Journal of Experimental & Theoretical Artificial Intelligence

ISSN: 0952-813X (Print) 1362-3079 (Online) Journal homepage: https://www.tandfonline.com/loi/teta20

Quantifying the exploration performed by metaheuristics

Paola Pellegrini & Daniela Favaretto

To cite this article: Paola Pellegrini & Daniela Favaretto (2012) Quantifying the exploration performed by metaheuristics, Journal of Experimental & Theoretical Artificial Intelligence, 24:2, 247-266, DOI: 10.1080/0952813X.2012.656327

To link to this article: https://doi.org/10.1080/0952813X.2012.656327

Journal of Experimental & Theoretical Artificial Intelligence
Vol. 24, No. 2, June 2012, 247–266

Taylor & Francis
Taylor & Francis Group

Quantifying the exploration performed by metaheuristics

Paola Pellegrini[ab]* and Daniela Favaretto[b]

[a]*IRIDIA, CoDE, Université Libre de Bruxelles (ULB), Brussels, Belgium;*
[b]*Department of Management, University Ca' Foscari, Cannaregio 873, Venice 30121, Italy*

(*Received 6 December 2010; final version received 20 November 2011*)

In this article we propose a formalisation of the concept of exploration performed by metaheuristics. In particular, we define and test a method for studying this aspect regardless of the specific approach implemented. Understanding the behaviour of metaheuristics is important for being able to boost their results. Measuring the exploration performed may help increase this understanding. We propose an experimental analysis to show how the measure of exploration defined may be used to this aim. We quantify the different level of exploration implied by different parameter settings in an ant colony optimisation and in a genetic algorithm for the travelling salesman problem. The results suggest that it may be possible to establish a relation between exploration and performance of the algorithm.

Keywords: metaheuristics; cluster analysis; exploration; ant colony optimisation; genetic algorithm

1. Introduction

Metaheuristics are stochastic approaches that are widely used for tackling difficult optimisation problems. Understanding and being able to guide their behaviour may contribute to the achievement of high quality results. As Wolpert and Macready (1997) state in their 'No Free Lunch Theorem for Optimisation', *for any algorithm, any elevated performance over one class of problems is exactly paid for in performance over another class*. Each single class of instances may require a procedure with particular characteristics in order to be solved effectively. Thus, both choosing the appropriate approach and fitting its behaviour to the specific problem to be tackled play a major role in the design of a well-performing algorithm. One concept that may guide the evaluation of an algorithm is the exploration it performs. Exploration is the act of investigating a space to the aim of gaining information on its structure and features. In optimisation, an algorithm explores the feasible region of a problem for identifying local and global optima according to the objective function.

The behaviour of metaheuristics may be described in terms of *exploration* and *exploitation* (Blum and Roli 2003). Finding the right balance of these concepts is crucial

*Corresponding author. Email: paola.pellegrini@ulb.ac.be; paolap@unive.it

ISSN 0952–813X print/ISSN 1362–3079 online

http://dx.doi.org/10.1080/0952813X.2012.656327
http://www.tandfonline.com

for quickly identifying regions of the search space with high-quality solutions. Blum and Roli (2003) affirm that *although the relevance of these two concepts is commonly agreed, so far there is no unifying description to be found in the literature. Descriptions are very generic and metaheuristic specific*. To the best of our knowledge, a formal definition of *exploration* is still missing in the field. A measurement of such phenomenon independent from the approach used is not yet available. The papers that deal with this topic focus on some specific procedures. Among others, Amor and Rettinger (2005), Eshelman and Schaffer (1991), Bhattacharya (2008) and Wineberg and Oppacher (2003a,b) deal with the behaviour of genetic algorithms (GAs). Herault (2000), Desai, (1999) and Orosz and Jacobson (2002) study simulated annealing, while Devarenne, Mabed, and Caminada (2006) and Watson, Whitley, and Howe (2005) propose interesting observations on tabu search. Finally, Dorigo and Stützle (2004), Merkle and Middendorf (2001) and Pellegrini and Ellero (2008) consider ant colony optimisation (ACO). Other authors describe the behaviour of metaheuristics based on intuitive observations (Blum and Roli 2003). Finally, some authors deal with the exploration performed by different approaches focusing on the specific characteristics of a problem (see, e.g. Schuurmans and Southey 2001).

In this article, we propose a definition and a measure of exploration independent both on the algorithms implemented and on the problem to be tackled. Through such a measure it is possible to observe the behaviour of any metaheuristic without needing a deep knowledge on the internal state of the algorithm. Moreover, it gives the possibility of comparing the behaviour of very different approaches. We show how this can be done through an experimental analysis. We tackle ten TSP instances with an ACO and a GA. First, we assess the difference in the level of exploration performed that is implied by different parameter settings for each algorithm. Second, we compare the exploration performed by the two algorithms. The results suggest that it may be possible to identify a relation between the exploration performed and the quality of the solutions obtained.

The rest of this article is organised as follows. In Section 2 we define the exploration and we describe how it can be measured in practice. In Section 3 we report the algorithms implemented for the experimental analysis depicted in Section 4. Finally, in Section 5 we draw some conclusions.

2. Exploration

In this section, we propose an original definition of the exploration performed by a metaheuristic algorithm, and we deal with some technical aspects that need to be tackled for turning the new definition into a practically implementable measure.

2.1. *Definition*

An optimisation problem is a pair (S, f) in which the solution space S is a set of feasible solutions and the cost function f is a mapping $f : S \to \mathbb{R}$. In the case of minimisation, the problem is to find a globally-minimal solution, i.e. a solution $s^* \in S$ such that $f(s^*) \leq f(s)$, $\forall s \in S$. A set of n variables is given, each in a defined domain: $x_i \in D_i,\ i = 1, \ldots, n$. A feasible solution is a vector $s = (x_1, x_2, \ldots, x_n)$ in which each variable is instantiated for satisfying the problem's constraints.

Let $\mathcal{B}$ be the set of metaheuristic algorithms of interest, $\mathcal{I}$ be the set of instances to be tackled and $\mathcal{R}$ be the set of possible combinations of computational resources, such as time and memory space. A run is a function

$$r : \mathcal{B} \times \mathcal{I} \times \mathcal{R} \times \mathbb{R} \to \mathcal{S}, \tag{1}$$

where the fourth independent variable is the seed of the random number generator and $\mathcal{S}$ is the set of all possible sets of feasible solutions. We consider the available computational resources as exogenous inputs: indeed, an algorithm is executed on a specific hardware. The characteristics of this hardware have an impact on the performance of the algorithm over time. Similarly, the set of solutions visited varies as a function of the elapsed computational time.

In this context, the exploration performed in a run can be defined as a function

$$E : \mathcal{B} \times \mathcal{I} \times \mathcal{R} \times \mathbb{R} \to \mathbb{N}\backslash\{0\} \tag{2}$$

such that, given a set of inputs $B_1,\ B_2 \in \mathcal{B},\ I \in \mathcal{I},\ R_1,\ R_2 \in \mathcal{R},\ h_1,\ h_2 \in \mathbb{R}$, the following relation must hold:

$$r(B_1, I, R_1, h_1) \subseteq r(B_2, I, R_2, h_2) \Rightarrow E(B_1, I, R_1, h_1) \leq E(B_2, I, R_2, h_2). \tag{3}$$

In this framework, we propose the following definition:

The exploration performed by a metaheuristic algorithm is represented by the number of clusters of solutions visited:

$$E(B, I, R, h) = |C(B, I, R, h)|, \tag{4}$$

where $C(B, I, R, h)$ is the set of clusters resulting from the solutions visited by algorithm $B \in \mathcal{B}$ when solving instance $I \in \mathcal{I}$ using resources $R \in \mathcal{R}$ and seed $h \in \mathbb{R}$ for the random number generator.

Given this definition, the representation of solutions in the space plays a central role in the quantification of exploration. We propose here a representation that takes into account the perspective of the algorithm that explores the space: we aim at capturing the exploratory attitude of the algorithm beyond the topological dispersion of the solutions visited.

