8

Using Retroviral Iterative Genetic Algorithm to Solve Constraint Global Optimization Problems

Renato Simões Moreira[1,3], Otávio Noura Teixeira[1,2,3], Walter Avelino da Luz Lobato[1,3], Hitoshi Seki Yanaguibashi[1,3] and Roberto Célio Limão de Oliveira[1]

[1]*Programa de Pós-Graduação em Engenharia Elétrica, Universidade Federal do Pará*
[2]*Laboratório de Computação Natural, Centro Universitário do Estado do Pará*
[3]*Movimento Evolucionário e Cooperativo para a Construçao do Artificial - MEC*[2]*A*
Brazil

1. Introduction

Over the years the viruses have been treated as villains in the destruction of organic structures, resulting from the disappearance of entire species and even causing long and fatal epidemics (such as HIV). However, its effectiveness to perpetuate themselves is really impressive, although their acceptance as life forms are still under debate. Retroviruses are the nature's swiftest forms (Carter & Saunders, 2007) and this retroviral feature could not be discarded to develop some computational structure, specifically in the field of evolutionary computation, that can use them as inspiration.

This chapter aims to present a new hybrid nature-inspired metaheuristic developed based on viral structures of family Retroviridae (retroviruses) called as Retroviral Iterative Genetic Algorithm (RIGA). The source of the name comes from the junction of its features: Genetic Algorithm for behaving like a GA, Retroviral for having retroviruses structures and Iterative because occurs every single generation. Also it is made a comparison with another approach based on the junction of Genetic Algorithms and Game Theory called Social Interaction Genetic Algorithm (SIGA).

Both approaches are compared considering the biological versus social approaches applied in the context of Genetic Algorithms and also these two metaheuristics are applied in solving four widespread engineering problems find in the literature, i. e., (1) Welded Beam Design; (2) Design of a Pressure Vessel; (3) Minimization of the Weight of a Tension/Compression String; (4) Speed Reducer Design. In this way, this chapter presents the necessary fundamentals for the conception of RIGA's Algorithm, its structure and the results obtained in the simulations.

2. Biological basement: from viruses to retroviruses

Viruses are compulsory intracellular parasites with a very simple structure. Their acceptance as life forms is very controversial, since they are very different from the most simple bacteria and they have unique features, like the absence cell membrane. They don't have any known organelles and their size is several smaller, thus, the only possible way to see them is by

electronic microscopy. They are also metabolically inert unless they are inside a host cell. It is important to notice also that they cannot contain simultaneously DNA and RNA molecules (Hogg, 2005).

The viruses are formed basically by two components: the capsid, consisting of viral proteins, and the core, which contains their genetic information; the combination of these two structures is known as nucleocapsid. The main objective of viruses is to replicate themselves. To achieve this, they need to penetrate a host cell, make copies of themselves and put those copies out of the host cell.

2.1 Retroviruses

Retroviruses are the only known entities that are able to convert RNA into DNA under normal circumstances. After the adsorption and the injection of their genetic material into the host cell, the process of retro transcription takes place in the cytoplasm of the infected cell, using the viral reverse transcriptase enzyme. This process will convert a single stranded molecule of RNA (ssRNA) into a doublestranded DNA (dsDNA) molecule that is larger than the original RNA and has a high error rate, creating DNAs sequences different of which should be (Agut, 2009).

The retroviruses replication process can be described basically in follow steps (Carter & Saunders, 2007):

1. Viral recognition by the receptors present in the host cell surface
2. Penetration into the host cell
3. Reverse transcription (RNA to DNA)
4. Viral integration to the hosts genome, where it will replicate
5. Viral DNA translation (produces viral mRNA that will translated in viral proteins)
6. Viral assembling
7. Viral shedding, when the new viruses leave the host cell

One of the proteins of the virus is the integrase, which is still associated with provirus. This enzyme cuts the chromosomal DNA of the host cell and inserts the viral converted DNA, integrating the provirus into the host cell chromosome as in Fig.1. The next time this infected cell divides, the provirus will be replicated to the daughter cells (Agut, 2009). After the viral genome is integrated in the host cell genome, the virus will be totally dependent of the cellular metabolism to continue its process of transcription, translation, genome replication, viral assembling and shedding.

The concepts of Charles Darwin about reproduction and Natural Selection applied to organics forms are also applied to viruses. Even though their acceptance as an organic form is very questionable, the viruses have genes that are striving to perpetuate the species. The main mechanisms used for viral evolution are mutation, recombination, reassortment and acquisition of cellular genes (Carter & Saunders, 2007).

3. Genetic algorithms

Genetic algorithms are part of probabilistic techniques that try to find different solutions in different executions with the same parameters, and sometimes with the same population

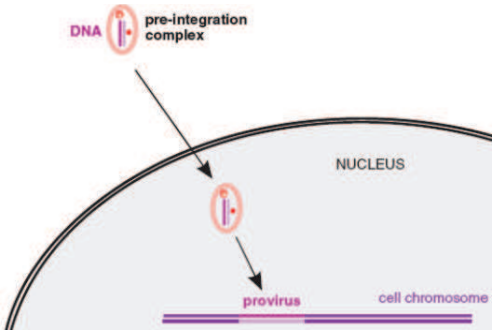


Fig. 1. Provirus creation and reverse transcription

(Goldberg, 1989). According to Haupt & Haupt (2000) some of the main advantages of GA are:

1. Optimization of discrete and continues values;
2. Simultaneous search;
3. Possibility to work with many variables;
4. Provide a set of optimum variables, instead of only specific solution.

