10.1063/1.2817338

# Monkey search: a novel metaheuristic search for global optimization

Antonio Mucherino and Onur Seref

*Center for Applied Optimization, University of Florida, Florida, USA*

**Abstract.** We propose a novel metaheuristic search for global optimization inspired by the behavior of a monkey climbing trees looking for food. The tree branches are represented as perturbations between two neighboring feasible solutions of the considered global optimization problem. The monkey mark and update these branches leading to good solutions as it climbs up and down the tree. A wide selection of perturbations can be applied based on other metaheuristic methods for global optimization. We show that Monkey Search is competitive compared to the other metaheuristic methods for optimizing Lennard-Jones and Morse clusters, and for simulating protein molecules based on a geometric model for protein folding.

**Keywords:** Metaheuristics, Global Optimization, Protein Folding
**PACS:** 87.55.de, 87.55.kd, 87.15.Cc

## 1. INTRODUCTION

Many meta-heuristic search methods have been developed over the years for solving difficult global or combinational optimization problems. Many of these methods were inspired by optimal behaviors in nature. These methods are designed to find the global optimum solution of a hard optimization problem. The main challenge these methods face is the existence of multiple local minima of these optimization problems, which the methods that are not well designed usually converge to and stop. The success of these search methods rely on their handling such local minima and converging towards a solution which is close to the global solution.

In the field of meta-heuristic optimization, many methods have been proposed, which are inspired by natural phenomena or animal behavior. For instance, the Simulated Annealing (SA) algorithm [10] simulates the physical annealing process by decreasing the temperature of a given system to reach a stable low-energy configuration, which can correspond to the global optimum of an optimization problem. The Genetic Algorithms (GA) [8] mimic the evolution of a population of solutions. As the Darwinian Theory suggests, good solutions with better objective function values are allowed to survive and create new children by the crossover mechanism, which may even undergo mutations to avoid local minima. The solutions with worse objective function values are destroyed by natural selection. Ant Colony (AC) algorithms [4] simulate the behavior of a colony of ants that searches for food. Since each ant leaves a chemical called pheromone to mark its path, a colony in which each ant seems to work randomly begins to follow an optimal path as soon as there is enough pheromone on it. The Harmony Search (HS) [7] simulates musical improvisation, in which the optimal harmony is sought by improvising on the notes of previously played harmonies . The problem with some

CP953, *Data Mining, Systems Analysis, and Optimization in Biomedicine*
edited by O. Seref, O. E. Kundakcioglu, P. M. Pardalos

global optimization problems is that the gradient of the objective function is hard to calculate. The Differential Evolution (DE) algorithm [19] extend the concept of gradient by defining the direction which leads one feasible solution to another. The DE algorithm works with a population of solutions and has similarities with GAs.

In this study, we propose a new method based on an agent discovering and searching trees of solutions for solving global optimization problems. The general behavior of the search agent resembles a monkey looking for food by climbing up trees, from which the method acquires its name. The basic assumption for such behavior is that the monkey explores trees and learns which branches lead to better food resources. In our search method, food is represented by desirable solutions and the branches of the tree of solutions are represented as perturbations between two neighboring feasible solutions in the considered optimization problem. These perturbations can be totally random; however they can also be customized and based on other strategies previously developed for global optimization. This structure of our search is not meant to be a method for global optimization to be compared to the other aforementioned methods, it rather exploits the perturbation mechanisms of these methods in defining its branches.

The movement of the monkey is up and down the branches of a tree. Unknown branches are discovered using perturbations and whenever a better solution is found, the monkey climbs down the unique path that connects this solution to the root, while marking the path with this better solution. After reaching the root, the monkey climbs back up the tree using the previously discovered branches. The path the monkey follows on the way up the tree is probabilistically determined at each branch, based on the previous marks, which is eventually intended to lead the monkey to branches with better solutions. The monkey climbs until it reaches an undiscovered frontier, and continues its search from there on with new unknown branches.

In this study, we test the Monkey Search method on the global optimization problem of finding stable conformations of clusters of atoms regulated by the *Lennard Jones* potential energy [11] and by the *Morse* potential energy [13]. These difficult to optimize energy functions have been already considered for testing methods for optimization [5, 6, 12, 16, 17, 20, 21]. We also consider an optimization problem arising from a model developed for studying the protein folding problem, which has a growing need of fast and efficient optimization methods. This model is called the "tube model" due to its protein representation, and it simulates protein conformations based on geometric features of proteins [1, 3, 9].

