A Grid-based Genetic Algorithm for Multimodal Real Function Optimization

Jose M. Chaquet and Enrique J. Carmona
Dpto. de Inteligencia Artificial, Escuela Técnica Superior de Ingeniería Informática, Universidad Nacional de Educación a Distancia, Madrid, Spain

Keywords: Genetic Algorithm, Multimodal Real Functions, Grid-based Optimization, Integer-real Representation.

Abstract: A novel genetic algorithm called GGA (Grid-based Genetic Algorithm) is presented to improve the optimization of multimodal real functions. The search space is discretized using a grid, making the search process more efficient and faster. An integer-real vector codes the genotype and a GA is used for evolving the population. The integer part allows us to explore the search space and the real part to exploit the best solutions. A comparison with a standard GA is performed using typical benchmarking multimodal functions from the literature. In all the tested problems, the proposed algorithm equals or outperforms the standard GA.

1 INTRODUCTION

Optimization of real problems are normally hard to solve because deals with multimodal functions and complex fitness landscapes. Issues as multiple local optimum, premature convergence, ruggedness or deceptiveness are some of the difficulties (Weise et al., 2009). To face this kind of problems, the use of Evolutionary Computing (EC) paradigms is very attractive. One of the most used EC paradigms for the tuning of multimodal real functions are Evolution Strategies (ES). Real coding representation and self-adaptation of the optimal mutation strengths make ES suitable to these type of domains. However, in this work, we want to investigate how to improve the performance of a standard Genetic Algorithm (GA) based on real-valued or floating-point representation. In fact, since this type of representation was proposed (Davis, 1991; Janikow and Michalewicz, 1991; Wright, 1991), there have been many works in the literature that have been devoted to this purpose. Each of them was focused in different aspects as, for example, new mutation operators (Deep and Thakur, 2007b; Korejo et al., 2010), new crossover operators (Deep and Thakur, 2007a; Garcia-Martinez et al., 2008; Tutkun, 2009), or new self-adaptive selection schemes (Affenzeller and Wagner, 2005).

The main two ideas of this paper are to use an integer-real vector for individual representation and to discretize the search space by using a grid. Such mixed representation allows breaking down the standard search process in two types of search made simultaneously. One of them is constrained to the grid and allows making a global search in the domain tuning the integer part (exploration). The other one tunes the real part making a local search of the best individuals (exploitation). The new algorithm implemented using that methodology is called Grid-based Genetic Algorithm (GGA).

The idea of using a grid to facilitate the search process has been also reported in the so-called cell-to-cell mapping method (Hsu, 1988). In a similar way, other approaches based on the subdivision of the search space into boxes were presented in (Dellnitz et al., 2001). In both works, stochastic search is introduced for the evaluation of the boxes, but each box is considered *once* during the search process which is not the spirit of GAs. On the other hand, a mixture of different type of numbers for representation was also used in (Li, 2009), where the coding involves using real, integer and nominal values. Nevertheless, in that work, each vector component represents a different dimension in the search space, that is, two or more components are not treated as forming a unique entity. Conversely, in GGA, each couple of integer-real components represents implicitly one dimension.

The rest of the paper is organized as follows. Section 2 describes the new proposed algorithm. Next, section 3 presents the problems used as benchmarking and the final configuration (parameters and operators) used for our GGA and a real-coded standard GA employed for comparison. Section 4 presents a com-

M. Chaquet J. and J. Carmona E..
A Grid-based Genetic Algorithm for Multimodal Real Function Optimization.
DOI: 10.5220/0004114401580163
In *Proceedings of the 4th International Joint Conference on Computational Intelligence* (ECTA-2012), pages 158-163
ISBN: 978-989-8565-33-4

parative of the results obtained using both algorithms. Finally, in section 5, the main conclusions and future works are given.

