In the Name of God

Journal of
Information Systems & Telecommunication
Vol. 5, No. 1, January-March 2017, Serial Number 17

Research Institute for Information and Communication Technology
Iranian Association of Information and Communication Technology

Affiliated to: Academic Center for Education, Culture and Research (ACECR)

Print ISSN: 2322-1437
Online ISSN: 2345-2773
Publication License: 91/13216

Editorial Office Address: No.5, Saeedi Alley, Kalej Intersection., Enghelab Ave., Tehran, Iran,
P.O.Box: 13145-799 Tel: (+9821) 88930150 Fax: (+9821) 88930157
E-mail: info@jist.ir
URL: www.jist.ir

Indexed by:
- Index Copernicus International www.indexcopernicus.com
- Islamic World Science Citation Center (ISC) www.isc.gov.ir
- Directory of open Access Journals www.Doaj.org
- Scientific Information Database (SID) www.sid.ir
- Regional Information Center for Science and Technology (RICeST) www.ricest.ac.ir
- Iranian Magazines Databases www.magiran.com

Publisher:
Regional Information Center for Science and Technology **(RICeST)**
Islamic World Science Citation Center **(ISC)**

This Journal is published under scientific support of
Advanced Information Systems (AIS) Research Group and
Digital & Signal Processing Research Group, ICTRC

Acknowledgement

JIST Editorial-Board would like to gratefully appreciate the following distinguished referees for spending their valuable time and expertise in reviewing the manuscripts and their constructive suggestions, which had a great impact on the enhancement of this issue of the JIST Journal.

(A-Z)

- Abbasi, Mahdi, Bu-Ali Sina University, Hamadan , Iran
- Aghabozorgi, Masoud Reza, Yazd University, Yazd, Iran
- Ardeshir, Gholamreza , Babol Noshirvani University of Technology, Babol, Iran
- Azimzadeh, Fatemeh, ACECR, Tehran, Iran
- Borna, Keyvan, Kharazmi University, Tehran, Iran
- Buyer, Asgar Ali, Azarbayjan Shahid Madani University, Tabriz, Iran
- Babak Nasersharif, khaje Nasir-edin Toosi University, Tehran, Iran
- Ebrahimzadeh Ataollah, Babol Noshirvani University of Technology, Babol, Iran
- Fathi, Abdolhossein, Razi University, Kermanshah, Iran
- Ghaffari Ali, Islamic Azad University, Tabriz Branch, Tabriz, Iran
- Haji Mohammadi, Zeinab, Amir kabir University, Tehran, Iran
- Hamidi, Hojjatollah, khaje Nasir-edin Toosi University, Tehran, Iran
- Hamidizadeh, Mohammadreza, Shahid Beheshti University, Tehran, Iran
- Khosravi Farsani, Hadi, Shahrekord University, Shahrekord, Iran
- Mansoorizadeh, Muharram, Bu-Ali Sina University, Hamedan, Iran
- Mosallanejad, Ali, Iran University of Science and Technology, Tehran, Iran
- Mardukhi, Farhad, Razi University, Kermanshah, Iran
- Nilforoushan, Zahra, Kharazmi University, Tehran, Iran
- Najimi, Maryam, Babol University of Technology, Babol, Iran
- Nabavi, Abdolreza, Tarbiat Modares University, Tehran, Iran
- Rafighi, Masoud, Qom University, Qom, Iran
- Rezavani, Mehdi, Arak University, Iran
- Ramezannia, Seyed Bagher, khaje Nasir-edin Toosi University, Tehran, Iran
- Ramezani, Amin, Tarbiat Modares University, Tehran, Iran
- Tavassoli, Sudeh, Technique University of Kaiserslautern, Kaiserslautern, Germany
- Tabatabaei, Hadi, Shahid Beheshti University, Tehran, Iran
- Taghizadeh Dehkordi, Maryam, Shahrekord University, Shahrekord, Iran

Table of Contents

A New Calibration Method for SAR Analog-to-Digital Converters Based on All Digital Dithering

Shabnam Rahbar
Department of Electrical, Faculty of Engineering, Shahid Chamran University of Ahvaz, Ahvaz, Iran
Sh_rahbar@hotmail.com
Ebrahim Farshidi*
Department of Electrical, Faculty of Engineering, Shahid Chamran University of Ahvaz, Ahvaz, Iran
farshidi@scu.com

Received: 11/Feb/2015 Revised: 07/Jan/2017 Accepted: 07/Mar/2017

Abstract

In this paper a new digital background calibration method for successive approximation register analog to digital converters is presented. For developing, a perturbation signal is added and also digital offset is injected. One of the main advantages of this work is that it is completely digitally and eliminates the nonlinear errors between analog capacitor and array capacitors due to converter's capacitors mismatch error by correcting the relative weights. Performing of this digital dithering method does not require extra capacitors or double independent converters and it will eliminate mismatches caused by these added elements. Also, No extra calibration overhead for complicated mathematical calculation is needed. It unlike split calibration, does not need two independent converters for production of two specified paths and it just have one capacitor array which makes it possible with simple architecture. Furthermore, to improve DNL and INL and correct the missing code error, sub radix-2 is used in the converter structure. Proposed calibration method is implemented by a 10 bit, 1.87-radix SAR converter. Simulation results with MATLAB software show great improvement in static and dynamic characteristics in applied analog to digital converter after calibration. So, it can be used in calibration of successive approximation register analog to digital converters.

Keywords: SAR; Converter; Calibration; Perturbation; Radix-2; DNL and INL.

1. Introduction

Today, due to the combination of analog and digital integrated circuits, having the appropriate speed and accuracy in data conversion is very important. Hence, successive-approximation-register (SAR) analog-to-digital converters (ADCs) because of establishing a good balance between speed and accuracy as well as lower complexity of circuits than sigma delta converters, flash and other types that either lower speed or accuracy and high circuits complexity and volume, have been considered [1-18]. SAR ADC due to good tradeoffs between speed and accuracy. However, capacitor mismatch in the SAR converters limits the accuracy and resolution of the converter. Commonly, the main factor that limits the linearity in these converters is capacitor mismatch error in digital-to-analog converter (DAC) [10]. Some SAR ADC calibration methods perform the calibration in the analog domain, which requires extra analog circuits [14, 17, 18]. This extra analog will cause additional mismatches between them and capacitors of core array. Also, they need higher circuit complexity. In [15, 19, 20] some SAR ADC digital calibration technique with the dithering method have been proposed. The disadvantage of this technique is that, it is analyzed based on complicated mathematical calculations using matrix and inversing of them, therefore, its implementation require very complex circuit in digital calibration part.

In this work, a new SAR ADC digital calibration technique with offset injecting perturbation signal in digital domain is presented. In the proposed method, and in two steps, this offset will be injected in the two least significant capacitors in the basic capacitor array, separately. Also, sub radix-2 architecture [14, 17] is utilized in order to not to lose any code and prevent redundancy.

The advantages of this method is that: firstly, for applying the perturbation signal extra analog capacitor [15], which leads to new uncompensated mismatching between this analog capacitor and array capacitors, is not required. Secondly, comparing to preceding digital dithering method [16, 19, 20], it doesn't need complicated mathematical calculation. Thirdly, unlike split calibration, that requires two independent converters for production of two specified paths [17, 18], it just employs basic capacitor array which makes it possible with simple architecture.

This paper organizes the following chapters: in the second chapter a review on procedure of SAR ADS converters has been provided. In third chapter, calibration architecture and error correction procedure is introduced. In the fourth and fifth chapters simulation and conclusion are provided.

* Corresponding Author

2. Basic Principle of SAR

2.1 Binary SAR Architecture

In Fig. 1, a conventional form of SAR analog-to-digital converters has been showed. Commonly, it includes a DAC with binary weight therein each capacitor with more valuable weight is equal to total low-valuable capacitors. Analog input will be converted in N clock cycles to N-bit digital output. In first phase all capacitors are connected to input sample v_{in}, then capacitor c_{N-1} is connected to capacitor $+v_{ref}$ and the remaining capacitors connect to $-v_{ref}$. By redistribution of capacitor charges, total node voltage is obtained as below [10, 14, 17]:

$$v_{sum} = -v_{in} + {}^{c_{N-1}}/_{C_T} . v_{ref} - \frac{\sum_{i=0}^{N-2} c_i + c_0}{c_T} . v_{ref} \quad (1)$$

$$c_T = \sum_{i=0}^{N-1} c_i + c_0 \quad (2)$$

Considering v_{sum} sign, the comparer specifies d_{N-1} bit. If $V_{sum} > 0, d_{N-1}$ is zero and otherwise will be 1. This process is repeated in N clock pulse and for N bit:

$$v_{sum} = -v_{in} + \sum_{i=0}^{N-1} (2d_{i-1} - 1) \frac{c_i}{c_T} v_{ref} - \frac{c_0}{c_T} v_{ref} \quad (3)$$

In (3), it is observed that the share of each digital output bit is determined by a weight w_i that is equal to ${C_i}/{C_T}$. In ideal mode, the weights are equal to ${1}/{2^i}$ (i is the number of relative bit). In this mode, converter transfer curve is as linear function x=y [13].

Although binary algorithm is more efficient as respect to the conversion stages but is exposed to the analog disorders while implementing the actual circuit. Mismatch of capacitor in binary SAR ADC is static error resources (DNL, INL). For reducing these errors, often ADC including unit elements and capacitive divisions is used that is cause of circuits complexity in logic part such as binary to thermometry decoder circuit that ultimately results in speed drops of converter. In the mode of having error due to the mismatch of capacitors, the decision-making surfaces may not be distributed equally all over the input range. This distortion may lead in losing some codes, but that missing code maybe improved by digital calibration [13].

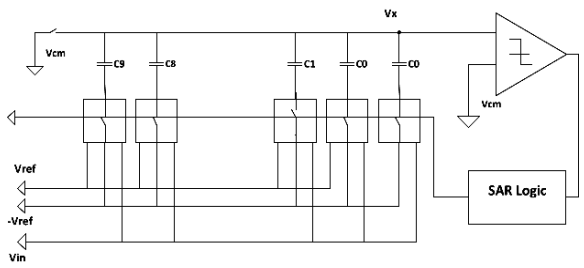


Fig. 1. SAR ADC structure

2.2 Sub Radix-2 Architecture DAC

If binary architecture without any modification is used, digital calibration cannot correct all types of mismatch errors. Fig. 2 illustrates several transfer curves for a 5-bit binary-scaled DAC [14]: Fig. 2a shows nominal curve, Fig. 2b shows curve in case of a positive DNL error in the MSB, and Fig. 2c shows curve in case of a negative DNL error in the MSB[14]. While considering the MSB only in this example, a comparable situation can occur with the other capacitors of the DAC. The large DNL error produced for DNL>0 cannot be reduced with calibration, as calibration can only re-map the input code to an existing combination of capacitors that approximates the desired output level. However, for DNL>0, there is no combination of capacitors available to fill the gap in the output range. On the other hand, the large DNL error for DNL<0 can be corrected with calibration, as there is a 'gap-free' consecutive of output levels. By digital re-mapping, the overlap of the curve can be removed to obtain a slick transfer curve. However, as a side effect of the overlap, the full-scale range of this converter will be slightly smaller than usual that is compensated by redundancy bit [13, 14].

Concisely, for the digital calibration to operate properly gaps (DNL>0) are not allowed but overlap (DNL<0) is allowed. By means of redundancy, the probability of a 'gap' can be reduced to an arbitrary low value by design: instead of designing the nominal transfer curve as in Fig. 2a (x=y), it is designed as in Fig. 2c. Thus, redundancy introduces intentional overlap (DNL<0) of the nominal transfer curve to guarantee that the consecutive of the output range remains, also in case of mismatch. While the figure illustrates redundancy for the MSB only, in reality this redundancy requirement needs to be implemented for each bit of the converter [13, 14, 17].

For the case that the more valuable weight is smaller than total capacitors, probably a code is lost.

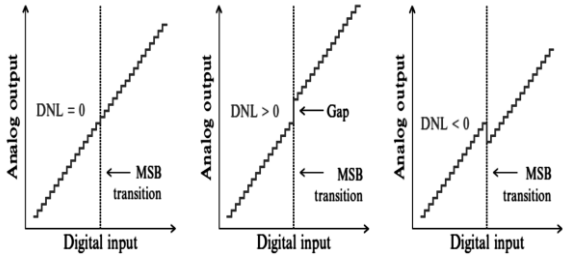


Fig. 2. Transfer curves for a 5-bit binary-scaled DAC:
left) Nominal curve middle) Curve case of a positive DNL error in the MSB right) Curve case of a negative DNL error in the MSB [14]

3. Calibration Method

Perturbation-based calibration algorithm has been introduced in [15]. In [15], perturbation signal in analog domain has been inserted to the inlet by two small capacitors. These capacitors increase the chip area. Furthermore, mismatch error between these inserted capacitors and the capacitors of SAR ADC cannot be compensated. Dithering method is used in [16], by applying a PN signal on the weights, the error is corrected by matrix and very complicated calculations which will lead to high amount of circuit a low speed in the

calibration unit. In this paper, the error has been compensated by adding two lower weights to the input.

In this paper a new structure of sub radix-2 algorithm has been used that reduces the complexity of circuit structure, in which by using the added weights w_0 and w_1 to the converter. Capacitor mismatch errors are corrected in digital domain.

Fig. 3 exhibits the first step of background calibration design of SAR ADC. As can be seen, to perturbation signal injection, the weight of lowest bit w_0 of the first stage is applied by V_{in}, in which the sum/subtract of these two makes the ADC input.

The operation is described as follows: a single SAR ADC digitizes each analog sample twice, with two offsets, where is the weight of lowest bit w_0. SAR ADC provides two outputs for each sample of analog input. D_+ is produced in lieu for input $V_{in} + w_0$ and D_- in lieu for $V_{in} - w_0$.

According to Fig. 3, the main output of ADC is produced out of the average of both outputs. The error is calculated by subtracting the outputs through reducing 2 w_0. So it can be written as:

$$\text{error} = \underbrace{\mathrm{D_+.W}}_{\mathrm{X_+}} - \underbrace{\mathrm{D_-.W}}_{\mathrm{X_-}} - 2\mathrm{w_0} \quad (4)$$

$$\mathrm{X_{out}} = \frac{\mathrm{X_+ + X_-}}{2} \quad (5)$$

where X_{out} is the final output of ADC.

Update equation of weights is obtained according to LMS method as following [15]:

$$\mathrm{w_i(n+1) = w_i(n) - \mu_w.error.(D_{+i}(n) - D_{-i}(n))} \quad (6)$$

Where μ_w is the convergence coefficient.

By this equation, all weights except w_0 will be updated and improved.

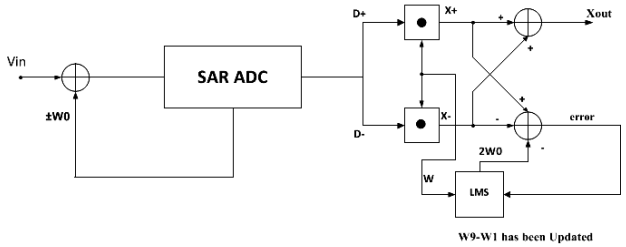


Fig. 3. Calibration diagram with injecting w0

In order to facilitate the signal injection of dither, as can be seen in the Fig. 4, the digital offset is injected through the lowest-valuable weight capacitor in capacitor banks of Fig. 1.

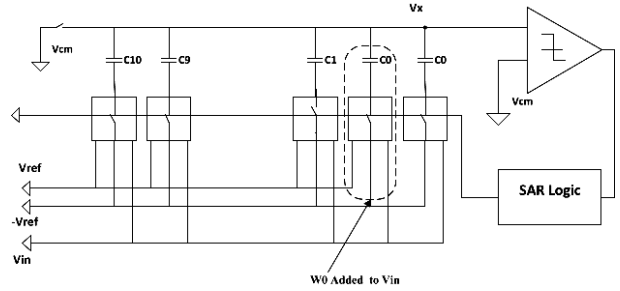


Fig. 4. Injecting w_0 to input

Adding and subtracting has been done by forward and reverse switching sequence of this capacitor, respectively [16]. So it can be said that:

$$\mathrm{V_{in} \pm w_0 = \sum_{i=1}^{N-1}(2b_i - 1)\,w_i + Q_e} \quad (7)$$

where V_{in} is the analog sample, Q_e is the quantization noise and w_i weight of the i'th bit.

In the second step calibration engine, and for updating w_0, the weight of $\pm w_1$ are added to V_{in} input (similar to w_0) and the error is obtained by calculating the difference of outputs and reducing $2w_1$ 2 as shown in Fig. 5 error is used to calibrate the weight of w_0 in terms of the following equation:

$$\text{error} = \underbrace{\mathrm{D_+.W}}_{\mathrm{X_+}} - \underbrace{\mathrm{D_-.W}}_{\mathrm{X_-}} - 2\mathrm{w_1} \quad (8)$$

and

$$\mathrm{w_0\,(n+1) = w_0(n) - \mu_w.error.(D_+(n) - D_-(n))} \quad (9)$$

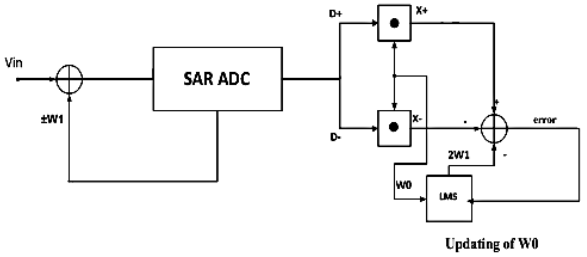


Fig. 5. Updating w_0 with injecting w_1

For the signal injection of second dither and for adding $\pm w_1$ to V_{in}, as can be seen in the Fig. 6 digital offset is injected through the second lowest-valuable weight capacitor in capacitor banks of Fig. 1(similar to preceding step). So it can be written as:

$$\mathrm{V_{in} \pm w_1 = w_0(2b_0 - 1) + \sum_{i-2}^{N-1}(2b_i - 1)w_i + Q_e} \quad (10)$$

Complete algorithm for the proposed calibration method is shown in Fig. 7. So it concluded that the proposed technique has several advantages: comparing with some preceding dithering method [15], for perturbation signal, this method does not use extra capacitor and employs same basic capacitor array. Furthermore, unlike [16,19,20], complicated mathematical and matrix calculation is not required. Also, unlike split calibration [17,18], double independent converters paths are not used.

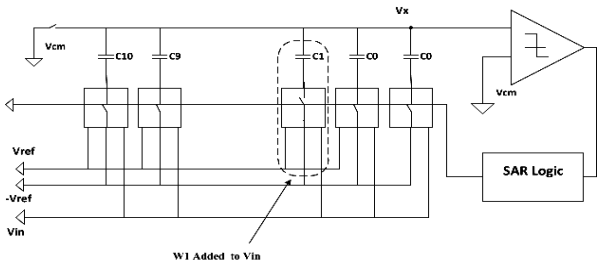


Fig. 6. Injecting w_1 to input

4. Simulation Results

A 10-bit SAR ADC has been simulated and calibrated by background digital calibration procedure and aiding MATLAB software. The capacity of capacitors in ADC has been specified according to sub radix-2 algorithm and mismatch 5% and with the base of 1.87, in accordance with first line of table I. The stability coefficients of LMS loop $\mu_w = 2^{-17}, \mu_e = 2^{-11}$ is chosen. INL and DNL diagrams before and after calibration modes are observed respectively in Fig. 8 and 9.

Fig. 8a and Fig. 8b show DNL diagrams and Fig. 9a and Fig. 9b show INL diagrams before and after calibration, respectively. In this figures the improvement of DNL and INL is completely evident. After calibration DNL and INL reduced from [-1, +12] LSB to [-0.1, +0.1] LSB and [-14.2, +14.2] LSB to [-0.34, +0.23] LSB, respectively. It should be pointed out that the effective number of bit is approximately is 10bit (which has maximum length of 2^{10}=1024 multiplied by LSB) and before calibration maximum length of DNL is about 14LSB. So DNL is less than 1.5% of maximum length in a 10bit converter. Upon reaching DNL to [-0.5, 0.5] LSB, missing code error is improved. Fig. 10 shows frequency response diagram before and after calibration. Static and dynamic specifications are shown in table II. After calibration, static and dynamic specifications have noticeable Improvement.

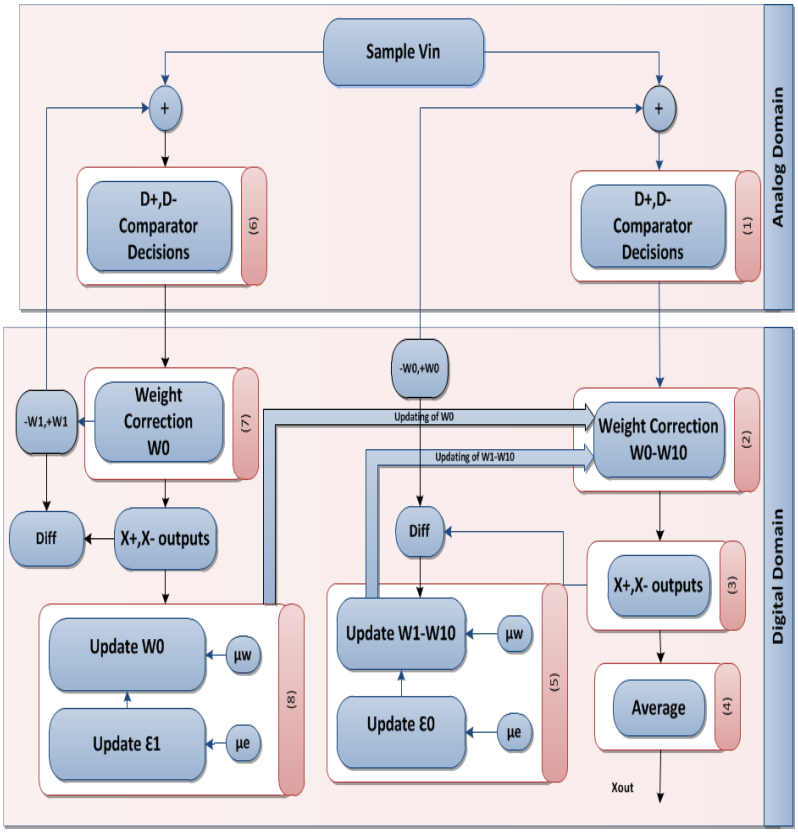
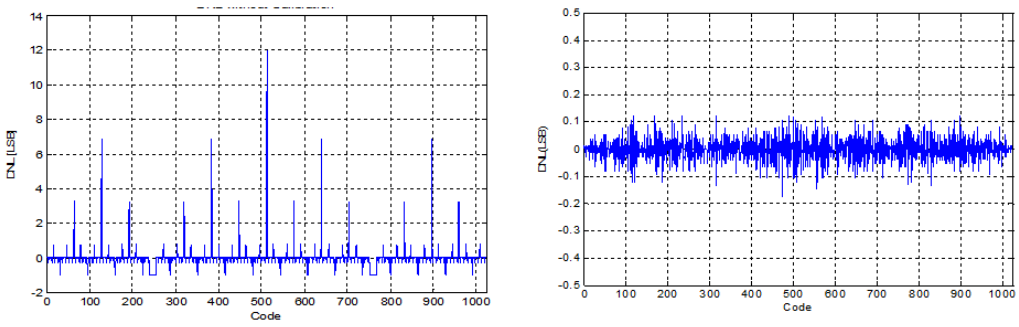
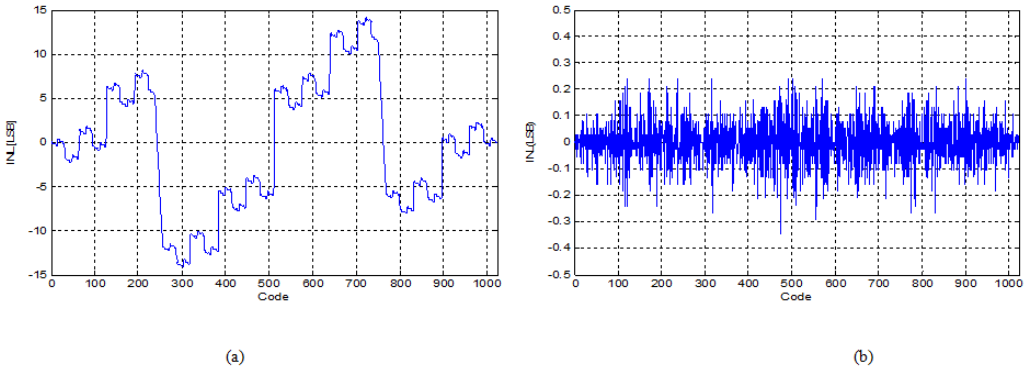


Fig. 7. Flowchart of calibration procedure



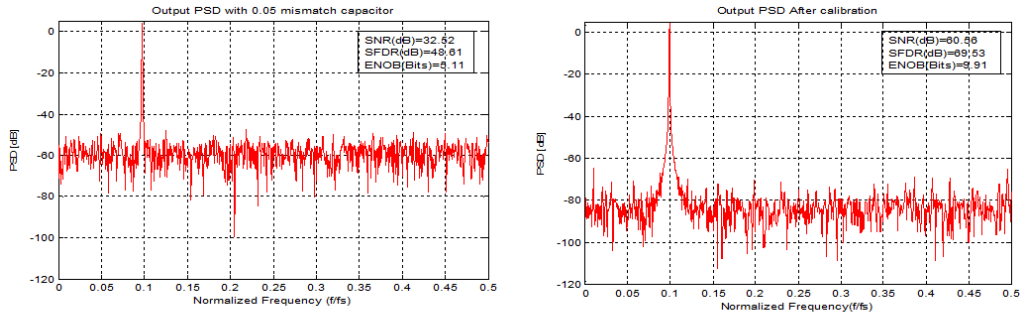
(a) (b)

Fig. 8. DNL Diagrams: a) Before calibration b) After calibration



(a) (b)

Fig. 9. INL Diagrams: a) Before calibration b) After calibration



(a) (b)

Fig. 10. Output PSD diagram: a) Before calibration b) After calibrat

Table 1. Static and dynamic specifications SAR ADC

	Before Calibration	After Calibration	Improvement Rate
DNL(LSB)	12	0.21	11.76
INL(LSB)	14.2	0.57	13.63
SNR (dB)	32.52	60.56	28.04
SFDR (dB)	48.61	69.53	20.92
ENOB (Bits)	5.11	9.91	4.8

5. Conclusion

A SAR ADC by digital background calibration method as well as sub radix-2 algorithm was presented. The dominant error in this converter was studied and improved by means of calibration method. Upon applying sub radix-2, no adaptation to the unit capacitor is required and it causes the simplicity of analog circuit and improvement of converter efficiency. Innovative result of this work is that perturbation signal is injected by the two least significant of existing elements in array capacitor of converter. So, this method does not need extra capacitor, so, related issue of mismatch between added element and existing elements in array capacitor will be eliminated. Furthermore, compared to the preceding works, it does not require complex mathematical calculations.

References

[1] F. Kuttner, “A 1.2-V 10-b 20-Msample/s nonbinary successive approximation ADC in 0.13- m CMOS,” in IEEE ISSCC Dig. Tech. Papers, Feb. 2002.

[2] J. Craninckx and G. Van der Plas, “A 65 fJ/conversion-step 0-to-50 MS/s 0-to-0.7 mW 9 b charge-sharing SAR ADC in 90 nm digital CMOS,” in IEEE ISSCC Dig. Tech. Papers, Feb. 2007.

[3] V. Giannini et al., “An 820 W 9 b 40 MS/s noise-tolerant dynamic-SAR ADC in 90 nm digital CMOS,” in IEEE ISSCC Dig. Tech.Papers, Feb. 2008.

[4] C. C. Liu et al., “A 10 b 100 MS/s 1.13 mW SAR ADC with binary scaled error compensation,” in IEEE ISSCC Dig. Tech. Papers, Feb. 2010.

[5] Y. Z. Lin et al., “A 9-bit 150-MS/s 1.53-mW subranged SAR ADC in 90-nm CMOS,” in Symp. VLSI Circuits Dig. Tech. Papers, Jun. 2010.

[6] C. C. Liu et al., “A 1 V 11 fJ/conversion-step 10 bit 10 MS/s asynchronous SAR ADC in 0.18 m CMOS,” in Symp. VLSI Circuits Dig. Tech. Papers, Jun. 2010.

[7] G. Promitzer, “12 bit low-power fully differential switched capacitor noncalibrating successive approximation ADC with 1 MS/s,” IEEE J. Solid-State Circuits, vol. 36, no. 7, pp. 1138–1143, Jul. 2001.

[8] C. P. Hurrell et al., “An 18 b 12.5 MHz ADC with 93 dB SNR,” in EEE ISSCC Dig. Tech. Papers, Feb. 2010.

[9] W. Liu, P. Huang, and Y. Chiu, "A 12 b 22.5/45 MS/s 3.0 mW 0.059 mm CMOS SAR ADC achieving over 90 dB SFDR," in IEEE ISSCC Dig. Tech. Papers, Feb. 2010.

[10] S. M. Louwsam et al., "A 1.35 GS/s, 10 b, 175 mW time-interleaved AD converter in 0.13 m CMOS," IEEE J. Solid-State Circuits, vol. 43, no. 4, pp. 778–786, Apr. 2008.

[11] B. E. Amazeen, M. C. W. Coln, and G. R. Carreau, "Quasi-differential successive- approximation structures and methods for converting analog signals into corresponding digital signals." U.S. Patent 6,400,302, June,2002.

[12] W. Liu, Y. Chang, S.-K. Hsien, B.-W. Chen, Y.-P. Lee, W.-T. Chen, T.-Y. Yang, G.-K. Ma, and Y. Chiu, "A 600MS/s 30mW 0:13_m CMOS ADC array achieving over 60dB SFDR with adaptive digital equalization," ISSCC2009 Digest of Technical Papers, vol. 52, 2009, pp. 82-83.

[13] W. Liu and Y. Chiu, "An equalization-based adaptive digital background calibration technique for successive approximation analog-to-digital converters," in ASIC, 2007. ASICON'07. 7th International Conference on, pp. 289-292, 2007.

[14] P. Harpe, H. Hegt, and A. Roermund, Smart AD and DA Conversion: Springer, 2010.

[15] W. Liu, P. Huang, and Y. Chiu, "A 12-bit, 45-MS/s, 3-mW Redundant Successive-Approximation-Register Analog-to-Digital Converter With Digital Calibration," IEEE Journal of Solid-State Circuits, vol. 46, pp. 2661-2672, 2011.

[16] R. Xu, B. Liu, and J. Yuan, "Digitally Calibrated 768-kS/s 10-b Minimum-Size SAR ADC Array With Dithering," IEEE Journal of Solid-State Circuits, vol. 47, pp. 2129-2140, 2012.

[17] S. Rahbar and E. Farshidi "Digital Background Calibration of Radix 1.83 Successive Approximations Register Analog-to-Digital Converter using the Split Architecture," Technical Journal of Engineering and Applied Sciences, vol. 3, no. 2, pp. 233-238, 2013.

[18] J. A. McNeill, K. Y. Chan, M. C. W. Coln, C. L. David, and C. Brenneman, "All-Digital Background Calibration of a Successive Approximation ADC Using the Split ADC Architecture," Circuits and Systems I: IEEE Transactions on Regular Papers, vol. 58, pp. 2355-2365, 2011.

[19] L. Du, N. Ning, S. Wu, and Y. Liu, "A digital background calibration technique for SAR ADC based on capacitor swapping," IEICE , vo11, no. 12, pp. 1-11, 2014.

[20] J. Wu, A. Wu, and Y. Du, "Dithering-based calibration of capacitor mismatch in SAR ADCs," Electronic Letters IET, vol. 52, no. 19, pp. 1198-1600, 2016.

Shabnam Rahbar was born in Brogerd, Iran, in 1984. She received the B.Sc degree in 2008 from Shiraz University, Shiraz, Iran, and the M.Sc degree in 2012 from Shahid Chamran University of Ahvaz, Ahvaz, Iran. Her main research area is calibration of data converters.

Ebrahim Farshidi was born in Shoushtar, Iran, in 1973. He received the B.Sc degree in 1995 from Amir Kabir University, Iran, the M.Sc degree in 1997 from Sharif University, Iran and the Ph.D degree in 2008 from electrical engineering at IUT, Iran, all in electronic engineering. He worked for Karun Pulp and Paper Company during 1997–2002. From 2002 he has been with shahid chamran university, Ahvaz, where he is currently Professor of electrical engineering department. He is author of more than 100 technical papers in electronics. His areas of interest include current-mode circuits design, and data converters.

A Model for Mobile Code Computing Paradigms in Computer Networks

Hodjat Hamidi*
Department of Industrial Engineering, K. N. Toosi University of Technology, Tehran, Iran
h_hamidi@kntu.ac.ir
Maryam Parvini
Department of Industrial Engineering, K. N. Toosi University of Technology, Tehran, Iran
parvini@mail.kntu.ac.ir

Received: 22/Feb/2016 Revised: 18/Feb/2017 Accepted: 08/Mar/2017

Abstract

This paper presents a reliable model for mobile codes in distributed networks, which represents reliable mobile agent execution. The model ensures non-blocking mobile agent execution and forces the once property without relying on correct fault detection. A mobile agent execution is blocking if a fault of agent prevents the agent from continuing in its execution. The once problem is related to non-blocking in the sense that solutions to the latter may lead to multiple executions of the mobile agent. A solution to reliable mobile agent execution needs to ensure both the non-blocking and once properties. The analytical results show new theoretical perceptions into the statistical behaviors of mobile agents and provide useful tools for executing mobile agents in networks. The results show that agents' behavior is influenced by places' characteristics and the agents' behavior can be managed to network. In this paper, we analyzed the average time consuming of mobile agents between two places. The approach, Fault-Tolerant approach for mobile codes offers a user-transparent fault tolerance which can be selected by the user for every single application given to the environment. Thereby, the user can decide for every application weather it has to be treated fault-tolerant or not. We proposed a reliable execution model of mobile codes and analyzed the life expectancy, including the average time consuming of mobile agents between two places, the average number of places agents will visit, and the agents' life expectancy.

Keywords: Mobile Computing; Mobile Code; Computer Network; Computing Paradigms.

1. Introduction

In view of the deficiencies of the client/server paradigm, the mobile code paradigm has been developed as an alternative approach for distributed application design. In the client/server paradigm, programs cannot move across different places and must run on the places they reside on. The mobile code paradigm, on the other hand, allows programs to be transferred among and executed on different computers. By allowing code to move between places, programs can interact on the same computer instead of over the network. Therefore, communication cost can be reduced. Besides, mobile agent [1-2] programs can be designed to work on behalf of users autonomously. This autonomy allows users to delegate their tasks to the mobile agents, and not to stay continuously in front of the computer terminal. The promises of the mobile code paradigm bring about active research in its realization. Most researchers, however, agree that security concerns are a hurdle [3]. In this paper, we investigate these concerns. A mobile agent is a software program which migrates from a site to another site to perform tasks assigned by a user. For the mobile agent system to support the agents in various application areas, the issues regarding the reliable agent execution, as well as the compatibility between two different agent systems or the secure agent migration, have been considered. Some of the proposed schemes are either replicating the agents [4-5] or checkpointing the agents [6-7]. For a single agent environment without considering inter-agent communication, the performance of the replication scheme and the checkpointing scheme is compared in [8] and [9]. In the area of mobile agents, only few work can be found relating to fault tolerance. Most of them refer to special agent systems or cover only some special aspects relating to mobile agents, e. g. the communication subsystem. Nevertheless, most people working with mobile agents consider fault tolerance to be an important issue [10]. Cluster, and therefore parallel applications running on them, are very susceptible for failures of components of the cluster. However, programmers and users of distributed applications are interested in their algorithms and solutions. They expect fault tolerance as a service from the underlying run time system. These considerations show the necessity for a design, which enables user-transparent fault tolerance in agent environments. Current agent systems, and also the underlying operating systems, provide this feature only insufficiently, if at all. In this paper we introduce an approach for such a design. It can be applied to different agent systems, if they fulfill certain requirements as discussed below. The approach, Fault-Tolerant approach for mobile agents offers a user-transparent fault tolerance which can be selected by the user for every single

* Corresponding Author

application given to the environment. Thereby, the user can decide for every application weather it has to be treated fault-tolerant or not.

