Continuous Mimetic Evolution

Antoine Ducoulombier[1] and Michèle Sebag[2,1]

(1) LRI, CNRS URA 410
Université d'Orsay
91405 Orsay Cedex
Antoine.Ducoulombier@lri.fr

(2) LMS, CNRS URA 317
Ecole Polytechnique
91128 Palaiseau Cedex
Michele.Sebag@polytechnique.fr

Abstract. There exists no memory of biologic evolution besides the individuals themselves. Indeed, the biologic milieu can change and a previously unfit action or individual can come to be more fit; it would be most dangerous to rely on the memory of the past.
This contrasts with artificial evolution most often considering a fixed milieu: the generation of an unfit individual previously explored is only a waste of time. This paper aims at constructing a memory of evolution, and using it to avoid such fruitless explorations. A new evolution scheme, called *mimetic evolution*, gradually constructs two models along evolution, respectively memorizing the best and the worst individuals of the past generations. Standard crossover and mutation are replaced by *mimetic mutation*: individuals are attracted or repelled by these models. Mimetic evolution is extended from binary to continuous search spaces. Results of experiments on large-sized problems are detailed and discussed.

1 Introduction

Biologic evolution takes place in a changing environment. Being able to repeat previously unsuccessful experiments is therefore vital. This could explain why Nature does not involve anything like an explicit memory: all the knowledge gathered by evolution is actually implicit and dispatched among the individuals.

Conversely, artificial evolution most often tackles optimization problems and considers *fixed* fitness landscapes. The history of evolution should thus provide reliable information; unfortunately, exploiting the list of all previously generated individuals gets soon intractable as evolution goes on. This paper focuses on constructing a tractable memory of evolution, and using it to guide the further evolution steps. This memory is explicit, in contrast with the implicit memory represented by the current population; and it is collective, i.e. accessible to all individuals, in contrast with the local memory carried by the individuals (e.g. the mutation step sizes [Sch81]).

Many works devoted to the control of evolution ultimately rely on some explicit collective memory of evolution. The memorization process can acquire numerical information; this is the case for the reward-based mechanism proposed by Davis to adjust the operator rates [Dav89], the adjustment of penalty factors in SAT problems [ER96] or the construction of discrete gradients [HOG95],

among others. The memorization process can also acquire symbolic information, represented as rules [RS96] or beliefs [Rey94].

Memory-based heuristics can control most steps of evolution: e.g. selection via penalty factors [ER96], operator rates [Dav89], operator effects [HOG95, RS96]... Memory can even be used to "remove genetics from the standard genetic algorithm" [Bal95]: the *Population Based Incremental Learning* (PBIL) algorithm memorizes the best individual of previous populations, and uses this memory to generate the next population from scratch.

Evolution by Inhibition symmetrically memorizes the worst individuals in each generation; this memory is used to modify the current population. The underlying metaphor is that of the *Loser*, virtual individual summarizing the past unfit individuals: offspring aim at be farther away from the loser, than their parents [SSR97].

This paper continues a previous work devoted to *Mimetic Evolution*, which combines PBIL and evolution by inhibition [PDR+97]. Mimetic evolution memorizes the best and the worst individuals previously met by evolution within two "models", the *Winner* and the *Loser*. Mimetic evolution is (remotely) inspired by the social evolution of individuals: any individual imitates, rejects, or ignores independently each one of the models. Practically, these models are used to evolve the genetic material via a single operator termed *social mutation*; social mutation is parameterized by the desired behavior of the individuals with respect to the models, so-called "social strategy". A range of social strategies has been defined, among which the *Entrepreneur* (which imitates the winner and ignores the loser); the *Conformist* (which imitates the winner and rejects the loser); the *Phobic* (which rejects the loser and ignores the winner; this behavior reproduces the Evolution by Inhibition scheme [SSR97])... Last, the *Ignorant* strategy (which ignores both models) serves as reference to check the relevance of the models.

