Electronic Colloquium on Computational Complexity, Report No. 120 (2006)

Evolvability*

Leslie G. Valiant
Division of Engineering and Applied Sciences
Harvard University
valiant@deas.harvard.edu

Abstract

Living cells function according to complex mechanisms that operate in different ways depending on conditions. Evolutionary theory suggests that such mechanisms evolved as result of a random search guided by selection and realized by genetic mutations. However, as some observers have noted, there has existed no theory that would explain quantitatively which mechanisms can so evolve, and which are too complex. In this paper we provide such a theory. As a starting point, we relate evolution to the theory of computational learning, which addresses the question of the complexity of recognition mechanisms that can be derived from examples. We derive a quantitative notion of evolvability. The notion quantifies the limited sizes of populations and the limited numbers of generations that need to be sufficient for the evolution of significant classes of mechanisms. It is shown that in any one phase of evolution where selection is for one beneficial behavior, certain specific classes of mechanisms are demonstrably evolvable, while certain others are demonstrably not.

1 Introduction

We address the problem of quantifying how complex mechanisms, such as those found in living cells, can evolve into existence without any need for unlikely events to occur. If evolution merely performed a random search it would require exponential time, much too long to explain the complexity of existing biological structures. Darwin made this observation eloquently in the context of the evolution of the eye and suggested selection as the critical controlling principle. He called the supposition that the eye could evolve " $\cdots$ absurd in the highest possible degree" were it not for the fact that eyes "vary ever so slightly" and might therefore evolve over time by selection [1].

This paper describes a quantitative theory of the possibilities and limitations of what selection can achieve in speeding up the process of acquiring complex mechanisms beyond mere exhaustive search. In particular, we show that, in a defined sense, selection for a given beneficial behavior can provably support the evolution of certain specific classes of mechanisms, and provably not support that of certain other classes.

*This work was supported by grants NSF-CCR-03-10882, NSF-CCF-04-32037 and NSF-CCF-04-27129.

ISSN 1433-8092

We approach this problem by viewing mechanisms from the quantitative viewpoint of the mathematical functions they realize. In particular the subject matter of the field of computational learning theory [2-5] can be viewed as that of delineating limits on the complexity of mechanisms that can be acquired with feasible resources without an explicit designer or programmer. A primary instance studied there is the acquisition of a recognition algorithm for a function given just positive and negative examples of it. The quantitative study of computational learning over the last two decades has shown that certain classes of recognition mechanisms can indeed be learned in a feasible amount of time, while others encounter apparently intractable computational impediments.

Our goal here is to give a quantitative theory of the evolution of mechanisms. What we formalize is concerned with four basic notions. First, since the biology of cells consists of thousands of proteins and operates with circuits with complex mechanisms, we seek mechanisms that can evaluate *many-argument functions*. This permits the behavior of circuits to vary in complex ways depending on the particular combination of values that a large number of input parameters take. Second, any particular mechanism or many-argument function has a measure of *performance* that is determined by the values of the function on inputs from a probability distribution over the conditions that arise. By applying the mechanism to a variety of conditions the organism will enjoy a cumulative expected benefit that is determined by this performance. Third, for any mechanism only a limited number of variants can be explored per generation, whether through mutations or recombination, since the organisms that can exist at any time have a *limited population*. Fourth, there is the requirement that mechanisms with significant improvements in performance evolve in a *limited number of generations*.

We show that our notion of evolvability is a restricted case of PAC learnability. This offers a unifying framework for the fields of evolution and cognition. The behavior of a biological organism is clearly affected both by the results of evolution and those of learning by the individual. Distinguishing between these two sources of behavior has proved problematic, and it will perhaps help to have a unifying viewpoint on them.

As far as the evolvability of specific classes of functions we have both positive and negative results. On the positive side we show that the classes of monotone Boolean conjunctions and disjunctions are evolvable over the uniform distribution of inputs for the most natural representation of these functions. On the negative side we show that the class of Boolean parity functions is not evolvable over the uniform distribution. Since the latter class is known to be learnable we can conclude that evolvability is more constrainted than learnability.

Our intention is to leave little doubt that functions in classes that are provably evolvable in the defined sense, do correspond to mechanisms that can logically evolve into existence over realistic time periods and within realistic populations, without any need for combinatorially unlikely events to occur. Previous quantitative theories of evolution had aims other than that of quantifying the complexity of the mechanisms that evolved. The major classical thrust has been the analysis of the dynamics of populations [6-9]. A more recent development is the study of evolutionary algorithms [10, 11], a field in which the goal is to develop good computer algorithms inspired by evolution, usually for optimization, but not necessarily those that model biological evolution. For example, these algorithms may choose mutations depending on both the current genome description and the current inputs or experiences, and may act on exponentially small increases in performance. In our model the candidate set of mutations is selected only on the basis of the current genome, and only statistically distinguishable improvements can have an influence. We note that the term evolvability has been used in a further different sense, that of measuring the intrinsic capacity of genomes to produce variants [12]

We also observe that while most discussions of evolution emphasize competition and survival rather than evolution towards targets, such discussions have not yielded quantified explanations of which mechanisms can evolve.

2 Many-Argument Functions

The structures or circuits that are the constituents of living cells have to respond appropriately to wide variations in the conditions. We shall represent the conditions as the values of a number of variables $x_1, \cdots, x_n$, each of which may correspond, for example, to the output of some previously existing circuit. The responses that are desirable for an organism under the various combinations of values of the variables $x_1, \cdots, x_n$ we view as the values of an *ideal* function $f(x_1, \cdots, x_n)$. This f will have a low value in circumstances when such a low value is the most beneficial, and a high value in those other circumstances when a high value is the most beneficial. It will be beneficial to evolve a circuit that behaves as closely as possible to this f. For simplicity we shall consider the variables x_i and the functions f to take just two possible values, a low value of -1, and a high value of +1.

We shall give here as illustrations two such functions. The first, the *parity* functions, will be shown in this paper not to be evolvable. The second, *conjunctions*, will be shown to be evolvable.

A parity function is odd or even. An odd (even) parity function f over $x_1, \cdots, x_n$ has value +1 iff an odd (even) number of the variables in a specified subset of the variables have value +1. The following table illustrates an odd parity function over $\{x_1, x_2, x_3, x_4\}$ that has value 1 if an odd number of $\{x_1, x_3, x_4\}$ have value +1. For example, it has value +1 if $x_1 = x_2 = x_3 = x_4 = +1$, and value -1 if $x_1 = x_2 = x_3 = +1$ and $x_4 = -1$.

