New Approaches to Designing Genes by Evolution in the Computer

Alexander V. Spirov and David M. Holloway
[1]*The I. M. Sechenov Institute of Evolutionary Physiology and Biochemistry,*
[2]*Stony Brook University,*
[3]*British Columbia Institute of Technology,*
[1]*USA*
[2]*Russia*
[3]*Canada*

1. Introduction

The field of Evolutionary Computation (EC) has been inspired by ideas from the classical theory of biological evolution, with, in particular, the components of a population from which reproductive parents are chosen, a reproductive protocol, a method for altering the genetic information of offspring, and a means for testing the fitness of offspring in order to include them in the population. In turn, impressive progress in EC – understanding the reasons for efficiencies in evolutionary searches - has begun to influence scientific work in the field of molecular evolution and in the modeling of biological evolution (Stemmer, 1994a,b; van Nimwegen et al. 1997; 1999; Crutchfield & van Nimwegen, 2001). In this chapter, we will discuss how developments in EC, particularly in the area of crossover operators for Genetic Algorithms (GA), provide new understanding of evolutionary search efficiencies, and the impacts this can have for biological molecular evolution, including directed evolution in the test tube.

GA approaches have five particular elements: encoding (the ‘chromosome’); a population; a method for selecting parents and making a child chromosome from the parents’ chromosomes; a method for altering the child’s chromosomes (mutation and crossover/recombination); criteria for fitness; and rules, based on fitness, by which offspring are included into the population (and parents retained). We will discuss our work and others’ on each of these aspects, but our focus is on the substantial efficiencies that can be found in the alteration of the child chromosome step. For this, we take inspiration from real biological reproduction mechanisms.

1.1 Biological evolution by random point mutations?

Traditional GA, using random point mutations, indicates that such a mechanism would be too slow to account for the observed speed of biological evolution (e.g. Shapiro, 2010). This suggests that other more complicated mutational mechanisms are acting (Shapiro, 1999, 2002; 2010). A number of projects are indicating, indeed, that the design of biological molecular machines, such as gene regulatory circuits, may be unreachable by an evolutionary search from scratch (von Dassow et al., 2000; Kitano, 2004; Shapiro, 2010). A likely solution is that evolution creates complicated molecular machines by operating on previously-evolved simpler domains (motifs, modules) (e.g. Botstein, 1980).

1.2 Building blocks in protein and nucleic acid molecules

In parallel with the computational literature, we use the term 'building blocks' (BBs) for these simpler domains. Biologically, BBs are found in proteins, in which amino acids combine to create functionally and physically distinct regions within the protein (e.g. Voigt et al., 2002); they are found in the semi-autonomous domains of RNA (Ancel-Myers & Fontana, 2005); and they are found in DNA, from nucleosomal and chromatin organization to the organization of gene regulatory regions (Fig. 1). Comparative studies show that BBs are maintained during evolution, and can be shared by quite diverse organisms (Voigt et al., 2002).

The striking conservation of BBs in biological evolution has been noted in GA. It is beginning to be understood how important conservation of BBs is for efficient evolutionary searches in GA (and other fields of EC) (Forrest & Mitchell, 1993a,b; Goldberg, 1989; Holland, 1975; Mitchell et al., 1992). This chapter will discuss recent developments of GA chromosome alteration rules which conserve BBs, and how these relate to developments in directed evolution in the laboratory.

1.3 Nontrivial mutagenesis for molecular evolution in the test tube

As well as increased understanding of the role of BBs in biology and in search mechanisms, there is a growing appreciation for the use of BBs in in vitro, directed evolution experiments. Numerous groups are using evolutionary principles to design and select macromolecules, and it is becoming apparent that random point mutations are not the most efficient means for doing this. The role of crossover in conserving BBs in GA has inspired new techniques in molecular evolution in the test tube (Stemmer, 1994a,b). Methods are now being used to recombine from specific crossover sites to maintain BBs (Fig. 2) and speed the generation of diverse usable progeny molecules.

DNA shuffling (or 'sexual PCR') and in vitro evolution are well advanced fields now, and have been successfully used to design many new biotechnologically valuable enzymes (Sen et al., 2007). Beyond the synthesis of macromolecules, a growing area in systems biology is to investigate the evolution of genes and gene networks, through computation and synthetic biology laboratory work.

1.4 Biological evolution requires complicated mutational mechanisms?

The role of complex methods of mutation vs. simple point mutation is currently an active area of discussion (e.g. Long et al., 2003). In particular, agents such as retroviruses (e.g. HIV) and retroposons are believed to work as highly effective and highly specific mutators (e.g. Brosius, 1999). The crossover mechanism evolved by retroviruses (Fig. 3) shares many similarities with the DNA shuffling/sexual PCR techniques used in in vitro evolution (Fig. 2).

A)…ACGTAATCCCcatagaaaaccggtggaaaattcgcagctcgc**TGCTAAGCT**ggccatc**CGCTAAGCT**
cccg**GATCATCCA**aatccaagtgcgcataatttttgtttctg**CTCTAATCCA**gaatggatcaagagcgcaatcctcaatcc
gcgatccgtgatcctcgattcccgaccgatccgcgacctgtacctgacttcccgtcacctctgccca**TCTAATCCC**…

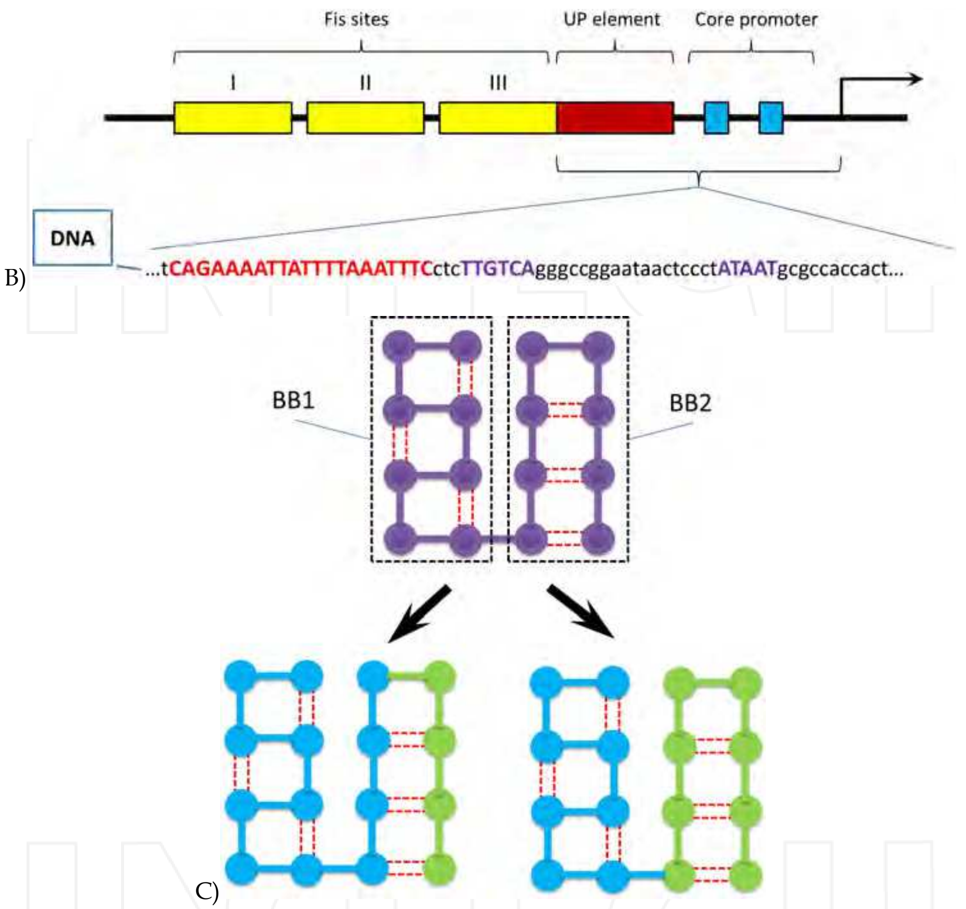

Fis sites
UP element
Core promoter
I
II
III
DNA
B)
…tCAGAAAATTATTTTAAATTTCctcTTGTCAgggccggaataactccctATAATgcgccaccact…
BB1
BB2
C)

Fig. 1. Some examples of "building blocks" (BBs) in biological macromolecules engaged in keeping and transferring genetic information – polypeptides and nucleic acids. A) The core of the anterior regulatory region of the *Drosophila* (fruit fly) gene *hunchback*, with a cluster of six BBs highly specific for recognition of the Bicoid protein. B) Organization of the bacterial promoter rrnP1 (ribosomal RNA operon promoter) into a series of highly conserved blocks, with between-block spacers of conserved length. C) Illustration of BB disruption in proteins. Black lines represent peptide bonds, red dotted lines represent interactions between amino acid (aa) side chains. Two hybrid proteins are shown. If the first 12 residues (aa's) are from one parent, and the last four residues are from the other parent, three side chain interactions can be disrupted. If the last eight residues come from the same parent, then there is no disruption. Hybridizations that maintained interactions would be most likely to fold properly. (After Voigt et al., 2002)

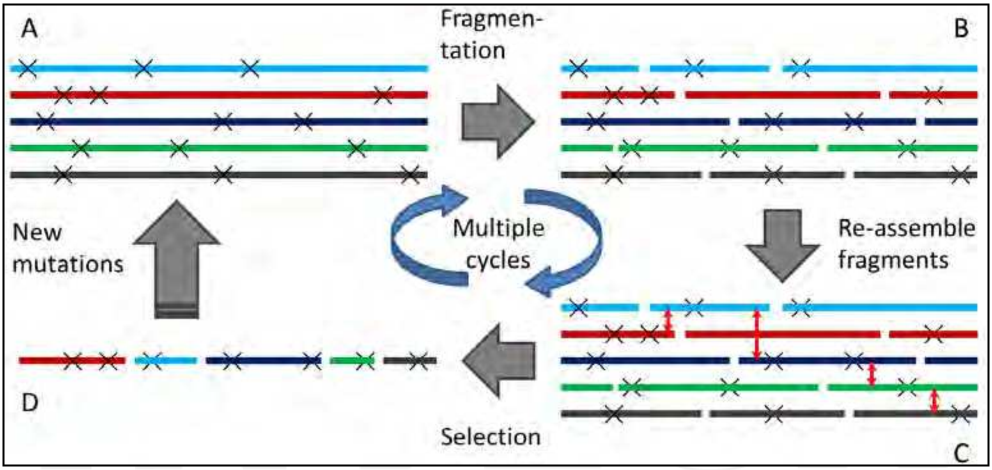


Fig. 2. Schematic of experimental mutation of DNA for in vitro evolution. Input strands can only be cut at specific sites; progeny are created by combining these fragments. BBs are maintained within the fragments, but novel combinations are created in the process. Keeping the BB sequences minimizes structural disruption in the products.

