A Study on the Efficiency of Neutral Crossover Operators in Genetic Algorithms Applied to the Bin Packing Problem

Eneko Osaba
Deusto Institute of Technology
University of Deusto
Bilbao, Spain
e.osaba@deusto.es

Fernando Diaz
Telecommunications Dept.
University of Deusto
Bilbao, Spain
fernando.diaz@deusto.es

Roberto Carballedo
Deusto Institute of Technology
University of Deusto
Bilbao, Spain
roberto.carballedo@deusto.es

Idoia de la Iglesia
Deusto Institute of Technology
University of Deusto
Bilbao, Spain
delaiglesia@deusto.es

Enrique Onieva
Deusto Institute of Technology
University of Deusto
Bilbao, Spain
enrique.onieva@deusto.es

Asier Perallos
Deusto Institute of Technology
University of Deusto
Bilbao, Spain
perallos@deusto.es

ABSTRACT

This paper examines the influence of neutral crossover operators in a genetic algorithm (GA) applied to the one-dimensional bin packing problem. In the experimentation 16 benchmark instances have been used and the results obtained by three different GAs are compared with the ones obtained by an evolutionary algorithm (EA). The aim of this work is to determine whether an EA (with no crossover functions) can perform similarly to a GA.

Categories and Subject Descriptors

I.2.8 [**Artificial Intelligence**]: Problem solving, Control Methods and Search—*Heuristic methods*

Keywords

Genetic Algorithm, Crossover Operator, Bin Packing.

1. INTRODUCTION

Today, the genetic algorithm (GA) is one of the most used techniques to solve complex optimization problems. Since its formulation [3], the GA has been applied to a wide range of problems. Annualy, a lot of research studies use this kind of algorithms, either to solve a problem [9], to analyze some theoretical aspects of GAs [8], or to compare them with other techniques [6]. GAs are widely used in the industrial, transport and logistics fields. This is because in those contexts there are many problems with a simple definition, but complex to be solved. Such problems are known as combinatorial optimization problems. There are a lot of combinatorial optimization problems, being the one-dimensional bin packing (1d-BPP) one of the best known.

In this paper, a study on the efficiency of blind crossover operators in GAs applied to the 1d-BPP is carried out. The goal of this work is to determine if an evolutionary algorithm (EA) can perform as good as a classic GA for the 1d-BPP. Thereby, it can be concluded whether blind crossover functions are valuable for addressing this problem.

2. ONE-DIMENSIONAL BIN PACKING

The packing of items into boxes or bins is a daily task in distribution and production. Depending on the item characteristics, as well as the form and capacity of bins, a wide amount of different packing problems can be formulated. In [4] an introduction to bin-packing problems can be found. The 1d-BPP is the simplest one, and it has been used frequently as benchmarking problem [1]. The 1d-BPP consists of a set of items $I = \{i_1, i_2, \dots, i_n\}$, each with an associated size s_i, and an unlimited supply of bins with the same capacity q. The objective of the 1d-BPP is to pack all the items into a minimum number of bins. In this way, the objective function is the number of bins, which has to be minimized.

In this study the solutions are encoded as a permutation of items. To count the number of bins needed in one solution, the item sizes are accumulated in a variable ($sumSize$). When $sumSize$ exceeds q, the number of bins is incremented in 1, and $sumSize$ is reset to 0. Thereby, supposing a simple instance of 10 items $I = \{i_1, i_2, \dots, i_{10}\}$, each one with the same size (30), and q=120. One possible solution could be $X = (i_1, i_4, i_2, i_7, i_9, i_{10}, i_6, i_5, i_3, i_8)$, and its fitness would be 3 (the number of bins needed to hold all the items).

3. EXPERIMENTATION

In the experimentation conducted in this work the performance of four different algorithms are compared. The first technique is an EA based only on mutations. The remaining are three classic GAs. All meta-heuristics use the same parameters and functions, with the exception of the crossover operator. While the EA has no recombination phase, each GA has a different function. The neutral operators employed are the Order Crossover (OX) [2], Order Based Crossover (OBX) [10], and the Half Crossover (HX) [7]. These functions are well-known, and they are widely used in the literature.

All the techniques have a population composed by 50 randomly created individuals. Each GA has a crossover probability (p_c) of 95%, and a mutation probability (p_m) of 5%. On the other hand, the EA has a p_c=0 and p_m=100%. Regarding the parents selection criteria of the GAs: first, each individual is selected with a probability equal to p_c. If

