Maximization of a Dissimilarity Measure for Multimodal Optimization

Fabrício Olivetti de Franca, *Member, IEEE*
Universidade Federal do ABC (UFABC)
Center of Mathematics, Computing and Cognition (CMCC)
R. Santa Adélia 166, CEP 09210-170, Santo André, Brazil
Email: folivetti@ufabc.edu.br

***Abstract*—Many practical problems are described by an objective-function with the intent to optimize a single goal. This leads to the important research topic of nonlinear optimization, that seeks to create algorithms and computational methods that are capable of finding a global optimum of such functions. But, many functions are multimodal, having many different global optima. Also, given the impossibility to create an exact model of a real-world problem, not every global (or local) optima is feaseable to be conceived. As such, it is interesting to find as many alternative optima in order to find one that is feaseable given unmodelled constraints. This paper proposes a methodology that, given a local optimum, it finds nearby local optima with similar objective-function values. This is performed by maximizing the approximation error of a Linear Interpolation of the function. The experiments show promising results regarding the number of detected peaks when compared to the state-of-the-art, though requiring a higher number of function evaluations on average.**

Keywords*—*multimodal optimization, niching, nonlinear optimization.

I. Introduction

Multimodal Optimization may refer to the task of finding one global optimum in multimodal functions or the problem of finding as many optima, global or local, as possible to be chosen by a decision maker afterwards. The first task motivated the creation of populational algorithms, notably the Evolutionary Algorithms [1], [2], [3], that manage to explore the search space while still exploiting the local neighborhood. Although the idea of a population of solutions improved the exploration when compared to traditional local search, in some cases it was proven insufficient due to a large number of deceiving local optima.

One way to deal with such problems were with the use of Niching techniques [4], [5], [6] that tries to create rewards when the algorithm maintains the diversity of a population or punishments when the population converges to a single local optimum.

The second definition of multimodal optimization was motivated by practical applications in which, after the end of the optimization process, the decision maker realized that the solution found was unfeasible [7]. This happens because many real-world problems are impossible to be perfectly modelled by a mathematical function and constraints or when the problem is affected by some uncertainties [8].

978-1-4799-7492-4/15/$31.00

This problem was also solved using Niching techniques enforcing the output of multiple solutions. Additionaly, this motivated the creation of specific computational methods specially crafted to find multiple optima, such as the immune-inspired approaches [8], [9], [10].

As these methods aim at exploring distinct regions of the search space, expecting to find the basis of attraction of different global optima, they often use a distance metric to determine whether two points are sufficiently far apart. Most algorithms use the Euclidean distance but this poses as a problem in order to find the right definition of what is sufficiently far apart.

In [8] a distance measure called *Line Distance* was introduced to solve this problem. This measures is defined as the error of the approximation of a linear interpolation between two points to the real function. In other words, if two points are located at the same side of the same basis of attraction of a local optimum, the linear interpolation will correctly approximate the objective-function, thus leading to a small error. If they are located in different optima, the error will be larger, bounded by the peak height.

Although this measure do not completly solve the aforementioned problem, it makes easier to correctly define a threshold for closeness that works for most objective-functions. It has been shown that the threshold setting is quite robust in many multimodal functions [6].

In this paper, the Line Distance between a solution and its nearest-neighbor will replace the original fitness function. By doing so, each solution is expected to evolve to a distinct local optima. The proposed algorithm will take inspirations from Artificial Immune Systems and Niching techniques.

The paper is organized as follows: Section II will explain the calculation of the Line Distance, further simplifying its calculation. Section III devises a computational method to find many local optima by maximizing this distance. Section IV will evaluate such method by means of a well-known benchmark for multimodal optimization. Finally, Section V concludes this papers with some insights for future work.

II. Line Distance

The Line Distance was first proposed in [8] in order to measure the similarity between two solutions in an Artificial Immune Systems Algorithm (AIS). The AIS algorithm, named dopt-aiNet (artificial immune Network for dynamic optimization), was conceived to generate a network of dissimilar solutions, each of which located at a different local optima. This dissimilarity measure was used in the context of a suppression mechanism to prune the network of solutions to retain only unique solutions.

The Line Distance between two points in Euclidean Space, $x_1, x_2 \in \Re^n$, with objective-function values of y_1 and y_2, respectively, is calculated by first creating a middle point $x_m = 0.5(x_1 + x_2)$ with objective-function value y_m, and then measuring the distance between the $n+1$-dimensional point $P_m = [x_m, y_m]$ to the segment defined by $P_1 = [x_1, y_1]$ and $P_2 = [x_2, y_2]$.

The distance between this point to this segment is given by:

$$LD(x_1, x_2) = \|P_{proj}\|$$

with

$$\begin{aligned} P_{proj} &= \overline{P_1 P_m} - (\overline{P_1 P_m} \cdot v)v \\ v &= \frac{P_2 - P_1}{\|P_2 - P_1\|} \\ \overline{P_1 P_m} &= [\frac{x_2 - x_1}{2}, y_m - y_1] \end{aligned}$$

By simplifying this equation [1] we have:

$$LD(P_1, P_2) = \sqrt{\frac{((\frac{y_1+y_2}{2} - y_m)^2 \sum_i (x_2^i - x_1^i)^2)}{\sum_i (x_2^i - x_1^i)^2) + (y_2 - y_1)^2}} \quad (1)$$

As we can see, the rationale for this measure is that, if the line created by x_1 and x_2 approximates the objective-function curve, then these points must be at the same basis of attraction. But, on the other hand, if the approximation has a larger error, it means they must be located at different local optimum (see Fig. 1).

