Genetic and Evolutionary Algorithms in the Real World

David E. Goldberg
Department of General Engineering
IlliGAL Report No. 99013
March 1999

Department of General Engineering
University of Illinois at Urbana-Champaign
117 Transportation Building
104 South Mathews Avenue
Urbana, IL 61801

Genetic and Evolutionary Algorithms in the Real World*

David E. Goldberg
Department of General Engineering
University of Illinois at Urbana-Champaign
Urbana, IL 61801
deg@uiuc.edu
http://www-illigal.ge.uiuc.edu/

1 Introduction

Since 1992, I have made regular trips to Japan to give talks about genetic algorithms (GAs)—search procedures based on the mechanics of natural selection and genetics. Back during my first visit, the use of genetic and evolutionary algorithms (GEAs) was restricted to a relatively small cadre of devoted specialists. Today, Japanese researchers and practitioners are ably advancing the state of GEA art and application across a broad front. Around the globe, from traditional and cutting-edge optimization in engineering and operations research to such non-traditional areas as drug design, financial prediction, data mining, and the composition of poetry and music, GEAs are grabbing attention and solving problems across a broad spectrum of human endeavor. Of course, science and technology go through fads and fashions much like those of apparel, food, and toys, and many practitioners—in Japan and elsewhere—are wondering whether GEAs, like so many methods that have come and gone in the past, will become a permanent part of the toolkit or will fade like some computational hoola hoop du jour. Thus, I believe it is quite important that we pose some tough questions for the present and future state of genetic and evolutionary computation (GEC), and in this short essay I ask and answer three questions:

1. Why have GEAs become so popular?
2. Is this popularity real, or is it just another fad?
3. What will the future hold for GEA technique and application?

In what follows, I will argue that the current popularity of GEAs is no accident, no fad, and that genetic algorithms—indeed all forms of genetic and evolutionary computation (GEC)—are here to stay and will play an increasingly important role in helping people innovate in many walks of life. This may seem like a strong assertion, especially to those practitioners who have had both positive and negative experiences with genetic algorithms, but cutting-edge research suggests that the techniques that are currently in widespread use are only the tip of the iceberg, and that the generation of GEAs that is currently in the lab promises relief from problems of scale up that some users have suffered in going from toy problems to real, industrial-strength problems. Moreover, as in so many other issues in the arena of applications, the primary determinants are often *economic*, not technical, and there, too, GEAs have much to offer.

*Portions of this essay are excerpted from Goldberg (in press-a; in press-b)

In the remainder, I start by offering the briefest of introductions to genetic and evolutionary computation. Thereafter, I will argue that one reason we find GEAs so appealing is that they connect quite directly with our own sense of human *innovation*. I will distinguish between different *modes* of innovation, and I believe that this is one of the reasons we find the methodology so appealing. I continue by answering the three questions above by appealing to five different user *motives* for using GEAs. Why, after all, do real users with real problems choose to use genetic and evolutionary algorithms? By closely examining each of the five motives, we will better understand why people are using GEAs today, and why they will continue to use GEAs indefinitely into the future.

2 The One Minute Genetic-Evolutionary Algorithmist

Elsewhere, I have written at length (Goldberg, 1989) about GEA basics, and in this section we quickly review the fundamental mechanics by discussing what GAs process and how they process it.

Suppose we are seeking to find a *solution* to some *problem*. To apply a genetic algorithm to that problem, the first thing we must do is encode the problem as an artificial *chromosome* or chromosomes. These artificial chromosomes can be strings of 1s and 0s, parameter lists, or even complex computer codes, but the key thing to keep in mind is that the genetic machinery will manipulate a finite representation of the solutions, not the solutions themselves. Another thing we must do in solving a problem is to have some means or procedure for discriminating good solutions from bad solutions. This can be as simple as having a human intuitively choose better solutions over worse solutions, or it can be an elaborate computer simulation or model that helps determine what good is. But the idea is that *something* must determine a solution's relative *fitness to purpose*, and whatever that is will be used by the genetic algorithm to guide the evolution of future generations.

