# Описание множества операторов для алгоритмов оптимизации. v. 1.5

А. Б. Сергиенко

9 февраля 2014 г.

**Аннотация**

В данном документе дано собрано множество всяких операторов, которые используются автором в своих исследований. В первую очередь это операторы модификаций генетического алгоритма, а также классические операторы алгоритма.

## Содержание

# 1 Введение

Это своеобразная «свалка» операторов, которые используются автором. На данный документ можно ссылаться в своих работах, чтобы указать, что та или иная модификация операторов подробно описана в этом документе. Тут нет исследований эффективности алгоритмов с данными операторами — это задача иных проектов. Здесь представлено только описание операторов.

Например, в работе может быть написано следующее: «Модифицированный генетический алгоритм основан на стандартном генетическом алгоритме (https://github.com/Harrix/Standard-Genetic-Algorithm). Предложенный алгоритм отличается только оператором скрещивания, и вместо двухточечного скрещивания используется двухточечное скрещивание с возможностью точек разрыва по краям хромосомы (подробное описание смотрите в https://github.com/Harrix/HarrixSetOfOperatorsAlgorithms)».

Данный документ представляет его версию 1.0 от 9 февраля 2014 г.

Последнюю версию документа можно найти по адресу:

https://github.com/Harrix/HarrixSetOfOperatorsAlgorithms

С автором можно связаться по адресу sergienkoanton@mail.ru или http://vk.com/harrix.

Сайт автора, где публикуются последние новости: http://blog.harrix.org/, а проекты располагаются по адресу http://harrix.org/.

# 2 Условные обозначения

$a \in A$ — элемент $a$ принадлежит множеству $A$.

$\bar{x}$ — обозначение вектора.

$\arg f(x)$ — возвращает аргумент $x$, при котором функция принимает значение $f(x)$.

$Random(X)$ — случайный выбор элемента из множества $X$ с равной вероятностью.

$Random\left(\{x^i \mid p^i\}\right)$ — случайный выбор элемента $x^i$ из множества $X$, при условии, что каждый элемент $x^i \in X$ имеет вероятность выбора равную $p^i$, то есть это обозначение равнозначно предыдущему.

$random(a, b)$ — случайное действительное число из интервала $[a; b]$.

$int(a)$ — целая часть действительного числа $a$.

$\mu(X)$ — мощность множества $X$.

**Замечание.** Оператор присваивания обозначается через знак «=», так же как и знак равенства.

**Замечание.** Индексация всех массивов в документе начинается с 1. Это стоит помнить при реализации алгоритма на C-подобных языках программирования, где индексация начинается с нуля.

**Замечание.** Вызывание трех функций: $Random(X)$, $Random\left(\{x_i \mid p_i\}\right)$, $random(a, b)$ – происходит каждый раз, когда по ходу выполнения формул, они встречаются. Если формула ите-

рационная, то нельзя перед ее вызовом один раз определить, например, $random(a, b)$ как константу и потом её использовать на протяжении всех итераций неизменной.

**Замечание.** Надстрочный индекс может обозначать как возведение в степень, так и индекс элемента. Конкретное обозначение определяется в контексте текста, в котором используется формула с надстрочным индексом.

**Замечание.** Если у нас имеется множество векторов, то подстрочный индекс обозначает номер компоненты конкретного вектора, а надстрочный индекс обозначает номер вектора во множестве, например, $\bar{x}^i \in X$ $(i = \overline{1, N})$, $\bar{x}^i_j \in \{0; 1\}$, $(j = \overline{1, n})$. В случае, если вектор имеет свое обозначение в виде подстрочной надписи, то компоненты вектора проставляются за скобками, например, $(\bar{x}_{max})_j = 0$ $(j = \overline{1, n})$.

**Замечание.** При выводе матриц и векторов элементы могут разделяться как пробелом, так и точкой с запятой, то есть обе записи $\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}^{\mathrm{T}}$ и $(1; 1; 1; 1; 1; 1; 1; 1; 1)^{\mathrm{T}}$ допустимы.

**Замечание.** При выводе множеств элементы разделяются только точкой с запятой, то есть допустима только такая запись: $\{1; 1; 1; 1; 1; 1; 1; 1; 1\}^{\mathrm{T}}$.

# 3 Операторы селекции

Селекция — оператор случайного выбора одного индивида из популяции, основываясь на значениях функции пригодности всех индивидов текущей популяции, для использования его в операторе скрещивания. При этом вероятность выбора у индивидов с более высокой пригодностью выше, чем у индивидов с более низкой пригодностью.

## 3.1 Пропорциональная селекция

Идентификатор: **ProportionalSelection**.

Вероятность выбора элемента пропорциональна значению пригодности индивида. Данный вид селекции может работать только с неотрицательными значениями пригодности.

Пропорциональная селекция определяется формулой:

$$Selection\left(Population, Fitness, DataOfSel\right) = Random\left(\left\{\bar{x}^i | p^i\right\}\right), \tag{1}$$

$$p^i = \begin{cases} \dfrac{f_{fit}\left(\bar{x}^i\right)}{\sum_{j=1}^{N} f_{fit}\left(\bar{x}^j\right)}, \text{ если } \exists f_{fit}\left(\bar{x}^k\right) \neq 0 \left(k = \overline{1, N}\right); \\ \dfrac{1}{N}, \text{ иначе.} \end{cases} \tag{2}$$

где $\bar{x}^i \in Population, i = \overline{1, N}$.

Как видим, формула определения вероятности выбора индивида имеет составной вид. Второе условие предназначено для маловероятного случая, когда в популяции все индивиды будут иметь пригодность равную нулю.

