2014 IEEE Congress on Evolutionary Computation (CEC)
July 6-11, 2014, Beijing, China

Visualizing the Population of Meta-heuristics During the Optimization Process Using Self-Organizing Maps

Marcelo Lotif

Abstract— **This study proposes a novel Visual Data Mining technique based on Self-Organizing Maps (SOM) to visualize the population points of metaheuristic algorithms while they execute their search process. The SOM is used to divide the search space of the optimization function into bi-dimensional regions, allowing one to perform a visual analysis by mapping the points into the 2-dimensional space, in order to compare various executions of the functions performed with different parameter configurations. The use of these maps as a Visual Data Mining tool aims to visually process the resulting data and identify behavioral patterns of the meta-heuristic instances.**

I. Introduction

THE correlated fields of Information Visualization and Visual Data Mining consist of a set of techniques that allow the analysis of a given set of data in an interactive manner. Those techniques aim to include the human knowledge, flexibility, creativity and perceptual abilities to the analysis process to improve the results [2] [3]. The human visual system has the unique capability of detecting important features and patterns. Therefore data visualization makes it possible for researchers, analysts, engineers, and the lay audience to obtain insight in these data in an efficient and effective way [4]. The advent of interactive data presentation and query resources tools have allowed domain experts to examine different scenarios and identify hidden data patterns more quickly [5].

Visualization techniques have been used in order to obtain a better understanding of how meta-heuristics perform the search for the optimal solution. This knowledge is crucial to choosing the most appropriate meta-heuristic and performing its subsequent calibration with respect to the problem being optimized. Visualization tools are very useful in this context to help analyze multidimensional data collected during the search process, so one can for instance trace the source of solutions, the survival of partial solutions, the effects of tuning the operators of the algorithm or obtain population statistics and trends [7]. These tools have been applied to a number of meta-heuristic algorithms serving different purposes [6]-[16].

Some of the Information Visualization and Visual Data Mining approaches employ Self-Organizing Maps (SOM) as a tool for data visualization [25]. SOMs [17] [18] are a specific class of Artificial Neural Networks in which the neurons are positioned in a space that is usually uni- or bi-dimensional, where they are selectively adapted to the patterns received as input (stimuli) during a process of competitive learning. SOMs have been used successfully in a variety of fields, including pattern recognition, prediction of time series, image compression, process monitoring, speech recognition and fault diagnosis, among others [19] [20]. As a Visual Data Mining tool, there are applications in clustering [21] [29] [33], visualization of protein sequences [22], demographic trajectories [34] and crime trajectories [35], for example. There are a number of characteristics of SOM that make it remarkably useful for visualization. As stated by [24], it implements an ordered dimensionality reduction mapping of the training data, follows the probability density function of the original data and is robust to missing data. It is simple, readily explainable and easy to visualize.

This paper presents a novel technique to visualize high dimensional data sets produced by collecting information during the search process of meta-heuristics using Self-Organizing Maps as the visualization tool. It allows to compare the data of meta-heuristic executions over time in a visual manner, aiming to take advantage of the positive features of SOMs to perform these comparisons. The rest of this paper is organized as follows: Section II presents relevant methods to visualize data produced by meta-heuristics and how SOMs are used as a visualization tool. Section III introduces the method to divide the search space of the optimization function into regions and map the data set to the resultant neuron lattice trained by the SOM. Section IV shows the application of the technique that compares different executions of one meta-heuristic algorithm. Conclusions are presented at Section V.

II. State of the art

A. Visualizing data collected regarding the search process of meta-heuristics

The simple analysis of the final outcome of an optimization algorithm and the comparison with other algorithms that have the same purpose has become a very common practice in many tests and proofs of concept in scientific publications that propose new algorithms and approaches to problem solving. Thus, the algorithm execution is treated as a black box, and the internal process of searching for the best solution to the problem is not being analyzed. What must also be taken into consideration is how these algorithms and approaches generate new results and how their operators work to reach the final result according to the calibration of its parameters. This information can be really valuable to the appropriate choice of meta-heuristics.

This paper presents and extends part of the study developed at the Graduate Program in Applied Informatics (PPGIA) of the University of Fortaleza (UNIFOR) by the author (e-mail: marcelolotif@gmail.com) as part of his master thesis [1]

978-1-4799-1488-3/14/$31.00 ©2014 IEEE

A number of studies have been carried in order to better understand how meta-heuristics explore and exploit the search space seeking the best solution. A set of techniques to visually analyze the optimization process of evolutionary algorithms is presented by [6]. Those techniques aim to produce various charts to display different data sets in order to give a visual impression of the evolutionary algorithm's progress and the actual state of the individuals of the population. These data sets are easy to generate and display using standard visualization tools, providing a base line for understanding the evolutionary process.