Let $P_i^z(x_i)$ be the probability distribution associated by the algorithm to the domain D_i of variable x_i, $i = 1, \ldots, n$, at iteration z, and let $P_i(x_i) = \max_z\{P_i^z(x_i)\}$. Each solution $s = (x_1, x_2, \ldots, x_n)$ is represented as $\bar{s} = (x_1 P_1(x_1), x_2 P_2(x_2), \ldots, x_n P_n(x_n))$. In this way, a solution embodies information both on the actual evaluation of each variable and on the probability associated to it. By considering the maximum value of the probability across all iterations, we obtain that $s_1 = s_2$ implies $\bar{s}_1 = \bar{s}_2$, regardless of the iterations in which s_1 and s_2 are visited. Moreover, suppose that at the beginning of a run the probability distribution is uniform. If, after some time, few values become very likely for a variable, the information that needs to be recorded is the emergent intensification.

2.2. *The clustering procedure*

For exploiting the definition of exploration proposed, an appropriate clustering procedure needs to be identified. An agglomerative hierarchical procedure (Jardine and Sibson 1968)

appears suitable: at each step, the two closest solutions are grouped together to form a cluster. In the following we define how to compute the distance between a new cluster and the others, and the stopping criterion implemented.

Initially, each cluster contains one solution. The distance between the starting clusters is the distance between solutions. For what concerns the distance between a new-born cluster and the others, let g_1 and g_2 be two clusters that are being grouped in $\hat{g}$. The distance between $\hat{g}$ and a third cluster $\bar{g}$ is

$$\text{distance}(\hat{g}, \bar{g}) = \max\{\text{distance}(g_1, \bar{g}), \text{distance}(g_2, \bar{g})\}. \tag{5}$$

For what concerns the stopping criterion implemented, the agglomerative procedure terminates when the distance between the two closest clusters is greater than a predefined threshold ϵ. Such a threshold must be coherent with the magnitude of the distance matrix analysed. As shown by Pellegrini et al. (2009), the exact value is not relevant to the results.

Given a run, the exploration is a function of ϵ: the higher ϵ, the larger and then the fewer the clusters. Still, for any pair ϵ' and ϵ'', and for any $B_1, B_2 \in \mathcal{B}$, $I \in \mathcal{I}$, $R_1, R_2 \in \mathcal{R}$, $h_1, h_2 \in \mathbb{R}$,

$$E_{\epsilon'}(B_1, I, R_1, h_1) \leq E_{\epsilon'}(B_2, I, R_2, h_2) \Rightarrow E_{\epsilon''}(B_1, I, R_1, h_1) \leq E_{\epsilon''}(B_2, I, R_2, h_2), \tag{6}$$

where $E_{\epsilon'}$ is the exploration computed by setting $\epsilon = \epsilon'$.

E_ϵ measures the exploration in relative terms: it identifies which algorithm, or which algorithm configuration, explores the most, rather than indicating an absolute value for the exploration of a single algorithm.

The pseudo-code of the procedure used for computing exploration is reported in Figure 1.

/* Input: solutions $\bar{s} \in S$, ϵ.
Initialize variables *minimumDist*, E */
Define the set of clusters $G = \{g_1, g_2, ..., g_{|S|}\} = \{\{\bar{s}_1\}, \{\bar{s}_2\}, ..., \{\bar{s}_{|S|}\}\}$
for each $g_h, g_k \in G$ do
  $d_{g_h,g_k} = \sqrt{\sum_{i=1}^{n} (x_i^h P_i(x_i^h) - x_i^k P_i(x_i^k))^2}$
  /* $\bar{s}_h = (x_1^h, x_2^h, ...x_n^h)$, $\bar{s}_k = (x_1^k, x_2^k, ...x_n^k)$ */
minimumDist := $\min_{g_h,g_k \in G} d_{g_h,g_k}$
while *minimumDist* $\leq \epsilon$
  randomly select $g_h, g_k \in G : d_{g_h,g_k} = minimumDist$
  $g_h := g_h \cup g_k$
  $G := G \setminus \{g_k\}$
  for each $g_w \in G \setminus \{g_h\}$
    $d_{g_h,g_w} = d_{g_w,g_h} = \max\{d_{g_h,g_w}, d_{g_k,g_w}\}$
  minimumDist := $\min_{g_h,g_k \in G} d_{g_h,g_k}$
$E := |G|$
return E

Figure 1. Pseudo-code of the procedure for computing exploration.

2.3. *Constructive and perturbative methods*

Several metaheuristic algorithms have been proposed in the literature for tackling optimisation problems. They can be grouped in two families: constructive and perturbative search. Constructive search is a search paradigm in which the process starts from an empty candidate solution and iteratively instantiates variables one at a time, until a complete candidate solution is obtained. ACO and bee swarm optimisation are typical examples of constructive search methods. Perturbative search is a paradigm in which candidate solutions are iteratively perturbed by modifying the value of one or more variable in each search step. Tabu search, simulated annealing, iterated local search and GAs are typical examples of perturbative search methods (Hoos and Stützle 2004).

Measuring exploration is quite straightforward for the algorithms belonging to the first family: the value of each variable is chosen according to an explicitly defined probability distribution. This distribution only needs to be recorded for representing solutions. In the second family of approaches, instead, probabilities are in general assigned to complete solutions. In this case, we define the probability associated to each instantiation of variable $x_i, i = 1, \ldots, n$, as the sum of the probabilities of all the solutions in which x_i assumes the same value.

3. The algorithms implemented

For the experimental analysis proposed for showing the potential application of the measure of exploration described in Section 2, we implement two metaheuristic algorithms, namely *MAX–MIN* ant system and a GA. We compute the exploration they perform both with and without the introduction of a local search procedure. In this section we report a short description of the algorithms and of the local search procedures tested.

3.1. *MAX–MIN ant system*

In the first part of the experimental analysis described in this article, we apply the measure of exploration proposed for studying the behaviour of *MAX–MIN* ant system (*MM*AS) for the travelling salesman problem (TSP) (Stützle and Hoos 1997, 1998, 2000; Dorigo and Stützle 2004).

The idea at the basis of *MM*AS consists in exploiting artificial ants that construct solutions incrementally. The variables of the problems are typically mapped into binary variables, discretising the continuous space if necessary. At each step, an ant chooses stochastically one variable in a feasible set. The value of this variable is set to 1. If the constraints of the problem impose it, the value of other variables is set consequently. The probability of setting variable $x_i = 1$ is determined by the heuristic measure η_i and by the pheromone trail τ_i. In *MM*AS the pheromone associated to setting $x_i = 1$, $i = 1, \ldots, n$, is updated after the activity of each colony of m ants, according to

$$\tau_i = (1 - \rho)\tau_i + \Delta\tau_i^s, \tag{7}$$

where ρ is a parameter of the algorithm, $0 < \rho < 1$, named evaporation rate. Being solution s the best one - considering either the last iteration (iteration-best solution) the whole run (best-so-far solution) or the best since a reinitialisation of the pheromone trails (restart-best solution) (Stützle and Hoos 2000) - $\Delta\tau_i^s = 1/C_s$ if $x_i = 1$, and $\Delta\tau_i^s = 0$

otherwise. C_s is the cost associated to solution s. The schedule according to which the solution to be used is chosen, is described by Dorigo and Stützle (2004).

Pheromone trails in *MM*AS are bounded between $\tau_{\min}$ and $\tau_{\max}$. Following Dorigo and Stützle (2004), we set $\tau_{\max} = 1/(\rho C_{\text{best-so-far}})$, and $\tau_{\min} = [\tau_{\max}(1 - \sqrt[n]{0.05}]/[(\frac{n}{2} - 1)\sqrt[n]{0.05}]$. At the beginning of a run, the best solution corresponds to the one found by a problem specific heuristic. The pheromone trails are set equal to $\tau_{\max}$ for all variables.

Another element characterising *MM*AS is the random-proportional rule. In particular, let X_k be the set of variables that may be instantiated by ant k. Each component $x_i \in X_k$ has a probability of being chosen p_i:

$$p_i = \frac{\tau_i^\alpha \eta_i^\beta}{\sum_{j:x_j \in X_k} \tau_j^\alpha \eta_j^\beta}. \tag{8}$$

In the TSP, the problem instance is mapped to a graph in which nodes correspond to cities, and edges correspond to routes connecting cities. Variables represent edges of the graph: given a solution s, $x_i = 1$ if edge i belongs to s, and $x_i = 0$ otherwise. The heuristic measure η_i is set equal to the inverse of the cost of edge i. In the current implementation, the first population is generated using the nearest neighbour heuristic with randomised starting node.

The definition of probabilities described in Equation (8) may not represent the cumulated knowledge on the convenience of setting a variable equal to 1. Consider, for example, the case of the last variable instantiated in a solution: its probability is 1, independently on the product of pheromone trail and heuristic measure. In the computation of exploration, this situation is an inconvenience. In fact, the values of probabilities may depend more on randomness that affects solution construction than on the real indication of the algorithm.