2 ALGORITHM DESCRIPTION

The objective of multimodal real function optimization, $\mathbb{R}^N \to \mathbb{R} : F(\mathbf{x})$ with $\mathbf{x} = (x_1, x_1, \cdots, x_N)$, is to find the global optimum in a N-dimensional space $\mathbb{R}^N$. In formal notation, the problem to be solved is the following:

$$\mathbf{x}_{min} | F(\mathbf{x}_{min}) \leq F(\mathbf{x}), \forall \mathbf{x} \in \mathbb{R}^n. \quad (1)$$

The proposed algorithm works with a grid in the N-dimensional space defined by $(\Delta_1, \Delta_2, \ldots, \Delta_N) \in \mathbb{R}^N$, where each Δ_i represents the uniform grid step size in each dimension. In GGA, the genotype of each individual is an integer-real vector, $(s_1, s_2, \ldots, s_N, \alpha_1, \alpha_2, \ldots, \alpha_N)$, where $s_i \in \mathbb{Z}$, and $\alpha_i \in \mathbb{R}$ must fulfil the constraint $0 \leq \alpha_i < \Delta_i$. Then the phenotype of each individual is computed in the following way:

$$x_i = s_i \cdot \Delta_i + \alpha_i, \; i = 1, \ldots, N. \quad (2)$$

The main idea of this special encoding is to discretize the search space in order to facilitate the global optimum seeking using two types of search simultaneously: global and local search. The former uses the integer part of each individual and is constrained to the grid. It enables the exploration in the search space. On the other hand, the later uses the real part of each individual to exploit those solutions whose integer part is close to an optimum. Fig. 1 shows an example of the mentioned encoding in a two-dimensional problem ($N = 2$). The grid step size, Δ_i, is defined by the user implicitly. Thus, the user chooses the number of grid intervals n_Δ. This value will be the same for all dimensions. If the definition domain range of each variable is $x_i \in [x_{i,min}, x_{i,max}]$, each Δ_i is computed by

$$\Delta_i = (x_{i,max} - x_{i,min}) / n_\Delta, \quad (3)$$

being Δ_i fixed for all the individuals and for all generations, i. e., it is not evolved by the algorithm.

The following describes the parameters and operators used by the GGA. The initial population is initialized randomly covering all the domain space, i. e., $s_i \in [\lfloor x_{i,min}/\Delta_i \rfloor, \lfloor x_{i,max}/\Delta_i \rfloor]$ and $\alpha_i \in [0, \Delta_i]$, where $\lfloor x \rfloor$ denotes the function which returns the largest integer not greater than x. As parent selection, it is used the tournament method with size t_{size}. Once parents are selected, the crossover operator is performed with a defined probability p_{cross}. If the cross operator is not invoked, the children are just an exact copy of their

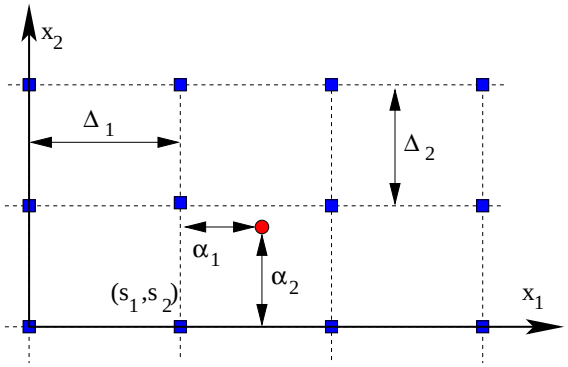

Figure 1: Graphical example of encoding in a two-dimensional problem. The point represents an individual, being its relative coordinates (α_1, α_2) respect to the grid node $(\Delta_1 s_1, \Delta_2 s_2)$. The absolute coordinates are $(s_1\Delta_1 + \alpha_1, s_2\Delta_2 + \alpha_2)$ where Δ_1 and Δ_2 are the grid step sizes in each dimension.

parents. Standard one-point crossover is used for the integers s_i and real numbers α_i and the same crossover point is used for both.

