Tomasz Dominik Gwiazda

Genetic algorithms reference

Volume I

Crossover for single-objective numerical optimization problems

tomaszgwiazda e-books

Genetic algorithms reference
Volume I Crossover for single-objective numerical optimization problems

Published by TOMASZGWIAZDA E-BOOKS
Łomianki Poland
http://www.tomaszgwiazda.pl
http://www.tomaszgwiazda.com
e-mail: contact@tomaszgwiazda.com

Cover designed by the author

Translation for the English edition: Agata Szczepańska

ISBN 10: 83-923958-1-6
ISBN 13: 978-83-923958-1-2

Contents

Introduction

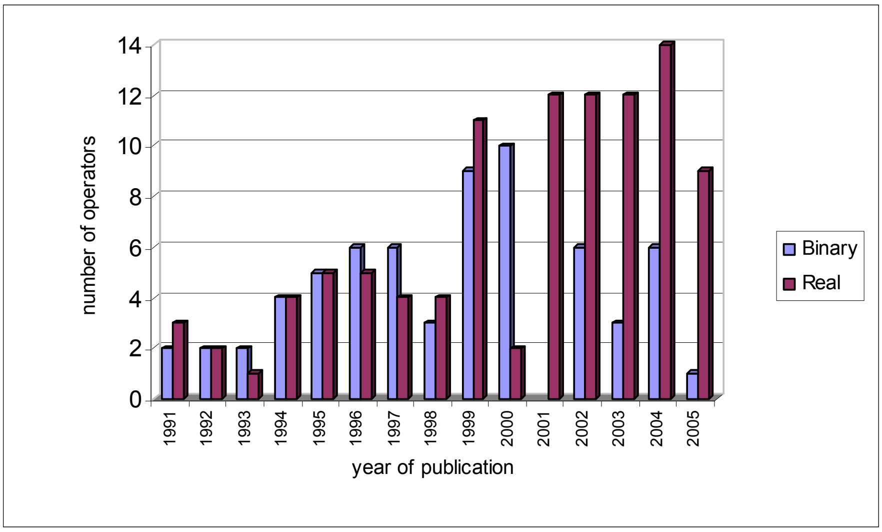
The literature on Genetic Algorithms or more widely on Evolutionary Computation is full of many excellent books and articles which are texts of introductory or review character. These text concentrate on presentation of fundamental (or most popular at a given time) methods of selection, recombination and mutation etc., so, at the same time, for obvious reasons they overlook (or only mention) most of the output from that field. The similar situation occurs as far as websites on Genetic Algorithms as well as software applying Genetic Algorithms are concerned – the scope of presented or applied methods is considerably limited. Because of that a researcher who is a beginner in that field (though not only beginner) is forced to individually dig in the source texts for less popular methods, new inspirations or answers to the question whether the method he is currently working on is new. Eventually, it is very often the case (which is proved in this book) that the *new* method which is published, is a duplication of an already existing method or is a slight and not very significant modification of it. The need for comprehensive study is, therefore, obvious and that is the motivation which led to the idea of preparation of this book. .

This book is the first of the series of reference books I am working on, with the aim to provide a possibly most comprehensive review of methods developed in the field of Genetic Algorithms. The necessity to concentrate on certain thematic areas is the result of the character of these books. The choice of those areas, even though performed arbitrarily will hopefully reflect their degree of importance and popularity. Hence, in this book which begins the whole series, an operator of the greatest importance for Genetic Algorithms will be presented i.e. crossover operator and its area of application will be single-objective numerical optimization problems. Following publications from this series will be dedicated to selection and mutation operators from the same area of application. After that I will be concentrating on multi-objective optimization problems to, in the end, cover the area of combinatorial optimization problems.

The layout of this book is the following. At the beginning I will present 11 *standard operators*, where by the term *standard* I mean those operators which most often appeared in source materials in the 80s and in the beginning of the 90s as a reference point for the newly published methods. The standard operators are presented in an abbreviated form in comparison to the other operators; therefore, some operators which could be undoubtedly qualified to the group of standard operators (e.g Arithmetical Crossover) are described in latter parts of this book in order to make a more complete presentation. The second part of this book presents 66 operators developed for the binary coded problems and the third one presents 89 operators developed for the real-coded problems. In many cases of the presented operators this division is somewhat artificial because they may be applied to solve one as well as the other class of problems. Hence, my decision to present a certain operator in the group of binary or real coded operators is based on the source texts in which, the authors usually specify the application area of that operator. The last part of this book includes the list of statistic-based operators indicating source texts as well as read-also texts..

Let Figure 1 represent quantitative summary, in which the number of operators described in this book is presented according to the years of their publication.

Figure 1 Number of operators described



In the second and third part of this book every operator is presented according to the same scheme, which is presented below:

- Keywords – are supposed to help with searching through the book and also with mutual association of the presented in it operators.
- Motivation – showing the motivation which was the base for development of a given operator. This motivation has been formulated by the authors éxplicite or has been drawn up arbitrarily.
- Source text – source text pointing to the website from which that text may be downloaded – most of these sites are free of charge.
- Read also – suggested additional texts, the subject matter of which is directly connected with the discussed operator. The choice of these texts, even though it is made arbitrarily, is based mainly on the bibliography list included in the source text or points to the texts describing further development of a given operator or other operators connected with it ideologically. Links to sites where the suggested texts may be downloaded from are also provided.
- See also – other operators that in my opinion it is worth to become acquainted with in connection with a given operator. The names of operators are in the same time hyperlinks to these pages of the book that they are discussed on.
- Algorithm – presents the discussed operator in the form of a pseudo-code, often in a couple of options. I decided to choose this form of presentation of an operator because it enables, in most cases, immediate application of that operator in practice. On the other hand I decided against usage of a specific programming language because elements appearing in the code additionally, resulting from grammar could make it difficult to understand the presented operator. A presented algorithm may often differ from its original form presented in the source text, it is often the case when the form of an operator

was closely connected with the problem for which a given operator has been developed. However, the key idea of an operator is always presented.

- Comments – commentary or description of the presented operator, depending on whether in my opinion pseudocode of an algorithm is a sufficient description or not.
- Experiment domains – problems that a given operator has been developed to solve or has been tested on, especially in consideration with standard testing functions.
- Compared to – list of other crossover operators the presented operator has been compared to (in the source text). The names of operators are in the same time hyperlinks to these pages of the book that they are discussed on.