**Пример.** Пусть $Fitness = \{0,5; 0,2; 0,1; 0,6; 0,2; 0,4\}$. Тогда вероятности выбора индивидов равны:

$$p_1 = \frac{0,5}{0,5+0,2+0,1+0,6+0,2+0,4} = 0,25;$$
$$p_2 = \frac{0,2}{0,5+0,2+0,1+0,6+0,2+0,4} = 0,1;$$
$$p_3 = \frac{0,1}{0,5+0,2+0,1+0,6+0,2+0,4} = 0,05;$$
$$p_4 = \frac{0,6}{0,5+0,2+0,1+0,6+0,2+0,4} = 0,3;$$
$$p_5 = \frac{0,2}{0,5+0,2+0,1+0,6+0,2+0,4} = 0,1;$$
$$p_6 = \frac{0,4}{0,5+0,2+0,1+0,6+0,2+0,4} = 0,2.$$

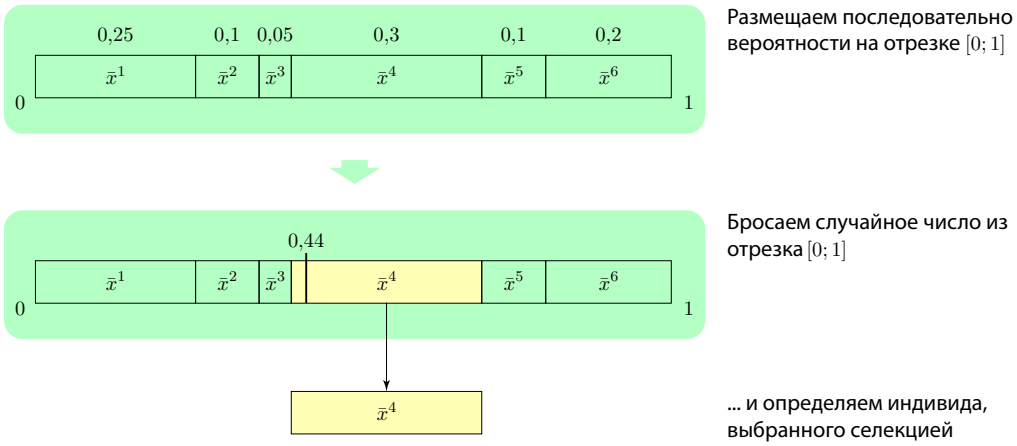


Рисунок 1. Механизм работы пропорциональной селекции

$DataOfSel$ также не содержит каких-либо параметров относительно данного типа селекции.

Нет ограничений на множество задач оптимизации, которые может решать алгоритм оптимизации с данной селекцией.

В библиотеке **HarrixMathLibrary** данная селекция реализуется через функции **MHL_ProportionalSelection**, **MHL_ProportionalSelectionV2** и **MHL_ProportionalSelectionV3**:

https://github.com/Harrix/HarrixMathLibrary

## 3.2 Ранговая селекция

Идентификатор: **RankSelection**.

Работает не с массивом пригодностей напрямую, а массивом нормированных рангов, присваиваемых индивидам на основе значений пригодности. Используется функция, которая проставляет ранги для элементов несортированного массива пригодностей, то есть номера, начиная с 1, в отсортированном массиве. Если в массиве есть несколько одинаковых элементов, то

ранги им присуждаются как среднеарифметические ранги этих элементов в отсортированном массиве. Если это не сделать, то вероятность выбора индивидов одинаковых по функции пригодности будет не равна друг другу, что противоречит идеи оператора селекции. Далее для выбора индивидов используется пропорциональная селекция, работающая с массивом рангов.

Значит, ранговая селекция определяется формулой:

$$Selection\left(Population, Fitness, DataOfSel\right) = Random\left(\left\{\bar{x}^i | p^i\right\}\right), \tag{3}$$

$$p^i = \frac{Rank\left(f_{fit}\left(\bar{x}^i\right)\right)}{\sum_{j=1}^{N} Rank\left(f_{fit}\left(\bar{x}^j\right)\right)}, \tag{4}$$

$$Rank\left(f_{fit}\left(\bar{x}^i\right)\right) = \frac{\sum_{j=1}^{N} NumberOfSorting\left(f_{fit}\left(\bar{x}^i\right), Fitness\right) \cdot S\left(f_{fit}\left(\bar{x}^i\right), f_{fit}\left(\bar{x}^j\right)\right)}{\sum_{j=1}^{N} S\left(f_{fit}\left(\bar{x}^i\right), f_{fit}\left(\bar{x}^j\right)\right)}, \tag{5}$$

$$S\left(f_{fit}\left(\bar{x}^i\right), f_{fit}\left(\bar{x}^j\right)\right) = \begin{cases} 1, \text{ если } f_{fit}\left(\bar{x}^i\right) = f_{fit}\left(\bar{x}^j\right); \\ 0, \text{ если } f_{fit}\left(\bar{x}^i\right) \neq f_{fit}\left(\bar{x}^j\right). \end{cases} \tag{6}$$

где $\bar{x}^i \in Population$, $i = \overline{1, N}$.

$NumberOfSorting\left(f_{fit}\left(\bar{x}^i\right), Fitness\right)$ — функция, возвращающая номер элемента $f_{fit}\left(\bar{x}^i\right)$ в отсортированном массиве $Fitness$ в порядке возрастания.

Формула (5) подсчитывает средние арифметические ранги при условии, что в массиве $Fitness$ могут встречаться одинаковые элементы.