In this paper, all cited schemes (PBIL, evolution by inhibition and mimetic evolution) are extended from binary to continuous search spaces. The interest of this extension is twofold. First, it allows evolution to directly consider the search space ($\mathbb{R}^N$) in many cases (e.g. most engineering optimization problems are numerical); and indeed, the discretization of the search space can hinder evolution [B95]. Second, it incidentally settles the main drawback of binary mimetic evolution, namely the adjustment of the mutation rate (metaphorically, the strength of the social pressure).

This paper is organized as follows. Section 2 first reviews some related work, and describes how the use of virtual individuals, imaginary individuals or models, can support evolution. Binary mimetic evolution is then briefly recalled, in order for this paper to be self-contained; and continuous mimetic evolution is detailed (section 3). Section 4 presents and discusses experiments on several large-sized problems in continuous search spaces. We last conclude and detail some perspectives of research.

2 State of the art

A major question in the field of artificial evolution is that of the respective roles of crossover and mutation. Though the question concerns both binary and continuous search spaces, only the binary case will be considered in this section.

Crossover traditionally relies on the Building Block hypothesis [Hol75, Gol89]. But a growing body of evidence suggests that crossover is efficient because it operates large step mutations. In particular, T. Jones has studied the macro-mutation operator defined as crossing over a parent with a random individual[1]. Macro-mutation obviously does not allow the offspring to combine the building blocks of their two parents; still, macro-mutation happens to outperform standard crossover on benchmark problems [Jon95].

More generally, standard crossover actually behaves like a biased mutation operator. The bias depends on the population and controls both the strength and the direction of the mutation. The "mutation rate" of standard crossover, e.g. the Hamming distance between parents and offspring, depends on average on the diversity of the population; and the "mutation direction" of standard crossover (which genes are modified) also depends on the population.

On the other hand, binary mutation primarily aims at preserving the genetic diversity of the population. This can be done as well through crossover with specific individuals, deliberately maintained in the population to prevent the loss of genetic diversity. For instance, the Surrogate GA [Eva97] maintains imaginary individuals such as the complementary of the best current individual, or all-0 and all-1 individuals; crossover alone thus becomes sufficient to ensure the genetic diversity of the population, and mutation is no longer needed. Another possibility is to deliberately introduce genotypic diversity by embedding the search space Ω into $\{0,1\} \times \Omega$ and identifying the individuals 0ω and $1\bar{\omega}$, as done in Dual Genetic Algorithms [PA94].

Evolution can also be supported by virtual individuals, i.e. individuals belonging neither to the population nor to the search space. This is the case in the PBIL algorithm, where the best individuals (elements of $\{0,1\}^N$) in the previous populations are memorized within an element of $[0,1]^N$. This vector noted $\mathcal{M}$ provides an alternative to crossover and mutation, in that it allows PBIL to generate the current population from scratch: for each individual X and each bit i, value X_i is randomly selected such that $P(X_i = 1) = \mathcal{M}_i$ (where A_i denotes as usual the i-th component of A). $\mathcal{M}$ is initialized to $(0.5, 0.5, ..., 0.5)$ and it is updated from the best individual X_{max} at each generation, by relaxation :

$$\mathcal{M} := (1 - \alpha)\mathcal{M} + \alpha X_{max}$$

where α in $[0,1]$) is the relaxation factor, which corresponds to the fading of the memory. The main advantage of PBIL is its simplicity: it does not involve any modification of the genetic material. The only information transmitted from one

[1] Note that this macro-mutation fairly resembles standard crossover on large populations during the first generations of evolution.

generation to another is related to the best individual; still, it is not necessarily sufficient to reconstruct this best individual. This might hinder the exploitation of narrow highly fit regions, such as encountered in the Long Path problem [HG95]. Practically, one sees that even if $\mathcal{M}$ has very little difference with an individual on the path, the population constructed from $\mathcal{M}$ might not overlap the path [SS96].