Even though it may be disapproved of, I treat the following terms as synonyms: „recombination–crossover", „solution vector–chromosome", „gene–variable", „generation–iteration" and use them as such throughout the text. Moreover, if it is not indicated explicite to be otherwise, I use following symbols throughout the text:

t – generation (iteration) counter

M – maximum number of generations (iterations)

$P()$ – population of solution vectors (chromosomes)

$P(0)$ – initial population

$P(t)$ – current population

$P(t+1)$ – next population

p_{cross} , p_c – crossover probability

p_m – mutation probability

n – length of solution vector (chromosome)

Binary operators

$A^{(t)} = (a_1^{(t)},..,a_n^{(t)})\ \forall i\ a_i^{(t)} \in \{0,1\}$ – binary solution vector (chromosome)

$f(A^{(t)})$ – fitness of binary solution vector $A^{(t)}$

$\{A_1^{(t)},..,A_k^{(t)}\} \in P(t)$ – binary solution vectors (chromosomes)

$A_j^{(t)} = (a_{j1}^{(t)},..,a_{jn}^{(t)}) \forall i, j\ a_{ji}^{(t)} \in \{0,1\}$

$f(A_j^{(t)})$ – fitness of binary solution vector $A_j^{(t)}$

Real operators

$X^{(t)} = (x_1^{(t)},..,x_n^{(t)}) \in R^n$ – real solution vector (chromosome)

$\forall i\ x_i^l \le x_i \le x_i^u$ where:

x_i^l – lower boundary of ith variable (gene),

x_i^u – upper boundary of ith variable (gene)

$f(X^{(t)})$ – fitness of real solution vector $X^{(t)}$

$\{X_1^{(t)},..,X_k^{(t)}\} \in P(t)$ – real solution vectors (chromosomes)

$\forall i, j\ X_j^{(t)} = (x_{j1}^{(t)},..,x_{jn}^{(t)}) \in R^n, x_i^l \le x_{ji} \le x_i^u$ where:

x_i^l – lower boundary of ith variable (gene)

x_i^u – upper boundary of ith variable (gene)

$f(X_j^{(t)})$ – fitness of real solution vector $X_j^{(t)}$

Upon publishing of whole series of the planned e-books, I plan to prepare supplements once every two years which will cover two years from the date that e-book was published. I plan to send them to all interested readers. Hence, if you wish to receive such a supplement in the future please contact me by e-mail.

Standard operators

1-Point Crossover

(1-PX)

Read also

- Nomura T. (1997), *An Analysis on Crossovers for Real Number Chromosomes in an Infinite Population Size*, ATR Human Information Processing Research Laboratories, Evolutionary Systems Department

WEB: http://citeseer.ifi.unizh.ch/62577.html
http://citeseer.ist.psu.edu/62577.html

Algorithm

```
1.  select two parents $A^{(t)}$ and $B^{(t)}$ from a parent pool
2.  create two offspring $C^{(t+1)}$ and $D^{(t+1)}$ as follows:
3.  randomly choose one crossover point $cp \in \{1,...,n-1\}$
4.          for $i$ = 1 to $cp$ do
5.          $c_i^{(t+1)} = a_i^{(t)}$
6.          $d_i^{(t+1)} = b_i^{(t)}$
7.          end do
8.          for $i$ = $cp$ + 1 to $n$ do
9.          $c_i^{(t+1)} = b_i^{(t)}$
10.         $d_i^{(t+1)} = a_i^{(t)}$
11.         end do
```

k-Point Crossover

(k-PX)

Read also

- Nomura T. (1997), *An Analysis on Crossovers for Real Number Chromosomes in an Infinite Population Size*, ATR Human Information Processing Research Laboratories, Evolutionary Systems Department

WEB: http://citeseer.ifi.unizh.ch/62577.html
http://citeseer.ist.psu.edu/62577.html

Algorithm

```
1.  select two parents $A^{(t)}$ and $B^{(t)}$ from a parent pool
2.  create two offspring $C^{(t+1)}$ and $D^{(t+1)}$ as follows:
3.  randomly choose $k$ crossover points $cp_1,...,cp_k \in \{1,...,n-1\}$
4.      for $i$ = 1 to $cp_1$ do
5.      $c_i^{(t+1)} = a_i^{(t)}$
6.      $d_i^{(t+1)} = b_i^{(t)}$
7.      end do
8.  $switch$ = 0
9.      for $j$ = 2 to $k$ do
10.         if $switch$ = 0 then
11.             for $i = cp_{j-1} + 1$ to $cp_j$ do
12.             $c_i^{(t+1)} = b_i^{(t)}$
13.             $d_i^{(t+1)} = a_i^{(t)}$
14.             end do
15.         $switch$ = 1
16.         else
17.             for $i = cp_{j-1} + 1$ to $cp_j$ do
18.             $c_i^{(t+1)} = a_i^{(t)}$
19.             $d_i^{(t+1)} = b_i^{(t)}$
20.             end do
21.         $switch$ = 0
```

```
22.             end if
23.     end do
24.     if $switch = 0$ then
25.             for $i = cp_k + 1$ to $n$ do
26.             $c_i^{(t+1)} = b_i^{(t)}$
27.             $d_i^{(t+1)} = a_i^{(t)}$
28.             end do
29.     else
30.             for $i = cp_k + 1$ to $n$ do
31.             $c_i^{(t+1)} = a_i^{(t)}$
32.             $d_i^{(t+1)} = b_i^{(t)}$
33.             end do
34.     end if
```

Shuffle Crossover

(SC)

Read also

- Burkowski F.J. (1999), Shuffle Crossover and Mutual Information, in *Proceedings of the 1999 Congress on Evolutionary Computation*, pp. 1574-1580

WEB: http://intl.ieeexplore.ieee.org/xpl/abs_free.jsp?arNumber=782671

Algorithm

```
1.  select two parents $A^{(t)}$ and $B^{(t)}$ from a parent pool
2.  create two offspring $C^{(t+1)}$ and $D^{(t+1)}$ as follows:
3.  randomly shuffle (in the same way) the genes in both parents
4.  randomly choose one crossover point $cp \in \{1,...,n-1\}$
5.      for $i$ = 1 to $cp$ do
6.      $c_i^{(t+1)} = a_i^{(t)}$
7.      $d_i^{(t+1)} = b_i^{(t)}$
8.      end do
9.      for $i$ = $cp$ + 1 to $n$ do
10.     $c_i^{(t+1)} = b_i^{(t)}$
11.     $d_i^{(t+1)} = a_i^{(t)}$
12.     end do
13. unshuffle the genes in both offspring
```

Reduced Surrogate Crossover

(SC)