The fundamental principle is to explore a population of chromosomes inside a search space, whose evolution depends on mutation and crossover operations, as in the natural evolutionary process. In Fig. 2 is possible to see its structure.

```
1. Begin;
2. Generate Initial Population;
3. Evaluate each indivivual of Initial Population;
4. Repeat until m generations
// Reproduction Phase
   4.1. Repeat until the number of descendents equal to desired quantity (crossover %):
        4.1.1. Select two individuals based on a selection method;
        4.1.2. Make crossover operation in the selected individuals;
        4.1.3. Make mutation operation in the descendent generated in the last step;
        4.1.4. Evaluate all individuals generated;
   4.2. Substitute old individuals by new individuals generated in reproduction;
5. The best individual of population is the problem's solution;
6. End.
```

Fig. 2. Basic structure of Genetic Algorithm

3.1 Genetic algorithm with viral infection (related works)

The viral infection is not an innovation in GA. It was discussed other times like in VEGA (Kubota et al., 1996) and GAVI (Guedes et al., 2005). In both methods is used another population composed by virus, called viral population and infection of chromosomes called transcription. In VEGA the viral population is a subset of chromosomes, created from initial hosts.

During the process of infection, a virus is selected by the same process of rank. However, as the virus has no fitness value, because it doesn't have a complete solution to be evaluated,

it is used a parameter of infection, called fitvirus (Kubota et al., 1996). This is an indicator of how well the virus has acted. After selecting the virus that will perform the infection (in a particular chromosome), the transcription is made in this moment, which consists in the modification of infected chromosome by the viral information that contains a section similar to the one represented by the infecting agent Guedes et al. (2005); Kubota et al. (1996).

When a virus is created or modified, its infectivity level is set at a fixed initial value. If the virus infects a chromosome (increase chromosome's fitness) their level of infectivity is increased by 1 (one), otherwise (if turn down the fitness) this value is reduced by 1 (one). If the virus infectivity value reaches 0 (zero) it discards its own parts and copies a part of chromosome for itself (Guedes et al., 2005).

The main difference between VEGA and GAVI is because GAVI uses viral infection as operator, ignoring the operator of mutation, and VEGA is a complete GA (Guedes et al., 2005; Kubota et al., 1996).

4. Retroviral Iterative Genetic Algorithm

The main reason for the use of viral structure in the algorithm is the fact that these viruses are associated with a source of genetic innovation, which is influenced by the rapid rate of replication and changing (Villarreal, 2009).

For biological inspiration of RIGA, the family of retroviruses was chosen. These viruses do not possess correction mechanisms to undo possible genetic mutations that occur naturally during viral multiplication, which causes a high mutation rate, arising genetically modified individuals at each generation, what is considered an important characteristic during the processing time of GA (Haupt & Haupt, 2000; Mitchell, 1999). The acquisition of cellular genes was chosen as a method to viral evolution, since it is quite common in retroviruses (Mitchell, 1999).

There are many differences between RIGA, GAVI and VEGA, some of them:

1. RIGA doesn't change any GA component, GAVI remove the mutation operation;
2. In the GAVI the worst viruses has them genetic material changed, in the RIGA they are completely changed, thereby, the viral population is constantly remade, increase the possibility of infection in chromosome population;
3. The biological basement of RIGA is very specific for the use of retroviral structure;
4. VEGA creates virus only from host chromosomes, RIGA creates virus from host chromosomes too, but, uses the main concept of a retroviruses: high mutation rate;
5. VEGA and GAVI handle a virus as a sequenced subset of chromosome, in the other hand, RIGA handle virus with dispersed information;
6. The virus lifecycle in RIGA is well-defined.

4.1 Viruses

Viruses in RIGA are structures that have the same size of a chromosome, however, with random empty spaces, because the idea is to share genetic material and avoid other population of chromosomes working in parallel. The amount of empty spaces and its positions are determined randomly. Thus for a problem that requires a binary representation of eight positions, some viruses have information as can seen in Fig. 3 .

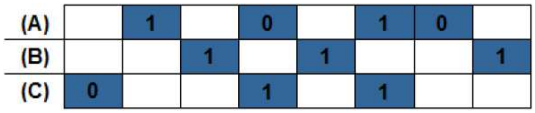


Fig. 3. Possible viruses for a binary chromosome with eight positions

4.2 Viral population and creation of new viruses

For creation of new viruses that will compose the viral population, RIGA was inspired by the natural process common in retroviruses called reverse transcription. The process consists basically of the steps showed in Fig. 4 .

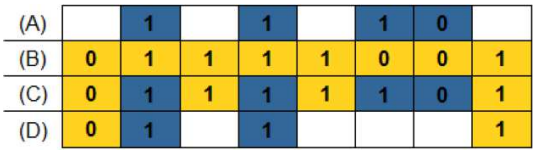


Fig. 4. Creating a new virus process (A)Random virus (B)Chromosome from population (C)Auxiliary chromosome contained the mix of genetic material (D)New virus contained genetic material from virus and host

4.3 Infection

Infection is the process of inclusion of the viral genetic material into the host chromosome, which is required a virus and a chromosome. The target chromosome will have changed their genetic material in the same positions where the genes are arranged on the virus, so all the viral genetic information, will be copied to the target chromosome, excepting the empty spaces, which will be filled by the host chromosome. The RIGA infection is represented in Fig. 5 .

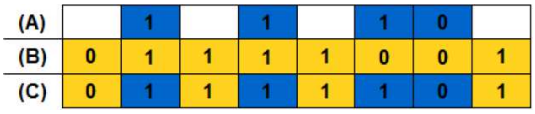


Fig. 5. Chromosome infection (A)Virus (B)Chromosome (C)Infected Chromosome

The general view about the creation and infection process is represented in Fig. 6 . In the same figure is possible verify the exchange between genetic material of all structures involved. It's possible verify in Fig. 6 as well, that the new virus (D) is made from an chromosome (B) infected by a virus (A) of viral population. The auxiliary chromosome (C) is made to be template to virus (D).

The infection process depends exclusively on one single factor: increasing chromosomal fitness. The infection is successful when an infected chromosome has an increase on its fitness and unsuccessful when it has a decrease on its fitness. This is an important factor because it determines which viruses will infect the next generation. The viruses with less infection rates will be extinct. For RIGA it is important to restrict the infection only when there is an increase in the chromosomal fitness, because, if any infection was considered, good chromosomes could turn into bad ones. Thus, only the successful infections are important to RIGA.

