J Comb Optim (2007) 14:9–33
DOI 10.1007/s10878-006-9036-2

An immune algorithm with stochastic aging and kullback entropy for the chromatic number problem

Vincenzo Cutello · Giuseppe Nicosia · Mario Pavone

Published online: 29 December 2006

Abstract We present a new Immune Algorithm, *IMMALG*, that incorporates a Stochastic Aging operator and a simple local search procedure to improve the overall performances in tackling the chromatic number problem (CNP) instances. We characterize the algorithm and set its parameters in terms of Kullback Entropy. Experiments will show that the IA we propose is very competitive with the state-of-art evolutionary algorithms.

Keywords Immune Algorithm · Information Gain · Graph coloring problem · Chromatic number problem · Combinatorial optimization

1 Introduction

In the last five years we have witnessed the establishment of an increasing number of algorithms, models and results in the field of Artificial Immune Systems (Dasgupta, 1998; De Castro and Timmis, 2002; Garrett, 2005). From an information processing point of view (Forrest and Hofmeyr, 2000) the Immune System (IS) can be seen as a problem learning and solving system. The *antigen* (Ag) is the computational problem to solve, the *antibody* (Ab) is the generated candidate solution. At the beginning of the *primary response* the problem-Ag is recognized by a poor candidate solution-Ab. At the end of the primary response the problem-Ag is defeated-solved by good

V. Cutello · G. Nicosia (✉) · M. Pavone
Department of Mathematics and Computer Science, University of Catania,
V.le A. Doria 6, 95125 Catania, Italy
e-mail: nicosia@dmi.unict.it

V. Cutello
e-mail: vctl@dmi.unict.it

M. Pavone
e-mail: mpavone@dmi.unict.it

candidate solutions-Ab. Consequently the primary response corresponds to a training phase while the *secondary response* is the testing phase where we will try to solve similar problems to the ones originally presented in the primary response (Nicosia et al., 2001a).

Recent studies show that when one tackles the Chromatic Number Problem (CNP), or Graph Coloring Problem, with evolutionary algorithms (EAs), the best results are often obtained by hybrid EAs with local search and specialized crossover (Galinier and Hao, 1999). In particular, the random crossover operator used in a standard genetic algorithm performs poorly for combinatorial optimization problems and, in general, the crossover operator must be designed carefully to identify important properties, building blocks, which must be transmitted from the parents population to the offspring population. Hence the design of a *good* crossover operator is crucial for the overall performance of the EAs. The drawback is that it might happen that recombining good individuals from different regions of the search space, having *different symmetries*, may produce poor offspring (Marino and Damper, 2000).

For this reason, we use an Immunological Algorithm (IA) to tackle the Chromatic Number Problem. IAs do not have a crossover operator, and the crucial task of designing an appropriate crossover operator is avoided at once.

Presented IA, called *IMMALG*, is based on human's clonal selection principle, and it uses a particular mutation operator and a local search strategy without having to incorporate specific domain knowledge, as proposed in Cutello et al. (2003).

The paper is structured as follows: Section 2 describes the chromatic number problem; in Section 3 we give the definitions and properties of the Kullback Entropy function; Section 4 presents the proposed immune algorithm for the chromatic number problem, that we call *IMMALG*, which is inspired by the human's clonal selection theory; moreover, in the same section is described how *IMMALG* uses the kullback entropy as a termination condition (Section 4.4); Section 5 and Section 6 detail the characteristic dynamics of the implemented Immune Algorithm; Section 7 presents the results obtained by *IMMALG* for the classical graph coloring benchmarks, and compares the proposed IA with some of the state-of-art evolutionary algorithms for the chromatic number problem; finally, concluding remarks are presented in Section 8.

2 The chromatic number problem

How many colours do we need to colour the countries of a map in such a way that adjacent countries are coloured differently? How many days have to be scheduled for committee meetings of a parliament if every committee intends to meet for one day and some members of parliament serve on several committees?

Partitioning a set of objects into classes according to certain rules is a fundamental process in mathematics. The theory of graph coloring deals with exactly this situation, in which the objects form the set of vertices and two vertices are joined whenever they are not allowed in the same class.

Formally, given an undirected graph $G = (V, E)$, with vertex set V and edge set E, a *proper coloring* of G is a map $f : V \to S \subseteq E$, such that $f(u) \neq f(v)$ whenever $\{u, v\} \in S$. The elements of the set S are called *colors* and, if $|S| = k$, G is said to be *k-colorable*; the minimum value of k represents the *chromatic number* of G, $\chi(G)$.

The problem of finding the minimum value for k, such that G is *k-colorable*, is called the *Chromatic Number Problem* (CNP), or in general *Graph Coloring Problem* (GCP). Such a problem is also a classical *constraint satisfaction problem* (CSP) (Tsang, 1993), where the vertices are the variables and the edges are the constraints.

There are two different approaches for tackling the graph coloring problem: assigning the colors to the vertices (*assignment approach*), or partitioning the set of vertices V into disjoint subsets, which represent the color classes (*partition approach*). In this last case, CNP is defined as the problem to partitioning V into a minimal number k of disjoint subsets $V_1, V_2, \ldots, V_k$ such that if $(u, v) \in E$ then $u \in V_i$, $v \in V_j$ and $i \neq j$ $(\forall i, j = 1, \ldots, k)$. Each color class forms an *Independent Set* of vertices, i.e. no two vertices are linked by an edge.

A fast algorithm to tackle the independent set problem is unlikely to exist, since Independent Set is a NP-Complete problem.

In turn, CNP is also NP-complete (Garey and Johnson, 1979), therefore exact solutions can be found for simple or medium instances (Mehrotra and Trick, 1996; Caramia and Dell'Olmo, 2001). Then, as described in Garey and Johnson (1979) and Jensen and Toft (1995), is possible to assert the following theorems:

1 3-*Colorable is NP-Complete problem.*

and more generalized:

2 *The Graph k-colorability problem is NP-Complete, for* $k > 2$.

The existence of the chromatic number follows from the well-ordering theorem of the set theory, and conversely, considering cardinals as special ordinals, the existence of the chromatic number easily implies the well-ordering theorem. However, even if it is not assumed that every set has a well-ordering, but maintains the property that every set has a cardinality, then the statement "any finite or infinite graph has a chromatic number" is equivalent to the *axiom of choice*, as proved in Galvin and Komjáth (1991).

Several real-world applications are directly influenced by the chromatic number problem, such as time tabling (de Werra, 1985), scheduling (Leighton, 1979), frequency assignment for use in the electromagnetic spectrum (Gamst, 1986), register allocation in compilers (Chow and Hennessy, 1990) and printed circuit board testing (Garey et al., 1976).

3 Kullback entropy

Shannon's entropy (Shannon, 1948) is a commonly used measure in information theory and it represents a good measure of randomness or uncertainty, where the entropy of a random variable is defined in terms of its probability distribution. Shannon's theory, *information* is represented by a numerically measurable quantity, using a probabilistic model. In this way, the solutions of a given problem can be formulated in terms of the obtained amount of information. Kullback and Leibler (Kullback, 1959) introduced a measure of information using two probability distributions of discrete random variable, called *relative information*, which found many applications in setting important theorems in information theory and statistics.

In this work we used such measures to evaluate the learning ability of the proposed *IMMALG*. In details, in this section first we give some definitions and properties relatively to the kullback entropy, and then we will introduce this entropic function as a metric to analyze the learning process.

3.1 Shannon entropy for diversity assessment

The function $f(\vec{x})$ maps the set Ω, the *problem domain*, i.e. the set of the feasible solutions, into the set Λ, which represents all possible values of the objective function (i.e. the function to optimize) that determines how good is the $\vec{x}$ candidate solution.

Let $\Gamma_i : \Lambda \to \Re$ be the *Influence Function*, such that Γ_i is a decreasing function of the distance to the i-th solution point. Examples of influence functions are: the parabolic function, the square wave function, or the Gaussian function:

$$\Gamma(r) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-r^2}{2\sigma^2}}$$

where the standard deviation of this function is chosen subjectively according to the required resolution.

The influence function of the i-th solution point, that is

$$\vec{y}_i^t \in \mathcal{P}^t$$

(with $|\mathcal{P}^t| = d$) is maximum at that point and decreases gradually as the distance from that point increases.

Density function. The density function in an arbitrary point $\vec{y} \in \Lambda$ can be obtained as follows

$$D(\vec{y}) = \sum_{i=1}^{N} \Gamma(r_{i\to\vec{y}}) \tag{1}$$

Here $r_{i\to\vec{y}}$ represents the distance of the point $\vec{y} \in \Lambda$ and the i-th solution point $\vec{y}_i^t \in \mathcal{PF}_{obs}^{*,t} \subseteq \Lambda$. Without loss of generality we consider an euclidean distance for simplicity.