The paper is organized as follows: Section 2 presents the main security challenge of mobile code. In Section 3, the security modeling for the mobile agent and model failures are explained. In Section 4, the fault-tolerant execution model is introduced. The simulation result is discussed in section 5. Conclusion is given in Section 6.

2. The Main Security Challenge of Mobile Code

The main security challenge of mobile code systems lies on the protection of agents. When an agent executes on a remote host, the host is likely to have access to all the data and code carried by the agent. If by chance a host is malicious and abuses the code or data of an agent, the privacy and secrecy of the agent and its owner would be at risk.

Seven types of attack by malicious hosts [2] can be identied:

- Spying out and manipulation of code;
- Spying out and manipulation of data;
- Spying out and manipulation of control flow;
- Incorrect execution of code;
- Masquerading of the host;
- Spying out and manipulation of interaction with other agents; and
- Returning wrong results of system calls to agents

There are a number of solutions proposed to protect agents against malicious hosts [10], which can be divided into three streams:

- Establishing a closed network: limiting the set of hosts among which agents travel, such that agents travel only to hosts that are trusted.
- Agent tampering detection: using specially designed state-appraisal functions to detect whether agent states have been changed maliciously during its travel.
- Agent tampering prevention: hiding from hosts the data possessed by agents and the functions to be computed by agents, by messing up code and data of agents, or using cryptographic techniques.

Depending on the choices made on the client and server sides, the following variants of mobile code computing paradigms can be identified [11-12]:

In the Remote Evaluation (REV) paradigm, component A sends instructions specifying how to perform a service to component B. These instructions can, for instance, be expressed in Java byte code. Component B then executes the request using its own resources, and returns the result, if any, to A. Java Servers are an example of remote evaluation [13].

In the Code on Demand (CoD) paradigm, the resources are collocated with component *A*, but *A* lacks the knowledge of how to access and process these resources in order to obtain the desired result. Rather, it gets this information from component *B*. As soon as *A* has the necessary know-how (i.e., has downloaded the code from *B*), it can start executing. The mobile agent computing paradigm is an extension of the REV paradigm. Whereas the latter focuses primarily on the transfer of code, the mobile agent paradigm involves the mobility of an entire computational entity, along with its code, the state, and potentially the resources required to perform the task. As developer-transparent capturing and transfer of the execution state (i.e., runtime state, program counter, and frame stacks, if applicable) requires global state models as well as functions to externalize and internalize the agent state, only few systems support this *strong mobility* scheme. In particular, Java-based mobile agent platforms are generally unsuitable for this approach, because it is not possible to access an agent's execution stack without modifying the Java Virtual Place. Most systems thus settle for the *weak mobility* scheme where only the data state is transferred along with the code. Although it does not implicitly transport the execution state of the agent, the developer can explicitly store the execution state of the agent in its member attributes. The values of these member attributes are transported to the next place. The responsibility for handling the execution state of an agent thereby resides with the developer. In contrary to REV, mobile agents can move to a sequence of places, i.e., can make multiple hops. The mobile code paradigm is actually a collective term, applicable wherever there is mobility of code. Different classes of code mobility can be identified, whereas Ghezzi and Vigna proposed three of them, namely *remote evaluation*, *code on demand* and *mobile agent* [14-15].

In particular, the code that is to be executed for the specific task. In the mobile code paradigms (*remote evaluation*, *code on demand*, and *mobile agent*), the *know-how* moves from one side to another side regarding where the computation takes place; while in the client/server paradigm, the *know-how* is stationary on the remote (server) side. *Resources* are the input and output for the code, whereas *processor* is the abstract place that carries out and holds the state of the computation. The arrows represent the directions in which the specific item should move before the required task is carried out. Ghezzi and Vigna's classification, [15], is found to be comprehensive and representative of most existing mobile code paradigms (such as the rsh utility, Java applets and mobile agent systems), and we will base our discussion on this classification.

3. Security Modeling

There are several fault tolerance issues that need to be addressed in our approach, just as in other schemes. For example, when storage space is exceeded in data bin services, some form of queue management is implemented (much like routers discard packets under certain load conditions). One or more trusted third parties can be used for data collection activities or task agent

hosting (instead of the originating host) to allow for disconnected host operations. Timeout of task agents that must wait for results of both the computation agent and the data collection agents can be mitigated by providing time-based services that determine when agents have been unreasonably detained or diverted.

As with any multi-agent or mobile agent system, recovery from errors when messages are not delivered or when migration is not possible needs to be addressed. Failure of data bin services would require an alternative or default data storage service in the network if the host facility becomes unavailable. Failure of the original task agent, failure of one or more computation agents, and failure of data collection agents can be mitigated by such approaches as the shadow model of [14]. Other work on fault-tolerance such as [15-42] provides approaches to mitigate host failures and malicious activity. Denial of service or random alterations of the code are not preventable because the agent server has ultimate power over an agent by having access to executable code and updatable state—though such activity can be detectable. When multi-hop agents with dependent (aggregated) data are used, the ability to mask or guard the function itself is needed to protect the computation agent against smart alterations of the code. We are currently researching other means to accomplish this aspect of agent protection [16] and plan to incorporate future results in consideration of multi-hop migrations. We also do not address the ability to keep keys used by both the computation and collection agent private, though it is an important issue with planned future research along the lines of work such as [17-18].

Several types of faults can occur in agent environments. Here, we first describe a general fault model, and focus on those types, which are for one important in agent environments due to high occurrence probability, and for one have been addressed in related work only insufficiently.

- Node failures: The complete failure of a compute node implies the failure of all agent places and agents located on it. Node failures can be temporary or permanent.
- Failures of components of the agent system: Failures of agent places, or components of agent places become faulty, e. g. faulty communication units or incomplete agent directory. These faults can result in agent failures, or in reduced or wrong functionality of agents.
- Failures of mobile agents: Mobile agents can become faulty due to faulty computation, or other faults (e. g. node or network failures).
- Network failures: Failures of the entire communication network or of single links can lead to isolation of single nodes, or to network partitions.
- Falsification or loss of messages: These are usually caused by failures in the network or in the communication units of the agent systems, or the underlying operating systems. Also, faulty transmission of agents during migration belongs to this type.

Especially in the intended scenario of parallel applications, node failures and their consequences are important. Such consequences are loss of agents, and loss of node specific resources. In general, each agent has to fulfill a specific task to contribute to the parallel application, and thus, agent failures must be treated. In contrast, in applications where a large number of agents are sent out to search and process information in a network, the loss of one or several mobile agents might be acceptable [19-20].

Places, or agents can fail and recover later. A component that has failed but not yet recovered is called down; otherwise, it is up. If it is eventually permanently up, it is called good [21]. In this paper, we focus on crash failures (i.e., processes prematurely halt). Benign and malicious failures (i.e., Byzantine failures) are not discussed. A failing place causes the failure of all agent running on it. Similarly, a failing node causes all places and agents on this node to fail as well. We do not consider deterministic, repetitive programming errors (i.e., programming errors that occur on all agent replicas or places) in the code or the place as relevant failures in this Context. Finally a link failure causes the loss of the messages or agents currently in transmission on this link and may lead to network partitioning. We assume that link failures (and network partitions) are not permanent. The failure of a component (i.e., agent, place, node, or communication link) can lead to blocking in the mobile agent execution. Assume, for instance that place P_2 fails while executing a_2 (Fig. 1). While P_2 is down, the execution of the mobile agent cannot proceed, i.e., it is blocked. Blocking occurs if a single failure prevents the execution from proceeding. In contrast, and execution is non-blocking if it can proceed despite a single failure, the blocked mobile agent execution can only continue when the failed component recovers.

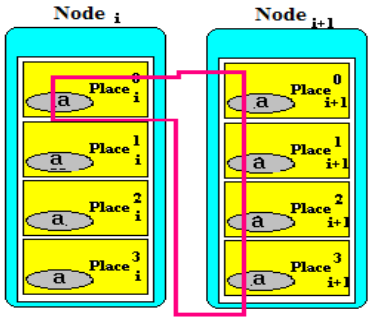


Fig. 1. the redundant places mask the place failure
(Shaded rectangles represent transactional message queues, whereas the dotted line indicates the borders of a node transaction), [2]

This requires that recovery mechanism be in place, which allows the failed component to be recovered. If no recovery mechanism exists, then the agent's state and, potentially, even its code may be lost. In the following, we assume that such a recovery mechanism exists (e.g., based on logging [22-23]. Replication prevents blocking. Instead of sending the agent to one place at the next node, agent replicas are sent to a set $node_i$ of places $P_i^0, P_i^1, \ldots$

(Fig. 1). We denote by a_i^j the agent replica of a_i executing on place P_i^j, but will omit the superscripted index if the meaning is clear from the context. Although a place may crash (i.e., $node_1$ in Fig. 1), the agent execution does not block. Indeed, P_2^1 can take over the execution of a1 and thus prevent blocking. Note that the execution at $node_0$ and $node_2$ is not replicated as the agent is under the control of the user. Moreover, the agent is only configured at the agent source and presents the results to the agent owner at the agent destination. Hence, replication is not needed at these nodes.

Despite agent replication, network partitions can still prevent the progress of the agent. Indeed, if the network is partitioned such that all places currently executing the agent at $node_i$ are in one partition and the places of $node_{i+1}$ are in another partition, the agent cannot proceed with its execution. Generally (especially in the Internet), multiple routing paths are possible for a message to arrive at its destination. Therefore, a link failure may not always lead to network partitioning. In the following, we assume that a single link failure merely partitions one place from the rest of the network .Clearly, this is a simplification, but it allows us to define blocking concisely. Indeed , in the approach presented in this article, progress in the agent execution is possible in a network partition that contains a majority of places .If no such partition exists , the execution is temporally interrupted until a majority partition is established again ,Moreover , catastrophic failures may still cause the loss of the entire agent. A failure of all places in $node_1$ (Fig. 1), for instance, is such a catastrophic. Failure (assuming no recovery mechanism is in place). As no copy of *a1* is available any more, the agent *a1* is lost and, obviously, the agent execution can no longer proceed .In other words, replication does not solve all problems. The definition of non-blocking merely addresses single failures per node as they cover most of the failures that occur in a realistic environment. In the next section, we classify the places in $node_i$ into iso-places and hetero – places according to their properties [16-17].

An agent *"a"* can commit if all or some of the surrogates commit depending on the commitment condition Com (*a*). Each agent is also realized by using the XA interface [18-20] which supports the two-phase commitment protocol. Each surrogate issues a prepare request to a server on receipt of a prepare message from the agent. If prepare is successfully performed, the surrogate sends a prepared message to the agent. Here, the surrogate is referred to as committable. Otherwise, the surrogate aborts after sending aborted to the agent. The agent receives responses from the agents after sending prepare to the surrogates. On receipt of the responses from surrogates, the agent makes a decision on commit or abort based on the commitment condition. In the atomic condition, the agent sends commit only if prepared is received from every surrogate. The agent sends abort to all committable servers if aborted is received from at least one surrogate. On receipt of abort, a committable surrogate aborts. In the at-least-one commitment condition, the agent sends commit to all committable servers only if prepared is received from at least one object server. Surrogate a_i asks the other surrogate if they had committed. Suppose the surrogate a_i is faulty before receiving prepared. Here, a_i is abort able. If the surrogate a_i is recovered, a_i unilaterally aborts (Fig.2).

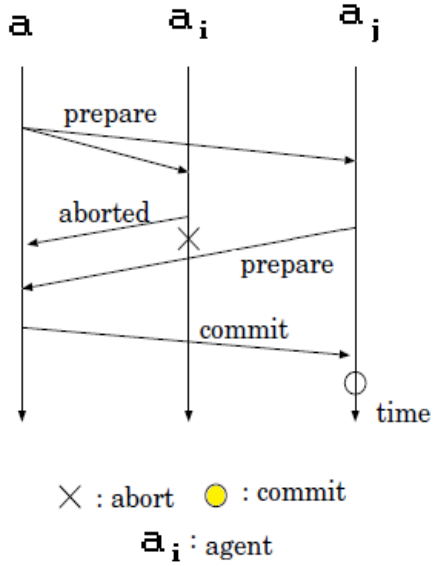


Fig. 2. Conditional commitment [4].

Now, let us consider a mobile agent travelling through *n* places on the network. Each place, and the agent itself, is modeled as an abstract node as in [17]. We consider only the standard attack phase described in [18] by malicious places. On arrival at a malicious place, the mobile agent is subject to an attack effort from the place. Because the place is modeled as a node, it is reasonable to estimate the attack effort by the number of instructions for the attack to carry out, which would be linearly increasing with time. On arrival at a non-malicious place, the effort would be constant zero. Let the agent arrive at place *i* at time T_i, for *i=1,2...n*. Then the effort of place *i* at total time / would be described by the *time-to-effort function* [1, 2]:

$$E_i(t) = k_i(t - T_i), \quad (1)$$

where *k* is a constant.

We may call the constant k_i the *coefficient of malice*. The larger the k_i, the more malicious place *i* is (k_i =0 if place *i* is non-malicious). Furthermore, let the agent stay on place *i* for an amount of time t_t, then there would be breach to the agent if and only if the following breach condition holds:

$E_i(t_i+T_i)$ > effort to next breach by place *i* (2)

i.e., $k_i t_i$ > effort to next breach by place *i*

As seen from [19-20], it is reasonable to assume exponential distribution of the effort to next breach, so we have the *probability of breach at place i*,

P (breach at place *i*) = P (breach at time t_i+T_i) (3)

= P (breach at effort $k_i t_i$)

$= 1 - exp(-vk_i t_i)$, v is a constant

$= 1 - exp(-\lambda_i t_i)$, $\lambda_i = vk_i$

We may call v the *coefficient of vulnerability* of the agent. The higher the v, the higher is the probability of breach to the agent. Therefore, the *agent security E* would be the probability of no breach at all places, i.e.

$$E = e^{-\sum_{i=1}^{n} \lambda_i t_i} \quad (4)$$

Suppose that we can estimate the coefficients of malice k_i's for places based on trust records of places, and also estimate the coefficient of vulnerability v of the agent based on testing and experiments, then we can calculate the desired time limits T_i's to achieve a certain level of security *E*. Conversely, if users specify some task must be carried out on a particular place for a fixed period of time, we can calculate the agent security *E* for the users based on the coefficients of malice and vulnerability estimates.

4. The Fault Tolerant Execution Model and Evaluation

For a large network with a large number of node and place, suppose that agents can be generated from every place on networks, provide mobile agents an execution environment. Initially, there are a pile of tasks generated in the network .Then a pile of agents, whose number is equal to that of the tasks, is generated. Each task is carried by an agent .Those agents wander among places in the network to search for their destinations. At each place, agents have local information about the error rate of each adjoin link, but they do not have global knowledge on the state of the network. The sequence of places visited by the agent compose the agent's itinerary. Agents' itineraries can be either static or dynamic. A static itinerary is entirely defined at the source and does not change during the agent traveling; whereas a dynamic itinerary is subject to modifications by the agent during its execution [21].

Since mobile agents are capable of sensing the execution environment and reacting autonomously to changes [22], a dynamic itinerary on the fly .Let P_i denote the i-th place in the itinerary and P (i) denote the set consisted by the neighbor places of P_i .The number of neighbor places in set P (i) is denoted by l_i, i.e., the connectivity degree of place P_i. Once an agent reaches a place it executes locally. After completed its execution, the agent selects a place from P (i) to move to. Suppose that there is an error rate for each candidate direction, mobile agents will prefer a route with a low error rate to shun faults. The selected place in P (i) is denoted by P_{i+1}^1. In case that a failure takes place on P_{i+1}^1, the agent is blocked and has to return to the previous place P_i. Then, it will reselect another neighbor place from P (i) and move to. The j-th selected place in P(i) is denoted P_{i+1}^1. An agent is supposed will not jump to the same neighbor place twice since in a general way a failure place will not recover in a very short time. This process will continue until the agent successfully enters a place and completes its execution there. The final visited place in P (i) is denoted by P_{i+1}.

Communication between consecutive nodes P_i and P_{i+1} is based on transactional message queues, shown as shaded rectangles in Fig. 1. At each node, a place retrieves the agent from its input queue, executes the agent, and places the resulting agent in the input queues of the next node's places as one transaction. A place P_i can only commit the distributed transaction when it is elected by the places in $node_i$, when it receives a majority of votes. Rothermel uses a 2-phase commit protocol [23] to commit the transactions, the election protocol thereby acting as a resource manager to the transaction manager. Modeling reliable mobile agent execution based on two different, interfering problems leads to a more complex solution than ours. In addition, understanding the weaknesses of such a solution is difficult and tedious. Our solution, however, is specified in terms of a *single* problem, the consensus problem, an intensively studied problem with well-understood solutions.

In Rothermel's model, the execution of the agent as well as the forwarding of the agent from node P_i to P_{i+1} run as a transaction. Our model, in contrast, clearly decouples the mechanisms that provide fault tolerance from the execution properties of the agent operations. In particular, the agent operations do not need to run as a transaction. If they do, they have their own transaction manager.

In an asynchronous distributed system, there are no bounds on transmission delays of messages or no relative process speeds. Therefore, when a mobile agent is blocked by reason of a failure in an asynchronous distributed system, the agent owner cannot correctly determine whether the agent has failed or is merely slow [24-28]. Therefore, the reliability of agents' execution is paramount for measuring the network performance. We treat this problem as a probability problem using the behavior of mobile.

Agents to build a probability estimation on the number of places an agent can visit. Let S_i denote the number of places selected by an agent in set P(i). The event indicates that the agent cannot enter the place P_i^j in set P (i), then the parameter p measures the incidence of failure in the network. The average number of selected places in set P (i), denoted by M (Si), and satisfies.

$$M(S_i) = (1-(p(1-p))^{d_i}) / (1-p+p^2))) \quad (5)$$

i=1,2… and j=1,2,…….

From Fig.3, it is easy to see that the average number of places an agent will selected in a neighbor place set is an increase function on both error rate p and the number of neighboring places d_i. Furthermore, if the time cost for passing a link approximates to a constant *k*, we have estimate the average time consumption for mobile agents entering a place in set P (i).

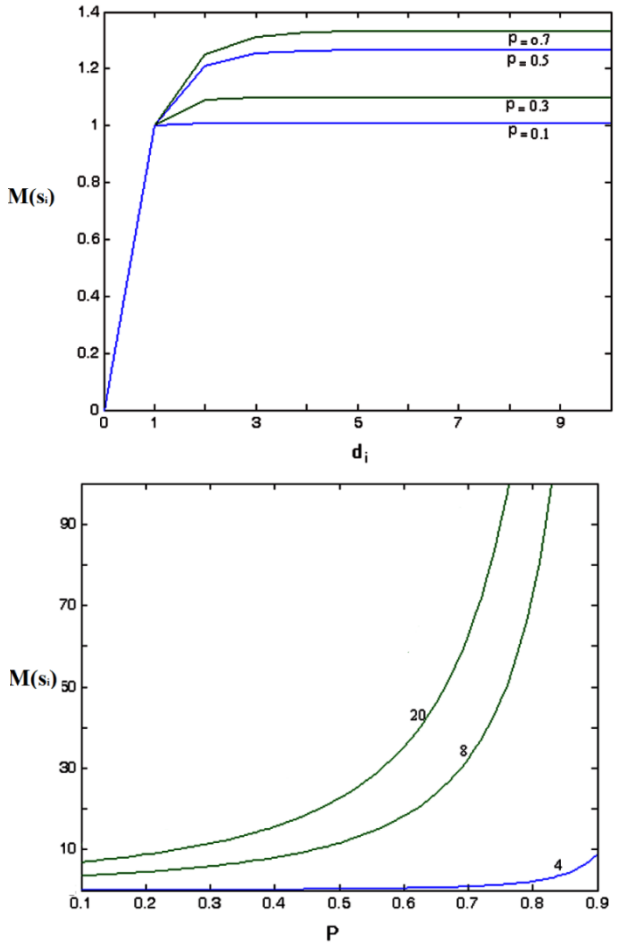
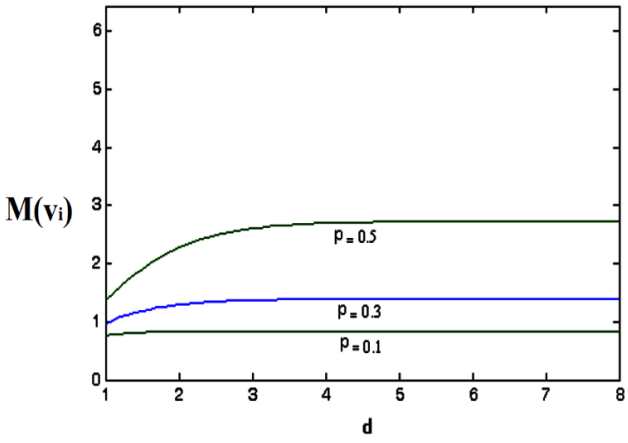


Fig. 3. The Changes of M (Si) over p and d_i

By the assumption that the time consumption for an agent passing a link is *q*, the time consumption of the period that an agent moves to a down place and returns to the previous place equals to *2q*. Hence, Agents' life expectancy satisfies.

$$M[v] = (q(1-q^d)/((1-q)(1+p))/((1-p((1-q^d)/(1-q))) \quad (6)$$



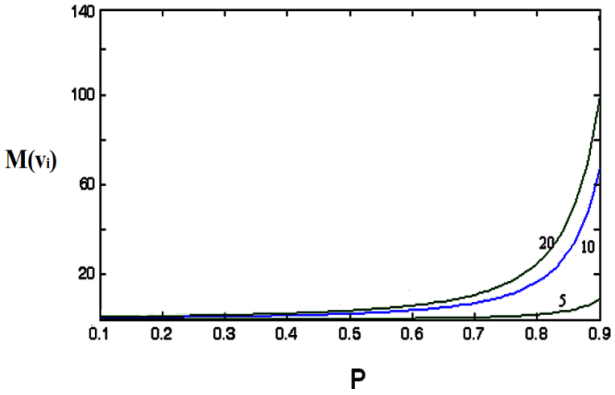


Fig. 4.The Changes of M (vi) over P and d

Fig. 4. shows the changes of agents' average life expectancy. It is easy to see that the average life expectancy is an increase function on both the error rate and the network connectivity. In particular, it is a convex function on the parameter *d* and a concave function on the parameter p.

5. Results

In this experiment, we change the number of nodes from 2 to 30 and use one mobile agent. The result is shown in Fig. 5(a). We observe that both the execution time and the energy consumption using either computing model grow as the number of nodes increases. But the execution time of the client/server model grows much faster than the mobile-agent-based model. This is because as the number of nodes increases, the server has to deal with more connections requested by the clients at the same time, which elongates the execution time. On the other hand, the mobile agent model is less influenced by the number of nodes because there are far fewer connections at one time for the mobile agent model. The figure also shows that the client/server model performs a little better than the mobile agent model from both the execution time and energy consumption perspectives. This happens when the mobile agent model needs more connections than the client/server model in order to send and receive mobile agents. It also happens when the overhead of the mobile agent surpasses the overhead of the client/ server model.

In this experiment, we fix the node number at 100 but change the number of mobile agents from 1 to 50. Without loss of generality, we assume each agent migrates the same number of nodes. We expect a constant profile from the client/server model since it is irrelevant to the number of mobile agents. We can see from Fig. 5(b) that the execution time of the mobile agent model is always less than that of the client/server model because the node number is large. Interestingly, the execution time of the mobile agent model decreases as the number of mobile agents increases and reaches the lowest point when there are five mobile agents. Then, the execution time begins to climb. This is because more mobile agents

will reduce the number of nodes each agent migrates, thus reducing the execution time. But more mobile agents also cause more connections and more overheads. As the number of mobile agents increases, the energy consumption also increases in linear and the mobile agent model actually consumes more energy when m>15.

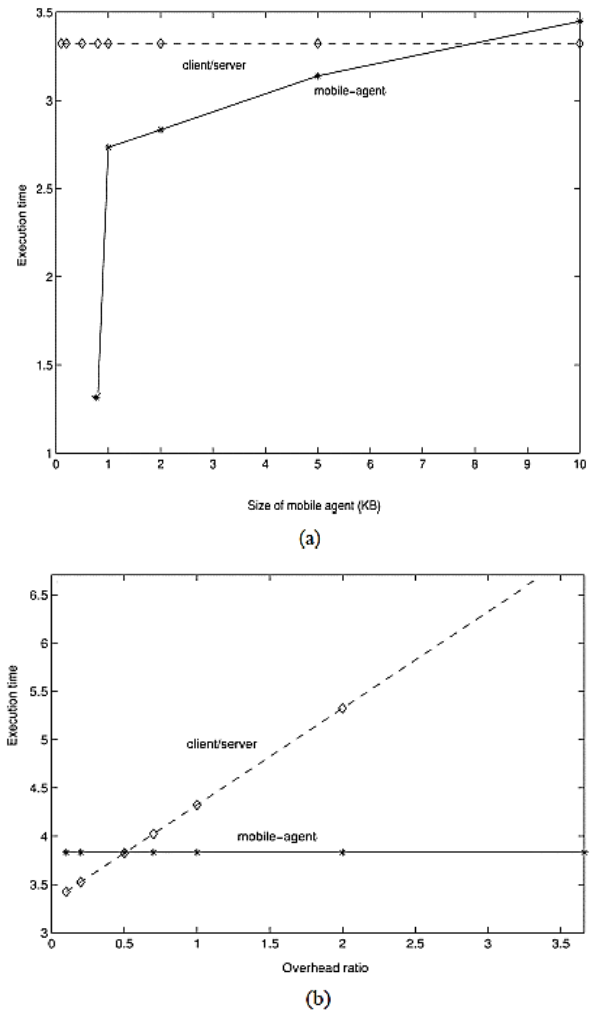


Fig. 5. (a) Effect of the mobile agent size (s). Execution time. (b) Effect of the overhead ratio and Execution time.

Access time from time when the application program starts to time when the application program ends, is measured for Agents and client-server model. Fig.6 shows the access time for number of object servers. The mobile agent's shows that Aglets classes are not loaded when an agent A arrives at an object server. Here, the agent can be performed after mobile agents are loaded. On the other hand, the mobile agent's with replication means that an agent manipulates objects in each object server where mobile agents are already loaded, i.e. the agent comes to the object server after other agents have visited on the object server. As shown in Fig.6, the client-server model is faster than the transactional agent. However, the transactional agent is faster than the client-server model if object servers are frequently manipulated, i.e. mobile agents with replication are a priori loaded.

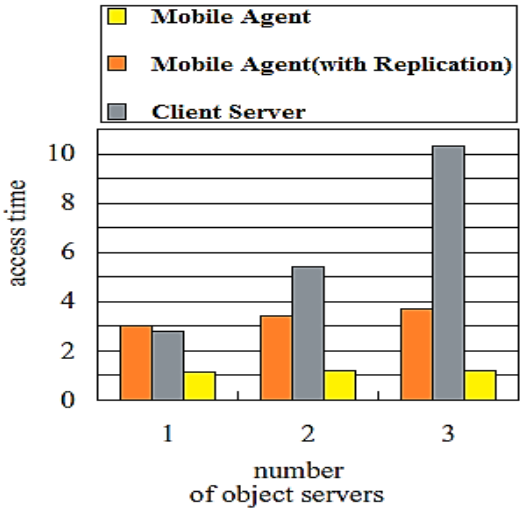


Fig. 6. Client server vs. agent.

6. Conclusion

To achieve reliable in the context of mobile codes, we first have specified reliable mobile agent execution in terms of two properties: non-blocking and once execution. Replication overcomes the blocking problem. This paper shows how the present approach to reliable mobile agent execution can be used to achieve non-blocking mobile agent execution. The use of mobile agent, however, is critical and requires reliability in regard to mobile agent failures that may lead to bad response time and hence the availability of the system may lost. In this paper, a fault tolerance technology is proposed in order that the system autonomously detect and recover the fault of the mobile agent due to a failure in a transmission link. The key idea is the use of stochastic regularities of mobile agent's behavior-all the mobile agents in the network as a whole can be stochastically characterized though a single mobile agent may act randomly. In this paper, we proposed a reliable execution model of mobile agents and analyzed the life expectancy, including the average time consuming of mobile agents between two places, the average number of places agents will visit, and the agents' life expectancy.

References

[1] H. Hamidi and K. Mohammadi, “Evaluation of Fault Tolerant Mobile Agents in Distributed Systems, “International Journal of Intelligent Information Technologies (IJIIT 5(1)), pp.43-60, Janauary-March 2009.

[2] H. Hamidi and A. Vafaei, "Evaluation of Security and Fault-Tolerance in Mobile Agents," Proc.Of the 5th IEEE Conf. on Wireless & Optical Communications Networks (WOCN2008), May 5, 6 and 7, 2008.

[3] H. Hamidi and K. Mohammadi, "Modeling and Evaluation of Fault Tolerant Mobile Agents in Distributed Systems," Proc. Of the 2th IEEE Conf. on Wireless & Optical Communications Networks (WOCN2005), pp. 91-95, March 2005.

[4] S. Pleisch and A. Schiper, "Modeling Fault-Tolerant Mobile Agent Execution as a Sequence of Agree Problems," Proc. of the 19th IEEE Symp. on Reliable Distributed Systems, pp. 11-20,2000.

[5] S. Pleisch and A. Schiper, "FATOMAS - A Fault-Tolerant Mobile Agent System Based on the Agent-Dependent Approach," Proc. 2001 Int'l Conf on Dependable Systems and networks, pp.215-224, 2001.

[6] M. Strasser and K. Rothermel, "System Mechanism for Partial Rollback of Mobile Agent Execution," Proc. 20th In!'l Conf on Distributed Computing Systems, 2000.

[7] T. Park, I. Byun, H. Kim and H.Y. Yeom, "The Performance of Checkpointing and Replication Schemes for Fault Tolerant Mobile Agent Systemss ," Proc. 21th IEEE Symp. On Reliable Distributed Systems, 2002.

[8] M. Izatt, P. Chan, and T. Brecht. Ajents: Towards an Environment for Parallel, Distributed and Mobile Java Applications. In Proc. ACM 1999 Conference on Java Grande, pages 15-24, June 1999.

[9] A. S. Tanenbaum, "Distributed Operating Systems", prentice Hall, Inc, 1995.

[10] H.W. Chan, K.M. Wong, R. Lyu “Design, Implementation, and Experimentation on Mobile Agent Security for Electronic Commerce Application,” Distributed systems, s. Mullender,ed., second ed., pp. 199-216, Reading, Mass.: Addison-wesley, 1993.

[11] X. Defago, A. schiper,and N. sergent, “semi-passive Replication,”proc. 17th IEEE symp. Reliable Distributed system (SRDS ‘98), pp. 43-50, oct. 1998.

[12] M.J. Fischer, N.A. Lynch and M.S. paterson, “Impossibility of Distributed consensus with one Faulty process,” Proc. second ACM SIGACT-SIGMOD symp. Principles of Database system, pp. 17-24, Mar.1983.

[13] M. S. Greenberg, J. C. Byington, and D. G. Harper. “Mobile Agents and Security”. In Volume 367, IEEE Communications Magazine. IEEE Press, July 1998.

[14] C. F. Tschudin. “Mobile Agent Security”. In M. Klusch, Intelligent Information Agents. Forthcoming LNCS. http://www.docs.uu.se/~tschudin/pub/cft-1999-iia.ps.gz, 1999.

[15] C. Ghezzi, G.Vigna. “Mobile Code Paradigms and Technologies: A Case Study”. In Kurt Rothermet, Radu Popescu-Zeletin, editors, Mobile Agents, First International Workshop, MA’97, Berlin, Germany, April 1997, Proceedings, LNCS 1219, p. 39-49. Springer, 1997.

[16] A. Fuggetta, G. P.Picco, & G.Vigna, Understanding code mobility. IEEE Transactions on Software Engineering, 24(5). pp. 342–361, 1998.

[17] W. Stallings. Cryptography and Network Security, Principles and Practice. Prentice Hall, 2nd edition, 1999.

[18] T. Sander and C. F. Tschudin. “Protecting Mobile Agents against Malicious Hosts”. In Giovanni Vigna, editor, Mobile Agents and Security, LNCS 1419, p. 44-60. Springer, 1998.

[19] F. Hohl. “Time Limited Blackbox Security: Protecting Mobile Agents from Mali cious Hosts”. In Giovanni Vigna, editor, Mobile Agents and Security, LNCS 1419, p. 92-113. Springer, 1998.

[20] M.K. Aguilera, w. chen, and s. Toueg, “Failure Detection and consensus in the crash-Recovery Model,” Distributed computing,vol. 13,no. 2,pp. 99-125,2000.

[21] R. A. Sahner, K. S. Trivedi, and A. Puliafito. Performance and Reliablity Analysis of Computer Systems. Kluwer Academic Publishers, Boston, 1996.

[22] S. Pleisch and A. Schiper, “Fault-Tolerant Mobile Agent Execution," IEEE TRANSACTIONS ON COMPUTERS, VOL. 52, NO .2, Feb 2003.

[23] M. Strasser and K. Rothermel, "System Mechanism for Partial Rollback of Mobile Agent Execution," Proc. 20th In!'l Conf on Distributed Computing Systems, 2000.

[24] H. Hamidi, “Modeling Fault Tolerant and Secure Mobile Agent Execution in Distributed Systems, "International Journal of Intelligent Information Technologies (IJIIT 2(1)), pp.21-36, 2006.

[25] H. Hamidi and A.Vafaei, “Evaluation and Check pointing of Fault Tolerant Mobile Agents Execution in Distributed Systems,” Journal of Networks (Academy Publisher), VOL. 5, NO. 7. July 2010.

[26] A. Vafaei, H. Hamidi., S.A. Monadjemi. “A Framework for Fault Tolerance Techniques in the Analysis and Evaluation of Computing Systems” International Journal of Innovative Computing, Information and Control (IJICIC), Vol.8, No.7, July 2012.

[27] S. Bimonte, L. Sautot, L. Journaux, & B. Faivre, Multidimensional Model Design using Data Mining: A Rapid Prototyping Methodology. International Journal of Data Warehousing and Mining (IJDWM), 13(1), pp.1-35. 2017.

[28] D. Chevers, A. Mills, E. Duggan, S. Moore. “An Evaluation of Software Development Practices among Small Firms in Developing Countries: A Test of a Simplified Software Process Improvement Model.” Journal of Global Information Management, 24(3), pp. 45-70. 2016.

[29] C. Esposito, & M. Ficco, “Recent Developments on Security and Reliability in Large-Scale Data Processing with MapReduce.” International Journal of Data Warehousing and Mining (IJDWM), 12(1), pp. 49-68. 2016.

[30] F. Gharagozlou, , G. A. Mazloumi, , A. Nahvi, Nasrabadi, A. M., Foroushani, A. R., Kheradmand, A.A, Ashouri, M., Samavati, M, Detecting Driver Mental Fatigue Based on EEG Alpha Power Changes during Simulated Driving, Iranian Journal of Public Health 2015.

[31] H. Hamidi, “A Combined Fuzzy Method for Evaluating Criteria in Enterprise Resource Planning Implementation.” International Journal of Intelligent Information Technologies (IJIIT), 12(2), pp.25-52. 2016.

[32] Hamidi, H. “A Model for Impact of Organizational Project Benefits Management and its Impact on End User”, JOEUC, Volume 29, Issue 1, pp. 50-64, 2017.

[33] R. D. Johnson, Y.Li, & J. H. Dulebohn, “Unsuccessful Performance and Future Computer Self-Efficacy Estimations: Attributions and Generalization to Other Software Applications.” Journal of Organizational and End User Computing, 28(1), Pp.1-14. 2016.