After crossover, the mutation operator is applied over each gene of each child with a specified probability p_{mut}. Unlike the crossover operator, two different mutation operators have been implemented for the real and integer part, called α-mutation and s-mutation, respectively. The first one is chosen with a predefined probability $p_{\alpha/s}$, whereas the second operator is chosen with a probability $1 - p_{\alpha/s}$. So, for each gene, one of them is chosen if $p < p_{mut}$. In other case, the gene is not mutated. Being $U(a,b)$ a sample of a random variable with a uniform distribution in the interval $[a,b]$, the α-mutation operator is defined as follows:

$$\begin{aligned} \alpha_i' &\leftarrow \alpha_i + U(-\Delta_i\sigma_\alpha, \Delta_i\sigma_\alpha) \\ if\ \alpha_i' < 0 \Rightarrow &\begin{cases} \alpha_i'' \leftarrow \alpha_i' + \Delta_i \\ s_i' \leftarrow s_i - 1 \end{cases} \\ if\ \alpha_i' > \Delta_i \Rightarrow &\begin{cases} \alpha_i'' \leftarrow \alpha_i' - \Delta_i \\ s_i' \leftarrow s_i + 1 \end{cases} \end{aligned}, \quad (4)$$

where the parameter $\sigma_\alpha \ll 1$ controls the mutation strength. As we can see, first a low random value is added to α_i. Then, it is checked if the new value is inside the feasible range. If not, both values s_i and α_i are updated accordingly. With this strategy, a fine tuning can be done during the exploitation phase because the mutation operator is not disruptive and allows changing the grid node if necessary.

Because the s_i components of each individual are thought to explore the search space, the s-mutation operator only modifies s_i and maintains invariable the α_i values. Here the idea is to use the typical nonuniform mutation with normal distribution but, because we are working with integers, that distribu-

tion is substituted by the difference of two geometrical distributed variables $Z(\sigma_s)$, which it is more suitable for integer numbers (Rudolph, 1994):

$$s_i' \leftarrow s_i + Z_1(\sigma_s) - Z_2(\sigma_s). \quad (5)$$

According to (Rudolph, 1994), a geometrical distributed variable Z with dispersion σ can be generated from a uniform distribution with the following procedure:

$$\begin{aligned} u &\leftarrow U(0,1) \\ \psi &\leftarrow 1 - \frac{\sigma}{1+\sqrt{1+\sigma^2}} \\ Z &\leftarrow \left\lfloor \frac{\ln(1-u)}{\ln(1-\psi)} \right\rfloor \end{aligned} \quad (6)$$

The dispersion of the two geometrical distributions, σ_s, is a control parameter of the mutation strength. The objective of this operator is to create enough diversity in the population to facilitate the exploration of the search space.

After the variation operators are applied, it is employed a generational model, i.e., the whole population is replaced by its offspring, which forms the next generation. All these steps are repeated for several generations until the stop condition is fulfilled. Two stop conditions are checked: a maximum number of generations or a fitness value close to the global optimum below a predefined threshold, $\theta_{solution}$.

Once the GGA is described, here are some notes about the implemented standard GA used for comparison with our algorithm. Real-valued representation has been employed. The population initialization is performed randomly over the predefined range for each dimension. As in GGA, tournament selection is employed for choosing the parents. Once all the parents are selected, the cross operator is applied with certain probability p_{cross} over parent couples. If crossover is not selected, the children are just a copy of their parents. As crossover operator, intermediate recombination is employed: $x_i^{child} = \varphi x_i^{parent1} + (1-\varphi) x_i^{parent2}$ for some $\varphi \in [0,1]$. The mutation operator is used with a certain probability p_{mut} for each gene and each individual. Nonuniform mutation with Gaussian distribution $N(0,1)$ is used: $x_i' \leftarrow x_i + \sigma N(0,1)$. In order to facilitate the exploration in the first part of the algorithm and a correct exploitation at the final generations, a linear variation of the standard deviation between the first generation (with a value $\sigma = \sigma_{initial}$) and a prescribed generation G_{linear} (decreasing until the value $\sigma = \sigma_{final}$) is employed. From G_{linear} generation till the end, the standard deviation is maintained constant and equal to σ_{final}. As in GGA, a generational model is used and the same stop criterion is adopted.

