*GECCO–99: Proceedings of the Genetic and Evolutionary Computation Conference*, in press.

# A DNA Implementation of the Max 1s Problem

**David Wood**
Computer Sci.
Univ. of Del.
Newark DE

**Junghuei Chen**
Chem. & Biochem.
Univ. of Del.
Newark DE

**Eugene Antipov**
Chemical Eng.
Univ. of Del.
Newark DE

**Bertrand Lemieux**
Plant & Soil Sci.
Univ. of Del.
Newark DE

**Walter Cedeño**
Hewlett-Packard
Centerville Rd
Wilmington DE

## Abstract

Elements of genetic algorithms, DNA computing, and *in vitro* evolution are combined into laboratory procedures. The traditional test problem for genetic algorithms, Max 1s problem is addressed. Preliminary laboratory results are shown.

## 1 Introduction

Evolution is a concept of obtaining adaptation through the interplay of selection and diversity. Analogies from evolution have been used in both computing and molecular biology. The paradigm in molecular biology is known as "*in vitro* evolution."

In this paper we identify elements of genetic algorithms and *in vitro* evolution that we recommend combining to address Max 1s problems. We choose Max 1s to test our techniques, which can also be used on some other problems [6]. We choose genetic algorithms because they manipulate bitstrings using operations of pointwise mutation and crossover. These operations can be performed by modifications of techniques from *in vitro* evolution. The crucial operation of physically separating DNA strands by their "fitness" is demonstrated using 2d denaturing gradient gel electrophoresis.

We are careful to make the following point. Evolutionary computation has controversial aspects. Evolutionary computation makes few assumptions and is ostensibly applicable to very broad classes of problems. Naturally, this makes it difficult to establish any provable guidelines. We do not take any stance on the virtues of any particular method of evolutionary computation. Instead, we aspire to *provide the means for assessing* some evolutionary computations using population sizes larger than is practical with conventional computers.

## 2 DNA Suggests Genetic Algorithms

Several means of DNA computation have been addressed. The first was, of course, by Adleman [1, 2]. Recent overviews can be found in [12] and [18]. See also the DNA computing bibliography of Dassen [8].

From the beginning of DNA based computing to the present there have been calls [10, 19, 24] to consider carrying out evolutionary computations using genetic materials in the laboratory. So far, there have been three such experiment designs proposed, including two in a recent DIMACS Workshop [3, 6]. The very first design was presented about two years ago [9], but has not yet been carried out in the laboratory.

Naturally, computing time using DNA is proportional to the number of generations required. This motivates incorporating both pointwise mutation and crossover, attempting to minimize the number of generations required. Of all computing paradigms inspired by evolution, genetic algorithms seem particularly suited to implementation using DNA. This is because genetic algorithms generally use bitstrings, crossover, and pointwise mutation.

### 2.1 DNA Attributes Suit Genetic Algorithms

DNA computing techniques are desirable for genetic algorithm computations for several reasons, some of which are listed below.

- These techniques might process, in parallel, populations which are billions of times larger than is usual for conventional computers. The expectation for larger populations is: they can sustain larger ranges of genetic variation and thus can generate high-fitness individuals in fewer generations.

- Massive information storage is available using DNA. For example, grams of DNA could eventually be used. A gram of DNA contains about $10^{21}$ bases. This information content is approximately $2 \times 10^{21}$ bits, greatly exceeding the 200 petabyte storage of all the digital magnetic tape produced in one year [30].
- Biolaboratory operations on DNA inherently involve errors. These are more tolerable in executing genetic algorithms than in executing deterministic algorithms. To some extent, errors may be regarded as contributing to desirable genetic diversity.
- Modifications to the current technology of *in vitro* evolution suffice to implement crossover and pointwise mutation.

However, selecting DNA strands for "breeding" in genetic algorithms is moderately challenging because one must physically separate DNA strands according to their "fitness."

### 2.2 DNA Genetic Algorithms Compared to Supercomputers

The following oversimplified estimates indicate DNA computing techniques could in the future compare favorably to supercomputers in some cases. These favorable cases include executing genetic algorithms having simple fitness evaluations and very large populations of candidate solutions.