VIS is an off-line visualization tool developed by [7] to allow the analysis of Genetic Algorithms (GA) executions. It was designed with two main goals: (a) allow users to examine the details of a GA run, providing easy access to the desired information and easy transition between related pieces of information; and (b) to develop methods and representations for displaying multi-dimensional data in coherent and informative manner. Some of the functionalities of this tool include viewing snapshots of the GA population through time, examination of specific individuals, and navigating forwards and backwards through an execution.

In [8], another off-line graphical tool called GAVEL is described to analyze the optimization process of the GA. Its main feature is to inspect the best solution and trace its ancestors, performing a backwards analysis in a form of an ancestry tree. One can not only trace the history of this individual, but also the history of each of its genes back to the one who has originated it. The access to such information enables the understanding whether the GA is the right algorithm for the problem being treated, as well as assisting in the calibration of their operators to reach an optimization process that best fits the situation.

In the work of [9], several studies are cited in the area of visualization of the GA, which are categorized according to the granularity of the information produced. Those categories separate the visualization methods that collect information in terms of fitness, genes, chromosomes and problem space. Additionally, a visualization framework is provided to observe the evolutionary process from the population's perspective, with the information being presented in the chart as a single visual point. The objective of the framework is to observe the overall changes in the population, as well as any relationship between the generations during the executions of the GA. When combined, the diagrams proposed by the authors allow the user to observe trends and evolutionary dynamics.

The method explained by [11] aims to produce a visual display of the chromosomes of the GA so that other information not available on the chart of fitness value over time become apparent. Using Sammon mapping as a method of dimensionality reduction and presenting it in an interesting way, the method allows one to get important information about what happens during the execution of the algorithm. In this study, a Genetic Algorithm is used to optimize 5-bit binary strings, projected to two dimensions with the aid of Sammon mapping. The circles represent the individuals in the search space: the larger the circle, the more people are in this area of the chart. The fill color of each circle indicates the value of the objective function of the individuals represented by it, while lighter colors in grayscale represent values closer to the global optimum, thus being the most promising regions of the search space.

A visualization tool called VIZ that displays ways to analyze trajectory meta-heuristics when subjected to the traveling salesman problem is presented in the work of [12]. The purpose of this tool is to find a meta-heuristic that best fits the instance of the problem being optimized, as well as the most suitable group of parameter values. It is observed that the study of the parameter calibration takes the majority of the efforts, when 90% of design and test time can be spent on the fine calibration of the algorithm. The purpose of the study as a whole, therefore, is to reduce this effort through a tool that assists decision making by analyzing the behavior of the internal mechanisms of meta-heuristics.

Another visual tool is proposed by [13] to help tuning the Tabu Search algorithm, which is called Visualizer for Metaheuristics Development Framework (V-MDF). Within it, there is a functionality called Distance Radar that measures the distance between the solutions that are found by the optimization algorithm over time, with the objective of clarifying the search process and aid in the selection of the best parameter configuration regarding the problem being optimized.

The studies of [14] are directed towards the display of the search process conducted by the Particle Swarm Optimization algorithm, so that it shows the entire search process, thus allowing for a better understanding of how to configure the algorithm. The proposed method produces a simulated particle motion animation over time by drawing the position of all particles of the swarm in each iteration into the two-dimensional space. Sammon mapping is used as the dimensionality reduction method. The purpose of this visualization algorithm is to map all the points in which the individuals of the population have been throughout the search process in order to generate a kind of trail of each individual through the space. The more the iterations drawn into the chart are distant in time, the clearer their trail appears. Therefore, darker trails represent points in the search space where the individual was most recently.

A method to visualize four variations of the Harmony Search algorithm, besides the algorithm itself in its canonical form, was proposed by [15]. The technique selected for dimensionality reduction and subsequent visualization was Viz3D. The purpose is to observe how variations of the Harmony Search algorithm effectively operate and to what extent the variations mentioned alter the search pattern displayed by the original algorithm, since each one introduces considerable conceptual modifications to it. The charts show the progress of the search through the iterations, where each point corresponds to the location of the centroid obtained from the individuals that form the population of the executions in those instants of time. These centroids are then

transformed from n-dimensional points to three-dimensional points using Viz3D. The dot size increases as the average value of the evaluation function of the population approaches the optimum value of known function being optimized. In other words, the smaller the average value of the evaluation function of the population, the greater its associated point appears in the chart. This gives a measure of how the population evolves during the iterations with respect to the optimization function.

B. The use of Self-Organizing Maps as an Information Visualization and Visual Data Mining tool

Self-organizing Maps [17] [18] have been proved to be effective techniques for clustering and visualization [21], being directly compared to other well-known dimensionality reduction techniques. Among the possible techniques of visualization with Self-Organizing Maps, the most interesting in regard to the scope of this work were addressed by [22]-[24].