Having encoded the problem in a chromosomal manner and having devised a means of discriminating good solutions from bad ones, we prepare to *evolve* solutions to our problem by creating an initial *population* of encoded solutions. The population can be created randomly or by using prior knowledge of possibly good solutions, but either way a key idea is that the GA will search from a population, not a single point.

With a population in place, *selection* and *genetic operators* can process the population iteratively to create a sequence of populations that hopefully will contain more and more good solutions to our problem as time goes on. There is much variety in the types of operators that are used in GAs, but quite often (1) *selection*, (2) *recombination*, and (3) *mutation* are used.

Simply stated, selection allocates greater survival to better individuals—this is the survival-of-the-fittest mechanism we impose on our solutions. This can be accomplished in a variety of ways. Weighted roulette wheels can be spun, local tournaments can be held, various ranking schemes can be invoked, but however we do it the main idea is to *prefer better solutions to worse ones*. Of course, if we were to only choose better solutions repeatedly from the original database of initial solutions, we would expect to do little more than fill the population with the best of the first generation. Thus, simply selecting the best is not enough, and some means of creating new, possibly better individuals must be found; this is where the genetic mechanisms recombination and mutation come into play.

Recombination is a genetic operator that *combines bits and pieces of parental solutions* to form, new, possibly better offspring. Again, there are many ways of accomplishing this, and achieving competent performance does depend on getting the recombination mechanism designed properly; but the primary idea to keep in mind is that the offspring under recombination will not be identical to any particular parent and will instead *combine parental traits in a novel manner*. By itself, recombination is not all that interesting of an operator, because a population of individuals processed under repeated recombination alone will undergo what amounts to a random shuffling of extant traits.

Where recombination creates a new individual by recombining the traits of two or more parents, mutation acts by simply modifying a single individual. There are many variations of mutation, but the main idea is that the offspring be identical to the parental individual except that *one or more changes is made to an individual's trait or traits* by the operator. By itself mutation represents a "random walk" in the neighborhood of a particular solution. If done repeatedly over a population of individuals, we might expect the resulting population to be indistinguishable from one created at random.

3 The Fundamental Intuition

The previous section described the mechanics of a genetic-evolutionary algorithm, but it gives us little idea of why these operators might promote a useful search. To the contrary, individually we saw how the operators acting alone were ineffectual, and it is something of an intellectual mystery to explain why such individually uninteresting mechanisms acting in concert might together do something useful. Starting in 1983 (Goldberg, 1983), I have developed what I call the *fundamental intuition of genetic algorithms* or the *innovation intuition* to explain this apparent mystery. Specifically, I liken the processing of selection and mutation together and that of selection and recombination taken together to *different facets of human innovation*, what I will call the *improvement* and *cross-fertilizing* types of innovation. We start first with the combination of selection and mutation and continue with the selection-recombination pair.

3.1 Selection + Mutation = Continual Improvement

When taken together, selection and mutation are a form of hillclimbing mechanism, where mutation creates variants in the neighborhood of the current solution and selection accepts those changes with high probability, thus climbing toward better and better solutions. Human beings do this quite naturally, and in the literature of total quality management this sort of thing is called *continual improvement* or as it is called in Japanese, *kaizen*. When I first introduced the innovation intuition, it was largely based on introspection, but others have had similar thoughts, for example the British author and politician Bulwer-Lytton (Asimov & Shulman, 1988):

> Invention is nothing more than a fine deviation from, or enlargement on a fine model. ...Imitation, if noble and general, insures the best hope of originality.

Although this qualitative description is a far piece from an algorithmic one, we can hear the echo of mutation and selection within these words. Certainly, continuing to experiment in a local neighborhood is a powerful means of improvement, although it will have a tendency to be fairly local in scope, unless a means can be found for intelligently jumping elsewhere when a locally optimal solution is found.