Consider a population represented by a total of $p$ bits. Executing a genetic algorithm *by any means* will require at least $g \times \mathcal{O}(p)$ fitness evaluations, where $g$ is the number of generations used. Now, assume the fitness evaluation of a candidate solution processes all the bits of the candidate solution. A state-of-the-art teraflop supercomputer performs about $10^{12}$ operations per second. Thus, we have a rough estimate for supercomputer time complexity,

$$T \approx g \times p \times 10^{-12} \text{ seconds} \approx g \times p \times 10^{-17} \text{ days.} \quad (1)$$

To compare this to DNA computing, let us assume the fitness evaluation of an entire population can be done in the laboratory in 24 hours [6]. Thus, the time complexity of a DNA approach to genetic algorithms is seen to be about

$$T \approx g \text{ days.} \quad (2)$$

Essentially no new laboratory techniques or equipment would be needed to use gram quantities of DNA. This corresponds to populations represented by about $p = 10^{21}$ bits. For populations of this size, we see from Eq 1 and from Eq 2 that DNA implementation of genetic algorithms compares favorably to supercomputer time complexity.

Some caution is needed in interpreting this comparison. The comparison is based on unprecedentedly large populations. (Still miniscule compared to the size of the search space, of course!) As far as we know, it is unclear exactly how beneficial large population sizes might be. The classical "schema theorem" of Holland [14] says a population of $P$ distinct candidates probes $\mathcal{O}(P^3)$ potential solutions. However the applicability of this result, like many others in evolutionary computation, is actively debated.

This may be an appropriate point to repeat our primary goal. We do not take any stance on the virtues of any particular method of evolutionary computation. Instead, we aspire to provide the means for assessing some evolutionary computations using population sizes larger than is practical with conventional computers.

### 2.3 DNA Genetic Algorithms Compared to *In Vitro* Evolution

Methods in molecular biology known as "*in vitro* evolution" are similar to those used in genetic algorithms.

Virtually all DNA sequences can be equally meaningful as inputs or outputs of genetic algorithms. In fact, in a DNA context, Max 1s problems with *various* targets (not just all 1s) are of interest: "Evolve a population of DNA strands highly similar to a given one."

In contrast, *in vitro* evolution focuses on variations of the *extremely rare* DNA sequences of biological or biochemical interest. Naturally, *in vitro* evolution methods use fitness criteria limited to properties of biomolecular interest. Indeed, finding means to physically separate biological materials by "fitness" has determined which problems are addressed by *in vitro* evolution.

The comparison can be summarized like this. The fitness and selection techniques of *in vitro* evolution are limited. They produce a relatively few, but important, possible outcomes. DNA implementation of Max 1s problems allow essentially arbitrary bitstrings as inputs (targets) and outputs.

*In vitro* methods suit the following very important but relatively small classes of problems.

- Seeking improved enzymes [23], which are proteins that serve as catalysts in biochemical reactions. This work emphasizes crossover more than mutation.

- Seeking ribozymes [13, 20, 26, 28], which are RNA strands that serve as biochemical catalysts. This work uses mutation but not crossover.

- Seeking binding sites [26, 27], which are sequences of DNA that are acted upon by small biomolecules. This work uses repeated selection but neither mutation nor crossover.

## 3 The Max 1s Problem

This is a traditional test problem for genetic algorithms. It involves binary bitstrings of fixed length. An initial population (usually randomly generated) is given. The objective is to evolve some bitstrings to match a prespecified "target" (generally taken to be all 1s). However, arbitrary targets may be of interest in working with DNA.

### 3.1 The Max 1s Problem via DNA

This section begins with an outline of a DNA implementation. The corresponding information is shown in Figure 1. In the remainder of the section, details and preliminary laboratory results are given. Throughout this section, information is grouped in the following categories: (1) candidate pool, (2) fitness evaluation, (3) selection and (4) breeding.

### 3.2 Outline of DNA Implementation

Our implementation is given by the following outline. The same information, with a few added details, is shown in Figure 1.

**DNA Genetic Algorithm for Max 1s**

Begin with an initial population of candidates.

1. Evaluate fitness by hybridizing to target strands and physically separate on a 2d gel.
2. Select and purify candidates to breed.
3. Amplify selected candidates incorporating pointwise mutation and reserve a portion.
4. Breed candidates, using crossover.
5. Combine reserved and bred candidates, obtaining a new generation.

