Missouri University of Science and Technology
Scholars' Mine

Opportunities for Undergraduate Research Experience Program (OURE)

Student Research & Creative Works

02 Apr 2007

Improving the Usability of Evolutionary Algorithms: Self-Adaptive Semi-Autonomous Democratic Parent Selection

Joshua M. Eads

Follow this and additional works at: https://scholarsmine.mst.edu/oure

 Part of the Computer Sciences Commons

Recommended Citation

Eads, Joshua M., "Improving the Usability of Evolutionary Algorithms: Self-Adaptive Semi-Autonomous Democratic Parent Selection" (2007). *Opportunities for Undergraduate Research Experience Program (OURE)*. 190.
https://scholarsmine.mst.edu/oure/190

This Presentation is brought to you for free and open access by Scholars' Mine. It has been accepted for inclusion in Opportunities for Undergraduate Research Experience Program (OURE) by an authorized administrator of Scholars' Mine. This work is protected by U. S. Copyright Law. Unauthorized use including reproduction for redistribution requires the permission of the copyright holder. For more information, please contact scholarsmine@mst.edu.

Improving the Usability of Evolutionary Algorithms: Self-Adaptive Semi-Autonomous Democratic Parent Selection

Joshua M. Eads

Faculty Advisor: Dr. Daniel Tauritz
Research Mentor: Ekaterina Smorodkina

Department of Computer Science

April 2, 2007

Abstract

One of the primary obstacles to Evolutionary Algorithms (EAs) fulfilling their promise as easy to use general-purpose problem solvers is the difficulty of correctly configuring them for specific problems such as to obtain satisfactory performance. This paper introduces the concept of democratic, semi-autonomous parent selection by encoding and evolving population rating operators as in Genetic Programming and shows the potential of extending self-adaptation by pairing mates using an adaptation of the Stable Roommates problem. Replacing the typical general parent selection algorithm with autonomously evolved individual selection parameters has the prospective to bring EAs a step closer to their promise as easy to use general-purpose problem solvers.

Introduction

In the field of computer science, many problems are too complex for computers to find exact answers for in reasonable time even with increasing computational power. En lieu of searching very large problem spaces for the exact solution, carefully designed heuristics are typically used to narrow the search space and find solutions that may be accurate within a certain degree of the exact solution. Evolutionary Algorithms (EAs) are a class of algorithms which evolve solutions based on random initialization and have been shown to find approximate and exact solutions to real world problems faster than general purpose search algorithms. The basic design of an evolutionary algorithm maps possible solutions to the current problem as individuals in a virtual population. These possible solutions have a fitness attached to them, or how close they are to the optimal solution, and undergo a Darwinian evolution cycle for a number of generations until an optimal solution has been evolved. The goal of EAs is to be able to have a generic algorithm that can easily adapt to many difficult problems and still provide high quality solutions with minimal human interaction.

Evolutionary Algorithm Cycle

A standard evolutionary algorithm has a basic five-step cycle that describes a generation. A population of individuals is initially generated that represents a random sampling of the problem space. Individuals typically contain a single genotype that best fits the problem, such as a bit-string, vector of real-valued numbers, or a parse-tree. The second step of the cycle picks a predetermined number of individuals from the population to be parents during the current generation. These parents are typically selected in a tournament selection or other competitive selection processes; however this paper will present a decentralized method to replace existing techniques. The third step of the cycle performs the biological equivalent having all the parents from the parent pool mate with their selected mate and create a set of children with genetic material from their parents' genotypes. The forth step of the cycle applies a typically small amount of mutation to the children's genes to stimulate genetic diversity. The final stage of each generation is to determine which of the parents and children will survive on to the next generation. Over multiple generations, the selective pressure to keep more fit individuals through Darwinian natural selection will generally push the population towards the global optimum of the problem.