[34] A. S. Kakar, "A User-Centric Typology of Information System Requirements." JOEUC, 28(1), pp. 32-55. 2016.
[35] S. Kumar, "Performance Evaluation of Novel AMDF-Based Pitch Detection Scheme," ETRI Journal, vol. 38, no. 3, pp. 425-434, 2016.
[36] Y. Liu, C.Tan, & J. Sutanto, "Selective Attention to Commercial Information Displays in Globally Available Mobile Application." Journal of Global Information Management, 24(2), pp.18-38. 2016.
[37] S.A. Monadjemi, A.Vafaei, H.Hamidi, "Analysis and Evaluation of a New Algorithm Based Fault Tolerance for Computing Systems." International Journal of Grid and High Performance Computing (IJGHPC), 4(1), pp. 37-51, 2012.
[38] S.A. Monadjemi, A.Vafaei, H.Hamidi. "ANALYSIS AND DESIGN OF AN ABFT AND PARITY-CHECKING TECHNIQUE IN HIGH PERFORMANCE COMPUTING SYSTEMS" Journal of Circuits, Systems, and Computers (JCSC), JCSC Volume 21 Number 3. 2012.
[39] A. Safdar, K. DoHyeun, "Enhanced power control model based on hybrid prediction and preprocessing/post-processing." Journal of Intelligent & Fuzzy Systems, vol. 30, no. 6, pp. 3399-3410. 2016.
[40] B. Shadloo, A. Motevalian, V. Rahimi-movaghar, M. Amin-Esmaeili, V. Sharifi, A. Hajebi, R. Radgoodarzi, M. Hefazi, Rahimi-Movaghar, "Psychiatric Disorders Are Associated with an Increased Risk of Injuries: Data from the Iranian Mental Health Survey." Iranian Journal of Public Health 45(5): pp. 623-635. 2016.
[41] J. Wu, F. Ding, M. Xu, Z. Mo, & A. Jin, "Investigating the Determinants of Decision-Making on Adoption of Public Cloud Computing in E-government." JGIM, 24(3), pp.71-89. 2016.
[42] X. Ye, T. Sakurai, "Robust Similarity Measure for Spectral Clustering Based on Shared Neighbors," ETRI Journal, vol. 38, no. 3, pp. 540-550. 2016.

Hodjat Hamidi born 1978, in shazand Arak, Iran, He got his Ph.D in computer engineering. His main research interest areas are Information Technology, Fault-Tolerant systems and applications and reliable and secure distributed systems and e- commerce. Since 2013 he has been a faculty member at the IT group of K. N. Toosi University of Technology, Tehran Iran. Information Technology Engineering Group, Department of Industrial Engineering, K. N. Toosi University of Technology.

Maryam Parvini is a master student of Information Technology Engineering at K. N. Toosi University of Technology. Her research interests include Machine learning, Knowledge Discovery and Data Mining and Customer Relationship Management.

A new Sparse Coding Approach for Human Face and Action Recognition

Mohsen Nikpour*
Department of Electrical and Computer Engineering, Babol Noushirvani University of Technology, Babol, Iran
Nikpour@mit.ac.ir
Mohammad Reza Karami Molaei
Department of Electrical and Computer Engineering, Babol Noushirvani University of Technology, Babol, Iran
Mkarami@nit.ac.ir
Reza Ghaderi
Department of nuclear Engineering, Shahid Beheshti University of Tehran, Tehran, Iran
R_ghaderi@sbu.ac.ir

Received: 27/Jul/2016 Revised: 07/Jan/2017 Accepted: 14/Jan/2017

Abstract

Sparse coding is an unsupervised method which learns a set of over-complete bases to represent data such as image, video and etc. In the cases where we have some similar images from the different classes, using the sparse coding method the images may be classified into the same class and devalue classification performance. In this paper, we propose an Affine Graph Regularized Sparse Coding approach for resolving this problem. We apply the sparse coding and graph regularized sparse coding approaches by adding the affinity constraint to the objective function to improve the recognition rate. Several experiments has been done on well-known face datasets such as ORL and YALE. The first experiment has been done on ORL dataset for face recognition and the second one has been done on YALE dataset for face expression detection. Both experiments have been compared with the basic approaches for evaluating the proposed method. The simulation results show that the proposed method can significantly outperform previous methods in face classification. In addition, the proposed method is applied to KTH action dataset and the results show that the proposed sparse coding approach could be applied for action recognition applications too.

Keywords: Sparse Coding; Manifold Learning; Graph Regularization; Affinity; Image Representation; Image Classification.

1. Introduction

Image classification is a significant task in image processing and computer vision studies. Sparse coding method can represent images using a few active coefficients [1]. Accordingly, interpreting and applying the sparse representations are easy and simplify efficient content-based image indexing and retrieval [2]. The authors in [5] have been proposed an improved CRC-RLS method for solving the poor robustness of Collaborative Representation based Classification with Regularized Least Square algorithm.

The range of sparse coding methods have been widened every day in many fields such as pattern recognition, machine learning and signal processing [3], face recognition [4,5], image classification [7,8] and action recognition [10] in recent years. Most important targets of sparse coding is the maximum signal fidelity preservation and also improving the quality of the sparse representation. To achieve these targets, many works have been done to modify the sparsity constraint. In [7] the authors to improve the sparse coding method, have added a nonnegative constraint to the objective function of basis sparse coding method. The authors in [8] have proposed a face recognition method based on the discriminative locality preserving vectors. This method is based on the discriminant analysis on the locality preserving vectors. A robust sparse coding to improve the signal fidelity has been proposed in [9]; however, in the case of the similar images, they may be transformed into identical visual words of the codebook and encoded with the same representations. The dictionary learned from the images cannot effectively encode manifold structure of the images face in this case, and the similar images from different classes may be classified in the same class accordingly. This similarity will highly challenge the robustness of existing sparse coding algorithms for image classification problems such as face images. Similar face images are lying on a manifold structure and the face images from different classes are lying on different manifold structures [10]. It has been shown that if the geometrical structure is used and the local invariance is considered, the learning performance can be significantly improved. Recently, many literatures have focused on manifold learning problems, which represent the samples from the different manifold structures. To preserve the geometrical information of the data, the authors in [11] proposed to extract a good feature representation through which the manifold structure of data is spotted.

Regarding the recent progress in sparse coding and manifold learning, we propose a novel Affine regularized sparse coding algorithm to construct robust sparse representations for classifying similar face images

* Corresponding Author

accurately. Specifically, the objective function of sparse coding has incorporated this criterion to make the new representations of the similar face images far from each other. Moreover, to improve the objective function with more discriminating power in data representation, we also incorporate the graph Laplacian term of coefficients [8] in our objective function. For more consideration, the proposed method is applied on a well-known human action recognition dataset. The experimental results verify the effectiveness of our sparse coding approach.

This paper is continued as follows: In Section 2, some related works are introduced. The sparse coding and graph regularized sparse coding is then described in Section 3. Section 4 contains the proposed method. The experiment setup and results on face and action datasets are indicated in Section 5 and consequently, some conclusions and future work are presented in Section 6.

2. Related Work

In this section, we discuss some prior papers in sparse coding and manifold learning area. In recent years, sparse coding has been widely used in many fields in computer vision. The authors in [3] proposed a feature sign search method. This method reduces the non-differentiable problem to an unconstrained quadratic programming (QP). This problem can be solved rapidly by the optimization process. Our work also uses their method to solve the proposed optimization problem. For adapting the dictionary to achieve sparse representation, the authors in [12] proposed a K-SVD method to learn the dictionary using orthogonal matching pursuit or basis pursuit. Adding nonnegative term to the sparse constraint is a method to improve the quality of sparse representations [7]. The other methods such as graph regularization [5,8] and using weighted ℓ2-norm constraint are also introduced for improving the sparse representation. In the machine learning literature, manifold learning has also attracted extensive research interest. The authors in [8] proposed a graph based algorithm, called Graph regularized Sparse Coding (GraphSC), to give sparse representations that well consider the local manifold structure of the data. By using graph laplacian as a smooth operator, the obtained sparse representations vary smoothly along the geodesics of the data manifold. Our work in addition to the affinity constraint, incorporates the graph laplacian term of coefficients [8] in the objective function, and can discover more discriminating representations for image classification.

3. Preliminaries

This section introduces sparse coding and affine graph regularized sparse coding.

3.1 Sparse Coding

Assuming a data matrix $Y = [y_1, ..., y_n] \in R_{m\times n}$ where n is the number of samples in the m-dimensional feature space. Let $\Phi = [\varphi_1, ..., \varphi_k] \in R_{m\times k}$ be the dictionary matrix where each column φ_i represents a basis vector in the dictionary, and $X = [x_1, ..., x_n] \in R_{k\times n}$ be the coding matrix where each column $\boldsymbol{x_i}$ is a sparse representation for a data point $\boldsymbol{y_i}$. Assuming the reconstruction error for a data point follows a zero-mean Gaussian distribution with isotropic covariance, while taking a Laplace prior for the coding coefficients and a uniform prior for the basis vectors, then the maximum posterior estimate of Φ and X given Y is reduced to:

$$\min_{\Phi,X} \|Y-\Phi X\|_F^2 + \alpha \sum_{i=1}^{n} |x_i| \quad st. \|\varphi_j\|^2 \le c, \forall i=1,2,...,k \tag{1}$$

In the above equation α is a parameter for regularizing the level of sparsity of the obtained codes and the approximation of initial data. The objective function in (1) is not convex in Φ and X, therefore solving the above equation is not easy in this case. But it is convex in either Φ or X. Therefore, solving this problem is done by alternatively optimizing Φ while fixing X and vice versa. As a result, the above mentioned problem can be split into two reduced least squares problems: an $\ell 1$-regularized and an $\ell 2$-constrained, both of which can be solved efficiently by existing optimization software [3,4].

3.2 Graph Regularized Sparse Coding

The authors in [8] have proposed a method called Graph Regularized Sparse Coding (GraphSC) method, which considers the manifold assumption to make the basis vectors with respect to the intrinsic geometric structure underlying the input data. This method assumes that if two data points y_i and y_j are close in the intrinsic geometry of data distribution, then their codes φ_i and φ_j are also close. Consider a set of n-dimensional data points $\{y_1, ..., y_n\}$, GraphSC constructs a p-nearest neighbor graph G with n vertices each representing a data point. Let W be the weight matrix of G, if y_i is among the p-nearest neighbor of y_j, $W_{i,j}$ =1; otherwise, $W_{i,j}$ =0. $d_i = \sum_{j=1}^{n} W_{i,j}$, $D = diag(d_1, \dots, d_n)$ and graph Laplacian $L = D-W$. A reasonable criterion for preserving the geometric structure in graph G is to minimize:

$$\frac{1}{2}\sum_{i,j=1}^{n} \|x_i - x_j\|^2 W_{ij} = \mathrm{Tr}(XLX^T) \tag{2}$$

By replacing the result into (1) the GraphSC [7] is obtained:

$$\min_{\Phi,X} \|Y-\Phi X\|_F^2 + \gamma \mathrm{Tr}(XLX^T) + \alpha \sum_{i=1}^{n} |x_i| \quad st. \|\varphi_i\|^2 \le c, \; i = 1, \dots, k \tag{3}$$

In (3) γ is a parameter for regularizing the weight between sparsity of the obtained codes and preserving the geometrical structure.

4. The Proposed Method: Affine Graph Regularized Sparse Coding

In this section, we present the Affine graph regularized sparse coding algorithm for robust image representation, which extends GraphSC by taking into account the affinity constraints on the samples.

4.1 Problem Definition

In linear sparse coding, a collection of k atoms $\varphi_1, \varphi_2, \dots, \varphi_k$ is given that forms the columns of the overcomplete dictionary matrix Φ. With a *l0*-minimization problem, the sparse codes of a feature vector $y \in R^m$ can be determined:

$$\min_{w \in R^m} \|w\|_0, \quad \text{s.t. } x = G_\Phi(w) \tag{4}$$

Where the function G_Φ is defined as $G_\Phi(w) = \Phi w$. In the proposed method the main technical difficulty is the proper interpretation of the function $G_\Phi(w)$ in the manifold setting, where the atoms $\varphi_1, \varphi_2, \dots, \varphi_k$ are now points in M and Φ now denotes the set of atoms, and because of the nonlinearity property in this case, it is no longer possible to create a matrix with atoms. Moving to the more general manifold setting, we have forsaken the vector space structure in R^m. In the linear sparse coding, each point is considered as a vector whose definition requires a reference point. However, in the affine graph regularized sparse coding approach, each point cannot be considered as a vector and therefore, must be considered as a point. This particular viewpoint is the main source of differences between linear and the proposed sparse coding.

In this paper, a new method is proposed to modify the usual notion of sparsity by adding an affine constraint to reduce the feature vectors dimension on a manifold. A vector y is defined as an affine sparse vector if it can be written as follows [13]:

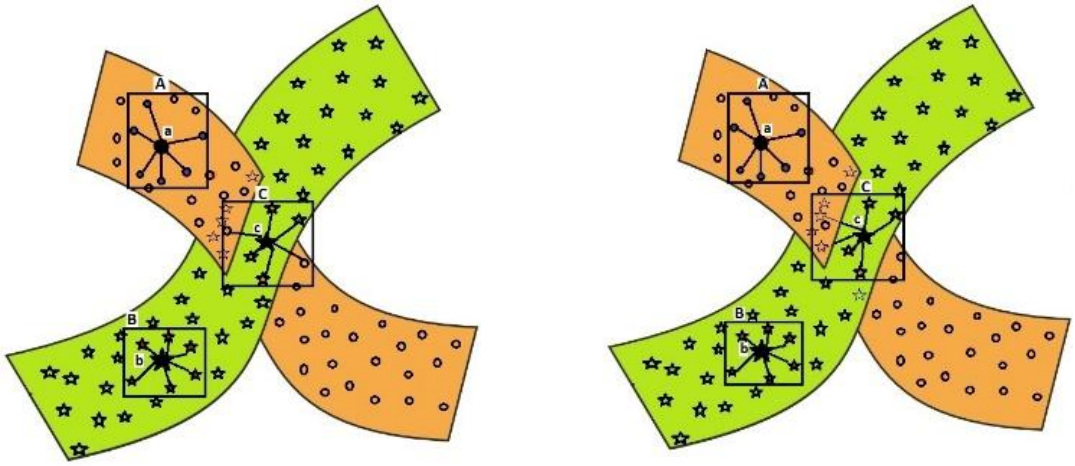
$$y = w_1\varphi_1 + w_2\varphi_2 + \cdots + w_n\varphi_n; \qquad w_1 + w_2 + \cdots + w_n = 1 \tag{5}$$

According to the definition, if the vector is constructed with combination of the affine samples, can be mapped on the space with the lower dimension. As we know, nevertheless the vectors are in the space with high dimension manifold, but they are locally on the low dimension manifold. Representing a vector in places where the manifolds are interferences is very challenging. In these cases, for representing the vectors, if the samples selection are done based on only the nearest neighbors and the sparsity term, maybe some selected samples are from the irrelevant manifold, however if the selected samples have the affinity constraint in addition, because one can consider the samples on the manifold with locally low dimension, only the samples on the relevant manifold could be selected.

For better perception of the proposed method, see Figure 1.

Two overlapped manifolds are shown in this Figure. Figure 1.a indicates a representation of the samples a,b and c regarding only the sparsity term and Figure 1.b indicates the representation of the same points regarding the manifold constraints in addition to the sparsity constraint. The samples a and b in the both Figures 1.a and 1.b are represented by the atoms from the corresponding manifolds correctly. These two samples hasn't any conflict with the other manifold.

The sample c is under different conditions. As indicated before, this sample is located on the green manifold. If you represent this sample with its adjacent atoms and only consider the sparsity term, we should consider the other manifold samples for representation same as Figure 1.a. However, if we consider $tr(XLX^T)$ and Affinity terms for its representation in addition, we will reach a better conclusion. As previously pointed out, the term $tr(XLX^T)$ emphasizes on the problem that if the samples of a manifold are closed to each other, their codes will be closed to each other as well.



a) Representation of samples a, b, c without the affinity constraint b) Representation of samples a, b, c by adding the affinity constraint

Fig. 1. The effectiveness of affinity constraint in representation of samples from the overlapped manifolds

Also, the Affinity constraint emphasizes on the problem that a collection of the closest neighbors of the concerned sample to represent every sample and then chooses a collection of weights for every sample in a way that every point is represented by the linear combination of its neighbors. The former samples are located on a manifold with high dimensions and the objective of the Affinity term is to reduce its dimensions. It is to be considered that despite the fact that the samples are on manifolds with many dimensions but they are locally located on manifolds with low dimensions. The characteristic of this new term causes the sample c to be represented with utilization of the concerned manifold data (Figure 1.b).

According to the above mentioned descriptions, we can add an affinity term to (1):

$$\min_{\Phi,X}\|Y-\Phi X\|_F^2+\gamma Tr(XLX^T)+\alpha\sum_{i=1}^{n}|x_i| \quad st.\sum_{i=1}^{n}x_i=1 \tag{6}$$

The constraint term $\sum_{i=1}^{n} x_i = 1$ is added to the main criterion as a lagrangian coefficient leading to:

$$\min_{\Phi,X}\|Y-\Phi X\|_F^2+\gamma Tr(XLX^T)+\alpha\sum_{i=1}^{n}|x_i|+\beta(1-\sum_{i=1}^{n}x_i)^2 \tag{7}$$

where β is a parameter for tuning the affinity constraint. For tuning α, β and γ parameters some experiments have been done that can be seen in the next section. In Figure 2, one can see the steps of the proposed method. After preprocessing the input data the samples are clustered using k-means algorithm to make the initial dictionary. Then the KSVD is applied for optimizing the dictionary. Finally the proposed sparse coding method is applied to extract the optimal coefficients and then classifying the test data based on the minimum error.

4.2 Solution of the proposed method

We apply the feature-sign search algorithm [3] to solve the optimization problem (7). For solving non-differentiable problems in non-smooth optimization methods, a necessary condition for a parameter vector to be a local minimum is that the zero-vector is an element of the sub-differential set containing all sub-gradients in the parameter vector [14].

Following [6,14], the optimization of the proposed sparse coding has been divided into two steps: 1) $\ell 1$-regularized least squares problem; the affine graph regularized sparse codes X are learned with dictionary Φ fixed and 2) $\ell 2$-constrained least squares problem; the dictionary Φ has been learned with affine graph regularized sparse codes X fixed. The above two steps are repeated respectively until a stop criterion is indulged.

The optimization problem in the first step can be solved by optimizing over each x_i individually.

Since (7) with *l1*-regularization is non-differentiable when x_i contains values of 0, for solving this problem, the standard unconstrained optimization methods cannot be applied.

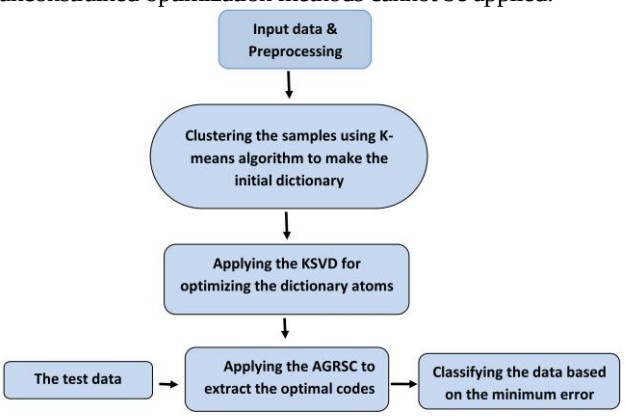


Fig. 2. The diagram of Proposed method

Several approaches have been proposed to solve the problem of this form [15,16]. In the following, we introduce an optimization method based upon coordinate descent to solve this problem [17]. It can easily be seen that (7) is convex, thus the global minimum can be achieved.

We update each vector individually by holding all the other vectors constant. In order to solve the problem by optimizing over each x_i, we should rewrite the (7) in a vector form. The reconstruction error $\|Y-\Phi X\|_F^2$ can be rewritten as:

$$\sum_{i=1}^{m}\|Y-\Phi X\|_F^2 \tag{8}$$

The Laplacian regularizer $tr(XLX^T)$ can be rewritten as:

$$Tr(XLX^T)=Tr\left(\sum_{i,j=1}^{n}L_{i,j}x_i x_j^T\right)=\sum_{i,j=1}^{n}L_{ij}x_i x_j^T=\sum_{i,j=1}^{n}L_{ij}x_i^T x_j. \tag{9}$$

□

Combining (7) , (8) , (9) the problem can be written as:

$$min\sum_{i=1}^{n}\|y_i-\Phi x_i\|_F^2+\gamma\sum_{i,j=1}^{n}L_{ij}x_i^T x_j+\alpha\sum_{i=1}^{n}|x_i|+\beta(1-\sum_{i=1}^{n}x_i)^2, \tag{10}$$

When updating x_i, the other vectors $\{x_j\}_{i\neq j}$ are fixed. Thus, we get the following optimization problem:

$$\min_{x_i}G(x_i)=\|y_i-\Phi x_i\|^2+\gamma L_{ii}x_i^T x_i+x_i^T H_i+\alpha\sum_{j=1}^{k}\left|x_i^{(j)}\right|+\beta(1-\sum_{i=1}^{n}x_i)^2 \tag{11}$$

Where $H_i=2\gamma\left(\sum_{j\neq i}L_{ij}s_j\right)$ and $x_i^{(j)}$ is the j'th coefficient of x_i.

Following the feature-sign search algorithm proposed in [18], the (11) can be solved as follows. In order to solve the non-differentiable problem, we adopt a sub-gradient strategy, which uses sub-gradients of $G(x_i)$ at non-differentiable points. Primarily we define:

$$p(x_i) = \|y_i - \Phi x_i\|^2 + \gamma L_{ii} x_i^T x_j + x_i^T H_i + \beta(1 - \sum_{i=1}^{n} x_i)^2 \quad (12)$$

Then,

$$G(x_i) = p(x_i) + \alpha \sum_{j=1}^{k} |x_i^{(j)}| \quad (13)$$

Recall that a necessary condition for a parameter vector to be a local minimum in nonsmooth optimizations is that the zero-vector is an element of the subdifferential, the set containing all subgradients at this parameter vector [14]. We define $\nabla_i^{(j)}|x_i|$ as the subdifferentiable value of the jth coefficient of x_i. If $|x_i^{(j)}| > 0$ then the absolute value function $|x_i^{(j)}|$ is differentiable, therefore $\nabla_i^{(j)}|x_i|$ is given by sign($x_i^{(j)}$). If $x_i^{(j)} = 0$ then the subdifferentiable value $\nabla_i^{(j)}|x_i|$ is set [-1,1]. So, the optimality conditions for achieving the optimal value of $G(x_i)$ is:

$$\begin{cases} \nabla_i^{(j)} p(x_i) + \alpha sign(x_i^{(j)}) & if \quad |x_i^{(j)}| > 0 \\ \left|\nabla_i^{(j)} p(x_i)\right| \le \alpha & if \quad x_i^{(j)} = 0 \end{cases} \quad (14)$$

Then, we consider how to select the optimal sub-gradient $\nabla_i^{(j)} G(x_i)$ when the optimality conditions are violated, i.e., in the case that $\left|\nabla_i^{(j)} p(x_i)\right| > \alpha$ if $x_i^{(j)} = 0$. When $x_i^{(j)} = 0$ we consider the first term in the previous expression $\nabla_i^{(j)} p(x_i)$. Suppose that $\nabla_i^{(j)} p(x_i) > \alpha$, this means that $\nabla_i^{(j)} G(x_i) > 0$ regardless the sign of $x_i^{(j)}$. In this case, in order to decrease $G(x_i)$, we will want to decrease $x_i^{(j)}$. Since $x_i^{(j)}$ starts at zero, the very first infinitesimal adjustment to $x_i^{(j)}$ will make it negative. Therefore, we can let $sign\left(x_i^{(j)}\right) = -1$. Similarly if $\nabla_i^{(j)} p(x_i) < -\alpha$ then we let $sign\left(x_i^{(j)}\right) = 1$. To update x_i suppose that we have known the signs of the $x_i^{(j)}{}'s$ at the optimal value, then we can remove the $l1$-norm on $x_i^{(j)}$ by replacing each term $|x_i^{(j)}|$ with either $x_i^{(j)}$ (if $x_i^{(j)} > 0$) or $-x_i^{(j)}$(if $x_i^{(j)} < 0$) or 0 (if $x_i^{(j)} = 0$). Thus, (13) is converted to a standard unconstrained quadratic optimization problem (QP). In this case, the problem can be solved by a linear system. The algorithmic procedure of learning affine graph regularized sparse codes is described in the following:

- for each x_i, search for signs of sign $\left(x_i^{(j)}\right)$ i = 1, … , k
- solve the reduced QP problem to get the optimal x_i^* which minimizes the objective function
- return the optimal coefficients matrix $X^* = [x_1^*, x_2^*, \dots, x_n^*]$

In the algorithm, we maintain an active set $A = \left\{j | x_i^{(j)} = 0, \left|\nabla_i^{(j)} p(x_i)\right| > \alpha\right\}$ for potentially nonzero coefficients and their corresponding signs $\theta = [\theta_1, \dots, \theta_k]$ while updating each x_i. Then, it systematically searches for the optimal active set and coefficient signs that minimize the objective function (9). In each activating step, the algorithm uses the zero-value whose violation of the optimality condition $\nabla_i^{(j)} p(x_i) > \alpha$ is the largest.

The detailed algorithmic procedure of learning affine graph regularized sparse codes is stated in Algorithm 1.

Algorithm 1: Learning Affine Graph Regularized Sparse codes

Input: Data set of n data points $Y = [y_1, \dots, y_n]$, the dictionary Φ, the graph laplacian matrix L, the parameters α, β, γ.

1- For all i such that $1 \le i \le n$ do

2- **Initializing**: $x_i = \vec{0}, \theta = \vec{0}$, and active set $A = \emptyset$,where $\theta_j \epsilon \{-1,0,1\}$denotes sign($x_i^{(j)}$).

3- **Activating:** from zero coefficient of x_i, select j=arg max$_j \left|\nabla_i^{(j)} p(x_i)\right|$. Activate $x_i^{(j)}$ (add j to the active set) only if it locally improves the objective function:

$if \quad \nabla_i^{(j)} p(x_i) > \alpha$, then set $\theta_j = -1$, $A = \{j\} \cup A$

$if \quad \nabla_i^{(j)} p(x_i) < -\alpha$, then set $\theta_j = 1$, $A = \{j\} \cup A$

4- Feature sign: let us separate Φ as some submatrix that contains only columns corresponding to the active set as $\hat{\Phi}$. Let $\hat{x}_i$ and $\hat{p}_i$ be subvectors of x_i and p.

The resulting unconstrained QP is as follows:

$$\min u(\hat{x}_i) = \left\|y_i - \hat{\Phi}\hat{x}_i\right\|^2 + \gamma L_{ii} \hat{x}_i^T \hat{x}_i + \hat{x}_i^T \hat{H}_i + \beta(1 - \sum_{i=1}^{n} \hat{x}_i)^2 + \alpha \hat{\theta}^T \hat{x}_i^T$$

Let $(\partial u(\hat{x}_i)/\partial \hat{x}_i) = 0$, the optimal value of x_i under the current active set is obtained as follows:

$$-2\hat{\Phi}^T(y_i - \hat{\Phi}\hat{x}_i) + 2\boldsymbol{\gamma} \boldsymbol{L_{ii}} \hat{x}_i + 2\boldsymbol{\gamma}\left(\sum_{\boldsymbol{j \neq i}} \boldsymbol{L_{ij}} \hat{x}_i\right) + 2\boldsymbol{\beta}(\mathbf{1} - \mathbf{1^T}\hat{\boldsymbol{x}}_i)\mathbf{1} + \boldsymbol{\alpha}\hat{\boldsymbol{\theta}} = \mathbf{0}$$

$$\hat{x}_i^{new} = \left(\hat{\Phi}^T\hat{\Phi} + \boldsymbol{\gamma}\, \boldsymbol{L_{ii}} \boldsymbol{I} + \boldsymbol{\beta}\mathbf{1}\mathbf{1}^{\boldsymbol{T}}\right)^{-1}(\hat{\Phi}^T \boldsymbol{y_i} + \boldsymbol{\beta}\mathbf{1} - \frac{\mathbf{1}}{\mathbf{2}}(\boldsymbol{\alpha}\hat{\boldsymbol{\theta}} + \hat{\boldsymbol{H}}_{\boldsymbol{i}}))$$

Where I is the identity matrix.

In the next step a discrete line search is performed on the line segment from $\hat{x}_i$ to $\hat{x}_i^{new}$ and checks the objective value at $\hat{x}_i^{new}$ and all points where the sign of any coefficient changes. Then the point with lowest objective value is replaced with $\hat{x}_i$. At last the zero coefficients of $\hat{x}_i$ are removed from the active set and update $\theta = sign(x_i)$.

5- **The optimality conditions**:

Condition (1): nonzero coefficients have the optimality condition as:

$$\nabla_i^{(j)} p(x_i) + \alpha sign\left(x_i^{(j)}\right) = 0, \forall \; x_i^{(j)} \neq 0$$

If condition(1) is not established go back to step 4
Else
Check condition(2).
Condition(2): zero coefficients have the optimality condition as:
$\left|\nabla_i^{(j)} p(x_i)\right| \leq \alpha, \forall x_i^{(j)} = 0$
If condition (2) is not established go back to step 3
Else
Return x_i as the solution.
6- End

4.3 Learning Dictionary

The learning dictionary Φ with the sparse codes X fixed is transformed to the following least square problem with quadratic constraints:

$$\min_{\Phi} \|Y - \Phi X\|_F^2, \quad s.t. \quad \|\varphi_i\|^2 \leq c \ , \forall i = 1, \dots, k \qquad (16)$$

Many methods have been proposed for solving this problem.

In this paper, we use the Lagrange dual method, which has been shown more efficient than gradient descent. The solution for this problem has been well described by prior works [6] and in this paper we do not consider it.

5. Experiments

In this section, for evaluating the proposed approach, some experiments for image classification has been performed.

5.1 Data Preparation

ORL, Yale face database are two well-known datasets widely used in computer vision and pattern recognition researches. The experiments has been done on these two datasets. In continuation we have introduced these two datasets.

- **ORL face dataset**

The ORL (Olivetti Research Laboratory) dataset contains 400 images consisting of 10 different images from 40 distinct persons [19]. The images of each person, were taken under different conditions such as times, lighting, facial expressions such as open / closed eyes, smiling / not smiling and facial details such as glasses / no glasses. The background of the whole images was homogeneous and dark. The size of each image was 92x112 pixels (Figure 3).

- **Yale face dataset**

The Yale Face Database [20] contains 165 images consisting of 11 images from 15 different persons under different conditions. The size of each image is 243×320. The conditions are consists different facial expression or configuration, center-light, with glasses, without glasses, left-light, right-light, normal, happy, sad, sleepy, surprised, and wink (Figure 4).

Fig. 3. some examples of the ORL dataset images

Fig. 4. some examples of the Yale face dataset images

5.2 Experimental Setup

For evaluation of the proposed approach, the results of this method on two defined datasets are compared with two state-of-the-art basic approaches, Sparse Coding (SC) [2] and Graph Regularized SC (GraphSC) [6] for image classification.

Each of the three methods can learn sparse representations for input data points. In particular, SC is a special case of the proposed method with $\beta = \gamma = 0$ and GraphSC is a special case with $\beta = 0$.

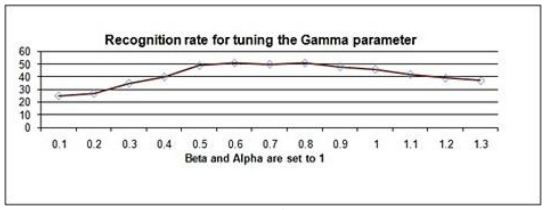
Following [6,21] the SC, GraphSC, and proposed methods are performed on data as an unsupervised dimensionality reduction procedure. For reducing a dimension of data, before applying the above algorithm, PCA is applied by keeping 98% information in the largest Eigen vectors.

Under our experimental setup, we have tuned the optimal parameters for the target classifier using leave one subject out cross validation method. Therefore, we evaluate the three baseline methods on datasets by empirically searching the parameter space for the optimal parameter settings, and report the best results of each method. For Sc and Graphsc the parameters have been set according to [11].

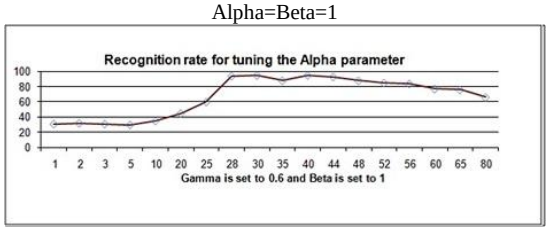
For the proposed method, we set the trade-off parameters α, β, γ through searching. In Figure 5a the plots are show the parameter value changes for ORL face dataset.

As can be seen from Figure 5, the parameters α, β, γ are set to 30, 0.1 and 0.6 respectively.

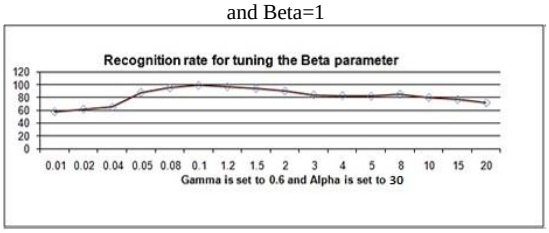
At first the value for γ parameter is achieved, for the best recognition rate assuming $\alpha, \beta = 1$. As can be seen from Figure 6a the highest recognition rate is achieved for $\gamma = 0.6$. At the next step, the value for α is achieved assuming $\gamma = 0.6$ and $\beta = 1$ for the best recognition rate. As can be seen from Figure 6b the best value for this parameter can be a number between 28 and 45. We set $\alpha = 30$ and using the same experiments the best value for β is achieved 0.1.



a) Recognition rate variations for gamma changes by setting Alpha=Beta=1



b) Recognition rate variations for alpha changes by setting Gamma=0.6 and Beta=1



c) Recognition rate variations for Beta changes by setting Gamma=0.6 and Alpha=30

Fig. 5. The parameters setting using ORL dataset

It should be noted that, the affinity constraint can be more successful when the sparsity is large enough because have the coefficients not enough sparsity, the coefficients may be selected from the hyperplane with higher dimensions than data's original dimension. In this case if the affinity constraint is added to the objective function, it may even worsen the performance with respect to the GraphSC method.

5.3 Experimental Results

For evaluating the proposed method, two experiments have been carried out. The first experiment is done on ORL face dataset for face recognition and the second one is done on Yale face dataset for face Expression recognition.

The classification accuracy of the proposed method and the two baseline methods on the two face image datasets ORL and Yale face dataset is illustrated in Figure 6 and Figure 7 respectively. As mentioned before the ORL dataset contain 40 classes of faces. Due to the lack of space in the Table only 10 classes are depicted. Among the whole dataset, classes 4 and 6, classes 8 and 10, classes 14 and 17, classes 5 and 18 are very similar to each other. Therefore, we use these classes in addition to classes 1 and 2 in the confusion matrix to show the superiority of the proposed method in classifying face datasets in Figure 6. Also in Figure 7 the results for Yale face dataset are shown. From the results we observe that the proposed method achieves much better performance than the two baseline methods.

The average classification accuracies of the proposed method on the two datasets are 91% and 63.46%, respectively for the only classes shown in the Figures. We have noticed that our proposed approach outperforms the first two baseline methods. Incorporating the graph Laplacian term of coefficients in addition to affinity constraint, leads to improve the sparse representations with more discriminating power to alleviate the classification problems.

The mean recognition rate for the selected classes are shown in Table 1. The performance improvements are 18%, 15.98% and 3.1%, 4.5% compared to baseline methods SC and GraphSC, respectively.