3.2 Selection + Recombination = Innovation

One way of promoting this kind of intelligent jumping is through the combined effect of selection and recombination, and we can start to understand this if we liken their effect to that of the processes of human cross-fertilizing innovation. What is it that people do when they are being innovative in a cross-fertilizing sense? Usually they are grasping at a notion—a set of good solution features—in one context, and a notion in another context and juxtaposing them, thereby speculating that the combination might be better than either notion taken individually. Again, my first thoughts on the subject were introspective ones, but again others have written along similar veins, for example, the French mathematician Hadamard (1945):

> We shall see a little later that the possibility of imputing discovery to pure chance is already excluded....Indeed, it is obvious that invention or discovery, be it in mathematics or anywhere else, takes place by combining ideas.

Likewise, the French poet-philosopher Valéry had a similar observation:

> It takes two to invent anything. The one makes up combinations; the other chooses, recognizes what he wishes and what is important to him in the mass of the things which the former has imparted to him.

Once again, verbal descriptions are far from our more modern computational kind, but something like the innovation intuition has been clearly articulated by others.

With a basic understanding of the mechanics of genetic algorithms and an intuitive understanding of their power, we now examine some of the motives for using GEAs in the real world.

4 Five Motives to Use GEAs

In the previous section, I suggested that GEAs appeal to our own sense of innovation, and this is indeed a strong motive for our adopting these procedures, but more practically what motivates a user to use genetic and evolutionary algorithms? Certainly there are as many answers to this question as there are GEA users, but some generalizations can be made. Here I identify motives of five types:

1. Motives from the buzz
2. Motives from nature
3. Motives from artificial systems
4. Motives from competence
5. Motives from economics

In the remainder of this section, we consider each of these in somewhat more detail.

4.1 Motives from Buzz

One of the first things that attracts new users to GEAs is what I will call the "buzz." GEAs and genetic and evolutionary computation in general are receiving media attention, both print and electronic, and various accounts of GEA discovery and invention are reverberating through popularizations of artificial life and complex systems. These accounts are often what attracts new users to the field, but *man does not solve problems by buzzwords alone.* At some point, a problem must be posed, methods engaged, and results obtained, so motives from the buzz—while helpful in attracting new users—do little to retain them.

4.2 Motives from Nature

The buzz of excitement draws us to GEAs, but what can keeps ou attention? One of the factors that certainly holds us is the *scientific reasonableness* of the endeavor. Since Darwin, we take it for granted that life on this planet in all its diverse and well adapted forms was created by natural selection and natural genetics. With this understanding comes the inkling that perhaps we might be able to *use* nature's "search algorithm of choice" and apply it to the solution of humankind's problems. Having

this thought and making it work are two different things; yet, the inkling is important because it acts as something of an *existence proof* to let us know that we are on the right track even though we haven't yet engineered the ultimate genetic algorithm. Surely, people have dreamt of human flight from their first observations of birds, and for many years all attempts were doomed to failure. The knowledge that something *could* fly certainly played the dual role of (1) providing specific inspiration for the design details of an airplane and (2) sustaining inquiry and continued trials, especially as the failures mounted. In the same way, researchers and practitioners are inspired by nature's example and are impelled to continue even when their efforts don't turn out as they wish.

4.3 Motives from Artificial Systems

Nature as a source of ideas and an existence proof provides inspiration and solace, largely for the GEA designer and researcher, but the practitioner's motives are rooted in the limitations of traditional optimization and operations research methods. On the one hand, there are a large number of such methods available. For example, when you have a linear problem with linear constraints, you can grab linear programming. When you have a stage-decomposable problem you can grab dynamic programming. When you have a nonlinear problem with nonlinear constraints, you can (sometimes) grab nonlinear programming, and so on. But the fact that you have a list of acceptable *methods* for particular *problem classes* is itself part of the problem. Traditional methods are well tuned to a particular problem class, but when a problem comes along that violates the assumptions of such methods, solution results can be particularly disappointing. Wouldn't it be nice if artificial search and optimization procedures would work well over a broader class of problems? Artificial genetic and evolutionary methods are a potential answer to this yearning, because the evolution of natural systems takes place via mechanisms that are in many ways invariant across species, and in so doing nature uses the same or similar search procedure almost regardless of environment. Many users turn to GEAs and GEC for exactly this breadth of solution quality with reasonable efficiency.