Evolution by Inhibition involves the opposite memory, that is, the memory of the worst individuals in the previous populations. This memory noted $\mathcal{L}$ (for *Loser*) is also an element of $[0,1]^N$, constructed by relaxation :

$$\mathcal{L} := (1-\alpha)\mathcal{L} + \alpha X_{min}$$

where X_{min} denotes the average of half the worst offspring, and α is the relaxation factor. In contrast with PBIL which uses $\mathcal{M}$ to generate a new population, $\mathcal{L}$ is actually used to evolve the current population via a specific operator termed *flee-mutation*. Flee-mutation replaces both mutation and crossover; for each individual X, it selects and flips the bits most similar to those of the loser (minimizing $|X_i - \mathcal{L}_i|$). The offspring thus is farther away from the loser, than the parent was. Metaphorically, the goal of this evolutionary scheme is: Be different from the Loser ! And incidentally, this reduces the chance for exploring again low fit regions.

The potential of evolution by inhibitions is demonstrated for appropriate settings of the flee-mutation rate (number of bits mutated): EBI then significantly outperforms PBIL [SSR97], which itself outperforms most standard discrete optimization algorithms [Bal95]. But the adjustment of the flee-mutation rate remains an open question.

3 Mimetic evolution

Mimetic evolution melts PBIL and evolution by inhibition: besides the Loser constructed by EBI, it uses the memory of best individuals constructed by PBIL, or *Winner*, to guide evolution. This section briefly recalls how mimetic evolution was implemented in binary search spaces (more detail is found in [PDR+97]), and details how it extends to continuous search spaces.

3.1 Binary mimetic evolution

Two elements of $[0,1]^N$, thereafter called *models*, are gradually constructed by relaxation from the population. These models, the winner $\mathcal{W}$ and the loser $\mathcal{L}$, respectively reflect the best and the worst individuals encountered by evolution so far (Table 1).

Let us first examine how $\mathcal{W}$ can help evolving individual X. Given the most fit individuals of the population (X, Y and Z), some possible causes for being fit are ($bit_2 = 1$), or ($bit_3 = 1$), or ($bit_5 = 1$) (a majority of the most fit individuals has those bits set to this value). Thus, one might want for instance to flip bit_2 and let bit_3 unchanged in X; this amounts to making X more similar to $d\mathcal{W}$,

which goes to $\mathcal{W}$ in the limit. Metaphorically, X thus "imitates" the winner $\mathcal{W}$. Practically, the bits mutated in X are selected by a tournament of bits, as those maximizing $|X_i - \mathcal{W}_i|$; this draws the offspring closer to $\mathcal{W}$ than X was.

	1	2	3	4	5	Fitness
X	0	0	1	0	0	Fit
Y	1	1	1	1	1	Fit
Z	0	1	1	0	1	Fit
$d\mathcal{W}$	0.33	0.66	1	0.33	0.66	
S	0	0	0	1	0	Unfit
T	1	0	1	1	1	Unfit
U	1	0	0	1	1	Unfit
$d\mathcal{L}$	0.66	0	0.33	1	0.66	

$$\mathcal{W} := (1 - \alpha_w)\mathcal{W} + \alpha_w d\mathcal{W}$$

$$\mathcal{L} := (1 - \alpha_l)\mathcal{L} + \alpha_l d\mathcal{L}$$

Table 1. *Individuals and virtual individuals*

This mechanism is refined by taking advantage of the loser too. For instance, according to $d\mathcal{W}$, it might be a good idea to mutate bit 5; but $d\mathcal{L}$ suggests that $(bit_5 = 1)$ is not a factor of high fitness. This leads to select the bits to mutate, so that the offspring "imitates" $\mathcal{W}$ and "rejects" $\mathcal{L}$.

Practically, a new operator termed *social mutation* is defined. In each individual X, social mutation modifies a user-supplied number M of bits; these bits are selected by tournament as those maximizing $|X_i - \mathcal{W}_i|$ - $|X_i - \mathcal{L}_i|$; the bits mutated thus depend on X and on the models.