Algorithm

```
1.  select two parents $A^{(t)}$ and $B^{(t)}$ from a parent pool
2.  create a list $PCP = (cp_1,...,cp_l)$ of possible $l$ crossover points as follows:
3.  $l = 0$
4.      for $i$ = 1 to $n$ do
5.          if $a_i^{(t)} \neq b_i^{(t)}$ then
6.          $l = l + 1$
7.          $cp_l = i$
8.          end if
9.      end do
10.     if $l > 0$ then
11.     create two offspring $C^{(t+1)}$ and $D^{(t+1)}$ as follows:
12.     randomly choose one crossover point $cp \in PCP(cp_1,...,cp_l)$
13.         for $i$ = 1 to $cp$ do
14.         $c_i^{(t+1)} = a_i^{(t)}$
15.         $d_i^{(t+1)} = b_i^{(t)}$
16.         end do
17.         for $i = cp$ + 1 to $n$ do
18.         $c_i^{(t+1)} = b_i^{(t)}$
19.         $d_i^{(t+1)} = a_i^{(t)}$
20.         end do
21.     else
22.     do nothing
23.     end if
```

Uniform Crossover

(UX)

Read also

➢ Spears W.M., De Jong K.A. (1991), On the Virtues of Parameterized Uniform Crossover, in *Proceedings of the Fourth International Conference on Genetic Algorithms*, Morgan Kaufman, pp. 230-236
WEB: http://citeseer.ifi.unizh.ch/spears91virtues.html
http://citeseer.ist.psu.edu/spears91virtues.html

➢ Belea R., Beldiman L. (2003), A new method of gene coding for a genetic algorithm designed for parametric optimization, in *The Annals of "Dunarea De Jos"*, University of Galati, pp. 66-71
WEB: http://www.ann.ugal.ro/eeai/archives/2003.htm

➢ Cordón O., Damas S., Santamaría J. (2003), A CHC Evolutionary Algorithm for 3D Image Registration, in *IFSA 2003*, Springer-Verlag, pp. 404-411
WEB: http://sci2s.ugr.es/publications/
http://springerlink.metapress.com/openurl.asp?genre=article&issn=0302-9743&volume=2715&spage=404

➢ Cotta C., Troya J.M. (2000), Using Dynastic Exploring Recombination to Promote Diversity in Genetic Search, in Parallel Problem Solving from Nature - 6th International Conference, Springer Verlag, pp. 16-20
WEB: http://citeseer.ifi.unizh.ch/cotta00using.html
http://citeseer.ist.psu.edu/cotta00using.html

Algorithm UX

```
1.  select two parents $A^{(t)}$ and $B^{(t)}$ from a parent pool
2.  create two offspring $C^{(t+1)}$ and $D^{(t+1)}$ as follows:
3.          for $i$ = 1 to $n$ do
4.          choose a uniform random real number $u \in$<0,1>
5.                  if $u \le p_s$ then (swap bits)
6.                  $c_i^{(t+1)} = b_i^{(t)}$
7.                  $d_i^{(t+1)} = a_i^{(t)}$
8.                  else (don't swap)
9.                  $c_i^{(t+1)} = a_i^{(t)}$
10.                 $d_i^{(t+1)} = b_i^{(t)}$
11.                 end if
12.         end do
```

where:
p_s – probability of swapping, in standard form p_s = 0.5

Heuristic Uniform Crossover, Highly Disruptive Crossover

(HUX)

Read also

- ➢ Spears W.M., De Jong K.A. (1991), On the Virtues of Parameterized Uniform Crossover, in *Proceedings of the Fourth International Conference on Genetic Algorithms*, Morgan Kaufman, pp. 230-236
WEB: http://citeseer.ifi.unizh.ch/spears91virtues.html
http://citeseer.ist.psu.edu/spears91virtues.html
- ➢ Belea R., Beldiman L. (2003), A new method of gene coding for a genetic algorithm designed for parametric optimization, in *The Annals of "Dunarea De Jos"*, University of Galati, pp. 66-71
WEB: http://www.ann.ugal.ro/eeai/archives/2003.htm
- ➢ Cordón O., Damas S., Santamaría J. (2003), A CHC Evolutionary Algorithm for 3D Image Registration, in *IFSA 2003*, Springer-Verlag, pp. 404-411
WEB: http://sci2s.ugr.es/publications/
http://springerlink.metapress.com/openurl.asp?genre=article&issn=0302-9743&volume=2715&spage=404
- ➢ Cotta C., Troya J.M. (2000), Using Dynastic Exploring Recombination to Promote Diversity in Genetic Search, in Parallel Problem Solving from Nature - 6th International Conference, Springer Verlag, pp. 16-20
WEB: http://citeseer.ifi.unizh.ch/cotta00using.html
http://citeseer.ist.psu.edu/cotta00using.html

Algorithm HUX

```
1.  select two parents $A^{(t)}$ and $B^{(t)}$ from a parent pool
2.  create two offspring $C^{(t+1)}$ and $D^{(t+1)}$ as follows:
3.  $C^{(t+1)} = A^{(t)}$
4.  $D^{(t+1)} = B^{(t)}$
5.  $number\_of\_different\_genes = 0$
6.        for $i = 1$ to $n$ do
7.              if $c_i^{(t+1)} \neq d_i^{(t+1)}$ then
8.              $number\_of\_different\_genes = number\_of\_different\_genes + 1$
9.              end if
10.       end do
11.       $swap\_counter = 0$
12.       do while $swap\_counter \leq number\_of\_different\_genes/2$
13.             for $i = 1$ to $n$ do
```

```
14.                         if $c_i^{(t+1)} \neq d_i^{(t+1)}$ and $c_i^{(t+1)} \neq b_i^{(t)}$ then
15.                         choose a uniform random real number $u \in <0,1>$
16.                                 if $u \leq 0.5$ then (swap bits)
17.                                 $c_i^{(t+1)} = b_i^{(t)}$
18.                                 $d_i^{(t+1)} = a_i^{(t)}$
19.                                 $swap\_counter = swap\_counter + 1$
20.                                 end if
21.                         end if
22.                 end do
23.         loop
```

Average Crossover

(AX)

Read also

- Nomura T. (1997), *An Analysis on Crossovers for Real Number Chromosomes in an Infinite Population Size*, ATR Human Information Processing Research Laboratories, Evolutionary Systems Department
WEB: http://citeseer.ifi.unizh.ch/62577.html
http://citeseer.ist.psu.edu/62577.html
- Fernandes C., Tavares R., Munteanu C., Rosa A. (2001), Using Assortative Mating in Genetic Algorithms for Vector Quantization Problems, in *Proceedings of the 2001 ACM symposium on Applied computing* , pp. 361 - 365
WEB: http://citeseer.ist.psu.edu/fernandes01using.html
http://citeseer.ifi.unizh.ch/fernandes01using.html

Algorithm

```
1. select two parents $X^{(t)}$ and $Y^{(t)}$ from a parent pool
2. create one offspring $X^{(t+1)}$ as follows:
3.         for $i = 1$ to $n$ do
4.         $x_i^{(t+1)} = \frac{x_i^{(t)} + y_i^{(t)}}{2}$
5.         end do
```