**Пример.** Пусть $Fitness = \{0,5; 0,2; 0,1; 0,6; 0,2; 0,4\}$. Тогда вероятности выбора индивидов равны:

$$p_1 = \frac{5}{5 + 2,5 + 1 + 6 + 2,5 + 4} = 0,238;$$
$$p_2 = \frac{2,5}{5 + 2,5 + 1 + 6 + 2,5 + 4} = 0,119;$$
$$p_3 = \frac{1}{5 + 2,5 + 1 + 6 + 2,5 + 4} = 0,047;;$$
$$p_4 = \frac{6}{5 + 2,5 + 1 + 6 + 2,5 + 4} = 0,286;$$
$$p_5 = \frac{2,5}{5 + 2,5 + 1 + 6 + 2,5 + 4} = 0,119;$$
$$p_6 = \frac{4}{5 + 2,5 + 1 + 6 + 2,5 + 4} = 0,190.$$

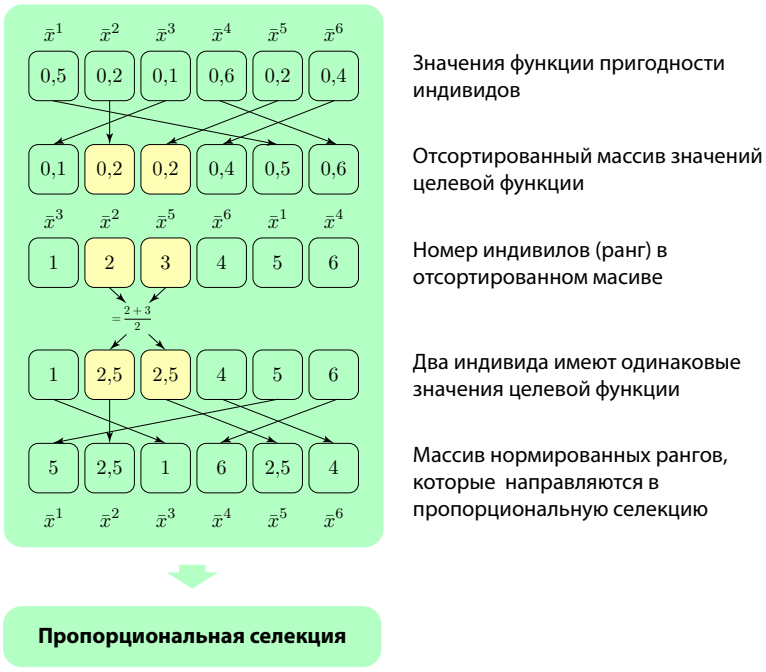


Рисунок 2. Механизм работы ранговой селекции

$DataOfSel$ также не содержит каких-либо параметров относительно данного типа селекции.

Нет ограничений на множество задач оптимизации, которые может решать алгоритм оптимизации с данной селекцией.

В библиотеке **HarrixMathLibrary** данная селекция реализуется через функцию **MHL_RankSelection**:

https://github.com/Harrix/HarrixMathLibrary

## 3.3 Турнирная селекция

Идентификатор: **TournamentSelection**.

Из популяции с равной вероятностью выбираются индивиды в количестве $T$ (размер турнира), где $2 \leq T \leq PopulationSize$ ($PopulationSize$ — размер популяции). При этом каждый индивид может попасть в группу (турнир) только один раз (турнирная селекция без возвращения). Из данной группы выбирается индивид с наибольшей пригодностью.

Значит, турнирная селекция определяется формулой:

$$Selection\left(Population, Fitness, DataOfSel\right) = \arg\max_{\bar{x} \in H} f_{fit}\left(\bar{x}\right), \text{где} \tag{7}$$
$$H = \left\{h^i | h_i = Random\left(Population / \left(\left\{h^1\right\} \cup \left\{h^2\right\} \cup \ldots \cup \left\{h^{i-1}\right\}\right)\right)\right\}, i = \overline{1,T}.$$

Турнирная селекция добавляет в $DataOfSel$ дополнительный параметр —- размер турнира $T$. Обычно выбирают значение этого параметра равное $T = 2$.

**Пример.** Массив возьмем тот же, что и в предыдущих примерах $Fitness = \{0,5; 0,2; 0,1; 0,6; 0,2; 0,4\}$. Размер турнира равен $T = 3$. Пример работы оператора показан на рисунке:

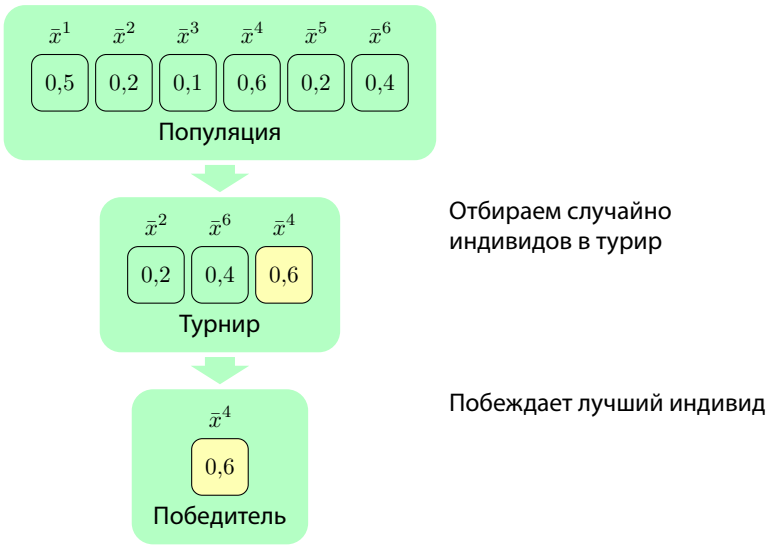


Рисунок 3. Механизм работы турнирной селекции

$$DataOfSel = \begin{pmatrix} TypeOfSel \\ T \end{pmatrix}. \tag{8}$$

Нет ограничений на множество задач оптимизации, которые может решать алгоритм оптимизации с данной селекцией.

В библиотеке **HarrixMathLibrary** данная селекция реализуется через функцию **MHL_TournamentSelection**:

https://github.com/Harrix/HarrixMathLibrary

## 3.4 Турнирная селекция с возвращением

Идентификатор: **TournamentSelectionWithReturn**.

