

```

1  switch (CrossoverOperator)
2  {
3      case CrossoverType.PMX:
4          // i1, i2 cut locations
5          // m[] partial map, m[i] => the mapping target of i
6
7          // define two crossover map for chromosome
8          index_1 = randomizer.Next(numberOfGenes);
9          while (index_2 == index_1)
10         {
11             index_2 = randomizer.Next(numberOfGenes);
12         }
13         // swap two numbers if index_1 is larger
14         if (index_2 < index_1)
15         {
16             temp_num = index_1;
17             index_1 = index_2;
18             index_2 = temp_num;
19         }
20         // initiate mapping
21         mapping = new int[numberOfGenes];
22         for (int i = 0; i < mapping.Length; i++) mapping[i] = -1;
23         // build up the mapping
24         for (int i = index_1; i < index_2; i++)
25         {
26
27             if (chromosomes[fatherIdx][i] == chromosomes[motherIdx][i])
28                 continue;
29             if (mapping[chromosomes[fatherIdx][i]] == -1 &&
30                 mapping[chromosomes[motherIdx][i]] == -1)
31             {
32                 mapping[chromosomes[fatherIdx][i]] =
33                     chromosomes[motherIdx][i];
34                 mapping[chromosomes[motherIdx][i]] =
35                     chromosomes[fatherIdx][i];
36             }
37             else if (mapping[chromosomes[fatherIdx][i]] == -1)
38             {
39                 mapping[chromosomes[fatherIdx][i]] =
40                     mapping[chromosomes[motherIdx][i]];
41                 try
42                 {
43                     mapping[mapping[chromosomes[motherIdx][i]]] =
44                         chromosomes[fatherIdx][i];
45                 }
46                 catch (System.IndexOutOfRangeException Exception)
47                 {
48
49                 }
50                 mapping[chromosomes[motherIdx][i]] = -2;
51             }
52             else if (mapping[chromosomes[motherIdx][i]] == -1)
53             {
54                 try
55                 {
56                     mapping[chromosomes[motherIdx][i]] =
57                         mapping[chromosomes[fatherIdx][i]];
58                 }
59                 catch (System.IndexOutOfRangeException Exception) { }
60                 try
61                 {
62                     mapping[mapping[chromosomes[fatherIdx][i]]] =
63                         chromosomes[motherIdx][i];
64                 }
65                 catch (System.IndexOutOfRangeException Exception) { }
66                 try
67                 {
68                     mapping[chromosomes[fatherIdx][i]] = -2;
69                 }
70             }
71         }
72     }
73 }

```

```

62     }
63     catch (System.IndexOutOfRangeException Exception) { }
64 }
65 else
66 {
67     try
68     {
69         mapping[mapping[chromosomes[motherIdx][i]]] =
            mapping[chromosomes[fatherIdx][i]];
70     }
71     catch (System.IndexOutOfRangeException Exception) { }
72
73     try
74     {
75         mapping[mapping[chromosomes[fatherIdx][i]]] =
            mapping[chromosomes[motherIdx][i]];
76     }
77     catch (System.IndexOutOfRangeException Exception) { }
78
79
80     mapping[chromosomes[fatherIdx][i]] = -3;
81     mapping[chromosomes[motherIdx][i]] = -3;
82 }
83 }
84
85 // crossover and make two children
86 for (int i = 0; i < numberOfGenes; i++)
87 {
88     if (index_1 <= i && i < index_2)
89     {
90         chromosomes[child1Idx][i] = chromosomes[motherIdx][i];
91         chromosomes[child2Idx][i] = chromosomes[fatherIdx][i];
92     }
93     else
94     {
95         if (mapping[chromosomes[fatherIdx][i]] < 0)
96             chromosomes[child1Idx][i] = chromosomes[fatherIdx][i];
97         else chromosomes[child1Idx][i] =
98             mapping[chromosomes[fatherIdx][i]];
99
100         if (mapping[chromosomes[motherIdx][i]] < 0)
101             chromosomes[child2Idx][i] = chromosomes[motherIdx][i];
102         else chromosomes[child2Idx][i] =
103             mapping[chromosomes[motherIdx][i]];
104     }
105 }
106 // crossover finished.
107
108 // check if all gene are distinct, if not let other chromosome
109 // replace it
110 for (int i = 0; i < populationSize * 3; i++)
111 {
112     temp = new int[numberOfGenes];
113     for (int j = 0; j < numberOfGenes; j++)
114     {
115         temp[j] = j;
116     }
117
118     if (chromosomes[i].Distinct().ToArray().Count() != numberOfGenes)
119     {
120         chromosomes[i] = temp.OrderBy(x =>
121             randomizer.Next()).ToArray();
122     }
123     else
124     {
125         break;
126     }
127 }
128 }

