Toward Truly "memetic" Memetic Algorithms: discussion and proofs of concept

Natalio Krasnogor and Steven Gustafson

School of Chemistry and School of Computer Sciences & IT
University of Nottingham, Nottingham, United Kingdom
http://slater.chem.nott.ac.uk/~natk/Public/index.html
{nxk | smg}@cs.nott.ac.uk

Abstract. A vast number of very successful applications of Memetic algorithms (MAs) have been reported in the literature in the last years for a wide range of problem domains. The majority of the papers dealing with MAs are the result of the combination of highly specialised **pre-existing** local searchers and usually purpose-specific genetic operators. Moreover, those algorithms require a considerable effort devoted to the tuning of the local search and evolutionary parts of the algorithm.
We have demonstrated in our previous work (see references below), that given a range of possible local search strategies available to a Memetic Algorithm, the optimal choice of which one must be used is not only problem and instance dependent but also tightly related to the state of the search process itself. We also showed that it is indeed possible to produce Memetic Algorithms that adapt on-the-fly to those situations for a variety of problem domains.
In this paper we continue our studies of the design of robust Memetic Algorithms by introducing the concept of "self-generating" Memetic Algorithms. As mentioned above the success of a Memetic Algorithm depends on the pre-existence of powerful local searchers. Here we allow the Memetic Algorithm to **create** its local searchers and to co-evolve the behaviours it needs to successfully solve a problem.

1 Introduction

Although a number of very successful applications of Memetic algorithms have been reported in the literature, many of these results not only arise from highly specific combinations of specialised operators (e.g. [36],[35]) but also require considerable tuning of the local search and evolutionary parts of the algorithm. We have demonstrated previously that, given a range of possible local search strategies available to an MA, the optimal choice is not only problem dependent but also strongly related to the state of the search process itself, and we showed that it is indeed possible to produce Memetic Algorithms that adapt to those situations[24],[25],[6],[28],[30] for a variety of problem domains and situations.

This highlights some important issues that must be addressed if the use of these new search algorithms is to progress beyond the research laboratory and

into the real-world. Foremost amongst these is the requirement that in order to have greater applicability, variants of MAs are needed that do not require a detailed knowledge of the problem structure and a subtle understanding of the interplay of local search and evolutionary operators; what it is needed is an MA that can automatically discover the specific kind of local search that is suitable to the instance the algorithm is trying to solve. Moreover, it would be expected that even when no "silver bullet" local search heuristic is known for a given NP-hard optimization problem (e.g. like Lin-Kernighan or K-opt heuristics for *TSP* or *Graph Partitioning*) these new Self-Generating Memetic Algorithms will still be competent[19].

In this paper we propose to investigate a novel means of automating the process of discovering successful local search strategies for MAs where the search strategies (themselves enconded as individuals of a GP population) are co-evolved alongside the population of potential solutions. The representation of local search operators (memes) can include features such as the acceptance strategy, the maximum number of neighborhood members to be sampled, the number of iterations for which the meme should be run, a decision function that will tell the meme whether it is worth or not to be applied on a particular individual and, perhaps more importantly, the move operator itself in which the meme will be based(see [24] for a discussion of these issues).

This new approach differs from standard adaptive and self-adaptive evolutionary algorithm (e.g. [13],[3],[47],[22]), adaptive memetic algorithms (e.g. [20],[32],[24]) and also hyperheuristics (e.g. [11],[10]). Those approaches mainly focus on adapting the probabilities of applying the genetic operators, the size of the populations (e.g [48]) or the decision of which operator to choose for local search out of a fix set of local searchers[28],[25],[6].

Of particular relevance to this paper are adaptive MAs which come in three main flavors. One option is to adapt on-line the decision on which points are worth of considering for local search in such a way that precious cpu time is not wasted with solutions that are local-optima or close to optima [20]. Land [32] adapted the intensity of local search, that is, how deep the local search must be done and also utilizes the concept of "sniffs" to gouge the potential benefits of doing local search in the vicinity of a solution.

In turn, Krasnogor[24] introduced Multimeme Algorithms where a finite set of local searchers (codified as memeplexes) where given to the memetic algorithm and the choice of which one must be used was learned on the fly. In Hyperheuristics, in contrast, there is no co-evolution neither a population of solutions is maintained. In hyperheuristics what is learn is the best way to apply in tandem a set of heuristic methods (mainly constructive methods) that incrementally builds up a solution to a problem. All these methods can be expected to perform well in problems for which a well studied and tested set of optimization heuristics is known **a priori**. Moreover they lack the ability to create or discover novel local searcher if the need arises (e.g. the search is stuck in a local optimum for a long period of time). It is this feature, the creative design of new local

search heuristics on-the-fly, that distinguishes Self-Generating Metaheuristics (in particular Self-Generating MAs) from the previous approaches mentioned above.

2 Toward truly "Memetic" Memetic Algorithms

Memetic algorithms are not the first kind of algorithms to draw inspiration from natural phenomena. In this case the inspiration came from memetic theory. However, unlike Simulate annealing, Ant Colony optimization, GAs, etc., scholars working on MAs, as will be argued later, departed considerably from the metaphor and ignored its main features.

The common use of the term "memetic algorithm" refers to an evolutionary algorithm that employs as a distinctive part of its main evolutionary cycle (mutation, crossover and selection), a local search stage.

The name "memetic algorithm" is a very contested label that stirs critics and controversies among researchers and practitioners. We take the position that by labeling these algorithms as memetic algorithms and not calling them something else (e.g. lamarckian GAs, genetic local search, hybrid GAs, etc.) there is much to be learned. But for this to happen we need to put back the "memetic" into memetic algorithms.

Memetic theory started as such with the definition given by R. Dawkins of a meme as a unit of cultural inheritance[14][1]. Many other researchers and philosophers "flirted" with the idea that cultural phenomena can somehow be explained in evolutionary terms even before Dawkins' introduction of a meme. Other symbols were introduced to refer to the elementary unit of cultural change and/or transmission (e.g. *m-culture* and *i-culture* [8], *culture-type* [44], etc.). See [18] for a comprehensive analysis. The merit of Dawkins contribution can be attributed to his insight into correctly assigning a new signifier, i.e. a label or symbol, to the thing being signified, i.e. the unit of cultural transmission. The term meme was a new word hence it was not loaded with preconceptions and misconceptions. From the computer sciences perspective it was appealing because it defined that concept as a discrete structure which can be easily harnessed in a computer program and, as we will show in this paper, can be evolved to suit different problems, different instances of the same problem and different stages of the search process itself.