In order to better represent this indication, the probability used for representing solutions is computed considering a different set for the normalisation of the product of pheromone and heuristic measure. The variables considered in this case are all the ones that would be alternative to each other if the partial solution under construction was empty: all the arcs exiting from a node are taken into account in the denominator.

3.2. *Genetic algorithm*

The second application of the method proposed for measuring exploration concerns a GA. The main procedure that characterises such an approach is inspired by the ability shown by populations of living beings to evolve and adapt to changing conditions, under the pressure of natural selection (Darwin 1859). This metaheuristic is based on the selection of individuals representing candidate solutions. From generation to generation, individuals evolve through *crossover* and *mutation*.

In the implementation considered here for tackling the TSP, the initial population is generated using the nearest neighbour heuristic with randomised starting node. At each iteration it is completely replaced by new individuals: *pop* new solutions are created starting from parent ones. Parents are chosen stochastically among the solutions belonging to the previous population with objective function value lower than or equal to the average. In this set, the probability associated to each individual is proportional to its fitness. According to a predefined probability p_{cross}, new individuals are obtained through crossover. The crossover operator consists in a procedure beginning with the

random selection of the starting element of the new solution. In the TSP the starting city is randomly selected. Then, the edges that, in the parent solutions, exit from this city are considered. One of them is randomly drawn. It may be the case in which they are both infeasible. In the TSP this happens if the ending node of both edges already belong to the new solution. Then, an edge is randomly selected among the feasible ones. The selection is based on uniform distributions. With probability $1 - p_{\text{cross}}$ the parent solutions are not modified. In addition to crossover, with probability p_{mut} the solutions are subjected to mutation, which is represented in this implementation by the swap of two consecutive cities in a randomly chosen position. Inserting edge i in solution s implies setting $x_i = 1$.

For computing the probability of setting each variable equal to 1, one must take into account all the potential parent solutions, with their probability of being actually selected. Then the probabilities p_{cross} and p_{mut} must be considered. In the crossover phase the algorithm is behaving as a typical constructive search procedure, actually attaching a probability to each edge. During mutation, instead, probability is attributed to solutions, and the method proposed in Section 2.3 for perturbative procedures is applied.

The mechanisms described have been selected without focusing on the efficiency of the algorithm. A multitude of possibilities are proposed in the literature for tackling the TSP (Radcliffe and Surry 1994, Potvin 1996, Back, Fogel, and Michalewicz 1997; Merz and Freisleben 2001). Nonetheless, obtaining state-of-the-art results is out of the scope of this article: the aim of this study is observing the characteristics of the exploration performed. The quality of the solutions obtained is considered only in terms of its relation with the emergent behaviour.

3.3. *Local search procedures*

In the literature, metaheuristics are very often hybridised with a local search procedure. Their performance are thus boosted through a problem-specific approach. Following this trend, we hybridise the algorithms described at the beginning of this section. We implement two classical local search approaches for the TSP, namely 2-opt and 3-opt. They are based on the idea of λ-optimality (Lin 1965): a solution is λ-optimal if it is not possible to improve it by replacing any λ of its edges. As the name itself tells, in 2-opt $\lambda = 2$ and in 3-opt $\lambda = 3$.

In the current implementation, local search is applied to all solutions generated at each iteration by *MM*AS and GA, respectively. The search is randomised: equal probability is associated to each feasible exchange. The number of possibilities tested is equal to the number of edges included in one solution (Dorigo and Stützle 2004). The first improvement pivoting rule is used (Hansen and Mladenović 2006). The search is fastened through the application of the *do not look bits* mechanism (Hoos and Stützle 2004).

4. Experimental analysis

The measure of exploration proposed in Section 2 is applied for analysing the behaviour of *MM*AS and GA for the TSP. The hybridisation of the same algorithms with two local search procedures is also tested. For *MM*AS, we use the ACOTSP program implemented by Thomas Stützle. The stopping criterion is the fulfillment of a fixed number of objective function evaluations. We perform runs of different duration. The computational resources available are equal for all the implementations tested. The code used for computing the

exploration is available on the web page http://www.paola.pellegrini.it. We solved ten instances from the TSPLIB repository. Here we report only the results achieved in instance `kroA200.tsp`. The conclusions that can be drawn are qualitatively equivalent in all the instances. In the following, we report the average value of exploration over 20 independent runs for each configuration and for each run-length. The difference among these runs is represented by the seed used for the random number generator. The variance of the values registered in the 20 runs is extremely low.

The algorithms are run with different parameter settings. According to the literature (Eshelman and Schaffer 1991; Wineberg and Oppacher 2003a; Pellegrini, Favaretto, and Moretti 2006; Bhattacharya 2008; Pellegrini and Ellero 2008), these values have an impact on the exploration. We start from a reference configuration, and we modify one element at a time. A configuration is a specific setting of all parameters. We analyse the impact of the various parameter settings on the exploration considering them independently from each other.

The parameters analysed for *MM*AS are α, β, ρ and m. The values $\alpha = 1$, $\beta = 3$, $\rho = 0.02$ and $m = 100$ represent the reference configuration. The ones tested are: $\alpha \in \{1, 2, 3, 4\}$, $\beta \in \{2, 3, 4, 5\}$, $\rho \in \{0.02, 0.05, 0.2, 0.7\}$, $m \in \{50, 100, 150, 200\}$. The parameters analysed for GA are p_{cross}, p_{mut} and *pop*. The values used in the reference configuration are 0.5, 0.5 and 100, respectively. The ones tested are: $p_{\text{cross}} \in \{0.1, 0.3, 0.5, 0.7\}$, $p_{\text{mut}} \in \{0.1, 0.3, 0.5, 0.7\}$ and $pop \in \{50, 100, 150, 200\}$.

4.1. *MAX–MIN ant system*

The exploration and the value of the best solution visited by various configurations of *MM*AS are depicted in Figure 2. We show the percentage error with respect to the optimal solution. Let us remark that, for example, if the graph concerns parameter α, then β, ρ and m are set as in the reference configuration, i.e. 3, 200 and 100, respectively. The graphs represent the number of objective function evaluations performed on the x-axis. In Figure 2(a), (c), (e) and (g), the y-axis reports the exploration. We introduce a logarithmic scale for a better visualisation of the results. In Figure 2(b), (d), (f) and (h), the y-axis represents the percentage error in terms of objective function value. The value of the threshold ϵ used in the clustering procedure is equal to the mean of the average distance between solutions, computed over the different run-lengths and configurations. It is set to 7.859.

Figure 2(a) shows the impact of different values of parameter α on the exploration. The higher the value of the parameter, the higher the exploration itself. The duration of the run appears quite relevant: in short runs, the difference in the level of exploration is very low. In longer ones, a clear distinction can be noticed. Finally, the separation of the curves decreases again. At the limit, if a very high number of objective function evaluations is allowed, the exploration is likely to become insensitive to the variation of α. This is suggested both by the intuition and the final crossing of the curves referred to $\alpha = 3$ and $\alpha = 4$ in the graph.

In Figure 2(b), we report the progressive value of the best solution found. In very short runs, the performance achieved by the different configurations is comparable. This comparability tends to re-emerge in the longest ones. When the run-length is intermediate, instead, the quality of the result increases as a function of the value of α. It follows an inverse trend with respect to exploration: the higher the exploration, the lower the cost of

