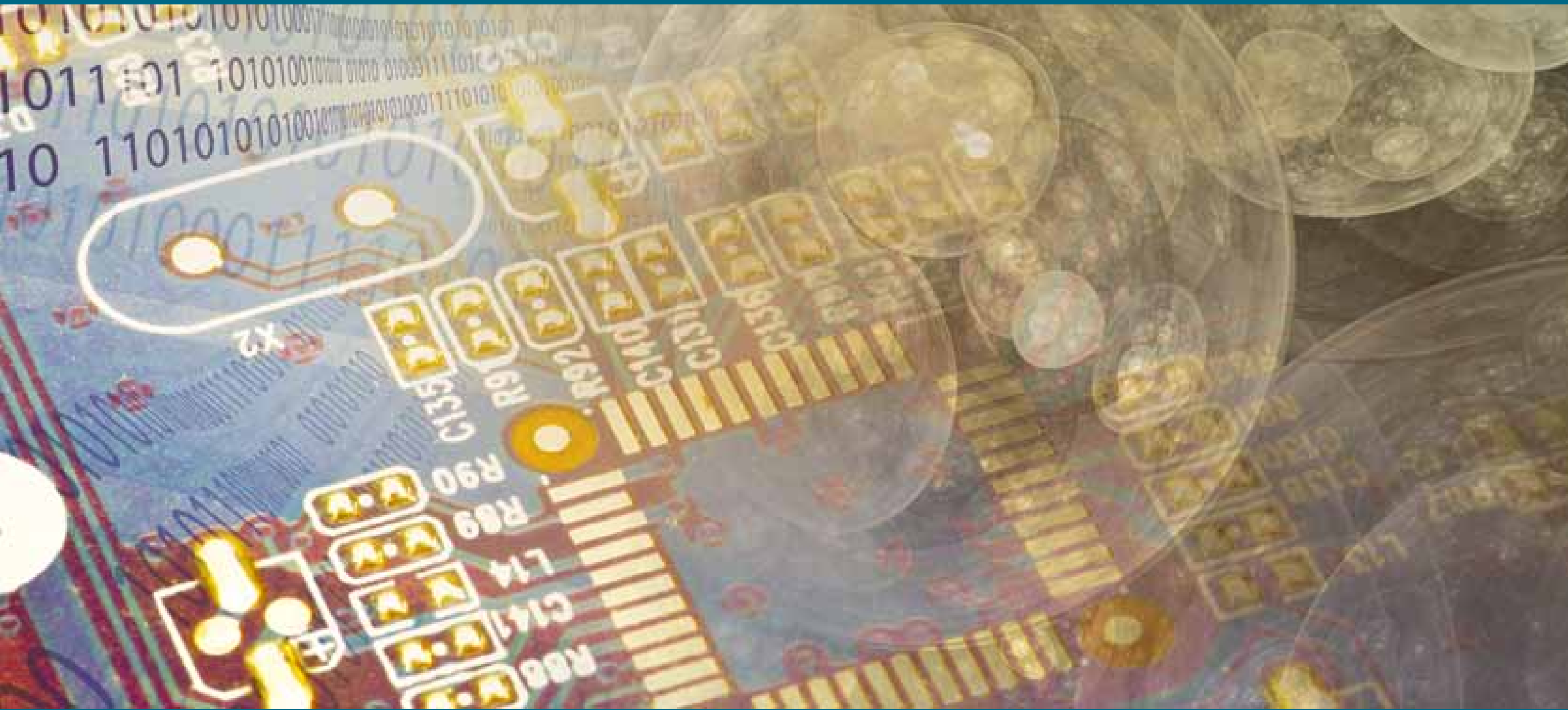
Journal of Artificial Evolution and Applications

Artificial Evolution Methods in the Biological and Biomedical Sciences

Guest Editors: Jason H. Moore, Janet Clegg, Elena Marchiori, Marylyn Ritchie, and Stephen Smith



Artificial Evolution Methods in the Biological and Biomedical Sciences

Journal of Artificial Evolution and Applications

Artificial Evolution Methods in the Biological and Biomedical Sciences

Guest Editors: Jason H. Moore, Janet Clegg,
Elena Marchiori, Marylyn Ritchie, and Stephen Smith

Editor-in-Chief

Advisory Editors

Associate Editors

Contents

Hindawi Publishing Corporation
Journal of Artificial Evolution and Applications
Volume 2009, Article ID 956380, 1 page
doi:10.1155/2009/956380

Editorial

Artificial Evolution Methods in the Biological and Biomedical Sciences

Jason H. Moore,[1] Janet Clegg,[2] Elena Marchiori,[3] Marylyn Ritchie,[4] and Stephen Smith[2]

[1] *Department of Genetics, Dartmouth College, Hanover, NH 03755, USA*
[2] *Department of Electronics, University of York, Heslington, York YO10 5DD, UK*
[3] *Faculty of Science, Radboud University, P. O. Box 9102, 6500 HC Nijmegen, The Netherlands*
[4] *Department of Molecular Physiology and Biophysics, Vanderbilt University, 2201 West End Avenue, Nashville, TN 37232, USA*

Correspondence should be addressed to Jason H. Moore, jason.h.moore@dartmouth.edu

Received 29 July 2009; Accepted 29 July 2009

Artificial evolution has emerged as a powerful computational strategy for solving complex problems in the biological and biomedical sciences. Inspired by biological evolution, artificial evolution is attractive because it employs stochastic search algorithms that are inherently parallel. As such, these novel approaches are ideal for biological and biomedical problems that are high-dimensional, noisy, and very complex. The papers that appear in this special issue of the Journal of Artificial Evolution and Applications were rigorously peer reviewed and represent a wide range of different algorithms and application areas. As such, the papers in this special issue represent both the depth and breadth of artificial evolution and its potential applications. Papers in this volume cover a range of artificial evolution algorithms including, for example, genetic algorithms and genetic programming. Application areas span multiple different areas including the classification of microarray data, classification and diagnosis of cancer, multiple sequence alignment, ecological simulation, population genetics, and neurological discrimination.

The application of artificial evolution to problems in medicine is particularly exciting. The use of evolutionary algorithms such as genetic programming is becoming more common and the paper "Classification of oncologic data with genetic programming" by Leonardo Vannesci et al. is a particularly good example in which the critical fitness function employed is based on a clinical evaluation, in this case the area under the receiver operating characteristic curve and a measure of correctly classified instances. Senhua Yen and Dipankar Dasgupta describe the development of a novel algorithm inspired by the immune system in their paper "Conserved self pattern recognition algorithm with novel detection strategy applied to breast cancer diagnosis." The aim is to aid breast cancer diagnosis by analysing the cytological characteristics of breast fine needle aspirates using this novel pattern recognition approach. Although the methods and algorithms employed are very different, these papers demonstrate the potential impact of applying artificial evolution to the whole spectrum of medical applications.

The biological and biomedical sciences may be the ideal application areas for artificial evolution given the complexity of the data and the complexity of the processes and patterns being studied. This special issue provides a sampling of the state of the art in the field and should provide many novel ideas for readers to try with their own problem-solving efforts.

Jason H. Moore
Janet Clegg
Elena Marchiori
Marylyn Ritchie
Stephen Smith

Hindawi Publishing Corporation
Journal of Artificial Evolution and Applications
Volume 2009, Article ID 725049, 13 pages
doi:10.1155/2009/725049

Research Article

Underdominance, Multiscale Interactions, and Self-Organizing Barriers to Gene Flow

Margaret J. Eppstein,[1] Joshua L. Payne,[1] and Charles J. Goodnight[2]

[1] *Department of Computer Science, University of Vermont, Burlington, VT 05405, USA*
[2] *Department of Biology, University of Vermont, Burlington, VT 05405, USA*

Correspondence should be addressed to Margaret J. Eppstein, maggie.eppstein@uvm.edu

Received 11 March 2009; Accepted 1 June 2009

Recommended by Stephen Smith

Understanding mechanisms for the evolution of barriers to gene flow within interbreeding populations continues to be a topic of great interest among evolutionary theorists. In this work, simulated evolving diploid populations illustrate how mild underdominance (heterozygote disadvantage) can be easily introduced at multiple loci in interbreeding populations through simultaneous or sequential mutational events at individual loci, by means of directional selection and simple forms of epistasis (non-linear gene-gene interactions). It is then shown how multiscale interactions (within-locus, between-locus, and between-individual) can cause interbreeding populations with multiple underdominant loci to self-organize into clusters of compatible genotypes, in some circumstances resulting in the emergence of reproductively isolated species. If external barriers to gene flow are also present, these can have a stabilizing effect on cluster boundaries and help to maintain underdominant polymorphisms, even when homozygotes have differential fitness. It is concluded that multiscale interactions can potentially help to maintain underdominant polymorphisms and may contribute to speciation events.

1. Introduction

Charles Darwin referred to speciation as the "mystery of mysteries" [1] and nearly 150 years later the mechanisms involved in speciation remain an important topic of debate in evolutionary biology (for recent reviews of this topic, see [2–8]). Historically, models of speciation have commonly invoked geographical isolation as a means for divergent evolution [9–11]. However, empirical evidence [12–14] suggests that speciation can also occur in the absence of geographical barriers to gene flow, and there has been a recent flurry of theoretical models providing support for these observations [15–24]. These models typically assume divergent evolution leading to speciation, subsequent to some form of premating reproductive isolating mechanism. For example, disruptive natural selection toward use of different parts of the available resource spectrum [17, 19] could alter the timing and/or location of mating events, resulting in two or more effectively reproductively isolated subpopulations that then continue to diverge, despite continuing to share the same geographic range. Similarly, assortative mating (due to sexual selection, e.g., where like prefers to mate with like) has also been proposed as a premating isolating mechanism [15, 18, 20], with several models employing a combination of these factors [15, 16, 21–24].

Spatially localized breeding interactions have been observed in a variety of both plant (e.g., [25–29]) and animal (e.g., [30–32]) populations, and the spatially explicit nature of these interactions has often been recognized as potentially important in speciation processes. Wright [33] derived statistical predictions that showed how spatially localized mating within interbreeding populations leads to nonadaptive differentiation in different parts of the population which are isolated from each other by distance. He felt that this process could be important for evolution within a species, but would only rarely represent first steps toward speciation itself. Subsequent spatially explicit individual-based models with localized mating have been employed to show how patches with distinct gene frequencies become quickly established and persist for many generations, even in the absence of selection [34, 35]. When selection is present, such evolving spatial self-organization of genotypes can help maintain high levels of genetic variation at multiple loci, when multiple genotypes have the same fitness [36]. Similarly, when male dispersal is dependent on mating success, theoretical models have demonstrated that populations will self-organize into groups of similar genotypes, promoting the evolution of assortative mating and thus facilitating the emergence of reproductively isolated groups [37]. Further, computational simulations have shown that environmental heterogeneity, such as the presence of gradual environmental gradients, can facilitate evolution of reproductive isolation [16]. Even with no assortative mating or environmental heterogeneity, simulated two-locus haploid populations experiencing disruptive selection (which are functionally equivalent to diploid populations with inviable heterozygotes, a.k.a. complete underdominance) will self-organize into two reproductively isolated species [38]. However, this occurs only if the hybrids are completely inviable, and such models have often been dismissed as unrealistic on the grounds that it is difficult to explain how the incompatible alleles became established in the same population in the first place.

If an ancestral diploid population is homozygous for a single allele at a given locus, it is difficult to envision how a new mutant allele with even mild underdominance (i.e., slight heterozygote disadvantage) could become established in the gene pool. In panmictic populations, the probability that an underdominant mutation becomes fixed decreases exponentially with both population size and the degree of underdominance [7], since this requires crossing a maladaptive valley in moving between fitness “peaks”. However, several possible mechanisms for the successful introduction of underdominance have been put forth. Conceivably, environmental changes could alter the fitness effects of previously fixed alleles so that they later become underdominant [38]. Alternatively, if there is strong disruptive selection toward different niches within the habitat, then a mutant hybrid may experience a transient fitness advantage by exploiting underutilized resources in a new niche, but then exhibit underdominance once the population stabilizes [39, 40]. Bateson [9], Dobzhansky [10], and Muller [11] proposed a means by which hybrid incompatibilities could evolve via epistatic (i.e., nonlinear gene-gene) interactions between mutations occurring at separate loci in allopatric (i.e., geographically isolated) populations, and Kondrashov [41] showed how this same process could also occur if mutations arise nearly simultaneously in different regions in a single interbreeding population where individuals have limited movement. Despite the theoretical difficulties regarding the introduction of underdominance, there is no question that natural populations do maintain a great deal of genetic variation, and there is ample empirical evidence of underdominance and even complete hybrid sterility [42–45]. For example, in a recent comprehensive genetic study in maize, direct evidence was found for several types of within locus nonadditivity, including allelic underdominance at multiple loci [46].

Epistasis has long been recognized as important in evolutionary processes [9], and our rapidly growing understanding of the complex interconnectedness of genetic [47–49] and metabolic [50] regulatory networks is spawning a new appreciation for the ubiquity of nonlinear gene-gene interactions [51, 52]. Empirical evidence suggests that epistasis may be an important factor leading to speciation [44] and some form of epistasis is a common assumption in theoretical models of speciation [9–11, 21]. Recent molecular evidence indicates that the distribution of genetic polymorphisms associated with complex diseases (i.e., diseases that are caused by epistatic interactions of many genetic polymorphisms) is not significantly different from the distribution of normal human variation (comprising apparently neutral polymorphisms) [53], indicating that some polymorphisms may be individually nearly neutral but become significantly deleterious only in certain combinations, or in response to certain environmental conditions.

In this paper, we use simulated diploid populations evolving on two-dimensional spatial grids to explore the specific question as to whether the cumulative effects of incomplete underdominance (mild heterozygote disadvantage) at multiple epistatically interacting loci can potentially drive speciation events, even in the absence of other premating isolation mechanisms, such as allopatry, environmental heterogeneity, or assortative mating. Two primary questions are tackled: (1) how can underdominant alleles become initially established in a single interbreeding population in a homogeneous environment?, (2) assuming that multiple mildly underdominant alleles exist in a population, can self-organization of genotypes result in a coalescence of mild incompatibilities such that two reproductively isolated species emerge?

2. Methods

2.1. Discrete Population Model. Populations of diploid individuals were modeled using two-dimensional stochastic cellular automata, wherein each lattice cell could be occupied by at most one individual at any discrete time step. Evolution was simulated in synchronous (nonoverlapping) generations. At each generation, each cell was repopulated by the offspring of two parents, stochastically selected using fitness proportionate selection from the parent population in the mating neighborhood centered on the cell. That is, the probability P_i of selecting parent i, from this neighborhood, was computed as

$$P_i = \frac{f_i}{\sum_{j=1}^{n} f_j}, \tag{1}$$

where j represents each of the n individuals in the mating neighborhood used for repopulating cell i, and f_i is the fitness of the ith individual. Individuals were not permitted to mate with themselves. For each pair of selected parents, a single offspring was produced to occupy cell i in the next generation. Genotypes of individuals comprised L bi-allelic loci, for L in the range 2 to 10, depending on the experiment in question. Loci were treated as unlinked, so parents donated alleles to their offspring via independent assortment (uniform recombination). If the offspring of selected parents was inviable ($f_i = 0$), then the cell was treated as empty for the subsequent generation. Reported experiments were conducted on a 100 × 100 cell lattice with nonperiodic boundary conditions (local neighborhoods were simply truncated at domain boundaries). Separate experimentation in our laboratory, not otherwise reported, showed that lattice size (at least for lattices of 100 × 100 or larger), boundary conditions (nonperiodic, periodic), selection mechanism (tournament, fitness proportionate), crossover strategy (uniform, single point), and asynchronous versus synchronous updates did not qualitatively affect the results, although the time scale of self-organizing events varied with these control parameters.

2.2. Interaction Topologies. Two different types of population structures were used for the determination of the individuals in the mating neighborhoods used in (1): (a) panmixia, wherein the mating neighborhood for each individual comprised the entire population, and (b) localized mating within overlapping 3 × 3 cell "Moore" neighborhoods centered on each cell i. Similar spatially localized interactions are variously referred to elsewhere by such phrases as nearest neighbor [54], isolation by distance [33, 34, 36], local neighborhoods [38], or absence of long-range interactions [41].

2.3. Fitness Models. Several different fitness models are discussed in this study, incorporating various types and degrees of additivity (linear within-locus fitness, where the heterozygote has fitness intermediate to that of the two homozygotes), dominance (nonlinear within-locus fitness), and epistatic (nonlinear between-locus fitness) interactions, including the two-locus fitness tables shown in Figure 1. In these fitness tables, a value of 0 means inviability, and positive values simply indicate relative fitnesses of the various genotypes. Before discussing the fitness functions used in our experiments, we briefly review two fitness functions used in related literature, for comparison.

2.3.1. Review of Fitness Models of Goldstein and Holsinger. Goldstein and Holsinger [36] employed two types of multilocus fitness functions in their study exploring the effects of self-organization in populations with localized mating, as exemplified by the two-locus fitness functions shown in Figures 1(a) and 1(b). These functions actually exhibit within-locus overdominance (i.e., the average fitness of each single-locus heterozygote (*Aa* or *Bb*) is higher than either single-locus homozygote (*AA*, *aa*, *BB*, or *bb*), for both loci), so polymorphism is maintained through selection, even though directional selection will favor homozygotes *aa* in combination with *BB*, or *bb* in combination with *AA*. Our study differs from theirs in that they were exploring self-organization under this form of *stabilizing* selection (due to heterozygote advantage), while we explore self-organization under *disruptive* selection (due to heterozygote disadvantage).

2.3.2. Review of Bateson, Dobzhansky, Muller Incompatibilities. Bateson [9], Dobzhansky [10], and Muller [11] proposed a mechanism for the introduction of hybrid incompatibilities between allopatric populations (commonly referred to as BDM incompatibilities). An example of a BDM type incompatibility is illustrated by the two-locus fitness table shown in Figure 1(c). If a common ancestral population includes only *AABB*, it is easy to see that mutations to *a* and *b* are each individually beneficial and so could each become fixed if they arise in allopatric populations, resulting in only *aaBB* in one population and *AAbb* in the other. The hybrid between these two populations *AaBb* is inviable, so speciation has occurred, even if the geographic barriers between the two populations are subsequently removed. This model is extendible to multiple loci with cumulative effects [56]. However, BDM incompatibilities require multiple allopatric mutations (or nearly simultaneous mutations in populations with localized mating [41]), and so cannot be used to explain the introduction of underdominance at individual bi-allelic loci. In this study, we demonstrate the introduction of *within-locus underdominance* at multiple loci in interbreeding populations with localized mating, from individual mutational events that may be simultaneous *or sequential.*

The remainder of the specific fitness models shown in Figure 1 are discussed in the next section in the context of the relevant experiments.

3. Experiments

3.1. Introducing within-Locus Underdominance

3.1.1. Additive by Dominance Epistasis. Goodnight [55] suggested, but did not demonstrate, that certain types of two-locus epistasis could result in the introduction of complete within-locus underdominance with a single mutation. For example, consider the fitness table for two loci shown in Figure 1(d) (which exhibits what Goodnight [55] refers to as pure "additive by dominance" epistasis). An ancestral population with only *A*, *B*, and *b* alleles will experience stabilizing selection, since the hybrid *AABb* genotype is the most fit, so both *B* and *b* alleles will be maintained in the population. As long as randomly interbreeding populations remain in Hardy-Weinberg equilibrium (where both alleles *B* and *b* have equal frequency, so the relative frequencies of diploid genotypes *BB* : *Bb* : *bb* are 1 : 2 : 1) a newly introduced *a* allele will be selectively neutral and could become fixed due to drift. However, any deviation away from a frequency of 0.5 at the *B* locus will result in directional selection for the *a* allele. In a panmictic population, either *aaBB* or *aabb* will take over the population, depending on which of *B* or *b* is most prevalent as a result of drift. However, in a population with localized mating, both *aaBB* and *aabb* can become established in different parts of the population, causing disruptive selection and subsequent self-organizing reproductive isolation (speciation) of these two genotypes. We have confirmed that such events can occur in simulated populations that were randomly initialized with spatially uncorrelated distributions of equal numbers of the *B* and *b* alleles but only a single randomly located *a* allele in a sea of *A* alleles, as illustrated by a representative

	BB	*Bb*	*bb*
AA	0.67	0.9	1
Aa	0.9	1	0.9
aa	1	0.9	0.67

(a)

	BB	*Bb*	*bb*
AA	0	0.75	1
Aa	0.75	1	0.75
aa	1	0.75	0

(b)

	BB	*Bb*	*bb*
AA	0.8	0.9	1
Aa	0.9	0	0.9
aa	1	0.9	0.8

(c)

	BB	*Bb*	*bb*
AA	0	1	0
Aa	0.5	0.5	0.5
aa	1	0	1

(d)

	BB	*Bb*	*bb*
AA	0.1	0.1	0.1
Aa	0.3	0.3	0.3
aa	0.6	0.5	0.6

(e)

	BB	*Bb*	*bb*
AA	0.1	0.1	0.1
Aa	0.5	0.5	0.5
aa	0.8	0.3	1

(f)

	BB	*Bb*	*bb*
AA	1	0.5	1
Aa	0.5	0	0.5
aa	1	0.5	1

(g)

	BB	*Bb*	*bb*
AA	1	0.5	0.92
Aa	0.5	0	0.5
aa	0.92	0.5	1

(h)

Figure 1: Sample fitness tables for various two-locus, bi-allelic genotypes referred to in this study, where fitness ranges from 0 (inviable) to 1 (most fit). (a) and (b) Two locus examples of the fitness functions described in Goldstein and Holsinger [36]; see text for details, (c) an example of one form of BDM incompatibilities, (d) "additive by dominance" epistasis [55], (e) an instantiation of the formulas in (2) with $\eta = 0.6$, $\alpha = 0.5$, $\beta = 0.3$, $\delta = 0.0$, and $\gamma = 0.1$, (f) an instantiation of the formulas in (2) with $\eta = 1.0$, $\alpha = 0.9$, $\beta = 0.5$, $\delta = 0.2$, and $\gamma = 0.5$, (g) an instantiation of fitnesses via (3) with $L = 2$ and $\varepsilon = 0.0$, and (h) an instantiation of fitnesses via (3) with $L = 2$ and $\varepsilon = 0.1$.

simulation depicted in Figure 2. If the mutant *a* allele is not lost due to early drift, it quickly begins to increase due to directional selection in its local environment, which almost inevitably leads to patches of reproductively isolated species with genotypes *aaBB* and *aabb*. In a 100 × 100 grid, such speciation events were observed in 30% of 20 trials when using 3 × 3 localized mating neighborhoods, whereas fixation to either *aabb* or *aaBB* occurred 100% of the time when mating was panmictic. This example is of interest because of the fact that complete underdominance at the *B* locus is the result of a single mutational event, resulting in speciation when mating is localized, despite the fact that all alleles and loci have identical average effects when the population is in Hardy-Weinberg equilibrium. However, the existence of perfect transient "additive by dominance" epistasis (Figure 1(d)) in natural populations is expected to be very rare, at best, given the very specific requirements that the *B* locus exhibit perfect underdominance, perfect neutrality, and perfect overdominance in *AA*, *Aa*, and *aa* backgrounds, respectively.

3.1.2. Directional Selection for within-Locus Underdominance at a Single Locus. It actually turns out to be quite simple to introduce within-locus underdominance into interbreeding populations via single mutational events and directional selection, when we allow for both additive and epistatic genetic variance. Consider a two-locus fitness table, where the (nonnegative) entries are calculated by the formulas below:

$$\begin{array}{c|c|c|c|} & BB & Bb & bb \\ \hline AA & \eta - \alpha & \eta - \alpha & \eta - \alpha \\ \hline Aa & \eta - \beta & \eta - \beta & \eta - \beta \\ \hline aa & \eta - \delta & \eta - \delta - \gamma & \eta \\ \hline \end{array} \qquad (2)$$

where $\eta > \alpha > \beta > \delta \geq 0$ and $\eta \geq \delta + \gamma$. This fitness table has the following properties: (i) polymorphisms at the *B* locus are neutral when in *AA* or *Aa* backgrounds, (ii) the maximum fitness is η, (iii) the predominant component of fitness variance at the *A* locus is additive (i.e., there is directional selection for the *a* allele, since $\eta - \alpha < \eta - \beta < \eta - \delta$), and (iv) there are varying degrees of epistatic additivity and underdominance at the *B* locus in combination with *aa*, depending on the values of δ (the degree of asymmetry in fitness between *aaBB* and *aabb*) and γ (the degree of underdominance of *aaBb* relative to *aaBB*), respectively. Two example instantiations of the fitness formulas in (2)

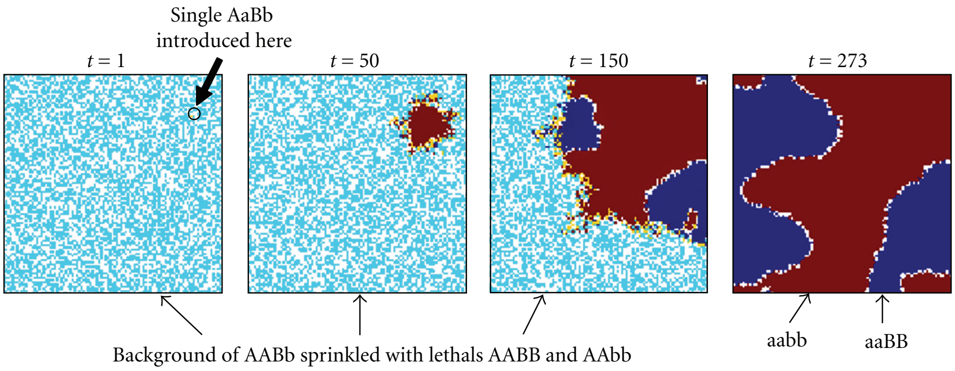


Figure 2: Self-organized speciation upon introduction of a single *a* allele (introduced location shown by small circle), with "additive by dominance" epistatic fitness as shown in Figure 1(d). Inviable genotypes (*AaBb*) are shown in white.

are shown in Figure 1(e) ($\delta = 0$, so there is no additive component to fitness at the *B* locus, so *bb* and *BB* have equal fitness) and Figure 1(f) ($\delta = 0.2$, so there is an additive component for the *B* locus, with *bb* being more fit than *BB*).

To illustrate how easily within-locus underdominance, even when asymmetric (i.e., $\delta > 0$), can be introduced, we ran the following experiments using $\eta = 1.0$, $\alpha = 0.9$, $\beta = 0.5$, $\gamma \in \{0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6\}$, and $\delta \in \{0, 0.1, 0.2, 0.3, 0.4\}$. Populations of 100 × 100 individuals were randomly initialized in spatially uncorrelated Hardy-Weinberg frequencies for the *B* and *b* alleles, but initially contained only the *A* allele at the *A* locus, so all genotypes in the ancestral population were of equal fitness. A single beneficial mutant *a* allele was then introduced into each population in a random location and the population was allowed to evolve until one of three events occurred: (i) the *a* allele was lost due to early drift (unsuccessful trial), (ii) only one genotype remained, usually *aaBB* but occasionally *aabb,* (unsuccessful trial), or (iii) the *A* allele was lost due to directional selection and both *B* and *b* remained, so that within-locus asymmetric underdominance has been established at the *B* locus (successful trial). The probability that the mutant *a* allele will become fixed is governed by $\alpha - \beta$ (i.e., how immediately beneficial the mutation is). However, once the *a* allele starts to increase in frequency, directional selection takes over and the *A* allele is soon lost, after which time either outcome (ii) or (iii) will occur.

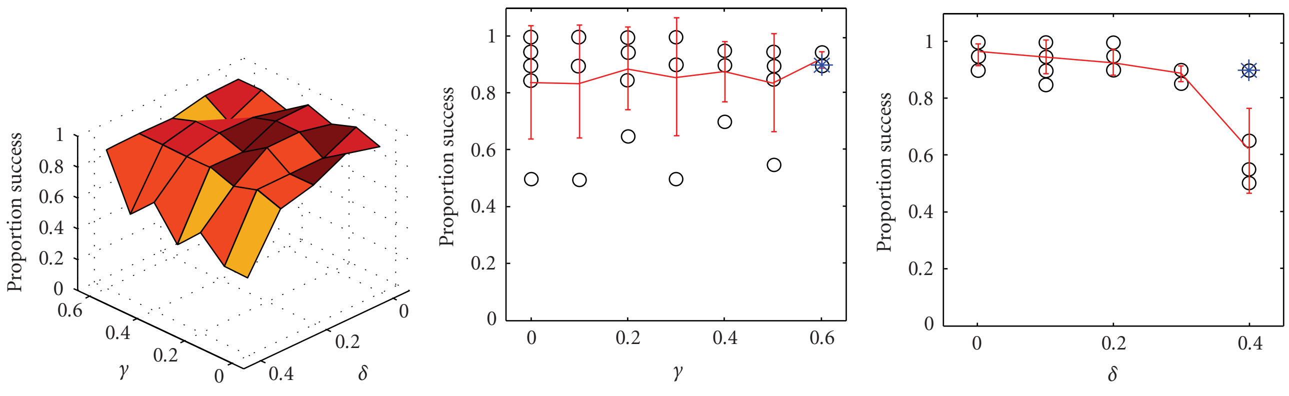
In Figure 3 we show how the proportion of successful introductions (out of 20 attempts at each parameter combination) of within-locus underdominance varies with 3 × 3 localized mating on a 100 × 100 grid, as a function of γ and δ, where the probability of fixation of the *a* allele is close to 1 (because $\alpha - \beta = 0.4$, so directional selection for *a* is strong). Somewhat surprisingly, the success of introduction of underdominance at the *B* locus is essentially independent of the degree of underdominance γ (Figure 3(b)). Indeed, even with complete underdominance (i.e., heterozygote inviability at $\eta - \delta - \gamma = 0$) shown by the asterisks in Figures 3(b) and 3(c), a single mutational event can cause speciation into reproductively isolated populations of *aaBB* and *aabb*. This is similar to the speciation event caused by the table in Figure 1(d) and shown in Figure 2, although in this case speciation can occur even when the *aaBB* genotype is less fit than the *aabb* genotype, because boundaries of inviable hybrids between clusters of these two genotypes act as barriers to gene flow that help to protect the less fit species. Thus, perfect genetic redundancy, where multiple homozygotes are equally fit, is not a strict requirement for self-organized speciation to occur. However, the frequency of successful trials is reduced as the asymmetry in fitness (δ) between *aaBB* and *aabb* is increased, because this increases the probability that the entire population converges on *aabb* (Figure 3(c)). In summary, *when mating is localized, even strong underdominance with mild asymmetry between homozygotes can be easily introduced into the population through a single mutational event*, when simple and biologically feasible forms of additivity and epistasis are considered. The fitness formulas in (2) are just one of many forms of epistatic fitness that can have this effect, as long as there is directional selection toward the newly introduced allele.

When mating is panmictic, weak underdominance and asymmetry can still be introduced in this manner, but the frequency of success is very sensitive to both γ and δ and unless these are both very weak the population rapidly converges to either *aaBB* or *aabb,* resulting in the failure to introduce within-locus underdominance at the *B* locus. This is shown by the results of an identical set of experiments to those described above, except where mating was panmictic (Figure 4). Even when underdominance is successfully introduced, it is not likely to persist for long in panmictic populations, as discussed later.

3.1.3. *Directional Selection for Introducing Underdominance at Multiple Loci.*

The method described in Section 3.1.2 for introducing underdominance within loci can be easily extended to introducing both within-locus and epistatic underdominance at two or more loci. For example, consider the 4-locus fitness table shown in Figure 5, where the ancestral population is in Hardy-Weinberg proportions for alleles *A*, *a*, *B*, and *b* but has only alleles *C* and *D* present.

Introduction of a mutant *c* allele will introduce underdominance at the *A* locus, through the directional selection process described in Section 3.1.2, and illustrated in the first column of 2-locus tables of Figure 5. Similarly, introduction

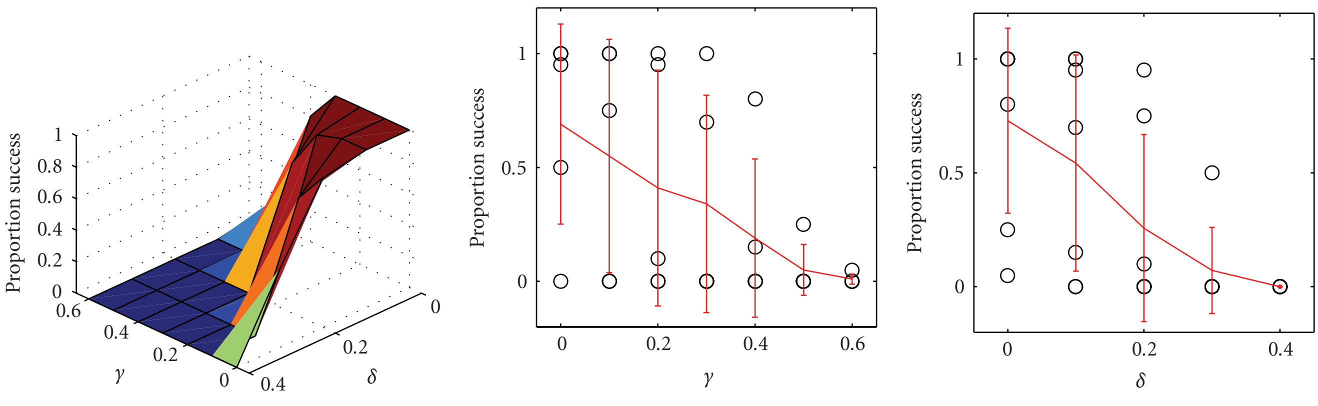


(a)

(b)

(c)

Figure 3: (a) Proportion of successful introductions (out of 20 trials on a 100×100 grid for each parameter combination, using 3×3 localized mating) of underdominance at the *B* locus via a single mutation to *a* at the *A* locus, using the fitness table shown in (2), as a function of the underdominance (γ) of *aaBb* relative to *aaBB*, and the fitness disadvantage (δ) of *aaBB* relative to *aabb*, for $\gamma \in \{0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6\}$ and $\delta \in \{0, 0.1, 0.2, 0.3, 0.4\}$, (b) success rate as a function of δ over all γ tested, and (c) success rate as a function of γ over all δ tested. In plots (b) and (c), open circles are the proportions of successful trials (out of 20) at each given parameter combination; solid lines are means and error bars are ± one standard deviation (averaged across all δ and γ, for plots (b) and (c), resp.), and the asterisk inside an open circle indicates the one case in each plot where *aaBb* is inviable.



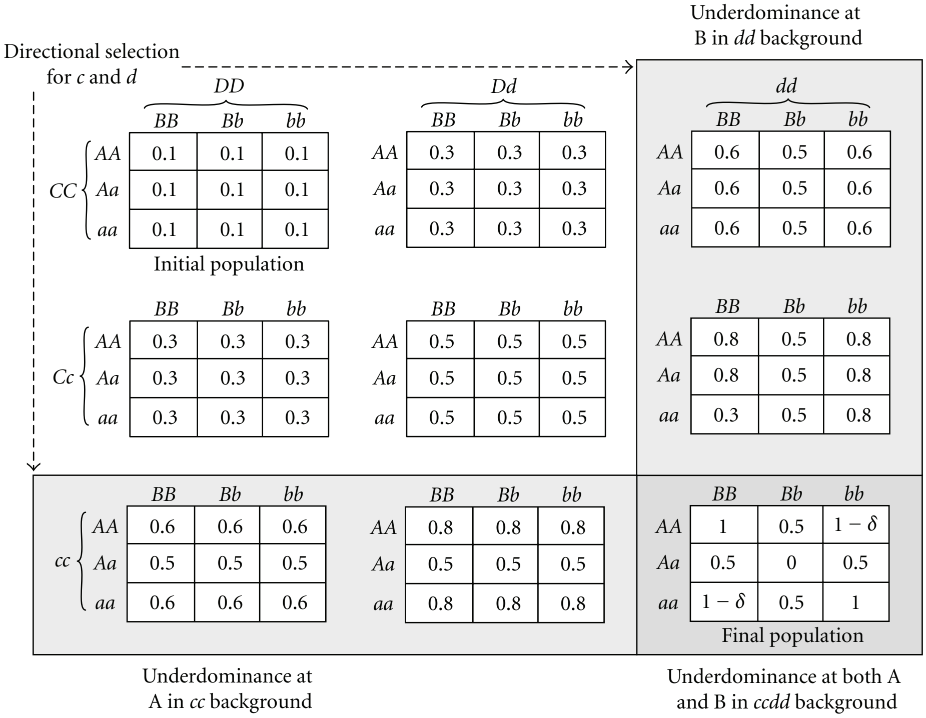
(a)

(b)

(c)

Figure 4: (a) Proportion of successful introductions (out of 20 trials on a 100 × 100 grid, using panmictic mating) of underdominance at the *B* locus via a single mutation to *a* at the *A* locus, using the fitness table shown in (2), as a function of the underdominance (γ) of *aaBb* relative to *aaBB*, and the fitness disadvantage (δ) of *aaBB* relative to *aabb*, for $\gamma \in \{0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6\}$ and $\delta \in \{0, 0.1, 0.2, 0.3, 0.4\}$, (b) success rate as a function of δ over all γ tested, and (c) success rate as a function of γ over all δ tested. In plots (b) and (c), open circles are the proportions of successful trials (out of 20) at each given parameter combination; solid lines are means and error bars are ± one standard deviation (averaged across all δ and γ, for plots (b) and (c), resp.).

of a mutant *d* allele will introduce underdominance at the *B* locus, as illustrated in the first row of 2-locus tables of Figure 5. Introduction of both *c* and *d* alleles can lead the population to the fitness table shown in the lower right 2-locus table of Figure 5, which is equivalent to Figure 1(g) (if $\delta = 0$) or Figure 1(h) (if $\delta = 0.08$). In order to test how frequently this occurs, we performed the following set of experiments, using the fitness table shown in Figure 5 with $\delta = 0.08$. In each case, a 100 × 100 population was randomly initialized in spatially uncorrelated Hardy-Weinberg proportions for *A*, *a*, *B*, and *b* but with only *C* and *D* alleles present. Then, *c* and *d* alleles were introduced in random locations; in one set of experiments, these two mutations were introduced simultaneously, whereas in another set of experiments the *c* allele was introduced first and, if it became fixed (i.e., if it replaced the *C* allele entirely), then the *d* allele was subsequently introduced. Both simultaneous and sequential introductions were tested in conjunction with both 3 × 3 localized mating and with panmixia, in 100 random trials for each of these four possible combinations. A trial was considered successful if and only if both *C* and *D* alleles disappeared while all of the *A*, *a*, *B*, *b*, *c*, and *d* alleles remained, so that the resulting fitnesses were as shown in Figure 1(h). In the case of panmixia, none of the trials were successful. However, with 3 × 3 localized mating 80% of the simultaneous introduction trials were successful


Directional selection for c and d
Underdominance at B in dd background
Initial population
Final population
Underdominance at A in cc background
Underdominance at both A and B in $ccdd$ background

		DD			Dd			dd		
		BB	Bb	bb	BB	Bb	bb	BB	Bb	bb
CC	AA	0.1	0.1	0.1	0.3	0.3	0.3	0.6	0.5	0.6
	Aa	0.1	0.1	0.1	0.3	0.3	0.3	0.6	0.5	0.6
	aa	0.1	0.1	0.1	0.3	0.3	0.3	0.6	0.5	0.6
Cc	AA	0.3	0.3	0.3	0.5	0.5	0.5	0.8	0.5	0.8
	Aa	0.3	0.3	0.3	0.5	0.5	0.5	0.8	0.5	0.8
	aa	0.3	0.3	0.3	0.5	0.5	0.5	0.3	0.5	0.8
cc	AA	0.6	0.6	0.6	0.8	0.8	0.8	1	0.5	$1-\delta$
	Aa	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0	0.5
	aa	0.6	0.6	0.6	0.8	0.8	0.8	$1-\delta$	0.5	1

FIGURE 5: A four-locus fitness table to illustrate how within-locus underdominance can be introduced and fixed at two loci. Starting from an ancestral population containing neutral alleles *A, a, B, b, C, D* (fitness as in upper left), independent simultaneous or sequential mutations to *c* and *d* alleles are both likely to become fixed via directional selection, resulting in a final population containing alleles *A, a, B, b, c, d* (fitness as in lower right), wherein both the *A* and *B* loci exhibit underdominance. Note that the resulting fitness shown in the lower right is equivalent to the fitness tables shown in Figures 1(g) and 1(h), for $\delta = 0$ and $\delta = 0.08$, respectively.

and 32% of the sequential introduction trials were successful in introducing the two-locus underdominance, even though in this example the underdominance is fairly strong and the homozygotes in the resulting population are not all equally fit. This process is easily generalizable to introducing underdominance at more than two loci, especially when the underdominance is mild.

3.2. Self-Organization in 2 Locus Systems Due to Multiscale Interactions. In the previous section, we established that underdominant polymorphisms, such as shown in Figures 1(g) and 1(h), can be easily introduced into populations with localized mating interactions. In this section, we tackle the question as to what happens in populations with multiple underdominant loci. Specifically, we wanted to see if self-organization of the genotypes would occur in spatially structured populations and if so, how this would affect the evolutionary dynamics.