Theorem 1 (Infinity Scaling): By scaling the objective-function $f(x)$ by a constant K, even if $K \to \infty$, it will have an insignificant effect on the Line Distance whenever the two points are located at the same basis of attraction.

Proof: If we scale the objective-function and replace these values on Eq. 1 we have:

$$LD(P_1, P_2) = \sqrt{\frac{K^2((\frac{y_1+y_2}{2} - y_m)^2 \sum_i (x_2^i - x_1^i)^2)}{\sum_i (x_2^i - x_1^i)^2) + K^2(y_2 - y_1)^2}}$$

$$LD(P_1, P_2) = \sqrt{\frac{((\frac{y_1+y_2}{2} - y_m)^2 \sum_i (x_2^i - x_1^i)^2)}{(\sum_i (\frac{x_2^i - x_1^i}{K})^2) + (y_2 - y_1)^2})}$$

[1] the step-by-step simplification of this equation is available at http://nbviewer.ipython.org/github/folivetti/LINKEDOPT/blob/master/LineDistance.ipynb

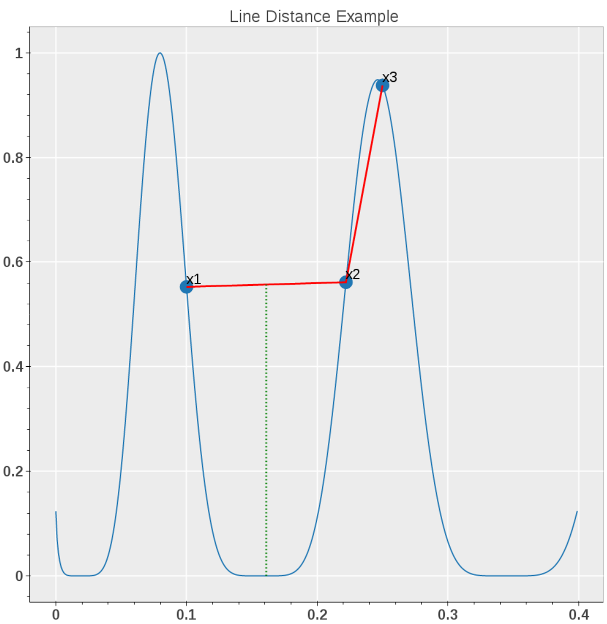

Fig. 1. Depiction of Line Distance with $LD(x_1, x_2) = 0.55$ and $LD(x_2, x_3) = 0.01$.

As $K \to \infty$, we have that $\frac{x_2 - x_1}{K} \to 0$ and, thus:

$$LD(P_1, P_2) = \sqrt{\frac{((\frac{y_1+y_2}{2} - y_m)^2 \sum_i (x_2^i - x_1^i)^2)}{(y_2 - y_1)^2}}.$$

Notice that if the two points are close to each other, the effects of this simplification will be minimal, since $x_2 - x_1$) will be close to 0. But, if they are far apart, the Line Distance will be increased since its denominator becomes smaller compared to Eq. 1.

To avoid a discontinuity whenever $y_1 = y_2$ we add:

$$LD(P_1, P_2) = \sqrt{\frac{((\frac{y_1+y_2}{2} - y_m)^2 \sum_i (x_2^i - x_1^i)^2)}{(y_2 - y_1)^2 + \epsilon}}. \quad (2)$$

■

With this simplification the dissimilarity values of the previous example become $LD(x_1, x_2) = 7.66$ and $LD(x_2, x_3) = 0.01$.

The Line Distance metric is closely related to the Hill Valley Algorithm [11], an algorithm that samples a number of intermediate points in order to infer whether two points were located at the same basis of attraction. For a maximization objective, if any intermediate point has a higher objective-function value than the tested points, then they are located at the same basis of attraction. They both exploit the approximation to the objective-function, but they do differ in some points:

- Line Distance is formulated as a mathematical function while Hill Valley is an algorithm.

- Hill Valley will return 0 whenever it detects that both points are at the same basis or a continuous value if they are not, Line Distance will always return a continuous value representing how close the points are from each other.
- If every interior point fails to find a higher peak, Hill Valley may report a false negative, Line Distance will report a false positive if the middle point hits a peak of the same height.
- If two points are at opposite sides from the same basis of attraction, Hill Valley will detect it, Line Distance will not.

This last point was an important factor to choose the Line Distance for the proposed algorithm, as it will be required that both sides of a basis of attraction are preserved for further exploration.

III. Maximization of Line Distance

The rationale for maximizing the Line Distance is that each solution of a populational algorithm will improve only if it moves farther away from its neighbors and into a local optima. Following this intuition, the first idea would be to just replace the objective-function of each solution with the average Line Distance of this solution to the whole population. But by doing so, it will lead to some problems.

First, if two points x_1 and x_2 are not located at two adjacents optima, the middle point x_m may be located at a different local optima of equal height that stands between them. This wrongly makes the Line Distance between these two points assume a value close to zero. Notice that this situation may not be rare in functions with evenly distributed local optima.

Second, the improvement of one solution will affect all other solutions thus defining a noisy objective-function that can be hard to deal with.