(A)		1		1		1	0	
(B)	0	1	1	1	1	0	0	1
(C)	0	1	1	1	1	1	0	1
(D)	0	1		1				1
(E)	1	0	0	0	1	1	1	0
(F)	0	1	0	1	1	1	1	1

Fig. 6. Creating a new virus and infection process (A) Random Virus (B) Chromosome from population (C) Auxiliary chromosome contained the mix of genetic material (D) New virus contained genetic material from virus and host (E) Target chromosome (F) Infected chromosome

4.4 Parameters

The RIGA uses the same parameters from classic GA (number of individuals, rate of mutation, crossover and elitism and the type of selection and crossover). However, to apply the concepts of viral infection by retroviruses, some other paramenters are necessary and they are:

1. Infection Population rate: the rate of chromosomes that will be infected;
2. Viral Elitism rate: the rate of viruses that will be kept in the next viral population;
3. Number of Viruses: the number of viruses of viral population;
4. Weakest Infection: this parameter forces the infection of weakest chromosome;
5. Single Infection: this parameter forces a unique infection per chromosome;
6. Internal Infection rate: this parameter indicates the maximum percentage of genetic material from any chromosome that will form a new virus.

4.5 The RIGA algorithm

The algorithm of RIGA method uses the basic structure of Genetic Algorithm presented in Fig. 2 and just extend it as shown in Fig. 7 in the way to include some new steps described as *Virus Application*. In next section the Social Interaction Genetic Algorihm is presented and then compared with RIGA.

5. Social Interaction Genetic Algorithm

The Social Interaction Genetic Algorithm is based on the fundaments of Genetic Algorithms and Game Theory where an individual can take your fitness value changed during the evolutionary process, in other words during the execution of GA. With this option, individuals can now increase your chances to survive and produce offspring, through the struggle of the games in order to maximize their individual gains.

5.1 Basic structure

In order to enable individuals to increase their fitness throughout the implementation process, was inserted a new step called *Social Interaction Phase* in the basic structure of a Genetic Algorithm shown in Figure 2. This new phase was placed before the *Reproductive Phase* and the individuals in it are exposed to an environment, which is nothing more than a strategic

```
1. Begin;
2. Generate Initial Population;
3. Evaluate each indivivual of Initial Population;
4. Repeat until m generations
// Reproduction Phase
   4.1. Repeat until the number of descendents equal to desired quantity (crossover %):
        4.1.1. Select two individuals based on a selection method;
        4.1.2. Make crossover operation in the selected individuals;
        4.1.3. Make mutation operation in the descendent generated in the last step;
        // Viral Application
        4.1.4. If it´s the first time, generate a random viral population;
        4.1.5. Generate new virus based on population;
        4.1.6. Infect individuals:
               a. Infect each chosen individual with each existent virus;
               b. For each successful infection, increment 1 to viral fitness otherwise
                  decrement 1;
               c. Check the virus with the highest level infection rate and keep
                  according to viral elitism rate;
        4.1.7. Evaluate all individuals generated;
   4.2. Substitute old individuals by new individuals generated in reproduction;
5. The best individual of population is the problem's solution;
6. End.
```

Fig. 7. RIGA Algorithm

game, where they have the opportunity to fight for their existence for some period of time. This new approach can be seen in Fig. 8 and the new phase can be expressed by three steps: (1) randomly select two individuals; (2) obtain the behavior of each individual, from their behavioral strategies; (3) change their adaptability (fitness value) based on the behavior adopted and the paytable of the game. Furthermore, this structure allows the use of any type of game, according to their paytable, and also any selection methods, such as roulette and tournament.

```
1. Begin;
2. Generate Initial Population;
3. Repeat until m generations
   // Social Interaction Phase
   3.1. Repeat until d contests
        3.1.1. Random select two individuals;
        3.1.2. Repeat until r rounds
               a. Obtain the behavior of each indivivuals based on their behavior strategy;
               b. Alter their fitness based on their behaviors and the payoff function of
                  the game;
   3.2. Evaluate each individual of population;
 // Reproduction Phase
   4.2. Repeat until the descendents number is equal to the desired quantity (crossover %)
        4.2.1. Select two individuals based on a selection method;
        4.2.2. Make crossover operation in the selected individuals;
        4.2.3. Make mutation operation in the descendent generated in the last step;
        4.2.4. Evaluate all individuals generated;
   4.3. Substitute old individuals by new individuals generated in reproduction;
5. The best individual of population is the problem's solution;
6. End.
```

Fig. 8. Structure of SIGA Algorithm

5.2 Implementation process

Based on the structure shown in Figure 8, the aspects related to implementation of SIGA that differ from the GA are: (1) individuals and behavioral strategies, and (2) calculating the value of fitness.

In the classical approach of the GA, individuals are represented by only one chromosome, which contains information about the problem to be solved. In the case of SIGA, there is the need to give individuals the ability to behave strategically. Thus, each now has more than one chromosome, which is responsible for the genetic code related to its strategy. This is randomly generated in the step of generating the initial population and is transmitted to offspring, where each child receives directly the strategy of each parent, and later, this mutated into one of two of their genes.

Thus, the behavioral strategies have been genetically encoded, as can be seen in the Table 1, through a chromosome from two positions and an alphabet composed of a ternary system, containing the values 0, 1 and 2. This system is sufficient to encode all four strategies. They are: *ALL-D*, *ALL-C*, *TFT* (Tit-For-Tat) and *Random*, with a distribution equal to 3:3:2:1, respectively, in the initial population.

Genotype	Chromossome	Phenotype
B^hB^h	00	ALL-D
B^hB^d	01	TFT
B^hb	02	ALL-D
B^dB^h	10	TFT
B^dB^d	11	ALL-C
B^db	12	ALL-C
B^hb	20	ALL-D
B^db	21	ALL-C
b b	22	Random

Table 1. Genetic Coding of Behavioral Strategies

This distribution can be observed by the amount of genetic codings for each strategy. Besides that the notation for the genetic gene is represented by the letter B (Behavior) and the dominance relationship between alleles is given as follows $B^h = B^d > b$.