	Class1	Class2	Class4	Class5	Class6	Class8	Class10	Class14	Class17	Class18
Class1	92	0	2	1	0	0	2	1	2	0
Class2	0	98	0	0	0	2	0	0	0	0
Class4	1	0	65	1	22	5	3	2	0	1
Class5	2	0	2	61	0	2	0	0	13	20
Class6	0	1	28	1	68	0	0	1	0	1
Class8	0	1	0	0	0	69	14	2	8	6
Class10	1	0	0	2	0	19	69	1	7	1
Class14	2	0	3	3	2	0	3	71	15	1
Class17	1	0	0	7	3	2	7	8	69	3
Class18	1	0	0	21	4	2	1	1	2	68

a) SC recognition rate for the selected dataset from ORL

	Class1	Class2	Class4	Class5	Class6	Class8	Class10	Class14	Class17	Class18
Class1	98	0	1	0	0	0	0	0	1	0
Class2	0	98	0	0	0	1	0	0	0	1
Class4	0	0	89	1	5	1	1	1	1	1
Class5	1	0	2	91	0	0	0	0	2	4
Class6	0	0	3	0	91	0	2	2	1	1
Class8	0	2	0	0	1	87	8	0	2	0
Class10	0	0	1	0	0	8	79	4	5	3
Class14	0	0	1	0	0	0	0	83	13	3
Class17	0	0	0	1	2	1	8	11	75	2
Class18	1	0	0	6	1	0	0	0	4	88

b) GraphSC recognition rate for the selected dataset from ORL

	Class1	Class2	Class4	Class5	Class6	Class8	Class10	Class14	Class17	Class18
Class1	99	0	1	0	0	0	0	0	0	0
Class2	0	100	0	0	0	0	0	0	0	0
Class4	1	0	95	1	2	0	0	0	1	0
Class5	0	0	2	96	0	0	0	1	1	0
Class6	0	0	2	1	95	0	1	0	0	1
Class8	0	0	0	0	1	92	4	1	2	0
Class10	0	0	0	0	0	7	84	3	5	1
Class14	0	0	0	0	0	0	2	79	14	5
Class17	0	0	0	1	2	1	5	11	78	2
Class18	0	0	0	1	1	0	2	4	0	92

c) The recognition rate for the selected dataset from ORL for the proposed method

Fig. 6. The image confusion matrix for ORL data between three methods Sc, GraphSc and the proposed method

	center-light	With glasses	happy	left-light	No glasses	normal	right-light	sad	sleepy	surprised	Wink
center-light	23.6	7.8	4.2	6	21.2	21.8	4.2	4.2	0	0.6	6
w/glasses	6	76.6	3	4.2	0	4.2	3	0	1.8	1.2	0
happy	7.8	4.2	65.2	6	7.8	4.2	0	1.2	1.8	0	1.8
left-light	1.8	3	4.2	45.1	15.7	21.2	1.8	1.2	0.6	3	2.4
w/no glasses	23.6	3	3	7.2	25.1	23.6	3	4.8	4.2	1.8	0.6
normal	23.3	0	4.2	10.9	21.8	23.6	4.8	1.2	3	3	4.2
right-light	6	1.8	6	4.8	3	1.2	57.2	6	3	6.8	4.2
sad	0	0	1.8	6	0.6	0	7.2	35.6	30.8	12	6
sleepy	5.4	0	1.8	3	1.8	1.2	6.8	30	39.8	6	4.2
surprised	1.8	1.2	4.2	4.2	0.6	0	6	12	6	60.4	3.6
wink	0	1.8	3	1.8	1.2	1.2	6	9	4.8	1.2	70

a) SC recognition rate for Yale face dataset

	center-light	With glasses	happy	left-light	No glasses	normal	right-light	sad	sleepy	surprised	wink
center-light	38	7.8	4.2	6	18	18.8	1.8	3	0	0	2.4
With glasses	6	82.6	1.8	3	1.2	1.8	1.2	0	1.8	0.6	0
happy	7.8	3	74.8	3	4.2	1.2	0	1.2	0	0	4.8
left-light	1.8	1.2	3	52.5	12	17.5	1.8	2.4	0.6	3	4.2
No glasses	20	1.8	1.8	7.4	46.1	13.3	1.8	4.2	1.8	1.2	0.6
normal	18	0	1.2	6	12.6	43	6	1.2	6	3	3
right-light	6	1.2	6	4.8	1.8	1.2	68.8	4.2	1.2	3	1.8
sad	0	1.2	1.8	6	1.2	1.8	6	54.2	20	6	1.8
sleepy	6	0	0	3	1.2	1.2	4.8	29	53.6	1.2	0
surprised	2.4	0.6	3	4.8	0	1.8	4.2	12	6.8	64.4	0
wink	0	1.2	1.8	3.6	0.6	1.2	3.6	4.8	2.4	10.3	70.5

b) GraphSC recognition rate for Yale face dataset

	center-light	w/glasses	happy	left-light	w/no glasses	normal	right-light	sad	sleepy	surprised	wink
center-light	47.4	6.8	4.2	6	15.7	16.9	1.8	3	0	0	1.2
w/glasses	6	83.4	1.8	1.8	1.2	0	0.6	1.2	1.8	1.2	0
Happy	6.8	1.8	78.2	3	4.2	1.2	0	1.2	0.6	0	3
left-light	1.8	1.2	2.4	60.3	12	13.9	1.8	1.2	0.6	1.8	3
w/no glasses	13.9	1.8	1.8	7.2	42.9	12	7.2	5.4	3	3	1.8
Normal	15	0	0.6	6	9.6	49.6	6	1.2	6	1.8	4.2
right-light	4.2	1.2	4.2	4.8	2.4	0.6	72.4	4.2	1.8	3	1.2
Sad	0	1.2	1.8	6	3	1.8	4.2	57.7	13.9	5.4	3
sleepy	4.8	1.2	0	3	1.8	1.2	1.2	17.5	68.1	1.2	0
surprised	1.8	1.2	3	4.8	2.4	1.2	1.8	6	8.4	68.8	0.6
Wink	0	1.2	1.2	3	3	0	0.6	0	4.8	11.5	74.3

c) Recognition rate for Yale face dataset for the proposed method

Fig. 7. The image confusion matrix for Yale face dataset between three methods Sc, GraphSc and the proposed method

Table 1. Mean Recognition rate for SC, GraphSc and the proposed methods

	Mean Recognition rate for ORL data	Mean Recognition rate for Yale data
SC method	**73%**	**47.48%**
GraphSc method	**87.9%**	**58.96%**
DLPV method [8]	**97.6%**	**-----**
Proposed method	**98.8%**	**63.46%**

If we add the ORL absent classes, the recognition rate is raised up to 98.8%. For better evaluating the proposed method, the recognition rate for all classes of ORL dataset is compared with a recent non-sparse based method in face recognition application at 2015 [8]. The authors in this paper have proposed a face recognition method based on the discriminative locality preserving vectors (DLPV). The best result in this paper for the ORL dataset is reported 97.6%. With comparing these two methods the superiority of the proposed algorithm becomes clear.

5.4 Experiment on action dataset

For more evaluation the proposed sparse coding method, this method is applied on KTH action dataset. The KTH dataset [22], contains six actions including walking, jogging, running, boxing, hand waving and hand clapping, performed several times by 25 subjects in four different scenarios. Overall, it contains 2391 sequences. Some samples of KTH dataset are shown in Figure 8.

Fig. 8. Some samples of KTH dataset

At first the dataset has been divided into two 50% portions as train and test data randomly for each class. Then the HOG3D descriptor is extracted from data. For reducing the dimension, PCA is applied with keeping 98% of information. Then the proposed sparse coding method is applied for feature extraction from the descriptor. We use SVM method for classifying the data.

The results show that the proposed method could classify the KTH data with about 94% precision. This result show that the proposed method can be applied in action recognition applications same as to the face recognition.

6. Conclusion and future work

In this paper, a novel approach for robust face recognition namely affine graph regularized sparse coding has been proposed. In the proposed method, the well-defined graph regularized sparse coding method has been improved by adding the affinity constraint. Using this term, until the sparsity is big enough the manifold structure of features is better preserved. The results indicate that the proposed method with comparison to some other approaches has the better performance for face recognition. In addition the results show that the proposed method could be applied for human action recognition datasets as well. In future works we would apply the proposed method for action recognition artificial and real world datasets for evaluating the proposed method.

References

[1] M. Long, G. Ding, J. Wang, J. Sun, Y. Guo, and P. S. Yu, "Transfer Sparse Coding for Robust Image Representation", IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2013.

[2] J.Zhang, D. Zhao, W. Gao, "Group-based Sparse Representation for Image Restoration", IEEE Transactions on Image Processing, vol. 23, no. 8, pp. 3336 – 3351, 2014.

[3] H. Lee, A. Battle, R. Raina, and A. Y. Ng., "Efficient sparse coding algorithms", In Advances in Neural Information Processing Systems, 2006, pp. 801-808.

[4] Y. Cheng, Z. Jin, T. Gao, H. Chen and N. Kasabov, "An Improved Collaborative Representation based Classification with Regularized Least Square (CRC-RLS) Method for Robust Face Recognition", Neuro computing, vol. 215, no. c, pp. 250-259, 2016.

[5] Y. Censor and S. Zenios, "Parallel Optimization: Theory, Algorithms, and Applications," 1st ed., New York: Oxford Univ. Press, 1997.

[6] M. Zheng, J. Bu, C. Chen, C. Wang, L. Zhang, G. Qiu, and D. Cai, "Graph regularized sparse coding for image representation", IEEE Transactions on Image Processing, Vol. 20, No.5, pp. 1327-1336, 2011.

[7] Y. N. Liu, F. Wu, Z. H. Zhang, Y. T. Zhuang, and S. C. Yan, "Sparse representation using nonnegative curds and whey", In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR, 2010.

[8] Y. Wen, L. Zhang, K. M. von Deneen, L. He, "Face recognition using discriminative locality preserving vectors", Digital Signal Processing, Vol. 50, pp. 103-113, March 2016.

[9] M. Yang, L. Zhang, J. Yang, and D. Zhang. "Robust sparse coding for face recognition", In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR, 2011.

[10] Z. Lu, Y. Peng, "Latent semantic learning with structured sparse representation for human action recognition", Pattern Recognition, vol. 46, no. 7, pp. 1799-1809, July 2013.

[11] M.Zheng, J.Bu, C.Chen, C.Wang, L.Zhang, G.Qiu and D.Cai, "Graph Regularized Sparse Coding for Image Representation", Journal of Latex Class Files, vol. 6, no. 1, Jan 2007.

[12] B. Quanz, J. Huan, and M. Mishra, "Knowledge transfer with low-quality data: A feature extraction issue", IEEE Transactions on Knowledge and Data Engineering, vol. 24, no.10, pp. 1789-1802, October 2012.

[13] Hazewinkel, Michiel, ed. (2001), "Affine transformation, Encyclopedia of Mathematics" , Springer, ISBN 978-1-55608-010-4.

[14] R. Fletcher, "Practical methods of optimization", 2nd ed., Wiley-Interscience, New York, 1987.

[15] M. Aharon, M. Elad, A. Bruckstein, and Y. Katz, "K-svd: An algorithm for designing overcomplete dictionaries for sparse representation", IEEE Transactions on Signal Processing, vol.57, no.11, pp. 4311-4322, October 2006.

[16] M. Belkin and P. Niyogi, "Laplacian eigenmaps and spectral techniques for embedding and clustering", In Advances in Neural Information Processing Systems 14, NIPS, 2001, pp. 585-591.

[17] X.Lu, Y. Yuan and P. Yan, "Alternatively Constrained Dictionary Learning for Image Super resolution", IEEE Transactions On Cybernetics, vol. 44, no. 3, March 2014.

[18] E. Cand`es and T. Tao, "Near-optimal signal recovery from random projections: niversal encoding strategies?", IEEE transactions on information theory, vol. 52, no. 12, pp. 5406–5425, 2006.

[19] F. Samaria, A. Harter, "Parameterisation of a Stochastic Model for Human Face Identification", Proceedings of 2nd IEEE Workshop on Applications of Computer Vision, Sarasota FL, 1994.

[20] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman, "Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection", IEEE Transaction on PAMI, vol. 19, no. 7, pp. 711-720, July 1997.

[21] S. J. Pan, I.W. Tsang, J. T. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis", IEEE Transactions on Neural Networks, vol.22, no.2, pp.199–210, Feb 2011.

[22] X. Wu, D. Xu, L. Duan, and J. Luo, "Action recognition using context and appearance distribution features", in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '11), 2011, pp. 489–496.

Mohsen Nikpour received the B.Sc And M.Sc degrees in electrical engineering from Mazandaran University, Iran, in 2004 and 2008 respectively and Know he is a Ph.D Student in electrical engineering in Babol noushirvani University, Islamic Republic of Iran. His current research interests include Image processing, video processing, video coding, image segmentation and image denoising.

Mohammad Reza Karami-Mollaei received the B.Sc in Electrical and Electronic Engineering in 1992, M.Sc of signal processing in 1994, and Ph.D in 1998 in Biomedical Engineering from I.N.P.L d'Nancy of France. He is now an associate professor with the Department of Electrical and Computer Engineering, Babol University of Technology. Since 1998 his research is in signal and speech processing. He published more than 100 articles in journals and conferences. His research interests include Speech, Image and signal processing.

Reza Ghaderi received B.Sc in 1989 from Ferdoosi University of Mashhad, IRAN, M.Sc in 1991 from Tarbiat Modaress University, IRAN and Ph.D in 2001 from Surrey University UK all in Electronic Engineering. Currently he is an associate prof. at Nuclear Eng. Dept. of Shahid Beheshti Univ., Tehran, Iran. His research interests are neural networks, pattern recognition, system modeling, signal processing, Fuzzy logic, artificial intelligent.

Towards Accelerating IP Lookups on Commodity PC Routers using Bloom Filter: Proposal of Bloom-Bird

Bahram Bahrambeigy*
Department of Datacenter, Pishgaman Tose Ertebatat (PTE) Inc., Tehran, Iran
b.bahrambeigy@pishgaman.net
Mahmood Ahmadi
Department of Computer Engineering, Razi University, Kermanshah, Iran
m.ahmadi@razi.ac.ir
Mahmood Fazlali
Department of Computer Science, Shahid Beheshti University (SBU), GC, Tehran, Iran
fazlali@sbu.ac.ir

Received: 21/May/2015 Revised: 16/Nov/2016 Accepted: 16/Dec/2016

Abstract

Nowadays, routers are the main backbone of computer networks specifically the Internet. Moreover, the need for high-performance and high-speed routers has become a fundamental issue due to significant growth of information exchange through the Internet and intranets. On the other hand, flexibility and configurability behind the open-source routers has extended their usage via the networks. Furthermore, after assigning the last remaining IPv4 address block in 2011, development and improvement of IPv6-enabled routers especially the open-sources has become one of the first priorities for network programmers and researchers. In IPv6 because of its 128-bits address space compared to 32-bits in IPv4, much more space and time are required to be stored and searched that might cause a speed bottleneck in lookup of routing tables. Therefore, in this paper, Bird as an example of existing open source router which supports both IPv4 and IPv6 addresses is selected and Bloom-Bird (our improved version of Bird) is proposed which uses an extra stage for its IP lookups using Bloom filter to accelerate IP lookup mechanism. Based on the best of our knowledge this is the first application of Bloom filter on Bird software router. Moreover, false positive errors are handled in an acceptable rate because Bloom-Bird scales its Bloom filter capacity. The Bloom-Bird using real-world IP prefixes and huge number of inserted prefixes into its internal FIB (Forwarding Information Base), shows up to 61% and 56% speedup for IPv4 and IPv6 lookups over standard Bird, respectively. Moreover, using manually generated prefix sets in the best case, up to 93% speedup is gained.

Keywords: Bird; Bloom Filter; Forwarding Information Base; IPv4; IPv6; Open Source Routers.

1. Introduction

The need for high-performance and high-speed routers has become a fundamental issue due to significant growth of information exchange through Internet and intranets. Due to adoption of Class-less Inter-domain Routing (CIDR) method, routers need to find best match between various prefix lengths that may differ from lengths 1 to 128 based on what version of IP and what prefix is used. This process of finding matching IPs is time consuming and a lot of hardware (e.g. TCAM and SRAM) and algorithmic approaches (e.g. binary searches) are proposed in the literature as will be discussed further in the related work section.

On the other hand, modern IP router solutions can be classified into three main categories, hardware routers, software routers, and programmable network processors (NPs) [1]. PC-based software routers have created reasonable networking platforms with easy development and programmability features. These features are the most important in comparison with hardware routers. Current software routers reported forwarding up to 40 Gbits/sec traffic on a single commodity personal computer [2]. On the other hand, existence of open-source software routers has brought the opportunity to study and change their codes to make the better routers based on researchers needs. Bird [3], Quagga [4], and Xorp [5] are examples of such open-source routers. Among them, the Bird is selected to implement a Bloom filter (BF) [6] on its internal FIB (Forwarding Information Base) in which all routing tables are based on this data-structure. Since searching in a FIB which stores a huge number of IP prefixes can cause speed bottleneck, we have accelerated it using an extra stage lookup on Bloom filter data-structure. Results show that BF as an extra stage on Bird IP lookups (i.e. Bloom-Bird) makes it up to 93% faster than its standard hashing mechanism for searching big FIBs in the best case. Also results indicate that there is even speedup when result of searching a particular prefix in the FIB is positive because of hash optimizations made in the Bloom-Bird.

The main concern in this paper is to show how a Bloom filter can help an open-source software router to speedup searches when number of inserted nodes and

* Corresponding Author

prefixes becomes huge. In order to have a fair comparison between two versions of Bird (i.e. standard Bird and Bloom-Bird), basic rules and structures of standard Bird is not changed. For example maximum length of Bird's main hash is 16-bits, so it is the same for Bloom-Bird too. The Bloom-Bird includes a Bloom filter array (thus a space overhead) to speedup simple searches for a given IP and length and also Longest Prefix Matching (LPM) lookups. The array can scale its capacity; therefore, False Positive (FP) errors are handled in an acceptable rate.

The main contribution of the paper is proposal of Bloom-Bird router to enhance the performance of Bird open-source router that utilizes a Bloom filter for both IPv4 and IPv6 addresses in its architecture.

The rest of the paper is organized as follows. Section 2 presents related work of Bloom filter and its applications in network processing. Section 3, contains two subsections, which first, is a brief introduction to FIB data structure of Bird and how Bloom-Bird is implemented conceptually, and in the latter subsection the pseudo-codes of implemented approach is presented. Section 4 contains four subsections, which the first two sub-sections are dedicated to IPv4 prefix sets and the last two sub-sections are based on IPv6 prefix sets. Therefore, in Section 4, the first sub-section presents an introduction of the scenario in order to evaluate Bloom-Bird and standard Bird using IPv4 prefixes, and in the second subsection, results of Bloom-Bird evaluation are presented; third and fourth sub-sections are similar to previous sub-sections but they are based on IPv6 prefix sets. Finally, section 5 concludes the paper.

2. Related Work

Bloom filter (BF) is a randomized and probabilistic data-structure proposed by Burton Bloom in the 1970s [6]. BF normally consists of a bit-array representing existence of inserted elements. By checking k hash functions and getting negative answer, it can be determined that the element is not inserted certainly. However some FP (False Positive) may occur. Which means BF may express some elements exist by mistake so it needs to check main hash table to make sure about positive answers. Bit-array of BF can reside in an on-chip memory by a hardware implementation to do k hash functions checks in parallel. The main advantage of BF structure is Space and Time efficiency in which consumes much less space than ordinary data structures because of its potential collisions and requires much less and more predictable time to query a member.

A lot of variants of BFs are proposed such that more than 20 variants of BF are presented in the literature [7]. Each and every one of them is used for a special manner. For example, standard Bloom filter (SBF) is used in order to check if a specific element is present or not. An important draw-back of SBF is that insertions cannot be undone. Counting Bloom filter (CBF) [8] and later, Deletable BF (DlBF) [9] proposed in order to gain the removability in BF. In CBF each bit in the Bloom array is replaced by a counter. Each insertion, increments counters related to k hash functions. Obviously, each deletion decrements related counters. Another variant of BF which supports deletions as mentioned is DlBF. It splits BF array into multiple regions and tracks regions of BF array in which collisions occur. A small fraction of bit-array is used in order to determine related area is collision-free or not. If bits are located in a collision-free region, then the bit can be reset safely, otherwise it will not be safe to delete. Therefore, some bits may not reset if they are located in a collisionary region. CBF is selected for Bloom-Bird because of its simplicity and consistency over deletions instead of DlBF. DlBF would be a good option if BF is going to be implemented in an on-chip memory.

BFs are used in various applications including network processing as discussed in [10] that can be classified into four major categories: Resource routing, Packet routing, Measurement, and Collaborating in overlay and peer-to-peer networks. Moreover, "IP Route lookup" and "Packet Classification" are important applications of BF in the network processing (e.g. [11]). Longest Prefix matching (LPM) or Best Matching Prefix (BMP) that can be classified into IP route lookup category is also an interesting area of BF application. There are a lot of proposed algorithms in order to speedup BMP in the literature. In [12] authors have classified BMP algorithms into "Trie-based algorithms", "Binary search on prefix values", and "Binary search on prefix lengths". "A Trie is a tree-based data-structure allowing organization of prefixes on a digital basis using the bits of prefixes to direct the branching" [12]. Trie-based schemes do a linear search on prefix length because they only compare one bit at a time. The worst case of memory accesses is W when prefix length is W. However, binary search algorithms on prefix values are proportional to log_2N which N is number of prefixes. Binary search on prefix lengths are proportional to log_2W. The first BF application for LPM proposed in [13] which parallel check on on-chip memory is performed to accelerate lookups before checking slower off-chip memory.

Although employing Bloom filters as an extra stage to accelerate IP lookups is well studied by Dharmapurikar et al. [13], Bloom-Bird is different because it is completely independent of specialized hardware implementation and it runs on commodity PC hardware. Therefore, number of hashes is kept as low as possible and the hash probes can be run sequentially and BF can reside in slow memory without loss of efficiency. Moreover, BF on Bloom-Bird helps to accelerate (prefix, length) pair searches in the FIB data structure by ignoring long chains of linked lists of the main hash table. In order to have a fair comparison, basic rules and chain orders of Bird are not modified and modifications are as low as possible.

Furthermore, because IPv6 uses more bits to represent IP addresses (128-bit) compared to IPv4 (32-bit), therefore, it is expected that number of IPv6 prefixes

becomes much bigger than current IPv4 prefixes in a near future. Therefore, trie-based schemes will become inefficient. Therefore, multi-bit tries [14] are proposed which compare more than one bit at a time at the cost of space overhead. However, whether Binary search or trie-based lookups is used, it is shown that Bloom filters can accelerate IPv6 lookups too [15], [16]. Nevertheless, the main advantages of our work over the two aforementioned approaches accelerating IPv6 lookups using Bloom filters is that they are based on a special hardware implementation but Bloom-Bird runs completely on commodity PC hardware. To assert again, we didn't change the main hash of standard Bird into better lookup approaches like trie-based schemes or binary searches, in order to have a fair comparison between standard Bird and Bloom-Bird.

The following section explains Bird open source router fundamental data-structures, pseudo-codes of them and how are they are improved in Bloom-Bird.

3. Bloom-Bird: A Better Bird

"The Bird project aims to develop a fully functional dynamic IP routing daemon primarily targeted on (but not limited to) Linux, FreeBSD and other UNIX-like systems" [3]. It supports latest versions of routing protocols such as BGP, RIP and OSPF. It also supports both IPv4 and IPv6 addresses and a simple command line interface to configure the router. There is a fundamental data-structure called FIB (Forwarding Information Base) in the Bird which routing tables are based on it. This data-structure stores IP prefixes and length of them. Searching in a FIB, where huge number of prefixes is stored can become a speed bottleneck, which can be faster using a CBF (Counting Bloom Filter) as will be presented. Storing in FIB of Bird router is a two stage mechanism. In the first stage, an *order*-bit hash is calculated based on prefix value to find bucket index of main hash table (*order* can be varied from 10 to 16). In the next stage, there is a chain of nodes in linked list structure which may become long due to huge number of nodes. Therefore, a BF can help to reduce of traversing these long chains for missing nodes which results in accelerating the IP lookup mechanism. Nodes are allocated in each chain by Bird Slab Allocator. The implementation of the memory allocator is based on what Bonwick proposed [17] that makes linked list traversing reasonably fast.

To be more specific, there are three important functions related to FIBs in the Bird named fib_get(), fib_find(), fib_route() which are responsible for adding, searching and longest prefix matching, respectively. Each FIB structure in Bird starts with a default 10-bit hash order (i.e. 2^{10}) and increases its hash table size when the number of prefixes increases. This expansion will stop at 16-bit; therefore, chain lengths start to become larger. Implemented BF helps the main hash table when this situation happens to prevent searching in this large FIB chains when an IP prefix cannot be found.

In the following subsection, the way Bloom-Bird implemented is presented conceptually. In the next subsection, the pseudo-codes of implemented approach are discussed.

3.1 Implementation Concepts

Bird uses dynamic hashing size to store prefixes which increases when number of inserted prefixes becomes huge. It starts from 10-bit and it expands until 16-bit order and never grows afterwards. It increments the order by 2 when the capacity limit is reached (e.g. it expands into 12-bit when 10-bit limit is reached). In order to have a fair comparison between standard Bird and Bloom-Bird, these rules are not changed. Therefore, Bloom-Bird includes an extra BF array in each FIB to help it responding faster when it is possible.

Hashing mechanism in the BF of Bloom-Bird is inspired by the main hash table of the standard Bird. If number of entries increases, Bloom-Bird changes size and order of BF array similar to the main hash table approach. Bloom-Bird starts with 18-bit order and it increases to 20-bit order if the capacity limit is reached. This expansion continues until 32-bit and it never grows afterwards. For simplicity, this expansion of BF array is not included in the pseudo-codes in the next subsection. Because of 32-bit order limit, capacity of BF array is limited to 32-bit when an acceptable FP error rate is expected. As the results will show, Bloom-Bird shows at most 15% FP error rate which is fairly good based on Eq. "(1)" in section 5 and tested elements.

In "Fig. 1" a simple Bird's FIB hashing table is depicted. In the aforementioned Figure, the order can be varied from 10 to 16 as mentioned before. Basic fib_find() function that searches for a given prefix and length in a FIB is shown which uses ipa_hash() function to determine which bucket in main hash table should be used. Main hash array is an *order*-bit array of fib_node type. Afterwards, the node will be inserted into a new free location in the linked lists chain.

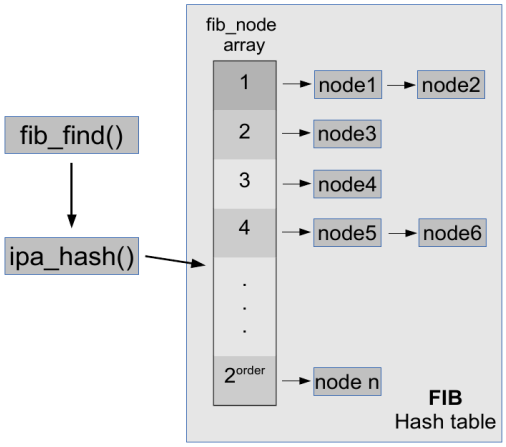


Fig. 1. FIB hashing architecture of standard Bird [3].

In "Fig. 2" the way that BF is implemented in the FIB is depicted. The fib_find() function checks BF for given prefix firstly. If BF confirms the existence of the prefix,

then the main hash table will be checked in order to determine pointer address of found node or a FP error may occur. On the other hand, (and more importantly) if BF returns negative answer, checking main hash table will be ignored. Therefore, the main advantage of BF is the latter part in which checking main hash table and maybe traversing long chains of linked lists can be avoided.

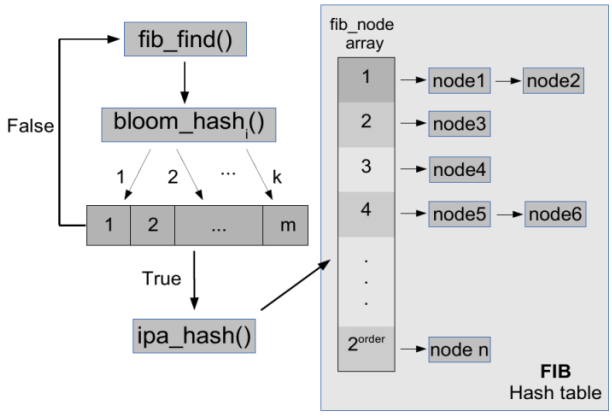


Fig. 2. FIB hashing architecture of Bloom-Bird.

3.2 Implementation Codes

The pseudo-code of fib_find() in the standard Bird (as discussed), is shown as SB_fib_find() function (in order to distinguish between standard Bird and Bloom-Bird functions, SB_ is prepended to Bird functions and BB_ is prepended to Bloom-Bird functions). In the first line, *e* variable points to the selected bucket which is traversed in order to find given prefix. In the second and third lines, the bucket chain is traversed to find the requested node. Two situations may happen after this loop. The loop may find the node, and then last line returns the node. Otherwise, traversing linked list may end with a null pointer; therefore, fourth line returns a null pointer indicating that the node cannot be found.

Psuedo-Code1. SB_fib_find()

```
SB_fib_find(fib, prefix, length)
1.  e = fib_table[ipa_hash(prefix)]
2.  while((not empty e) AND (not found e))
3.    e = e->next
4.  return e
```

In the Bloom-Bird version of this function, in the first line, BF array and its hashing mechanism is used in order to ignore prefixes that do not exist as discussed earlier. The function is changed as BB_fib_find(). Three first lines are dedicated to BF search method for a given prefix. There are *k* hash probes in order to search BF. In each step of the loop, if a location of the BF array represents an empty location then the search returns false answer immediately (i.e. NULL pointer). As it is shown in second line, *k* independent hash functions are used for BF to check the array locations. These hash functions are also depicted in "Fig. 2" *as bloom_hashi()*.

Psuedo-Code 2. BB_fib_find()

```
BB_fib_find(fib, prefix, length)
1.  for(i=1 to k)
2.    if(filter[bloom_hash_i(prefix)]        is
    empty)
3.      return NULL
4.  e = fib_table[hash(prefix)]
5.  while((not empty e) AND (not found e))
6.    e = e->next
7.  return e
```

Bird uses very simple longest prefix matching (LPM) mechanism that starts from a given length and decrements it until longest prefix match is found or returns a NULL pointer. The pseudo-code is shown as SB_fib_route() function.

Since fib_route() uses fib_find() as its main function to determine existence of the prefixes, BF can help fib_route() very effectively because BF causes no false negative errors and it does not need to go through the main hash chains for lengths that cannot be found. Therefore, there is no need to change anything in the fib_route() function and BF helps LPM indirectly. Although there are better solutions like binary searches on prefix values and lengths as mentioned in related work section, Bird's LPM algorithm is not changed in order to show BF performance over standard Bird.

Psuedo-Code 3. SB_fib_route()

```
SB_fib_route(fib, prefix, length)
1.  while (length ≥ 0)
2.    if(fib_find(fib, prefix, length))
3.      return found node
4.    else
5.      length = length – 1
6.  return NULL
```

The last important function is fib_get() which searches for given prefix and length and if does not exist, it adds the prefix into the FIB. The pseudo-code of this function is presented as SB_fib_get() function. It is shown in the first line that the fib_get() uses fib_find() function as its searching mechanism. If it finds the node then the found node pointer will be returned. Otherwise the node will be inserted into selected bucket of the hash table.

Psuedo-Code 4. SB_fib_get()

```
SB_fib_get(fib, prefix, length)
1.  if(fib_find(fib, prefix, length))
2.    return found node
3.  else
4.    Go down through hash chains
5.    And add new node
```

To check existence of nodes in this function also BF can help through fib_find() when a node does not exist. Therefore, in the first line, BF is checked before its main hash table and when a node is not inserted before, BF counters should be incremented (because CBF is used). Therefore, only one change is needed in this function. This function is been shown as BB_fib_get() function.

Psuedo-Code 5. BB_fib_get()

```
BB_fib_get(fib, prefix, length)
1.  if(fib_find(fib, prefix, length))
2.    return found node
3.  else
4.    Go down through hash chains
5.    And add new node
6.    for(i=1 to k)
7.      filter[bloom_hash_i(prefix)] += 1
```

Extra lines 6 and 7 of BB_fib_get() function are responsible for updating BF array due to newly added node. This does not count as overhead since hash functions are optimized using bit-wise operations and simplifications compared to standard hash of Bird.

There are two different types of hash functions used in the Bloom-Bird. First type is much like Bird's original hash function which returns a 16-bit hash based on prefix value but optimized using bit-wise operations (ipa_hash() function). Second type hash functions are used for BF which has much less collisions than Bird's original hash function. These second type hash functions return a variety of bit sizes based on BF array length. Number of BF hash functions (*k*) is set to the lowest possible value. Two possible minimum values of k has been tested (i.e. *k=3* and *k=2*). In which *k=3* tests showed a little speed overhead compared to *k=2* tests while the FP error rate was almost the same. Therefore, the *k=2* value is selected for Bloom-Bird to compare its performance with standard Bird.

Therefore, in the Bloom-Bird, *k* is constant and is set to 2 because the loop of checking *k* hash functions becomes speed bottleneck for bigger *k*.

In next Section the scenario and prefix sets in order to compare Bloom-Bird and standard Bird and evaluation are presented and discussed.

4. Evaluation of Bloom-Bird and Results

4.1 IPv4 Scenario

In order to evaluate standard Bird and Bloom-Bird three real IPv4 prefix sets from [18] are gathered from years 2008, 2010 and 2013 sorted by date which latest and more updated one contains more than 482 thousands unique IPv4 prefixes as "Table 1" shows.

Table 1. IPv4 Prefix sets to test the two versions of Bird.

Prefix set alias	# of nodes
Prefix1	262,039
Prefix2	351,645
Prefix3	482,500
Prefix4	1,179,648
Prefix5	1,310,720
Prefix6	2,490,368

The two versions of Bird i.e. standard Bird and Bloom-Bird are evaluated by inserting these real prefix sets and querying them. Prefix sets 4 and 5 are manually generated which contains all possible 24 length prefixes starting with 1-19 and 20-39 octets respectively. Prefix set 6 is concatenation of two prefix sets 4 and 5 in order to test searching FIBs with even bigger prefix sets and make sure about the results. These last three prefix sets contain 99% missing (not existing) prefixes compared to the other three real prefixes (i.e. prefix sets 1-3) in order to show performance of BF when most queries return negative answer (best case). These last three prefix sets are not real prefix traces; therefore, they are only used for searching, not for inserting into FIBs.

Percentage of number of missing nodes when each prefix set is searched is presented in "Table 2". For example when all prefixes in prefix set 2 are inserted into a FIB and all prefixes in the prefix set 1 are queried afterwards, 24.53% of searches return negative answer.

Table 2. Percentage of missing nodes when searching for IPv4 prefix sets

Inserted prefix set / Searched prefix set	Prefix1	Prefix2	Prefix3
Prefix1	0	24.53%	36.27%
Prefix2	43.65%	0	22.92%
Prefix3	65.34%	43.82%	0
Prefix4	99.8%	99.77%	99.56%
Prefix5	99.86%	99.77%	99.39%
Prefix6	99.83%	99.77%	99.47%

There are 0 values in the above Table because the same prefix set is inserted and searched. Results of evaluation are included and discussed in the following subsection.

4.2 IPv4 Results of Bloom-Bird and Discussion

As discussed in the previous subsection, three prefix sets 1-3 are inserted into a FIB at three different times in two versions of standard Bird and Bloom-Bird and all prefix sets 1-6 are queried afterwards. Percentages of speedups of Bloom-Bird (fib_find() and fib_route() functions) over standard Bird and FP error rate are presented in "Tables 3, 4 and 5" respectively. These results are gained on a home PC with 2.88 MHz dual core CPU, 6 MB cache and 4 GB RAM which runs unmodified (vanilla) Linux kernel 3.12.

