Modified growing neural gas algorithm for faster convergence on signal distribution sudden change

Article · October 2009

DOI: 10.1109/ICAT.2009.5348398

2 authors:

Stojanco Gancev
Ss. Cyril and Methodius University in Skopje
4 PUBLICATIONS **13** CITATIONS
SEE PROFILE

Andrea Kulakov
Ss. Cyril and Methodius University in Skopje Macedonia
87 PUBLICATIONS **819** CITATIONS
SEE PROFILE

Modified Growing Neural Gas Algorithm for Faster Convergence on signal distribution sudden change

Stojanco Gancev, Andrea Kulakov
Faculty of Electrical Engineering and IT
University of Sts Cyril and Methodius
Skopje, Macedonia

gancevstojan@yahoo.com, kulak@feit.ukim.edu.mk

***Abstract*— The paper deals with the problem of faster optimal coverage of a Growing Neural Gas algorithm for random signals appearing with non-stationary distributions. A modification of the algorithm that successfully solves this problem will be presented with simulations in a 2-D environment and statistical results that will show its efficiency. A comparison with a previous solution for the same problem using so called Utility measure will be also given.**

Keywords-component; *growing neural gas; faster convergence; fuzzy algorithm;* ***non-stationary distribution;***

I. INTRODUCTION

Non-stationary data can be found in many systems and the use of the models with the incremental networks called Growing Neural Gas (GNG), developed by Fritzke [1] can cope with slowly changing non-stationary distributions of the input signals. Still, rapid changes in the distribution cannot be handled properly, leaving so many called dead units that are far away from the new distribution (see Fig. 1).

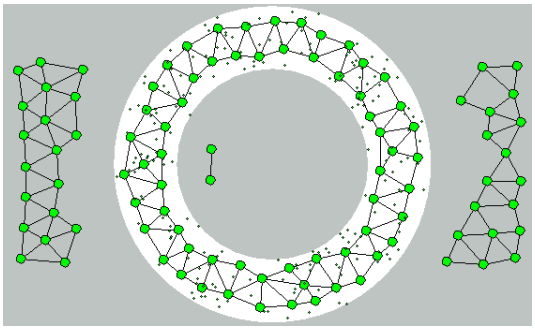

Figure 1. Nodes outside the ring are dead units because they are too far away from the new distribution of the signals (the ring) and disconnected from the nodes inside the ring.

Proposed algorithm modification from Fritzke [2] introduced as GNG-U solves this problem using so called Utility measure that is removing dead nodes and inserting them again, can be found and compared in [3], [4] and [5]. Improved algorithm GNG-U in combination with wireless sensor networks where rapid re coverage of the special environmental regions is required is applied in [6] and [7]. The need of quick response to non-stationary distribution changes in web data mining (some documents disappear or become obsolete, while other enter analysis) is presented in paper [8] where improved GNG with the utility measure is used to achieve system requirements.

In the paper, we will present algorithm on top of GNG and GNG-U that converges faster to the optimal coverage than the GNG-U when the signal distribution suddenly changes. The paper will present algorithm build on top of GNG and GNG-U that converges faster to the optimal coverage on sudden distribution changes. The modification is based on a dynamic fuzzy change of the parameter epsilon neighbor for defined step of iterations explained in section two of this paper.

The modified algorithm notation is GNG-F and GNG-UF when our modification is applied to the improved algorithm from Fritzke [2]. The 'F' letter notation in the algorithm defines fuzzy calculation of epsilon neighbor value when sudden change of signal distribution occurs.

In order to compare the algorithm effectiveness statistical results are presented in section four in the paper where all of the measurements are conducted for the same set of distribution changes for the GNG, GNG-U, GNG-F and GNG-UF algorithms.

II. THE ALGORITHM

A. Original GNG algorithm

For the sake of completeness and clarity, the paper will present major part of the original algorithm again, and later our improvement will be described. All the units have some random positions w_i in $\boldsymbol{R}^n$, although only simulation results from $\boldsymbol{R}^2$ will be given.

