Scout Algorithms and Genetic Algorithms: A Comparative Study

Fabio Abbattista
Dipartimento di Informatica
Universita' di Bari
Via E. Orabona 4
70126 Bari, Italy
email: fabio@di.uniba.it

Valeria Carofiglio
Dipartimento di Informatica
Universita' di Bari
Via E. Orabona 4
70126 Bari, Italy
email: valeria@di.uniba.it

Mario Köppen
Department Pattern Recognition
Fraunhofer IPK–Berlin
Pascalstr. 8-9
10587 Berlin, Germany
email: mario.koeppen@ipk.fhg.de

Summary

A comparative study of the recently proposed Scout algorithm (Scout_O) [1] and the standard genetic algorithm (SGA) is presented. The main results are: update rules can be given, for which Scout_O performs similar to SGA; schemata of SGA can be identified for Scout_O as well; and Scout_O fulfills Hollands admissible detector configuration paradigm [2], a necessary pre-cursor of SGA-like algorithms.

A Scout algorithm explores binary search spaces, represented by all the possible binary strings $S = (S_i)$. The basic element of the algorithm is represented by the probability vector (PV) $P = (P_i)$, in which each element P_i gives the probability that the i-th element S_i will be 1. The main goal of Scout_O is to identify a vector P^{opt} in such a way that it is possible to generate an optimal solution for the problem at hand.

A Scout algorithm iteratively performs the three steps: generation of K solutions using the PV; selection of solutions, which performs better than the foregoing generation; and updating the PV according to the fitness gain achieved by the selected solutions.

Two new approaches for updating the PV are presented. The first one (Scout_M) was designed in order to resemble the evolutionary progress of a standard genetic algorithm and uses the second-order statistics of the fitness gain. Given

$$\lambda_i^j = \frac{\sqrt{\bar{f}_{\{S_i^p(c)\}^j}\bar{f}_{\{S^p(c)\}}}}{\sqrt{\sigma_{\{S(c)\}}}}$$

with $\{S_i^p(c)\}^j$ representing the set of all better performing solutions $\{S^p(c)\}$, having bit j at position i. Then, the first new update rule is

$$P_i(c+1) = P_i(c) + \alpha\lambda_i^1.$$

where α represents the learning rate (a real number belonging to the range [0,1]).

The second new version (denoted Scout_F) combines the Scout_M approach with the original Scout algorithm. Its update rule is

$$P_i(c+1) = \frac{P_i(c) + \lambda_i^1}{1 + \lambda_i^0 + \lambda_i^1}.$$

In order to verify the effectiveness of the proposed algorithms we tested them on two well-known problems: For the Royal Road Functions (RRF) [3], the original Scout algorithm achieved an average fitness of 6.91. Both modified versions achieved the maximum possible fitness value of 12.8 in 9 out of of 10 runs. For the Tanese functions [4], which are constructed from Walsh Polynomials (WP) and are known as hard-to-solve problems for SGA, the original Scout algorithm achieved an average fitness of 96.02% of the maximum possible fitness value. For Scout_M a maximum fitness of 97.89% and for Scout_F a fitness of 98.34% was achieved in ten runs on five WP.

References

[1] Abbattista, F., Dalbis, D., *The Scout Algorithm to Explore Unknown Spaces*, Proc. of the Int. Conference on Evolutionary Computation, ICEC'98, 1998.

[2] Holland, J.H., *Goal-Directed Pattern Recognition*, Proc. of the Int. Conference on Methodologies of Pattern Recognition, Honolulu, Hawaii, pp. 287–296, 1969.

[3] Jones, T., *A Description of Holland's Royal Road Function*, Evolutionary Computation, 2(3) pp. 409-415, 1995.

[4] Tanese, R., *Distributed Genetic Algorithms for Function Optimization*, PhD thesis, The University of Michigan, Ann Arbor, MI, 1989.