	$x_3 = -1, x_4 = -1:$	$x_3 = -1, x_4 = +1:$	$x_3 = +1, x_4 = -1:$	$x_3 = +1, x_4 = +1:$
$x_1 = -1, x_2 = -1:$	-1	$+1$	$+1$	-1
$x_1 = -1, x_2 = +1:$	-1	$+1$	$+1$	-1
$x_1 = +1, x_2 = -1:$	$+1$	-1	-1	$+1$
$x_1 = +1, x_2 = +1:$	$+1$	-1	-1	$+1$

We shall show that for evolving arbitrary parity functions over n variables either the number of generations or the population size of the generations would need to be exponentially large in terms of n.

In contrast we shall also show that there are classes with similarly substantial structure that are evolvable in a strong sense. An example of such a class is that of conjunctions. An example of a conjunction over x_1, x_2, x_3, x_4 is the function that is true if and only if $x_1 = +1$, and $x_4 = -1$. We abbreviate this function as $x_1 x_4'$. This function is shown below.

	$x_3 = -1, x_4 = -1:$	$x_3 = -1, x_4 = +1:$	$x_3 = +1, x_4 = -1:$	$x_3 = +1, x_4 = +1:$
$x_1 = -1, x_2 = -1:$	-1	-1	-1	-1
$x_1 = -1, x_2 = +1:$	-1	-1	-1	-1
$x_1 = +1, x_2 = -1:$	$+1$	-1	$+1$	-1
$x_1 = +1, x_2 = +1:$	$+1$	-1	$+1$	-1

We denote by X_n the set of all 2^n combinations of values that the variables $x_1, \cdots, x_n$ can take. We define D_n to be a probability distribution over X_n that describes the relative frequency with which the various combinations of variable values for $x_1, \cdots, x_n$ occur in the context of the organism. Evolution algorithms that work for all distributions would be particularly compelling.

Definition 2.1 *The* performance *of function $r : X_n \to \{-1, 1\}$ with respect to ideal function $f : X_n \to \{-1, 1\}$ for probability distribution D_n over X_n is*

$$\mathrm{Perf}_f(r, D_n) = \sum_{x \in X_n} f(x) r(x) D_n(x).$$

The performance is simply a measure of the correlation between the ideal function f and a hypothesis r we have at hand. The value will always be a real number in the range [-1, 1]. It will have value 1 if f is identical to r on points with nonzero probability in D_n. It will have value -1 if there is perfect anti-correlation on these points. (In general one could have a further multiplicative weight function $w(x)$ in this definition that would weight different conditions differently.)

The interpretation is that every time the organism encounters a condition, in the form of a set of values x of the variables $x_1, \cdots, x_n$, it will undergo a benefit amounting to $+1$ if its circuit r agrees with the ideal f on that x or a penalty -1 if r disagrees with f. Over a sequence of life experiences (i.e. different points in X_n) the total of all the benefits and penalties will be accumulated. Organisms or groups for which this total is high will be selected preferentially to survive over organisms or groups with lower such totals.

An organism or group will be able to test the performance of a function r by sampling a limited set $Y \subseteq X_n$ of size *s(n)* of inputs or experiences.

Definition 2.2 *For a positive integer s, ideal function $f : X_n \to \{-1, 1\}$ and probability distribution D_n over X_n the* empirical performance *$Perf_f(r, D_n, s)$ of function $r : X_n \to \{-1, 1\}$ is a random variable that makes s selections independently with replacement according to D_n and for the multiset Y so obtained takes value*

$$\sum_{x \in Y} f(x)r(x).$$

In our basic definition of evolvability we insist that evolution be able to proceed from any starting point. Otherwise, if some initialization process is assumed, then proceeding to the reinitialized state from another state might incur a decrease in performance. We also insist that at any time any one of the available mutations that gives a measurable improvement in performance should be acceptable and that the very best is not needed.

The evolution algorithm for conjunctions that we describe in detail in Section 5 behaves as follows. The learning of the previously described target function $x_1x'_4$ will be achieved by an algorithm that maintains a conjunction of a number of literals, a literal being a variable x_i or its negation x'_i. Clearly the aim is to evolve the function $x_1x'_4$ which has performance $+1$ since it is identical with the target, and is the only conjunction with that performance. The mutations will consist of *adding or deleting a single literal* for the current conjunction, or *swapping one literal for another*. Since there are then about n^2 possible mutations at each step it is feasible to explore the space of all mutations with a population of (polynomial) size, namely n^2.

3 Definition of Evolvability

Given the existence of an ideal function f the question we ask is whether it is possible to evolve a circuit for a function r that closely approximates f. Roughly, we want to say that a class C of ideal functions f is evolvable if any f in the class C satisfies two conditions. (i) From any starting function r_0 the sequence of functions $r_0 \Rightarrow r_1 \Rightarrow r_2 \Rightarrow r_3 \Rightarrow \cdots$ will be such that r_i will follow from r_{i-1} as a result of a single step of mutation and selection in a moderate size population, and (ii) after a moderate number i of steps r_i will have a performance value significantly higher than the performance of r_0, so that it is detectable after a moderate number of experiences. The conditions will be sufficient for evolution to start from any initial starting function and progress towards f, predictably and inexorably.

While the ideal function may be viewed as an abstract mathematical function, the hypothesis r needs to be represented concretely in the organism and should be viewed as a *representation* of a function. We generally consider C to be a class of ideal functions and R a class of representations of functions from which the organism will choose an r to approximate the f from C. We shall assume throughout that the representation R is *polynomial evaluatable* in the sense that there is a polynomial $u(n)$ such that given the description of an r in R and an input x from X_n, the value of $r(x)$ can be computed in $u(n)$ steps. This reflects the one assumption we make about biology, that its processes can be simulated in polynomial time on a computer. For brevity, and where it introduces no confusion, we shall denote by r both the representation as well as the function that that representation computes. We denote by C_n, R_n, and D_n, the restrictions of C, R, and D to n variables, but sometimes omit these distinctions where the meaning is clear. Also, we shall denote by ε the error parameter of the evolution, which describes how close to optimality the performance of the evolved representation has to be. We shall be prepared to expend resources that are polynomial in n, the number of arguments of the ideal function, and also in $1/\varepsilon$. Hence our resource bounds will be polynomial functions, such the as $p(n, 1/\varepsilon)$ in the following definition.