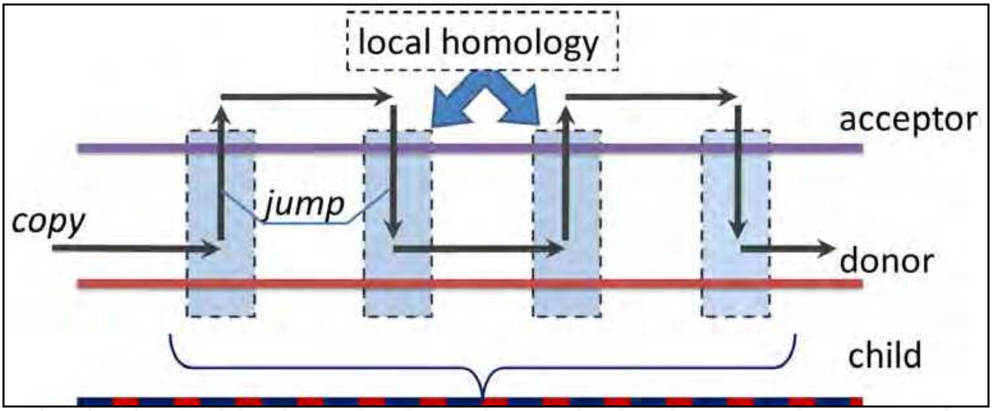


Fig. 3. The overall idea of recombination between two parental RNA strings used by retroviruses. The child sequence is read alternately from the parent strings, jumping between the parent templates at regions of homology (marked by gray rectangles). This is an effective mechanism for genetic diversity in the child, while retaining BBs.

Retroviral recombination usually takes two parent RNA strands to create a child DNA strand (Negroni & Buc, 2001; An & Telesnitsky, 2002; though a three strand mechanism is also a possibility). Development of GA crossover operators that use the retroviral scheme, or extend it to the multiple parent case seen in sexual PCR, can give quantitative understanding of the efficiencies of these techniques, and can provide insight into the biological evolution of retroviruses and retroposons.

1.5 Test tube evolution needs new theoretical considerations

Some in vitro evolution groups are using DNA 'soft computing' on well-defined benchmark computing problems (Chen & Wood, 2000; Henkel et al., 2007). However, theoretical (mathematical) studies of these computations are still at an initial stage (Crutchfield, 2002; Sun, 1999; Maheshri & Schaffer, 2003). A central question is whether theory can offer new approaches to speed up the evolutionary search for macromolecules with desired characteristics or features. This chapter will survey prospective ways to apply new developments in GA crossover operations in order to further the theory and efficiency of in vitro evolution.

We have long been interested in the design of genes with multiple autonomous regulatory elements – these are critical in formation of the early body plan (in particular, we study these genes in the fruit fly, *Drosophila*). We have found that evolutionary searches for such highly structured sequences are very similar to the well-defined Royal Road (RR) and Royal Staircase (RS) computational test functions. By developing GA crossover operators that perform well on RR and RS functions, we are developing computational techniques for solving real design problems for biological and synthetic macromolecules. In particular, we are introducing GA crossover operators that work like retroviral or sexual PCR recombination, and which have the ability to preserve BB architecture. We name our approach Retroviral GA, or retroGA.

This chapter will first (section 2) introduce the RR and RS test functions, and introduce the retroGA technique. Section 3 will show the performance of this approach on the test functions, and on biological gene-structure problems, from bacteria and fruit flies (each with particular challenges as a search problem). retroGA results will be compared with standard GA (point mutation). In section 4 we will discuss the prospects for extending the analytical approach developed by van Nimwegen and co-workers for RR and RS functions to real biological genetic problems, such as the bacteria and fruit fly examples in section 3. We will conclude on the use of the retroGA approach in understanding real biological evolution problems and for aiding the efficiency of directed (forced) molecular evolution in the laboratory.

2. Our approach

We will first discuss the RR and RS benchmark functions, which allow for standardized testing and analysis of BB type evolutionary problems. We will then present our retroGA approach, using retroviral recombination methods (crossover) to preserve BBs during evolutionary searches.

2.1 GA benchmark functions as models of molecular biological evolution

Among the many benchmark tests in EC, the RR and RS fitness functions were specifically invented to study the preservation and destruction of BBs by crossover operators. As such, they can serve as models for many cases of natural and test tube evolution, in which searches proceed with BB preservation. Four RR functions, of increasing complexity, were invented and introduced by Forrest, Mitchell, and Holland to specifically test crossover operations in GA (Forrest & Mitchell, 1993a,b; Mitchell et al., 1992). The related RS functions were devised and introduced by van Nimwegen & Crutchfield (2000; 2001). These well-defined functions allow for analytical (mathematical) study of the evolutionary search behaviour and parameter dependence.

2.1.1 Royal road functions

Royal Road functions were devised to award fitness for the preservation of BBs, and thus to serve as models for natural evolution (van Nimwegen & Crutchfield, 2000). R1, the simplest function, calculates bit string fitness by the number of order-8 schema, or words, in the string. Order does not matter:

```
s₁ = 11111111********************************************************; c₁  =  8
s₂ = ********11111111************************************************; c₂  =  8
...............................................................................
s₇ = ************************************************11111111********; c₇  =  8
s₈ = ********************************************************11111111; c₈  =  8
sopt= 1111111111111111111111111111111111111111111111111111111111111111; copt= 64
```

where the * is a random bit (0 or 1). The fitness value R1 (for string x) is the sum of the coefficients c_s corresponding to each given schema of which x is an instance (c_s is equal to order). The fitness of an intermediate step (such as the combination of s_1 and s_8) is a linear combination of the fitness of the lower level components (e.g. the combination of s_1 and s_8 has fitness 16). The genotype space consists of all bit-strings of length 64 and contains 9 neutral subbasins of fitness 0, 8, 16, 24, 32, 40, 48, 56 and 64. There is only one sequence with fitness 64, 255 strings with fitness 56, 65534 strings with fitness 48, etc. Because fitness proceeds by the build up of words, the fitness landscape for RRs has a subbasin-portal architecture (Fig. 4), in which evolution tends to drift in neutral subbasins, with rare jumps to the next level via portals (creation or destruction of a word).

In searching for fitness functions that are easy for GA and difficult for non-evolutionary methods, a whole family of increasingly complex RR functions was devised (R1, R2, R3, R4; Mitchell, 1996). These showed that standard GA is superior to non-evolutionary techniques on harder problems, but also brought to light that standard GA has substantial weaknesses in the crossover operator. Due to their formal simplicity, theoretical analysis can be carried out on the RR functions to understand the parameters which promote efficiencies.

R2 is very similar to R1, but allows for higher fitness at certain intermediate steps. R3 allows for random-bit spacers of set length between BBs (which are not calculated in the fitness score). The optimal string for R3 is:

s_{opt}=11111111********11111111********11111111*******11111111********11111111********11111111********11111111*******11111111.

The most difficult RR function is R4, since one, two, or even four non-neighboring elementary (in our case 8-bit) BBs in the string gives the same exact score (Level 1). The score will only increase if words neighbour, e.g. a pair of juxtaposed 8-bit BBs creates a Level 2 16-bit BB. A Level 3 BB consists of 4 neighboring 8-bit BBs (32 bits in total). Level 4 BBs are 64-bit, composed of eight 8-bit elementary BBs. Level 5 BBs would consist of 16 8-bit BBs. Most current optimization techniques cannot effectively deal with the R4 fitness function.

From the viewpoint of molecular biology, R1 and especially R3 are reminiscent of a key element of the regulatory region of a gene: a cluster of binding sites (BS's), made of short BBs separated by spacer sequences. While the analogy is good, there are several differences to bear in mind. First, usually the positions and order of BS's in such clusters are less restricted (than in the comparable RR), but this depends on the particular gene in question (enhanceosomes vs. "billboards"; see Jeziorska et al., 2009). Second, any BS is not a unique sequence: it is usually a family of related sequences with varying strength (fitness), usually with a conserved core sequence (Stormo, 2000). Finally, proximity of BSs to each other is important for the action of activators and repressors. This is analogous to R4 (e.g. with sub-clustering represented by R4, Level 3), but the biological spacing is somewhat less restricted than in R4. Because of the general parallels between RR and biological structure, we expect RR analysis to shed some light on the evolution of gene regulatory regions, and to be useful as a theory for forced molecular evolution of bacterial and yeast gene promoters. (Where modified or completely artificial promoters can become new molecular tools for bio-sensing, etc.: Haseltine & Arnold, 2007; Lu et al., 2009.)

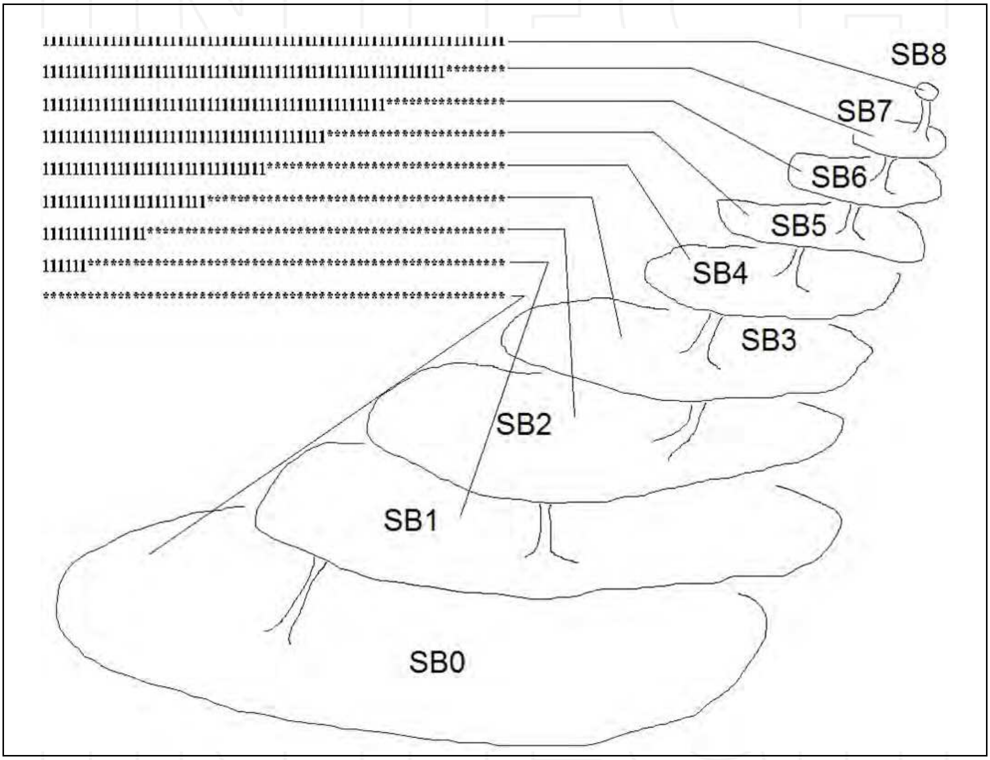


Fig. 4. Subbasin-portal architecture for the R1 function.

2.1.2 Royal staircase functions

These are a generalization of the RR functions in which the subbasin-portal architecture is expressed in a more explicit form (van Nimwegen & Crutchfield, 2000). The RS function we use in this chapter is similar to the 8-bit word, 64-bit string R1 and R2 of the previous section, but order matters (i.e. the string is built up from one end), and fitness for N=8, K=8 RS ranges from 0 to 80:

```
s1 = 11111111********************************************************; c1 = 10
s2 = 1111111111111111************************************************; c2 = 20
.....................................................................................
s6 = 111111111111111111111111111111111111111111111111****************; c6 = 60
s7 = 11111111111111111111111111111111111111111111111111111111********; c7 = 70
sopt= 1111111111111111111111111111111111111111111111111111111111111111; copt = 80
```

This version of RS was used by van Nimwegen and Crutchfield, and we use it in this chapter to be able to compare our results to theirs.