1. Pay attention to the input signal ξ distributed according to some distribution $P(\xi)$.
2. Find the nearest unit s_1 and the second-nearest unit s_2.
3. Increment the age of all edges emanating from s_1
4. Add the squared distance between the input signal and the nearest unit in input space to a local error variable:

$$\Delta error(s_1) = \left\| w_{s_1} - \xi \right\|^2 \quad (1)$$

5. Move s_1 and its direct topological neighbors toward the signal ξ by fractions ε_b and ε_n, respectively, of the total distance:

$$\Delta w_{s_1} = \varepsilon_b (\xi - w_{s_1}) \quad (2)$$

$$\Delta w_n = \varepsilon_n (\xi - w_n) \quad (3)$$

for all direct neighbors n of s_1.

6. If s_1 and s_2 are connected by an edge, set the age of this edge to zero. If such an edge does not exist, create it.
7. Remove edges with an age larger than a_{max}. If this results in sensor nodes having no emanating edges, set these nodes as "free".
8. Decrease all local error variables by multiplying them with a constant d.
9. If a stopping criterion (e.g., network size or some performance measure like minimal overall error) is not yet fulfilled, go to step one.

B. *Modified GNG-F algorithm*

Our improvement of GNG algorithm is by means of manipulating the parameter ε_n that determines the ratio by which the neighboring units will be affected to move towards the winning node that was closest to the input signal.
Modified algorithm used three additional values:

1. ***eI*** - Effective iterations. Representing the number of iterations in which parameter ε_n is set to 1.0

2. ***cN*** – count of iterations needed to calculate approximate error.

$$aproxError = \sum_{i=n-cN}^{i=n} error(i) \Big/ cN \quad (4)$$

3. ***tH*** – Threshold factor. Multiplying factor used to trigger the state of effective iterations.

$$tFactor(n) = aproxError(n) + tH * \sum_{i=n-cN}^{i=n} error(i) \Big/ cN \quad (5)$$

Calculation of ε_n value in the GNG-UF is with the following fuzzy rule:

If error(n+1)> tFactor(n) then

$$From \ \varepsilon_{n+1} \text{ to } \varepsilon_{n+eI+1} = 1.0 \quad (6)$$

where tFactor(n) is calculated in (5)

Namely when the signal distribution suddenly changes, then the overall error, which is a sum of all local error variables, will enormously increase. In the GNG-F or GNG-UF algorithm, this triggers an increase of the parameter ε_n (0.1) for the following effective iterations *eF* that results in more vivid mobility of the nodes toward the new distribution of the signals. After effective iterations, the parameter is switch back to its low value (0.0006) which causes less mobility of the nodes and allows better spreading of the nodes within the region of the current distribution of the signals.

When GNG-U used with our modification (GNG-UF) the effectiveness is noticeable with big utility factor value (bigger or equal to 600.0), which is solving the problem if some of the nodes are left out of the signal area after the effective iterations.

III. THE SIMULATION

Experiments have been conducted with the simulation test provided courtesy of Hartmut S. Loos and Bernd Fritzke at [9] on which we implemented our algorithm and add configuration capabilities to GNG-UF parameters. For better understanding, the user interface of the modified application is presented in Fig. 2.

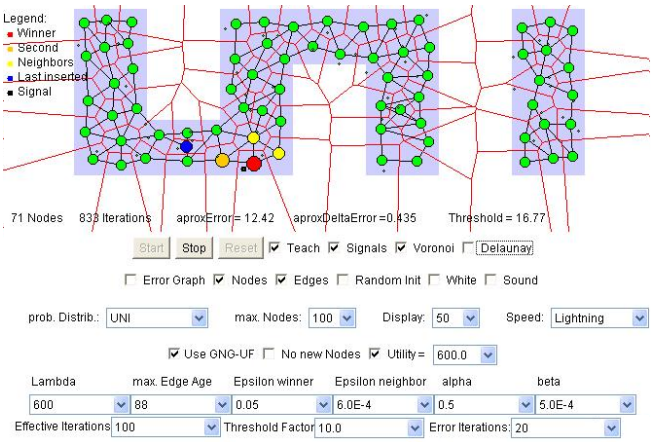

Figure 2. Screen of the user interface of the modified GNG application. In the bottom of the panel are the settings for effective iterations, threshold factor and error iteration that are part of the modified algorithm.

The improvement of the algorithm have been compared with the proposed solution GNG and GNG-U by Fritzke in [2] where he introduces a measure of Utility of each node, with possibility of eliminating nodes which are not used for a long period. His solution gives bigger initial errors since it eliminates many, if not all, nodes and gradually adds new nodes according the new distribution. The proposed improvement rather redistributes the nodes with bigger agility according to the new signal distribution.