Repeat.

### 3.3 A New Fitness Selection is Added to *In Vitro* Evolution, and Combined With Existing Breeding Techniques

As mentioned earlier, modifications to current technology suffice to implement crossover and pointwise mutation. However, selecting DNA strands for "breeding" in genetic algorithms is more challenging because one must physically separate DNA strands according to their "fitness." Thus, we combine modified forms of three robust biolaboratory techniques for implementing DNA genetic algorithms [6].

- A new application of 2d denaturing gradient gel electrophoresis (2d DGGE): the more fit candidate strands of DNA can be physically separated from the less fit candidates according to how well they match (hybridize with) "target" DNA strands.

- Existing "dirty" polymerase chain reaction (dPCR) [13, 20, 26, 27, 28] incorporates pseudorandom mutation.

- An existing *in vitro* crossover technique due to Stemmer [22, 23, 25] shuffles DNA by blocks. We modify this to give single-point crossover to agree with a common choice for genetic algorithms.

### 3.4 Design of Candidate Solutions and Target DNA Strands

Figure 2 shows our DNA strand design. A target

```
      /-----------------TARGET------------------\

      /-----CG-CLAMP--------\/----- 80 A's------\

5' -> CGCCCTCCGCCCGTCGCCCGCCCAAAAAA.......AAAAAAA -> 3'

3' <- GCGGGAGGCGGGCAGCGGGCGGGTTTTTT.......TTTTTTTGTGATCACTCAGCATAAT <- 5'

      \--CG-CLAMP COMPL-----/\----- 80 T's------/\----- TAIL -----/

      \---------------PERFECT CANDIDATE---------------------------/
```

Figure 2: Design of target and a perfect candidate. Imperfect candidates would have a mixture of 80 Ts and Cs in place of the 80 Ts in the perfect candidate.

strand and a perfect candidate strand are shown in the figure. Imperfect candidate strands have a mixture of 80 Ts and Cs instead of 80 consecutive Ts. At the $3'$ end of all candidate strands there is a universal section complementary to the CG clamp section of the target strands. The CG clamp has been designed to encourage correct alignment and to avoid secondary structure (hybridizing to itself). All $5'$ ends of the

