition sequences in PPM communications. IEEE Transactions on Information Theory **33** (1987) 738–744
5. Klove, T.: Bounds and construction for difference triangle sets. IEEE Transactions on Information Theory **35** (1989) 879–886
6. Robinson, J., Bernstein, A.: A class of binary recurrent codes with limited error propagation. IEEE Transactions on Information Theory **13** (1967) 106–113
7. Feeney, B.: Determining optimum and near-optimum Golomb rulers using genetic algorithms. Master thesis, Computer Science, University College Cork (2003)
8. Rankin, W.: Optimal Golomb rulers: An exhaustive parallel search implementation. Master thesis, Duke University Electrical Engineering Dept., Durham, NC (1993)
9. Dollas, A., Rankin, W.T., McCracken, D.: A new algorithm for Golomb ruler derivation and proof of the 19 mark ruler. IEEE Transactions on Information Theory **44** (1998) 379–382
10. OGR project: `http://www.distributed.net/ogr/` (on-going since September 14, 1998)
11. Mirchandani, P., Francis, R.: Discrete Location Theory. Wiley-Interscience (1990)
12. Dewdney, A.: Computer recreations. Scientific American (1986) 14–21
13. McCracken, D.: Minimum redundancy linear arrays. Senior thesis, Duke University, Durham, NC (1991)
14. Shearer, J.B.: Golomb ruler table. Mathematics Department, IBM Research, `http://www.research.ibm.com/people/s/shearer/grtab.html` (2001)
15. Shearer, J.: Some new optimum Golomb rulers. IEEE Transactions on Information Theory **36** (1990) 183–184
16. Garry, M., Vanderschel, D., et al.: In search of the optimal 20, 21 & 22 mark Golomb rulers. GVANT project, `http://members.aol.com/golomb20/index.html` (1999)
17. Smith, B., Walsh, T.: Modelling the Golomb ruler problem. In: Workshop on non-binary constraints (IJCAI'99), Stockholm (1999)
18. Galinier, P., Jaumard, B., Morales, R., Pesant, G.: A constraint-based approach to the Golomb ruler problem. In: 3rd International Workshop on Integration of AI and OR Techniques (CP-AI-OR'2001). (2001)
19. Koziel, S., Michalewicz, Z.: A decoder-based evolutionary algorithm for constrained parameter optimization problems. In Bäeck, T., Eiben, A., Schoenauer, M., Schwefel, H.P., eds.: Parallel Problem Solving from Nature V. Volume 1498 of Lecture Notes in Computer Science. Springer-Verlag, Berlin Heidelberg (1998) 231–240
20. Soliday, S., Homaifar, A., Lebby, G.: Genetic algorithm approach to the search for Golomb rulers. In Eshelman, L., ed.: 6th International Conference on Genetic Algorithms (ICGA'95), Pittsburgh, PA, USA, Morgan Kaufmann (1995) 528–535
21. Pereira, F., Tavares, J., Costa, E.: Golomb rulers: The advantage of evolution. In Moura-Pires, F., Abreu, S., eds.: Progress in Artificial Intelligence, 11th Portuguese Conference on Artificial Intelligence. Number 2902 in Lecture Notes in Computer Science, Berlin Heidelberg, Springer-Verlag (2003) 29–42

22. Bean, J.: Genetic algorithms and random keys for sequencing and optimization. ORSA Journal on Computing **6** (1994) 154–160
23. Cotta, C., Fernández, A.: A hybrid GRASP - evolutionary algorithm approach to Golomb ruler search. In Yao, X., et al., eds.: Parallel Problem Solving From Nature VIII. Number 3242 in Lecture Notes in Computer Science, Birmingham, UK, Springer (2004) 481–490
24. Resende, M., Ribeiro, C.: Greedy randomized adaptive search procedures. In Glover, F., Kochenberger, G., eds.: Handbook of Metaheuristics. Kluwer Academic Publishers, Boston MA (2003) 219–249
25. Radcliffe, N.: Equivalence class analysis of genetic algorithms. Complex Systems **5** (1991) 183–205
26. Wright, S.: The roles of mutation, inbreeding, crossbreeding and selection in evolution. In Jones, D., ed.: 6th Intenational Congress on Genetics. Volume 1. (1932) 356–366
27. Diaz, D., Codognet, P.: GNU Prolog: beyond compiling Prolog to C. In Pontelli, E., Costa, V., eds.: 2nd International Workshop on Practical Aspects of Declarative Languages (PADL'2000). Number 1753 in LNCS, Boston, USA, Springer-Verlag (2000) 81–92
28. Barták, R.: Practical constraints: A tutorial on modelling with constraints. In Figwer, J., ed.: 5th Workshop on Constraint Programming for Decision and Control, Gliwice, Poland (2003) 7–17
29. Bierwirth, C., Mattfeld, D., Watson, J.P.: Landscape regularity and random walks for the job shop scheduling problem. In Gottlieb, J., Raidl, G., eds.: Evolutionary Computation in Combinatorial Optimization. Volume 3004 of Lecture Notes in Computer Science., Berlin, Springer-Verlag (2004) 21–30
30. Jones, T.: Evolutionary Algorithms, Fitness Landscapes and Search. Phd thesis, Santa Fe Institute, University of New Mexico, Alburquerque (1995)
31. Jones, T., Forrest, S.: Fitness distance correlation as a measure of problem difficulty for genetic algorithms. In Eshelman, L., ed.: Proceedings of the 6th International Conference on Genetic Algorithms, San Francisco, CA, Morgan Kaufmann Publishers (1995) 184–192
32. Reeves, C.: Landscapes, operators and heuristic search. Annals of Operational Research **86** (1999) 473–490
33. Boese, K., Kahng, A., S., M.: A new adaptive multi-start technique for combinatorial global optimizations. Operations Research Letters **16** (1994) 101–113
34. Mladenović, N., Hansen, P.: Variable neighborhood search. Computers and Operations Research **24** (1997) 1097–1100