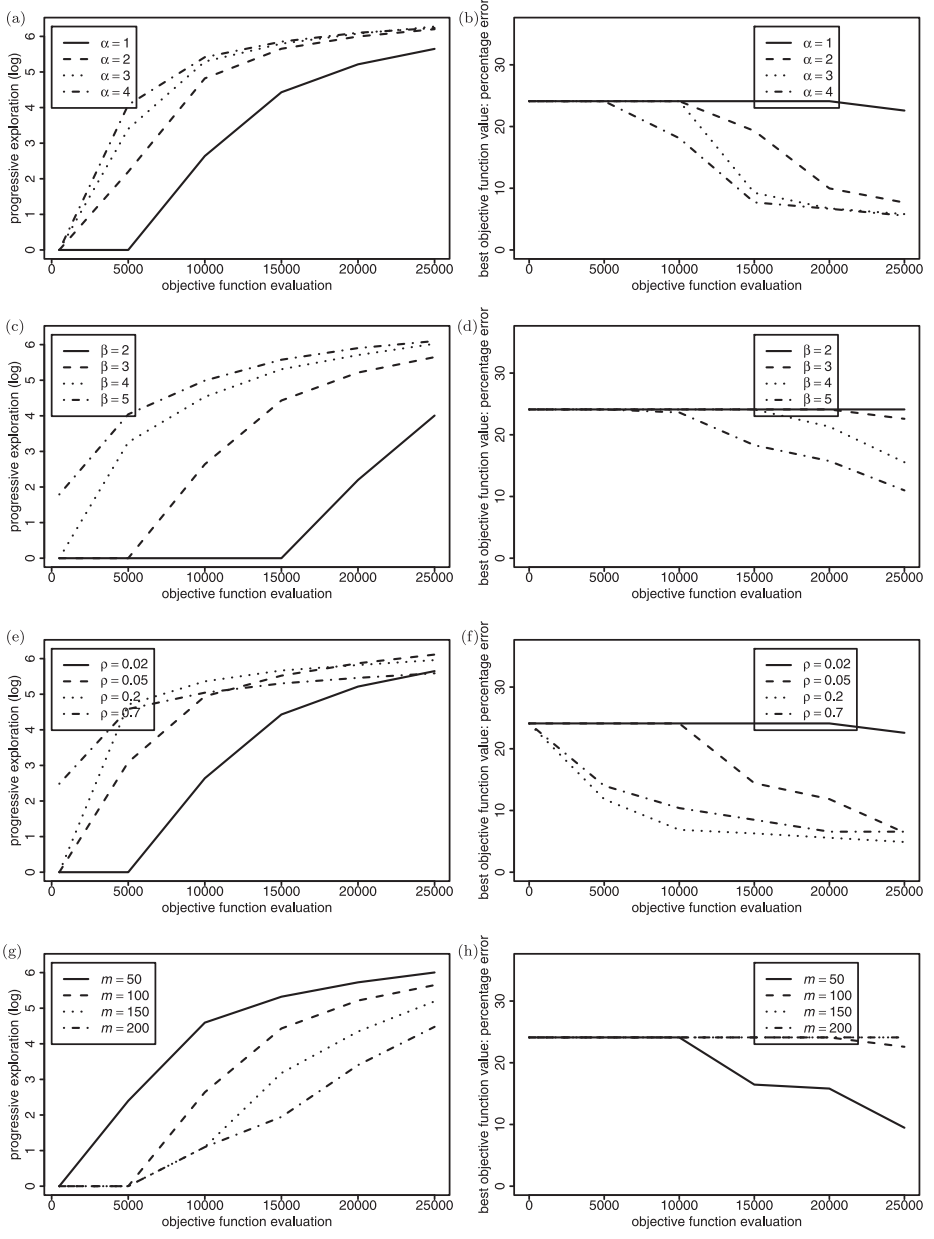

Figure 2. Exploration and value of of the best solution visited by *MM*AS with no local search. (a) Exploration with own $\epsilon - \alpha$. (b) Best solution value – α. (c) Exploration with own $\epsilon - \beta$. (d) Best solution value – β. (e) Exploration with own $\epsilon - \rho$. (f) Best solution value – ρ. (g) Exploration with own $\epsilon - m$. (h) Best solution value – m.

the best solution found, and thus the better the performance. This last observation, of course, is strictly related to the problem instances and to the computational resources considered. It cannot be taken as a general rule for obtaining good quality results in combinatorial optimisation.

Figure 2(c) and (d) report the same analysis for parameter β. The trend followed is very similar to the one just described. The impact of the value of this parameter on the exploration is quite strong. The increase of the magnitude of this impact as a function of run-length appears smoother than in the case of the different values tested for α. The value of the best so far solution varies consistently with the above observation.

The dependence of the exploration from the values of α and β is coherent with the role of these parameters: consider the ratio between the probabilities associated to two solution components x_i and x_j at a specific time instant

$$\frac{P_i}{P_j} = \left[\frac{\tau_i}{\tau_j}\right]^{\alpha}\left[\frac{\eta_i}{\eta_j}\right]^{\beta}. \tag{9}$$

The further from one the value resulting from Equation (9), the stronger the probability of selecting one component with respect to the other. During a run, the only variable elements in Equation (9) are τ_i and τ_j. A high value of parameter α implies that a slight change in the ratio between these pheromone trails, going from slightly below to slightly above one (or vice versa), implies a strong variation of the ratio between probabilities. A similar reasoning can be made for parameter β: a high value of this parameter amplifies a lot any variation of τ_i/τ_j. The different impact of α and β on the slope of the curve is due to the fact that β amplifies both the ratio of pheromone trails and the ratio of heuristic measures. The former varies in an iteration-by-iteration basis. The latter is constant throughout the run. This implies that the effect of the amplification of τ_i/τ_j imposed by α is very low until some significant differences are reached in the pheromone trails, i.e. after some pheromone updates. Instead, both τ_i/τ_j and η_i/η_j are amplified through β, and thus the impact can be observed immediately, thanks to the different values of the heuristic measure.

Figures 2(e) and (f) report the results achieved with different values of parameter ρ. The relation between this parameter and the exploration is more complicated than in the previous cases (Pellegrini et al. 2009). Up to a certain threshold the exploration is an increasing function of ρ. The opposite holds above this threshold. For very low values of ρ the amount of pheromone deposited and its evaporation are very low: several updates may be necessary before changing the ranking of the pheromone trails on the edges, and then before changing the area to be investigated with high probability. For $\rho = 0.02$ and $\rho = 0.05$ the curves representing exploration are convex: marginal exploration is higher for longer runs. The situation changes for high values of ρ: even with one single update, the evaporation and the deposit of pheromone are quite marked. The pheromone trails on the edges included in the best solutions, then, become much greater than the others in a very short time. The higher the value of ρ, the stronger the effect. This implies a smaller distance between solutions and a lower consequent exploration. For $\rho = 0.2$ and $\rho = 0.7$ the curves representing exploration are concave.

Finally, Figures 2(g) and (h) report that the exploration is decreasing with respect to the value of m. The greater m, the lower the number of iterations performed in a run. Consequently, the lower the number of different probability distributions used. A high value of m implies that the likely edges are often the same.

For what concerns the relation between exploration and performance, the observations made for parameters α and β are mostly respected both for parameters ρ and m.

4.2. *Genetic algorithm*

In Figure 3, we report the exploration and the value of the best solution visited by GA for the different values of the parameters. In addition, we show the percentage error with respect to the optimal solution. As in Figure 2, the graphs represent the number of objective function evaluations performed on the x-axis. The y-axis reports, in Figure 3(a), (c) and (e), the exploration expressed in logarithmic scale, and, in Figure 3(b), (d) and (f), the percentage error in terms of objective function value. The threshold used as stopping criterion for the clustering procedure is $\epsilon = 1.003$.

The impact of parameter p_{cross} on the exploration is quite evident in Figure 3(a). In particular, the higher the value of the parameter, the lower the exploration.