Table 3. IPv4 Speedups of Bloom-Bird fib_find() over standard Bird - Simple Search Function (*) means the same prefix set is inserted

Inserted prefix set / Searched prefix set	Prefix1	Prefix2	Prefix3
Prefix1	(*) 20%	45%	55%
Prefix2	53%	(*) 17%	47%
Prefix3	61%	58%	(*) 28%
Prefix4	81%	93%	91%
Prefix5	82%	91%	91%
Prefix6	81%	93%	90%

Table 4. IPv4 Speedups Bloom-Bird of fib_route() over standard Bird - LPM Search Function (*) means the same prefix set is inserted

Inserted prefix set / Searched prefix set	Prefix1	Prefix2	Prefix3
Prefix1	(*) 14%	33%	41%
Prefix2	26%	(*) 26%	36%
Prefix3	33%	43%	(*) 32%
Prefix4	41%	62%	64%
Prefix5	44%	63%	63%
Prefix6	42%	63%	64%

Table 5. IPv4 False Positive Percentage of Bloom-Bird
(*) means the same prefix set is inserted

Inserted prefix set / Searched prefix set	Prefix1	Prefix2	Prefix3
Prefix1	(*) 0	1.04%	2.14%
Prefix2	5.46%	(*) 0	1.41%
Prefix3	7.58%	1.25%	(*) 0
Prefix4	8.69%	0.86%	1.66%
Prefix5	9.49%	1.07%	2.3%
Prefix6	9.11%	0.97%	1.88%

In the "Table 3", speedups of fib_find() function which is responsible for simple searching for a given prefix and length is presented. In the "Table 4", speedups of fib_route() function which is responsible for Longest Prefix Matching (LPM) is presented (starting length for LPM is set to 32 for all searches). In the "Table 5", percentage of FP error rate is presented. Also there are 6 rows in the all aforementioned tables, representing what prefix set is searched. The smallest speedup is 14% and biggest speedup is 93%. Smaller speedups are gained when most prefixes are found after search (i.e. number of existing nodes are bigger than missing nodes). On the other hand, bigger speedups are gained when most prefixes are not found after search.

It is well known that FP error can be estimated using following equation [13]:

$$fpr = \left[1 - (1 - \frac{1}{m})^{k*n}\right]^k \quad (1)$$

In which *k*, *m* and *n* represent number of hash functions, size of array, and number of inserted elements, respectively. It gives a nearly accurate estimation and it is used in this paper to evaluate resulted FP errors. The optimal number of hashes (*k*) can be calculated using following equation:

$$k = \frac{m}{n} \ln 2 \quad (2)$$

Although Eq. "(2)" gives us optimal number of hashes (*k*) but we need to keep it at its lowest possible value in the Bloom-Bird. Because the higher the *k* value becomes, the more overhead is caused. That is because of sequential execution of BF array probes in the Bloom-Bird. Therefore, as mentioned before, the number of hashes (*k*) is set constant number equal to 2.

In order to guarantee its FP rate and performance, the Bloom-Bird calculates its BF array size based on following equation:

$$m = 2^{(n+2)} \quad (3)$$

Therefore, for its default 18-bits order (maximum number of inserted elements can be up to $n=2^{18}$), size of BF array can be calculated based on Eq. "(3)" which leads to $m=2^{20}$. Consequently, given these values of *m,n* and *k=2*, Eq. "(1)" results 15% FP error rate. Experimental results also show the expected value even in lower rates; As "Table 5" shows, the most FP error rate is 9.49 percent. Therefore, Bloom-Bird handles its FP error rate even better than expected.

The practical FP rate of Bloom-Bird is calculated based on the following equation:

$$fpr = \frac{Observed\ false\ positives}{Tested\ prefixes} \quad (4)$$

Based on the experiments, for small number of prefixes, BF counts only as a memory overhead on Bird i.e. no valueable speedup will be gained. Therefore, BF feature of Bloom-Bird will remain deactivated until its main hash table reaches into 16-bit order. Afterwards, BF array will be allocated and initialized to zero. Hashing mechanism in the BF of Bloom-Bird is inspired by the main hash table of Bird as mentioned before. If number of enteries increases, Bloom-Bird changes size and order of BF feature like the way main hash table does. Bloom-Bird starts with 18-bit order and it increases to 20-bit if the capacity limit is reached (i.e. number of inserted elements reaches $n=2^{18}$). This expansion continues until 32-bit.

In the three "Tables 3, 4, and 5" results show how scaling feature helps accelerating the Bloom-Bird when prefix set 2 is inserted in comparison when prefix set 1 is inserted. Since number of prefixes in the prefix set 2 is bigger than Bloom-Bird default hash order (i.e. 18-bits), the order of BF is scaled up to 20-bits and consequently the FP is decreased in comparison when prefix set 1 is inserted. This situation also shows how FP error rate is important and can make the searches faster.

When "Tables 3 and 4" are compared, the speedups of fib_route() function are lower than its similar situation in the fib_find(). That is because of fib_find() tires just once for given prefix and length but fib_route() tries *W(n)* times in worst case which *n* is 32 for IPv4. Therefore, fib_route() in most cases finds the best match.

For memory usage, number of bits in the BF array can be calculated using Eq. "(3)" as mentioned before. Although 4-bit counters are generally used for counters in CBF, in the Bloom-Bird FIBs, 8-bit counters are used in the CBF because of simplicity and lower overhead of increment operations in the PC for Byte data-type. Since k is constant and is set to 2 and maximum number of inputs by default is 18-bits (i.e. $n=2^{18}$); therefore, the memory requirement for Bloom-Bird in the 18-bits order based on Eq. "(3)" is 1 MB. When the capacity limit is reached, it will be incremented by 2; therefore, it will be 20-bits order. This order requires 4 MB of memory. This expansion continues until 32-bit and the memory requirement can be calculated using Eq. "(3)".

Similar to two previous sub-sections which IPv4 scenario and results discussed, in the following two sub-sections, the same approach is used but IPv6 prefix sets are used. In the first subsection, scenario is discussed and in the next subsection, results are discussed.

4.3 IPv6 Scenario

Compared to IPv4, unfortunately, latest traces from RouteViews [19] show that existing IPv6 prefixes are very fewer. For example number of latest IPv6 unique prefixes were 16,500 compared to IPv4 which were more than 480,000 unique prefiex. Therefore, in order to show BF advantage of Bloom-Bird over standard Bird, we had to increase the IPv6 prefix sets. For this purpose, ipv6gen [20]

tool is used in order to increase number of unique prefixes by calculating possible subnets from exitsing real prefixes. Moreover, just like previous scenario in the first sub-section, three completely manually generated prefixes (not real) are generated using ipv6gen in order to show BF efficiency. The IPv6 prefix sets are shown in the "Table 6".

It should be noted that Bird router converts all IPv6 prefixes into a single 32-bits IP structure using bit-wise OR. It means all previous IPv4 functions can be applied to IPv6 prefixes. The 128-bits prefixes are split into four 32-bits and they are bit-wise ORed into a single 32-bits prefix. Therefore, Bloom-Bird functions can be applied easily to the IPv6 prefixes.

Table 6. IPv6 Prefix sets to test the two versions of Bird.

Prefix set alias	# of nodes
Prefix1	491,136
Prefix2	762,816
Prefix3	1,042,176
Prefix4	2,103,152
Prefix5	2,109,152
Prefix6	4,212,304

Prefix sets 1-3 are based on real IP6 prefix sets gathered from RouteViews [19] from years 2011, 2012 and 2013 sorted by date respectively. The two versions of Bird i.e. standard Bird and Bloom-Bird are evaluated by inserting these real prefix sets and querying them. Prefix sets 4 and 5 are manually generated using ipv6gen [20] tool. Prefix set 6 is concatenation of two prefix sets 4 and 5 in order to test searching FIBs with even bigger prefix sets and make sure about the results. These last three prefix sets contain 99% missing (not existing) prefixes compared to the other three real prefixes (i.e. prefix sets 1-3) in order to show performance of BF when most queries return negative answer. These last three prefix sets are not real prefix traces; therefore, they are only used for searching, not for inserting into FIBs.

Percentage of number of missing nodes when each prefix set is searched, is presented in "Table 7". For example when all prefixes in prefix set 2 are inserted into a FIB and all prefixes in the prefix set 1 are queried afterwards, 12.03% of searches return negative answer.

Table 7. Percentage of missing nodes when searching for prefix sets

Inserted prefix set / Searched prefix set	Prefix1	Prefix2	Prefix3
Prefix1	0	12.03%	19.3%
Prefix2	43.36%	0	10.56%
Prefix3	61.9%	36.83%	0
Prefix4	99.74%	99.72%	99.76%
Prefix5	99.54%	99.47%	99.5%
Prefix6	99.64%	99.6%	99.63%

Results of evaluation based on IPv6 prefix sets are included and discussed in the following subsection.

4.4 IPv6 Results of Bloom-Bird and Discussion

As discussed in the previous subsection, three IPv6 prefix sets 1-3 are inserted into a FIB at three different times in the two versions of standard Bird and Bloom-Bird and all prefix sets 1-6 are queried afterwards. Percentages of speedups of Bloom-Bird (fib_find() and fib_route() functions) over standard Bird and FP error rate are presented in "Tables 8, 9 and 10", respectively. As mentioned in the second sub-section, these results are gained on a home PC with 2.88 MHz dual core CPU, 6 MB cache and 4 GB RAM which runs unmodified (vanilla) Linux kernel 3.12.

Table 8. IPv6 Speedups of Bloom-Bird fib_find() over standard Bird - Simple Search Function (*) means the same prefix set is inserted

Inserted prefix set / Searched prefix set	Prefix1	Prefix2	Prefix3
Prefix1	(*) 8%	30%	33%
Prefix2	49%	(*) 19%	32%
Prefix3	56%	46%	(*) 18%
Prefix4	90%	91%	90%
Prefix5	70%	67%	60%
Prefix6	83%	80%	79%

Table 9. IPv6 Speedups Bloom-Bird of fib_route() over standard Bird - LPM Search Function (*) means the same prefix set is inserted

Inserted prefix set / Searched prefix set	Prefix1	Prefix2	Prefix3
Prefix1	(*) 10%	18%	22%
Prefix2	18%	(*) 14%	22%
Prefix3	20%	24%	(*) 17%
Prefix4	22%	63%	64%
Prefix5	54%	60%	62%
Prefix6	31%	62%	63%

Table 10. IPv6 False Positive Percentage of Bloom-Bird (*) means the same prefix set is inserted

Inserted prefix set / Searched prefix set	Prefix1	Prefix2	Prefix3
Prefix1	(*) 0	2.82%	5.58%
Prefix2	5.85%	(*) 0	3.81%
Prefix3	7.27%	6.5%	(*) 0
Prefix4	4.33%	8.21%	13.18%
Prefix5	3.67%	7.39%	12.07%
Prefix6	4.00%	7.8%	12.62%

In the "Table 8", speedups of fib_find() function which is responsible for simple searching for a given prefix and length is presented. In the "Table 9", speedups of fib_route() function which is responsible for Longest Prefix Matching (LPM) is presented (starting length for LPM is set to 128 for all searches). In the last "Table 10", percentage of FP error rate is presented. Also there are 6 rows in the aforementioned tables, representing what prefix set is searched. The smallest speedup is 8% and biggest speedup is 91%. Smaller speedups are gained when most prefixes are found after search (i.e. number of existing nodes are bigger than missing nodes). On the other hand, bigger speedups are gained when most prefixes are not found after search.

As mentioned before, in order to guarantee FP rate and performance, the Bloom-Bird calculates its BF array size based on Eq. "(3)". Therefore, for its default 18-bits order (maximum number of inserted elements can be up to $n=2^{18}$), size of BF array can be calculated based on Eq. "(3)" which leads to $m=2^{20}$. Consequently, given these values of m,n and $k=2$, Eq. "(1)" results 15% FP error rate. Experimental results also prove the resulted value even in lower values which "Table 10" shows the most FP error rate resulted is 13.18 percent. Therefore, Bloom-Bird handles its FP error rate even better than expected.

When "Tables 8 and 9" are compared, the speedups of fib_route() function are lower than their similar situation in the fib_find(). That's because of fib_find() tires just once for given prefix and length but fib_route() tries *W(n)* times in worst case which *n* is 128 for IPv6. Consequently, fib_route() in most cases finds the best match.

Although comparing IPv6 scenario to IPv4 scenario is not fair in general because of different number of prefixes and length distribution of them, but speedup of IPv6 compared to IPv4 is a little lower and False Positive errors are a little higher. The only reason for that can be simple hashes that cannot distribute the IPv6 prefixes as well as IPv4 prefixes that use fewer bits to represent prefixes. Therefore, False Positive errors because of simple IPv6 hashes become bigger and speedups become lower compared to IPv4 scenario.

5. Conclusion

The paper showed and presented another application of Bloom filter on a practical open-source router. The BF implementation on Bird's FIB data structure showed that it can help Bird to search and route faster when number of inserted prefixes into a FIB becomes huge. Bloom-Bird which utilizes a Bloom filter in its architecture, evaluated using various prefix sets gathered from real routers traces and also manually generated prefix sets to make the tests more accurate and reliable. Bloom-Bird employs a Bloom-filter in Bird's FIB data structure in order to accelerate the IP lookups when FIB's linked list chains become long. Comparison using different prefix sets showed that up to 93% speedup is gained when most searches return negative answer. This improvement is achieved at the cost of Bloom filter space overhead. Moreover, it is showed how Bloom-Bird can handle its FP error rate when number of inserted prefixes increases by scaling the Bloom filter capacity. The results presented and discussed for both IPv4 and IPv6 prefix sets.

Regardless whether Bloom filter is going to be used as an extra stage before hashing mechanism or other searching data structures (e.g. trie), it can help to avoid traversing chains and paths when result of a search is negative. Therefore, our software based approach is applicable to any other software based routers to accelerate their IP lookups when their FIBs become huge.

References

[1] Y. Zhu, Y. Deng, and Y. Chen. "Hermes: an integrated CPU/GPU microarchitecture for IP routing," presented at the 48th Conf. Design Automation Conference, San Diego, California, 2011.

[2] S. Han, K. Jang, K. Park, and S. Moon. "PacketShader: a GPU-accelerated software router," ACM SIGCOMM Computer Communication Review, vol. 41, pp. 195-206, 2010.

[3] O. Filip. "The BIRD Internet Routing Daemon Project" Internet: www.bird.network.cz/?index, Jun. 15, 2013 [Mar. 7, 2017].

[4] P. Jakma. "Quagga Software Routing Suite." Internet: www.nongnu.org/quagga, Dec. 6, 2015 [Mar. 7, 2017].

[5] M. Handley, O. Hodson, and E. Kohler. "XORP: an open platform for network research," ACM SIGCOMM Computer Communication Review, vol. 33, pp. 53-57, 2003.

[6] B. Bloom. "Space/time trade-offs in hash coding with allowable errors," Communications of the ACM, vol. 13, pp. 422-426, 1970.

[7] S. Tarkoma, C. E. Rothenberg, and E. Lagerspetz. "Theory and practice of bloom filters for distributed systems," Communications Surveys & Tutorials, IEEE, vol. 14, pp. 131-155, 2012.

[8] L. Fan, P. Cao, J. Almeida, and A. Broder. "Summary cache: a scalable wide-area web cache sharing protocol," IEEE/ACM Transactions on Networking (TON), vol. 8, pp. 281-293, 2000.

[9] C. Rothenberg, C. Macapuna, F. Verdi, and M. Magalhães. "The deletable bloom filter: a new member of the bloom family," IEEE Communications Letters, vol. 14, pp. 557-559, 2010.

[10] A. Broder and M. Mitzenmacher. "Network Applications of Bloom Filters: A Survey," Internet Mathematics, vol. 1, pp. 636-646, 2002.

[11] M. Ahmadi and S. Wong. "Modified collision packet classification using counting Bloom filter in tuple space," presented at the 25th Int. Multi-Conference: Parallel and distributed computing and networks, Innsbruck, Austria, 2007.

[12] L. Hyesook and L. Nara. "Survey and proposal on binary search algorithms for longest prefix match," Communications Surveys & Tutorials, IEEE, vol. 14, pp. 681-697, 2012.

[13] S. Dharmapurikar, P. Krishnamurthy, and D. Taylor. "Longest prefix matching using bloom filters," presented at the Conf. on Applications, technologies, architectures, and protocols for computer communications, Karlsruhe, Germany, 2003.

[14] S. Sahni and K. S. Kim. "Efficient construction of multibit tries for IP lookup," IEEE/ACM Transactions on Networking, vol. 11, pp. 650-662, 2003.

[15] K. Lim, K. Park, and H. Lim. "Binary search on levels using a Bloom filter for IPv6 address lookup," presented at the 5th ACM/IEEE Symposium on Architectures for Networking and Communications Systems, Princeton, New Jersey, 2009.

[16] S. Haoyu, H. Fang, M. Kodialam, and T. V. Lakshman. "IPv6 Lookups using Distributed and Load Balanced Bloom Filters for 100Gbps Core Router Line Cards," in INFOCOM 2009, IEEE, 2009, pp. 2518-2526.

[17] J. Bonwick. "The slab allocator: an object-caching kernel memory allocator," presented at the Technical Conf. on USENIX Summer 1994, Boston, Massachusetts, 1994.

[18] D. Meyer. "RouteViews IPv4 BGP RIBs. 2008, 2010, 2013.", Internet: www.routeviews.org/bgpdata Feb. 28, 2017 [Mar. 7, 2017].

[19] D. Meyer. "RouteViews IPv6 BGP RIBs. 2011, 2012, 2013.", Internet: www.routeviews.org/bgpdata Feb. 28, 2017 [Mar. 7, 2017].
[20] V. Kotal, "IPv6 prefix generator.", Internet: www.github.com/vladak/ipv6gen Jan. 29, 2011 [Mar. 7, 2017].

Bahram Bahrambeigy received his B.Sc degree in Information Technology Engineering and M.Sc degree in Computer Networks Engineering both from Islamic Azad University (IAU) in 2011 and 2013, respectively. He is currently working at Pishgaman Tosee Ertebatat (PTE) Tehran as Datacenter Engineer. His research interests include Computer Networks, Software Routers and High-performance Computing.

Mahmood Ahmadi received the B.Sc degree in Computer engineering from Isfahan University, Isfahan, Iran in 1995. He received the M.Sc degrees in Computer architecture and engineering from Tehran Polytechnique University, Tehran, Iran in 1998. From 1999 to 2005, he was a faculty member at Razi university in Kermanshah in Iran. In October 2005, he joined the Faculty of Electrical Engineering, Mathematics, and Computer Science (EEMCS), Delft University of Technology, Delft, The Netherlands, as fulltime Ph.D student. He got his Ph.D in May 2010. His research interests include Computer architecture, network processing, Bloom filters, software defined networking, and high-performance computing. He is working as an assistant professor at Computer Engineering Department in Razi University of Kermanshah, Iran.

Mahmood Fazlali Mahmood Fazlali received B.Sc in computer engineering from Shahid Beheshti University (SBU) in 2001. Then he received M.Sc from University of Isfahan in 2004, and Ph.D from SBU in 2010 in computer architecture. He performed researches on reconfigurable computing systems in computer engineering lab of Technical University of Delft (TUDelft) as a postdoc researcher. Now, he is working as an assistant professor at computer science department at SBU. His research interest includes high performance computing, parallel processing, reconfigurable computing and computer aided design.

An Efficient Noise Removal Edge Detection Algorithm Based on Wavelet Transform

Ehsan Ehsaeyan*
Department of Electrical and Computer engineering, Sirjan University of Technology, Sirjan, Iran
ehsaeyan@sirjantech.ac.ir

Received: 18/Apr/2016 Revised: 15/Aug/2016 Accepted: 15/Sep/2016

Abstract

In this paper, we propose an efficient noise robust edge detection technique based on odd Gaussian derivations in the wavelet transform domain. At first, new basis wavelet functions are introduced and the proposed algorithm is explained. The algorithm consists of two stage. The first idea comes from the response multiplication across the derivation and the second one is pruning algorithm which improves fake edges. Our method is applied to the binary and the natural grayscale image in the noise-free and the noisy condition with the different power density. The results are compared with the traditional wavelet edge detection method in the visual and the statistical data in the relevant tables. With the proper selection of the wavelet basis function, an admissible edge response to the significant inhibited noise without the smoothing technique is obtained, and some of the edge detection criteria are improved. The experimental visual and statistical results of studying images show that our method is feasibly strong and has good edge detection performances, in particular, in the high noise contaminated condition. Moreover, to have a better result and improve edge detection criteria, a pruning algorithm as a post processing stage is introduced and applied to the binary and grayscale images. The obtained results, verify that the proposed scheme can detect reasonable edge features and dilute the noise effect properly.

Keywords: Wavelet Transform; Edge Detection; Gaussian Filter; Multiscale Analysis; Noise Removal; Gaussian Bases; Wavelet Function Derivation; Admissibility Condition; Edge Criteria; N-connected Neighborhood.

1. Introduction

Nowadays, image processing has an important role in the proceeding of new science. Edge detection has various applications and is a useful tool in the registration, pattern recognition, topological recognition, image compression and other computer vision fields. Classical edge detectors like Roberts, Sobel and Prewitt have a simple structure which helps the time consumption saving. However, they have problems in the noisy condition and cannot discriminate the noise and background points properly. Furthermore, they cannot present images in automatic zoom and different scales.

One of the most popular algorithms is Gaussian-based edge detection due to the noise removal [1-2]. Canny proposed an edge detector based on Gaussian filter and identified three criteria for the optimal edge detector (good detection, good localization and low spurious response), which was successful in the noise free and high-contrast images [3]. Canny detector is a popular method which has been revised many times since it has been introduced [4-5]. But the noise interference is inevitable and natural images are almost polluted by the noise. The edge detection would be challenging and time consuming when the noise contaminates the image unexpectedly [6].

Another noise removal solution is the usage of scale-space theory. Multi-scale edge detection using wavelet transform has been introduced by Mallat [7]. Selecting a large scale can block the noise effect. In this condition, spurious and false responses are weakened and disappear. But it occurs with the dislocation edge error too. On the other hand, selecting the low scale results in the noise sensitive detection. Therefore, there is a tradeoff between the good detection and the edge localization in noisy images. Multiresolution analysis has been introduced to obtain an intermediated compromised result. Zhang has continued Sadler [8] idea and proposed the scale multiplication to compromise between the localization error and the noise sensitive detection [9]. Zhu has used the scale multiplication technique based on the odd Gabor transform domain for the noise overcoming in the edge detection [10].

A number of Cellular Automata (CA)-based edge detectors have been developed recently due to the simplicity of the model and the potential for simultaneous removal of different types of noise in the process of detection [11-13].

With the increasing requirements of the accuracy of algorithms in the image edge detection, some intelligent algorithms are used, such as artificial neural network [14], fuzzy optimization [15], Genetic algorithm, ant colony optimization [16] and Particle Swarm Optimization.

Also, some new techniques have been developed in this field, which improve the edge detection performance, such as designing edge detector filters in potential field [17], Krawtchouk orthogonal polynomials [18], arctangent edge model [20], wavelet transform [19,21] and gravity field [22-23].

In this paper, we focus on the Gaussian edge detection and develop Canny edge detector by the derivation of

* Corresponding Author

Gaussian wavelet function and improve results by introducing an algorithm which joints different edge maps. We show that this technique reduces spurious responses and improves edge detection criteria.

Our paper is organized as follows: Section 2 discusses the principal of the edge detection idea by the wavelet transform and introduces new wavelet functions based on *n*th derivative Gaussian, which are used in this paper for the edge detection. Section 3 deals with our scheme description and famous edge detection criteria and Section 4 demonstrates experimental results.

2. New Bases Introduction

Canny has used the first order derivative of the Gaussian filter as the wavelet function. We develop this idea to *n*th order derivative of the Gaussian filter. The wavelet functions derived from $g(x,y)$ in the direction of *x* and *y* as:

$$\psi_x^n(x,y) = \frac{\partial^n g(x,y)}{\partial x^n} \quad (1a)$$

$$\psi_y^n(x,y) = \frac{\partial^n g(x,y)}{\partial y^n} \quad (1b)$$

$n = 1,2,\ldots$

These bases satisfy the admissibility condition and tend to 0 in $\pm\infty$. Assume $g_s(x,y)$ be the smoothing function at the scale s

$$g_s(x,y) = \frac{1}{2\pi s^2} e^{-\frac{1}{2s^2}(x^2+y^2)} \quad (2)$$

Hence, scaled wavelet bases are defined as

$$\psi_{s,x}^n(x,y) = s^n \frac{\partial^n g_s(x,y)}{\partial x^n} \quad (3a)$$

$$\psi_{s,y}^n(x,y) = s^n \frac{\partial^n g_s(x,y)}{\partial y^n} \quad (3b)$$

For an image $f(x,y)$, its wavelet transform has two elements in the x and y directions.

$$W_s^x(x,y) = f(x,y) * \psi_s^x(x,y) \quad (4-a)$$

$$W_s^y(x,y) = f(x,y) * \psi_s^y(x,y) \quad (4-b)$$

$$M_s f(x,y) = \sqrt{W_s^x(x,y)^2 + W_s^y(x,y)^2} \quad (4-c)$$

The points at which their modulus values ($M_s f(x,y)$) are the local maximums correspond to abrupt change points in the corresponding positions of the smooth image or the position of the sharp and steep changes, whose sizes reflect the gray strengths in the positions. So, as long as we detect the local maximum value points of the wavelet transform series modulus along the gradient direction, the edge points of the image are gained.

3. Method Description and Analysis Parameters

In this section, we describe the proposed method. Then some famous criteria are discussed briefly, which are used in experimental results.

3.1 Method Description

Traditionally, first derivation of the Gaussian wavelet is considered as an edge detector [3]. Finding local maxima of absolute $M(x,y)$ in $A(x,y)$ direction yields edge points (magnitude and orientation). Another method is the usage of the second derivation of the Gaussian filter response and finding zero-crossing points, which is very sensitive to the noise. Zhang improved results by the scale multiplication method [9]. We complete Zhang method and introduce here a novel technique of the Gaussian wavelet edge detection to refine the noise interference. This technique is based on the multiplication of $\psi_s^n(x,y)$ edge responses not only across the scale s, but across the derivation *n*. This procedure has two freedom degree parameters to adjust noise and the edge dislocation. Fig 1 shows the block diagram of the proposed method. Our method consists of five steps:

1- Input noisy image which has been corrupted by AWGN.
2- Calculate $\psi_{s,x}^n(x,y)$ and $\psi_{s,y}^n(x,y)$ according to Eq (3) in unit *scale* and $n = 1,2,3$. The length of filters is 7.
3- Obtain relevant coefficients in x and y directions by convolution of calculated bases in step 2 with the noisy image.
4- Prepare the edge maps of every derivation (i.e. n=1, 3, 5) according to Eq (4-c) which named $W_s^n(x,y)$. These results are shown in the first stage of Figure 1.
5- Apply pruning algorithm to different edge maps and yield a result with higher quality and lower fake edges.

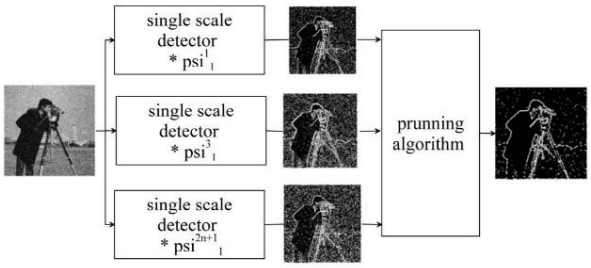


Fig. 1. Block diagram of proposed method

3.2 Analysis Criteria

Pratt [25] introduced a criterion that shows the quantity performance of edge detection, which is used in much research [10, 24]. This parameter is called *figure of merit* and defined as:

$$F = \frac{1}{max\{N_I, N_A\}} \sum_{i=1}^{N_A} \frac{1}{1 + \alpha d^2(i)} \quad (5)$$

Where N_I is the number of true edges, and N_A represents the number of marked edges by the detector algorithm. α is a penalty scaling number that controls false edges and is set on 1/9 in this paper like Pratt work. *d* means the Euclidian distance between the point detected by the algorithm procedure and marked as the edge point and its actual edge in the reference map. There

are three types of distance definition between two pixels (x_1, y_1) and (x_2, y_2) which are used in this paper as follows:

- Cityblock: in 2D space, the cityblock distance is defined as $|x_1\text{-}x_2|+|y_1\text{-}y_2|$
- Chessboard: which is identified by max ($|x_1\text{-}x_2|$,$|y_1\text{-}y_2|$)
- Quasi-Euclidean: the quasi-Euclidean distance is calculated by:

$$d_{quasi-euclidean} = \begin{cases} |x_1 - x_2| + (\sqrt{2}-1)|y_1 - y_2| & |x_1 - x_2| > |y_1 - y_2| \\ (\sqrt{2}-1)|x_1 - x_2| + |y_1 - y_2| & otherwise \end{cases} \quad (6)$$

Second parameter is based on the distance between marked edges and true edges. Root mean square localization error is denoted by *D* and designed by Zhang [9]:

$$D = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(p_g(i) - p_d(i))^2} \quad (7)$$

where *N* in this formula is the number of edge points. The actual edge position is denoted by p_g and the detected edge point by the algorithm is denoted by p_d.

The edges are classified in four groups in the edge detection process:

True positive (TP): these edges are actual edges, and we detected them correctly. An edge detector would be more powerful that has a higher TP edges result.

False positive (FP): this criterion refers to the amount of the edges that are not real edges; but we detected them as true edges.

True negative (TN): these points are not edges and we ignored them correctly. TP has a reverse relationship with FP in most cases. The greater true negative means the better edge detection result.

False negative (FN): we neglected this type of points as edges. But they are true edges. Lower FP means better edge quality.

True positive rate (TPR): this normalized criterion contains correct and false detection pixels, and shows the sensitivity of results. True positive rate is calculated as:

$$TPR = \frac{TP}{TP + FN} \quad (8)$$

False positive rate (FPR): conversely, FPR refers to the error of edge detecting. False positive rate is between 0 to 1 and specified by:

$$FPR = \frac{FP}{FP + TN} \quad (9)$$

Precision (PREC): another evaluation parameter declares how the percentage of marked edge points is true. Precision is obtained from [2]:

$$PREC = \frac{TP}{TP + FP} \quad (10)$$

F alpha-measure (Fα): F alpha-measure shows the overall quality and is given by [26]:

$$F_\alpha = \frac{PREC\ .TPR}{\alpha PREC + (1-\alpha)TPR} \quad (11)$$

Where α is a scaling constant between 0 and 1.

Accuracy: it shows the precision of diagnostic true edges in edge detection. Accuracy is delivered by percentage and obtained from:

$$accuracy\ (\%) = 100 \times \frac{TP + TN}{TP + TN + FP + FN} \quad (12)$$

4. Experimental Results

In this section, the ability of proposed edge detector is shown. We present results in visual and statistical modes.

4.1 Visual Results

In this part, the wavelet response of proposed bases is investigated. We applied three wavelet bases $\psi^1(x,y), \psi^3(x,y)$ and $\psi^5(x,y)$ to the images. The results are illustrated in Fig 2. Two types of image were considered in this paper: binary ('WAVELET TRANSFORM') and grayscale ('Lena' and 'cameraman') with the size of 256*256 pixels.

Each image has two columns. First one is a noise free image and its $W_1^1(x,y)$, $W_1^3(x,y)$ and $W_1^5(x,y)$ wavelet edges respectively. And second column shows the image with an exploited edge from the wavelet coefficient at scale $s = 1$. These results are achieved after applying the threshold to wavelet coefficients at the scale *s*=1. The images are corrupted by additive white Gaussian noise with the different variance. We find out, there is a little difference between $W_s^1(x,y)$, $W_s^3(x,y)$ and $W_s^5(x,y)$. So all of them can be used in edge detection separately. In much research, first order derivative of the smooth function is used as the edge detector.

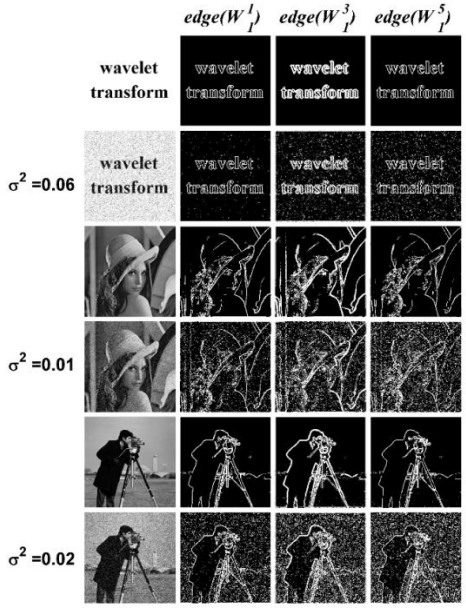
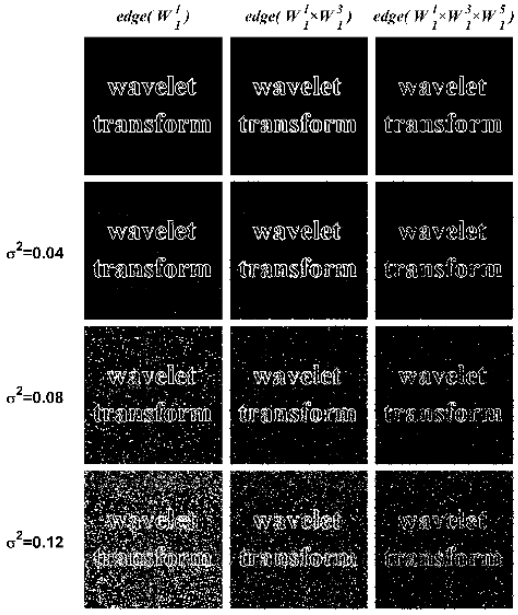
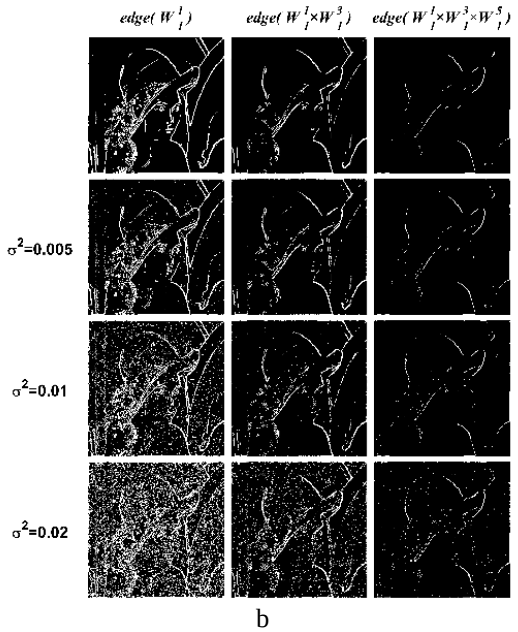


Fig. 2. First, third and fifth Gaussian derivation wavelet response with different noise power density in scale s=1. First columns of images are noise free.

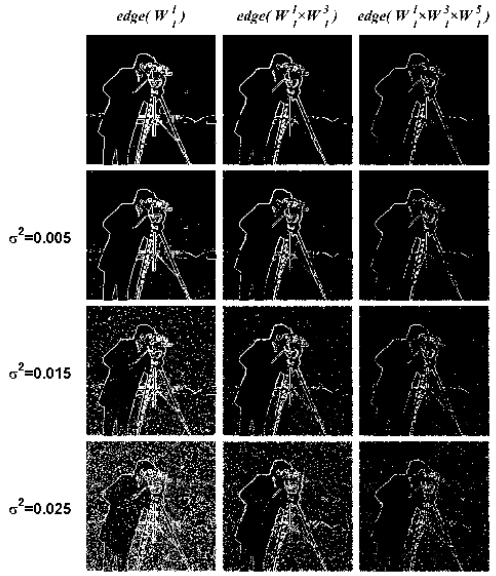
Fig 3 shows the results of the edge detection based on first, third and fifth order derivation of the Gaussian smooth function (response of $\psi^1(x,y), \psi^3(x,y)$ and $\psi^5(x,y)$) in ideal (first rows) and various noise power (second and third rows) cases. The first rows indicate the noise free response where coefficients multiplication, $W_s^1(x,y) \times W_s^3(x,y)$ or $W_s^1(x,y) \times W_s^3(x,y) \times W_s^5(x,y)$ extenuates thick edges. When the image is polluted a little by noise, the best choice is the use of single $W_s^i(x,y)$ for the edge detection. When the noise power density is considerable, and we cannot ignore the effect of noise, it is better to use $W_s^1(x,y) \times W_s^3(x,y)$ instead of the traditional one for edge detection. Assume that the image is contaminated by noise significantly. The best choice is $W_s^1(x,y) \times W_s^3(x,y) \times W_s^5(x,y)$ where the wrong edges number is lowest and true edges remain meaningful.



a



b



c

Fig. 3. Multiplication of wavelet coefficients according to Gaussian bases in different noise condition

4.2 Statistical Results

The previous part shows the edge detection in the visual scene. In this part, the statistical results of the parameters which have been introduced in section 2, are calculated and presented in tables for the discussion and comparison. We can investigate the edge responses and obtain results similar to the ones in the previous part. Table 1 shows the calculated statistical parameters of the three images ('wavelet transform', 'Lena', 'cameraman') with the different noise power.