As such, to avoid these problems, a new meta-heuristic will be proposed to take advantage of the maximization of the Line Distance. This meta-heuristic is inspired by two already established algorithms, Artificial Immune Network for Dynamic Optimization (dopt-aiNet) [8], that has a dynamic population size in order to adapt to the number of local optima, and Nearest-better Clustering Algorithm (NEA2) [12] that works by segmenting the search space by measuring the distance of a solution to its nearest neighbor, then applying the CMA-ES algorithm [13] to each of the found regions.

The whole idea is to devise a heuristic that, given a local optimum, finds another nearby local optimum. By starting with a single solution, the population will be dynamically created by applying this heuristic to each existing solution and expanding the explored search space. In order to organize the population of solutions, they will be organized as a network of local optima.

So, the proposed meta-heuristic starts with a single node network representing a solution located at the middle of the search space. At every iteration, it samples a set of nodes from the population and, for each sampled node, it applies a local search heuristic to connect it to its nearest optimum given a random direction, thus creating a new edge. After some iterations, the network is rebuilt by eliminating redundant nodes that are within a radius of each other. The edges are then recreated by connecting each node to its nearest neighbor.

A general description of this algorithms, named *LinkedOpt* [2], is given in Alg. 1 and it is followed by a more descriptive explanation.

input : max iterations $maxIT$, number of expanded nodes per iteration $nnodes$, radius of similarity $thrE$, $thrL$, lower and upper bound of the objective-function lb, ub
output: network of solutions G

$x_0 \leftarrow lb + 0.5 \cdot (ub - lb)$;
$InsertNode(G, x_0)$;
for $it \leftarrow 1$ **to** $maxIT$ **do**
  $nodes \leftarrow SelectNodes(G, nnodes)$;
  $G \leftarrow ExpandNodes(G, nodes, thrE)$;
  $G \leftarrow Suppress(G, thrE, thrL)$;
end

Algorithm 1: LinkedOpt meta-heuristic

The function *SelectNodes()* will sample $nnodes$ nodes from the network at random but with probability inversely proportional to their degree. This probability ensures that the nodes located at unexplored areas (the leaves of a tree-like structure) will have preference to be expanded, while those at already explored areas are less likely to continue its expansion.

After the node selction, each selected node is expanded by sampling an unity vector d representing a random direction. Then, the function $F(\alpha) = LD(x, x + \alpha \cdot d)$ is maximized. There are many methods capable of tackling unidimensional search if the objective-function is convex, unimodal and has no discontinuity. But since there may be many local optima with different spacing along the random direction, this function becomes multimodal.

As pointed in [8], this can be dealt with by segmenting the one-dimensional search space into smaller spaces, alleviating the multimodality problem. In this algorithm, the bounds of each segment will be defined as $[P(i), P(i+1)]$ taken from the geometric progression $P(i) = x + thrE * 2.0^{i-1}$, beggining with $i = 0$ and $P(0) = x$; $thrE$ is the radius in which two solutions are considered redundant. The final segment will be the last one of this progression that stays within the objective-function bounds.

The rationale for this segmentation is that the search radius will be smaller nearby the current node in order to improve the already found solutions and it will be larger farther away from the node, allowing the exploration of the search space. The unidimensional optimization approach used here is the Brent method [14].

After the optimization of each segment, some decisions are performed to whether to replace the current node or to connect it to the new solution. Given an optimal α of one segment, we define $x^* = x + \alpha \cdot d$ and $x_m = 0.5 \cdot (x - x^*)$, also, y, y^*, y_m will refer to the objective-function values of the corresponding points.

[2]The source-code is available at https://github.com/folivetti/LINKEDOPT

If $y_m > y, y^*$ or $y^* > y, y_m$, then x cannot possibly be a global maximum and then its corresponding node is replaced by x_m or x^* accordingly. If $y_m < y, y^*$, then it is assumed that x_m is a local minima, and x and x^* are located at the basis of attraction of different optima. In this situation, x and x^* are both kept on the population and x^* is linked to the closest node inside the network. This is depicted in Alg. 2.

```
input : network of solutions G, list of nodes to expand
        nodes, initial τ0
output: network of solutions G
for x ← nodes do
    τ ← τ0;
    while lb ≤ x + 2·τ·d ≤ ub do
        α ← arg max_α LD(x, x + α·d), τ ≤ α ≤ 2τ;
        x* ← x + α·d;
        x_m ← 0.5·(x + x*);
        if f(x_m) > f(x), f(x*) then
            x ← x_m;
        end
        else if f(x*) > f(x), f(x_m) then
            x ← x*;
        end
        else if f(x_m) < f(x), f(x*) then
            InsertNode(G, x*);
        end
        τ ← 2τ;
    end
end
```

Algorithm 2: ExpandNodes function

Finally, at the suppression step the nodes are grouped by the distance to each other, i.e., every node with Euclidean distance less than $thrE$ or Line Distance less than $thrL$ will be grouped together. The node with the maximum objective-function of each group is kept in the network and connected with its nearest-neighbor afterwards. Despite its drawbacks, the role of the Euclidean Distance at this step is to avoid the maintanence of a high number of nodes in regions with many local optima, thus guaranteeing to keep just one optimum per region.

After the suppression, a local search was perform to finetune each of the nodes. For the problems with just a few local optima, the L-BFGS-B algorithm [15] is performed for each representative node. The problems with many local optima or higher dimensions (≤ 5), the CMA-ES algorithm is performed centered by the representative point and with variance calculated by the clustered points.