The chapter is organized as follows. Section 2 provides a detailed description of the Monkey Search method. In Section 3 computational results and comparisons between Monkey Search and other metaheuristics are presented on Lennard-Jones and Morse clusters, and also on the tube model. Finally in Section 4, conclusions are drawn.

## 2. MONKEY SEARCH METHOD

Monkey Search (MS) is an agent based search algorithm, in which the searching agent discovers branches of a tree of solutions [18]. This behavior is not an exact replica of the behavior of monkeys in real life, however the general behavior of the search agent resembles a monkey wandering trees in search of food.

The monkey in our algorithm randomly climbs and discovers new branches of a tree. Each time a better solution is found, the monkey climbs down the unique path that connects the solution to the root, and marks this path with this solution value. The monkey then climbs up the tree again, choosing the previously marked branches randomly with higher probability towards the branches with better values, until it reaches an undiscovered frontier of the tree. The behavior of preferring branches with better values is intended for the monkey to converge to a specific area of the tree in which the global solution may be discovered. The monkey is not limited with a single tree; the search domain for the objective function may rather be considered as a jungle.

The branches of trees contain feasible solutions. Starting from any solution, which is at the root of a branch, a new neighbor solution is given at the tip of the same branch. The functional distance between the two solutions is determined by the random perturbation. The trees have a binary tree structure, which means that there are two new branches extending from each branch, each with a neighbor solution. Each time the monkey finds a solution with a better objective function value, it stops climbing and stores this solution as the current best solution. The monkey also stops when it reaches the top of the tree, which is the depth of the tree. Whenever either stopping condition is realized, the monkey starts climbing down branch by branch and looking for some other tree branch which could lead to better solutions. On its way back, the monkey updates and marks all the branches it passes for later exploration. The tree is considered to contain a preset number of paths from the root that reach the entire height of the tree. Each time the monkey reaches the allowed height of the tree, one of such paths is assumed to be explored. When all such paths are explored, the monkey starts climbing a new tree starting from the best solution, or a combination of better solutions.

At each step of the algorithm, new feasible solutions are created starting from the current one, and using a randomized perturbation function, which generates a new solution when applied to the current one. We aim to exploit other metaheuristic searches in defining these perturbation functions. For instance, one of the perturbations we apply is the crossover between solutions from genetic algorithms. The other perturbations we apply are inspired by other metaheuristics mentioned in the Introduction. However, perturbation functions are not limited by other heuristics, and can be derived from the optimization problem itself, leading to a wide domain of applications for Monkey Search.

At the end of each branch while climbing up, the monkey chooses between the left or the right branch. Therefore, at each step, we need to apply two different perturbations to the current solution, and the perturbation to apply can be chosen among a set of perturbations adaptively to provide more flexibility. The choice of perturbations may start uniformly, and converge towards those with higher success percentage. For avoiding that some kind of perturbation becomes predominant, the choice of perturbations is randomly reset back to uniform.

The immediate information available to the monkey at any point is the current solution $X_{cur}$, and the best solution $X_{best}$ on the current tree. Both these solutions can be used in the perturbations for generating new solutions. The perturbations we apply in Monkey Search in this study are as follows:

- Random changes to $X_{cur}$, as in the Simulated Annealing methods;

- Crossover operator applied for generating a child solution from the parents $X_{cur}$ and $X_{best}$, as in Genetic Algorithms;
- The mean solution built from $X_{cur}$ and $X_{best}$, inspired by Ant Colonization;
- Directions that lead $X_{cur}$ to $X_{best}$, as in Directional Evolution;
- Creating harmonies (solutions) from $X_{cur}$ and $X_{best}$ and introducing random notes, as in Harmony Search;

Note that our perturbations do not consider a population of solutions like in in Genetic Algorithms. They consider $X_{cur}$ and $X_{best}$ only, which assumes the existence of a predominant individual, $X_{best}$, which may be the case in some species such as in a pride of lions.

For avoiding local optima, we keep a predetermined number of $n_m$ best solutions updated by each successive tree. Since we use the solutions in this set frequently, we call this set the *memory* of our search algorithm. In order to force the search in different spaces of the search domain, we start with $n_w$ trees with randomly generated solutions as root. Usually $n_m < n_w$, and we keep the best $n_m$ of them in the memory. After the first $n_w$ trees, the root solution of a new tree is randomly chosen from the memory. We update the memory whenever a new solution is better than the ones that are already in the set, by including this solution to the memory and removing the closer solution in memory that is worse than the new one. In order to avoid local optima, if the new solution is too similar to an existing solution in the memory, it is ignored.

