Local Intermediary Recombination

Operator

Real

Gevol.evolution.evoperator.real

Description

Randomly selects two parents from the population. It takes each real gene from both parents and modify it using random numbers according to the formula:

$$g = u * p1 + (1 - u) * p2$$

Where:

- u - random number between 0 and 1, the same for every gene
- p1 - gene from first parent
- p2 - gene from second parent

Parameters

No parameters.

Pseudocode

```
P1 = select randomly one individual from population P
P2 = select randomly one individual from population P
u = random number between 0 and 1
For each gene g in new chromosome
      g = u * P1 + (1 – u) * P2
```

Implementation details

First, new chromosome is being created. The real individual is built with three chromosomes: values, sigma and alpha. New genes are calculated for each chromosome.

```
//rand modify parameter
double u = Randomizer.NextDouble(0, 1);
//generate new individual
RealIndividualChromosome p1chromosome =
(RealIndividualChromosome)population[indv1].Chromosome;
RealIndividualChromosome p2chromosome =
(RealIndividualChromosome)population[indv2].Chromosome;
RealIndividualChromosome newChromosome = new
RealIndividualChromosome(p1chromosome.Sigma.Count, p1chromosome.Alpha.Count);
newChromosome.Age = 0;
for (int i = 0; i < p1chromosome.Sigma.Count; i++)
{
      newChromosome.Sigma.Add((u * p1chromosome.Sigma[i]) + ((1-
u)*p2chromosome.Sigma[i]));
      newChromosome.Values.Add((u * p1chromosome.Values[i]) + ((1 - u) *
p2chromosome.Values[i]));
}
for (int i = 0; i < p1chromosome.Alpha.Count; i++)
```

```
{
       newChromosome.Alpha.Add((u * p1chromosome.Alpha[i]) + ((1 - u) *
p2chromosome.Alpha[i]));
}
```

Then new population is returned with one new individual.

```
Population newPopulation = new Population();
RealIndividual newIndividual = new RealIndividual(newChromosome);
newPopulation.Add(newIndividual);
return newPopulation;
```