## **Evolutionary Algorithms: Real-Parameter Evolutionary Algorithms**

# Jiří Kubalík Czech Institute of Informatics, Robotics, and Cybernetics Czech Technical University in Prague



http://cw.felk.cvut.cz/doku.php/courses/a4m33bia/start

## **Evolutionary Algorithms for Real-Parameter Optimizations**

#### **Real Coding Advantages**

- The use of real parameters makes it possible to use large domains (even unknown domains!)
   for the variables.
- The capacity to exploit the graduality of the functions with continuous variables. The concept
  of graduality refers to the fact that slight changes in the variables correspond to slight changes
  in the function.
- The capacity for the local tuning of the solutions.
- The coding and decoding processes are avoided, thus increasing the EA's speed.

## **Evolutionary Algorithms for Real-Parameter Optimizations**

#### **Real Coding Advantages**

- The use of real parameters makes it possible to use large domains (even unknown domains!)
   for the variables.
- The capacity to exploit the **graduality** of the functions with continuous variables. The concept of graduality refers to the fact that slight changes in the variables correspond to slight changes in the function.
- The capacity for the local tuning of the solutions.
- The coding and decoding processes are avoided, thus increasing the EA's speed.

#### **Evolutionary Algorithms**

- Standard selecto-recombinative genetic algorithms with real representation.
- Evolution strategies.
- Differential Evolution.

#### Standard recombination operators

- Simple Crossover (1-point)
- Average Crossover
- Arithmetic Crossover
- Flat Crossover
- Blend Crossover BLX- $(\alpha)$

## **Specialized recombination operators**

- Unimodal Normal Distribution Crossover (UNDX)
- Simplex Crossover (SPX)
- Parent-Centric (PCX)

Let us assume that  $P_1=(p_1^1\dots p_n^1)$  and  $P_2=(p_1^2\dots p_n^2)$  are two parental chromosomes.

■ Simple Crossover – a position  $i \in 1, 2, ..., n-1$  is randomly chosen and the two offspring chromosomes  $O_1$  and  $O_2$  are built as follows

$$O_1 = (p_1^1, p_2^1, \dots, p_i^1, p_{i+1}^2, \dots, p_n^2)$$

$$O_2 = (p_1^2, p_2^2, \dots, p_i^2, p_{i+1}^1, \dots, p_n^1)$$

- Average Crossover an offspring  $O=(o_1,\ldots,o_i,\ldots,o_n)$  is created such that  $o_i=(p_i^1+p_i^2)/2$ , for  $i=1\ldots n$
- Arithmetical Crossover two offspring  $O_k=(h_1^k,\dots,h_n^k)$ , where k=1,2, are built as follows  $o_i^1=\lambda p_i^1+(1-\lambda)p_i^2$ , for  $i=1\dots n$   $o_i^2=\lambda p_i^2+(1-\lambda)p_i^1$

where  $\lambda$  is a constant or varies with regard to the number of generations made.

Let us assume that  $P_1=(p_1^1\dots p_n^1)$  and  $P_2=(p_1^2\dots p_n^2)$  are two parental chromosomes.

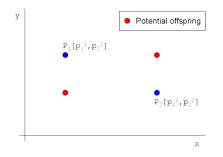
■ Simple Crossover – a position  $i \in 1, 2, ..., n-1$  is randomly chosen and the two offspring chromosomes  $O_1$  and  $O_2$  are built as follows

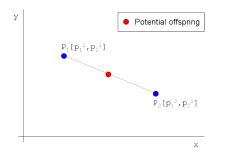
$$O_1 = (p_1^1, p_2^1, \dots, p_i^1, p_{i+1}^2, \dots, p_n^2)$$

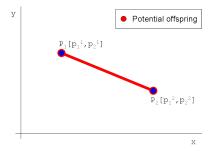
$$O_2 = (p_1^2, p_2^2, \dots, p_i^2, p_{i+1}^1, \dots, p_n^1)$$

- Average Crossover an offspring  $O=(o_1,\ldots,o_i,\ldots,o_n)$  is created such that  $o_i=(p_i^1+p_i^2)/2$ , for  $i=1\ldots n$
- Arithmetical Crossover two offspring  $O_k=(h_1^k,\ldots,h_n^k)$ , where k=1,2, are built as follows  $o_i^1=\lambda p_i^1+(1-\lambda)p_i^2$ , for  $i=1\ldots n$   $o_i^2=\lambda p_i^2+(1-\lambda)p_i^1$

where  $\lambda$  is a constant or varies with regard to the number of generations made.