Monkey Search stops when the difference between the objective function values of all solutions in the memory is below a threshold. It is worth to note that Monkey Search is based on the following parameters which influence the convergence of our algorithm: the height $h_t$ of the trees, the number $n_t$ of times the monkey reaches the top of the tree, the size of the memory $n_m$, and the starting number $n_w$ of random trees.

## 3. APPLICATIONS AND COMPUTATIONAL RESULTS

In this section, we present the application of Monkey Search on two sets of biomedical problems. The first set of problems involve Lennard-Jones and Morse potential energy functions for cluster conformations, both of which are well-known hard global optimization problems. The second application is the tube model based solely on the number of amino acids in a protein that determines a folding pattern from its mechanical properties. Monkey Search is applied to different problem sizes of both problem sets and compared to other metaheuristics.

### 3.1. Lennard-Jones and Morse clusters

Finding the ground state conformations of clusters of atoms represents one of the hardest global optimization problems. Many researches have been studying the so-called Lennard Jones (LJ) clusters [12, 16, 20] and Morse clusters [5, 6], previously used for testing global optimization methods [6].

The LJ energy is very simple to explain but only sufficiently accurate for the noble gases, and it also provides approximations for the structures of nickel and gold clusters. The LJ pairwise potential is given conventionally in reduced terms

$$v(r) = \frac{1}{r^{12}} - \frac{2}{r^6},$$

where $r$ is the inter-particle Euclidean distance between any two atoms of the cluster. All the atoms are considered to be identical. The problem is to find the global minimum of the potential energy defined by

$$V_{LJ} = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} v(r_{ij}),$$

where $r_{ij}$ is the inter-particle distance between atoms $i$ and $j$. Note that the $V_{LJ}$ values do not change by translating or rotating the clusters. This simple function is not easy to optimize, because it has high number of local minima. For instance, a cluster containing $n = 13$ atoms has about 1000 local minima, and the number of these local minima grows faster as $n$ increases [20]. It has been proven in [17] that, when the size of the cluster is less than 1600, the stable structures have an icosahedral structure, even if there are exceptions, in which the clusters have lower energy when they have a decahedral or octahedral structure.

The Morse potential is a simple model, and, compared to LJ potential, it has a parameter ρ, which determines the width of the potential well (bottom point of the potential curve) and allows to model a wide variety of materials [5]. This potential depends on the interatomic distances $r$:

$$v_m(r) = [\exp\{\rho(1-r)\} - 1]^2 - 1,$$

and the potential energy is given by

$$V_M = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} v_m(r_{ij}),$$

where $r_{ij}$ is the distance between the atoms $i$ and $j$.

The LJ energy has the same curvature at the bottom of the well as the Morse energy when $\rho = 6$, and the Morse clusters show a wide variety of structural behavior as a function of ρ. When Morse clusters have less than 8 atoms, all the global minima of these clusters are independent of ρ. However, when the clusters have 8 atoms or more, motifs may change from icosahedral to decahedral to close-packed as the value of ρ is increased. There are, however, exceptional dimensions for which the transition from the icosahedral to the close-packed structure is direct.

Since the optimization process of both LJ and Morse clusters may be very hard, many proposed methods [5, 16, 20] for finding the stable cluster conformations subject to such energies are based on the structural motifs cited above. In this way, the degrees of freedom of each cluster can be reduced, and the optimization process can be easier. However, such methods are doomed to failure if the solution does not belong to the

**TABLE 1.** Percent success and mean computational time obtained by carrying out experiments with different values for the parameters $n_t$ and $h_t$

| $n_t$ / $h_t$ | 30 | | 50 | | 70 | | 100 | |
|---|---|---|---|---|---|---|---|---|
| 30 | 4% | 30s | 26% | 60s | 38% | 90s | 70% | 2m |
| 50 | 14% | 50s | 40% | 90s | 74% | 2m | 86% | 4m |
| 70 | 38% | 1m | 60% | 2m | 82% | 4m | 100% | 6m |
| 100 | 42% | 1m | 72% | 3m | 94% | 5m | 100% | 8m |

chosen motif, and they cannot be applied for solving any problem but the one they have been designed for.