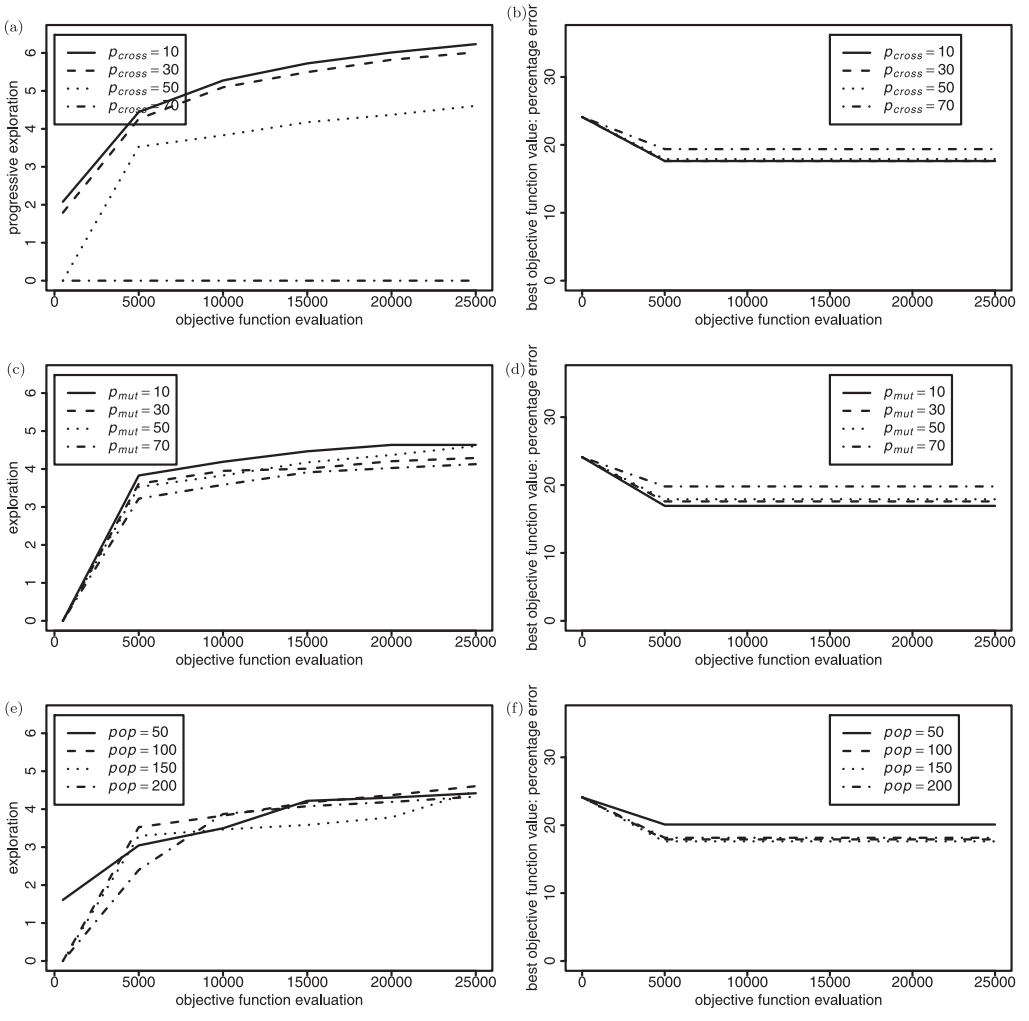

Figure 3. Exploration and value of of the best solution visited by GA with no local search. (a) Exploration with own ϵ – P_{cross}. (b) Best solution value – P_{cross}. (c) Exploration with own ϵ – p_{mut}. (d) Best solution value – p_{mut}. (e) Exploration with own ϵ – *pop*. (f) Best solution value – *pop*.

This observation is strictly linked with the specific crossover operator included in the algorithm. This operator implies that a high number of components are similarly likely to be chosen. Their probability, then, is very small. As a consequence, if p_{cross} is high, several solutions with low positive coordinates (equal to the probability $P_i(1)$, $i = 1, \ldots, n$) are visited. Their distance is, consequently, quite small. The impact of this parameter is not very marked in terms of performance.

Figure 3(c)–(f) depict the analysis for p_{mut} and *pop*. The trend of the performances is very similar to the one observed for p_{cross}. The exploration does not appear very sensitive to the values of these parameters. In fact, the mutation operator implemented here is not extremely powerful. On the other hand, for what concerns the impact of parameter *pop*, remark that the values of both parameters p_{cross} and p_{mut} set in the reference configuration is 0.5. Intuitively, then, an extreme diversity between subsequent populations is not likely. Hence, observing a set of solutions constituting a unique generation will not be extremely different from observing a set of solutions with the same cardinality, but corresponding to two subsequent evolution steps.

The GA implemented tends to explore quite extensively a small area of the search space. This behaviour allows to find quite early a local minimum, but then the search stagnates with no further significant improvement.

4.3. *Hybridisation with local search*

Figures 4 and 5 depict the analysis on *MM*AS ant system hybridised with 2-opt and 3-opt local search, respectively. The threshold ϵ is equal to 17.520 in the first case and 35.842 in the second. The influence of parameters is quite low in terms of both exploration and quality of the results. The same conclusion can be drawn by observing the results obtained hybridising the GA with local-search, reported in Figures 6 and 7. The respective values of ϵ are 16.564 and 46.588.

The sensitivity of the algorithms to the values of parameters is strongly reduced when a local search procedure is applied. Moreover, the regularity of the trend followed by the exploration as a function of parameter values is not clear under these experimental conditions. This result is due to two main reasons. First of all, the type of exploration during the local search runs is the same regardless of the values of the parameters of either *MM*AS or GA. Second, the hybridisation implies that the role of parameters cannot be defined as clearly as in the implementation with no local search. For example, let us consider parameter ρ in *MM*AS. If no local search is applied, ants use with high-probability solution components on which a high level of pheromone is present. With high probability the solution used for pheromone deposit according to Equation (7) includes such components. Thus, pheromone is gradually increased on the components that appear to be convenient, and it is gradually decreased on the others. The role of parameter ρ is controlling this gradualness. If local search is applied, instead, even if ants use component with a high level of pheromone, the exploration of the neighbourhood through either 2-opt or 3-opt may imply the inclusion of several components with a low pheromone level in the final solution, with the exclusion of several others with a high pheromone level. Hence, the gradualness in the variation of pheromone trails may be lost, and the role of parameter ρ becomes less clear. The absence of a clear trend of exploration as a function of the value of this parameter supports this intuition.

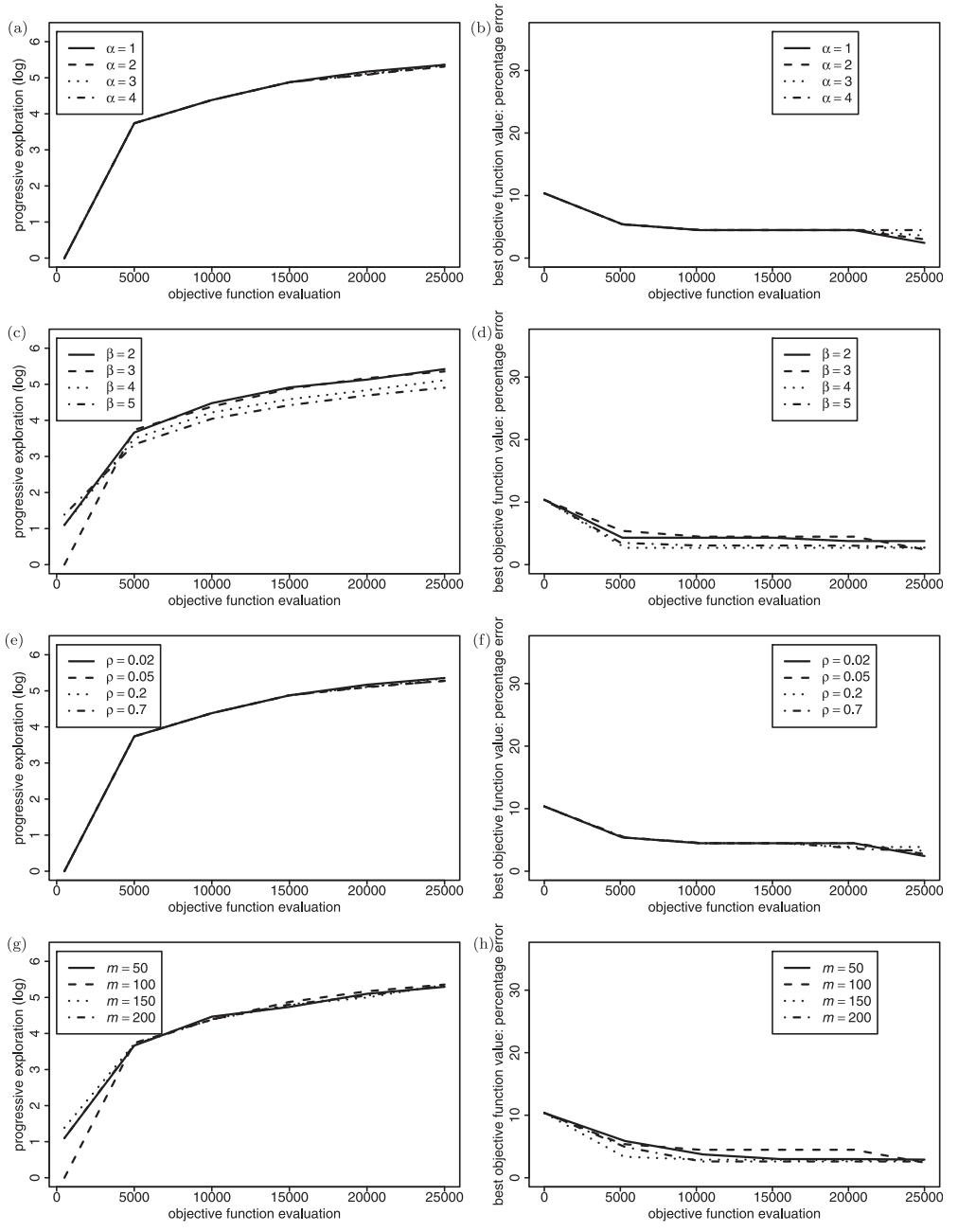

Figure 4. Exploration and value of of the best solution visited by *MM*AS with 2-opt local search. (a) Exploration with own $\epsilon - \alpha$. (b) Best solution value – α. (c) Exploration with own $\epsilon - \beta$. (d) Best solution value – β. (e) Exploration with own $\epsilon - \rho$. (f) Best solution value – ρ. (g) Exploration with own $\epsilon - m$. (h) Best solution value – m.