However, there is no reason why one could only imitate the winner and reject the loser. A straightforward generalization is to define a pair (δ_W, δ_L) in $\mathbb{R}^2$, and to select the bits to mutate as those maximizing :

$$\delta_W |X_i - \mathcal{W}_i| + \delta_L |X_i - \mathcal{L}_i|$$

One sees that X imitates model $\mathcal{M}$ $(= \mathcal{W}$ or $\mathcal{L})$ if $\delta_M > 0$, rejects $\mathcal{M}$ if $\delta_M < 0$, and ignores $\mathcal{M}$ if $\delta_M = 0$. Social mutation finally gets parameterized by the pair (δ_W, δ_L), termed *social strategy.* Some of these strategies have been given metaphorical names for the sake of convenience; obviously, other metaphors could have been imagined. We distinguish mainly:

- The *conformist*, that imitates the winner and rejects the loser;
- The *phobic*, that rejects the loser and ignores the winner;
- The *ignorant*, that ignores both the loser and the winner.

One notices that the social strategy is unchanged if δ_W and δ_L are multiplied by a positive coefficient. Social strategies can then be represented as angles. This angle gives the preferred direction of the individuals, in the changing system of coordinates given by the winner and the loser. Figure 1 shows the possible

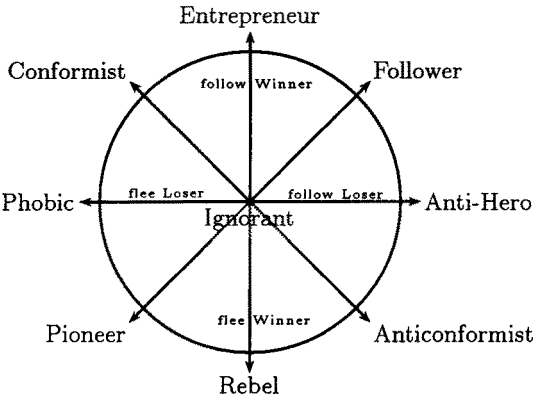

Figure 1. Mimetic Strategies

directions, with angle 0 corresponding to rejecting the loser, and angle $\pi/2$ to following the winner.

The main weakness of binary mimetic evolution is the adjustment of the social mutation rate, that is the number of bits to mutate in each individual. The difficulties encountered have been discussed in [SSR97, PDR+97].

3.2 Continuous social mutation

Mutation offers rather different difficulties depending on whether the search space is binary or continuous.

In a continuous search space, mutation usually proceeds by adding a gaussian perturbation $N(0, \sigma_i)$ to each component X_i of an individual X. The question is how much each gene should be modified, that is, how to set σ_i. To the best of our knowledge, the most efficient answer so far is given by self-adaptive mutation, stemmed from the Evolution Strategy scheme [Sch81, BS93]: the genotypic material of the individual is enhanced with the vector of step sizes $(\sigma_1, \ldots, \sigma_N)$, and evolution then adjusts "for free" the σ_i. Practically, σ_i first undergoes a gaussian mutation with a fixed variance depending on the size of the problem (the recommended values of the parameters are indicated below; see [Sch81] for more detail). The modified σ_i is then used to modify X_i :

$$\begin{array}{ll} \tau_{glob} := \tau N(0,1) & \tau \approx \frac{1}{\sqrt{2N}} \\ for\ i = 1..N & \\ \quad \sigma_i := \sigma_i * exp(\tau_{glob} + \tau_{loc} N(0,1)) & \tau_{loc} \approx \frac{1}{\sqrt{2\sqrt{N}}} \\ \quad X_i := X_i + N(0, \sigma_i) & \end{array}$$

Evolution thereby hopefully favors individuals having both accurate phenotypes (i.e. with high performance) and accurate step sizes.

The extension of social mutation to continuous search spaces mostly requires to define how the winner and the loser are used to guide the mutation. Indeed, the computation of the winner and the loser straightforwardly extends from binary to continuous search space, with α_w and α_l denoting the relaxation factors of respectively the winner and the loser:

$$\mathcal{W} := (1 - \alpha_w)\mathcal{W} + \alpha_w \; d\mathcal{W}$$
$$\mathcal{L} := (1 - \alpha_l)\mathcal{L} + \alpha_l \; d\mathcal{L}$$

where $d\mathcal{W}$ and $d\mathcal{L}$ respectively stand for the average of the best (resp. worse) offspring. The relaxation factors α_w and α_l are equal in the experiments.