Definition 3.1 *For a polynomial $p(n, 1/\varepsilon)$ and a representation class R a $p(n, 1/\varepsilon)$-*neighborhood *N on R is a pair M_1, M_2 of randomized polynomial time Turing machines such that on input a number n in unary and representation $r \in R_n$ act as follows: M_1 first outputs the members of a set $Neigh_N(r) \subseteq R_n$, that contains r and has size at most $p(n, 1/\varepsilon)$. Subsequently M_2 is run on this output of M_1, and outputs each member r_1 of $Neigh_N(r)$ with a probability $Pr_N(r, r_1) \geq 1/p(n, 1/\varepsilon)$.*

The interpretation here is that for each genome the number of variants that can be searched effectively is not unlimited, because the population at any time is not unlimited, but is polynomial bounded. One possible implementation, clearly, is to regard R as the set of possible genomes, restrict mutations to a fixed constant number of base pairs, and regard the genome length as a polynomial in the relevant n. We consider exponentially many such variants to be impractical, while modest polynomial bounds such as n or n^2 are feasible. As in other areas of algorithmic analysis it turns out that natural polynomially bounded processes usually have reasonably modest polynomial bounds, and hence such results are meaningful [13-14]. The theory, as presented here, aims to distinguish between polynomial and exponential resources, insisting as it does that population sizes, numbers of generations, and numbers of computational steps all have to be upper bounded by a polynomial in the number of variables on which a circuit depends, and in the inverse of the error. Clearly, using more careful analysis finer distinctions can also be made.

Note that in our model, for each r process M_1 is run once and process M_2 run possibly many times. This ensures that from among a polynomial number of mutations, each one can be produced many times so that its possible superiority can be reliably determined. In biology this may be implemented either by the same mutation occurring in many individuals, or by the one mutation being spread by reproduction. Estimates of actual mutation rates in various organisms are available [15-17].

Definition 3.2 *For positive integers n and s, an ideal function $f \in C_n$, a representation class R with $p(n, 1/\varepsilon)$-neighborhood N on R, a distribution D_n, a representation $r \in R_n$ and a real number t, the mutator $Mu(f, p(n, 1/\varepsilon), R, N, D_n, s, r, t)$ is a random variable that on input $r \in R_n$ takes a value $r_1 \in R_n$ determinded as follows. For each $r_1 \in Neigh_N(r)$ it first computes an empirical value of $v(r_1) = Perf_f(r, D_n, s)$. Let $Bene$ be the set $\{r_1 \mid v(r_1) \geq (1+t)v(r)\}$ and $Neut$ be the set difference $\{r_1 \mid v(r_1) \geq (1-t)v(r)\}$ – $Bene$. Then*

(i) if $Bene \neq \phi$ then output $r_1 \in Bene$ with probability

$$Pr_N(r, r_1) / \sum_{r_1 \in Bene} Pr_N(r, r_1)$$

(ii) if $Bene = \phi$ but $Neut \neq \phi$ then output an $r_1 \in Neut$, the probability of a specific r_1 being

$$Pr_N(r, r_1) / \sum_{r_1 \in Neut} Pr_N(r, r_1)$$

(iii) if $Bene = Neut = \phi$ then output an arbitrary member $r_1 \in Neigh_N(r)$ with probability $Pr_N(r, r_1)$.

In this definition a distinction is made between beneficial and neutral mutations as revealed by a set of s experiments. In the former the empirical performance after the mutation exceeds that of the current representation r by an additive tolerance of at least t, a quantity which will, in general, be large enough, in particular some inverse polynomial, that it can be reliably distinguished from a zero increase in performance. In neutral mutations no increase in performance is expected, but it is expected that the performance is not worse than that of the current r by more than t. If some beneficial mutations are available one is chosen according to the relative probabilities of their generation by N. If none is available then one of the neutral mutations is taken according to the relative probabilities of *their* generation by N. If all mutations have empirical performance less than that of r by more than t then an arbitrary mutation is made. In practice by including the current representation r in $Neigh_N(r)$ and having s a large enough polynomial in comparison with the inverse of t, we can ensure that the latter possibility occurs with probability exponentially small in terms of n.

Definition 3.3 *For a mutator Mu(f, $p(n, 1/\varepsilon)$, R, N, D_n, s, r, t) a* t-evolution step *on input $r_1 \in R_n$ is the random variable Mu(f, $p(n, 1/\varepsilon)$, R,N, D_n, s, r_1, t). If r_2 is so generated we say $r_1 \rightarrow r_2$ or $r_2 \leftarrow$ Evolve(f, $p(n, 1/\varepsilon)$, R, N, D_n, s, r_1, t).*

Definition 3.4 *For a mutator Mu(f, $p(n, 1/\varepsilon)$, R, N, D_n, s, r, t) a* t-evolution sequence *for $r_1 \in R_n$ is a random variable that takes as values sequences $r_1, r_2, r_3, \cdots$ such that for all i $r_i \leftarrow Evolve(f, p(n, 1/\varepsilon), R, N, D_n, s, r_{i-1}, t)$.*

We shall find that if we want to evolve to performance very close to one, say $1 - \varepsilon$, we shall need numbers of experiments s or numbers of generations g that grow inversely with ε, and the tolerances t to diminish with ε. We therefore now regard these as functions of n and ε, and denote them by $s(n, 1/\varepsilon), g(n, 1/\varepsilon)$ and $t(1/n, \varepsilon)$.

Definition 3.5 *For polynomials $p(n, 1/\varepsilon)$, s(n, 1/ε) and t(1/n, ε), a representation class R_n, and $p(n, 1/\varepsilon)$-neighborhood N on R, the class C is t(1/n, ε)-*evolvable by *(p(n, 1/ε), R, N, s(n, 1/ε))* over distribution *D if there is a polynomial g(n, 1/ε) such that for every positive integer n, every $f \in C_n$, every $\varepsilon > 0$, and every $r_0 \in R_n$ it is the case that with probability greater than 1 - ε, a t(n, 1/ε)-evolution sequence $r_0, r_1, r_2, \cdots$, where $r_i \leftarrow Evolve(f, p(n, 1/\varepsilon), R, N, D_n, s(n, 1/\varepsilon), r_{i-1}, t(n, 1/\varepsilon))$, will have $Perf_f(r_{g(n,1/\varepsilon)}, D_n) > 1 - \varepsilon$.*

The polynomial $g(n, 1/\varepsilon)$, the generation polynomial, upper bounds the number of generations needed for the evolution process.

Definition 3.6 *A class C is* evolvable by $(p(n, 1/\varepsilon),\ R,\ N,\ s(n,\ 1/\varepsilon))$ over D *iff for some polynomial* $t(1/n,\ \varepsilon)$*, C is* $t(1/n,\ \varepsilon)$*-evolvable by* $(p(n, 1/\varepsilon),\ R,\ N,\ s(n,\ 1/\varepsilon))$ *over D.*

Definition 3.7 *A class C is* evolvable by R over D *iff for some polynomials* $p(n, 1/\varepsilon)$ *and* $s(n,\ 1/\varepsilon)$*, and some* $p(n, 1/\varepsilon)$*-neighborhood N on R, C is evolvable by* $(p(n, 1/\varepsilon),\ R,\ N,\ s(n,\ 1/\varepsilon))$ *over D.*

Definition 3.8 *A class C is evolvable over D if for some R it is evolvable by R over D.*

Definition 3.9 *A class C is* evolvable *if it is evolvable over all D.*

Our definition of evolvability is closely related to that of learnability [2-5], but it includes the extra ingredients that each step of learning (i) chooses from a polynomial size set of hypotheses, (ii) improves a performance measure monotonically, and further (iii) the progress of the algorithm depends on the candidate hypotheses only through their performance on inputs, and not through their syntax. Note that by allowing neutral mutations we do not insist on the performance improving strictly monotonically at each step.