Because of the unidimensional search step and the local search, this meta-heuristic is very costly regarding function evaluations, but in the next section an experimental setup will be devised in order to compare if the higher costs are compensated by a better set of solutions.

IV. Experimental Results

The results obtained by LinkedOpt will be assessed following the benchmark for multimodal optimization proposed in [16]. This benchmark is composed of 12 functions with different characteristics determing a total of 20 different problems of varying dimensions. The performance of each algorithm is measured by means of Average Peak Rate (PR), Average Success Rate (SR) and Function Evaluations (FES). The PR is defined as:

$$PR = \frac{1}{N} \sum_{i=1}^{N} \frac{NP_i}{NP_{opt}}, \tag{3}$$

where NP_i is the number of peaks found at the i-th run, NP_{opt} is the total number of peaks of the studied test function and N is the number of runs on the benchmark.

The SR is defined as:

$$SR = \frac{1}{N} \sum_{i=1}^{N} SR_i, \tag{4}$$

where SR_i is 1 if the algorithm found every peak on the i-th run, and 0 otherwise.

Those measures are calculated for different accuracy levels, i.e., how close the solution must be to the global optima to be considered as such. For these experiments, the number of runs (N) is defined as 50 and the accuracy levels are set to $\{1e-1, 1e-2, 1e-3, 1e-4, 1e-5\}$.

The benchmark was propose for a competition hosted by the IEEE Conference of Evolutionary Computation 2013. The results will be compared with the first and second place of this competition: Niching Evolutionary Algorithm [12] (NEA2) and Dynamic Archive Niching Differential Evolution Algorithm [17] (dADE1), respectively.

Since the proposed approach can be very demanding regarding the number of function evaluations, it will be allowed for LinkedOpt to go past the imposed limits [3]. In order to fine tune the parameters, first the algorithm was run for a large number of iterations, noting the total number of peaks found. After that, the parameters were empirically adjusted so that the number of function evaluations were minimized while maintaining the same number of peaks per run. To minimize the number of FES, the suppression and local search were performed every 20 iterations.

The results reported here was those obtained after the parameter tuning (Table I) but without restricting the number of function evaluations. In the end, the average and standard deviation of FES will be reported as well. The obtained results regarding PR and SR are reported in Tables II to VI.

For the sake of simplicity, we will focus first on the analysis of the lowest accuracy level. The first 5 problems from the benchmark are very simple one and two-dimensional problems, so all of the considered methods could reach a perfect score regarding both PR and SR. The next 5 problems creates a more challenging scenario and some of the global optima cannot be found within the specified maximum function evaluation. We can see that LinkedOpt can perform equally or better than NEA2 (considering both PR and SR) but worse than dADE1 on three of those problems.

[3] for this reason, this paper will not be a part of the competition

TABLE II. RESULTS OBTAINED WITH AN ACCURACY LEVEL OF $1e-1$.

Alg.	$F_1(1D)$		$F_2(1D)$		$F_3(1D)$		$F_4(2D)$		$F_5(2D)$	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
LinkedOpt	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
NEA2	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
dADE1	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
Alg.	$F_6(2D)$		$F_7(2D)$		$F_8(2D)$		$F_6(3D)$		$F_7(3D)$	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
LinkedOpt	0.99	0.9	0.97	0.54	0.28	0.0	0.87	0.0	1.0	1.0
NEA2	0.96	0.48	0.95	0.16	0.24	0.0	0.62	0.0	1.0	1.0
dADE1	1.0	1.0	1.0	1.0	1.0	0.46	0.83	0.0	1.0	1.0
Alg.	$F_9(2D)$		$F_{10}(2D)$		$F_{11}(2D)$		$F_{11}(3D)$		$F_{12}(3D)$	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
LinkedOpt	0.67	0.0	0.96	0.0	0.67	0.0	0.67	0.0	0.75	0.0
NEA2	0.98	0.88	0.85	0.18	0.98	0.86	0.83	0.16	0.74	0.02
dADE1	0.87	0.54	0.75	0.0	0.74	0.16	0.94	0.78	1.0	1.0
Alg.	$F_{11}(5D)$		$F_{12}(5D)$		$F_{11}(10D)$		$F_{12}(10D)$		$F_{12}(20D)$	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
LinkedOpt	1.0	1.0	0.75	0.0	0.67	0.0	0.75	0.0	0.75	0.0
NEA2	0.67	0.0	0.7	0.0	0.67	0.0	0.67	0.0	0.36	0.0
dADE1	0.89	0.6	0.96	0.84	0.66	0.02	0.5	0.24	0.08	0.02

TABLE I. LIST OF PARAMETERS EMPIRICALLY SET FOR LINKEDOPT IN ORDER TO OBTAIN THE HIGHEST NUMBER OF GLOBAL OPTIMA WITH A MINIMUM NUMBER OF FUNCTION EVALUATIONS.

Problem	**Iterations**	**Samples**	**thrE**	**thrL**
1	10	10	0.5	0.1
2	20	10	0.01	0.001
3	10	10	0.01	0.01
4	20	10	0.1	0.1
5	20	10	0.1	0.1
6	50	30	0.1	0.1
7	50	10	0.1	0.1
8	50	10	0.1	0.1
9	50	10	0.1	0.1
10	50	10	0.1	0.1
11	50	10	0.1	0.1
12	30	10	0.1	0.1
13	30	10	0.1	0.1
14	20	10	0.5	0.1
15	20	20	0.5	0.1
16	20	10	0.1	0.1
17	20	10	0.1	0.1
18	20	50	0.1	0.1
19	20	50	0.1	0.1
20	20	20	1.0	0.1

TABLE VII. NUMBER OF FUNCTION EVALUATIONS FOR EACH ALGORITHM.

