Spector, L., T. Helmuth, and K. Harrington. 2011. Fecundity and Selectivity in Evolutionary Computation. In *GECCO'11 Posters, Genetic and Evolutionary Computation Conference*. ACM Press. pp. 129-130.

Fecundity and Selectivity in Evolutionary Computation

Lee Spector
Cognitive Science
Hampshire College
Amherst, MA 01002 USA
lspector@hampshire.edu

Thomas Helmuth
Computer Science
University of Massachusetts
Amherst, MA 01003 USA
thelmuth@cs.umass.edu

Kyle Harrington
Computer Science
Brandeis University
Waltham, MA 02453 USA
kyleh@brandeis.edu

ABSTRACT

The number of offspring produced by each parent—that is, the fecundity of reproducing individuals—varies among evolutionary computation methods and settings. In most prior work fecundity has been tied directly to selectivity, with higher selection pressure giving rise to higher fecundity among individuals selected to reproduce. In nature, however, there is a wider variety of strategies, with different organisms producing different numbers of offspring under the influence of a range of factors including not only selection pressure but also other factors such as environmental stability and competition within a niche. In this work we consider possible lessons that may be drawn from nature's approaches to these issues and applied to evolutionary computation systems. In particular, we consider ways in which fecundity can be dissociated from selectivity and situations in which it may be beneficial to do so. We present a simple modification to the standard evolutionary algorithm, called *decimation*, that permits high fecundity in conjunction with modest selection pressure and which could be used in various forms of evolutionary computation. We also present a simple example, showing that decimation can improve the problem-solving performance of a genetic algorithm when applied to a deceptive problem.

Categories and Subject Descriptors

I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search—*Heuristic methods*; I.2.2 [**Artificial Intelligence**]: Automatic Programming—*Program synthesis*

General Terms

Algorithms

Keywords

Selection, fecundity, truncation, decimation, Deb's deceptive problem

1. DECIMATION

Natural organisms exhibit a wide range of reproductive strategies. Reproductive strategies vary in several different ways, with one being that typical members of some species produce large numbers of offspring while typical members of other species produce relatively small numbers of offspring. This is a difference in *fecundity*. Presumably these variations in fecundity are adaptive, producing greater overall reproductive success than alternatives would produce in the same ecological circumstances. Variation of fecundity, and of reproductive strategy more generally, is widely discussed in the biological literature (e.g. [3]) but it has not yet received the same attention in evolutionary computation. To the best of our knowledge no major approaches in evolutionary computation provide the kind of flexible decoupling of fecundity and selectivity that nature permits. Most approaches couple fecundity and selectivity explicitly, with configurations that exert greater selection pressure leading to higher fecundity and with low fecundity being an unavoidable by-product of low selection pressure. We argue, both on the basis of biological evidence and on the basis of an experiment with a deceptive problem from the evolutionary computation literature, that this is a limitation of evolutionary computation that we should work to overcome.

Here we describe a simple modification to the standard evolutionary algorithm called *decimation*.[1] Decimation is a step that occurs during each generation of an evolutionary algorithm (EA) between fitness testing and reproduction. It is used for *survivor selection* (known sometimes as *replacement*), an aspect of an EA that is highlighted in some dialects of evolutionary computing but largely ignored in others. Decimation performs survivor selection by discarding all parents and then substantially decreasing the size of the offspring population from which new parents will be chosen. Because the pool of potential parents is significantly smaller than the pool of created offspring, the surviving individuals will likely each produce multiple offspring. The reduction of the population is performed in a way that favors high fitness individuals but does not rule out the survival of some relatively low fitness individuals. We accomplish this by performing "elimination tournaments" in which some number of individuals (typically 2) are selected and the individuals with better fitness are retained while the individual with the worst fitness is removed from the population. This process is iterated until the population has been reduced to the required size (typically 10% of the original population size). Subsequently, during parent selection, we choose individuals from the decimated population uniformly at random. This

[1]This technique is sufficiently similar to the technique called decimation by Koza [2] that we thought it was reasonable to use the same name, although there are differences in motivation and algorithmic details.

GECCO'11, July 12–16, 2011, Dublin, Ireland.
ACM 978-1-4503-0690-4/11/07.

means that all individuals that survive decimation will have an equal chance of producing a large "litter." While overall selectivity will be relatively low (because of the low tournament size that we generally use in the elimination tournaments), fecundity will nonetheless be high because each surviving individual—some of which may have relatively poor fitness—will produce many offspring. Decimation resembles truncation, a survivor selection method used in various evolutionary algorithms—it is known as (μ, λ) selection in evolution strategies—but in contrast to truncation, decimation lets some less-fit individuals reproduce.