Discrete Crossover

(DC)

Read also

- Voigt H.-M., Mühlenbein H., (1995), Cvetković D., Fuzzy recombination for the Breeder Genetic Algorithm, in *Proceedings of the Sixth International Conference on Genetic Algorithms*, Morgan Kaufman, pp. 104-111

WEB: http://citeseer.ifi.unizh.ch/voigt95fuzzy.html
http://citeseer.ist.psu.edu/voigt95fuzzy.html
http://www.amspr.gfai.de/publications_de.htm

- Mühlenbein H., Schlierkamp-Voosen D. (1993), Predictive Models for the Breeder Genetic Algorithm I. Continuous Parameter Optimization, in *Evolutionary Computation*, vol. 1, pp. 25-49

WEB: http://citeseer.ifi.unizh.ch/mtihlenbein93predictive.html
http://citeseer.ist.psu.edu/mtihlenbein93predictive.html
http://www.ais.fhg.de/~muehlen/pegasus/publications.html

Algorithm

```
1.  select two parents $X^{(t)}$ and $Y^{(t)}$ from a parent pool
2.  create one offspring $X^{(t+1)}$ as follows:
3.        for $i$ = 1 to $n$ do
4.        choose a uniform random real number $u \in <0,1>$
5.              if $u \leq 0.5$ then
6.              $x_i^{(t+1)} = x_i^{(t)}$
7.              else
8.              $x_i^{(t+1)} = y_i^{(t)}$
9.              end if
10.       end do
```

Flat Crossover

(FC)

Read also

- Herrera F., Lozano M., Verdegay J.L. (1998), Tackling Real-Coded Genetic Algorithms: Operators and Tools for Behavioural Analysis, in *Artificial Intelligence Review*, Kluwer, vol. 12, pp. 254-319

WEB: http://citeseer.ifi.unizh.ch/herrera98tackling.html
http://citeseer.ist.psu.edu/herrera98tackling.html

Algorithm

```
1. select two parents $X^{(t)}$ and $Y^{(t)}$ from a parent pool
2. create one offspring $X^{(t+1)}$ as follows:
3.         for $i$ = 1 to $n$ do
4.         choose a uniform random real number
           $\alpha \in < \min(x_i^{(t)}, y_i^{(t)}), \max(x_i^{(t)}, y_i^{(t)}) >$
5.         $x_i^{(t+1)} = \alpha$
6.         end do
```

Heuristic Crossover /Intermediate Crossover (HC/IC)

Read also

- Voigt H.-M., Mühlenbein H., (1995), Cvetković D., Fuzzy recombination for the Breeder Genetic Algorithm, in *Proceedings of the Sixth International Conference on Genetic Algorithms*, Morgan Kaufman, pp. 104-111

WEB: http://citeseer.ifi.unizh.ch/voigt95fuzzy.html
http://citeseer.ist.psu.edu/voigt95fuzzy.html
http://www.amspr.gfai.de/publications_de.htm

- Mühlenbein H., Schlierkamp-Voosen D. (1993), Predictive Models for the Breeder Genetic Algorithm I. Continuous Parameter Optimization, in *Evolutionary Computation*, vol. 1, pp. 25-49

WEB: http://citeseer.ifi.unizh.ch/mtihlenbein93predictive.html
http://citeseer.ist.psu.edu/mtihlenbein93predictive.html
http://www.ais.fhg.de/~muehlen/pegasus/publications.html

- Herrera F., Lozano M., Verdegay J.L. (1998), Tackling Real-Coded Genetic Algorithms: Operators and Tools for Behavioural Analysis, in *Artificial Intelligence Review*, Kluwer, vol. 12, pp. 254-319

WEB: http://citeseer.ifi.unizh.ch/herrera98tackling.html
http://citeseer.ist.psu.edu/herrera98tackling.html

Algorithm

```
1.  select two parents $X^{(t)}$ and $Y^{(t)}$ from a parent pool
2.  create one offspring $X^{(t+1)}$ as follows:
3.          for $i = 1$ to $n$ do
4.          assume that $x_i^{(t)} \le y_i^{(t)}$
5.          choose a uniform random real number $\alpha \in <0,1>$
6.          $x_i^{(t+1)} = x_i^{(t)} + \alpha(y_i^{(t)} - x_i^{(t)})$
7.          end do
```

Comments

Parameter α may be of constant value equal to 0.5 or may be selected by a draw from interval <0,1> (row: 5).

Blend Crossover

(BLX-α, BLX-α-β)

Read also

- Takahashi M., Kita H. (2001), A Crossover Operator Using Independent Component Analysis for Real-Coded Genetic Algorithm, in *Proceedings of the 2001 Congress on Evolutionary Computation*, pp. 643-649
WEB: http://ieeexplore.ieee.org/xpl/abs_free.jsp?arNumber=934452
- Herrera F., Lozano M., Sánchez A.M. (2003), A taxonomy for the Crossover Operator for Real-Coded Genetic Algorithms: An Experimental Study, in *International Journal of Intelligent Systems*, Wiley, vol. 18, pp. 309-338
WEB: http://www3.interscience.wiley.com
http://dx.doi.org/10.1002/int.10091

Algorithm BLX-α

```
1. select two parents $X^{(t)}$ and $Y^{(t)}$ from a parent pool
2. create two offspring $X^{(t+1)}$ and $Y^{(t+1)}$ as follows:
3.         for $i$ = 1 to $n$ do
4.         $d_i = | x_i^{(t)} - y_i^{(t)} |$
5.         choose a uniform random real number
           $u \in \left\langle \min(x_i^{(t)}, y_i^{(t)}) - \alpha d_i, \max(x_i^{(t)}, y_i^{(t)}) + \alpha d_i \right\rangle$
6.         $x_i^{(t+1)} = u$
7.         choose a uniform random real number
           $u \in \left\langle \min(x_i^{(t)}, y_i^{(t)}) - \alpha d_i, \max(x_i^{(t)}, y_i^{(t)}) + \alpha d_i \right\rangle$
8.         $y^{(t+1)} = u$
9.         end do
```

where:
α – positive real parameter

Algorithm BLX-α-β

```
1. select two parents $X^{(t)}$ and $Y^{(t)}$ from a parent pool
2. assume that $X^{(t)}$ is better than $Y^{(t)}$
3. create two offspring $X^{(t+1)}$ and $Y^{(t+1)}$ as follows:
4.         for $i$ = 1 to $n$ do
5.         $d_i = | x_i^{(t)} - y_i^{(t)} |$
```

```
6.              if $x_i^{(t)} \le y_i^{(t)}$ than
7.                  choose a uniform random real number
                    $u \in \left\langle x_i^{(t)} - \alpha d_i , y_i^{(t)} + \beta d_i \right\rangle$
8.                  $x_i^{(t+1)} = u$
9.                  choose a uniform random real number
                    $u \in \left\langle x_i^{(t)} - \alpha d_i , y_i^{(t)} + \beta d_i \right\rangle$
10.                 $y_i^{(t+1)} = u$
11.             else
12.                 choose a uniform random real number
                    $u \in \left\langle y_i^{(t)} - \beta d_i , y_i^{(t)} + \alpha d_i \right\rangle$
13.                 $x_i^{(t+1)} = u$
14.                 choose a uniform random real number
                    $u \in \left\langle y_i^{(t)} - \beta d_i , y_i^{(t)} + \alpha d_i \right\rangle$
15.                 $y_i^{(t+1)} = u$
16.         end do
```

where:
α, β – positive real parameters

Binary operators

Random Respectful Crossover

(R3)(RRC)