Proposition 3.1 *If C is evolvable by R over D then C is learnable by R over D. In particular, if C is evolvable by R then C is learnable by R.*

Proof If C is evolvable over D then, by definition, for some polynomials $p(n, 1/\varepsilon), s(n, 1/\varepsilon), g(n, 1/\varepsilon)$ and $t(1/n, \varepsilon)$, some polynomial evaluatable representation R and some $p(n, 1/\varepsilon)$-neighborhood N on R, C is $t(1/n, \varepsilon)$-evolvable by $(p(n, 1/\varepsilon), R, N, s(n, 1/\varepsilon))$ over distribution D with generation polynomial $g(n, 1/\varepsilon)$. The main observation here is that we can replicate this evolution algorithm exactly in terms of the PAC learning framework. At each stage the evolution algorithm takes fixed size samples of $s(n, 1/\varepsilon)$ labelled examples from the distribution, computes for its current hypothesis the empirical performance, and from that generates the next hypothesis in a polynomially bounded fashion. But computing this performance is equivalent to computing the fraction of examples on which the hypothesis predicts correctly. Hence the access required to examples is that of random labelled examples from D, and every step is a polynomial time computational process. All this is permitted within the PAC model. Also the final hypothesis of the evolution model satisfies the requirements of the learning model since it ensures that the performance is at least $1-\varepsilon$, and hence accurate on at least $1 - \varepsilon/2$ of D. ■

We can strengthen the above statement by observing that evolvability implies learnability in the more restricted sense of statistical queries defined by Kearns [18]. In that model oracles provide not individual examples but estimates, to within inverse polynomial additive error, of the fraction of examples that satisfy polynomially evaluatable properties. This is clearly sufficient for the proof of Proposition 3.1, which therefore supports also the following.

Proposition 3.2 *If C is evolvable by R over D then it is efficiently learnable from statistical queries using R over D.*

Evolvability for all D is a very strong and desirable notion. As mentioned previously, it guarantees evolution independent of any assumptions about the distribution. It also means that evolution can continue even if the D changes. Of course, a change in D can cause a reduction in the value of Perf for any one r, and hence may set back the progress of evolution. However, the process of finding improvements with respect to whatever the current D is will continue. It remains an open problem as to whether such distribution-free evolution is possible for a significant class of functions.

Note also that the representation class R may represent a class of functions that differs from C. For example an R richer than C may be helpful. Alternatively, a weaker class may still produce good enough approximations and may have better properties for evolution. In general if we wish to identify or emphasize the class R that supports an evolution algorithm we say that *C is evolvable by R for D*, or *C is evolvable by R*.

One can make variations on the definition of evolvability. One restriction is *evolvability with initialization.* In that case in Definition 3.5, instead of requiring convergence from any starting point $r_0 \in R_n$, we require only that there is convergence from one fixed starting point r_0. The more general definition given is more robust, allowing for successive phases of evolution, each phase responding to new conditions. The evolved representation for one phase can then serve as the starting representation for the next, without a decrease in performance at any step. In evolution with initialization, the steps of going from the end of one phase to a reinitialized new state may suffer a performance decrease.

Another variant is *evolvability with optimization.* Here we insist that in Definition 3.2 the representation r_1 selected is any one with empirical performance within tolerance t of the best empirical performance in $\text{Neigh}_N(r)$. However, it is easy to see that this variant is no more powerful than the main definition. One can simulate the search for the best representation as required by one step of the optimized evolution, in no more than $6/t$ basic steps of looking for a representation with an empirical improvement of at least $t/2$, each step using a new sample. (Note that the actual performance can increase cumulatively by at most 2. Using the Hoeffding Bound (Fact 4.2) one can show that the cumulative empirical performance increase on the different samples can be limited to 3 with overwhelming probability.) For this simulation we change the representation to $i \cdot r^M \cdot r^P$ where $i \leq 6/t$ is an integer denoting which basic step we are in, $r^M \in R$ is the representation that generates the mutations for each of the up to $6/t$ basic steps, and $r^P \in R$ is the one with best performance found so far. (In other words r^P is the function this representation is computing, but the representation also has a memory of r^M from which it can generate new mutations in R, that may not be generatable from r^P alone.) After $i = 6/t$ basic steps the final r^P is adopted as the starting r^M and r^P of the next step of the optimized evolution. Note that the constructed representation in this reduction is a *redundant* representation in the sense that there are many representations that correspond to the same function r^P.

Proposition 3.3 *If C is evolvable with optimization by R over D, then C is evolvable by*

R over D. If C is evolvable with initialization and optimization by R over D, then C is evolvable with initialization by R over D.

4 Limits to Evolvability

The obvious question arises as to whether the converse of Proposition 3.1 holds: does learnability imply evolvability? Our next result answers this in the negative, saying as it does that for a certain function class there is a distribution that defeats all combinations of representations and neighborhoods.

We define Lin_n to be the set of odd parity functions $f(x_1, \cdots, x_n)$ over $\{-1, 1\}^n$. Each such f corresponds to some subset of the variables $x_{i[1]}, \cdots, x_{i[k]} \in \{x_1, \cdots, x_n\}$. The function f has value 1 if and only if an odd number of the variables $\{x_{i[1]}, \cdots, x_{i[k]}\}$ have value 1. (Equivalently f specifies a linear subspace of over GF[2]). Clearly there are 2^n functions in Lin_n. We define U to be the uniform distribution over $\{-1, 1\}^n$. We note that the functions in Lin are easy to compute, and further the class is known to be learnable not only for U but for all distributions [18, 19].

Proposition 4.1 *Lin is not evolvable for U by any representation R.*

Proof Kearns [20] shows that Lin is not efficiently learnable from statistical queries over U using any representation. The result then follows from Proposition 3.1 above. ■

Another important class of functions that is known to be learnable is that of linear halfspaces $\{\mathbf{a}.\mathbf{x} \geq b \mid \mathbf{a} \in \mathbb{R}^n, b \in \mathbb{R}\}$ in n-dimensional space $\mathbb{R}^n$. This class is learnable for all distributions by the natural representation of linear halfspaces, even if the coefficients $\mathbf{a}$, b are represented as rational numbers with n digits of accuracy, by virtue of the existence of polynomial time algorithms for linear programming [4]. However, if both the class and its representation is restricted to {0,1} coefficients then we have the following.