Function	Allowed	LinkedOpt
F1 (1D)	$50,000$	$16,607 \pm 649$
F2 (1D)	$50,000$	$17,889 \pm 394$
F3 (1D)	$50,000$	$7,477 \pm 412$
F4 (2D)	$50,000$	$34,023 \pm 263$
F5 (2D)	$50,000$	$18,616 \pm 311$
F6 (2D)	$200,000$	$162,199 \pm 175$
F7 (2D)	$200,000$	$411,658 \pm 179$
F8 (2D)	$400,000$	$321,286 \pm 240$
F6 (3D)	$400,000$	$399,145 \pm 497$
F7 (3D)	$200,000$	$44,248 \pm 561$
F9 (2D)	$200,000$	$394,487 \pm 6,260$
F10 (2D)	$200,000$	$222,752 \pm 6,433$
F11 (2D)	$200,000$	$610,439 \pm 38,758$
F11 (3D)	$400,000$	$299,336 \pm 14,583$
F12 (3D)	$400,000$	$587,466 \pm 54,325$
F11 (5D)	$400,000$	$469,280 \pm 40,121$
F12 (5D)	$400,000$	$603,508 \pm 45,308$
F11 (10D)	$400,000$	$505,460 \pm 39,544$
F12 (10D)	$400,000$	$394,26 \pm 39,068$
F12 (20D)	$400,000$	$461,564 \pm 46,182$

In the next 5 problems, encompassing composition functions, LinkedOpt had a worse performance than both contenders. But, by further examination of the obtained solutions, LinkedOpt had difficulties on finding the two global optima of Weierstrass function, that contains many local optima.

But, the surprising results were obtained on the final set of 5 problems, on higher dimensions. The results show that LinkedOpt was unaffected by the dimension increase while just slightly decreasing the number of optima found on the previous set. This, though, comes with the cost of requiring more function evaluations.

When inspecting the other level of accuracy, another drawback of LinkedOpt is noticed. This algorithm can roughly find the regions containing local optima but lacking accuracy in comparison to other approaches.

Finally, in Table VII, the average and standard deviation of the number of function evaluations for each problem are compared to the allowed number of evaluations for the competition. This table shows that LinkedOpt needs a higher number of FES to find the global optima whenever the function has many local optima. This is to be expected because the main operator of LinkedOpt tries to enumerate all of the local optima on a given random direction, so if a function has many local optima it will try to explore each one of them.

To illustrate the behavior of LinkedOpt, Figs. 2 to 6 depicts the network created after 5 iterations and after th suppression step. Notice that for F12 (20D) it was plotted just the first two axis. It is interesting to notice that, with just few iterations, the solutions are already spread accross the search space and around the local optima. This indicates that the proposed algorithm is balanced regarding exploration and exploitation, a feature sought by any populational meta-heuristics.

V. CONCLUSION

In this paper a new metaheuristic for multimodal optimization was proposed based on the Line Distance measure introduced on a prior work. The Line Distance is capable of asserting whether two solutions are located at the basis of attraction of the same optima or not. By exploiting this property, this metaheuristic, called LinkedOpt, tried to explore the search space by maximizing this Line Distance between a given point and the points along a random direction.

Given enough time, this approach converges toward a

TABLE III. RESULTS OBTAINED WITH AN ACCURACY LEVEL OF $1e-2$.

Alg.	$F_1(1D)$		$F_2(1D)$		$F_3(1D)$		$F_4(2D)$		$F_5(2D)$	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
LinkedOpt	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
NEA2	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
dADE1	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
Alg.	$F_6(2D)$		$F_7(2D)$		$F_8(2D)$		$F_6(3D)$		$F_7(3D)$	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
LinkedOpt	0.99	0.9	0.29	0.0	0.28	0.0	0.12	0.0	1.0	1.0
NEA2	0.96	0.48	0.92	0.08	0.24	0.0	0.59	0.0	1.0	1.0
dADE1	1.0	1.0	1.0	0.18	1.0	0.5	0.59	0.0	1.0	1.0
Alg.	$F_9(2D)$		$F_{10}(2D)$		$F_{11}(2D)$		$F_{11}(3D)$		$F_{12}(3D)$	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
LinkedOpt	0.67	0.0	0.75	0.0	0.67	0.0	0.67	0.0	0.75	0.0
NEA2	0.97	0.8	0.85	0.18	0.97	0.82	0.82	0.1	0.72	0.0
dADE1	0.67	0.0	0.75	0.0	0.67	0.0	0.67	0.0	0.64	0.0
Alg.	$F_{11}(5D)$		$F_{12}(5D)$		$F_{11}(10D)$		$F_{12}(10D)$		$F_{12}(20D)$	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
LinkedOpt	0.67	0.0	0.75	0.0	0.67	0.0	0.75	0.0	0.75	0.0
NEA2	0.67	0.0	0.7	0.0	0.67	0.0	0.67	0.0	0.36	0.0
dADE1	0.67	0.0	0.48	0.0	0.63	0.0	0.12	0.0	0.0	0.0

TABLE IV. RESULTS OBTAINED WITH AN ACCURACY LEVEL OF $1e-3$.