Keywords
schema, similarity

Motivation

- Offspring generation from a similarity set of the parents.

Source text

➢ Radcliffe N.J. (1991), Forma Analysis and Random Respectful Recombination, in *Proceedings of the Fourth International Conference on Genetic Algorithms*, pp. 222-229
WEB: http://users.breathe.com/njr/formaPapers.html
http://citeseer.ist.psu.edu/radcliffe91formal.html

Read also

➢ Watson R.A., Pollack J.B. (2000), Recombination Without Respect: Schema Combination and Disruption in Genetic Algorithm Crossover, in *Proceedings of GECCO 2000*, Morgan Kaufman, pp. 112-119
WEB: http://citeseer.ifi.unizh.ch/watson00recombination.html
http://citeseer.ist.psu.edu/watson00recombination.html

➢ Ozugur T. (2005), Hierarchical Provisioning for Cellular networks, in *IEEE Transactions on Wireless Communications*, vol. 4(2), pp. 775-791
WEB: http://ieeexplore.ieee.org/xpl/abs_free.jsp?arNumber=1413243

➢ Nomura T. (1997), *An Analysis on Crossovers for Real Number Chromosomes in an Infinite Population Size*, ATR Human Information Processing Research Laboratories, Evolutionary Systems Department
WEB: http://citeseer.ifi.unizh.ch/62577.html
http://citeseer.ist.psu.edu/62577.html

See also

- Hierarchical Crossover
- Disrespectful Crossover
- Asymmetric Two-point Crossover
- Variation of Asymmetric Two-point Crossover
- Homologous Crossover
- Schema-Based Crossover
- Adaptive Probability Crossover-4

Algorithm

```
1.  select two parents $A^{(t)}$ and $B^{(t)}$ from a parent pool
2.  create a similarity vector $S^{AB} = (s_1^{AB},...,s_n^{AB})$ as follows:
3.      for $i = 1$ to $n$ do
4.          if $a_i^{(t)} = b_i^{(t)}$ then
5.          $s_i^{AB} = a_i^{(t)}$
6.          else
7.          $s_i^{AB} = NULL$
8.          end if
9.      end do
10. create two offspring $C^{(t+1)}$ and $D^{(t+1)}$ as follows:
11.     for $i = 1$ to $n$ do
12.         if $s_i^{AB} = 1$ then
13.         $c_i^{(t+1)} = 1$
14.         $d_i^{(t+1)} = 1$
15.         else if $s_i^{AB} = 0$ then
16.         $c_i^{(t+1)} = 0$
17.         $d_i^{(t+1)} = 0$
18.         else if $s_i^{AB} = NULL$ then
19.         choose a uniform random real number $u \in <0,1>$
20.             if $u \leq 0.5$ then
21.             $c_i^{(t+1)} = 1$
22.             else
23.             $c_i^{(t+1)} = 0$
24.             end if
25.         choose a uniform random real number $u \in <0,1>$
26.             if $u \leq 0.5$ then
27.             $d_i^{(t+1)} = 1$
```

```
28.                 else
29.                     $cd_i^{(t+1)} = 0$
30.                 end if
31.             end if
32.     end do
```

Comments

- The R3 algorithm duplicates genes of parents in an offspring at every position at which they are identical (rows: 12-17). At positions where values of the parent genes are different they are determined at random (rows: 18-31).

Experiment domains

- n/a

Compared to

- n/a

Masked Crossover

(MX)

Keywords
adaptive, fitness driven crossover, schema preservation, epistasis

Motivation
- Protecting good schemata from destruction.
- Searching through the solution space in promising directions depending on fitness.

Source text

➢ Louis S.J., Rawlins G.J.E. (1991), Designer Genetic Algorithms: Genetic Algorithms in Structure Design, in *Proceedings of the Fourth International Conference on genetic Algorithms*, Morgan Kaufman, pp.53-60
WEB: http://citeseer.ifi.unizh.ch/louis91designer.html
http://citeseer.ist.psu.edu/louis91designer.html

Read also

➢ Maini H., Mehrotra K., Mohan Ch., Ranka S. (1994), Knowledge-Based Nonuniform Crossover, in *Proceedings of IEEE International Conference on Evolutionary Computation*, Orlando
WEB: http://intl.ieeexplore.ieee.org/xpl/abs_free.jsp?arNumber=350048

➢ Vekaria K., Clack C. (1999), Biases Introduced by Adaptive Recombination Operators, in *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 670-677
WEB: http://citeseer.ifi.unizh.ch/vekaria99biases.html
http://citeseer.ist.psu.edu/vekaria99biases.html

➢ Chou Ch.-H., Chen J.-N. (2000), Genetic Algorithms: initialization schemes and genes extraction, in *Proceedings of The Ninth IEEE International Conference on Fuzzy Systems*, pp. 965 - 968
WEB: http://intl.ieeexplore.ieee.org/xpl/abs_free.jsp?arNumber=839167

See also
- Knowledge-Based Nonuniform Crossover

Algorithm

```
1.  select two parents $A^{(t)}$ and $B^{(t)}$ from a parent pool
2.  create two offspring $C^{(t+1)}$ and $D^{(t+1)}$ as follows:
3.      for $i = 1$ to $n$ do
4.      $c_i^{(t+1)} = a_i^{(t)}$
5.      $d_i^{(t+1)} = b_i^{(t)}$
6.      end do
```

```
7.     for $i = 1$ to $n$ do
8.         if $p_i^B = 1$ and $p_i^A = 0$ then
9.         $c_i^{(t+1)} = b_i^{(t)}$
10.        end if
11.        if $p_i^A = 1$ and $p_i^B = 0$ then
12.        $d_i^{(t+1)} = a_i^{(t)}$
13.        end if
14.    end do
```

where:

$P^A = (p_1^A, ..., p_n^A)$ – crossover mask of the parent $A^{(t)}$, $\forall i\ p_i^A \in \{0,1\}$

$P^B = (p_1^B, ..., p_n^B)$ – crossover mask of the parent $B^{(t)}$, $\forall i\ p_i^B \in \{0,1\}$

Comments

- The MX operator uses a mask vector to determine which bits of which parent are inherited by the offspring. The first step is the duplication of the bits of the parents. The bits of the first parent are copied to the first offspring and, accordingly, of the second parent to the second offspring (rows: 3-6). In the second step, the offspring exchange bits among each other (rows: 9 and 12) at those positions where the mask vectors of the parent were equal to 1, indicated domination of that parent at that position and the mask vectors of the other parent were equal to 0 (rows: 8 and 11).
- The mask vectors are initiated in *P*(*0*) randomly. During every GA iteration, the mask vectors are inherited by each offspring from its parent. Then the mask vectors of the offspring as well as the parents undergo modification. The modification process (not described here) is based on the comparison of fitness of the offspring and the parents. If *good* offspring were created, the masks of the parents do not need to be modified and the masks of the offspring may be very similar to those of the parents. In a situation where *bad* offspring were created the masks of the parents as well as of the offspring need to be modified.

Experiment domains

- n-bit parity checker
- n-bit adder

Compared to

- 1-Point Crossover

1bit Adaptation Crossover

(1BX)

Keywords
adaptive, recombination model, combination of crossovers

Motivation

- Obtaining different trajectories of searching the solution space through simultaneous application of two operators of diametrically opposite characteristics.

Source text

➢ Spears W.M. (1992), *Adapting Crossover in Evolutionary Algorithms*, Technical Report AIC-92-025, Naval Research Laboratory, Navy Center for Applied Research on Artificial Intelligence.
WEB: http://www.aic.nrl.navy.mil/~spears/ea.html

Read also

➢ Vrajitoru D. (2004), Intra and Extra-Generation Schemes for Combining Crossover Operators, in *Proceedings of the Fifteenth Midwest Artificial Intelligence and Cognitive Science Conference MAICS 2004*, pp. 86-91
WEB: http://www.maics.us/proceedings.htm

➢ Herrera F., Lozano M., Sánchez A.M., Hybrid Crossover Operators for Real-Coded Genetic Algorithms: An Experimental Study, in *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, Springer, vol. 9(4), pp. 280-298
WEB: http://sci2s.ugr.es/publications/
http://dx.doi.org/10.1007/s00500-004-0380-9

See also

- k-Point Crossover
- Uniform Crossover
- Combined Balanced Crossover
- Adaptive Strategies of Mixing Crossovers

Algorithm

```
1. select two parents $A^{(t)}$ and $B^{(t)}$ from current population $P(t)$
2. choose a uniform random real number $u \in <0, 1>$
3.     if $a_n^{(t)} = b_n^{(t)} = 1$ then
4.     create two offspring $C^{(t+1)}$ and $D^{(t+1)}$ offspring by the 2-Point Crossover
       as follows:
5.     randomly choose two crossover points $cp_1, cp_2 \in \{1,...,n-1\}$ $(cp_1 < cp_2)$
6.         for $i$ = 1 to $cp_1$ do
```

```
7.                  $c_i^{(t+1)} = a_i^{(t)}$
8.                  $d_i^{(t+1)} = b_i^{(t)}$
9.                  end do
10.                 for $i = cp_1 + 1$ to $cp_2$ do
11.                 $c_i^{(t+1)} = b_i^{(t)}$
12.                 $d_i^{(t+1)} = a_i^{(t)}$
13.                 end do
14.                 for $i = cp_2 + 1$ to $n$ do
15.                 $c_i^{(t+1)} = a_i^{(t)}$
16.                 $d_i^{(t+1)} = b_i^{(t)}$
17.                 end do
18.         else if $a_n^{(t)} = b_n^{(t)} = 0$ then
19.         create two offspring $C^{(t+1)}$ and $D^{(t+1)}$ by the Uniform Crossover as
            follows:
20.                 for $i = 1$ to $n$ do
21.                 choose a uniform random real number $u \in <0,1>$
22.                         if $u \le p_s$ then (swap bits)
23.                         $c_i^{(t+1)} = b_i^{(t)}$
24.                         $d_i^{(t+1)} = a_i^{(t)}$
25.                         else (don’t swap)
26.                         $c_i^{(t+1)} = a_i^{(t)}$
27.                         $d_i^{(t+1)} = b_i^{(t)}$
28.                         end if
29.                 end do
30.         else
31.         choose a uniform random real number $u \in <0, 1>$
32.                 if $u < 0.5$ then
33.                 create two offspring $C^{(t+1)}$ and $D^{(t+1)}$ by the Uniform Crossover
                    as follows:
```

```
34.                 for $i$ = 1 to $n$ do
35.                 choose a uniform random real number $u \in <0,1>$
36.                         if $u \le p_s$ then (swap bits)
37.                         $c_i^{(t+1)} = b_i^{(t)}$
38.                         $d_i^{(t+1)} = a_i^{(t)}$
39.                         else (don't swap)
40.                         $c_i^{(t+1)} = a_i^{(t)}$
41.                         $d_i^{(t+1)} = b_i^{(t)}$
42.                         end if
43.                 end do
44.         else
45.         create two offspring $C^{(t+1)}$ and $D^{(t+1)}$ by the 2-Point Crossover
            as follows:
46.         randomly choose two crossover points
            $cp_1, cp_2 \in \{1,...,n-1\}$ $(cp_1 < cp_2)$
47.                 for $i$ = 1 to $cp_1$ do
48.                 $c_i^{(t+1)} = a_i^{(t)}$
49.                 $d_i^{(t+1)} = b_i^{(t)}$
50.                 end do
51.                 for $i = cp_1 + 1$ to $cp_2$ do
52.                 $c_i^{(t+1)} = b_i^{(t)}$
53.                 $d_i^{(t+1)} = a_i^{(t)}$
54.                 end do
55.                 for $i = cp_2 + 1$ to $n$ do
56.                 $c_i^{(t+1)} = a_i^{(t)}$
57.                 $d_i^{(t+1)} = b_i^{(t)}$
58.                 end do
59.         end if
60.     end if
```

where:
p_s – probability of swapping, in standard form $p_s = 0.5$

Comments

- In the 1BX method the last bit of the solution vector is reserved for the code of one of the two of the applied crossover operators. Assuming that "0" corresponds with the Uniform Crossover (UX) operator and "1" corresponds with the 2-Point Crossover (2-PX) operator, the choice of one of them is made according to the rule: if the last bit of the parents is off the same value (rows: 3 and 18) then choose the operator indicated by this bit (rows: 4 and 19). Otherwise choose the operator through selection by a draw (rows: 32 and 44).
- Application of the described crossover scheme combines the choice of the operator with the solution vector. Moreover, this choice is carried out separately for each parent pair; hence this scheme is called *local adaptation. Global adaptation* version has been also presented, but as it was emphasized by the author, significantly worse results were obtained by its application.