Proposition 4.2 *If C is the class of Boolean Threshold Functions* $\{\mathbf{a}.\mathbf{x} \leq b \mid \mathbf{a} \in \{0, 1\}^n, b \in \mathbb{R}\}$ *in* $\mathbb{R}^n$ *and R is the given representation of it, then C is not evolvable by R, unless NP = RP.*

Proof In [3] it is shown that this class is not learnable by its natural representation unless NP = RP. (The proof there shows that an NP-complete problem, integer programming, can be mapped to instances of learning Boolean threshold functions for a certain distribution to accuracy better than $1/n$.) The result follows from Proposition 3.1 above. ■

Note that the classes considered in the previous two propositions may appear to be biologically unnatural. This is exactly the prediction of the theory, which asserts that such structures cannot evolve. The existence of such structures in biology would challenge the validity of our approach to evolvability.

There appear to be at least four impediments that can be identified to evolvability in our sense, the first three of which derive from general impediments to learnability, while the last is particular to evolvability: (i) A purely information theoretic impediment [21]: the complexity of the mechanism that is to evolve exceeds the number of experiments. (ii) A representational limit such as Proposition 4.2 above, where learnability by a fixed representation would imply solving a computational problem that is believed to be hard. (iii) An intrinsic complexity limitation [22]: the function class is so extensive that learning it by *any* representation would imply an efficient algorithm for a problem believed to be hard to compute. (iv) Limits such as Proposition 4.1 above, that show that for information theoretic reasons evolvability cannot proceed because no empirical test of a polynomial number of hypotheses in a neighborhood can guarantee sufficient convergence in performance. Note that impediments (i) and (iv) are absolute, requiring no unproven computational assumptions.

5 Some Provably Evolvable Structures

We now describe some basic classes of Boolean functions and distributions that are provably evolvable. Here disjunction or Boolean "or" is denoted by +, conjunction or Boolean "and" by the multiplication sign, and Boolean negation of a variable x_i by x'_i. In general we shall have n variables $x_1, \cdots, x_n$. A *q-disjunction* is a disjunction of $k \leq q$ of the n variables or their negations, while a *q-conjunction* is a conjunction of $k \leq q$ of the n variables or their negations. Thus a q-disjunction is $y_{i[1]} + \cdots + y_{i[k]}$ where $1 \leq i[1], \cdots, i[k] \leq n$ and $y_{i[j]} \in \{x_1, \cdots, x_n, x'_1, \cdots, x'_n\}$, and a q-conjunction as $y_{i[1]} \cdots y_{i[k]}$. The uniform distribution over $\{-1, +1\}$ will be denoted again by U. A conjunction or disjunction is *monotone* if it contains no negated literals. We note that Ros (Section B2.8 in [10]; [23]) has analyzed evolutionary algorithms for learning conjunctions and disjunctions. However, a step of his algorithm is allowed to depend not just on the value of his current hypothesis on an input, but on more detailed information such as the number of bits on which the hypothesis and input differ. Such computation on the input condition and the hypothesis description we consider unrealistic for evolution, and is outside our model.

Fact 5.1 *Over the uniform distribution U for any conjunction* $Pr_U(y_{i[1]} \cdots y_{i[k]} = 1) = 2^{-k}$ *and for any disjunction* $Pr_U(y_{i[1]} + \cdots + y_{i[k]} = 1) = 1 - 2^{-k}$.

For our probabilistic arguments below it will be sufficient to appeal to the following:

Fact 5.2 *(Hoeffding [24]). The probability that the mean of s independent random variables each taking values in the range [a, b] is greater than or less than the mean of their expectations by more than δ is at most* $exp(-2s\delta^2/(b-a)^2)$, *where* $exp(x)$ *denotes* e^x.

Fact 5.3 *(Coupon Collector's Problem) Suppose that there is a bucket of n balls and M is a subset of m of them. Then after* $j = CC(n,\ m,\ \eta) = n(\log_e m + \log_e(1/\eta))$ *samples with replacement the probability that some member of the chosen set M has been missed is less than* η.

Proof This probability is upper bounded by $m(1-1/n)^j < m(1-1/n)^{jn/n} = me^{-j/n} < \eta$.

Theorem 5.1 *Monotone conjunctions and disjunctions are evolvable over the uniform distribution for their natural representations.*

Proof We first note that for our definition of evolvability it is sometimes advantageous for a local search procedure to introduce literals that do not appear in the ideal function f. For example, suppose $f = x_1x_2x_3$ and we start with a hypothesis 1, the conjunction of zero literals. Then the hypothesis will disagree with f on 7/8 of the distribution, and the introduction of the literal x_4 will be an improvement, reducing this probability from 7/8 to 1/2. Similarly, it is sometimes advantageous to remove such a non-ideal literal.

If we are evolving to accuracy ε with a $p(n, 1/\varepsilon)$-neighborhood N we let $q = \lceil \log_2(dn/\varepsilon) \rceil$ for some constant d to be determined later. We choose the effective representation class R to be montone q-conjunctions. In case the initial conjunction is of length greater then q we allow for a preliminary evolution phase in the following way. We define the neighborhood of each such long conjunction to be all the conjunctions obtained by removing one or none of its literals, each one being generated with equal probability. Clearly the removal from the hypothesis of a literal not in the true monomial will be always at least neutral and hence will be always available as a mutation. After this process runs its course, which will take $O(n)$ stages except with exponentially small probability, either a short conjunction of length at most q will be reached, or there will be left a long conjunction of length at least $q+1$ consisting of literals all from f. The latter will differ from f with probability less than $2^{-q-1} \leq \varepsilon/(2dn)$, and hence will have performance at least $1-\varepsilon$ if $d \geq 1$.

For the main phase of the evolution it is therefore sufficient to consider that the current hypothesis $r \in R$ consists of the conjunction of the set V of at most q literals. We denote by r^+ and r^- the sets of conjunctions containing the literals in V with one literal added, and with one taken away, respectively. In the case that V has the maximum number q of literals then r^+ is empty. In the case that $|V| = 0, r^-$ is empty. Also we define r^{+-} to be the conjunctions consisting of the literals in V with one further literal added and then one literal taken away. Clearly $r \in r^{+-}$. We then choose the neighborhood structure N to be such that

$$Neigh_N(r) = r^+ \cup r^- \cup r^{+-}.$$

Finally r and the members of r^+ and r^- will each have equal probabilities, so that their total is 1/2, while the remaining members of r^{+-} will also have equal probabilities, again totaling 1/2. Clearly N is a $p(n, 1/\varepsilon)$-neighborhood structure where $p(n, 1/\varepsilon) = O(n^2)$.