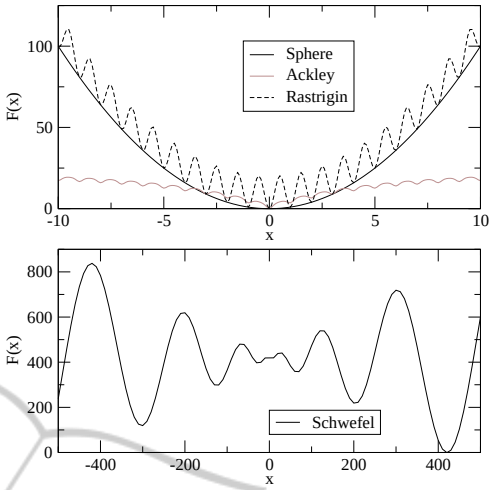

Figure 2: Sphere, Ackley and Rastrigin (top) and Schwefel (down) functions in one-dimensional problem $N = 1$.

3 BENCHMARKING AND ALGORITHM SETUP

In this section the multimodal functions used for benchmarking are described. Also the specific parameters employed in the GA and GGA are given. Four functions extracted from the literature are used: *Sphere* (unimodal), *Ackley*, *Rastrigin* and *Schefwel*:

$$F_{Sphere}(\mathbf{x}) = \sum_{i=1}^{N} x_i^2, \quad (7)$$

$$F_{Ackley}(\mathbf{x}) = 20 + e - 20\exp\left(-0.2\sqrt{\sum_{i=1}^{N}\frac{x_i^2}{N}}\right) - \exp\left(\sum_{i=1}^{N}\frac{\cos(2\pi x_i)}{N}\right), \quad (8)$$

$$F_{Rastrigin}(\mathbf{x}) = 10N + \sum_{i=1}^{N} x_i^2 - 10\sum_{i=1}^{N}\cos(2\pi x_i), \quad (9)$$

$$F_{Schwefel}(\mathbf{x}) = 418.982988N - \sum_{i=1}^{N} x_i \sin\sqrt{|x_i|}. \quad (10)$$

All the functions, except the Schwefel one, have a global minimum in the origin with zero value, being the definition range in all dimensions $[-10,10]$. Fig. 2 (top) shows the first three functions in one-dimensional problem. Schwefel function is defined in $-500 \le x_i \le 500$ and its global minimum (as well with zero value) is located in $x_i = 420.968746$. See Fig. 2 (down) for one-dimensional representation. The search of the global optimum for the three functions that have the global minimum in the origin should produce an individual with all its α_i equal to

zero. This fact is extremely unusual in real applications. So in order to study the behaviour of the GGA in more realistic scenarios, three new functions, called π-*functions*, are built doing a shift in all dimensions in the following way:

$$F_\pi(\mathbf{x}) = F(\mathbf{x} - \Pi), \tag{11}$$

where F is the original Sphere, Ackley or Rastrigin function and Π is a vector with all its components equal to π.

On the other hand, all the above functions are isotropic (same behaviour in all dimensions). Some real problems are non-isotropic. Therefore it has been implemented a new function family called *modified* functions doing a change of variable

$$x_i = 2^{-i+1} y_i. \tag{12}$$

In this way, for example, the modified Sphere or *m-Sphere* function is defined as:

$$F_{m-Sphere}(\mathbf{y}) = \sum_{i=1}^{N} 2^{-2i+2} y_i^2. \tag{13}$$

Likewise, obtaining the other modified functions is immediate. For all the modified functions, the global minimum is in $y_i = 0$, except for the *m-Schwefel*, where the minimum is in $y_i = 420.968746 \cdot 2^{i-1}$. The definition ranges for the four modified function are now $x_{i,min} 2^{i-1} \leq y_i \leq x_{i,min} 2^{i-1}$, being $x_{i,min}$ and $x_{i,max}$ the range limits of the original functions. Note that for the first dimension x_1, the original range is preserved.