2.1.3 Evolution of gene regulatory regions

Here we develop two prototypical cases in the evolution of gene regulatory regions, highlighting their similarities to the RR and RS test functions. We will present retroGA results on these problems in section 3.

2.1.3.1 Directed (forced) evolution of prokaryotic promoters

Bacterial gene promoters, being simpler than eukaryotic promoters, present good cases for investigating the details of the evolutionary searches producing their structure. For a target sequence (solution) for these sorts of problems, we have selected the sequence of the ribosomal RNA (rRNA) operon promoter rrnP1 in *E. coli*, since it is very well studied and well characterized (Schneider et al., 2003). Core promoters in *E. coli* are approximately 60 base pairs (bp) long and are characterized by several conserved sites with spacers in between. It is believed that while the sequences of these spacers are not significant, their lengths are of extreme importance (Schneider et al., 2003). There are at least four well-conserved features in a bacterial promoter: the starting point (usually 'CAT'); the -10 sequence ('TATAAT' consensus); the -35 sequence ('TTGACA' consensus); and the distance between the -10 and -35 sequences. The rrnP1 promoter sequence contains an AT-rich sequence called the upstream (UP) element (Ross et al., 1998) upstream of the -35 element. UP elements increase transcription 20-to 50-fold (Hirvonen et al., 2001). Its consensus is AAA a/t a/t T a/t TTTT**AAAA, where * indicates a random base, a/t means A or T. In addition, three to five BS's for the Fis protein (FisBS) increase transcription three- to eight-fold (Ross et al., 1990). The weight matrix for the binding sites of this transcription factor has been defined (Hengen et al., 1997). The desired sequence for the rrnP1 promoter therefore includes:

```
[FisBS]**<~5 bp>**[FisBS]**<~5 bp>**[FisBS]**<~15 bp>**AAA a/t a/t T a/t TTTT**AAAA**<~4 bp>**TTGACA**<16-19 bp>**TATAAT**<5-9 bp>**CAT.
```

Evolution of the rrnP1 promoter can be viewed as a Royal Staircase fitness function. Starting with the core promoter (s_1), evolution could add the powerful UP element (s_2) and then sequentially add FisBSs (s_3, s_4):

```
s1=**********************************…*************************************************TTGACA***...***TATAAT***...***CAT, c1 = Δ
s2=****************…******************************AAA a/t a/t T a/t TTTT**AAAA***...***TTGACA***...***TATAAT***...***CAT, c2 = ~35 Δ
s3=*************…******************[FisBS]*** …***AAA a/t a/t T a/t TTTT**AAAA***...***TTGACA***...***TATAAT***...***CAT, c3 = ~100 Δ
.....................................................................................................................................
sopt=***[FisBS]***…***[FisBS]***…***[FisBS]*** …***AAA a/t a/t T a/t TTTT**AAAA***...***TTGACA***...***TATAAT***...***CAT, copt = ~150Δ
```

where Δ is an arbitrary small fitness value.

Like RS, rrnP1 probably evolved by sequential finding and adding of BBs, with each addition raising the fitness of the promoter sequence (transcriptional efficiency). As with RS, the length and positions of the BBs are conserved during evolution, though not as strictly as the RS function in section 2.1.2.

The simplified rrnP1 test

A major difference between RR and RS functions and functional clusters of BBs in biological macromolecules is the redundant character of the blocks. Functionally very similar blocks can have different sequences, sharing only a common core sequence. I.e. BBs usually are not unique, but are a family of related sequences. Also, compared to RR and RS, biological clusters of BBs include longer spacers (of variable length), and they are usually longer than 64 or 128 elements. Finally, they are not binary, but quaternary (DNA and RNA) (or even consisting of 20 letters, in the case of proteins).

To begin studying the rrnP1 problem (and do so within the RR, RS framework), we can simplify some of these complications: we ignore the redundant character of its 8 BBs and the variability of spacer lengths (see Fig. 1A,B); we assume that all the elements are fixed and/or unique in sequence; and we consider five elements only. The first of these represents the whole core promoter and is modelled by only 6 letters. The second element is the proximal half of the UP element, assumed to have a length of 5 letters. The spacer between the 1st and the 2nd elements is 24 letters. The 3rd to 5th elements are given the same length, with spacers of 15-letter length. We will present results on computing this simplified target in section 3.

2.1.3.2 Genes with multiple regulatory units

In eukaryotic organisms (i.e. non-bacterial), the organization of gene regulatory regions is far more complex. Genes are regulated from cis-regulatory modules (CRMs), which have clusters of BS's for activators and inhibitors, with very important spacer lengths between them to allow for quenching (inhibition) and cooperativity (activation). CRM's can be an arbitrary distance from the gene coding region. Compared to a prokaryotic model, like that for rrnP1, a eukaryotic CRM model must account for evolution of the BS locations and strengths, and be tested, fitness-wise, against a global production capacity. If the BS's are words in the language of gene regulation, CRM's order those words into sentences. Where rrnP1 could be treated as analogous to an RS problem with spacers, a eukaryotic CRM is more analogous to an R4 function, to account for clustering, with the level of the R4 representing the number of BS's in a functional cluster. Since the number of BS's is frequently 2 or 3, this begins to present major computational challenges, since most algorithms are insufficient at R4 Level 2 or 3. If we now begin to consider genes with multiple CRM's, which is common, we must consider at least R4 Level 4, a point at which most algorithms tend to fail. In analogy to language, organization of multiple CRM's is at the level of the paragraph directing a gene's regulation. Such problems may need to be more realistically thought of as higher level RS functions.

Evolution of multiple CRM's in Drosophila

The genes responsible for early body segmentation in the fruit fly, *Drosophila*, form a highly studied network of interacting regulations. These genes code for proteins which transcriptionally regulate the other segmentation genes, and their spatial expression patterns determine where different body parts will form. The regulatory regions for segmentation genes involve multiple CRM's, each of which can control different aspects of the spatial gene expression. It is believed the complex modern regulatory regions evolved by addition of CRM's to a simpler primitive antecedent. We are running computations on the evolution of a number of the segmentation genes.

One example is evolution of the regulatory region of the *hunchback* (*hb*) gene, which forms an anterior-high 'step function' pattern which differentiates the head from the tail end in the embryo. Fig. 5 shows the organization of the 3 *hb* CRM's, and the spatial expression that each is primarily responsible for. *hb* expression is controlled by at least 5 transcriptional regulators (protein products of other segmentation genes): Bicoid, Caudal, Tailless, Huckebein, Hunchback, Giant, Kruppel & Knirps. Available information on the organization of the *hb* regulatory regions is collected in the HOX pro (Spirov et al., 2000; 2002) database (http://www.iephb.nw.ru/hoxpro/hb-CRMs.html).

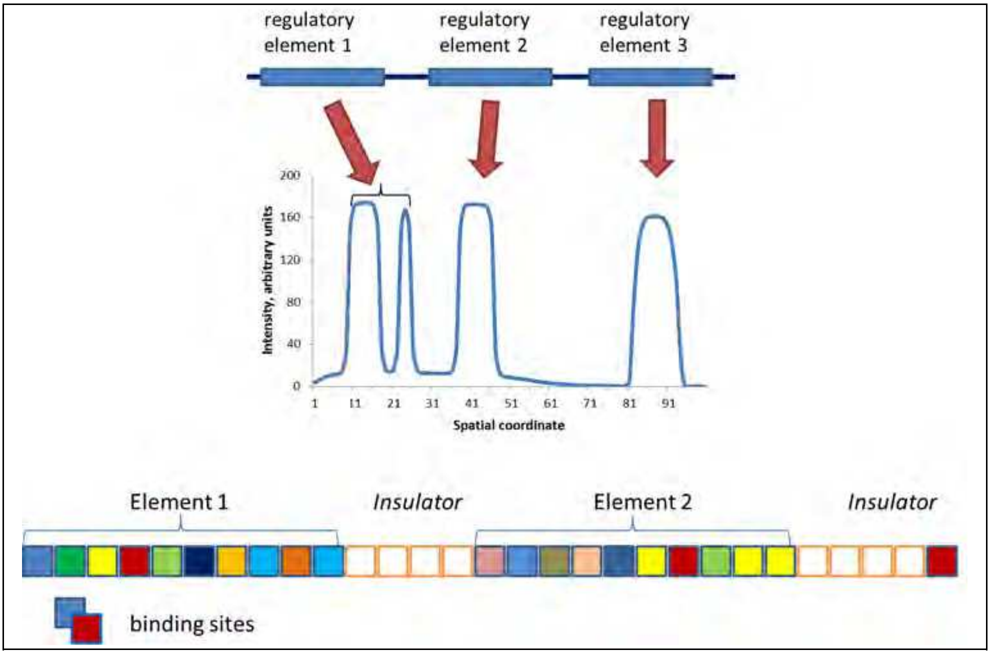


Fig. 5. One of the best studied examples of a gene from the *Drosophila* segmentation gene network – the *hunchback* (*hb*) gene. Bottom: the schematic organization of the *hb* regulatory region, with three separate autonomous regulatory elements (CRMs). Each regulatory element is a cluster of binding sites for, at least, five transcription factors (Bicoid, Caudal, Tailless, Hunchback, Huckebein, Giant, Kruppel & Knirps), shown as colored bars. Spacers (insulators) are also shown. Middle: mature expression pattern for the *hb* gene in an early fruit fly embryo (one-dimensional spatial expression profile, along the main head-to-tail embryo axis). Top: representation of the gene regulatory structure, each responsible for a different aspect of the *hb* expression pattern.

We can study the building up of the modern *hb* regulatory region through computational evolution from a single CRM ancestor. Starting from a single CRM with fitness score = Δ, the evolutionary search finding the 2nd CRM would double the score (2Δ); and so on sequentially to completion (score=3Δ; Fig. 6).

