Impact of manipulation on initial population in heuristics

Alicja Winnicka
Institute of Mathematics
Silesian University of Technology
Kaszubska 23, 44-100 Gliwice, Poland
Email: Alicja.Lidia.Winnicka@gmail.com

***Abstract*—The optimization problem is important due to practical use in various areas of our lives. Unfortunately, quite often there is a situation that we need to find the minimum values with certain criteria. Finding a solution using a classic approach is impossible to apply, which is due to an infinite number of solutions. That is why heuristic algorithms are used to find the optimal solution in a finite time. In this paper we propose the idea of manipulating the initial population due to better distribution of individuals in the whole space. The proposed solution has been described, tested and discussed.**

I. Introduction

Optimization algorithms find great practical use in various areas of our lives. Designing a construction such as a house requires an ideal weight distribution on the columns. From a mathematical point of view, the pressure, length and width of walls or columns can be described with the help of several equations, where the unknown value will depend on all elements. However, finding the best values to make this construction also as cheap as possible is quite a complicated problem. For this purpose, various algorithms are used that allow finding solutions that meet all criteria.

However, the problem is to search for these values. Quite often the set of values is an infinite one. And this prevents the use of iterative algorithms. These types of problems have contributed to the creation of heuristics, ie methods that do not guarantee ideal (and sometimes even correct) solutions in a finite period of time. Although, a large number of optimization problems are possible to solve using them.

The development of heuristics is driven by practical use. This is particularly important in the methods of artificial intelligence, where training the classifier occurs by minimizing weights in order to get the least error [1], [2] or the optimization used in the operation of various systems [3]–[6]. Another important element is the optimization in smart grids and microgrids [7]–[9]. In this paper, we want to show how the manipulation of the initial distribution of individuals in the population depends on two classic heuristic algorithms.

II. Optimization problem

The optimization problem is understood as finding global extremes for a specific function $f(x) : \mathbb{R}^n \to \mathbb{R}$ for a given points $\mathbf{x} = (x_0, x_1, \ldots, x_{n-1})$, $x_i \in \langle L_i, R_i \rangle$ for $i = 0, 1, \ldots, n-1$. In the case of minimization problem, it can be formulated as

$$\begin{aligned} &\text{Minimize} && f(\mathbf{x}) \\ &\text{subject to} && g(\mathbf{x}) \geq 0 \\ & && L_i \leq x_i \leq R_i \qquad i = 0, 1, \ldots, n-1, \end{aligned} \tag{1}$$

where $g(\cdot)$ is inequality constraint.

III. Cuckoo Search Algorithm

Cuckoo Serch Algorithm (CSA) is one of the heuristic algorithms based on living creatures – in this case on cuckoos [10]–[12]. After observation of these birds, people noticed, that cuckoos have specific way to hatch their eggs. They actually do not hatch them by themselves, but search another bird's nests to leave eggs there. In that case, they need to find the nest whose host does not recgnize unfamiliar egg.

The behaviour of these birds inspired to extend minimization algorithm. CSA assumes that cuckoo is a point $\mathbf{x} = (x_0, \ldots, x_{n-1})$ on the n-dimension solution space. Of course, modeling the natural life of these creatures is impossible, so it is simplified due to the following points

- the size of population is constant in all iterations,
- each cuckoo can lay only one egg per iteration,
- cuckoo is identified with egg,
- host may detect unfamiliar egg with some propability, and in that case when egg is found, the cuckoo has to look for new place in the solution space.

At the beginning of algorithm, the population is created. Each cuckoo is placed on the solution space in random way. In each iteration, each cuckoo performs two actions. The first is a flight to another place which is modeled using Levy's equation (called *Levy's flight*) defined as

$$L(\mathbf{x}, \zeta, \eta) = \sqrt{\frac{\eta}{2\pi}} \frac{e^{-2\eta(\mathbf{x}-\zeta)}}{\sqrt{(\mathbf{x}-\zeta)^3}}. \tag{2}$$

In the above formula $\zeta, \eta \in \langle 0, 1 \rangle$ are coefficient. Using this formula, the cuckoo's movement is done using the following equation

$$\mathbf{x} = \mathbf{x} \pm L(\mathbf{x}, \zeta, \eta). \tag{3}$$

The second step is host decision because after the cuckoo flight, her egg is tossed to the nest. In fact, cuckoos can mask

the tossed egg to make it look like other ones – the probability of detection is minimized. Modeling this phenomenon involves defining a threshold value Ξ, with respect to which the egg will be detected. For random probability $\xi \in \langle 0, 1\rangle$ the following condition is checked

$$H(\mathbf{x}) = \begin{cases} \xi > \Xi & \text{the egg remains in the nest} \\ \xi \leq \Xi & \text{the egg is removed from the nest} \end{cases}. \tag{4}$$

In the case when the egg is removed from the nest, a new position is choosen for this cuckoo (in random way) – this action guarantees a constant size of the population. The algorithm is executed to satisfy a stop condition, which is most often the number of iteration in the algorithm. At the end, the best cuckoo is returned, which has the smallest or largest value of fitness function in the entire population (depending on the optimization problem being considered).