We investigate two evolution operators. The first one, termed *Fixed social mutation*, involves a fixed social strategy (δ_W, δ_L). If we consider the bi-dimensional space including the individual at hand, the winner and the loser, a social strategy defines a direction in this 2D space (Figure 2). For a given mutation step size, this direction defines a target offspring. The fixed social mutation is built from a standard self-adaptive gaussian mutation, and biased so as to produce an offspring closer to the target offspring, than the parent.

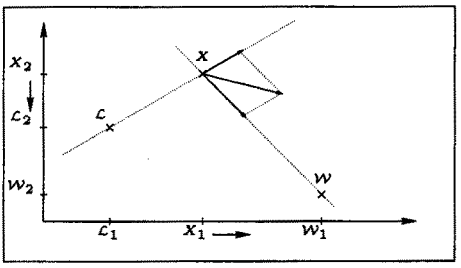

Figure 2. Fixed social mutation in $\mathbb{R}^2$, for a Conformist strategy.

More precisely the sign of the gaussian perturbation is determined so as to move the offspring in the desired direction:

$$X_i := X_i + sign(\delta_L(X_i - \mathcal{L}_i) + \delta_W(X_i - \mathcal{W}_i)) \times |N(0, \sigma_i)|$$

where $sign(A)$ is 1 if A is positive, -1 otherwise.
This evolution scheme is much dependent on the user-supplied strategy: the only degree of freedom is provided by the fact that the system of reference given by the winner and the loser evolves itself. Still it will be hard to recover from a bad social strategy.

A second evolution operator is termed self-adaptive social mutation, as it self-adapts the social strategy of the individuals. The self adaptation of the social strategy parallels that of the mutation step size in self-adaptive mutation.

More precisely, the individual is enhanced with the description of its personal strategy, given as two positive or negative scalars δ_W and δ_L. Evolution thus adjusts for free the social strategy most suited to each individual [2].

The self-adapted social strategy again determines a target offspring with:

$$X_i := X_i + \delta_L(X_i - \mathcal{L}_i) + \delta_W(X_i - \mathcal{W}_i)$$

This extends naturally to self adaptive *vectorial* social mutation where δ_W and δ_L are vectors.

3.3 Continuous PBIL

To the best of our knowledge, there has been only one other attempt to extend PBIL to continuous search space so far [STMS97]. This extension is based on uniform sampling of the domains of the genes, and the PBIL mechanism is used to gradually restrict the domains explored. Let $[Min_i, Max_i]$ be the initial domain of the i-th gene, and let Moy_i denote the half sum of Min_i and Max_i. When generating an individual, one first decides whether X_i should belong to $[Min_i, Moy_i]$ or $[Moy_i, Max_i]$, with:

$$Proba(X_i > Moy_i) = \mathcal{W}_i$$

One then draws X_i with uniform probability in the selected half interval. When $\mathcal{W}_i$ reaches a given threshold (.1 or .9), the domain $[Min_i, Max_i]$ shrinks accordingly (being respectively replaced with $[Min_i, Moy_i]$ or $[Moy_i, Max_i]$). One disadvantage of this procedure is that the search space can only shrink: there is no way to recover from a bad previous choice.

The approach investigated here relies on the natural extension of the computation of $\mathcal{W}$, from the two best individuals X^1_{max} and X^2_{max} and the worst individual X_{min}, as employed in [Bal95] to decrease the odds of premature convergence of $\mathcal{W}$. One finally has:

$$\mathcal{W} := (1-\alpha)\mathcal{W} + \alpha(X^1_{max} + X^2_{max} - X_{min})$$

$\mathcal{W}$ thus gives the "center" of the region to be sampled in the next population. The sampling involves independent gaussian distributions for each gene i, centered on $\mathcal{W}_i$, and we investigate three mechanisms for determining the variance σ_i of the distributions.