4.4 Motives from Competence

The promise of quality and efficiency—the promise of *robustness*—has indeed attracted many practitioners to GEAs, but for some of them, a funny thing happened on the way to their applications. At first, when working with small toy problems in their application domain, the GEA works quite well, but when they turn to larger or harder problem instances they find that solution times increase, solution quality decreases, or both. The response of different users to these problems of *scale up* are many. Some fiddle around with operators or codings, trying different possibilities, until something works. Others abandon genetic and evolutionary computation entirely, quite frustrated with the whole affair. Others still simply remain puzzled, and question why such ostensibly robust algorithms exhibit such poor scale-up behavior.

For years these difficulties were swept under the rug, but we now know that simple genetic and evolutionary algorithms with fixed crossover and mutation operators are fairly limited in what they can do (Thierens & Goldberg, 1993). Mathematical analyses have been performed to support this assertion fairly convincingly, and this would seem to be a deal breaker if it weren't for companion results that show that adaptive and self-adaptive operators can overcome these difficulties quite effectively. Here at the University of Illinois, we have pioneered the development of so-called *competent* genetic algorithms—GAs that solve hard problems, quickly, reliably, and accurately.

The development of the *messy GA* in 1989 (Goldberg, Korb, & Deb, 1989) led to the first competent GA, the so-called *fast messy GA* (Goldberg, Deb, Kargupta, & Harik, 1993), and the fmGA has spawned a whole family of offspring, including the gene expression messy GA (Kargupta, 1996), the

linkage learning GA (Harik, 1997), the linkage detection GA using nonlinearity detection (Munetomo, & Goldberg, in press), and the Bayesian optimization algorithm or BOA (Pelikan, Goldberg, & Cantú-Paz, in press) to name a few. These implementations have not been well integrated into practice yet, but as more and more practitioners become aware of them, the frustration with the problems of scale up will become decidedly less. Moreover, as these new operators take their place in everyday GEA practice, users will be surprised to find that hard problems can be solved reliably and accurately in times that may grow no more quickly than a quadratic function of the number of decision variables.

4.5 Motives from Economics

The foregoing discussion has given a number of fairly sophisticated reasons why users are motivated to use genetic algorithms, but for many practitioners the bottom line is often the *bottom line*. That is, practitioners are often interested in receiving economic benefits from the performance of a genetic optimization. In many cases, the economic prime movers are fairly direct. Using genetic and evolutionary algorithms may enable a practitioner to optimize or improve a system that is otherwise not amenable to algorithmic improvement, thereby resulting in a direct economic benefit from the use of the GEA. In other circumstances, the economic benefits are somewhat less direct, but they may be critical to the choice of a GEA nonetheless. We examine three such circumstances briefly:

1. economics of investment in method
2. economics of model investment
3. economics of GEA speedup

One economic reason that users turn to GEAs has to do with their investment in optimization methods. If one has limited resources and is concerned with computing improved solutions to problems with either (1) a broadly competent method such as a GEA, or (2) a panoply of disparate techniques from OR or traditional optimization, the investment necessary to learn and use a single broad method should be lower than that associated with a collection of techniques. In the case of a collection of techniques, not only must many different methods be mastered, but the user must also learn when to choose which technique. These costs can add up, and other things being equal, the user may prefer to trade off the use of a perfectly tuned solver for one that does an adequate job without additional investment in knowledge of method.