Indifference region. Let the quantity $\mu(0 \leq \mu \leq 1)$ be a number specified by the user, which can be used to divide the k-dimensional objective space, Λ, into $1/(\mu^k)$ number of small grids. For simplicity, $1/\mu$ is taken to be an integer. Each grid refers to a square (hypercube in k-dimension), the *indifference region* $T_{\mu(q)}$, wherein any two solution points $\vec{y}_i$ and $\vec{y}_j$ within the region are considered similar to one another or that the designer is indifferent to such solutions.

The point q, located at any intersection of the k-grid lines in the objective space Λ, has coordinates $(q_1, q_2, \ldots, q_k)$.

Hence, one divides the set Λ into a $a_1 \times a_2$ grid of cells, where a_1 and a_2 are determined so that the size of each cell ij becomes less than or equal to the indifference region.

First, we compute the density function in each cell, D_{ij}, using Eq. (1); and then we determine a normalized density, $\rho_{i,j}$:

$$\rho_{ij} = \frac{D_{ij}}{\sum_{h_1=1}^{a_1} \sum_{h_2=1}^{a_2} D_{h_1 h_2}} \tag{2}$$

Now we have:

$$\sum_{h_1=1}^{a_1} \sum_{h_2=1}^{a_2} \rho_{h_1 h_2} = 1 \quad \rho_{h_1 h_2} \geq 0 \quad \forall h_1 h_2$$

The Shannon's definition of Entropy (Shannon, 1948) of such a distribution can then be defined as:

$$H = -\sum_{h_1=1}^{a_1} \sum_{h_2=1}^{a_2} \rho_{h_1 h_2} ln(\rho_{h_1 h_2})$$

In general, for a k-dimensional objective space, the feasible region is divided into $a_1 \times a_2 \times \ldots a_k$ cells and the Shannon Entropy is defined as:

$$H = -\sum_{h_1=1}^{a_1} \sum_{h_2=1}^{a_2} \cdots \sum_{h_k=1}^{a_k} \rho_{h_1 h_2 \ldots h_k} ln(\rho_{h_1 h_2 \ldots h_k}) \tag{3}$$

Finally, the Shannon Entropy of a solution point can be calculated by constructing a grid of cells and applying Eq. (3). As it can be easily verified, the entropy of a solution set with a relatively flat density hyper-surface is large while a set of unevenly distributed solution points generally yields a lower entropy. This leads to the following conclusion: a solution set with higher entropy is spread more evenly throughout the feasible region and provides a better *coverage of the space*.

In this work, the Shannon Entropy is used to measure the flatness of the *information distribution* provided by a set of solution points, and hence it is desirable to maximize it.

We do not consider the entropy H as a measure of the uncertainty in a stochastic event with a given probability distribution.

Now we have a metric for the goodness of the spread of points in a k-dimensional objective space.

3.2 Kullback-leibler information G

The Kullback-Leibler entropy (Kullback, 1959; Jaynes, 2003; Sivia, 1996) is perhaps the most frequently used information-theoretic "distance" measure from the viewpoint

of theory. The Kullback-Leibler entropy G is a functional of two PDFs $p(x)$ and $r(x)$,

$$G \equiv -\int dx \quad p(x) ln[p(x)/r(x)] \tag{4}$$

If $r(x)$ is a constant G becomes essentially the Shannon Entropy H. The Kullback-Leibler Information G has the following important property:

$$G \geq 0$$

The equality sign holds if and only if $p(x) = r(x)$.

The discrete form of Eq. (4) is

$$G = -\sum_{n} p(x_n) ln[p(x_n)/r(x_n)]$$

Now, lets consider the information distribution defined in Eq. (2) for a given optimization problem obtained by a single-objective optimization method at time step t:

$$\rho^{t}_{h_1 h_2 \ldots h_k};$$

analogously $\rho^{t_0}_{h_1 h_2 \ldots h_k}$ represents the information distribution of the candidate solution points at time step t_0, with $t_0 \ll t$. Hence the Kullback-Leibler Entropy is defined as follows:

$$G(t, t_0) = -\sum_{h_1=1}^{a_1} \sum_{h_2=1}^{a_2} \cdots \sum_{h_k=1}^{a_k} \rho^{t}_{h_1 h_2 \ldots h_k} ln(\rho^{t}_{h_1 h_2 \ldots h_k} / \rho^{t_0}_{h_1 h_2 \ldots h_k})$$

$G(t, t_0)$ is the quantity of information the system discovers during the convergence process of the single-objective optimization method.

The gain is the amount of information the system has already learned from the given optimization problem, *OP*, with respect to the "initial" (time step t_0) distribution function (e.g., for the evolutionary optimization algorithms, the initial distribution is the randomly generated initial population $P^{(t_0=0)}$). Once the solving (or learning or searching) process starts, the information gain G increases monotonically until it reaches a final steady state. This is consistent with the idea of a *maximum entropy principle* (Jaynes, 2003; Sivia, 1996) of the form

$$\frac{dG}{dt} \geq 0.$$

Since $\frac{dG}{dt} = 0$ when the solving process ends, one will use it as a termination condition for the optimization methods. Moreover, in general the information gain is a kind of entropy function useful both "on-line" and at "run-time" to understand the algorithms' behavior and to set the procedure's parameters.

Another useful definition is the following equation:

$$G(t, t-1) = -\sum_{h_1=1}^{a_1}\sum_{h_2=1}^{a_2}\cdots\sum_{h_k=1}^{a_k}\rho^t_{h_1h_2\ldots h_k} ln\big(\rho^t_{h_1h_2\ldots h_k}/\rho^{(t-1)}_{h_1h_2\ldots h_k}\big)$$

In some application domains it is important to compute the total amount of information between two adjacent time steps.

To analyze the learning process, we use the notion of *Kullback Entropy*, or *Kullback information* (also called *information gain*) (Nicosia and Cutello, 2002; Cutello et al., 2004a), an entropy function associated to the quantity of information the system discovers during the learning phase. To this end, we define the solution points distribution function $f^{(t)}_m$ as the ratio between the number, B^t_m, of solution points at time t with fitness function value m, (m is the distance from the antigen–problem) and the total number of B cells:

$$f^{(t)}_m = \frac{B^t_m}{\sum_{m=0}^{h} B^t_m} = \frac{B^t_m}{d}.$$

It follows that the kullback entropy can be defined as:

$$K(t, t_0) = \sum_m f^{(t)}_m \log\left(f^{(t)}_m/f^{(t_0)}_m\right).$$

4 The *IMMALG* algorithm

In this work we present a simplified model of the natural immune system which gives rise to an algorithm *IMMALG*, proposed in Cutello et al. (2003). We have only two entities: Ag and *B cells*. Ag is a set of variables that models the problem, and, B cells are defined as strings of integers of finite length $\ell = |V|$.

Input is the antigen–problem and output is basically the candidate solutions-B cells that solve–recognize the Ag.

By P^t_d we denote a population of d individuals, B cells, of length ℓ, which represent a subset of the space of feasible solutions of length ℓ, S^ℓ, obtained at time t. The initial population of B cells, i.e. the initial set P^0_d, is created randomly.

Like a classical clonal selection algorithm, *IMMALG* can be divided into three steps: *Interaction*, *Cloning Expansion*, and *Aging*.

4.1 Interaction phase

In the *Interaction phase* the population P^t_d is evaluated. For the CNP, the fitness function value $f(\vec{x}) = m$ indicates that there exists a m–coloring for G, that is, a partition of vertices $V = S_1 \cup S_2 \cup \cdots \cup S_m$ such that each $S_i \subseteq V$ is a subset of vertices which are pairwise not adjacent (i.e. each S_i is an *independent set*).

For the color assignment the vertices of the solution represented by a B cell are examined and assigned colors, following a deterministic scheme based on the order

in which the graph vertices are visited: vertices are examined according to the order given by the B cell and assigned the first color not assigned to adjacent vertices.

This method is very simple. In literature there are more complicated and effective methods (Galinier and Hao, 1999; Marino and Damper, 2000; Johnson and Trick, 1996). We do not use these methods because we want to investigate the learning and solving capabilities of *IMMALG*. In fact, the described IA does not use specific domain knowledge and does not make use of problem-dependent local searches. Thus, our IA could be improved simply including *ad hoc* local search and immunological operators using specific domain knowledge.

4.2 Cloning expansion phase

The *Cloning expansion phase* is composed of two steps: *cloning* and *hypermutation*. The cloning expansion events are modeled by a *cloning potential V* and a *mutation number M*, which depend upon on the fitness function value f. If we exclude all the adaptive mechanisms (Eiben et al., 1999) in EA's (e.g., adaptive mutation and adaptive crossover rates which are related to the fitness function values), the immune operators, contrary to standard evolutionary operators, depend upon the fitness function values (Leung et al., 2001).

The cloning potential determines the probability that a given B cell will be cloned, producing *dup* copies of it, where *dup* is a parameter. Such probability depends on the fitness function and it is represented by a truncated exponential, using the following equation:

$$V(f(\vec{x})) = 1 - \left(e^{-k(\ell - f(\vec{x}))}\right), \tag{5}$$

where the parameter k determines the sharpness of the potential. The cloning operator generates an intermediate population P_{Nc}^{clo}, where Nc is the number of the clones produced.