The first one, termed *Constant PBIL*, explores a fixed region centered on $\mathcal{W}$, with $\sigma_i = .1$.

The second one, termed *Adaptive PBIL*, computes σ_i as the variance on gene i

[2] Incidentally, this scheme is more satisfactory from the point of view of social modeling (but indeed social modeling is far beyond the scope of this paper), as it constructs populations combining various types of social strategies. It would be most interesting to get, as a by-product of evolution, the social strategy most adapted to the last explored regions of the fitness landscape.

of the best half of the population.
The third one, *Self-Adaptive PBIL*, self adapts the variance σ_i per individual as follows. One considers a standard $(\mu + \lambda)$ evolution strategy, where an offspring is generated from $\mathcal{W}$ and the variance σ_i of the parent at hand. Here, ES evolves the behavior of the individual with respect to model $\mathcal{W}$, so that the set of behaviors retained is most susceptible to improve $\mathcal{W}$.

4 Experimental Validation

4.1 Problems

The experiments consider the same functions as [Bal95].

$$y_1 = x_1 \qquad y_i = x_i + y_{i-1},\ i \geq 2 \qquad F_1 = \frac{100}{10^{-5} + \Sigma_i |y_i|}$$

$$y_1 = x_1 \qquad y_i = x_i + sin(y_{i-1}),\ i \geq 2 \qquad F_2 = \frac{100}{10^{-5} + \Sigma_i |y_i|}$$

$$F_3 = \frac{100}{10^{-5} + \Sigma_i |.024 \times (i+1) - x_i|}$$

Functions F_1, F_2 and F_3 are defined on $[-2,56, 2.56[^{100}$.

4.2 Experimental setting

The evolution scheme is a (10+50)-ES: 10 parents produce 50 offspring and the 10 best individuals among parents plus offspring are retained in the next population. A run is allowed 200,000 evaluations; all results are averaged on 20 independent runs. The relaxation factors α_w and α_l are both set to .01.

The results obtained are represented in polar coordinates (ρ, θ), where θ stands for the social strategy (see section 3.1) and ρ denotes the average best performance obtained for this strategy (each point on the circle thus represents $4,000,000$ evaluations). The unit circle serves as reference: it corresponds to the *ignorant* strategy, that is, a standard (10+50)-ES.

The results of adaptive social mutation and continuous PBIL are also indicated.

4.3 Continuous results and Discussion

In a continuous search space, the ignorant strategy coincides with a standard ES; no wonder that it gets good results, and is hard to be caught up.

The bad performance of fixed mimetic evolution can partly be blamed on what follows. The direction of evolution of the individuals is given in the changing system of coordinates defined by the winner and the loser — and this direction does not change for non-adaptive social mutation. Still, the loser $\mathcal{L}$ changes

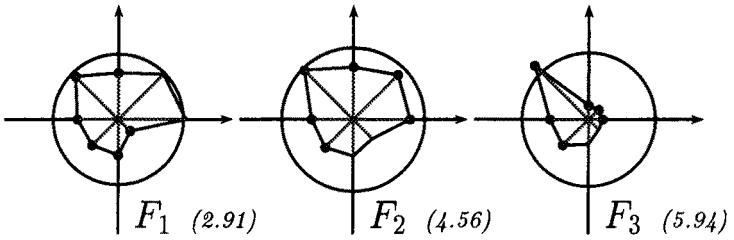

Figure 3. Fixed continuous social strategies on F_1, F_2 and F_3. The reference circle represents the performance of the ignorant (given in parenthesis)

faster than the winner $\mathcal{W}$ as $\alpha_w = \alpha_l$ and the best individuals of the population vary less than the worst individuals. But following an invariant direction defined with respect to both a fixed and a changing reference point can result in satellizing the individual around the fixed reference point; this also holds for vectorial social strategy.
Additional experiments show that setting $\alpha_w << \alpha_l$ actually improves the performance of fixed mimetic evolution for some strategies, though it still does not catch up the ignorant strategy.

