Independent Sampling Genetic Algorithms and the Applications

Chien-Feng Huang
Center for the Study of Complex Systems
University of Michigan, Ann Arbor, MI 48109
cfhuang@umich.edu

Premature convergence has long been recognized as one notorious factor that leads to the unsatisfactory performance of genetic algorithms (GAs). In particular, due to the resulting loss of diversity of building blocks, it impedes the perfect implementation of implicit parallelism and thus hinders the efficacy of crossover. In this paper, a new strategy for independent sampling of building blocks (independent sampling genetic algorithms (ISGAs)) is proposed in order to nicely implement implicit parallelism. Simply stated, each individual independently samples candidate schemata and creates population diversity in the first phase (independent sampling phase), subsequently elitism and population structures are adopted for breeding to attain the global optimum (breeding phase). The following addresses this methodology in more detail.

INDEPENDENT SAMPLING PHASE

Step 1. Choose an empty set called *EstSchPos* for collecting possible positions of useful schemata for future processing. Then initialize a new string at random with uniform probability for each bit. If its fitness is greater than zero, go to step 2; if not, continue to pick up a new string until its fitness exceeds zero, and then go to step 2. Record the current fitness as *OldFit*.

Step 2. From the first bit to the last bit of this string except the elements in *EstSchPos*, successively flip a bit each time and evaluate the fitness of the resulting string. Record the new fitness as *NewFit*. If *NewFit* is less than *OldFit*, record the position of this bit (by appending this locus to *EstSchPos*) and its original bit value as a candidate gene. (After step 2, we should be able to collect a record for bit values and loci of candidate schemata.)

Step 3. Generate again a new string at random and replace the alleles at positions *EstSchPos* with the corresponding schema values discovered at step 2 until the fitness of the resulting string exceeds *OldFit*. Record the new fitness as *OldFit*.

Step 4. Go to steps 2 and 3 until an optimum string has been found, or the length of *EstSchPos* has reached the string length, or until a maximum number of evaluations has been performed.

Step 5. Return the current fitness value and the number of function evaluations.

The philosophy embedded in the above algorithm is that once several beneficial genes appear simultaneously, they together contribute a fitness increase of the string containing them; thus any loss of one of these genes leads to the fitness decrease of the string.

BREEDING PHASE

After the independent sampling phase, the population is expected to contain enough diverse candidate schemata, because a number of individuals independently build up an evolutionary avenue by various building blocks. However, factors such as deception and incompatible schemata could lead individuals to only arrive at suboptimal solutions. To overwhelm this difficulty, in each cycle, we first pick out the best string, and then look for a mating partner with the highest rank according to the exponential rank selection, and with the largest diversity based on "hamming distance" to the best string.

To demonstrate the capability of the ISGA, we choose as the testbed a 30-bit deceptive function whose fitness landscape consists of ten copies of an order-3 fully deceptive function as defined in the paper of messy GAs (Goldberg, Korb, & Deb, 1989). Then we compare its performance with those of several different types of GAs: mGAs, modified mGAs (Goldberg, Deb, Kargupta, & Harik, 1993), Breeder GAs (BGAs) (Mühlenbein & Schlierkamp-Voosen, 1993). (The experimental result of the ISGA is based on population size 50 and exponential base 1.5.) The mean function evaluations to optimum is 2128 for ISGA, 40600 for mGA, 26650 for modified mGA, and 16000 for BGAs, respectively; from which we see that the ISGA significantly outperforms other GAs.

References

Goldberg, D. E., Deb, K., Kargupta, H., & Harik, G. 1993). Rapid, Accurate optimization of difficult problems using fast messy genetic algorithms. *Proceedings of the Fifth International Conference on Genetic Algorithms*, 56-64.

Goldberg, D. E., Korb, B., & Deb, K. (1989). Messy genetic algorithms: Motivation, analysis, and first results. *Complex Systems 3*, 493-530.

Mühlenbein, H., & Schlierkamp-Voosen, D. (1993). *Optimal interaction of mutation and crossover in the breeder genetic algorithm*. Technical Report 93-042, GMD.