In this study, we do not consider any information available on the low-energy cluster conformations, such as the structural motifs they may have. We check the efficiency of our algorithm in solving these problems without reducing the degrees of freedom of the atoms in the cluster. A two-phase method for optimizing LJ and Morse clusters has been proposed in [12], in which the knowledge on the particular motifs the cluster may assume is not exploited, even if the search is guided towards the global optimum by using information derived from the global shape of the expected solutions. We compared our Monkey Search to Simulating Annealing, Harmony Search, and the two-phase method, proving that Monkey Search performs competitively. Monkey Search algorithm is implemented using C++ programming language running on Windows XP operating system. A 4GHz duo-processor AMD Athlon is used for computational experiments. An experiment is considered to be successful if the obtained LJ or Morse cluster has an energy corresponding to those available on the Cambridge Cluster Database [2].

In order to investigate the efficiency of Monkey Search by using different values for the parameters $h_t$ and $n_t$, computational experiments are carried out, in which different values for $h_t$ and $n_t$ are used for optimizing an LJ cluster formed by 15 atoms. In all of the computational experiments reported below, the memory size $n_m$ is fixed and set to 10, the number of trees the monkey starts climbing from a random solution is set to $2n$, and the maximum number $n_{max}$ of trees the monkey is allowed to climb is set to 3000, but this limit has never been reached in our experiments.

In Table 1, percent success over 50 runs and mean computational times are reported for each couple ($n_t$,$h_t$). As we can see from this table, by increasing the parameter values, Monkey Search can obtain better results, but naturally, with the increased computational burden. Note that our method's performance improves faster as $h_t$ is increased. Since we obtained a 100% percent success when $n_t = 70$ and $h_t = 100$, we fixed these two parameters in the following computational experiments.

In Table 2, the results related to the optimization of LJ clusters are presented and compared to the results obtained by the two-phase method [12]. In Table 3, the results related to the Morse clusters with different choices of the parameter ρ are presented. As we can see from the Tables 2 and 3, the percent success of Monkey Search decreases by increasing the number $n$ of atoms in the cluster and by increasing the ρ value in the case of Morse clusters. As pointed out above, optimizing a LJ cluster is as hard as optimizing a Morse cluster with $\rho = 6$, and this is reflected on the obtained percent

**TABLE 2.** Percent success and mean computational time related to LJ clusters having different dimensions compared to the percent success of the two-phase method

| Lennard Jones | | | |
|---|---|---|---|
| $n$ | % success | time | % success in [12] |
| 10 | 100% | 1m | 52% |
| 12 | 100% | 4m | 94% |
| 15 | 100% | 6m | 22% |
| 18 | 58% | 11m | 19% |
| 20 | 28% | 13m | 36% |
| 22 | 10% | 16m | 24% |
| 23 | 6% | 17m | 29% |
| 25 | 10% | 20m | 35% |

**TABLE 3.** Percent success and mean computational time related to Morse clusters having different dimensions and ρ values

| Morse | | | | | |
|---|---|---|---|---|---|
| $n$ | % success | | | | time |
| | $\rho = 3$ | $\rho = 6$ | $\rho = 10$ | $\rho = 14$ | |
| 10 | 100% | 100% | 100% | 86% | 4m |
| 12 | 100% | 100% | 96% | 58% | 5m |
| 15 | 100% | 88% | 42% | 14% | 10m |
| 18 | 88% | 24% | 24% | 2% | 12m |
| 20 | 34% | 30% | 4% | 0% | 15m |
| 22 | 14% | 22% | 2% | 0% | 18m |
| 23 | 10% | 0% | 0% | 0% | 20m |
| 25 | 10% | 0% | 0% | 0% | 22m |

success. Monkey Search results are found to be very competitive compared to the two-phase method. Moreover, as the results in Table 1 suggest, the percent success obtained by Monkey Search can be improved by modifying the parameters $n_t$ and $h_t$.