The 2-D signal distributions that we will work within the paper are presented in the Fig. 3 below.

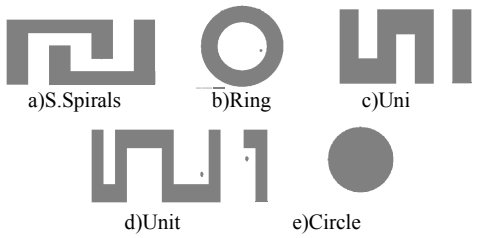

Figure 3. Different signal distributions (signal area is in gray color).

In Fig. 3, the nodes of the sensor network are divided into two tightly connected sub-networks of nodes. Small dots in the subsequent figures show the appearance of the last 50 input signals.

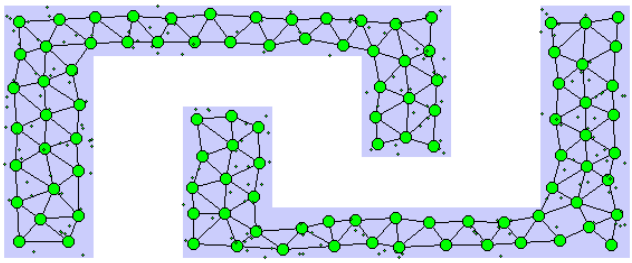

Figure 4. Small Spiral signal distribution covered with 100 nodes.

The distributions areas shown in figures present the probability distributions in the 2-D plane, where the concentration of the input signals is uniform. In Fig. 5 a sudden change of the probability distribution can be seen from two rectangular areas (Small Spiral distribution), where the nodes were previously distributed, into one circle.

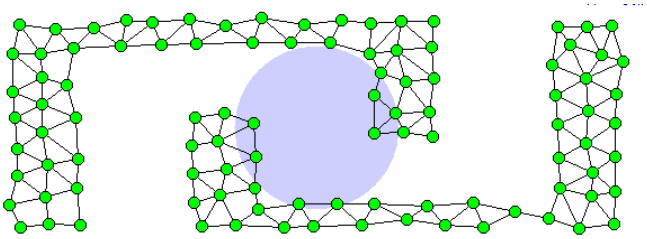

Figure 5. Sudden change of the distribution of the signals from Small Spiral to Circle results in poorly covered signals.

In the Fig. 6, 7 and 8 below can be seen the reaction of each algorithm GNG, GNG-U and GNG-UF respectively to the signal distribution change. In Fig. 6 where the GNG algorithm is presented we can see the high number of dead nodes that is resulting with big error. In Fig. 7 is the GNG-U where the utility factor eliminates all the dead nodes despite the GNG-UF where nodes are swiftly moved to the signal area.

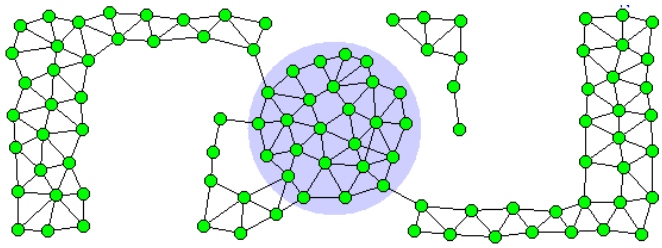

Figure 6. GNG Most of the nodes are out of the signal area and are unusable which results with high error

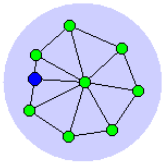

Figure 7. GNG-U (utility factor=3.0). Most of the nodes are pronounced death and will be created again.

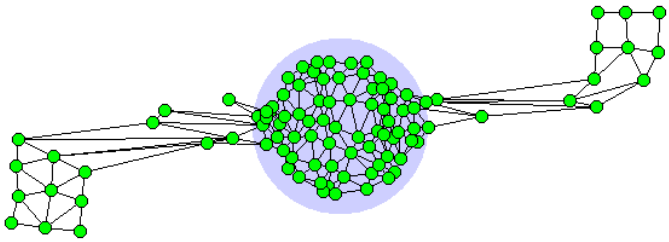

Figure 8. GNG-UF With the increase of the parameter ε_n =0.1, the nodes start swiftly to move towards the area of biggest concentration of input signals. This shows much faster convergence than the GNG-U algorithm.