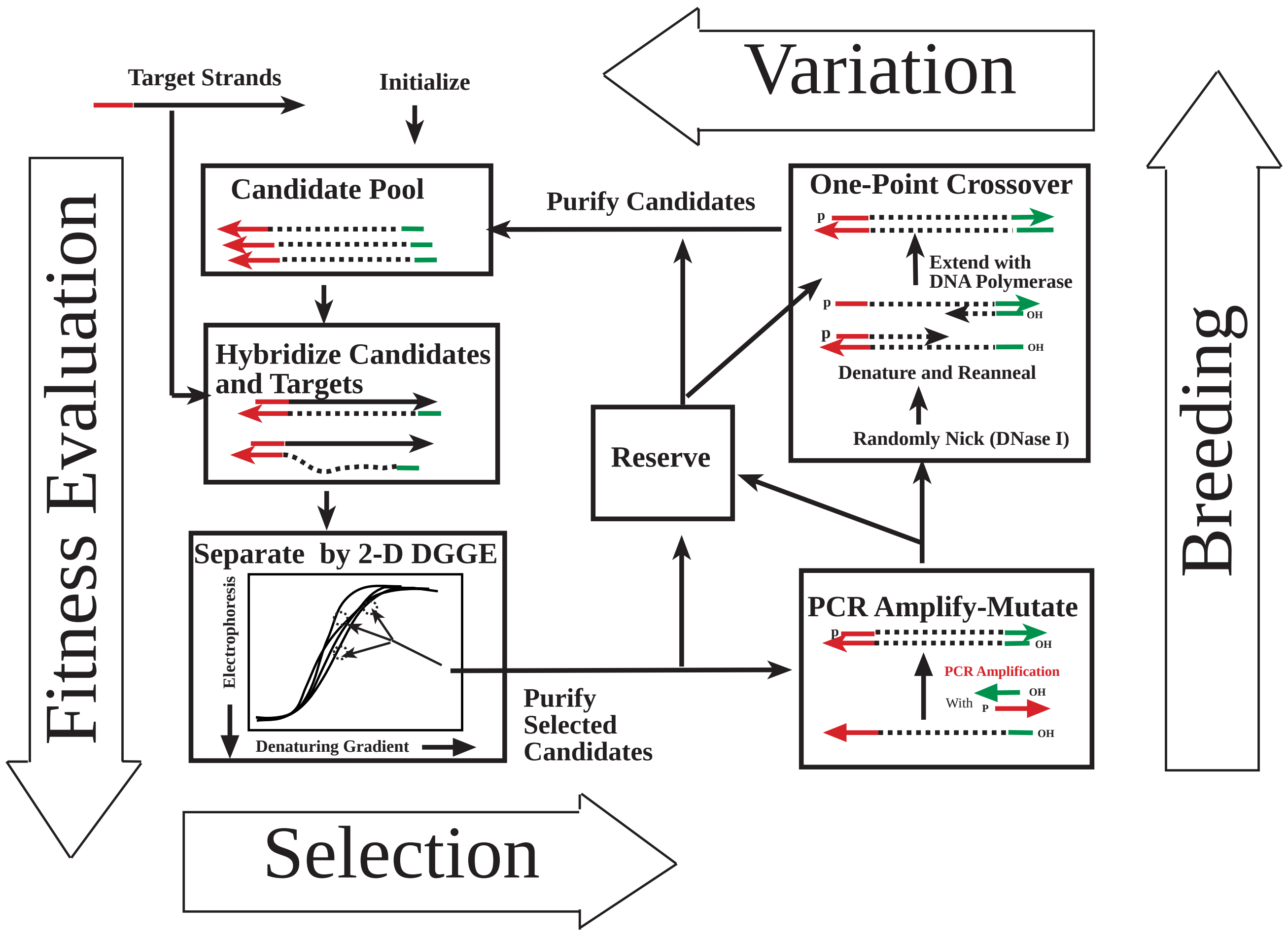

Figure 1: Outline of DNA implementation of genetic algorithms for Max 1s problem. The candidate pool appears in the upper left. Selection using 2d DGGE appears at the lower left. Purification and amplification of the more fit candidate strands appears at the lower right. Breeding using crossover appears at the upper right.

candidate strands are extended by a universal tail sequence. Since candidate strands have known sequences (used as primer sites) at both ends, they can be amplified by PCR. The candidate strands are longer to facilitate eventual separation of target and candidate strands using denaturing gel electrophoresis.

### 3.5 Fitness Evaluation by 2d DGGE Physical Separation of DNA

The most challenging part of the DNA implementation of genetic algorithms is to identify a laboratory process that will physically separate DNA strands according to their "fitness." For this task we use so-called 2d denaturing gradient gel electrophoresis (2d DGGE), which we push far beyond its established domain of application [15].

- A first important fact for DNA computing is that *2d DGGE can detect even a single base mismatch* in DNA strands. Indeed, this is a common application of 2d DGGE in molecular biology [15].

- A second important fact is that our experiments with 2d DGGE (see Figure 4) demonstrate a surprisingly *smooth transition through a large dynamic range of mismatching.*

To the best of our knowledge, no one has ever before demonstrated what happens when 2d DGGE is used with a population having a very great diversity of mismatchings. (The question has not arisen in molecular biology applications.) Populations of strands all having considerable mismatches might regrettably not be distinguishable at all. Fortunately, we found otherwise.

Let us review the nature of DGGE. Figure 3 shows a control case: a 2d DGGE from our laboratory having perfect candidates combined with target strands. The target strands hybridize (stick) to the perfect candidate strands, with a tail of unmatched bases at the $5'$ end. The mixture of hybridized strands is placed uniformly along the top of the gel. The hybridized strands travel vertically downward in the gel as a result of an applied electric field. However, their speed of migration is determined by their initial placement from left to right; that is, by how strongly they are denatured (pulled apart). On the left, where no denaturant is encountered, the strands move relatively quickly downward. In the center, they move more slowly because they encounter intermediate denaturing. At the extreme right, the stands are able to move only very slowly because the strands are almost completely pulled apart except in the more resistant CG clamp region.