Finally a new family of functions called modified π or *m-*π *functions* are defined shifting and scaling the variables simultaneously applying Eq. (11) and (12) .

Once the test functions are introduced, Tables 1 and 2 give the parameter settings used in GA and GGA. Note that the two algorithms have the most similar parameter values as possible in order to do a fair comparison. The values have been experimentally chosen. The parameter tuning is straightforward, and the same values used for all the runs demonstrate the robustness of the method.

Table 1: GA parameters.

Parameter	Value
Population size	200
Tournament t_{size}	3
p_{cross}	0.8
p_{mut}	0.05
Max. generations	2000
G_{linear}	1000
$\sigma_{initial} / (x_{i,max} - x_{i,min})$	0.3
$\sigma_{final} / (x_{i,max} - x_{i,min})$	10^{-6}
Threshold $\theta_{solution}$	$\leq 10^{-4}$

Table 2: GGA parameters.

Parameter	Value
Intervals n_Δ	20
Population size	200
Tournament t_{size}	3
p_{cross}	0.8
p_{mut}	0.05
$p_{\alpha/s}$	0.9
σ_α	10^{-2}
σ_s	6
Max. generations	2000
Threshold $\theta_{solution}$	$\leq 10^{-4}$

4 RESULTS

A total of 14 test functions have been employed, including the original Sphere, Ackley, Rastrigin and Schwefel, and the variants π, modified and π-modified functions. The number of dimensions considered in all the cases are $N = 10$. Each run has been repeated 20 times using different seeds for the random number generator and averages were taken. Tables 3 and 4 give the percentage of successful runs and the average number of generation for achieving the stop condition both for GA and GGA respectively. As well it is given the standard deviation of the generation number. A run is considered successful when the fitness of the best individual is less than $\theta_{solution}$ in a generation number less or equal than the maximum generation number. Only successful runs are considered for the generation number averages.

According to the results, the proposed algorithm equals or outperforms the GA, both in percentage of successful runs and in number of generations required. Because of the linear variation of the standard deviation in the GA during the first 1000 generations, the required generations for achieving the stop condition is larger than 1000. This limitation is not observed in GGA, because no variation in the parameters is imposed with the generation number. A shorter linear variation step has been tested in GA using $G_{linear} = 500$, but although the average number of generations is reduced in some cases, the successful rates are reduced as well.

Fig. 3, 4 and 5 shows a typical run of GGA for Schwefel function. For this example $\Delta_i = 1000/20 = 50$ and the optimum is placed in $x_i = s_i \Delta_i + \alpha_i = 8 \cdot 50 + 20.968746$. In Fig. 3, the fitness of the best individual and the population average fitness are plotted versus the generation. We can differentiate two phases. The first one where the majority of the inte-

Table 3: Experimental results obtained for the 14 test functions using GA. Percentage of successful runs, average number of generations, and standard deviations of the generations are provided.

Case	% Success	Generations	$\sigma_{generations}$
Sphere	100	1013	43
Ackley	100	1212	73
Rastrigin	25	1134	375
Schwefel	0	-	0
π-Sphere	100	1003	1
π-Ackley	100	1233	108
π-Rastrigin	10	1310	208
m-Sphere	100	1003	1
m-Ackley	100	1231	82
m-Rastrigin	35	1205	383
m-Schwefel	0	-	0
m-π-Sphere	100	1003	1
m-π-Ackley	100	1256	99
m-π-Rastrigin	20	1140	345

Table 4: Experimental results obtained for the 14 test functions using GGA. Percentage of successful runs, average number of generations, and standard deviations of the generations are provided.