Experiment domains

- n-peak problems

Compared to

- k-Point Crossover
- Uniform Crossover

Multivariate Crossover

(MC)

Keywords
variable-to-variable recombination

Motivation

- Effective optimization of multivariate functions.

Source text

➢ Konstam A.H., Hartley S.J., Carr W.L. (1992), Optimization in a Distributed Processing Environment using Genetic Algorithm with Multivariate Crossover, in *Proceedings of the 1992 ACM annual conference on Communications*, pp. 109-116
WEB: http://portal.acm.org/citation.cfm?id=131228

Read also

➢ Yang S.Y., Park L.-J., Park C.H., Ra J.W. (1995), A Hybrid Algorithm using Genetic Algorithm and Gradient-Based Algorithm for Iterative Microwave Inverse Scattering, in *IEEE International Conference on Evolutionary Computation*, pp.450-455
WEB: http://intl.ieeexplore.ieee.org/xpl/abs_free.jsp?arNumber=489190

➢ Deb K., Goyal M., Optimizing Engineering Designs Using a Combined Genetic Search, in *Proceedings of the Seventh International Conference on Genetic Algorithms*, Morgan Kaufman, pp. 521-528
WEB: http://citeseer.ifi.unizh.ch/deb95optimizing.html
http://citeseer.ist.psu.edu/deb95optimizing.html

See also

- Chromosome Shuffling
- 2N-Parent Parameter Wise Crossover

Algorithm

```
1. select two parents $A^{(t)}$ and $B^{(t)}$ form a parent pool
2. assume that each parent vector is divided into $q$ substrings $s_{ij}^{(t)}$, where $q$ is the
   number of parameters represented in each parent vector i.e. each
   $s_{ij}^{(t)}$ ($i$=$A$,$B$; $j$=1,...,$q$) represents a $j$th parameter; hence $A^{(t)} = (s_{A1}^{(t)},...,s_{Aq}^{(t)})$,
   $B^{(t)} = (s_{B1}^{(t)},...,s_{Bq}^{(t)})$
3. create two offspring $C^{(t+1)}$ and $D^{(t+1)}$ as follows:
4.     for $j$ = 1 to $q$ do
5.         if $Rnd \le p_c$ then
```

```
6.              perform crossover between $s_{Aj}^{(t)}$ and $s_{Bj}^{(t)}$
7.              $s_{Cj}^{(t+1} = s_{Aj}^{(t)} \otimes s_{Bj}^{(t)}$
8.              $s_{Dj}^{(t+1} = s_{Aj}^{(t)} \otimes s_{Bj}^{(t)}$
9.              else
10.             $s_{Cj}^{(t+1} = s_{Aj}^{(t)}$
11.             $s_{Dj}^{(t+1} = s_{Bj}^{(t)}$
12.             end if
13.      end do
```

where:
$\otimes$ – standard 1-Point Crossover operator
Rnd – uniform random real number , *Rnd* $\in <0,1>$

Comments
- The most fundamental difference between the MC operator and other operators using variable-to-variable recombination is that the answer to the question "whether to crossover" is checked in the MC method separately for each substring (row: 5). As for the other operators, the answer to that question refers to the parent vector as a whole.

Experiment domains
- function taken from the National Crime Survey

Compared to
- k-Point Crossover

Homologous Crossover

(HX)

Keywords
information exchanging, information destruction, convergence speed, non-disruptive crossover, optimal crossover points

Motivation

- Reduction of the destructive action of a multi-point crossover operator resulting from random selection of crossover points

Source text

➢ Park J., Park J., Lee Ch., Han M. (1993), Robust and Efficient Genetic Crossover Operator: Homologous Recombination, in *Proceedings of 1993 International Joint Conference on Neural Networks*, pp. 2975-2978
WEB: http://ieeexplore.ieee.org/xpl/abs_free.jsp?arNumber=714347

Read also

➢ Beyer H.-G. (1995), Toward a Theory of Evolution Strategies: On the Benefits of Sex – the ($\mu/\mu,\lambda$) Theory, in *Evolutionary Computation*, vol. 3(1), pp. 81-111
WEB: http://citeseer.ifi.unizh.ch/361714.html
http://citeseer.ist.psu.edu/361714.html

See also

- Spontaneous Crossover
- Circle-ring Crossover
- Sufficient Exchanging
- Intermediate Crossover
- Intermediate Crossover2
- Adaptive Number of Crossover Points

Algorithm

```
1. select two parents $A^{(t)}$ and $B^{(t)}$ from a current population $P(t)$
2. randomly choose $m$ crossover points $\{pc_1,...,pc_m\}$
3. create two offspring $A^{(t+1)}$ and $B^{(t+1)}$ by restricted $m$-point crossover as follows:
4. for every pair of strings $ST_A = (a^{(t)}_{pc_k},...,a^{(t)}_{pc_{k+1}}) \in A^{(t)}$ and
   $ST_B = (b^{(t)}_{pc_k},...,b^{(t)}_{pc_{k+1}}) \in B^{(t)}$ between two successive crossover points $pc_k$ and
   $pc_{k+1}$ do:
5.        if length_of_$ST_A$ (= length_of_$ST_B$ ) $\geq \varpi$ then
6.        compute the degree of similarity $DS$ of strings $ST_A$ and $ST_B$ as follows:
7.        number_of_1 = 0
```

```
8.              for $i = k$ to $k+\varpi$ do
9.                  if $a_{pc_i}^{(t)}$ XOR $b_{pc_i}^{(t)} = 1$ then
10.                 number_of_1 = number_of_1 + 1
11.                 end if
12.             end do
13.         $DS$ = number_of_1/length_of_$ST_A$
14.             if $DS \geq \tau$ then
15.             swap bits
16.             else
17.             do nothing
18.             end if
19.         else
20.         do nothing
21.         end if
22. end do
```

where:
ϖ and τ – parameters of the method

Comments

- The HX operator is based on the standard k-Point Crossover operator. Introduced modification relies on the fact that only strings of bits which are at least of a certain length (row: 5) or of an admissible degree of similarity (row: 14) are allowed to crossover. Determination of the degree of similarity is based on the XOR operator (rows: 9-13).
- This strategy is aimed at transfering (hence also protecting) strings with specified parameters to the next generation.
- In HX the value of ϖ and τ is determined à priori as constant or dynamicly changed (increased) in the GA run.