The calculation of fitness in the context of the classical theory of Genetic Algorithms is based on the solutiont's genotype of an individual ($fitness_{Solution}$) and considering in the case of SIGA that an individual has two chromossomes, there is a need to change the way to calculate it.

Thus, besides having a fitness value related to the solution of the problem, individuals are given a fitness value based on the sum of the gains achieved in the stage of social interaction ($fitness_{Strategy}$), according to the Eq. 1.

$$fitness_{Total} = \alpha(fitness_{Solution}) + \beta(fitness_{Strategy}) \quad (1)$$

The total fitness of an individual ($fitness_{Total}$) is calculated from the weighted sum of fitness values of the solution($fitness_{Solution}$) and the gains made in the disputes of the games, through strategic behavior ($fitness_{Strategy}$).

It is worth mentioning that for the category of problems used here, Constraint Global Optimization Problems, in addition to calculating the fitness, the number of violations of

restrictions on each individual is taken into consideration in the selection process. Thus, there is the need to minimize priority violations and subsequently the value of $fitness_{Total}$. Moreover, the approach of SIGA does not take into account the implementation of the fee value of fitness, due to the number of violated constraints, commonly found in other approaches available in the literature, such as in (Coello, 2002).

5.3 Simulation of social interaction phase

In order to illustrate the functioning of the social interaction phase a simulation is shown in Figure 3. The population is composed of four individuals (id, fitness value and strategy behavior), the values of T, R, P and S are respectively 30, 25, 15 and 10 and, at each iteration are held four rounds of disputes the Iterative Prisoner's Dilemma game.

It is interesting to note that in the first interation, for example, the individuals 2 and 3 were selected. In each round in the dispute, the second individual always betrays while the third individual cooperates in the first round and then acts the same way as his opponent in the previous round, ie, he also betrayed only. In each round are assigned values of gain and therefore this change influence the outcome of a tournament selection operation.

Moreover, it is important to note that first is performed the social interation phase and after the tournament selection method. This selection method was used for testing. Thus, two individuals are selected randomly and checked their fitness values and the number of constraintt's violations. Then there are the following:

1. if both do not violate any constraint, then the lowest fitness wins the tournament;
2. if one did not have and the other did have violations, then the individual without violations wins;
3. if both violate constraints wins the one with the least amount of violations;
4. if they violate the same amount of constraints, then the smaller fitness gains.

At the bottom of Fig. 9, it is possible to see the change on the graphic configuration of winners in each one of the games through the partial results of the tournaments. Each individual is represented by a different texture. It is possibile to observe the influence of social interaction phase, which allows less fit individuals to evolve and be selected for crossover operation, as is the case of individual #1 who was less able than individual #4 and became more fit.

6. Evaluation of RIGA and SIGA

In the way to evaluate RIGA and SIGA approaches, they were applied to four widespread engineering problems in literature and they are also well-known as *Constraint Global Optimization Problems*.

6.1 Problem 1: Welded beam design

According to Rao (1996) this problem aims to minimize the production cost of a Welded Beam subject to some constraints on shear stress, bending stress in the beam, buckling load on the bar, end deflection of the beam and side constraints. There are four design variables: (1) thickness of the weld (h); (2) width of the beam (t); (3) thickness of the beam (b); and, (4) length of welded joint (l), as shown in Fig. 11.

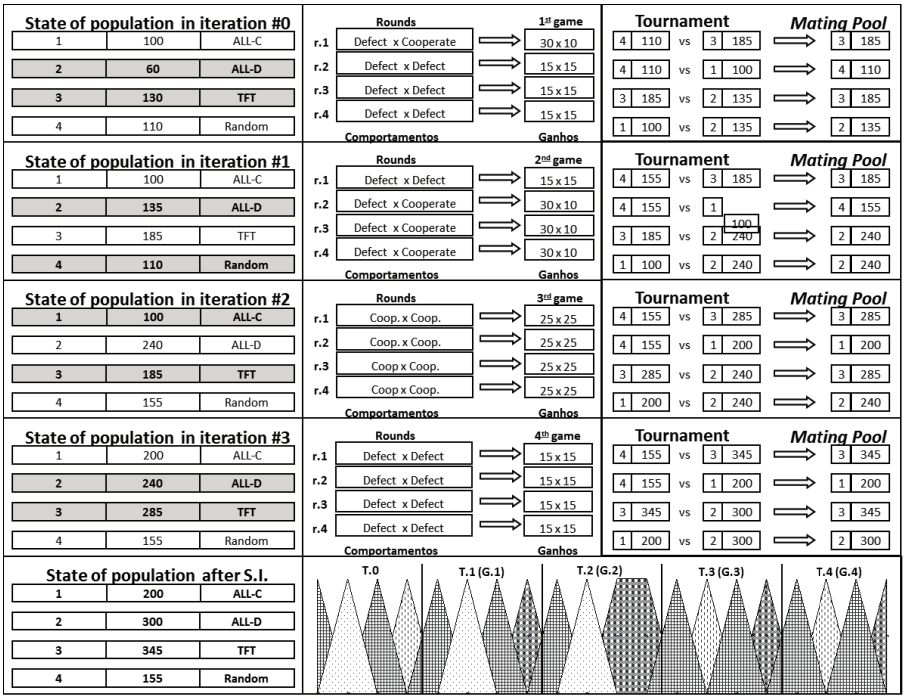


Fig. 9. Simulation of Social Interaction Phase

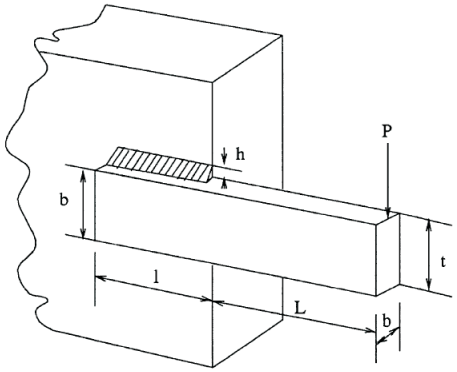


Fig. 10. Welded Beam Design (Alfares & Esat, 2007)