Alg.	$F_1(1D)$		$F_2(1D)$		$F_3(1D)$		$F_4(2D)$		$F_5(2D)$	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
LinkedOpt	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
NEA2	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
dADE1	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
Alg.	$F_6(2D)$		$F_7(2D)$		$F_8(2D)$		$F_6(3D)$		$F_7(3D)$	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
LinkedOpt	0.99	0.9	0.25	0.0	0.28	0.0	0.12	0.0	1.0	1.0
NEA2	0.96	0.44	0.92	0.06	0.24	0.0	0.58	0.0	1.0	1.0
dADE1	1.0	1.0	0.88	0.0	1.0	0.3	0.55	0.0	1.0	1.0
Alg.	$F_9(2D)$		$F_{10}(2D)$		$F_{11}(2D)$		$F_{11}(3D)$		$F_{12}(3D)$	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
LinkedOpt	0.67	0.0	0.75	0.0	0.67	0.0	0.67	0.0	0.75	0.0
NEA2	0.97	0.8	0.84	0.18	0.96	0.76	0.81	0.08	0.72	0.0
dADE1	0.67	0.0	0.74	0.0	0.67	0.0	0.67	0.0	0.62	0.0
Alg.	$F_{11}(5D)$		$F_{12}(5D)$		$F_{11}(10D)$		$F_{12}(10D)$		$F_{12}(20D)$	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
LinkedOpt	0.67	0.0	0.75	0.0	0.67	0.0	0.75	0.0	0.75	0.0
NEA2	0.67	0.0	0.7	0.0	0.67	0.0	0.67	0.0	0.36	0.0
dADE1	0.67	0.0	0.42	0.0	0.63	0.0	0.08	0.0	0.0	0.0

TABLE V. RESULTS OBTAINED WITH AN ACCURACY LEVEL OF $1e-4$.

Alg.	$F_1(1D)$		$F_2(1D)$		$F_3(1D)$		$F_4(2D)$		$F_5(2D)$	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
LinkedOpt	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
NEA2	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
dADE1	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
Alg.	$F_6(2D)$		$F_7(2D)$		$F_8(2D)$		$F_6(3D)$		$F_7(3D)$	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
LinkedOpt	0.99	0.9	0.25	0.0	0.28	0.0	0.12	0.0	1.0	1.0
NEA2	0.95	0.38	0.91	0.04	0.24	0.0	0.58	0.0	0.99	0.86
dADE1	1.0	0.78	0.81	0.0	0.96	0.16	0.44	0.0	1.0	1.0
Alg.	$F_9(2D)$		$F_{10}(2D)$		$F_{11}(2D)$		$F_{11}(3D)$		$F_{12}(3D)$	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
LinkedOpt	0.67	0.0	0.0	0.0	0.67	0.0	0.67	0.0	0.12	0.0
NEA2	0.96	0.76	0.84	0.16	0.96	0.74	0.81	0.06	0.72	0.0
dADE1	0.67	0.0	0.74	0.0	0.67	0.0	0.67	0.0	0.6	0.0
Alg.	$F_{11}(5D)$		$F_{12}(5D)$		$F_{11}(10D)$		$F_{12}(10D)$		$F_{12}(20D)$	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
LinkedOpt	0.67	0.0	0.0	0.0	0.67	0.0	0.5	0.0	0.31	0.0
NEA2	0.67	0.0	0.7	0.0	0.67	0.0	0.67	0.0	0.36	0.0
dADE1	0.67	0.0	0.4	0.0	0.61	0.0	0.02	0.0	0.0	0.0

TABLE VI. RESULTS OBTAINED WITH AN ACCURACY LEVEL OF $1e-5$.

Alg.	$F_1(1D)$		$F_2(1D)$		$F_3(1D)$		$F_4(2D)$		$F_5(2D)$	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
LinkedOpt	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
NEA2	1.0	1.0	1.0	1.0	1.0	1.0	0.99	0.96	1.0	1.0
dADE1	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
Alg.	$F_6(2D)$		$F_7(2D)$		$F_8(2D)$		$F_6(3D)$		$F_7(3D)$	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
LinkedOpt	0.0	0.0	0.25	0.0	0.28	0.0	0.12	0.0	1.0	1.0
NEA2	0.0	0.0	0.91	0.04	0.24	0.0	0.58	0.0	0.98	0.76
dADE1	0.0	0.0	0.71	0.0	0.95	0.1	0.35	0.0	1.0	1.0
Alg.	$F_9(2D)$		$F_{10}(2D)$		$F_{11}(2D)$		$F_{11}(3D)$		$F_{12}(3D)$	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
LinkedOpt	0.42	0.0	0.0	0.0	0.38	0.0	0.38	0.0	0.0	0.0
NEA2	0.96	0.76	0.83	0.14	0.95	0.7	0.8	0.06	0.71	0.0
dADE1	0.67	0.0	0.74	0.0	0.67	0.0	0.67	0.0	0.62	0.0
Alg.	$F_{11}(5D)$		$F_{12}(5D)$		$F_{11}(10D)$		$F_{12}(10D)$		$F_{12}(20D)$	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
LinkedOpt	0.0	0.0	0.0	0.0	0.0	0.0	0.25	0.0	0.19	0.0
NEA2	0.67	0.0	0.7	0.0	0.66	0.0	0.67	0.0	0.35	0.0
dADE1	0.67	0.0	0.36	0.0	0.6	0.0	0.0	0.0	0.0	0.0