Coding the *hb* problem for computation highlights the levels of abstraction necessary to represent multiple CRMs:

DNA

*****TTAATCCGTT***…***CGAGATTATTAGTCAATTGC***…***GGATTAGC***…***GAAAGTCATAAAAACACATAATA***…**
BS for Bicoid&Kruppel Bicoid&Giant Bicoid Hunchback&Giant

Symbolically, CRM level (B for Bicoid, K for Kruppel, H for Hunchback, N for Knirps, G for Giant):

*****B K B G G B K B H G B K***…***N H H/N N H H N H K H H H*****
Element 1 Element 2

Symbolically, in octal numbers:

*****0 1 0 4 4 0 1 0 2 4 0 1***…***3 2 2 3 3 2 2 3 2 1 2 2 2*****
Element 1 Element 2

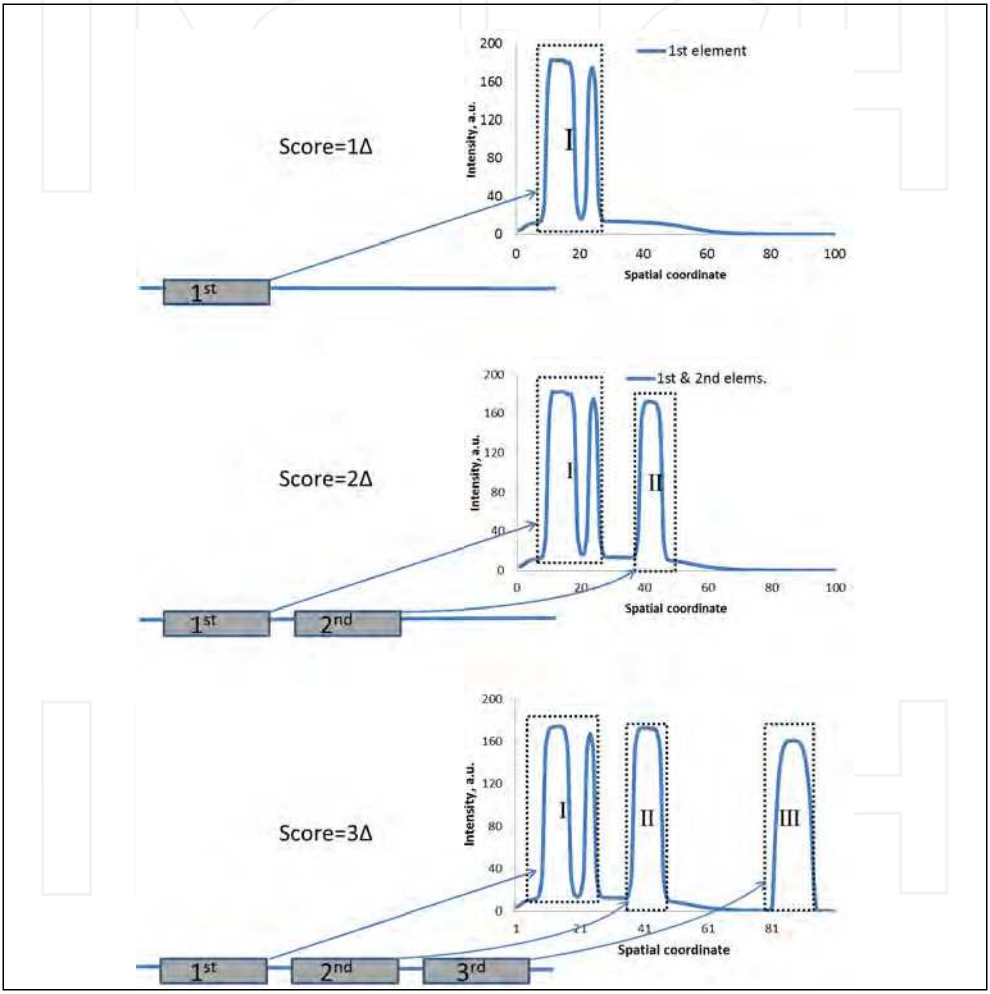


Fig. 6. A simplified scenario for the evolutionary origin of *hunchback* gene organization. A single element would insert into an ancestral gene with no elements, and, due to increased fitness, build up to the gene with three elements. Gene organization and the corresponding patterns of gene expression are shown schematically.

The BS's are finally coded as decimal pairs, where the 1st digit identifies the transcriptional factor and the 2nd digit represents its binding strength. To capture activator cooperativity and inhibitor quenching, neighbouring BS's can be allowed to alter binding strengths. GA and retroGA algorithms can perform crossover operations on these strings to evolve them. In contrast to the rrnP1 problem, where fitness is rated by transcriptional efficiency of the gene, fitness of the *hb* regulatory string depends on how well it produces the required spatial expression pattern. The strings are formal representations of real functional connections between genes in a network. Candidate strings must be solved in a reaction-diffusion model for spatial patterning, and the resulting pattern scored for fitness against experimental data (e.g. profile in Fig. 5).

2.2 The retroGA technique

As discussed in the Introduction, standard GA techniques, specifically through the use of point mutations to generate diversity in the chromosome, can destroy BB's which are important for fitness, slowing evolutionary searches. We have taken inspiration from the mechanisms of retroviral recombination to create crossover operators which preserve BB's. Our innovations are only in the crossover operators, all other actions of the algorithm are as in classical GA.

As discussed above, homology-based PCR techniques (DNA shuffling, sexual PCR) used in test tube evolution may be naturally interpreted as a generalization of retroviral recombination processes (Fig. 3), using n instead of 2 parent strings. Our retroGA operator generates a child string from a given "parent set", combining the function of reproduction and crossover. Crossover points are determined by regions of homology in the parent strings. The parent strings are selected from the population, as in standard GA, by one of several predetermined strategies, such as *truncation, roulette-wheel*, etc. One string is selected as a donor, the others as acceptors (Fig. 7).

In our reproduction and crossover procedure, a first pair of parent candidates is selected. These are the donor and acceptor-1 (Fig. 7). Their sequences are then compared going from left to right for a short distance L_{acc} (where $L_{acc} < L$, L is the length of the whole sequence). If the required zone of local homology is not found, another candidate for acceptor-1 is selected. The number of attempts to find a suitable acceptor is at most N_{acc}. If, and only if, a zone of complete homology of a size no less than L_{hom} symbols ($L_{hom} < L$) is found during an attempt to scan two sequences, do these two sequences become the donor and acceptor-1 pair. Replica generation is then initiated, and takes place in the first n symbols of the donor, from the first element to the last element of the region homologous between the two parents. Replication then jumps to acceptor-1, and acceptor-2 candidates are selected. A search for local homology takes place between acceptor-1 and the putative acceptor-2. If no such region is found, the next candidate is searched. This process is iteratively repeated until the replica (child) is completed, or until the N_{acc} limit is exceeded.

retroGA with point mutations: As discussed in the Introduction, crossover of BB's is more efficient than point mutation. In real retroviral recombination, however, it appears that both processes are present. Template switching between parent RNA strings tends to introduce mutations in the child sequence. For our retroGA, we include this effect by introducing one point mutation in one of a few starting sites in the portion of the child string being copied from the new acceptor. This addition provided speed-up for retroGA on RS, rrnP1-gene and hb-gene searches, but not on RR searches. Further analysis is needed to understand this difference.

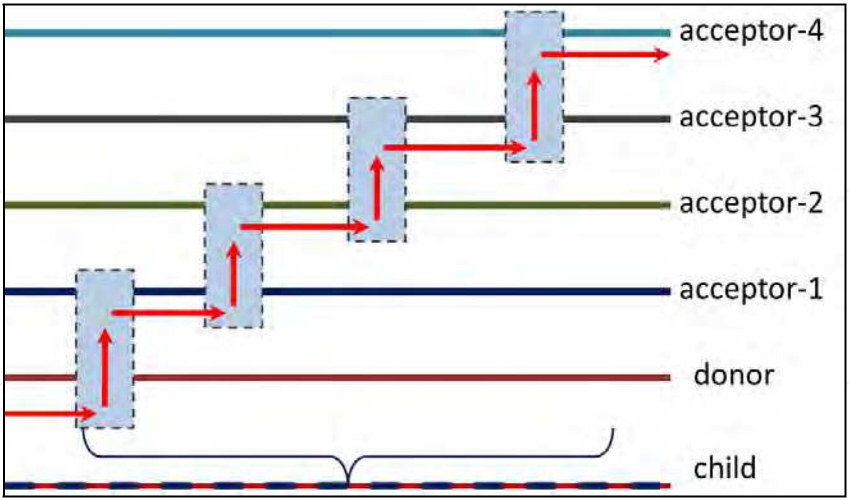


Fig. 7. Principle of the retroGA operator, an analogy to in vitro DNA shuffling techniques. The process of creating the child sequence by the operator starts with the donor-parent. Replication of the child from the donor-parent occurs if there is at least one region of homology (identity) between the donor and acceptor -1 (marked by gray rectangles). The process then jumps onto the acceptor-1 string. An acceptor-2 is found with a region of homology to acceptor-1, and the process repeats, copying from acceptor -1 and jumping to acceptor-2 (which becomes the third parent of the child sequence). The process of jumping from acceptor to acceptor continues until the creation of the child sequence is complete.

3. Results

In this section we present results on the efficiency of retroGA in comparison with standard GA (point mutations only). We do the comparison on RR and RS benchmark tests, as well as on the biological rrnP1 and *hb* gene sequence problems. Because all of these problems share a subbasin-portal type architecture, such computations allow us to begin to characterize the degree to which RR and RS test functions can predict behavior in gene searches. This is especially relevant if we can begin to use the analytical (mathematical) tools that have been developed for the RR and RS test functions to understand the gene search dynamics.

3.1 Crossover operators for RS problems

As a baseline, we have corroborated the RS results of van Nimwegen & Crutchfield (2000; 2001) with point mutation GA. Following their analytical and computational work provides a framework from which to understand the efficacy of our retroGA technique (including for the RS-like rrnP1 problem). In particular, they derived the dependence of the number of evaluations E to achieve the global optimum on the frequency of point mutations q and size of population M (point mutations only and roulette-wheel selection strategy). They found theory and computational experiments to be in good agreement.

We have reproduced their computational experiments and analyzed how average time to achieve a given fitness n empirically depends on n. The case of N = 4, K = 10 is shown in Fig. 8. The averaged time (in the average number of candidate string evaluations) to achieve the n+1 fitness level rises exponentially (Fig. 8A); plotting in semi-log scale confirms this (Fig. 8B).