In Figs. 1 and 2 is showed the Eq. (5) for the classical CNP dimacs instances QUEEN6_6 and FLAT300_20_0. Several fitness function values were used, from χ chromatic number, or best known, to worst coloring (i.e. number of vertices), and varying k. From these figures, one can see how small values of k are betters when *IMMALG* tackles a CNP instance with a high number of vertices.

The mutation number is given by an inversely proportional function to the fitness value, and it has the shape of a hyperbola branch:

$$M(f(\vec{x})) = (1 - opt/f(\vec{x}))\beta, \tag{6}$$

where $opt \leq f(\vec{x}) \leq |V|$, with *opt* being the chromatic number χ, or the lowest value known; and, $\beta = c \cdot \ell$ models the shape of the curve.

This function indicates the number of swaps between vertices in $\vec{x}$. The mutation operator chooses randomly $M(f(\vec{x}))$ times two vertices i and j in $\vec{x}$ and then swaps them. The hypermutation function applied on population P_{Nc}^{clo} generates the population P_{Nc}^{hyp}. The cell receptor mutation mechanism is modeled by the mutation number

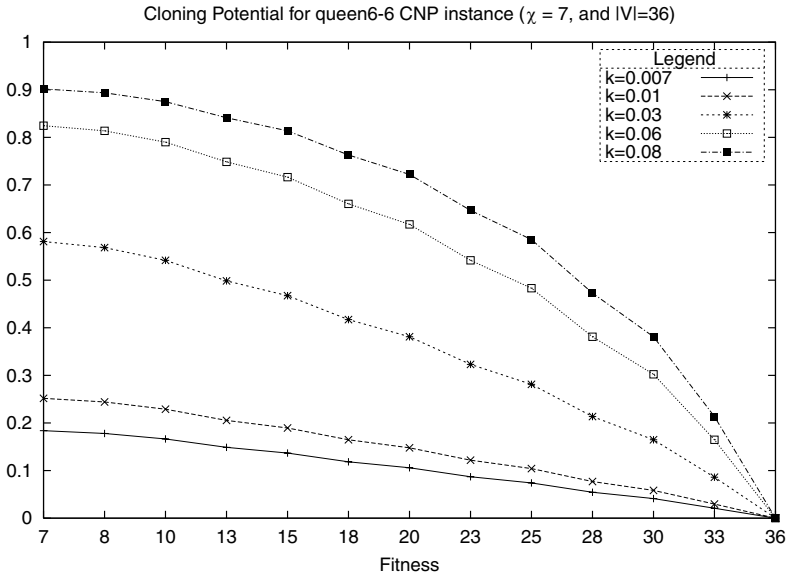

Fig. 1 Cloning potential versus fitness function value, for the CNP instance QUEEN6_6

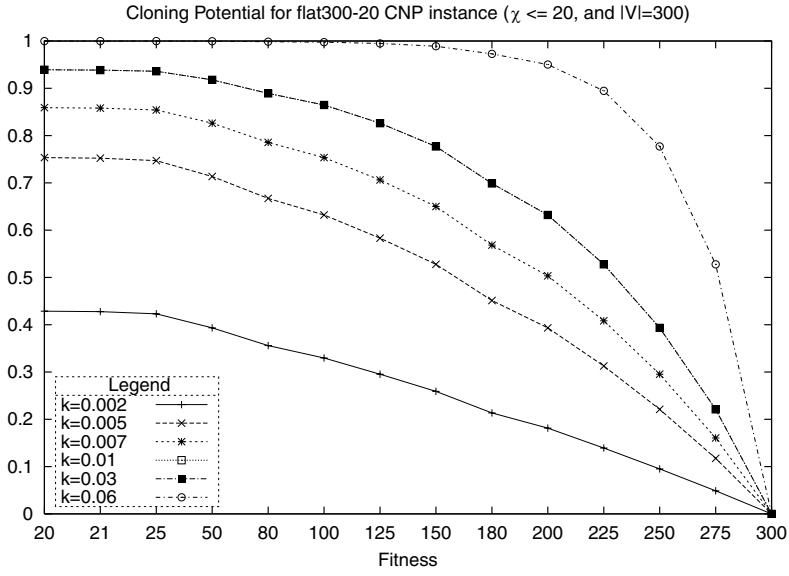

Fig. 2 Cloning potential versus fitness function value, for the CNP instance FLAT300_20_0

M, obtained by Eq. (6). The cloning expansion phase triggers the growth of a new population of high-value B cells centered around a higher fitness function value.

Figures 3 and 4 show the mutation number obtained by Eq. (6), at different fitness function qualities, from best to worst. These experiments were obtained varying c in $\{0.1, 0.3, 0.5, 0.8, 1.0\}$. In general, for the results obtained by *IMMALG* (see

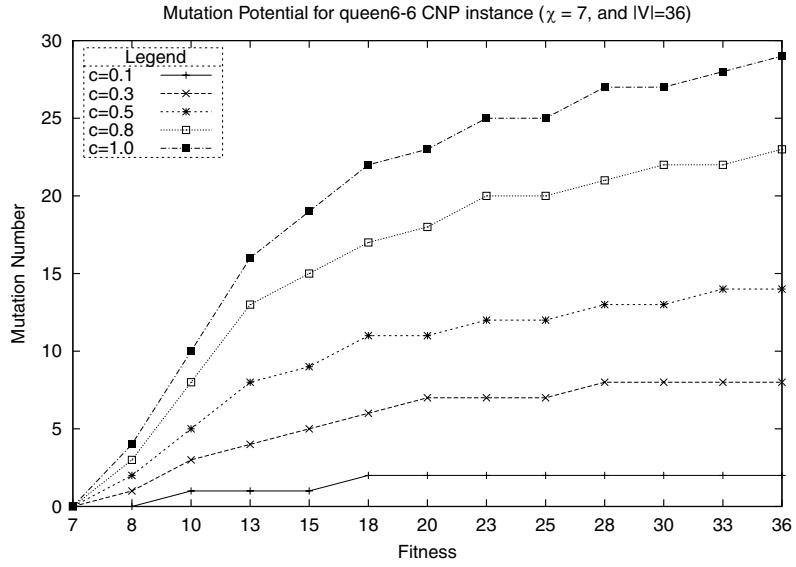

Fig. 3 Mutation potential obtained by Eq. (6) varying $c \in \{0.1, 0.3, 0.5, 0.8, 1.0\}$, for the CNP instance QUEEN6_6

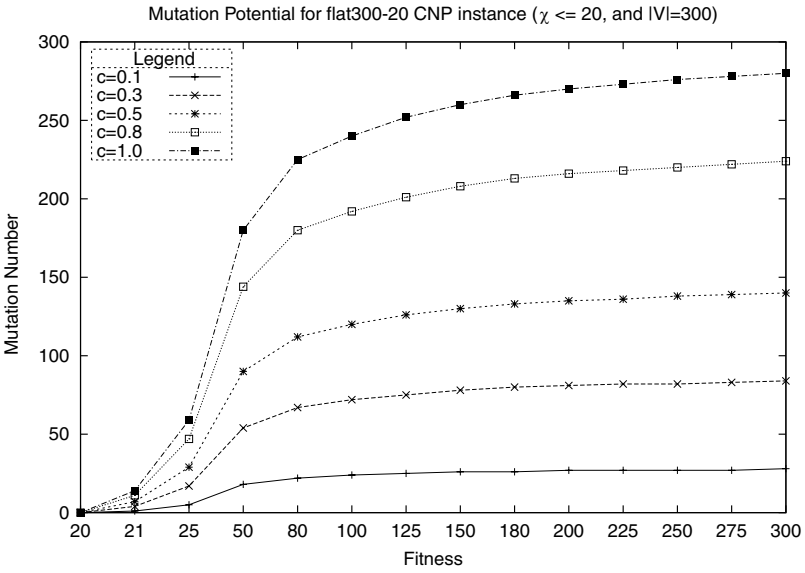

Fig. 4 Mutation potential obtained by Eq. (6) varying $c \in \{0.1, 0.3, 0.5, 0.8, 1.0\}$, for the CNP instance FLAT300_20_0

Section 7), we used high values of c for small and medium CNP instances ($c = 0.8$ for QUEEN6_6), and small values of c for large instances ($c = 0.1$ for FLAT300_20_0). it is also possible to see that a low mutation rate corresponds to good fitness values, whereas a high mutation rate corresponds to bad values, much like in natural immune system.

To improve the performances of *IMMALG* when it tackles a generic graph with a high number of vertices, we used a simple local search procedure, together with the hypermutation operator. Local search algorithms for combinatorial optimization problems generally rely on a definition of neighborhood. In our case, a B cell y is a neighbor of B cell x if it can be obtained from x, by swapping two elements of x. Thus, neighbors are generated by swapping vertex values: every time a proposed swap reduces the number of used colors, it is accepted and the process continues with the sequence of swaps, until it explores the whole neighborhood. Because swapping all pairs of vertices is time consuming, we used a reduced neighborhood: all $n = |V|$ vertices in a B cell are tested for a swap, but only with the closed ones in the permutation represented by the B cell itself. So we introduce the concept of neighborhood with radius R. Then, all the vertices were swapped only with their R nearest neighbors, to the left and to the right.