Table. 1. Statistical results of edge detection. *d1*='cityblock', *d2*='chessboard' and *d3*='quasi-Euclidian' distance definition (refer to section 3)

			'wavelet transform' binary image		
			$\sigma^2 = 0.04$	$\sigma^2 = 0.08$	$\sigma^2 = 0.12$
W_1^1 *(traditional wavelet edge detection)*	*F*	*d1*	0.9295	0.6522	0.4367
		d2	0.9303	0.6601	0.4489
		d3	0.9299	0.6556	0.4418
	D	*d1*	0.2495	0.8323	1.3432
		d2	0.2330	0.8542	1.3686
		d3	0.2367	0.8183	1.3130
	TPR		0.9168	0.9143	0.9141
	FPR		0.0032	0.0546	0.1821
	PREC		0.9576	0.5655	0.2806
	F_α		0.9367	0.6988	0.4294
	ACCURACY		**99.1074**	94.3176	82.4844
$W_1^1 \times W_1^3$ *(proposed in this paper)*	*F*	*d1*	0.9284	0.8157	0.6417
		d2	0.9306	0.8201	0.6500
		d3	0.9293	0.8175	0.6453
	D	*d1*	0.3926	0.6173	0.9527
		d2	0.4099	0.6499	0.9683
		d3	0.3835	0.6099	0.9303
	TPR		0.8993	0.8506	0.8129
	FPR		0.0101	0.0266	0.0670
	PREC		0.8737	0.7127	0.4852
	F_α		0.8863	0.7756	0.6077
	ACCURACY		98.3368	96.4508	92.4316
$W_1^1 \times W_1^3 \times W_1^5$ *(proposed in this paper)*	*F*	*d1*	0.7685	0.7238	0.7324
		d2	0.7687	0.7249	0.7371
		d3	0.7686	0.7242	0.7344
	D	*d1*	0.1242	0.3149	0.6543
		d2	0.1285	0.3520	0.6702
		d3	0.1190	0.3268	0.6458
	TPR		0.7642	0.6988	0.6472
	FPR		7.7289e-04	0.0061	0.0232
	PREC		0.9872	0.8992	0.6843
	F_α		0.8615	0.7865	0.6652
	ACCURACY		98.2285	**97.2641**	**95.3033**

d1='cityblock' d2='chessboard' d3='quasi-Euclidean'

Table. 1. Continue

			lenna		
			$\sigma^2 = 0.005$	$\sigma^2 = 0.01$	$\sigma^2 = 0.02$
W_1^1 *(traditional wavelet edge detection)*	*F*	*d1*	0.8432	0.7083	0.5671
		d2	0.8563	0.7378	0.6096
		d3	0.8486	0.7203	0.5847
	D	*d1*	0.6980	1.2090	1.6069
		d2	0.7536	1.2742	1.6667
		d3	0.6575	1.1517	1.5292
	TPR		0.7804	0.7877	0.7663
	FPR		0.0222	0.0880	0.2224
	PREC		0.8183	0.5342	0.3062
	F_α		0.7989	0.6367	0.4376
	ACCURACY		**95.5383**	89.7903	77.6306
$W_1^1 \times W_1^3$ *(proposed in this paper)*	*F*	*d1*	0.3829	0.4289	0.6989
		d2	0.3847	0.4329	0.7293
		d3	0.3836	0.4305	0.7114
	D	*d1*	0.3558	0.5734	1.2330
		d2	0.3902	0.6179	1.3280
		d3	0.3449	0.5449	1.1702
	TPR		0.3738	0.3992	0.5003
	FPR		0.0028	0.0086	0.0664
	PREC		0.9450	0.8564	0.4913
	F_α		0.5357	0.5446	0.4957
	ACCURACY		92.6422	**92.4179**	88.4430
$W_1^1 \times W_1^3 \times W_1^5$ *(proposed in this paper)*	*F*	*d1*	0.0930	0.1190	0.2251
		d2	0.0930	0.1191	0.2284
		d3	0.0930	0.1191	0.2265
	D	*d1*	0	0.2042	0.7327
		d2	0	0.1957	0.7575
		d3	0	0.1667	0.6865
	TPR		0.0930	0.1182	0.2012
	FPR		1.7213e-05	1.8935e-04	0.0071
	PREC		0.9986	0.9877	0.7846
	F_α		0.1701	0.2112	0.3202
	ACCURACY		89.6988	89.9704	**90.3015**

d1='cityblock' d2='chessboard' d3='quasi-Euclidean'

Table. 1. Continue

			cameraman		
			$\sigma^2 = 0.005$	$\sigma^2 = 0.015$	$\sigma^2 = 0.025$
W_1^1 *(traditional wavelet edge detection)*	*F*	*d1*	0.8955	0.7175	0.4991
		d2	0.8972	0.7298	0.5216
		d3	0.8961	0.7221	0.5074
	D	*d1*	0.2807	0.9335	1.4477
		d2	0.3041	0.9681	1.5200
		d3	0.2631	0.9012	1.4111
	TPR		0.8803	0.8378	0.8565
	FPR		0.0061	0.0526	0.1879
	PREC		0.9363	0.6170	0.3157
	F_α		0.9074	0.7107	0.4614
	ACCURACY		**98.3490**	93.7302	81.6223
$W_1^1 \times W_1^3$ *(proposed in this paper)*	*F*	*d1*	0.6944	0.7255	0.6653
		d2	0.6967	0.7316	0.6809
		d3	0.6953	0.7278	0.6710
	D	*d1*	0.3453	0.5952	1.0569
		d2	0.3935	0.6434	1.1130
		d3	0.3389	0.5902	1.0267
	TPR		0.6724	0.6474	0.6605
	FPR		0.0056	0.0199	0.0662
	PREC		0.9242	0.7668	0.5024
	F_α		0.7785	0.7020	0.5707
	ACCURACY		96.4828	**94.9493**	90.8676
$W_1^1 \times W_1^3 \times W_1^5$ *(proposed in this paper)*	*F*	*d1*	0.4128	0.4109	0.4783
		d2	0.4128	0.4115	0.4827
		d3	0.4128	0.4111	0.4800
	D	*d1*	0.0201	0.2745	0.7557
		d2	0.0201	0.2933	0.7950
		d3	0.0201	0.2671	0.7351
	TPR		0.4126	0.4026	0.4272
	FPR		1.0082e-04	0.0023	0.0155
	PREC		0.9976	0.9469	0.7358
	F_α		0.5837	0.5650	0.5405
	ACCURACY		94.5923	94.3024	**93.3258**

d1='cityblock' d2='chessboard' d3='quasi-Euclidean'

5. Improved Algorithm

In the previous section, we saw that $W_s^1(x,y)$ (traditional wavelet edge detection) was useful in low noise condition and had good results in parameters listed in Table 1. But its drawback is the acting on the medium and high noise level. In other words, it is very sensitive to noise contaminating. In this condition, $W_s^1(x,y) \times W_s^3(x,y)$ or $W_s^1(x,y) \times W_s^3(x,y) \times W_s^5(x,y)$ is introduced as a method to refine noise. $W_s^1(x,y) \times W_s^3(x,y)$ and $W_s^1(x,y) \times W_s^3(x,y) \times W_s^5(x,y)$ are powerful to remove spurious responses where created by the noise. Their suppressing noise parameters such as D, FPR, PREC, F_α and Accuracy have been better than $W_s^1(x,y)$ parameters.

But the edge quality was low, and it could detect only main and thick edges and had low TPR and Figure of merit criteria. $W_s^1(x,y) \times W_s^3(x,y)$ or $W_s^1(x,y) \times W_s^3(x,y) \times W_s^5(x,y)$ kills noise and details simultaneously. So in high polluted images another algorithm is essential to pick up good characteristics of $W_s^1(x,y)$ such as TPR and F and pick up good characteristics of $W_s^1(x,y) \times W_s^3(x,y) \times W_s^5(x,y)$ such as FPR and PREC to improve the edge detection criteria and handles a reasonable response. This algorithm must be applied as a post-processing to result edges, i.e. wavelet coefficients after a thresholding process. An improved edge detection algorithm named pruning algorithm is introduced here. Pruning algorithm is a post-processing stage that applied to binary image (detected edge map). It is useful where there are similar edge maps such as multiresolution levels of an image, and we want to fuse them. In pruning algorithm, a binary frame with the complete edge and also with the high polluted noise is chosen as the initial edge image. Other frames are used to improve edges in the initial edge image.

This algorithm uses $W_s^1(x,y)$ as a basic edge frame. In this process, all the detected points in $W_s^1(x,y)$ are considered as candidate edges and tries to remove false edges by searching the neighborhood of pixels in other frames like $W_s^1(x,y) \times W_s^3(x,y)$ or $W_s^1(x,y) \times W_s^3(x,y) \times W_s^5(x,y)$.

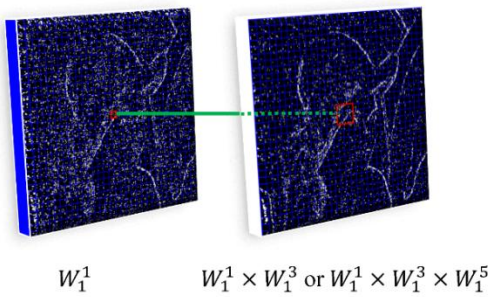


Fig. 4. N-connected neighborhood for searching areas

The pruning algorithm is as follows:

1- Define an n-connected neighborhood for searching areas as shown in Fig 4.

2- For each pixel belonging to $W_s^1(x,y)$, determine this pixel and study n-connected neighborhood in $W_s^1(x,y) \times W_s^3(x,y)$ or $W_s^1(x,y) \times W_s^3(x,y) \times W_s^5(x,y)$. If FPR is more important to us, $W_s^1(x,y) \times W_s^3(x,y) \times W_s^5(x,y)$ is selected for searching area and if accuracy and TPR are more important in the edge detection, $W_s^1(x,y) \times W_s^3(x,y)$ is a better choice.

3- To achieve noise reduction, if the number of detected edges in $W_s^1(x,y) \times W_s^3(x,y)$ or $W_s^1(x,y) \times$

$W_s^3(x,y) \times W_s^5(x,y)$ is more than N, $p(i,j)$ is denoted as a real pixel, otherwise $p(i,j)$ in $W_s^1(x,y)$ changes to zero.

This algorithm applied to study images and results are shown in Fig 5 and Table 2.

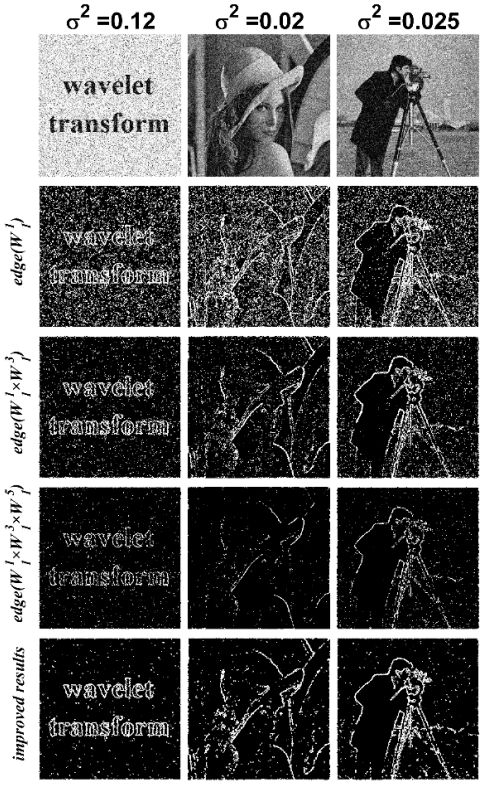


Fig. 5. Applied pruning algorithm to noisy images

The results of the proposed algorithm have a less edge loss and higher noise blocking. According to Table 2, most of the statistical parameters like TPR are compensated after applying the pruning algorithm. Meanwhile, the improved results have the greatest F, F_α and Accuracy.

This procedure would be mixed with the scale edge i.e. the scaled edge frames with s=1,2,… are chosen for the searching area procedure. In this condition due to refining noise in the higher scale, the rate of false edge detection reduced significantly. Also by selecting a lower scale as the initial edge frame, true location is preserved. However, a computational cost is exposed to the edge detection.

Table. 2. Statistical results of the applied pruning algorithm

		F			*D*		
		d1	*d2*	*d3*	*d1*	*d2*	*d3*
'wavelet transform' image	W_1^1	0.437	0.449	0.442	1.343	1.369	1.313
	$W_1^1 \times W_1^3$	0.642	0.65	0.645	0.953	0.968	0.93
	$W_1^1 \times W_1^3 \times W_1^5$	0.732	0.737	0.734	0.654	0.67	0.646
	Improved	0.7627	0.7691	0.7655	0.7475	0.7575	0.7252
lenna	W_1^1	0.5671	0.6096	0.5847	1.6069	1.6667	1.5292
	$W_1^1 \times W_1^3$	0.6989	0.7293	0.7114	1.233	1.328	1.1702
	$W_1^1 \times W_1^3 \times W_1^5$	0.2251	0.2284	0.2265	0.7327	0.7575	0.6865
	Improved	0.7043	0.7231	0.7119	0.9906	1.0555	0.9472
cameram an	W_1^1	0.499	0.522	0.507	1.448	1.52	1.411
	$W_1^1 \times W_1^3$	0.665	0.681	0.671	1.057	1.113	1.027
	$W_1^1 \times W_1^3 \times W_1^5$	0.478	0.483	0.48	0.756	0.795	0.735
	Improved	0.8106	0.8203	0.8144	0.7839	0.8272	0.7553

Table. 2. Continue

		TPR	*FPR*	*PREC*	F_α	*ACCURACY*
'wavelet transform' image	W_1^1	0.914	0.182	0.281	0.429	82.4844
	$W_1^1 \times W_1^3$	0.813	0.067	0.485	0.608	92.4316
	$W_1^1 \times W_1^3 \times W_1^5$	0.647	0.023	0.684	0.665	95.3033
	Improved	0.904	0.035	0.665	0.767	**96.0297**
lenna	W_1^1	0.7663	0.2224	0.3062	0.4376	77.6306
	$W_1^1 \times W_1^3$	0.5003	0.0664	0.4913	0.4957	88.443
	$W_1^1 \times W_1^3 \times W_1^5$	0.2012	0.0071	0.7846	0.3202	90.3015
	Improved	0.5657	0.041	0.6387	0.6	**91.4337**
cameram an	W_1^1	0.857	0.188	0.316	0.461	81.6223
	$W_1^1 \times W_1^3$	0.661	0.066	0.502	0.571	90.8676
	$W_1^1 \times W_1^3 \times W_1^5$	0.427	0.016	0.736	0.541	93.3258
	Improved	0.7523	0.0317	0.706	0.7284	**94.8441**

6. Conclusions

An efficiency edge detection algorithm to remove spurious noise based on nth order derivative of Gaussian wavelet is presented in this paper. To approach the goal, first a new set of wavelet bases is introduced. After that, a new algorithm based on the wavelet coefficients multiplication is presented. We showed that how the use of higher order of Gaussian derivations can improve edge detection criteria. Our algorithm is applied to noisy binary and grayscale images in order to verify the efficiency of the proposed scheme for these two types of images and the results are carried out in both visual and statistical data. The results are compared with the traditional wavelet transform edge detection and investigated edge detection parameters. Our method has two freedom parameters (nth order derivative and the scale) to compare the basic Gaussian wavelet edge detection, which has a single parameter (scale) to adjust the resolution and noise refining. Finally, a neighborhood searching algorithm as a post processing stage is applied to improve the proposed method. The experimental results verified that our scheme is capable of improving image criteria on demand.

References

[1] F. Guo, Y. Yang, B. Chen, and L. Guo, "A novel multi-scale edge detection technique based on wavelet analysis with application in multiphase flows," Powder Technology, vol. 202, no. 1-3, pp. 171–177, Aug. 2010.

[2] C. Lopez-Molina, B. De Baets, H. Bustince, J. Sanz, and E. Barrenechea, "Multiscale edge detection based on Gaussian smoothing and edge tracking," Knowledge-Based Systems, vol. 44, pp. 101–111, May 2013.

[3] J. Canny, "A computational approach to edge detection," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. PAMI-8, no. 6, pp. 679–698, Nov. 1986.

[4] W. McIlhagga, "The canny edge detector revisited," International Journal of Computer Vision, vol. 91, no. 3, pp. 251–261, Oct. 2010.

[5] L. Ding and A. Goshtasby, "On the canny edge detector," Pattern Recognition, vol. 34, no. 3, pp. 721–725, Mar. 2001.

[6] R. C. Gonzalez, R. E. Woods, D. J. Czitrom, and S. Armitage, Digital image processing, 3rd ed. United States: Prentice Hall, 2007.

[7] S. Mallat and S. Zhong, "Characterization of signals from multiscale edges," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 14, no. 7, pp. 710–732, Jul. 1992.

[8] B. M. Sadler and A. Swami, "Analysis of multiscale products for step detection and estimation," IEEE Transactions on Information Theory, vol. 45, no. 3, pp. 1043–1051, Apr. 1999. [9] L. Zhang and P. Bao, "Edge detection by scale multiplication in wavelet domain," Pattern Recognition Letters, vol. 23, no. 14, pp. 1771–1784, Dec. 2002.

[9] Z. Zhu, H. Lu, and Y. Zhao, "Scale multiplication in odd Gabor transform domain for edge detection," Journal of Visual Communication and Image Representation, vol. 18, no. 1, pp. 68–80, Feb. 2007.

[10] M. Hasanzadeh Mofrad, S. Sadeghi, A. Rezvanian, and M. R. Meybodi, "Cellular edge detection: Combining cellular automata and cellular learning automata," AEU - International Journal of Electronics and Communications, vol. 69, no. 9, pp. 1282–1290, Sep. 2015.

[11] S. Uguz, U. Sahin, and F. Sahin, "Edge detection with fuzzy cellular automata transition function optimized by PSO," Computers & Electrical Engineering, vol. 43, pp. 180–192, Apr. 2015.

[12] S. Amrogowicz and Y. Zhao, "An edge detection method using outer totalistic cellular Automata," Neurocomputing, Jun. 2016.

[13] J. Gu, Y. Pan, and H. Wang, "Research on the improvement of image edge detection algorithm based on artificial neural network," Optik - International Journal for Light and Electron Optics, vol. 126, no. 21, pp. 2974–2978, Nov. 2015.

[14] C. I. Gonzalez, P. Melin, J. R. Castro, O. Castillo, and O. Mendoza, "Optimization of interval type-2 fuzzy systems for image edge detection," Applied Soft Computing, vol. 47, pp. 631–643, Oct. 2016.

[15] X. Liu and S. Fang, "A convenient and robust edge detection method based on ant colony optimization," Optics Communications, vol. 353, pp. 147–157, Oct. 2015.

[16] G. Ma, C. Liu, and D. Huang, "The removal of additional edges in the edge detection of potential field data," Journal of Applied Geophysics, vol. 114, pp. 168–173, Mar. 2015.

[17] D. Rivero-Castillo, H. Pijeira, and P. Assunçao, "Edge detection based on Krawtchouk polynomials," Journal of Computational and Applied Mathematics, vol. 284, pp. 244–250, Aug. 2015.

[18] N. Decoster, S. G. Roux, and A. Arnéodo, "A wavelet-based method for multifractal image analysis. II. Applications to synthetic multifractal rough surfaces," The European Physical Journal B, vol. 15, no. 4, pp. 739–764, Jun. 2000.

[19] Q. Sun, Y. Hou, and Q. Tan, "A subpixel edge detection method based on an arctangent edge model," Optik - International Journal for Light and Electron Optics, vol. 127, no. 14, pp. 5702–5710, Jul. 2016.

[20] G. J. Tu and H. Karstoft, "Logarithmic dyadic wavelet transform with its applications in edge detection and reconstruction," Applied Soft Computing, vol. 26, pp. 193–201, Jan. 2015.

[21] B. Zuo and X. Hu, "Edge detection of gravity field using eigenvalue analysis of gravity gradient tensor," Journal of Applied Geophysics, vol. 114, pp. 263–270, Mar. 2015.

[22] J. Wang, X. Meng, and F. Li, "Improved curvature gravity gradient tensor with principal component analysis and its application in edge detection of gravity data," Journal of Applied Geophysics, vol. 118, pp. 106–114, Jul. 2015.

[23] M.-Y. Shih and D.-C. Tseng, "A wavelet-based multiresolution edge detection and tracking," Image and Vision Computing, vol. 23, no. 4, pp. 441–451, Apr. 2005.

[24] Pratt, W.K.: 'Digital Image Processing', (John Wiley & Sons, New York, USA, 2001. Eased)

[25] M. K. Geetha and S. Palanivel, "Video classification and shot detection for video retrieval applications," International Journal of Computational Intelligence Systems, vol. 2, no. 1, pp. 39–50, 2009.

Ehsan Ehsaeyan received the B.Sc degree in electrical engineering from Shahed University, Tehran, Iran in 2005. He received the M.Sc degree in communication engineering from Shahid Bahonar University, Kerman, Iran, in 2009. His area research interests include Image Processing and Digital Signal Processing.

The Separation of Radar Clutters using Multi-Layer Perceptron

Mohammad Akhondi Darzikolaei*
Faculty of Electrical and Computer Engineering, Babol Noshirvani University of Technology, Babol, Iran
m.akhondi@stu.nit.ac.ir
Ata Ebrahimzade
Faculty of Electrical and Computer Engineering, Babol Noshirvani University of Technology, Babol, Iran
E_zade@nit.ac.ir
Elahe Gholami
Faculty of Electrical and Computer Engineering, Babol Noshirvani University of Technology, Babol, Iran
e.gholami8869@yahoo.com

Received: 15/May/2016 Revised: 31/Dec/2016 Accepted: 31/Jan/2017

Abstract

Clutter usually has negative influence on the detection performance of radars. So, the recognition of clutters is crucial to detect targets and the role of clutters in detection cannot be ignored. The design of radar detectors and clutter classifiers are really complicated issues. Therefore, in this paper aims to classify radar clutters. The novel proposed MLP-based classifier for separating radar clutters is introduced. This classifier is designed with different hidden layers and five training algorithms. These training algorithms consist of Levenberg-Marquardt, conjugate gradient, resilient back-propagation, BFGS and one step secant algorithms. Statistical distributions are established models which widely used in the performance calculations of radar clutters. Hence In this research, Rayleigh, Log normal, Weibull and K-distribution clutters are utilized as input data. Then Burg's reflection coefficients, skewness and kurtosis are three features which applied to extract the best characteristics of input data. In the next step, the proposed classifier is tested in different conditions and the results represent that the proposed MLP-based classifier is very successful and can distinguish clutters with high accuracy. Comparing the results of proposed technique and RBF-based classifier show that proposed method is more efficient. The results of simulations prove that the validity of MLP-based method.

Keywords: Clutter; Classifier; Feature; Neural Network; Radar.

1. Introduction

The term radar is an abbreviation for radio detection and ranging. The rudimentary concept of radar system is inspired by echolocation animals such as bats and dolphins. Radar is a system which detects, locates and measures the speed of objects using echo electromagnetic waves. It transmits electromagnetic waves into environments and receives the echoes from objects. It is apparent that radar system is effected by progression of modern technology. This improvement makes the analysis of radar performance more complicated. Many negative factors can have destructive influences on radar performances. Clutter has really the most negative role on radar echo signals. Clutter is any unwanted signal which can disorder echoes from radar. Clutter can be reflected from any things such as lands, sea, forests, mountains and weather conditions. Because of stochastic and variable nature of clutters, radar specialists usually apply probability density functions for describing the traits of clutters. Gaussian, Weibull, Rayleigh, K-distribution and log-normal are most popular and widely used models. Adaptive techniques for detection, tracking and classification of clutters are very crucial. Artificial Neural Networks and Heuristic Algorithms have been used for radar signal processing which requires high capacity to match with different conditions. [1], [2], [3]

In the field of soft computing, Artificial Neural Networks[1] is one of the most important methods. Its invention is inspired by the neurons of human brain. Mc culloch and Pitts [4] were the first ones modeled mathematically the neural networks. The simplicity, low computational cost and high performance are some significant characteristics of this computational approach. Feed forward Neural Networks [5] are very popular tools among other kinds of Neural Networks. They receive data as inputs on one side and prepare outputs from other side by connections between neurons in various layers. Multi-layer perceptron[2] [6,7] is the one type of feed forward neural networks. MLP has more than one perceptron in different layers. This helps it to solve nonlinear problems. Pattern classification [8], data prediction [9], pattern recognition, remote sensing and optimization are few applications of MLPs. The amazing trait of MLPs is learning [10]. Similar to human brain, MLPs have aptitude to learn from experiences. This part is common in all neural networks. Back Propagation algorithms are the instances of learning algorithms which are widely used with MLPs.

As mentioned, Neural Networks and Soft computing algorithms have been used successfully for radar signal

[1] ANN
[2] MLP

* Corresponding Author

processing. Authors in [11] classified various kinds of clutters. They tried to categorize birds, weather and ground clutters. Their data was obtained from Air Traffic Control. In fact, data were experimental were included the amplitude and phase of echo signals. Haykin et al in this reference extracted a set of best features which can differentiate among different clutter models. In reference [12] the radar target detection was done with Artificial Neural Network. Authors used Prony's algorithm to extract time-domain target features. Multi-Layer Perceptron and the Self Organizing Maps were utilized. These networks had been tested and each network had been trained on a wide range of SNR and with various data to appraise the training invariant traits of each network. Authors in [13] considered radar signal detection using Artificial Neural Networks in just K-distributed environment. They tested two training algorithms. Back propagation and Genetic algorithms for an MLP structure were used. In [14] authors present the problem of detecting targets in simulated land clutter. The clutter was modeled by Weibull distribution. Authors in this reference were tried to find a detection scheme to determine the target position easily. Because high-level clutter echoes were received, they proposed detection scheme based on Neural Network, where feedforward multilayer perceptron was used. Then, they compared their proposed scheme with a coherent detector commonly used for Weibull-distributed clutter and concluded the performance improvement achieved by using their proposed method. Reference [15] described the classification of radar returns Sea and ground clutters. Authors first explained an analysis of radar clutter data to validate the K amplitude distribution and the autoregressive modelling of the spectrum. Then, they briefly introduced a multi-layer neural network classifier. The Neural Network inputs were included the shape parameter of the K-distribution, the magnitude and the phase of the poles and the reflection coefficients which were calculated by using the Burg's algorithm. In [16] authors aimed to improve the detection radar performance in presence of snow clutter. They extracted suitable features which were occupied to separate targets and snow clutter. A Bayes classifier, a multilayer perceptron and a radial basis function network were tested and compared. In this paper, we classify the different types of radar clutters. The general procedure of this classification is shown in Fig.1.

Fig. 1. the general procedure of classification of radar clutters

In This paper, we use MLP method to classify four different kinds of clutters. We use five different training algorithms to form our neural networks. Using four clutters and different training algorithms were novel concepts and it was not done in any papers. We compared MLP and RBF to show that MLP is more suitable for clutter data. Fortunately the results prove this issue. These Three characteristics are the preferences of our paper.

This paper has following procedure: first radar clutter models which they are used as input data are introduced. Rayleigh, Log normal, Weibull and K-distribution clutters are modeled. In section 3, three suitable features for clutter data are described. Burg's reflection coefficients, Skewness and kurtosis are these three features. In next part, MLP as classifier is explained and 5 training algorithms for MLP-based classifier is described. Section 5 represents some results of simulations. These results show the validity of proposed MLP-based classifier. In this part also the results of simulations are compared with results of RBF-based classifier. In the last part, the conclusions of simulations and this research are mentioned.

2. Radar Clutter Models

This section introduces radar clutter models. These models can be used for sea and land and weather. Because clutters are variable and random echoes, statistical distributions are used to describe the characteristics of clutters.

2.1 Rayleigh Distribution

The probability density distribution function of Rayleigh distribution [17] is

$$f(x) = \frac{x}{\sigma_v^2} exp\left(-\frac{x^2}{2\sigma_v^2}\right) \; x \geq 0 \qquad (1)$$

Where x is clutter amplitude, σ_v is standard deviation of clutter. The distribution function is

$$F(x) = 1 - exp(-\left(\frac{x}{\sigma_v}\right)^2) \qquad (2)$$

In order to well describe the relationship of parameter σ_v and environment, let $\sigma_v = \frac{\sigma}{\lambda_0}$ into (1), we can get:

$$f(x) = \frac{x\lambda_0}{\sigma^2} exp\left(-\frac{x^2\lambda_0^2}{2\sigma^2}\right), \; x > 0 \qquad (3)$$

Where λ_0 is radar wavelength. Fig. 2 shows Rayleigh distribution with different parameters.

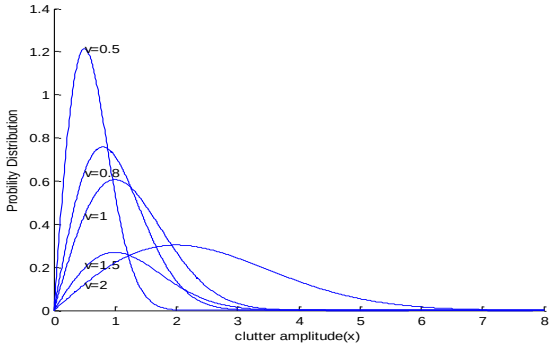


Fig. 2. Rayleigh PDF with different parameters

2.2 Weibull Distribution

Weibull distribution is used usually for modeling of clutter in low grazing angle.it can be used as weather, sea and land clutter. Weibull density function is:

$$f(x) = \frac{\beta x^{\beta-1}}{c^{\beta}} exp\left(\left(\frac{-x}{c}\right)^{\beta}\right) \;\; ; \;\; x \geq 0 \qquad (4)$$

Where β is shape parameter and c is scale parameter. Weibull distribution with different parameters is shown in Fig.3.

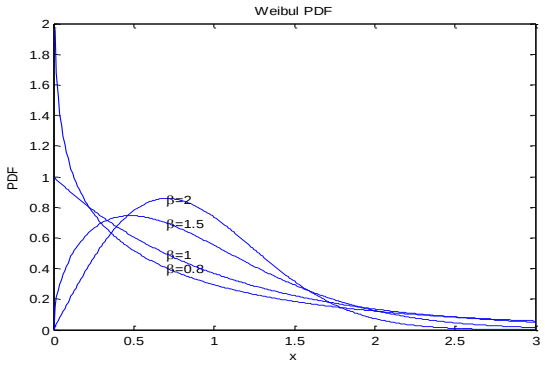


Fig. 3. Weibull PDF with different parameters

2.3 K-distribution

This probability distribution for modeling the statistics of clutter is described as a compound distribution that consists of two components the local power and the speckle component. The K-distribution [18] probability density function describing amplitude x is

$$f(x) = \frac{2b^{\frac{v+1}{2}} x^{\frac{v-1}{2}}}{\Gamma(v)} K_{v-1}(2\sqrt{bx}) \qquad (5)$$

This is characterized by a scale parameter, b, and a shape parameter, v. In Fig. 4 K-distribution with different shape parameters is shown.

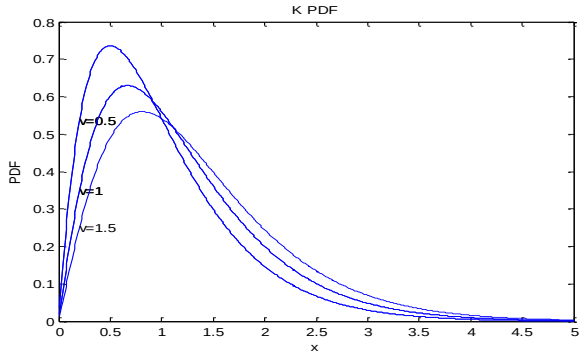


Fig. 4. K-distribution PDF with different shape parameters

2.4 Log Normal Distribution

One of the first models used to describe non Rayleigh clutters was Log normal [19] because it has longer tail than Rayleigh. In the Log normal probability density function the clutter echo power which is expressed in decibel (dB) is Gaussian. The log normal probability density function is:

$$f(x) = \frac{1}{\sqrt{2\pi}sx} exp[\frac{1}{2s^2}(ln\, x/x_m)^2] \;\; ; \; x \geq 0 \qquad (6)$$

Where s is standard deviation and x_m is average of x.

Fig.5 shows the variations of log normal pdf when 's' changes and $x_m = 1$.

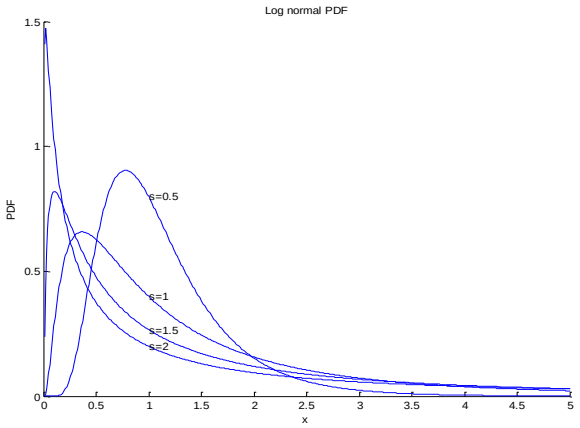


Fig. 5. Log normal PDF with' x_m=1'and different values of 's'

3. Feature Extraction

Feature extraction plays an essential role in each classification problems. It is necessary to extract the set of features which can be applied to distinguish the members of input types. The study of clutter statistical features has been performed to recognize the most suitable set to be used as classifier inputs. This is suitable way for controlling the computational cost and improving the capabilities of classifiers. In this paper we have considered four statistical distributions such as Log normal, Rayleigh, Weibull and K-distribution for classification. We use three features for radar clutters. Skewness and kurtosis as high order moments and Burg reflection coefficients which are described below.

3.1 Burg Reflection Coefficient

Burg's reflection coefficients [20] are utilized as spectral features for clutters. These coefficients are obtained from maximum entropy method (MEM) of spectral analysis [21].

The coefficients arise out of the lattice implementation of the prediction error filter (PEF) which attempts to minimize the prediction error power at each stage. This minimization results in a whitening filter and as such the reflection coefficients represent incremental predictable information extracted from the time series at each stage. Therefore we use burg's reflection coefficients to extract the best features. These coefficients are defined as [22]:

$$\rho_m = \frac{-2\sum_{i=1}^{L}\sum_{k=m+1}^{K} f_{m-1,i}(k)\, b^*_{m-1}(k-1)}{\sum_{i=1}^{L}\sum_{k=m+1}^{K}(|f_{m-1,i}(k)|^2 + |b_{m-1,i}(k-1)|^2)} \qquad (7)$$

Where $f_{m,i}(k)$ and $b_{m,i}(k)$ are the forward and backward prediction errors. They are obtained as:

$$f_{m,i}(k) = f_{m-1,i}(k) + \rho_m b_{m-1,i}(k-1) \qquad (8)$$

$$b_{m,i}(k) = b_{m-1,i}(k-1) + \rho_m^* f_{m-1,i}(k) \quad (9)$$

The asterik in equations (7) and (9) represent complex conjugation. For a specified index i, the prediction errors $f_{0,i}(k)$ and $b_{0,i}(k)$ are intialized with the input data as follows:

$$f_{0,i}(k) = b_{0,i}(k) = x_i(k) \quad (10)$$

For each of the L lattice filters, the index i implies to i^{th} time series part of length K. m^{th} part of the prediction-error filter is also determined with m and the k^{th} sample in a time series is shown with k.