Из популяции с равной вероятностью выбираются индивиды в количестве $T$ (размер турнира), где $2 \leq T \leq PopulationSize$ ($PopulationSize$ — размер популяции). При этом каждый индивид может попасть в группу (турнир) сколько угодно раз (турнирная селекция с возвращением). То есть после того, как мы выбрали индивида, мы запоминаем его номер и возвращаем обратно в популяцию, где он на равных правах может попасть заново в этот турнир. Из данной выбранной группы выбирается индивид с наибольшей пригодностью.

Технически данную селекцию проще организовать путем генерации целых чисел по равномерному закону распределению от 1 до $PopulationSize$ (на C++ от 0 до $PopulationSize - 1$) в количестве $T$ штук. При этом в полученной выборке могут попадаться одинаковые числа. Данные числа будут обозначать номера индивидов в популяции. И затем выберем лучшего индивида.

Значит, турнирная селекция определяется формулой:

$$\begin{gathered} Selection\left(Population, Fitness, DataOfSel\right) = \arg\max_{\bar{x} \in H} f_{fit}\left(\bar{x}\right), \text{где} \\ H = \left\{h^i | h_i = Random\left(Population\right)\right\}, i = \overline{1, T}. \end{gathered} \tag{9}$$

Турнирная селекция с возвращением добавляет в $DataOfSel$ дополнительный параметр –– размер турнира $T$. Обычно выбирают значение этого параметра равное $T = 2$.

$$DataOfSel = \begin{pmatrix} TypeOfSel \\ T \end{pmatrix}. \tag{10}$$

Нет ограничений на множество задач оптимизации, которые может решать алгоритм оптимизации с данной селекцией.

В библиотеке **HarrixMathLibrary** данная селекция реализуется через функцию **MHL_TournamentSelectionWithReturn**:

https://github.com/Harrix/HarrixMathLibrary

# 4 Операторы скрещивания

**Скрещивание (кроссовер)** — оператор случайного формирования нового индивида из двух выбранных родителей с сохранением признаков обоих родителей.

## 4.1 Одноточечное скрещивание

Идентификатор: **SinglepointCrossover**.

Данный оператор скрещивания используется для бинарных векторов.

Пусть имеется два родителя (родительские хромосомы) $\overline{Parent}^1$ и $\overline{Parent}^2$. В случайном месте происходит разрыв между двумя позициями генов в обеих хромосомах. После этого хромосомы обмениваются частями, в результате чего образуются два потомка. Из них выбирается случайно один потомок, который и передается в качестве результата оператора скрещивания. То есть скрещивание происходит по формулам:

$$Crossover\left(\overline{Parent}^1, \overline{Parent}^2, DataOfCros\right) = Random\left(\left\{\overline{Offspring}^1; \overline{Offspring}^2\right\}\right), \tag{11}$$

$R = Random\left(\{2; 3; \ldots; n\}\right);$

$\overline{Offspring}_i^1 = \overline{Parent}_i^1, i = \overline{1, R-1};$

$\overline{Offspring}_i^1 = \overline{Parent}_i^2, i = \overline{R, n};$

$\overline{Offspring}_i^2 = \overline{Parent}_i^2, i = \overline{1, R-1};$

$\overline{Offspring}_i^2 = \overline{Parent}_i^1, i = \overline{R, n};$

$\overline{Offspring}^1 \in X, \overline{Offspring}^2 \in X.$

**Пример.** Для всех видов скрещивания будем использовать двух родителей: $\overline{Parent}^1 = (0; 1; 0; 1; 1; 1; 0; 0)^{\mathrm{T}}$ и $\overline{Parent}^2 = (1; 1; 0; 0; 1; 0; 1)^{\mathrm{T}}$. Одноточечное скрещивание показано на рисунке:

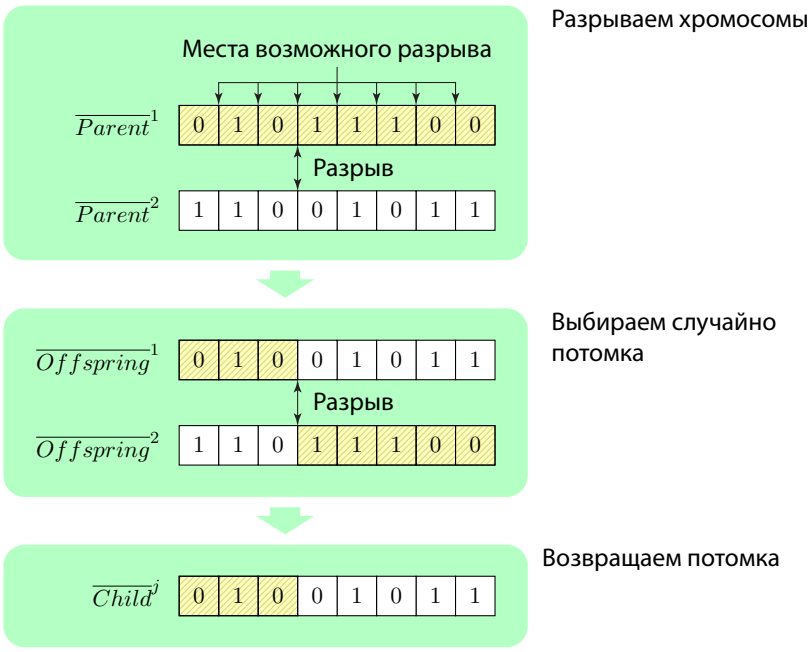


Рисунок 4. Механизм работы одноточечного скрещивания

$DataOfCros$ не содержит каких-либо параметров относительно данного типа скрещивания.