The construction will ensure that every mutation in N either causes an improvement in performance of at least 2^{-2q} or causes no improvement (and a possible degradation.) We choose the tolerance $t(1/n, \varepsilon) = 2^{-2q-1}$ and the number of samples $s(n, 1/\varepsilon) = t^{-3}$. It will then follow that the empirical test will, except with exponentially small probability, correctly identify an available mutation that has true improvement $2t$, distinguishing it from one that gives no improvement in performance. This can be seen by substituting $a = -1, b = 1, \delta = t$ and $s = \delta^{-3}$ in the Hoeffding Bound above to obtain that the probability of s trials each with expected improvement 2δ will produce a mean improvement of less than $t = \delta$ is at most $\exp(-2s\delta^2/(b-a)^2) = \exp(-(dn/\varepsilon)^2)$. A similar argument also shows that the same test will not mistakenly classify a mutation with no performance increase with one with an increase of $2t$. In the same way the same tolerance will distinguish a mutation with a nonnegative performance increase from one whose performance decreases by at least $2t$.

In a run of the evolution algorithm there will be $g(n, 1/\varepsilon)$ stages, and in each stage up to $p(n, 1/\varepsilon)$ mutations will be tested, where g and p are polynomials. We will want that in all $p(n, 1/\varepsilon)g(n, 1/\varepsilon)$ empirical tests the probability of even one failure to make such a distinction be less than $\varepsilon/2$. But $p(n, 1/\varepsilon)g(n, 1/\varepsilon)\exp(-(dn/\varepsilon)^2) < \varepsilon/2$ for all n, ε for a suitable constant d.

Suppose that W is the true set of h literals in the ideal conjunction, and that the current hypothesis r is the conjunction of a set of k literals V. We claim that:

Claim 5.1 *Suppose $k \leq q$. Then*
(a) If $k < q$ then adding to r any literal z in W - V will increase the performance of r by at least 2^{1-q}.
(b) Removing from r any literal z in $V \cap W$ will decrease the performance of r by at least 2^{1-q}.
(c) Adding to r a literal in W - V and removing from r a literal in V - W will increase the performance by at least 2^{-q-h}.
(d) Adding to r some literal not in W, and removing from r a literal in $V \cap W$ will decrease the performance of r by at least 2^{-q-h}.
(e) Adding to r a literal in W - V and removing from r a literal in $V \cap W$ will leave the performance unchanged, as will also adding to r a literal not in W, and removing one in V - W.
(f) If r contains all the literals in W, then removing a z in V - W will increase the performance of r by at least 2^{1-q}.
(g) If r contains all the literals in W then adding a z in V - W will decrease the performance of r by at least 2^{-q}.
(h) If $h > q$ then adding a z to an r of length at most q - 2 will increase performance by at least 2^{-q}, and removing a z from an r of length at most $q - 1$ will decrease performance by at least 2^{1-q}.

To verify the above eight claims suppose that r is $y_{i[1]} \cdots y_{i[k]}$.

(a) Consider a z in $W - V$. Then conjoining z to $y_{i[1]} \cdots y_{i[k]}$ will change the hypothesis from $+1$ to the value of -1 on the points satisfying $z' y_{i[1]} \cdots y_{i[k]}$. Clearly, the ideal function f takes value -1 at all these points since $z = -1$. These points will, by Fact 5.1, have probability $2^{-(k+1)} \geq 2^{-q}$. Hence, the performance will improve by at least twice this quantity, namely 2^{1-q}.

(b) Suppose that $z = y_{i[1]}$. Removing it from r will change the hypothesis from -1 to the value of $+1$ on the points satisfying $z' y_{i[2]} \cdots y_{i[k]}$. Clearly, the ideal function takes value -1 at all these points. These points will, by Fact 5.1, have probability $2^{-k} \geq 2^{-q}$ and hence the performance will degrade by at least twice this quantity.

(c) Suppose the added literal is z and the removed literal is $y_{i[1]}$. Then the hypothesis changes (i) from 1 to -1 on the points where $z' y_{i[1]} \cdots y_{i[k]} = 1$, and (ii) from -1 to +1 on the points such that $z y'_{i[1]} y_{i[2]} \cdots y_{i[k]} = 1$. Now (i) changes from incorrect to correct at all such points, and applies with probability $2^{-(k+1)}$. Also (ii) applies to a set of points with the same total probability $2^{-(k+1)}$, but the change on some of the points may be from correct to incorrect. To show that the net change caused by (i) and (ii) in combination is beneficial as claimed it is sufficient to observe that (ii) is nondetrimental on a sufficient subdomain. To see this we consider the literals Z in W that are missing from r but other than z, and suppose that there are m of these. Then on the domain of points $z y'_{i[1]} y_{i[2]} \cdots y_{i[k]} = 1$ that specifiy (ii) we note that on the fraction 2^{-m} of these the correct value of the ideal function is indeed 1. Hence the improvement due to (i) is not completely negated by the degradation due to (ii). The improvement in performance is therefore at least $2^{-m-k} \geq 2^{-h-q}$.

(d) Suppose the added literal is z and the removed literal is $y_{i[1]}$. Then the hypothesis changes (i) from 1 to -1 on the points where $z' y_{i[1]} \cdots y_{i[k]} = 1$, and (ii) from -1 to +1 on the points such that $z y'_{i[1]} y_{i[2]} \cdots y_{i[k]} = 1$. Now (ii) is an incorrect change at every point and applies with probability $2^{-(k+1)}$. Also (i) applies to a set of points with the same total probability $2^{-(k+1)}$. To show that the net change caused by (i) and (ii) in combination is detrimental to the claimed extent, it is sufficient to observe that (i) is detrimental on a sufficient subdomain. To see this we consider the literals Z in W that are missing from r but other than z, and suppose that there are m of these. Then on the domain of points $z' y_{i[1]} y_{i[2]} \cdots y_{i[k]} = 1$ that specifiy (i) we note that on the fraction 2^{-m} of these the correct value of the ideal function is indeed 1. Hence (i) suffers a degradation of performance on a fraction 2^{-m} of its domain, and hence the rest cannot fully compensate for the degradation caused in (ii). The combined decrease in performance is therefore at least $2^{-m-k} \geq 2^{-h-q}$.

(e) Suppose the added literal is z and the removed literal is $y_{i[1]}$. Then the hypothesis changes (i) from 1 to -1 on the points where $z' y_{i[1]} \cdots y_{i[k]} = 1$, and (ii) from -1 to +1 on the points such that $z y'_{i[1]} y_{i[2]} \cdots y_{i[k]} = 1$. Now (ii) is an incorrect change at every point and applies with probability $2^{-(k+1)}$. Also (i) applies to a set of points with the same total probability $2^{-(k+1)}$ but is a correct change at every point. The second part of the claim follows similarily. Again each of the two conditions holds with probability 2^{-k-1}. But now if there are m literals in W missing from r, then over each of the two conditions stated in (i) and (ii) function f is true on a fraction z^{-m}. Hence the effect of the two changes is again to cancel and keep the performance unchanged.

