PARAMETRIC SYNTHESIS OPTIMIZATION WITH GENETIC ALGORITHMS

During components parametrical synthesis problem solution is complicate to show the analytic expressions, that describe objective function (as objective function the quality criteria, item price or availability ratio may be used) dependence from the various factor range [1-4]. The classical methods of optimization application is limited because of prior information lack. That is why the problem of methods that are capable to find solutions under the absence of prior information about the object is very actual. One of such methods the evolution algorithms are and genetic algorithms in particular. In the paper author’s analyzed the binary, continuous GA and hybrid genetic algorithms. Both algorithms follow the same procedure of modeling genetic recombination and natural selection [6].

Binary GA represents variables as an encoded binary string and works with the binary strings to minimize the cost. Continuous GA works with the continuous variables themselves to minimize the cost. Since GAs originated with a binary representation of the variables, the
binary method is presented first. On the Fig.1.A. the flowchart of a binary algorithm is presented. On the Fig.2. the flowchart of a continuous algorithm is presented. The flowchart in Fig. 2, B provides a “big picture” overview of a continuous GA. The average minimum cost of a population as a function of generation is shown in Fig. 2. The continuous parameter GA outperformed the binary GA by finding a much lower minimum cost over 25 generations. The binary GA only took one more generation to find the minimum than the continuous parameter GA. A hybrid GA combines the power of the GA with the speed of a local optimizer. The GA excels at gravitating toward the global minimum. It is not especially fast at finding the minimum when in a locally quadratic region. Thus the GA finds the region of the optimum, and then the local optimizer takes over to find the minimum. Hybrid GA can take one of the following forms [6]:

Running a GA until it slows down, then letting a local optimizer take over. Hopefully the GA is very close to the global minimum.

Seeding the GA population with some local minima found from random starting points in the population.

Every so many iterations, running a local optimizer on the best solution or the best few solutions and adding the resulting chromosomes to the population.

The GA used for the analysis has a population size of 16 and a mutation rate of 0.2.

Convergence results appear in Fig.3, averaged over 200 independent random runs. The dashed line for the GA and the solid line for the hybrid GA are not identical prior to the start of the local optimizer due to the random nature of both algorithms.
On the average the hybrid GA finds the minimum in many fewer function calls than the GA by itself. The several types of GA were compared with application of test function. The convergence results were received.

**Literature**