Method investment costs can be significant, but for many users the lion's share of investment is tied up in *modeling* or *simulation*. Most complex optimization involves a fairly sophisticated objective function that may itself rely on finite-element models, approximations to the solutions of nonlinear equations, discrete-event simulations, or the like. Prior to using such models for optimization or design, users expend considerable time and effort inputing data, running test cases, tuning the model to agree with the real world, and then using the models for analysis. After such a large investment in modeling, no user likes to be told that in order to perform an optimization, the model must be shoehorned into a form preferred by a particular optimization method, but many optimization methods require exactly this kind of model transformation. Genetic algorithms, on the other hand, take their function evaluations as they come, thereby respecting the significant investment that users may have in analysis code, using that code without substantial modification or transformation.

This laissez faire attitude toward function evaluations comes at a cost, however. Because GEAs make relatively few assumptions about the solution space, and because the interface between GEA and evalution involves only the passing of function evaluation values (no derivatives or higher order information), a GEA solution may require hundreds or thousands of function evaluations. As was suggested earlier, this number can be reduced through the use of competent GEAs to times that may be

as good as subquadratic, but nonetheless, in large problems, fairly large numbers of function evaluations will be necessary. By itself, this would be cause for some concern if there weren't corresponding ways to ways to speed up the GEA itself through various *efficiency improvements*, including (1) space utilization (parallelization), (2) time utililzation, (3) evaluation relaxation, and (4) hybridization. Advances are begin made rapidly along all these fronts (Cantú-Paz & Goldberg, 1999, Goldberg, 1999; Goldberg & Voessner, 1999; Miller & Goldberg, 1996), and practitioners should soon expect to see practical means of speeding their solutions day in and day out.

5 Toward a Golden Era of Computational Innovation and Creativity

Beyond the use of competent and efficient GEAs, lies a new era, which only has been hinted at by existing efforts. If, as I suggest, GEAs do indeed represent a form of computational innovation, and if GEAs are indeed becoming competent in the sense that hard problems will be solved quickly, reliably, and accurately as a simple matter on a day-to-day basis, then it seems that we are embarking on a time, when *human intellect will be regularly augmented by innovation of a computational kind.* This is an exciting prospect—but it can be a scary one, too. In this section, I consider some of the ramifications of this coming era *of computational innovation and creativity.* Specifically, I predict that

1. Competent GEAs will become an ordinary part of corporate problem solving.
2. Creative algorithms will be developed using extensions of the techniques, analyses, and methodology used to design competent GEAs.
3. Winners and losers in business will be determined at least partially by rapid mastery of these techniques.
4. Computational innovation and creativity will have scientific importance beyond their utility as problem solvers.

In the remainder, I consider each of these points in somewhat more detail.

First, it is becoming increasingly clear that GEAs are joining the repetoire of everyday business tools. As I have traveled around the globe, many people have used the analogy of *finite element methods* (FEM) in describing the importance of GEAs. FEM has permitted the regular everyday solution of just about any field problem from electromagnetics, to fluid mechanics to solid mechanics and any combination thereof. Practitioners are starting to think of GEAs as a similarly regular kind of computational tool when design or innovation is the object, and this trend will continue.

Second, I believe that the strides that have been made in GEA theory, method, and implementation are just the beginning. Though important, the techniques at our command right now, I believe, are a mere shadow of the kind of thing that will be available in the coming months and years. Proving this without demonstration is difficult if not impossible, but our language helps us know that there is more for us to discover. I believe it is correct to talk about existing GEAs as being a form of computational innovation as I discussed earlier, but it would be wrong to talk about existing GEA as being *creative* in any human sense. We humans seem to reserve the word "creative" as a category that goes beyond innovative, but in what way? I would suggest that the word "creativity" is reserved for people and things that are able to *transfer* knowledge from one domain to another. A creative book or piece of art is often one that alludes to other works, and thereby brings over thoughts and notions from other domains. A creative scientific discovery is one that is inobvious, oftentimes having been created through the transfer of a seemingly unrelated idea. In a similar manner, I believe that we shall shortly have *creative algorithms* that transfer ideas from one domain to another, and these ideas will build

upon the methods of GEAs in important ways, perhaps integrating with neural, fuzzy, and other soft computation to give us systems more powerful that what we can now imagine.