This problem can be stated as follows:

$$Minimize\ f(X) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14.0 + x_2) \tag{2}$$

Subject to:

$$g_1 = \tau - \tau_{max} \leq 0 \tag{3}$$

$$g_2 = \sigma - \sigma_{max} \leq 0 \tag{4}$$

$$g_3 = h - b \leq 0 \tag{5}$$

$$g_4 = 0.10471h^2 + 0.04811tb(14.0 + l) - 5.0 \leq 0 \tag{6}$$

$$g_5 = 0.125 - h \leq 0 \tag{7}$$

$$g_6 = \delta - \delta_{max} \leq 0 \tag{8}$$

$$g_7 = P - P_c \leq 0 \tag{9}$$

Where:

$$\tau = \sqrt{\left(\tau'\right)^2 + 2\tau'\tau''\frac{1}{2R} + \left(\tau''\right)^2} \tag{10}$$

$$\tau' = \frac{P}{\sqrt{2}hl} \tag{11}$$

$$\tau'' = \frac{MR}{J} \tag{12}$$

$$M = P\left(L + \frac{1}{2}\right) \tag{13}$$

$$R = \sqrt{\frac{t^2}{4} + \left(\frac{h+t}{2}\right)^2} \tag{14}$$

$$J = 2\left\{\sqrt{2}hl\left[\frac{t^2}{12} + \left(\frac{h+t}{2}\right)^2\right]\right\} \tag{15}$$

$$\sigma = \frac{6PL}{bt^2} \tag{16}$$

$$\delta = \frac{4PL^3}{Eb^3} \tag{17}$$

$$P_c = \frac{4.013E\sqrt{\frac{t^2b^6}{36}}}{L^2} + \left(1 - \frac{t}{2L}\sqrt{\frac{E}{4G}}\right) \tag{18}$$

Besides that, the following values were considered: $P = 6000lb$, $L = 14in$, $E = 30x10^6psi$, $G = 12x10^6psi$, $\tau_{max} = 13600psi$, $\sigma_{max} = 30000psi$, $\delta_{max} = 0.25in$ and variable ranges: $0.1 \leq$ h ≤ 2.0; $0.1 \leq$ l ≤ 10.0; $0.1 \leq$ t ≤ 10.0; $0.1 \leq$ b $\leq$ s 2.0.

Both algorithms, RIGA and SIGA, were executed 30 times for different parameters configurations and in this way it was established a set of parameters with the best results. For SIGA was applied the following values for parameters: {selection method = tourament; tournament size = 2; number of generations = 2000; population size = 200; crossover tax = 85%; mutation tax = 10%; alpha = 1.0; beta = 1.0; games = 100; rounds = 10; R = 5; T = 3; P = 1; S = 0}. And for RIGA was applied the following values for parameters: {selection method = tourament; tournament size = 2; number of generations = 2000; population size = 100; crossover tax = 85%; mutation tax = 10%; Infection Population Rate: 50%; Number of Viruses: 100}.

The best result out of ten plays, one for each of the algorithms, with the parameter set can be seen in Table 2.

Comparing the results of RIGA and SIGA with others results from literature, it is possible to see that they got the best two results, but SIGA was just a little better than RIGA.

Variables	RIGA	SIGA	He & Wang (2007)	Coello & Montes (2002)	Coello (2000)	Deb (1991)
$x1(h)$	0.171937	0.171937	0.202369	0.205986	0.208800	0.248900
$x2(l)$	4.131627	4.122129	3.544214	3.471328	3.420500	6.173000
$x3(t)$	9.587429	9.587429	9.048210	9.020224	8.997500	8.178900
$x4(b)$	0.183010	0.183010	0.205723	0.206480	0.210000	0.253300
$g1(x)$	-32.388778	-8.067400	-12.839796	-0.074092	-0.337812	-5758.603777
$g2(x)$	-39.336830	-39.336800	-1.247467	-0.266227	-353.902604	-255.576901
$g3(x)$	-0.011073	-0.011070	-0.001498	-0.000495	-0.001200	-0.004400
$g4(x)$	-3.466349	-3.467150	-3.429347	-3.430043	-3.411865	-2.982866
$g5(x)$	-0.236389	-0.236390	-0.079381	-0.080986	-0.083800	-0.123900
$g6(x)$	-16.024295	-16.024300	-0.235536	-0.235514	-0.235649	-0.234160
$g7(x)$	-0.046937	-0.046940	-11.681355	-58.666440	-363.232384	-4465.270928
$f(x)$	1.665485	1.664373	1.728024	1.728226	1.748309	2.433116

Table 2. Comparative results for Welded Beam Design

6.2 Problem 2: Design of a pressure vessel

The aim of the problem is to minimize the total cost of designing a pressure vessel, including the cost of the material, forming and welding.A cylindrical vessel is capped at both ends by hemispherical heads as shown in Fig. 11. There are four design variables: (1) thickness of the shell (T_s, x_1); (2) thickness of the head (T_h, x_2), inner radius (R, x_3) and the length of the cylindrical section of the vessel (L, x_4), not including the head. Among the four variables, T_s e T_h are integer multiples of 0.0625in, that are the available thickness of rolled steel plates and R and L are continuous variables.