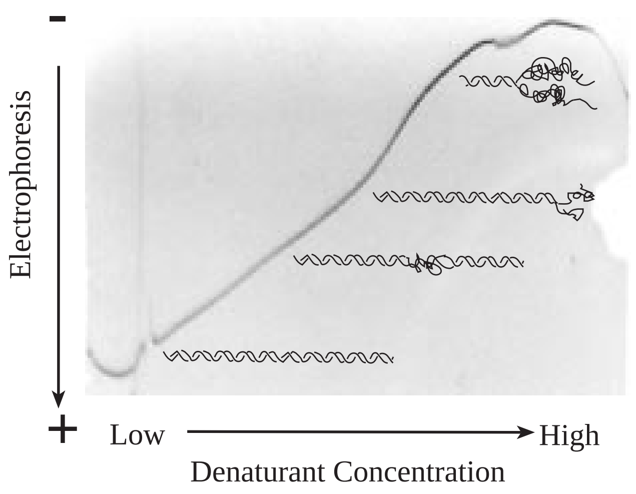

Figure 3: DGGE using perfect candidates. DNA strands move downward from a reservoir at the top of the figure. The speed of vertical strand migration is retarded as strands come apart (denature) as shown schematically on the figure.

We heuristically reason that when we repeat the above experiment with a mixture of targets and imperfect candidates, we expect that everywhere across the gel the candidate strands that best match (hybridize to) the targets will migrate downward relatively faster. In fact, a mixture of imperfect matches exhibits vertical spreading in our experiments. See Figure 4. In this figure the 80 variable positions of the imperfect candidates are chosen to be Ts with .8 probability and Cs with .2 probability.

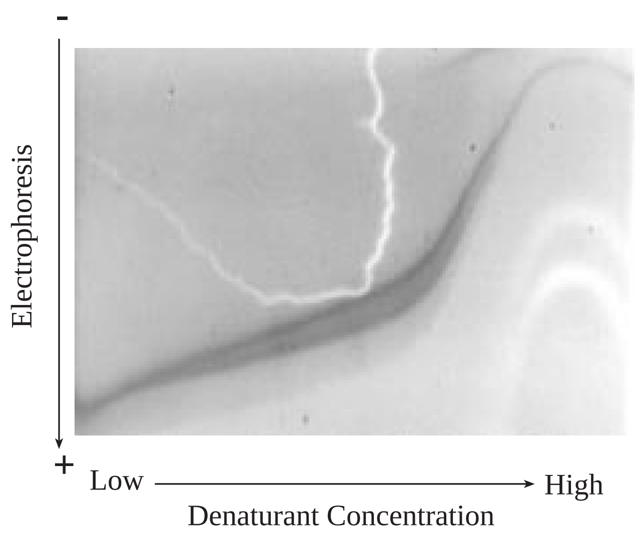

Figure 4: DGGE vertically separates imperfect candidates. The speed of vertical strand migration is retarded as strands come apart (denature). This is due to two factors: increasing denaturant concentration and decreasing quality of target-candidate matching (hybridization).

### 3.6 Fitness Selection of Candidate Solutions

Selection is done by literally cutting out a portion of the 2d gel and extracting the DNA strands from it. This allows a wide latitude for selection criteria. The most fit candidates are presumably lowest on any vertical line. However, the nature of variation from left to right is not clear. Further experiments will be needed to optimize a selection strategy. Experience in genetic algorithm computing demonstrates the desirability of maintaining genetic diversity to prevent the loss of genetic materials which may be needed in later stages of evolution.

The selected DNA is purified (for example, separated by length using conventional denaturing gel electrophoresis) to get rid of target strands. The purified candidate strands are amplified by PCR, which can also induce pointwise mutation at a rate of $10^{-2}$ to $10^{-4}$ [4, 5, 7, 16, 17, 29, 31]. One of the PCR primers, the one that makes strands complementary to the candidates, is phosphorylated on its $5'$ end so that these strands can later be digested with $\lambda$-exonuclease.

A portion of the resulting double stranded product is temporarily reserved; the remainder is used for breeding.