Third, if these first two suggestions are correct, it is quite clear that it will be increasingly important that businesspeople embrace these methods quickly and enthusiastically. Companies and workers who do will be rewarded with *competitive advantage* over those who don't. At first, specialists will be responsible for the corporate computational innovation and creativity function, but as the tools become better, this function will be distributed down to the point of application. At the present moment, companies should examine themselves to see where GEA has been embraced and how to encourage its spread. GEA education will be a critical need in these early years, especially since these concepts are not yet widely taught in the undergraduate experience.

Finally, I believe that GEA and its extension will have important *scientific* impact beyond the utility of solving problems. If genetic and evolutionary computation is helping us to devise innovative and possibly creative algorithms, it seems that there are whole areas of scientific research that would benefit from using these algorithms as *models* of human innovation and creativity. All of the social sciences, for example, attempt to model what people do with one another in matters of conflict, exchange, social interaction, and mental life. But at present these studies proceed without explicit modeling of human innovation and creativity, and the models of GEA and beyond should help fill this important gap. Indeed, artificial agents in economics, political science, and elsewhere are starting to show the way; I believe the trend is toward (1) increasing utilization of our models in these scientific endeavors and (2) increasing understanding that mechanistic understanding of innovative and creative processes fundamentally alters the social sciences and related disciplines. Moreover, I believe that biologists will come—somewhat belately—to understand that GEA research, despite its utilitarian approach—no, because of its utilitarian approach—has helped illuminate critical biological processes that were not properly understood through direct scientific inquiry alone.

6 Conclusions

I started this essay by trying to understand whether GEAs are some passing fad or fancy, or whether they will become a permanent part of the problem-solving toolkit. I continued by reviewing a basic GEA, and trying to tie some of the processes therein to a kind of computational innovation. Thereafter, I examined five facets or dimensions of user motivation, including motives from the buzz, from nature, from artificial systems, from competence, and from economics. Surely the real user is motivated by some combination of these factors and perhaps many others. Initially users are drawn to GEAs by some combination of the first three of these reasons, but they stay for hard-headed reasons of competence, economics, or both.

I suggested that many of the first-generation evolutionary and genetic algorithms currently in use are incapable of solving hard problems, quickly, reliably, and accurately; in short, they don't scale up. This would be bad news if it weren't for cutting-edge research at Illinois and elsewhere that shows us how to design GEAs that overcome these difficulties. Beyond the design of such *competent* genetic algorithms, users come and stay with GEAs for a variety of good economic reasons. Certainly GEAs can help directly impact the economics of design by giving us better or more cost-effective designs as the output of the optimization process. Beyond such direct impacts, users come and stay with GEAs because they can reduce investment costs in methods development, because they can fully utilize existing investment in modeling and simulation, and because they can be extended to provide quality solutions more efficiently through parallelization, time utilization, relaxed function evalution, and hybridization. Together, these factors suggest that GEAs will become—are becoming—a permanent part of the designer's tool kit.

Beyond these immediate concerns, I have suggested that these methods are taking us toward a golden era of computational innovation and creativity where GEAs are used as a commonplace and newer techniques increase the innovative-creative horsepower even further. I have even suggested that these utilitarian inventions will be helpful in delivering new knowledge in the social and biological sciences. Although these possibilities are exciting, this new era is not without risk. Those who fail to master the new technology may find themselves at a disadvantage to those who do. Therefore, perhaps the wise course of action for enlightened companies and knowledge workers in Japan and around the globe is to actively—and quickly—seek mastery of this growing, exciting technology.