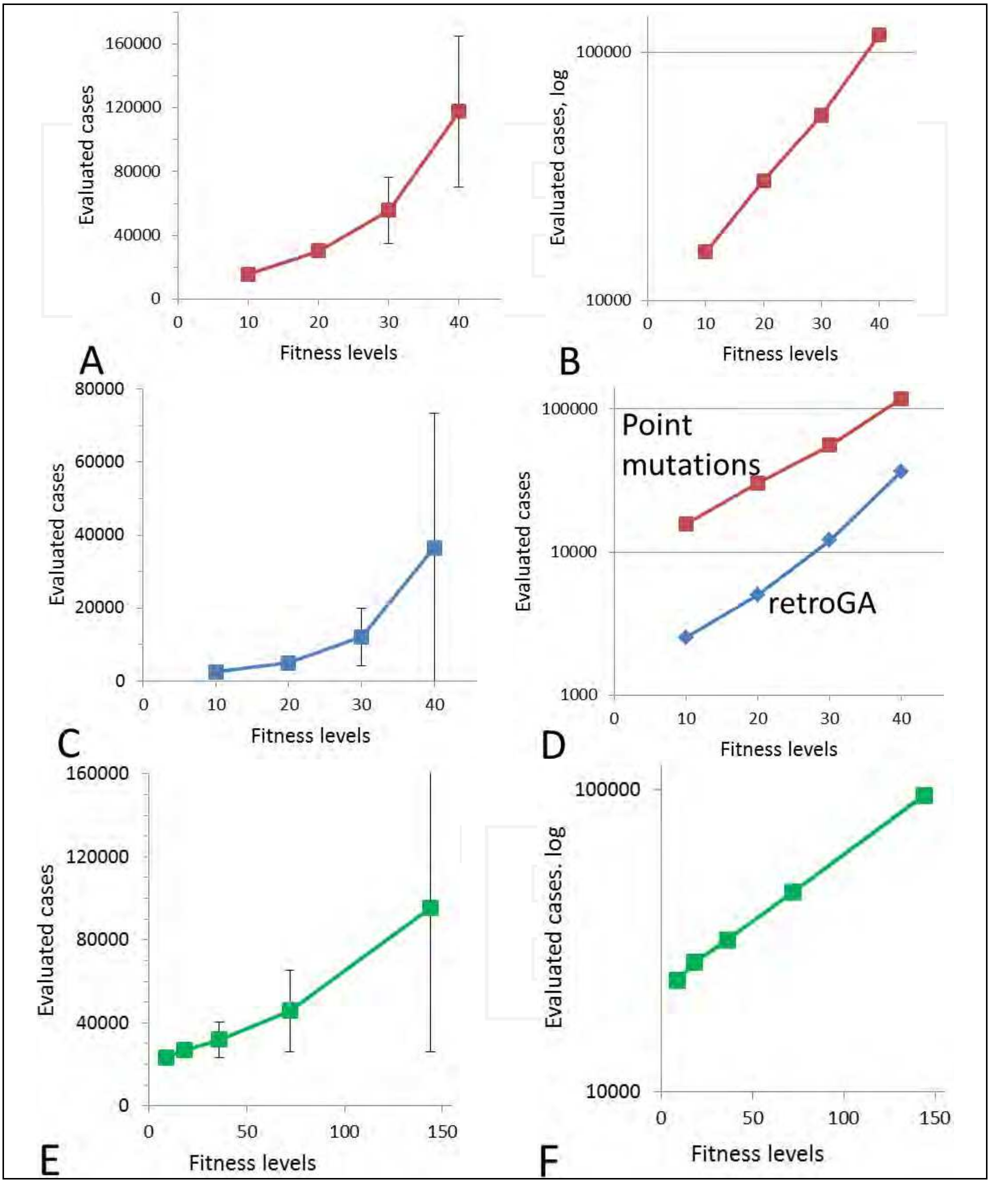


Fig. 8. Average total number of fitness function evaluations to achieve n^{th} fitness level for the Royal Staircase with N=4 blocks of length K=10, in comparison with the rrnP1 gene test.

Evolutionary search with point mutations only (A, B; Cf. van Nimwegen & Crutchfield (2000; 2001)) vs. search by retroGA algorithms (C,D). Each data point was obtained as an average over 200 GA runs. A, B) Corroboration of the van Nimwegen and Crutchfield tests with point mutation only (no crossover). Population size M=30,000; mutation rate q=0.01; roulette-wheel selection strategy. C,D) Our tests with the retroGA operator as the mutation procedure (see text for details). retroGA speeds the search by about 5.5 times compared to point mutations only. M=5,000; truncation selection strategy. E,F) retroGA results on the simplified rrnP1 test function, M=24,000.

3.1.1 retroGA speeds up search on RS fitness functions

Testing both versions of our retroGA operators – crossover without point mutations and crossover with associated point mutations (see section 2.2) clearly shows that the combined crossover/point mutation mechanism is the most effective procedure on RS tests, speeding up searches about five-fold. Fig. 8C,D shows the retroGA crossover/point mutation results for the same RS fitness function as in previous section (N=4, K=10). The RS optimum was achieved in 36,469±36,991 solution evaluations, about 5.5 times faster than by standard GA (point mutations only; ~200,000 evaluations). It can also be seen in Fig. 8C,D, that the retroGA search shows a nearly exponential dependence between search efficacy and the n level, like GA with mutations only (Fig. 8A,B).

3.2 Crossover operators for the rrnP1 problem

We found that the simplified version of the rrnP1 test behaved very closely to the RS tests with N=4, K=10. Though we had initially thought of rrnP1 in terms of RS organization (section 2.1.3.1), we were surprised at the closeness, because the rrnP1 test is specified by quaternary strings (the four DNA letters A, T, G, C) and the string length is about twice the RS test, owing to spacers. The dependence of the search efficacy on the n level is still exponential (Fig. 8E,F) for retroGA on rrnP1, as on RS. retroGA on the simplified rrnP1 (with five blocks) was over five times faster than GA with one-point crossover (crossover rate = 0.01): 95,618±69,575 (Fig. 8E,F) vs. 512,040±48,378 average evaluations. Success on the rrnP1 problem, and the parallels to the well-characterized RS function, suggests that retroGA is an effective technique for prokaryotic gene search problems, and could contribute to real problems of forced (directed) evolution of bacterial promoters in the test tube. We will follow up these connections with modern synthetic biology in the Discussion.

3.3 Crossover operators for R1 - R3 functions

In this section we characterize retroGA performance on R1 to R3. These functions have been well-studied in the literature, and as discussed above, have some of the fundamental motifs necessary for modeling gene organization. Testing retroGA both with and without point mutations after crossover showed little effect (in contrast to RS). The results shown here are for retroGA crossover without accompanying mutation.

We have already reported on the several-fold speed-up of retroGA vs. standard GA for RR problems (Spirov & Holloway, 2010). Here we will focus on the dependence of retroGA performance on key computational parameters. It is known that the R1 – R3 functions behave similarly in computational experiments (Forrest & Mitchell, 1993a,b; Mitchell et al., 1992; Mitchell, 1996). Therefore, we will focus on R1 tests, and present comparisons to R3 performance.

3.3.1 Dependence on population size

Theoretical and computational studies have shown that many performance parameters of R1 depend on population size M (van Nimwegen et al., 1999). With an aim to applying retroGA to real directed molecular evolution problems (in vitro), it is important to characterize the population size dependence (and to connect the theoretical knowledge of R1 to real biological problems). We tested M dependence (Fig. 9) for a set of parameters found to be close to optimal in other tests (see next sub-sections).

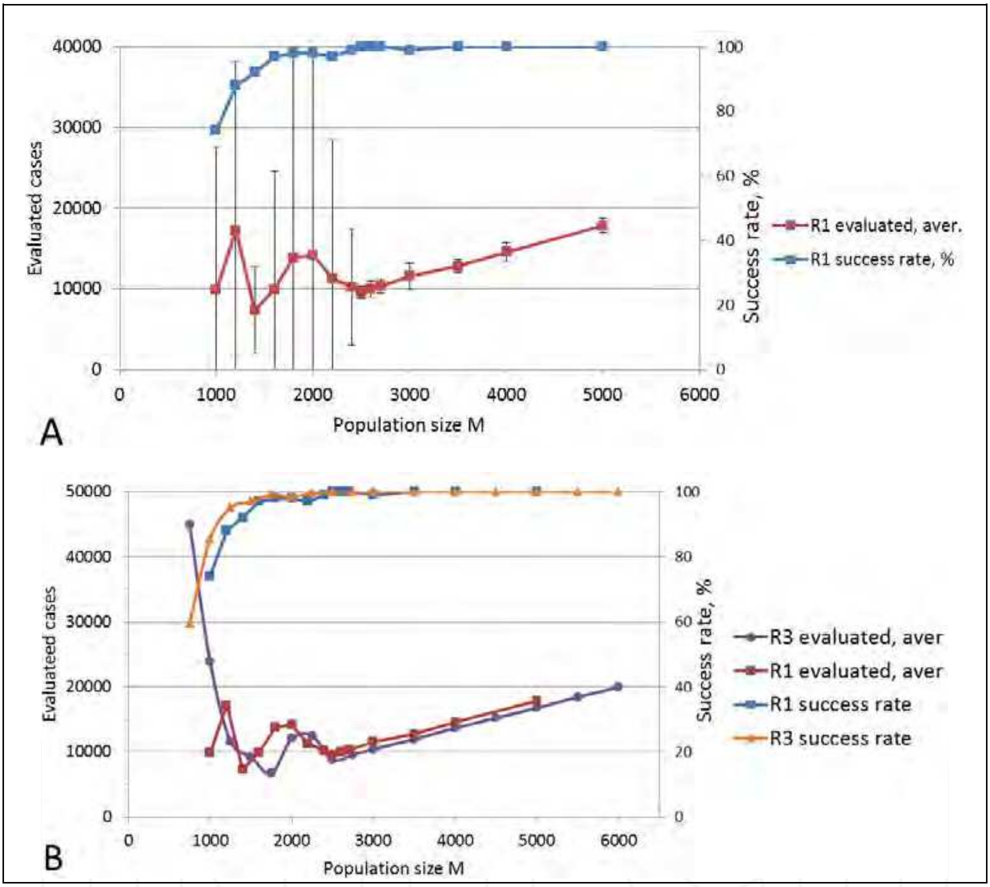


Fig. 9. Royal Road fitness functions 1 & 3: Empirical dependence of average total number of fitness function evaluations on population size M. Each data point was obtained as an average over 100 retroGA runs. Other parameters: limit of acceptor parents N_{acc}=100. A) The R1 tests. B) R1 vs. R3 results.

While retroGA performance, as number of evaluations, is relatively consistent across population size (Fig. 9A, red), there is a relatively narrow window of M from ~2,400 to 2,700 in which number of evaluations and the standard deviation on these are both low. In this range of M, retroGA is about 6 times faster than standard GA (~10,000 vs. ~60,000 evaluations). While M=<2,400 can achieve fast results, the standard deviation is higher and a lower percentage of runs achieve the global optimum (Fig. 9A, blue). For M>2700, the number of evaluations steadily rises with population size. We find that retroGA behaves very similarly on R3 as on R1 (Fig. 9B), as seen earlier with standard GA (Mitchell, 1996). The main conclusion here is that the retroviral crossover is most efficient just as the success rate approaches 100% - small populations are enough for efficient and reliable retroGA searches.

3.3.2 Dependence on retroGA parameters

retroGA algorithms have only three parameters: the maximum number of acceptors N_{acc} to use in synthesizing a child-string (see Fig. 7); the maximum acceptor length to search for local homology L_{acc}; and the maximum length of the local homology region L_{hom} (see section 2.2).

Dependence on number of acceptors, N_{acc}: Even for such a simple problem as R1, a high number of acceptors helps greatly (Fig. 10A). Having only a few acceptors gives a very high number of evaluations; adding acceptors, up to about 40, drops the number of evaluations many-fold. More parents provides a more effective evolutionary search.

Dependence on maximum acceptor length to search for local homology L_{acc}, and on maximum length of local homology region L_{hom}: As explained in section 2.2, in this work we scan each acceptor for local homology for a certain distance from the jump point (see Fig. 7), using length from 2 to L_{acc} to find a homologous sequence of length from 2 to L_{hom}. Fig. 10B shows results for the dependence of efficacy on L_{acc} for the R1 function. The algorithm shows a great increase in efficiency going from L_{acc} ~10% to 20% bit-string length: for the 64 bit test strings, L_{acc} should be over 13 bits. Tests with L_{hom} show a smoother increase in efficiency, and indicate that L_{hom} should be kept over ~40% bit-string length (Fig. 10C).