Populations of 100 × 100 individuals with two bi-allelic loci were subject to fitnesses according to either the table shown in Figure 1(g) (within-locus underdominance with no epistasis) or the table shown in Figure 1(h) (within-locus underdominance with mild epistasis). The populations were randomly initialized in Hardy-Weinberg equilibrium, with all alleles having initially equal frequencies and spatially uncorrelated random uniform distribution across the spatial domain (e.g., Figure 6(a)), to preclude the introduction of initial bias in average effects or spatial organization. The random Hardy-Weinberg initialization is conservative when examining self-organization, since any initial clustering or local biases in fitness will only serve to nucleate cluster formation more quickly and speed up the process of self-organization. Groups of individuals are considered different species only if all hybrids between the groups are inviable. Experiments consisted of 10 random replications from each of 10 random starting domains, for both 3 × 3 localized mating and panmictic mating neighborhoods.

With panmixia, populations without epistasis (fitness as in Figure 1(g)) became completely fixed to one of the four possible homozygotes, with equal probability. With epistasis (fitness as in Figure 1(h)), panmictic populations became fixed to one of the two fittest homozygotes, with equal probability. These results are consistent with mean field predictions that underdominance cannot be maintained in populations with random mating.

When populations experience 3 × 3 localized mating, however, the results are more interesting. Without epistasis (fitness as in Figure 1(g)), the populations quickly self-organize into a patchy structure of the four possible homozygotes and the sizes of these clusters coarsens over time (e.g., Figures 6(b), 6(c), and 6(d)). In this case, speciation does not occur since gene flow remains possible between all four homozygotes (Figure 1(g)). In contrast, with disruptive epistasis present (where the most fit genotypes

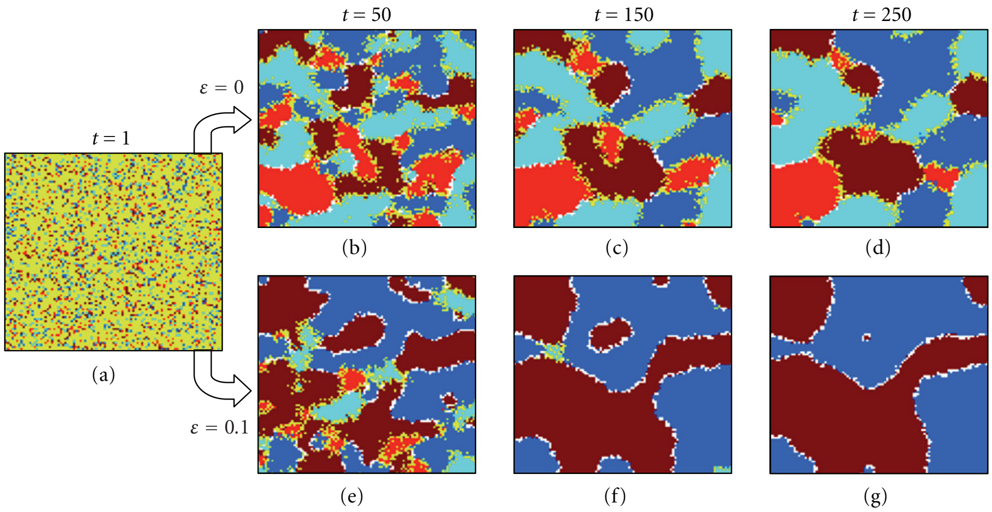


Figure 6: (a) A representative two-locus, bi-allelic, diploid population initialized in Hardy Weinberg equilibrium. (b)–(d) With no epistasis (per fitness table in Figure 1(g)), the population self-organizes into clusters of the four possible homozygotes separated by hybrid zones, most of which are permeable, so no speciation occurs. (e)–(g) With epistasis (per fitness table in Figure 1(h)), the boundaries coalesce into impermeable hybrid zones, leaving reproductively isolated populations (i.e., species) of the two most fit homozygotes (*AABB* and *aabb*). The variable *t* refers to the number of generations. Inviable genotypes (*AaBb*) are shown in white.

are genetically incompatible with each other, such as with fitness in Figure 1(h)), populations with localized mating invariably self-organize into reproductively isolated clusters of the two fittest genotypes (e.g., Figures 6(e), 6(f), and 6(g)), despite the absence of any environmental heterogeneity, externally imposed barriers to gene flow, or assortative mate preference. Thus, multiscale interactions comprising within-locus underdominance, between-locus epistasis, and localized mating interactions between individuals can result in self-organized speciation.

It should be noted that if allowed to run indefinitely, stochastic events in these finite and homogeneous simulated spatial domains ultimately favor one or the other species. However, real ecological domains are heterogeneous and once reproductive isolation has occurred, it is likely that two species will continue to diverge and, therefore, may not continue to be in direct competition for the same set of resources.

3.3. Extension to More than 2 Loci. For speciation to occur due to self-organization of only two underdominant loci, the degree of underdominance must be significant, so that the double heterozygote is completely inviable. However, we now consider a more biologically realistic scenario in which mild underdominance exists at several loci. Will such populations still exhibit self-organized speciation? In order to tackle this question we created a generalized fitness function exhibiting underdominance with optional epistasis, as follows:

$$f_i = \frac{1 - \; U_i + E_i}{1 + E_{\max}}, \tag{3}$$

where f_i is the fitness of individual *i*. Genotypes of individuals comprised *L* bi-allelic underdominant loci, where the two alleles at a given locus are identified by uppercase or lowercase letters. In (3) a maximum potential fitness of 1 is reduced by an underdominance penalty *U*, increased by an epistatic bonus *E*, and then renormalized so that the maximum possible fitness is brought back to 1. The underdominance penalty *U* is computed as the proportion of underdominant loci that are heterozygous. Thus, the more loci in the genotype, the milder the underdominance, and only genotypes heterozygous at all underdominant loci are inviable. Note that this strict inverse dependence of the degree of individual within-locus underdominance on the number of interacting loci is the most conservative approach for examining whether speciation will occur, since we construct these genomes so that, in all cases, there is only one possible genotype that is completely inviable. Speciation would be more likely to occur if there were multiple inviable genotypes, and would never occur if the within-locus underdominance penalty U were less than 1.0 for all genotypes. The epistatic bonus *E* is computed as the product of an epistatic coefficient ε and the maximum of the number of homozygous loci with the same case (upper or lower), such that only the two most genetically distinct homozygous genotypes (e.g., *AABBCC* and *aabbcc*) experience equal and maximal fitness.

This simple fitness function was employed because it allows easy control of both the degree of underdominance (by changing the number of loci) and the degree of epistasis (by changing ε) being modeled, while still maintaining identical average effects for each locus and each allele in the initial populations (which were randomly initialized in spatially uncorrelated Hardy-Weinberg equilibrium for all alleles). Note that for a 2-locus system, the fitnesses for the 9 genotypes shown in the tables in Figures 1(g) and 1(h) can be generated from (3), where $\varepsilon = 0$ and $\varepsilon = 0.1$, respectively (note that these same fitnesses could

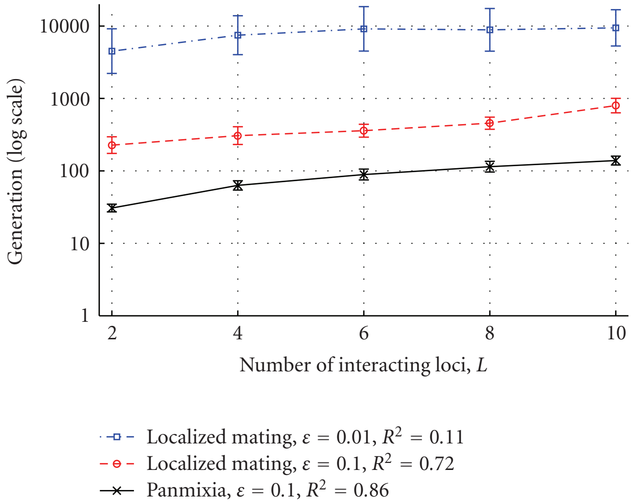
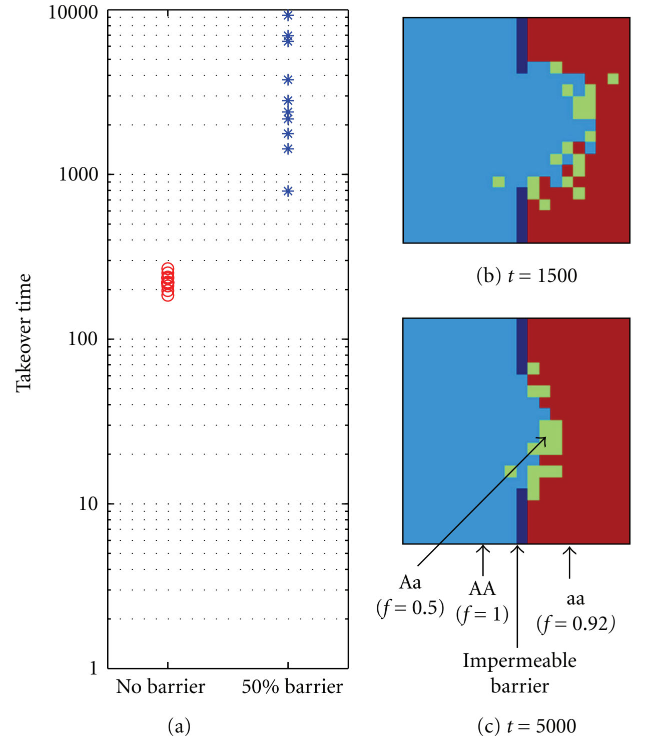


Figure 7: The number of generations until either speciation (with localized mating and epistasis $\varepsilon = 0.01$ or $\varepsilon = 0.1$) or fixation of a single genotype (with panmixia and $\varepsilon = 0.1$). Each data point represents the mean of 100 trials (10 random runs from each of 10 random initial conditions), with vertical bars representing standard deviations.



(a)

(b) $t = 1500$

(c) $t = 5000$

Figure 8: (a) Takeover times for *AA* when there is no barrier (red circles) or a 50% barrier (blue asterisks); see text for details. (b) and (c) Snapshots of population structure of a representative run with a 50% barrier at 1500 and 5000 generations, respectively.

have resulted from the evolutionary process described in Section 3.1.3 and illustrated in Figure 5). Experiments were performed with $L \in \{2, 4, 6, 8, 10\}$ underdominant loci and epistasic coefficients of $\varepsilon \in \{0.01, 0.1\}$, with both panmixia and 3×3 localized mating. Each experimental configuration was run for 100 trials (10 random runs from each of 10 random initial populations).

As before, when mating is panmictic, the population rapidly fixes to one of the two fittest genotypes with equal probability, and speciation does not occur (Figure 7, bottom line). However, with 3×3 localized mating, speciation was observed in 100% of the trials for both values of epistasis tested, (Figure 7, top two lines). As the number of loci increases (and consequently the degree of within-locus underdominance decreases) the fitness valleys of heterozygotes at each locus become less pronounced, allowing increasingly easy traversal of fitness valleys and enabling underdominant polymorphisms to persist longer in the population. For example, with mild epistasis ($\varepsilon = 0.1$), the number of generations to speciation events increased exponentially ($R^2 = 0.72$) with the number of interacting loci L (Figure 7, middle line). Decreasing the epistasis coefficient by an order of magnitude (to $\varepsilon = 0.01$) increased the mean of the log of time to speciation by an order of magnitude ($P < .0001$, ANOVA) but also increased the variance ($P < .0001$, O'Brien's test), with a corresponding drop in correlation ($R^2 = 0.11$, Figure 7, top line). In the latter case, the asymmetry in fitness between any of the homozygotes is almost negligible, enabling the underdominant polymorphisms to persist longer before speciation occurs.

The results of these experiments demonstrate that, with localized mating and mild underdominance, clusters of homozygous genotypes spontaneously form and can coexist for long periods of time. With even a small amount of disruptive epistasis, leaky genetic boundaries between these clusters tend to coalesce over time to form impermeable genetic barriers to gene flow, even when individual loci are nearly neutral. Thus, speciation can occur as an emergent property from the self-organization of multiple underdominant polymorphisms in populations with localized mating.

3.4. *The Effect of External Barriers to Gene Flow.*

As shown in Figure 7, underdominant alleles and less fit genotypes can persist for long periods in a single interbreeding population, if mating is spatially localized, even when the domain is completely homogeneous and no niche differentiation occurs. However, the presence of external barriers to gene flow can further enhance the persistence of underdominance and less fit genotypes in an interbreeding population. If even partial external (e.g., geographic) barriers to gene flow are present, self-organized cluster boundaries will tend to become stabilized at external boundaries [57]. Consider a simple single-locus 20×20 population, where the left half of the domain is initially populated with the homozygote *AA* with fitness 1.0, while the right half is populated with the homozygote *aa* with fitness 0.92, and the heterozygote *Aa* has fitness 0.5 (i.e., fitness is as in column 1 of the table shown in Figure 1(h)). When there is no physical barrier between them, the more fit *AA* takes over the entire domain in an average of only 224 generations (10 trials, standard deviation = 25), whereas when 10 of the 20 cells are blocked by an impermeable external boundary, leaving a 10-cell window in the center, the takeover time increases by over an order of magnitude to an average of 3777 (10 trials, standard deviation = 2809), with one takeover time as high as 9254 generations (Figure 8(a)). The reason for this dramatic slowdown in takeover by the more fit genotype is illustrated by snapshots from a representative run. At 1500 generations (Figure 8(b)), the more fit *AA* genotype has made a bulge into the half if the domain initially occupied by *aa*. However, the fitness advantage of *AA* is countered by the fact that the local mating neighborhoods at the convex cluster boundary have a larger proportion of *aa* genotypes, which increases their probability of being selected by (1). In fact, the bulge tends to grow and shrink in size over time; in this example it was much smaller at 5000 generations (Figure 8(c)) than it was at 1500 generations (Figure 8(b)). Ultimately, if given enough time, a fitter genotype will "break through" the barrier and then rapidly take over the rest of the domain (this particular run took 7166 generations for complete takeover).

4. Discussion and Conclusions

Underdominance can conceivably enter the genome of an interbreeding population via a variety of potential mechanisms. Previously proposed mechanisms include environmental change [38], disruptive selection caused by niche differentiation [39, 40], and "additive by dominance" epistasis [55]. Here, we demonstrate a simple alternative and biologically reasonable mechanism by which within-locus underdominance can easily become established at one or more loci, either simultaneously or sequentially. Specifically, the proposed mechanism requires (i) an initial condition comprising a preexistent neutral polymorphism at a locus (ii) an advantageous mutation at a second locus (which thus becomes fixed by directional selection), and (iii) an epistatic interaction between the two loci, such that the first (previously neutral) locus becomes underdominant in the background of the newly fixed favorable allele at the second locus. We note that these requirements are consistent with the existence of a large amount of observed neutral polymorphism, occasional advantageous mutations, and pervasive epistatic genetic interactions in biological organisms [47]. Our simulations show that, if mating is panmictic, then only mild underdominance can be introduced in this manner and is not likely to persist for long. However, when mating is spatially localized, even strong underdominance with mild asymmetry can be easily introduced and maintained in interbreeding populations for long durations. Our model thus illustrates how underdominance at multiple loci can easily be introduced into interbreeding populations with localized interactions.

We also demonstrate that in locally mating populations exhibiting mild underdominance at multiple loci, the populations self-organize into clusters of compatible genotypes. Gene flow persists between clusters unless the hybrids between clusters are completely inviable. Even in the extreme case, where boundaries for different underdominant loci are initially independent of each other, over time they become aligned. Thus, leaky genetic boundaries coalesce to form harder genetic boundaries (deeper fitness valleys). When certain forms of mild epistasis are present, speciation can be an emergent property in this model, arising as the result of multiscale interactions (within-locus, between-locus, and between individuals) without any geographic, niche-based, mate preference, or other premating isolating mechanisms. In contrast, self-organization cannot occur when mating is panmictic, in which case the populations invariably converge on a single genotype.

Just as localized mating can promote maintenance of genetic polymorphisms at multiple diploid loci in patchy structures when selection is stabilizing [36], we have shown that a similar process can occur when selection is disruptive. In both cases, genetic redundancy (where multiple genotypes have the same fitness) help to stabilize the polymorphisms. However, under disruptive selection even clusters of unequal fitness can persist long enough for speciation to occur, since the fitness valleys in the hybrid zones between unequally fit genotypes slow the takeover by the fitter genotype. If external barriers to gene flow are also present, then these can increase persistence of even asymmetric underdominant polymorphisms by further stabilizing cluster boundaries.

In the experiments reported here, mating interactions were either panmictic or used overlapping 3 × 3 localized mating neighborhoods. However, even when mating is generally localized in natural populations, there are still likely to be occasional long range interactions (e.g., long range migration events in animals or unusually long dispersal of pollen or seeds in plants). In a separate set of experiments reported elsewhere [58], we assessed the sensitivity of simulated self-organized speciation to relaxations in the assumption of strictly localized mating. Specifically, we altered the interaction topology from nearest neighbor interactions to panmictic interactions in two ways: (i) by increasing the size of the contiguous mating neighborhoods and (ii) by allowing for long-distance dispersal of individuals with increasing probability. The results of that study [58] show self-organized speciation to be robust to mating neighborhood sizes significantly larger than nearest neighbor interactions (relative neighborhood size to domain size is actually shown to be the governing parameter, as in cellular evolutionary algorithms [59]) and to probabilities of long-distance dispersal that fall well into the range of so called "small-world" [60] interaction topologies.

Spatially explicit models, such as employed here, are not generally analytically tractable, and the lack of closed form solutions has led some to claim that this limits the generality of theoretical conclusions [6]. However, in complex biological systems, the generality of theoretical conclusions may be even more severely limited by the assumptions necessary for analytical tractability, and by the principle of computational irreducibility [61] simulations can be necessary in order to gain insight into complex multiscale spatiotemporal evolutionary processes. We do not dispute that analytical solutions based on assumptions such as panmixia or haploidy can certainly lead to useful generalizations in some circumstances. Yet, as demonstrated in this contribution as well as a variety of other studies of both simulated and natural populations (e.g., [16, 34, 36, 38, 54, 57, 62–65]), essential evolutionary dynamics often emerge as a consequence of spatially-constrained interactions. While the model employed herein is highly idealized, it nonetheless manifests properties observed in natural populations, while removing the confounding effects of differences in initial average effects of different alleles or different loci, heterogeneity in the environment, or premating isolation of similar genotypes due to mate selection or geographic isolation. The three primary assumptions in our model of self-organized speciation are that populations can exhibit (i) underdominant polymorphisms, (ii) epistatic genetic interactions, and (iii) spatially localized mating, all of which have been widely observed in natural populations, as discussed in the introduction. These simulations yield potentially useful generalizations and insights, demonstrate the sensitivity of evolutionary processes to spatial and multiscale aspects of interactions, and underscore the importance of taking these complexities into account.

The degree to which epistatic underdominance is a significant driving force in natural evolution is difficult to say. Certainly, hybrid zones of reduced fitness are commonly observed between closely related species, but when and how these hybrid incompatibilities evolved is impossible to determine in retrospect. However, while this study cannot answer the question of whether or not recombination and self-organization of many nearly neutral underdominant alleles has led to emergent intrinsic barriers to gene flow in natural systems, we argue that it does indicate that such processes may be feasible and even parsimonious mechanisms for genetic divergence without premating isolation. We conclude that multiscale interactions can potentially help to maintain underdominant polymorphisms and may contribute to speciation events. This model shows one way that the emergent properties in complex biological communities can drive evolutionary change. It is probable that, in natural systems, many mechanisms are operating simultaneously to cause speciation.

Acknowledgments

This work was supported in part by US Department of Energy through its EPSCoR Program (DOE-FG02-00ER45828) for a graduate fellowship to support Joshua L. Payne, and Vermont EPSCoR (NSF EPS 0701410) for a graduate fellowship to support Joshua L. Payne and a pilot project award granted to Margaret J. Eppstein.

References

[1] C. Darwin, *The Origin of Species*, Avenal Books, New York, NY, USA, 1859.

[2] J. A. Coyne and H. A. Orr, “The evolutionary genetics of speciation,” *Philosophical Transactions of the Royal Society B*, vol. 353, no. 1366, pp. 287–305, 1998.

[3] J. A. Coyne and H. A. Orr, *Speciation*, Sinauer Associates, Sunderland, Mass, USA, 2004.

[4] U. Dieckmann, M. Doebeli, J. A. J. Metz, and D. Tautz, *Adaptive Speciation*, Cambridge University Press, Cambridge, UK, 2004.

[5] M. Doebeli, U. Dieckmann, J. A. J. Metz, and D. Tautz, “What we have also learned: adaptive speciation is theoretically plausible,” *Evolution*, vol. 59, no. 3, pp. 691–695, 2005.

[6] S. Gavrilets, “Perspective: models of speciation—what have we learned in 40 years?” *Evolution*, vol. 57, no. 10, pp. 2197–2215, 2003.

[7] S. Gavrilets, *Fitness Landscapes and the Origin of Species*, Princeton University Press, Princeton, NJ, USA, 2004.

[8] M. Kirkpatrick and V. Ravigné, “Speciation by natural and sexual selection: models and experiments,” *American Naturalist*, vol. 159, supplement 3, pp. S22–S35, 2002.

[9] W. Bateson, “Heredity and variation in modern lights,” in *Darwin and Modern Science*, Cambridge University Press, Cambridge, UK, 1909.

[10] T. Dobzhansky, *Genetics and the Origin of Species*, Columbia University Press, New York, NY, USA, 1937.

[11] H. J. Muller, “Isolating mechanisms, evolution, and temperature,” *Biology Symposium*, vol. 6, pp. 71–125, 1942.

[12] U. K. Schliewen, D. Tautz, and S. Paabo, “Sympatric speciation suggested by monophyly of crater lake cichlids,” *Nature*, vol. 368, no. 6472, pp. 629–632, 1994.

[13] D. Schluter, “Experimental evidence that competition promotes divergence in adaptive radiation,” *Science*, vol. 266, no. 5186, pp. 798–801, 1994.

[14] E. B. Knox and J. D. Palmer, “Chloroplast DNA variation and the recent radiation of the giant senecios (Asteraceae) on the tall mountains of Eastern Africa,” *Proceedings of the National Academy of Sciences of the United States of America*, vol. 92, no. 22, pp. 10349–10353, 1995.

[15] U. Dieckmann and M. Doebeli, “On the origin of species by sympatric speciation,” *Nature*, vol. 400, no. 6742, pp. 354–357, 1999.

[16] M. Doebeli and U. Dieckmann, “Speciation along environmental gradients,” *Nature*, vol. 421, no. 6920, pp. 259–264, 2003.

[17] J. D. Fry, “Multilocus models of sympatric speciation: Bush versus Rice versus Felsenstein,” *Evolution*, vol. 57, no. 8, pp. 1735–1746, 2003.

[18] M. Higashi, G. Takimoto, and N. Yamamura, “Sympatric speciation by sexual selection,” *Nature*, vol. 402, pp. 523–526, 1999.

[19] T. J. Kawecki, “Sympatric speciation via habitat specialization driven by deleterious mutations,” *Evolution*, vol. 51, no. 6, pp. 1751–1763, 1997.

[20] A. S. Kondrashov and M. Shpak, “On the origin of species by means of assortative mating,” *Proceedings of the Royal Society B*, vol. 265, no. 1412, pp. 2273–2278, 1998.

[21] A. S. Kondrashov and F. A. Kondrashov, “Interactions among quantitative traits in the course of sympatric speciation,” *Nature*, vol. 400, no. 6742, pp. 351–354, 1999.

[22] W. R. Rice, “Disruptive selection on habitat preference and the evolution of reproductive isolation: a simulation study,” *Evolution*, vol. 38, no. 6, pp. 1251–1260, 1984.

[23] W. R. Rice, “Speciation via habitat specialization: the evolution of reproductive isolation as a correlated character,” *Evolutionary Ecology*, vol. 1, no. 4, pp. 301–314, 1987.

[24] D. Udovic, “Frequency-dependent selection, disruptive selection, and the evolution of reproductive isolation,” *American Naturalist*, vol. 116, pp. 621–641, 1980.

[25] C. B. Fenster, "Gene flow in *Chamaecrista fasciculata* (Leguminosae). I. Gene dispersal," *Evolution*, vol. 45, no. 2, pp. 398–409, 1991.

[26] D. A. Levin and H. W. Kerster, "Local gene dispersal in *Phlox*," *Evolution*, vol. 22, pp. 130–139, 1968.

[27] D. L. Marr, J. Leebens-Mack, L. Elms, and O. Pellmyr, "Pollen dispersal in *Yucca filamentosa* (Agavaceae): the paradox of self-pollination behavior by *Tegeticula yuccasella* (Prodoxidae)," *American Journal of Botany*, vol. 87, no. 5, pp. 670–677, 2000.

[28] K. B. Park and M. G. Chung, "Indirect measurement of gene flow in *Hosta capitata* (Liliaceae)," *Botanical Bulletin of Academia Sinica*, vol. 38, no. 4, pp. 267–272, 1997.

[29] J. J. Robledo-Arnuncio and L. Gil, "Patterns of pollen dispersal in a small population of *Pinus sylvestris* L. revealed by total-exclusion paternity analysis," *Heredity*, vol. 94, no. 1, pp. 13–22, 2005.

[30] J. A. Endler, *Geographic Variation, Speciation, and Clines*, Princeton University Press, Princeton, NJ, USA, 1977.

[31] R. K. Grosberg, "Limited dispersal and proximity-dependent mating success in the clonal ascidian *Botryllus schlosseri*," *Evolution*, vol. 41, pp. 412–429, 1987.

[32] M. A. Smith and D. M. Green, "Sex, isolation and fidelity: unbiased long-distance dispersal in a terrestrial amphibian," *Ecography*, vol. 29, no. 5, pp. 649–658, 2006.

[33] S. Wright, "Isolation by distance," *Genetics*, vol. 28, pp. 114–138, 1943.

[34] F. J. Rohlf and G. D. Schnell, "An investigation of the isolation-by-distance model," *American Naturalist*, vol. 105, pp. 295–324, 1971.

[35] M. E. Turner, J. C. Stephens, and W. W. Anderson, "Homozygosity and patch structure in plant populations as a result of nearest-neighbor pollination," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 79, no. 1, pp. 203–207, 1982.

[36] D. B. Goldstein and K. E. Holsinger, "Maintenance of polygenic variation in spatially structured populations: roles for local mating and genetic redundancy," *Evolution*, vol. 46, no. 2, pp. 412–429, 1992.

[37] R. J. H. Payne and D. C. Krakauer, "Sexual selection, space, and speciation," *Evolution*, vol. 51, no. 1, pp. 1–9, 1997.

[38] H. Sayama, L. Kaufman, and Y. Bar-Yam, "Symmetry breaking and coarsening in spatially distributed evolutionary processes including sexual reproduction and disruptive selection," *Physical Review E*, vol. 62, no. 5B, pp. 7065–7069, 2000.

[39] S. Gavrilets, "The Maynard Smith model of sympatric speciation," *Journal of Theoretical Biology*, vol. 239, no. 2, pp. 172–182, 2006.

[40] D. S. Wilson and M. Turelli, "Stable underdominance and the evolutionary invasion of empty niches," *American Naturalist*, vol. 127, no. 6, pp. 835–850, 1986.

[41] A. S. Kondrashov, "Accumulation of Dobzhansky-Muller incompatibilities within a spatially structured population," *Evolution*, vol. 57, no. 1, pp. 151–153, 2003.

[42] F. Fel-Clair, T. Lenormand, J. Catalan, et al., "Genomic incompatibilities in the hybrid zone between house mice in Denmark: evidence from steep and non-coincident chromosomal clines for Robertsonian fusions," *Genetical Research*, vol. 67, no. 2, pp. 123–134, 1996.

[43] L. F. Galloway and J. R. Etterson, "Population differentiation and hybrid success in *Campanula americana*: geography and genome size," *Journal of Evolutionary Biology*, vol. 18, no. 1, pp. 81–89, 2005.

[44] D. C. Presgraves, L. Balagopalan, S. M. Abmayr, and H. A. Orr, "Adaptive evolution drives divergence of a hybrid inviability gene between two species of *Drosophila*," *Nature*, vol. 423, no. 6941, pp. 715–719, 2003.

[45] N. M. Waser, M. V. Price, and R. G. Shaw, "Outbreeding depression varies among cohorts of *Ipomopsis aggregata* planted in nature," *Evolution*, vol. 54, no. 2, pp. 485–491, 2000.

[46] R. A. Swanson-Wagner, Y. Jia, R. DeCook, L. A. Borsuk, D. Nettleton, and P. S. Schnable, "All possible modes of gene action are observed in a global comparison of gene expression in a maize F_1 hybrid and its inbred parents," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 103, no. 18, pp. 6805–6810, 2006.

[47] M. Nei, "Selectionism and neutralism in molecular evolution," *Molecular Biology and Evolution*, vol. 22, no. 12, pp. 2318–2342, 2005.

[48] S. R. Proulx and P. C. Phillips, "The opportunity for canalization and the evolution of genetic networks," *American Naturalist*, vol. 165, no. 2, pp. 147–162, 2005.

[49] A. H. Y. Tong, G. Lesage, G. D. Bader, et al., "Global mapping of the yeast genetic interaction network," *Science*, vol. 303, no. 5659, pp. 808–813, 2004.

[50] E. Ravasz, A. L. Somera, D. A. Mongru, Z. N. Oltvai, and A.-L. Barabasi, "Hierarchical organization of modularity in metabolic networks," *Science*, vol. 297, no. 5586, pp. 1551–1555, 2002.

[51] J. H. Moore, "The ubiquitous nature of epistasis in determining susceptibility to common human diseases," *Human Heredity*, vol. 56, no. 1–3, pp. 73–82, 2003.

[52] T. A. Thornton-Wells, J. H. Moore, and J. L. Haines, "Genetics, statistics and human disease: analytical retooling for complexity," *Trends in Genetics*, vol. 20, no. 12, pp. 640–647, 2004.

[53] P. D. Thomas and A. Kejariwal, "Coding single-nucleotide polymorphisms associated with complex vs. Mendelian disease: evolutionary evidence for differences in molecular effects," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 101, no. 43, pp. 15398–15403, 2004.

[54] M. E. Turner, J. C. Stephens, and W. W. Anderson, "Homozygosity and patch structure in plant populations as a result of nearest-neighbor pollination," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 79, no. 1, pp. 203–207, 1982.

[55] C. J. Goodnight, "Quantitative trait loci and gene interaction: the quantitative genetics of metapopulations," *Heredity*, vol. 84, no. 5, pp. 587–598, 2000.

[56] H. A. Orr, "The population genetics of speciation: the evolution of hybrid incompatibilities," *Genetics*, vol. 139, no. 4, pp. 1805–1813, 1995.

[57] H. Sayama, L. Kaufman, and Y. Bar-Yam, "Spontaneous pattern formation and genetic diversity in habitats with irregular geographical features," *Conservation Biology*, vol. 17, no. 3, pp. 893–900, 2003.

[58] J. L. Payne, M. J. Eppstein, and C. J. Goodnight, "Sensitivity of self-organized speciation to long-distance dispersal," in *Proceedings of IEEE Symposium on Artificial Life (CI-ALife '07)*, pp. 1–7, 2007.

[59] E. Alba and B. Dorronsoro, "The exploration/exploitation tradeoff in dynamic cellular genetic algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 9, no. 2, pp. 126–142, 2005.

[60] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks," *Nature*, vol. 393, no. 6684, pp. 440–442, 1998.

[61] S. Wolfram, "Undecidability and intractability in theoretical physics," *Physical Review Letters*, vol. 54, no. 8, pp. 735–738, 1985.

[62] B. Kerr, M. A. Riley, M. W. Feldman, and B. J. M. Bohannan, "Local dispersal promotes biodiversity in a real-life game of rock-paper-scissors," *Nature*, vol. 418, no. 6894, pp. 171–174, 2002.

[63] T. L. Czárán, R. F. Hoekstra, and L. Pagie, "Chemical warfare between microbes promotes biodiversity," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 99, no. 2, pp. 786–790, 2002.

[64] K. Johst, M. Doebeli, and R. Brandl, "Evolution of complex dynamics in spatially structured populations," *Proceedings of the Royal Society B*, vol. 266, no. 1424, pp. 1147–1154, 1999.

[65] J. M. J. Travis and C. Dytham, "The evolution of dispersal in a metapopulation: a spatially explicit, individual-based model," *Proceedings of the Royal Society B*, vol. 265, no. 1390, pp. 17–23, 1998.

Hindawi Publishing Corporation
Journal of Artificial Evolution and Applications
Volume 2009, Article ID 963150, 10 pages
doi:10.1155/2009/963150

Research Article

Multiple Sequence Alignment Using a Genetic Algorithm and GLOCSA

Edgar D. Arenas-Díaz,[1] Helga Ochoterena,[2] and Katya Rodríguez-Vázquez[1]

[1] *Instituto de Investigaciones en Matemáticas Aplicadas y en Sistemas, Universidad Nacional Autónoma de Mexico, Circuito exterior s/n, Ciudad Universitaria, 04510 Mexico, DF, Mexico*
[2] *Instituto de Biología, Universidad Nacional Autónoma de Mexico, Apdo. Postal 70-367, 04510 Mexico, DF, Mexico*

Correspondence should be addressed to Edgar D. Arenas-Díaz, xaltonalli@gmail.com

Received 14 November 2008; Revised 4 April 2009; Accepted 13 June 2009

Recommended by Jason Moore

Algorithms that minimize putative synapomorphy in an alignment cannot be directly implemented since trivial cases with concatenated sequences would be selected because they would imply a minimum number of events to be explained (e.g., a single insertion/deletion would be required to explain divergence among two sequences). Therefore, indirect measures to approach parsimony need to be implemented. In this paper, we thoroughly present a Global Criterion for Sequence Alignment (GLOCSA) that uses a scoring function to globally rate multiple alignments aiming to produce matrices that minimize the number of putative synapomorphies. We also present a Genetic Algorithm that uses GLOCSA as the objective function to produce sequence alignments refining alignments previously generated by additional existing alignment tools (we recommend MUSCLE). We show that in the example cases our GLOCSA-guided Genetic Algorithm (GGGA) does improve the GLOCSA values, resulting in alignments that imply less putative synapomorphies.

1. Introduction

The use of DNA (deoxyribonucleic acid) or protein sequences for different purposes has greatly increased as the technology for DNA and protein sequencing has improved with the consequent cost reduction. A proof for this is the enormous amount of information available in the Protein Data Bank [1] or in GenBank [2]. The exponential growth in size of these data repositories goes in parallel with the increasing need for tools to manage and analyze the valuable information therein contained. The first step to make this information manageable is to device tools to identify comparable proteins or DNA fragments, as well as comparable protein or DNA sequence units (amino acids and nucleotides, resp.). This process is referred to as sequence alignment. By aligning sequences, phylogenetic analyses can be carried out, PCR (polymerase chain reaction) primers constructed, secondary or tertiary structures predicted, among other applications. Being such a central topic, algorithms to tackle sequence alignment have already been developed. Nevertheless, as we explain more thoroughly in Section 2.2.1, sequence alignment is not a trivial problem. To reduce this complex issue to trackable problems, most available softwares consider at once pairs of sequences. Measures for alignment quality that globally use the entire data sets (matrices consisting of more than two sequences) are currently unavailable.

In this paper, we thoroughly present a Global Criterion for Sequence Alignment (GLOCSA) that uses a scoring function to globally rate multiple alignments aiming to indirectly use the parsimony criterion. We also propose an evolutionary computation technique suitable to optimize it. So this novel objective function is coupled with a Genetic Algorithm (GA), the GLOCSA-Guided Genetic Algorithm (GGGA), which uses a compact representation of the alignments and five different mutation operators to explore the solution landscape. Although GGGA can be used for completely unaligned data sets, it is more efficient for refining alignments previously generated by additional existing tools. Using GLOCSA as the scoring parameter, GGGA is capable of improving alignments generated by other tools (MUSCLE (multiple sequence comparison by log-expectation) v3.6 [3] was used in this work to prealign the matrices).

2. Sequences and Alignments

2.1. Sequences. DNA consists of a unique sequence of repeated four nucleotides. Each nucleotide is characterized by a corresponding nitrogenous base representing the *primary structure* of a real or hypothetical DNA molecule or strand, with the capacity to carry information. Such sequences analogously exist for RNA (ribonucleic acid) and proteins [4, 5].

In biochemistry, the primary structure of a biological molecule is the exact specification of its atomic composition and the chemical bonds connecting those atoms. For molecules of DNA, RNA, or proteins, the primary structure is equivalent to specify the sequence of its monomeric subunits, that is, the nucleotides or aminoacids sequence [4, 5].

2.2. Sequence Alignment. DNA sequences, RNA sequences and the protein sequences encoded change through time, evolving mainly under the action of mutation. The simplest types of mutation are point mutations, which are *substitutions* of nucleotides or aminoacids, and insertions/deletions, also known as *indels*. When one or two comparable sequences suffered insertion and/or deletion mutations, they will differ in length (i.e., they will have a different number of nucleotides or amino acids). Because these mutations are normally not observable, it is necessary to deduce where they occurred in order to identify which nucleotides or amino acids originally occupy the same position (which ones are homologous). This is the alignment process. Although this could appear trivial, it is a complex task due to the fact that a limited and a priori known number of minimum observable units exist for each aligned position (e.g., in the case of DNA only four nucleotides) and that all positions have the same potential alternative conditions (e.g., in DNA each unaligned position needs to have one of the four nucleotides). In this way, a gap (an inferred indel) in a sequence can be placed in many positions without making a big difference with respect to the comparable sequence. To align two or more sequences, they are put together in a $(S \cdot C)$ matrix, where S is the number of sequences, and C is the maximum number of residues in a sequence(positions in the alignment); the shorter sequences are filled at the end with gap codifications ("$-$") to fit the matrix perfectly. With a sequence in each line of the matrix, the process of alignment, represents the insertion of $-$ in the sequences (see Table 11). In order to choose the *best* alignment, it is considered that, in biological terms, the process of alignment has the objective to align homologous residues (having the same evolutionary origin). Assuming that evolution is parsimonious, when performing an alignment the aim is to minimize the number of evolutionary changes (events of substitutions or indels) that the alignment implies [6].

Alignments can be either pairwise, two sequences only, or multiple, more than two sequences up to an arbitrary number. For pairwise alignments *dynamic programming* algorithms have been developed to address this problem, such as Needleman-Wunsch [7] and Smith-Waterman [8] algorithms. Pairwise alignments might be regarded as special cases of multiple alignment. In practice, however, the computational complexity of aligning multiple sequences is such that the corresponding algorithms are usually not straight extensions of the pairwise approaches. Instead, multiple alignments are often constructed by repeatedly merging pairwise alignments (*progressive alignment*) [6].

2.2.1. The Number of Possible Alignments of Two Sequences. In order to define the complexity of finding an optimal alignment given an objective function, the number of possible alignments can be computed.

Having two sequences $S_1 = S_1[1]S_1[2]\cdots S_1[m]$ and $S_2 = S_2[1]S_2[2]\cdots S_2[n]$ of size m and n, respectively, $f(m,n)$ can be defined as the number of alignments that can be formed between them.

Any possible alignment of S_1 and S_2 ends in one of these specific ways [6]:

$$\left(\frac{S_1[m]}{-}\right), \left(\frac{S_1[m]}{S_2[n]}\right), \quad \text{or} \quad \left(\frac{-}{S_2[n]}\right). \tag{1}$$

That is, the last residue of S_1 aligned with a gap codification $-$, the last residue of S_1 and S_2 aligned, or the last residue of S_2 aligned with a gap codification $-$.

Considering the effect of these three possible ends on the number of alignments that can be formed out of the remaining residues in the alignment, the ending $(S_1[m]/-)$ removes one residue from S_1, $(S_1[m]/S_2[n])$ removes one residue from both sequences and $(-/S_2[n])$ removes one residue from S_2. Following this, the next recursion can be written [6]:

$$f(m,n) = f(m-1,n) + f(m-1,n-1) + f(m,n-1). \tag{2}$$

Each of the terms in the righthand side of (2) represents a possible end. In addition to this recursion, a *stop criterion* or *boundary condition* is needed:

$$f(m,0) = f(0,n) = f(0,0) = 1. \tag{3}$$

Using the recursion in (2) and the stop criterion in (3), the number of possible alignments of two sequences of equal length from 1 to 10, $m = n = 1,2,\ldots,10$, can be computed (Table 1), where it is obvious that the number of alignments grows exponentially as the length of the sequences increments. Then, it is straightforward to assume that, with more sequences involved in an alignment, the number of possibilities grows even faster.

3. Previous Work

3.1. Sequence Alignment and Evolutionary Computation. Evolutionary Computation (EC) has been previously used

Table 1: Number of possible alignments for two sequences; **m** and **n** are the respective sizes of two given sequences.

m, n	No. of possible alignments
1,1	3
2,2	13
3,3	63
4,4	321
5,5	1683
6,6	8989
7,7	48639
8,8	265729
9,9	1462563
10,10	8097453

in the problem of multiple sequence alignment (MSA) [5], from Genetic Algorithms [9–11] to Evolutionary Programming [12, 13]. From these applications, SAGA (sequence alignment by genetic algorithm) is considered [10] the most relevant to the topic of this paper's research.

One of the main advantages of EC is to allow a good separation between the optimization process and the evaluation criterion (objective function). It is the objective function that defines the aim of any optimization procedure and in the case of sequence alignment, it is also the objective function that summarizes the biological knowledge that is intended to be projected into the alignment.

3.1.1. Objective Functions. An alignment is considered to be *correct* if it reflects, at least in the case of DNA, the *evolutionary history* of the species of the sequences being aligned. But, at the time of assessing the quality of an alignment, such evolutionary information is not frequently available, or even more, not known. It may also be the case that aligning a set of sequences is an intermediate step to produce an evolutionary hypothesis. Hence, alternatives must be sought, and measures of sequence similarity are an useful option. It is assumed that similar sequences share the same evolutionary origin [14], as long as the level of identity is outside the twilight zone (more than 30% identity over 100 positions). Nevertheless, to assess sequence homology by similarity has also been questioned [15, 16].