Let us assume that  $P_1=(p_1^1\dots p_n^1)$  and  $P_2=(p_1^2\dots p_n^2)$  are two parental chromosomes.

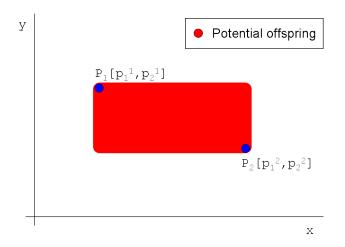
- Flat Crossover an offspring  $O=(o_1,\ldots,o_i,\ldots,o_n)$  is created such that  $o_i$  is randomly (uniformly) chosen value from interval  $[min(p_i^1,p_i^2),max(p_i^1,p_i^2)]$ .
- Blend Crossover an offspring  $O=(o_1,\ldots,o_i,\ldots,o_n)$  is created such that  $o_i$  is randomly (uniformly) chosen number from interval

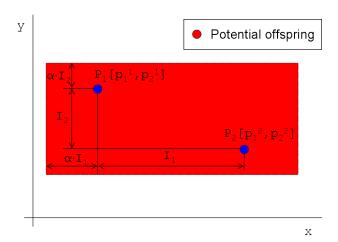
```
c_{min}-I\cdot \alpha, c_{max}+I\cdot \alpha, for i=1\dots n, where c_{min}=min(p_i^1,p_i^2), c_{max}=max(p_i^1,p_i^2), I=c_{max}-c_{min}
```

Let us assume that  $P_1=(p_1^1\dots p_n^1)$  and  $P_2=(p_1^2\dots p_n^2)$  are two parental chromosomes.

- Flat Crossover an offspring  $O=(o_1,\ldots,o_i,\ldots,o_n)$  is created such that  $o_i$  is randomly (uniformly) chosen value from interval  $[min(p_i^1,p_i^2),max(p_i^1,p_i^2)]$ .
- Blend Crossover an offspring  $O=(o_1,\ldots,o_i,\ldots,o_n)$  is created such that  $o_i$  is randomly (uniformly) chosen number from interval

$$c_{min}-I\cdot lpha, c_{max}+I\cdot lpha$$
, for  $i=1\dots n$ , where  $c_{min}=min(p_i^1,p_i^2)$ ,  $c_{max}=max(p_i^1,p_i^2)$ ,  $I=c_{max}-c_{min}$ 





### **Parent-Centric Crossover**

### Parent-Centric Crossover (PCX)

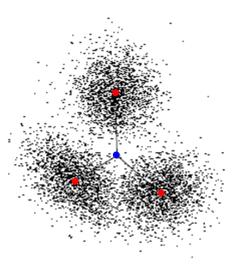
- offspring are centered around each parent; it assigns more probability for an offspring to remain closer to the parents than away from parents,
- the complexity for creating one offspring is  $O(\mu)$ .

PCX creates offspring as follows

$$\overrightarrow{o} = \overrightarrow{p} + w_1 |\overrightarrow{d}^{(p)}| + \sum_{i=1, i \neq p}^{\mu} w_2 \overline{D} \overrightarrow{e}^{(i)}$$

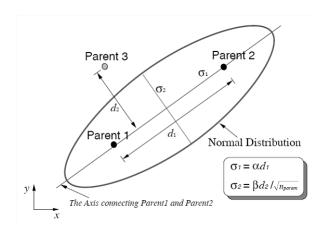
where

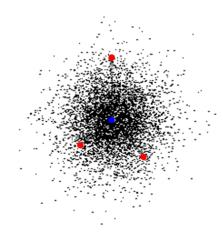
- $\blacksquare$   $\overrightarrow{m}$  is the mean vector of  $\mu$  parents
- $\overrightarrow{p}$  is a parent chosen as the basis for the offspring
- $\overrightarrow{d}^{(p)} = \overrightarrow{p} \overrightarrow{m}$
- $\blacksquare$   $\overline{D}$  is the average over all perpendicular distances calculated from each of the other  $(\mu-1)$  parents to the vector  $\overrightarrow{d}^{\,(p)}$
- $w_1$  and  $w_2$  are zero-mean normally distributed variables.