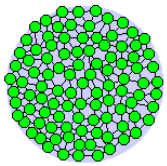

Figure 9. Later on, low ε_n parameter (0.0006), leads to better dispersion of nodes and thus to lower overall error

When the overall error from signals obtained with the proposed algorithm is compared with overall error from the GNG-U algorithm of Fritzke[2] we show that the time needed to converge to optimal distribution of sensor nodes depending on the probability distribution of the input signals, are many times shorter in case of the proposed algorithm.

IV. THE RESULTS

Results used to compare the algorithm effectives will be presented with graphical data taken from the simulation we did and statistical data measurements of various signal distribution changes.

A. Error graphs

Error graphs from the simulation can be seen in Fig. 10 where GNG–U error is presented and Fig. 11 with GNG-UF when signal distribution changes from Small Spiral to Circle.

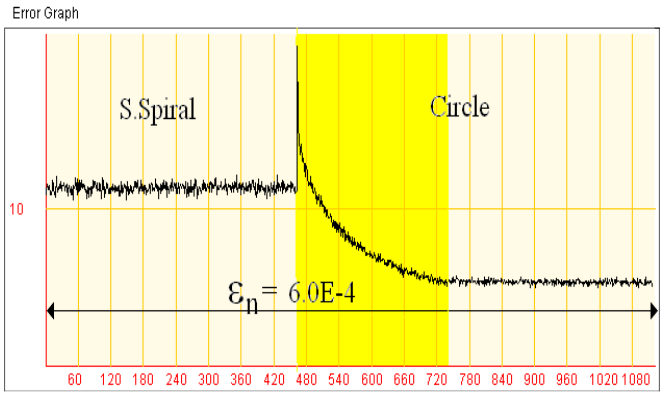

Figure 10. Error graph for GNG-U when distribution is goes from small spiral to circle. With yellow color transition, time is marked. (Utility =3.0; signals= 200 and ε_n = 6.0E-4).

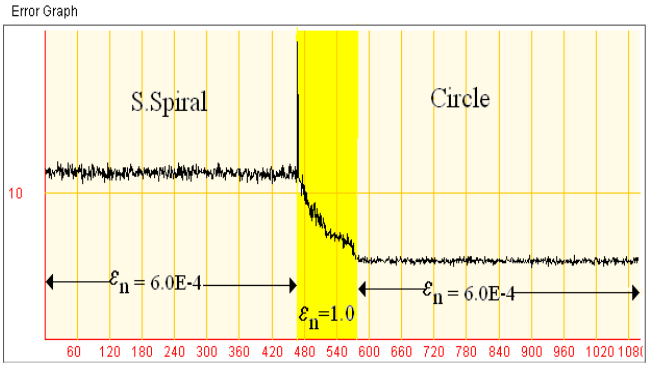

Figure 11. Error graph for GNG-UF when distribution is goes from small spiral to circle. With yellow color transition, time is marked.

(Utility =600.0; signals= 200; default ε_n = 6.0E-4, *eI*=100; *tH*=10.0; *cN*=20).

Error graphs comparison for the same distribution change shows that the GNG-U needs 270 iterations to distribute nodes despite 110 iterations of GNG-UF. This shows much faster response on distribution change and smaller error when used GNG- UF algorithm.

B. Statistical data comparison

We conducted multiple measurements from different signal distributions changes and the results are presented in Table I and Table II where values are calculated as average of 15 measurements per result. Approximate error variation is +/- 0.1 and number of iterations variations are +/-10. In the tables, we made comparison between different setups where U letter is for the utility factor and F is for fuzzy calculation of Epsilon neighbor.

TABLE I. RESULTS TAKEN WHEN TESTING WITH 50 SIGNALS. IF THE MEASURED VALUE IS WITH THE UTILITY FACTOR THE VALUE IS SET IN PRETENSES (UTILITY) NEXT TO THE ALGORITHM.

