

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

Copyright © 2006 by Tomasz Dominik Gwiazda

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

1. Introduction	4
2. Standard operators	8
1-Point Crossover	8
k-Point Crossover	9
Shuffle Crossover	11
Reduced Surrogate Crossover.....	12
Uniform Crossover	13
Heuristic Uniform Crossover, Highly Disruptive Crossover	14
Average Crossover.....	16
Discrete Crossover	17
Flat Crossover	18
Heuristic Crossover /Intermediate Crossover	19
Blend Crossover.....	20
3. Binary operators	22
Random Respectful Crossover.....	22
Masked Crossover	25
1bit Adaptation Crossover	27
Multivariate Crossover	31
Homologous Crossover.....	33
Count-preserving Crossover	35
Elitist Crossover.....	37
4. Index of keywords, authors and experiment domains	402

Introduction

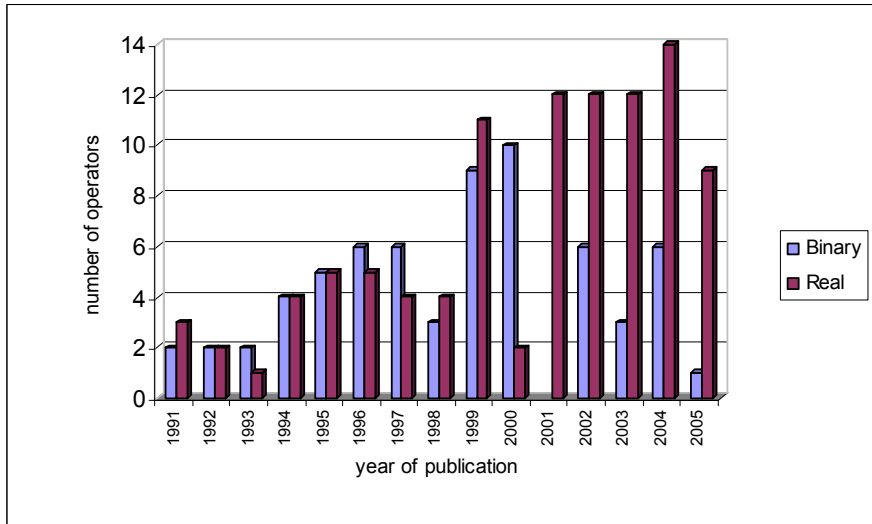
The literature on Genetic Algorithms or more widely on Evolutionary Computation is full of many excellent books and articles which are texts of introductory or review character. These texts concentrate on presentation of fundamental (or most popular at a given time) methods of selection, recombination 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 explicit 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

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

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

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

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

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

Real operators

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

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

x_i^l – lower boundary of i^{th} variable (gene),

x_i^u – upper boundary of i^{th} variable (gene)

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

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

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

x_i^l – lower boundary of i^{th} variable (gene)

x_i^u – upper boundary of i^{th} 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 $cp \in \{1, \dots, n-1\}$
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 $cp_1, \dots, cp_k \in \{1, \dots, n-1\}$
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 $cp \in \{1, \dots, n-1\}$
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 $PCP = (cp_1, \dots, cp_l)$ 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 $cp \in PCP(cp_1, \dots, cp_l)$
 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>

➤ Cordon O., Damas S., Santamaria 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 Dynamic 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 $u \leq p_s$ 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>

➤ Cordon O., Damas S., Santamaria 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 Dynamic 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