Given the possible large size of the neighborhood and n, we found it convenient to apply the local search procedure only on the population's best B cell, obtaining a new population P_N^{ls}, where N is the number of the B cell produced by local search process. We note that if $R = 0$ the local search procedure is not executed. This case is used for simple CNP instances, to avoid unnecessary fitness function evaluations. The applied local search is not critical to the searching process. Once a maximum number of generations has been fixed, the local search procedure increases only the success rate on a certain number of independent runs and, as a drawback, it increases the average number of evaluations to solutions. However, if we omit such procedure, *IMMALG* needs more generations, hence more fitness function evaluations, to obtain the same results of *IMMALG* using local search.

4.3 Aging phase

In the *Aging phase,* after the evaluation of P_{Nc}^{hyp} at time step t, the algorithm eliminates old B cells. Such an elimination process is stochastic, and, specifically, the probability that a B cell remains into the current population is governed by an exponential negative law with parameter τ_B, (expected mean life for the B cells):

$$P_{live}(\tau_B) = e^{-\frac{h}{\tau_B}}. \tag{7}$$

The aging process hopes to capitalize on the immunological fact that B cells have a limited life span. This operator is designed to generate diversity and turn-over into the population, in an attempt to avoid getting trapped in local minimum.

In Fig. 5 is showed the probability that a B cell remains into the population at different mean life (τ_B) values. In such a plot we considered $h \in \{0.5, \ln(0.2), 1.0, 2.0, 3.0, 4.0\}$. In general, for the comparisons and results obtained by *IMMALG*, shown in Section 7, we used $h = \ln(2)$.

Finally, the new population P_d^{t+1} of d elements is produced. It is possible to use two kinds of Aging phases: *pure aging phase* and *elitist aging phase*. With the elitist aging, when a new population for the next generation is generated, it contains the B cells with the best fitness function value. While in the pure aging the best B cells can be eliminated as well. We observe that the exponential rate of aging, $P_{live}(\tau_B)$, and the

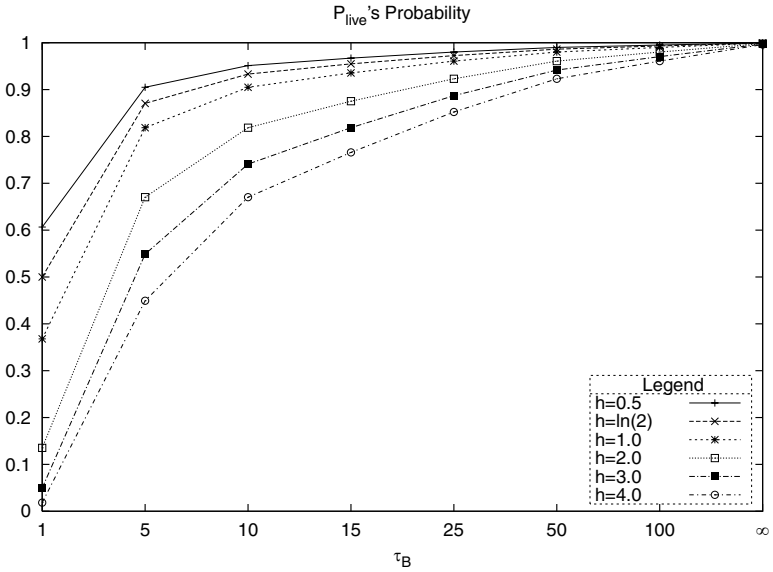

Fig. 5 P_{live} probabilty versus τ_B, varying h in {0.5, ln(0.2), 1.0, 2.0, 3.0, 4.0}

Table 1 Pseudo-code of *IMMALG*

IMMALG (d,dup,τ_B,R)
$t := 0$;
Random_Initialize $P_d^t = \boldsymbol{x_1}, \boldsymbol{x_2}, \triangleright\triangleright\triangleright \boldsymbol{x_d}\diamondsuit \in S^\ell$
while ($\frac{dK}{dt} \neq 0$) **do**
  InterAction(Ag, P_d^t);
  P_{Nc}^{clo} := Cloning (P_d^t,dup);
  P_{Nc}^{hyp} := Hypermutation (P_{Nc}^{clo});
  Evaluate (P_{Nc}^{hyp});
  P_N^{ls}:=Local_Search(P_{Nc}^{hyp},R);
  P_d^{t+1}:=Stochastic_Aging($P_d^t \cup P_{Nc}^{hyp} \cup P_N^{ls}$,$\tau_B$);
  $K(t,t_0)$:=KullbackEntropy();
  $t := t + 1$;
end_while

cloning potential, $V(f(\vec{x}))$, are inspired by biological processes (Seiden and Celada, 1992).

Sometimes it might be useful to apply a *birth phase* to increase the population diversity. This extra phase must be combined with an aging phase with a longer expected mean life time τ_B. For the CNP we did not use the birth phase because it produced a higher number of fitness function evaluations to solutions. In Table 1 is showed the pseudo–code of *IMMALG*.

4.4 Using kullback entropy as termination criterion

Kullback entropy function, described in Section 3, was used to analyze the learning process of *IMMALG*.

The gain is the amount of information the system has already learned from the given Ag-problem with respect to the initial distribution function (the randomly generated initial population $P_d^{(t_0=0)}$). Once the learning process starts, the kullback entropy increases monotonically until it reaches a final steady state (see Figs. 6 and 7). This is consistent with the idea of a *maximum entropy principle* (Jaynes, 2003; Sivia, 1996) of the form $\frac{dK}{dt} \geq 0$. Since $\frac{dK}{dt} = 0$ when the learning process ends, we use it as a termination condition for the proposed Immune Algorithm.

Moreover, kullback entropy function is useful to understand the *IMMALG*'s behavior and to set its parameters, as we will show in Section 6.

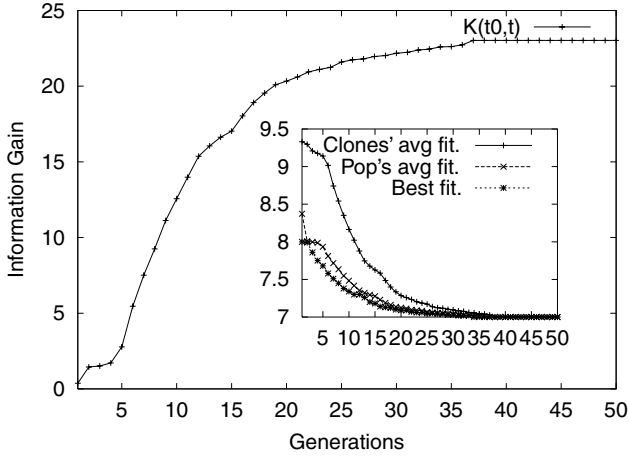

Fig. 6 Kullback entropy versus generations for the CNP instance QUEEN6_6

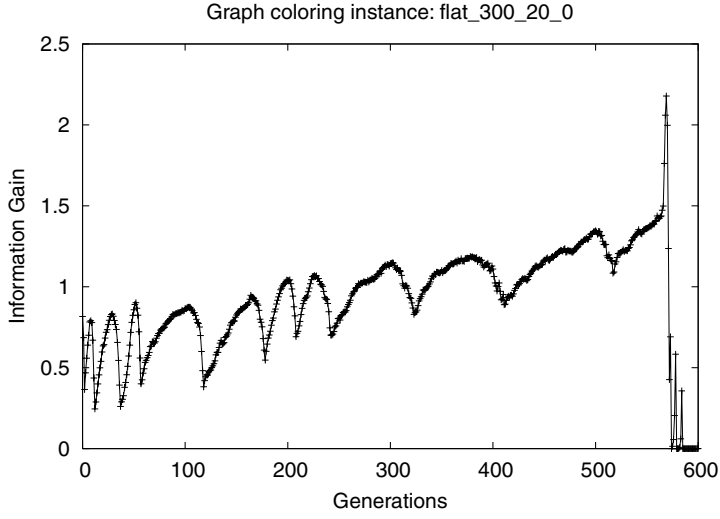

Fig. 7 Kullback entropy versus generations for the CNP instance FLAT300_20_0

Figure 6 shows the kullback entropy function when *IMMALG* faces the CNP instance QUEEN6_6 with vertex set $|V| = 36$, edge set $|E| = 290$ and optimal coloring 7. In particular, in the inset plot one can see the corresponding average fitness of population P_{Nc}^{hyp}, the average fitness of population P_d^{t+1} and the best fitness value. All the values are averaged on 100 independent runs.

In Fig. 7 is showed the kullback entropy function versus generation, tackling the CNP instance FLAT300_20_0, with $|V| = 300$, $|E| = 21,375$ and $\chi \leq 20$. In this plot one can see, that the kullback entropy has several fluctuations, until it reaches the peak, which corresponds to the maximum information obtained. After this point, the function quickly decreases to 0.