### **Unimodal Normal Distribution Crossover**

UNDX creates offspring solutions around the center of mass of  $\mu$  parents



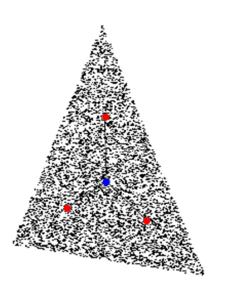


- ullet generates offspring by using the normal distribution which is defined by  $\mu$  parents,
- offspring are generated around the mean; the probability of creating an offspring away from the mean vector reduces,
- preserves the correlation among parameters well efficiently solves problems with strong epistasis among parameters,
- there can be some areas where it cannot generate offspring from a given initial population,
- the complexity of creating one offspring is  $O(\mu^2)$ ,
- has difficulties in finding the optimal point(s) near the boundaries.

## **Simplex Crossover**

## Simplex Crossover (SPX)

- generates offspring around the mean of parents parents,
- offspring restricted within a region that is a simplex  $\gamma = \sqrt{\mu + 1}$  times bigger than the parent simplex,
- assigns a uniform probability distribution for creating solutions in the restricted area,
- the computational complexity for creating one offspring is  $O(\mu)$ .

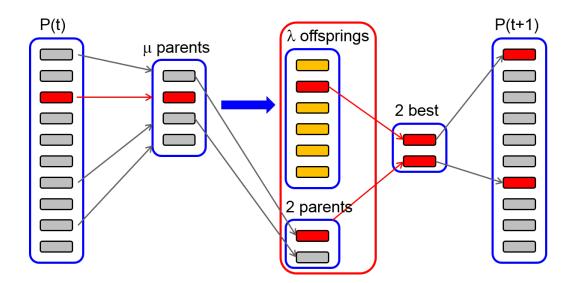


# Generalized Generation Gap (G3) Evolutionary Algorithm Model

#### Need for higher sampling rate

**G3** – elite-preserving, steady-state, and computationally fast.

- 1. From the population P(t), select the best parent and  $(\mu 1)$  other parents randomly.
- 2. Generate  $\lambda$  offspring from  $\mu$  parents using a recombination scheme.
- 3. Choose two parents at random from  $\mu$  parents.
- 4. Form a combined subpopulation of chosen two parents and  $\lambda$  offspring, choose the best two solutions and replace the chosen two parents with these solutions.

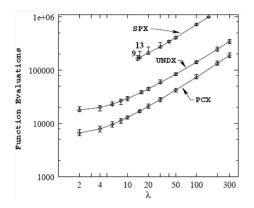


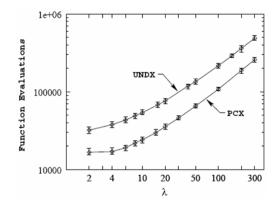
# Generalized Generation Gap (G3) Evolutionary Algorithm Model

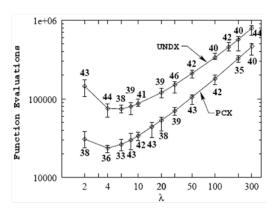
**G3** – elite-preserving, steady-state, and computationally fast.

- 1. From the population P(t), select the best parent and  $(\mu 1)$  other parents randomly.
- 2. Generate  $\lambda$  offspring from  $\mu$  parents using a recombination scheme.
- 3. Choose two parents at random from  $\mu$  parents.
- 4. Form a combined subpopulation of chosen two parents and  $\lambda$  offspring, choose the best two solutions and replace the chosen two parents with these solutions.

Comparisons of UNDX, SPX and PCX with the G3 model on Ellipsoidal, Schwefel's, and Generalized Rosenbrock's functions and n=20.







©Deb K. et al.: Real-Coded Evolutionary Algorithms with Parent-Centric Recombination.

## **Evolution Strategies**

#### **Evolution strategy (ES)**

- **ES** is an optimization technique based on ideas of **adaptation and evolution**.
- ES uses natural problem-dependent representations, and primarily mutation and selection as search operators.
- For real-valued search spaces, mutation is performed by adding a normally distributed random value to each vector component.
- The step size is often governed by self-adaptation. Individual step sizes for each coordinate or correlations between coordinates are either governed by self-adaptation or by covariance matrix adaptation (CMA-ES).

#### Two Membered ES

The two membered ES - (1+1)-ES – is a simple mutation-selection scheme ES.

- Population consists of one parent individual and one descendant, created by means of adding normally distributed random numbers.
- The better of both individuals then serves as the ancestor of the following iteration.