Standard operators

```
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  $u \in \langle 0, 1 \rangle$ 
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 $u \in \langle 0, 1 \rangle$
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
 $\alpha \in \langle \min(x_i^{(t)}, y_i^{(t)}), \max(x_i^{(t)}, y_i^{(t)}) \rangle$
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 $x_i^{(t)} \leq y_i^{(t)}$
5. choose a uniform random real number $\alpha \in \langle 0, 1 \rangle$
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 $\langle 0, 1 \rangle$ (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

$$u \in \left(\min(x_i^{(t)}, y_i^{(t)}) - \alpha d_i, \max(x_i^{(t)}, y_i^{(t)}) + \alpha d_i \right)$$
6. $x_i^{(t+1)} = u$
7. choose a uniform random real number

$$u \in \left(\min(x_i^{(t)}, y_i^{(t)}) - \alpha d_i, \max(x_i^{(t)}, y_i^{(t)}) + \alpha d_i \right)$$
8. $y_i^{(t+1)} = u$
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 $x_i^{(t)} \leq y_i^{(t)}$ than
 7. choose a uniform random real number
$$u \in \langle x_i^{(t)} - \alpha d_i, y_i^{(t)} + \beta d_i \rangle$$
 8. $x_i^{(t+1)} = u$
 9. choose a uniform random real number
$$u \in \langle x_i^{(t)} - \alpha d_i, y_i^{(t)} + \beta d_i \rangle$$
 10. $y_i^{(t+1)} = u$
 11. else
 12. choose a uniform random real number
$$u \in \langle y_i^{(t)} - \beta d_i, y_i^{(t)} + \alpha d_i \rangle$$
 13. $x_i^{(t+1)} = u$
 14. choose a uniform random real number
$$u \in \langle y_i^{(t)} - \beta d_i, y_i^{(t)} + \alpha d_i \rangle$$
 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 $S^{AB} = (s_1^{AB}, \dots, s_n^{AB})$ 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 $u \in \langle 0,1 \rangle$
 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 $u \in \langle 0,1 \rangle$
 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:

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

$P^B = (p_1^B, \dots, p_n^B)$ – 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 $u \in \langle 0, 1 \rangle$
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 $cp_1, cp_2 \in \{1, \dots, n-1\}$ ($cp_1 < cp_2$)
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  $u \in \langle 0, 1 \rangle$ 
22.             if  $u \leq p_s$  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  $u \in \langle 0, 1 \rangle$ 
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  $u \leq p_s$  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
          $cp_1, cp_2 \in \{1, \dots, n-1\}$  ( $cp_1 < cp_2$ )
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 $s_{ij}^{(t)}$ ($i=A,B; j=1, \dots, q$) represents a j^{th} parameter; hence $A^{(t)} = (s_{A1}^{(t)}, \dots, s_{Aq}^{(t)})$,
 $B^{(t)} = (s_{B1}^{(t)}, \dots, s_{Bq}^{(t)})$
3. create two offspring $C^{(t+1)}$ and $D^{(t+1)}$ as follows:
4. for $j = 1$ to q do
5. if $Rnd \leq p_c$ 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 $\{pc_1, \dots, pc_m\}$
3. create two offspring $A^{(t+1)}$ and $B^{(t+1)}$ by restricted m -point crossover as follows:
4. for every pair of strings $ST_A = (a_{pc_k}^{(t)}, \dots, a_{pc_{k+1}}^{(t)}) \in A^{(t)}$ and

$ST_B = (b_{pc_k}^{(t)}, \dots, b_{pc_{k+1}}^{(t)}) \in B^{(t)}$ between two successive crossover points pc_k and

pc_{k+1} do:

5. if $\text{length_of_}ST_A (= \text{length_of_}ST_B) \geq \varpi$ then
6. compute the degree of similarity DS of strings ST_A and ST_B as follows:
7. $\text{number_of_}1 = 0$

```

8.           for  $i = k$  to  $k + \varpi$  do
9.               if  $a_{pc_i}^{(t)} \text{ XOR } b_{pc_i}^{(t)} = 1$  then
10.                  number_of_1 = number_of_1 + 1
11.              end if
12.          end do
13.       $DS = \text{number\_of\_1} / \text{length\_of\_ST}_A$ 
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 transferring (hence also protecting) strings with specified parameters to the next generation.
- In HX the value of ϖ and τ is determined a priori as constant or dynamically 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 $A^{(t)} = (a_1^{(t)}, \dots, a_n^{(t)})$ and $B^{(t)} = (b_1^{(t)}, \dots, b_n^{(t)})$ from a parent pool
2. create two lists of differences L^{up} and L^{down} as follows:
3. $L^{up} = \text{empty_list}$, $L^{down} = \text{empty_list}$, $L_length = 0$
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 , $Rnd \in \langle 0,1 \rangle$

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.ieeeexplore.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 $P(t) = \{A_1^{(t)}, \dots, A_{Population_size}^{(t)}\}$
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:

- ⊗ – 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

A

adaptive..... 28, 30, 53, 94, 116, 145, 156,
158, 163, 185, 203, 233, 274, 288,
291, 294, 298, 306, 318, 323, 326,
329, 358, 366, 368, 379, 381

adaptive culture model..... 390

adaptive length chromosome 296

allele distribution 156

alternative to crossover . 68, 75, 102, 106,
126, 308

approximation 251, 253

Author

Abbas H.A..... 274, 288, 291, 323
Abdel-Aty-Zohdy A.S..... 161
Acan A..... 158, 303, 308, 326
Adachi N..... 251, 253
Agrawal R.B..... 211
Aitken J.M..... 320
Alfonso H..... 237, 261
Alliot J.-M..... 245
Altinçay H..... 158, 303, 326
Anand A..... 397
Anderson K.S..... 251, 253, 306
Angantyr A..... 333
Angelov P.P..... 268
Antonsson E.K..... 148, 200
Aydin M.E..... 276, 344, 347, 355, 364
Bäck T..... 55
Bandyopadhyay S..... 62, 91
Barrido S.C..... 294
Beasley J.E..... 87
Beldiman L..... 16, 17
Belea R..... 16, 17
Bendisch J..... 213
Bentley P.J..... 62
Bersini H..... 208, 235
Beyer H.-G..... 36, 211
Bhattacharyya S..... 156
Bi Z..... 172, 266
Blaise Madeline..... 298, 300
Bort E..... 257, 392
Buczak A.L..... 274
Bullinaria J.A..... 166
Burdsall B..... 239, 286
Burke D.S..... 148

Burkowski F.J..... 14
Cabido R..... 121
Callaghan V..... 203
Cantú-Paz E..... 137, 139
Cao H..... 249
Caprani O..... 198, 257, 392
Cen L..... 185
Chakraborty G..... 255, 370
Chan Ch.-H..... 370
Chan K.P..... 397
Chan K.Y..... 170, 276, 344, 347, 355,
364
Chang Ch.-Y..... 379
Chang H.C..... 340
Chang Y.-H..... 38, 97, 179
Chen J.-N..... 28
Chen P.H..... 340
Chen S..... 123
Chen Y..... 249
Cheng H..... 288, 370
Cheung Y.-M..... 174
Chou Ch.-H..... 28
Chou J.-H..... 344, 351, 355, 388
Chu P.C..... 87
Chuwu P..... 145, 384
Clack C..... 28, 40, 94
Clarke G..... 203
Coello Coello C. A..... 394
Coghill G..... 358
Coli M..... 40, 55, 141, 145
Colley M..... 203
Colorni A..... 390
Cordón O..... 16, 17
Costa E..... 102, 106, 126, 308
Cotta C..... 16, 17, 239, 386
Covic G..... 358
Cremonezi R.R..... 329
Cvetković D..... 20, 22, 213
Dadios E.P..... 294
Dallaali M.A..... 372
Damas S..... 16, 17
De Falco I..... 386
De Jong K.A..... 16, 17, 148
De San Pedro M.E..... 237
de' Garis H..... 249
Deb K..... 34, 55, 116, 143, 211, 397
Delgado M.R..... 329