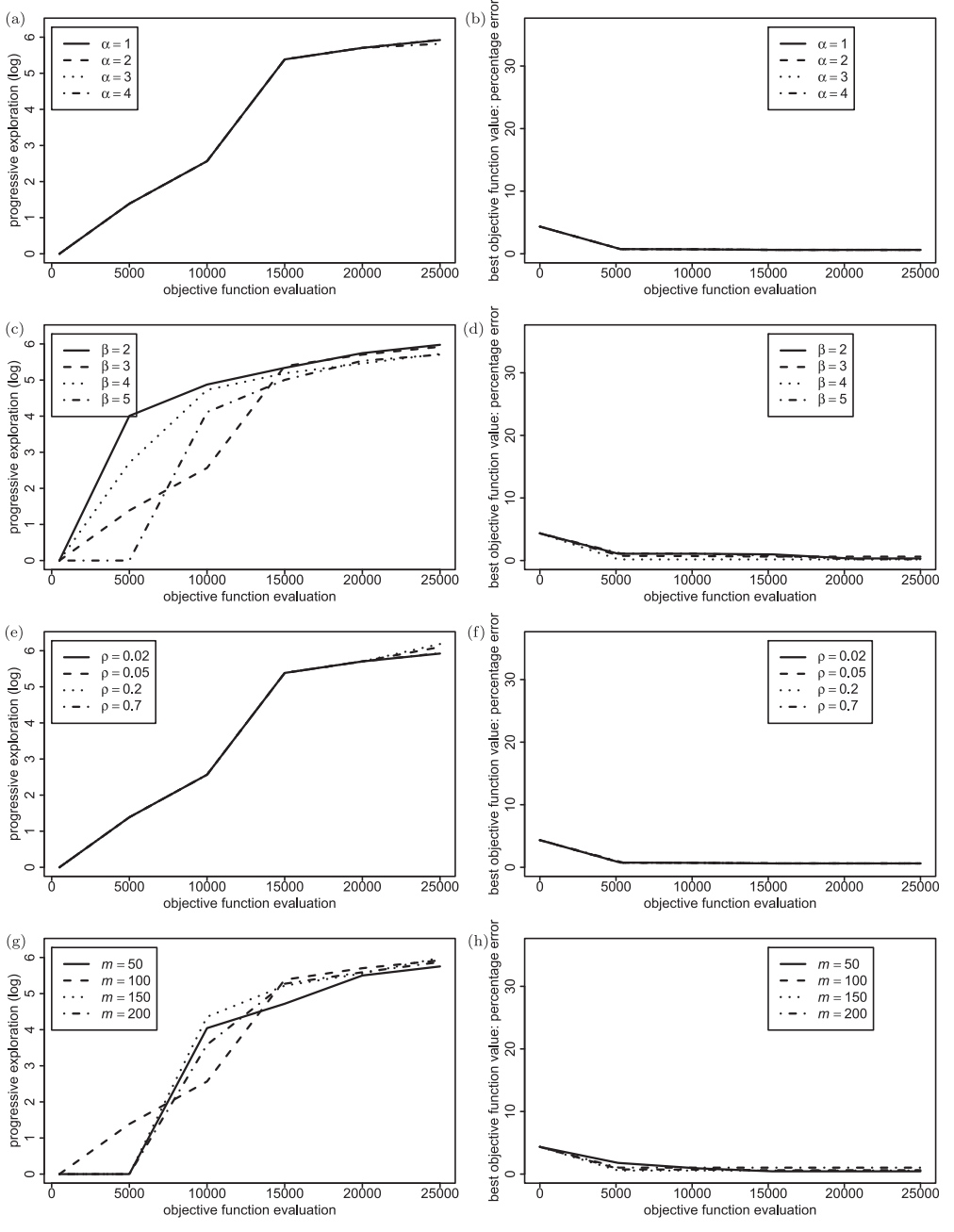

Figure 5. Exploration and value of of the best solution visited by *MM*AS with 3-opt local search. (a) Exploration with own $\epsilon - \alpha$. (b) Best solution value – α. (c) Exploration with own $\epsilon - \beta$. (d) Best solution value – β. (e) Exploration with own $\epsilon - \rho$. (f) Best solution value – ρ. (g) Exploration with own $\epsilon - m$. (h) Best solution value – m.

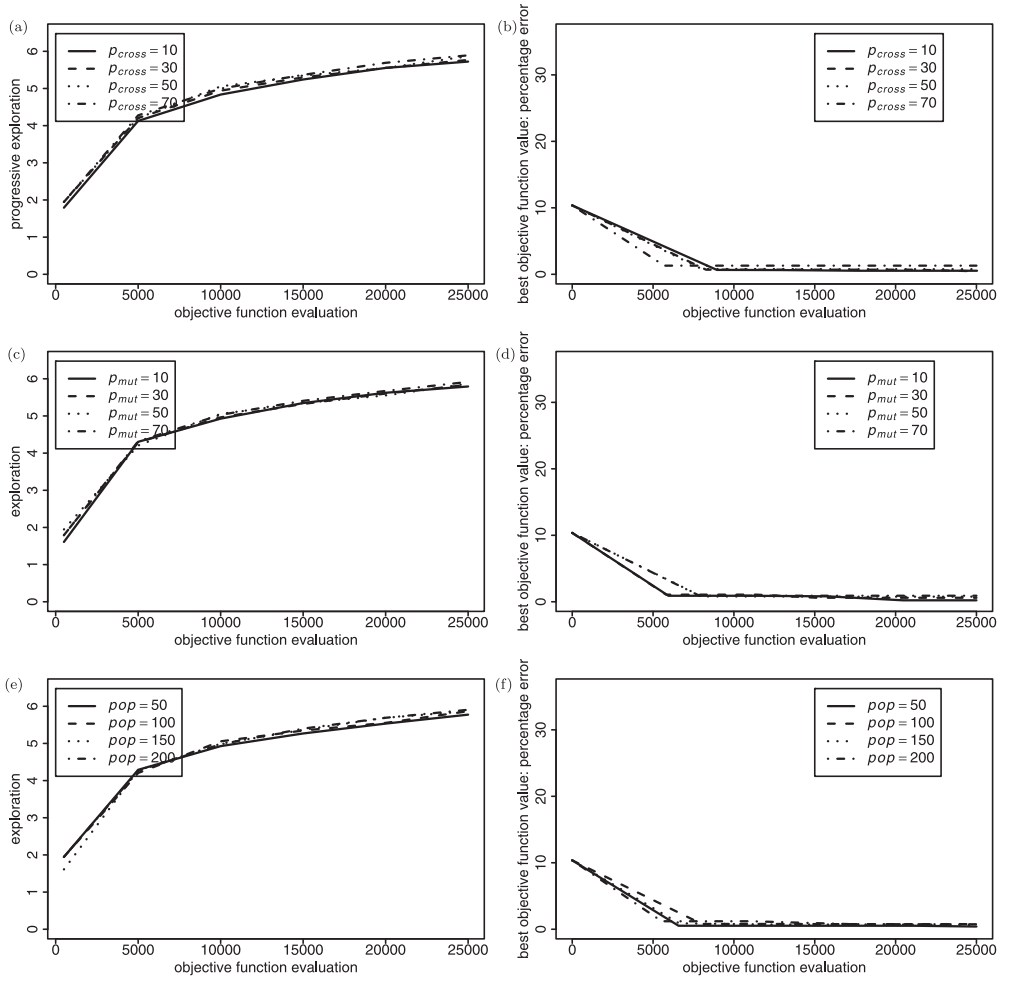

Figure 6. Exploration and value of of the best solution visited by GA with 2-opt local search. (a) Exploration with own ϵ – P_{cross}. (b) Best solution value – P_{cross}. (c) Exploration with own ϵ – P_{mut}. (d) Best solution value – P_{mut}. (e) Exploration with own ϵ – *pop*. (f) Best solution value – *pop*.

