Scalable Evolutionary Computation

Hillol Kargupta
School of Electrical Engineering and Computer Science
Washington State University
hillol@eecs.wsu.edu

Scalability is an important issue in algorithm design that studies the characteristics of an algorithm with respect to the changes in the factors that influence its performance. Such factors typically include the "size" of the problem, different facets of problem-difficulties, desired accuracy and reliability of the algorithm-performance, and the model of computation. It is important to a researcher since it characterizes the performance of an algorithm and identifies the suitability of the algorithm for solving large, difficult problems using different computing environments. It is also important to a practitioner since it points out the cost of running an algorithm for the expected level of success. Therefore, it makes sense to study scalability on both theoretical and experimental grounds.

From the theoretical perspective, *computational complexity* [1] analysis offers a useful way to quantify the scalability of an algorithm. This perspective of scalability is a widely accepted norm in almost every matured discipline that deals with computation. However, the presence of computational complexity-based analysis in the Evolutionary Computation (EC) literature has been minimal in the past. This could be because enumerative search takes exponential time at the worst case for almost every interesting application domain. Since complexity analysis often takes a conservative route, it exposes the worst case exponential-time performance of "random variation and selection"-based evolutionary algorithms (EAs) unless we restrict the scope of the EAs.

Similar situations can be found in many scientific fields. For example, consider the field of machine learning [9]. The fundamental difficulty of learning from data for the general class of all concepts was realized [8] in the early days of this field. The initial development of a primarily empirical field finally lead to a more matured discipline where new algorithms are expected to come with more rigorous performance characteristics. The development of the field of *computational learning theory* is a clear example of this transition. The pioneering work by Valiant [12], Haussler [4] and others lead to deeper understanding about the performance of many early machine learning algorithms and the development of the class of Probably-Approximately-Correct (PAC) learning algorithms. Today there exist a wide spectrum of different machine learning algorithms [11] that come with rigorous performance bounds for specific classes of problems and models of computation. Algorithms with limited theoretical analysis are expected to be backed by empirical experimental results for large and difficult problems.

The field of combinatorial search offers another similar example. The fundamental hardness of many interesting combinatorial search problem did not hinder the development of the field of *approximation algorithms* [3, 6]. Today, there exist many good polynomial-time approximation algorithms for several NP-hard problems [5].

EC faces a similar situation. We know that evolutionary search, optimization, and machine learning are hard in general. However, such worst case result should not prohibit us from quantifying the performance of EAs for different classes of problems, different performance qualities, and different models of computation. We need to study the scalability of EAs for different scenarios and compare their performance with that of other existing algorithms. This will certainly help settling the dust in the field of EC.

The scalability issue in EC shows up in many different facets. Typically, EAs are applied to search, optimization, and machine learning problems. The factors controlling the difficulty-level of such problems include the number of search variables, epistatic interaction among the search variables, local optima, misleadingness and other structural properties of the search space. Scalability of EAs with respect to growing degrees of difficulties along these dimensions is an important issue. We need to quantify the performance characteristics of our favorite EAs as we increase our demand for better reliability and accuracy. Moreover, the speedup in problem solving by EAs with increasing number of processors in different models of computation needs to be quantified. We should also explore the practitioner's perspective toward scalability. We need to study successful, large-scale EC applications and identify the main issues behind their scalability. Undoubtedly, this will play a critical role in the future of EC for the next millennium.

Fortunately, an increasing number of EC researchers and practitioners are investigating the scalability of popular evolutionary algorithms. They are developing EAs that can be rigorously claimed to be efficient on both theoretical and experimental grounds. This special issue offers a flavor of this growing body of work by putting together several interesting papers that address different facets of scalability in EAs.

The first paper by Dirk Thierens discusses the scalability of simple genetic algorithm (GAs). Typically in simple GA, the search space structure is supposed to be implicitly detected using schemata (similarity-based equivalence classes). This paper considers the scalability of the process of schema detection and their subsequent exploitation in simple GA. It essentially shows that unless the user hand-picks the representation in such a way that the information about the features defining good schemata are available to the crossover operator, simple GA faces some serious scalability problems. This paper underscores the seriousness of the problem that we are facing and its importance in the future of EC. It also points out the need for a new generation of evolutionary algorithms that are scalable for large, difficult problems.

Restriction on the problem-space is essential for developing efficient search algorithms. The early work on inductive learning [10, 13] and the relatively recent developments [14] clearly point that out. However, restricting the application domain too much may destroy the basic purpose of black-box search (searching in absence of "adequate" domain knowledge) algorithm. We need to find a middle ground where "bias" on the targeted class of problems is flexible enough to cover a reasonable class of interesting problems. Can it be done? Yes, and people have already found such middle ground in a related discipline. The futility of truly general inductive machine learning algorithm *with no bias* is quite well known [10, 13]. In an inductive learning or guessing process, bias is often introduced in many ways, including the representation and operators. It simply captures the preference of a certain algorithm for certain representation, operators and other parameters over other possibilities. Bias of a learning algorithm restricts its applicability to certain problems. However, an appropriate bias also improves the performance of an algorithm for certain classes of problems. Therefore, inductive bias should be considered as a trade off and a suitable choice of bias should require finding a domain specific middle ground. The field of machine learning appears to have found such a middle ground. The widespread application of inductive learning algorithms in many different domains and the recent development of the billion-dollar data mining [2] industry would not have been possible without that.