Finally, we note that our experimental protocol can have other termination criteria, such as the maximum number of evaluations or generations.

5 *IMMALG*'s dynamics

In Figs. 8 and 9 we show the dynamics of the fitness function. In both plots, we show the dynamics of the average fitness value of population P_{Nc}^{hyp}, P_d^{t+1}, and the best fitness value of population P_d^{t+1}.

Note that the average fitness value of P_{Nc}^{hyp} shows the diversity in the current population. When this value is equal to average fitness of population P_d^{t+1}, we are close to premature convergence or in the best case we are reaching a sub-optimal or optimal solution. It is possible to use the difference between the P_{Nc}^{hyp} average fitness value and the P_d^{t+1} average fitness value, as a standard to measure population diversity:

$$\left|avg_{fitness}\left(P_{Nc}^{hyp}\right) - avg_{fitness}\left(P_d^{t+1}\right)\right| = Pop_{div}.$$

When Pop_{div} rapidly decreases, this is considered as the primary reason for premature convergence.

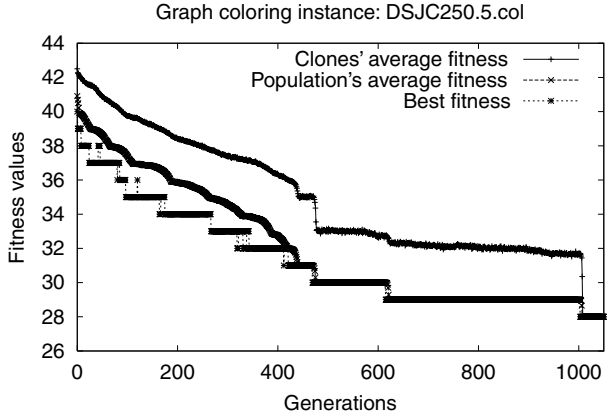

Fig. 8 *IMMALG* using pure aging phase: average fitness of population P_{Nc}^{hyp}, average fitness of population P_d^{t+1}, and best fitness value vs generations

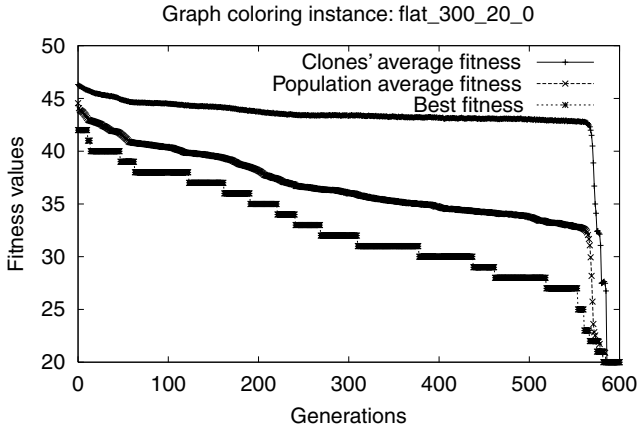

Fig. 9 *IMMALG* using elitist aging: average fitness of population P_{Nc}^{hyp}, average fitness of population $P_d^{(t+1)}$, and best fitness value vs generations

Figure 8 shows the *IMMALG* dynamics when it faces the DSCJ250.5.COL CNP instance ($|V| = 250$ and $|E| = 15,668$). For this plot, we used population size $d = 500$, duplication parameter $dup = 5$, expected mean life time $\tau_B = 10.0$ and neighborhood's radius $R = 5$. For this instance we use *pure aging* and we obtain the optimal coloring.

In Fig. 9 we tackled the FLAT_300_20 CNP instance ($|V| = 300$ and $|E| = 21,375$), with the following *IMMALG* parameters: $d = 1000$, $dup = 10$, $\tau_B = 10.0$ and $R = 5$. For this instance the optimal coloring is obtained using *elitist aging*. In general, with elitist aging the convergence is faster, even though it can trap the algorithm on a local optimum.

Although, with pure aging the convergence is slower and the population diversity is higher, our experimental results indicate that elitist aging seems to work well.

We can define the ratio $S_p = \frac{1}{dup}$ as the *selective pressure* of the algorithm: when $dup = 1$, obviously we have that $S_p = 1$ and the selective pressure is low, as we increase *dup* also the *IMMALG*'s selective pressure increases.

Experimental results show that high values of d denote high clone population average fitness and, in turn, high population diversity but, also, a high computational effort during the evolution.

6 Tuning of parameters

To understand how to set the *IMMALG* parameters, we performed several experiments on the CNP instance QUEEN6_6.

Firstly, we want to set the B cell's mean life, τ_B. Hence, we fix the population size $d = 100$, the duplication parameter $dup = 2$, the local search radius $R = 2$ and the total generations $gen = 100$. For each experiment we performed $runs = 100$ independent runs.

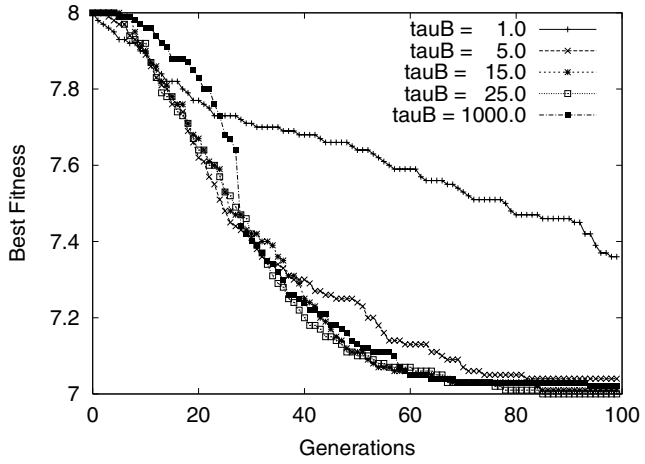

Fig. 10 Best fitness values vs generations

In Fig. 10 the best fitness values is showed, whilst in Fig. 11 the Kullback Entropy is displayed with respect to the following τ_B values $\{1.0, 5.0, 15.0, 25.0, 1000.0\}$.

When $\tau_B = 1.0$ the B cells have a shorter mean life, only one time step, and with this value *IMMALG* performed poorly. With $\tau_B = 1.0$ the maximum entropy obtained at generation 100 is about 13.

As τ_B increases, the best fitness values decrease and the kullback entropy increases. The best value for τ_B is 25.0. With $\tau_B = 1000.0$, and in general when τ_B is greater than a number of fixed generations gen, we can consider the B cells mean life infinite and obtain a pure elitist selection scheme.

In this special case, the behavior of *IMMALG* shows slower convergence in the first 30 generations in both plots. For values of τ_B greater than 25.0 we obtain slightly

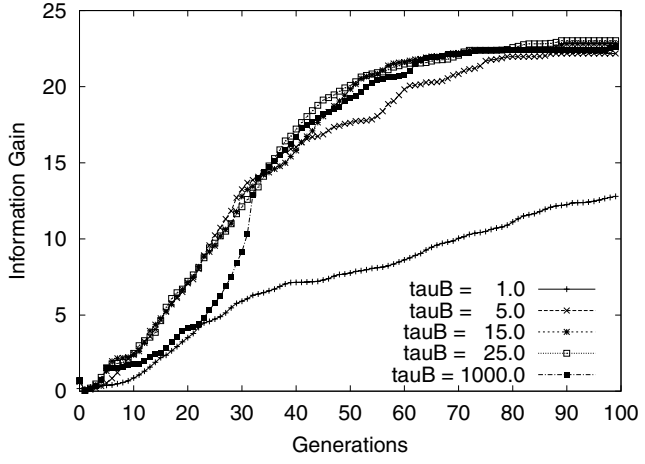

Fig. 11 Kullback entropy vs generations

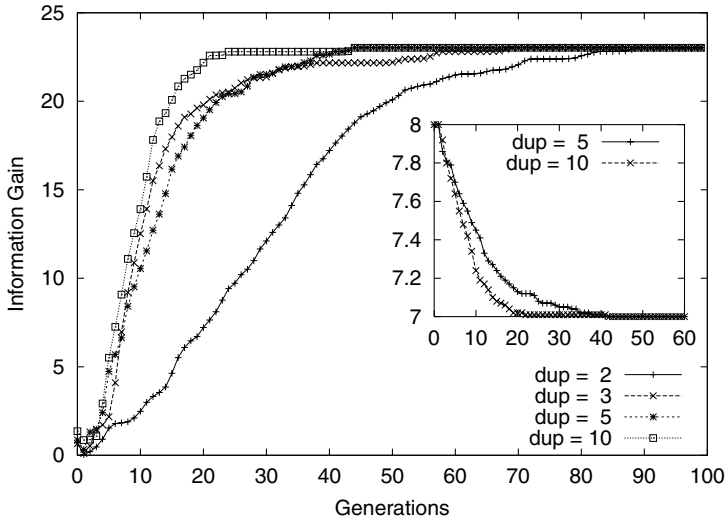

Fig. 12 Kullback entropy and Best fitness value for *dup*

worse results. Moreover, when $\tau_B \leq 10$ the success rate (SR) on 100 independent runs is less than 98 while when $\tau_B \geq 10$ *IMMALG* obtains a $SR = 100$ with a lower Average number of Evaluations to Solutions (AES) located when $\tau_B = 25.0$.