В библиотеке **HarrixMathLibrary** данная селекция реализуется через функцию **TMHL_SinglepointCrossover**:

https://github.com/Harrix/HarrixMathLibrary

## 4.2 Одноточечное скрещивание с возможностью полного копирования одного из родителей

Идентификатор: **SinglepointCrossoverWithCopying**.

Данный оператор скрещивания используется для бинарных векторов.

Отличается от стандартного одноточечного скрещивания тем, что точки разрыва могут происходить по краям родителей, что может привести к полному копированию родителя.

Пусть имеется два родителя (родительские хромосомы) $\overline{Parent}^1$ и $\overline{Parent}^2$. В случайном месте происходит разрыв между двумя позициями генов в обеих хромосомах. После этого хромосомы обмениваются частями, в результате чего образуются два потомка. Из них выбирается случайно один потомок, который и передается в качестве результата оператора скрещивания. То есть скрещивание происходит по формулам:

$$Crossover\left(\overline{Parent}^1, \overline{Parent}^2, DataOfCros\right) = Random\left(\left\{\overline{Offspring}^1; \overline{Offspring}^2\right\}\right), \quad (12)$$

$R = Random\left(\{1; 3; \ldots; n+1\}\right);$

$\overline{Offspring}_i^1 = \overline{Parent}_i^1, i = \overline{1, R-1};$

$\overline{Offspring}_i^1 = \overline{Parent}_i^2, i = \overline{R, n};$

$\overline{Offspring}_i^2 = \overline{Parent}_i^2, i = \overline{1, R-1};$

$\overline{Offspring}_i^2 = \overline{Parent}_i^1, i = \overline{R, n};$

$\overline{Offspring}^1 \in X, \overline{Offspring}^2 \in X.$

**Пример.** Для всех видов скрещивания будем использовать двух родителей: $\overline{Parent}^1 = (0; 1; 0; 1; 1; 1; 0; 0)^{\mathrm{T}}$ и $\overline{Parent}^2 = (1; 1; 0; 0; 1; 0; 1)^{\mathrm{T}}$. Одноточечное скрещивание с возможностью полного копирования одного из родителей показано на рисунке:

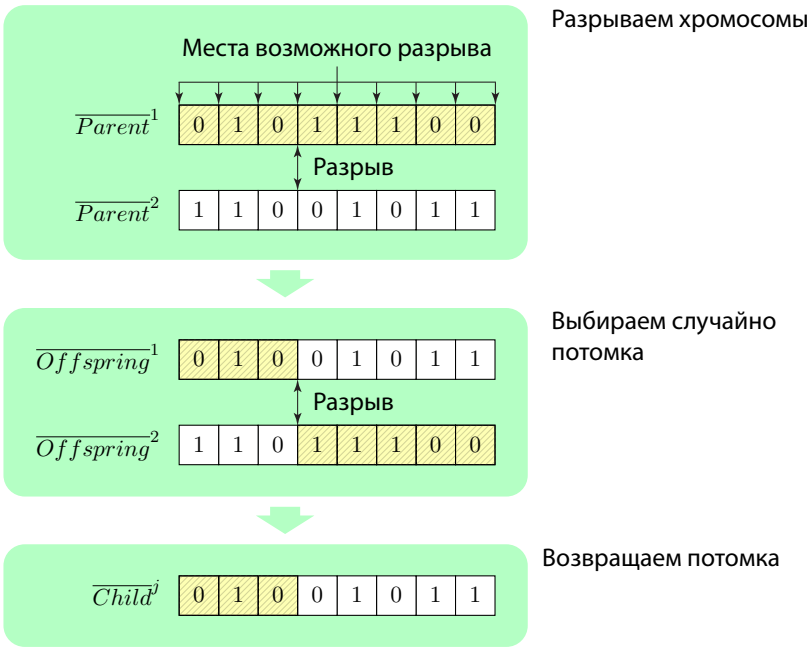


Рисунок 5. Механизм работы одноточечного скрещивания с возможностью полного копирования одного из родителей

$DataOfCros$ не содержит каких-либо параметров относительно данного типа скрещивания.

В библиотеке **HarrixMathLibrary** данная селекция реализуется через функцию **TMHL_SinglepointCrossoverWithCopying**:

https://github.com/Harrix/HarrixMathLibrary

## 4.3 Одноточечное скрещивание для вещественных векторов

Идентификатор: **SinglepointCrossoverForReal**.

Данный оператор скрещивания используется для вещественных векторов.

По сути ничем не отличается от SinglepointCrossover, кроме типа векторов, на котором работает оператор.

Пусть имеется два родителя (родительские хромосомы) $\overline{Parent}^1$ и $\overline{Parent}^2$. В случайном месте происходит разрыв между двумя позициями генов в обеих хромосомах. После этого хромосомы обмениваются частями, в результате чего образуются два потомка. Из них выбирается случайно один потомок, который и передается в качестве результата оператора скрещивания. То есть скрещивание происходит по формулам:

$$Crossover\left(\overline{Parent}^1, \overline{Parent}^2, DataOfCros\right) = Random\left(\left\{\overline{Offspring}^1; \overline{Offspring}^2\right\}\right), \quad (13)$$

$$R = Random\left(\{2; 3; \ldots; n\}\right);$$

$$\overline{Offspring}_i^1 = \overline{Parent}_i^1, i = \overline{1, R-1};$$

$$\overline{Offspring}_i^1 = \overline{Parent}_i^2, i = \overline{R, n};$$

$$\overline{Offspring}_i^2 = \overline{Parent}_i^2, i = \overline{1, R-1};$$

$$\overline{Offspring}_i^2 = \overline{Parent}_i^1, i = \overline{R, n};$$

$$\overline{Offspring}^1 \in X, \overline{Offspring}^2 \in X.$$

$DataOfCros$ не содержит каких-либо параметров относительно данного типа скрещивания.