Existing measures of similarity are obtained using substitutions matrices ([17] for proteins). A substitution matrix assigns a cost for each possible substitution or conservation accordingly to the probability of occurrence, computed from data analysis. In this approach insertions and deletions are penalized (gap penalty). The most common scheme for that purpose is giving a cost for gap opening and gap extension (*affine gap penalties* model), in order to favor alignments with smaller numbers of indels (each gap can be regarded as an indel event). The main disadvantage of these substitutions matrices is that they are intended to rate the similarity between two sequences at a time only, and in order to extend them to multiple sequences, it is common to find that they are scaled by adding up each pairwise similarity to obtain the score for the multiple sequence alignment [5].

Every objective function defines a mathematical optimum (or a set of them), which is not necessarily the same as the biological optimum that is sought when aligning genetic sequences. This biological optimum can be said that arises as a consequence of the evolutionary history of the sequences in the alignment. An objective function is only as good as its mathematical optimum resembles the biological one. In order to make this two optima converge, biological knowledge must be integrated to the objective function [5].

SAGA [10] was used to optimize two different objective functions. A brief description of them are given as follows.

Weighted Sums of Pairs. Weighted Sums of Pairs is the objective function used by MSA [18]. The sums-of-pairs principle associates a cost to each pair of aligned codifications in each column of an alignment (substitution cost) and another, similar cost to the gaps (gap cost). The sum of these costs yields the global cost of the alignment. Major variations involve using (1) different sets of costs for substitutions (PAM Matrices [17], BLOSUM tables [19]), (2) different schemes for the cost of gaps (*quasinatural* and *natural* [20]), and (3) different sets of weights associated with each pair of sequences due to evolutionary distance [21].

SAGA was first used to optimize the sums of pairs with quasinatural gap penalties.

COFFEE Score. COFFEE stands for *Consistency-Based Objective Function For alignment Evaluation* and is a measure of the level of consistency between multiple alignments of a set of sequences and a library of all possible pairwise alignments of the same set of sequences. Evaluation is made by comparing each pair of aligned residues observed in the multiple alignments with the list of residue pairs that constitute the library. The consistency score is equal to the number of pairs of residues that are found simultaneously in the multiple alignment and in the library, divided by the total number of pairs observed in the multiple sequence alignment [5].

The main difference between the COFFEE function and the Weighted Sum of Pairs is the use of the library instead of the substitution matrix.

4. GLOCSA—A New Objective Function

The Global Criterion for Sequence Alignment (GLOCSA) is a new proposed function to assess the quality of multiple sequence alignments of DNA. It has been build from the ground up with simplicity and a global approach in mind. By global it is understood that it rates the alignment as a whole, that is, all sequences considered simultaneously, not taking pairs of sequences to score their corresponding alignment. It also takes into account the gaps, seeking to favor parsimony.

GLOCSA is composed of three individual criteria: *Mean Column Homogeneity (MCH)*, *Reciprocal of Gap Blocks (RGB)* and *Columns Increment (CI)*. These are combined in a polynomial with a set of corresponding weights (w_{mch}, w_{rgb}, and w_{ci}). These weights are set by default to the values shown in Table 2. This default values were determined

Table 2: GLOCSA weights.

w_{mch}	= 1000
w_{rgb}	= 20
w_{ci}	= −20

Table 3: Nucleic acid codifications supported.

A	Adenosine
C	Cytosine
G	Guanine
T	Thymine
R	G A (puRine)
Y	T C (pYrimidine)
K	G T (Ketone)
M	A C (aMino group)
S	G C (Strong interaction)
W	A T (Weak interaction)
B	G T C (not A) (B comes after A)
D	G A T (not C) (D comes after C)
H	A C T (not G) (H comes after G)
V	G C A (not T, not U) (V comes after U)
N	A G C T (aNy)
−	Gap
?	Any base or gap

empirically, adjusting them to assign better scores to better alignments using a set of artificial examples and some real-world alignments:

$$\text{GLOCSA} = w_{\text{mch}}\text{MCH} + w_{\text{rgb}}\text{RGB} + w_{\text{ci}}\text{CI}. \tag{4}$$

The main problem faced when scoring alignments is that the exact evolutionary history of the involved sequences is never known. Theories can be stated about which alignment reflects the more plausible or probable evolutionary history (which is what produces the differences in the sequences) but certainty cannot be guaranteed.

Compared to the other schemes of sequence alignment evaluation rating them on a pair basis, such as *weighted sum of pairs* [18], GLOCSA has the advantage of rating the whole alignment at a time (with the *Mean Column Homogeneity* criterion). It also has the advantage of considering parsimony, favoring more concentrated *gaps* (with *Reciprocal of Gap Blocks*) and smaller alignment matrices (with *Columns Increment*).

At the moment it is intended to rate only multiple sequences of DNA composed of the standard IUB/IUPAC codifications for nucleic acids, shown in Table 3.

To score an alignment of multiple sequences, a matrix with C columns and S lines is considered, where C is the maximum number of positions in a sequence, and S is the number of sequences in the alignment. Initially, to perfectly fit all the sequences in the matrix, gap positions are appended ("−") at the end of the shorter sequences.

4.1. Mean Column Homogeneity. In the alignment matrix each position is represented in a column, and the column homogeneity has the purpose of rating the grade of diversity in the elements of a given position, scoring higher the more homogeneous columns.

The basic idea is that the occurrences of each of the four bases in a column are counted. A, C, G, and T are counted with a weight of 1.0 while polymorphisms are counted as an equal fraction of a unit for each base they represent (e.g., A counts 1.0 for A while R is either G or A, so it counts 0.50 for G and 0.50 for A). Gaps are also counted, with a unit for each. Using these counts the column homogeneity for each column is computed.

The count of bases and gaps are computed in wc_{jt} $\forall 0 \leq t \leq 4$, where t is the index for a base or gap which is being counted and j is the column. These weighted counts are the result of adding up to wc_{jt} the corresponding weight (shown in Table 4) for the codification of each sequence in the column. This can be expressed as,

$$wc_{jt} = \sum_i T_w(t, \text{am}(i, j)), \tag{5}$$

where $\text{am}(i, j)$ is a function that retrieves the codification in the sequence i at column j of the alignment, and the function $T_w(t, P_c)$ looks up the weight associated with the base t and the codification P_c (in this case P_c is given by $\text{am}(i, j)$) in Table 4.

After counting, the column homogeneity of a given column is computed using the following formula:

$$\text{CH}_j = \frac{\sum_{t=0}^{3} (wc_{jt})^2}{\left(\sum_{t=0}^{4} wc_{jt}\right)^2}. \tag{6}$$

It is to be noted that in the numerator of the fraction only the four bases are considered (A, C, G, and T indexed by 0, 1, 2, and 3), and in the denominator the gap (−, indexed by 4) is considered along with the bases. This is considered in order to penalize the insertion of gaps, assuming that as the gaps are not counted in the numerator but they are counted in the denominator, the column homogeneity value decreases when there are more gaps.

In the case that a position in a sequence has a ? codification, that position for that sequence is discarded (as it was not observed) for the computing of that column homogeneity value. This is because a ? implies that in that position the sequence has no information.

A special consideration is taken when all the elements in a column are gap codifications (−) in that case the column homogeneity is given a value of zero, to penalize the existence of such columns.

When the column homogeneity value for all the columns has been computed, the mean value is obtained and that is the *Mean Column Homogeneity*.

This criterion gives higher scores to more homogeneous columns, penalizing diversity of bases in a column (as shown in the examples of Table 5).

4.2. Reciprocal of Gap Blocks. The gap codifications ("−") which are contiguous are grouped into blocks, and the

Table 4: Base count weights matrix.

t		A	C	G	T	R	Y	K	M	S	W	B	D	H	V	N	–
0	A	1	0	0	0	1/2	1/2	0	1/2	0	1/2	0	1/3	1/3	1/3	1/4	0
1	C	0	1	0	0	0	0	0	1/2	1/2	0	1/3	0	1/3	1/3	1/4	0
2	G	0	0	1	0	1/2	0	1/2	0	1/2	0	1/3	1/3	0	1/3	1/4	0
3	T	0	0	0	1	0	1/2	1/2	0	0	1/2	1/3	1/3	1/3	0	1/4	0
4	–	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table 5: Column Homogeneity evaluation examples.

	Column												
	0	1	2	3	4	5	6	7	8	9	10	11	12
seq0	A	A	A	A	A	A	A	A	A	A	–	A	A
seq1	A	A	A	A	A	A	A	A	A	A	–	–	G
seq2	A	A	A	A	A	A	A	A	A	G	–	–	–
seq3	A	A	A	A	A	A	A	A	A	G	–	–	–
seq4	A	A	A	A	A	A	A	A	G	T	–	–	–
seq5	A	A	A	A	A	A	A	A	G	T	–	–	–
seq6	A	A	A	A	A	A	A	G	T	T	–	–	–
seq7	A	A	A	A	A	A	G	G	T	C	–	–	–
seq8	A	A	–	A	G	G	T	T	C	C	–	–	–
seq9	A	–	–	G	G	T	C	T	C	C	–	–	–
CH	1.00	0.81	0.64	0.82	0.68	0.66	0.52	0.44	0.28	0.26	0.00	0.01	0.02

reciprocal of the number of gap blocks is calculated, as shown in the next equation:

$$\mathrm{RGB} = \frac{1}{\mathrm{GB}}, \tag{7}$$

where GB is the number of gap blocks in the alignment. If there are no gap blocks, the Reciprocal of Gap Blocks criterion is given a value of 1.0.

This criterion serves the purpose of rewarding the alignments where the gap codifications are located in a more concentrated manner, that is, where there are fewer larger blocks of gap codifications rather than more blocks of smaller length. Fewer blocks imply less evolutionary events to be explained and a more parsimonious alignment.

In Tables 6, 7, and 8 three alignments of a hypothetical set of sequences are shown. The three alignments have the same number of "–", but the example in Table 6 has them in 3 blocks, the example in Table 7 in 2 blocks, and finally the example in Table 8 in just 1 block, a difference which is noticeable in the reciprocal gap blocks criterion, and thus favoring the alignment which implies less evolutionary events (parsimony).

4.3. Columns Increment. Inserting gaps to align a set of sequences is common, and the number of columns increases. *Columns Increment* is the ratio of this augmentation, defined by

$$\mathrm{CI} = \frac{C}{C_0} - 1, \tag{8}$$

where C is the number of columns after aligning, and C_0 the number of columns before aligning, which is equivalent to the number of nucleotides of the longest sequence.

An example of a hypothetical set of sequences for which two different alignments are shown in Tables 9 and 10. Each alignment has a different value for the *Columns Increment* criterion. A smaller alignment is preferred because a smaller matrix probably implies less evolutionary events (parsimony).

5. GGGA—a GA Using GLOCSA

Having a new objective function to evaluate the quality of multiple sequence alignments, and considering the complexity of the problem (as it is explained in Section 2.2.1), using a genetic algorithm to optimize alignments and its GLOCSA score was considered a viable option.

GGGA, *GLOCSA-Guided Genetic Algorithm* is the Genetic Algorithm implemented to optimize the GLOCSA value. GGGA is a variant of the *Simple Genetic Algorithm* where a custom representation is proposed, along with a specific mutation operator. There is no crossover operator, selection is performed by tournament, and elitism is used.

The initialization of the population is done using the mutation operator and a seed alignment which is an input to the algorithm. To produce each individual of the next generation, an individual is selected from the previous generation, using the tournament selection operator and then submitted to the mutation operator to generate the new individual (under a mutation probability).

Table 6: Alignment to exemplify the *Reciprocal of Gap Blocks* criterion. RGB = 0.33.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
seq0	A	A	A	A	G	G	C	A	T	C	A	T	C	A	T	C	A	G	G	A	A	A	A
seq1	A	A	A	A	G	G	—	—	—	C	—	—	—	A	—	—	—	G	G	A	A	A	A

Table 7: Alignment to exemplify the *Reciprocal of Gap Blocks* criterion. RGB = 0.50.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
seq0	A	A	A	A	G	G	C	A	T	C	A	T	C	A	T	C	A	G	G	A	A	A	A
seq1	A	A	A	A	G	G	—	—	—	—	—	—	C	A	—	—	—	G	G	A	A	A	A

5.1. Representation of Individuals. Each individual in the population represents a possible alignment. The alignment matrix (described in Section 4, e.g., in Table 11) used to rate an alignment with GLOCSA is the base for the representation of individuals. But not everything in the matrix is necessary to reconstruct any given alignment. Therefore it is processed to obtain a much more manageable representation.

Since the solution space explored by the algorithm consists of the possible alignments of a given set of sequences which do not change, the only piece of information which is necessary to represent any alignment, is the location of every gap codification. Furthermore, if gap codifications are grouped into gap blocks (groups of contiguous gap codifications), the position and size of these blocks are the only information needed to reconstruct an alignment.

If the bases in every sequence of the alignment are indexed with consecutive numbers, starting from 0 for the first base to ease its implementation, the position of the gap blocks can be determined by the base index it precedes.

Thus, the alignment can be represented by having for each sequence a list of the positions and sizes of every gap block in them, that is, each gap block represented as two nonnegative integers (position and size). As a simple illustrative example the alignment matrix of Table 11 is transformed to its corresponding representation in Table 12. In this example, the sequence 0 has only one gap block of size 2, before the *A* with index 2 (the third one), hence the list of gap blocks for this sequence only has one element which is $[2, 2]$; sequence 1 has two gap blocks $[2, 2]$, $[3, 1]$; sequence 2 has only one $[0, 2]$, the two gap codifications at the end of the sequence were appended to fit it in the alignment matrix; so there is no need to include them in the representation (trailing gaps are a consequence of the different lengths of the sequences); sequence 3 has just one gap block $[3, 3]$.

5.2. Mutation Operator and Suboperators. The mutation operator is basically in charge of changing the gap codification appearances in the alignment represented by an individual, in order to explore the solution space. It works with a mutation probability, which determines the number of expected mutations in an individual when the operator is applied to it. As it is more manageable to refer to the mutation probability in terms of the number of mutations expected per individual, as it is more informative in the context of the problem, this approach will be used in the results.

For each mutation operation five types of changes to the gap codification appearances are proposed: insertion of new gap blocks, increment of the size of a gap block, decrease of the size of a gap block, shift of positions of gap blocks and deletion of a gap block. These five types of changes are denominated *suboperators*, and the selection of which one will be applied is determined by its individual probability, dynamically adapted throughout the generations. These suboperators were selected because in the opinion of the authors they make the algorithm capable of searching the solution space in a relatively efficient way. A crossover operator (interchanging entire sequences between alignments) was also considered but was discarded in early stages because it gave no apparent advantage to the algorithm.

It is noteworthy that while performing these changes to the alignments no penalization is done other than the modification in the GLOCSA score these changes imply.

5.2.1. Insertion Suboperator. This suboperator chooses randomly a sequence and inserts a gap block in it. The size of the new gap block is also random, but with an exponential distribution with mean fitted from the gap block sizes in the seed alignment.

The size of the new gap blocks to insert is biased toward small sizes; this is because large gap blocks are not very common, but still exist.

5.2.2. Increment Suboperator. The Increment Suboperator chooses a sequence at random and an existing gap block from it, increasing in one unit its size. If the selected sequence does not have any gap block at all, this operator leaves the sequence without change.

5.2.3. Decrease Suboperator. As the previous operator, it chooses randomly a sequence and a gap block from it, whose size will decrease by one; if the size is 1 gap codification, this operator deletes the gap block totally. Again if the selected sequence does not have any gap block at all, it remains unchanged.

5.2.4. Shift Suboperator. In a sequence chosen at random, this operator selects first a gap block in it, then a position is selected randomly in that sequence; if a gap block exists in that position, the sizes of them are interchanged. If there

Table 8: Alignment to exemplify the *Reciprocal of Gap Blocks* criterion. RGB = 1.0.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
seq0	A	A	A	A	G	G	C	A	T	C	A	T	C	A	T	C	A	G	G	A	A	A	A
seq1	A	A	A	A	G	G	C	—	—	—	—	—	—	—	—	—	A	G	G	A	A	A	A

Table 9: Alignment to exemplify the *Columns Increment* criterion. In this case, the number of columns remains the same after aligning. CI = 0.

	0	1	2	3	4	5	6	7	8
seq0	A	T	C	A	T	C	A	T	C
seq1	A	T	C	A	T	C	A	T	C
seq2	A	T	C	A	T	C	A	T	C

is not a gap in the selected position, the position of the first selected gap is set to the other position. If the selected sequence does not have any gap block at all, no modification is done. This operator mixes information within a given sequence in a single alignment (individual) it does not recombine information from two individuals as a crossover operator.

5.2.5. Deletion Suboperator. This operator selects randomly a sequence and then a gap block. This gap block is completely deleted from the list of gap blocks. If no gap block exists in the selected sequence, it remains without any change.

5.2.6. Adaptation of Mutation Suboperators Probability. The probability of applying each of the subopertors is dynamically adapted as the generations pass; it is changed accordingly to their effect in the GLOCSA score of the alignments represented by the individuals.

This adaptation is done once at the end of every generation, and the procedure is as follows.

For the first generation of the genetic algorithm the five suboperators have the same probability, each with 0.20 of probability of being used. Every time the mutation operator is applied, in a *record* are stored the GLOCSA scores before and after the mutation and a vector which represents the use count of each suboperator (e.g., in Table 13).

After generating the entire new population, the *attributed difference by suboperator* (dSO) is computed by dividing the suboperators use count by the total number of mutations performed, and then multiplying it by the difference between the after and before scores of GLOCSA. This is shown in (9), where dSO_s is the *attributed difference* for a given suboperator s, sOUC_s is the use count for suboperator s, tM is the total number of mutations performed ($\text{tM} = \sum \text{sOUC}_s$), and aS and bS are the GLOCSA scores after and before the mutation suboperators action:

$$\text{dSO}_s = \left(\frac{\text{sOUC}_s}{\text{tM}}\right)(\text{aS} - \text{bS}) \quad \forall s = \{\text{mutation suboperators}\}. \tag{9}$$

Then, the *attributed difference by suboperator* for all the records is summed up in the *total attributed difference by suboperator* (tDSO, see (10)):

$$\text{tDSO}_s = \sum_R \text{dSO}_s \quad \forall s = \{\text{mutation suboperators}\}. \tag{10}$$

These tDSO_s values are then normalized by dividing them by the largest absolute value of them:

$$\text{tDSO}_s = \frac{\text{dSO}_s}{\max(\{|\text{dSO}_s|\ \forall s\})}. \tag{11}$$

Afterward, the probability (p_s) of each suboperator is added $p_S \cdot \text{SCh} \cdot \text{tDSO}_s$:

$$p_s = p_s + (p_S \cdot \text{SCh} \cdot \text{tDSO}_s), \tag{12}$$

where SCh is a constant which sets how big the steps of the adaptation are. It was set for the experiments to the value of 0.10.

Finally the values of p_s are scaled to make the sum of all the probabilities equal to 1:

$$p_s = \frac{p_s}{\sum_S p_s}. \tag{13}$$

5.3. Population Initialization. To initialize the population a given alignment is used as a starting point. The individuals of the initial generation are mutations of it, obtained by applying the mutation operator. The mutation operator is applied discarding the adaptation stage; therefore the five suboperators have the same probability while initializing the population.

6. Tests with Real Data

6.1. Test Bench. To test the ability of GGGA to optimize the GLOCSA scoring function, three multiple sequence alignment problems were proposed, which are shown in Table 16 along with relevant information. The set of sequences *exmpl17* is a subset of *exmpl19*; the two shortest sequences were eliminated, thus presumably reducing the complexity of the alignment.

6.2. GA Test Parameters. Each set of sequences was first aligned with *MUSCLE* [3], a popular progressive alignment tool. The resulting alignment was seeded as a starting point for the initialization of the population; thus the aim of the test is to see if further improvements to the alignment of MUSCLE can be performed, guided by the GLOCSA scoring function.

The genetic algorithm for all the experiments was run over 1000 generations with a population of 100, with

Table 10: Alignment to exemplify the *Columns Increment* criterion. Here, the number of columns increased to **6** after aligning. CI = 0.66.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
seq0	A	T	C	A	T	C	—	—	—	A	T	C	—	—	—
seq1	A	T	C	—	—	—	A	T	C	A	T	C	—	—	—
seq2	A	T	C	—	—	—	—	—	—	A	T	C	A	T	C

Table 11: Alignment matrix example.

Sequence #								
0	G	A	—	—	A	C	A	G
1	G	A	—	—	A	—	A	G
2	—	—	T	C	A	C	—	—
3	G	A	T	—	—	—	A	G

Table 12: Alignment matrix example—GA representation.

Sequence #	
0	[2, 2]
1	[2, 2], [3, 1]
2	[0, 2]
3	[3, 3]

Table 13: Sample records used for adaptation.

	Suboperator					GLOCSA score	
	0	1	2	3	4	Before score	After score
					. . .		
15	0	1	1	0	0	34.9	34.5
16	1	0	1	0	0	34.9	35.1
					. . .		

Table 14: Default Test Parameters. For each experiment these default parameters were used.

GLOCSA weights	$w_{\text{mch}} = 1000$ $w_{\text{rgb}} = 20$ $w_{\text{ci}} = -20$
Number of generations	1000
Individuals in population	100
Elite individuals	5
Individuals in tournament	5

Table 15: Test Experiments. Using the default test parameters listed in Table 14, the number of expected mutations were tested in the range of [0.1,3] with increments of 0.1, performing 30 experiments for each configuration, for each alignment in the test bench.

Alignment	No. of expected mutations	Experiments performed
exmpl19	0.1	30
	0.2	30
	⋮	⋮
	2.9	30
	3.0	30
exmpl17	0.1	30
	⋮	⋮
	3.0	30
exmpl29	0.1	30
	⋮	⋮
	3.0	30

5 individuals of elitism. Selection is performed using a tournament of 5 individuals.

The GLOCSA Scoring function used as the objective function has the default weights defined in Table 2.

The rate of the mutation was in the range of [0.1, 3] number of expected mutations with increments of 0.1. For each of this combination of values (the previously mentioned parameters and the number of expected mutations) 30 experiments were performed.

The only parameter tested within a range was the number of expected mutations. Because it was considered the most important and performing a parameter sweep across all parameters would have been too computationally expensive.

6.3. Experiments Results. Results of these experiments are shown in Figures 1, 2, and 3, using box and whiskers plots; the box has lines at the lower quartile, median, and upper quartile values; whiskers extend from each end of the box to the minimum and maximum scores obtained.

It was observed that the GLOCSA-Guided Genetic Algorithm always improved (at least slightly) the solution previously found by MUSCLE (the score of the initial alignment is the lower range of the GLOCSA Scores in the chart), and as expected the amount of improvement is strongly related with the *number of expected mutations*; lower (near zero) and higher (close and beyond three mutations per individual) numbers of expected mutations produce less improvements while values in or in the vicinity of the range of [0.5, 1.0] produce the higher optimization values. This trend is certainly due to the exploration/exploitation balance, with fewer mutations there is not enough exploration, and with too many mutations there is excess exploration in detriment of exploitation.

It is important to notice that the range of the GLOCSA values in Figures 1, 2, and 3 is different for each of them. This is because GLOCSA values are relative to the alignment they are scoring, prominently the *column homogeneity*. In particular this criterion would have a value of 1000 (multiplied by its default weight) when aligning a set of copies of a single sequence (every column will score 1.0).

While the improvements for the alignments exmpl17 and exmpl19 are about the same, for exmpl29 these are bigger.

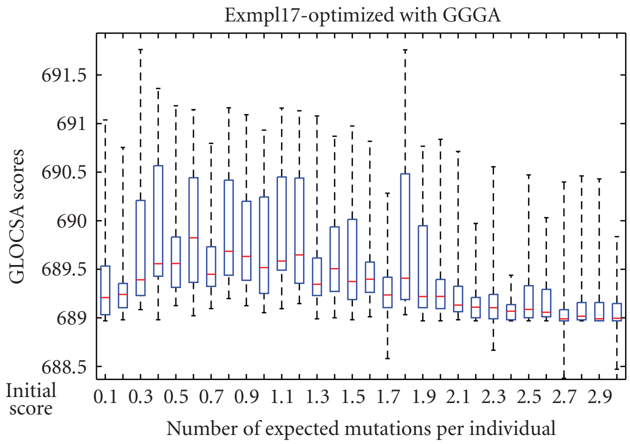


Figure 1: Box and whisker plot of the results of the experiments of exmpl17.

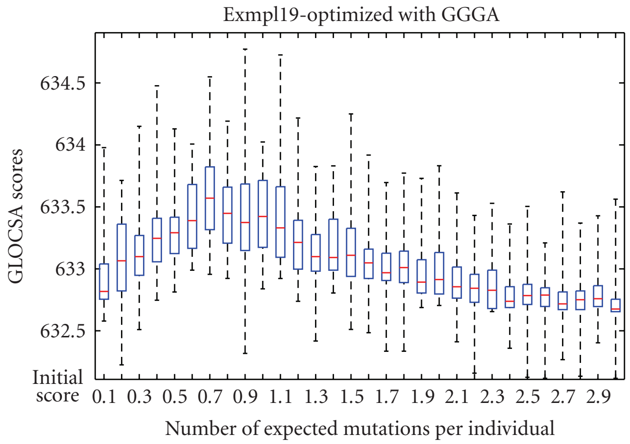


Figure 2: Box and whisker plot of the results of the experiments of exmpl19.

Table 16: Test Bench.

	No. of seq.	max. no. of pos.	Total no. of bases
exmpl19	19	649	10908
exmpl17	17	649	10149
exmpl29	29	245	6150

This is explained by the fact that exmpl29 is less complex than exmpl17 and exmpl19, which are about the same difficulty (exmpl17 easier that exmpl19, as the sequences in the first are a subset of those in the second, but not enough to make a noticeable difference).

In Table 17, the mean elapsed times for the 30 experiments of each alignment are shown. All the experiments were performed in a personal computer with an Intel Pentium D CPU 2.80 GHz processor (though not using its two cores for a single experiment) with 2 GB in RAM.

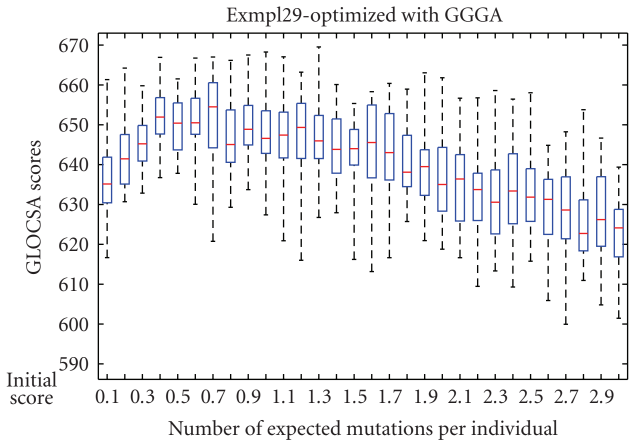


Figure 3: Box and whisker plot of the results of the experiments of exmpl29.

Table 17: Tests elapsed times.

Test	Elapsed time (minutes)
exmpl19	12.37
exmpl17	11.25
exmpl29	7.03

7. Conclusions

For the assessment of the quality of multiple sequence alignments, scoring functions have been previously defined, but in the opinion of the authors, the results obtained so far are not satisfactory enough, and therefore the GLOCSA measure was devised. It aims to be considered an alternative scoring function for multiple sequence alignments, one with the advantages of being simple, of rating the whole alignment at once, and being parsimonious.

Given the complexity of the problem of multiple sequence alignment, the techniques of Evolutionary Computation—Genetic Algorithms in particular—seem useful for optimizing this new proposed scoring function. Although it is not efficient, compared with the fast progressive alignment heuristics (e.g., MUSCLE, a run of it over the larger alignment tested in this work takes less than 4.5 seconds in the same machine) the GGGA has the ability to optimize GLOCSA as the objective function. In the light of performing it as a refinement over previously aligned data with more efficient methods (as in the test experiments where MUSCLE alignments were inserted as starting points) is a promising application.

Even though a set of sequences can be aligned from scratch optimizing its GLOCSA score with the GA, it would be too time consuming. Then, an initial starting point given by another tool seems like a good idea, in the light that progressive alignment delivers good results, but these can be further refined. The seed alignment can be the product of any alignment tool, which gives this approach additional flexibility.

8. Future Work

Currently GLOCSA only rates DNA sequence alignments; the next step would be to extend its application scope to protein sequences.

GLOCSA as a quality measure has been validated empirically, but tests to assess its reliability are still pending. This will be done with the aid of defined sets of reference alignments such as BALiBASE (protein sequence alignments) [22, 23] and the GLOCSA-Guided Genetic Algorithm, thus resulting in the assessment of both, the scoring function and the genetic algorithm implementation.

A new crossover operator (across columns) will also be implemented in the Genetic Algorithm, and its adaptation mechanism will be explored further. The performance of the Genetic Algorithm will be compared to a Random Search, to see how much the evolutionary nature of the algorithm is contributing to the results.

Acknowledgments

The authors would like to thank CONACYT under the project 61507 and SIBA-UNIBIO (UNAM) for the funding that made this work possible.

References

[1] F. C. Bernstein, T. F. Koetzle, G. J. B. Williams, et al., "The protein data bank: a computer based archival file for macromolecular structures," *Journal of Molecular Biology*, vol. 112, no. 3, pp. 535–542, 1977.

[2] D. A. Benson, I. Karsch-Mizrachi, D. J. Lipman, J. Ostell, and D. L. Wheeler, "Genbank," *Nucleic Acids Reseach*, vol. 34, pp. D16–D20, 2006.

[3] R. C. Edgar, "Muscle: multiple sequence alignment with high accurracy and high throughput," *Nucleic Acids Reseach*, vol. 32, no. 5, pp. 1792–1797, 2004.

[4] W. S. Klug, M. R. Cummings, and C. Spencer, *Concepts of Genetics*, Benjamin Cummings, Essex, UK, 2005.

[5] "Using genetic algorithms for pairwise and multiple sequence alignments," in *Evolutionary Computation in Bioinformatics*, G. B. Fogel and D. W. Corne, Eds., chapter 5, Morgan Kaufman, San Francisco, Calif, USA, 2003.

[6] B. Haubold and T. Wiehe, *Introduction to Computational Biology: An Evolutionary Approach*, Birkhäuser, Basel, Switzerland, 2007.

[7] S. B. Needleman and C. D. Wunsch, "A general method applicable to the search for similarities in the amino acid sequence of two proteins," *Journal of Molecular Biology*, vol. 48, no. 3, pp. 443–453, 1970.

[8] T. F. Smith and M. S. Waterman, "Comparison of biosequences," *Advances in Applied Mathematics*, vol. 2, no. 4, pp. 482–489, 1981.

[9] M. Ishikawa, T. Toya, and Y. Tokoti, "Parallel iterative aligner with genetic algorithm," in *Proceedings of the 13th International Conference on Artificial Ingelligence and Genome Workshop*, pp. 84–93, 1993.

[10] C. Notredame and D. G. Higgins, "SAGA: sequence alignment by genetic algorithm," *Nucleic Acids Research*, vol. 24, no. 8, pp. 1515–1524, 1996.

[11] C. Notredame, E. A. O'Brien, and D. G. Higgins, "RAGA: RNA sequence alignment by genetic algorithm," *Nucleic Acids Research*, vol. 25, no. 22, pp. 4570–4580, 1997.

[12] K. Chellapilla and G. Fogel, "Multiple sequence alignment using evolutionary programming," in *Proceedings of the IEEE Congress on Evolutionary Computation*, vol. 1, p. 452, Washington, DC, USA, July 1999.

[13] L. Cai, D. Juedes, and E. Liakhovitch, "Evolutionary computation techniques for multiple sequence alignment," in *Proceedings of the IEEE Conference on Evolutionary Computation (ICEC '00)*, vol. 2, pp. 829–835, 2000.

[14] C. Sander and R. Schneider, "Database of homology-derived protein structures and the structural meaning of sequence alignment," *Proteins: Structure, Function and Genetics*, vol. 9, no. 1, pp. 56–68, 1991.

[15] J. I. Davis and J. J. Doyle, "Homology in molecular phylogenetics: a parsimony perspective," in *Molecular Systematics of Plants II*, pp. 101–131, Kluwer Academic Publishers, Boston, Mass, USA, 1998.

[16] H. Ochoterena, "Homology in coding and non-coding DNA sequences: a parsimony perspective," *Plant Systematics and Evolution*.

[17] M. O. Dayhoff, *Atlas of Protein Sequence and Structure*, National Biomedical Research Fundation, Washington, DC, USA, 1978.

[18] D. J. Lipman, S. F. Altschul, and J. D. Kececioglu, "A tool for multiple sequence alignment," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 86, no. 12, pp. 4412–4415, 1989.

[19] S. Henikoff and J. G. Henikoff, "Amino acid substitution matrices from protein blocks," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 89, no. 22, pp. 10915–10919, 1992.

[20] S. F. Altschul, "Gap costs for multiple sequence alignment," *Journal of Theoretical Biology*, vol. 138, no. 3, pp. 297–309, 1989.

[21] S. F. Altschul and D. J. Lipman, "Trees, stars, and multiple biological sequence alignment," *SIAM Journal on Applied Mathematics*, vol. 49, no. 1, pp. 197–209, 1989.

[22] J. D. Thompson, F. Plewniak, and O. Poch, "BAliBASE: a benchmark alignment database for the evaluation of multiple alignment programs," *Bioinformatics*, vol. 15, no. 1, pp. 87–88, 1999.

[23] A. Bahr, J. D. Thompson, J.-C. Thierry, and O. Poch, "BAliBASE (Benchmark Alignment dataBASE): enhancements for repeats, transmembrane sequences and circular permutations," *Nucleic Acids Research*, vol. 29, no. 1, pp. 323–326, 2001.

Hindawi Publishing Corporation
Journal of Artificial Evolution and Applications
Volume 2009, Article ID 803973, 10 pages
doi:10.1155/2009/803973

Research Article

An Evolutionary Method for Combining Different Feature Selection Criteria in Microarray Data Classification

Nicoletta Dessì and Barbara Pes

Dipartimento di Matematica e Informatica, Università degli Studi di Cagliari, Via Ospedale 72, 09124 Cagliari, Italy

Correspondence should be addressed to Barbara Pes, pes@unica.it

Received 20 November 2008; Accepted 1 June 2009

Recommended by Stephen Smith

The classification of cancers from gene expression profiles is a challenging research area in bioinformatics since the high dimensionality of microarray data results in irrelevant and redundant information that affects the performance of classification. This paper proposes using an evolutionary algorithm to select relevant gene subsets in order to further use them for the classification task. This is achieved by combining valuable results from different feature ranking methods into feature pools whose dimensionality is reduced by a wrapper approach involving a genetic algorithm and SVM classifier. Specifically, the GA explores the space defined by each feature pool looking for solutions that balance the size of the feature subsets and their classification accuracy. Experiments demonstrate that the proposed method provide good results in comparison to different state of art methods for the classification of microarray data.

1. Introduction

Microarray technologies provide an unprecedented opportunity for uncovering the molecular basis of cancer and other pathologies. Any microarray experiment assays the expression levels of a large number of genes in a biological sample. These assays provide the input to a wide variety of computational efforts aiming at defining global gene expression profiles of pathological tissues and comparing them with corresponding normal tissues. Generally, this process is carried on by selecting a small informative set of genes that can distinguish among the various classes of pathology, by choosing an appropriate mathematical model (i.e., a classifier), by estimating the parameters of the model based on a training set of samples whose classification is known in advance.

A relevant problem in microarray data classification, and in machine learning in general, is the risk of "overfitting" that arises when the number of training samples is small and the number of attributes or features (i.e., the genes) is comparatively large. In such a situation, we can easily learn a classifier that correctly describes the training data but performs poorly on an independent set of data. In order to improve the performance of learning algorithms [1–3], it is of paramount importance to reduce the dimensionality of the data by deleting unsuitable features [4].

Indeed, the selection of an optimal subset of features by exhaustive search is impractical and computationally intensive when the number of attributes is high, as it is for microarray data, and a proper learning strategy must thus be devised. The relevance of good feature selection methods has been discussed by [5], but the recommendations in literature do not give evidence for a single best method for either the feature selection or the classification of microarray data [6].

Recent studies on evolutionary algorithms (EAs) have revealed their success on microarray classification. Particularly, these methods not only converge to high quality solutions, but also search for the optimal set of features on complex and large spaces of possible genes [7, 8]. One of the most influential factors in the quality of the solutions found by an evolutionary algorithm is a suitable definition of the search space of the potential solutions.

This paper proposes an evolutionary approach that combines results from different ranking methods to assess the merits of the individual features by evaluating their strength of class predictability. This gives us the ability to find feature subsets with small size and high classification performance that we call feature pools (FPs). Each FP is assumed as an initial set of informative genes and is further refined by a wrapper approach involving a genetic algorithm (GA) and SVM classifier. Specifically, the GA explores the space defined by each FP looking for solutions that balance the size of the feature subsets and their classification accuracy.

Our extensive experiments on a public microarray dataset, namely the Leukemia dataset (Available at http://www.broad.mit.edu/cgi-bin/cancer/publications/.), demonstrate that the proposed approach is highly effective in selecting features and outperforms some proposed methods in literature.

The rest of the paper is organized as follows. In Section 2, we provide background information on microarray data analysis and discuss some related works. Section 3 illustrates the rationale for the proposed approach and describes the adopted evolutionary algorithm. We provide our extensive results and their interpretations in Section 4. Section 5 contains a detailed discussion as well a comparison with the results of different state-of-art methods from the literature. Finally, in Section 6 we conclude with some final remarks and suggest future research directions.

2. Background and Related Work

The "curse of dataset sparsity" [9, 10] is a major concern in microarray analysis, since microarray data include a large number of gene expression values per experiment (several thousands of features), and a relatively small number of samples (a few dozen of patients). Giving a large number of features to learning algorithms can make them very inefficient for computational reasons. In addition, irrelevant data may confuse algorithms making them to build inefficient classifiers while correlation between feature sets causes the redundancy of information and may result in the counter effect of overfitting [5]. Therefore, it is more important to explore data and utilize independent features to train classifiers, rather than increase the number of features we use.

The problem of feature selection has received a thorough treatment in machine learning and pattern recognition. Most of the feature selection algorithms approach the task as a search problem, where each state in the search specifies a distinct subset of the possible features [11]. The search problem is combined with a criterion in order to evaluate the merit of each candidate subset of features. There are a lot of possible combinations between each search procedure and each feature evaluation measure [12].

Based on the evaluation measure, feature selection algorithms can broadly fall into the *filter model* and the *wrapper model* [13]. The filter model relies on general characteristics of the training data to select predictive features (i.e., features highly correlated to the target class) without involving any mining algorithm. Conversely, the wrapper model uses the predictive accuracy of a predetermined mining algorithm to give the quality of a selected feature subset, generally producing features better suited to the classification task at hand. However, it is computationally expensive for high-dimensional data [11, 13]. As a consequence, the filter model is often preferred in gene selection due to its computational efficiency.

Hybrid and more sophisticated feature selection techniques have been explored in recent microarray research efforts [14]. Among the most promising approaches, evolutionary algorithms have been applied to microarray analysis in order to look for the optimal or near optimal set of predictive genes on complex and large search spaces [15]. For example, references [16–18] address the problem of gene selection using a standard genetic algorithm which evolves populations of possible solutions, the quality of each solution being evaluated by an SVM classifier. Genetic algorithms have been employed in conjunction with different classifiers, such as k-Nearest Neighbor in [19] and Neural Networks in [20]. Moreover, evolutionary approaches enable the selection problem to be treated as a multiobjective optimization problem, minimizing simultaneously the number of genes and the number of misclassified examples [18, 21].

3. The Evolutionary Method

Most of the evolutionary algorithms approach the task of microarray classification as a search problem where each state in the search specifies a distinct subset of the possible relevant features. If the search space is too large, it is possible that the evolutionary algorithm cannot discover the most selective genes within the search space. Moreover, having too many redundant or irrelevant genes increases computational complexity and cost and degrades estimation in classification error. On the other hand, if the initial gene space is too small, it is possible that some predictive genes are not included in the search space.

Feature ranking (FR) is a traditional evaluation criterion that is used by most popular search methods for assessing individual features and assigning them weights according to their relevance to the target class. Often the top-ranked genes are selected and evaluated by search algorithms in order to find the best feature subset. Although several search strategies exist, most of them cannot be applied to microarray datasets due to the large number of genes. Furthermore FR algorithms cannot discover redundancy and correlation among genes.

These limitations suggest us to pursue a hybrid method that attempts to take advantage from the combination of FR and evolutionary algorithms by exploiting their best performance in two steps. First, different FR methods are used for ranking genes. Since it is unfeasible to search for every possible subset of genes through the search space, only the top ranked genes are considered; they provide distinct lists of ordered genes that are combined in subsets, namely feature pools, of potentially "good" features. Second, each feature pool is further reduced by a genetic algorithm (GA) that tries to discover gene subsets having smaller size and/or better classification performance.