Index of keywords, authors and experiment domains

Della Cioppa A.....	386	Hasegawa K.....	68, 75, 102, 106, 126, 308
Demiröz G.....	227	Herrera F. 21, 22, 23, 30, 70, 158, 174, 206, 213, 215, 303, 316, 326, 353	
Dolezal O.....	121	Hervás-Martínez C.....	398
Dongarra J.....	396	Higuchi T.....	263
Dorigo M.....	390	Hinterding R.....	200, 329
Dotoli M.....	222	Hiroyasu T.....	137, 139, 145, 396
Drake S.....	276	Hirsh H.....	259
Duarte A.....	121	Ho S.-Y.....	344
Durand N.....	245	Hofmeister T.....	121
Eberhart R.C.....	390	Hong I.....	70, 174
Eiben A.E..... 42, 58, 82, 91, 100, 152, 166, 170, 182, 241, 243, 247, 329, 362		Hoshi K.....	255, 370
Emmanouilidis C.....	123	Hou C.-Z.....	291, 368
Erbatur F.....	141, 143, 394	Hou Y.-Ch.....	38, 97, 179
Esquivel S.C.....	82, 237, 261, 303	Hsu Y.....	251, 253, 306
Estivill-Castro V.....	84	Huang Ch.-H.....	366
Ewing R.L.....	161	Huang D.-S.....	177
Fernandes C.....	19, 270, 272	Huang J.-H.....	366
Fernandez F.....	121	Hunter A.....	123
Ferrero S.....	237, 261	Husbands P.....	276
Fogarty T.C..... 53, 170, 188, 276, 298, 344, 347, 355, 364		Ishibuchi H.....	158
Foley Ch.M.....	102, 106	Jain L.....	100, 247
Franceschini G.....	257, 392	Janikow C.Z.....	194, 196
Fuller A.T.G.....	222	Jannet T.C.....	390
Furuhashi T.....	68, 75	Jian W.....	203, 368, 381
Gagliardi F.....	239, 386	Jian Y.....	306
Galla T.M.....	89	Jianfu T.....	296
Gallant P.J.....	320	Jiao L.....	266, 368, 381
Gallard R.H.....	82, 237, 261, 303	Jichang G.....	296
Gang Z.....	145, 384	Jin G.....	251, 253
Gantala D.....	161	Jones T. C.....	218, 220
García-Pedrajas N.....	398	Joshi D.....	397
Genusso P.....	40, 55, 141, 145	Joyner C.R.....	72, 282, 344, 347, 355
Ghosh A.....	100, 241, 243, 247, 375	Ju P.....	288, 370
Gibson G.M.....	222, 333	Kahng A.B.....	70, 174
Giraud-Carrier Ch.....	239, 286	Kang L.....	249, 318
Goldberg D.E.....	40, 80, 116, 263, 397	Kanoh H.....	68, 75, 102, 106, 126, 308
Gong D.....	163, 182, 268, 362, 375	Kargupta H.....	116
Gow H.-J.....	379	Kato K.....	372
Goyal M.....	34, 55, 143	Kato N.....	68, 75, 102, 106, 126, 308
Grefenstette J.J.....	148, 255	Keller B.....	66
Guerra-Salcedo C.....	123	Kendall G.....	296, 320
Guo T.....	249	Kita H.....	23, 245, 280, 396
Guokun Z.....	172	Kobayashi S.....	280, 318, 396, 398, 399
Güvenir H.A.....	227	Kok J.N.....	42, 58
Gwiazda T.D.....	177, 303, 394	Konstam A.H.....	34, 38, 97, 179
Hagras H.....	203	Krahenbuhl L.....	342
Han L.....	296, 320	Krasnogor N.....	353
Han M.....	36, 49, 51, 163	Ku S.....	286
Harik G.R.....	80, 116	Kubalik J.....	154
Hartley S.J.....	34, 38, 97, 179	Kundu M.K.....	45, 97
Harvey I.....	68, 75, 148	Lai S.....	323
Hasançebi O.....	141, 143, 394	Lam H.K.....	340
		Lasso M.....	237
		Lazansky J.....	154

Index of keywords, authors and experiment domains

Lazarescu V.	284	Moreira F.J.S.	342
Lee B.	123, 208, 235, 286	Mühlenbein H.	20, 22, 60, 206, 213, 397
Lee C.-Y.	148, 200	Munteanu C.	19, 270, 272, 284
Lee Ch.	36, 49, 51, 163	Murata T.	158
Lee J.-D.	366	Murthy A.S.	45
Lee K.Y.	314	Murthy C.A.	62
Lee S.-W.	364	Nakano R.	208
Lefmann	121	Nandi S.	45, 97
Leiva A.	82, 237, 261, 303	Narayanan A.	399
Leung F.H.F.	340	Naso D.	222
Leung Y. W.	72, 266, 282	Nawa N.E.	68, 75
Li Bin	399	Nazhiyat G.	229, 231
Li F.	145, 338	Noel M.M.	390
Li J.	318	Nomura T.	11, 12, 19, 25, 225, 270, 272, 314
Li Y.-H.	360	Nong Y.	338
Li Yan-Da	49, 51	Nossal R.	89
Lian Y.	185	Nowrouzian B.	222
Lima C.F.	397	Ohmukai I.	137, 139, 145
Lin W.	384	Ono I.	280, 318, 396
Ling S.H.	340	Ortiz-Boyer D.	398
Lis J.	82, 91, 166	Ozugur T.	25, 188
Liu G.	370	Paaß G.	60
Liu Q.-H.	145	Pal N.R.	45, 97
Liu T.-K.	344, 351, 355, 388	Pal S.K.	62, 91
Liu Z.	323	Palazzari P.	40, 55, 141, 145
Lobo F.	397	Pan F.	163
Logan T.D.	190, 192, 194, 196, 198, 227	Pan X.-Q.	291, 368
Louis S.J.	28, 47	Pandolfi D.	237
Lozano M.	21, 22, 23, 30, 70, 158, 174, 206, 213, 215, 303, 316, 326, 353	Park C.H.	34, 55
Lutz R.	66	Park J.	36, 49, 51, 163
MacIntyre J.	123	Park L.-J.	34, 55
Maini H.	28, 47	Patnaik L.M.	203, 255, 274, 288, 291, 323, 368
Maione G.	222	Pelikan M.	397
Maniezzo V.	390	Pham D.T.	251, 253
Mao Zhi-Hong	49, 51	Pi L.	296
Massa A.	257, 392	Pollack J.B.	25, 112, 130, 132, 135, 200
Mayer H.A.	89, 116, 168	Pounds-Cronish A.	203
Mehrotra K.	28, 47, 80	Premaratne M.	372
Mezura-Montes E.	394	Price K.V.	233
Michalewicz M.	229, 231	Qiang L.	296
Michalewicz Z.	190, 192, 194, 196, 198, 227, 229, 231, 257, 329, 392	Qiao J.-F.	268, 375
Miki M.	137, 139, 145, 396	Qifu X.	338
Min F.	145	Ra J.W.	34, 55
Ming L.	174	Radcliffe N.J.	25
Mingshu Z.	145, 384	Ramirez J.A.	342
Miyoshi T.	225, 314	Ramos R.M.	342
Mohamed P.S.	314	Randolph D.	235
Mohan C.K.	78, 80, 118	Ranka S.	28, 47
Mohan Ch.	28, 47	Rasheed K.	259
Molina D.	353	Raué P.-E.	42, 58
Moon B.R.	70, 158, 174, 326	Rawlins G.J.E.	28, 47
Moore M.	399	Ray T.	397