В библиотеке **HarrixMathLibrary** данная селекция реализуется через функцию **MHL_SinglepointCrossoverForReal**:

https://github.com/Harrix/HarrixMathLibrary

## 4.4 Двухточечное скрещивание

Идентификатор: **TwopointCrossover**.

Данный оператор скрещивания используется для бинарных векторов.

Пусть имеется два родителя (родительские хромосомы) $\overline{Parent}^1$ и $\overline{Parent}^2$. В двух случайных местах происходят разрывы между двумя позициями генов в обеих хромосомах. После этого хромосомы обмениваются частями, в результате чего образуются два потомка. Из них выбирается случайно один потомок, который и передается в качестве результата оператора скрещивания. То есть скрещивание происходит по формулам:

$$Crossover\left(\overline{Parent}^1, \overline{Parent}^2, DataOfCros\right) = Random\left(\left\{\overline{Offspring}^1; \overline{Offspring}^2\right\}\right), \quad (14)$$

$r_1 = Random\left(\{2; 3; \ldots; n\}\right);$

$r_2 = Random\left(\{2; 3; \ldots; n\}\right);$

$R_1 = \min\left(r_1, r_2\right);$

$R_2 = \max\left(r_1, r_2\right);$

$\overline{Offspring_i}^1 = \overline{Parent_i}^1, i = \overline{1, R_1 - 1};$

$\overline{Offspring_i}^1 = \overline{Parent_i}^2, i = \overline{R_1, R_2 - 1};$

$\overline{Offspring_i}^1 = \overline{Parent_i}^1, i = \overline{R_2, n};$

$\overline{Offspring_i}^2 = \overline{Parent_i}^2, i = \overline{1, R_1 - 1};$

$\overline{Offspring_i}^2 = \overline{Parent_i}^1, i = \overline{R_1, R_2 - 1};$

$\overline{Offspring_i}^2 = \overline{Parent_i}^2, i = \overline{R_2, n};$

$\overline{Offspring}^1 \in X, \overline{Offspring}^2 \in X.$

**Пример.** Двухточечное скрещивание показано на рисунке:

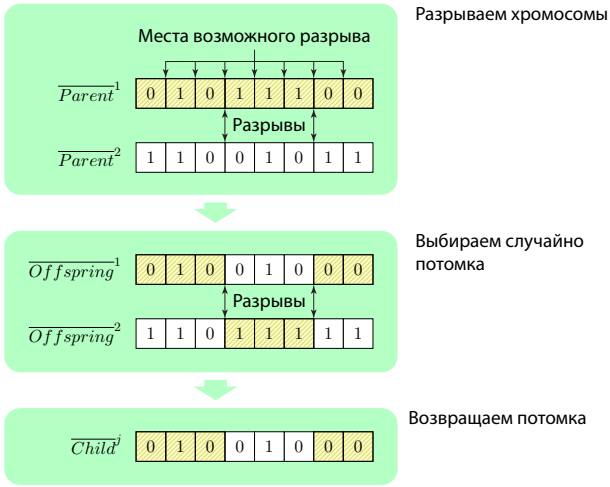


Рисунок 6. Механизм работы двухточечного скрещивания

$DataOfCros$ не содержит каких-либо параметров относительно данного типа скрещивания.

В библиотеке **HarrixMathLibrary** данная селекция реализуется через функцию **TMHL_TwopointCrossover**:

https://github.com/Harrix/HarrixMathLibrary

## 4.5 Двухточечное скрещивание с возможностью полного копирования одного из родителей

Идентификатор: **TwopointCrossoverWithCopying**.

Данный оператор скрещивания используется для бинарных векторов.

Отличается от стандартного двухточечного скрещивания тем, что точки разрыва могут происходить по краям родителей, что может привести к полному копированию родителя.

Пусть имеется два родителя (родительские хромосомы) $\overline{Parent}^1$ и $\overline{Parent}^2$. В двух случайных местах происходят разрывы между двумя позициями генов в обеих хромосомах. После этого хромосомы обмениваются частями, в результате чего образуются два потомка. Из них выбирается случайно один потомок, который и передается в качестве результата оператора скрещивания. То есть скрещивание происходит по формулам:

$$Crossover\left(\overline{Parent}^1, \overline{Parent}^2, DataOfCros\right) = Random\left(\left\{\overline{Offspring}^1; \overline{Offspring}^2\right\}\right), \quad (15)$$

$$r_1 = Random\left(\{1; 2; \ldots; n+1\}\right);$$
$$r_2 = Random\left(\{1; 2; \ldots; n+1\}\right);$$
$$R_1 = \min\left(r_1, r_2\right);$$
$$R_2 = \max\left(r_1, r_2\right);$$
$$\overline{Offspring}_i^1 = \overline{Parent}_i^1, i = \overline{1, R_1 - 1};$$
$$\overline{Offspring}_i^1 = \overline{Parent}_i^2, i = \overline{R_1, R_2 - 1};$$
$$\overline{Offspring}_i^1 = \overline{Parent}_i^1, i = \overline{R_2, n};$$
$$\overline{Offspring}_i^2 = \overline{Parent}_i^2, i = \overline{1, R_1 - 1};$$
$$\overline{Offspring}_i^2 = \overline{Parent}_i^1, i = \overline{R_1, R_2 - 1};$$
$$\overline{Offspring}_i^2 = \overline{Parent}_i^2, i = \overline{R_2, n};$$
$$\overline{Offspring}^1 \in X, \overline{Offspring}^2 \in X.$$