The fundamental innovation of memetic theory is the recognition that a dual system of inheritance, by means of the existence of two distinct replicators, mold human culture. Moreover, these two replicators interact and co-evolve shaping each other's environment. As a consequence evolutionary changes at the gene level are expected to influence the second replicator, the memes. Symmetrically, evolutionary changes in the meme pool can have consequences for the genes.

Critics of memetic theory often rebuff that the meme idea is useless as it has not been found yet the basic encoding*unit* or representation building block of what constitute a meme. In our opinion this is an unscientific approach that

[1] The definition was later refined in [15]

bears no relation with reality. The reader must note that before the double-helix nature of the DNA was discerned, and even before the discovery of the DNA as the carrier of genetic information, the Darwins (Charless and Erasmus), Wallace and Mendel already built the basis of our current understanding of evolutionary theory; molecular biology came much later. It is not reasonable then, to dismiss Memetic theory on the previously mentioned ground. The reader is invited to refer to[1] for recent papers on Memetic theory.

2.1 Memetic Theory in Evolutionary Computation

In any of the major evolutionary computation paradigms, e.g. GAs, Evolution Programs, Evolutionary Strategies, GPs, etc, the computation cycle shown in graph 1 takes place.

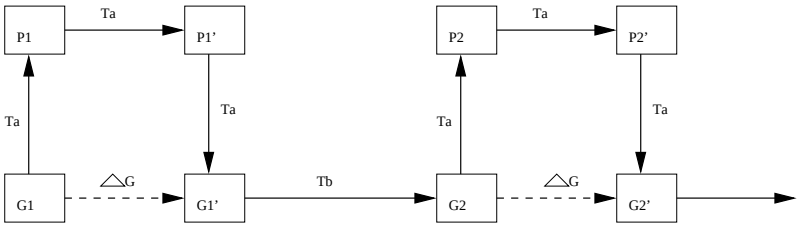


Fig. 1. *Evolutionary genetic cycle.*

In figure 1 a hypothetical[2] population of individuals is represented at two different points in time, generation 1 (G1) and at a later generation (G2). In the lower line, G_i for $i = 1, 2$ represents the distribution of genotypes in the population. In the upper line, P_i represents the distribution of phenotypes at the given time. Transformations T_A account for epigenetic phenomena, e.g. interactions with the environment, in-migration and out-migrations, individual development, etc., all of them affecting the distribution of phenotypes and producing a change in the distribution of genotypes during this generation. On the other hand transformations T_B account for the Mendelian principles that govern genetic inheritance and transforms a distribution of genotypes G_1' into another one G_2. Evolutionary computation endeavors concentrate on the study and assessment of many different ways the cycle depicted in 1 can be implemented. This evolutionary cycle implicitly assumes the existence of only one replicator: genes.

On the other hand what memetic algorithmicists should somehow investigate, if they were more faithful to the natural phenomena that inspired the methodology, is the implementation of a more general and complex dual evolutionary cycle where two replicators co-exist. This[3] is shown in 2.

[2] This graph is adapted from [18] page 114.
[3] Graph adapted from [18] page 186.

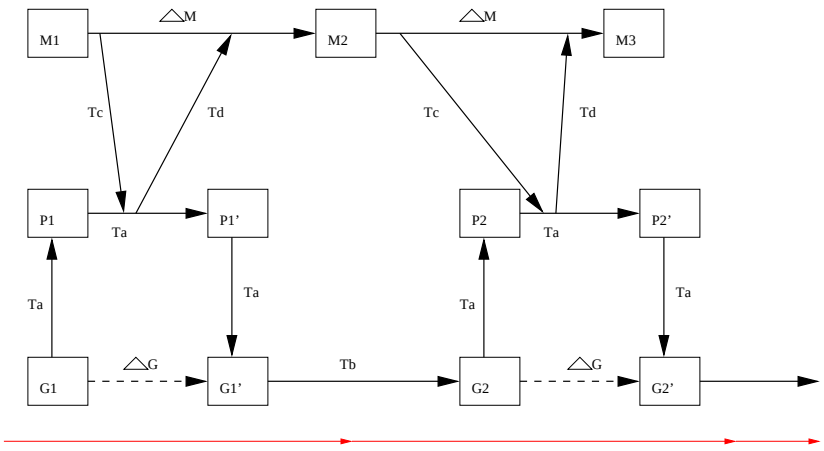


Fig. 2. *Coevolutionary memetic-genetic cycle.*

In the context of memetic algorithms, memes represent instructions to self-improve. That is, memes specify sets of rules, programs, heuristics, strategies, behaviors, etc, individuals can use in order to improve their own fitnesses under certain metric.

As we mentioned earlier, the fundamental difference between the later graph and the former resides in the fact that graph 2 reflects a coevolutionary system where two replicators of a different nature interact. Moreover the interactions between genes and memes are indirect and mediated by the common carrier of both: individuals. A truly memetic system should not be confused with other coevolutionary approaches where different "species", sub-populations or just different individuals interact by ways of a combination of cooperation, competition, parasitism, symbiosis, etc. In coevolutionary approaches like those described by [21],[38],[39],[40],[41],[43] and others, only Mendelian transformations are allowed and sometimes in-migration and out-migration operators are also included. In a memetic system, memes can potentially change and evolve using rules and time scales other than the traditional genetic ones. In the words of Feldman and Cavalli-Sforza[7] memetic evolution is driven by:

> ...the balance of several evolutionary forces: (1) *mutation*, which is both purposive (innovation) and random (copy error); (2) *transmission*, which is not as inert as in biology [i.e., conveyance may also be horizontal and oblique]; (3) *cultural drift* (sampling fluctuations); (4) *cultural selection)* (decisions by individuals); and (5) *natural selection* (the consequences at the level of Darwinian fitness) ...

In graph 2 we have the same set of transformations as before between genes and phenotypes, but also meme-phenotypes and memes-memes interactions are shown. There are mainly two transformations for memes that are depicted, T_C and T_D. Transformations T_C represents the various ways in which "cultural"

instructions can re-shape phenotypes distributions, e.g. individuals learn, adopt or imitate certain memes or modify other memes. T_D, on the other hand, reflects the changes in memetic distribution that can be expected from changes in phenotypic distributions, e.g. those attributed to teaching, preaching, etc.