Instance		EA			GA with OX			GA with OBX			GA with HX		
Name	Optima	Avg.	S. dev.	Time	Avg.	S. dev.	Time	Avg.	S. dev.	Time	Avg.	S. dev.	Time
N2C1W1_A	48	**53.0**	0.73	0.02	53.4	0.75	0.35	53.6	0.68	0.09	54.5	1.19	0.03
N2C1W1_B	49	**53.4**	0.50	0.02	54.0	0.79	0.21	53.9	0.79	0.09	54.7	0.86	0.03
N2C2W1_A	42	**45.9**	0.69	0.01	46.3	0.72	0.23	46.1	0.93	0.06	47.5	0.83	0.03
N2C2W1_B	50	54.0	1.28	0.02	**53.8**	0.70	0.24	54.3	0.81	0.06	55.6	0.94	0.03
N3C2W2_A	107	**120.4**	1.43	0.07	121.1	1.41	1.64	121.8	1.64	0.45	124.2	1.52	0.18
N3C2W2_B	105	117.0	1.00	0.07	**116.9**	1.57	1.92	117.7	2.00	0.37	118.8	1.40	0.18
N3C3W1_A	66	**73.0**	0.94	0.06	73.7	0.80	1.42	73.4	0.94	0.35	75.0	1.00	0.12
N3C3W1_B	71	**78.9**	0.89	0.06	80.1	0.94	1.33	79.7	1.42	0.37	81.1	0.91	0.13
N3C3W4_A	89	**99.9**	1.23	0.09	100.3	1.21	1.48	101.2	1.36	0.40	102.4	1.50	0.15
N3C3W4_B	88	**98.0**	1.10	0.06	98.8	1.51	1.44	98.7	1.38	0.40	100.3	1.50	1.76
N4C1W1_A	240	**272.9**	1.57	0.35	278.6	2.54	7.83	273.0	7.34	5.91	278.1	1.72	2.10
N4C1W1_B	262	**295.5**	2.16	0.43	300.5	3.02	7.75	298.5	1.57	6.52	301.0	2.25	2.15
N4C1W1_C	241	**273.4**	1.54	0.42	278.2	3.18	7.03	277.2	2.69	5.23	278.2	2.00	2.24
N4C2W1_A	210	**242.5**	1.82	0.46	246.0	2.45	7.90	244.5	2.16	5.40	247.9	2.35	2.24
N4C2W1_B	213	**246.5**	1.32	0.44	249.8	3.11	7.84	250.0	3.19	5.08	252.1	1.79	2.31
N4C2W1_C	213	**246.2**	1.77	0.49	250.2	3.02	7.69	248.3	2.74	5.96	252.0	2.51	2.10

Table 1: Results of the four techniques applied to the 1d-BBP

one individual is selected for the recombination, the other participant is selected randomly. In relation to the survivor function, the 50% of the surviving population is selected by the elitist method. The remaining 50% is selected at random. About the ending criteria, the execution of each meta-heurstic ends when there are $n + \sum_{k=1}^{n} k$ generations without improvements in the best solution, where n is the size of the problem.

All the instances have been picked from the Scholl/Klein benchmark[1]. Each instance has been executed 30 times, and for each one, the results average, standard deviation and average runtime (in seconds) are shown (Table 1).

Looking at these results, it can be seen how the EA obtains better results for the 87.5% of the instances (14 out of 16). In the remaining two instances, GA with OX performs better. Additionally, in general (62.5% of the cases), the standard deviation of the EA is lower than the one of the other algorithms. This characteristic gives robustness to the EA, something important in real environments. Finally, regarding runtimes, the EA needs less execution time in the 100% of the instances.

The reason why the EA needs less execution time can be explained in the following way: the mutation operator consists of a simple modification in one chromosome, so it can be made in a short time. On the other hand, the crossover operates with two individuals, and its working way is more complex, needing more runtime. The reason why EA gets better results can also be explained. For the 1d-BPP, crossover helps to the exploration capacity of the technique, but it does not help to perform an exhaustive search. To perform a deeper search, a function that conducts small jumps in the space of solutions becomes necessary. For the 1d-BPP, the mutation function can handle this goal. These arguments are also based on a recent study on the traveling salesman problem [5].

4. CONCLUSIONS

In this paper a short study on the efficiency of the crossover phase in GAs applied to the 1d-BPP has been presented. Using the same parameters and funcions (except the crossover function), the performance of three different classic GAs has been compared with the one of an EA. According to the experimentation conducted, it can be concluded that the use of crossover functions does not lead to an improvement in results. In addition, the use of this kind of functions increases the runtime and the complexity of the technique without providing any visible improvement.

[1] *http://www.wiwi.uni-jena.de/entscheidung/binpp/index.htm.*

5. REFERENCES

[1] M. J. Brusco, H. F. Köhn, and D. Steinley. Exact and approximate methods for a one-dimensional minimax bin-packing problem. *Annals of Operations Research*, 206(1):611–626, 2013.

[2] L. Davis. Applying adaptive algorithms to epistatic domains. In *International joint conference on artificial intelligence*, volume 1, pages 161–163, 1985.

[3] J. H. Holland. *Adaptation in natural and artificial systems.* Michigan Press, 1975.

[4] S. Martello and P. Toth. *Knapsack problems. Algorithms and Computer Implementations.* John Wiley & Sons, 1990.

[5] E. Osaba, R. Carballedo, F. Diaz, and A. Perallos. Analysis of the suitability of using blind crossover operators in genetic algorithms for solving routing problems. In *Symposium on Applied Computational Intelligence and Informatics*, pages 17–22, 2013.

[6] E. Osaba, F. Diaz, and E. Onieva. Golden ball: a novel meta-heuristic to solve combinatorial optimization problems based on soccer concepts. *Applied Intelligence*, pages 1–22, 2014.

[7] E. Osaba, E. Onieva, R. Carballedo, F. Diaz, A. Perallos, and X. Zhang. A multi-crossover and adaptive island based population algorithm for solving routing problems. *Journal of Zhejiang University SCIENCE C*, 14(11):815–821, 2013.

[8] M. Serpell and J. E. Smith. Self-adaptation of mutation operator and probability for permutation representations in genetic algorithms. *Evolutionary Computation*, 18(3):491–514, 2010.

[9] S. Swain and D. Mohapatra. Genetic algorithm-based approach for adequate test data generation. In *Intelligent Computing, Networking, and Informatics*, pages 453–462. Springer, 2014.

[10] G. Syswerda. Schedule optimization using genetic algorithms. *Handbook of genetic algorithms*, pages 332–349, 1991.