	F_1	F_2	F_3
Reference (Ignorant)	2.91	4.56	5.94
CME with Self Adaptation of δ_W, δ_L	1.18	2.57	3.40
CME with Self Adaptation of $\overrightarrow{\delta_W}, \overrightarrow{\delta_L}$	0.07	0.87	0.87
Constant-PBIL	3.13	3.55	13.69
Adaptive-PBIL	1.09	2.31	6.20
Self-Adaptive-PBIL	1.34	1.85	2.69

Table 2. Continuous Mimetic Evolution (CME) and PBIL

The adaptive social mutation encounters other problems. Let us first consider the scalar case. The social strategy (δ_L, δ_W) controls both the direction of mutation, and the mutation step size (section 3.2). Coefficients δ_L and δ_W must therefore be unbounded and can be both positive and negative (to explore all directions of the bi-dimensional space defined by the winner and the loser). Still, the update of δ_L and δ_W, copied from the self-adaptive mutation, primarily aims at exploring $\mathbb{R}^+$ rather than $\mathbb{R}$. The bad performance of adaptive social mutation is thus explained by the fact that the social strategy is not adjusted with sufficient flexibility. Same holds when δ_L and δ_W are vectors rather than scalars. Further research is concerned with designing other mechanisms to evolve the social strategy with more flexibility.

The continuous PBIL obtains good results. The fact that *Constant PBIL* happens to supersede the binary PBIL and ES, satisfactorily demonstrates that the winner is accurately determined and duly wanders in the desired regions. Still, Table 2 shows that none of our attempts so far to adjust the range of exploration, was successful as the best results are obtained for a fixed σ_i. Empirically, the adaptive and self-adaptive adjustments of σ_i rapidly lead to small values of σ_i, which hinders the search as they can only slowly increase when the models rapidly change.

5 Conclusion

This paper investigates how the memory of evolution can support and speed up evolution. Given the fact that the exhaustive history of evolution cannot be tractably exploited after the first generations, the individuals previously met by evolution are packed in form of *models*. The PBIL algorithm [BC95] and Evolution by Inhibitions [SSR97] demonstrated how evolution can respectively take advantage of the model memorizing the best and the worst individuals. A major drawback of these approaches is that evolution easily gets stuck, as individuals only observe one model and adopt a single predetermined behavior (imitation or avoidance) with respect to this model.

Mimetic evolution combines these schemes and uses both models to evolve the current population in binary search spaces [PDR+97]. As one can combine *ad libitum* the influence (basically attractive, repulsive or indifferent) of each model, an individual is offered a rich variety of directions of evolution, metaphorically the "social strategies" of evolution. And indeed, the use of two models avoids some deadlocks of evolution, for the influence of one model acts as a perturbation with respect to the influence of the other one: it gets more difficult to get stuck.

Still, the extension of mimetic evolution to continuous search spaces presented in this paper, shows the limits of the memory mechanism proposed so far. In particular, we clearly need an indicator telling when a model gets stuck, so that to modify the social strategy of an individual regarding this model. Moreover, the continuous mimetic machinery mixes up two different notions, namely the recommended direction of evolution in the changing system of coordinates defined by the models, and the social pressure, namely how far should an individual go in this direction.

Further research is concerned with implementing two kinds of memory, updated at different speed rates. The comparison would hopefully allow one to detect that the models get irrelevant or ineffective to the current stage of evolution: the model gets stuck if the fast recent memory is closer and closer to the slow antique one. Obviously, such a mechanism could take a clue from the long term *versus* short term memories of human beings.

References

B95. T. Bäck. *Evolutionary Algorithms in theory and practice.* New-York:Oxford University Press, 1995.

BC95. S. Baluja and R. Caruana. Removing the genetics from the standard genetic algorithms. In A. Prieditis and S. Russel, editors, *Proc. of ICML95*, pages 38–46. Morgan Kaufmann, 1995.