3.3.3 Time to achieve fitness level *n*

In addition to total number of evaluations, RR and RS functions have been evaluated in terms of epoch duration, the time a population stays at a given level n searching for the solution to the next level n+1. In developing a theory for the R1 problem, van Nimwegen and colleagues (1999), predicted that epoch duration depends exponentially on epoch number (fitness level) n. Computationally, we do see a roughly exponential dependence for standard GA (no crossover; Fig. 11C), though it is not strictly exponential (Fig. 11D, semi-log plot; this in contrast to RS, which shows strict exponentiality, Fig. 8). Interestingly, retroGA with a high level of acceptors shows a linear relationship between number of evaluations and n (Fig. 11A). The dependence becomes exponential again for low acceptor numbers (Fig. 11B). The retroGA operator with many acceptors is far more efficient than standard GA (Fig. 11D) in terms of keeping epoch duration low.

3.3.4 Tests with ternary strings

While the R3 fitness function has strong parallels to the typical clustering of binding sites in gene regulatory regions (Fig. 1A), a major difference is that DNA "strings" are quaternary (four-letter) ones. Here, we check how such dimensionality affects overall efficacy in GA tests. Specifically, we tested R1 with optimized parameter sets (c.f. Fig. 11A) with ternary strings. (Quaternary strings were too computationally intensive for GA for test purposes.)

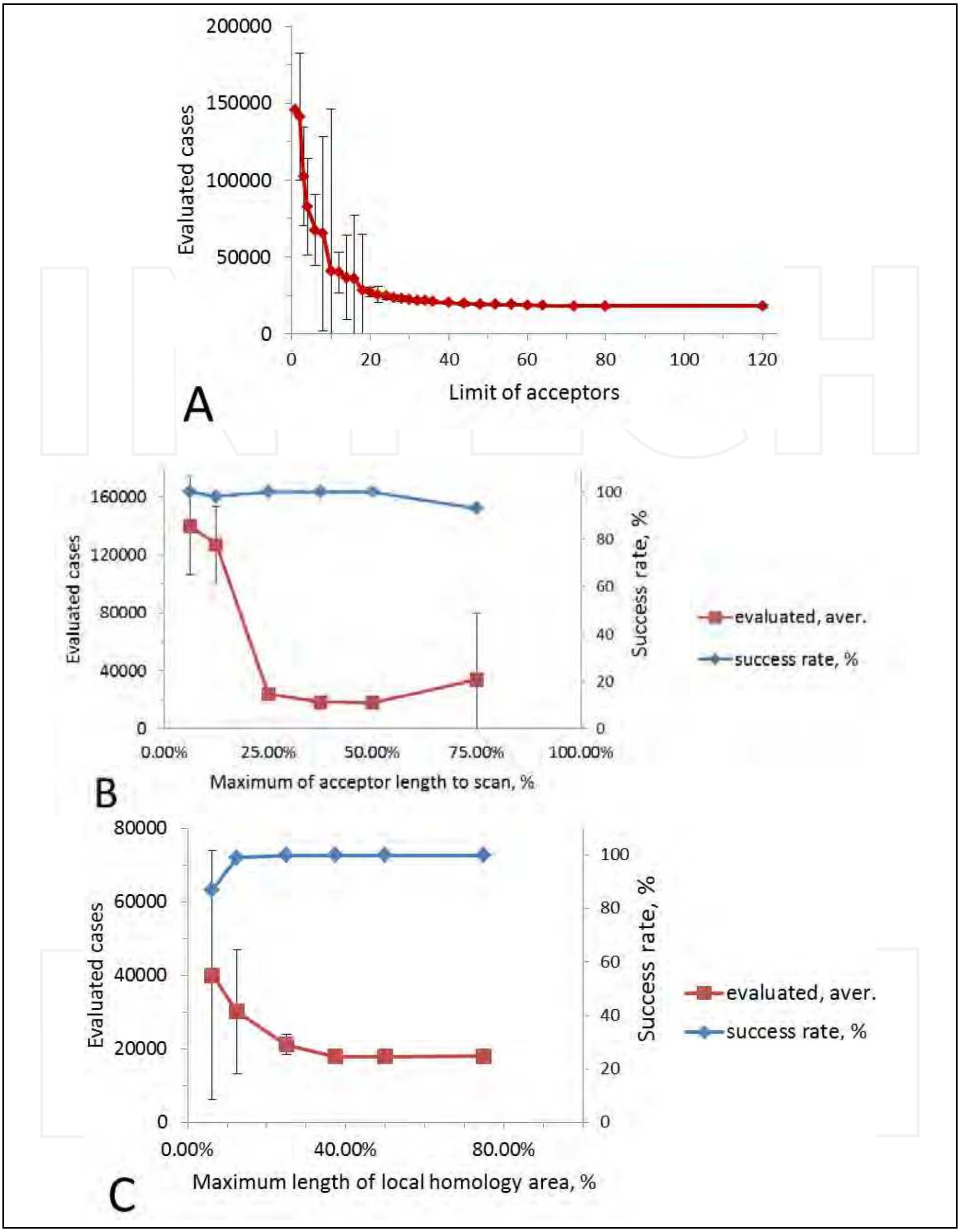
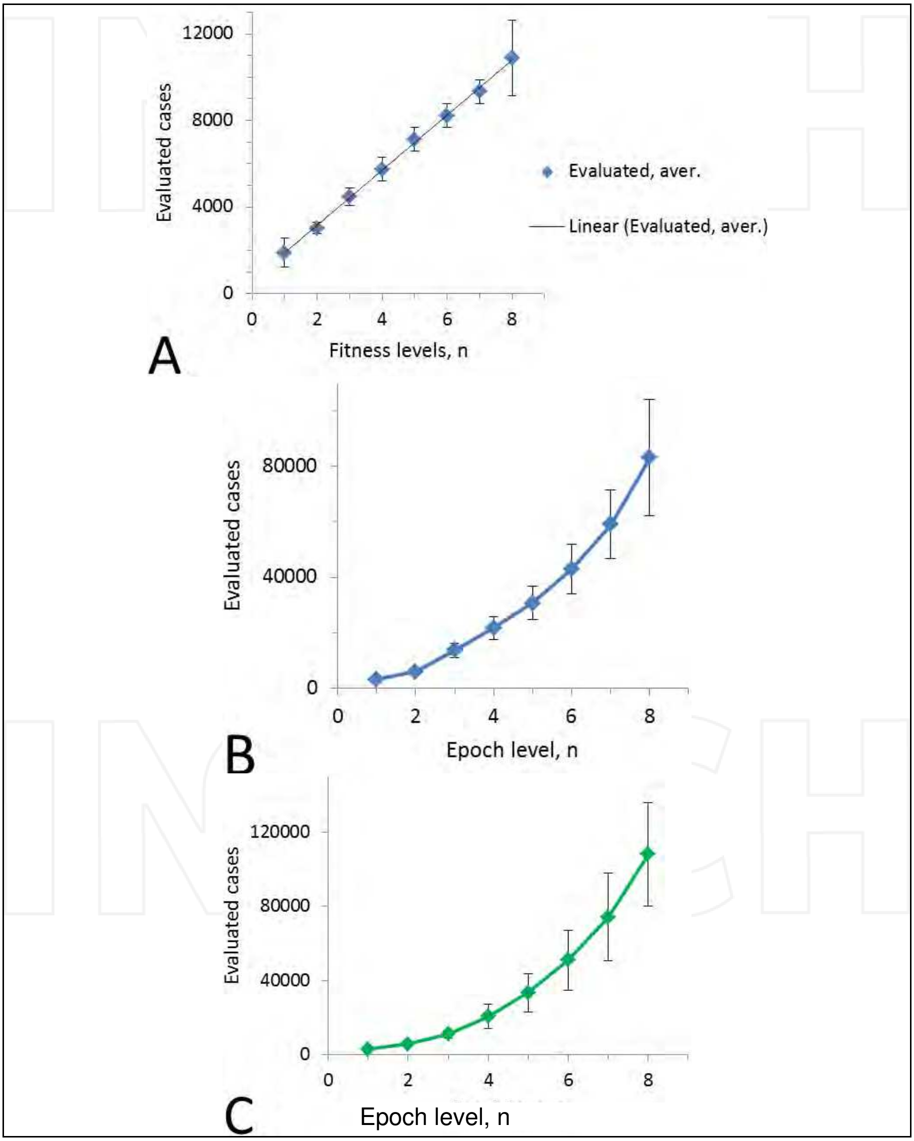


Fig. 10. Tests with the retroGA parameters. Each data point was obtained as an average over 200 retroGA runs. A) Tests on the number of acceptors to use, N_{acc}, for the R1 fitness function. B) Tests on the maximum acceptor length to search for local homology L_{acc}. C) Tests on the maximum length of local homology to find, L_{hom}. M=5,000 in all computations.

Fig. 12 shows the same linear relationship of epoch duration on n found in Fig. 11A for retroGA on binary strings, but the number of evaluations goes up dramatically, with the ternary problem taking ~20 times more evaluations on average than the binary problem (10,893 vs 228,419 – Fig. 11A vs. Fig. 12). There is a large price to pay for increasing the dimension of the problem; we would expect the quaternary problem to be many times slower again.



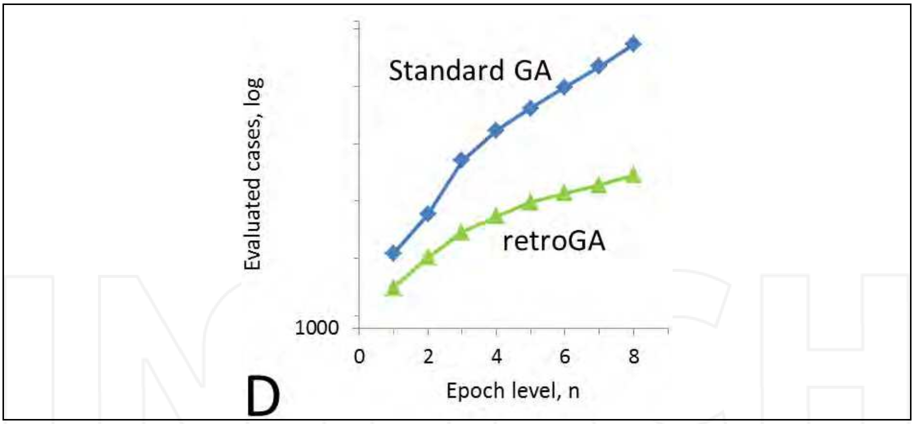


Fig. 11. Epoch durations for the R1 problem: time to achieve fitness level n. Each data point was obtained as an average over 100 retroGA runs. A) retroGA - M=2600; N_{acc}=48; L_{acc}=32; L_{hom}=32. B) retroGA - M=2600; N_{acc}=4; L_{acc}=32; L_{hom}=32. C) Standard GA - M=2600; crossover rate = 0.1; mutation rate = 1/L . D) retroGA vs. standard GA: plots A (green) and C (blue) in semi-log coordinates.