4.4. *Comparison of different configurations*

Through the measure proposed, it is possible to compare the exploration performed by different metaheuristic algorithms. In the experimental analysis described in this section, *MM*AS and GA are analysed jointly, in the implementations with no local search, with 2-opt and 3-opt. The value of ϵ is the mean over all the run-length, all the algorithms and all the implementations. It is set to 18.813.

Figure 8 reports the exploration performed and the corresponding value of the best solution visited. For ease of visualisation, we depict only the behaviour of the reference configurations. The analysis of the other configurations tested suggests the same conclusions. In particular, the exploration performed by the 3-opt implementations

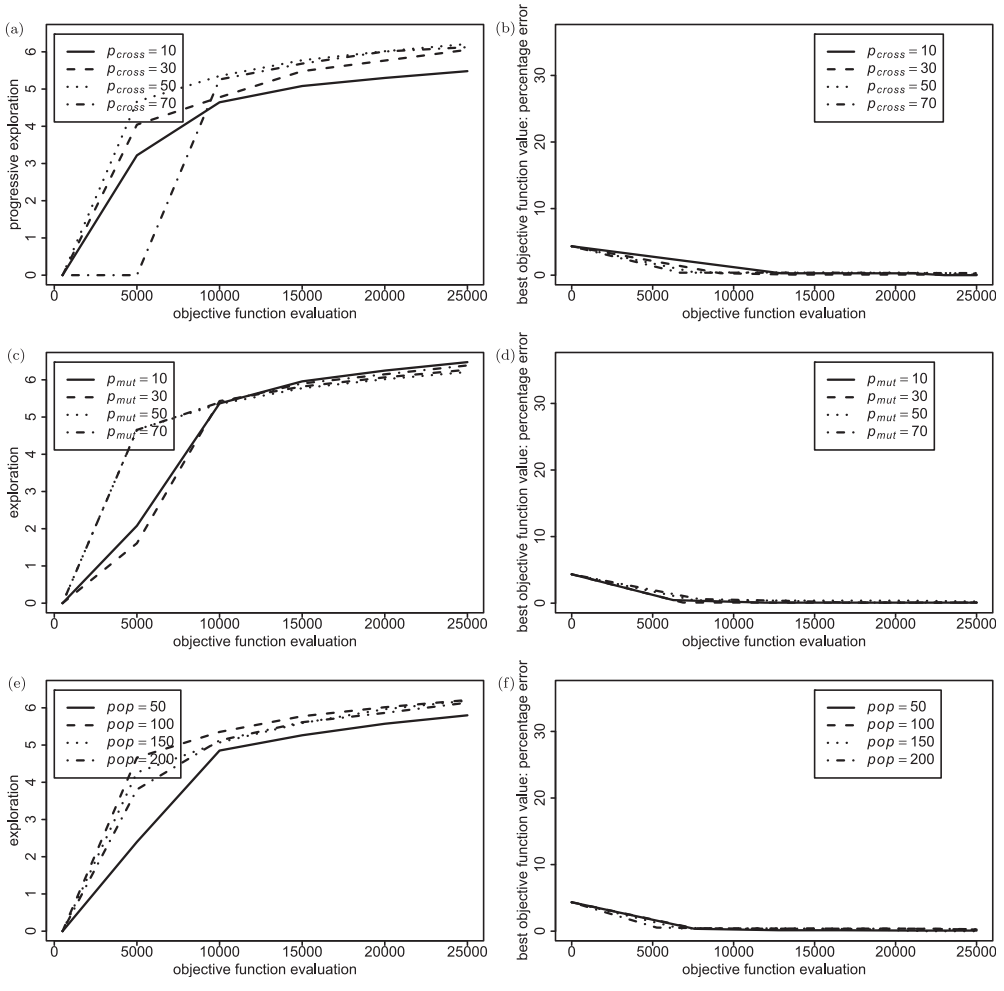

Figure 7. Exploration and value of of the best solution visited by GA with 3-opt local search. (a) Exploration with own ϵ – P_{cross}. (b) Best solution value – P_{cross}. (c) Exploration with own ϵ – P_{mut}. (d) Best solution value – P_{mut}. (e) Exploration with own ϵ – *pop*. (f) Best solution value – *pop*.

is the highest, followed by the 2-opt ones. When no local search is applied, the exploration performed by both *MM*AS and GA is negligible, if compared to the implementations including local search. 3-opt eliminates the difference between the exploration of the two algorithms. When 2-opt is implemented, instead, GA explores more than *MM*AS. Figure 8(b) shows that the performance of GA with 2-opt is better than *MM*AS hybridised with the same local search.

5. Conclusions

In this article we propose a definition of exploration of the search space that helps in analysing the behaviour of metaheuristic algorithms, independently on the specific

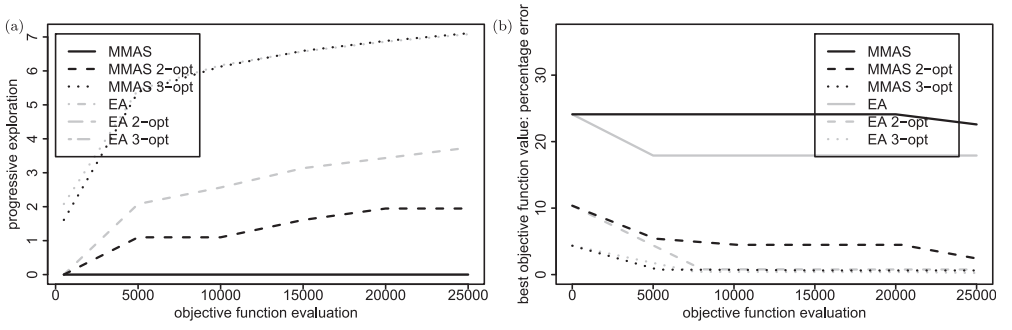

Figure 8. Exploration and value of of the best solution visited by *MM*AS and GA with no local search, with 2-opt and with 3-opt. We report the results achieved with the reference configurations. (a) Exploration with common own ϵ. (b) Best solution value.

approach considered. We describe the steps that must be undertaken for translating this definition into a practically usable tool, and we show how to apply it. We implement an ACO algorithm and a GA for the TSP. We assess the impact on the exploration of the values of the main parameters of the algorithms, evaluating the different behaviours in absence of any local search procedure, or when either the 2-opt or the 3-opt are implemented. The results achieved support the intuitions on the impact of parameter values and of local search on the behaviour of the algorithms.

Measuring exploration with the novel methodology presented consents to point out some connections between the behaviour of the algorithms and their performance. In fact, the results obtained show that the higher the exploration, the better the performance. Of course, this particular conclusion is strictly dependent on the problem instances and on the computational resources available. In general, the best results are obtained by finding the proper balance of exploration and exploitation. The measure proposed in this article can be used for quantifying this concepts.

The results reported are not surprising for practitioners and researchers, experts in the field of ACO and GA. Nonetheless, predictability does not coincide with obviousness. A measure for testing the validity of predictions is extremely important in the field of metaheuristics, where theoretical proofs are often unavailable. Furthermore, an objective method for computing the exploration of metaheuristics helps in liberalising the research in this field. In fact, for gaining the expertise that is necessary for understanding the behaviour of a metaheuristic in the absence of a specific measure, requires a long time and a lot of effort devoted to the analysis of the algorithm. Thanks to a formalised method as the one proposed in this article, an implementer may skip this costly analysis, at least in some part, and he may base his reasoning on hypothesis supported by experimental results.

The research presented in this article is to be read in the context of a quite ambitious project: it aims at giving a tool for getting some insight in the behaviour of metaheuristics. Bezdek, Keller, Krishnapuram, and Nikhil (1999) state that in optimisation sometimes we do not know why things work the way they do, but we should be happy if they work right this time. Observing the peculiar behaviour of metaheuristics in terms of exploration may

help in reducing the unawareness on the reasons why they do or do not work properly under some given conditions.