Self-Adaptation

This paper will describe a new method for bringing evolutionary algorithms closer to being fully autonomous. Previous research has worked to automate the mutation step-size (a complete field as Evolution Strategies) as well as crossover and mutation parameters [3] [1]. This paper expands on recent research working toward fully automating the parent

selection process by decentralizing the selection and giving each individual in the population control over who it mates with.

Semi-Autonomous parent selection

In a typical EA, a central algorithm is used to select the mates of all individuals. This method typically takes a pool of individuals and performs a tournament where the fittest individuals at each level move on until two final individuals are left. These individuals are selected as mates and the process continues for all mates. This process is easy to implement, but does not represent the autonomous mate selection process that is present in nature - independent selection of mates based on internal preferences of both individuals. This paper presents a new process termed as Self-Adaptive Semi-Autonomous Democratic Parent Selection (SASADEPS) which decentralizes the mate selection process and creates a more autonomous and adaptive method for selecting parents in any environment.

Overview of paper

The next section of this paper will present an in-depth look at how the parent selection process has been decentralized along with reasoning for allowing individuals to choose their own mate. Details of how previous work for finding stable matching for roommates and couples has been applied to matching the virtual individuals in the EA will follow. The three problems that were used to determine how the SASADEPS method compares to previous work will then be described along with other details of the experiments. Finally, the results of the experiments and analysis will conclude this paper along with future areas of research to decouple evolutionary algorithms from human interaction.

SASADEPS Methodology

It has recently been shown that steps to decentralizing the parent selection step of a typical EA cycle can produce results on par with previous centralized implementations. Smorodkina showed in the Self-Adaptive Semi-Autonomous Dictatorial Parent Selection (SASADIPS) algorithm that replacing the parent selection method with an independently evolved method for each individual decoupled the system from a human-determined method and instead could adapt to the situation. The experiments performed had each individual pick from basic selection primitives (e.g., higher fitness, more diversity, less uniformity) and previous selection methods (e.g., tournament selection, roulette wheel) and created decision trees from these primitives. This method was similar to Genetic Programming (GP) in that it had individuals create mating decision trees that would select one other mate for each individual and then during breeding the decision trees would also evolve.

This paper takes the next step from Smorodkina's research and allows the individuals to evolve decision trees that rank other individuals instead of allowing each individual

complete control of its mate. The SASADEPS methodology allows each individual to rank all other individuals based on their evolved decision tree and then the individuals which prefer each other the most will be paired together. In the previous method, individuals were paired based only on one of the pair's preferences, the new method incorporates preferences evolved from both individuals, thus giving allowing the individuals to more autonomously control their destiny. The algorithm used to pair all individuals together is described in the next section.

Stable Roommates Problem

Once all the individuals in the population have rated all other individuals based on their internally evolved decision tree, an algorithm must be used to optimally pair all individuals together. Gale and Shapley first present one such method for pairing individuals when they can be separated into two sets, such as pairing males and females based on their preferences [2]. Irving expands on this method by allowing a pairing of all individuals in a single set, rather than two distinct sets, based on their preferences towards each other [4].

The SASADEPS method uses an adaptation of Irving's stable roommates algorithm to optimally pair all individuals after they have ranked each other. On larger population sizes, the algorithm has an increasing chance of not being able to find a globally optimal matching for all individuals. When the algorithm detects that a matching cannot be made, a heuristic is used where all individuals are paired to their most preferred mate that has not already been taken.

1	4, 6, 2, 5, 3
2	6, 3, 5, 1, 4
3	4, 5, 1, 6, 2
4	2, 6, 5, 1, 3
5	4, 2, 3, 6, 1
6	5, 1, 4, 2, 3

Table 1: Example Preference List

An example of how a population of six individuals could rank each other is given in Table 1 where individual 1 prefers 4 over 6 and so on. The basic technique for matching individuals is to have each "propose" to their preferred mates in order and then following a set of rules on who accepts and rejects the proposals. In this example, the algorithm will progress to the second of two phases and detect a cycle in the reduced preference list, thus signaling that no stable matching exists. Again, the heuristic would then be used to roughly pair all individuals with others that they prefer.