Although Burg's reflection coefficients are features which are used for phase of clutter data but it is common to use magnitude of $|\rho_1|$ directly as the feature of amplitude of clutter data. So in this paper, we use $|\rho_1|$ as feature for our clutter data, because our input data are amplitude of log normal, Rayleigh, weibull and K-distribution.

3.2 Skewness and Kurtosis

In addition to mathematical description, Skewness and kurtosis have physical meanings [23]. The skewness represents the asymmetry of the distribution from its mean and kurtosis measures how peaky or flat with respect to Gaussian distribution.

In this paper we apply these two high order moment as the feature of amplitude of radar clutters. They are defined as:

$$Skewness(x) = \sum_M \frac{(x-\mu(x))^3}{\sigma^3} \quad (11)$$

$$Kurtosis(x) = \sum_M \frac{(x-\mu(x))^4}{\sigma^4} \quad (12)$$

Where μ refers to average and σ shows the standard deviation of x.

4. Classifiers

This section explains the classifiers which are used in this paper.

4.1 Multi layer Perceptron Network (MLP)

In this paper, we have utilized MLP Neural Networks as classifiers. An MLP neural network includes various layers. An input layer of source nodes, one or more hidden layers of computation nodes (neurons) and output layers. It should be mentioned that each layer is fully connected to next one. Inputs are spread through the network layer by layer and MLP gives a non-linear mapping of the inputs at output layers.Fig.6 shows MLP topology [24]. The recognition basically consists of two training and testing phases. In training stage, weights are calculated according to the chosen learning algorithm. Learning algorithms and their speeds are very essential problems for MLP. The aim of training is to minimize the global error (E) which is defined as:

$$E = \frac{1}{P}\sum_{p=1}^{P} E_p \quad (13)$$

Where P is the total number of training patterns and E_p is the error for training pattern *(p)*. E_p is obtained by below formula:

$$E_p = \frac{1}{2}\sum_{i=1}^{N}(o_i - t_i)^2 \quad (14)$$

Where N is the total number of output neurons, o_i is the network output at the i^{th} output neuron and t_i is the target output at the i^{th} output neuron. In every training algorithm, the objective is to decrease this global error by adjusting the weights and biases.

One of the popular learning algorithm is Back Propagation (BP) [25]. In BP, a simple gradient descent algorithm updates weight values by using following formula[26]:

$$w_{k+1} = w_k - a_k\, g_k \quad (15)$$

Where, g_k is the current gradient, a_k is learning rate and w_k is the vector of current weights.

Although BP is still popular, in some conditions, BP network classifiers generate non-robust responses and converge to local mininmum. New algorithms have been introduced for training stage. However, some algorithms needs much computing power to get good training especially when dealing with a large training set.

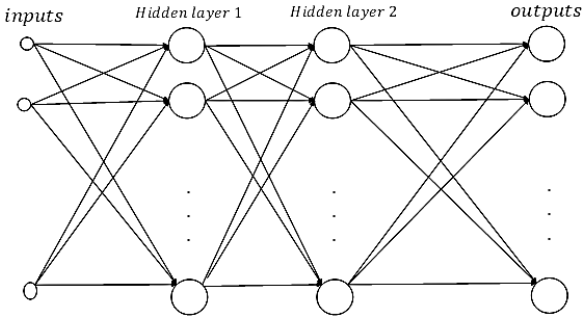


Fig. 6. MLP topology [24]

In this paper following learning algorithms are considered.

4.1.1 Levenberg-Marquardt Algorithm

The objective of designing Levenberg-Marquardt (LM) algorithm was to achieve second order training speed without having to calculate the Hessian matrix [26]. When the performance function has the form of a sum of squares, then then Hessian matrix can be approximated from following formula:

$$H = J^T J \quad (16)$$

And also gradient is calculated as:

$$g = J^T e \quad (17)$$

Where J is the Jacobian matrix, which cosists first derivatives of the network errors with respect to the weights and biases, and e is a vector of network errors.

The Levenberg-Marquardt algorithm [27] utilizes this approximation to the Hessian matrix in the following Newton-like update:

$$w_{k+1} = w_k - [J^T J + \mu I]^{-1} J^T e \quad (18)$$

Where J is jacobian matrix, e is a vector of network errors and μ is a constant.

4.1.2 Conjugate Garadient Algorithm

The basic back propagation algorithm controls the weights in the steepest descent direction (the most negative of the gradients). This is the direction in which the performance function is declining very quickly. Though the function declines most swiftly along the negative gradient, this does not necessarily generate the fastest convergence. In the conjugate gradient algorithms, searching is done along conjugate directions, which converge faster than steepest descent directions.

Conjugate gradient algorithms commence by searching in the steepest descent direction on the first iteration.

$$p_0 = -g_0 \quad (19)$$

A line search is then done to choose optimal distance to move along the current search direction:

$$w_{k+1} = w_k + a_k p_k \quad (20)$$

Then the next search direction is chosen so that it is conjugate to previous search directions. Combining the new steepest descent direction with the previous search direction is popular method for determining the new search direction:

$$p_k = -g_k + \beta_k \, p_{k-1} \quad (21)$$

The way in which β_k is computed with Fletcher-Reeves update as:

$$\beta_k = \frac{g_k^T \, g_k}{g_{k-1}^T \, g_{k-1}} \quad (22)$$

Above formula is the ratio of the norm squared of the current gradient to norm squard of the previous gradient [28].

4.1.3 Resilient Back-Propagation (RPROP) Algorithm

RPROP considers the sign of derivatives as the indication for the direction of the weight update [29]. In doing so, the size of the partial derivative does not influence the weight step. The following equation shows the adaptation of the update values of Δ_{ij} (weight changes) for the RPROP algorithm. For initialization, all Δ_{ij} are set to small positive values:

$$\Delta_{ij}(t) = \begin{cases} \eta^+ * \Delta_{ij}(t-1) ; & if \frac{\delta E}{\delta w_{ij}}(t-1) \frac{\delta E}{\delta w_{ij}}(t) > 0 \\ \eta^- * \Delta_{ij}(t-1) ; & if \frac{\delta E}{\delta w_{ij}}(t-1) \frac{\delta E}{\delta w_{ij}}(t) < 0 \\ \eta^0 * \Delta_{ij}(t-1) ; & otherwise \end{cases} \quad (23)$$

Where $\eta^0 = 1$, $0 < \eta^- < 1 < \eta^+$ and $\eta^{-,0,+}$ are known as the update factors. Whenever the derivative of the corresponding weight changes its sign, it implies that the previous update value is too large and it has skipped a minimum. Therefore, the update value is then reduced η^- as shown above. However, if the derivative retains its sign, the update value is η^+ increased. This will help to accelerate convergence in shallow areas. To avoid over- acceleration, in the epoch following the application of η^+, the new update value is neither increased nor decreased (η^0) from the previous one. Note that the values of Δ_{ij} remain non-negative in every epoch. This update value adaptation process is then followed by the actual weight update process, which is governed by the following equations:

$$\Delta_{ij}(t) = \begin{cases} -\Delta_{ij}(t) ; & if \frac{\delta E}{\delta w_{ij}}(t) > 0 \\ +\Delta_{ij}(t) ; & if \frac{\delta E}{\delta w_{ij}}(t) < 0 \\ 0 ; & otherwise \end{cases} \quad (24)$$

Weight values are updated with below formula:

$$w_{ij}(t+1) = w_{ij}(t) + \Delta_{ij}(t) \quad (25)$$

The update values and weights are changed every time the whole pattern set has been presented once to the network (learning by epoch).

4.1.4 BFGS Algorithm

Newton's method is an alternative to the conjugated gradient methods for fast optimization. The basic step of Newton's method is:

$$w_{k+1} = w_k - A^{-1} g_k \quad (26)$$

Where A_k is the Hessian matrix of performance index at the current values of the weights. Because of high computational cost of the Hessian matrix, usually some of algorithms which don not need to the computation of second derivatives are introduced. These are called Quasi-Newton (or secant) method. The quasi-Newton method, which has been most successful in published studies, is the Broyden, Fletcher, Goldfarb and shanno (BFGS) update [30].

4.1.5 One Step Secant (OSS) Algorithm

The one step secant (OSS) method is an attempt to bridge the gap between the conjugate gradient algorithms and the quasi-Newton algorithms. OSS algorithm does not save the complete Hessian matrix, it assumes that at each iteration, the previous Hessian was the identity matrix.

4.2 Radial Basis Function (RBF) Network

Radial basis function neural networks (RBFN) are popular tools for multivariate approximation, time series forecasting, image processing, speech recognition, classification and etc., since their properties of localization, robustness and stability [31]. The basic structure of a RBFN is a two layer, feed-forward network in which the activation functions of the neurons of the hidden layer are radial basis functions (RBF). Each hidden neuron calculates the distance from its input to the neuron's central point, c, and applies the RBF to that distance. The neurons of the output layer perform a weighted sum between the outputs of the hidden layer and weights of the links that connect both output and hidden layer, in other words linear function is existed between the hidden layer and the output layer:

$$h_i(x) = \phi(\frac{||x - c_i||^2}{r_i^2}) \quad (27)$$

$$f_j(x) = \sum w_{ij} h_i(x) + w_0 \quad (28)$$

where x is the input, ϕ is the RBF, c_i is the center of the i^{th} hidden neuron, r_i is its radius, w_{ij} is the weight links that connect hidden neuron number i and output neuron number j, and w_0 is a bias for the output neuron.

The problem of automatic RBFN design is an important subject. The original regularization RBF theory, proposed that the number of basis functions should be equal to the number of training samples. The basis functions are centered on the training samples and the only unknown parameters are the linear weights, which can be determined efficiently by solving the system of linear equations. However, the resulting networks are complex and often ill-conditioned. Generalized RBFNs are designed with fewer nodes than there are samples in the training set, which results in less complex networks. However, the number of basis functions, their centers and widths, have to be determined. In this paper we have proposed an efficient method based on evolutionary RBF neural networks by implementing improved bees algorithm. The aim of this model is to fit a given data set with sufficient accuracy, and more importantly, generalizes well to unseen data while the neural network is maintaining a reasonable size.

5. Simulation Results

This section represents some of the simulation results of the proposed method for classification of radar clutters. We have used MATLAB as simulator.

In this paper, we design a classifier by using artificial neural networks. The first step in this classifier is producing suitable data set. As mentioned, our input data includes radar clutters. Because our works were done in university and educational environment, we do not access to real and experimental clutter data. So, like other papers, we generate clutters with SIRP and ZMNL methods. These two methods are very common for generate clutter data. We produce 12000 clutter patterns. Log normal, Rayleigh, Weibull and K-distribution were four clutters which are simulated. Our input data were included:

a) Log normal: 3000 patterns with unity mean and standard deviation equal to 0.9.

b) Rayleigh: 3000 patterns with unity standard deviation

c) Weibull: 3000 patterns which its shape parameter is 1.8 and scale parameter is 1.2.

d) K-distribution: 3000 patterns with shape parameter equal to 2 and unity scale parameter.

After generation of input data, we extract their suitable features. In this simulation, we use skewness and kurtosis and $|\rho_1|$ as feature for clutter data.

Then we give these suitable input data to classifier based on MLP neural networks. We design various MLP classifier with different number of hidden layers and neurons. Another characteristic of our work is that we train our data with different learning algorithms. Table 1 shows the list of algorithms which we use for training. In all conditions, we choose 8000 patterns of input data as training data and others 4000 patterns as test data. Since we prove that our proposed method is valid and has high accuracy, we compare it with RBF neural network. All results show that the proposed method has high accuracy.

Table 1: Five different training algorithms used for training of MLP neural networks

Algorithm	Acronym
Resilient	RP
Scaled Conjugate Gradient	SCG
Broyden, Fletcher, Goldfarb and Shanno (BFGS) Quasi-Newton	BFGS
One Step Secant Quasi-Newton	OSS
Levenberg-Marquardt	LM

Because the results of artificial neural network are random, we repeat simulations for 10 iterations and put the average of values in following tables. Note that all of values in tables are respect to percentage. In these experiments, we test our proposed MLP-based classifier in various conditions. First, for different hidden layers which have various neurons we have examined our classifier. Theses simulation were done in four cases. In first case, classifier has two hidden layers which one of them has 20 neurons and another has 15 neurons. In second condition, MLP-based classifier also has two hidden layers but first layer has 30 and second one has 15 neurons. In two other cases, we design one hidden layer for classifier and in each layer are 25 or 35 neurons. All of these cases were repeated for 5 different training algorithms and MLP was trained by LM, RP, OSS, SCG and BFGS training algorithms. The results of all these four cases are shown in tables 2 to 5.

Table 2. confusion matrix of proposed MLP-based classifier with 2 hidden layers and (20*15) neurons

Training algorithm		Rayleigh	Log normal	weibull	K dist.
LM	**Rayleigh**	**98.6%**	0	0.4%	1%
	Log normal	0	**93.2%**	6.8%	0
	weibull	0.6%	6.4%	**93%**	0
	K dist.	0.2%	0	0	**99.8%**
RP	**Rayleigh**	**91.8%**	2.3%	0	5.9%
	Log normal	0	**93.5%**	6.5%	0
	weibull	0	9%	**89.7%**	1.3%
	K dist.	0	0.1%	1.6%	**98.3%**
BFGS	**Rayleigh**	**94%**	0	0	6%
	Log normal	0	**90%**	10%	0
	weibull	0.7%	10.1%	**89.2%**	0
	K dist.	1.8%	0.5%	0	**97.7%**
OSS	**Rayleigh**	**99%**	0	0	1%
	Log normal	0	**96.3%**	3.7%	0
	weibull	0	8%	**90.7%**	1.3%
	K dist.	0.1%	0	2.3%	**97.6%**
SCG	**Rayleigh**	**99.1%**	0	0	0.9%
	Log normal	0	**93.7%**	6.3%	0
	weibull	0	9.3%	**90.4%**	0.3%
	K dist.	0.4	0	0.7	**98.9%**

Table 3. confusion matrix of proposed MLP-based classifier with 2 hidden layers and (30*15) neurons

Training algorithm		Rayleigh	Log normal	weibull	K dist.
LM	**Rayleigh**	**95.4%**	0	0	4.6%
	Log normal	0	**94.5%**	5.5%	0
	weibull	0.1%	6.4%	**93.5%**	0
	K dist.	0	0.3%	0.5%	**99.2%**
RP	**Rayleigh**	**95.2%**	0	1.2%	3.6%
	Log normal	0	**94.1%**	5.9%	0
	weibull	0	9.2%	**90.8%**	0
	K dist.	1.2%	0	0.7%	**98.1%**
BFGS	**Rayleigh**	**99.3%**	0	0	0.7%
	Log normal	0	**96.3%**	3.7%	0
	weibull	0	7.9%	**92.1%**	0
	K dist.	0	1.4%	0	**98.6%**
OSS	**Rayleigh**	**99.8%**	0	0.2%	0
	Log normal	0	**96.4%**	3.6%	0
	weibull	0.1%	6.4%	**93.5%**	0
	K dist.	0.2%	0.4%	0	**99.4%**
SCG	**Rayleigh**	**99.1%**	0.9%	0	0
	Log normal	0	**92.5%**	7.5%	0
	weibull	0.7%	8.2%	**91.1%**	0
	K dist.	0	0.1	0.2	**99.7%**

Table 4. confusion matrix of MLP classifier with 1 layer and 25 neurons

Training algorithm		Rayleigh	Log normal	weibull	K dist.
LM	**Rayleigh**	**91.9%**	0	6.2%	1.9%
	Log normal	0	**92.4%**	7.6%	0
	weibull	0.1%	7.3%	**89.2%**	3.4%
	K dist.	0	0.6%	0.7%	**98.7%**
RP	**Rayleigh**	**98.1%**	0	0	1.9%
	Log normal	0	**97.8%**	2.2%	0
	weibull	1%	8.4%	**90.6%**	0
	K dist.	6.8	0	0	**93.2%**
BFGS	**Rayleigh**	**99.2%**	0	0	0.8%
	Log normal	0	**98.8%**	1.2%	0
	weibull	1.6%	0	**90.2%**	8.2%
	K dist.	0	2.3%	0.4%	**97.3%**
OSS	**Rayleigh**	**99.4%**	0	0.1	0.5%
	Log normal	0	**96.1%**	3.9%	0
	weibull	1.2%	5.3%	**93.5%**	0
	K dist.	0	0.4%	1.6%	**98%**
SCG	**Rayleigh**	**99.8%**	0.2%	0	0
	Log normal	0	**95%**	5%	0
	weibull	0	6.6%	**92.6%**	0.8
	K dist.	4.3%	0.3%	0	**95.4%**

Table 5. confusion matrix of proposed MLP-based classifier with 1 hidden layer and (35) neurons

Training algorithm		Rayleigh	Log normal	weibull	K dist.
LM	**Rayleigh**	**98.4%**	0	0	1.6%
	Log normal	0	**93.6%**	6.4%	0
	weibull	1.1%	5.6%	**93.3%**	0
	K dist.	0	2.3%	0	**97.7%**
RP	**Rayleigh**	**96.3%**	0	0	3.7%
	Log normal	0	**95.4%**	4.6%	0
	weibull	0.5%	7.4%	**92.1%**	0
	K dist.	0	0	1.2	**98.8%**
BFGS	**Rayleigh**	**99.8%**	0	0	0.2%
	Log normal	0	**94.7%**	5.3%	0
	weibull	0	8.3%	**91.1%**	0.6%
	K dist.	3.6%	0.6%	0	**95.8%**
OSS	**Rayleigh**	**98.8%**	0.8%	0.4%	0
	Log normal	0	**96.6%**	3.2%	0.2%
	weibull	8.4%	0	**89.3%**	2.3%
	K dist.	0.6%	0.8%	0	**98.6%**
SCG	**Rayleigh**	**99.2%**	0	0	0.8%
	Log normal	0	**96%**	4%	0
	weibull	4.8%	3.4%	**91.8%**	0
	K dist.	0	1.6%	5%	**97.9%**

It is realizable from tables 2 to 5 that the proposed MLP-based classifier can separate radar clutters very successfully. Because in each case, the percentage of recognition of output is very high and all of them are almost more than 90%. These values prove the validity and accuracy of proposed technique.

For emphasizing on accuracy of proposed classifier, we compared it with RBF neural network. The confusion matrix of RBF-based classifier for radar clutters is shown in table 6. Although the results show that RBF-based classifier is also good but MLP-based classifier is much better than it and has a higher accuracy.

Table 6. confusion matrix of proposed RBF-based classifier

		Rayleigh	Log normal	weibull	K dist.
RBF	**Rayleigh**	**93.5%**	1.1%	0	5.4%
	Log normal	2.4%	**89.3%**	0	8.3%
	weibull	0	0.6%	**99.1%**	0.3%
	K dist.	1.4%	2.2%	4.9%	**91.5%**

In table 7, the comparison of results of proposed MLP-based method with different training algorithms and hidden layers with the results of RBF-based classifier are mentioned. The values of this table represent that proposed technique is more efficient.

Table 7. comparison of the accuracy of proposed MLP-based classifier and RFB-based classifier

Hidden layers	Training algorithm	% Accuracy of classifier
20* 15	**LM**	96.15%
30*15		95.65%
25		93.05%
35		95.75%
20* 15	**RP**	93.325%
30*15		94.55%
25		94.925%
35		95.65%
20* 15	**BFGS**	92.725%
30*15		96.575%
25		96.375%
35		95.35%
20* 15	**OSS**	95.9%
30*15		97.275%
25		96.75%
35		95.825%
20* 15	**SCG**	95.525%
30*15		95.6%
25		95.7%
35		96.225%
RFB		93.35%

6. Conclusions

Classification of radar clutters is very essential issue in radar researches. So in this paper is tried to classify four important models of clutter. Rayleigh, Log normal, Weibull and K distribution are these four models. Since to decrease the complexity of the classifier, a feature extraction has been designed for providing the classifier inputs. The proposed MLP-based classifier could classify clutters successfully. The results show that more than 90%, the proposed classifier was successful. The results of proposed technique were compared with the results of RBF-based method. All results prove the validity of proposed method.

Future researches will be focused on radar clutter classification with other kinds of Neural Networks and Soft computing algorithms. We will improve the Neural Networks by using soft computing algorithms like Genetic and Ant Algorithms. In the next works we will use some other features which can describe clutters better.

References

[1] J. Anderson, M. T. Gately, P. A. Penz, D. R. Collins, and others. "Radar signal categorization using a neural network," Proceedings of the IEEE, 1990 ,vol. 78, no. 10, pp. 1646–1657.

[2] J.-H.Lee, I.-S.Choi, H.-T.Kim. "Natural frequency-based neural network approach to radar target recognition," IEEE Trans. Signal Process, vol. 51, no. 12, p. 3191, December 2003.

[3] R. Rouveure, P. Faure, and Monod, "Multi-layer feed-forward perceptron for microwave signals processing," in Geoscience and Remote Sensing, Symposium. IGARSS '03, 2003, vol. 6, pp. 3519-3521.

[4] O. L. Mangasarian and W. H. Wolberg. Cancer diagnosis via linear programming. University of Wisconsin-Madison: Computer Sciences Department, 1990.

[5] T. L. Fine, Feedforward neural network methodology. Springer Science & Business Media, 2006.

[6] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," . The bulletin of mathematical biophysics, vol. 5, no. 4, pp. 115–133, 1943.

[7] S. Mirjalili, S. Z. M. Hashim, and H. M. Sardroudi. "Training feedforward neural networks using hybrid particle swarm optimization and gravitational search algorithm," Applied Mathematics and Computation, vol. 218, no. 22, pp. 11125–11137, 2012.

[8] S. Mirjalili and S. Z. M. Hashim. "A new hybrid PSOGSA algorithm for function optimization," in Computer and information application (ICCIA), 2010 international conference on, 2010, pp. 374–377.

[9] Z. X. Guo, W. K. Wong, and M. Li, "Sparsely connected neural network-based time series forecasting," Information Sciences, vol. 193, pp. 54–71, 2012.

[10] P. Auer, H. Burgsteiner, and W. Maass, "A learning rule for very simple universal approximators consisting of a single layer of perceptrons," Neural Networks, vol. 21, no. 5, pp. 786–795, 2008.

[11] Haykin and C. Deng, "Classification of radar clutter using NN," IEEE Trans. Neural Netw., vol. 2, November 1991.

[12] R. Soleti, L. Cantini, F. Berizzi, A. Capria, and D. Calugi, "Neural Network for polarimetric radar target classification," in Signal Processing Conference, 14th European, 2006, pp. 1–5.

[13] K. Cheikh and F. Soltani, "Application of neural networks to radar signal detection in k-distributed clutter," Radar, Sonar and Navigation, IEE Proceedings, vol. 153, no. 5, pp. 460-466, 2006.

[14] R. Vicen-Bueno, M. Rosa-Zurera, M. P. Jarabo-Amores, and R. Gil-Pita, "Automatic target detection in simulated ground clutter (Weibull distributed) by multilayer perceptrons in a low-resolution coherent radar," Radar, Sonar & Navigation, IET, vol. 4, no. 2, pp. 315–328, 2010.

[15] C. Bouvier, L. Martinet, G. Favier, H. Sedano, and M. Artaud, "Radar clutter classification using autoregressive modelling, K-distribution and neural network," in Acoustics, Speech, and Signal Processing, ICASSP-95., International Conference on, 1995, vol. 3, pp. 1820–1823.

[16] L. Pierucci and L. Bocchi, "Improvements of radar clutter classification in air traffic control environment," in Signal Processing and Information Technology, IEEE International Symposium on, 2007, pp. 721–724.

[17] L. Teng, H. Dan, "Model for spatial correlated clutter and its application to temporal spatialcorrelated clutter", IET Microwaves, Antennas & Propagation, Vol. 5 ,No. 3, pp. 298-304, 2011.

[18] K. D. Ward, S. Watts, and R. J. Tough, "Sea clutter: scattering, the K distribution and radar performance", IET, vol. 20, 2006.

[19] A. Farina, A. Russo, and F. A. Studer, "Coherent radar detection in log-normal clutter," Communications, Radar and Signal Processing, IEE Proceedings F,1986, vol. 133, no. 1, pp. 39–53.

[20] W. Stewing. Parametric spectral analysis of radar clutter. McMaster University, 1983.

[21] J. P. Burg, "A new analysis technique for time series data," presented at the NATO Advanced Study Institute on Signal Processing with Emphasis on Underwater Acoustics, Enschede, The Netherlands, 1968.

[22] S. S. Haykin, Adaptive filter theory. Pearson Education India, 2008.

[23] J. R. Barry, B. K. Carter, R. J. Erdahl, R. L. Harris, J. T. Miller, G. D. Smith, and R. M. Barnes, "Angel clutter and the ASR air traffic control radar," Applied Physics Laboratory, John Hopkins University, Silver Spring, MD, Final Report under Federal Aviation Administration Contract DOT-FA72WA-2705, 1973.

[24] M. Kubat, Neural networks: a comprehensive foundation by Simon Haykin, Macmillan: Cambridge Univ Press, 1999.

[25] H. Adeli and S.-L. Hung, Machine learning: neural networks, genetic algorithms and fuzzy systems. John Wiley & Sons, Inc., 1994.

[26] J. J. Moré, "The Levenberg-Marquardt algorithm: implementation and theory," in Numerical analysis, Springer, pp. 105–116, 1978.

[27] M. T. Hagan and M. B. Menhaj, "Training feedforward networks with the Marquardt algorithm," Neural Networks, IEEE Transactions on, vol. 5, no. 6, pp. 989–993, 1994.

[28] R. Fletcher and C. M. Reeves, "Function minimization by conjugate gradients," The computer journal, vol. 7, no. 2, pp. 149–154, 1964.

[29] A. Ebrahimzadeh, A. Khazaee, "An efficient technique for classification of electrocardiogram signals, Advances in Electrical and Computer Engineering", Volume 9, pp. 89-93, 2009.
[30] M. Riedmiller and H. Braun, "A direct adaptive method for faster backpropagation learning: The RPROP algorithm," in Neural Networks, IEEE International Conference on, 1993, pp. 586–591.
[31] S. Haykin, Neural Networks: A Comprehensive Foundation. New York: MacMillan, 1999.

Mohammad Akhondi Darzikolaei received the B.Sc and M.Sc degree from Babol Noshirvani University of technology. He worked on radar clutters and radar signal processing. He is now Telecommunication Ph.D student in Babol Noshirvani University of technology and his researches focus on radar and sonar signal processing, speech processing and pattern recognition and artificial neural networks.

Ata Ebrahimzade received the Ph.D degree in electrical engineering from Ferdosi University, Mashhad, Iran in 2007.
He is currently Professor with the Faculty of Electrical and Computer Engineering, Babol University of Technology, Babol, Iran. He has authored or co-authored more than 70 papers in international journals and conferences. His current research interests include signal processing and artificial intelligence.
Dr. Ebrahimzade is a reviewer of international conferences and journals.

Elahe Gholami received the B.Sc and M.Sc degree from Babol Noshirvani University of technology. She worked on cognitive radio spectrum sensing. Her researches focus on WSN, Cognitive radio, Signal processing, artificial neural networks and soft computing algorithms.

Speech Emotion Recognition Based on Fusion Method

Sara Motamed
Department of Computer Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran
samotamed@yahoo.com
Saeed Setayeshi*
Department of Medical Radiation, Amirkabir University of Technology, Tehran, Iran
setayesh@aut.ac.ir
Azam Rabiee
Department of Computer Science, Dolatabad Branch, Islamic Azad University, Isfahan, Iran
azrabiee@gmail.com
Arash Sharifi
Department of Computer Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran
a.sharifi@srbiau.ac.ir

Received: 27/Jul/2016 Revised: 07/Jan/2017 Accepted: 14/Jan/2017

Abstract

Speech emotion signals are the quickest and most neutral method in individuals' relationships, leading researchers to develop speech emotion signal as a quick and efficient technique to communicate between man and machine. This paper introduces a new classification method using multi-constraints partitioning approach on emotional speech signals. To classify the rate of speech emotion signals, the features vectors are extracted using Mel frequency Cepstrum coefficient (MFCC) and auto correlation function coefficient (ACFC) and a combination of these two models. This study found the way that features' number and fusion method can impress in the rate of emotional speech recognition. The proposed model has been compared with MLP model of recognition. Results revealed that the proposed algorithm has a powerful capability to identify and explore human emotion.

Keywords: Speech Emotion Recognition; Mel Frequency Cepstral Coefficient (MFCC); Fixed and Variable Structures Stochastic Automata; Multi-constraint; Fusion Method.

1. Introduction

Emotion recognition refers to the ability of detecting humans' feelings and conditions. It is also one of the most efficient methods of analyzing information collected from humans to identify the interaction between man and machines [1,2]. Most researches of recognition have focused either on its recognition or classification [3]. Seehapoch et al. have claimed that there are two elements in the speaker feeling including prosodic and spectral which may influence on speech emotion recognition since both of them contain emotional information [4]. Many researchers have tried to separate the speech features including pitch, energy, frequency, formant and vibration [5].

There are many debates regarding identification of speech emotion recognition about having insufficient knowledge for identification of speech sounds, lack of powerful database and even different accents and dialect expression. Hozjan et al. have assumed that there should be no difference in the culture of the speakers, so they ignored cultural changes in their behavior in analyzing problems associated with emotional speech recognition [6]. Cahn proved that acoustic features including intonation, sound quality and enumeration have powerful relationship with speech recognition [7].

To increase the rate of recognition, some researchers combined the extracted features for speech recognition. Zhang et al. used the Gaussian mixture model (GMM), Mel frequency Cepstral coefficient (MFCC) and autocorrelation function coefficients (ACFC) in the feature extraction level. Their results indicated a suitable rate of recognition [8]. Similarly, Bojanic et al. employed a fusion method of speech emotion recognition to identify the speaker feelings through the use of neural network method [9]. Since identification of speech recognition is an important problem to extract speech meaning, this paper uses a special fusion method to classify emotional speech through using the parallel stochastic learning automata.

This paper introduces a new classification model which employs multi-constraints with the use of stochastic learning automata to recognize speech emotion. The rest of the paper is organized as follows. Section 2 discusses about feature extraction methods. The proposed method and stochastic learning automata are explained in sections 3 and 4, respectively. The test results and conclusion are given in sections 5 and 6, respectively.

2. Features Extraction

2.1 Mel Frequency Cepstrum Coefficients (MFCC)

One of the most important steps in speech emotion recognition is to extract suitable features. Speech features are divided into the four main groups and numerous

* Corresponding Author

methods are introduced by researchers to obtain a powerful feature extractor. These four groups include speech continuous, qualitative, spectral and TEO based classes [4]. Price worked on the speech sound and energy features and grouped these under spectral classes [10]. MFCC algorithm is one of the efficient methods grouped under spectral features [10].

Many useful features can be extracted from the speech signals such as energy, MFCC (Mel frequency cepstral coefficients), LPC (linear predictive coding) and so on. This set of features has important information for discriminating different types of emotions [11] [12]. In this work, we have selected the MFCC to extract the emotional features [2,13]. This method includes the following mathematical approaches:

Step 1- Pre-emphasize:

The signal passes a filter of high frequency emphasized by equation (1) [14].

$$Y[n] = X[n] - 0.95\, X[n-1] \quad (1)$$

Step 2- Framing:

To classify speech samples, analogue conversations are changed into digital conversations in small frames length of 20 to 40 ms.

The speech signal is divided in N frames. Vicinity of separated frames by M is (M<N) where M=100 and N=256 [14].

Step 3- Hamming Window:

Windows are defined with W(n) and $0 \leq n \leq N-1$. N is the number of samples in each frame and Y[n] is the output signal. Input signals are shown by X(n) and windowing results are shown in equations (2) and (3) [14].

$$Y(n) = X(n) \times W(n) \quad (2)$$

$$w(n) = 0.54 - 0.4\cos\left[\frac{2\Pi n}{N} - 1\right]$$
$$0 \leq n \leq N-1 \quad (3)$$

Step 4- Fast Fourier Transform:

Each frame is changed into frequency amplitude from time amplitude with N. The Fourier Transform is used to reverse pulse complexity related to glottal U[n] and instigation of vocal tract H[n] in time domain and is calculated by equation (4), [14].

$$y(w) = FFT\,[h(t) * X(t)] = H(w) * X(w) \quad (4)$$

Step 5- Processing Mel Filter bank:

Cepstral coefficients are mainly obtained from the output of a set of filter banks, which suitably cover the full range of defined frequency spectrum. Generally the set of 24 buffer filters were used, which are similar to human ear performance. Filters were placed along the frequency axes variably. More filters were allocated to part of the spectrum below 1 KHz since they have more information about sound. Filtering is referred to as conceptual weighting.

Filter outputs are a set of their filtered spectral components: equation (5) indicates calculation of Mel for given frequencies in Hz [14]:

$$F(mel) = [2595 * \log 10\,[1 + f]700] \quad (5)$$

Step 6- Cos transform:

This step is a process of transferring Mel spectrum to time span using discrete Cosine transform (DCT). Results are Cepstral coefficients for Mel frequency and the set of coefficients are referred to as acoustic vectors [14]. The obtained features vector is put into the next step for partitioning. The block diagram for our recommended model is shown in fig 1.

2.2 Auto Correlation Function Coefficients (ACFC)

Auto Correlation Function of periodic signal generates a maximal value when the delay equals to the function cycle [8]. So, this method can find the maximum value pith period of the signal. The autocorrelation function calculated by equation 6:

$$p(k) = \lim_{n\to\infty} \frac{1}{2N+1} \sum_{-\infty}^{+\infty} x(m)x(m+k) \quad (6)$$

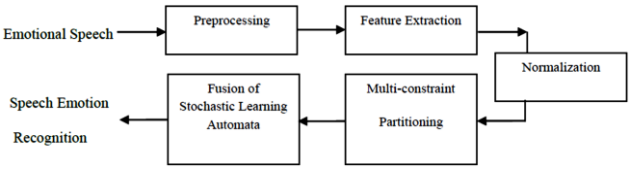


Fig. 1 Recommended system block diagram

Unvoiced signal and its autocorrelation function have no periodicity, and no significant peak. P (k) rapidly decays while k increases. Voiced signal has a quasi-periodicity, and its auto-correlation function P (k) has the same period with k [8].

3. Multi- constraint Partitioning

In order to classify obtained features, the MFCC features are used to allocate collected features of speech emotion recognition signals to the groups with the highest similarity. The proposed model is an efficient approach to allocate P features vectors values obtained by MFCC, with the |P| elements, to N classes (N is the number of people in our database) where each class has Ni Nodes with the certain capacity (Ni the number of different actions of each person). This means that each set of feature vectors has limited capacity considering constraints on the connections. External and internal equations to explore limitations and relationships are presented here.