Experiment domains

- De Jong's function (F1)

Compared to

- k-Point Crossover
- Uniform Crossover
- Intermediate Crossover
- Intermediate Crossover2

Count-preserving Crossover

(CPC)

Keywords
number of "1" preservation

Motivation

- Preserving constant number of bits equal to "1" in every chromosome of a population.

Source text

➢ Hartley S.J., Konstam A.H. (1993), Using Genetic Algorithms to Generate Steiner Triple Systems, in *ACM Conference on Computer Science*, pp.366-371
WEB: http://citeseer.ifi.unizh.ch/hartley93using.html
http://citeseer.ist.psu.edu/hartley93using.html

Read also

➢ Hou Y.-Ch., Chang Y.-H. (2004), A New Efficient Encoding Mode of Genetic Algorithms for the Generalized Plant Allocation Problem, in *Journal of Information Science and Engineering*, vol. 20, pp. 1019-1034
WEB: http://www.iis.sinica.edu.tw/JISE/2004/200409_12.html

See also

- Count-preserving Crossover-2
- Set-Oriented Crossover
- Self Crossover

Algorithm

```
1.  select two parents $A^{(t)} = (a_1^{(t)},...,a_n^{(t)})$ and $B^{(t)} = (b_1^{(t)},...,b_n^{(t)})$ from a parent
    pool
2.  create two lists of differences $L^{up}$ and $L^{down}$ as follows:
3.  $L^{up}$= empty_list, $L^{down}$= empty_list, L_length = 0
4.      for $i$ = 1 to $n$ do
5.          if $a_i$ = 1 and $b_i$ = 0 then
6.          append $i$ to $L^{up}$
7.          L_length = L_length + 1
8.          else if $a_i$ = 0 and $b_i$ = 1 then
9.          append $i$ to $L^{down}$
10.         end if
11.     end do
```

```
12. create two offspring $A^{(t+1)}$ and $B^{(t+1)}$ as follows:
13. copy all bits from parent $A^{(t)}$ to offspring $A^{(t+1)}$
14. copy all bits from parent $B^{(t)}$ to offspring $B^{(t+1)}$
15.         for $j = 1$ to $L\_length$ do
16.                 if $Rnd < 0.5$ then
17.                 at position determined by $L_j^{up}$ exchange the bits between offspring
                    $A^{(t+1)}$ and $B^{(t+1)}$
18.                 at position determined by $L_j^{down}$ exchange the bits between
                    offspring $A^{(t+1)}$ and $B^{(t+1)}$
19.                 end if
20.         end do
```

where:
Rnd – uniform random real number , $Rnd \in <0,1>$
L_length – number of elements in the L^{up} and L^{down}
L_j^{up} – j^{th} element of L^{up}
L_j^{down} – j^{th} element of L^{down}

Comments

- The CPC operator carries out its task (see: motivation) assuming, that the number of bits equal to "1" in every chromosome in the initial population $P(0)$ is the same.
- CPC may guarantee preservation of the constant number of bits equal to "1" due to application of two lists noting the differences between the parents (rows: 3-11). List L^{up} includes positions (numbers) of those bits, on which there are differences between the parents, but the first parent at a given position holds a bit equal to "1" and the second equal to "0" (row: 5). List L^{down} similarly notes the positions of differences, but the first parent at a given position holds a bit equal to "0" and the second equal to "1" (row: 8). The offspring creation process making use of those lists is based on the exchange of bits between the offspring at those positions which, are indicated by subsequent element pairs from lists L^{up} and L^{down} (rows: 17 and 18).
- Number of elements in L^{up} and in L^{down} is the same, which is a direct result of the assumption, that the number of bits equal to "1" is constant for all chromosomes in $P(0)$.

Experiment domains

- n/a

Compared to

- n/a

Elitist Crossover

(EX)

Keywords
selection, exploration, exploitation, exploration/exploitation balance, competition for survival

Motivation
- Assessing the effectivity of integrating the selection and crossover processes.

Source text
➢ Thierens D., Goldberg D.E. (1994), Elitist Recombination: an integrated selection recombination GA, in *Proceedings of the First IEEE World Congress on Computational Intelligence*, pp. 508-512
WEB: http://intl.ieeexplore.ieee.org/xpl/abs_free.jsp?arNumber=349898
http://www.cs.uu.nl/groups/DSS/publications/

Read also
➢ Vekaria K., Clack C. (1998), Selective Crossover in Genetic Algorithms: An Empirical Study, in *Proceedings of the fifth Conference on Parallel Problem Solving from Nature*, Springer-Verlag, pp. 438-447
WEB: http://citeseer.ifi.unizh.ch/vekaria98selective.html
http://citeseer.ist.psu.edu/vekaria98selective.html
➢ Coli M., Genusso P., Palazzari P. (1996), A New Crossover Operator for Genetic Algorithms, in *IEEE International Conference on Evolutionary Computation*, pp. 201-206
WEB: http://citeseer.ifi.unizh.ch/coli96new.html
http://citeseer.ist.psu.edu/coli96new.html

See also
- Best Schema Crossover
- Selective Crossover-2
- Partially Randomized Crossover
- Direct Design Variable Exchange Crossover

Algorithm
```
1. for every generation of GA do
2. randomly shuffle the entire population $P(t) = \{A_1^{(t)}, ..., A_{Population\_size}^{(t)}\}$
3.     for $i$ = 1 to $Population\_size$ do
4.     create two vectors $V_1$ and $V_2$:

       $V_1 = A_i^{(t)} \otimes A_{i+1}^{(t)}$

       $V_2 = A_i^{(t)} \otimes A_{i+1}^{(t)}$
```

```
5.          compute the fitness value of $V_1$ and $V_2$
6.          insert best two vectors of $\{A_i^{(t)}, A_{i+1}^{(t)}, V_1, V_2\}$ into the next population
             $P(t+1)$ as offspring
7.          $i = i + 2$
8.          end do
```

where:
$\otimes$ – preferred crossover method

Comments

- In the standard genetic algorithm, the selection process is always preceded by the crossover process. In the EX method both of the processes are integrated. During the first step the entire population is randomly shuffled (row: 2). Then, from each successive pair of parental vectors, two new vectors are created by crossover (row: 4). From a "family" created in this way, two best vectors are singled out and implemented as offspring to the next population (row: 6).
- Application of elitist selection in the traditional way that is on the level of the entire population may often be the reason for the premature convergence of the algorithm. An EX elitist selection applied on the "family" level (row: 6) eliminates this danger according to the authors.

Experiment domains

- bit counting function
- fully deceptive trap function

Compared to

- Uniform Crossover

Index of keywords, authors and experiment domains