### 3.7 Breeding Using Single Point Crossover

The portion of double stranded product to be used for breeding is partially digested with the enzyme *DNase I* to nick (cut only one strand) at random locations. The nicked strands are combined with a similar amount of reserved unnicked strands. The mixture is denatured (strands are melted apart) and allowed to reanneal forming new combinations. Many, many possible configurations could be formed. But among these, some will be intact complements of candidate strands annealed to a $5'$ end of a candidate strand including its CG clamp, which enforces alignment. These are featured in the upper right corner of Figure 1. By adding DNA polymerase, the partial candidate strand is extended to a full length legal candidate combining its genetic information with that encoded in the intact strand. The net result is single point crossover. The offspring candidate strand has a block of genetic information from one parent followed by another block from a different parent.

The Sexual PCR (gene shuffling) technique of Stemmer [25, 22] is similar to our crossover operation. Sexual PCR would be limited to populations of candidate DNA strands which are very similar and nonuniformly structured. These two properties are needed to ensure alignment of the DNA fragments. We avoid these restrictions on the variety of candidate strands by adding the universal CG sequences at the ends of the candidate strands to enforce alignment. However, our present approach limits us to using single point crossover (which is usually used in genetic algorithms).

Finally, the reaction products are combined with the remainder of the reserved material and complementary strands are digested with $\lambda$-exonuclease. Our crossover reactions may produce many products besides offspring candidates, but they will be benign and rarely the same length as a candidate. Purification by length (using denaturing gel electrophoresis) completes the breeding operation.

The new generation is ready to be processed.

## Acknowledgment

We want to acknowledge partial support under DARPA/NSF Grant No. 9725021 for Chen and Wood. Antipov was partially supported under an NSF Research Experience for Undergraduates supplement to this grant.

## References

[1] Leonard M. Adleman. Molecular computation of solutions to combinatorial problems. *Science*, 266:1021–1024, November 1994.

[2] Leonard M. Adleman. Computing with DNA. *Scientific American*, 279(2):54–61, August 1998.

[3] Thomas Bäck, Joost N. Kok, and Grzegorz Rozenberg. Evolutionary computation as a paradigm for DNA-based computing. In Laura Landweber, Erik Winfree, Richard Lipton, and Stephan Freeland, editors, *Preliminary Proceedings DIMACS Workshop on Evolution as Computation*, pages 67–88, DIMACS, Piscataway NJ, January 1999. Available on request from DIMACS. Paper found at URL: http://www.wi.LeidenUniv.nl/˜joost.

[4] R. Craig Cadwell and Gerald F. Joyce. Randomization of genes by PCR mutagenesis. *PCR Methods and Applications*, 2:28–33, 1992.

[5] R. Craig Cadwell and Gerald F. Joyce. Mutagenic PCR. In Carl W. Dieffenbach and Gabriela S. Dveksler, editors, *RCP Primer: A Laboratory Manual*. Cold Spring Harbor Laboratory Press, Plainview, New York, 1995.

[6] Junghuei Chen, Eugene Antipov, Bertrand Lemieux, Walter Cedeño, and David Harlan Wood. DNA computing implementing genetic algorithms. In Laura Landweber, Erik Winfree, Richard Lipton, and Stephan Freeland, editors, *Preliminary Proceedings DIMACS Workshop on Evolution as Computation*, pages 39–49, DIMACS, Piscataway NJ, January 1999. Available on request from DIMACS. Paper found at URL: www.cis.udel.edu/˜wood/papers/DIMACS_99.ps.

[7] R. Cheynier, S. Gratton, J. P. Vartanian, A. Meyerhans, and S. WainHobson. G$\rightarrow$A hypermutation does not result from polymerase chain reaction. *AIDS Research and Human Retroviruses*, 13(12):985–986, August 10, 1997.

[8] J. H. M. Dassen. A bibliography of molecular computation and splicing systems. HTML source: http://www.wi.LeidenUniv.nl/jdassen/dna.html, http://www.wi.LeidenUniv.nl/ jdassen/dna.bib. This bibliography is also hooked into http://liinwww.ira.uka.de/bibliography/, The Collection of Computer Science Bibliographies.

[9] R. Deaton, R. C. Murphy, J. A. Rose, Max H. Garzon, Donald R. Franceschetti, and

S. E. Stevens Jr. A DNA based implementation of an evolutionary search for good encodings for DNA computation. In *IEEE International Conference on Evolutionary Computation*, pages 267–271, Indianapolis, Illinois, April 13–16, 1997.