The use of different ranking methods promotes the selection of important subsets without losing informative genes while reducing the search space for the genetic algorithm.

```
INPUT: $D$—Dataset of $N$ features
       $M$—Number of ranking methods to be considered
       Met—Ranking method
       $T$—Threshold

OUTPUT: FeaturePools—A list of $M$ sets of features

(1)  list RankedSets = { }
(2)  AllFeatures = { }
(3)  for $k = 1$ to $M$
(4)      $\text{Set}_k = \{\}$
(5)      for each feature $f_i \, \varepsilon \, D$
(6)          score = rank($f_i$, $\text{Met}_k$, $D$)
(7)          append $f_i$ to $\text{Set}_k$ according to score
(8)      end for
(9)      $\text{Set}_k$ = top ($\text{Set}_k$, $T$)
(10)     AllFeatures = AllFeatures $\cup$ $\text{Set}_k$
(11)     append $\text{Set}_k$ to RankedSets
(12) end for

(13) list FeaturePools = { }
(14) $\text{FP}_0 = \{\}$
(15) list Combinations = { }
(16) for $k = M$ to 2
(17)     Combinations = Combine($M$, $k$)
(18)     shared = CommonFeatures(RankedSets,
         Combinations)
(19)     $\text{FP}_{M+1-k}$ = shared $\cup$ $\text{FP}_{M-k}$
(20)     append $\text{FP}_{M+1-k}$ to FeaturePools
(21) end for
(22) $\text{FP}_M$ = AllFeatures
(23) append $\text{FP}_M$ to FeaturePools
```

Algorithm 1: Pseudocode describing the first step of the proposed evolutionary method.

Being hard to apply evolutionary methods directly to high-dimensional datasets [22], reduced feature pools provide the possibility of putting into practice genetic algorithms, usually effective for small or middle scale datasets, for microarray data classification. In the rest of this section, we give a description of these steps.

3.1. First Step: Ranking Genes and Building Feature Pools. Algorithm 1 describes the first step that aims to reduce the dimensionality of the initial problem by identifying pools of candidate genes to be further selected by the GA.

Firstly, the genes are ranked using M ranked methods (lines 1–8). Ranking is carried out separately by each method and results in M ranked sets of genes each of ones contains all the genes in descending order of relevance. Then, we reduce the dimensionality by considering only the T top-ranked genes from each set (line 9), where T is a fixed threshold. This process results in a list of M ranked sets (line 11).

The basic idea of our approach is to absorb useful knowledge from these M sets and to fuse their information by considering the features they share (lines 13–23). In more detail, given a positive integer k ($2 \leq k \leq M$), we build a list of all possible k-combinations of the first M integers starting from 1 (line 17). For example, if $M = 4$ and $k = 2$, the list of combinations is as follows: $\{(1,2)\,(1,3)\,(1,4)\,(2,3)\,(2,4)\,(3,4)\}$. Each integer indexes a ranked set and we use these combinations (line 18) for determining the features shared by M, $M-1, \ldots, 2$ of the M sets, respectively.

Next (lines 19–23), the shared features are employed for building a list of nested feature pools $\text{FP}_1 \subseteq \text{FP}_2 \cdots \subseteq \text{FP}_M$, where FP_1 contains the features shared by all the M sets, FP_2 the features shared by at least $M-1$ of the M sets, FP_3 the features shared by at least $M-2$ of the M sets, …, FP_{M-1} the features shared by at least 2 of the M sets. Finally, FP_M contains all the features belonging to the M sets.

3.2. Second Step: Gene Selection by GA/SVM. In the second step, we implement a wrapper model that combines GA and SVM. The latter is a popular classification technique, however other classifiers could be incorporated in our approach. To sum up, the GA selects some features as an individual and SVM evaluates them by classification, and the result is used for estimating the fitness of the individual. The possible choices of feature pools FP_i define the evolutionary search space.

Figure 1 shows the whole structure of this second step. This is carried out separately on each FP_i. At the start of the search, a population of individuals (i.e., feature subsets) is randomly initialized from the feature pool FP_i. Each individual of the current population is evaluated according to a fitness function. Each time the fitness is evaluated, an SVM classifier is built and tested on the feature subset under investigation. Then, a new population is generated by applying genetic operations (selection, crossover and mutation) and the fitness is again evaluated until a prespecified number of generations G is reached. This evolution process results in a best individual that we try to further refine by initializing from it a new population that is used as a starting point of a new evolution process. The refinement is iterated until a prespecified stopping criterion is met. When the entire round of search is completed, the final feature subset is returned.

The basic components of our GA are as follows.

3.2.1. Representation of Individuals. Generally, a genetic algorithm represents the individual as a string or a binary array. Considering the large number of genes, if we represent all the genes as a binary vector, this results in a very long chromosome. Since the pre-processing step reduces the dimensionality of initial gene set, we limit the maximum size of each individual, that is, the length of chromosome, to a predetermined parameter size $M * T$ that denotes the maximum cardinality of a feature pool. The individuals are encoded by n-bit binary vectors. If a bit is "1" it means that the corresponding feature is included in the gene subset, while the bits with value 0 mean the opposite.

3.2.2. Fitness Function. The fitness function is a key factor which affects the performance of GAs. Our aim is to define a function to scale the merit of a feature subset in terms of both classification accuracy and degree of dimensionality

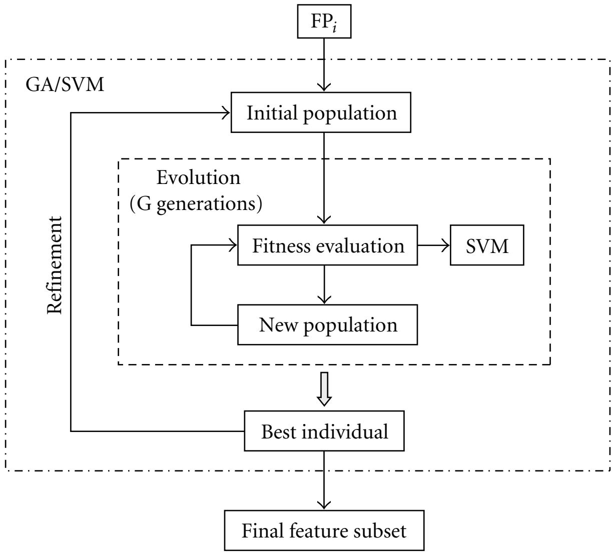


Figure 1: The architecture of the GA/SVM algorithm.

in order to see how good your approach is in situations where there is a large number of genes. The main idea is to achieve a tradeoff between the accuracy and the size of the obtained feature subsets. As a compromise between these two evaluation criteria, the fitness is defined as follows:

$$F = w \cdot C(x) + \frac{1-w}{S(x)}, \tag{1}$$

where w is a parameter between 0 and 1, x is a feature vector representing an individual, $C(x)$ is the classification accuracy of a classifier built on x, and $S(x)$ is the x size, that is, the number of genes included into x.

Here, the first term measures the weighted classification accuracy from a classifier and the second one evaluates the weighted size of the feature subset x. The parameter w is a fitness scaling mechanism for assessing the relevance of each term. Increasing the value of w will give more relevance to accuracy and reducing it will set more penalties on the size.

This multiobjective fitness makes it possible to obtain diverse solutions of high accuracy, while conventional approaches tend to be converged to a local optimum. We will analyze systematically the usefulness of the adopted function in our experiments.

3.2.3. Genetic Operators

Selection. Roulette wheel selection is used to probabilistically select individuals from a population for later breeding. The probability $P(h_i)$ of selecting the individual h_i is proportional to its own fitness $F(h_i)$ and inversely proportional to the fitness of other competing hypotheses in the current population. It is defined as follows:

$$P(h_i) = \frac{F(h_i)}{\sum_i F(h_i)}. \tag{2}$$

Crossover. We use the single point crossover, which is enough for our application. One crossover point i is chosen at random so that the first i bits are contributed by one parent and the remaining bits by the second parent.

Mutation. Each individual has a probability p_m to mutate. We randomly choose a number of n bits to be flipped in every mutation stage.

3.2.4. Stopping Criteria. A single evolution process is terminated when a predefined number of generations G is reached or an individual of maximum accuracy (100%) and minimum size (1) is obtained. The best individual produced by the evolution is iteratively refined by starting a new evolution process (Figure 1) until the fitness cannot be further improved (or a predefined number of iterations I is reached): the results show the possibility of improvement even if in few cases.

P trails of search are carried out using the GA/SVM approach previously described. The resulting gene subsets, as well as the partial results of the refinement process in each trail of search, are recorded in an archive for further analysis. All recorded gene subsets will be used in further evaluation and compared with respect to dimensionality and classification accuracy. This allows the identification of optimal subsets along with summary information such as the average classification accuracy and the average size of the gene subsets selected in different rounds of search.

4. Experimental Results

We verify the proposed method with Leukaemia [2] which is a popular public microarray dataset. Leukemia contains 72 samples among which 25 samples are collected from acute myeloid leukaemia (AML) patients and 47 samples are from acute lymphoblastic leukaemia (ALL) patients. Gene expression levels of 7129 genes are reported.

4.1. Methods and Parameters Settings. In the first step (see Section 3.1) we used the following ranking methods:

(i) information Gain (IG),
(ii) chi-squared (CHI),
(iii) symmetrical Uncert (SU),
(iv) one Rule (OR).

CHI measures the degree of independence between the feature and the target class. Inspired by information theory, IG evaluates the reduction of uncertainty (entropy) in classification prediction when knowing the feature. SU allows the discriminatory power of each feature to be found and OR operates by using a one rule classifier to evaluate each feature.

For genetic operations (see Section 3.2) the parameters were set as follows:

(i) population size: 25,
(ii) number of generations: $G = 10$, $G = 20$, $G = 30$,

(iii) probability of crossover: 1,

(iv) probability of mutation : 0.001,

(v) number of refinement iterations: $I = 10$.

SVM error estimation was by using leave-one-out cross validation (LOOCV). That is, one of the samples was left out to be a pseudotest data and the classifier was built based on all but the left out sample. This evaluation was repeated for each sample, and the estimated accuracy is a mean over all considered samples. We notice that LOOCV is a straightforward technique for estimating error rates and it is also an almost unbiased estimator.

The ranking methods and the SVM classifier were provided by the Weka library [4]. In particular, we must take account that in the Weka library SVM is trained using the SMO algorithm [23].

The evolutionary algorithm is run using GALib [24], a C++ library of genetic algorithm objects. The library includes tools for using genetic algorithms to do optimization in any C++ program using any representation and any genetic operators.

4.2. First Step. As already mentioned, the first step is done over ranking genes and, in the experiments, four ($M = 4$) ranking methods (IG, CHI, SU, OR) were used for it. First, each ranking method was applied to Leukemia and four ranked lists were generated. Then, we carried through preliminary experiments to compare the effectiveness of the considered methods.

Specifically, we ordered features according to their predictive power within each list and studied the behavior of SVM classifier on nested subsets of top-ranked features (i.e., top-2, top-4, top-8, etc.) from each list. Table 1 shows the LOOCV accuracy of SVM, respectively, by each nested subset and each ranking method. We note the similarity between results obtained with the four methods. The maximum accuracy (i.e., 98,6%) was reached by running SVM on 1024 features, except for CHI method where a peak was achieved on 32 features. We observe that when the number of selected features further increases, the accuracy does not improve, due to the inclusion of uninformative or redundant genes.

Results in Table 1 seem to suggest that no single feature selection criterion is optimal in identifying a small subset of highly discriminative features. This may be caused by the complex interactions, correlations, and redundancy between features and the biases embedded in the feature ranking criteria. On this premise, our experimental study aims to explore the effectiveness of combining useful outcomes from different methods, according to the methodology presented in Section 3.

As a first step, we cut off the $T = 20$ top ranked genes from each list, where the threshold of 20 is chosen based on a common practice in microarray studies. Table 2 shows the index of the 20 top-ranked genes (i.e., features) ordered by the relevance that each gene is assigned by each single ranking method. As we can see, some genes are shared by two or more ranking methods while some genes are specific to a single method.

Table 1: LOOCV accuracy (%) of different groups of top ranked features.

Top-ranked features	IG	CHI	SU	OR
2	93.1	93.1	93.1	91.7
4	93.1	93.1	93.1	88.9
8	93.1	93.1	93.1	94.4
10	94.4	93.1	93.1	93.1
16	94.4	94.4	94.4	95.8
20	94.4	94.4	95.8	97.2
25	95.8	97.2	97.2	95.8
32	97.2	98.6	97.2	97.2
64	95.8	97.2	97.2	97.2
128	94.4	97.2	97.2	97.2
256	97.2	97.2	97.2	97.2
512	97.2	97.2	97.2	97.2
1024	98.6	98.6	98.6	98.6
2048	98.6	98.6	98.6	98.6
4096	98.6	98.6	98.6	98.6
7129	98.6	98.6	98.6	98.6

Table 2: The 20 top-ranked genes from each ranking method.

	Top-20 IG	Top-20 CHI	Top-20 SU	Top-20 OR
1	3252	1834	1834	4847
2	4847	4847	4847	760
3	1834	1882	1882	6041
4	1882	3252	3252	1882
5	6041	6855	760	1685
6	2288	2288	2288	6376
7	760	760	6041	6855
8	6855	6041	6855	2288
9	1685	1685	1685	3252
10	1779	6376	6376	1834
11	2128	4373	2354	1779
12	6376	2128	4373	4366
13	2354	4377	4377	4328
14	4366	2354	4366	2402
15	4377	1779	2402	4196
16	4373	2402	758	1745
17	4328	1144	4328	1144
18	758	4366	1144	2020
19	1144	6281	3320	1928
20	2642	2121	2642	6347

Table 3 shows the composition of the feature pools FP_i ($i = 1, \ldots, 4$) as well as the LOOCV accuracy of the SVM classifier trained on each FP_i (*baseline model*). The letter

Table 3: FP$_i$ composition and accuracy of the corresponding baseline model.

	FP$_1$	FP$_2$	FP$_3$	FP$_4$
1	3252**r**	3252**r**	3252**r**	3252**r**
2	4847**r**	4847**r**	4847**r**	4847**r**
3	1834**r**	1834**r**	1834**r**	1834**r**
4	1882**r**	1882**r**	1882**r**	1882**r**
5	6041**r**	6041**r**	6041**r**	6041**r**
6	2288**r**	2288**r**	2288**r**	2288**r**
7	760**r**	760**r**	760**r**	760**r**
8	6855**r**	6855**r**	6855**r**	6855**r**
9	1685**r**	1685**r**	1685**r**	1685**r**
10	6376**r**	6376**r**	6376**r**	6376**r**
11	4366**r**	4366**r**	4366**r**	4366**r**
12	1144**r**	1144**r**	1144**r**	1144**r**
13		1779**b**	1779**b**	1779**b**
14		2354**b**	2354**b**	2354**b**
15		4377**b**	4377**b**	4377**b**
16		4373**b**	4373**b**	4373**b**
17		4328**b**	4328**b**	4328**b**
18		2402**b**	2402**b**	2402**b**
19			2128**g**	2128**g**
20			758**g**	758**g**
21			2642**g**	2642**g**
22				6281**y**
23				2121**y**
24				3320**y**
25				4196**y**
26				1745**y**
27				2020**y**
28				1928**y**
29				6347**y**
Accuracy	94.4%	94.4%	94.4%	98.6%

following each feature denotes the corresponding feature colour defined as follows:

(i) **r** marks the *red features*, that is, genes selected by all methods;

(ii) **b** marks the *blue features*, that is, genes selected by three methods;

(iii) **g** marks the *green features*, that is, genes selected by two methods;

(iv) **y** marks the *yellow features*, that is, genes selected by just one method.

The choice of different colours is a useful heuristic we adopted for revealing the features shared by different ranking methods.

4.3. Second Step. Starting from the different feature pools obtained in the previous step, we performed a further gene selection according to the evolutionary approach described in Section 3.2. Specifically, we studied the behavior of the proposed algorithm in four ways: with respect to the parameter w (ranging from 0.70 to 0.95), with respect to the number of generations ($G = 10, G = 20, G = 30$), with respect to the classification accuracy, and with respect to the dimensionality of the feature subset.

Table 4: Performance of GA on the feature pool FP$_1$.

w	Number of generations	Average accuracy (%)	Maximum accuracy (%)	Average size	Minimum size
0.70	10	94.2	95.8	4	3
	20	94.2	95.8	4	3
	30	93.3	95.8	3	2
0.75	10	94.4	97.2	4	3
	20	94.4	97.2	3	2
	30	93.9	97.2	3	2
0.80	10	96.4	98.6	5	4
	20	95.5	97.7	4	4
	30	95.0	97.2	4	2
0.85	10	95.0	97.2	4	3
	20	96.7	98.6	4	4
	30	95.8	98.6	4	2
0.90	10	96.9	98.6	4	3
	20	96.4	97.2	6	3
	30	96.9	98.6	5	3
0.95	10	95.8	97.2	4	3
	20	96.9	98.6	4	2
	30	97.2	98.6	4	3

Table 5: Performance of GA on the feature pool FP$_2$.

w	Number of generations	Average accuracy (%)	Maximum accuracy (%)	Average size	Minimum size
0.70	10	95.3	97.2	6	5
	20	98.1	100	6	4
	30	97.5	98.6	5	4
0.75	10	97.2	98.6	7	5
	20	97.2	98.6	7	6
	30	96.9	97.2	5	3
0.80	10	95.8	97.2	6	4
	20	96.1	97.2	5	3
	30	96.9	98.6	6	3
0.85	10	97.2	98.6	6	3
	20	97.8	98.6	5	3
	30	98.1	98.6	6	3
0.90	10	98.3	100	4	3
	20	97.5	98.6	4	3
	30	97.2	100	4	3
0.95	10	97.8	98.6	4	3
	20	97.5	98.6	4	3
	30	98.1	98.6	4	3

Table 6: Performance of GA on the feature pool FP_3.

w	Number of generations	Average accuracy (%)	Maximum accuracy (%)	Average size	Minimum size
0.70	10	96.7	98.6	6	3
	20	96.4	97.2	6	3
	30	97.8	100	7	5
0.75	10	96.7	98.6	8	7
	20	97.8	100	8	4
	30	97.8	100	10	5
0.80	10	96.9	98.6	7	3
	20	98.9	100	5	3
	30	98.1	98.6	10	5
0.85	10	97.8	100	5	3
	20	98.3	100	5	3
	30	98.9	100	6	4
0.90	10	98.6	100	6	3
	20	98.6	100	4	3
	30	98.9	100	4	3
0.95	10	99.4	100	5	3
	20	98.3	100	4	3
	30	98.6	100	4	3

Table 7: Performance of GA on the feature pool FP_4.

w	Number of generations	Average accuracy (%)	Maximum accuracy (%)	Average size	Minimum size
0.70	10	98.6	98.6	12	11
	20	98.3	98.6	12	6
	30	98.3	98.6	9	4
0.75	10	98.6	100	10	6
	20	98.9	100	9	6
	30	98.6	98.6	11	10
0.80	10	98.6	98.6	12	7
	20	98.6	98.6	9	3
	30	98.6	98.6	8	5
0.85	10	98.6	98.6	7	5
	20	98.6	98.6	9	3
	30	98.6	98.6	9	6
0.90	10	98.9	100	5	5
	20	99.2	100	9	4
	30	98.9	100	6	3
0.95	10	98.3	98.6	10	7
	20	98.6	98.6	5	4
	30	98.9	100	6	3

Table 8: The proposed method versus seven state-of-art methods.

The proposed method	100 (3)
[25]	94.10 (-)
[27]	100 (8)
[16]	100 (6)
[26]	95.0 (-)
[21]	100 (4)
[3]	100 (2)
[17]	100 (25)

Since the evolutionary algorithm performs a stochastic search, we consider the average accuracy and the average dimensionality of the selected subsets over a number $P = 5$ of trials. Within each FP_i ($i = 1, \dots, 4$), Tables 4, 5, 6, and 7 report the accuracy (average and maximum) and the number of selected genes (average and minimum), respectively, by each value of w and the number of generations.

Compared with the baseline model of FP_1 (*red features* in Table 3), whose accuracy is 94,4% on 12 features, we can see from Table 4 that the proposed evolutionary approach results in gene subsets of smaller size for each combination of w and number of generations. As well, the average accuracy outperforms the baseline model only if $w \geq 0.80$, meaning that we should give more priority on the classification accuracy over the size when evaluating the fitness of each feature subset. Moreover, the number of generations seems to not significantly affect the performance of the algorithm, suggesting that few generations are sufficient for GA to converge on the best individual.

Compared with the baseline model (accuracy: 94,4%, size: 18) of FP_2 (*red* and *blue features* in Table 3*)*, Table 5 shows a clear improvement in terms of both classification accuracy and dimensionality for each combination of w and number of generations. Interestingly enough, increasing w (that means the fitness is evaluated giving more priority on the accuracy over the size) does not significantly increase the accuracy of the selected subset, while the size of the selected subset tends to decrease as w increases. This seems to suggest that the optimization of the accuracy (first term in the fitness function) implies optimizing the dimensionality too. As in the case of FP_1, the performance does not improve when increasing the number of generations.

Our GA achieves the best results on the feature pool FP_3 (*red*, *blue,* and *green features* in Table 3), as we can see in Table 6. Indeed, the comparison with the baseline model (accuracy: 94,4%, size: 21) shows an improved performance for each combination of w and number of generations. Moreover, for 13 different settings of parameters, a classifier with 100% accuracy is identified by the algorithm. Higher values of w, in particular $w \geq 0.85$, lead to the best performance not only in terms of accuracy but also in terms of dimensionality, confirming that optimizing the accuracy means automatically reducing the size of the selected subset. Again, the number of generations seems to be not important, especially for higher values of w.

Finally, in the case of FP_4 (*red*, *blue*, *green,* and *yellow features* in Table 3), each combination of parameters results in the selection of gene subsets whose classification accuracy

Table 9: Features belonging to the perfect predictors in Table 10.

FP	Selected feature	Frequency
FP_2	1144**r**	**3** (3)
	6855**r**	**2** (3)
	1834**r**	**1** (3)
	6376**r**	**1** (3)
	2354**b**	**3** (3)
	4377**b**	**2** (3)
	4373**b**	**1** (3)
FP_3	1144**r**	**15** (18)
	1834**r**	**10** (18)
	6855**r**	**5** (18)
	1685**r**	**4** (18)
	760**r**	**3** (18)
	1882**r**	**1** (18)
	2288**r**	**1** (18)
	6376**r**	**1** (18)
	2354**b**	**12** (18)
	4377**b**	**9** (18)
	4373**b**	**7** (18)
	2402**b**	**1** (18)
	4328**b**	**1** (18)
	2642**g**	**8** (18)
	758**g**	**7** (18)
FP_4	1685**r**	**3** (7)
	6855**r**	**3** (7)
	1144**r**	**2** (7)
	1834**r**	**2** (7)
	4366**r**	**2** (7)
	1882**r**	**1** (7)
	2288**r**	**1** (7)
	6041**r**	**1** (7)
	2354**b**	**6** (7)
	4377**b**	**3** (7)
	2402**b**	**2** (7)
	4373**b**	**1** (7)
	2642**g**	**4** (7)
	758**g**	**2** (7)
	2128**g**	**1** (7)
	2020**y**	**5** (7)
	6281**y**	**5** (7)
	6347**y**	**5** (7)
	1928**y**	**4** (7)
	2121**y**	**1** (7)
	4196**y**	**1** (7)

is, on average, the same as the baseline model (98,6%) and no further improvement was achieved by the evolutionary algorithm in terms of accuracy. On the other hand, the dimensionality of the selected subsets is much lower than the initial number of features (29), which reveals a high degree of correlation and redundancy between the genes belonging to FP_4.

5. Discussion

A basic question is to discuss the change in accuracy when varying the number of selected features and their combinations. In general, we believe that there is not a rule to determine an optimal number of features to get the best accuracy even for a specific classifier since that number may change from data to data and also may vary from different feature selection methods as our experiments demonstrate.

The threshold of 20 used to cut off top-ranked features is an arbitrary number, though it is based on our experience as we consider that biologists like a small number of features to separate two classes of cells and building a classifier would need a long time if many discriminatory features are selected.

However, this arbitrary choice does not pay when we simply consider use SVM on the 20 top-ranked features (baseline model) or on nested subsets of top-ranked features (i.e., top-2, top-4, top-8, etc.): accuracy is poor but this is not surprising and means that many features interact closely.

Our method demonstrated its efficiency in discovering the size of optimal subsets selected on the subsets of common features. Results show that the SVM classifier performs better on these optimal subsets. However, features common to all ranking methods (i.e., the *red features* belonging to FP_1) define a search space that is too small and the performance of the classifier did not increase when the search was refined by an additional number of generations. When this search space was enlarged by adding *blue*, *green*, and *yellow features* our approach shows an excellent performance, not only at providing a very good average accuracy, but also with respect to the number of selected features and the computational cost. Resulting from the union of *red*, *blue*, and *green features*, the pool FP_3 seems to define the most effective search space for the GA.

Table 8 summarizes our results with the results of seven state-of-art methods from the literature. The conventional criteria are used to compare the results, the classification accuracy in terms of the rate of correct classification (first number) and the number of used genes (the number in parenthesis, "-" indicating that the number of genes is not available). For our approach, the classification rate we presented is the maximum accuracy obtained on FP_3 and the corresponding number of genes (see Table 6 for details). As it can be observed, we obtain a maximum classification rate of 100% using 3 genes (the corresponding average accuracy was 99,4%) which is much better than that reported in [25, 26]. This same performance is achieved by [3, 16, 17, 21, 27]. However, the number of genes selected by [16, 17, 21, 27] is greater than the one obtained by our method whose number of selected genes is greater than the one reported in [3].

We also observe that increasing the number of generations does not greatly affect the performance of the algorithm. This may be because the size of the initial gene pool FP_3 gives search space enough to the evolutionary algorithm. As well, the performance increases within high

Table 10: Perfect predictors identified by the proposed approach.

FP	Size	Features
FP_2	4	1144**r** 2354**b** 4373**b** 4377**b**
	4	1144**r** 1834**r** 6855**r** 2354**b**
	5	1144**r** 6855**r** 6376**r** 2354**b** 4377**b**
FP_3	3	1144**r** 1834**r** 2642**g** (**4**_times_)
	4	1144**r** 2354**b** 4373**b** 4377**b** (**3**_times_)
	4	1144**r** 1834**r** 2354**b** 758**g** (**2**_times_)
	4	1834**r** 2354**b** 4328**b** 2642**g**
	4	1834**r** 1685**r** 2354**b** 2642**g**
	5	1144**r** 1834**r** 1685**r** 4373**b** 758**g**
	5	1144**r** 1834**r** 2354**b** 4377**b** 758**g**
	6	2288**r** 6855**r** 2354**b** 4377**b** 758**g** 2642**g**
	6	1144**r** 1685**r** 6855**r** 2354**b** 4373**b** 4377**b**
	7	760**r** 1144**r** 6376**r** 6855**r** 2354**b** 4373**b** 4377**b**
	8	760**r** 1144**r** 1685**r** 1882**r** 6855**r** 4373**b** 4377**b** 758**g**
	8	760**r** 1144**r** 6855**r** 2354**b** 2402**b** 4377**b** 758**g** 2642**g**
FP_4	5	2354**b** 4377**b** 2020**y** 6281**y** 6347**y**
	5	2354**b** 2642**g** 2020**y** 6281**y** 6347**y**
	5	1685**r** 2354**b** 1928**y** 2020**y** 6347**y**
	6	2354**b** 2128**g** 2642**g** 2020**y** 6281**y** 6347**y**
	6	6855**r** 2354**b** 2402**b** 4377**b** 2642**g** 1928**y**
	14	1144**r** 1834**r** 1882**r** 1685**r** 4366**r** 6855**r** 2354**b** 2402**b** 4373**b** 758**g** 2642**g** 1928**y** 2121**y** 6281**y**
	14	1144**r** 1685**r** 1834**r** 2288**r** 4366**r** 6041**r** 6855**r** 4377**b** 758**g** 1928**y** 2020**y** 4196**y** 6281**y** 6347**y**

values of the parameter w. This means that the tradeoff between the two objectives of the fitness function is best represented when we give more importance to the accuracy since a high level of accuracy was automatically reached with a low number of features.

Another topic to address is the number of features subsets that reach the 100% accuracy (*perfect predictors*) and the frequency of selection of the genes that are member of the best predictors. Table 10 shows the perfect predictors discovered by the proposed approach. Interesting, no perfect predictor was discovered on the search space defined by FP_1. It seems to confirm that this space is not large enough and contains groups of correlated features. *Blue* and *green features* mitigate the presence of this correlation by enlarging the search space. As well, the presence of *yellow features* in FP_4 seems to influence the size of the optimal predictors since there is a notable difference when we consider the size of optimal predictors originated by FP_2 and FP_3. We observe that all features belonging to a perfect predictor are *multicoloured*, that is, they denote top-ranked genes shared by different groups of ranking methods. This indicates that combinations of features are beneficial.

Table 9 shows the frequency of the genes belonging to the optimal predictors (the number in parenthesis indicates the total number of perfect predictor within each feature pool). These results can be used by biologists for further evaluation.

6. Conclusions

We presented a new evolutionary approach to select relevant features subsets in order to use them for the classification task. With respect to speeding-up the EA evaluation, we worked in proposing the combination of different ranking methods with two goals: to incorporate information to the GA to be used by genetic operators, and to reduce the computational time of the classification process by means of a pre-processing step from the data. The EA incorporates information in the early stage, when different ranking methods are applied before running the classification process, by organizing the top-ranked features into different feature pools. The main concern is the formulation of the feature selection issue as an optimization problem so that the predictors with maximum accuracy and minimum size can be found. We demonstrated that the proposed approach solves this optimization problem in efficient way and experimental results show that our method outperforms different state-of-art methods for the classification of microarray data. As future work, we will apply the proposed method to a variety of datasets and study the feature overlapping.

Acknowledgment

The authors are very grateful to anonymous reviewers for the useful comments and suggestions.

References

[1] J. Khan, J. S. Wei, M. Ringnér, et al., "Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks," *Nature Medicine*, vol. 7, no. 6, pp. 673–679, 2001.

[2] T. R. Golub, D. K. Slonim, P. Tamayo, et al., "Molecular classification of cancer: class discovery and class prediction by gene expression monitoring," *Science*, vol. 286, no. 5439, pp. 531–527, 1999.

[3] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, "Gene selection for cancer classification using support vector machines," *Machine Learning*, vol. 46, no. 1–3, pp. 389–422, 2002.

[4] I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, Elsevier, Amsterdam, The Netherlands, 2nd edition, 2005.

[5] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *Journal of Machine Learning Research*, vol. 3, pp. 1157–1182, 2003.

[6] E. Pranckeviciene and R. Somorjai, "On classification models of gene expression mieroarrays: the simpler the better," in *Proceedings of the IEEE International Conference on Neural Networks*, pp. 3572–3579, 2006.

[7] Y. S. Ong and A. J. Keane, "Meta-Lamarckian learning in memetic algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 2, pp. 99–110, 2004.

[8] I.-S. Oh, J.-S. Lee, and B.-R. Moon, "Hybrid genetic algorithms for feature selection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 11, pp. 1424–1437, 2004.

[9] H. Simon, "Supervised analysis when the number of candidate features (p) greatly exceeds the number of cases (n)," *SIGKDD Explorations*, vol. 5, no. 2, pp. 31–36, 2003.

[10] R. L. Somorjai, B. Dolenko, and R. Baumgartner, "Class prediction and discovery using gene microarray and proteomics mass spectroscopy data: curses, caveats, cautions," *Bioinformatics*, vol. 19, no. 12, pp. 1484–1491, 2003.

[11] A. L. Blum and P. Langley, "Selection of relevant features and examples in machine learning," *Artificial Intelligence*, vol. 97, no. 1-2, pp. 245–271, 1997.

[12] H. Liu and L. Yu, "Toward integrating feature selection algorithms for classification and clustering," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 4, pp. 491–502, 2005.

[13] R. Kohavi and G. H. John, "Wrappers for feature subset selection," *Artificial Intelligence*, vol. 97, no. 1-2, pp. 273–324, 1997.

[14] Y. Saeys, I. Inza, and P. Larrañaga, "A review of feature selection techniques in bioinformatics," *Bioinformatics*, vol. 23, no. 19, pp. 2507–2517, 2007.

[15] T. Jirapech-Umpai and S. Aitken, "Feature selection and classification for microarray data analysis: evolutionary methods for identifying predictive genes," *BMC Bioinformatics*, vol. 6, article 148, 2005.

[16] S. Peng, Q. Xu, X. B. Ling, X. Peng, W. Du, and L. Chen, "Molecular classification of cancer types from microarray data using the combination of genetic algorithms and support vector machines," *FEBS Letters*, vol. 555, no. 2, pp. 358–362, 2003.

[17] E. B. Huerta, B. Duval, and J.-K. Hao, "A hybrid GA/SVM approach for gene selection and classification of microarray data," in *Proceedings of the EvoWorkshops*, vol. 3907 of *Lecture Notes in Computer Science*, pp. 34–44, 2006.

[18] F. Tan, X. Fu, Y. Zhang, and A. G. Bourgeois, "Improving feature subset selection using a genetic algorithm for microarray gene expression data," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '06)*, pp. 2529–2534, Vancouver, Canada, July 2006.

[19] L. Li, C. R. Weinberg, T. A. Darden, and L. G. Pedersen, "Gene selection for sample classification based on gene expression data: study of sensitivity to choice of parameters of the GA/KNN method," *Bioinformatics*, vol. 17, no. 12, pp. 1131–1142, 2002.

[20] V. Bevilacqua, G. Mastronardi, F. Menolascina, A. Paradiso, and S. Tommasi, "Genetic algorithms and artificial neural networks in microarray data analysis: a distributed approach," *Engineering Letters*, vol. 13, no. 3, pp. 335–343, 2006.

[21] A. R. Reddy and K. Deb, "Classification of two-class cancer data reliably using evolutionary algorithms," Tech. Rep., KanGAL, 2003.

[22] J. Bins and B. A. Draper, "Feature selection from huge feature sets," in *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2, pp. 159–165, 2001.

[23] J. C. Platt, "Fast training of support vector machines using sequential minimal optimization," in *Advances Kernel Methods-Support Vector Learning*, chapter 12, pp. 41–65, MIT Press, Cambridge, Mass, USA, 1998.

[24] M. Wall, "GAlib: A C++ Library of Genetic Algorithm Components," Massachusetts Engineering Department, August 1996.

[25] T. S. Furey, N. Cristianini, N. Duffy, D. W. Bednarski, M. Schummer, and D. Haussler, "Support vector machine classification and validation of cancer tissue samples using microarray expression data," *Bioinformatics*, vol. 16, no. 10, pp. 906–914, 2000.

[26] S. Chao and C. Lihui, "Feature dimension reduction for microarray data analysis using locally linear embedding," in *Proceedings of the Asia Pacific Bioinformatics Conference (APBC '05)*, pp. 211–217, 2005.

[27] Y. Wang, F. S. Makedon, J. C. Ford, and J. Pearlman, "HykGene: a hybrid approach for selecting marker genes for phenotype classification using microarray gene expression data," *Bioinformatics*, vol. 21, no. 8, pp. 1530–1537, 2005.

Hindawi Publishing Corporation
Journal of Artificial Evolution and Applications
Volume 2009, Article ID 848532, 13 pages
doi:10.1155/2009/848532

Research Article

Classification of Oncologic Data with Genetic Programming

Leonardo Vanneschi,[1] Francesco Archetti,[1, 2] Mauro Castelli,[1] and Ilaria Giordani[1]

[1] *Department of Informatics, Systems and Communication (D.I.S.Co.), University of Milano-Bicocca, 20126 Milan, Italy*
[2] *Consorzio Milano Ricerche, 20126 Milan, Italy*

Correspondence should be addressed to Leonardo Vanneschi, vanneschi@disco.unimib.it

Received 14 November 2008; Revised 2 April 2009; Accepted 13 June 2009

Recommended by Jason Moore

Discovering the models explaining the hidden relationship between genetic material and tumor pathologies is one of the most important open challenges in biology and medicine. Given the large amount of data made available by the DNA Microarray technique, Machine Learning is becoming a popular tool for this kind of investigations. In the last few years, we have been particularly involved in the study of Genetic Programming for mining large sets of biomedical data. In this paper, we present a comparison between four variants of Genetic Programming for the classification of two different oncologic datasets: the first one contains data from healthy colon tissues and colon tissues affected by cancer; the second one contains data from patients affected by two kinds of leukemia (acute myeloid leukemia and acute lymphoblastic leukemia). We report experimental results obtained using two different fitness criteria: the receiver operating characteristic and the percentage of correctly classified instances. These results, and their comparison with the ones obtained by three nonevolutionary Machine Learning methods (Support Vector Machines, MultiBoosting, and Random Forests) on the same data, seem to hint that Genetic Programming is a promising technique for this kind of classification.

1. Introduction

High-throughput microarrays have become one of the most important tools in functional genomics studies, and they are commonly used to address various biological questions, like disease classification and treatment prognosis. Although cancer detection and class discovery have often been studied over the past years, no general way to work out this problem has been found yet, probably because there can be many pathways causing cancer, and a tremendous number of varieties exist. Recently, array technologies have made it straightforward to measure and monitor the expression levels of thousand of genes during cellular differentiation and response. It has been shown that specific patterns of gene expression occur during different biological states such as embryogenesis, cell development, and during normal physiological responses in tissues and cells [1]. The expression of a gene provides a measure of its activity under certain biochemical conditions. The key problem of evaluation of gene expression data is to find patterns in the apparently unrelated values measured. With increasing numbers of genes spotted on microarrays, visual inspection of these data has become impossible, and, hence, the importance of computer analysis, in particular by means of Machine Learning, has substantially increased in recent years. Well-studied datasets of different phenotypes are publicly available to train and evaluate supervised pattern analysis algorithms for classification and diagnosis of unknown samples. Therefore, there is a strong need to build molecular classifiers made of a small number of genes, especially in clinical diagnosis, where it would not be practical to have a diagnostic assay to evaluate hundreds of genes in one test.

In this study, we present an application of Genetic Programming (GP) [2] for molecular classification of cancer. In particular, we study two publicly available oncologic datasets: the first one contains data from healthy colon tissues and colon tissues affected by cancer; the second one contains data from patients affected by two different kinds of leukemia (acute myeloid leukemia and acute lymphoblastic leukemia). Four versions of GP are studied on those datasets; those GP variants differ by the way of handling the training set and by the fact that they may or may not affect training data with noise. We test the usefulness of GP using two different fitness functions: the receiver operating characteristic (ROC) area under curve (AUC) and the measure of correctly classified instances (CCIs). For both these performance measures, results returned by GP are compared with the ones returned by three well-known nonevolutionary Machine Learning methods: Support Vector Machines, MultiBoosting and Random Forests.

Even though (as described in the next section) GP has been previously applied by other authors to microarray data, we believe that the present manuscript contains the following interesting sources of novelty: it contains a study of various different GP versions with two different fitness measures on two different datasets, and it compares the results returned by GP with the ones of other, nonevolutionary, Machine Learning methods.

The paper is structured as follows. Section 2 presents an overview of previous and related contributions. Section 3 presents the datasets that we have used, describes the four presented GP frameworks, and also introduces the three nonevolutionary Machine Learning methods whose results have been compared with the GP ones. Section 4 reports experimental results. In Section 5 we present the genotype of some of the best solutions found by GP. Finally, Section 6 concludes the paper and proposes some ideas for future research. The paper is terminated by two appendices where the most recurrent genes contained in the best solutions found by GP are defined.

2. State of the Art

Given the large amount of data coming from DNA microarray analysis, in the last few years researchers have started paying a growing attention to cancer classification using gene expression. Studies have shown that gene expression changes are related with different types of cancers. Many different stochastic Machine Learning methods [3] have already been applied for microarray data analysis, like k-nearest neighbors [4], hierarchical clustering [5], self-organizing maps [6], Support Vector Machines [7, 8], or Bayesian networks [9]. All this different classification methods share some common issues that make classification a nontrivial task applied on gene expression data. In fact, the attribute space, or the number of genes, of the data is often huge: there are usually thousands to hundred thousands of genes present in each dataset. Also, if the samples are mapped to points in the attribute space, they often can be viewed as very sparse points in a very high dimensional space. Most of existing classification algorithms were not designed with this kind of data characteristics in mind. Thus, such a situation represents a challenge for most classification algorithms. Overfitting is a major problem due to the high dimension, and the small number of observations makes generalization even harder. Furthermore, most genes are irrelevant to cancer distinction: some researchers proposed to perform a gene selection prior to cancer classification to reduce data size, thus improving the running time and remove a large number of irrelevant genes which improves the classification accuracy [3].

In the last few years Evolutionary Algorithms (EAs) [10] have been used for solving both problems of selection and classification in gene expression data analysis. Genetic Algorithms (GAs) [11] have been employed for building selectors where each allele of the representation corresponds to one gene, and its state denotes whether the gene is selected or not [12]. GP on the other hand has been shown to work well for recognition of structures in large datasets [13]. GP has been applied to microarray data to generate programs that reliably predict the health/malignancy states of tissue or classify different types of tissues. An intrinsic advantage of GP is that it automatically selects a small number of feature genes during the evolution [14]. The evolution of classifiers from the initial population seamlessly integrates the process of gene selection and classifier construction. In fact, in [15] GP is used to cancer expression profiling data to select potentially informative feature genes, build molecular classifiers by mathematical integration of these genes, and classify tumour samples. Furthermore, GP has been shown a promising approach for discovering comprehensible rule-based classifiers from medical data [16] as well as gene expression profiling data [17]. Results presented in those contributions are encouraging and pave the way to a further investigation of GP for this kind of datasets, which is the goal of this paper.