Data: number of iterations t, number of cuckoos k, fintess function $f(\cdot)$, the solution space, threshold value Ξ
Result: The best cuckoos
Start;
Generate a population of k cuckoos in random way in solution space;
$i := 0$;
while $i \leq t$ **do**
  $k = 0$;
  while $j \leq k$ **do**
    Change the position of j-th cuckoo using Eq. (2);
    Choose $\xi \in \langle 0, 1\rangle$ in random way;
    if $\xi > \Xi$ **then**
      Generate new position for j-th cuckoo;
    end
  end
end
Return cuckoo with the best fitness value $f(\cdot)$;
Stop;

Algorithm 1: Cuckoo Search Algorithm.

IV. Polar Bear Algorithm

Another heuristic algorithm is inspired by polar bear's behavior during hunting [13]. Polar bear must find the prey and if he doesn't find anything, he must use the ice floak as a way to move a certain distance. He uses it to drift to another location, where possibly he will find the prey (especially seals).

The basis of this algorithm are similar to the CSA, we have a population of k bears, where each of them can be represented as point $\mathbf{x} = (x_0, x_1, \ldots, x_{n-1})$ in some solution space. At the beginning, an initial population is created with one difference – only 75% of individuals are generated, the remaining 25% depend on Arctic conditions and reproduction.

Each bear moves only if the new position is better than the current one in relation to the fitness function $f(\cdot)$. It can be defined as the following equation

$$(\mathbf{x}_j^t)^{(i)} = (\mathbf{x}_j^{t-1})^{(i)} + \text{sign}\,(\omega)\,\alpha + \gamma, \tag{5}$$

where α is a random number in $\langle 0, 1\rangle$, ω is the distance between two spatial coordinates and γ is a random value in the range of $\langle 0, \omega\rangle$. The distance is defined using simple Euclidean metric between two points and described as

$$d\left((\mathbf{x})^{(i)}, (\mathbf{x})^{(j)}\right) = \sqrt{\sum_{k=0}^{n-1}\left((x_k)^{(i)} - (x_k)^{(j)}\right)^2}. \tag{6}$$

Polar bears do not hunt only on the surface, but also in the water. In a situation where the bear flows on the ice floe, he closely observes the surrounding water. In the case of noticing the future victim, the bear dives into the water and attack. There are occasions when a bear very suddenly throws himself on the victim, if there is a chance that he will run away. In the modeling of this phenomenon, the polar bear's area of view should be considered as

$$r = 4a\cos(\phi_0)\sin(\phi_0), \tag{7}$$

where $a \in \langle 0, 0.3\rangle$ is a visible distance, $\phi_0 \in (0, \frac{\pi}{2}\rangle$ is the angle of approaching to the victim. This equation is used to describe the local movement of these individuals, where each spatial coordinate is modified using

$$\begin{cases} x_0' = x_0 \pm r\cos(\phi_1) \\ x_1' = x_1 \pm [r\sin(\phi_1) + r\cos(\phi_2)] \\ x_2' = x_2 \pm [r\sin(\phi_1) + r\sin(\phi_2) + r\cos(\phi_3)] \\ \ldots \\ x_{n-2}' = x_{n-2} \pm \left[\sum_{k=1}^{n-2} r\sin(\phi_k) + r\cos(\phi_{n-1})\right] \\ x_{n-1}' = x_{n-1} \pm \left[\sum_{k=1}^{n-2} r\sin(\phi_k) + r\sin(\phi_{n-1})\right] \end{cases}, \tag{8}$$

where $\phi_1, \phi_2, \ldots \phi_{n-1} \in \langle 0, 2\pi\ \rangle$.

Difficult living conditions on arctic surfaces may be detrimental to the individuals living there, for this purpose a random value $\kappa \in \langle 0, 1\rangle$ is introduced, which allows to introduction the conditions of freezing or dying from hunger as the following rule

$$\begin{cases} \text{Death} & \text{if } \kappa < 0,25 \\ \text{Reproduction} & \text{if } \kappa > 0,75 \end{cases}, \tag{9}$$

where, if the first condition is met, the weakest individual dies (on the condition that the population size is greater than half). In the second case, the two individuals in a given iteration (one is the best one in whole population relative to the fitness function and when the size of population is smaller than assumed) reproduce as

$$(\mathbf{x}_j^t)^{(new)} = \frac{(\mathbf{x}_j^t)^{(best)} + (\mathbf{x}_j^t)^{(i)}}{2}, \tag{10}$$

Table I: Test functions used in optimization.