Now we fix $\tau_B = 25.0$ and vary *dup*. In Fig.12 we note that *IMMALG* quickly gains more information (kullback entropy) at each generation with $dup = 10$, whilst the best fitness value is reached faster with $dup = 5$.

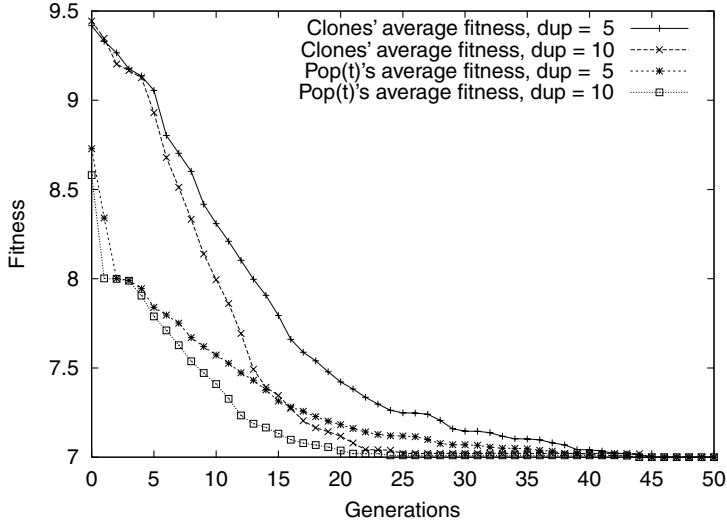

Fig. 13 Average fitness of Clones and P_d^t for $dup \in \{5, 10\}$

With both values of dup the largest kullback entropy is obtained at generation 43. Moreover, with $dup = 10$ the best fitness value is obtained at generation 22, whereas with $dup = 5$ at generation 40.

One may deduce that $dup = 10$ is the best value for the cloning of B cells since with this value of dup we obtain more kullback entropy in less time.

This is not always true. Indeed, if we observe Fig. 13 we can see how *IMMALG* with $dup = 5$ obtains a higher clone average fitness and hence a greater diversity. This

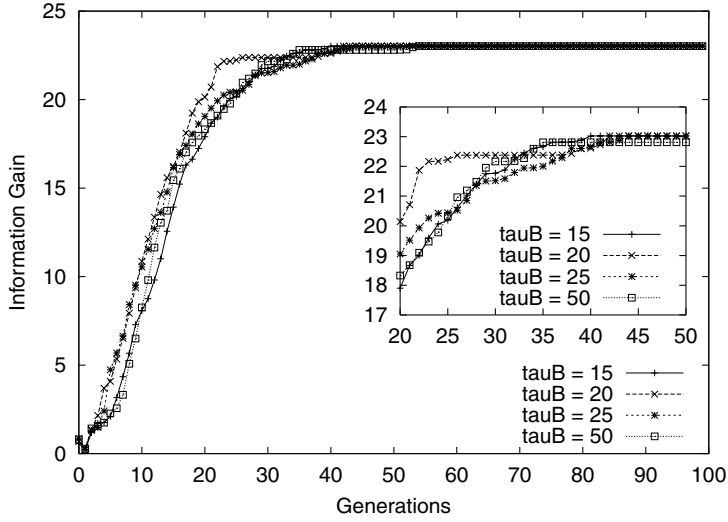

Fig. 14 Kullback entropy for $\tau_b \in \{15, 20, 25, 50\}$

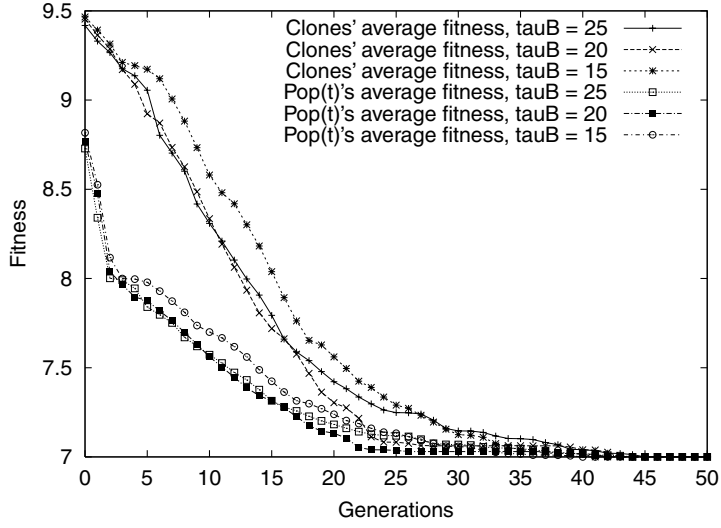

Fig. 15 Average fitness of population P_{Nc}^{hyp} and population P_d^t for $\tau_b \in \{15, 20, 25\}$

characteristic can be useful in avoiding premature convergence and in finding more optimal solutions for a given combinatorial problem.

From Figs.10 and 11, we saw that for $dup = 2$, the best value of τ_B is 25.0; but last experimental results described, show a better performance for $dup = 5$. Then, we set $dup = 5$ and vary τ_B; we obtain the results shown in Figs. 14 and 15.

From these plots, one can see that for $\tau_B = 15$ *IMMALG* reaches the maximum entropy at generation 40 (Fig. 14) and more diversity (Fig. 15). Hence, when $dup = 2$ the best value of τ_B is 25.0, i.e. on average we need 25 generations for the B cells to reach a mature state. On the other hand, when $dup = 5$ the correct value is 15.0 Thus, as increases the value of *dup*, the average time for the population of B cells, to reach a mature state, decreases.

Local search is useful for large instances (see Table 2). The cost of local search, though, is high. Figure 16 shows how the AES increases as the neighborhood radius increases. The plot reports two classes of experiments performed with 1000 and 10000 independent runs.

In Fig. 17 we show the values of the parameters d and *dup* as functions of the Success Rate (SR). Each point has been obtained averaging 1000 independent runs. One can see that there is a certain relation between d and *dup*, which has to be satisfied in order to reach a $SR = 100$.

With respect to the QUEEN6_6 instance, for low values of the population size we need a high value of *dup* to reach $SR = 100$. For $d = 10$, $dup = 10$ is not sufficient to obtain the maximum *SR*. On the other hand, as the population size increases, smaller values for *dup* work better.

Small values of *dup* are a positive factor. We recall that *dup* is similar to the temperature in Simulated Annealing (Johnson et al., 1991). Low values of *dup* correspond to a system that cools down slowly and has a high EAS.

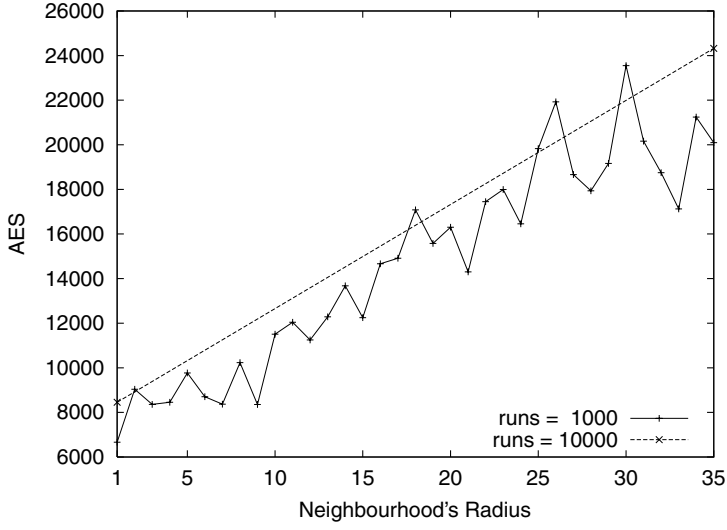

Fig. 16 Average number of Evaluations to Solutions versus neighborhood's radius

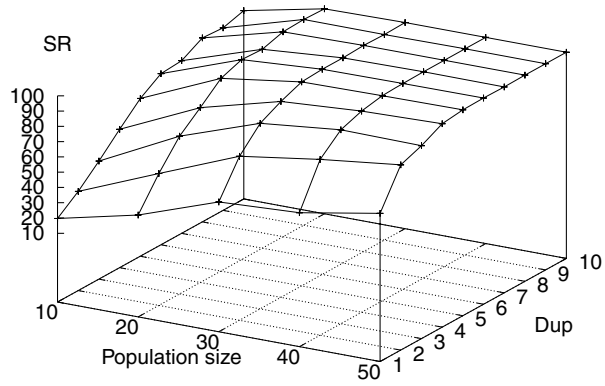

Fig. 17 3D plot of d, *dup* versus Success Rate (SR)

Finally, in Fig. 18 we show the average fitness of the current population, best fitness and entropy function, when *IMMALG* tackled CNP instance QUEEN6_6. Such an experiment, was performed on 100 independent runs.