The Self-Organizing Map has many beneficial features that makes it an useful method for Data Mining. It implements a sort of dimensionality reduction of the data that is submitted to it as an input for the training process. The map follows the probability density function of the data and is robust when it comes to missing data. It is easy to explain, simple and – perhaps most importantly – easy viewing. The visualization of complex multidimensional data is indeed one of the main areas of application of Self-Organizing Maps [24] [25].

Other visualization techniques with Self-Organizing Maps have been proposed using distance matrices, being the U-Matrix [26] [27] the most widely implemented method, where distances between points are usually represented by color codes [25]. Other ways of representing the distances, such as the size of the cells of the map, are also used [24]. In this method, the wider the cell that separates two points, the farther the points are in the multidimensional space.

One of the variations of SOMs, called Emerging Self-Organized Maps (ESOMs) [28] is also very useful in data visualization. We can notice in works such as [29], [30] and [31] that the use of ESOMs greatly increase the quality of the diagrams, as it generates more accurate results and mappings [32], and thus facilitate the visualization of data.

As stated in Section I, there are many uses of the Self Organizing Maps as a Visual Data Mining tool. The applications range from clustering [21] [29] [33], to visualization of protein sequences [22], demographic trajectories [34] and crime trajectories [35], for example.

III. Dividing the Search Space Into Regions

Among the various useful information that may be collected regarding the execution of meta-heuristics is to know what areas of the search space are being visited during the optimization process, and when they are visited. Thus, it can be determined which regions of the search space are being explored, how frequent this exploration is, how long the algorithm takes to find the most promising region, and so on. Furthermore, one can compare the executions of various instances of meta-heuristics to determine which one is the most appropriate for the optimization problem in question. Information about the exploration of the search space has been assessed by other studies with the objective of gaining insight into the behavior of meta-heuristic algorithms [8]-[14].

The technique proposed here consists in dividing the search space into regions by mapping it into a SOM. The idea is, given a specific search space associated with an objective function, to obtain a population made of samples of that space generated via uniform distribution, so that a homogeneous area is covered by the population. A neural network is then created and trained based on these points, so that to a certain extent this layer reflects the topology of the search space at the end of the training process. An illustration of this procedure is shown in Fig. 1.

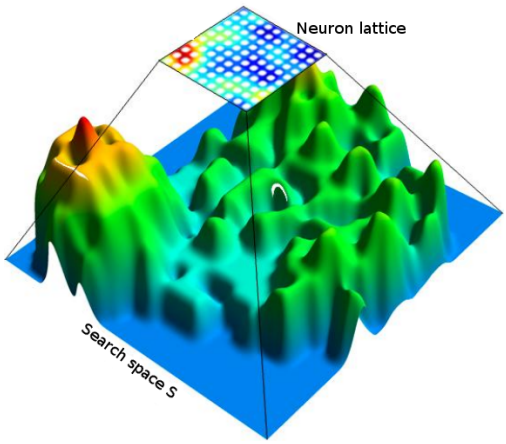

Fig. 1. Mapping a three dimensional search space S into a two-dimensional 10×10 neuron lattice of a Self-Organizing Map

Each neuron of the resultant network can be understood as corresponding to a certain region of the n-dimensional space, since the sample individuals of the population that was used as the input are uniformly scattered throughout the search space. Therefore, one can determine which region of the search space a particular individual of the population is in by using this individual as an input for the layer of neurons, and verifying which one neuron is activated by it.

Consider a population $P = \{p_1, p_2, \dots p_N\}$, where N is the number of individuals of the population, and a map M consisting of a set of neurons $K = \{k_1, k_2, \dots k_w\}$, where w is the number of neurons, trained according to a random population generated in normal distribution for the function f being optimized. In order to define which regions of the search space of f are being covered by the population P, its N points are subjected to the map M. When determining which and how often the k_w neurons of the map are being activated when subjected to these stimuli, one can visualize what regions of the search space are actually being visited by the points in N.

Fig. 2 depicts a sample of how these regions are being revealed for a three dimensional search space into a two

dimensional neuron lattice. Given the population P of individuals throughout the search space S, one can easily identify the areas visited by them in the search space by verifying which of the k_w neurons of the lattice are being activated by those individuals, which are marked in red. One may notice that a neuron can be activated by multiple points of the population and, because of that, these points must be categorized as being exploring the same region of the space. At the same time, one can state that, for any point p that can be generated in the search space S, there is a corresponding neuron k_w in the map M. Extrapolating this model to the n-dimensional search space, it can be determined that actually a n-to-2-dimensional mapping is being performed, in a way that any n dimensional point that can be generated for the optimization function f has a corresponding neuron in the network.