Index of keywords, authors and experiment domains

Riff-Rojas M.C.....	298	Tsutsui S.....	100, 241, 243, 247, 263, 375, 396, 397
Rosa A.....	19, 270, 272, 284	Turchiano B.....	222
Rothkrantz L.J.M.....	154	Unal R.....	72, 282, 344, 347, 355
Ruan X.....	182, 268, 362, 375	Ünveren A.....	158, 303, 326
Russo M.....	388	van Kemenade C.H.M.....	42, 58, 170
Ruttkey Zs.....	42, 58	Vasconcelos J.A.....	342
Sakawa M.....	372	Vekaria K.....	28, 40, 94
Sakuma J.....	396, 398, 399	Venkatarayalu N.....	397
Salcic Z.....	358	Verdegay J.L.....	21, 22, 215
Saldanha R.R.....	342	Villagra A.....	237
Salman A.A.....	80	Villar P.....	303, 316
Salto C.....	237, 261	Voigt H.-M.....	20, 22, 60, 213
Sánchez A.M.....	23, 30, 70, 121, 158, 174, 206, 303, 316, 326	Voss M.S.....	102, 106
Sanchez-Velazco J.....	166	Vrajitoru D.....	30, 114, 166, 172, 174, 358
Sano M.....	396	Wakefield J.P.....	62
Santamaria J.....	16, 17	Wang H.....	274
Sastry K.....	263, 397	Wang W.Y.....	360
Sastry V.U.K.....	45	Wang Y.....	174, 266, 368, 381
Savage M.J.W.....	358	Wang Y.-P.....	174
Schlierkamp-Voosen D.....	20, 22, 206	Watson R.A.....	25, 112, 130, 132, 135, 200
Schoenauer M.....	229, 231	Wei L.....	266
Schütz M.....	237, 261	Wei P.....	377
Schwefel H.-P.....	55	Whitley L.D.....	123
Seetharaman G.S.....	338	Won K.S.....	397
Sharawi M.S.....	161	Wright A.H.....	190, 192
Shimosaka H.....	396	Wright J.A.....	268
Shu L.-S.....	344	Wu C.-Y.....	338
Simões A.....	102, 106, 126, 308	Wu Z.....	318
Smith J.....	53, 188	Xiaohui Hu.....	390
Smith J.E.....	298	Xiong W.-Q.....	377
Smith P.....	68, 75	Xue Y.....	203, 368, 381
Smith S.F.....	123	Yamada T.....	208
Someya H.....	398	Yamamura M.....	263, 398
Spears W.M.....	16, 17, 30, 70, 358	Yan W.....	198
Spirkovska L.....	152	Yang G.-W.....	145
Spitzlinger M.....	89, 168	Yang H.....	249
Srinivas M.....	203, 255, 274, 288, 291, 323, 368	Yang Junan.....	399
Stanley D.O.....	72, 282, 344, 347, 355	Yang Q.....	203, 368, 381
Stidsen T.J.....	198, 257, 392	Yang S.....	156, 366
Su L.-M.....	291, 368	Yang S.Y.....	34, 55
Sun X.....	163	Yang Y.-M.....	384
Swaminathan S.....	190, 192, 194, 196, 198, 227	Yanjiao K.....	266
Takahashi M.....	23, 245, 280	Yen J.....	123, 208, 235
Takahashi R.H.C.....	342	Yimin Y.....	266
Tarantino E.....	386	Yongquan Y.....	172
Tavares R.....	19, 270, 272	Yoon H.-S.....	158, 326
Tekol Y.....	158, 303, 308, 326	Yoshida J.....	137, 139, 145
Thierens D.....	40	Yoshida Y.....	251, 253
Troutt M.D.....	156	Yu J.....	249
Troya J.M.....	16, 17	Yu N.....	338
Tsai J.-T.....	344, 351, 355, 388	Yuan B.....	132, 135, 379
Tsay M.-T.....	379	Yuhui Shi.....	390
Tsui H.-T.....	364	Yunping C.....	306