3. Material and Methods

3.1. Dataset. We test our methods on two publicly available oncologic datasets: the first one contains data from healthy colon tissues and colon tissues affected by cancer and will be called *Colon Dataset* from now on; the second one contains data from patients affected by two different kinds of leukemia (acute myeloid leukemia and acute lymphoblastic leukemia) and will be called *Leukemia Dataset* from now on. These two datasets are described as follows.

3.1.1. Colon Dataset. The Colon Dataset is a collection of expression measurements from colon biopsy samples reported in [5]. The dataset consists of 62 samples of colon epithelial cells collected from colon-cancer patients. In particular the “tumour” biopsies were extracted from tumours, and the “normal” biopsies were collected from healthy parts of the colons of the same patients. The final assignments of the status of biopsy samples were made by pathological examination. Gene expression levels in these 62 samples were measured using high-density oligonucleotide arrays. Of the about 6000 genes represented in these arrays, 2000 genes were selected based on the confidence in the measured expression levels. The dataset, 62 samples over 2000 genes, is available at http://microarray.princeton.edu/oncology/affydata/index.html.

3.1.2. Leukemia Dataset. The Leukemia Dataset (first introduced in [18]) contains data from 72 patients, half of which affected by acute myeloid leukemia and the remaining ones affected by lymphoblastic leukemia. For these patients, 7070 genes have been monitored. For measuring the expression level of those genes, oligonucleotides microarrays produced by Affimetrix have been used. Thus, the dataset is composed by 7070 columns and 72 lines, each of which labelled with “myeloid” or “lymphoblastic” in order to separate these two kinds of leukemia. This dataset and a detailed description of it can be found at http://genecruiser.broadinstitute.org/cgi-bin/cancer/publications/pub_paper.cgi?mode=view&paper_id=43.

3.2. Classification Methods. After a discussion of our GP framework and variants, the Machine Learning methods used for comparing results, that is, Support Vector Machines (SVM), MultiBoosting, and Random Forests, are described here in a deliberately synthetic way, since they are well-known and well-established techniques. References to master those methods are quoted.

3.2.1. Genetic Programming for Classification. Candidate classifiers (individuals) that are evolved by GP are Lisp-like tree expressions built using the function set F $= \{+, *, -, /\}$ and a terminal set T composed by M floating point variables, where M is the number of columns in the dataset (i.e., $M = 2000$ for the Colon Dataset and $M = 7070$ for the Leukemia Dataset). Thus, GP individuals are arithmetic expressions (exactly the same method as in [19] has been used to avoid expressions containing divisions with a denominator equal to zero). These expressions can be transformed into binary classifiers (class “normal” for healthy tissues and class “tumour” for ill ones for the Colon Dataset; class “myeloid” for acute myeloid leukemia and class “lymphoblastic” for acute lymphoblastic leukemia for the Leukemia Dataset) by using a threshold. Here, we use two fitness functions: ROC-AUC and CCI. In the first case each classifier is evaluated by a fitness function defined as the area under the ROC curve [20, 21]. In this work, the ROC curve is obtained by considering 20 different threshold values uniformly distributed in the interval $[-1, 1]$. For each one of these threshold values, a point is drawn having as abscissa the false positive rate and as ordinate the true positive rate obtained by the candidate classifier using that threshold. The area is calculated using the trapezoids method. The second type of fitness function is instead obtained by fixing a particular threshold value (equal to 0.5 in this work, following [14]) and calculating the CCI. CCI is defined as the correctly classify instances rate, that is, CCI $= (\mathrm{TP} + \mathrm{TN})/N$, where TP indicates True Positives, TN specifies True Negatives, and N is the number of rows in the dataset.

For calculating both these fitness values during the presented GP simulations, we have considered a static and a dynamic way of handling the training set, and we have considered training data as they are (i.e., without any explicit modification) or perturbing them with noise. These different strategies, used for improving GP generalization ability as suggested in [19], are described as follows.

Static Training Set Handling. Fitness has been calculated using each line in the training set at each generation for all individuals in the population.

Dynamic Training Set Handling. The training set is partitioned into 5 subsets, and at each generation only 4 of those subsets are used to calculate fitness, while one of them is not used. At each 5 generations, one of the 4 used subsets is selected and replaced by the subset that was previously left unused. In this way, the training set is modified in a cyclic way at each 5 generations. The number of subsets in which the dataset has been partitioned (5) and the period of training set modifications (5 generations) have been chosen by means of a set of experiments, whose results are not reported here.

No Noise Added to Data. When calculating fitness, each GP terminal symbol x_i has been replaced exactly by the values in the ith column of the training set.

Gaussian Noise Added to Data. Data have not been used exactly as they are in the original dataset, but a Gaussian noise (with average equal to zero and with a standard deviation equal to the datum value divided by 100) has been added to them. Each time a GP terminal symbol has to be evaluated, a new Gaussian perturbation of the original value is generated (in this way, the same variable is likely to have two slightly different values in two different fitness evaluations).

Combining these different methods of handling training set and data have lead us to define four different versions of GP, that we call GP0, GP1, GP2, and GP3 for simplicity.

(i) GP0 uses the static training set handling and data with no noise. This corresponds to *standard* GP.

(ii) GP1 uses the static training set handling and data perturbed with Gaussian noise.

(iii) GP2 uses the dynamic training set handling and data with no noise.

(iv) GP3 uses the dynamic training set handling and data perturbed with Gaussian noise.

The other parameters we have used in our GP experiments are population size of 200 individuals, ramped half-and-half initialization, tournament selection of size 7, maximum tree depth equal to 10, subtree crossover rate $p_c = 0.95$; subtree mutation rate $p_m = 0.1$, maximum number of generations equal to 500; furthermore, we have used generational tree-based GP with elitism, that is, unchanged copy of the best individual on the training set into the next population at each generation.

3.2.2. Other Machine Learning Methods. In this paragraph we briefly describe the other machine learning methods used for our tests. For more details on these algorithms and their use, the reader is referred to the respective references quoted here and after.

Support Vector Machines. Support Vector Machines (SVMs) were originally introduced in [22]. Their aim is to device a computationally efficient way of learning separating hyperplanes in a high dimensional feature space. In this work we use the implementation of John Platt's [23] sequential minimal optimization (SMO) algorithm for training the support vector classifier. Training an SVM requires the solution of a large quadratic programming (QP) optimization problem. SMO works by breaking this large QP problem into a series of smallest ones. Parameter values used in this work are complexity parameter c equal to 1.0, size of the kernel cache equal to 1000003, epsilon value for the round-off error equal to $1 \cdot 10^{-12}$, exponent for the polynomial kernel equal to 1.0, and tolerance parameter equal to 0.001. All these parameter values correspond to the standard values offered by the Weka software [24]. These parameters are defined, for instance, in [23].

MultiBoosting. MultiBoosting is an extension to the classification method Adaptive Boosting (AdaBoost) [25]. AdaBoost is a meta-algorithm and can be used in conjunction with other learning algorithms to improve their performance. AdaBoost is adaptive in the sense that subsequent classifiers built are tweaked in favor of those instances misclassified by previous classifiers. Multiboosting can be viewed as combining AdaBoost with wagging. It is able to harness both AdaBoost's high bias and variance reduction with wagging's superior variance reduction. Using C4.5 as the base learning algorithm, multiboosting is demonstrated to produce decision committees with lower error than either AdaBoost or wagging significantly more often than the reverse over a large representative cross-section of datasets. It offers the further advantage over AdaBoost of suiting parallel execution. For more information, see [26]. Parameter values used in this work are 100 iterations, 3 subcommittees, and weight threshold for weight pruning equal to 100. All these parameter values correspond to the standard values offered by the Weka software [24].

Random Forests. Random Forests is an improved Classification and Regression Trees method [27]. It works by creating a large number of classification trees or regression trees. Every tree is built using a deterministic algorithm, and the trees are different owing to two factors. First, at each node, a best split is chosen from a random subset of the predictors rather than all of them. Secondly, every tree is built using a bootstrap sample of the observations. The out-of-bag data, approximately one-third of the observations, are then used to estimate the prediction accuracy. Unlike other tree algorithms, no pruning or trimming of the fully grown tree is involved. In this work we use the Breiman implementation presented in [28]. A number of trees equal to 300 have been used in this work. All the other parameters that we have used have been set to the standard values offered by the Weka software [24].

4. Experimental Results

Results obtained by the nonevolutionary methods and by the different GP variants on the Colon Dataset and on the Leukemia Dataset are reported in Sections 4.1 and 4.2, respectively.

To obtain these results, we have generated 10 different partitions of the dataset into training and test set. For each one of these partitions, 70% of the lines in the dataset chosen at random (with uniform probability distribution) form the training set and the remaining 30% the test set (we have explicitly checked that the same training-test partition does not appear more than once). To report results in this paper, for each one of these partitions we have proceeded as follows.

(i) For nondeterministic methods such as GP, Multiboosting and Random Forest, we have performed 100 independent executions, and we have retained the best values of CCI and ROC found on the test set.

(ii) For SVM, which is deterministic in this work, we have retained the values of CCI and ROC on the test set of the returned solution.

Thus, we have 10 values of CCI and 10 values of ROC for each method. We finally report the best, the average, and the standard deviation of these 10 solutions, both for CCI and ROC.

Furthermore, we have also randomly generated 500 different training-test set partitions (also in this case 70% of the lines in the dataset chosen at random with uniform probability distribution form the training set and the remaining 30% the test set, where we have explicitly checked that the same training-test partition does not appear more than once), and we have executed one run of each one of the studied methods (both nonevolutionary ones and GP) for each one of these partitions. Results of these further experiments are reported in Section 4.3.

Note on Computational Time. We have calculated the computational time for all the executions whose results are reported in Section 4.3. (i.e., 500 different executions for each Machine Learning method, each one with a different training-test partition) on a dedicated machine Intel Pentium III-500 with 128 M RAM, and we have calculated the averages of all these computational times. The various GP runs returned an average time of about 153 seconds; approximately the same average amount of time was requested by Boosting (about 155 seconds). Random Forests requested a larger average amount of time for one run (about 260 seconds); finally SVM was the fastest method (one run of SVM requested about 12 seconds on average).

4.1. Results on the Colon Dataset. Table 1 summarizes the experimental results obtained by the non-evolutionary methods on the Colon Dataset.

SVM is the method that returns the best average results, both for CCI and ROC, while the best CCI results are returned by Random Forests and SVM, and the best ROC results are returned by Random Forests. We point out that we have applied these classification methods to our datasets without any explicit feature selection algorithm nor preprocessing. The motivation for this is that we wanted to compare these results with the ones obtained by GP, pointing out that GP is able to perform an automatic feature selection, while the other non-evolutionary methods do not have this capability.

Table 1: Results returned by the nonevolutionary methods on the Colon Dataset. 10 different partitions of the dataset into training and test set have been considered. The best, average and standard deviations of the best CCI and ROC results obtained on each one of these 10 partitions are reported.

	CCI			ROC		
	Best	Average	Std. Dev.	Best	Average	Std. Dev.
Random Forests	0.9444	0.7417	0.0810	1	0.8250	0.0755
SVM	0.9444	0.8778	0.0438	0.9545	0.8525	0.0874
Multi Boosting	0.8889	0.7850	0.0577	0.9861	0.8152	0.0488

Table 2: Results returned by the studied GP variants on the Colon Dataset. The same 10 partitions of the dataset into training and test set as in Table 1 have been considered. The best, average and standard deviations of the best CCI and ROC results obtained on each one of these 10 partitions are reported.

	CCI			ROC		
	Best	Average	Std. Dev.	Best	Average	Std. Dev.
GP0	1	0.8926	0.038	1	0.9437	0.0472
GP1	1	0.8946	0.042	1	0.9444	0.0455
GP2	1	0.8947	0.039	1	0.9437	0.0455
GP3	1	0.895	0.042	1	0.9555	0.0466

Table 2 reports the results obtained by the different GP variants studied using the same 10 training-test partitions as in Table 1. Comparing the results reported in Table 2 with the ones reported in Table 1, we can remark that all GP variants are able to find an ideal solution both for CCI and ROC, which is not the case for the non-evolutionary methods (with the exception of Random Trees for ROC). Also comparing the average values, we can remark that all GP variants outperform all non-evolutionary methods, and the respective standard deviations seem to hint that the difference between GP performances and the ones of the other methods is statistically relevant.

Differences between the various GP variants seem marginal, which hints that both the dynamic dataset handling and the use of Gaussian noise are not useful to improve GP generalization ability, at least for this application. By the way, it has to be remarked that performances of standard GP (GP0) are already (informally) rather “high”, and thus difficult to improve. In the future, we plan to investigate the gain in using GP1, GP2, and GP3 for more complex problems, where GP0 is not able to find good solutions.

4.2. Results on the Leukemia Dataset. Results obtained by the studied non-evolutionary methods are summarized in Table 3. For the Leukemia Dataset, MultiBoosting is the method that has returned both the best results and the best average results, both for CCI and ROC.

Table 4 reports the results obtained by the different GP variants studied using the same 10 training-test partitions as in Table 3. Also in this case, all GP variants outperform all non-evolutionary methods, and standard deviation values seem to hint that the differences between the average results obtained by GP and the average ones obtained by the best non-evolutionary method on this dataset (MultiBoosting) are statistically relevant.

Table 3: Results returned by the nonevolutionary methods on the Leukemia Dataset. 10 different partitions of the dataset into training and test set have been considered. The best, average, and standard deviations of the best CCI and ROC results obtained on each one of these 10 partitions are reported.

	CCI			ROC		
	Best	Average	Std. Dev.	Best	Average	Std. Dev.
Random Forests	0.9048	0.7191	0.0939	0.9500	0.6999	0.1270
SVM	0.8571	0.7476	0.0552	0.8375	0.7274	0.0924
Multi Boosting	0.9524	0.7548	0.0733	1	0.7500	0.0895

Table 4: Results returned by the studied GP variants on the Leukemia Dataset. The same 10 partitions of the dataset into training and test set as in Table 3 have been considered. The best, average and standard deviations of the best CCI and ROC results obtained on each one of these 10 partitions are reported.

	CCI			ROC		
	Best	Average	Std. Dev.	Best	Average	Std. Dev.
GP0	1	0.8323	0.0390	1	0.8491	0.0047
GP1	1	0.8592	0.0425	1	0.8777	0.0400
GP2	1	0.8325	0.0395	0.9778	0.8500	0.0392
GP3	1	0.8607	0.0407	0.9904	0.8778	0.0381

All GP variants have been able to produce ideal solutions for CCI, while only GP0 and GP1 have been able to generate ideal ROC values. We finally remark that, also for the Leukemia Dataset, perturbing data with Gaussian noise or handling the training set in a dynamic way is not beneficial.

4.3. Further Experiments. In Sections 4.1 and 4.2, 10 different training-test set partitions were considered and, for each partition, 100 independent runs of each one of the nondeterministic methods (random forests, multiboosting and GP) were executed.

In this section we present the results that we have obtained by considering 500 different training-test partitions and executing one run of each method for each different partition. Best, average, and standard deviations of the obtained results are reported. The other used parameters are exactly the same as the ones used to produce the results of Sections 4.1 and 4.2.

Table 5 shows the results obtained by the nonevolutionary methods on these 500 different training-test partitions for the Colon dataset. The method that returns the best average results is SVM, both for CCI and ROC, even though SVM is the only method that is not able to obtain 1 as the best ROC.

Results obtained by the GP variants on the same 500 training-test set partitions are shown in Table 6. All the GP variants have returned better average CCI and ROC

Table 5: Results returned by the non-evolutionary methods on the Colon Dataset. 500 different partitions of the dataset into training and test set have been considered. The best, average, and standard deviations of the CCI and ROC results obtained on each one of these 500 partitions are reported.

	CCI			ROC		
	Best	Average	Std. Dev.	Best	Average	Std. Dev.
Random Forests	0.9444	0.8368	0.0688	1	0.8578	0.0627
SVM	0.9444	0.8567	0.0396	0.9545	0.864	0.075
Multi Boosting	0.9444	0.8295	0.051	1	0.823	0.0436

Table 6: Results returned by the studied GP variants on the Colon Dataset. The same 500 partitions of the dataset into training and test set as in Table 5 have been considered. The best, average, and standard deviations of the best CCI and ROC results obtained executing one run on each one of these 500 partitions are reported.

	CCI			ROC		
	Best	Average	Std. Dev.	Best	Average	Std. Dev.
GP0	1	0.8999	0.0497	1	0.9472	0.0440
GP1	1	0.9038	0.0499	1	0.9596	0.0345
GP2	1	0.9042	0.0446	1	0.9528	0.0385
GP3	1	0.9017	0.0454	1	0.9600	0.0368

Table 7: Results returned by the non-evolutionary methods on the Leukemia Dataset. 500 different partitions of the dataset into training and test set have been considered. The best, average, and standard deviations of the CCI and ROC results obtained on each one of these 500 partitions are reported.

	CCI			ROC		
	Best	Average	Std. Dev.	Best	Average	Std. Dev.
Random Forests	0.9048	0.7728	0.0747	1	0.7581	0.0873
SVM	0.9444	0.8153	0.0438	0.8375	0.7368	0.0835
Multi Boosting	0.9444	0.8267	0.0611	1	0.7974	0.081

Table 8: Results returned by the studied GP variants on the Leukemia Dataset. The same 500 partitions of the dataset into training and test set as in Table 7 have been considered. The best, average, and standard deviations of the best CCI and ROC results obtained executing one run on each one of these 500 partitions are reported.

	CCI			ROC		
	Best	Average	Std. Dev.	Best	Average	Std. Dev.
GP0	1	0.8348	0.0419	1	0.8469	0.0488
GP1	1	0.8569	0.0427	1	0.8890	0.0481
GP2	1	0.8304	0.0477	1	0.8406	0.0413
GP3	1	0.8560	0.0392	1	0.8871	0.0470

than the non-evolutionary methods. Furthermore, all the GP variants have returned a best CCI and best ROC equal to 1. The differences between the GP variants seem to be marginal. Finally, all the GP variants show a rather stable behavior given by the relatively small values of the standard deviations.

Table 7 reports the values returned by the non-evolutionary methods on 500 different training-test partitions of the Leukemia dataset. This time, the method that has returned the best average ROC and CCI results is MultiBoosting.

The results returned by the GP variants on the same 500 training-test set partitions of the Leukemia dataset are presented in Table 8. Also in this case, all the studied GP versions overcome all the studied non-evolutionary methods both for the average CCI and the average ROC. A best CCI and a best ROC value equal to 1 is found by each GP variant, and standard deviations are rather small, thus confirming that also on this dataset the studied GP variants have a rather stable behavior.

5. The Best Solutions Found by GP

In this section, we report the genotype of some of the best solutions found by GP in the form of expressions in infix notation, and successively we describe the most recurrent genes contained in them (Appendices A and B). These expressions are reported here to allow the reader to have an idea of how the best solutions found by GP on the test sets look like; we do not pretend them to necessarily be *the* model explaining the relationships between gene expressions and the studied pathologies. In order to build such a model, collaborations with domain experts are needed (and we are planning them in our future activity). Nevertheless, we hope that reporting those expressions here may be a starting point for this new and challenging research. Furthermore, we also report scatterplots of the Z-scores of the different genes contained in the best solutions found by GP (like, e.g., in [15]), and we show how those values are correlated when ROC and CCI are used as fitness functions.

5.1. Colon Dataset. We first report a solution with CCI $= 1$ on the test set found by GP0. Reported as an expression in infix notation, this solution is found in Algorithm 1.

We remark that GP has performed an automatic feature selection; in fact, this solution contains only 15 over the 2000 possible genes. This fact distinguishes GP from the other studied Machine Learning, that can use a subset of features only if an explicit feature selection algorithm is executed before training (preprocessing).

One of the solutions with area under the ROC curve on the test set equal to 1 returned by GP0 is

```
K03460%X59131 * (X66924 + H20709)
- (T74896 + U28963) * (R61359 + T86444)
- (U20659 - T81460) * R53941.
```

In this case, GP's feature selection has been even stronger: only 11 of the 2000 available genes are used by GP.

```
IF     ((X51416+R99200*X06614)%(H23544*X61123
       -T47213+M34344+(H79575-R50864)*U18920
       +R46739%(U20659+H04333)
       -R53941+L09604)>0.5)
THEN    Class = "tumour"
ELSE   Class = "normal"
```

Algorithm 1

```
IF     (X05409%M28130+(U94855-M84526)%(U04270
       *X55668%D28473
       -(D38498-Z37976)%M96326)> 0.5)
THEN    Class = "tumour"
ELSE   Class = "normal"
```

Algorithm 2

It is a widely agreed upon idea that only a restricted number of genes are correlated with tumour pathologies (those genes are often identified by domain experts as *biomarkers*). For this reason, the ability of GP to retain a limited number of genes into the proposed solutions is interesting. In order to identify and study the most important genes found by GP, for each one of the 4000 GP independent runs that we have performed to obtain the results reported in this paper (100 independent runs for each one of the 10 training-test different partitions and for each one of the 4 GP variants), we have retained the best solution found on the test set, both for CCI and ROC. In all those 8000 solutions, we have counted the number of occurrences of each gene in the dataset. We finally have extracted the 30 most recurrent genes. A detailed description of those genes is contained in Appendix A.

Furthermore, we have considered all the genes that have appeared in at least one best solution found by GP using CCI *and* in at least one best solution found by GP using ROC (i.e., we have considered the set of genes contained in the best solutions found by GP using CCI, set of genes contained in the best solutions found by GP using ROC, and we have considered the intersection between these two sets). In Figure 1 we show the normalized Z-Score of these genes.

Gene's normalized Z-Score has been studied, for instance, in [15], and it is defined as follows: for a given gene i, Z-Score $= (S_i - E(S_i))/\sigma$, where S_i denotes the number of times genes i being contained in the studied GP solutions, $E(S_i)$ is the expected number of times for gene i being contained in those solutions, and σ denotes the square root of the variance. The calculation of $E(S_i)$ is $ES_i =$ (number of genes contained in the studied GP solutions)/(number of genes in the initial gene pool).

Figure 1 shows the correlation between gene's normalized Z-Score for the two fitness criteria for the four versions of GP that we have studied. For all these GP versions, normalized Z-scores seem positively correlated (Figure 1 also reports the axis bisector, which represents the ideal correlation).

5.2. Leukemia Dataset. The genotype of one of the solutions with CCI = 1 found by GP0 is found in Algorithm 2.

Also in this case, GP has operated an automatic feature selection, given that this solution contains only 10 of the 7070 possible genes.

The genotype of a solution with area under the ROC curve on the test set equal to 1 returned by GP0 is

```
(U15782 - J04990)%X04707
+ X62822 - M27891 * M96326.
```

It contains only 6 of the 7070 possible genes.

Also for the Leukemia dataset for each one of the 4000 GP independent runs, we have retained the best solutions found on the test set, both for CCI and ROC. In all those 8000 solutions, we have counted the number of occurrences of each gene in the dataset. We finally have extracted the 30 most recurrent genes. A detailed description of those genes is contained in Appendix B.

In Figure 2 we report the correlation between the normalized Z-Scores of the genes that appear at least once in the best solutions found by GP using CCI and at least once in the best solutions found by GP using ROC. Also in this case, Z-Scores seem positively correlated (we also report the axis bisector in figure, to give an intuition of the ideal correlation).

6. Conclusions and Future Work

Four different variants of Genetic Programming (GP) for classification have been presented in this paper. The difference between these four versions is that they may/may not use a cyclic algorithm to dynamically handle the training set and they may/may not perturb input data with (Gaussian) noise. These GP variants have been applied to two publicly available biomedical microarray datasets representing a collection of expression measurements from colon biopsy experiments and leukemia. One the main characteristic of these datasets is that they both contain a large number of features—that is, information about gene expressions—(2000 in the case of the colon dataset and 7070 in the case of the leukemia dataset) and a low number of samples (62 in the case of the Colon dataset and 72 for the leukemia one). We believe that GP may be a suitable method to mine these datasets, given the ability of GP to deal with complex expressions and structures and to perform an automatic feature selection.

GP experiments have been executed using two different fitness functions: the ROC and the CCI. The first one of these fitness measures is calculated using a set of threshold values (20 uniformly distributed values in the range $[-1, 1]$ in this work), while the second one is obtained by fixing a predefined threshold value (0.5 in this work, following [14]). Both those fitness measures have received a noteworthy attention in past literature, but (to the best of our knowledge) they have never been studied together before in GP applications.

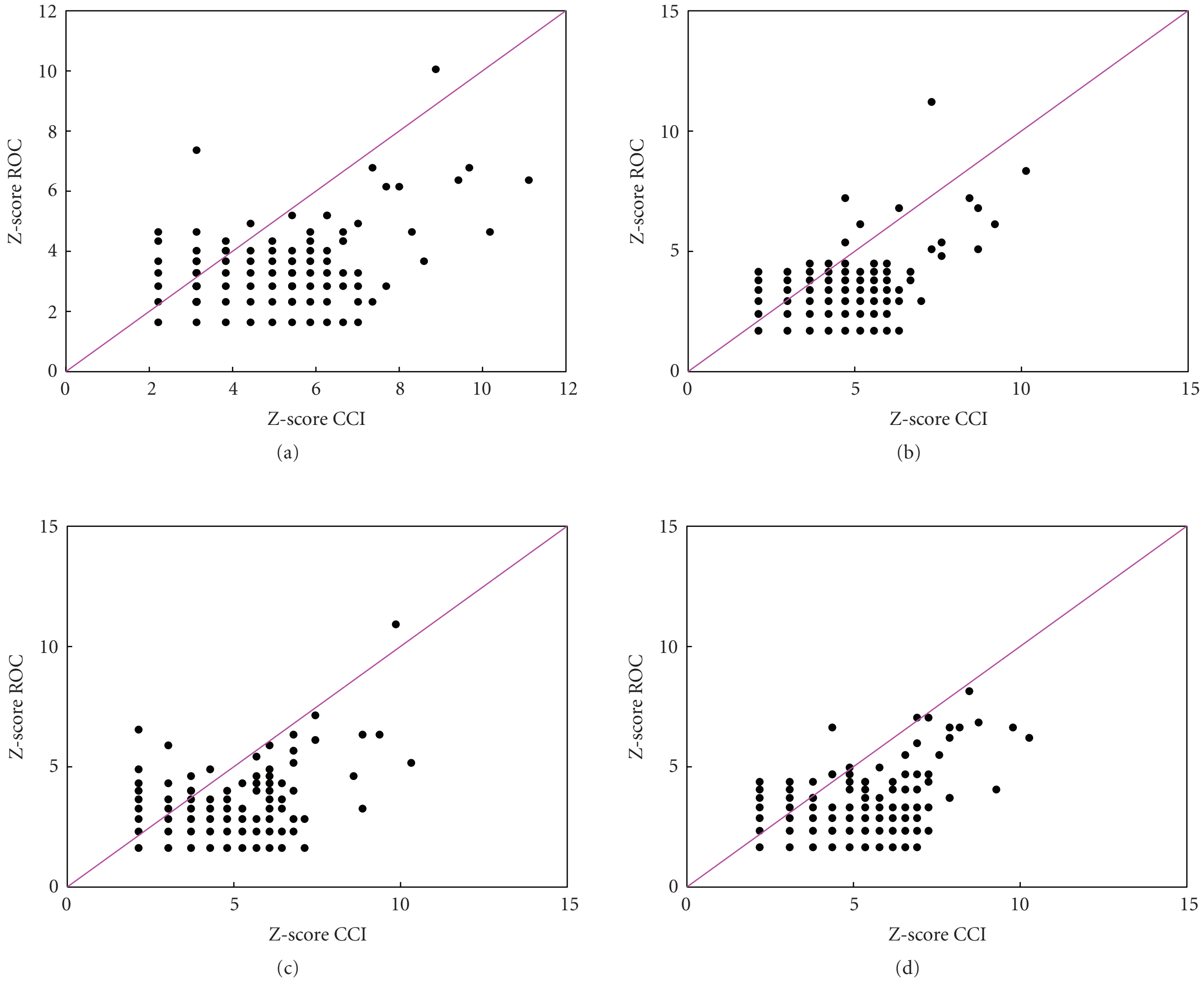


FIGURE 1: Normalized Z-score of the most recurrent common genes in the best solutions found by GP using CCI and ROC as fitness functions for the Colon dataset. (a): GP0, (b): GP1, (c): GP2, (d): GP3.

The experimental results returned by GP have been compared with the ones of three non-evolutionary Machine Learning methods (Support Vector Machines, MultiBoosting, and Random Forests). They show that GP is able to find better CCI and ROC results than the best non-evolutionary methods for both datasets. Even more interestingly, average results returned by GP (over a number of runs performed with different training-test partitions of the dataset) are better than the best ones returned by all the other non-evolutionary methods.

Furthermore, the reported results have shown no clear difference in the performances of the different GP variants, and this seems to hint that using the proposed dynamic algorithm to handle the training set or perturbing input data with Gaussian noise is not helpful to improve GP generalization ability, at least for this particular application.

We suspect that this is due to the fact that "standard GP" has good performances on these datasets, which are difficult to improve. The other GP variants deserve to be further tested on more difficult problems, where standard GP fails to find good quality solutions or requests too large amounts of computational resources.

These results are promising, even though they represent just a first preliminary step of a long term work, in which we wish to employ GP for cancer classifications in a more structured way and large scale. Many future activities are planned. First of all, we will train our GP system in a more sophisticated way, in order to improve its generalization ability. For instance, we could use more than one fitness criteria on the training set, following the idea presented in [29], where multioptimization on training is shown to increment GP generalization ability in many applications. For classification, it would be particularly interesting to use both ROC and CCI during training. Furthermore, we are planning to improve GP using more sophisticated methods for seeding the initial population compared to the standard ramped half and half method used here.

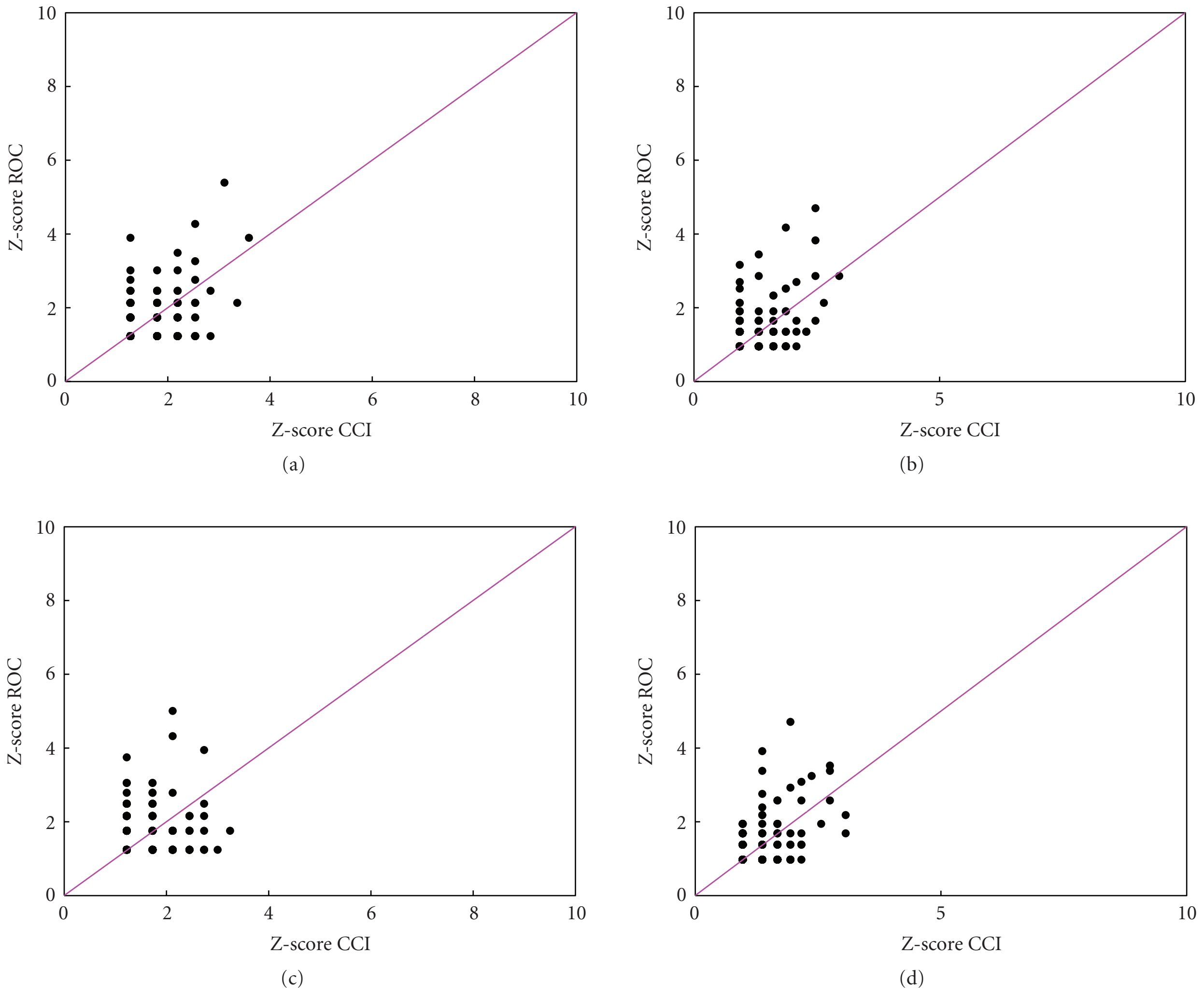


FIGURE 2: Normalized Z-score of the most recurrent common genes in the best solutions found by GP using CCI and ROC as fitness functions for the Leukemia dataset. (a): GP0, (b): GP1, (c): GP2, (d): GP3.

One of the main limitations of this work is that we did not use any application specific problem knowledge: a “semantic” analysis of the best solutions found by GP could have helped us to generate new and possibly more effective solutions. We are currently working in this direction: we are trying to develop a sort of “application-based” feature selection, and in parallel we are trying to give a biological interpretation to solutions found by GP, trying to infer interesting properties.

Appendices

A. Most Recurrent Genes Contained in the Best Solutions Found by GP for the Colon Dataset

In Table 9 we describe the most recurrent genes contained in the best solutions on the test set of the Colon Dataset returned by GP. For a more detailed discussion of these genes, see http://microarray.princeton.edu/oncology/affydata/index.html.

The first column of this table contains the gene IDs. They are entries of the GenBank database (see e.g., http://www.ncbi.nlm.nih.gov/Genbank/ for a description of this database of known genes). Other informations about these genes can be obtained by using these IDs as entries at the page: http://smd.stanford.edu/cgi-bin/source/sourceBatchSearch.

B. Most Recurrent Genes Contained in the Best Solutions Found by GP for the Leukemia Dataset

In Table 10 we present the most recurrent genes contained in the best solutions on the test set of the Leukemia Dataset returned by GP. For a more detailed discussion of these genes, see http://genecruiser.broadinstitute.org/cgi-bin/cancer/publications/pub_paper.cgi?mode=view&paper_id=43.

Also in this case, as for the table presented in Appendix A, the first column of this table contains the gene IDs.

Table 9: Definition of the most recurrent genes contained in the best solutions found by GP for the Colon Dataset.

GENE ID	Gene description
H75955	C2H2-type zinc finger proteins, such as ZNF238, act on the molecular level as transcriptional activators or repressors and are involved in chromatin assembly.
L41268	Killer cell immunoglobulin-like receptors (KIRs) are transmembrane glycoproteins expressed by natural killer cells and subsets of T cells. The KIR genes are polymorphic and highly homologous and they are found in a cluster on chromosome 19q13.4 within the 1 Mb leukocyte receptor complex (LRC). The KIR proteins are classified by the number of extracellular immunoglobulin domains (2D or 3D) and by whether they have a long (L) or short (S) cytoplasmic domain. The ligands for several KIR proteins are subsets of HLA class I molecules; thus, KIR proteins are thought to play an important role in regulation of the immune response.
R99200	The protein encoded by this gene is a beta-amyloid peptide-binding protein. Beta-amyloid peptide has been established to be a causative factor in neuron death and the consequent dimunition of cognitive abilities observed in Alzheimer's disease. This protein may be a target of neurotoxic beta-amyloid peptide and may mediate cellular vulnerability to beta-amyloid peptide toxicity through a G protein-regulated program of cell death.
R53941	The protein encoded by this gene is a GTPase which belongs to the RAS superfamily of small GTP-binding proteins. Members of this superfamily appear to regulate a diverse array of cellular events, including the control of cell growth, cytoskeletal reorganization, and the activation of protein kinases.
R51502	This gene encodes one of four subunits of the splicing factor 3B. The protein encoded by this gene cross-links to a region in the pre-mRNA immediately upstream of the branchpoint sequence in pre-mRNA in the prespliceosomal complex A. It also may be involved in the assembly of the B, C, and E spliceosomal complexes. In addition to RNA-binding activity, this protein interacts directly and highly specifically with subunit 2 of the splicing factor 3B.
X61123	The BTG1 gene locus has been shown to be involved in a t(8;12)(q24;q22) chromosomal translocation in a case of B-cell chronic lymphocytic leukemia. It is a member of a family of antiproliferative genes. BTG1 expression is maximal in the G0/G1 phases of the cell cycle and downregulated when cells progressed through G1. It negatively regulates cell proliferation.
H05814	This gene encodes a DEAD box protein. DEAD box proteins, characterized by the conserved motif Asp-Glu-Ala-Asp (DEAD), are putative RNA helicases. They are implicated in a number of cellular processes involving alteration of RNA secondary structure.
X66363	It may play a role in signal transduction cascades in terminally differentiated cells. This gene is thought to escape X inactivation.
K03460	Tubulin is the major constituent of microtubules. It binds two moles of GTP, one at an exchangeable site on the beta chain and one at a nonexchangeable site on the alpha-chain.
K02566	The active peptide bradykinin that is released from HMW-kininogen shows a variety of physiological effects: (4A) influence in smooth muscle contraction; (4B) induction of hypotension; (4C) natriuresis and diuresis; (4D) decrease in blood glucose level; (4E) it is a mediator of inflammation and causes (4E1) increase in vascular permeability; (4E2) stimulation of nociceptors (4E3) release of other mediators of inflammation (e.g., prostaglandins); (4F) it has a cardioprotective effect (directly via bradykinin action, indirectly via endothelium-derived relaxing factor action).
H05978	It could have a dual role in dynein targeting and in ACTR1A/Arp1 subunit of dynactin pointed-end capping. It could be involved in ACTR1A pointed-end binding and in additional roles in linking dynein and dynactin to the cortical cytoskeleton.
U20659	This gene encodes the seventh largest subunit of RNA polymerase II, the polymerase responsible for synthesizing messenger RNA in eukaryotes. In yeast, the association of this subunit with the polymerase under suboptimal growth conditions indicates that it may play a role in regulating polymerase function.
X17042	This gene encodes a protein best known as a hematopoietic cell granule proteoglycan. Proteoglycans stored in the secretory granules of many hematopoietic cells also contain a protease-resistant peptide core, which may be important for neutralizing hydrolytic enzymes. This encoded protein was found to be associated with the macromolecular complex of granzymes and perforin, which may serve as a mediator of granule-mediated apoptosis.
Z49269	This gene, CCL14, is one of several CC cytokine genes clustered on 17q11.2. The CC cytokines are secreted proteins characterized by two adjacent cysteines. The cytokine encoded by this gene induces changes in intracellular calcium concentration and enzyme release in monocytes.
H41017	Mitochondrial creatine (MtCK) kinase is responsible for the transfer of high-energy phosphate from mitochondria to the cytosolic carrier, creatine. Many malignant cancers with poor prognosis have shown overexpression of ubiquitous mitochondrial creatine kinase.
L09159	It regulates a signal transduction pathway linking plasma membrane receptors to the assembly of focal adhesions and actin stress fibers.
U31216	L-glutamate is the major excitatory neurotransmitter in the central nervous system and activates both ionotropic and metabotropic glutamate receptors. Glutamatergic neurotransmission is involved in most aspects of normal brain function and can be perturbed in many neuropathologic conditions.

Table 9: Continued.

GENE ID	Gene description
H20709	Myosin is a hexameric ATPase cellular motor protein. This gene encodes a myosin alkali light chain, that is, expressed in smooth muscle and nonmuscle tissues.
R15876	This gene encodes subunit 3 of the splicing factor 3a protein complex.
R43914	DNA- and RNA-binding protein is involved in several nuclear processes such as pre-mRNA splicing, apoptosis, and transcription regulation. In association with FUBP1 it regulates MYC transcription at the P2 promoter through the core-TFIIH basal transcription factor, involved in apoptosis induction when overexpressed in HeLa cells. Isoform 6 failed to repress MYC transcription and inhibited FIR-induced apoptosis in colorectal cancer. Isoform 6 may contribute to tumor progression by enabling increased MYC expression and greater resistance to apoptosis in tumors than in normal cells.
H79575	This gene encodes fibronectin, a glycoprotein present in a soluble dimeric form in plasma, and in a dimeric or multimeric form at the cell surface and in extracellular matrix. Fibronectin is involved in cell adhesion and migration processes including blood coagulation, host defense, and metastasis.

Table 10: Definition of the most recurrent genes contained in the best solutions found by GP on the Leukemia Dataset.