The next generation of EAs will have to find such a middle ground. This special issue samples a few papers that attempt to do that. The second and the third paper assume restrictions on the interaction among the search variables of the problem. In a given objective function, a search variable may depend on all other variables, resulting in epistatic interaction among all the variables. In the so called n-k Landscape framework[7] this is similar to the cases where k is $O(n)$; n is the number of search variables. These two papers assume that k is bounded by some constant. In other words, they consider problems with Bounded Variable Interaction (BVI). Yes, this restricts the scope of the algorithms. However, that is the point. Until we come up with a better way to introduce restrictions on the problem space, this offers a reasonable way to gain efficiency and scalability at the cost of restricted scope.

The second paper by Heinz Mühlenbein and Thilo Mahnig describes the Factorized Distribution Algorithm (FDA) that estimates the underlying search space distribution using a-priori chosen structure of the model. The FDA algorithm is designed for the class of additively decomposable functions. It runs in $O(n\sqrt{n})$ time. The FDA assumes the knowledge about the decomposition structure of the objective function. However, the authors propose to eliminate this problem using a Bayesian algorithm called the LFDA. The LFDA algorithm inductively estimates the structure of the search space by detecting the mutually interacting variables.

The third paper by Masaharu Munetomo and David Goldberg offers an alternate approach for efficiently detecting the underlying search space structure for problems with BVI. It develops the LIMD algorithm that notes the dependencies among the search variables by a non-monotonicity detection procedure. The LIMD algorithm works by checking second order dependencies in $O(n^2)$ objective function evaluations. This algorithm works for problems where all the search variables can be grouped into a set of equivalence classes based on their mutual interaction. However, this paper also extends the LIMD approach for detecting the variable interaction for problems where the different classes of mutually interacting variables overlap.

The paper by William Langdon explores a different aspect of scalability in EC. It focuses on properties of the distribution of program-fitness in genetic programming. It investigates the characteristics of such distributions with respect to the changes in size of the programs. It makes use of the enumerative and Monte Carlo techniques to study these characteristics. It also reports that once a minimum size threshold is reached, the fitness distribution appears to be almost independent of the program length. This work offers a systematic study of the convergence property of the fitness distributions for a large class of problems and an analytical proof of this for linear programs.

The paper by Erick Cantú-Paz and David Goldberg explores another angle to scalability in EC. It explores the performance speed-up of different models of parallel GAs as the number of processors is increased. It investigates two important classes of parallel GAs: (1) a single population-based parallel GA with master-slave architecture where different processors are used for parallel evaluation of the objective function and (2) a multi-population-based parallel GA where separate instances of the algorithm are executed in parallel along with migration-based exchange of population members.

This special issue presents an exposure to some important issues in scalable EC. Hopefully, this will take a small step toward crystallizing the growing effort to analyze and develop scalable evolutionary algorithms that come with a performance guarantee. The editor would like to thank all the reviewers and acknowledge NSF-Grant IIS 9830660 for supporting his effort.

References

[1] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to algorithms.* MIT Press, Massachussets; McGraw-Hill, New York, 1990.

[2] U. Fayyad, G. P. Shapiro, P. Smyth, and R. Uthurusamy. *Advances in Knowledge Discovery and Data Mining.* AAAI/MIT Press, USA, 1996.

[3] M. R. Garey, R. L. Graham, and J. D. Ullman. Worst case analysis of memory allocation algorithms. In *Proceedings of the 4th ACM Sympossium on Theory of Computing*, pages 143–150. ACM Press, New York, NY, 1972.

[4] D. Haussler. Quantifying inductive bias: AI learning algorithms and Valiant's learning framework. *Artificial Intelligence*, 2(36):177–222, 1989.

[5] D. Hochbaum. *Approximation Algorithms for NP-hard Problems.* PWS Publishing Company, Massachusetts, 1997.

[6] D. S. Johnson. Approximation algorithms for combinatorial problems. *Journal of Computing System Science*, 9:256–278, 1974.

[7] S. Kauffman. *The Origins of Order.* Oxford University Press, New York, 1993.

[8] M. Minsky and S. Papert. *Perceptrons.* MIT Press, MIT, USA, 1968.

[9] T. Mitchell. *Machine Learning.* McGraw-Hill, USA, 1st edition, 1997.

[10] T. M. Mitchell. The need for biases in learning generalizations. Rutgers Computer Science Tech. Rept. CBM-TR-117, Rutgers University, New Brunswick, New Jersy, 1980.

[11] B. K. Natarajan. *Machine Learning, A Theoretical Approach.* Morgan Kaufmann, San Mateo, CA, 1991.

[12] L. G. Valiant. A theory of the learnable. *Communications of the Association for Computing Machinery*, 27(11):1134–1142, 1984.

[13] S. Watanabe. *Knowing and guessing - A formal and quantitative study.* John Wiley & Sons, Inc., New York, 1969.

[14] D. H. Wolpert and W. G. Macready. No free lunch theorems for search. Tech. Rep. No. SFI-TR-95-02-010, Santa Fe Institute, Santa Fe, NM, 1995.