(f) Suppose that $z = y_{i[1]}$. Removing z from $y_{i[1]} \cdots y_{i[k]}$ will change the value of the hypothesis from -1 to +1 on the points satisfying $z' y_{i[2]} \cdots y_{i[k]}$. But all such points have true value +1 if r contains all the literals in W. Hence this gives an increase in performance by an inverse polynomial amount $2^{1-k} \geq 2^{1-q}$.

(g) Consider a z in $V - W$. Then conjoining z to $y_{i[1]} \cdots y_{i[k]}$ will change the hypothesis from $+1$ to -1 on the points satisfying $z' y_{i[1]} \cdots y_{i[k]}$. But all such points have true value $+1$ if r contains all the literals in W. Hence conjoining z will cause a decrease in performance by an inverse polynomial amount $2^{-k} \geq 2^{-q}$.

(h) If $h > q$ then the hypothesis equals -1 on a large fraction of at least $1 - 2^{-1-q}$ of the points. A conjunction of length $k \leq q - 2$ will equal -1 on $1 - 2^{-k} \leq 1 - 2^{1-q}$ points, and a conjunction of length $k + 1$ on $1 - 2^{-k-1} \leq 1 - 2^{1-q}$ of the points. Hence the fraction of points on which the -1 prediction will be made increases by $(1 - 2^{1-q}) - (1 - 2^{2-q}) \geq 2^{1-q}$ with the addition of a literal if $k \leq q - 2$, and decreases by at least 2^{1-q} with the removal of one, if $k \leq q - 1$. If $h > q$ then the corresponding increase/decrease in the fraction of points on which predictions are correct is at least 2^{-q}, since the fraction of predicted -1 points changes by twice this quanlity, and the true $+1$ points amount to at most a half this quantity.

To prove the proposition, first suppose that the number h of literals in the ideal function is no more than q. Then the intended evolution sequences will have two phases. First, from any starting point of length at most q the representation will increase the number of its literals that are in W by a sequence of steps as specified in Claims (a) and (c). Interspersed with these steps there may be other steps that cause similar inverse polynomial improvements, but add or remove non-ideal literals. Once the conjunction contains all the literals of the ideal conjunction, it enters into a second phase in which it contracts removing all the non-ideal literals via the steps of Claim(f).

The assertious of the above paragraph can be verified as follows. Claims (a) and (c) ensure that *as long as some ideal literal is missing from* r, beneficial mutations, here defined as those that increase performance by at least 2^{-2q} will be always available and will add a missing ideal literal. Further, Claims (b), (d) and (e) ensure that mutations that remove or exchange ideal literals will be deleterious, reducing the performance by at least 2^{-2q}, or neutral, and hence will not be executed. Some beneficial mutations that add or remove non-ideal literals may however occur. However, since each literal not already in the conjunction, will be generated by N with equal probability as a target for addition or swapping in, the Coupon Collectors model (Fact 5.3) can be applied. If the ideal conjunction contains m literals then after $\mathrm{CC}(n, m, \varepsilon/2) = O((n \log n + n \log(1/\varepsilon))$ generations all m will have been added or swapped in, except with probability $\varepsilon/2$.

Once r contains all the ideal literals then the only beneficial mutations are those that remove non-ideal literals (Claims (f)). Adding non-ideal literals (Claim(g)), exchanging a literal for an ideal literal (Claims (d),(e)), or exchanging for a non-ideal literal (Claim (e)) are all deleterious or neutral. Hence in this second phase after $\mathrm{O}(n)$ steps the ideal conjunction with perfect performance will be reached.

We conclude that in the case that the number of literals h in the ideal conjunction is at most q then the evolution will reach the correct hypothesis in the claimed number of stages, except with probability ε, which accounts for both the empirical tests being unrepresentative, as well as the evolution steps failing for other reasons.

Finally we consider the alternative case that the number h of literals in the ideal conjunction is more than q. Then in the evolution beneficial steps (h) that add literals will be always available until the hypothesis becomes of length $q-1$. Further, steps (h) that remove literals will be never taken since these are deleterious. Once length $q-1$ is achieved the literals in it may churn, but the performance will be at least $1-2(2^{1-q}+2^{-h}) = 1-5.2^{-q} = 1-5\varepsilon/(dn)$.

The result for conjunctions therefore follows with $g(n, 1/\varepsilon) = O(n\log(n/\varepsilon))$.

The claimed result for disjunctions follows by Boolean duality: Given an expression representing a Boolean function, by interchanging "and" and "or" operators and negating the inputs will yield the negation of the original function. ■

The algorithm as described above may be applied to conjunctions with negated variables, but will then fail sometimes. For example, if the starting configuration contains many literals that are negations of literals in the ideal function, then it may have high performance because it predicts -1 everywhere. However, it would appear difficult in that case to find an improved hypothesis by local search.

If initialization is allowed then the above results can be obtained much more easily, and then also allow negations.

Proposition 5.1 *Conjunctions and disjunctions are evolvable with initialization over the uniform distribution.*

The reader can verify this by considering conjunctions with initial representation 1. If the hypothesis is of length k and consists only of literals that occur in the true conjunction of length $h > k$, then adding a literal from the true conjunction will increase performance by 2^{-k}, while adding one not in the true conjunction will increase performance by $2^{-k} - 2^{1-h}$. If $h = k$ then adding a literal will decrease performance by at least 2^{-k}. Then if we let $q = \log_2(d/\varepsilon)$ for an appropriate constant d, choose tolerance 2^{-q-1}, have a neighborhood that either adds a literal or does nothing, and stop adding new literals if the conjunction reaches length q, then evolution with optimization will proceed through conjunctions of literals exclusively from W until performance at least $1-\varepsilon$ is reached. It follows that this evolution algorithm will work with optimization and initialization, and hence, by Proposition 3.3 it is evolvable with initialization alone, but for a redundant representation.

6 Discussion

We have introduced a framework for analyzing the possibilities of and limitations on the evolution of mechanisms. Our definition of evolvability has considerable robustness. It can be weakened in several ways, separately and in combination, to yield notions that impose less onerous requirements. First, one can entertain the definition for just one specific distribution as we did for our positive results in Section 5. The question whether significant classes are provably evolvable for all distributions is perhaps the most important question that our formulation raises. Second, the requirement of having the performance be able to approach arbitrarily close to the best possible can be relaxed. This permits processes where computations are feasible only for obtaining approximations to the best possible. Third, the starting point need not be allowed to be arbitrary. There may be a tradeoff between the robustness offered by allowing arbitrary starting points, and the complexity of the mechanisms that can evolve. Wider classes may be evolvable in any of these less onerous senses than in the most robust sense. We can equally study, in the opposite direction, the quantitative tradeoffs obtained by constraining the model more, by disallowing, for example, neutral mutations or redundant representations.