We also compared our Monkey Search to other metaheuristic algorithms, such as Simulated Annealing and Harmony Search for LJ clusters. Both of these metaheuristics were not able to find the global optimum. In Table 4, the best results for some choices of the dimension $n$ are presented for 50 runs. Harmony Search generally stops far from the optimal solutions, and Simulated Annealing can get closer to the optimal solution but it is not able to find the precise location of the optimal clusters, whereas Monkey Search always found the optimal conformations. In the Harmony Search experiments, the harmony memory size *hms* has been set to 10, 25, 50 and 100, for each of the 50

**TABLE 4.** The LJ energy related to the best results obtained by Harmony Search, Simulated Annealing, and Monkey Search, after 50 runs for each dimension $n$

| $n$ | HS | SA | MS |
|---|---|---|---|
| 10 | -26.957428 | -28.422322 | -28.422532 |
| 12 | -32.293843 | -37.967146 | -37.967600 |
| 15 | -40.959475 | -52.321407 | -52.322627 |
| 18 | -41.079693 | -66.280907 | -66.530949 |
| 20 | -49.213888 | -77.170622 | -77.177043 |
| 22 | -51.988805 | -86.194146 | -86.809782 |
| 23 | -50.888812 | -91.363302 | -92.844472 |
| 25 | -56.919838 | -100.675583 | -102.372663 |

runs. The presented results in which $n = 10$ to 15 have been obtained with $hms = 25$, whereas $hms = 10$ for each result for dimension $n = 18$ to 25. The starting temperature in the Simulated Annealing experiments has been set to 10, $100 \cdot n$ steps are performed at constant temperature, and then the temperature parameter is decreased by 1%.

## 3.2. The tube model

The tube model is a geometric model for protein folding proposed in [1, 9] and recently modified in [3]. Protein folding is one of the major challenges in biology and medicine: nobody knows how a chain of amino acids folds forming the typical compact and functional structures the proteins have inside the cells. The most promising approach in this area is the *ab-initio* one, in which the energy in the proteins is modeled and is minimized in order to find the energetically stable conformations. The considered tube model can be classified as ab-initio, but it is very different from the others because it is mainly based on geometric features of proteins. One cannot predict the protein conformation by using this model, but just simulating conformations which may be similar to those proteins actually have.

The model is named as the tube model because a protein conformation is modeled as a tube with a certain diameter such that the tube does not cross itself. The diameter of the tube has particular values when the protein secondary structure such as $\alpha$-helices and $\beta$-sheet are considered. The model leads to the formulation of a global optimization problem, in which some requirements on the protein conformations are imposed. The optimization problem can be formulated in different ways, it depends on the requirements one wants to consider and the relationships among them. More details on the tube model and on the different formulations of the optimization problem can be found in [14].

The global optimization problem we consider in this work can be described as follows:

$$\min_X f(X)$$

where $X = \{x_1, x_2, \ldots, x_n\}$ is a protein conformation represented through the spatial coordinate of each $C_\alpha$ Carbon atom (one for each amino acid), and $f$ is the objective function which takes into account three geometric requirements. Its mathematical formulation is:

$$\begin{aligned} f(X) = \quad & \gamma_1 \sum_{i=1}^{n-3} \sum_{j=i+3}^{n} d(x_i, x_j) + \\ & \gamma_2 \sum_{i=1}^{n-2} \sum_{j=i+2}^{n} exp_+(th - d(x_i, x_j)) + \\ & \gamma_3 \sum_{i=1}^{n-2} (d(x_i, x_{i+2}) - c)^2 \end{aligned}$$

where $\gamma_1$, $\gamma_2$, $\gamma_3$, $th$ and $c$ are five positive and real constants. The function $exp_+$ corresponds to the exponential function when its argument is non-negative, and it is zero otherwise; $d$ represents the Euclidean distance.

The considered optimization problem simulates protein conformations which are compact and contain α-helices. The first term of $f(X)$ minimizes all the relative Euclidean distances; the second one avoids the case that the conformation falls on itself, by keeping a threshold value of $th = 4.30$; finally, the last term forces each $x_i$ and $x_{i+2}$ to have a relative distance equal to $c \in [5.0, 6.0]$ units, which is a typical distance in α-helices. Even though only a $C_\alpha$ atom is considered for each amino acid, we represent the protein conformation by the backbone dihedral angles representation, in order to avoid unnatural conformations. Hence, the objective function $f(X)$ is evaluated after a change of representation. The constants $\gamma_1$, $\gamma_2$ and $\gamma_3$ are set, respectively, to 1.0, 10.0 and 1.0.