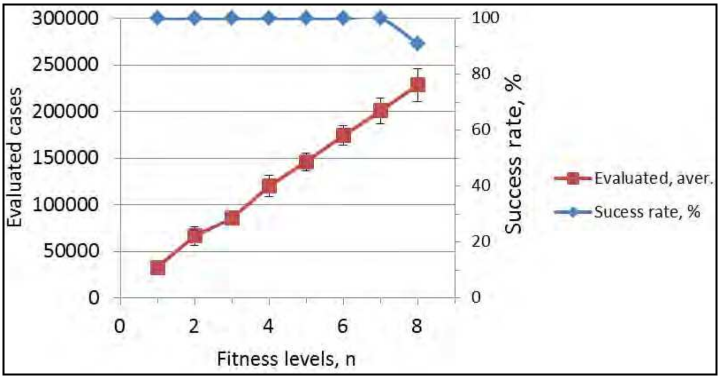


Fig. 12. Epoch duration for retroGA on the R1 problem with ternary strings. Cf. Fig. 11A. Each data point is an average over 100 retroGA runs.

3.4 Crossover to design the *hunchback* gene model

The *hb* gene problem as formulated in section 2.1.3.2 is like R3 in form (N=3; K=16; spacers of length 4), but with octal-digit strings and a substantial level of degeneracy in its three building blocks. The sequential search for CRM's also gives this problem an RS-like character. While the hb search has RR- and RS-like qualities, which should aid in analysis of the problem, the degeneracy of the building blocks is not captured by the test functions, but this does bring the problem closer to real life problems of forced evolution.

We found retroGA (crossover and point mutations) to be an effective method for solving the *hb* gene problem. Specifically, retroGA was over 4-fold faster than standard GA (325,594±59,456 vs. 1,373,246±198,698; averaged over 100 runs).

The *hb* gene problem is the only one in this chapter which has redundant building block sequences. Results show that these blocks are highly redundant. Fig. 13 shows 100 good solutions for the *hb* regulatory sequence. Each row is a solution, with octal-digit represented on an 8-bit grey-scale. There is no discernible pattern outside of the spacer regions (black stripes), illustrating how high the redundancy is in such a problem (for solutions which match the data *well*).

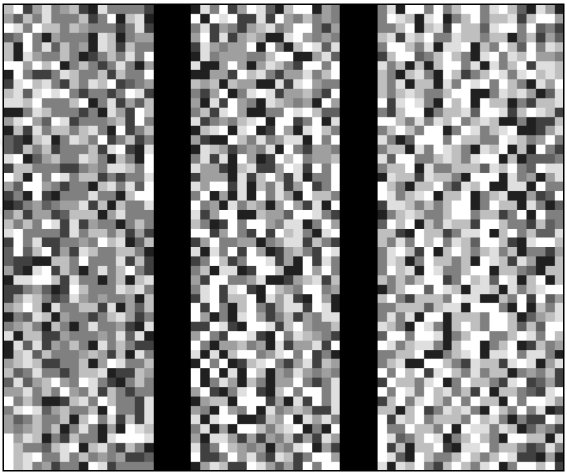


Fig. 13. Grey-scale representation of an aligned stack of 100 good solutions of the *hb* gene problem. Each row corresponds an octal-digit string. The two vertical black columns correspond to two spacer regions of four elements in length each.

4. Discussion

A major aim with this work is to bridge evolutionary computations from benchmark cases, such as Royal Road and Royal Staircase, which are well-understood theoretically (in terms of mathematical analysis), to biological cases, which can serve as a basis for more efficient directed molecular evolution in the test tube and for understanding the mechanisms of biological evolution at the level of gene regulatory sequences.

4.1 Towards a theory of evolution of biological macromolecules

Using analytical tools from statistical mechanics, dynamical systems theory, and mathematical population genetics, van Nimwegen and co-authors (van Nimwegen & Crutchfield, 2000; 2001; van Nimwegen et al., 1999; Crutchfield & Nimwegen, 2001) developed a detailed and quantitative description of the search dynamics for the RS and RR class of problems that exhibit epochal evolution. From this, the authors could analytically predict optimal parameter settings for this class of problems. More generally, the detailed understanding of the behaviour for this class of problems provides valuable insights into the emergent mechanisms that control the dynamics in more general settings of evolutionary searches and in other population-based dynamical systems. By establishing the RR and RS characteristics of gene regulatory problems, we can use this theoretical background to anchor our understanding of more realistic biological search cases.

4.1.1 Royal staircase theory

For RS (point mutation only (no crossover) and roulette-wheel selection strategy), van Nimwegen & Crutchfield (2000) derived an analytical expression for the dependence of the number of evaluations E to achieve the global optimum on the frequency of point mutation q and population size M. Numerical tests (Fig. 14 upper, a) closely follow the analytically predicted dependence (Fig. 14 upper, b). With numerical tests, we found (Fig. 14, (c)) the retroGA operator to have a similar dependence of E on M and Q (Q is the point mutation rate for retroGA; also see Fig. 8). retroGA uses substantially larger populations (M>=2,500) but is several times faster than the standard GA studied by van Nimwegen and Crutchfield. This similar general character of the dependence is promising for extending the van Nimwegen–Crutchfield theory to the case of retroGA crossover.

4.1.2 Royal road theory

van Nimwegen and colleagues (1999) developed an analytical theory for the R1 problem (without crossover and with roulette-wheel selection) and deduced a series of expressions describing the behaviour of this evolutionary search at low mutation rate q. From these, they could predict that high mutation rate would be associated with lower average fitness; they also derived a basis for the exponential dependence of number of evaluations on fitness level. Such predictions are very intriguing for understanding searches and diversity in test tube directed evolution. However, our numerical results with the retroGA operator show a linear relationship of number of evaluations on fitness. This indicates that the analytical results, for the inefficient point mutation operator, may not be seen in biological situations which use more efficient BB-preserving (crossover) operators. Further work is needed to establish the applicability of the point mutation analysis to crossover mechanisms.

4.2 Future prospects for applying the computational results to directed evolution of gene circuits

Our aim is to use the techniques developed in this chapter to aid the directed evolution of bacterial and yeast gene promoters in the laboratory. Several approaches to improve and/or analyze such promoters via directed evolution have been undertaken by experimentalists (Schmidt-Dannert, 2001; Haseltine & Arnold, 2007; Collins et al., 2006). While there is still some gap between the gene models in this chapter and real macromolecular evolution, we hope to have outlined the directions that can be taken for the computational work to provide a stronger theoretical basis for directing and analysing experiments.

We have focused on evolution of sequences, with biological applications in gene promoter structure. The growing field of synthetic biology also includes a great deal of work on designing gene circuits, where large numbers of genes affect each other's expression (e.g. the *Drosophila* segmentation network). Haseltine & Arnold (2007) have identified three primary limitations in using directed evolution to design gene circuits: (a) the evolutionary search space for a genetic circuit composed of many genes is generally too large to explore efficiently; (b) detuning parameters (reducing function) is much easier than improving function; and (c) although selecting for independent properties is possible, it usually requires setting up multiple rounds of screening or selection. In this area, using

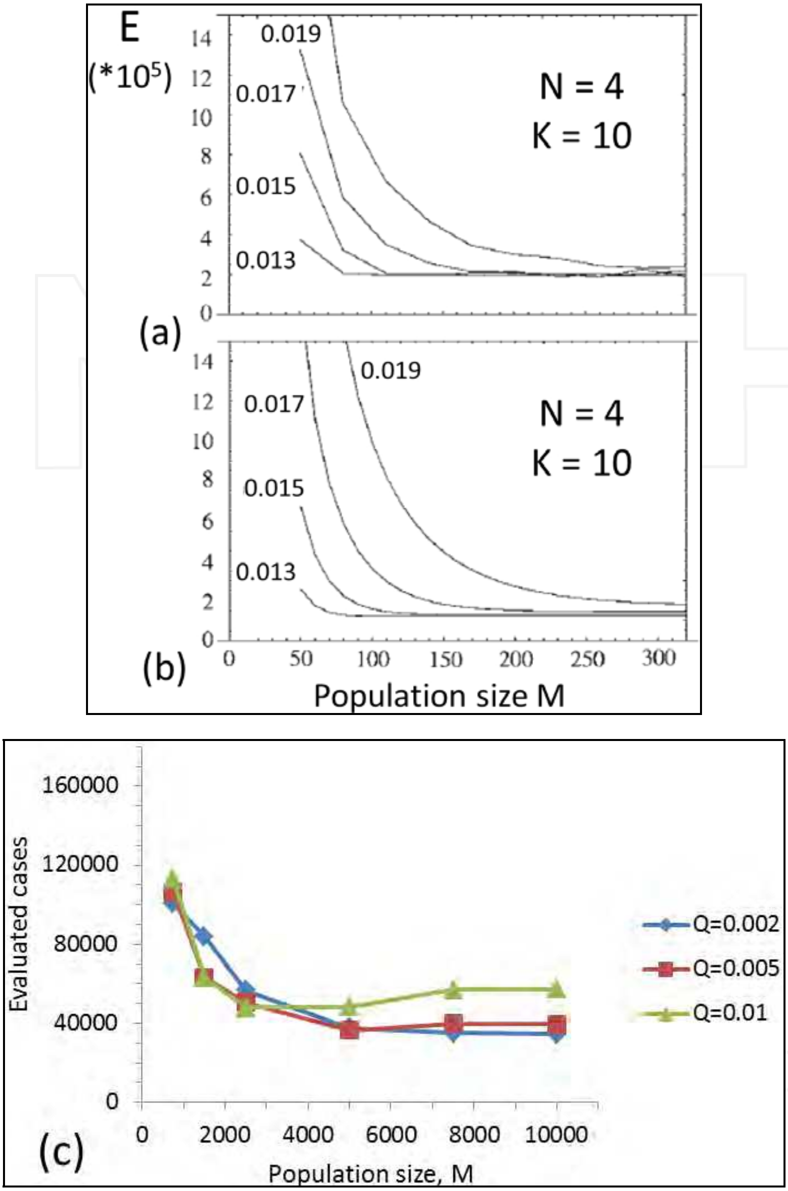


Fig. 14. Dependence of search efficacy E on population size M and mutation rate q. Upper figure: (a) experimentally obtained dependence of E on population size M, each data point is an average over 250 GA runs; (b) shows the theoretical predictions for E as a function of M (van Nimwegen & Crutchfield, 2000). In both, N=4 blocks of length K=10 (c.f. Fig. 8) for four different mutation rates: q ∈ {0.013, 0.015, 0.017, 0.019}. Lower figure (c): Tests with retroGA, showing the empirical dependence of E on M and parameter Q (the point mutation rate for retroGA).

mathematical models to suggest mutational targets can greatly speed up the process and help overcome each of these limitations.

In the wider perspective, an appropriate theory of molecular evolution in the test tube, which includes effective mathematical analysis of new experimental recombination techniques, as described in this chapter, would give a new way to design gene circuits effectively. We hope that the theoretical and computational results discussed in this chapter can facilitate progress in this direction.

5. Conclusion

In this chapter, we have discussed some of the computational issues for evolutionary searches to find gene regulatory sequences. One of the challenges for such searches is to maintain building blocks (meaningful 'words') through genetic change operators. Mutation operators in standard GA frequently destroy such BB's and slow searches. We have introduced the retroGA operator, inspired by retroviral recombination and in vitro DNA shuffling mechanisms, to copy blocks of genetic information. The Royal Road (RR) and Royal Staircase (RS) benchmark functions have been developed for analysing evolutionary searches which preserve BB's. RR and RS theory provide a mathematical framework for understanding the dynamics of searches which have subbasin-portal fitness landscapes. Empirically, we see that retroGA searches share many of the characteristics of RR and RS, but that features, such as multiple parent strings, which can greatly speed up searches, also produce different optimization dynamics than RR and RS. We aim to bridge between RR and RS functions and real biological applications. Through working on specific cases, the rrnP1 and *hb* gene regulatory regions, we are altering simple, binary RR functions to take into account BS clustering and non-binary coding. While real biological problems have a yet higher degree of complexity, our aim is to sketch how EC computations can be used to aid experimental biological work. Computational theory can contribute to both understanding how real gene structures have evolved and to speeding up laboratory work on directed evolution of macromolecules in the test tube.