[10] Alan Dove. From bits to bases: Computing with DNA. *Nature Biotechnology*, 16(9):830–832, September 1998.

[11] A. D. Ellington, M. P. Robertson, and J. Bull. In vitro evolution - Ribozymes in wonderland. *Science*, 276(5312):546–547, April 25, 1997.

[12] Tino Gramß, Stephan Bornholdt, Michael Gramß, Melanie Mitchell, and Thomas Pellizzari. *Non-Standard Computation.* Wiley-VCH, Weinheim, 1998.

[13] Rachel Green, Andrew D. Ellington, David P. Bartel, and Jack W. Szostak. *In vitro* genetic analysis: Selection and amplification of rare functional nucleic acids. *Methods*, 2:75–86, 1991.

[14] John H. Holland. *Adaptation in natural and artificial systems.* MIT Press, Boston, 1992.

[15] L. S. Lerman, K. Silverstein, and E. Grinfeld. Searching for gene defects by denaturing gradient gel electrophoresis. *Trends in Biochemical Sciences*, 172(3):89–93, 1992.

[16] M. M. Ling and B. H. Robinson. Approaches to DNA mutagenesis: An overview. *Analytical Biochemistry*, 254(2):157–178, December 15, 1997.

[17] J. L. LinGoerke, D. J. Robbins, and J. D. Burczak. PCR-based random mutagenesis using manganese and reduced dNTP concentration. *Biotechniques*, 23(3):407–, September 1997.

[18] Carlo C. Maley. DNA computation: Theory, practice, and prospects. *Evolutionary Computation*, 6(3):201–229, 1998.

[19] Robert Pool. Forget silicon, try DNA. *New Scientist*, 151(2038):26–31, July 13, 1996.

[20] D. L. Robertson and F. G. Joyce. Selection *in vitro* of an RNA enzyme that specifically cleaves single-stranded DNA. *Nature*, 344(6265):467–468, March 29, 1990.

[21] A. Skandalis, L. P. Encell, and L. A. Loeb. Creating novel enzymes by applied molecular evolution. *Chemistry and Biology*, 4(12):889–898, December 1997.

[22] Willem P. C. Stemmer. DNA shuffling by random fragmentation and reassembly: *In vitro* recombination for molecular evolution. *Proceedings of the National Academy of Science, U.S.A.*, 91:389–391, 1994.

[23] Willem P. C. Stemmer. Rapid evolution of a protein by DNA shuffling. *Nature*, 370:389–391, 1994.

[24] Willem P. C. Stemmer. The evolution of molecular computation. *Science*, 270:1510–1510, December 1, 1995.

[25] Willem P. C. Stemmer. Sexual PCR and assembly PCR. In Robert M. Meyers, editor, *The Encyclopedia of Molecular Biology and Molecular Medicine*, volume 5, pages 447–457. VCH, New York, 1996.

[26] Jack W. Szostak. *In vitro* genetics. *Trends in Biochemical Sciences*, 172(3):89–93, 1992.

[27] H. J. Thiesen and C. Bach. Target detection assay (TDA) — A versatile procedure to determine DNA-binding sites as demonstrated on SP1 protein. *Nucleic Acids Research*, 18(11):3203–3209, 1990.

[28] C. Tuerk and L. Gold. Systematic evolution of ligands by exponential enrichment — RNA ligands to bacteriophage-T4 DNA-polymerase. *Science*, 249(4968):505–510, August 3, 1990.

[29] J. P. Vartanian, M. Henry, and S. WainHobson. Hypermutagenic PCR involving all four transitions and a sizeable proportion of transversions. *Nucleic Acids Research*, 24(14):2627–2631, July 15, 1996.

[30] Roy Williams. Data powers of ten. Web page at http://www.ccsf.caltech.edu/˜roy/dataquan.

[31] M. Zaccolo, D. M. Williams, D. M. Brown, and E. Gherardi. An approach to random mutagenesis of DNA using mixtures of triphosphate derivatives of nucleoside analogues. *Journal of Molecular Biology*, 255(4):589–603, February 2, 1996.