Acknowledgments

My research was sponsored by the Air Force Office of Scientific Research, Air Force Materiel Command, USAF, under grant F49620-97-1-0050. Research funding for my work was also provided by a grant from the U. S. Army Research Laboratory under the Federated Laboratory Program, Cooperative Agreement DAAL01-96-2-0003. The U. S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon.

The views and conclusions contained herein are my own and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Office of Scientific Research, U. S. Army Research Laboratory, or the U. S. Government.

References

Asimov, I., & Shulman, J. (Eds.) (1988). *Isaac Asimov's book of science and nature quotations.* New York: Weidenfeld & Nicolson.

Cantú-Paz, E., & Goldberg, D. E. (1997). Modeling idealized bounding cases of parallel genetic algorithms. Genetic Programming 1997: Proceedings of the Second Annual Conference, 353–361.

Goldberg, D. E. (1983). *Computer-aided pipeline operation using genetic algorithms and rule learning.* Doctoral dissertation, University of Michigan, Ann Arbor.

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization, and machine learning.* Reading, MA: Addison-Wesley.

Goldberg, D. E. (in press-a). A meditation on the application of genetic algorithms. In Y. Rahmat-Samii & E. Michielssen (Eds.), *Electromagnetic system design using evolutionary optimization: Genetic algorithms.* New York: Wiley.

Goldberg, D. E. (in press-b). The race, the hurdle, and the sweet spot. In P. Bentley (Ed.), *Evolutionary design by computers.* London: Academic Press.

Goldberg, D. E. (in press-c). Using time efficiently: Genetic-evolutionary algorithms and the continuation problem. *GECCO-99: Proceedings of the 1999 Genetic and Evolutionary Computation Conference.* (Also published as IlliGAL Report No 99002)

Goldberg, D. E., Deb, K., Kargupta, H., & Harik, G. (1993). Rapid, accurate optimization of difficult problems using fast messy genetic algorithms. *Proceedings of the Fifth International Conference on Genetic Algorithms,* 56-64. (First published in 1993 as IlliGAL Report No. 93004)

Goldberg, D. E., Korb, B., & Deb, K. (1989). Messy genetic algorithms: Motivation, analysis, and first results. *Complex Systems, 3*(5). 493-530. (First published in 1989 as TCGA Report No. 89003)

Goldberg, D. E., & Voessner, S. (in press). Optimizing global-local search hybrids. *GECCO-99: Proceedings of the 1999 Genetic and Evolutionary Computation Conference.* (Also published as IlliGAL Report No. 99001)

Hadamard, J. (1945). *The psychology of invention in the mathematical field.* Princeton University Press: Princeton, NJ.

Harik, G. (1997). *Learning gene linkage to efficiently solve problems of bounded difficulty using genetic algorithms.* Doctoral dissertation, University of Michigan, Ann Arbor. (Also published as IlliGAL Report No. 97005)

Kargupta, H. (1996). *SEARCH, evolution, and the gene expression messy genetic algorithm* (Unclassified Report LA-UR 96-60). Los Alamos, NM: Los Alamos National Laboratory.

Miller, B. L., & Goldberg, D. E. (1996). Optimal sampling for genetic algorithms. *Proceedings of the Artificial Neural Networks in Engineering (ANNIE '96) Conference,* 291–297.

Munetomo, M., & Goldberg, D. E. (in press). Identifying linkage groups by nonlinearity/non-monotonicity detection. *GECCO-99: Proceedings of the 1999 Genetic and Evolutionary Computation Conference.*

Pelikan, M., Goldberg, D. E., Cantú-Paz (in press). BOA: The Bayesian optimization algorithm. *GECCO-99: Proceedings of the 1999 Genetic and Evolutionary Computation Conference.*

Thierens, D., & Goldberg, D. E. (1993). Mixing in genetic algorithms. *Proceedings of the Fifth International Conference on Genetic Algorithms,* 38-45.