Memetic Algorithms, as they were used so far, failed completely (or almost completely) to implement this dual inheritance system to any degree. Consequently, it is not surprising that researchers hesitate to call a GA (or other evolutionary approach) that uses local search a memetic algorithm.

3 Two Show Case Applications

In this paper we intend to investigate the ability of a memetic algorithm (in the Dawkins sense) to discover concurrently the behaviors that are appropriate to use as local searchers and ultimately improve the solutions to the problem the MA needs to solve. To illustrate the potentialities of this approach we will first revisit some previous work we have done on these lines using the *Protein Structure Prediction Problem* and then, as a second show case and experimental test bed, we will use *NK-Landscapes*.

3.1 The Protein Structure Prediction Case

The *primary structure* of a protein is defined by its amino acid sequence. When this linear arrangement of monomers folds in space, the protein adopts its *native state*. This tridimensional structure, also known as tertiary structure, determines the protein's functionality. It's assumed that the amino acid sequence completely determines the tertiary structure.

Proteins are too complex to allow exact modeling with today's computational power, moreover, scientist disagree on what constitute the right model that must be used. To study optimization techniques for the PFP, reduced models are used.

The most simple models used to represent proteins are based on lattices (of 2 and 3 dimensions), where each site in the lattice is filled by at most one amino acid; the correspondence between amino acids and positions is called *embedding* of the protein, and when the embedding is injective, it is called *self avoiding*. Rectangular and triangular lattices have been used (See figures 3(a) and 3(b)).

The mostly used lattice model is the so called HP Model or Dill's model [16]. In this model the twenty amino acids are classified in just two kinds, **H**ydrophobics and hydrophilics (or **P**olars), so a protein w could be thought as a word in a binary alphabet, $w \in \{H, P\}^+$. However an over simplification this might seem, it is known that the major driving force for protein folding is the hydrophobic force [17]. The energy function takes into account the interactions between topological neighbors of type H (these interactions are called bonds or contacts). Topological neighbors are pairs of adjacent amino acids in the lattice that are not consecutive in the sequence. Different energy functions could be obtained considering alternative amino acid interactions.

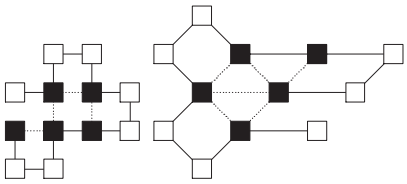


Fig. 3. Embedding in a Rectangular Lattice (a) and a Triangular Lattice (b). Black squares represent H's and white squares represent P's. Dotted lines between two amino acids represent a contact.

The problem and several variations were shown NP-Complete by [2], [12] and [4] among others.

Figure 4 taken from our earlier work[23] is a graphical representation of the kind of Memetic Algorithms we intend to construct.

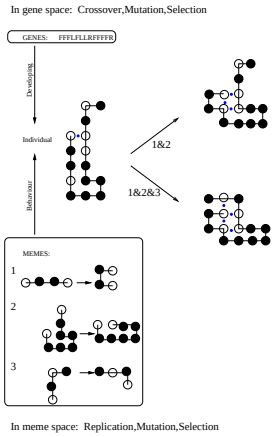


Fig. 4.

In the figure, a protein in the HP model and its structure is presented. The quality of the structure is measured by the number of hydrophobic topological neighbors (marked with a small dot) as prescribed by the HP-model. In our MAs, individuals are encoded by a pair $(chromosome, meme)$, where a chromosome represents a structure for the protein instance that is being solved and the meme represents a behaviour, or more precisely, the pattern of local search to use. In figure 4 the "raw" individual (i.e. the individual that did not use its memes to improve itself) has a fitness of 1 (represented by the blue dot). During local search individuals can select one or more memes (from a memepool) to optimize themselves. Individuals can also imitate others individuals behaviours by

copying their memes[5]. The memes themselves can evolve by an independent evolutionary process or any other conceivable creative process. When generations goes by, individuals will come up with a set of behavioural or local search memes that reshape the structures encoded by their chromosomes. In the figure two alternative manifestations of the original individual have better fitness that the raw genetically-encoded protein structure (fitness 3 and 4 respectively).

In [31],[26] and [42] we encoded the local search memes as rules of the form $initialPattern \rightarrow endPattern$. As it should be obvious the rule set cannot, in general, be constructed by hand because it is very difficult to deduce which local interactions lead to global behavior, so some kind of automatic procedure has to be used. In those papers a GA was used to evolve the rules. In figures 5(a) and 5(b) (reproduced from [31]) we show the two types of rules used.

In figure 5(a) the main aim is to discover highly designable substructures. Moreover, the use of rules with a length of at least 6 allows the discovery of substructures supporting folding patterns [29],[27].

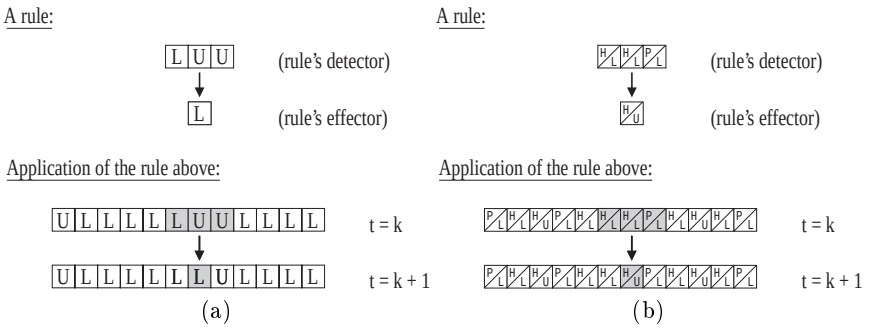


Fig. 5. A CA Rule in the first approach(a) and the second approach(b).

The second approach (shown in figure 5(b)) differs from the first in that a meme models not only substructures but also the sequence-structure association. The experimental details and a more complete account of this approach for *PSP* cannot be done here for reasons of space. However we want to emphasize that our algorithm was able to discover folding patterns, that is, secondary structure sub-units. For details please refer to the references mentioned before

3.2 The NK-Landscapes Case

NK-Landscapes can be defined as:

(n,k) *Landscapes*
Instance: Two integers $n,k|0<n,0\leq k\leq n-1$
and a $n\times 2^{k+1}$ matrix E with elements
sampled randomly from the $U(0,n)$ distribution.
E represents the epistatic interactions of k bits.
Solution: A binary string S, such that $|S|=n$.
Measure: $fitness(S)=\frac{1}{n}*\sum_{i=1}^{i=n}f_i(S_i,S_{i_1},\ldots,S_{i_k})$
with $f_i(\cdot)$ an entry into E, S_i the value of string S
at position i and S_{i_j} is the value of string S
at the $j-th$ neighbour of bit i.
The neighbours, not necessarily adjacent, j of bit i are part of the input.