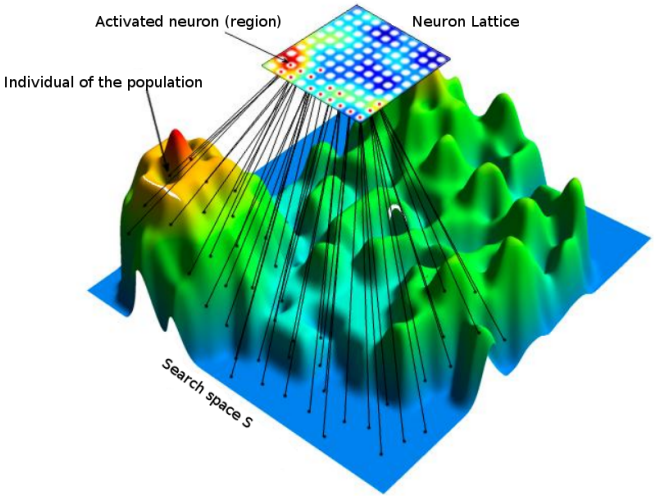

Fig. 2. Activation of the neurons in the lattice by the N individuals of a population P from a meta-heuristic while optimizing a function represented by the search space S. The neurons in red are the ones that were activated by at least one individual of the population

Using the method described above and with the aid of tools such as the Databionics ESOM Tools [32], one is able to generate images like the ones shown in Fig. 3. To create this picture, the Rastrigin multimodal test function (1) [36], whose global minimum is $x_i = 0, i = 1 \ldots n$, was used for $n = 30$ as the search space to be mapped. For the training process, the SOM algorithm was configured to be as close as the canonical specification as possible, more precisely as follows. The standard incremental training algorithm described in [18] was chosen and executed for 20 epochs. The Euclidean distance was defined as the space distance function. The initial and final learning rate were set at 0.5 and 0.1 respectively. The initial and final neighborhood set radius were 24 and 1 respectively. A linear cooling strategy was chosen for both learning rate and neighborhood set radius. For the neighborhood function, the Gaussian Kernel smoother was chosen.

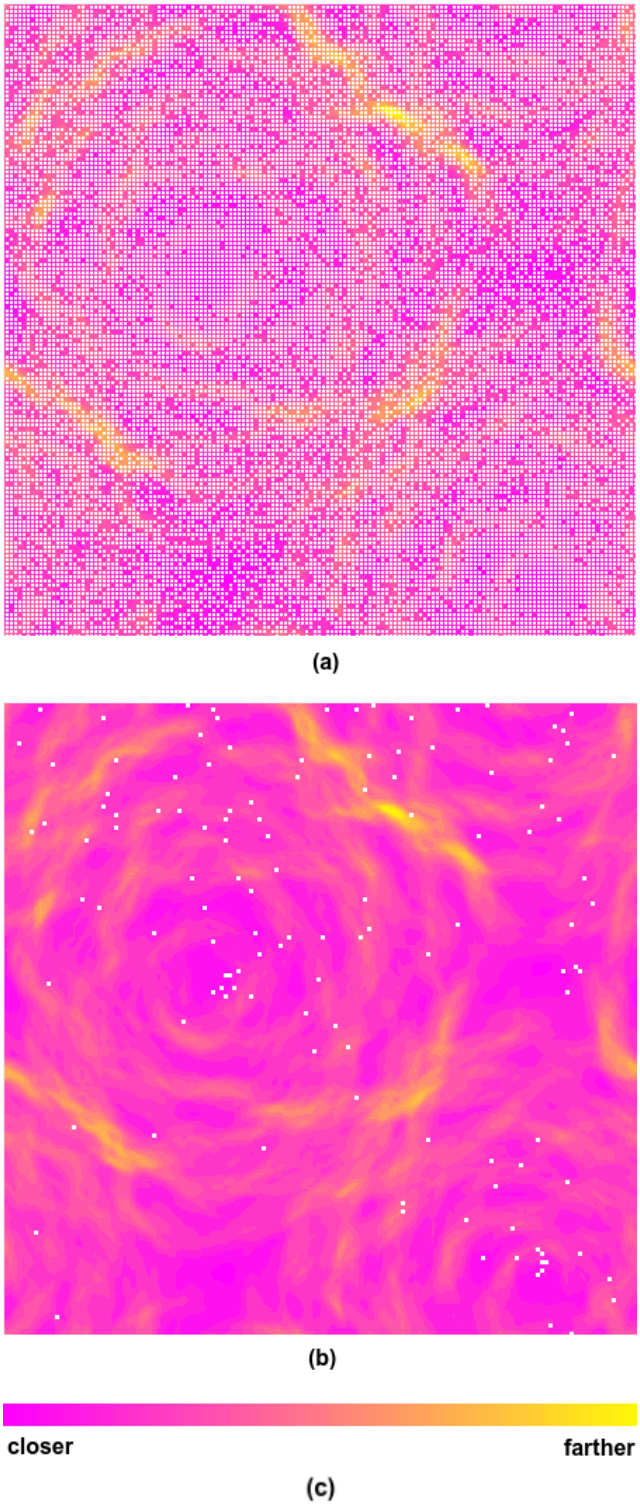

Fig. 3. A two-dimensional Self-Organized Map with 150×150 neurons trained with an input of 60,000 points generated in normal distribution for the Rastrigin function with 30 dimensions. Image (a) shows all the neurons trained by the SOM, while image (b) shows the neurons that were activated when a set of 125 random points was used as an input. Image (c) explains the color of the points in the map, where neurons separated by shades of pink are closer to each other then the ones separated by shades of yellow