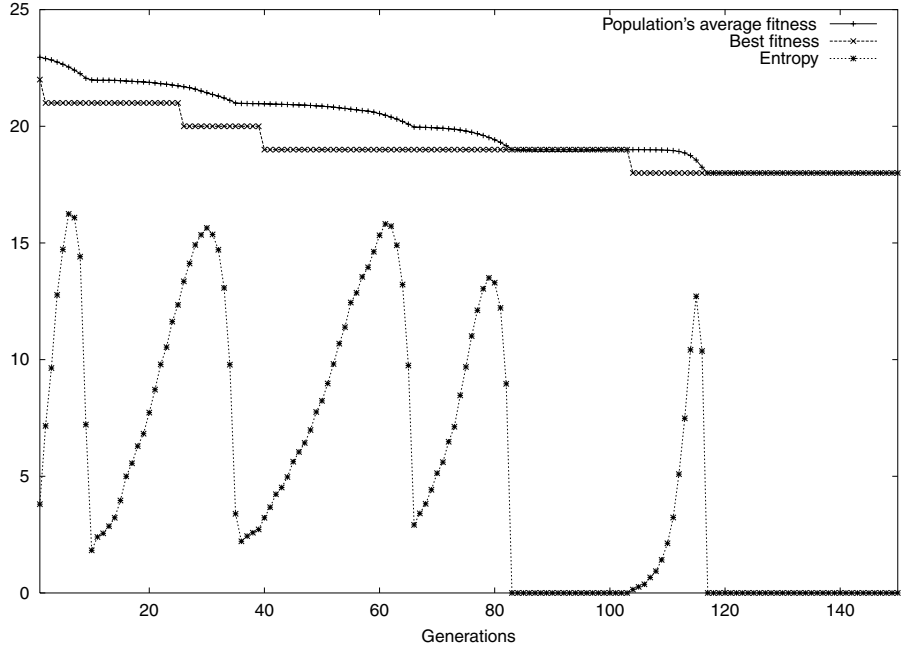

Fig. 18 Average fitness of population P_d^{t+1}, best fitness and entropy function versus generations

7 Comparisons and results

In this section we report our experimental results. We worked with classical benchmark graphs (Johnson and Trick, 1996): the MYCIELSKI, QUEEN, DSJC and LEIGHTON chromatic number instances. Results are reported in Tables 2 and 3. In these experiments the *IMMALG*'s best found value is always obtained with $SR = 100$. For all the results presented in this section, we used *elitist aging*.

To evaluate the goodness of the performances, we have compared *IMMALG* to several approaches presented on the DIMACS challenge's proceedings (Johnson and Trick, 1996) and we show the relative results in Table 4.

The algorithms we have considered for comparison are: I_GREEDY, an iterated version of the greedy procedure (Culberson and Luo, 1996); T_B&B, a branch-and-bound algorithm using tabu search (Glover et al., 1996); DIST (Morgenstern, 1996); PAR, a different version of DIST using a parallel branch-and-bound approach (Lewandowski and Condon, 1996); T_GEN_1, an hybrid genetic algorithm with tabu search (Fleurent and Ferland, 1996). Examining the table, one can see as the proposed *IMMALG* is competitive and comparable to the several methodologies used in literature for the coloring of a generic graph.

Table 2 Mycielsky and Queen graph instances. We fixed $\tau_B = 25.0$, and the number of independent runs 100. OC denotes the Optimal Coloring

Instance G	$\|V\|$	$\|E\|$	OC	(d,dup,R)	Best found	AES
Myciel3	11	20	4	(10,2,0)	4	30
Myciel4	23	71	5	(10,2,0)	5	30
Myciel5	47	236	6	(10,2,0)	6	30
Queen5_5	25	320	5	(10,2,0)	5	30
Queen6_6	36	580	7	(50,5,0)	7	3750
Queen7_7	49	952	7	(60,5,0)	7	11,820
Queen8_8	64	1,456	9	(100,15,0)	9	78,520
Queen8_12	96	2,736	12	(500,30,0)	12	908,000
Queen9_9	81	1,056	10	(500,15,0)	10	445,000
School1_nsh	352	14,612	14	(1000,5,5)	15	2,750,000
School1	385	19,095	9	(1000,10,10)	14	3,350,000

Table 3 Experimental results on subset instances of DSJC and Leighton graphs. We fixed $\tau_B = 15.0$, and the number of independent runs 10

Instance G	$\|V\|$	$\|E\|$	OC	(d,dup,R)	Best found	AES
DSJC125.1	125	736	5	(1000,5,5)	5	1,308,000
DSJC125.5	125	3,891	12	(1000,5,5)	18	1,620,000
DSJC125.9	125	6,961	30	(1000,5,10)	44	2,400,000
DSJC250.1	250	3,218	8	(400,5,5)	9	1,850,000
DSJC250.5	250	15,668	13	(500,5,5)	28	2,500,000
DSJC250.9	250	27,897	35	(1000,15,10)	74	4,250,000
le450_15a	450	8,168	15	(1000,5,5)	15	5,800,000
le450_15b	450	8,169	15	(1000,5,5)	15	6,010,000
le450_15c	450	16,680	15	(1000,15,10)	15	10,645,000
le450_15d	450	16,750	9	(1000,15,10)	16	12,970,000

Table 4 *IMMALG* versus six different algorithms presented on the DIMACS challenge. The values are averaged on 5 independent runs. With $b.k.$ we indicated the best known coloring number for each instances

Instance G	χ	$b.k.$	I_GREEDY	T_B&B	DIST	PAR	T_GEN_1	*IMMALG*
DSJC125.5	12	12	18.9	20	**17**	**17.0**	**17.0**	18.0
DSJC250.5	13	13	32.8	35	**28**	29.2	29.0	**28.0**
flat300_20_0	≤20	20	20.2	39	**20**	**20.0**	**20.0**	**20.0**
flat300_26_0	≤26	26	37.1	41	**26**	32.4	**26.0**	27.0
flat300_28_0	≤28	29	37.0	41	**31**	33.0	33.0	32.0
le450_15a	15	15	17.9	16	**15**	**15.0**	**15.0**	**15.0**
le450_15b	15	15	17.9	**15**	**15**	**15.0**	**15.0**	**15.0**
le450_15c	15	15	25.6	23	**15**	16.6	16.0	**15.0**
le450_15d	15	15	25.8	23	**15**	16.8	16.0	16.0
mulsol.i.1	–	49	49.0	**49**	**49**	**49.0**	**49.0**	**49.0**
school1_nsh	≤14	14	14.1	26	20	**14.0**	**14.0**	15.0

Table 5 *IMMALG* versus Evolve_AO algorithm. The values are averaged on 5 independent runs

Instance G	$\chi(G)$	Best-Known	Evolve_AO	*IMMALG*	Difference
DSJC125.5	12	12	17.2	18.0	+0.8
DSJC250.5	13	13	29.1	28.0	**−0.9**
flat300_20_0	≤20	20	26.0	20.0	**−6.0**
flat300_26_0	≤26	26	31.0	27.0	**−4.0**
flat300_28_0	≤28	29	33.0	32.0	**−1.0**
le450_15a	15	15	15.0	15.0	**0**
le450_15b	15	15	15.0	15.0	**0**
le450_15c	15	15	16.0	15.0	**−1.0**
le450_15d	15	15	19.0	16.0	**−3.0**
mulsol.i.1	–	49	49.0	49.0	**0**
school1_nsh	≤14	14	14.0	15.0	+1.0

Table 6 *IMMALG* versus Galinier and Hao's algorithm with and without tabu search, respectively HCA and GPB. The number of independent runs is 10, except for the GPB where for each instance the values are averaged on 3 independent runs (SR*)

Instance G	HCA's Best-found & (SR)	GPB's Best-found & (SR*)	*IMMALG*'s Best-found & (SR)
DSJC250.5	28 (90)	28 (100)	28 (100)
flat300_28_0	31 (60)	31 (100)	32 (100)
le450_15c	15 (60)	15 (100)	15 (100)
le450_25c	26 (100)	26 (100)	25 (100)

In Tables 5 and 6 we compare *IMMALG* with three of the best evolutionary algorithms, respectively Evolve_AO algorithm (Barbosa et al., to appear) and the Galinier and Hao's Algorithm with and without tabu search, HCA (Galinier and Hao, 1999) and GBP (Glass and Prügel-Bennet, 2003). For all the chromatic number instances we ran *IMMALG* with the following parameters: $d = 1000$, $dup = 15$, $R = 30$, and $\tau_B = 20.0$. For these classes of experiments the goal is to obtain the best possible coloring, no matter what the value of AES may be.

Table 5 shows how *IMMALG* outperforms the Evolve_AO algorithm, while is similar in results to the two different versions of the Galinier and Hao's Algorithm and better in SR values with respect to HCA, i.e. using tabu search (see Table 6). We note that for the experiments shown by GPB (Glass and Prügel-Bennet, 2003), each instance was averaged on 3 independent runs.