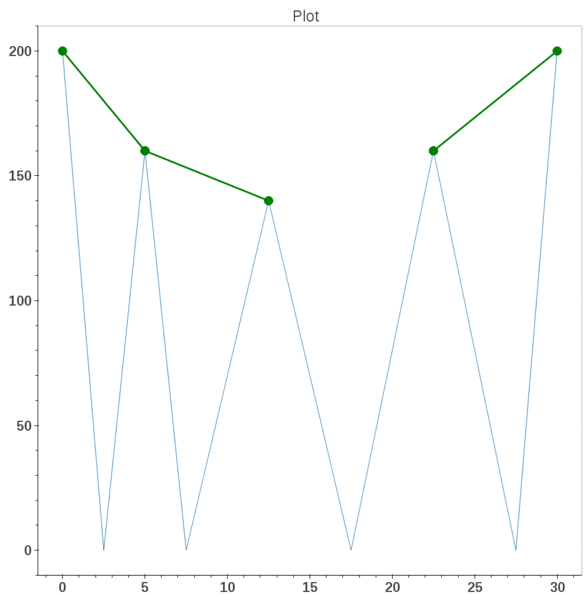

Fig. 2. Network created by LinkedOpt for Function F1 (1D) after 5 iterations.

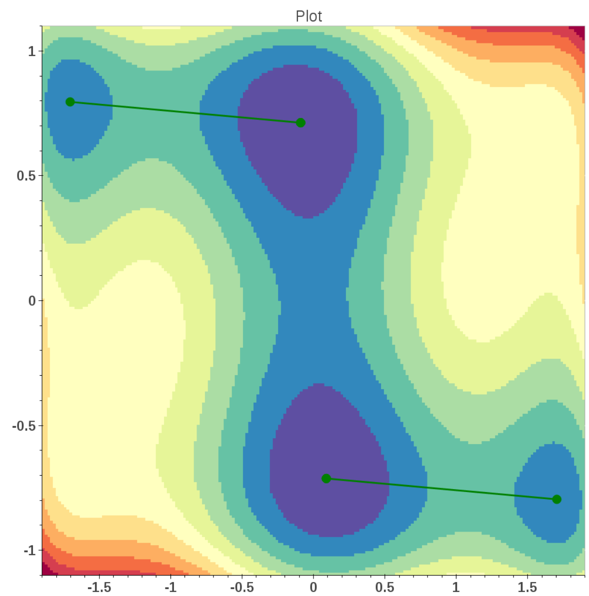

Fig. 3. Network created by LinkedOpt for Function F4 (2D) after 5 iterations.

network of local optima, where each node is a local optima and each edge connects two nearby optima.

The performance of LinkedOpt was assessed by means of a multimodal optimization benchmark proposed in 2013. These experiments showed that while LinkedOpt demands a higher number of function evaluations, it seems to consistently explore the search space, skipping from one optima to another, even on higher dimensions.

As for future works, there are some points that needs further investigation:

- Use of a better data structure to express the relationship between the solution taking the Euclidean and Line Distances into account.
- Exploit such data structure in order to minimize the number of function evaluations.
- Favor unexplored regions when sampling the distance unity vector.
- Improve the Line Distance maximization step in order to avoid the need for local search.

Finally, it should be noted that the expanding node algorithm can be incorporated in other multimodal optimization algorithms as a new local search or mutation operator.

ACKNOWLEDGMENT

The author would like to thank CAPES for the financial support.

REFERENCES

[1] T. Bäck and H.-P. Schwefel, "An overview of evolutionary algorithms for parameter optimization," *Evolutionary computation*, vol. 1, no. 1, pp. 1–23, 1993.

[2] J. Horn and D. E. Goldberg, "Genetic algorithm difficulty and the modality of fitness landscapes," in *Foundations of genetic algorithms 3*. Citeseer, 1994.

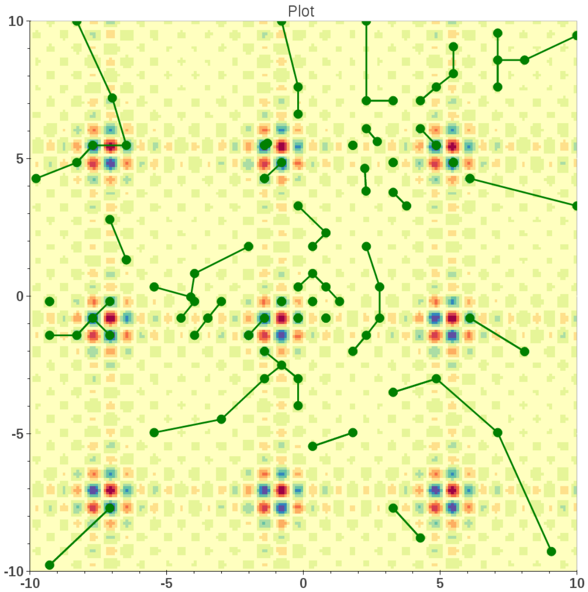

Fig. 4. Network created by LinkedOpt for Function F6 (2D) after 5 iterations.