Name	Function formula	Domain	Point **x**	Minimum
Ackley	$-20\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right)-\exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right)$	$\langle -32.8, 32.8\rangle$	$(0,\ldots,0)$	0
Booth	$(x_1+2x_2-7)^2+(2x_1+x_2-5)^2$	$\langle -10, 10\rangle$	$(1,3)$	0
Easom	$-\cos(x_1)\cos(x_2)\exp\left(-(x_1-\pi)^2-(x_2-\pi)\right)$	$\langle -10, 10\rangle$	(π,π)	-1
McCormick	$\sin(x_1+x_2)+(x_1-x_2)^2-1.5x_1+2.5x_2+1$	$x_1\in\langle -1.5, 4\rangle$, $x_2\in\langle -3, 4\rangle$	$(-0.54719, -1.54719)$	-1.9133
Sphere	$\sum_{i=1}^{n}x_i^2$	$\langle -10, 10\rangle$	$(0,\ldots,0)$	0
Trid	$\sum_{i=1}^{n}(x_i^2-1)^2-\sum_{i=2}^{n}x_ix_{i-1}$	$\langle -10, 10\rangle$	$(i(n+1-i))$	$\frac{-n(n+4)(n-1)}{6}$
Zakharov	$\sum_{i=1}^{n}x_i^2+\left(\sum_{i=1}^{n}0.5ix_i\right)^2+\left(\sum_{i=1}^{n}0.5ix_i\right)^4$	$\langle -5, 10\rangle$	$(0,\ldots,0)$	0

where $(\mathbf{x}^t)^{(best)}$ and $(\mathbf{x}^t)^{(i)}$ are two individuals selected from the best in whole population.

Data: number of iterations t, number of polar bears k, fintess function $f(\cdot)$, the solution space
Result: The best polar bear
Start;
Generate a population of $0.75\cdot k$ bears in random way in a given solution space;
$i := 0$;
while $i \leq t$ **do**
  $k = 0$;
  while $j \leq k$ **do**
    Calculate new position using Eq. (5);
    if *new position is better* **then**
      Change the position of j-th bear;
    end
    Calculated a view distance using Eq. (7);
    Move bear closer to the victim using Eq. (8);
    Choose $\kappa\langle 0, 1\rangle$ in a random way;
    if $\kappa > 0.75$ *and the population size is correct* **then**
      Remove the weakest individual from the population;
    end
    if $\kappa < 0.25$ *and the population size is correct* **then**
      Take the two best individuals for the reproduction process according to Eq. (10);
    end
  end
end
Return polar bear with the best fitness value $f(\cdot)$;
Stop;

Algorithm 2: Polar Bear Algorithm.

V. Manipulation of the initial population

The idea is to place individuals at similar distances, which has a chance to guarantee a search of a larger area. The reason is that the placement of individuals in the original algorithms is based on random choice. This can lead to the situation that all individuals will be located in a similar area, omitting the rest.

Each individual is positioned according to the solution space, i.e. $\langle L_i, R_i\rangle$. Having k individuals, the area can be divided into m parts as

$$\bigcup_{j=0}^{m-2}\left\langle L_i + j\cdot\frac{R_j}{m}, L_i + (j+1)\cdot\frac{R_j}{m}\right\rangle. \tag{11}$$

In each subset, $\frac{k}{m}$ individuals are generated in random way. The boundary values of subsets are repeated in neighboring elements, but it does not extend the initial range.

VI. Experiments

As part of checking the validity of the proposed solution, the test function described in Tab. I were used (with dimension equal to 10). Tests for described algorithms with space manipulation and without were conducted for two different size of population – 100 and 1000 during 300 iterations. Note that the individuals will be placed at random, so to authenticate the results, each algorithm will be made 10 times, and the result averaged. Obtained solutions are presented in Tab. III and II.

According to the tables, the average results are in almost every case more precise, which allows us to stalk that it is worth manipulating the solution space and dividing it into smaller subsets in which individuals will move in heuristic algorithms.

VII. Conclusions

In this paper, we proposed an even distribution of individuals in the initial populations in order to increase the search of the full area. The tests were carried out using two algorithms inspired by nature – Cuckoo Search Algorithm and Polar Bear Algorithm. The obtained results indicate a much faster finding of the extreme, and compared to the same number of iterations – the solutions are more accurate. This allows to say that heuristics can work more effectively when their randomness is minismized.

Table II: Obtained average results for 10 individuals.