Our result that some structures, namely montone conjunctions and disjunctions are evolvable over the uniform distribution, we interpret as evidence that the evolution of significant algorithmic structure is a predictable and analyzable phenomenon. This interpretation is further supported by the observation that the theory, analogously to learning theory, analyzes only the *granularity* of the structure that can evolve in a single phase with a single ideal function. If multiple phases and ideal functions are allowed in succession then arbitrarily complex structures can evolve. For example, in response to various initial ideal functions some set of conjunctions and disjunctions may evolve first. At the next phase the outputs of these functions can be treated as additional basic variables, and a second layer of functionality can evolve on top of these in response to other ideal functions. This process can proceed for any number of phases, and build up circuits of arbitrary complexity, as long as each layer is on its own beneficial.

A unified theory for learning and evolution is of potential significance to the studies of cognition and of its emulation by machine. A major challenge in understanding cognition is that in biological systems the interplay between the knowledge that is learned through experience by an individual and the knowledge inherent in the genes, is complex, and it is difficult to distinguish between them. In attempts to construct computer systems for cognitive functions, for example for vision, this challenge is reflected in the difficulty of providing an effective split between the preprogrammed and the learning parts. The unification of learning and evolution suggests that cognitive systems can be viewed as pure learning systems. The knowledge and skills a biological organism possesses can be viewed as the accumulation of what has been learned by its ancestors over billions of years, and what it has learned from its individual experience since conception. Robust logic [25] is a mathematical framework based on learning that aims to encompass cognitive tasks beyond learning, particularly reasoning. The pragmatic difficulty of finding training data for systems to be built along such principles has been pointed out [26]. By acknowledging that the training data may also need to cover knowledge learned through evolution one is acknowledging what happens in existing cognitive systems, namely the biological ones. It is possible that learning is the only way of guaranteeing sufficient robustness in large-scale cognitive systems. In that case it would follow that the construction of cognitive systems with human level performance should be conceptualized as a learning task that encompasses knowledge acquired in biological systems through evolution as well as experience.

We have shown that with regard to the acquisition of complex mechanisms evolvability can be viewed as a restricted form of learnability. While evolvability may be technically the more constrained, it is not inherently more mysterious.

7 Acknowledgements

The author is grateful to Daniel Fisher and Martin Nowak for stimulating discussions and encouragement, and to Rocco Servedio for his helpful comments on an earlier draft of this paper. This work was supported by grants NSF-CCR-0310882, NSF-CCF-0432037 and NSF-CCF-0427129 from the National Science Foundation.

References

[1] C. Darwin, *On the origin of species by means of natural selection.* London: John Murray, 1859.

[2] L. G. Valiant, A theory of the learnable. *C. ACM* **27**:11 (1984) pp.1134-1142.

[3] L. Pitt, and L.G. Valiant. Computational limitations on learning from examples,*J. ACM*, **35**:4 (1988) 965-984.

[4] A. Blumer, A. Ehrenfeucht, D. Haussler, and M.K. Warmuth. Learnability and the Vapnik Chervonenkis dimension, *J. ACM* **36**:4 (1989) 929-965.

[5] M. Kearns and U. Vazirani. *An Introduction to Computational Learning Theory.* MIT Press, 1994.

[6] R. A. Fisher (1930) *The Genetical Theory of Natural Selection.* Oxford: Oxford University Press.

[7] S. Wright, *Evolution and the Genetics of Populations, A Treatise.* Chicago: University of Chicago Press (1968-78).

[8] D. A. Roff, *Evolutionary Quantitative Genetics.* New York: Chapman & Hall, 1997.

[9] R. Bürger, *The Mathematical Theory of Selection, Recombination, and Mutation.* Wiley, Chichester (2000).

[10] T. Bäck, D.B. Fogel and Z. Michalewicz, (eds.) *Handbook of Evolutionary Computation*, Oxford Univ. Press, Oxford, (1997).

[11] I. Wegener. Theoretical aspects of evolutionary algorithms. *Proc. 28th Int. Colloq. on Automata, Languages and Programming,* LNCS 2076, Springer-Verlag (2001) 64-78.

[12] G.P. Wagner and L. Altenberg. Complex adaptations and the evolution of evolvability. *Evolution*, **50**:3 (1996) 967-976.

[13] M. R. Garey and D. S. Johnson, *Computers and Intractability: a Guide to the Theory of NP-Completeness*, Freeman, San Francisco, 1979.

[14] C. H. Papadimitriou, *Computational Complexity*, Addison-Wesley, Reading, Mass. (1994).

[15] M. Kimura, Evolutionary rate at the molecular level, *Nature* **217** (1968) 624-626.

[16] S. Kumar and S. Subramanian, Mutation rates in mammalian genomes, *Proc. Nat. Acad. Sci.*, **99** (2002) 803-808.

[17] J.W. Drake, *et al.*, Rates of spontaneous mutation, *Genetics* **148** (1998) 1667-1686.

[18] P. Fischer and H.U. Simon. On learning ring-sum expressions. *SIAM J. Computing*, **21**(1):181-192, (1992).

[19] D. Helmbold, R. Sloan and M. K. Warmuth, Learning integer lattices. *SIAM J. Computing* **21**(2): 240-266, (1992).

[20] M. Kearns. Efficient noise tolerant learning from statistical queries. *J.ACM* **45**(6): 983-1006, (1998).

[21] A. Ehrenfeucht, D. Haussler, M. Kearns and L. G. Valiant, A general lower bound on the number of examples needed for learning. *Inf. and Computation.* **82**:2 (1989) 247-261.

[22] M. Kearns and L. G. Valiant, Cryptographic limitations on learning Boolean formulae. *J. ACM*, **41**:1 (1994) 67-95.

[23] J.P. Ros. Learning Boolean functions with genetic algorithms: A PAC analysis. In *Foundations of Genetic Algorithms* (L.D. Whitley, Ed.), Morgan Kaufmann, San Mateo, CA (1993), 257-275.

[24] W. Hoeffding, Probability inequalities for sums of bounded random variables. *J. Amer. Stat. Assoc.* **58**, 13 (1963).

[25] L.G. Valiant, Robust logics, *Artificial Intelligence Journal*, **117** (2000) 231-253.

[26] L.G. Valiant, Knowledge infusion. *Proc. 21st National Conference on Artificial Intelligence, AAAI06*, (2006) 1546-1551.