NK-Landscapes are particularly useful to understand the dynamics of evolutionary search[45] (particularly MAs) as they can be tunned to represent low or high epistasis regimes (i.e. low or high k values respectively) with the extreme of an uncorrelated random fitness landscape for the case of $k=n-1$. Moreover, the optimization version of this problem can be solved in polynomial time by dynamic programming if the neighborhood structure used is that of *adjacent neighbours* or can be NP-Hard if the structure used is that of *random neighbours*[49].

NK-Landscapes have been the subject of intensive and varied studies. In [33] Kaufmann et.al. explore a phase change in search when a parameter τ of a local search algorithm reaches a certain critical value on some *NK-Landscape* problems . In their paper the authors show experimentally that the quality of the search follows an $s-$shape curve when plotted against τ making evident a change in phase. M. Oates et.al. in [37] showed performance profiles for evolutionary search based algorithms where phase changes were also present. Krasnogor and Smith in [30] and Krasnogor in [24] showed the existence of the "solvability" phase transition for GAs (instead than LS) and demonstrated that a self-adapting MA can learn the adequate set of parameters to use. Merz [34] devotes at least one whole chapter of his Ph.D. dissertation to the development of efficient Memetic Algorithms for this problem (we will return to his MAs later on). With a different target as the object of research O.Sharpe in [46] performs some analysis on the parameter space of evolutionary search strategies for NK landscapes.

As the *NK-Landscapes* represent a rich problem they are an ideal test case for our purposes. We will describe the behavior of our Self-Generating Memetic Algorithms in 4 different regimes: *low epistasis and poly-time solvable*, *high epistasis and poly-time solvable*, *low epistasis and NP-hard* and *high epistasis and NP-hard*.

In [24] and in previous sections we argued briefly about the need to creatively adapt every aspect of the local searchers. In this part of the paper we will focus only on the self-generation of the move operator itself as a proof of concept. The other aspects are actively being investigated.

Following the terminology of [24] our MAs will use self-adapting helpers, i.e., memes for which the code that represent their behaviour is subject to changes (in

this case artificial evolution). Moreover, the MA is a $D = 4$ Memetic Algorithm which implies that local search occurs as an independent process of Mutation and Crossover[4]. The MA will be composed of the two simultaneous processes as depicted in 2. Individuals in the MA population will be composed of genetic and memetic material. The genetic material will basically represent a solution to *NK-Landscapes* problems (i.e. a bit string) while the memetic part will represent "mental constructs" to optimize the *NK-Landscape* string. As such we will be evolving individuals whose goal is to self-optimize.

After [31],[26],[23] and building on knowledge gained through the experimental design for the results reported in [24] the local searchers are going to be represented as rules of the form $initialPattern \rightarrow endPattern$. As we mention earlier the rules were associated to local search patterns for the *PSP* and the same will be the case here.

The Self-Generating Memetic Algorithm The pseudocode in figure 6 depicts the algorithm use to solve the *NK-Landscapes*.

```
Memetic_Algorithm():
Begin
  t = 0;
  /* We put the evolutionary clock (generations), to null */
  Randomly generate an initial population P(t);
  Repeat Until ( Termination Criterion Fulfilled ) Do
    Variate individuals in M(t);
    Improve_by_local_search( M(t));
    Compute the fitness f(p) ∀p ∈ M(t) ;
    Generate P(t+1) selecting some individuals from P(t) and M(t);
    t = t + 1;
  Od
  Return best p ∈ P(t − 1);
End.
```

Fig. 6. The memetic algorithm employed.

The initial population in P is created at random. As mentioned before, each individual is composed of genetic material in the form of a bit string (B). The bit string represent the solution to the NK instance being solved. The memetic material is of the form $* \rightarrow S$ where the $*$ symbol matches any bit in the solution string and S is another bit string. The meaning of memes will be explained later on. The only variation mechanism is bitwise mutation (applied with probability

[4] See the Memetic Algorithms taxonomy in [24].

0.05) to the chromosomes. The replacement strategy was a $(20, 50)$. There is no genetic crossover but the SIM mechanism, as described in [30], was used to transfer memes between individuals. Memetic mutation occurs with an innovation rate[24] of 0.2. A meme can be mutated (with equal probability) in three ways: either a random bit is inserted in a random position, or a bit is deleted from a random position, or a bit is flipped at a random position. The length of memes cannot decrease below 0 nor increase beyond $3 * k$ for an $(n, k)-$problem.

The Local Search Procedure As mentioned before, a meme is represented as a rule of the form $* \rightarrow S$. During the local search stage this meme is interpreted as follows:

Every bit in the chromosome B has the opportunity to be improved by steepest hill-climbing. In general *NK-Landscapes* are epistatic problems so flipping only one bit at a time cannot produce reasonable improvements except of course in problems with very low k. To accommodate that fact, for each bit, one wants to optimize the value of that bit and that of $|S|$ other bits. A sample of size n is taken from all the $(|S| + 1)!$ possible binary strings. Based on the content of S, these sample strings serve as bits template with which the original chromosome B will be modified. If $|S| = 0$ then only B_i (the i^{th} bit of B) will be subjected to hill-climbing. On the other hand, if $|S| > 0$ then the local searchers scans the bits of S one after the other. If the first bit of S is a 0, then the bit $B_{(i+1)}$ will be set accordingly to what one of the n samples template mandates. On the other hand, if B_i is a 1 then bit $B_{(i+r)\%n}$ will be set as what one of the n samples template mandates. Here r is a random number between 0 and $n - 1$. By distinguishing in S between ones and ceroes memes can reflect the adjacent neighbour or the random neighbour version of the NK-landscapes. The larger the size of S the more bits will be affected by the local search process. As an example consider the case where the rule is $* \rightarrow 0000$. This rules implies $S = 0000$. In this case, for every bit i in B we will produce a sample of size n out of the possible 2^5 binary strings. Each one of these samples will be used as a template to modify B, in this case as S is built out of all 0 a fully-adjacent neighbourhood is considered. Suppose $B = 101010101010111110$ and the bit to be optimized is the fourth bit. $B_4 = 0$ in the example and its four adjacent neighbours are $B_5 = 1$,$B_6 = 0$,$B_7 = 1$,$B_8 = 0$. If one of the n samples is 11111 then B will be set to $B' = 101111111010111110$ provided B' has better fitness than B. The process is repeated in every bit of B once for every sample in the sample set.