```

```

123         break;
124     case CrossoverType.OX:
125         // order crossover
126         // define two crossover map for chromosome
127         index_1 = randomizer.Next(numberOfGenes);
128         while (index_2 == index_1)
129         {
130             index_2 = randomizer.Next(numberOfGenes);
131         }
132         // swap two numbers if index_1 is larger
133         if (index_2 < index_1)
134         {
135             temp_num = index_1;
136             index_1 = index_2;
137             index_2 = temp_num;
138         }
139         // start creating crossover children
140         temp_num = 0;
141         temp_num_2 = 0;
142         for (int i = index_1; i < index_2; i++)
143         {
144             chromosomes[child1Idx][i] = chromosomes[fatherIdx][i];
145             chromosomes[child2Idx][i] = chromosomes[motherIdx][i];
146         }
147         for (int i = 0; i < numberOfGenes; i++)
148         {
149             if
150             (!chromosomes[child1Idx].Contains(chromosomes[motherIdx][i]))
151             {
152                 if (temp_num == index_1) temp_num = index_2;
153                 chromosomes[child1Idx][temp_num] = chromosomes[motherIdx][i];
154                 temp_num += 1;
155             }
156             if
157             (!chromosomes[child2Idx].Contains(chromosomes[fatherIdx][i]))
158             {
159                 if (temp_num_2 == index_1) temp_num_2 = index_2;
160                 chromosomes[child2Idx][temp_num_2] = chromosomes[fatherIdx][i];
161                 temp_num_2 += 1;
162             }
163         }
164         break;
165     case CrossoverType.POX:
166         // position-based crossover
167         temp_num = (int)Math.Round(numberOfGenes * crossoverRate);
168         temp_1 = new int[temp_num];
169         // create random index array
170         for (int i = 0; i < temp_num; i++)
171         {
172             temp_1[i] = randomizer.Next(numberOfGenes);
173         }
174         // crossover children
175         for (int i = 0; i < numberOfGenes; i++)
176         {
177             if (temp_1.Contains(i))
178             {
179                 chromosomes[child1Idx][i] = chromosomes[fatherIdx][i];
180                 chromosomes[child2Idx][i] = chromosomes[motherIdx][i];
181             }
182             else
183             {
184                 chromosomes[child1Idx][i] = chromosomes[motherIdx][i];
185                 chromosomes[child2Idx][i] = chromosomes[fatherIdx][i];
186             }
187         }
188         break;
189     case CrossoverType.OSS:
190         // order-based crossover

```

```

190     temp_num = (int)Math.Round(numberOfGenes * crossoverRate);
191     temp_1 = new int[temp_num];
192     // create random value array
193     for (int i = 0; i < temp_num; i++)
194     {
195         temp_1[i] = randomizer.Next(numberOfGenes);
196     }
197     // crossover children
198     for (int i = 0; i < numberOfGenes; i++)
199     {
200         if (!(temp_1.Contains(chromosomes[fatherIdx][i])))
201         {
202             chromosomes[child1Idx][i] = chromosomes[motherIdx][i];
203         }
204         else {
205             chromosomes[child1Idx][i] = chromosomes[fatherIdx][i];
206         }
207
208         if (!(temp_1.Contains(chromosomes[motherIdx][i])))
209         {
210             chromosomes[child2Idx][i] = chromosomes[fatherIdx][i];
211         }
212         else
213         {
214             chromosomes[child2Idx][i] = chromosomes[motherIdx][i];
215         }
216     }
217
218     break;
219 case CrossoverType.OCCC:
220     break;
221 }

```