Meta-Optimizing Semantic Evolutionary Search

Moshe Looks
Department of Computer Science and Engineering
Washington University in St. Louis
Saint Louis, MO 63130, USA
moshe@metacog.org

Categories and Subject Descriptors

I.2.2 [**Artificial Intelligence**]: Automatic Programming – *Program synthesis*

General Terms

Algorithms, Design, Experimentation

Keywords

Empirical Study, Heuristics, Optimization

A powerful heuristic allowing many optimization problems of interest to be solved quickly is to attempt *decomposition* – breaking problems down into smaller subproblems that may be solved independently. For example, the hierarchical Bayesian optimization algorithm (hBOA) [5] dynamically learns a problem decomposition in terms of solution parameters. The effectiveness of this approach hinges on the existence of some compact and reasonably correct decomposition in the space (of decompositions, not solutions).

Difficulty arises when no such decomposition exists, or when an effective decomposition cannot be formulated directly as a model over solution parameters. In other words, how successfully an optimization algorithm can exploit near-decomposability depends on how clever an encoding has been chosen by humans to represent the problem. I posit that the characteristics of program spaces and the typically chaotic mapping from programs to outputs tend to *scramble* problems – even if the mapping from program outputs to fitness levels is nearly decomposable, the overall problem will not be (in terms of parameters of program spaces).

MOSES (meta-optimizing semantic evolutionary search) [4] is a new estimation-of-distribution approach to program evolution. Distributions are not estimated over the entire space of programs. Rather, a novel representation-building procedure that exploits domain knowledge[1] is used to dynamically select program subspaces for estimation over. This leads to a system of demes consisting of alternative representations (i.e. program subspaces) that are maintained simultaneously and managed by the overall system. The hBOA is applied to learn new programs *within* existing demes, which may in turn spawn new demes. My hypothesis is that incorporating domain knowledge into the representation-building process facilitates search through subspaces of the overall program space that are nearly decomposable, leading to competent program evolution.

[1]E.g., in the ant problem we know that a left turn immediately followed by a right turn has no effect, that three left turns are equivalent to a right turn, etc. In the domain of Boolean formulae, we know that *x AND x* is always equivalent to *x*, that *x AND not(x)* is always false, etc.

GECCO'07, July 7–11, 2007, London, England, United Kingdom.
ACM 978-1-59593-697-4/07/0007.

Table 1: Computational effort/1000 ($p = .99$) based on 100 runs, PM = 2-parity-3-multiplexer.

Problem	GP	MOSES	MOSES, no model-building
Ant	450 [2], 104 [3]	23	36
PM	1088 [4]	218	375

MOSES has been applied to a number of domains and problems: results are presented above for the artificial ant [2] and hierarchically composed parity-multiplexer [1] problems. For comparison, results for MOSES without probabilistic model-building (i.e., instead of hBOA, univariate modeling and sampling) are also shown. The performance differential between GP[2] and MOSES sans model-building demonstrates the effectiveness of representation-building at reducing/improving the search space and deme management at preserving diversity. To understand the improvements derived from model-building, we can examine the models being build by MOSES to see what linkages are learned.

For the ant problem, the most common pairwise dependencies uncovered are between the variables representing different rotations – i.e., MOSES is exploiting the symmetry in the space between left and right. For the parity-multiplexer, the linkage analysis shows that the inner parity variables are most tightly linked, indicating that MOSES is exploiting the inherent compositional structure of this problem.

1. REFERENCES

[1] M. V. Butz. *Rule-based Evolutionary Online Learning Systems*. PhD thesis, UIUC, 2004.
[2] J. R. Koza. *Genetic Programming*. MIT Press, 1992.
[3] W. B. Langdon and R. Poli. Better trained ants for GP. Technical report, University of Birmingham, 1998.
[4] M. Looks. *Competent Program Evolution*. PhD thesis, Washington University in St. Louis, 2006.
[5] M. Pelikan and D. E. Goldberg. A hierarchy machine. *Complexity*, 2003.

[2][3] used a modified fitness function to improve performance.