$$f_{Rast} = 10n + (\sum_{i=1}^{n} x_i^2 - 10\cos(2\pi x_i)), \quad -5.12 \leq x_i \leq 5.12 \tag{1}$$

Firstly, a set of 60,000 random points of the function (1) was generated in normal distribution, and a Self-Organized Map of 150×150 neurons – configured as described earlier – was trained using this set of points as input. Fig. 3 (a) depicts the neurons of the map that were activated by the set. This way, the search space of the 30-dimensional Rastrigin function is reduced to a bi-dimensional lattice of 22,500 neurons, each one representing a given section of this search space. Secondly, a trial population of 125 random points were generated and used as an input on the trained map, to verify which neurons are being activated by the points of this trial population and consequently in which sections of the search space those points are located, as can be seen on Fig. 3 (b).

By repeating the second step of the aforementioned process to any given set of points collected during the execution of a meta-heuristic, one can visually analyze how it explores the search space of the optimization function during its search process, as can be seen on Section IV.

The colors of the neurons in the images are detailed on Fig. 3 (c), which explains that neurons separated by shades of pink are closer to each other than neurons separated by shades of yellow. To calculate the distances between them, the unified distance matrix (U-Matrix) [26][37] was used, which is a distance matrix between prototype vectors of neighboring neurons of the network. It defines a value for each one of the k_w neurons of the network based on the distance between k_w and its set of neighbors so that the farther away the neuron is of its neighbors, the higher is its corresponding value in the matrix. By projecting this matrix in the map and assigning colors to each distance value, one can have a visual insight on how the neurons are organized in the map.

IV. Experimental Application

To exemplify the use of the technique explained on Section III, the following experiment is proposed: the Rastrigin function (1) with $n = 30$ was optimized by three instances of a Genetic Algorithm, and the visualization method proposed here was used to gain insight on how their populations explore the search space through six different steps of the optimization process.

The GA was implemented according to [38], changing the chromosome codification approach from binary string to floating point array, in order to adjust it to the optimization function in question. For the selection operator of the GA, the Roulette Wheel approach was chosen. It probabilistically selects individuals based on their fitness values, giving a higher chance of reproduction for the chromosomes with best values in the fitness function [39].

The three instances of the GA that were used in this experiment are configured as described in Table I. They were initialized with a set of points generated randomly. Snapshots of the populations were collected during the optimization process on iterations 0, 5, 15, 25, 35 and 45 of a sample run of each one of the instances. Those snapshots were then used as input on the Self-Organized Map trained for a set of random points of the Rastrigin function, as detailed in Section III, to identify which regions of the search space are being visited by the points as the search process advances. The maps resulting from this process are displayed in Fig. 4, 5 and 6.

TABLE I
Configurations of the three Genetic Algorithm instances used to optimize the Rastrigin function

Instance Name	Mutation Probability	Crossover Probability	Population Size
GA1	10%	75%	125
GA2	5%	95%	125
GA3	10%	85%	85

The white dots in each image of those figures represent the neurons that were activated on the map when the populations are used as input. Since a single neuron can be activated by more than one point at the same time, one white dot on the image may represent several points of the input population. If two or more points activate the same neuron, it indicates that they are sufficiently close in the n-dimensional space to be interpreted by the map as being part of the same region, thus they are represented by the same point in the 2-dimensional space.

Analyzing the images at iteration 0 in Fig. 4, 5 and 6, one can observe that the initial population starts with random points dispersed on the search space for all three instances. As the optimization process advances, the points of the GA3 instance (Fig. 6) rapidly converge to a single region of the map between iterations 15 and 25. There is a subset of points that migrate to a different region between iterations 25 and 35, in an attempt of escaping from local minima by the algorithm. The combination of high mutation and crossover probabilities for this instance (85% and 10% respectively) may have played an active role on this behavior. However, those points return to the region they first converged between the 35th and 45th iterations, and the search continues with the exploitation of this one region.

For the GA1 instance (Fig. 4), it can be observed that 7 different regions are being explored at iteration 35, still an elevated number at this point of the search when compared to GA2 and GA3 instances. It may be a consequence of the combination of a high mutation probability (10%) with a fairly low crossover probability (75%). Soon after, this number decreases significantly to a single region on the 45th iteration. It indicates that a point with a very good fitness value as compared to the rest of the population was found on that time frame, which makes it recombine with others more frequently than the less fitted points, leading to this steep convergence.