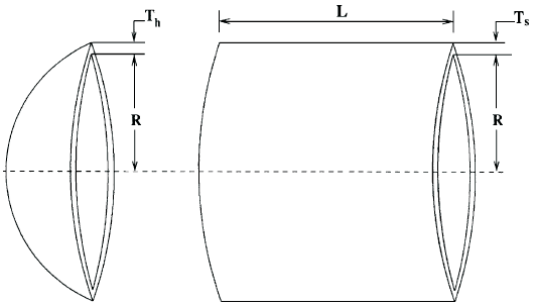


Fig. 11. Design of a Pressure Vessel (He & Wang, 2007)

This problem can be stated as follows:

$$Minimize\ f(X) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3 \tag{19}$$

Subject to:

$$g_1 = -x_1 + 0.0193x_3 \leq 0 \tag{20}$$

$$g_2 = -x_2 + 0.00954x_3 \leq 0 \tag{21}$$

$$g_3 = -\pi x_3^2x_4 - \frac{4}{3}\pi x_3^3 + 1296000 \leq 0 \tag{22}$$

$$g_4 = x_4 - 240 \leq 0 \tag{23}$$

Variables	RIGA	SIGA	He & Wang (2007)	Coello & Montes (2002)	Deb (1997)	Kannan & Kramer (1994)
$x1(Ts)$	0.812500	0.812500	0.812500	0.812500	0.937500	1.125000
$x2(Th)$	0.437500	0.437500	0.437500	0.437500	0.500000	0.625000
$x3(R)$	41.960321	42.092732	42.091266	42.097398	48.329000	58.291000
$x4(L)$	178.379290	176.947780	176.746500	176.654050	112.679000	43.690000
$g1(x)$	-0.002666	-0.000110	-0.000139	-0.000020	-0.004750	0.000016
$g2(x)$	-0.037199	-0.035935	-0.035949	-0.035891	-0.038941	-0.068904
$g3(x)$	-130.284087	-1337.994634	-116.382700	-27.886075	-3652.876838	-21.220104
$g4(x)$	-61.620710	-63.052220	-63.253500	-63.345953	-127.321000	-196.310000
$f(x)$	6077.156337	6066.029360	6061.077700	6059.946300	6410.381100	7.198.042800

Table 3. Comparative Results for Design of a Pressure Vessel

Besides that, the following variable ranges were considered: 1x0.0625 $\leq x_1 \leq$ 99x0.0625; 1x0.0625 $\leq x_2 \leq$ 99x0.0625; 10 $\leq x_3 \leq$ 200; 10 $\leq x_4 \leq$ 200.

Both algorithms, RIGA and SIGA, were executed 50 times for different parameters configurations and in this way it was established a set of parameters with the best results. For SIGA was applied the following values for parameters: {selection method = tourament; tournament size = 2; number of generations = 2000; population size = 200; crossover tax = 85%; mutation tax = 20%; alpha = 1.0; beta = 1.0; games = 100; rounds = 10; R = 5; T = 3; P = 1; S = 0}. And for RIGA was applied the following values for parameters: {selection method = tourament; tournament size = 2; number of generations = 2000; population size = 100; crossover tax = 85%; mutation tax = 20%; Infection Population Rate: 50%; Number of Viruses: 100}.

The best result out of ten plays, one for each of the algorithms, with the parameter set can be seen in Table 3.

Comparing the results of RIGA and SIGA with others results from literature, it is possible to see that they got just the 4^{th} and 3^{rd} best results, but SIGA again was just a little better than RIGA.

6.3 Problem 3: Minimization of the weight of a tension/compression string

This problem aims to minimize the weight of a tension/compression string subject to constraints on minimum deflection, shear stress, surge frequency, limits on outside diameter and on design variables, according to (Arora, 1989) e (Belegundu, 1982). The design variables are: (1) the wire diameter (d, x_1); (2) the mean coil diameter (D, x_2); and, (3) the number of active coils (P, x_3), as shown in Fig. 12.

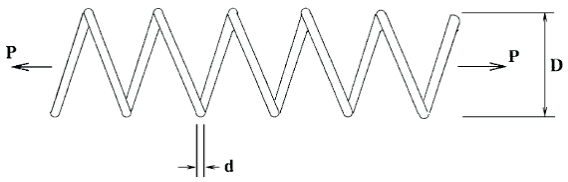


Fig. 12. Minimization of the Weight of a Tension/Compression String (He & Wang, 2007)

Variables	RIGA	SIGA	He & Wang (2007)	Coello & Montes (2002)	Arora (1989)	Belegundu (1982)
$x1(d)$	0.050021	0.050180	0.051728	0.051989	0.053396	0.050000
$x2(D)$	0.282322	0.279604	0.357644	0.363965	0.399180	0.315900
$x3(P)$	2.000075	2.087959	11.244543	10.890522	9.185400	14.250000
$g1(x)$	-0.000290	-0.002840	-0.000845	-0.000013	0.000019	-0.000014
$g2(x)$	-0.235797	-0.249450	-0.0000126	-0.000021	-0.000018	-0.003782
$g3(x)$	-43.106825	-42.176000	-4.051300	-4.061338	-4.123832	-3.938302
$g4(x)$	-0.778519	-0.780140	-0.727090	-0.722698	-0.698283	-0.756067
$f(x)$	0.002824	0.002878	0.0126747	0.0126810	0.0127303	0.0128334

Table 4. Comparative results for Minimization of the Weight of a Tension/Compression String

This problem can be stated as follows:

$$Minimize\ f(X) = (x_3 + 2)\, x_2 x_1^2 \tag{24}$$

Subject to:

$$g_1 = 1 - \frac{x_2^3 x_3}{71785 x_1^4} \leq 0 \tag{25}$$

$$g_2 = \frac{4x_2^2 - x_1 x_2}{12566\left(x_2 x_1^3 - x_1^4\right)} + \frac{1}{5108 x_1^2} - 1 \leq 0 \tag{26}$$

$$g_3 = 1 - \frac{140.45 x_1}{x_2^2 x_3} \leq 0 \tag{27}$$

$$g_4 = \frac{x_1 - x_2}{1.5} - 1 \leq 0 \tag{28}$$

Both algorithms, RIGA and SIGA, were executed 20 times for different parameters configurations and in this way it was established a set of parameters with the best results. For SIGA was applied the following values for parameters: {selection method = tourament; tournament size = 2; number of generations = 2000; population size = 200; crossover tax = 85%; mutation tax = 20%; alpha = 1.0; beta = 1.0; games = 100; rounds = 10; R = 5; T = 3; P = 1; S = 0}. And for RIGA was applied the following values for parameters: {selection method = tourament; tournament size = 2; number of generations = 2000; population size = 100; crossover tax = 85%; mutation tax = 20%; Infection Population Rate: 50%; Number of Viruses: 100}.