GENE ID	Gene description
M20203	Elastases form a subfamily of serine proteases that hydrolyze many proteins in addition to elastin. Humans have six elastase genes which encode the structurally similar proteins elastase 1, 2, 2A, 2B, 3A, and 3B. Elastase 2 hydrolyzes proteins within specialized neutrophil lysosomes, called azurophil granules, as well as proteins of the extracellular matrix following the protein's release from activated neutrophils.
M28130	The protein encoded by this gene is a member of the CXC chemokine family. This chemokine is one of the major mediators of the inflammatory response. This chemokine is secreted by several cell types. It functions as a chemoattractant and is also a potent angiogenic factor. This gene is believed to play a role in the pathogenesis of bronchiolitis, a common respiratory tract disease caused by viral infection.
M84526	The protein encoded by this gene is a member of the trypsin family of peptidases. The encoded protein is a component of the alternative complement pathway best known for its role in humoral suppression of infectious agents. This protein is also a serine protease, that is, secreted by adipocytes into the bloodstream. Finally, the encoded protein has a high level of expression in fat, suggesting a role for adipose tissue in immune system biology.
M96326	Azurophil granules, specialized lysosomes of the neutrophil, contain at least 10 proteins implicated in the killing of microorganisms. The protein encoded by this gene is an azurophil granule antibiotic protein, with monocyte chemotactic and antibacterial activity. It is also an important multifunctional inflammatory mediator.
Z69881	This gene encodes one of the SERCA Ca(2+)-ATPases, which are intracellular pumps located in the sarcoplasmic or endoplasmic reticula of muscle cells. This enzyme catalyzes the hydrolysis of ATP coupled with the translocation of calcium from the cytosol to the sarcoplasmic reticulum lumen and is involved in calcium sequestration associated with muscular excitation and contraction. Alternative splicing results in multiple transcript variants encoding different isoforms.
D80006	It may provide positional cues for axon pathfinding and patterning in the central nervous system.
J04990	The protein encoded by this gene, a member of the peptidase S1 protein family, is found in azurophil granules of neutrophilic polymorphonuclear leukocytes. The encoded protease has a specificity similar to that of chymotrypsin C, and may participate in the killing and digestion of engulfed pathogens and in connective tissue remodeling at sites of inflammation. Transcript variants utilizing alternative polyadenylation signals exist for this gene.
U32944	Cytoplasmic dyneins are large enzyme complexes with a molecular mass of about 1200 kD. They contain two force-producing heads formed primarily from dynein heavy chains and stalks linking the heads to a basal domain, which contains a varying number of accessory intermediate chains. The complex is involved in intracellular transport and motility. The protein described in this record is a light chain and exists as part of this complex but also physically interacts with and inhibits the activity of neuronal nitric oxide synthase. Binding of this protein destabilizes the neuronal nitric oxide synthase dimer, a conformation necessary for activity, and it may regulate numerous biologic processes through its effects on nitric oxide synthase activity.
X55668	Polymorphonuclear leukocyte serine protease degrades elastin, fibronectin, laminin, vitronectin, and collagen types I, III, and IV (in vitro) and causes enphysema when administered by tracheal insufflation to hamster.
X74262	This gene encodes a ubiquitously expressed nuclear protein which belongs to a highly conserved subfamily of WD-repeat proteins. It is present in protein complexes involved in histone acetylation and chromatin assembly. It is part of the Mi-2 complex which has been implicated in chromatin remodeling and transcriptional repression associated with histone deacetylation. This encoded protein is also part of corepressor complexes, which is an integral component of transcriptional silencing. It is found among several cellular proteins that bind directly to retinoblastoma protein to regulate cell proliferation. This protein also seems to be involved in transcriptional repression of E2F-responsive genes.

Table 10: Continued.

GENE ID	Gene description
M26602	Defensins are a family of microbicidal and cytotoxic peptides thought to be involved in host defense. The protein encoded by this gene, defensin, alpha 1, is found in the microbicidal granules of neutrophils and likely plays a role in phagocyte-mediated host defense.
M57731	It is produced by activated monocytes and neutrophils and expressed at sites of inflammation. Hematoregulatory chemokine, which, in vitro, suppresses hematopoietic progenitor cell proliferation. GRO-beta(5-73) shows a highly enhanced hematopoietic activity.
M27891	This gene is located in the cystatin locus and encodes the most abundant extracellular inhibitor of cysteine proteases, which is found in high concentrations in biological fluids and is expressed in virtually all organs of the body. A mutation in this gene has been associated with amyloid angiopathy.
U05259	The B lymphocyte antigen receptor is a multimeric complex that includes the antigen-specific component, surface immunoglobulin (Ig). Surface Ig noncovalently associates with two other proteins, Ig-alpha and Ig-beta, which are necessary for expression and function of the B-cell antigen receptor. This gene encodes the Ig-alpha protein of the B-cell antigen component.
U85767	This gene is one of several cytokine genes clustered on the q-arm of chromosome 17. Cytokines are a family of secreted proteins involved in immunoregulatory and inflammatory processes. The CC cytokines are proteins characterized by two adjacent cysteines. The cytokine encoded by this gene displays chemotactic activity on resting T lymphocytes and monocytes, lower activity on neutrophils and no activity on activated T lymphocytes. The protein is also a strong suppressor of colony formation by a multipotential hematopoietic progenitor cell line.
J04615	The protein encoded by this gene is one polypeptide of a small nuclear ribonucleoprotein complex and belongs to the snRNP SMB/SMN family. The protein plays a role in pre-mRNA processing, possibly tissue-specific alternative splicing events. Although individual snRNPs are believed to recognize specific nucleic acid sequences through RNA-RNA base pairing, the specific role of this family member is unknown.
X17042	This gene encodes a protein best known as a hematopoietic cell granule proteoglycan. Proteoglycans stored in the secretory granules of many hematopoietic cells also contain a protease-resistant peptide core, which may be important for neutralizing hydrolytic enzymes. This encoded protein was found to be associated with the macromolecular complex of granzymes and perforin, which may serve as a mediator of granule-mediated apoptosis.
M63438	HLA-C belongs to the HLA class I heavy chain paralogues. This class I molecule is a heterodimer consisting of a heavy chain and a light chain (beta-2 microglobulin). The heavy chain is anchored in the membrane. Class I molecules play a central role in the immune system by presenting peptides derived from endoplasmic reticulum lumen.
X95735	Focal adhesions are actin-rich structures that enable cells to adhere to the extracellular matrix and at which protein complexes involved in signal transduction assemble. Zyxin is a zinc-binding phosphoprotein that concentrates at focal adhesions and along the actin cytoskeleton. Zyxin has an N-terminal proline-rich domain and three LIM domains in its C-terminal half. The proline-rich domain may interact with SH3 domains of proteins involved in signal transduction pathways while the LIM domains are likely involved in protein-protein binding. Zyxin may function as a messenger in the signal transduction pathway that mediates adhesion-stimulated changes in gene expression and may modulate the cytoskeletal organization of actin bundles.
M69043	It inhibits the activity of dimeric NF-kappa-B/REL complexes by trapping REL dimers in the cytoplasm through masking of their nuclear localization signals. On cellular stimulation by immune and proinflammatory responses, it becomes phosphorylated promoting ubiquitination and degradation, enabling the dimeric RELA to tranlocate to the nucleus and activate transcription.
U49869	This gene encodes ubiquitin, one of the most conserved proteins known. Ubiquitin is required for ATP-dependent, nonlysosomal intracellular protein degradation of abnormal proteins and normal proteins with a rapid turnover. Ubiquitin is covalently bound to proteins to be degraded and presumably labels these proteins for degradation. Ubiquitin also binds to histone H2A in actively transcribed regions but does not cause histone H2A degradation, suggesting that ubiquitin is also involved in regulation of gene expression. This gene consists of three direct repeats of the ubiquitin coding sequence with no spacer sequence. Consequently, the protein is expressed as a polyubiquitin precursor with a final amino acid after the last repeat.
M17733	This gene encodes an actin sequestering protein which plays a role in regulation of actin polymerization. The protein is also involved in cell proliferation, migration, and differentiation. This gene escapes X inactivation and has a homolog on chromosome Y.
M19507	Myeloperoxidase (MPO) is a heme protein synthesized during myeloid differentiation that constitutes the major component of neutrophil azurophilic granules.
U46751	It is an adapter protein which binds ubiquitin and may regulate the activation of NFKB1 by TNF-alpha, nerve growth factor (NGF), and interleukin-1. It may play a role in titin/TTN downstream signaling in muscle cells, may regulate signaling cascades through ubiquitination, may be involved in cell differentiation, apoptosis, immune response, and regulation of K(+) channels.
X52056	This gene encodes an ETS-domain transcription factor that activates gene expression during myeloid and B-lymphoid cell development.

They are entries of the GenBank database (see e.g., http://www.ncbi.nlm.nih.gov/Genbank/ for a description of this database of known genes). Other informations about these genes can be obtained by using these IDs as entries at the page http://smd.stanford.edu/cgi-bin/source/sourceBatchSearch.

References

[1] P. Russel, *Fundamentals of Genetics*, Addison-Wesley, Reading, Mass, USA, 2000.

[2] J. Koza, *Genetic Programming*, MIT Press, Cambridge, Mass, USA, 1992.

[3] Y. Lu and J. Han, "Cancer classification using gene expression data," *Information Systems*, vol. 28, no. 4, pp. 243–268, 2003.

[4] D. Michie, D.-J. Spiegelhalter, and C.-C. Taylor, *Machine Learning, Neural and Statistical Classification*, Prentice-Hall, Upper Saddle River, NJ, USA, 1994.

[5] U. Alon, N. Barkai, D. A. Notterman, et al., "Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 96, no. 12, pp. 6745–6750, 1999.

[6] A. L. Hsu, S.-L. Tang, and S. K. Halgamuge, "An unsupervised hierarchical dynamic self-organizing approach to cancer class discovery and marker gene identification in microarray data," *Bioinformatics*, vol. 19, no. 16, pp. 2131–2140, 2003.

[7] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, "Gene selection for cancer classification using support vector machines," *Machine Learning*, vol. 46, no. 1–3, pp. 389–422, 2002.

[8] J. C. Hernandez, B. Duval, and J.-K. Hao, "A genetic embedded approach for gene selection and classification of microarray data," in *Proceedings of the 5th European Conference on Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics (EvoBIO '07)*, vol. 4447 of *Lecture Notes in Computer Science*, pp. 90–101, Springer, Valencia, Spain, April 2007.

[9] N. Friedman, M. Linial, I. Nachman, and D. Pe'er, "Using Bayesian networks to analyze expression data," *Journal of Computational Biology*, vol. 7, no. 3-4, pp. 601–620, 2000.

[10] J. H. Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, Mich, USA, 1975.

[11] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Reading, Mass, USA, 1989.

[12] J. J. Liu, G. Cutler, W. Li, et al., "Multiclass cancer classification and biomarker discovery using GA-based algorithms," *Bioinformatics*, vol. 21, no. 11, pp. 2691–2697, 2005.

[13] J.-H. Moore, J.-S. Parker, and L.-W. Hahn, "Symbolic discriminant analysis for mining gene expression patterns," L. De Raedt and P. Flach, Eds., vol. 2167 of *Lecture Notes in Artificial Intelligence*, pp. 372–381, Springer, Berlin, Germany, 2001.

[14] M. Rosskopf, H. A. Schmidt, U. Feldkamp, and W. Banzhaf, "Genetic programming based DNA microarray analysis for classification of tumour tissues," Tech. Rep. 2007-03, Memorial University of Newfoundland, 2007.

[15] J. Yu, J. Yu, A. A. Almal, et al., "Feature selection and molecular classification of cancer using genetic programming," *Neoplasia*, vol. 9, no. 4, pp. 292–303, 2007.

[16] C. C. Bojarczuk, H. S. Lopes, and A. A. Freitas, "Data mining with constrained-syntax genetic programming: applications to medical data sets," in *Proceedings of the Intelligent Data Analysis in Medicine and Pharmacology*, 2001.

[17] J.-H. Hong and S.-B. Cho, "The classification of cancer based on DNA microarray data that uses diverse ensemble genetic programming," *Artificial Intelligence in Medicine*, vol. 36, no. 1, pp. 43–58, 2006.

[18] T. R. Golub, D. K. Slonim, P. Tamayo, et al., "Molecular classification of cancer: class discovery and class prediction by gene expression monitoring," *Science*, pp. 531–537, 1999.

[19] M. Keijzer, "Scaled symbolic regression," *Genetic Programming and Evolvable Machines*, vol. 5, no. 3, pp. 259–269, 2004.

[20] C. E. Metz, "Basic principles of ROC analysis," *Seminars in Nuclear Medicine*, vol. 8, no. 4, pp. 283–298, 1978.

[21] M. H. Zweig and G. Campbell, "Receiver-operating characteristic (ROC) plots: a fundamental evaluation tool in clinical medicine," *Clinical Chemistry*, vol. 39, no. 4, pp. 561–577, 1993.

[22] V. Vapnik, *Statistical Learning Theory*, John Wiley & Sons, New York, NY, USA, 1998.

[23] J. Platt, "Fast training of support vector machines using sequential minimal optimization," in *Advances in Kernel Methods: Support Vector Learning*, B. Schoelkopf, C. Burges, and A. Smola, Eds., MIT Press, Cambridge, Mass, USA, 1998.

[24] Weka, a multi-task machine learning software developed by Waikato University, http://www.cs.waikato.ac.nz/ml/weka/.

[25] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[26] G. I. Webb, "MultiBoosting: a technique for combining boosting and wagging," *Machine Learning*, vol. 40, no. 2, pp. 159–196, 2000.

[27] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*, Wadsworth International Group, Belmont, Calif, USA, 1984.

[28] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[29] L. Vanneschi, D. Rochat, and M. Tomassini, "Multi-optimization for generalization in symbolic regression using genetic programming," in *Proceedings of the 2nd Annual Italian Workshop on Artificial Life and Evolutionary Computation (WIVACE '07)*, G. Nicosia, et al., Ed., 2007.

Hindawi Publishing Corporation
Journal of Artificial Evolution and Applications
Volume 2009, Article ID 130498, 12 pages
doi:10.1155/2009/130498

Research Article

Conserved Self Pattern Recognition Algorithm with Novel Detection Strategy Applied to Breast Cancer Diagnosis

Senhua Yu and Dipankar Dasgupta

Department of Computer Science, University of Memphis, Memphis, TN 38152, USA

Correspondence should be addressed to Senhua Yu, senhuayu@gmail.com

Received 1 October 2008; Accepted 22 May 2009

Recommended by Marylyn Ritchie

This paper presents a novel approach based on an improved Conserved Self Pattern Recognition Algorithm to analyze cytological characteristics of breast fine-needle aspirates (FNAs) for clinical breast cancer diagnosis. A novel detection strategy by coupling domain knowledge and randomized methods is proposed to resolve conflicts on anomaly detection between two types of detectors investigated in our earlier work on Conserved Self Pattern Recognition Algorithm (CSPRA). The improved CSPRA is applied to detect the malignant cases using clinical breast cancer data collected by Dr. Wolberg (1990), and the results are evaluated for performance measure (detection rate and false alarm rate). Results show that our approach has promising performance on breast cancer diagnosis and great potential in the area of clinical diagnosis. Effects of parameters setting in the CSPRA are discussed, and the experimental results are compared with the previous works.

1. Introduction

Normally, breast cells grow and then rest in cycles. The periods of growth and rest in each cell are controlled by genes in the cell's nucleus. Genes sometimes develop abnormalities, which cause to lose their ability to control the cycle resulting in uncontrolled growth of breast cells (cancer). Breast cancer is the most common cancer and the second largest cause of cancer deaths among women [1]. Based on the estimation (from www.BreastCancer.org) in 2007, there are about 178 480 new cases of invasive breast cancer and 62 030 new cases of noninvasive breast cancer diagnosed in the United States. Early detection of this disease via accurate diagnosis and treatment can greatly improve the chances for survival.

Most breast cancers are symptomatic of lump in the breast but the majority of breast lumps are benign. Therefore, it is important to distinguish benign lumps from malignant ones. The methods for diagnosing breast cancer include mammography, fine-needle aspirates (FNAs) with visual interpretation, and surgical biopsy. The detection ability of both mammography and fine-needle aspirates with visual interpretation to correctly diagnose breast cancer is unstable [2, 3]. Surgical biopsy, although accurate, is invasive, time consuming, and costly. The anticipated course of the cancer not only determines whether chemotherapy is needed but also affects the mental state and personal goal of the patient. Therefore, developing an accurate, efficient, and inexpensive method for breast cancer diagnosis is an important and challenging goal for the treatment of this disease.

The great economic and social values of breast cancer diagnosis have attracted many researchers. Some methods such as linear programming [4, 5], neural network [6, 7], and ant colony-based system [8] were applied to breast cancer diagnosis. Over the last decade, immunity-based approaches have been applied to solve problems in a wide variety of domains such as anomaly detection, pattern recognition, data mining, computer security, adaptive control, and fault detection [9]. The majority of breast lumps are benign, which could in practice provide a large training data set of normal class. Hence, Artificial Immune System (AIS) can be applied to breast cancer diagnosis by taking advantage of one-class classification. In this paper, we describe an improved Conserved Self Pattern Recognition Algorithm (CSPRA) for breast cancer diagnosis. The details of the algorithm are described in Section 2 and Section 3 outlines the breast cancer diagnosis problem. Section 4 describes how the algorithm can be applied to the breast cancer diagnosis problem and reports some experimental results. Section 5 provides discussions on algorithmic parameters and comparative results. The last section provides conclusive remarks and future work.

2. A Novel Detection Strategy in Conserved Self Pattern Recognition Algorithm (CSPRA)

2.1. A Brief Overview of CSPRA. The immune system plays roles by discriminating self (defined early in life) and nonself (anything that comes later), tolerating self, and attacking nonself [10]. This elegantly simple idea, known as the Self-Nonself model, has dominated the field for over 50 years. However, it has failed to explain a great number of findings such as alerted self, pregnancy, and aging [10]. Pattern Recognition Receptors (PRRs) model was published in 1989 to accommodate incompatible new findings [11]. The PRRs model suggested that Antigen Presenting Cells (APCs) can recognize evolving pathogens. The lymphocytes (T cell or B cell) would die if it recognized antigen (Signal 1) without the costimulation from APC (Signal 2) but APCs do not costimulate unless activated via encoded PRRs that recognize conserved pathogen-associated molecular patterns (PAMPs) on bacteria [11]. Inspired by the biological PRRs model, we recently proposed a novel algorithm called Conserved Self Pattern Recognition Algorithm (CSPRA) [12]. The basic steps involved in the CSPRA are as follows.

(1) Build up the "Self" training samples from the collected normal data and store the samples as a multiset S_0 of equal length strings, L over a finite alphabet.

(2) Establish a model of normal behavior by generating the set of T detectors (R_1) and a specific APC detector (R_2), respectively. The negative selection strategy [13] is employed in the generation of T detectors. However, the generation of the specific APC detector includes two major steps.

   (a) Based on the relationship between the antigen objects and the dimensions of their feature space, define the *conserved self pattern*. It can be predefined from the empirical data based on the scientists' lab results. This paper introduces a new technique for finding conserved self pattern, as described in Section 4.1.

   (b) Within the conserved self pattern consisting of the features located in loc 1, loc 2, . . . , generate APC detector $R_2\{\langle \text{loc}\,1, \text{min}, \text{max}, \text{mean}\rangle, \langle \text{loc}\,2, \text{min}, \text{max}, \text{mean}\rangle, \ldots\}$ by calculating maximum, minimum, and mean of all of the values in the features (or descriptors) of loc 1, loc 2, . . . , respectively.

(3) Monitor the system by detecting anomaly in the incoming new data in the testing data set S_1 using the generated T detectors and APC detector in R_1 and R_2. The matching rule (Euclidean distance) is used for T detectors to report anomaly. The distance between APC detector, and the new sample is calculated by (1). If it is *greater* than the predefined threshold, then the anomaly is detected by APC detector. In (1), w is the number of the dimensions for the conserved pattern; m_i and n_i represent the lower and upper bounds of the ith attribute in the entire training data, respectively; $p = (p_1, p_2, \ldots, p_w)$, p_i is the value of the ith attribute for the antigen object to be examined; $d = (d_1, d_2, \ldots, d_w)$, d_i is the mean of all of the values in the ith attribute in the entire training data:

$$\text{Dist}(p, d) = \sum_{i=1}^{w} \frac{|\, p_i - d_i \,|}{m_i - n_i}. \qquad (1)$$

(4) The new sample (antigen) is firstly checked with T detectors. The costimulation of APC detector is conducted if and only if *both* of the following conditions are fulfilled during the phase of T cell detection.

   (a) The affinity between the T cell detector and the new sample is very low, that is, the Euclidean distance between the T cell detector and the incoming sample is greater than predefined suspicious threshold.

   (b) The decision for abnormal (non-self) is made based on the other antigen epitope instead of the antigen peptide, where the conserved self pattern is located.

   The new sample satisfying the above conditions is named *suspicious antigen*. APC detector finally determines whether it is anomaly in this case.

Now, we extend the proposal in [12] aiming at resolving the conflicts in anomaly detection between two types of detectors in the detection phase of the algorithm. By wisely using domain knowledge and randomized method, the novel detection strategy we present in this paper has been proven to greatly enhance the efficiency for anomaly detection by the unpublished results when testing with multiple data sets. In this paper, we center on the application of this modified Conserve Self Pattern Recognition Algorithm to breast cancer diagnosis. In this section, we are going to describe the novel detection strategy, the interesting readers can refer to [12] for the details of the algorithm.

In biology, the PRRs model added additional layer of pathogen-associated molecular patterns (PAMPs) to the self-nonself model. Inspired by this metaphor, our earlier proposal in [12] combined both APCs Pattern Recognition and T cell Negative Selection to detect anomalies in new samples, which had been proven to efficiently reduce high false positive error rate that often occurred in Negative Selection Algorithm (NSA). By exploring this idea, a question is naturally raised: if the conflicts on anomaly decision happen when testing the new samples using APC detector and T cell detector, respectively, for example, T cell detector

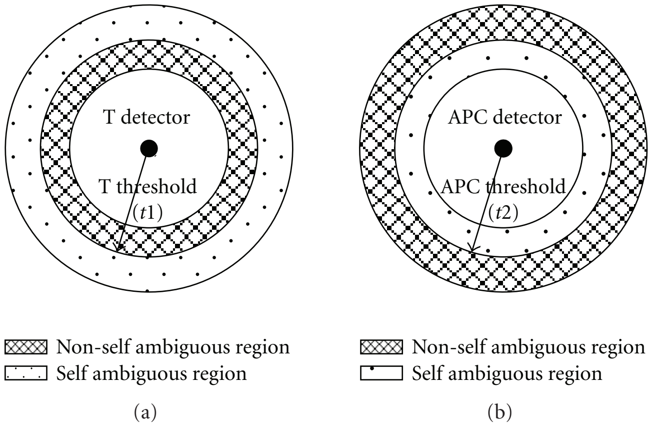


Figure 1: Interpretations of ambiguous boundary on anomaly detection.

recognizes the new sample as "Nonself" whereas the same sample is classified as "Self" by APC detector, how does the system make the final decision for the new sample? The solution in the previous proposal mirrors the biological metaphor: APCs are quiescent until they are activated via encoded PRRs that recognize conserved PAMPs [11]. The costimulation of APC detector will not be conducted until the detection from T cell detectors becomes unsure, that is, the *suspicious antigen* is encountered in the system. Although this solution shows its strength in terms of algorithmic complexity, its performance relies on the application domain since the definition of suspicious antigen is not always in accordance with a specific application. Our recent work proposes the mathematical methods to resolve the conflicts.

2.2. Ambiguous Boundary in Anomaly Detection. An important concept in our proposal is *ambiguous boundary*. Figure 1 illustrates the interpretations of ambiguous boundaries for T cell detection and APC detection, respectively. The representation for the single T detector in Figure 1(a) is different in two detection cases.

(i) If the new sample matches with *any* of the T detectors, that is, the distance between the new sample and the current T detector is less than the predefined threshold, then the new sample is detected as "non-self." The single T detector in Figure 1(a) represents the one in the set of T detectors that recognizes the new sample that is currently being tested.

(ii) The new sample is detected as "self" if it *does not* match *any* T detectors. In this case, the single T detector in Figure 1(a) is the one in the set of T detectors that is the nearest to the new sample that is currently being tested. From the viewpoint of algorithm implementation, the Euclidean distance between the new sample and each T detector is calculated while the testing sample is checked against all T detectors but only the shortest distance is returned.

Now, it becomes very straightforward to explain the ambiguous boundary for both T detector and APC detector. The *ambiguous coefficient* α in this proposal is applied to adjust the range of the ambiguous region, that is, to increase/decrease the shadow area in Figure 1.

(i) As shown in Figure 1(a), the ambiguous region for T detector is a ring-shaped region contained between the two circles (upper ambiguous boundary and lower ambiguous boundary) with the radius of $(1+\alpha_t)^* t_1$ (outer circle) and $(1-\alpha_t)^* t_1$ (inner circle), where t_1 represents the threshold for T detector activation and α_t is the ambiguous coefficient for T detector detection.

(ii) Similarly, as shown in Figure 1(b), the ambiguous region for APC detector is also a ring-shaped region contained between the two circles with the radius of $(1+\alpha_p)^* t_2$ (outer circle) and $(1-\alpha_p)^* t_2$ (inner circle), where t_2 represents the threshold for APC detector activation and α_p is the ambiguous coefficient for APC detector detection.

2.3. A Novel Detection Strategy to Resolve the Conflicts on Anomaly Detection. Once the ambiguous boundary is defined, the following rules are set to resolve the conflicts in anomaly decision between T detector and APC detector by using domain knowledge and randomized method.

2.3.1. Domain Knowledge. When the antigen (the new incoming sample) is loaded into the system, we check the matching rule of Euclidean distance against each T detector. If the matching is found, we keep the matching distance in record. If the matching is *not* found after *all* T detectors are checked, we only keep the distance to the nearest T detectors in record. Similarly, the matching distance is also kept in record when the antigen is checked against the APC detector. The information from the matching distances for both T detector and APC detector obtained in the detection phase is considered as valuable domain knowledge, which is used in the proposed detection strategy to resolve the conflicts in the following cases.

Case 1. The anomaly decision from APC detector is granted but the results from T detector detection are discarded if and only if *both* of the following conditions are fulfilled:

(i) the new sample falls into the ambiguous region that is predefined by the ambiguous boundary when testing with T detector;

(ii) the new sample is *not* in the ambiguous region when testing with APC detector.

Case 2. It is opposite of Case 1. T detector finally determines whether the new sample is anomalous but APC detection is simply ignored if and only if *both* of the following conditions are fulfilled:

(i) the new sample is *not* in the ambiguous region when testing with T detector;

(ii) the new sample is in the ambiguous region when testing with APC detector.

2.3.2. Randomized Method. Randomized method is selected to resolve the conflicts for all other cases rather than Cases 1 and 2 described in Section 2.3.1. In other words, randomized method is used under one of the following conditions as (1) the new sample falls into the ambiguous region when it is matched against with either T detector or APC detector; (2) the new sample does *not* fall into the ambiguous region when it is matched against with either T cell detector or APC detector.

When implementing this algorithm, a list $L(n_1, n_2, n_3, n_4, n_5 \ldots)$ is established in which to store these ambiguous testing samples while the system loops through every testing sample. Another parameter called *pattern weight* w is introduced in the algorithm to control the randomized decision making. When the loop is terminated, the algorithm randomly picks up n^*w samples, provided that the list L contains n ambiguous samples, to form a new sublist $l(m_1, m_2, m_3, \ldots)$. For every sample in the sublist l, APC detector *solely* determines whether it is anomalous regardless of the results from T detector decision. Anomaly decision for the remaining $n^*(1-w)$ samples in the parent list L follows T detector decision process.

2.4. Pseudocode for the Detection Algorithm. Listed below is the pseudocode for the detection algorithm that reflects the novel detection strategy we have outlined in the previous sections.

As noted in Algorithm 1, in lines 2 and 3, each sample is examined by T detector and APC detector, respectively. We keep the distance (d_t and d_p) in record when we check the matching rule for both APC detector and T detector, which are used to determine the resolution of the conflicts, as seen between lines 9 and 15. In lines 14 and 15, we save the index (i), T detector detection result (*TDecision*), and APC detector detection result (*APCDecision*) to the defined *struct* for each ambiguous testing sample (*ambiguousAg*), and then add *ambiguousAg* to the list of the ambiguous testing samples (*randList*). Hence, we no longer loop through all testing samples again when we determine the anomaly for the new sample using randomized method in lines 31 and 34, which greatly enhances the algorithm efficiency. As shown in line 20 in Algorithm 1, in the list of the ambiguous testing samples, the total number of the samples to be determined by APC conserved pattern recognition is affected by the pattern weight w. Line 23 through line 35 show how to determine the anomalies using randomized method. The pseudocode shown in Algorithm 1 is only part of the algorithm implementation showing the difference from our previous proposal in the detection phase for this algorithm. Readers interested in this information can refer to [12] for the entire pseudocode for the algorithm implementation.

3. Breast Cancer Diagnosis Problem

This breast cancer database is downloaded from the UCI machine learning repository [14], which was collected by Dr. William H. Wolberg from the University of Wisconsin Hospitals, Madison, USA [4]. The dataset is comprised of elements that consist of various scalar observations. The total number of the original samples is 699 but 16 samples with missing values are removed to construct a new dataset with 683 samples that are actually used in our experiments. The dataset contains two classes referring to benign and malignant samples. There are 444 samples in the dataset that are assigned to benign, and the other 239 samples are malignant. The original dataset contains 11 attributes including both sample id number and class label, which are removed in the actual dataset that are used in our experiments. The remaining 9 attributes represent 9 cytological characteristics of breast fine-needle aspirates (FNAs), as shown in Table 1. The cytological characteristics of breast FNAs were valued on a scale of one to ten, with one being the closest to benign and ten the most malignant.

Samples arrived periodically as Dr. Wolberg reported his clinical cases [4, 5]. The original database therefore includes the information about this chronological grouping of the data, having been removed from the data itself. Some brief statistical analysis is presented in Table 2. The calculation of class correlation in Table 2 is introduced in Section 4.1.

Prior to the experiments, we normalize the raw data by columns in the range of [0,1] with max-min normalization, as shown in

$$f(x) = \frac{x - \min(\text{column})}{\max(\text{column}) - \min(\text{column})}. \tag{2}$$

4. Application of CSPRA to Breast Cancer Diagnosis

In this section we begin by describing the data preprocessing, then further to discuss the algorithm parameters and report the experimental results showing that the algorithm is able to predict the breast cancer quite efficiently.

4.1. Finding Conserved Self pattern. The first task to use CSPRA is to find the conserved self pattern based on training data. The *Pearson Product Moment Correlation Coefficient* that was developed by Karl Pearson is the most widely used

```
//detection phase
$S$: set of testing samples
$t_1$: T detector threshold
$t_2$: APC detector threshold
$\alpha_t$: ambiguous coefficient for T detector detection
$\alpha_p$: ambiguous coefficient for APC pattern detection
$d_t$: distance between T detector and the current sample
$d_p$: distance between APC detector and the current sample
$w$: pattern weight used in random decision
Decision*: bool variable for the final detecting conclusion for the new sample
TDecision*: bool variable for the detecting conclusion from T detector
APCDecision*: bool variable for the detecting conclusion from APC detector
ambiguousAg: struct for each new sample to be decided on anomaly with randomized method
randList: list of ambiguousAg
*For these bool variables, true for anomaly and false for self
for every $s_i$ in $S = \{s_i, i = 1, 2, \ldots\}$
    TDecision = CheckWithTDetector($s_i, t_{1,} d_t$)
    APCDecision = CheckWithAPCDetector($s_i, t_{2,,} d_p$)
    if(TDecision && APCDecision)
        Decision = true;
    else if((!TDecision) && (!APCDecision))
        Decision = false;
    else
        if( $((d_t > (1+\alpha_t)^* t_1) \| (d_t < (1-\alpha_t)^* t_1))$ &&
        $((d_p > (1-\alpha_p)^* t_2)$ && $(d_p < (1+\alpha_p)^* t_2))$
            Decision = TDecision
        else if( $((d_t < (1+\alpha_t)^* t_1)$ && $(d_t > (1-\alpha_t)^* t_1))$ &&
        $((d_p > (1+\alpha_p)^* t_2) \| (d_p < (1-\alpha_p)^* t_2))$
            Decision = APCDecision
        else
            save $i$, TDecision, and APCDecision to the struct ambiguousAg
            Add ambiguousAg to the list randList
        end else
    end else
end for
int total_ambiguous = size of the list randList
int total_apc_decided = (int) total_ambiguous $^* w$
APCList: list of the index of the samples to be decided by APC detector in randList
TList: list of the index of the samples to be decided by T detector in randList
while(size of APCList < total_apc_decided)
    int val = rand()%total_ambiguous
    if(val doesn't exist in APCList)
        Add val to APCList
end while
for(int $i = 0$; $i <$ total_ambiguous; $i$ ++)
    if($i$ doesn't exist in APCList)
        Add $i$ to TList
end for
for(int $i = 0$; $i <$ size of APCList; $i$ ++)
    Decision = randList[APCList[$i$]].APCDecision
end for
for(int $i = 0$; $i <$ size of TList; $i$ ++)
    Decision = randList[TList[$i$]].TDecision
end for
```

Algorithm 1: Detection algorithm with new decision strategy.

Table 1: Attributes in the breast cancer databases.

No.	Attribute
0	Clump thickness
1	Uniformity of cell size
2	Uniformity of cell shape
3	Marginal adhesion
4	Single epithelial cell size
5	Bare nuclei
6	Bland chromatin
7	Normal nucleoli
8	Mitoses

Table 2: Statistical analysis of attribute values in 683 samples.

Attribute no.	Mean	Standard deviation	Class correlation
0	4.44	2.82	0.7148
1	3.15	3.07	0.8208
2	3.22	2.99	0.8219
3	2.83	2.86	0.7063
4	3.23	2.22	0.6910
5	3.54	3.64	0.8227
6	3.44	2.45	0.7582
7	2.87	3.05	0.7187
8	1.60	1.73	0.4234

measure of correlation between two variables X and Y [15]. The correlation coefficient r can be simply described as the sum of the product of the Z-scores for the two variables divided by the number of scores as shown in

$$r = \frac{\sum z_X z_Y}{N}. \tag{3}$$

However, it is fairly difficult to calculate the correlation coefficient using (3). We use the computational formula that is mathematically identical but is much easier to use, as shown in (4), to calculate the Pearsonian r between the values (X) in the column of each attribute and their corresponding class labels (Y) in breast cancer data. The corresponding class correlations for different sample size in breast cancer data are listed in Table 3

$$r = \frac{N\sum XY - (\sum X)(\sum Y)}{\sqrt{N\sum X^2 - (\sum Y)^2}\sqrt{N\sum Y^2 - (\sum Y)^2}}. \tag{4}$$

From the results in Table 3, the cytological characteristics of breast fine-needle aspirates such as uniformity of bare nuclei, uniformity of cell shape and cell size are ranked first three places in terms of class correlation. Hence, the subset from 1st, 2nd, and 5th (zero-based index) dimension of the original dimensions in training data is considered as *conserved self pattern*.

As noted, the abnormal samples (malignant samples for this application) are required if the conserved self pattern is defined by computing the class correlation. However, as shown in Table 3, the same conclusion about conserved self pattern is obtained when we calculate the class correlation using 100% currently collected samples, 50% of the samples from each class, and 25% of the samples from each class. It indicates that the conserved self pattern can be found with Pearsonian r method even if the samples from abnormal behaviors are smaller. With this observation, the Pearsonian r method to find conserved self pattern becomes more practical since it does not require the collection of a large number of the abnormal samples.

4.2. Effectiveness Measurement. It is a step forward from the previous works that we consider not only to what extent the system is able to detect the anomalies (malignant samples) but also to what extent the system possibly misclassifies the normal samples (benign samples) when we evaluate the system performance. Two measures of effectiveness for detecting anomaly are calculated as follows:

$$\text{DetectionRate} = \frac{TP}{TP + FN},$$
$$\text{FalseAlarmRate} = \frac{FP}{FP + TN}, \tag{5}$$

where TP, TN, FP, and FN are the counts of *true positives* (anomalous elements identified as anomalous), *true negatives* (normal elements identified as normal), *false positives* (normal elements identified as anomalous), and *false negatives* (anomalous elements identified as normal).

Various system parameters, that is, detector thresholds, control the sensitivity of the system. By employing various strategies to change the parameter values, different values for detection rate and false alarm rate are obtained that are used for plotting the Receiver Operating Characteristics (ROCs) curve, which reflects the tradeoff between false alarm rate and detection rate. Since high detection rate and low false alarm rate are the two goals between which balance is needed, the ROC curve is used in this paper to evaluate the influence of the various parameters on the system performance for breast cancer diagnosis.

4.3. Experimental Results. The experiments are carefully designed to objectively evaluate the performance of the algorithm and analyze the experimental results. By combining the idea of k-fold cross validation and the features of one-class classification of our method, three groups of training data, as seen in Table 5, are generated with the following schema to effectively reduce the ordering effect that perhaps exists in the original collections.

(i) *Scheme 1* for training data of 100% benign samples: the first training data are obtained by shuffling the original benign samples; the reordered samples are shuffled again to output the second training data. Repeating this step 10 times generates 10 different training data sets.

(ii) *Scheme 2* for training data of 25% benign samples: pick up 3 training data sets that are generated in the last three rounds of the scheme 1 to guarantee that the collected samples have been fully randomly reordered. Each dataset is partitioned into 4 subsamples, and thus total 12 subsamples are obtained.

Table 3: Attribute class correlation in breast cancer data.

No.	Attribute	Class correlation		
		100% sample	50% sample	25% sample
0	Clump thickness	0.7148	0.7513	0.7184
1	Uniformity of cell size	0.8208	0.8024	0.7598
2	Uniformity of cell shape	0.8219	0.8156	0.7905
3	Marginal adhesion	0.7063	0.6848	0.6241
4	Single epithelial cell size	0.6910	0.6889	0.6665
5	Bare nuclei	0.8227	0.8029	0.7238
6	Bland chromatin	0.7582	0.6823	0.6046
7	Normal nucleoli	0.7187	0.7025	0.6937
8	Mitoses	0.4234	0.4509	0.4439

10 subsamples are randomly selected from the total 12 subsamples to make up 10 training data of 25% benign samples.

(iii) *Scheme 3* for training data of 50% benign samples: pick up 5 randomly reordered training data sets that are generated in the last five rounds of the scheme 1. Each dataset is partitioned into 2 subsamples, and thus total 10 subsamples are obtained. The 10 subsamples are the actual 10 training data of 50% benign samples.

When all the available benign samples are used to train the enhanced CSPRA, the total 683 samples, including the same benign samples used in the training phase, are considered as testing data. Although such testing case can demonstrate the system's capability to recognize known normal data, the false alarm rate could be deceptive because the resulted model may overfit the training data. Therefore, the training data are removed in our testing with training data of 50% benign samples and 25% benign samples. When 50% benign samples are used as training data, the remaining 50% benign samples and all malignant samples are combined as testing data. Similarly, if the training data are 25% benign samples, the testing data include the remaining 75% benign samples and all malignant samples.

Through repeatedly running the algorithm and observing the generated results, the parameter settings listed in Table 4 produce better results. As noted, when 25% benign samples are used as training data, 10 different randomly reordered training data with 25% benign samples, known as *subsamples,* are generated from the original collections. The 10 subsamples are used to train our proposed model, respectively. Since T detectors are randomly generated with negative selection in the CSPRA, different values for detection rate and false alarm rates are observed. Therefore, each subsample undergoes 100 repeated runs, and the statistics for these repeated experiments is summarized and recorded. When the experiments with all 10 subsamples are completed, the average values of the statistics from 10 subsamples are calculated, which are reported in Table 5. The same procedures are applied in our experiments when 50% benign samples and 100% benign samples are used as training data, respectively.

Table 4: The values of the parameters that are used in the experiments.

Parameters	Value
T detector threshold (t_1)	0.1
APC detector threshold (t_2)	0.5
T ambiguous coefficient (α_t)	0.5
APC ambiguous coefficient (α_p)	0.4
Pattern weight (w)	0.7
T detector size	300
APC detector size	1
Sliding window size	3

As shown in Table 5, the performance of the enhanced CSPRA regarding to breast cancer diagnosis is very promising, which indicates the great potential of the AIS methods in the area of clinical diagnosis. The experimental results also show that the detection accuracy with the enhanced CSPRA is very high even if we use the smaller training data. Table 6 shows the best experimental results we picked out from 100 repeated experiments for each training data of 100%, 50%, 25% benign samples by comprehensively considering the balance of the detection rate and false alarm rate. To clarify this, the best result is actually chosen among the experiments having the highest accuracy rate in the 100 repeated experiments. Because there are 10 randomly-reordered subsamples for each group of training data, the values of detection rate and false alarm rate reported in Table 6 are the average of the best results obtained from the experiments with 10 subsamples. As seen in Table 6, the detection rate is 98.74% companying with the false alarm rate of 2.23% with the training data of 100% benign samples. When training with 50% benign samples, the detection rate is 99.54% at the false alarm rate of 4.23%. When we use the smaller training data (25% benign samples), the result is also very positive: the detection rate is 99.62% and the false alarm rate is 6.82%.

By closely observing the influence of unknown normal data when using our algorithm, less normal training data only slightly increase the false alarm rate in breast cancer diagnosis. However, in a negative selection algorithm and its

Table 5: Detection rate and false alarm rate with different training data.

Training data	Detection rate (%)				False alarm rate (%)			
	Max	Min	Mean	SD	Max	Min	Mean	SD
100% benign samples	99.08	95.10	97.15	0.84	3.13	2.05	2.60	0.24
50% benign samples	99.87	96.53	98.40	0.67	7.48	3.69	5.44	0.83
25% benign sample	99.96	97.36	99.08	0.55	10.45	6.67	8.47	0.77

Table 6: Best results picked out from 100 repeated experiments for each training data of 100%, 50%, 25% benign samples.