Experiment Design

A set of function primitives were given to each individual for generation of its mate ranking decision tree. These primitives need to form a basic language that the individual can use to express what features of other individuals it prefers and what features it does not want to mate with. Basic arithmetic primitives and an If-Then-Else clause were used in these experiments along with basic feature values such as fitness, uniformity, and hamming distance. An example decision tree for an individual in the population is shown in Figure 1.

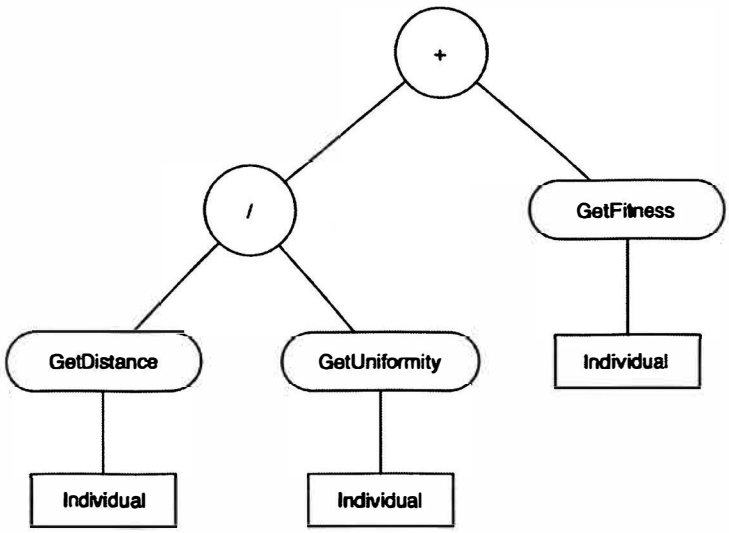

Figure 1: Example mate preference decision tree

To determine how the democratic semi-autonomous parent selection compares to Smorodkina's previous dictatorial method, experiments were run on the *onemax* and *trap* problems. The onemax problem consists of a bit-string of n bits where the solution to the problem is found when all bits are set to one. Although this problem is very simplistic to a logical mind, an EA has no domain knowledge about the problem and must essentially learn that an individual with more ones in it's genotype will be rewarded. The second problem used to gauge the EAs performance is known as the 4-bit deceptive trap problem. As the name suggests, in trap, a bit-string is again used but this time split into 4-bit sections. The values assigned to each 4-bit value are shown in Equation 1. The reason for the seemingly random value assignments is because the problem tries to trap the algorithm into local optimum instead of finding the global optimum of all ones.

$$\text{fitness}(\text{trap}_i) = \begin{cases} 3 & \text{if trap}_i \text{ contains zero 1's} \\ 2 & \text{if trap}_i \text{ contains one 1} \\ 1 & \text{if trap}_i \text{ contains two 1's} \\ 0 & \text{if trap}_i \text{ contains three 1's} \\ 8 & \text{if trap}_i \text{ contains four 1's} \end{cases} \quad (1)$$

The operators and their relevant parameter values are listed in Table 2.

Operator	Relevant parameters
Random initialization	population size: 50 each bit is initialized to either 0 or 1 with probability 0.5
Tournament parent selection	tournament size: 5
One point crossover	crossover point is uniform randomly selected
Bit flip mutation	each bit is flipped with probability 0.01%
Tournament competition	tournament size: 5 (bit string uniqueness is enforced); elitism size: 1

Table 2: Evolutionary operators and their parameters used in the experiments

Experimental Results

In the first experiment, both SASADEPS and SASADIPS implementations were run for 2000 generations with a starting maximum fitness of around 1500 for each of 30 trials of onemax. Both parent selection methods are compared over all generations in Figure 2.

In the second experiment, the SASADEPS and SASADIPS implementations were run for 1000 generations with a starting maximum fitness of around 450 for each of 30 trials of trap. Both parent selection methods are compared over all generations in Figure 3.