The best result out of ten plays, one for each of the algorithms, with the parameter set can be seen in Table 4.

Comparing the results of RIGA and SIGA with others results from literature, it is possible to see that they got the two best results, but this time RIGA was better than SIGA, just a little bit. Also, they results are much better than the others.

6.4 Problem 4: Speed reducer design

In Fig. 13 is possible to see the design of a speed reducer where its weight has to be minimized subject to constraints on bending stress of the gear teeth, surface stress, transverse deflections of the shafts and stresses in the shaft. The design variables are: (1) face width (x_1); (2) module

of teeth (x_2); (3) number of teeth on pinion (x_3); (4) length of the first shaft between bearings (x_4); (5) length of the second shaft between bearings (x_5); (6) diameter of the first shaft (x_6); and, (7) diameter of the second shaft (x_7).

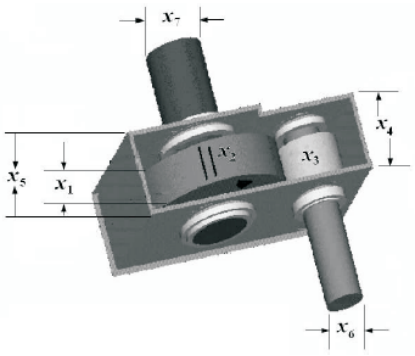


Fig. 13. Speed Reducer Design (Brajevic et al., 2010)

This problem can be stated as follows:

$$\begin{aligned} \textit{Minimizar } f(X) = {} & 0.7854x_1x_2^2\left(3.3333x_3^2 + 14.9334x_3 - 43.0934\right) \\ & -1.508x_1\left(x_6^2 + x_7^2\right) + 7.4777\left(x_6^2 + x_7^2\right) \\ & +0.78054\left(x_4x_6^2 + x_5x_7^2\right) \end{aligned} \tag{29}$$

Subject to:

$$g_1 = \frac{27}{x_1x_2^2x_3} - 1 \leq 0 \tag{30}$$

$$g_2 = \frac{397.5}{x_1x_2^2x_3} - 1 \leq 0 \tag{31}$$

$$g_3 = \frac{1.93x_4^3}{x_2x_3x_6^3} - 1 \leq 0 \tag{32}$$

$$g_4 = \frac{1.93x_5^3}{x_2x_3x_7^3} - 1 \leq 0 \tag{33}$$

$$g_5 = \frac{1.0}{110x_6^3}\sqrt{\left(\frac{750.0x_4}{x_2x_3}\right)^2 + 16.9 \times 10^6} - 1 \leq 0 \tag{34}$$

$$g_6 = \frac{1.0}{85x_7^3}\sqrt{\left(\frac{750.0x_5}{x_2x_3}\right)^2 + 157.5 \times 10^6} - 1 \leq 0 \tag{35}$$

$$g_7 = \frac{x_2x_3}{40} - 1 \leq 0 \tag{36}$$

$$g_8 = \frac{5x_2}{x_1} - 1 \leq 0 \tag{37}$$

$$g_9 = \frac{x_1}{12x_2} - 1 \leq 0 \tag{38}$$

Variables	RIGA	SIGA	Brajevic et al. (2010)	Canigna & Esquivel (2008)
$x1$	3.500098	3.500459	3.500000	3.500000
$x2$	0.700000	0.700020	0.700000	0.700000
$x3$	17.000337	17.005030	17.000000	17.000000
$x4$	7.300075	7.300251	7.300000	7.300000
$x5$	7.800014	7.800195	7.800000	7.800000
$x6$	2.900055	2.900041	3.350215	3.350214
$x7$	5.286690	5.286863	5.286683	5.286683
$g1(x)$	-0.073960	-0.074364	-0.073915	-0.073915
$g2(x)$	-0.198053	-0.198624	-0.197996	-0.197998
$g3(x)$	-0.108012	-0.108202	-0.499172	-0.499172
$g4(x)$	-0.901474	-0.901443	-0.901471	-0.901471
$g5(x)$	-1.000000	-1.000000	-2.220E-16	0.000000
$g6(x)$	-0.000004	-0.000102	-3.331E-16	-5.000E-16
$g7(x)$	-0.702494	-0.702403	-0.702500	-0.702500
$g8(x)$	-0.000028	-0.000103	0.000000	-1.000E-16
$g9(x)$	-0.795828	-0.795801	-0.583333	-0.583333
$g10(x)$	-0.143833	-0.143857	-0.051326	-0.051325
$g11(x)$	-0.010853	-0.011074	-0.010852	-0.010852
$f(x)$	2896.372448	2897.531422	2996.348165	2996.348165

Table 5. Comparative results for Speed Reducer Design

$$g_{10} = \frac{1.5x_6 + 1.9}{x_4} - 1 \leq 0 \tag{39}$$

$$g_{11} = \frac{1.1x_7 + 1.9}{x_5} - 1 \leq 0 \tag{40}$$

Besides that, the following variable ranges were considered: $2.6 \leq x_1 \leq 3.6$; $0.7 \leq x_2 \leq 0.8$; $17 \leq x_3 \leq 28$; $7.3 \leq x_4 \leq 8.3$; $7.3 \leq x_5 \leq 8.3$; $2.9 \leq x_6 \leq 5.0$; $7.3 \leq x_7 \leq 5.5$.

Both algorithms, RIGA and SIGA, were executed 50 times for different parameters configurations and in this way it was established a set of parameters with the best results. For SIGA was applied the following values for parameters: {selection method = tourament; tournament size = 2; number of generations = 2000; population size = 200; crossover tax = 85%; mutation tax = 20%; alpha = 1.0; beta = 1.0; games = 100; rounds = 10; R = 5; T = 3; P = 1; S = 0}. And for RIGA was applied the following values for parameters: {selection method = tourament; tournament size = 2; number of generations = 2000; population size = 100; crossover tax = 85%; mutation tax = 20%; Infection Population Rate: 50%; Number of Viruses: 100}.