Yushu L.....338
 Zeng B..... 384
 Zhang G.-Z.....177
 Zhang H.291, 368
 Zhang Liang-Jie49, 51
 Zhang Q.72, 266, 282
 Zhang Y.-P.....377
 Zhao J.....266, 368, 381
 Zhao S.....266, 368, 381
 Zhou J.....323
 Zhou X.....288, 370
 Zhu Z.....198
 Zhuang Zhenquan399
 auto-adjust scheme.....379
 averaging effect.....270, 272, 284

B

bacterial recombination..... 68, 75
 biological site-specific recombination276
 biologically inspired..... 68, 75, 102, 106,
 126, 308
 blindness of algorithm.....377, 384
 boundary of a feasible region.....229, 231

C

categorical variables.....333
 center of gravity268
 chromosome differentiation91
 chromosome structure.....116
 closed operator..... 194, 196, 198, 227
 combination of crossovers30, 70, 158,
 174, 326
 combination of differences.....130
 competition for survival.....40
 computational complexity.....55, 84, 185,
 208, 213, 235, 298, 300, 314
 constraints satisfaction .. 72, 87, 143, 194,
 196, 229, 231, 251, 253, 298, 300,
 372, 394
 contribution of parents to their offspring
182, 362
 convergence speed 36, 49, 51, 55, 60,
 123, 145, 161, 172, 185, 188, 213,
 233, 245, 253, 255, 259, 276, 288,
 291, 314, 320, 323, 333, 366, 370,
 384, 392
 convex continuous parameter space..194,
 196
 cross-gender cooperation166
 crossover performance114
 crossover probability.....255, 370

D

data clustering 386
 deception 80, 154
 dependency of variables 333
 derivatives 253
 deterministic crowding 132, 135
 diploid chromosomes..... 94
 directional search..... 306, 342
 disparate parameters 222
 diversity...68, 75, 89, 102, 106, 126, 132,
 135, 163, 172, 215, 225, 251, 253,
 255, 268, 272, 284, 306, 308, 316,
 320, 323, 333, 351, 353, 379
 dominance 94
 double string coding 372
 dynamic representation of gene position
 80

E

economy in computation 237, 261
 effectiveness 66, 141, 143, 161, 172, 177,
 190, 192, 206, 229, 231, 233, 270,
 272, 286, 294, 303, 338, 340, 342,
 351, 353, 368, 370, 392
 epistasis 28, 318
 exchanging relationships of genes 364
Experiment domains
 10-bar truss problem 142, 144
 25-bar truss problem 142, 144
 2-dimensional shape matching
 problem..... 151
 4-bit parity problem 205
 6-bit deceptive problem 219
 6-bit easy problem 219
 6-Hump Camel Back function 111,
 184
 72-bar truss problem 142, 144
 Ackley's function59, 67, 217, 312,
 317
 active power security correction on
 power market 307
 actuator hysteresis identification and
 compensation 371
 aligning multiple protein sequence
 problem..... 178
 assembling parts into objects problem
 365
 automatic generating numerical
 control rules problem 385
 Baluja's function 279
 benchmark set of 65 set covering
 problems 88
 bimodal equal spread function..... 212
 bimodal function..... 61, 79