**Пример.** Двухточечное скрещивание с возможностью полного копирования одного из родителей показано на рисунке:

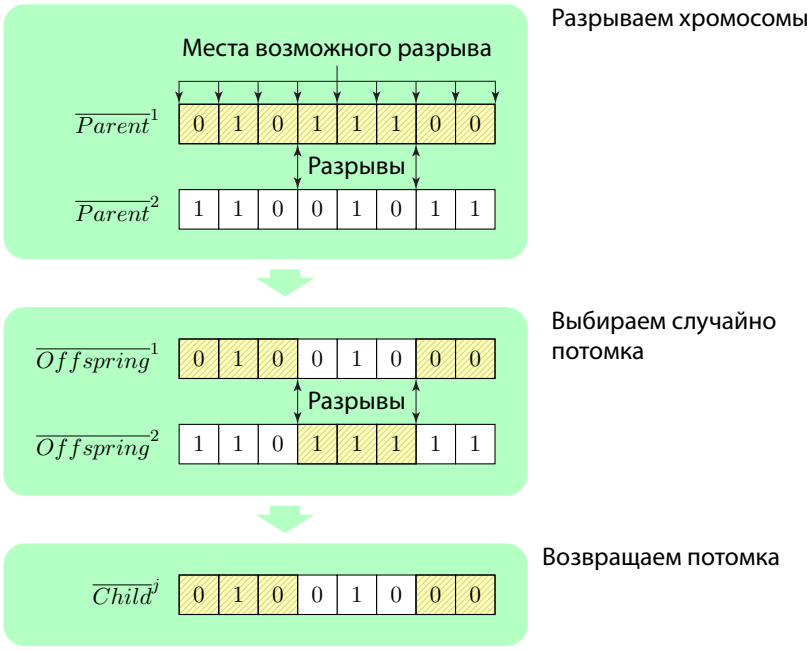


Рисунок 7. Механизм работы двухточечного скрещивания с возможностью полного копирования одного из родителей

$DataOfCros$ не содержит каких-либо параметров относительно данного типа скрещивания.

В библиотеке **HarrixMathLibrary** данная селекция реализуется через функцию **TMHL_TwopointCrossoverWithCopying**:

https://github.com/Harrix/HarrixMathLibrary

## 4.6 Двухточечное скрещивание для вещественных векторов

Идентификатор: **TwopointCrossoverForReal**.

Данный оператор скрещивания используется для вещественных векторов.

По сути ничем не отличается от TwopointCrossover, кроме типа векторов, на котором работает оператор.

Пусть имеется два родителя (родительские хромосомы) $\overline{Parent}^1$ и $\overline{Parent}^2$. В двух случайных местах происходят разрывы между двумя позициями генов в обеих хромосомах. После этого хромосомы обмениваются частями, в результате чего образуются два потомка. Из них выбирается случайно один потомок, который и передается в качестве результата оператора скрещивания. То есть скрещивание происходит по формулам:

$$Crossover\left(\overline{Parent}^1, \overline{Parent}^2, DataOfCros\right) = Random\left(\left\{\overline{Offspring}^1; \overline{Offspring}^2\right\}\right), \quad (16)$$

$r_1 = Random\left(\{2; 3; \ldots; n\}\right);$
$r_2 = Random\left(\{2; 3; \ldots; n\}\right);$
$R_1 = \min\left(r_1, r_2\right);$
$R_2 = \max\left(r_1, r_2\right);$
$\overline{Offspring}_i^1 = \overline{Parent}_i^1, i = \overline{1, R_1 - 1};$
$\overline{Offspring}_i^1 = \overline{Parent}_i^2, i = \overline{R_1, R_2 - 1};$
$\overline{Offspring}_i^1 = \overline{Parent}_i^1, i = \overline{R_2, n};$
$\overline{Offspring}_i^2 = \overline{Parent}_i^2, i = \overline{1, R_1 - 1};$
$\overline{Offspring}_i^2 = \overline{Parent}_i^1, i = \overline{R_1, R_2 - 1};$
$\overline{Offspring}_i^2 = \overline{Parent}_i^2, i = \overline{R_2, n};$
$\overline{Offspring}^1 \in X, \overline{Offspring}^2 \in X.$

$DataOfCros$ не содержит каких-либо параметров относительно данного типа скрещивания.

В библиотеке **HarrixMathLibrary** данная селекция реализуется через функцию **MHL_TwopointCrossoverForReal**:

https://github.com/Harrix/HarrixMathLibrary

## 4.7 Равномерное скрещивание

Идентификатор: **UniformCrossover**.

Данный оператор скрещивания используется для бинарных векторов.

Пусть имеется два родителя (родительские хромосомы) $\overline{Parent}^1$ и $\overline{Parent}^2$. Потомок состоит из генов, каждый из которых выбран случайно из генов родителей на соответствующих позициях. То есть скрещивание происходит по формулам:

$$\begin{aligned} &Crossover\left(\overline{Parent}^1, \overline{Parent}^2, DataOfCros\right) = \overline{Offspring}; \\ &\overline{Offspring}_i = Random\left(\left\{\overline{Parent}_i^1; \overline{Parent}_i^2\right\}\right), i = \overline{1,n}; \\ &\overline{Offspring} \in X. \end{aligned} \tag{17}$$

**Пример.** Равномерное скрещивание показано на рисунке:

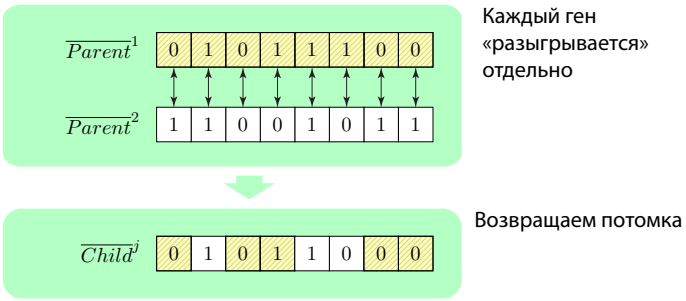


Рисунок 8. Механизм работы равномерного скрещивания

$DataOfCros$ не содержит каких-либо параметров относительно данного типа скрещивания.