Case	% Success	Generations	$\sigma_{generations}$
Sphere	100	410	58
Ackley	100	601	87
Rastrigin	100	322	61
Schwefel	100	678	293
π-Sphere	100	402	45
π-Ackley	100	592	77
π-Rastrigin	100	355	64
m-Sphere	100	399	49
m-Ackley	100	623	74
m-Rastrigin	100	325	72
m-Schwefel	100	640	125
m-π-Sphere	100	444	48
m-π-Ackley	100	621	77
m-π-Rastrigin	100	372	75

ger numbers s_i are adjusted (see Fig. 4, first 100 generations), and a second phase where the fine tuning of the reals α_i are sought (Fig. 5, from generations 100 to 700). Note that in the second phase, when a α_i exits from the feasible range $[0, \Delta_i]$, the integer counterpart s_i is updated accordingly and the α_i can evolve in the new interval (Figs. 4 and 5).

5 CONCLUSIONS

A new algorithm for multimodal real optimization, called GGA, is presented. Here the definition domain

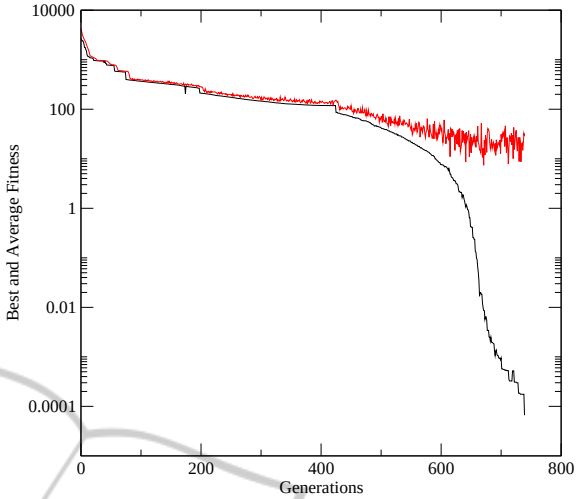

Figure 3: Convergence history of a typical run of Schwefel function: fitness of the best individual and population average fitness.

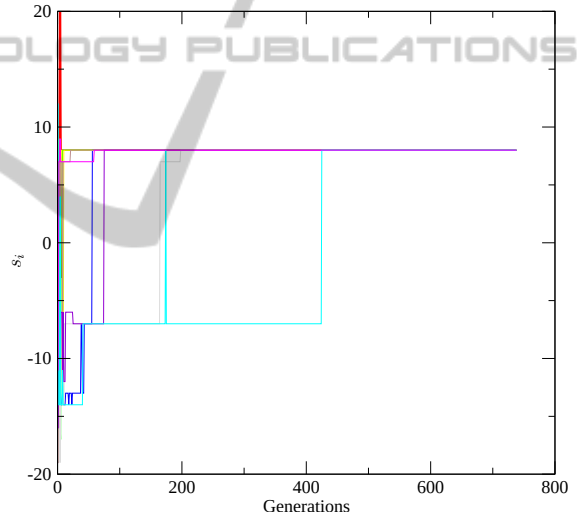

Figure 4: Convergence history of a typical run of Schwefel function: steps s_i of the best individual.

of the optimization problem is discretized using a grid and each individual is represented by integer and real number couples. This frame facilities the search process allowing two types of search simultaneously: a global search for exploration and a local search for exploitation. The global optimum of 14 test multimodal functions have been correctly found with a 100% successful rate. A comparison with a standard real-coded GA has been also performed. The proposed algorithm equals or outperforms the standard GA in percentage of successful runs and number of generations needed to reach the global optimum.

These preliminary results are encouraging. Nevertheless, the new method should be tested in more functions and real problems, and compared with other