The GA2 instance (Fig. 5) had a smoother convergence

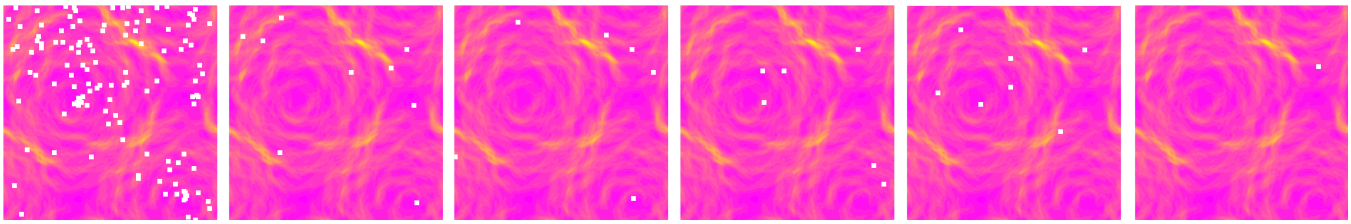

Fig. 4. The Genetic Algorithm instance GA1 optimizing the Rastrigin function with 30 dimensions at iterations 0, 5, 15, 25, 35 and 45. This instance is configured with 125 individuals, 75% of crossover probability and 10% of mutation probability.

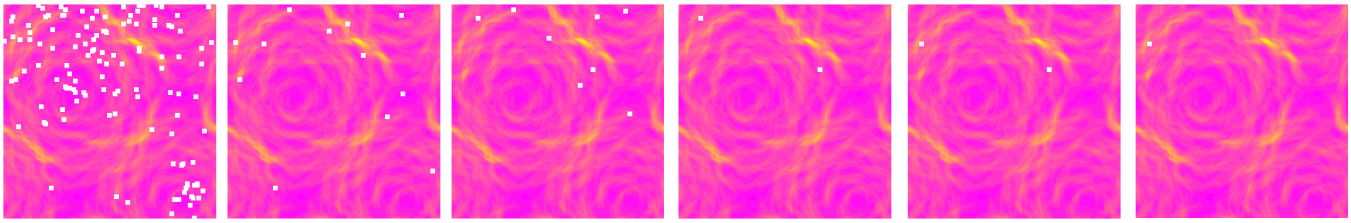

Fig. 5. The Genetic Algorithm instance GA2 optimizing the Rastrigin function with 30 dimensions at iterations 0, 5, 15, 25, 35 and 45. This instance is configured with 125 individuals, 95% of crossover probability and 5% of mutation probability.

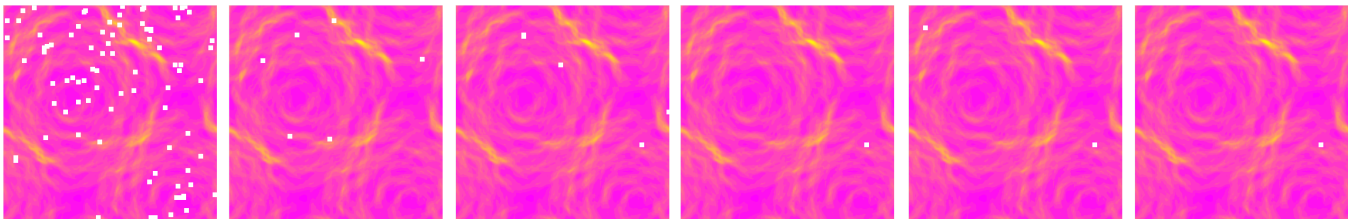

Fig. 6. The Genetic Algorithm instance GA3 optimizing the Rastrigin function with 30 dimensions at iterations 0, 5, 15, 25, 35 and 45. This instance is configured with 85 individuals, 85% of crossover probability and 10% of mutation probability.

than the others, going from 12 regions explored at the 5th iteration, to 8 at the 15th, 2 at the 25th and 35th, and finally concentrating all points in one single region at the 45th iteration. Combining the highest crossover probability (95%) with the lowest mutation probability (10%) may have caused this distinct behavior in comparison to the other two instances.

V. Final Remarks

This paper presented a novel technique to visualize n-dimensional points produced during the optimization process of meta-heuristics. By resorting to Self-Organizing Maps [17][18] and their well-known characteristics when used for dimensionality reduction [24], it was possible to design a method to visualize high dimensional points into a bi-dimensional space, in a way that one can verify which regions of the search space are being visited by the points of a meta-heuristic population through various moments of the optimization process. Furthermore, one can also compare executions of different instances of a meta-heuristic, to assess how distinct parameter calibrations affect the search process, or even compare different optimization algorithms to determine which is the most appropriate one to be used for a given optimization problem. This technique is flexible enough to support both trajectory and population-based meta-heuristics, as well as a large range of numeric functions.