In our implementation, and because our experiments are meant only as a proof of concept, we did not use all the code optimization described in [34] nor we use an exhaustive $k - opt$ or $Lin - kernighan$ heuristic as Merz employed. Certainly his recommendations on how to improve the efficiency of the code (in particular those related to the fitness updates) will be needed if larger problems are going to be studied. The evolved memes induce a variable-sampled $k - opt$ local searcher. We say variable as k varies with the size of S and it can be as small as 0 or as large as $3 * k$. It is sampled as we do not exhaustively explore

all the 2^{k+1} possible ways of settings the bits in a chromosome but rather take a reduced sample of size n.

4 Experiments

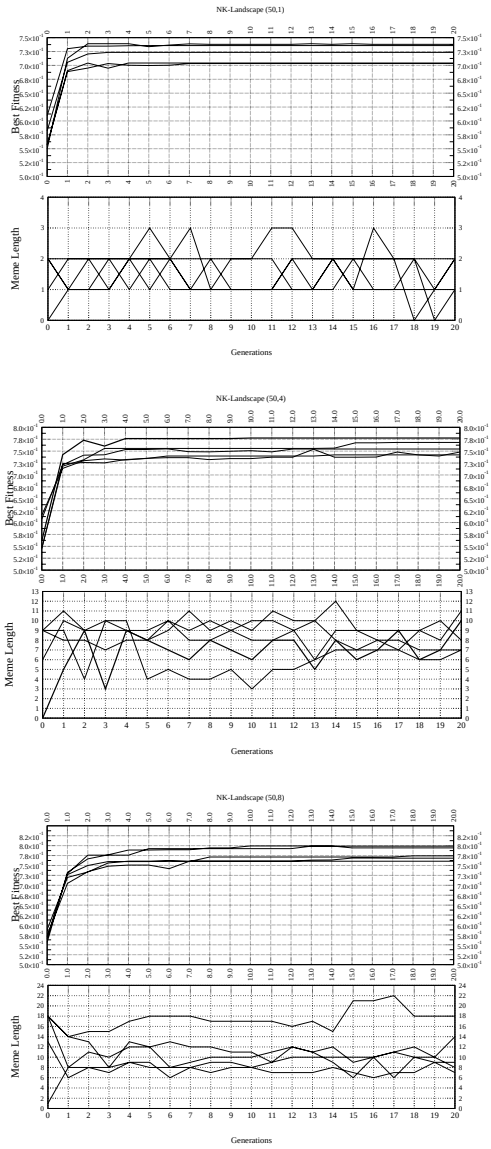
In previous sections we described our self-generating MAs. What sort of behaviors we expect to see emerging? Four different scenarios needs to be analysed: low epistasis-poly-time solvable, high epistasis-poly-time solvable, low epistasis-NP-hard and high epistasis-NP-hard landscapes. The level of epistasis is controlled by the n and k. The closer k is to 0 the more negligible the epistatic interactions among loci. If k grows up to $n-1$ then the induced problems is a random field. The transition between polynomial time solvability to NP-hardness depends on the type of neighborhood used as it was explained before. We should expect the emergence of short strings (i.e. $|S|$ not too big) for the low epistasis regimes while longer strings will be favored in high epistasis cases. We should be able to compare the length of the evolved local searcher with the k of the problem that is being solved, that is we expect to see memes emerging with lengths close to k. We should probably also see distinct patterns of activity for the different problem regimes. The range of problems we experimented with are:

- low epistasis, poly-time solvable: $(50, 1), (50, 4)$ with adjacent neighbours.
- high epistasis, poly-time solvable: $(50, 8), (50, 10), (50, 12), (50, 14)$ with adjacent neighbours
- low epistasis, NP-hard: $(50, 1), (50, 4)$ with random neighbours.
- high epistasis, NP-hard: $(50, 8), (50, 10), (50, 12), (50, 14)$ with random neighbours.

4.1 Results

In the following graphs we plot the evolution of the length of the meme associated with the fittest individual as a function of time and the evolution of fitness. For clarity, just 5 runs are depicted.

Low epistasis, poly-time solvable: In figures 7(a) and 7(b) we can observe the behaviour of the system. For the case $n = 50, k = 1$ the main activity occurs at the early generations (before generation 4). After that point the system becomes trapped in a local (possible global) optimum. The length of the memes evolved oscillates between 1 and 2. As the allowed length are restricted to be in the range $[0, 3 * k]$, the expected length of memes is 1.5. It is evident that the problem is solved before any creative learning can take place. When the Self-Generating MA is confronted with problem $n = 50, k = 4$ (a value of k just before the phase transitions mentioned in previous sections) the length of the meme in the best run oscillates between a minimum value of 3 (after generation 1) and a maximum of 10 for the run marked with a thick line (the best run). In this case the expected length (if a purely random rule was chosen) for a meme



(c)

Fig. 7. *NK(50,1) in (a), NK(50,4) in (b) and NK(50,8) in (c). Adjacent neighbours.*

is 6 which is the most frequently visited value. For these, the simplest possible *NK-Landscape* regimes, it does not seem to be of benefit to learn any specific meme, but rather, a random rule seems to suffice.

High epistasis, poly-time solvable: In figure 7(c) we can see the system's behaviour for a value of k after the phase transition mentioned in [33] and [24]. In this case there is effective evolutionary activity during the whole period depicted and we can see clearly that the length of the meme employed by the most successful individual converges towards the value of k(in this case 8). If a purely random rule was used the expected length would have been 12. The case shown in figure 8(a) is even clearer. All but one of the runs converge towards a meme length almost identical to $k = 10$, except for one that is very close to the expected length of 15.

The same trends can be seen in figures 8(b) and 8(c) where meme lengths converge to values around to $k = 12$ and $k = 14$ respectively. It is interesting to note that although the values are very close to our predictions they do not remain at a fixed value but rather oscillates. This is a very intriguing behaviour as it resembles the variable-neighborhood nature of Lin-Kernighan, the most successful local search strategy for *NK-Landscapes* and other combinatorial problems. It will be interesting to investigate on the range of values that the Memetic Algorithms presented in [34] (which uses $K - opt$ and Lin-Kernighan) effectively employs; we speculate that the range of changes, i.e. the number of bits modified in each iteration of LS, will be close to the epistatic parameter of the problem instance.