This research offers two main hints for future developments. On the one hand, the high-level analysis of the behaviour of metaheuristics may be further elaborated by finding a functional relation linking the exploration performed, the topology of the search space and the performance of the algorithm throughout the run. Finding such a functional relation would allow making the most of the algorithms in virtually any optimisation framework. The quantification of exploration represents a first step in this direction. On the other hand, the measure of exploration we have defined may find a direct application in the design of the algorithm. With this measure, the designer may control the behaviour of the algorithm in various phases of the search, solving effectively the trade-off between exploration and exploitation. This is typically done through the on-line parameter adaptation (Eiben, Michalewicz, Schoenauer, and Smith 2007; Stützle et al. 2011). These adaptation methods adjust the behaviour of the algorithm in response to the current state of the search, often using the value of the objective function associated to the solutions visited as a proxy of the current exploration. Replacing the proxy with the measure proposed in this article might improve significantly the performance of these methods.

References

Amor, H.B., and Rettinger, A. (2005), 'Intelligent Exploration for Genetic Algorithms: Using Self-organizing Maps in Evolutionary Computation', in *GECCO '05: Proceedings of the 2005 Conference on Genetic and Evolutionary Computation*, New York: ACM, pp. 1531–1538.

Back, T., Fogel, D.B., and Michalewicz, Z. (1997), *Handbook of Evolutionary Computation*, Bristol, UK: IOP Publishing Ltd.

Bezdek, J.C., Keller, J., Krisnapuram, R., and Nikhil, R.P. (1999), *Fuzzy Models and Algorithms for Pattern Recognition and Image Processing*, Norwell, MA: Kluwer Academic Publishers.

Bhattacharya, M. (2008), 'A Synergistic Approach for Evolutionary Optimization', in *GECCO '08: Proceedings of the 2008 GECCO Conference Companion on Genetic and Evolutionary Computation*, New York: ACM, pp. 2105–2110.

Blum, C., and Roli, A. (2003), 'Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison', *ACM Computing Surveys*, 35, 268–308.

Darwin, C.R. (1859), *On The Origin of Species by Means of Natural Selection, or the Preservation of Favoured Races in the Struggle for Life*, London: John Murray.

Desai, M.P. (1999), 'Some results Characterizing the Finite time Behaviour of the Simulated Annealing Algorithm', *Sadhana*, 24, 317–337.

Devarenne, I., Mabed, H., and Caminada, A. (2006), 'Intelligent Neighborhood Exploration in Local Search Heuristics', in *ICTAI '06: Proceedings of the 18th IEEE International Conference on Tools with Artificial Intelligence*, Washington, DC: IEEE Computer Society, pp. 144–150.

Dorigo, M., and Stützle, T. (2004), *Ant Colony Optimization*, Cambridge, MA, USA: MIT Press.

Eiben, A.E., Michalewicz, Z., Schoenauer, M., and Smith, J.E. (2007), 'Parameter Control in Evolutionary Algorithms', in *Parameter Setting in Evolutionary Algorithms*, Studies in Computational Intelligence (Vol. 54), eds. F.G. Lobo, C.F. Lima and Z. Michalewicz, Berlin, Germany: Springer Verlag, pp. 19–46.

Eshelman, L.J., and Schaffer, J.D. (1991), 'Preventing Premature Convergence in Genetic Algorithms by Preventing Incest', *ICGA Proceedings of the 4th International Conference on Genetic Algorithms*, 115–122.

Hansen, P., and Mladenović, N. (2006), 'First vs. Best Improvement: An Empirical Study', *Discrete Applied Mathematics*, 154, 802–817.

Herault, L. (2000), 'Rescaled Simulated Annealing-accelerating Convergence of Simulated Annealing by Rescaling the States Energies', *Journal of Heuristics*, 6, 215–252.

Hoos, H.H., and Stützle, T. (2004), *Stochastic Local Search. Foundations and Applications*, San Francisco, CA, USA: Morgan Kaufmann Publishers.

Jardine, N., and Sibson, R. (1968), 'The Construction of Hierarchic and Non-Hierarchic Classifications', *The Computer Journal*, 11, 177–184.

Lin, S. (1965), 'Computer Solutions of the Traveling Salesman Problem', *Bell System Technical Journal*, 44, 2245–2269.

Merkle, D., and Middendorf, M. (2001), 'On the Behaviour of aco Algorithms: Studies on Simple Problems', in *Proceedings of MIC'2001 Meta-heuristics International Conference*, pp. 573–577.

Merz, P., and Freisleben, B. (2001), 'Memetic Algorithms for the Traveling Salesman Problem', *Complex Systems*, 13, 297–345.

Orosz, J.E., and Jacobson, S.H. (2002), 'Finite-time Performance Analysis of Static Simulated Annealing Algorithms', *Computational Optimization and Applications*, 21, 21–53.

Pellegrini, P., Favaretto, D., and Moretti, E. (2006), 'On *MAX–MIN* ant system's parameters', in *Ant Colony Optimization and Swarm Intelligence, 5th International Workshop, ANTS 2006* (Vol. 4150 of *LNCS*), eds. M. Dorigo, L.M. Gambardella, M. Birattari, A. Martinoli, R. Poli and T. Stützle, Berlin, Germany: Springer Verlag, pp. 203–214.

Pellegrini, P., and Ellero, A. (2008), 'The Small World of Pheromone Trails', in *ANTS '08: Proceedings of the 6th International Conference on Ant Colony Optimization and Swarm Intelligence*, Berlin, Germany: Springer Verlag, pp. 387–394.

Pellegrini, P., Favaretto, D., and Moretti, E. (2009), 'Exploration in Stochastic Algorithms: An Application on *MAX–MIN* Ant System', in *Nature Inspired Cooperative Strategies for Optimization (NICSO 2008)*, Studies in Computational Intelligence (Vol. 236), eds. N. Krasnogor, B. Melián-Batista, J.A. Moreno-Pérez, J.M. Moreno-Vega and D.A. Pelta, Berlin, Germany: Springer Verlag, pp. 1–13.

Potvin, J.Y. (1996), 'Genetic Algorithms for the Traveling Salesman Problem', *Annals of Operations Research*, 63, 339–370.

Radcliffe, N.J., and Surry, P.D. (1994), 'Formal Memetic Algorithms', in *Evolutionary Computing: AISB Workshop* (Vol. 865 of *LNCS*), ed. T. Fogarty, Berlin, Germany: Springer Verlag, pp. 1–16.

Schuurmans, D., and Southey, F. (2001), 'Local Search Characteristics of Incomplete Sat Procedures', *Artificial Intelligence Journal*, 132, 121–150.

Stützle, T., and Hoos, H.H. (1997), 'The *MAX–MIN* Ant System and Local Search for the Traveling Salesman Problem', in *Proceedings of the 1997 IEEE International Conference on Evolutionary Computation* (*ICEC*'97), eds, Bäck, T., Michalewicz, Z., and Yao, X., Piscataway, NJ: IEEE Press, pp. 309–314.

Stützle, T., and Hoos, H.H. (1998), 'Improvements on the Ant System: Introducing the *MAX–MIN* Ant System', in *Artificial Neural Networks and Genetic Algorithms*, eds. R.F. Albrecht, G.D. Smith and N.C. Steele, Vienna, Austria: Springer Verlag, pp. 245–249.

Stützle, T., and and Hoos, H.H. (2000), '*MAX–MIN* ant System', *Future Generation Computer Systems*, 16, 889–914.

Stützle, T., López-Ibánez, M., Pellegrini, P., Maur, M., Montes de Oca, M.A., Birattari, M., and Dorigo, M. (2012), 'Parameter Adaptation in Ant Colony Optimization', in *Autonomous Search*, eds. Y. Hamadi, et al., Berlin, Germany: Springer, pp. 191–215.

Watson, J.P., Whitley, L.D., and Howe, A.E. (2005), 'Linking Search Space Srtucture, Run-Time Dynamics, and Problem Difficulty: A Step Toward Demystifying Tabu Search', *Journal of Artificial Intelligence Research*, 24, 221–261.

Wineberg, M., and Oppacher, F. (2003a), 'Distance Between Populations', *Genetic and Evolutionary Computation Conference (GECCO-2003)*, 1481–1492.

Wineberg, M., and Oppacher, F. (2003b), 'The Underlying Similarity of Diversity Measures Used in Evolutionary Computation', *Genetic and Evolutionary Computation Conference (GECCO-2003)*, 1493–1504.

Wolpert, D.H., and Macready, W.G. (1997), 'No free Lunch Theorems for Optimization', *IEEE Transactions on Evolutionary Computation*, 1, 67–82.