В библиотеке **HarrixMathLibrary** данная селекция реализуется через функцию **TMHL_UniformCrossover**:

https://github.com/Harrix/HarrixMathLibrary

## 4.8 Равномерное скрещивание для вещественных векторов

Идентификатор: **UniformCrossoverForReal**.

Данный оператор скрещивания используется для вещественных векторов.

По сути ничем не отличается от UniformCrossover, кроме типа векторов, на котором работает оператор.

Пусть имеется два родителя (родительские хромосомы) $\overline{Parent}^1$ и $\overline{Parent}^2$. Потомок состоит из генов, каждый из которых выбран случайно из генов родителей на соответствующих позициях. То есть скрещивание происходит по формулам:

$$Crossover\left(\overline{Parent}^1, \overline{Parent}^2, DataOfCros\right) = \overline{Offspring};$$
$$\overline{Offspring}_i = Random\left(\left\{\overline{Parent}_i^1; \overline{Parent}_i^2\right\}\right), i = \overline{1,n};$$
$$\overline{Offspring} \in X. \tag{18}$$

$DataOfCros$ не содержит каких-либо параметров относительно данного типа скрещивания.

В библиотеке **HarrixMathLibrary** данная селекция реализуется через функцию **MHL_UniformCrossoverForReal**:

https://github.com/Harrix/HarrixMathLibrary

# 5 Операторы формирования нового поколения из родителей и потомков

**Формирование нового поколения** — оператор формирования нового поколения из массива родителей и получившихся потомков с использованием уже известных значений функции пригодности, как родителей, так и потомков.

## 5.1 Только потомки

Идентификатор: **OnlyOffspringGenerationForming**.

Данный оператор используется для векторов любой природы.

Данная схема предполагает формирование нового поколения из потомков и родителей таким, что в новое поколение попадают только потомки. Данный тип формирования нового поколения определяется формулой:

$$Forming\begin{pmatrix} Population \\ MutChildPopulation \\ Fitness \\ FitnessOfMutChild \\ DataOfForm \end{pmatrix} = \begin{pmatrix} MutChildPopulation \\ FitnessOfMutChild \end{pmatrix}. \tag{19}$$

Здесь $Population$ — множество родителей, $MutChildPopulation$ — множество потомков, $Fitness$ – множество значений целевой функции множества родителей, $FitnessOfMutChild$ – множество значений целевой функции множества родителей.

$DataOfForm$ не содержит каких-либо параметров относительно данного типа формирования нового поколения.

## 5.2 Только потомки и копия лучшего индивида

Идентификатор: **OnlyOffspringWithBestGenerationForming**.

Данный оператор используется для векторов любой природы.

Данная схема предполагает формирование нового поколения из потомков и родителей таким, что в новое поколение попадают только потомки (без одного) и копия лучшего индивида $\overline{Best}$ (лучшего за всё время работы генетического алгоритма, а не только текущего поколения). В русской литературе данный способ часто называют селекцией элитизма. Данный тип формирования нового поколения определяется формулой:

$$Forming\begin{pmatrix} Population \\ MutChildPopulation \\ Fitness \\ FitnessOfMutChild \\ DataOfForm \end{pmatrix} = \begin{pmatrix} \{\bar{x}^i\} \\ \{f^i\} \end{pmatrix}, \tag{20}$$

$$\bar{x}^i = \begin{cases} \overline{Best}, \text{ если } i = 0; \\ \overline{MutChild}^i, \text{ если } i \neq 0; \end{cases}$$

$$f_i = f\left(\bar{x}_i\right), i = \overline{1, N}.$$

Здесь $Population$ — множество родителей, $MutChildPopulation$ — множество потомков, $Fitness$ – множество значений целевой функции множества родителей, $FitnessOfMutChild$ – множество значений целевой функции множества родителей.

$DataOfForm$ не содержит каких-либо параметров относительно данного типа формирования нового поколения.

# 6 Операторы «малого» изменения решения

## 6.1 Мутация

Идентификатор: **Mutation**.

Данный оператор используется для бинарных векторов.

**Мутация** — оператор случайного изменения всех потомков из популяции. Цель данного оператора не получить более лучшее решение, а разнообразить многообразие рассматриваемых индивидов. Обычно мутация предполагает незначительное изменение вектора. При выполнении оператора каждый ген каждого индивида с некоторой заданной вероятностью $ProbabilityOfMutation$ мутирует, то есть меняет свое значение на противоположное. Пусть у нас имеется некий бинарный вектор $\overline{x}$. Мутация происходит по формулам:

$$\overline{MutChild}_j = \begin{cases} \overline{x}_j, \text{ если } random\,(0,1) > ProbabilityOfMutation; \\ 1 - \overline{x}_j, \text{ иначе }. \end{cases} \tag{21}$$

Обычно в генетическом алгоритме вероятность мутации выбирается из трех вариантов: слабая ($Weak$), средняя ($Average$) и сильная ($Strong$) мутация. Отсюда вероятность мутации

определяется формулой:

$$ProbabilityOfMutation\left(TypeOfMutation\right) = \tag{22}$$
$$= \begin{cases} \dfrac{1}{3n}, \text{ если } TypeOfMutation = Weak; \\ \dfrac{1}{n}, \text{ если } TypeOfMutation = Average; \\ min\left(1, \dfrac{3}{n}\right), \text{ если } TypeOfMutation = Strong. \end{cases}$$

Здесь

$$TypeOfMutation \in \{Weak; Average; Strong\}, \tag{23}$$

$n$ — длина вектора $\bar{x} \in X$ бинарной задачи оптимизации.

В библиотеке **HarrixMathLibrary** данная селекция реализуется через функцию **TMHL_MutationBinaryMatrix** (сразу мутируют все решения):

https://github.com/Harrix/HarrixMathLibrary