Low epistasis, NP-hard: With the graph in figure 9(a) we start to investigate the behaviour of the Self-Generating MA on the NP-Hard regime (i.e. the random neighborhood model). The picture in figure 9(a) is similar to the adjacent neighborhoods version of 7(a) except that oscillations are more frequent in the former. Comparisons of 9(b) and 7(b) reveal very similar trends.

High epistasis, NP-hard: The experiments with $(n = 50, k = 8)$ under the random neighbours model reveal marked differences with the consecutive neighbour model (see figures 9(c) and 7(c) respectively). While in the later all the runs converged toward a meme length very close to k, the random model shows a richer dynamics. Meme length were divided into 3 groups. In one group, the emerged meme length were very close to the value of k, 8 in this case. The other two groups either continually increase the size of the memes or decreased it. Two of the most successful runs are identified with a cross or circle and each belong to a different group. Interestingly enough, the run that converges first to the local optimum is the one that uses very short memes, in contrast, the one that uses memes length equivalent to a value of k shows a continued improvement. It is important to note that none of the evolved memes converged towards the expected length of 12.

Figure 10(a) seems to reveal a similar 3-grouped pattern.

The runs that correspond to instances of $(n = 50, k = 12)$ differ notably from previous ones.

The meme length seems to be converging towards a value well below the expected length of 18 and even the epistatic value $k = 12$ for these problems. However, between generation 34 and 68 the meme lengths oscillates very close to $k = 12$ values.

The next figure, 10(c), presents similar features as that of 10(b) but two clusters appear, one that suggest length around the value of k and another with length values of 6.

From the analysis of the previous graphs we can see that our expectations, namely, that memes of length proportional to k will arise were confirmed. However, other interesting features are evident. There are clear differences between memes that are evolved to solve the poly-time solvable cases and the NP-hard cases. In the first case all memes length and for $k > 4$ converged toward values in the proximity of k. However, for the random neighborhood model and for high epistasis ($k > 4$) problems, the runs were clustered mainly around either meme lengths of values close to k or to lengths around 6 (regardless the value of k). This is indeed a very interesting behaviour that deserves further studies as values of k in the range $[4, 5, 6]$ are *on the edge* of the phase transitions described in [33],[24] and [30], that is, between 4,5 or 6 bits were the optimum number of bits that need to be considered to boost the efficiency of the search. Moreover, in the case of the NP-hard random neighbourhood with $k = 8$ three clusters are noted; we speculate that problems in this range are on the so called "edge of caos" where emergent behaviours are more likely to occur[9],[45].

5 Conclusions

In this paper we argued that the label "Memetic Algorithm" can be contested as long as truly memetic systems are not developed. We argued in favor in changing the biological metaphor by introducing a second replicator, i.e. memes, into Memetic Algorithms and by doing this putting back the memetic aspect into play. From the optimization point of view there are obvious advantages on doing so. MAs that follow the Dawking meaning can self-generate the local searchers they need to use in order to improve solutions for a given problem. Moreover, they will be able to adapt to each problem, to every instance within a class of problem and to every stage of the search.

We presented insights on the self-generation of folding patterns for the Protein Structure Prediction Problem and also we explored the use of self-generating MAs for *NK-Landscapes*. We found that the self-generating memes can adapt to the kind of instance where they are being used for a wide range of problems regimes.

It is our hope that researchers confronted with new problems for which there are not "silver bullet" local search heuristics (like is the case for TSP and Graph

Partitioning where $K-$opt and Lin-Kernighan are known to be extremely efficient) with which to hybridize a Memetic Algorithm will try the obvious: the Dawkins method of self-generation of local search behaviors, that is, the use of memes.

6 Acknowledgments

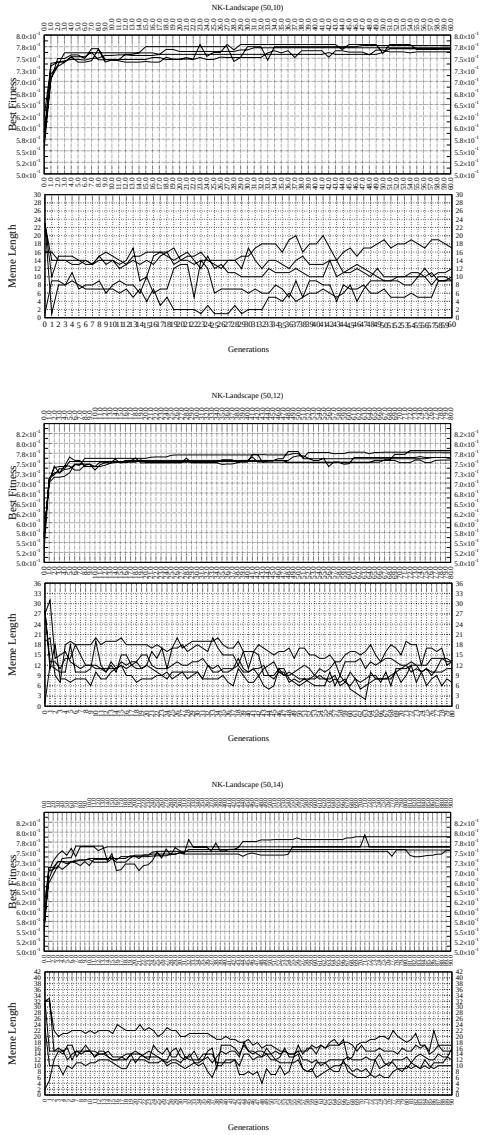
Parts of this work were developed while Dr. Krasnogor was atending the XXIII Madeira Math Encounters in the Universidade Da Madeira, Centro de Ciencias Matematicas, Isla Madeira.