Formal definition of (1+1)-ES:

$$(1+1) - ES = (P^0, m, s, c_d, c_i, f, g, t)$$

where

$$P^0 = (x^0, \sigma^0) \in I$$
 population  $x^0$  is a vector of solution parameters  $\sigma^0$  is vector of standard deviations  $I = R^n \times R^n$   $m : I \to I$  mutation operator  $s : I \times I \to I$  selection operator  $c_d, c_i \in R$  step-size control  $f : R^n \to R$  objective function  $g_j : R^n \to R$  constraint functions  $t : I \times I \to \{0,1\}$  termination criterion

### Two Membered ES cnd.

 $P^0$  denotes the initial population consisting of a single parent.

Mutation operator m is applied to all components of the object parameter  $x^t$ . It is realized by adding normally distributed random numbers

$$x^{\prime t} = x^t + \mathbf{N}_0(\sigma^t)$$

where  $N_0$  denotes a vector of independent Gaussian random numbers with zero mean and standard deviations  $\sigma_i^t = (i = 1, \dots, n)$ .

Selection operator just determines the fitter individual of the parent  $p^t$  and offspring  $o^t$  to become the parent for the next generation.

- $\bullet$   $o^t = m(p^t) = (x'^t, \sigma^t)$  survives if it is not worse than the parent  $p^t$ ,
- $p^t = (x^t, \sigma^t)$  survives otherwise.

Termination criterion can be

- elapsed CPU time,
- elapsed number of generations,
- absolute or relative progress per generation, etc.

# Step-size Control: 1/5 Success Rule

Progress rate  $(\varphi)$  – a quotient of the distance covered towards the optimum and the number of trials needed for this distance.

- Calculated for two model functions
  - Corridor model:  $f_1(x) = F(x_1) = c_0 + c_1 x_1$ ,  $\forall i \in \{2, ..., n\} : -b/2 \le x_i \le b/2$ Sphere model:  $f_2(x) = \sum_{i=1}^n x_i^2$
- lacksquare In both cases the step size  $\sigma_i^{opt}$  is inversely proportional to the number of object variables n.
- ullet The maximum progress rate is also inversely proportional to n.

For the optimum step sizes Rechenberg obtained the following probabilities for a successful mutation,

- $p_1^{opt} = 1/(2e) \approx 0.184$
- $p_2^{opt} \approx 0.270$

1/5 success rule – In order to obtain nearly optimal (local) performance of the (1+1)-ES in real-valued search spaces, tune the mutation strength in such a way that the (measured) success rate is about 1/5.

• If it is greater than 1/5, increase the variance; if it is less, decrease the mutation variance.

# **Extended Mutation Operator**

Mutation operator m modifies all components of the object parameter  $x^t$  according to

$$x^{\prime t} = x^t + \mathbf{N}_0(\sigma^t)$$

while the  $\sigma^t$  variances are dynamically adjusted (all at the same time) according to

$$\begin{array}{lll} \sigma^{t+n} &=& c_d \cdot \sigma^t & \text{, if } p_s^t < 1/5 \\ & c_i \cdot \sigma^t & \text{, if } p_s^t > 1/5 \\ & \sigma^t & \text{, if } p_s^t = 1/5 \end{array}$$

where  $p_s^t$  is the frequency of successful mutations, measured over 10n trials.

Recommended setup:

- increase step size:  $c_i = 1/0.82$ ,
- decrease step size:  $c_d = 0.82$ ,
- adjustment takes place every n mutations.

### **Multimembered EAs**

Multimembered  $(\mu+1)$ -ES where  $\mu>1$  parents can participate in the generation of one offspring:

$$(\mu + 1) - ES = (P^0, \mu, r, m, s, c_d, c_i, f, g, t)$$

where

#### Multimembered EAs: Recombination

#### Recombination operator r:

$$r(P^{t}) = a' = (x', \sigma') \in I, x' \in R^{n}, \sigma' \in R^{n}$$

$$x'_{i} = x_{a,i}, \chi \leq 0.5 \quad \forall i \in 1, ..., n$$

$$x_{b,i}, \chi > 0.5$$

$$\sigma'_{i} = \sigma_{a,i}, \chi \leq 0.5 \quad \forall i \in 1, ..., n$$

$$\sigma_{b,i}, \chi > 0.5$$

where  $a=(x_a,\sigma_a)$  and  $b=(x_b,\sigma_b)$  are two parents.