Bal95. S. Baluja. An empirical comparizon of seven iterative and evolutionary function optimization heuristics. Technical Report CMU-CS-95-193, Carnegie Mellon University, 1995.

BS93. T. Bäck and H.-P. Schwefel. An overview of evolutionary algorithms for parameter optimization. *Evolutionary Computation*, 1(1):1–23, 1993.

Dav89. L. Davis. Adapting operator probabilities in genetic algorithms. In J. D. Schaffer, editor, *Proc. of the* 3^{rd} *ICGA*, pages 61-69. M. Kaufmann, 1989.

ER96. A.E. Eiben and Z. Ruttkay. Self-adaptivity for constraint satisfaction: Learning penalty functions. In T. Fukuda, editor, *Proc. of the* 3^{rd} *IEEE ICEC*, pages 258–261. IEEE Service Center, 1996.

Eva97. I.K. Evans. Enhancing recombination with the complementary surrogate genetic algorithm. In T. Bäck, Z. Michalewicz, and X. Yao, editors, *Proc. of the Fourth IEEE ICEC*, pages 97–102. IEEE Press, 1997.

Gol89. D. E. Goldberg. *Genetic algorithms in search, optimization and machine learning.* Addison Wesley, 1989.

HG95. J. Horn and D.E. Goldberg. Genetic algorithms difficulty and the modality of fitness landscapes. In L. D. Whitley and M. D. Vose, editors, *Foundations of Genetic Algorithms 3*, pages 243–269. Morgan Kaufmann, 1995.

HOG95. N. Hansen, A. Ostermeier, and A. Gawelczyk. On the adaptation of arbitrary normal mutation distributions in evolution strategies: The generating set adaptation. In L. J. Eshelman, editor, *Proc. of the* 6^{th} *ICGA*, pages 57–64. Morgan Kaufmann, 1995.

Hol75. J. H. Holland. *Adaptation in natural and artificial systems.* University of Michigan Press, Ann Arbor, 1975.

Jon95. T. Jones. Crossover, macromutation and population-based search. In L. J. Eshelman, editor, *Proc. of the* 6^{th} *ICGA*, pages 73–80. Morgan Kaufmann, 1995.

PA94. P.Collard and J.P. Aurand. Dual ga: An efficient genetic algorithm. In *Proc. of ECAI*, pages 487–491. Amsterdam, Wiley and sons, August 1994.

PDR+97. M. Peyral, A. Ducoulombier, C. Ravisé, M. Schoenauer, and M. Sebag. Mimetic evolution. In *Artificial Evolution'97.* To appear, Springer-Verlag.

Rey94. R.G. Reynolds. An introduction to cultural algorithms. In *Proc. of the* 3^{rd} *Annual Conference on EP*, pages 131–139. World Scientific, 1994.

RS96. C. Ravisé and M. Sebag. An advanced evolution should not repeat its past errors. In L. Saitta, editor, *Proc. of the* 13^{th} *ICML*, pages 400–408, 1996.

Sch81. H.-P. Schwefel. *Numerical Optimization of Computer Models.* John Wiley & Sons, New-York, 1981. 1995 - 2^{nd} edition.

SS96. M. Sebag and M. Schoenauer. Mutation by imitation in boolean evolution strategies. In H.-M. Voigt, W. Ebeling, I. Rechenberg, and H.-P. Schwefel, editors, *Proc. of the* 4^{th} *Conference on PPSN*, pages 356–365. Springer-Verlag, LNCS 1141, 1996.

SSR97. M. Sebag, M. Schoenauer, and C. Ravisé. Toward civilized evolution: Developing inhibitions. In Th. Bäeck, editor, *Proc. of the* 7^{th} *ICGA*. Morgan Kaufmann, 1997.

STMS97. I. Servet, L. Trave-Massuyes, and D. Stern. Telephone network traffic overloading diagnosis and evolutionary computation technique. In *Artificial Evolution'97.* To appear, Springer-Verlag.