bimodal unequal spread function ..212	function taken from the National
bit counting function41	Crime Survey..... 35
blocked function.....212	fuzzy controller design 50, 52
Bohachewsky's function 111, 315, 317	fuzzy rule extraction problem..... 226
Branin's function ... 184, 359, 363, 376	Goldstein-Price's function..... 169
busy beaver problem219	graph bipartitioning 79
camel function.....325	graph bisection problems..... 160
chaotic time-series prediction problem	graph coloring problem... 44, 299, 302
.....322	graph partitioning 48
clique finding problem46	Griewank's function 59, 111, 115, 129,
Colville's function317	138, 140, 176, 184, 207, 214, 238,
concatenations of bipolar deceptive	242, 244, 246, 248, 262, 265, 269,
problems..... 79, 120	317, 319, 354, 363, 376
concatenations of order-3 deceptive	Hamiltonian Circuit problems 115,
problems..... 79, 120	176
conceptual design of a supersonic	HIFF problem113, 131, 134, 136
aircraft260	induction motor parameter estimation
contrast stretching and detail 290
enhancements of satellite images	industrial simulation 224
.....285	integrated steelmaking optimum cast
Corana's function 246, 279	plan 369
data clustering benchmark sets.....240	JSSP 155
data clustering in dermatological	Keane's function230, 232, 252, 254
semeiotics.....387	knapsack problem 202, 287
De Jong's function.. 27, 37, 39, 44, 59,	knowledge aquisition problem..... 293
67, 77, 83, 85, 101, 111, 115, 129,	L-SAT problem generator..... 96, 157
155, 171, 176, 184, 189, 191, 193,	Mahfoud's function..... 134, 136
199, 205, 212, 221, 234, 236, 238,	max-cut problem..... 122
242, 244, 248, 256, 265, 269, 279,	Michalewicz & Schoenauer's
319, 325, 359, 363, 376, 391	functions 305, 395
deceptive trap functions96	Michalewicz's function ... 44, 59, 105,
design of a three-stage membrane	111, 129, 210, 236
separation process260	microwave imaging problem 393
design of aerodynamic shape of a car	mixed variable bearing problem ... 337
.....65	multi-dimensional pattern recognition
design of digital finite impulse	problems 93
response filters187	multimodal function 56, 165, 173
design of two-way crossover circuits	multi-objective function..... 267
.....153	multi-objective optimization of logic
determination of a bus access	circuits 383
schedule for a real-time LAN90	n-bit adder..... 29
dynamic biochemical sensor	n-bit parity checker..... 29
measurement characterization .. 162	non-stationary function..... 252, 254
Easom's function ... 184, 238, 262, 363	n-peak problmes 33
economic load-dispatch problem...341	One-Max problem..... 54, 157, 219
enhanced Steiner problems in graphs	optical components benchmark data
..... 74	sets 374
feature selection problem 125	order-3 deceptive problem..... 205
feature selection problem of pattern	pattern matching problem 50, 52
recognition99	plant allocation problem 181
feature weights learning problem ..228	pole problem.....212
Fletcher-Powell's function.....171	power plant control system design
frequency assignment problem.....328	problem.....315
fully deceptive 4-bit problem81	Rastrigin's function.59, 111, 115, 129,
fully deceptive trap function41	138, 140, 176, 212, 214, 217, 242,