The study performed in Section IV aims to present an application of this technique and to have an insight on the possible analyses one can conduct with it. Some more interesting and conclusive information regarding meta-heuristic behavior can be assessed by this visualization method if a deeper and more detailed study is performed. The steps to generate the images analyzed here can be extended to any optimization function or meta-heuristic with little or no adjustment, making it a flexible technique to visually gain information about the points into the n-dimensional space. It was demonstrated in this paper that it can be very useful to investigate the execution of meta-heuristics, as well as its optimization process in general, in order to make it more suitable to the problem being optimized.

Although useful as it is, the method described here can be improved in a variety of ways. For instance, the value of the fitness function of each point may be added to the charts, as well as a visual indication of the number of points that are currently activating a single neuron, similarly to what was presented by [11] and [15]. Additionally, it can be extended to work with other classes of optimization problems that are different from the multimodal numeric function used here as example, or be used as an on-line visualization tool.

REFERENCES

[1] M. L. Araujo, "Analise Comparativa do Comportamento de Metaheuristicas Populactionais para Otimizacao Continua: Uma Abordagem Baseada em Mapas Auto-Organizaveis," M. S. thesis, Univ. of Fortaleza, Fortaleza, Brazil, 2012.

[2] D. Keim, "Information Visualization and Visual Data Mining," *IEEE Transactions on Visualization and Computer Graphics*, vol. 8, no. 1, pp. 1-8, 2002.

[3] D. Keim, J. Kohlhammer, G. Ellis and F. Mansmann, *Mastering the Information Age: Solving Problems with Visual Analytics.* Germany: Eurographics Association, 2010.

[4] J. van Wijk, "The Value of Visualization," *Procceedings of IEEE Conference on Visualization*, pp. 79-86, 2005.

[5] M. Oliveira, H. Levkowitz, "From visual data exploration to visual data mining: A survey," *IEEE T. Vis. Comput. Gr.*, vol. 9, pp. 378-394, 2003.

[6] H. Pohlheim, "Visualization of evolutionary algorithm - Set of standard techniques and multidimensional visualization," *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 533-540, 1999.

[7] A. Wu, K. De Jong, D. Burke, J. Grefenstette, C. Ramsey, "Visual analysis of evolutionary algorithms," *Proc. of IEEE Conference on Evolutionary Computation*, pp. 1419-1425, 1999.

[8] E. Hart, P. Ross, "GAVEL - A new tool for genetic algorithm visualization," *IEEE T. Evolut. Comput.*, vol. 5, pp. 335-348, 2001.

[9] H-C Wu, C-T Sun, S-S Lee, "Visualization of evolutionary computation processes from a population perspective," *Fifth World Congress on Intelligent Control and Automation*, vol. 3, pp. 2077-2081, 2004.

[10] M. Kadluczka, P. C. Nelson, T. M. Tirpak, "N-to-2-space mapping for visualization of search algorithm performance," *Proc. of IEEE International Conference on Tools with Artificial Intelligence*, pp.508-513, 2004.

[11] R. Dybowsky, T.D. Collins, P.R. Weller, "Visualization of Binary String Convergence by Sammon Mapping," *Proceedings of The First Annual Conference on Evolutionary Programming*, pp. 27-29, 1996.

[12] S. Halim, R.H.C. Yap, H.C. Lau, "Viz: A Visual Analysis Suite for Explaining Local Search Behavior," *Proceedings of the 19th Annual ACM Symposium on User Interface Software and Technology*, pp 57-66, 2006.

[13] S. Halim, H.C. Lau, "Tuning Tabu Search Strategies via Visual Diagnosis" in *Meta-Heuristics: Progress as Complex Systems Optimization*, K.F. Doerner, M. Gendreau, P. Greistorfer, W. Gutjahr, R.F. Hartl and M. Reimann, Eds. Berlin, Germany: Springer, 2007, pp. 365-388.

[14] Y. Kim, K. Lee, Y. Yoon, "Visualizing The Search Process of Particle Swarm Optimization," *Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation (GECCO) 09*, pp. 49-56, 2009.

[15] M. Lotif, A. Coelho, "Visually Inspecting the Search Behavior of Harmony Search and Its Variants With Viz3D," *2011 IEEE Congress on Evolutionary Computation (CEC)*, pp. 616-623, 2011.

[16] J. Perez, A. Mexicano, R. Santaolaya, I. Alvarado, M. Hidalgo, R. De la Rosa, "A visual tool for analyzing the behavior of metaheuristic algorithms," *International Journal of Combinatorial Optimization Problems and Informatics*, vol. 3, no. 2, pp. 31-43, 2012.

[17] T. Kohonen, "The Self-Organizing Map," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1464-1480, 1990.