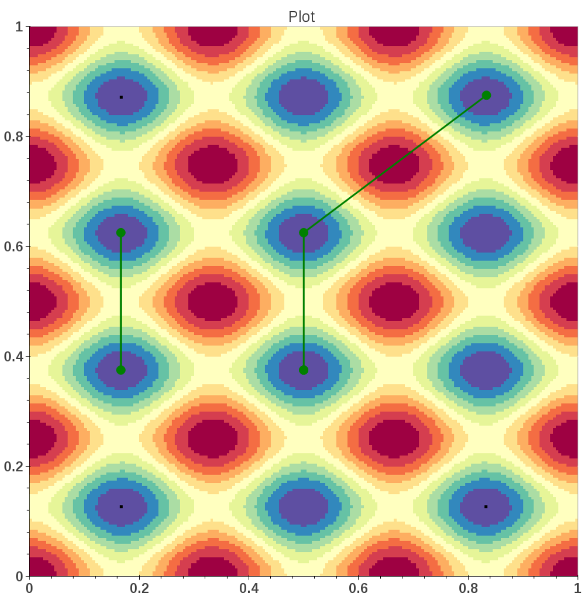

Fig. 5. Network created by LinkedOpt for Function F8 (2D) after 5 iterations.

[3] J. Vesterstrom and R. Thomsen, "A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems," in *Evolutionary Computation, 2004. CEC2004. Congress on*, vol. 2. IEEE, 2004, pp. 1980–1987.

[4] D. E. Goldberg and J. Richardson, "Genetic algorithms with sharing for multimodal function optimization," in *Genetic algorithms and their applications: Proceedings of the Second International Conference on Genetic Algorithms*. Hillsdale, NJ: Lawrence Erlbaum, 1987, pp. 41–49.

[5] B. Sareni and L. Krahenbuhl, "Fitness sharing and niching methods revisited," *Evolutionary Computation, IEEE Transactions on*, vol. 2, no. 3, pp. 97–106, 1998.

[6] F. de Franga, G. P. Coelho, and F. J. Von Zuben, "On the diversity mechanisms of opt-ainet: A comparative study with fitness sharing," in *Evolutionary Computation (CEC), 2010 IEEE Congress on*. IEEE, 2010, pp. 1–8.

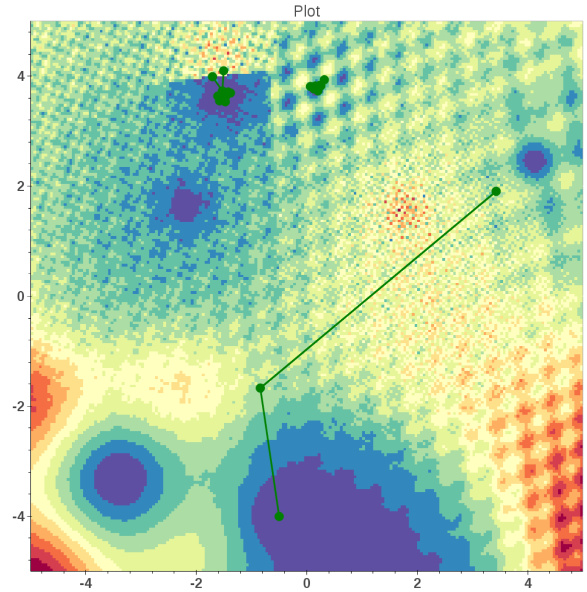

Fig. 6. Network created by LinkedOpt for Function F12 (20D) after 5 iterations.

[7] Z. Michalewicz, "Quo vadis, evolutionary computation?" in *Advances in Computational Intelligence*. Springer, 2012, pp. 98–121.

[8] F. O. de Franca, F. J. Von Zuben, and L. N. de Castro, "An artificial immune network for multimodal function optimization on dynamic environments," in *Proceedings of the 2005 conference on Genetic and evolutionary computation*. ACM, 2005, pp. 289–296.

[9] F. O. de França, G. P. Coelho, P. A. Castro, and F. J. Von Zuben, "Conceptual and practical aspects of the ainet family of algorithms," *International Journal of Natural Computing Research (IJNCR)*, vol. 1, no. 1, pp. 1–35, 2010.

[10] L. N. de Castro and J. Timmis, "An artificial immune network for multimodal function optimization," in *Evolutionary Computation, 2002. CEC'02. Proceedings of the 2002 Congress on*, vol. 1. IEEE, 2002, pp. 699–704.

[11] M. Jelasity, "Uego, an abstract niching technique for global optimization," in *Parallel Problem Solving from NaturePPSN V*. Springer, 1998, pp. 378–387.

[12] M. Preuss, "Improved topological niching for real-valued global optimization," in *Applications of Evolutionary Computation*. Springer, 2012, pp. 386–395.

[13] N. Hansen, "The cma evolution strategy: a comparing review," in *Towards a new evolutionary computation*. Springer, 2006, pp. 75–102.

[14] R. P. Brent, "An algorithm with guaranteed convergence for finding a zero of a function," *The Computer Journal*, vol. 14, no. 4, pp. 422–425, 1971.

[15] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu, "A limited memory algorithm for bound constrained optimization," *SIAM Journal on Scientific Computing*, vol. 16, no. 5, pp. 1190–1208, 1995.

[16] X. Li, A. Engelbrecht, and M. G. Epitropakis, "Benchmark functions for cec2013 special session and competition on niching methods for multimodal function optimization," *Evolutionary Computation and Machine Learning Group, RMIT University, Melbourne, Australia, Tech. Rep*, 2013.

[17] M. G. Epitropakis, V. P. Plagianakos, and M. N. Vrahatis, "Finding multiple global optima exploiting differential evolution's niching capability," in *Differential Evolution (SDE), 2011 IEEE Symposium on*. IEEE, 2011, pp. 1–8.