References

1. *Journal of Memetics - Evolutionary Models of Information Transmission.*
2. Jonathan Atkins and William E. Hart. On the intractability of protein folding with a finite alphabet. *Algorithmica*, pages 279–294, 1999.
3. Thomas Back. Self adaptation in genetic algorithms. In F. Varela and P. Bourgine, editors, *Towards a Practice on Autonomous Systems: Proceedings of the First European Conference on Artificial Life*, pages 263–271. MIT Press, 1992.
4. B. Berner and T. Leighton. Protein folding in the hydrophobic-hydrophilic (hp) model is np-complete. *Journal of Computational Biology*, 5(1):27–40, 1998.
5. S. Blackmore. *The Meme Machine.* Oxford University Press, 1999.
6. R.D. Carr, W.E. Hart, N. Krasnogor, J.D. Hirst, E.K. Burke, and J.E Smith. Alignment of protein structures with a memetic evolutionary algorithm. In *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2002.* Morgan Kaufmann, 2002.
7. L.L. Cavalli-Sforza and M.W. Feldman. *Cultural Transmission and Evolution: A Quatitative Approach.* Princeton University Press, Princeton, NJ., 1981.
8. F.T. Cloak. Is a cultural ethology possible. *Human Ecology*, 3:161–182, 1975.
9. P. Coveney and R. Highfield. *Frontiers of Complexity, the search for order in a chaotic world.* faber and faber (ff), 1995.
10. P. Cowling, G. Kendall, and E. Soubeiga. A hyperheuristic approach to scheduling a sales summit. In E. Burke and W. Erben editors, editors, *Theory of Automated Timetabling PATAT 2000, Springer Lecture Notes in Computer Science,*, pages 176–190. Springer, 2001.
11. P. Cowling, G. Kendall, and E. Soubeiga. Hyperheuristics: a tool for rapid prototyping in scheduling and optimisation. In *Proceedings of the 2nd European Conference on Evolutionary Computation, EvoCop 2002. Lecture notes in computer science.* Springer, 2002.
12. P. Crescenzi, D. Goldman, C. Papadimitriou, A. Piccolboni, and M. Yannakakis. On the complexity of protein folding. In *Proceedings of the Second International Conference on Computational Molecular Biology ReComb 98*, 1998.
13. L. Davis. Adapting operator probabilities in genetic algorithms. In J.J. Grefenstette, editor, *Proceedings of the Third International Conference on Genetic Algorithms.* Morgan Kaufmann, 1989.
14. R. Dawkins. *The Selfish Gene.* Oxford University Press, New York, 1976.
15. R. Dawkins. The extended phenotype. 1982.

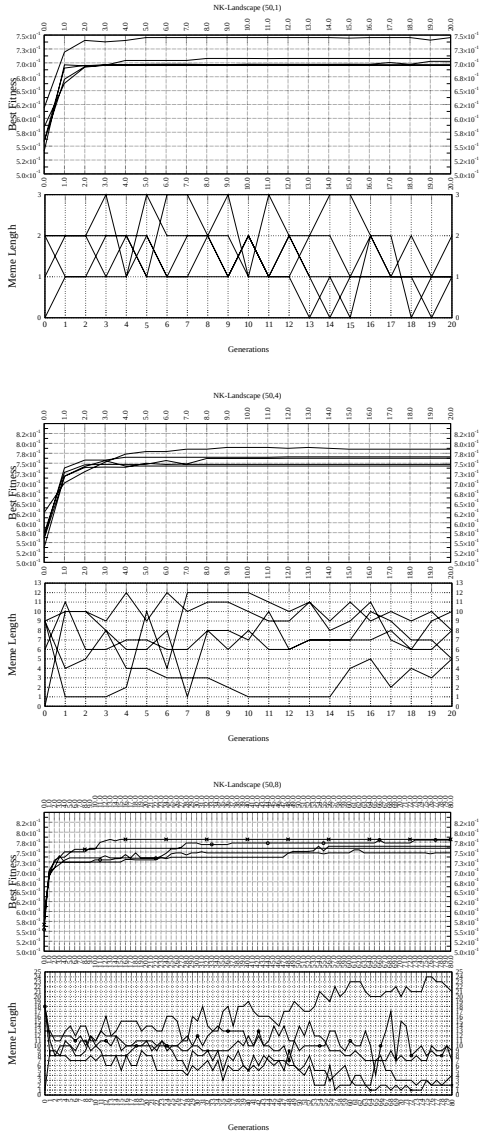
16. Ken A. Dill. *Biochemistry*, 24:1501, 1985.
17. Ken A. Dill. *Biochemistry*, 29:7133, 1990.
18. W.H. Durham. *Coevolution: Genes, Culture and Human Diversity*. Stanford University Press, 1991.
19. David E. Goldberg. *The Design of Innovation: Lessons from and for Competent Genetic Algorithms*. Kluwer Academic Publishers, 2002.
20. W. E. Hart. Adaptive global optimization with local search. *Ph.D. Thesis, University of California, San Diego*, 1994.
21. D. Hillis. Co-evolving parasites improve simulated evolution as an optimization procedure. In *Artificial Life II: Proc. of the 2nd Conf. on Artificial Life*, 1992.
22. R. Hinterding, Z. Michalewicz, and T.C. Peachey. Self-adaptive genetic algorithms for numeric functions. In *Proceedings of Parallel Problem Solving from Natur - PPSN IV*, 1996.
23. N. Krasnogor. Co-evolution of genes and memes in memetic algorithms. In A.S. Wu, editor, *Proceedings of the 1999 Genetic And Evolutionary Computation Conference Workshop Program*, 1999.
24. N. Krasnogor. *Studies on the Theory and Design Space of Memetic Algorithms. (Submitted)*. Ph.D. Thesis, Faculty of Engineering, Computer Science and Mathematics. University of the West of England, 2002.
25. N. Krasnogor, B.P. Blackburne, E.K. Burke, and J.D Hirst. Multimeme algorithms for protein structure prediction. In *Proceedings of Parallel Problem Solving From Nature (PPSN VII). LNCS*. Springer-Verlag, 2002.
26. N. Krasnogor, E. de la Cananl, D.A. Pelta, D.H Marcos, and W.A. Risi. Encoding and crossover mismatch in a molecular design problem. In P. Bentley, editor, *AID98: Proceedings of the Workshop on Artificial Intelligence in Design 1998*, 1998.
27. N. Krasnogor, P.E. Martinez Lopez, P. Mocciola, and D. Pelta. Protein folding meets functional programming. In *Poster Proceedings of the First International Conference on Computational Molecular Biology. Also, in Proceedings of the International Conference on Functional Programming (ACM Press)*, 1997.
28. N. Krasnogor and D. Pelta. Fuzzy memes in multimeme algorithms: a fuzzy-evolutionary hybrid. *Book chapter in Fuzzy Sets based Heuristics for Optimization*, 2002.
29. N. Krasnogor, D. Pelta, P. Martinez Lopez, and E. de la Canal. Enhanced evolutionary search of foldings using parsed proteins. In *Proceedings of Simposio de Investigacion Operativa 97, Argentine Operational Research Symposium*, 1997.
30. N. Krasnogor and J.E. Smith. Emergence of profitable search strategies based on a simple inheritance mechanism. In *Proceedings of the 2001 Genetic and Evolutionary Computation Conference*. Morgan Kaufmann, 2001.
31. Natalio Krasnogor, David Pelta, Daniel H. Marcos, and Walter A. Risi. Protein structure prediction as a complex adaptive system. In *Proceedings of Frontiers in Evolutionary Algorithms 1998*, 1998.
32. M.W.S. Land. Evolutionary algorithms with local search for combinatorial optimization. *Ph.D. Thesis, University of California, San Diego*, 1998.
33. W.G. Macready, A.G. Siapas, and S.A. Kauffman. Criticality and parallelism in combinatorial optimization. *Science*, 261:56–58, 1996.
34. P. Merz. *Memetic Algorithms for Combinatorial Optimization Problems:Fitness Landscapes and Effecitve Search Strategies*. Ph.D. Thesis, Parallel Systems Research Group. Department of Electrical Engineering and Computer Science. University of Siegen., 2000.