#### Remarks:

■ Parents a and b are determined randomly (uniform).

All individuals in the population have the same mating probabilities.

- $\blacksquare$  A uniform random variable  $\chi$  is sampled anew for each component of the vectors  $x^{'}$  and  $\sigma^{'}.$
- ullet The selection operator s removes the least fit individual out of  $\mu$  parents plus 1 offspring from the population.

# $(\mu + \lambda)$ - and $(\mu, \lambda)$ - Multimembered EAs

Motivation to extend  $(\mu + 1)$ -ES to  $(\mu + \lambda)$ - and  $(\mu, \lambda)$ -ES:

- To enable self-adaptation of strategic parameters standard deviations.
  - The  $\sigma^t$  parameters are considered a part of the individual's genome that is subject to recombination and mutation.
  - Those individuals with better adjusted strategy parameters are expected to perform better.
- To make use of parallel computers, where several newly generated individuals can be processed (evaluated) simultaneously.

# $(\mu + \lambda)$ -EA

#### Model:

- ullet  $\mu$  parents produce  $\lambda$  offspring which are reduced again to the  $\mu$  parents for the next generation.
- ullet Selection s operates on joined set of parents and offspring. Some very well adapted individuals can survive in the population for ever.

#### Deficiencies:

Can get stuck at some good location if the internal parameter setting becomes unsuitable to jump to the new field of possible improvements (dynamic opt. problems, problems with noisy objective function).

# $(\mu, \lambda)$ -EA

#### Model:

- ullet  $\mu$  parents produce  $\lambda$  offspring, only the offspring undergo selection;
- Forgetting principle the lifetime of every individual is limited to one generation if  $\mu < \lambda$ .

#### Remarks:

- Possible short phases of recession.
- It avoids long stagnation phases due to misadapted strategy parameters.

# **Handling the Internal Strategy Parameters**

Internal strategy parameters  $\sigma^t$  are part of the genetic information of an individual

$$a_i^t = (x^t, \sigma^t) \in I$$

which are not controlled by the 1/5 success rule anymore.

**Mutation operator works on both** the  $x^t$  and  $\sigma^t$ :

$$\begin{array}{lll} a_i'^t &=& r(P^t) & // \text{ recombination} \\ m(a_i'^t) &=& a_i''^t = (x''^t, \sigma''^t) \\ \sigma''^t &=& \sigma'^t exp(\mathbf{N}_0(\triangle \ \sigma)) \ // \text{ strategy mutation} \\ x''^t &=& x'^t + \mathbf{N}_0(\sigma''^t) \ // \text{ mutation of solution params } x \end{array}$$

#### Remark:

• Unsuitable  $\sigma''^t$  are removed by means of selection.

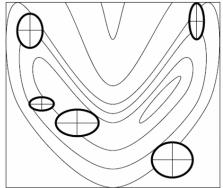
### **ES** with Correlated Mutations

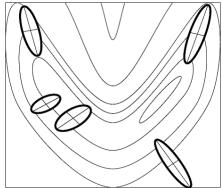
#### Motivation:

- With dedicated  $\sigma_i$  for each object variable  $x_i$  preferred directions of search can be established only along the axes of the coordinate system.
- The optimum rate of progress is achieved by chance when suitable mutations coincide.
  The trajectory of the population through the search space does not straightforwardly follow the gradient.

Correlated mutations – additional strategy vector  $\theta$  is added to the genotype  $a_i^t = (x^t, \sigma^t, \theta^t)$ , where a set of inclination angles  $\theta \in R^w$  define n-dim rotating hyperellipsoids, w = n(n-1)/2.

Ellipsoid contours of equal prob. density to place an offspring for simple and correlated mutations.





©B ack T. et al.: A Survey of Evolution Strategies.

### **Differential Evolution**

Differential Evolution (DE) – an evolutionary algorithm named after a special kind of differential operator, which is used to create new offspring.

DE evolves a population of N D-dimensional variable vectors. The ith vector of the population at time t is represented as

$$\overrightarrow{X}_{i}(t) = [x_{i,1}(t), x_{i,2}(t), \dots, x_{i,D}(t)]$$

Values  $x_{i,j}$  can be restricted within certain range between  $x_j^L$  and  $x_j^U$ .