We are interested in simulating a large number of protein conformations in a short amount of time. Therefore we rely on simple requirements for conformations and check if some protein conformation which actually exist in Nature is found. For this reason, Monkey Search must perform very fast. We set the maximum number $n_{max}$ of trees the monkey is allowed to climb to 40 only. Since the number of variables which are involved is greater than those in the LJ and Morse clusters, and since $n_{max}$ is limited to 40, we used larger values for the parameters $h_t$ and $n_t$. The memory dimension $n_m$ is fixed at 10, the initial random number of trees $n_w$ is $2n$, and the computational hardware and software setup is the same we used for carrying out the LJ and Morse experiments.

As the reference protein, we have chosen the one which is referenced in the Protein Data Bank (PDB) [15] by the code `2cro`. This protein contains 65 amino acids and it is one of the smallest all-alpha proteins currently known. Table 5 shows the results related to experiments, in which different values for the parameters $h_t$ and $n_t$ have been used. For each pair of $h_t$ and $n_t$ values, 50 runs have been carried out and the best result has been reported. From this table, the best result has been obtained with $h_t = 200$ and $n_t = 125$. Fig. 1 shows the related protein conformation and the `2cro` protein. The conformation found is very compact, it contains α-helices and it has the same globular shape of our reference protein. This is surprising because such a conformation has been simulated by using a model which is very simple with respect those usually used in the field of protein

**TABLE 5.** Simulation of a protein conformation having 65 amino acids by using different values for $h_t$ and $n_t$. For each pair of $h_t$ and $n_t$ values, the best result over 50 runs is reported

| $h_t$ | $n_t$ | $f(X)$ | time | $h_t$ | $n_t$ | $f(X)$ | time |
|---|---|---|---|---|---|---|---|
| 100 | 10 | 656185 | 40s | 200 | 50 | 652685 | 3m |
| 100 | 25 | 585785 | 50s | 200 | 100 | 594501 | 5m |
| 100 | 50 | 586891 | 1m | 200 | 125 | **549413** | 7m |
| 100 | 75 | 577496 | 2m | 200 | 150 | 597495 | 9m |
| 100 | 85 | 586284 | 3m | 200 | 175 | 552085 | 10m |
| 100 | 100 | 644809 | 5m | 200 | 200 | 598234 | 13m |

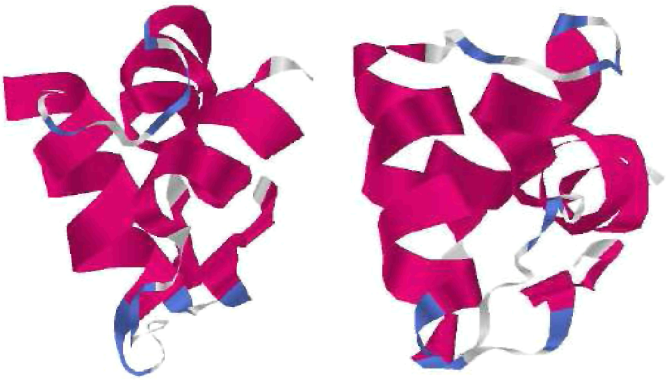

**FIGURE 1.** On the left, the best protein conformation obtained by Monkey Search; on the right, the actual conformation of the `2cro` protein

folding. Even if the other simulated conformations have a larger objective function value, they preserve the globular shape of `2cro` as well, but they are less compact and contains less $\alpha$-helices. Further details on the simulated protein conformations are reported in [14].

As for the LJ and Morse clusters, we compared the performance of Monkey Search on this problem to Simulated Annealing and Harmony Search. Simulated Annealing performed $100 \cdot n$ (with $n = 65$) steps at constant temperature, where the temperature was initially set to 10 and decreased by 1% at once. After 50 runs, Simulated Annealing found a conformation having a function value equal to 848922 as the best solution, with a mean computational time of 6 minutes. In the experiments related to Harmony Search, the harmony memory size *hms* has been set to 10, 25, 50 and 100, and 50 runs have been carried out for each of them. The best solution Harmony Search found had a function value equal to 729381 with a mean computational time of 4 minutes. As we can see, Monkey Search performs much better than Simulated Annealing and Harmony Search on this problem. Moreover, Harmony Search works better than Simulated Annealing on the tube model, whereas Simulated Annealing can provide better LJ and Morse clusters.

In fact, each metaheuristic method can be more efficient for solving some optimization problem and less efficient for solving another. The fact that the perturbations in Monkey Search are inspired by Simulated Annealing, Harmony Search and also by other metaheuristic searches may be the reason why Monkey Search performs better than the other two metaheuristics.