8 Conclusion and final remarks

We have designed a new IA, *IMMALG*, that incorporates a stochastic aging operator and a simple local search procedure to improve the overall performances obtained in tackling the chromatic number instances. The presented algorithm *IMMALG* has only four parameters. In order to correctly set these parameters we use the kullback entropy function, a particular entropy function useful to understand the IA's behavior. The kullback entropy measures the quantity of information that the system discovers during the learning process. We choose the parameters that maximize the discovered information and that moderately increase the kullback entropy monotonically. To our knowledge, this is the first time that IAs, and in general EAs, are characterized in terms of kullback entropy. We define the average fitness value of population P_{Nc}^{hyp} as the diversity in the current population. When this value is equal to the average fitness value of population $P_d^{(t+1)}$, we are close a premature convergence. Using a simple coloring method we have investigated the *IMMALG*'s learning and solving capabilities. The experimental results show how *IMMALG* is comparable to and, in many CNP instances, outperforms the best evolutionary algorithms. Finally, the designed IA is directed to solving CNP instances although the solutions' representation and the variation operators are applicable more generally, for example to the Protein Structure Prediction problem for the 2D HP model (Cutello et al., 2004a, b, 2005a, b,) and to global numerical optimization (Cutello et al., 2005b, c).

References

Ausiello G. Crescenzi P, Gambosi G, Kann V, Marchetti-Spaccamela A, Protasi M (1999) Complexity and approximation. Springer-Verlag

Barbosa VC, Assis CAG, do Nascimento JO (to appear) Two novel evolutionary formulations of the graph coloring problem. J Combin Optim

Bollobas B (1998) Modern graph theory. Graduate texts in mathematics, vol. 184. Springer-Verlag, Berlin Heidelberg New York

Brooks RL (1941) On colouring the nodes of a network. Cambridge Phil Soc 37:194–197

Caramia M, Dell'Olmo P (2001) Iterative coloring extension of a maximum clique. Naval Res Logis 48:518–550

Chow FC, Hennessy JL (1990) The priority-based coloring approach to register allocation. ACM Trans Program Languages Syst 12:501–536

Culberson JC, Luo F (1996) Exploring the k-colorable landscape with iterated greedy. Cliques, coloring and satisfiability: second DIMACS implementation challenge. American Mathematical Society, Providence, RI, pp 245–284

Cutello V, Morelli G, Nicosia G, Pavone M (2005) Immune algorithms with aging operators for the string folding problem and the protein folding problem. Lecture Notes Comput Sci 3448:80–90

Cutello V, Nicosia G, Pavone M (2004) Exploring the capability of immune algorithms: a characterization of hypermutation operators. Lecture Notes Comput Sci 3239:263–276

Cutello V, Narzisi G, Nicosia G, Pavone M (2005) Clonal selection algorithms: a comparative case study using effective mutation potentials. Lecture Notes Comput Sci 3627:13–28

Cutello V, Narzisi G, Nicosia G, Pavone M (2005) An immunological algorithm for global numerical optimization. In: Proc. of the seventh international conference on artificial evolution (EA'05). To appear

Cutello V, Nicosia G, Pavone M (2003) A hybrid immune algorithm with information gain for the graph coloring problem. In: Proceedings of genetic and evolutionary computation conference (GECCO) vol. 2723. Springer, pp 171–182

Cutello V, Nicosia G, Pavone M (2004) An immune algorithm with hyper-macromutations for the Dill's 2D hydrophobic-hydrophilic model. Congress on Evolutionary Computation, vol. 1. IEEE Press, pp 1074–1080

Dasgupta D (ed) (1999) Artificial immune systems and their applications. Springer-Verlag, Berlin Heidelberg New York

De Castro LN, Timmis J (2002) Artificial immune systems: a new computational intelligence paradigm. Springer-Verlag, UK

De Castro LN, Von Zuben FJ (2000) The clonal selection algorithm with engineering applications. In: Proceedings of GECCO 2000, workshop on artificial immune systems and their applications, pp 36–37

De Castro LN, Von Zuben FJ (2002) Learning and optimization using the clonal selection principle. IEEE Trans Evolut Comput 6(3):239–251

de Werra D (1985) An introduction to timetabling. European J Oper Res 19:151–162

Diestel R (1997) Graph theory. Graduate texts in mathematics, vol. 173. Springer-Verlag, Berlin Heidelberg New York

Eiben AE, Hinterding R, Michalewicz Z (1999) Parameter control in evolutionary algorithms. IEEE Trans Evolut Comput 3(2):124–141

Fleurent C, Ferland JA (1996) Object-oriented implementation of heuristic search methods for graph coloring, maximum clique and satisfability. Cliques, coloring and satisfiability: second DIMACS implementation challenge. American Mathematical Society, Providence, RI, pp 619–652

Forrest S, Hofmeyr SA (2000) Immunology as information processing. Design principles for immune system & other distributed autonomous systems. Oxford Univ. Press, New York

Galinier P, Hao J (1999) Hybrid evolutionary algorithms for graph coloring. J Comb Optim 3(4):379–397

Galvin F, Komjáth P (1991) Graph colorings and the axiom of the choice. Period Math Hungar 22:71–75

Gamst A (1986) Some lower bounds for a class of frequency assignment problems. IEEE Trans Vehicular Techn 35:8–14

Garey MR, Johnson DS, So HC (1976) An application of graph coloring to printed circuit testing. IEEE Trans Circ Syst CAS-23:591–599

Garey MR, Johnson DS (1979) Computers and intractability: a guide to the theory of NP-completeness. Freeman, New York

Garrett SM (2005) How do we evaluate artificial immune systems, vol. 13, no. 2 ? Evolutionary Computation, Mit Press, pp 145–178

Glass CA, Prügel-Bennet A (2003) Genetic algorithm for graph coloring: exploration of galinier and hao's algorithm. J Combinat Optim 7(3):229–236

Glover F, Parker M, Ryan J (1996) Coloring by Tabu branch and bound. Cliques, coloring and satisfiability: second DIMACS implementation challenge. American Mathematical Society, Providence, RI, pp 285–307

Halldórsson MM (1993) A still better performance guarantee for approximate graph coloring. Inf Proc Lett 45:19–23

Hamiez J, Hao J (2003) An analysis of solution properties of the graph coloring problem. Metaheuristics: Computer Decision-Making. Kluwer, Chapter 15, pp 325–326

Jaynes ET (2003) Probability theory: the logic of science. Cambridge University Press

Jensen TR, Toft B (1995) Graphs coloring problems. Wiley-Interscience Series in Discrete Mathematics and Optimization

Johnson DR, Aragon CR, McGeoch LA, Schevon C (1991) Optimization by simulated annealing: an experimental evaluation; part II, graph coloring and number partitioning. Oper Res 39:378–406

Johnson DS, Trick MA (eds) (1996) Cliques, coloring and satisfiability: second DIMACS implementation challenge. Am Math Soc, Providence, RI

Kullback S (1959) Statistics and information theory. J. Wiley and Sons, New York

Leighton FT (1979) A graph coloring algorithm for large scheduling problems. J Res National Bureau Standard 84:489–505

Leung K, Duan Q, Xu Z, Wong CW (2001) A new model of simulated evolutionary computation—convergence analysis and specifications. IEEE Trans Evolut Comput 5(1):3–16

Lewandowski G, Condon A (1996) Experiments with parallel graph coloring heuristics and applications of graph coloring. Cliques, coloring and satisfiability: second DIMACS implementation challenge. American Mathematical Society, Providence, RI, pp 309–334

Marino A, Damper RI (2000) Breaking the symmetry of the graph colouring problem with genetic algorithms. Workshop proc. of the genetic and evolutionary computation conference (GECCO'00). Morgan Kaufmann, Las Vegas, NV

Mehrotra A, Trick MA (1996) A column generation approach for graph coloring. INFORMS J Comput 8:344–354

Morgenstern C (1996) Distributed coloration neighborhood search. Cliques, coloring and satisfiability: second DIMACS implementation challenge. American Mathematical Society, Providence, RI, pp 335–357

Nicosia G, Castiglione F, Motta S (2001) Pattern recognition by primary and secondary response of an artificial immune system. Theory Biosci 120:93–106

Nicosia G, Castiglione F, Motta S (2001) Pattern recognition with a multi-agent model of the immune system. Int. NAISO symposium (ENAIS'2001). ICSC Academic Press, Dubai, UAE, pp 788–794

Nicosia G, Cutello V (2002) Multiple learning using immune algorithms. In: Proceedings of the 4th international conference on recent advances in soft computing. RASC, Nottingham, UK

Seiden PE, Celada F (1992) A model for simulating cognate recognition and response in the immune system. J Theor Biol 158:329–357

Shannon CE (2004) A mathematical theory of communication. Congress on evolutionary computation, vol. 1. IEEE Press, pp 1074–1080 Bell System Technical Journal, vol. 27, pp 379–423 and 623–656 (1948)

Sivia DS (1996) Data analysis. A Bayesian Tutorial. Oxford Science Publications

Tsang EPK (1993) Foundations of constraint satisfaction, vol. 37. Academic Press