Training data	Detection rate (%)	False alarm rate (%)
100% benign samples	98.74	2.23
50% benign samples	99.54	4.23
25% benign samples	99.62	6.82

variants, the false alarm rate usually increases dramatically when the normal training data decrease [16]. This phenomenon exactly demonstrates the strength of APC detector in the CSPRA. By comparison of canonical negative selection algorithm, the complexity of the proposed algorithm has not been increased. As shown in Table 4, the size of the APC detector is only 1. The proposed algorithm acts as adding only one robust detector (APC detector), which is used to detect conserved self pattern, to the total size of the detectors in a negative selection algorithm. The only observed side-effect for APC detector in the CSPRA is that the system produces the average false alarm rate of 2.23% even if we use 100% training normal samples whereas the false alarm for this case is zero in the negative selection algorithm. However, as already discussed, the false alarm rate is deceptive when all normal samples are used to train the system and are included in the testing data.

4.4. Influence of Various Parameters in the Algorithm. To further explore the effects of various control parameters in the algorithm on the performance of breast cancer prediction, experiments are done by changing the values for a certain parameter while the values for other parameters remain unchanged as listed in Table 4. The results reported in this section are obtained from the experiments with training data set of 50% benign samples randomly selected from the original collections.

Figures 2, 3, 4, and 5 show that both T detector threshold and APC detector threshold strongly affect the system performance. Both detection rate and false alarm rate increase when the T detector threshold increases. As shown in Figure 3, the detection rate rises rapidly before the value for T detector threshold reaches 0.1. However, when the T detector threshold is over 0.1, the false alarm rate starts a slow increment. APC detector threshold influences both the detection rate and false alarm rate along the opposite direction as seen in Figure 5; that is, both detection rate and false alarm rate decrease when the APC detector threshold increases. We find that the false alarm rate drops dramatically when the APC threshold gradually increases to 0.5 whereas the decrement of the detection rate speeds up when the APC threshold is greater than 0.5.

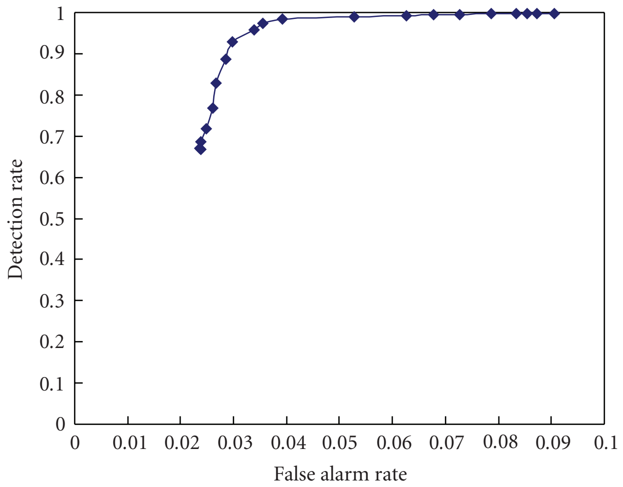


Figure 2: Performance (ROC) curve obtained by changing T detector threshold with the training data of 50% normal samples.

The influence of the parameter of pattern weight on the system performance is also studied. However, the curves of both detection rate and false alarm rate are close to being a horizontal line in Figure 6, which indicates that the influence of the parameter of pattern weight is very slight when the breast cancer data are tested with the proposed algorithm. The influence of T detector size on the system performance is also limited based on the experimental results reported in Figure 7. The detection rate almost arrives at 100% when the size of the T detectors increases to 250. The larger T detector set (greater than 250) does not help the detection rate. Instead, it sometimes lowers the detection rate. The Figure 7 also illustrates that the size of the T detectors has little impact on the false alarm rate.

5. Discussion

5.1. The Sensitivity of Parameters. To make the algorithm reliable and usable on the application domain, the tradeoff between the number of the control parameters and the algorithm performance is worthy of being considered when we design the algorithm. Less control parameters make the algorithm simpler but less flexible. On the contrary, if more control parameters are provided, the algorithm becomes more flexible and thus makes it possible to maximize the algorithm performance by tuning various parameters specific for the application. Providing a good value for the

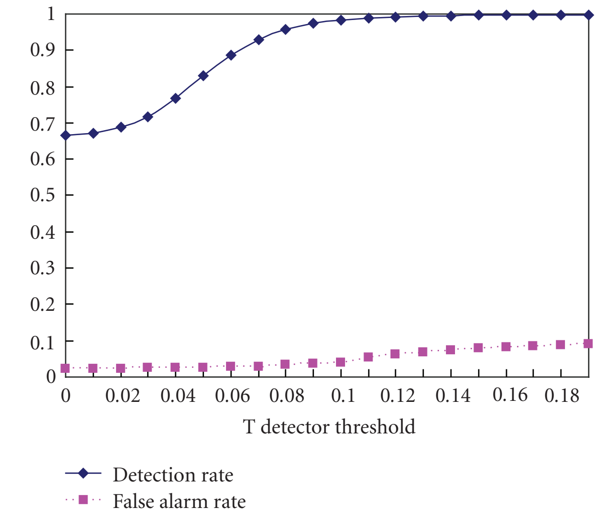


Figure 3: Influence of T detector threshold on both detection rate and false alarm rate with the training data of 50% normal samples.

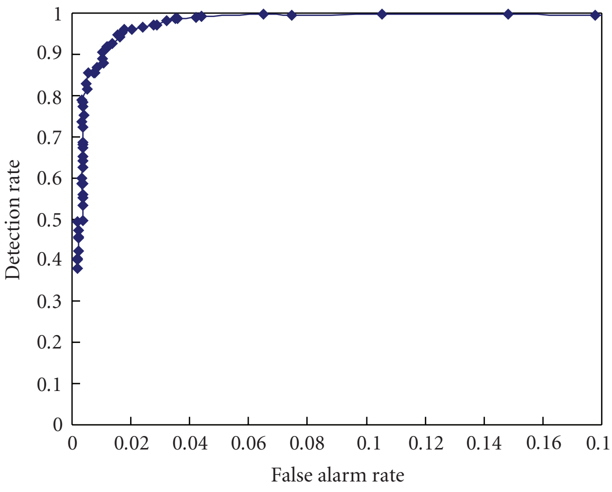


Figure 4: Performance (ROC) curve obtained by changing APC detector threshold with the training data of 50% normal samples.

certain parameter, however, may not be very intuitive. More control parameters make the algorithm operation more complicated, so it is hard to make the algorithm work well if we have poor intuition on the parameters. The results reported above show that the performance of the modified CSPRA is very sensitive to the threshold values of both T detector and APC detector, so both thresholds are key parameters to balance between sensitivity and generation and are required to be tuned with the application data to obtain better performance. The result in Figure 7 indicates that a good value for the parameter of T detector size could easily be found because the detection rate quickly reaches 100% with lower false alarm rate when the size of the T detector increases. We find in our experiments that the influence of the parameters of both ambiguous coefficient and pattern weight on the algorithm performance is minor; so the values for both ambiguous coefficient and pattern weight could be set to 0.5 by default when we start tuning the proposed algorithm for a specific application. When the other parameters have been optimized, the ambiguous coefficient and pattern weight can be fine tuned to enlarge the algorithm performance. Better understanding the sensitivity of the parameters on the algorithm performance will direct us to first tune the parameters with higher sensitivity while ignoring the parameters with lower sensitivity by setting these parameters with the default values. Therefore, it not only reduces the workloads of tuning multiple parameters but also makes full use of the flexibility of multiple parameters.

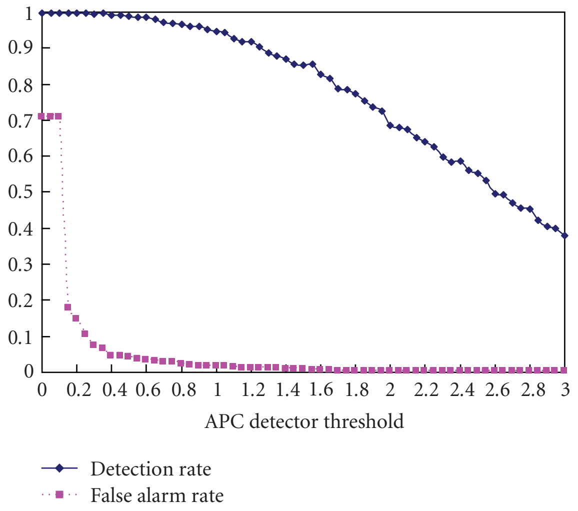


Figure 5: Influence of APC detector threshold on both detection rate and false alarm rate with the training data of 50% normal samples.

5.2. Comparison with Previous Works. The typical methods applied to breast cancer diagnosis in previous works include linear programming [4, 5], neural network [6, 7], and ant colony-based system [8]. In this section we compare our approach against these previous works. To make the comparison more appropriate, our experiments also calculate the accurate rate of the breast cancer prediction using (6). The corresponding results are reported in Table 7

$$\text{AccurateRate} = \frac{TP + TN}{TP + FP + TN + FN}. \tag{6}$$

The earliest work on breast cancer diagnosis [4] employed linear programming as the basic computational tool to determine two planes in an n-dimensional real space, as close together as possible, so that only the region between contains points from both sets of benign samples and malignant samples. Classification of the two sets is achieved by checking whether each point lies outside the region between the first pair of parallel planes. The validity of the method was tested with 369 samples. When 50% of the samples (185 samples) were used as a training set, 6.5% of the testing samples (the remaining 50% of the samples)

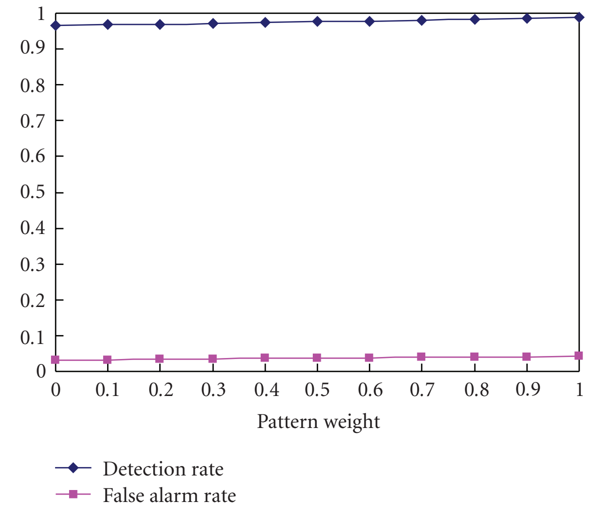


Figure 6: Influence of pattern weight on both detection rate and false alarm rate with the training data of 50% normal samples.

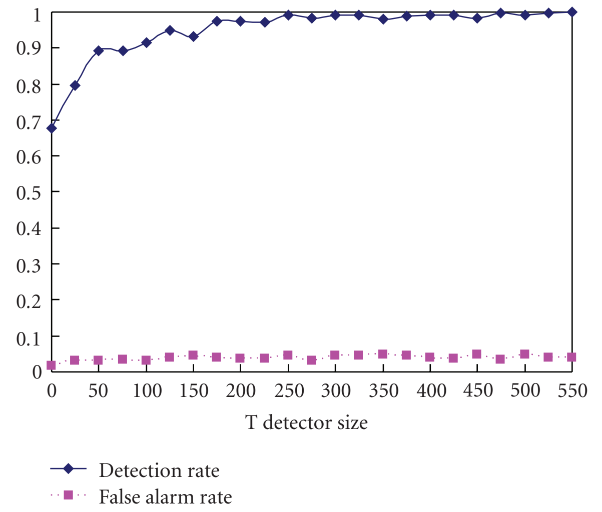


Figure 7: Influence of T detector size on both detection rate and false alarm rate with the training data of 50% normal samples.

were misclassified. This work was later extended in [5] by replacing 9 cytological characteristics for each sample with 30-dimensional feature vector that are generated by Xcyt system. The work in [5] used 569 vectors to train a linear programming-based diagnostic system, and the highest predicted accuracy, estimated with cross-validation, was 97.5% with three features (extreme area, extreme smoothness, and mean texture), a subset of the 30-dimensional feature vector. Compared to these previous works, we tested the larger dataset (683 samples) and achieved a better result (the highest accurate rate is 97.72% with training data of 50% of benign samples). Moreover, the requirement of ample malignant samples that are hardly collected in the clinic is mandatory in the training phase for these previous works [4, 5], whereas our approach only requires normal samples (the majority of the clinical cases) to train the system.

Table 7: Accurate rate with different training data.

Training data	Max (%)	Min (%)	Mean (%)	SD
100% benign samples	98.11	96.43	97.31	0.36
50% benign samples	97.72	95.21	96.55	0.49
25% benign samples	95.87	93.44	94.69	0.51

Abbass presented an evolutionary artificial neural network approach based on the pareto differential evolution algorithm augmented with local search for breast cancer diagnosis [6]. He used the same dataset as the one used in this paper. He chose the first 400 instances as the training set and the remaining 283 as the test set and the average test accuracy for his method is 98.1%. Although Abbass's method produces a higher accuracy rate than that obtained in our experiments, the larger training data used in his experiments could potentially overtrained the classifier. Because Abbass used the order of collections (*first* 400 instances) to define the training set, there could be ordering effects. Our experiment design, as described in Section 4.3, takes into account the problem with ordering effect, so our experiments are more reliable and reasonable.

In [7], the Wisconsin breast cancer database is used to train three different feedforward artificial neural network, including Cancer-Bin, Cancer-Norm, and Cancer-Cont networks. Cancer-Bin is trained with binary input patterns; Cancer-Norm is trained with normalized input patterns; Cancer-Cont is trained with the original data set. Three different rule extraction techniques (Binarized Rule Extraction, Partial Rule Extraction, and Full Rule Extraction), along with the rule ordering and integration mechanism are used to extract rules from these networks. Total 683 samples are evenly divided into a training set and a test set. The dimensionality of the breast-cancer input space is reduced from 9 to 6 inputs. When using the test set, the match rates for the trained networks of Cancer-Bin, Cancer-Norm, and Cancer-Cont are 96.20%, 95.91%, and 95.61%, respectively. By comparison with our method presented in this paper, the system used in [7] was apparently complicated. Based on the results in Table 7, our proposed method outperformed the hybrid symbolic-connectionist system in [7].

An ant colony-based system was also applied to breast cancer diagnosis [8]. The artificial Ant Colony System first specifies how ants construct or modify a solution for the problem domain, that is, the discovery of classification rules, in a probabilistic way and then update the pheromone trail considering the evaporation rate and the quality of the current solution. The predictive accuracy obtained with this method is 95.47%. In addition to the poor performance compared to our proposed method, the ant colony system was computationally expensive, especially when the search space (number of predicting attributes) is too large [8].

6. Conclusions

In this paper, a novel strategy on anomaly detection for the Conserved Self pattern Recognition Algorithm (CSPRA) is proposed. To explore the new detection strategy, we first put forward the concept of ambiguous boundary which determines a ring-shaped region where unsure antigen objects reside. The novel detection strategy employs the domain knowledge and randomized method to resolve the conflicts in anomaly detection between two types of detectors (T detectors and APC detectors) during the detection phase of the algorithm. In particular, when the conflicts emerge, there are two cases that can be observed in the detection phase: (1) the new sample falls not only into the ambiguous region of the T detector but also into the ambiguous region of APC detector; (2) the new sample falls into neither the ambiguous region of the T detector nor the ambiguous region of APC detector. The randomized method is selected to resolve the conflicts if one of the above cases is observed. Domain knowledge is used to arrive at the final anomaly decision for a new testing sample for the observed cases other than the cases suitable for randomized method.

The improved CSPRA is applied to breast cancer diagnosis, and the promising results reported in this paper show that the potential usage of the CSPRA is an efficient and reliable technique to diagnose the breast cancer. The total 683 clinic samples collected by Dr. Wolberg are experimented with our proposed method after max-min normalization. The conserved self pattern is discovered by computing Pearsonian r between the values for each column of the attribute and the class labels. The experimental results are evaluated by considering not only the capability of the system to detect the anomalies (malignant samples) but also the extent of the system to misclassify the normal samples (benign samples). When only 25% of the normal benign samples (111 samples) are used to train the proposed model, the result is still very promising with the detection rate of 99.62% and the false alarm rate of 6.82% in the best case. Importantly, the best detection can remain unchanged because the set of T detectors and the indices of the randomized method can be on-line recorded and reused for future detection.

The influence of various control parameters on the performance of predicting breast cancer is also studied. The results indicate that the performance of the modified CSPRA is very sensitive to the threshold parameters of both T detector and APC detector. By comparison with the results in the literature from the previous works on breast cancer diagnosis, our method outperforms the other methods in addition to the advantage of its one-class classification. However, there is still a long way to apply our proposed method to the actual diagnosis in the clinic. There may be two reasons for immaturity of the proposed approach. One, the available breast cancer data are unbalanced between the two cases. The number of benign samples is almost double than that of malignant samples. Because of this limitation, we can not exclude that the promising performance of our method is due to overfitting towards benign cases. Two, the available data set is too small and the variation of the attributes is too low. For some attributes in raw breast cancer data, the same values are observed in most samples. Therefore, we still question that these observations in the source data might contribute to the good results generated by our method and the previous works. These problems could be lessoned when we can obtain a much larger number of samples balanced over two cases. Although relatively limited, the encouraging results of the CSPRA on the available breast cancer data still give weight to further study this technique with more data sets and real world applications and compare this technique with other AIS methods and machine learning algorithms, which constitutes the direction of our future work.

References

[1] E. Marshall, “Search for a killer: focus shifts from fat to hormones.,” *Science*, vol. 259, no. 5095, pp. 618–621, 1993.

[2] S. W. Fletcher, W. Black, R. Harris, B. K. Rimer, and S. Shapiro, “Report of the international workshop on screening for breast cancer,” *Journal of the National Cancer Institute*, vol. 85, no. 20, pp. 1644–1656, 1993.

[3] R. W. M. Giard and J. Hermans, “The value of aspiration cytologic examination of the breast: a statistical review of the medical literature,” *Cancer*, vol. 69, no. 8, pp. 2104–2110, 1992.

[4] W. H. Wolberg and O. L. Mangasarian, “Multisurface method of pattern separation for medical diagnosis applied to breast cytology,” *Proceedings of the National Academy of Sciences of the United States of America*, vol. 87, no. 23, pp. 9193–9196, 1990.

[5] O. L. Mangasarian and W. H. Wolberg, “Cancer diagnosis via linear programming,” *SIAM News*, vol. 23, no. 5, pp. 1–18, 1990.

[6] H. A. Abbass, “An evolutionary artificial neural networks approach for breast cancer diagnosis,” *Artificial Intelligence in Medicine*, vol. 25, no. 3, pp. 265–281, 2002.

[7] I. Taha and J. Ghosh, “Characterization of the Wisconsin breast cancer database using a hybrid symbolic-connectionist system,” in *Proceedings of the Intelligent Engineering Systems through Artificial Neural Networks Conference (ANNIE ’96)*, St Louis, Mo, USA, November 1996.

[8] R. S. Parpinelli, H. S. Lopes, and A. A. Freitas, “An ant colony based system for data mining: applications to medical data,” in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO ’01)*, pp. 791–798, San Francisco, Calif, USA, 2001.

[9] D. Dasgupta, “Advances in artificial immune systems,” *IEEE Computational Intelligence Magazine*, vol. 1, no. 4, pp. 40–43, 2006.

[10] P. Matzinger, “The danger model: a renewed sense of self,” *Science*, vol. 296, no. 5566, pp. 301–305, 2002.

[11] C. A. Janeway Jr., “Approaching the asymptote? Evolution and revolution in immunology,” in *Cold Spring Harbor Symposia on Quantitative Biology*, vol. 54, pp. 1–13, 1989.

[12] S. Yu and D. Dasgupta, “Conserved self pattern recognition algorithm,” in *Proceedings of the 7th International Conference in Artificial Immune System (ICARIS ’08)*, vol. 5132, pp. 279–290, Phuket, Thailand, August 2008.

[13] S. Forrest, A. S. Perelson, L. Allen, and R. Cherukuri, “Self-nonself discrimination in a computer,” in *Proceedings of the IEEE Computer Society Symposium on Research in Security and Privacy*, pp. 202–212, IEEE Computer Society Press, Los Alamitos, Calif, USA, 1994.

[14] UCI Machine Learning Repository, Center for Machine Learning and Intelligent Systems, University of California, Irvine, Calif, USA, http://archive.ics.uci.edu/ml.

[15] D. Moore, *Basic Practice of Statistics*, W. H. Freeman, San Francisco, Calif, USA, 2006.

[16] Z. Ji, D. Dasgupta, Z. Yang, and H. Teng, "Analysis of dental images using artificial immune systems," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '06)*, pp. 528–535, Vancouver, Canada, July 2006.

Hindawi Publishing Corporation
Journal of Artificial Evolution and Applications
Volume 2009, Article ID 179680, 9 pages
doi:10.1155/2009/179680

Research Article

Evolutionary Selection of Features for Neural Sleep/Wake Discrimination

Peter Dürr, Walter Karlen, Jérémie Guignard, Claudio Mattiussi, and Dario Floreano

Laboratory of Intelligent Systems, Ecole Polytechnique Fédérale de Lausanne, 1015 Lausanne, Switzerland

Correspondence should be addressed to Peter Dürr, peter.duerr@epfl.ch

Received 15 November 2008; Accepted 19 February 2009

Recommended by Janet Clegg

In biomedical signal analysis, artificial neural networks are often used for pattern classification because of their capability for nonlinear class separation and the possibility to efficiently implement them on a microcontroller. Typically, the network topology is designed by hand, and a gradient-based search algorithm is used to find a set of suitable parameters for the given classification task. In many cases, however, the choice of the network architecture is a critical and difficult task. For example, hand-designed networks often require more computational resources than necessary because they rely on input features that provide no information or are redundant. In the case of mobile applications, where computational resources and energy are limited, this is especially detrimental. Neuroevolutionary methods which allow for the automatic synthesis of network topology and parameters offer a solution to these problems. In this paper, we use analog genetic encoding (AGE) for the evolutionary synthesis of a neural classifier for a mobile sleep/wake discrimination system. The comparison with a hand-designed classifier trained with back propagation shows that the evolved neural classifiers display similar performance to the hand-designed networks, but using a greatly reduced set of inputs, thus reducing computation time and improving the energy efficiency of the mobile system.

1. Introduction

The traditional way to craft an artificial neural network (ANN) for a classification task is to hand design a network topology and to find a set of network parameters using a gradient-based error-minimization algorithm such as back propagation [1]. However, in real-world applications, such as the classification of biomedical signals, the network topology can be difficult to design by hand. Additionally, in many cases, it is desirable to minimize the computational cost of the network, for example, by reducing the number of inputs used by the classifier. Evolutionary methods for the design of ANNs can provide an answer to both issues [2]. In this paper, we study the application of a neuroevolutionary method called analog genetic encoding (AGE) [3] to the problem of synthesis and optimization of neural networks for the processing of biological signals aimed at sleep and wake classification.

Continuous monitoring of the sleep/wake state of high-risk professionals such as pilots, truck drivers, or shift workers can potentially decrease the risk of accidents and help scheduling breaks and resting times. However, implementing such a classification in a wearable device is a challenging task. Limited energy and processing resources as well as the increased noise level due to movement artifacts and a constantly changing environment put tight restrictions on the choice of sensors and algorithms. Traditionally, the states of sleep and wake are classified based on the analysis of brain wave patterns (EEG) [4]. EEG recording requires gluing electrodes to the scalp and is typically susceptible to different sources of noise. Methods relying on EEG measurements are thus more suited for sleep analysis in controlled hospital environments than for mobile applications.

For mobile sleep/wake pattern screening, a commonly used technique is actigraphy [5]. In actigraphy, the acceleration of the wrist of the subject is recorded, and phases of weak activity—as judged by the levels of acceleration—are classified as sleep. Actigraphy devices can be small, inexpensive, and low power, which makes them suitable for mobile applications. However, as the signals provided by actigraphy devices are not directly linked to physiological states, it is difficult to derive a reliable prediction from them. Activities characterized by low levels of motion, such as reading or watching TV, are often misclassified as sleep [6]. In [7, 8], we have suggested to use electrocardiogram (ECG) and respiratory effort (RSP) signals for wearable sleep/wake classification (see Figure 1). Both signals depend on properties of the activity of the autonomous nervous system, which differ in sleep and wake [9]. Furthermore, they are measurable with portable sensor systems such as the Heally system (see Figure 2, Koralewski Industrie Elektronik, Celle, Germany).

An additional difficulty is that the generation of a set of labeled data for the training of the classifier is typically a time-consuming activity for both the subjects from whom data is collected and the technicians who must label the data [7]. It is thus desirable to design a classifier that can be trained on a set of data and can then be used on further subjects without additional training. In [7], we have shown that using the frequency content of the ECG and RSP signals as input features for a single layer ANN, a mean accuracy of 86.7% can be achieved when the network was trained and tested on data obtained from different subjects. A limitation of the hand-designed ANN used in [7] is its large number of inputs. Some of these inputs are presumably redundant and might not contribute significantly to the classifier performance. For the targeted mobile application, the power consumption of the classifier is critical. In order to reduce processing time and thus power consumption for mobile applications, it would be desirable to minimize the number of inputs. In this paper, we show how to automatically synthesize networks that use a small subset of the spectral components associated with the signals as inputs while maintaining the performance of the classifier.

2. Evolutionary Synthesis of Neural Networks

Neural networks can be described as directed graphs, where the nodes represent a neuron model, and the edges of the graph are associated with the weighted connections between the neurons, the so-called synaptic weights. The design of a network for a particular task thus involves the choice of the topology of the graph (i.e., the network architecture) and a suitable set of numerical parameters (i.e., the synaptic weights and the parameters of the neuron model). The automatic synthesis of the topology and parameters of a neural network requires a computer representation for both aspects of the network, combined with an algorithm capable of performing a search in the space defined by this representation. Evolutionary algorithms have been extensively used to evolve neural classifiers because these algorithms can combine a flexible representation with a high potential of stochastic exploration of the search space [10–13].

The simplest approach to this, the so-called *direct encoding*, represents all the neurons, synaptic connections, and parameters of the network explicitly (see, e.g. [14–16]). This has the advantage that the resulting networks can easily be decoded from the genome. However, with increasing size of the network, the length of the corresponding genome grows rapidly, which can affect the evolvability. In order to mitigate this problem, it has been suggested to encode a program or a sequence of instructions that, when executed, builds the network. This *developmental encoding* can lead to very compact representations of large networks (see, e.g., [17, 18]). However, the definition of a set of mutation and recombination operators which guarantees that only valid networks are generated during the search is typically very difficult.

A promising alternative to direct and developmental representations that is getting more and more popular is *implicit encoding* [19–23]. In this paper, we use an implicit representation called analog genetic encoding (AGE). AGE has been shown to be very effective for the automatic synthesis of various kinds of networks and, in particular, of neural networks [2, 3, 24–26].

The concept of implicit encodings like AGE is loosely inspired by the working of biological gene regulatory networks (GRNs). In biological GRNs, the interactions between the genes are not explicitly encoded in the genome but follow implicitly from the physical and chemical environment in which the genome is immersed. Simplifying a bit the picture, the activation of a biological gene depends on the interaction of molecules produced by another gene with parts of the activated gene called regulatory regions (Figure 3(a)). AGE abstracts this picture and defines an artificial genome composed of sequences of characters, for example, the uppercase ASCII set (Figure 3(b)). Similar to the function of promoter and terminal regions in GRNs, special sequences (the so-called tokens) identify regions of the artificial genome as artificial genes, which encode individual neurons. The sequences delimited by the tokens are interpreted analogous to coding regions and regulatory regions in biological GRNs. The strength of the connection between two neurons is implicitly determined by the coding region of one neuron and the regulatory region of another neuron via a function called *interaction map*. The interaction map can be seen as an abstraction of the biochemical process of gene regulation. It takes sequences of characters as arguments and outputs a real-valued connection strength. In our implementation, this is obtained by mapping the local alignment score [27] of the two sequences exponentially to the interval that spans all possible weight values (see [24]).

In summary, the AGE genome can be decoded first by extracting the neurons with the associated (coding and regulatory) sequences of characters. This is realized by scanning the genome for tokens which indicate the presence of a neuron (GN). Together with predefined terminator sequences (TE), these tokens delimit the part of the genome associated with the respective neuron. The enclosed sequences of characters are interpreted as the coding and regulatory sequences of the respective neuron. Subsequently, the interaction map I can be applied to all pairs of coding and regulatory sequences to obtain the synaptic weights w_{ij} connecting the neurons (see Figure 4).

In this framework, there are several different possibilities to implement connections from external inputs to external outputs (see [28] for more details). Here, we encoded the coding sequences associated to the input neurons and the regulatory sequences associated to the output neurons in

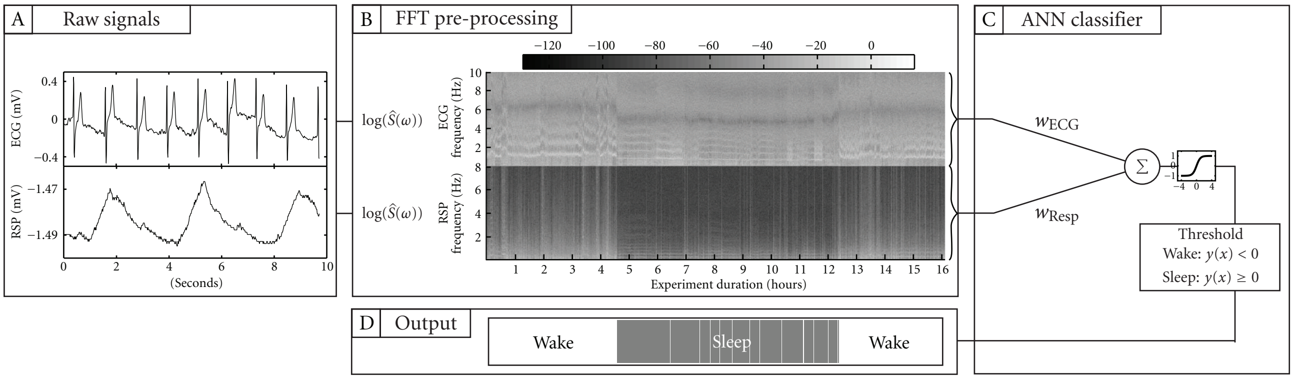


Figure 1: Overview of the sleep/wake classification system. (a) The raw electrocardiogram (ECG) and respiratory effort (RSP) signals are cut into windows of 40.06 seconds. (b) A short-time fast Fourier transformation (FFT) is used to calculate the spectral power of the windowed signals. (c) The resulting frequency data are fed to a feed-forward artificial neural network (ANN) and (d) a symmetric threshold classifies the ANN output into sleep or wake state estimates.

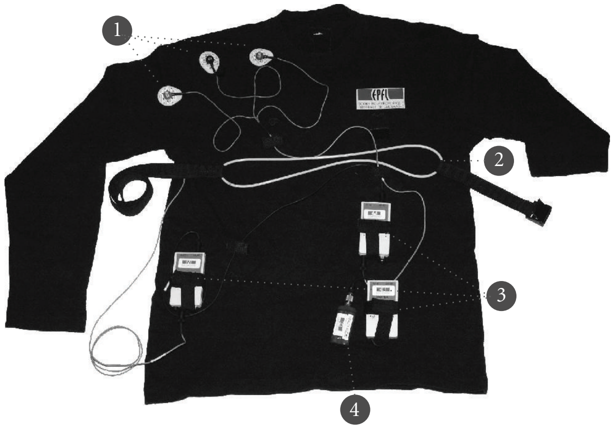


Figure 2: Portable Heally recording system mounted on a shirt. (1) ECG gel electrodes; (2) inductive belt sensor; (3) electronics modules; (4) NiMH battery.

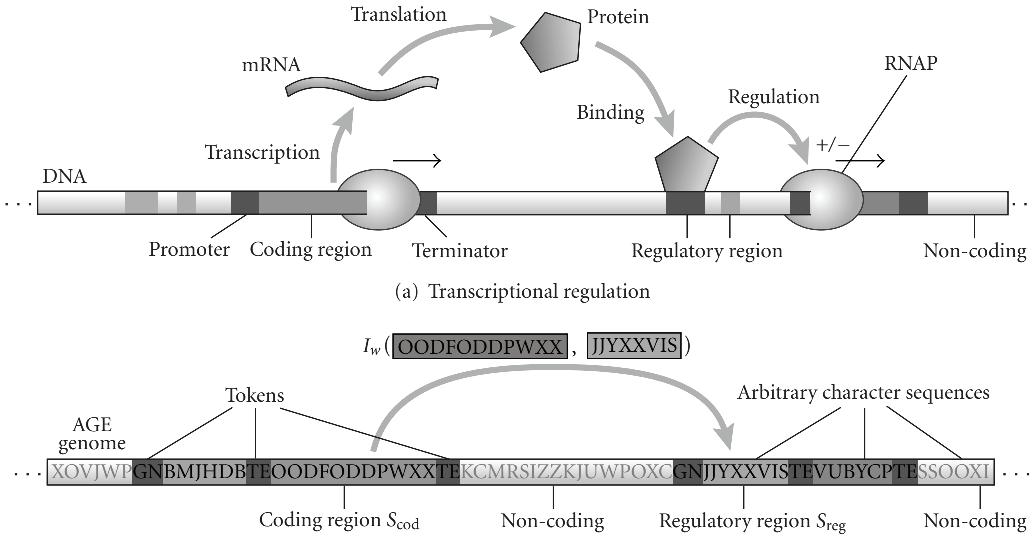
separated parts of the genome (see Figure 5). In this case, the connections from the input neurons to the network can be obtained by applying the interaction map to all pairs of coding sequences (associated with the input neurons and the hidden neurons) and regulatory regions (associated to the hidden neurons and the output neuron). Note that the interaction map can associate a null weight value, thus leaving the respective neurons unconnected. When this feature is applied to the connections stemming from the input neurons, it gives evolution the freedom to select a subset of the set of inputs that contains the information necessary to realize the classification task.

As the sequences which define the strength of the synaptic connections can have a variable length and the interaction map is defined to operate on sequences of arbitrary length, a large class of genetic operators can be used to alter the network. In particular, we use the biologically plausible insertion, substitution, and deletion of characters and the transposition, duplication, and deletion of fragments of genome. The changes in the genome caused by these mutation operators can reflect both changes in the parameters of the network as well as changes in the network structure. For example, the insertion of a character in the genome can lead to a change of the synaptic weight connecting a particular input to the output neuron. The deletion of a fragment of genome associated with an input of the network can lead to the removal of this particular input from the network. Furthermore, the number of hidden neurons in the network can increase (e.g., after a genome fragment duplication) or decrease (e.g., after a character substitution) over the course of evolution. Given the fact that parts of the genome can be noncoding (i.e., they are not part of the description of a neuron) and that the interaction map is defined to be highly redundant, many mutations do not have an effect on the decoded networks. This allows for a high neutrality in the search space, which can improve evolvability [29].

3. Experiments

To compare the performance of the classical approach to classifier synthesis and training with the state-of-the-art neuroevolution method based on AGE, we performed a set of experiments, where we compared the performance of a neural network with fixed hand-designed topology and variable weights trained with back propagation, with that of neural networks synthesized with an evolutionary algorithm-based on AGE. As anticipated, we are interested in the performance in a sleep/wake detection task, where data from a set of users is available for network synthesis and training, but the performance is expected to generalize to additional users. We thus investigated the performance of the two methods when trained on ECG and RSP data collected on multiple subjects, and tested on data from a different subject.

3.1. Data. The data used in the following experiments are identical with those described in [8], where a hand-designed classifier with back propagation was used. They stem from



(a) Transcriptional regulation

(b) The AGE abstraction

FIGURE 3: (a) In biological gene networks, the link between genes is realized by molecules that are synthesized from the coding region of one gene and interact with the regulatory region of another gene. (b) Analog genetic encoding abstracts this mechanism using an artificial genome containing markers that identify the artificial genes, and an interaction map that creates links between pairs of artificial genes by associating with them a numerical value that represents the strength of the link.

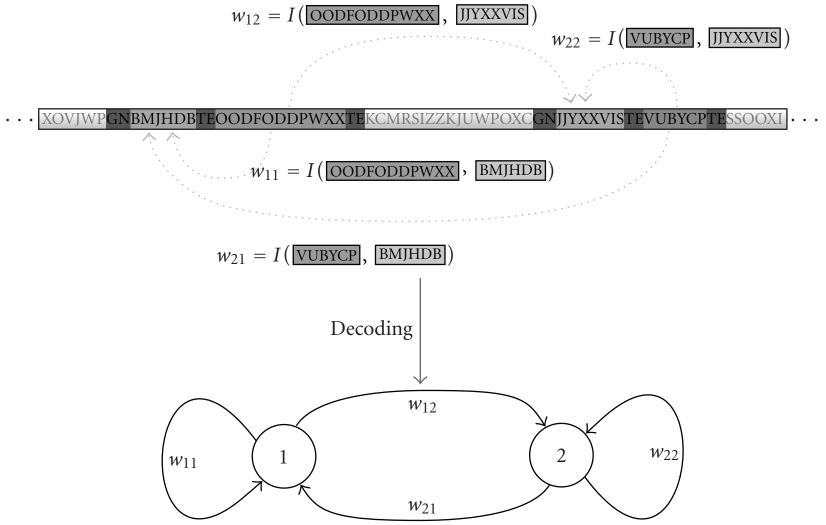

$w_{12} = I($ OODFODDPWXX , JJYXXVIS $)$
$w_{22} = I($ VUBYCP , JJYXXVIS $)$
XOVJWPGNBMJHDBTEOODFODDPWXXTEKCMRSIZZKJUWPOXCGNJJYXXVISTEVUBYCPTESSOOXI
$w_{11} = I($ OODFODDPWXX , BMJHDB $)$
$w_{21} = I($ VUBYCP , BMJHDB $)$
Decoding
w_{12}
w_{11}
1
2
w_{22}
w_{21}

FIGURE 4: A simple artificial neural network represented with analog genetic encoding. The interaction strengths w_{ij} are computed by the interaction map $I(s_i, s_j)$ which takes the sequence of characters s_i associated to the output of neuron i and the sequence of characters s_j associated to the input of neuron j as inputs.

recording sessions with six young healthy male subjects of a mean(± SD) age of 26(± 3) years. The subjects wore a Heally recording device (see Figure 2) for a total of 18 recording sessions which lasted 16 hours each and contained an overnight sleep. The datasets are composed of ECG and RSP recordings sampled at 100 Hz and 50 Hz, respectively. The *a priori* sleep and wake states of the subjects were determined by a trained technician who labeled the signals in 10-second intervals based on electromyogram, electrooculogram, and video recordings. The data were preprocessed and fed to the ANN. As in [7], the preprocessing step consisted of calculating the power spectrum of each signal using a short-time fast Fourier transform with a window length of 40.96 seconds (see Figure 1(b)). For each of these segments, we calculated a feature vector as $\vec{v} = \log(\hat{S}(\omega))$, where $\hat{S}(\omega)$ is the periodogram of the segment. Experiments described in [8] revealed that frequency components above 10 Hz for ECG and 8 Hz for RSP do not contribute to the hand-designed classifier performance and can be removed. The resulting two input vectors are thus composed of 409 spectral inputs

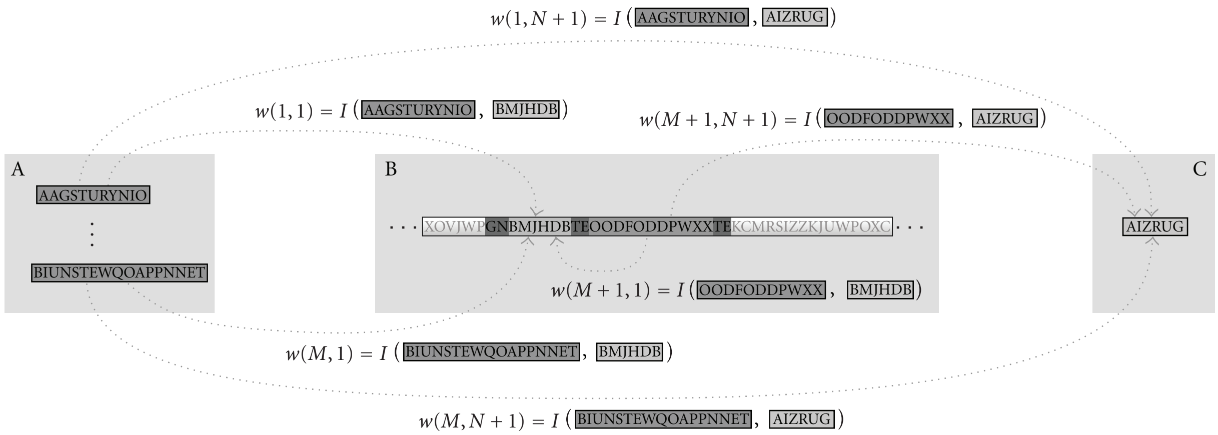

$w(1, N+1) = I$ (AAGSTURYNIO , AIZRUG)
$w(1, 1) = I$ (AAGSTURYNIO , BMJHDB)
$w(M+1, N+1) = I$ (OODFODDPWXX , AIZRUG)
A
AAGSTURYNIO
⋮
BIUNSTEWQOAPPNNET
B
· · · XOVJWPGNBMJHDBTEOODFODDPWXXTEKCMRSIZZKJUWPOXC · · ·
C
AIZRUG
$w(M+1, 1) = I$ (OODFODDPWXX , BMJHDB)
$w(M, 1) = I$ (BIUNSTEWQOAPPNNET , BMJHDB)
$w(M, N+1) = I$ (BIUNSTEWQOAPPNNET , AIZRUG)

Figure 5: There are different ways to implement external inputs and outputs in AGE [28]. Here, the genome is split in three parts: (a) contains the coding sequences of the M input neurons, (b) contains the definition of the N hidden neurons, and (c) contains the regulatory sequence of the output neuron. In the decoding process, the coding sequences and the regulatory sequences of all neurons present in the genome are identified. The connection weights $w(x, y)$ can then be obtained by applying the interaction map to all pairs of coding sequences x and regulatory sequences y.