Function name	CSA	CSA with modification	PBA	PBA with modification
Ackley	0.02972880281961	0.0177714446875131	0.0897574672427762	0.071868300285129
Booth	-0.0158455184268977	-0.0089278921386823	0.063578710734648	0.0355476169081161
Easom	-0.905359405654184	-0.987498435744782	-0.992768202439308	-0.982191178613431
McCormick	-1.97623352281811	-1.94638200721307	-1.9460962896008	-1.86267427586381
Sphere	-0.0536565001372511	0.0149251524428488	-0.0643595185430532	0.0481632181667552
Trid	-209.889724268061	-210.042710510819	-210.021059083669	-210.065610355961
Zakharov	0.00851286580251198	-0.0125859739783155	-0.0245003661254888	0.0547493551181393

Table III: Obtained average results for 100 individuals.

Function name	CSA	CSA with modification	PBA	PBA with modification
Ackley	0.0234407469855811	0.011506596170644	-0.0205448465051804	0.0162699407465658
Booth	-0.0176004870876672	-0.0171003493475348	-0.0215751469119336	0.00856991690897929
Easom	-0.979836908462381	-0.996544835418547	-1.01681687238944	-0.989967549610368
McCormick	-1.90125860997073	-1.92803074770287	-1.92798707448323	-1.99966222023081
Sphere	0.0134152536808584	0.00213265699349087	0.0206986147541081	-0.0172196125691848
Trid	-210.005388625923	-209.982567193072	-210.007645864532	-209.994710194309
Zakharov	-0.0181133843577064	-0.0170223669058917	0.000409970479276949	0.0166918954551055

REFERENCES

[1] R. Damaševičius, "Optimization of svm parameters for recognition of regulatory dna sequences," *Top*, vol. 18, no. 2, pp. 339–353, 2010.

[2] D. Połap, "Human-machine interaction in intelligent technologies using the augmented reality," *Information Technology And Control*, vol. 47, no. 4, pp. 691–703, 2018.

[3] Y. Gao and Y.-J. Liu, "Adaptive fuzzy optimal control using direct heuristic dynamic programming for chaotic discrete-time system," *Journal of Vibration and Control*, vol. 22, no. 2, pp. 595–603, 2016.

[4] S. Lange, S. Gebert, T. Zinner, P. Tran-Gia, D. Hock, M. Jarschel, and M. Hoffmann, "Heuristic approaches to the controller placement problem in large scale sdn networks," *IEEE Transactions on Network and Service Management*, vol. 12, no. 1, pp. 4–17, 2015.

[5] Z. Pooranian, M. Shojafar, J. H. Abawajy, and A. Abraham, "An efficient meta-heuristic algorithm for grid computing," *Journal of Combinatorial Optimization*, vol. 30, no. 3, pp. 413–434, 2015.

[6] C. Jagtenberg, S. Bhulai, and R. van der Mei, "An efficient heuristic for real-time ambulance redeployment," *Operations Research for Health Care*, vol. 4, pp. 27–35, 2015.

[7] G. Capizzi, G. L. Sciuto, C. Napoli, and E. Tramontana, "Advanced and adaptive dispatch for smart grids by means of predictive models," *IEEE Transactions on Smart Grid*, vol. 9, no. 6, pp. 6684–6691, 2017.

[8] G. Graditi, M. L. Di Silvestre, R. Gallea, and E. R. Sanseverino, "Heuristic-based shiftable loads optimal management in smart micro-grids," *IEEE Transactions on Industrial Informatics*, vol. 11, no. 1, pp. 271–280, 2015.

[9] C. Napoli, G. Pappalardo, G. M. Tina, and E. Tramontana, "Cooperative strategy for optimal management of smart grids by wavelet rnns and cloud computing," *IEEE transactions on neural networks and learning systems*, vol. 27, no. 8, pp. 1672–1685, 2015.

[10] X.-S. Yang and S. Deb, "Cuckoo search via lévy flights," in *Nature & Biologically Inspired Computing, 2009. NaBIC 2009. World Congress on*. IEEE, 2009, pp. 210–214.

[11] D. Połap, M. Wozniak, C. Napoli, and E. Tramontana, "Is swarm intelligence able to create mazes?" *International Journal of Electronics and Telecommunications*, vol. 61, no. 4, pp. 305–310, 2015.

[12] D. Połap, M. Woźniak, C. Napoli, E. Tramontana, and R. Damaševičius, "Is the colony of ants able to recognize graphic objects?" in *International Conference on Information and Software Technologies*. Springer, 2015, pp. 376–387.

[13] D. Połap *et al.*, "Polar bear optimization algorithm: Meta-heuristic with fast population movement and dynamic birth and death mechanism," *Symmetry*, vol. 9, no. 10, p. 203, 2017.