## 4. CONCLUSIONS

A novel metaheuristic method for global optimization has been proposed which resembles the behavior of a monkey climbing trees in its search for food. Monkey Search discovers a tree of solutions where branches contain neighbor solutions on their ends. The monkey marks these branches on its way climbing down whenever a better solution is found, such that these marks can be used to choose branches to climb, on the way up. This marking scheme is intended for the monkey to focus on a part of tree where it can find better solutions. The flexibility of Monkey Search comes from the definition of the perturbations which define the neighbor solutions. Such perturbations exploit ideas and strategies from other metaheuristics and optimization methods and combines them all into an unique method.

In this study, we show that Monkey Search performs significantly better than some other metaheuristic methods such as Simulated Annealing and Harmony Search for solving a set of global optimization problems emerging from molecular structures. Regarding the LJ and Morse clusters, Monkey Search is able to find better solutions with a higher percent success, and it is also very competitive compared to the two-phase method, which partially exploits information on the known solutions. Regarding the tube model for protein folding, after 50 trials, MS is able to provide better solutions compared to Simulated Annealing and Harmony Search, providing protein conformations satisfying the considered geometric requirements.

Further experiments on the optimal monkey behavior and the use of other global optimization problems are considered next as future research to explore the potential of Monkey Search in a more general sense. Since the structure of Monkey Search allows other search methods to be embedded in it, Monkey Search provides endless possibilities, which would potentially produce competitive results compared to these individual metaheuristics.

## REFERENCES


1. J. R. Banavar, A. Maritan, C. Micheletti, and A. Trovato, *Geometry and Physics of Proteins* **47**, 315-322 (2002).
2. Cambridge Cluster Database, `http://www-wales.ch.cam.ac.uk/CCD.html`.
3. G. Ceci, A. Mucherino, M. D'Apuzzo, D. di Serafino, S. Costantini, A. Facchiano, and G. Colonna, "Computational Methods for Protein Fold Prediction: an Ab-Initio Topological Approach, Data Mining in Biomedicine", *Optimization and Its Applications*, edited by Panos Pardalos et al., vol. 7, Springer, 2007.
4. M. Dorigo, and G. Di Caro, "Ant Colony Optimization: A New Meta-Heuristic", in *Proceedings of the Congress on Evolutionary Computation*, 1999, pp. 1470–1477.
5. J.P.K. Doye, and D.J. Wales, *J. Chem. Soc. Faraday Trans.* **93**, 4233–4244 (1997).

6. J.P.K. Doye, R.H. Leary, M. Locatelli, and F. Schoen, *INFORMS Journal On Computing* **16**, 371–379 (2004).
7. Z. W. Geem, J. H. Kim, G. V. Loganathan, *SIMULATIONS* **76**, 60–68 (2001).
8. D. E. Goldberg, *Genetic Algorithms in Search, Optimization & Machine Learning*, Reading, MA: Addison-Wesley, 1989.
9. T. X. Hoang, A. Trovato, F. Seno, J. R. Banavar, and A. Maritan, *PNAS* **101**, 7960–7964 (2004).
10. S. Kirkpatrick, C. D. Jr. Gelatt, and M. P. Vecchi, *Science* **220**, 671–680 (1983).
11. J. E. Lennard-Jones, "Cohesion", in *Proceedings of the Physical Society*, **43**, 1931, pp. 461–482.
12. M. Locatelli, and F. Schoen, *Computational Optimization and Applications* **21**, 55–70 (2002).
13. P. M. Morse, *Physical Review* **34**, 57–64 (1929).
14. A. Mucherino, O. Seref, P.M. Pardalos, working paper.
15. Protein Data Bank, `http://www.rcsb.org/pdb/`.
16. J.A. Northby, *J. Chem. Phys.* **87**, 6166–6177 (1987).
17. B. Raoult, J. Farges, M.F. De Feraudy, and G. Torchet, *Philos. Mag.* **60**, 881–906 (1989).
18. O. Seref, and E. Akcali, "Monkey Search: A New Meta-Heuristic Approach", in INFORMS Annual Meeting, San Jose, CA, 2002.
19. R. Storn, and K. Price, *Journal of Global Optimization* **11**, 341–359 (1997).
20. Y. Xiang, H. Jiang, W. Cai, and X. Shao, *J. Phys. Chem.* **108**, 3586–3592 (2004).
21. T. Zhou, W.-J. Bai, L. Cheng, and B.-H. Wang, *Physical Review E* **72**, 016702 (2005).