Both of these experiments were run with varying initial populations, with the same populations used for SASADIPS and SASADEPS. In the trap experiments, ten different initial populations were run three times each; in the onemax experiments there were thirty initial populations used.

Discussion

In both experiments, SASADIPS statistically outperformed SASADEPS in both *convergence quality* (the maximum fitness at the end of the experiment) and *convergence speed* (the rate at which the fitness increases). In Smorodkina's previous work with SASADIPS, the quality and speed of convergence closely matched that of a standard evolutionary algorithm. During the onemax problem, SASADEPS closely matched the previous EAs fitness

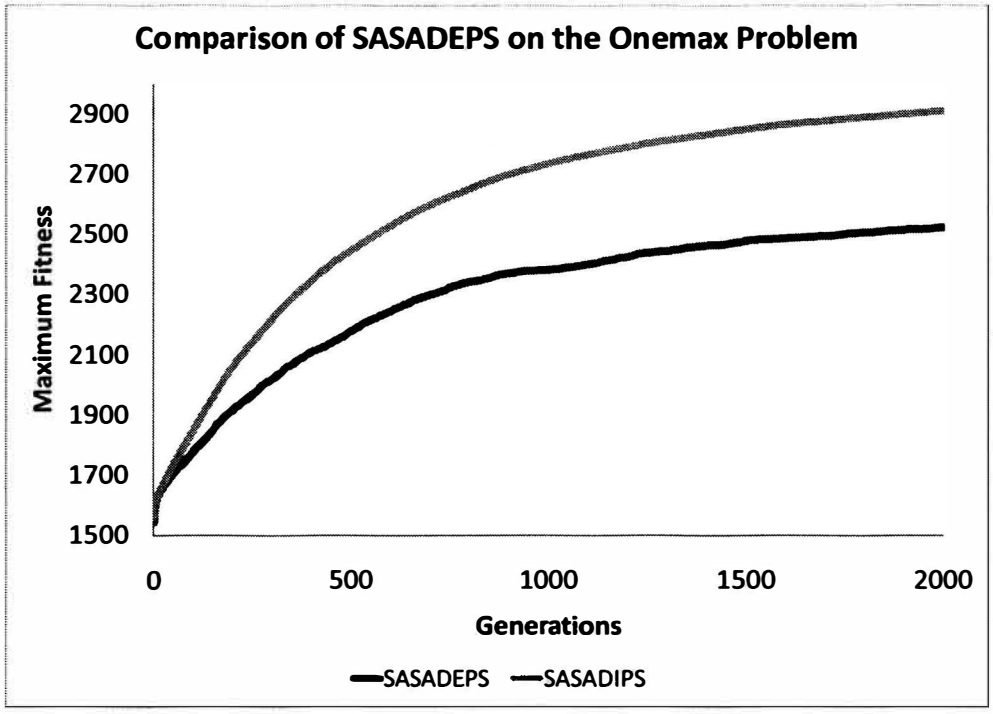

Figure 2: Comparison of SASADEPS and SASADIPS with Onemax Problem

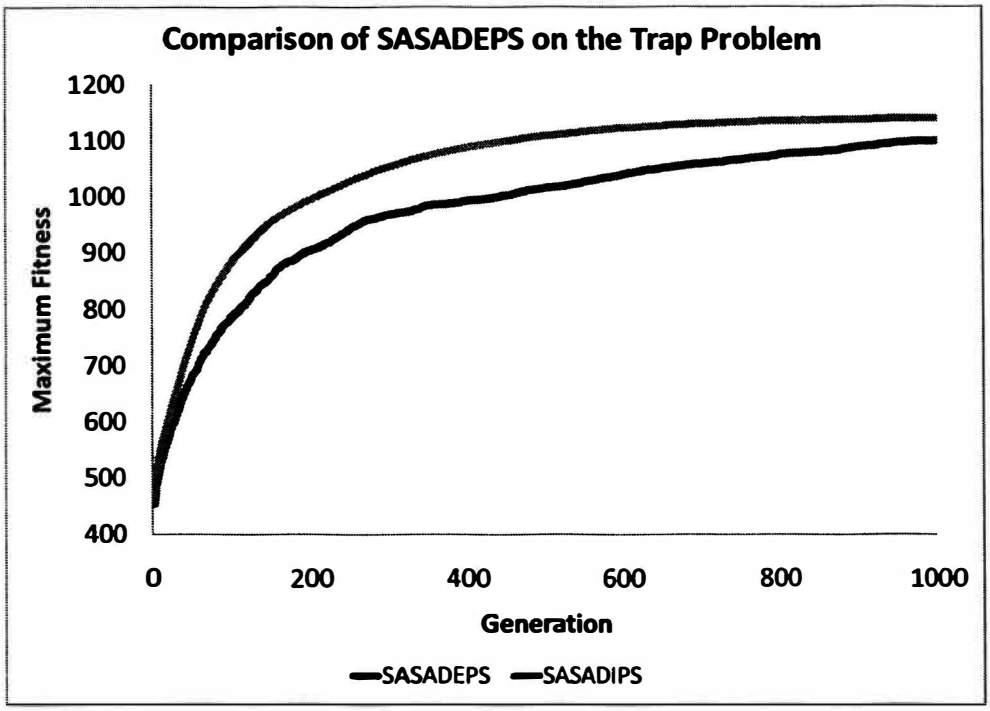

Figure 3: Comparison of SASADEPS and SASADIPS with Trap Problem

increase rate except for the beginning of the experiment where the convergence speed was not as fast. In the trap problem, however; the initial convergence speed matches that of SASADIPS, then decreases during the middle of the run and finally starts to increase again while the population using the SASADIPS method has slowed and gotten stuck in a local optimum.

It is significant to note that the trap experiment continued to show significant increases of fitness until the end of the trial. Typically, populations in EAs struggle to not get stuck in local optima due to a lack of diversity in later generations. Because individuals can continuously evolve their mate selection algorithms with more control in SASADEPS than in SASADIPS, the continual increase in fitness may be due to individuals adapting to the fitness landscape.

Conclusion and Future Work