2. DEB'S DECEPTIVE PROBLEM

To demonstrate one case in which decimation may be useful, we present some results using a traditional bit-string genetic algorithm on a straightforward deceptive problem from the literature. We use "Deb's deceptive 4-bit function" problem [1, p. 269] with a 40-bit genome that is assessed in consecutive 4-bit blocks.[2] For each block an error value from 0 to 5 is computed, and the error for the entire individual (its fitness, where lower is better) is the sum of the individual block errors. The error of a block is 0 if the block is all 1s, but otherwise it is higher the more 1s that the block contains; this is what makes the problem deceptive. In particular, we assign an error of 2 for a block of all 0s, an error of 3 for a block with a single 1, an error of 4 for a block with two 1s, and an error of 5 for a block with three 1s. We conducted $1,000$ runs in each condition with a population size of 500 and a maximum of 501 generations (the initial, random population and then 500 generations of reproduction). We used only mutation (no crossover), with our mutation operator flipping one bit chosen at random. Each generation 95% of the population was produced from selected parents by mutation while the remaining 5% was cloned, unchanged, from a selected parent. We conducted runs under three selection regimes: tournament selection (with various tournament sizes), decimation, and truncation. In the tournament selection runs each parent was chosen using an independent tournament among randomly selected members of the parental generation. In the decimation runs we ran elimination tournaments to reduce the population to 10% of its original size and then selected parents randomly from the surviving individuals. In the truncation runs we randomly chose parents from among the top 10% of the population. Figure 1 shows the number of successes achieved over the $1,000$ runs in each condition. Tournament selection works quite well with the right tournament size—the best is tournament size 4, which succeeded 948 times—but it performs poorly in with other tournament sizes and abysmally (34 successes) with tournament size 2. Truncation also performs quite poorly here, succeeding only 165 times. Decimation performs the best, succeeding 998 times out of $1,000$. Analysis (not shown here) with respect to Koza's "computational effort" statistic [2] supports this conclusion, as do results on a 5-bit version of this problem.

3. CONCLUSIONS AND FUTURE WORK

Our results (including many that cannot be shown here) make the case that evolutionary computation researchers should give greater attention to the relationship between fecundity and selectivity. We presented a simple method, called decimation, for adding one type of control for fecundity to an evolutionary computation system. We have also shown how decimation improved the performance of a simple bit-string genetic algorithm applied to a deceptive problem from the literature.

[2]Our formulation here is different than that in [1], but the problem is formally equivalent.

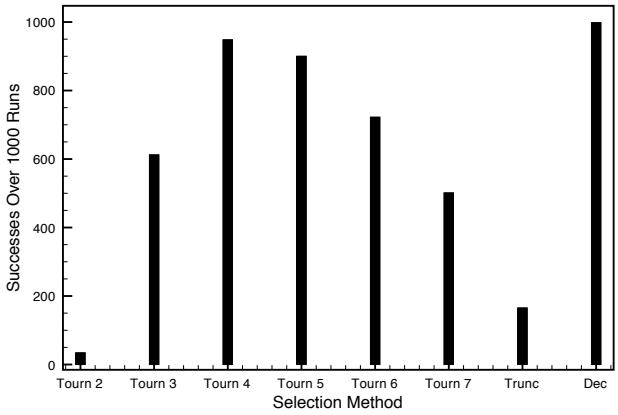

Figure 1: Numbers of successes, over 1000 runs in each condition, for Deb's deceptive 4-bit problem with various selection methods. "Tourn n" is standard tournament selection with tournament size n. "Trunc" is truncation to 10% of the population size. "Dec" is decimation to 10% of the population size.

Acknowledgments

Thanks to Daniel Gerow, Brian Martin, Nathan Whitehouse, Rebecca Neimark , Jordan Pollack, Emma Tosch, and other members of the Brandeis DEMO lab for feedback that helped us to improve this work in several ways, to Josiah Erikson for systems support, and to Hampshire College for support for the Hampshire College Institute for Computational Intelligence. This material is based upon work supported by the National Science Foundation under Grant No. 1017817. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the National Science Foundation.

4. REFERENCES

[1] A. E. Eiben and J. E. Smith. *Introduction to Evolutionary Computing*. SpringerVerlag, 2003.

[2] J. R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, Cambridge, MA, USA, 1992.

[3] R. H. MacArthur and E. O. Wilson. *The theory of island biogeography*. Princeton University Press, April 2001.