6. Acknowledgments

AVS wishes to thank Vladimir Fomin for help with hardware and software, Vladimir Levchenko and Marat Sabirov for discussion the results. This work was supported by the Joint NSF/NIGMS BioMath Program, 1-R01-GM072022 and the National Institutes of Health, 2R56GM072022-06. DMH thanks NSERC Canada and BCIT for financial support.

7. References

An W., and Telesnitsky A. (2002). HIV-1 genetic recombination experimental approaches and observations, *AIDS Rev*, Vol.4, pp. 195-212.

Ancel-Myers, L.W. & Fontana W. (2005). Evolutionary Lock-in and the Origin of Modularity in RNA Structure, In *Modularity –Understanding the Development and Evolution of Natural Complex Systems*, pp. 129-141, W.Callebaut and D.Rasskin-Gutman (editors), MIT Press, Cambridge, MA.

Botstein D. (1980). A theory of modular evolution for bacteriophages, *Ann N Y Acad Sci*, Vol.354, pp. 484–491.

Brosius, J. (1999). Genomes were forged by massive bombardments with retroelements and retrosequences, *Genetica,* Vol.107, pp. 209-238.

Chen J. & Wood D. H. (2000). Computation with Biomolecules, *Proc. Nat. Acad. Sci. USA,* Vol.97, pp. 1328-1330.

Collins CH, Leadbetter JR, & Arnold FH. (2006). Dual selection enhances the signaling specificity of a variant of the quorum-sensing transcriptional activator LuxR, *Nature Biotech,* Vol.24, pp. 708–712.

Crutchfield J. P. & van Nimwegen E. (2001). The Evolutionary Unfolding of Complexity, In *Evolution as Computation, DIMACS workshop*, Springer-Verlag, New York.

Crutchfield J. P. (2002). When Evolution is Revolution—Origins of Innovation, In *Evolutionary Dynamics—Exploring the Interplay of Selection, Neutrality, Accident, and Function*, J. P. Crutchfield and P. Schuster (eds.), Santa Fe Institute Series in the Science of Complexity, Oxford University Press, New York, pp. 101-133.

Forrest S., & Mitchell M. (1993a). What Makes a Problem Hard for a Genetic Algorithm? Some Anomalous Results and Their Explanation, *Machine Learning,* Vol.13, (No 2), pp. 285 -319.

Forrest S. & Mitchell M. (1993b). Relative building-block fitness and the buildingblock hypothesis, In D. Whitley (ed.), *Foundations of Genetic Algorithms,* Vol.2, pp. 109-126, San Mateo, CA Morgan Kaufmann.

Goldberg D. E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning* (Addison-Wesley, Reading, MA).

Haseltine EL, & Arnold FH. (2007). Synthetic gene circuits Design with directed evolution, *Annu Rev Biophys Biomol Struct,* Vol.36, pp.1–19.

Hengen PN, Bartram SL, Stewart LE, & Schneider TD. (1997). Information analysis of Fis binding sites, *Nucleic Acids Res,* Vol.25, pp. 4994-5002.

Henkel C. V., Back T., Kok J. N., Rozenberg G., & Spaink, H. P. (2007). DNA computing of solutions to knapsack problems, *Biosystems,* Vol.88, (No 1), pp.156 – 162.

Hirvonen CA, Ross W, Wozniak CE, Marasco E, Anthony JR, Aiyar SE, Newburn VH, & Gourse RL. (2001). Contributions of UP elements and the transcription factor FIS to expression from the seven rrn P1 promoters in Escherichia coli, *J Bacteriol,* Vol.183, pp. 6305-6314.

Holland J. H. (1975). *Adaptation in Natural and Artificial Systems* (Ann Arbor, MI Univ. Michigan Press).

Jeziorska DM, Jordan KW, & Vance KW. (2009). A systems biology approach to understanding cis-regulatory module function, *Sem. Cell & Dev. Biol,* Vol.20, pp. 856–862.

Kitano, H. (2004). Biological robustness. *Nat. Rev. Genet.* Vol.5, pp. 826–837.

Long M., Betran E., Thornton K., et al. (2003). The origin of new genes Glimpses from the young and old, *Nat Rev Genet,* Vol.4, pp. 865–875.

Lu T. K., Friedland A. E., Wang X., Shi D., Church G. M., & Collins J. J. (2009). Synthetic Gene Networks that Count, *Science,* Vol.324, (No 5931), pp. 1199-1202

Maheshri, N. & Schaffer D.V. (2003). Computational and experimental analysis of DNA shuffling, *Proc Natl Acad Sci USA,* Vol.100, (No 6), pp. 3071-6.

Mitchell M. (1996). *An Introduction to Genetic Algorithms.* MIT Press.

Mitchell M., Forrest S. & Holland J. H. (1992). The Royal Road for genetic algorithms Fitness landscapes and GA performance, In *Proceedings of the First European Conference on Artificial Life,* Cambridge, MA MIT Press/Bradford Books.

Negroni M., & Buc H. (2001). Mechanisms of retroviral recombination, *Annu Rev Genet,* Vol.35, pp. 275-302.

Ross W, Aiyar SE, Salomon J, & Gourse RL. (1998). Escherichia coli promoters with UP elements of different strengths modular structure of bacterial promoters, *J Bacteriol,* Vol.180, pp. 5375-5383.

Ross W, Thompson JF, Newlands JT, & Gourse RL. (1990). E.coli Fis protein activates ribosomal RNA transcription in vitro and in vivo, *EMBO J,* Vol.9, pp. 3733-3742.

Schmidt-Dannert C., (2001). Directed evolution of single proteins, metabolic pathways, and viruses, *Biochemistry,* Vol.40, pp. 13125–13136.

Schneider DA, Ross W, & Gourse RL. (2003). Control of rRNA expression in Escherichia coli, *Curr Opin Microbiol,* Vol.6, pp. 151-156.

Sen S., Venkata Dasu V., & Mandal B. (2007). Developments in directed evolution for improving enzyme functions, *Applied biochemistry and biotechnology,* Vol.143, pp. 212–223.

Shapiro, J.A. (1999). Transposable elements as the key to a 21st century view of evolution, *Genetica,* Vol.107, pp. 171–179.

Shapiro, J.A. (2002). Repetitive DNA, genome system architecture and genome reorganization, *Res Microbiol,* Vol.153, pp. 447-53.

Shapiro, J.A. (2010). Mobile DNA and evolution in the 21st century. *Mobile DNA*, Vol.1 (No4).

Spirov A.V., Borovsky M., & Spirova O.A. (2002). HOX Pro DB The functional genomics of hox ensembles, *Nucleic Acids Research,* Vol.30, No 1, pp. 351 – 353.

Spirov A. V., & Holloway D. M. (2010). Design of a dynamic model of genes with multiple autonomous regulatory modules by evolutionary computations, *Procedia Comp. Sci,* Vol.1, (No 1), pp. 1005-1014.

Spirov A.V., Bowler T. & Reinitz J. (2000). HOX-Pro A Specialized Database for Clusters and Networks of Homeobox Genes, *Nucleic Acids Research,* Vol.28, (No 1), pp. 337-340.

Stemmer W.P. (1994a). DNA shuffling by random fragmentation and reassembly in vitro recombination for molecular evolution, *Proc Natl Acad Sci USA,* Vol.91, pp. 10747-10751.

Stemmer W.P. (1994b). Rapid evolution of a protein in vitro by DNA shuffling, *Nature,* Vol.370, (No 6488), pp. 389-391.

Stormo GD. (2000). DNA binding sites representation and discovery, *Bioinformatics,* Vol.16, (No 1), 16–23.

Sun F. (1999). Modeling DNA shuffling, *J Comput Biol,* Vol.6, (No 1), pp. 77-90.

van Nimwegen E., & Crutchfield J. P. (2001). Optimizing Epochal Evolutionary Search Population-Size Dependent Theory, *Machine Learning Journal*, Vol.45, pp. 77-114.

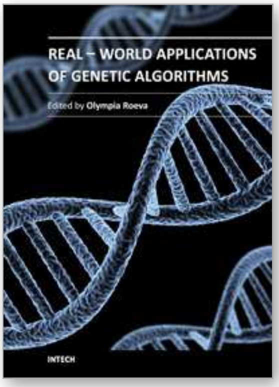
van Nimwegen E., & Crutchfield J. P. (2000). Optimizing Epochal Evolutionary Search Population-Size Independent Theory, *Computer Methods in Applied Mechanics and Engineering,* Vol.186, (No 2-4), pp. 171-194.

van Nimwegen E., Crutchfield J. P., & Huynen M. (1999). Neutral Evolution of Mutational Robustness, *Proc Natl Acad Sci USA,* Vol.96, pp. 9716-9720.

van Nimwegen E., Crutchfield J. P., & Mitchell M. (1997). Finite Populations Induce Metastability in Evolutionary Search, *Physics Letters A,* Vol.229, pp. 144-150

Voigt, C. A., Martinez, C., Mayo, S.L., Wang, Z-.G., & Arnold, F.H. (2002). Protein building blocks preserved by recombination, *Nature Structural Biology,* Vol.9, pp. 553-558.

von Dassow, G., Meir., E., Munro, E. M., & Odell, G. M. (2000). The segment polarity network is a robust developmental module, *Nature* Vol. 406, pp.188 - 192.



Real-World Applications of Genetic Algorithms
Edited by Dr. Olympia Roeva

ISBN 978-953-51-0146-8
Hard cover, 376 pages
Publisher InTech
Published online 07, March, 2012
Published in print edition March, 2012

The book addresses some of the most recent issues, with the theoretical and methodological aspects, of evolutionary multi-objective optimization problems and the various design challenges using different hybrid intelligent approaches. Multi-objective optimization has been available for about two decades, and its application in real-world problems is continuously increasing. Furthermore, many applications function more effectively using a hybrid systems approach. The book presents hybrid techniques based on Artificial Neural Network, Fuzzy Sets, Automata Theory, other metaheuristic or classical algorithms, etc. The book examines various examples of algorithms in different real-world application domains as graph growing problem, speech synthesis, traveling salesman problem, scheduling problems, antenna design, genes design, modeling of chemical and biochemical processes etc.

How to reference
In order to correctly reference this scholarly work, feel free to copy and paste the following:

Alexander V. Spirov and David M. Holloway (2012). New Approaches to Designing Genes by Evolution in the Computer, Real-World Applications of Genetic Algorithms, Dr. Olympia Roeva (Ed.), ISBN: 978-953-51-0146-8, InTech, Available from: http://www.intechopen.com/books/real-world-applications-of-genetic-algorithms/new-approaches-to-designing-genes-by-evolution-in-the-computer

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821