244, 248, 265, 269, 281, 317, 319, 343, 354	fixed mapping problem..... 211
Rechenberg's function207	fixed-locus scheme 112
registration of computer tomography image and facial surface data problem367	
Ridge's function 138, 140	G
robot tracking problem.....295	gene pool 60
Rosenbrock's function .. 169, 207, 217, 269, 281, 317, 319, 354	general applicability 78
Royal Road function 54, 157, 219	global optima localization 66
SAT problems 115	global search ability..... 139
satisfiability problem.....69	grammatical inference 66
Schaffer & Eshelman's trap function54	
Schaffer's function.. 67, 115, 147, 176, 184, 191, 193, 325, 363, 376	H
Schubert's function..... 67, 111, 359	Hamming cliff problem 211
Schwefel's function 59, 111, 115, 129, 138, 140, 169, 176, 242, 244, 248, 265, 269, 317, 354, 376	hill-climbing 123, 208, 235, 276
Shekel's function 184, 363	hybridizing 123, 208, 235, 276
soft decision decoding of linear block codes48	
sphere function.....217, 317, 354	I
stellar wave-front slopes prediction problem 322	idependent parameters 222
trainer scheduling problem.....297	illegitimate transposition ... 102, 106, 126
transportation problems..... 195, 197	implicit mutation 318
TSP 44, 48, 160, 167, 205, 312	indepencece form coordinate system. 263
unimodal function214	independent component analysis 280
V-cliff function 212	infection..... 68, 75
vector quantization problem..271, 273	information destruction 36, 49, 51
V-function..... 212	information exchanging..... 36
VLSI standard cell placement problem 71	intra-gender competition 166
Watson's function.....317	
Whitley's function 155	K
XOR problem..... 205	King strategy 384
zero/one multiple knapsack problem 117	knowledge accumulation 94
experimental design methods..... 72, 170, 266, 282, 344, 347, 355	knowledge-based crossover ... 47, 78, 118
exploitation . 40, 141, 143, 237, 253, 257, 261, 303, 318, 392	
exploration 40, 58, 89, 141, 143, 154, 251, 253, 257, 261, 318, 392, 394	L
exploration/exploitation balance ...40, 94, 154, 203, 215, 259, 261, 263, 274, 316	landscape of the problem 257, 392
	large optimization problems 253
F	Latin square 170
factorial design methods ... 344, 347, 355	linear constraints..... 194, 196
fitness driven crossover.....28	linear convergence..... 213
fitness surface257, 392	linear improvement..... 251
	linear non-convex combination 318
	linkage..... 53
	linkage equilibrium..... 60
	linkage probabilities 80
	local improvement..... 123, 208
	local search ability 137
	M
	macromutation..... 220
	main effect analysis 344, 347
	mathematical relationships 364

mechanics of crossover218, 220
 messy GA..... 272
 mixed variable.....333
 morphological filters338
 multi-chromosomal recombination168
 multi-chromosomal representation.....89,
 168
 multi-parent crossover 42, 53, 55, 58, 60,
 72, 82, 100, 143, 152, 170, 182, 198,
 208, 229, 231, 235, 241, 243, 247,
 249, 257, 261, 263, 303, 318, 360,
 362, 375, 392, 394
 multiple data types222, 333

N

neighborhood-based318
 neurocomputing model375
 non-coding segments116
 non-convex linear combiantion249
 non-disruptive crossover ..36, 49, 51, 116
 non-separable fitness function.....280
 number of “1” preservation....38, 45, 97,
 179
 number of crossover points163

O

optimal crossover points ..36, 49, 51, 163
 orthogonal array ...72, 282, 344, 347, 355
 orthogonal design.....282, 344, 347, 355
 orthogonal GA72, 282, 344, 347, 355
 orthogonal Latin square170

P

panmictic crossover394
 parabolic model.....257, 392
 parallel distributed GA137, 139
 parameter control298
 parameters estimation249
 parent differentiation.....377
 parent-centric crossover353
 partial commitment112
 partially specified representation112
 performance ..42, 58, 156, 172, 203, 294,
 381
 precision.....208, 211, 235, 351
 premature convergence132, 135, 145,
 163, 185, 203, 215, 251, 253, 288,
 291, 294, 306, 368, 377, 379, 381
 preservation of similarity130
 principal component analysis.....280
 probability of gene crossover.....366
 problem-specific knowledge ..47, 78, 118

processing time.....237, 261
 projections253
 promoter116

Q

quadratic representation of search
 domain253
 quality of parents388
 quality of solution.....276
 quantization266, 282, 344, 347

R

ranking.....323
 real-time system design89
 recombination model30, 70, 158, 174,
 326
 reduced-form genetic algorithm360
 reliability208, 235

S

schema25
 schema preservation28, 53, 123, 145,
 154, 188
 schema ranking.....188
 search direction.....251, 253
 selection.....40
 selective pressure.....255, 316
 self-similarity132, 135
 semantic hierarchy.....62
 separability of fitness function..245, 280,
 318
 set-oriented286
 sex82, 91, 166
 sex separation166
 signal-to-noise ratio355
 similarity25, 62
 simplex method208, 235
 small population sizes.....274
 sociological model.....390
 speed of algorithm84, 294, 377, 381
 speed of searching360
 statistics-based.....156
 structural information121
 sum of values preservation227

T

Taguchi method.....344, 347, 355
 terminator116
 tightly linked building blocks116

U

utilization of genetic information.....251,
253
underspecified representation.....112
uniform array266
uniform design266
uniform GA.....266
usefulness of crossover218, 220

V

variable length chromosome62, 148, 200,
222, 239, 286, 296, 308, 320, 386
variable length parameters..... 222
variable-to-variable recombination34, 55,
143