35. P. Merz and B. Freisleben. A comparison of memetic algorithms, tabu search, and ant colonies for the quadratic assignment problem. In *Proceedings of the 1999 International Congress of Evolutionary Computation (CEC'99), Washington DC, USA*, 1999.
36. P. Merz and B. Friesleben. A genetic local search approach to the quadratic assignment problem. In editor T. Back, editor, *Proceedings of the 7th International Coonference on Genetic Algorithms*, pages 465–472. Morgan Kauffman publisher, 1997.
37. M.J. Oates, D.W. Corne, and R.J. Loader. Tri-phase profile of evolutionary search on uni- and multi-modal search spaces. *Proceedings of the Congress on Evolutionary Computation (CEC2000)*, 1:357–364, 2000.
38. B. Olsson. A host-parasite genetic algorithm for asymmetric tasks. In C. Nédellec and C. Rouveirol, editors, *Machine Learning: ECML-98 (Proceedings of the 9th European Conference on Machine Learning)*, pages 346–351. Springer-Verlag, 1998.
39. B. Olsson. *Algorithms for Coevolution of Solutions and Fitness Cases in Asymmetric Problem Domains*. PhD thesis, University of Exeter, 1999.
40. B. Olsson. Handling asymmetric problems with host-parasite algorithms, 1999.
41. J. Paredis. Chapter c7.4: Coevolutionary algorithms. In T. Back, D.B. Fogel, and Z. Michalewicz, editors, *Handbook of Evolutionary Computation*, page C7.4. IOP publishing Ltd and Oxford Unviersity Press, 1997.
42. David Pelta and Pablo Daniel Ona. Aplicacion de tecnicas evolutivas para el problema de plegado de proteinas. 1998. Licentiate Thesis, Departamento de Informatica, Facultad de Ciencias Exactas, Universidad Nacional de La Plata.
43. M.A. Potter and K.A. De Jong. A cooperative coevolutionary approach to function optimization. In *Parallel Problem Solving From Nature*, 1994.
44. P.J. Richerson and R. Boyd. A dual inheritance model of the human evolutionary process: I. basic postulates and a simple model. *Journal of Social and Biological Structures*, I:127–154, 1978.
45. Kauffman S.A. *The Origins of Order, Self Organization and Selection in Evolution*. Oxford University Press, 1993.
46. O. Sharpe. Introducing performance landscapes and a generic framework for evolutionary search algorithms. *Proceedings of the Congress on Evolutionary Computation (CEC2000)*, 1:341–348, 2000.
47. Jim Smith and T.C. Fogarty. Self adaptation of mutation rates in a steady state genetic algorithm. In *Proceedings of the Third IEEE International Conference on Evolutionary Computing*, pages 318–323. IEEE Press, 1996.
48. R.E. Smith and E. Smuda. Adaptively resizing populations: Algorithms, analysis and first results. *Complex Systems*, 1(9):47–72, 1995.
49. E.D. Weinberger and A. Fassberg. Np completness of kauffman's n-k model, a tuneably rugged fitness landscape. In *Santa Fe Institute Technical Reports*, 1996.



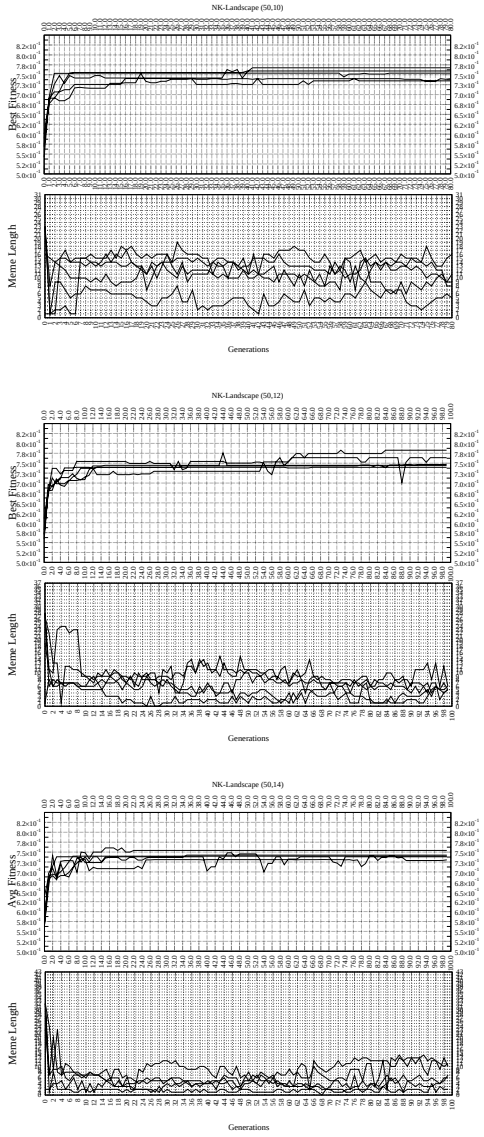
(c)

Fig. 8. *NK(50,10) in (a), NK(50,12) in (b) and NK(50,14) in (c). Adjacent neighbours.*



(c)

Fig. 9. *NK(50,1) in (a), NK(50,4) in (b) and NK(50,8) in (c). Random neighbours.*



(c)

Fig. 10. *NK(50,10) in (a), NK(50,12) in (b) and NK(50,14) in (c). Random neighbours.*