## **Differential Evolution: Differential Operator**

Differential Operator – generates an offspring for each population member  $\overrightarrow{X}_i(t)$  as follows

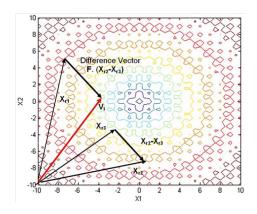
1. Donor vector  $\overrightarrow{V}_i(t)$  is created.

DE/rand/1 scheme:

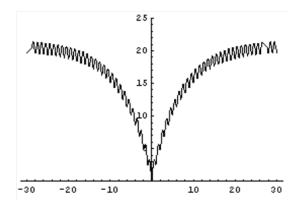
- Three population members (indexed  $r_1$ ,  $r_2$  and  $r_3$ ) are chosen at random.
- A difference between any two of the three vectors scaled by value F is added to the third one.

$$v_{i,j}(t) = x_{r1,j}(t) + F \cdot (x_{r2,j}(t) - x_{r3,j}(t)), \text{ for } j = 1 \dots D.$$

Illustration of Creation of the Donor Vector in 2D



©Das S. et al.: Particle Swarm Optimization and Differential Evolution Algorithms.



Projection of the Ackley's function

### **Differential Evolution: Crossover**

2. Crossover – to increase a diversity of the population.

The donor vector exchanges its components with the target vector  $\overrightarrow{X}_i(t)$  under one of the following schemes:

#### Exponential scheme

L – the number of components the donor vector will contribute to the target.

n – a starting point in the target vector  $\overrightarrow{X}_i(t)$ , from where the crossover with the donor vector  $\overrightarrow{V}_i(t)$  starts.

CR – crossover parameter determining  $Prob(L \ge m) = (CR)^{m-1}$ , for any m > 0.

Trial vector  $\overrightarrow{U}_i(t)$  (offspring) is formed as follows

$$u_{i,j}(t) = v_{i,j}(t)$$
 for  $j = \langle n \rangle_D, \langle n+1 \rangle_D, \dots, \langle n+L-1 \rangle_D,$   
=  $x_{i,j}(t)$  for remaining components

where  $<>_D$  denote a modulo function with modulus D.

■ Binomial scheme. The crossover is performed on each of the D variables whenever a randomly picked number between 0 and 1 is within CR value

$$u_{i,j}(t) = v_{i,j}(t)$$
 if rand(0,1)<  $CR$   
=  $x_{i,j}(t)$  otherwise

### **Differential Evolution: Selection**

3. **Selection** – determines which one of the target vector and the trial vector will survive in the next generation.

$$\overrightarrow{X}_i(t+1) = \overrightarrow{U}_i(t) \text{ if } f(\overrightarrow{U}_i(t)) \leq f(\overrightarrow{X}_i(t))$$
 
$$= \overrightarrow{X}_i(t) \text{ otherwise}$$

The population either gets better (w.r.t. fitness function) or remains constant.

### **Differential Evolution: Pseudocode**

```
Initialize population;
Evaluate fitness;
For i = 0 to max-iteration do
Begin
Create Difference-Offspring;
Evaluate fitness;
If an offspring is better than its parent
Then replace the parent by offspring in the next generation;
End If;
End For;
End.
```

### **Various DE schemes** – DE/x/y

- 1. x type of the vector to be perturbed by the differential operator (rand/best).
- 2. y number of difference vectors considered for perturbation of x.

## Reading

- Deb K. et al.: Real-Coded Evolutionary Algorithms with Parent-Centric Recombination, 2001
   http://www.iitk.ac.in/kangal/papers/tech\_rep2001003.pdf
- Herrera F. et al.: Tackling Real-Coded Genetic Algorithms: Operators and Tools for Behavioural Analysis, 2004

http://www.springerlink.com/content/nh44700284m80307/

 Ono I. et al.: A Robust Real-Coded Genetic Algorithm using Unimodal Normal Distribution Crossover Augmented by Uniform Crossover: Effects of Self-Adaptation of Crossover Probabilities, 1999

http://www.cs.bham.ac.uk/wbl/biblio/gecco1999/Ono\_gecco99.pdf

■ Bäck T. et al.: A Survey of Evolution Strategies, 1991

http://www6.uniovi.es/pub/EC/GA/papers/icga91.ps.gz

 Das S. et al.: Particle Swarm Optimization and Differential Evolution Algorithms: Technical Analysis, Applications and Hybridization Perspectives, 2008

http://www.softcomputing.net/aciis.pdf