The best result out of ten plays, one for each of the algorithms, with the parameter set can be seen in Table 5.

Comparing the results of RIGA and SIGA with others results from literature, it is possible to see that they got the two best results, and RIGA again was better than SIGA.

7. Conclusion

This chapter presents a new hybrid nature-inspired metaheuristic called Retroviral Iterative Genetic Algorithm (RIGA) applied in the resolution of four well-known engineering problems in literature. For its design were based on the study of Retroviruses because they are very

effective to perpetuate themselves. To establish a comparison at different levels of inspiration in nature, it has been presented the Social Interaction Genetic Algorithm (SIGA), also applied to the same four problems. Besides that some other methods results were used for comparison too. The pratical results show that RIGA is a promising approach and has to be considered in the resolution of Constraint Global Optimization Problems.

8. Acknowledgments

This work has the academic support of Natural Computation Laboratory of the University Center of Para State (LCN-CESUPA) and the Technological Institute of Federal University of Para (ITEC-UFPA).

9. References

Agut, A. (2009). Um sistema estratégio de reprodução, *Scientific American Brasil* pp. 14–19.

Alfares, F. S. & Esat, I. I. (2007). Real-coded quantum inspired evolution algorithm applied to engineering optimization problems, *Proceedings of the IEEE Second International Symposium on Leveraging Applications and Formal Methods, Verification and Validation*, pp. 169–176.

Arora, J. S. (1989). *Introduction to Optimum Design*, McGraw-Hill, New York.

Belegundu, A. D. (1982). A study of mathematical programming methods for structural optimization, *Technical report*, Department of Civil and Environmental Engineering. University of Iowa, Iowa, USA.

Brajevic, I., Tuba, M. & Subotic, M. (2010). Improved artificial bee colony algorithm for constrained problems, *Proceedings of the 11th WSEAS International Conference on Neural Networks, Fuzzy Systems and Evolutionary Computing*, pp. 185–190.

Canigna, L. C. & Esquivel, S. C. (2008). Solving engineering optimization problems with the simple constrained particle swarm optimizer, *Informatica* 32: 319 – 326.

Carter, J. B. & Saunders, V. A. (2007). *Virology: Principles and Applications*, John wiley & Sons Ltd.

Coello, C. A. C. (2000). Use of a self-adaptive penalty approach for engineering optimization problems, *Computers in Industry* 41: 113–127.

Coello, C. A. C. (2002). Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art, *Computer Methods in Applied Mechanics and Engineering* 191(11-12): 1245–1287.

Coello, C. A. C. & Montes, E. M. (2002). Constraint-handling in genetic algorithms through the use of dominance-based tournament selection, *Advanced Engineering Informatics* 16(11-12): 193–203.

Deb, K. (1991). Optimal design of a welded beam via genetic algorithms, *AIAA Journal* 29 (11): 2013–2015.

Deb, K. (1997). Geneas: a robust optimal design technique for mechanical component design, *in* D. Dasgupta & Z. Michalewicz (eds), *Evolutionary Algorithms in Engineering Applications*, Springer, pp. 497–514.

Goldberg, D. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison Wesley Longman, Inc.

Guedes, A., Leite, J. & Aloise, D. (2005). Um algoritmo genético com infecção viral para o problema do caixeiro viajante.

Haupt, R. L. & Haupt, S. E. (2000). *Pratical Genetic Algorithms*, John Wiley & Sons, Inc.

He, Q. & Wang, L. (2007). An effective co-evolutionary particle swarm optimization for constrained engineering design problem, *Engineering Applications of Artificial Intelligence, Elsevier* 20 (1): 89–99.

Hogg, S. (2005). *Essential Microbiology*, John Wiley & Sons, Inc.

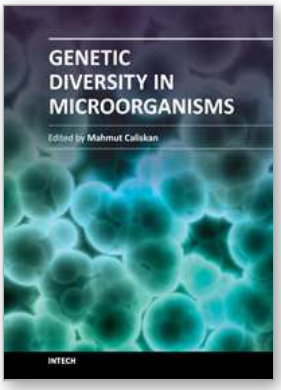
Kannan, B. K. & Kramer, S. N. (1994). An augmented lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design, *Transactions of the ASME Journal of Mechanical Design* 116: 318–320.

Kubota, N., Fukuda, T. & Shimojima, K. (1996). Virus-evolutionary genetic algorithm for a self-organizing manufacturing system, *Computers & Industrial Engineering* 30: 1015–1026.

Mitchell, M. (1999). *An Introduction to Genetic Algorithms*, MIT Press.

Rao, S. S. (1996). *Engineering Optimization*, Wiley, New York, USA.

Villarreal, L. (2009). Vírus são seres vivos?, *Scientific American Brasil* 28: 21–24.



Genetic Diversity in Microorganisms

Edited by Prof. Mahmut Caliskan

ISBN 978-953-51-0064-5
Hard cover, 382 pages
Publisher InTech
Published online 24, February, 2012
Published in print edition February, 2012

Genetic Diversity in Microorganisms presents chapters revealing the magnitude of genetic diversity of microorganisms living in different environmental conditions. The complexity and diversity of microbial populations is by far the highest among all living organisms. The diversity of microbial communities and their ecologic roles are being explored in soil, water, on plants and in animals, and in extreme environments such as the arctic deep-sea vents or high saline lakes. The increasing availability of PCR-based molecular markers allows the detailed analyses and evaluation of genetic diversity in microorganisms. The purpose of the book is to provide a glimpse into the dynamic process of genetic diversity of microorganisms by presenting the thoughts of scientists who are engaged in the generation of new ideas and techniques employed for the assessment of genetic diversity, often from very different perspectives. The book should prove useful to students, researchers, and experts in the area of microbial phylogeny, genetic diversity, and molecular biology.

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821