From /To		GNG-U (3.0)	GNG-UF (3.0)	GNG	GNG - F	GNG-UF (600)
S.Spiral /Circle	error	5.4	5.4	9.6	6.6	5.4
	steps	1150	750	*750	*600	700
Ring /Circle	error	5.4	5.4	10.0	5.4	5.4
	steps	1171	1280	*1000	155	155
Uni /Ring	error	8,6	8,6	11.4	10.4	8,6
	steps	1050	1000	*800	*1000	550
Uni /Unit	error	8.6	8.6	9.7	8.8	8.7
	steps	800	800	*1900	400	350
Unit /Uni	error	10.7	10.7	10.9	10.9	10.7
	steps	1150	1150	*1400	*400	280

TABLE II. RESULTS TAKEN WHEN TESTING WITH 200 SIGNALS. IF THE MEASURED VALUE IS WITH THE UTILITY FACTOR THE VALUE IS SET IN PRETENSES (UTILITY) NEXT TO THE ALGORITHM.

From /To		GNG-UF (600)	GNG-U (3.0)	GNG-U/ GNG-UF	% Steps
S.Spiral /Circle	error	5.4	5.4	2.45	145%
	steps	110	270		
Ring / Circle	error	5.4	5.4	2.63	163%
	steps	110	290		
Uni/ Ring	error	8,6	8,6	2.26	126%
	steps	115	260		
Uni/ Unit	error	8.8	8.7	1.14	14%
	steps	220	250		
Unit/ Uni	error	10.8	10.7	2.20	120%
	steps	150	330		

*There are nodes out of the signal area due to a fact that utility parameter is off.

Measured results prove that the GNG-UF is big improvement to GNG-U when the distribution signals suddenly change. In the Table 2, we give comparison between best performing results of GNG-U and GNG-UF when the distribution coverage is with 200 signals; Results show that GNG-UF improves steps to recover the distribution area of GNG-U algorithm up to 163 percents when change is from Ring to Circle.

We see that our algorithm shows the best result when used with utility factor bigger than 600 because in some scenarios of GNG-F dead nodes can occur (presented in table 1).

V. CONCLUSION

The improvement of GNG-U with the proposed modification presented in GNG-UF shows much bigger flexibility and faster convergence on signal distribution sudden change. The modified algorithm presented in this paper is integrated with in the current GNG demo [9] where the parameters of GNG-UF can be configured and tested for further comparison and experiments.

As we can see in the referenced papers the use and the need of faster coverage is requirement in different systems like wireless sensor networks, data mining or video shot motion categorization. In our future works, we plan to integrate the modified algorithm in real-time application and compare the effectiveness of the modified GNG.

ACKNOWLEDGMENT

The work described in this article has been partially funded by the Ministry of Education and Science of Macedonia.

REFERENCES

[1] Fritzke, B., A Growing Neural Gas Network Learns Topologies, in Tesauro, G., Touretzky, D.S., and Leen, T.K. (eds.) Advances in Neural Information Processing Systems 7, MIT Press, Cambridge MA, USA, 1995.

[2] Fritzke, B., A self-organizing network that can follow non-stationary distributions. In ICANN'97: International Conference on Artificial Neural Networks, pages 613-618. Springer.

[3] Costa, J.A.F.; Oliveira, R.S. Cluster Analysis using Growing Neural Gas and Graph Partitioning Neural Networks, International Joint Conference, pages 3051-3056, August 2007

[4] J Holmström, "Growing neural gas experiments with gng, gng with utility and supervised gng," Master's thesis, Uppsala University. Aug 2002

[5] X Cao and P N Suganthan, Video Shot Motion Characterization based on Hierarchical Overlapped Growing Neural Gas Networks. In Multimedia Systems, pages 378--385, October 2003.

[6] Y Sun, L Li, Dynamic Coverage Based on Neural Gas Learning Algorithm for Wireless Sensor Network Source, International Journal of Distributed Sensor Networks archive - volume 5 , pages 87-87 , Issue 1, January 2009

[7] Y Sun, J Qian, L Li, Hybrid Learning Algorithm for Effective Coverage in Wireless Sensor Networks, Proceedings of the 2008 Fourth International Conference on Natural Computation - volume 05, pages 227-231, 2008

[8] K Ciesielski, M l Drami´nski, M law,D Czerski, S T Wierzcho´n, Incremental Document Map Formation: Multi-stage Approach,XXI Autumn Meeting of Polish Information Processing Society Conference Proceedings pp. 27–37, 2005

[9] ftp://ftp.neuroinformatik.ruhr-uni-bochum.de/pub/software/NN/DemoGNG/DemoGNG-1.5.zip