[18] T. Kohonen, *Self-Organizing Maps*, 3 ed. New York: Springer-Verlag, 2001.

[19] J. Mwasiagi, *Self Organizing Maps: Applications and Novel Algorithm Design*, Croatia: InTech, 2011.

[20] U. Seiffert, L. C. Jain, *Self-Organizing Neural Networks: Recent Advances and Applications*, New York: Springer, 2002.

[21] A. Flexer, "On the Use of Self-organizing Maps for Clustering and Visualization" in *Principles of Data Mining and Knowledge Discovery*, vol. 1704. J. Zytkow and J. Rauch, Eds. Berlin: Springer, 1999, pp. 80-88.

[22] J. Hanke, J. Reich, "Kohonen Map as a Visualization Tool for the Analysis of Protein Sequences: Multiple Alignments, Domains and Segments of Secondary Structures," *Computer applications in the biosciences (CABIOS)*, vol. 12, no. 6, pp. 447-454, 1996.

[23] J. Ong, S. Abidi, "Data Mining Using Self-Organizing Kohonen maps: A Technique for Effective Data Clustering & Visualisation," *International Conference on Artificial Intelligence (IC-AI'99)*, pp. 1-4, 1999.

[24] H. Amor, A. Rettinger, "Intelligent exploration for genetic algorithms: using self-organizing maps in evolutionary computation," *Proceedings of the 7th annual conference on Genetic and evolutionary computation (GECCO '05)*, 1531-1538, 2005.

[25] J. Vesanto, "SOM-based data visualization methods," *Intelligent Data Analysis*, vol. 3, no. 2, pp. 111-129, 1999.

[26] D. Dzemyda, O. Kurasova, J. Zilinskas, *Multidimensional data visualization: Methods and applications.* Springer Optimization and Its Applications, vol. 75, Germany, 2013.

[27] A. Ultsch, "Maps for the visualization of high-dimensional data spaces," *Proc. Workshop on Self organizing Maps*, pp. 225-230, 2003.

[28] A. Ultsch, H. Siemon, "Kohonen's Self Organizing Feature Maps for Exploratory Data Analysis," *Proceedings of International Neural Network Conference (INNC90)*, pp. 305-308, 1990.

[29] A. Ultsch, "Data Mining and Knowledge Discovery with Emergent Self-Organizing Feature Maps for Multivariate Time Series," in *Kohonen Maps*, E. Oja and S. Kaski, Eds. Amsterdam, The Netherlands: Elsevier, 1999, pp. 33-46,

[30] H. Jin, W.-H. Shum, K.-S. Leung, M.-L. Wong, "Expanding Self-Organizing Map for data visualization and cluster analysis," *Information Sciences*, vol. 163, no. 1-3, pp. 157-173, 2002.

[31] E. Pampalk, A. Rauber, D. Merkl, "Using Smoothed Data Histograms for Cluster Visualization in Self-Organizing Maps," *Proceedings of the International Conference on Artificial Neural Networks*, pp. 871-876, 2002.

[32] P. Stefanovic, O. Kurasova, "Visual Analysis of Self-Organizing Maps," *Nonlinear Analysis: Modelling and Control*, vol. 16, no. 4, pp. 488-504, 2011.

[33] A. Ultsch, F. Morchen, "ESOM-Maps: Tools for Clustering, Visualization and Classification With Emergent SOM," Databionics Research Group, University of Marburg, Gemany, Tech. Rep. 46, Mar. 2005.

[34] J. Schatzmann, "Using Self-Organizing Maps to Visualize Clusters and Trends in Multidimensional Datasets," Final Year Individual Project Report, Imperial College London, United Kingdom, 2003.

[35] A. Skupin, R. Hagelman, "Visualizing Demographic Trajectories with Self-Organizing Maps," *Journal Geoinformatica*, vol. 9, no. 2, pp. 159-179, 2005.

[36] J. Hagenauer, M. Helbich, M. Leitner, "Visualization of crime trajectories with self-organizing maps: a case study on evaluating the impact of hurricanes on spatio-temporal crime hotspots," *25th Conference of the International Cartographic Association*, 2011.

[37] J. Digalakis, K. Magaritis, "An Experimental Study of Benchmarking Functions for Generic Algorithms," *International Journal of Computer Mathematics*, vol. 79, no. 4, pp. 403-416, 2002.

[38] A. Ultsch, "Self-Organizing Neural Networks for Visualization and Classification," in *Information and Classification*, O. Opitz, B. Lausen and R. Klar, Eds. London, England: Springer, 1993, pp. 307-313.

[39] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs,* 3 ed. New York: Springer-Verlag, 1996.

[40] O. Jadaan, L. Rajamani, C. Rao, "Improved Selection Operator for GA," *Journal of Theoretical and Applied Information Technology*, pp. 269-277, 2005.