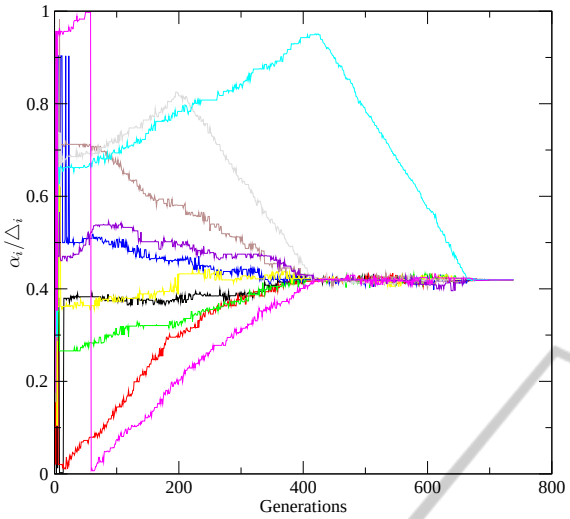

Figure 5: Convergence history of a typical run of Schwefel function: α_i/Δ_i ratios of the best individual.

paradigms of evolutionary algorithms. A sensitivity analysis of the algorithm to best tune the parameters should be necessary. As future work, auto-adaption techniques for some of the algorithm parameters could be investigated, such as Evolutionary Strategies do for the mutation strengths. Auto-adaptation is attractive because simplifies the setup of the algorithm (low number of parameters are required) and better results can be obtained (the parameters are automatically evolved using the best one according to the environment).

ACKNOWLEDGEMENTS

This work was suported by the Spanish Ministerio de Economía y Competitividad under the Project TIN2010-20845-C03-02.

REFERENCES

Affenzeller, M. and Wagner, S. (2005). *Offspring Selection: A New Self-Adaptive Selection Scheme for Genetic Algorithms*, pages 218–221. Number 2 in Adaptive and Natural Computing Algorithms. Springer Vienna.

Davis, L. (1991). *Hybridization and numerical representation*, volume The Hanbook of Genetic Algorithms, pages 61–71. New York: Van Nostrand Reinhold.

Deep, K. and Thakur, M. (2007a). A new crossover operator for real coded genetic algorithms. *Applied Mathematics and Computation*, 188(1):895–911.

Deep, K. and Thakur, M. (2007b). A new mutation operator for real coded genetic algorithms. *Applied Mathematics and Computation*, 193(1):211–230.

Dellnitz, M., Schutze, O., and Sertl, S. (2001). Finding zeros by multilevel subdivision techniques. *IMA Journal of Numerical Analysis*, 22:2002.

Garcia-Martinez, C., Lozano, M., Herrera, F., Molina, D., and Sanchez, A. (2008). Global and local real-coded genetic algorithms based on parent-centric crossover operators. *European Journal of Operational Research*, 185(3):1088–1113.

Hsu, C. (1988). Cell-to-cell mapping. a method of global analysis for nonlinear systems. *ZAMM - Journal of Applied Mathematics and Mechanics*, 68(12):654–655.

Janikow, C. and Michalewicz, Z. (1991). *An experimental comparison of binary and floating point representations in genetic algorithms*, volume Proceedings of the Fourth International Conference on Genetic Algorithms, pages 31–36. Morgan Kaufmann.

Korejo, I., Yang, S., and Li, C. (2010). A directed mutation operator for real coded genetic algorithms. In *EvoApplications (1)*, volume 6024 of *Lecture Notes in Computer Science*, pages 491–500. Springer.

Li, R. (2009). *Mixed-Integer Evolution Strategies for Parameter Optimization and their Applications to Medical Image Analysis*. PhD thesis, Leiden.

Rudolph, G. (1994). An evolutionary algorithm for integer programming. In *Parallel Problem Solving from Nature - PPSN III, Lecture Notes in Computer Science*, pages 139–148. Springer.

Tutkun, N. (2009). Optimization of multimodal continuous functions using a new crossover for the real-coded genetic algorithms. *Expert Systems with Applications*, 36(4):8172–8177.

Weise, T., Zapf, M., Chiong, R., and Nebro, A. J. (2009). *Why Is Optimization Difficult?*, volume 193 of *Studies in Computational Intelligence*, chapter 1, pages 1–50. Springer-Verlag Berlin Heidelberg.

Wright, A. (1991). Genetic algorithms for real parameter optimization. In *Foundations of Genetic Algorithms*, pages 205–218. Morgan Kaufmann.