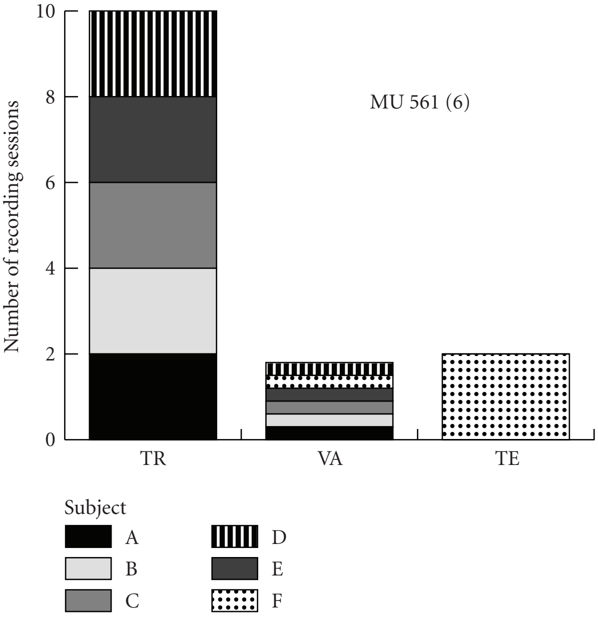


Figure 6: Distribution of the experimental data used for the training, validation, and test of the hand-designed and the evolutionary synthesized neural classifiers. The numbers indicate users in the training set (TR), users in the validation set (VA), and users in the test set (TE). There are six repetitions with different combinations of users/sessions in training, validation, and test sets.

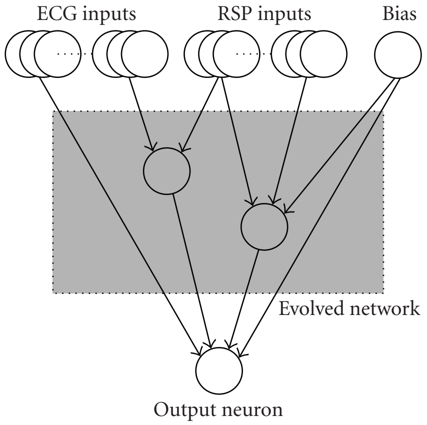


Figure 7: The neural classifier is automatically synthesized with analog genetic encoding. The evolved network can connect to an arbitrary subset of the 409 inputs from the ECG data, the 327 inputs from the RSP data and a bias unit. As the size of the network is not fixed, the number of hidden units in the network can increase or decrease over the course of evolution. The output unit indicates sleep or wake states using a simple threshold at an activation level of zero.

from ECG and 327 spectral inputs from RSP. Together, they compose the set of 736 inputs that were fed to the ANN classifier (see Figure 1(c)).

3.2. Experimental Design. In order to evaluate the performance of the two classifiers, we divided the data into three different sets: a training set (TR), a validation set (VA), and a test set (TE) (see Figure 6). The training set contains a subset of the data from five of the six subjects. The validation set is composed of 2 hours of data from each subject, randomly sampled over the two available sessions and containing an equal amount of samples labeled as sleep and wake. This data is not used for training or for testing. The test set contains data from the subject that has not been used in the training. Five independent runs of each experiment are performed from different randomly assigned initial conditions. In order to prevent performance biases due to the choice of sessions, we repeat each experiment with all possible combinations of users in the test and trainingsets, making sure that the same sessions do not appear both in the training and in the

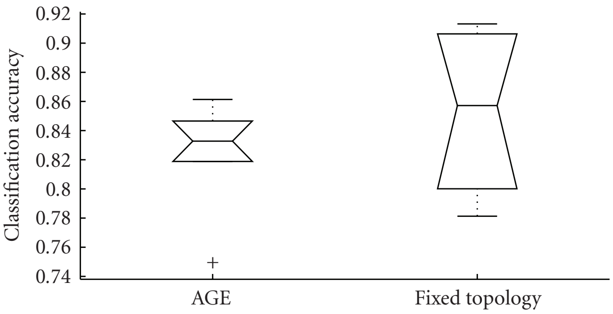


Figure 8: The average classification accuracy of the evolved networks (AGE) and the fixed topology networks trained with back propagation (*fixed topology*). The midline in each box is the median, the borders of the box represent the upper and the lower quartiles. The whiskers outside the box represent the minimum and maximum values obtained, except when there are outliers which are shown as small crosses. We define outliers as data points which differ more than 1.5 times the interquartile range from the border of the box. The notches permit the assessment of the significance of the differences of the medians. When the notches of two boxes overlap, the corresponding medians are not significantly different at (approximately) the 95% confidence level [34].

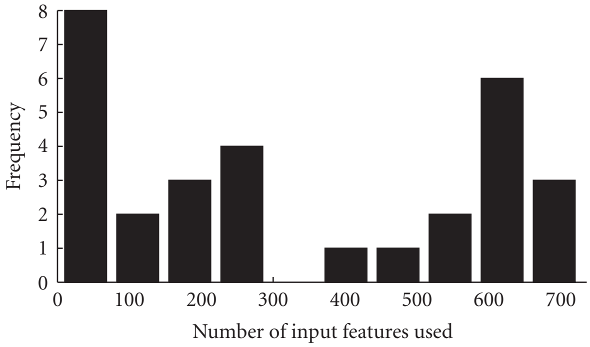


Figure 9: Histogram of the number of input features used by the evolved networks in the five repetitions of each of the six training cases. From the 30 networks, 8 used from 3 to 45 inputs, 2 used from 95 to 113 inputs, 3 used from 162 to 196 inputs, 4 used from 231 to 282 inputs, 1 used 411 and 1 used 484 inputs, 2 networks used from 522 to 533 inputs, 6 networks used from 602 to 647 inputs, and 3 networks used from 628 to 732 inputs.

testsets. This leads to a total of six different cases with five independent replications for each case.

3.3. Algorithms

3.3.1. Hand-Designed Fixed Topology Network. As a baseline for the classification accuracy, we used a feed-forward ANN with no hidden layers and a single output unit with a tangent-sigmoid transfer function. Additional experiments not reported here showed that the use of ANNs with a hidden layer does not improve the performance of the classifier. A similar finding has been reported by [30]. The synaptic weights of this fixed topology network were initialized with the Nguyen-Widrow method [31] and trained with a Levenberg-Marquardt back-propagation algorithm [32].

3.3.2. Network Synthesized with AGE. For the automatic synthesis of the network topology and parameters, the AGE representation was combined with a standard genetic algorithm (see [24] for more details). Using the above-mentioned possibility of feature selection, the evolved network could connect to an arbitrary subset of the 409 inputs from the ECG data, the 327 inputs from the RSP data, and a constant bias unit (see Figure 7). Additionally, the evolutionary process might insert hidden neurons in the network in order to generate more complex network structures. The activation y_i of the hidden neuron i was computed as

$$y_i = \sigma_i\left(\sum_{k=1}^{N} w(i,k)y_k + \sum_{l=1}^{M} w(i,N+l)I_l + w(i,N+M+1)\right), \tag{1}$$

where N is the number of hidden neurons in the network, $w(x,y) = w_{xy}$ are the entries of the weight matrix, $M = 736$ is the number of available inputs, I_l is the value of input l, and

$$\sigma_i(z) = \frac{2}{1+e^{-\alpha_i z}} - 1, \tag{2}$$

is a sigmoid transfer function with slope parameter α_i. The activation of the output neuron was computed analogously to the activation of the hidden neurons. The slope parameters α_i for the hidden neurons were encoded using the center of mass encoding [33].

Selection was performed using tournament selection and elitism. The algorithm parameters and mutation probabilities are listed in Table 1. In order to prevent bootstrap problems, the population was initialized with the best 100 networks out of 1000 randomly created genomes. Additionally, to save computation time, only a randomly selected subset of 10% of each training set was used for training. However, validation and testingwere always performed using 100% of the respective dataset. For each evolutionary run, the synthesized network was the network with the best performance on the validation set, in the collection of all the best performing networks observed at each of the 1000 generations that compose a run.

For both the back-propagation training and the evolutionary process, the measure of quality of the classifier was the sum over the data points of the squares of the difference between the actual and the desired classifier output.

4. Results and Discussion

As shown in Figure 8, the evolved networks and the fixed topology networks trained with back propagation do not display a significantly different classification accuracy (Wilcoxon rank sum test $P = .48$). However, while the hand-designed fixed topology networksemploy all of the 736 input features, many of the evolved networks used a

Table 1: The parameters used in the evolutionary algorithm.

Parameter	Value
Population size	100
Tournament size	2
Elite size	1
Recombination probability	.1
Probability of character substitution (per character)	.001
Probability of character insertion (per character)	.001
Probability of character deletion (per character)	.0015
Probability of fragment transposition	.01
Probability of fragment duplication	.01
Probability of fragment deletion	.015
Probability of neuron insertion	.01

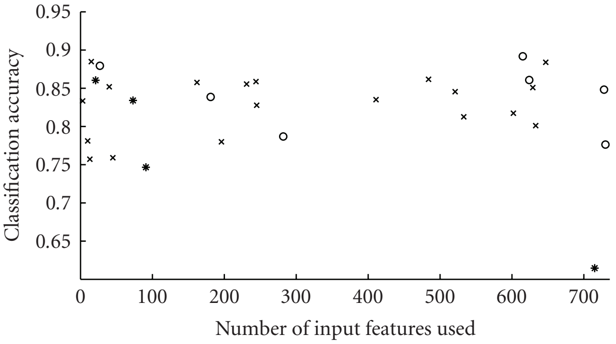


Figure 10: The performance of the evolved networks in the five repetitions of each of the six training cases. The horizontal axis represents the number of input features used by the network and the vertical axis gives the corresponding classification performance. The symbols indicate the number neurons in the hidden layer of the network. A cross indicates 0 hidden neurons, a circle indicates 1 hidden neuron, a star indicates 2 hidden neurons. Both the number of inputs and the number of hidden neurons are not significantly correlated with classification accuracy (see text).

drastically reduced set of inputs (see Figure 9, the median of the number of inputs used is 244.5). Figure 10 shows that there is no correlation between the number of inputs used by the evolved networks and their performance (Spearman's rank correlation coefficient $P = .02$, $P = .94$). This indicates that many input features are indeed redundant and that it is possible to synthesize networks with a very small number of inputs which perform as well as the hand-designed network using all inputs. However, all networks use input features from both ECG and RSP data (see Figure 11). Given the results of [7], it is not surprising that the presence of both types of data is beneficial for the classification accuracy and thus selected during evolution. Note that in the evolutionary experiments, no additional penalty term was added to the objective function to bias the search toward small networks. This explains the presence of both networks using a significantly reduced set of inputs, and networks using almost the whole set of available inputs in the evolutionary results.

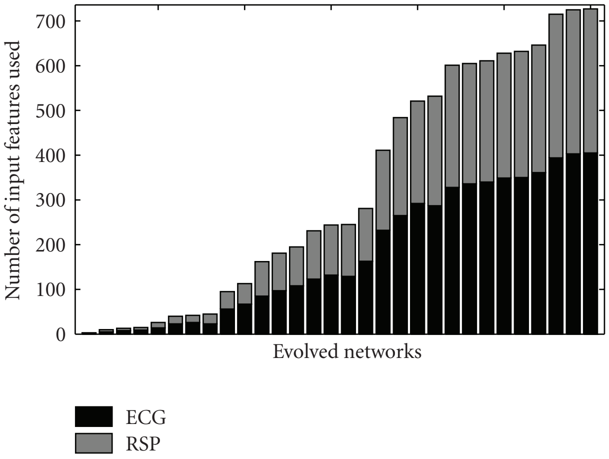


Figure 11: The evolved networks for the five repetitions of each of the six cases, sorted by the number of used input features. All networks use input features from both ECG and RSP data.

As mentioned above, the fixed topology network has no hidden layer. Of the 30 evolved networks, 19 feature no hidden neurons, 7 feature one hidden neuron, and 4 feature two hidden neurons. However, there is no correlation between the number of hidden neurons and the classification accuracy (Spearman's rank correlation coefficient $P = -.06$, $P = .74$). This substantiates the conjecture formulated in [7] that a hidden layer is not necessary for optimal performance in this task. Note, however, that this conjecture applies to this specific problem and does not extend to general classification applications.

5. Conclusion

Portable devices for biomedical signal analysis, like sleep/wake classification, have the potential to alleviate health problems and prevent accidents. Recent advances in sensor development and miniaturization allow for the construction of small mobile devices which integrate biomedical sensors and a microprocessor with sufficient processing power for many applications. However, one of the critical challenges, that remains, is the design of efficient classifiers which can be implemented on these small mobile systems. While the classification accuracy has to be as high as possible, the computational effort and thus the energy requirements for classification have to remain low. The results presented in this paper demonstrate that analog genetic encoding (AGE) permits the automatic evolutionary synthesis of compact neural classifiers for the problem of sleep/wake classification. Compared to a hand-designed classifier trained with back propagation, the possibility of the evolutionary selection of a subset of the available inputs permits a drastic reduction of the number of inputs without significant degradation of the classifier performance. For example, in the experiments presented here, the evolutionary synthesis with AGE found a classifier with the accuracy of 88.49%, using only 15 of the 736 input features used by the hand-designed network. The implementation of this evolved solution on a digital signal controller of the dsPIC33 product family (Microchip Technology Inc., USA) requires only 5.13% of the instructions used by an implementation of the hand-designed network on the same processor. This is a reduction of the computational cost of almost 95%. Moreover, the savings in computational cost and energy can be increased even further by adapting the sensory modalities and preprocessing steps to the reduced set of input features.

Acknowledgments

This work was supported by the Swiss National Science Foundation, Grant no. 200021-112060 and the Solar Impulse Project grant of Ecole Polytechnique Fédérale de Lausanne (EPFL). Thanks to Daniel Marbach and Sara Mitri for their comments on an earlier version of this manuscript and the anonymous reviewers for their helpful suggestions.

References

[1] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagation of errors," *Nature*, vol. 323, pp. 533–536, 1986.

[2] D. Floreano, P. Dürr, and C. Mattiussi, "Neuroevolution: from architectures to learning," *Evolutionary Intelligence*, vol. 1, no. 1, pp. 47–62, 2008.

[3] C. Mattiussi and D. Floreano, "Analog genetic encoding for the evolution of circuits and networks," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 5, pp. 596–607, 2007.

[4] A. Rechtschaffen, A. Kales, R. Berger, and W. Dement, *A Manual of Standardized Terminology, Techniques and Scoring System for Sleep Stages of Human Subjects*, Public Health Service, U.S. Government Printing Office, Washington, DC, USA, 1968.

[5] A. Sadeh and C. Acebo, "The role of actigraphy in sleep medicine," *Sleep Medicine Reviews*, vol. 6, no. 2, pp. 113–124, 2002.

[6] C. P. Pollak, W. W. Tryon, H. Nagaraja, and R. Dzwonczyk, "How accurately does wrist actigraphy identify the states of sleep and wakefulness?" *Sleep*, vol. 24, no. 8, pp. 957–965, 2001.

[7] W. Karlen, C. Mattiussi, and D. Floreano, "Adaptive sleep/wake classification based on cardiorespiratory signals for wearable devices," in *Proceedings of the IEEE on Biomedical Circuits and Systems Conference (BIOCAS '07)*, pp. 203–206, Montreal, Canada, November 2007.

[8] W. Karlen, C. Mattiussi, and D. Floreano, "Sleep and wake classification with ECG and respiratory effort signals," to appear in *IEEE Transactions on Biomedical Circuits and Systems*.

[9] R. D. Ogilvie, "The process of falling asleep," *Sleep Medicine Reviews*, vol. 5, no. 3, pp. 247–270, 2001.

[10] G. P. Zhang, "Neural networks for classification: a survey," *IEEE Transactions on Systems, Man and Cybernetics, Part C*, vol. 30, no. 4, pp. 451–462, 2000.

[11] M. Rocha, P. Cortez, and J. Neves, "Evolution of neural networks for classification and regression," *Neurocomputing*, vol. 70, no. 16–18, pp. 2809–2816, 2007.

[12] M. Čepek, M. Šnorek, and V. Chudáček, "ECG signal classification using GAME neural network and its comparison to other classifiers," in *Proceedings of the 18th International Conference on Artificial Neural Networks (ICANN '08)*, vol. 5163 of *Lecture Notes in Computer Science*, pp. 768–777, Prague, Czech Republic, September 2008.

[13] L. Chen and D. Alahakoon, "NeuroEvolution of augmenting topologies with learning for data classification," in *Proceedings of the International Conference on Information and Automation (ICIA '06)*, pp. 367–371, Shandong, China, December 2006.

[14] X. Yao, "Evolving artificial neural networks," *Proceedings of the IEEE*, vol. 87, no. 9, pp. 1423–1447, 1999.

[15] K. O. Stanley and R. Miikkulainen, "Evolving neural networks through augmenting topologies," *Evolutionary Computation*, vol. 10, no. 2, pp. 99–127, 2002.

[16] R. S. Zebulum, M. Vellasco, and M. A. Pacheco, "Variable length representation in evolutionary electronics," *Evolutionary Computation*, vol. 8, no. 1, pp. 93–120, 2000.

[17] F. Gruau, "Automatic definition of modular neural networks," *Adaptive Behavior*, vol. 3, no. 2, pp. 151–183, 1994.

[18] J. R. Koza, *Genetic Programming II: Automatic Discovery of Reusable Programs*, MIT Press, Cambridge, Mass, USA, 1994.

[19] J. Bongard, "Evolving modular genetic regulatory networks," in *Proceedings of the Congress on Evolutionary Computation (CEC '02)*, vol. 2, pp. 1872–1877, Honolulu, Hawaii, USA, May 2002.

[20] T. Reil, "Dynamics of gene expression in an artificial genome-implications for biological and artificial ontogeny," in *Proceedings of the 5th European Conference on Artificial Life (ECAL '99)*, pp. 457–466, Lausanne, Switzerland, September 1999.

[21] T. Reil, "Artificial genomes as models of gene regulation," in *On Growth, Form and Computers*, pp. 256–277, Academic Press, London, UK, 2003.

[22] C. Mattiussi, D. Marbach, P. Dürr, and D. Floreano, "The age of analog networks," *AI Magazine*, vol. 29, no. 3, pp. 63–76, 2008.

[23] J. Reisinger and R. Miikkulainen, "Acquiring evolvability through adaptive representations," in *Proceedings of the 9th Annual Genetic and Evolutionary Computation Conference (GECCO '07)*, pp. 1045–1052, ACM Press, London, UK, July 2007.

[24] P. Dürr, C. Mattiussi, and D. Floreano, "Neuroevolution with analog genetic encoding," in *Proceedings of the 9th International Conference on Parallel Problem Solving from Nature (PPSN '06)*, vol. 9, pp. 671–680, Springer, Reykjavik, Iceland, September 2006.

[25] A. Soltoggio, P. Dürr, C. Mattiussi, and D. Floreano, "Evolving neuromodulatory topologies for reinforcement learning-like problems," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '07)*, P. Angeline, M. Michaelewicz, G. Schonauer, X. Yao, and Z. Zalzala, Eds., pp. 2471–2478, IEEE Press, Singapore, September 2007.

[26] P. Dürr, C. Mattiussi, A. Soltoggio, and D. Floreano, "Evolvability of neuromodulated learning for robots," in *Proceedings of the ECSIS Symposium on Learning and Adaptive Behaviors for Robotic Systems (LAB-RS '08)*, pp. 41–46, Edinburgh, Scotland, August 2008.

[27] G. Gusfield, *Algorithms on Strings, Trees, and Sequences*, Cambridge University Press, Cambridge, UK, 1997.

[28] C. Mattiussi, *Evolutionary synthesis of analog networks*, Ph.D. dissertation, EPFL, Lausanne, Switzerland, 2005.

[29] A. Wagner, "Robustness, evolvability, and neutrality," *FEBS Letters*, vol. 579, no. 8, pp. 1772–1778, 2005.

[30] J. Principe and A. Tome, "Performance and training strategies in feedforward neural networks: an application to sleep scoring," in *Proceedings of the International Joint Conference on Neural Networks (IJCNN '89)*, vol. 1, pp. 341–346, Washington, DC, USA, June 1989.

[31] D. Nguyen and B. Widrow, "Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights," in *Proceedings of International Joint Conference on Neural Networks (IJCNN '90)*, pp. 21–26, San Diego, Calif, USA, June 1990.

[32] M. T. Hagan and M. B. Menhaj, "Training feedforward networks with the Marquardt algorithm," *IEEE Transactions on Neural Networks*, vol. 5, no. 6, pp. 989–993, 1994.

[33] C. Mattiussi, P. Dürr, and D. Floreano, "Center of mass encoding: a self-adaptive representation with adjustable redundancy for real-valued parameters," in *Proceedings of the 9th Annual Genetic and Evolutionary Computation Conference (GECCO '07)*, pp. 1304–1311, London, UK, July 2007.

[34] R. McGill, J. W. Tukey, and W. A. Larsen, "Variations of box plots," *The American Statistician*, vol. 32, no. 1, pp. 12–16, 1978.

Hindawi Publishing Corporation
Journal of Artificial Evolution and Applications
Volume 2009, Article ID 783647, 5 pages
doi:10.1155/2009/783647

Research Article

Kuri: A Simulator of Ecological Genetics for Tree Populations

Benn R. Alle,[1] Lupe Furtado-Alle,[1] Cedric Gondro,[2] and João Carlos M. Magalhães[1]

[1] Department of Genetics, Polytechnic Center, Federal University of Parana, Jardim das Americas, 81531-990 Curitiba, PR, Brazil
[2] The Institute for Genetics and Bioinformatics (TIGB), University of New England, Armidale, NSW 2351, Australia

Correspondence should be addressed to Benn R. Alle, bennalle@gmail.com

Received 6 October 2008; Accepted 19 May 2009

Recommended by Marylyn Ritchie

This paper presents Kuri, a software package developed to simulate the temporal and spatial dynamics of genetic variability in populations and multispecies communities of trees, as well as their interactions with environmental factors. A conceptual model using agents inspired on Echo models is used to define the environment, the hierarchical structures, and the low-level rules of the system. At the individual agent (tree) level a genetic algorithm is used to model the genotypic structure and the genetic processes, from a small set of simple rules, complex higher-order population, and environmental interactions emerge. The program was written in Delphi for the Windows environment, and was designed to be used for educational and research purposes.

1. Introduction

Computational simulations have been widely used to represent and simulate genetic processes. Some examples that fall within the scope of this work include simulations that were mainly developed for educational purposes, such as Populus [1], WinPop [2], Sigex [3], and Genup [4]. Others were developed for practical applications and are used in, for example, programs for forestry management [5, 6]. Simulations are also used to understand complex adaptive systems from a "first-principles" approach. Conceptual models such as Holland's Echo model are widely used [3, 7]. In this paper we discuss the software Kuri, a simulator of ecological genetics for tree populations. The program allows investigation of genetic and microevolutionary phenomena of tree populations or entire forest communities. Kuri can be used to study the dynamics of neutral genetic markers under certain biological factors and environmental constraints, such as dispersion mechanisms and geographical barriers, among others. Either real field data or artificial genetic and environmental parameters can be used for a given simulation. The latter allows creation and testing of hypothetical situations for theoretical and/or educational purposes.

Along the same lines used in the Sigex simulator [3], Kuri mechanistically implements low level elementary biological rules, for example, Mendelian segregation and mating, which interact to produce patterns that are analogous to those observed in natural populations, such as Hardy-Weinberg equilibrium. Thus, population data generated in Kuri is not obtained from sampling from a distribution, but is instead, a quantifiable element at the population level which emerges from the low level mechanistic interactions at the genetic level.

2. Software Kuri

Kuri was developed using the Delphi programming language, an object oriented derivative of Pascal. It uses a modular construct which allows easy implementation of new functions and applications and also enables seamless integration with the other modules. The program needs limited computational resources and will run on a 1.2 GHz processor with 512 M RAM and 2 GB free space on the hard disk. The operating system can be Windows XP or above. The current version of Kuri consists of three main modules: the graphical user interface (GUI), a dispersion module, and a genetic operators (KGOP) module.

In Kuri, environmental factors that affect germination/viability of seeds are combined to create a heatmap in which the colors represent different germination probabilities (Figure 1 shows a screenshot of Kuri with a probability heatmap based on satellite images). The GUI allows the user to import images, such as satellite photographs or schematic pictures to represent features of interest in a given area. Up to five images at a time can be used to represent different environmental parameters in a given simulation. Each image could represent, for example: (1) inhospitable areas where seeds cannot germinate, (2) areas of human intervention, (3) soil depth, (4) soil quality, and (5) hydrology. Note that each environmental parameter can be altered by the user. For instance, the map of soil depth can be replaced by a topographic map of the region, if it is more relevant for a particular research topic. Currently Kuri works with bitmap image files which are easy to generate or to convert from other file formats with available imaging software.

For each of these (up to 5) environmental parameters, probabilities of germination success on its respective map can be assigned to either discrete features or interval ranges for continuous features. Probabilities are color coded on the map and resolved at the pixel level. This means that each pixel can be assigned its own independent probability, irrespective of neighboring probabilities, allowing for a discontinuous probability landscape. The color scheme of probabilities is user defined which makes it easy to identify features. For example, areas where the germination of seeds is impossible such as buildings, streets, water masses, or rocky terrain are by default represented in black (Figure 1). Since colors and probabilities are linked, it is simply a matter of changing the probability associated with a specific color to update all points in the map to a new probability.

The overall germination probability map (Figure 1) is generated by multiplying the probabilities for each of these five environmental parameters at each individual pixel. Thus probability at pixel px_i is simply

$$P(px_i) = \prod_{j=1}^{5} P\left(ep_{ij}\right), \qquad (1)$$

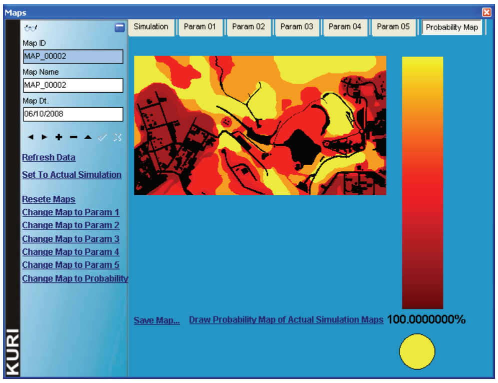
where ep is an environmental parameter. Color coding is used to represent the final probabilities on a scale between 0% and 100%. This assumes rather simplistically that the overall probabilities are independent terms with no interactions between parameters. To model interactions an additional proceeding can be used. If one of the parameters is a map of soil fertility and another map holds hydrology information, a table can be used to model the interaction between them, a page control called *interaction function*. This could be a simple scaling function, such that

$$P(px_i) = \lambda P\left(ep_{i1}\right) P\left(ep_{i2}\right), \qquad (2)$$

where λ is a scalar (in practice λ is simply a monochromatic map with a scalar attached to the single color). More complex nonlinear interactions can be envisioned (e.g., a mapping interval derived from the order terms of a random regression) provided (1) holds.

To simulate the dispersion of pollen and seeds, the total simulation area is divided into cells of user defined granularity, with height and width in pixels defined by the user. For each grain of pollen and for each seed in a particular cell, the probability of dispersing to another cell depends on the wind. This is achieved through a simple probabilistic function, where an integer ranging between 0 and n (n is a user defined parameter between zero and the total number of grid cells) is randomly sampled from a uniform distribution and multiplied by the probability of the wind direction (Figure 2). The value of n effectively sets the dispersion boundaries. Wind direction is also a user defined parameter consisting of a set of probabilities for each cardinal point and a decay rate from the center of dispersion.

The KGOP module is essentially a relational database that holds information on the biological community, the various species and their respective biological features, the genetic features of the species, and the genetic composition (essentially all allelic frequencies across all genes) of the population of each species, including the chromosome



Map ID: MAP_00002
Map Name: MAP_00002
Map Dt.: 06/10/2008
100.0000000%

Figure 1: Graphical user interface of Kuri showing the heatmap of seeding probabilities based on satellite imagery of Tangua Park in Curitiba, Brazil. Each color represents the combined probability of up to five different environmental parameters for each cell in the grid. Black is used to indicate nonviable regions (roads, rivers, built up areas, etc.).

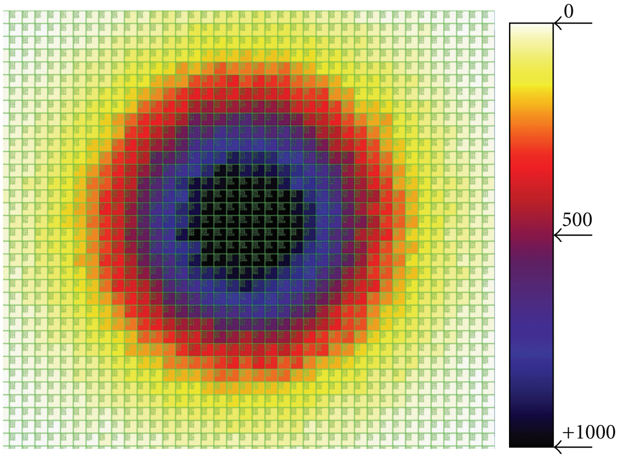


Figure 2: Heatmap of the dispersion of 1000000 pollen grains from a common origin in the center of the figure. Darker colors indicate more pollen in a given cell. In this example the wind direction probabilities were the same for all coordinates—hence the symmetric pattern of dispersion.

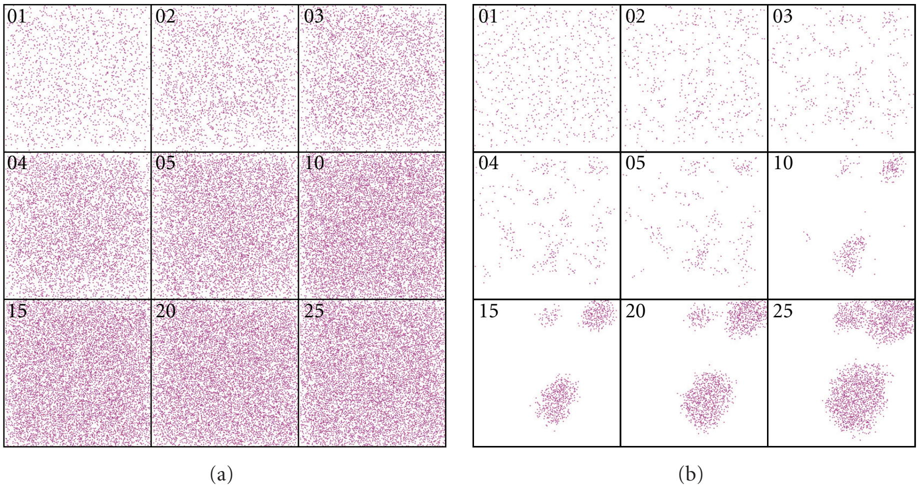


Figure 3: (a) Distribution of organisms in generations 1, 2, 3, 4, 5, 10, 15, 20, and 25 of replicate 1 under the scenario of strong winds. (b) Distribution of organisms in generations 1, 2, 3, 4, 5, 10, 15, 20, and 25 of replicate 1 under the scenario of mild winds. Each point represents the area occupied by an organism. Note how wind strength can affect the population structure and promote a shift from panmixia in (a) to endogamy in (b).

sets for each species with the number of loci in each chromosome, the linkage map between loci, and the number of alleles in each locus.

For each species the following biological parameters can be stored: the individual occupation range (species boundaries), the dispersion of pollen and seeds, the maximum and minimum ages of reproduction and images for each age group of the specimens. For this last parameter, Kuri's image collection can be used, or the user can import and add his/her own images. All parameters relate directly back to their original biological meaning and can be used quite intuitively.

For each new species added to the database, the user should specify the number of chromosomes that will be used in the simulation and the number of loci per chromosome. Up to 26 allele slots are available for each locus. The chromosomes and genes that will effectively be used in a simulation can be selected prior to a run. Recombination frequencies between genes should also be specified by the user. Mutation rates are the same for all genes/alleles, but can be changed across runs. Note that mutation in Kuri does not generate new allelic variants; it simply swaps an allele for another one from the database with a uniform probability. Initial populations are by default generated in Hardy-Weinberg equilibrium based on the given allelic frequencies (allelic and genotypic frequencies and chi-squared values for Hardy-Weinberg equilibrium tests are given in Kuri), but different initial population structures can be defined.

Computationally, the genetic mechanisms of the species are simulated using a Genetic algorithm (GA) [8]. In previous work we have [3] detailed how to implement these genetic processes and shown that they conform to theoretical predictions of population genetics. But briefly, GAs are the class of Evolutionary Computation algorithms which most closely mimic evolutionary processes at the genetic level. GA organisms are represented as linear strings which are referred to as chromosomes. The value in each position of the string is an allele and the position itself is a gene or locus. The combination of values (alleles) in the string (chromosome) can be mapped to a phenotypic expression (note that in Kuri all alleles are neutral). Thus GAs operate at two structural levels: a genotypic and a phenotypic one. Crossover swaps chromosome parts between selected parents to form the offspring while mutation changes the value of alleles at randomly selected loci.

The practical limits for the software (i.e., number of individuals, size of geographic area, number of generations, etc.) relate to the limits of the MySQL database. The effective size of the tables for the database is normally restricted by the operating system's filesystem. The total number of loci are limited to 128.

3. A Simulation Example: Dispersion Effects

In this section we discuss a simple simulation of seed dispersion effects to illustrate the use of Kuri in population genetics. We created a single species population in a homogeneous environment with a single locus and two segregating alleles of interest. Initially all plants were heterozygous. We ran the simulation under two scenarios with different wind intensities (strong and mild winds). Wind intensities affect the dispersion process and, consequently, the distribution of genetic variability.

For each scenario, five simulation runs of 25 generations each were performed. In Figure 3(a) the distribution pattern of the plants across generations is depicted under strong winds for the first replicate. Note that the distribution pattern remains homogeneous over the generations, meaning that dispersion occurs with a high level of panmixia, that is, random matting. Figure 3(b) shows the mild wind scenario over generations for the first replicate. Note the formation of endogamic groups, that is, most matings occur within

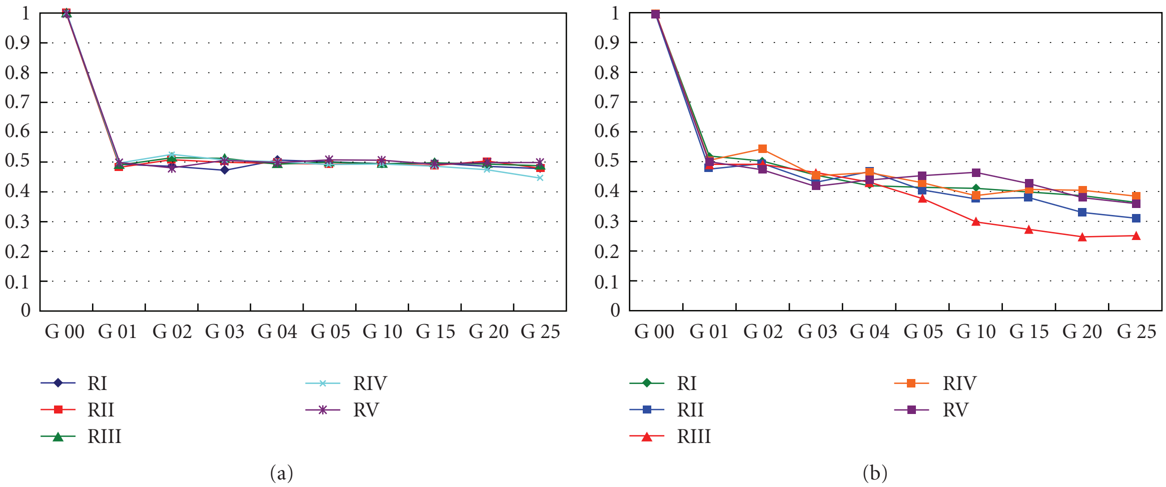


Figure 4: Changes in frequencies of heterozygotes observed across 25 generations in 5 repetitions. Initially the entire population was heterozygous. (a) Frequencies under the influence of strong winds. Equilibrium is reached after the first generation, oscillating around 0.5. (b) Frequencies under the influence of mild winds. Heterozygosity decreases due to population subdivision—Wahlund effect.

subpopulations, which are to be expected in an environment that does not favor dispersion.

The dynamics over time of the frequencies of heterozygotes for the five repeats are shown in Figure 4(a) (strong wind) and Figure 4(b) (mild wind). In the former, the frequencies of the heterozygotes reach equilibrium after the first generation, oscillating around 0.5. In the second case, a decrease in heterozygosity is noticeable since the subdivision creates a new population structure—an example of a genetic phenomenon known as Wahlund effect. In all strong wind repeats, equilibrium was reached and maintained across generations whilst with mild winds the number of homozygotes increases over time.

Even this simple scenario can provide insights about natural populations. Jump and Penuelas [9] showed that habitat fragmentation caused by human activity led to high levels of inbreeding due to a Wahlund effect. This was the first study showing that even widespread wind-pollinated trees are negatively affected by habitat fragmentation. Argumentatively, Kuri could be used to estimate genetic effects under different scenarios. For example, a satellite image of a forested area can be artificially fragmented in different patterns and these used to estimate the genetic effects of deforestation. This has implications for urbanization decisions and can assist in finding a solution that minimizes human impact. Clearly, for realistic results, there has to be reliable data and detailed knowledge of the ecology of the species.

For population studies the simulated data can be treated and analyzed as if it were *real data*, with the advantage of having full knowledge of the population structure and a handle on the mechanisms that yielded the dataset. For example, data from only the last generation could be used to make inferences about the evolutionary processes that were acting on the population. The degree of deviation from HW equilibrium can be calculated and used to estimate parameters such as F_{ST} [10]. These results can then be compared to the original experimental model to provide insights about the dynamics of the system.

Kuri was designed to simulate microevolutionary phenomena which can be detected through molecular markers which are usually selectively neutral. Neutral markers have the advantage that since they are not being selected for or against, any observed fluctuations in allelic frequencies are only due to population structure and environmental effects.

4. Concluding Remarks

Kuri can be used to simulate a wide range of biological scenarios. It allows manipulation of the genomes, alleles, and genotypes of different plant species and the interactions of these populations with the ecosystem. Kuri's database can be used to store different genetic models of species, being these based on real data of species or virtual organisms tailored for educational purposes. Alongside the biological parameters, the user can manipulate and/or create environmental parameters based on field data (such as satellite imagery) to study how these affect the genetic composition and size of populations. The software meets theoretical expectations, but it still has to be tested under realistic scenarios for which real data is available and results can be compared. Due to the lack of real data testing it is still unclear how detailed field data and knowledge of the ecology of the species has to be able to make valid inferences. Future work and user feedback may assist in answering these questions.

The software is modular. It was designed so that it can be modified and expanded to simulate other phenomena. For example, in the current version all genes/alleles are neutral, but it is straightforward to implement environmental constrains associated to the genotypes in order to simulate natural selection, or even simulate molecular evolution by adding another module that allows handling each allele as a DNA base pair. Kuri is open source and freely available from the web address: http://www.allesys.com.br/kuri/.

References

[1] D. N. Alstad, "Populus: simulations of population biology," 2007, http://www.cbs.umn.edu/populus.

[2] P. A. S. Nuin and P. A. Otto, "A program for representing and simulating population genetic phenomena," *Genetics and Molecular Biology*, vol. 23, no. 1, pp. 53–60, 2000.

[3] C. Gondro and J. C. M. Magalhães, "A simple genetic algorithm for studies of Mendelian populations," in *Recent Advances in Artificial Life*, H. Abbass, T. Bossamaier, and J. Wiles, Eds., pp. 85–98, World Scientific, London, UK, 2005.

[4] B. P. Kinghorn, "GENUP—a suite of programs to help teach animal breeding theory," in *Proceedings of the 10th Australian Association of Animal Breeding and Genetics*, pp. 555–559, 1992.

[5] M. Kanashiro, I. S. Thompson, J. A. G. Yared, et al., "Improving conservation values of managed forests: the Dendrogene project in the Brazilian Amazon," *Unasylva*, vol. 53, no. 209, pp. 25–33, 2002.

[6] B. Degen, H.-R. Gregorius, and F. Scholz, "ECO-GENE, a model for simulation studies on the spatial and temporal dynamics of genetic structures of tree populations," *Silvae Genetica*, vol. 45, no. 5-6, pp. 323–329, 1996.

[7] P. T. Hraber, T. Jones, and S. Forrest, "The ecology of Echo," *Artificial Life*, vol. 3, no. 3, pp. 165–190, 1997.

[8] C. Gondro and B. P. Kinghorn, "Solving complex problems with evolutionary computation," in *Proceedings of the 17th Australian Association for the Advancement of Animal Breeding and Genetics*, pp. 272–279, 2007.

[9] A. S. Jump and J. Penuelas, "Genetic effects of chronic habitat fragmentation in a wind-pollinated tree," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 103, no. 21, pp. 8096–8100, 2006.

[10] S. Wright, *Evolution and the Genetic of Populations, Vol. II: The Theory of Gene Frequencies*, The University of Chicago Press, Chicago, Ill, USA, 1969.