The results show promise that control can be taken away from centralized, pre-determined algorithms and instead evolved in real time for each individual. The individuals in the SASADEPS experiments are mainly limited by their decision tree operators, or vocabulary. Without a broad enough language, rating algorithms equal to or significantly better than standard parent selection techniques will be unable to evolve. Based on the initial results shown in this paper, further experimentation with additional operators and primitives in the decision trees along with optimal GP tree size should bring autonomous parent selection closer to competing or surpassing current methods.

Future work will be focused on developing new primitives for the decision tree which will hopefully make the SASADEPS more competitive with current evolutionary algorithm parent selection methods. Additional testing will be needed on a wider array of problems in addition to adding support for problems that require representations other than bit-strings. After increasing the robustness of the SASADEPS method, a paper will also be submitted to either an EA conference or journal for publication.

Acknowledgements

I would like to thank Dr. Tauritz for helping me find a research project that motivates me and keeping me motivated throughout the project. I would also like to thank Kate for helping me learn all the details of the programming framework and working with me to develop the methods and techniques used throughout the research.

References

[1] T. Bäck. *Evolutionary Computation 2*, volume chapter 21 of *pages 188-211.* Institute of Physics Publishing, 2000.

[2] D.Gale and L. S. Shapley. College admissions and the stability of marriage. *The American Mathematical Monthly*, 69(1):9 15, January 1962.

[3] J. Gomez. Self adaptation of operator rates in evolutionary algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 1162 1173, June 2004.

[4] R.W. Irving. An efficient algorithm for the "stable roommates" problem. *Journal of Algorithms*, 6:577 595, 1985.