Xi,n is an index which has a value of either 0 or 1 ($X_{i,n} \epsilon\{0,1\}$) and if features vector P_i is allocated to N_n node, then its quantity would be 1 otherwise it would be 0 [12]. External relationship is said to be a node with the supposition that P_i has been allocated to this specific Node [12]. Then the

external connection of P_i process is then applied to all feature vectors for P_j, $\sum_{j=1}^{|P|}(1-X_{i,n})w_{i,j}$. If each P_j process is allocated to N_n node then $X_{j,n}=1$ and this process has no participation with the above sum. Equation 7 calculates the external relationship of the nodes [12].

$$\sum_{i=1}^{|P|}\sum_{j=1}^{|P|}(X_{i,n}-X_{j,n}X_{j,n})W_{i,j} \le 1$$
$$n=1,\dots,|N| \quad (7)$$

The above formula divides features vectors sets into the subsets and adds the connection from one set to another. The only guiding quantity is that of $w_{i,j}$ connection which indicates connection from P_i to the node P_j (remote features vectors which is not connected to the node). Such internal connection may therefore be obtained by a similar formula. However, by adding the connection to $w_{j,i}$ feature vector to the node from remote feature vector the result of which would be equation (8), [12].

$$\sum_{i=1}^{|P|}\sum_{j=1}^{|P|}(X_{i,n}-X_{j,n}X_{j,n})W_{j,i} \le 1$$
$$n=1,\dots,|N| \quad (8)$$

The set of limitations in equation (8), limits the total calculation time for feature vectors allocated to each node with normalized capacity of node. The set of limitations in equation (9) assures that each feature vector would be placed on one node only [12].

$$\sum_{i=1}^{|P|}X_{i,n}\square_i \le 1$$
$$n=1,\dots,|N| \quad (9)$$

$$\sum_{i=1}^{|N|}X_{i,n} \le 1$$
$$n=1,\dots,|P \quad (10)$$

4. Stochastic Learning Automata

4.1 Fixed Structure Stochastic Automata (FSSA)

In an identification cycle, an automaton would select a behavior considering the reward and penalty received from the environment [15]. The automaton then uses the collected answer and the knowledge of former behavior toward defining the next measure. The objective of learning automaton is to perform an optimized measure beyond permitted behaviors. The automaton matches with the environment by learning optimal operation. The node with the most trespass from equation 7 – external connection for all vectors- is selected for pairing and computing of w_i. A feature vector, for instance P_A elected randomly between feature vectors on the node considering experimental distribution from their τ_i weight, is allocated to this node. The P_A feature vector randomly selects another P_B feature vector According to W_A distribution probability. The total feature vectors $\langle P_A , P_B\rangle$ are considered as a successful pair. If two feature vectors belong to the same node then those two feature vectors would receive a reward, unless their pairs are unsuccessful in which case both are penalized [15].

Learning samples modeled by learner automaton applications haves been found in systems with insufficient knowledge about the environment and their startup. FSSA output functions and transfer do not change over time. The problem is based on the fact that a stable map of a subclass is obtained from learning automaton solutions and is used for solving object partitioning problems.

4.2 Variable Structure Stochastic Automata (VSSA)

VSSA are a replacement for FSSA and their transfer and output matrixes are changed in time [15,16,17]. VSSA are defined as a possible operation vector P(K) where P_i (K) is the possibility of ith operation in the set of A operation which is selected in K time from available |A| operations. As $\sum_i P_i(K)=1$ for all Ks, updating the law for possibility vectors would be continuous or discontinuous. The VSSA family has the quickest learning automaton convergence. Combining the VSSA family with automata for solving the specified problems will hopefully improve the speed of obtaining a solution [15,16,17]. The main characteristic of estimating algorithms is that they maintain estimations of possible rewards from each operation and use them in updating probability equations. In essence, an automaton selects an operation in the first step of the function cycle and produces the environment for answering the operation. Estimation of possible rewards for that operation is updated according to the estimating algorithm answer.

4.2.1 Discrete Generalized Pursuit Learning Automata (DGPA)

One of the problems of standard learning algorithms is their relatively slow convergence in selecting the optimum operation in static environments where several solutions have been introduced. These solutions are based on the disconnection of possibility space, in which the possibility of operation selection can pick only certain quantities in the range of [0,1] [15,16,17]. An existing problem in new models is premature convergence of learning algorithms with non-optimized operations, which is rooted in the limitation of possibility space. To improve learning algorithms convergence, Thathachar and Sastry introduced a new route in estimator algorithms. The most important specification of these algorithms is maintaining a continuous estimation based on the possibility of receiving the reward for any operation, and using it to update automata equations. In other words, in the first step of an operation cycle, automata would select an operation and then produces an environmental response for it. According to this response, the estimating algorithm would present reward possibility estimation for

that operation. One group of estimating algorithms is called pursuit algorithms.

Pursuit algorithms are divided in two classes of continuous and discrete [18]. The difference between these two classes lies in updating the law for operation possibilities. According to results in [18], partitioning on the basis of pursuit automata with variable structure would only work for small classes. Therefore, this paper have been used the fusion of parallel discrete pursuit learning automata, that is defined in section 4-2-2, to solve multi-constraint problems.

4.2.2 Fusion of DGPA

Learning rate and convergence speed equivalence are important paramters in learning automata. The parallel operations method is considered to increase convergence speed in environment.

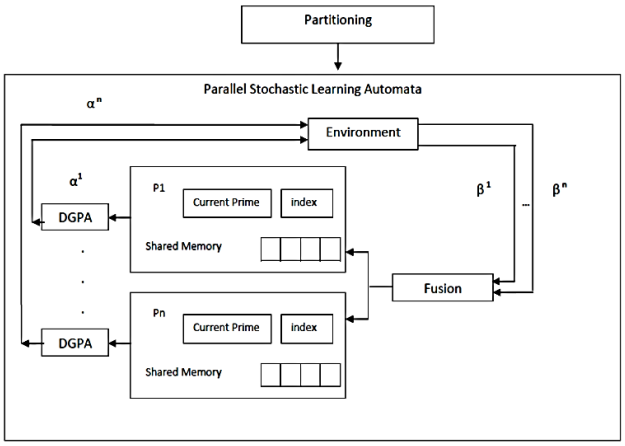


Fig. 2. Fusion of stochastic learning automata

Our proposed model is considered to be parallel instead of single learning automaton. Fig. 2 presents the allocation of |P| feature vector to |N| classes where next limitations are applied. At this stage, speech signals are divided into n parts and given to processors that the learning of discrete pursuit automata is applied to them in a parallel manner. In fig.2, n parameters are a total number of n={1,…,n} operations and P is the number of processors where each processor in this case is an automaton. "Indexes" in our model are stable and "Current Primes" in each moment are variable. Input vectors are divided by each processor to a certain number. Processors output is the input of DGPA.

In the fusion section, each set of operations is given by α, while the operation possibility vector P(K) is common to all n automata. Each instance of discrete pursuit learning is based on the common operation possibility vector, selected from ($\alpha^i(k)$). This vector obtains its own reinforcement signal (($\beta^i(k)$). It is supposed that $\beta^j(k) \epsilon [0,1]$ is obtained for all i and k and the fusion vector are computed by the equation (11) [18].

$$q_i(k) = \sum_{j=1}^{n} \beta^j(k) I\{\alpha^j(k) = \alpha_i\} \quad (11)$$

Equation (12) calculates the total response by simple summation:

$$q(k) = \sum_{j=1}^{n} \beta^j(k) = \sum_{j=1}^{n} q_i(k) \quad (12)$$

The output of fusion step is obtained by equation (13):

$$p_i(k+1) = p_i(k) + \tilde{\lambda}\left(q_i(k) - q(k)p_i(k)\right), i = 1, \ldots, r \quad (13)$$

$\lambda \epsilon (0,1)$ is learning parameter and $\tilde{\lambda} = \lambda/n$ is its normalized value. Considering computations in equation (13), P(K) has been updated only once. The updated value of p(k+1) is shared by all automata for selection of the next operation. This algorithm is suitable for n sizes and speeding up the rate of convergence.

5. Test Result and Conclusion

5.1 Dataset

The "Berlin Dataset of Emotional Speech" was used to train and test the algorithm in this paper [19,20]. The Berlin Dataset consists of 535 speech samples, consisting of German utterances related to emotions such as anger, disgust, fear, happiness, sadness, surprise, and neutrality, as performed by five male and five female voice actors. Each one of the ten professional actors expresses ten words and five sentences covering each of the emotional categories. The corpus was evaluated by 25 judges who classified each emotion with a score rate of 80%.

This Dataset was chosen for the following reasons: (i) the quality of its recording is very good and (ii) it is a public and popular Dataset of emotion recognition that is recommended in the literature [21]. This paper has used on the Berlin Dataset in order to achieve a higher and more accurate rate of recognition.

5.2 Lab Results

This paper introduces a new method of classification for speech emotion recognition. Signals are normalized and then fed to the MFCC and ACFC. After feature extraction by MFCC and ACFC, all features vectors are sent to the proposed classification model. Tables 1 to 7 indicate the experimental results on emotional speech signals.

Table 1. Speech emotion recognition using MFCC and single FSSA

FSSA	ANGER	JOY	NEUTRAL	DISGUST	FEAR	SADNESS
ANGER	**85.0**	10.0	10.00	15.0	10.0	0.0
JOY	0.0	**15.0**	0.20	0.0	0.0	0.0
NEUTRAL	15.0	65.0	**89.80**	30.0	55.5	40.0
DISGUST	0.0	0.0	0.0	**25.0**	0.5	10.0
FEAR	0.0	10.0	0.0	5.0	**20.0**	22.0
SADNESS	0.0	0.0	0.0	25.0	10.0	**28.0**

Table 1 shows that the highest rate of emotional learning is 89.80 percent by using MFCC and FSSA learning models for the neutral state and 85% for the state of anger.

Table 2. Speech emotion recognition using ACFC single FSSA

FSSA	ANGER	JOY	NEUTRAL	DISGUST	FEAR	SADNESS
ANGER	**83.25**	8.0	9.0	10.0	10.0	4.0
JOY	0.0	**10.0**	5.0	8.0	0.01	1.0
NEUTRAL	6.75	72.0	**86.0**	25.0	45.00	35.0
DISGUST	0.0	0.0	0.0	**20.0**	5.0	10.0
FEAR	0.0	10.0	0.0	9.0	**20.0**	21.0
SADNESS	0.0	0.0	0.0	28.0	10.0	**29.0**

Table 2 shows that the rate of emotional learning is 86.00% by using ACFC and FSSA learning models for the neutral state and 83.25% for the state of anger. By comparison of table 1 and table 2, the highest rate of recognition belongs to MFCC and FSSA.

To improve the rate of recognition, the fusion of MFCC and ACFC have been employed. Table 3 presents the rates of speech emotion recognition by using the fusion of MFCC, ACFC and FSSA classification.

Table 3. Speech emotion recognition using MFCC and ACFC with single FSSA

FSSA	ANGER	JOY	NEUTRAL	DISGUST	FEAR	SADNESS
ANGER	86.0	8.5	9.0	19.0	10.0	0.0
JOY	1.0	15.0	0.0	0.0	0.0	0.0
NEUTRAL	13.0	67.00	91.0	25.0	40.0	44.5
DISGUST	0.0	0.0	0.0	30.5	5.0	5.5
FEAR	0.0	9.5	0.0	0.5	30.0	10.0
SADNESS	0.0	0.0	0.0	25.0	15.0	40.0

According to table 3, the highest and lowest rate of recognition belong to neutral and joy, respectively. Table 4 indicates the rate of emotional speech recognition using the combination of MFCC, ACFC and fusion of FSSAs learning model.

Table 4. Speech emotion recognition using the combination of MFCC, ACFC and fusion of FSSAs

Fusion of FSSAs	ANGER	JOY	NEUTRAL	DISGUST	FEAR	SADNESS
ANGER	**88.50**	10.0	7.30	10.0	15.0	0.0
JOY	0.0	**17.0**	0.0	0.0	0.0	0.0
NEUTRAL	11.50	62.5	**92.70**	25.0	67.5	58.0
DISGUST	0.0	0.0	0.0	**30.0**	0.5	0.0
FEAR	0.0	10.5	0.0	1.0	**17.0**	22.0
SADNESS	0.0	0.0	0.0	25.0	0.0	**20.0**

According to table 4, the highest rate of recognition belongs to neutral state by 92.70% followed by anger with the value of 88.50%. Therefore, it can be claimed that FSSAs fusion algorithm gives better results than FSSA single model.

Table 5 shows the rate emotional speech recognition by using the combination of MFCC, ACFC and fusion of DGPAs classification method with two parallel processors.

Table 5. Speech emotion recognition using the combination of MFCC, ACFC and fusion of DGPAs with two parallel processors

Fusion of DGP as with two parallel processors	ANGER	JOY	NEUTRAL	DISGUST	FEAR	SADNESS
ANGER	**87.0**	28.0	8.0	5.0	20.0	0.01
JOY	3.0	**22.0**	1.0	0.0	6.0	5.0
NEUTRAL	10.0	50.0	**91.0**	55.0	52.0	25.0
DISGUST	0.0	0.0	0.0	**25.0**	1.0	10.0
FEAR	0.0	0.0	0.0	5.01	**21.0**	15.0
SADNESS	0.0	0.0	0.0	0.0	0.0	**35.0**

Table 5 indicates that he highest rate of learning occurs in the neutral speech state with the identification rate of 91%.

Table 6 shows the rate emotional speech recognition by using the combination of MFCC, ACFC and fusion of DGPAs classification method with two parallel processors.

Table 6. Speech emotion recognition using the combination of MFCC, ACFC and fusion of DGPAs with three parallel processors

Fusion of DGP as with three parallel processors	ANGER	JOY	NEUTRAL	DISGUST	FEAR	SADNESS
ANGER	**89.00**	15.0	7.10	5.0	0.0	10.0
JOY	0.0	**8.0**	0.0	5.0	0.0	15.0
NEUTRAL	11.00	62.0	**92.90**	50.0	60.0	24.3
DISGUST	0.0	0.0	0.0	**20.0**	9.0	10.0
FEAR	0.0	15.0	0.0	10.0	**20.50**	15.0
SADNESS	0.0	0.0	0.0	10.0	50.10	**25.7**

According to table 6 the highest rate of emotional speech recognition belong to the proposed model in neutral emotion by 92.90%. Table 7 indicates speech emotion recognition using the combination MFCC, ACFC to extract the features and three layers MLP methods to classify the feature's vector [14].

Table 7. Speech emotion recognition using the combination of MFCC, ACFC and three layer MLP

Fusion of DGP as with three parallel processors	ANGER	JOY	NEUTRAL	DISGUST	FEAR	SADNESS
ANGER	**85.0**	-	-	-	-	-
JOY	-	**30.0**	-	-	-	-
NEUTRAL	-	-	**87.50**	-	-	-
DISGUST	-	-	-	**25.0**	-	-
FEAR	-	-	-	-	**30.0**	-
SADNESS	-	-	-	-	-	**10.0**

According to table 7 the highest rate of emotional speech recognition by MLP classification belongs to neutral. But by comparing table 6 and table 7, the highest rate of speech emotion recognition belongs to the proposed model.

6. Conclusions

This paper introduces a new classification method based on multi- constraint partitioning by learning automata on emotional speech signals.

We have used six emotional states (anger, joy, neutral, disgust, fear and sadness), on the Berlin dataset, and the simulation environment has been MATLAB 2014. The proposed model consists of two main parts: feature extraction and classification. In feature extraction part, the proposed model used the combination of MFCC and ACFC models. In the part of classification multi-constraint partitioning, different type of learning automata are used including variable structure, fixed structure learning automata and DGPA.

According to experimental results, each model of classification has some disadvantages such as poor learning speech and low performance. Therefore we introduced a fusion model of learning automata with different number of parallel processors. Experimental result shows that the combination of MFCC, ACFC and fusion of DGPAs with three parallel processors has a higher performance on emotional speech signals than other methods.

Also, by comparison between tables 1 to 6 we have found that the highest rate of speech emotion recognition belong to neutral emotion. Although the proposed model have been compared by MLP model.

References

[1] R´azuri, J.G., D. Sundgren, R. Rahmani, A. Larsson, A.M. Cardenas, and I. Bonet. "Speech emotion recognition in emotional feedback for human - robot interaction ". International Journal of Advanced Research in Artificial Intelligence, vol. 4, pp. 20-27, 2015.

[2] Ayadi, M., S. Kamel, and F. Karray. "Survey on speech emotion recognition: features, classification schemes and databases". Pattern Recognition, vol. 443, pp. 572- 587, 2011.

[3] Cowie, R., S. Tspatsoulis, S. Kollias, and J. Taylor. "Emotion recognition in human-computer interaction". IEEE signal Processing, vol. 18, pp. 32-80, 2001.

[4] Seehapoch, T. and S. Wongathanavasu. "Speech emotion recognition using support vector machines". 5th International Conference on Knowledge and Smart Technology (KST), 2013, pp. 621 - 625.

[5] Ververidis, D. and C. Kotropoulos. "Emotional speech recognition: resources, features and methods". Elsevier Speech communication, vol. 489, pp. 1162- 1181, 2006.

[6] Hozjan, V. and Z. Kacic. "Context-independent multilingual emotion recognition from speech signal". Int. J. Speech Technol, vol. 6, pp. 311-320, 2003.

[7] cahn, J. "The generation of affect in synthesized speech". Journal of the American Voice I/O Society, pp. 1-19, 1990.

[8] Zhang, Q., N. An, K. Wang, F. Ren, and L. Li. "Speech Emotion Recognition using Combination of Features". Forth International Conference on Intelligent control and Information Processing (ICICIP), 2013.

[9] Bojanic, M., V. Crnojevic, and V. Deliv. "Application of neural network in emotional speech recognition". IEEE, pp. 20-22, 2012.

[10] Price, J. and A. Eydgahi. "Design an automatic speech recognition system using Matlab". 9th International Conference on Engineering Education, 2006, pp. 100-106.

[11] Adell, J., A. Bonafonte, and D. Escudero. "Analysis of prosodic features: towards modelling of emotional and pragmatic attributes of speech". Sociedad Española para el Procesamiento del Lenguaje Natural, vol. 35, pp. 277- 284, 2005.

[12] Wu, D., T.D. Parsons, and S.S. Narayanan. "Acoustic feature analysis in speech emotion primitives estimation". Interspeech, pp. 785-788, 2010.

[13] Lotfi, E. "Mathematical modeling of emotional brain for classification problems". Proceedings of IAM, vol. 21, pp. 60- 71, 2013.

[14] Muda, L., M. Begam, and Elamvazuthi. "Voice recognition algorithms using mel frequency cepstral coefficient (MFCC) and dynamic time warping (DTW techniques)". Journal of Computing, vol. 23, pp. 138-143, 2010.

[15] Thathachar, M.A.L. and P.S. Sastry. "Varieties of learning automata". An Overview in IEEE Transaction on System, vol. 326, pp. 711-722, 2002.

[16] Narendra, K.S. and M.A.L. Thathachar. "Learning automata". An Introduction in Prentice Hall, 1974.

[17] Narendra, K.S. and M.A.L. Thathachar. "Learning automata - A survey". IEEE Transactions on Systems, Man and Cybernetics, vol. 44, pp. 323-334, 1974.

[18] Horn, G. and B.J. Oommen. "Solving Multiconstraint Assignment Problems Using Learning Automata". IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), vol. 401, pp. 6 - 18, 2009.

[19] Eyben, F., M. Wöllmer, and B. Schuller. "Opensmile: the munich versatile and fast open source audio feature extractor". 10 Proceedings of the International Conference on Multimedia, 2010, pp. 1459-1462.

[20] Harimi, A., A. Shahzadi, A.R. A. R. Ahmadyfard, and K. Yaghmaie. "Classification of emotional speech spectral pattern features". Journal of AI and Data Mining, vol. 21, pp. 53-61, 2014.

[21] Burkhardt, F., A. Paeschke, M. Rolfes, W.F. Sendlmeier, and B. Weiss. "A database of German emotional speech". Interspeech, vol. 5, pp. 1517-1520, 2005.

Sara Motamed She is a Ph.D candidate of artificial intelligence engineering in Science and Research University of Tehran, Iran. She is a lecturer at Islamic Azad University of Fouman, Iran. Her interests are cognitive science image processing, machine vision, signal processing, and etc. She is reviewer of Journal of Information Systems and Telecommunication (JIST).

Saeed Setayeshi Associate Professor at Amirkabir University of technology, Tehran, Iran. His research interests include medical imaging systems, neural networks and fuzzy control, Medical radiation instrumentation, etc. He has presented and published many articles in scientific journals and conferences.

Azam Rabiee Assistant Professor from 2012, and lecturer from 2004 at Islamic Azad University, Dolatabad Branch, Isfahan, Iran.

Formerly, she was with Computational NeuroSystems Lab., Brain Science Research Center, KAIST in Daejeon, Korea from 2010 to 2011, as a visiting researcher. Her research interest includes speech processing, machine learning and biologically-inspired artificial intelligence approaches.

Arash Sharifi Received the B.Sc degree in computer hardware engineering from Islamic Azad university South Tehran branch, the M.Sc and Ph.D degree in computer artificial intelligence engineering from Islamic Azad University, Science and Research branch (SRBIAU), Tehran, in 2004, 2006, and 2010, respectively. He is currently assistant professor and the head of Department of Computer artificial intelligence engineering in SRBIAU University. His current research interests include machine learning, neural networks, deep learning and evolutionary algorithms.

Coverage Improving with Energy Efficient in Wireless Sensor Networks

Amir Pakmehr
Department of computer engineering, Tabriz branch, Islamic azad university, Tabriz, Iran
Amir.Pakmehr@gmail.com
Ali Ghaffari*
Department of computer engineering, Tabriz branch, Islamic azad university, Tabriz, Iran
A.Ghaffari@iaut.ac.ir

Received: 17/Aug/2016 Revised: 07/Feb/2017 Accepted: 07/Mar/2017

Abstract

Wireless sensor networks (WSNs) are formed by numerous sensors nodes that are able to sense different environmental phenomena and to transfer the collected data to the sink. The coverage of a network is one of the main discussion and one of the parameters of service quality in WSNs. In most of the applications, the sensor nodes are scattered in the environment randomly that causes the density of the nodes to be high in some regions and low in some other regions. In this case, some regions are not covered with any nodes of the network that are called covering holes. Moreover, creating some regions with high density causes extra overlapping and consequently the consumption of energy increases in the network and life of the network decreases. The proposed approach causes an increase in life of the network and an increase in it through careful selection of the most appropriate approach as cluster head node and form clusters with a maximum length of two steps and selecting some nodes as redundancy nodes in order to cover the created holes in the network. The proposed scheme is simulated using MATLAB software. The function of the suggested approach will be compared with Learning Automata based Energy Efficient Coverage protocol (LAEEC) approach either. Simulation results shows that the function of the suggested approach is better than LAEEC considering the parameters such as average of the active nodes, average remaining energy in nodes, percent of network coverage and number of control packets.

Keywords: Wireless Sensor Networks; Clustering; Network Coverage; Covering Holes; Energy Efficient.

1. Introduction

Recent technological advancements in the realm of micro-electro-mechanical systems (MEMS), wireless telecommunications and digital electronics led to the production of sensors which are cheap, low-power and compact. These sensors can fulfill multi-functions such as monitoring and surveilling environments in unprotected areas and carry out telecommunication operations. Indeed, these sensors can accomplish different functions such as sensing, data processing and transmitting information which establish the rationale behind wireless sensor networks (WSNs) [1, 5, 7, 8, 24, 28]. WSNs consist of a high number of sensor nodes which are closely distributed with high density within a phenomenon. In general, there is no need for continuous monitoring of the location and position of sensor nodes; neither is it required to predetermine their locations [4, 6, 9-11]. Hence, these nodes can be randomly arranged and set in the network environment. This feature expands the application and use of WSNs to remote or dangerous areas where they can be used with minimum surveillance and monitoring. For achieving this purpose, protocols of WSNs should have the capability of self-organization. That is, the protocols and algorithms operated on WSNs are managed by the sensors themselves. Instead of transmitting all the raw data to the sink node, each sensor node has a processor which can process the data in a limited way; then, it can transmit the semi-processed data to the sink node. Since lots of sensor nodes with high-density are used in an environment, they might be very close to one another. Thus, it can be assumed that multi-hop communications can transmit data to the sink by using less power and energy than the single-hop communications. In contrast with the traditional wired networks, on the one hand, configuration and makeup costs are reduced in WSNS; on the other hand, instead of installing thousands of meters of wire, only tiny sensors the size of about a few cubic centimeters have to be distributed in the environment. By merely adding some nodes, these networks can be extended and there is no need for complex reconfiguration. One of the most important challenges in WSNs is phenomenon and area coverage [12-20]. Indeed, the method proposed in this paper is a cluster-based method for network. In this method, the maximum length of clusters was considered to be two hops which led to the reduction of hop length. Hence, the algorithm is executed within the cluster rather than being executed by all the hops. Consequently, the network can reach a steady and stable state earlier and its coverage begins earlier.

The rest of the paper is organized as follows. Section 2 briefly reviews related work on existing methods about Coverage and Energy issues. In Section 3, the proposed area

* Corresponding Author

coverage algorithm is presented. Section 4 shows the performance of the proposed algorithm through simulation experiments and comparison with the existing methods. Section 5 concludes the paper.

2. Related Work

Recently, many important related works in area coverage have proposed for WSNs [21-25]. The connected dominating set (CDS) [11] concept has recently emerged as a promising approach to the area coverage in WSN. However, the major problem affecting the performance of the existing CDS-based coverage protocols is that they aim at maximizing the number of sleep nodes to save more energy. This places a heavy load on the active sensors (dominators) for handling a large number of neighbors. The rapid exhaustion of the active sensors may disconnect the network topology and leave the area uncovered. Therefore, to make a good trade-off between the network connectivity, coverage, and lifetime, a proper number of sensors must be activated. Paper [11] presents a degree-constrained minimum-weight extension of the CDS problem called Degree-constrained CDS (DCDS) to model the area coverage in WSNs. The proper choice of the degree-constraint of DCDS balances the network load on the active sensors and significantly improves the network coverage and lifetime. A learning automata-based heuristic named as LAEEC [11] is proposed for finding a near optimal solution to the proxy equivalent DCDS problem in WSN. The computational complexity of the proposed algorithm to find a $\frac{1}{1-\epsilon}$ optimal solution of the area coverage problem is approximated.

In energy-limited wireless sensor networks [3], network clustering and sensor scheduling are two efficient techniques for minimizing node energy consumption and maximizing network coverage lifetime. When integrating the two techniques, the challenges include how to decide the most energy-efficient cluster size and how to select cluster heads and active nodes. [3] paper provide a computation method for the optimal cluster size to minimize the average energy consumption rate per unit area. In the proposed coverage-aware clustering protocol, define a cost metric that favors those nodes being more energy-redundantly covered as better candidates for cluster heads and select active nodes in a way that tries to emulate the most efficient tessellation for area coverage.

An energy-aware distributed unequal clustering protocol in multi-hop heterogeneous wireless sensor networks [23] is proposed. It elects cluster heads based on the ratio between the average residual energy of neighbor nodes and the residual energy of the node itself, and uses uneven competition ranges to construct clusters of uneven sizes. The cluster heads closer to the BS have smaller cluster sizes to preserve some energy for the inter cluster data forwarding, which can balance the energy consumption among cluster heads and prolong the network lifetime.

In [2], the authors provide a novel algorithm using trees and graph theory to detect and describe the existing holes in the region of interest. Simulation results show that the tree-based method can indicate the location, size, and shape of coverage holes accurately. Based on the results for hole detection, a tree-based healing method is also proposed. The method is divided into two phases, namely, hole dissection and optimal patch position determination. On [2] Results obtained from the experimental evaluation reveal that the proposed healing method can increase the coverage rate with only a few additional sensors compared to other related methods.

On connected target *k*-Coverage in heterogeneous wireless sensor networks in [26], focus on the connected target *k*-coverage (CTCk) problem in heterogeneous wireless sensor networks (HWSNs). A centralized connected target *k*-coverage algorithm (CCTCk) and a distributed connected target *k*-coverage algorithm (DCTCk) are proposed so as to generate connected cover sets for energy-efficient connectivity and coverage maintenance. To be specific, the proposed scheme in [26] aims at achieving minimum connected target *k*-coverage, where each target in the monitored region is covered by at least *k* active sensor nodes.

In [27], the authors proposed an integrated and energy-efficient protocol for Coverage, Connectivity and Communication (C3). C3 protocol runs in six steps. 1. *Formation of rings:* The C3 protocol divides the network into virtual concentric rings, based on the communication range (*Rc*), using received signal strength indicator (RSSI) distance estimator. 2. *Formation of clusters*: A cluster head is selected alternately from even or odd numbered rings, in a round. 3. *Formation of dings*: A ding is a subsection of ring with a cluster head. The cluster head identifies the nodes which are at the distance of $\sqrt{3}R_s$ to form the ding. Therefore, there might be multiple dings inside a ring. 4. I*dentification of redundant nodes*: C3 protocol uses triangular tessellation based on *Rs* inside the dings to identify redundant nodes. The redundant nodes can enter into sleep state for a time duration of T. 5. *Establish connectivity:* C3 protocol establishes connectivity between neighboring nodes and cluster head. 6. *Communication: Finally*, in C3 protocol data are transmitted to the sink node with the help of cluster heads.

3. Proposed Work

Before introducing the novel proposed method, the drawbacks of LAEEC [11] method are first outlined and some solutions for sorting out those shortcomings by means of the proposed method are mentioned. In the LAEEC [11] protocol, the whole network nodes are fixed and are randomly distributed in the network. Also, the network is not clustered in this protocol. The way of executing the LAEEC [11] algorithm in the network is as follows. At first, one node is randomly selected by the

sink node; then, the algorithm is executed by the selected node. Then, the node selected by the sink randomly chooses another node for executing the algorithm. This procedure is repeated until all the respective nodes of the network execute the algorithm. This operation will cause delays in algorithm execution in the network. In LAEEC algorithm, the failure or death of a node has the role of cut-vertex which leads to the division of the network into two distinctive parts. Consequently, firstly, the algorithm is executed in an infinite loop by the nodes of the first area of the network and it is never ended, unless a specific time is considered for the execution of the algorithm. Secondly, due to the lack of communication between the first and second areas, the nodes of the second area are never selected by the nodes of the first zone for executing the algorithm. as a result, the algorithm is executed incompletely. Cut-vertex is a vertex of the graph which will lead to an increase in the number of the connective variables of graph if it is removed. In case the graph is connective before cut-vertex is removed, then, it will be non-connective after it is removed. Cut-vertex is of high significance in computer networks.

In the LAEEC [11] algorithm, a node might not be located within the coverage range of the last node which executes the algorithm; hence, it is not selected as the node for executing the algorithm. the unselected node will result in the placement of the algorithm in an infinite loop. It should be noted that any particular predictions were not considered for such unselected nodes. However, in the proposed method, since the network is clustered, there is no need for gauging and scaling all the nodes for algorithm execution. In case a node does not receive any messages from its neighbors within a specific time, it will announce itself as the cluster head.

One more shortcoming of the LAEEC [11] approach is that environment impact was used for rewarding actions but the selection of action was done randomly. Nevertheless, in the proposed method, when a node is gone due to failure or energy exhaustion, the cluster head and redundant node make choices based on the parameters of the node degree and the remaining energy of the node. The respective parameters and assumptions of the proposed method are discussed below.

3.1 Parameters and Assumptions used in the Proposed Method

Table 1 shows the parameters and their descriptions.

Table 1. Parameters and their description

Parameter	description
S	The set of homogenous nodes which are distributed in the network
S_u, S_v	Two different nodes which are the subset of S
S_{CH}	The set of cluster heads
CH_k	Cluster head for the cluster number k
S_{H1}	The set of the single-hop members of cluster
S_{H2}	The set of the two-hop members of cluster
$S(ni)$	The set of nodes which overlap with the n_i node
K	The number of overlapping neighbors, in case a node has K overlapping neighbors, it can be regarded as a redundant node.
$C(ni)$	The set of nodes which have the minimum required coverage in comparison with n_i. The minimum required coverage refers to the ratio of the number of the positions of the neighboring nodes to the total positions of the respective node which is an optional number. In the proposed method, the minimum required coverage was regarded 60%.
Frame-number	The maximum number of the transmitted data frames throughout a cycle which was 100 frames in the present study.
Node-degree (S_v)	the number of neighbors of the S_v node so that: NodeDegree(S_v) = count({Su | dist(Sv, S_u)<Rc , Su ∈S, $S_u \neq S_v$})

The node S_u will be regarded as the neighbor of S_v node if node S_u is located within the radio range of S_v. In the following, the stages of the proposed method are described. It should be pointed out that the proposed method has the four stages of set-up, cluster head selection and cluster formation, identification of redundant nodes, gap detection and the movement of redundant nodes for covering the available gaps. The detailed description of each stage is given below.

3.2 Set-up

In this stage, a number of sensor nodes are randomly distributed in a two-dimensional space. These sensors have the capability of dynamicity. In the proposed method, it was assumed that each sensor node is aware of the geographical position by means of a Global Positioning System (GPS) machine or via other network positioning methods. It was assumed that all the sensor nodes are active in the network set-up stage. After sensors are distributed in the network, these sensors broadcast a hello message in the respective environment to identify all of their neighbors and transmit some data to them regarding their won state.

3.3 Selecting Cluster Head and Cluster Formation

In this stage, three messages are used which include the followings:

A. *State message:* using this message, each sensor transmits some information and data to their neighbors about their own state including node degree, the remaining energy, etc. for being selected as the cluster head.
B. *Cluster head (CH) message:* a node which considers itself as a cluster head candidate uses this message to announce its candidacy and some information about itself to neighbors.
C. *Join message:* each sensor node uses this message to introduce itself as the single-hop or two-hop member of a particular cluster head.

In this phase, an attempt is made to select cluster heads from the congested areas of the network. For achieving this objective, a variable called node degree was defined. Indeed, node degree refers to the number of single-hop neighbors of a node in the network. The way of measuring degree for each node is as follows. After network set-up stage and the deployment of the sensors in the intended environment, a set-up message is broadcast by the sink to all the network nodes. Upon receiving this message, each sensor node transmits its data to all of its neighbors within a random time interval from zero to T_{max}. By receiving this message, each of the neighbors can

determine their own degree by investigating the contents of the message. The reason for using a random time interval between zero and T_{max} is to reduce collision while transmitting hello messages. In the next step, each sensor node waits for a specific time (*Tni*) so that it can receive the message of cluster head candidacy from the closest cluster head candidate. In case it does not receive a message within a specific time interval, it will announce itself as the cluster head and will broadcast its cluster head message along with some other data about its state in the network. For selecting a node with the maximum degree as the cluster head, the T_{ni} time of this message in proportion to its degree should be measured based on Equation (1) [25].

$$T_{n_i} = \alpha . e^{\frac{1}{degree_{(ni)}}} \quad (1)$$

On equation (1) where α is a constant coefficient and factor which should be arranged and adjusted in a way that $0 < T_{n_i} \leq T_{max}$.

it might happen that two nodes within the same area not only have higher degrees than their neighbors, but also they have the identical degrees. Hence, in that area, two nodes which are close to one another will simultaneously announce cluster head candidacy. Thus, instead of becoming members of one cluster head, some nodes become members of one cluster head and some other nodes become members of the other cluster head. Consequently, the number of clusters will increase in the area which contradict the objective of minimizing the number of clusters in the network. For sorting out this problem and selecting the most efficient node as the cluster head in the area, the remaining energy of the candidates are taken into consideration. As a result, equation (1) changes into equation (2) and the node with the highest remaining energy is selected as the cluster head. Hence, there will be only one cluster rather than two clusters in one area.

$$T_{n_i} = \alpha . e^{k} \quad (2)$$

And k defined as Equation (3)

$$k = \frac{1}{\left(degree_{(ni)} + \left(\frac{E_{res(ni)}}{E_{ini_{(ni)}}} \right) \right)} \quad (3)$$

Where in Equation (3), $E_{res(ni)}$ is residual energy of node and $E_{ini(ni)}$ is the initial energy of node.

According to Equation (3), each sensor node should measure the valid threshold energy for cluster head before it announces itself as a candidate for cluster head. Hence, in case its remaining energy is less than the threshold value, it will consider T_{ni} as the maximum value so that it will not introduce itself as the cluster head as far as possible.

According to Equation (4), *L* stands for the length of data and *dist(BS,Sv)* denotes the distance between S_y and the sink [25].

$$E_{th}(S_v) = L \times ((E_{elec} + E_{DA}) \times (S_v + 1) + \in_{mp} \times d\,(BS, S_v)^4) \times F_n \quad (4)$$

In the next stage, upon receiving the message about cluster head candidacy from the closest candidate node, it transmits a message and introduces itself as the member of that cluster head. Also, as the cluster head receives this message, it stores the data of the member node in its database and, from that moment, it regards it as a single-hop member. Furthermore, as the neighboring nodes of that single-hop member receive the cluster head candidacy message, they introduce themselves as the two-hop members of that cluster head. In general, single-hop and two hop neighbors for one sensor are defined in the following way: if the distance between A and B sensors is less than *Rs* and *d(A, B)= Rs* is true, these two neighbors are single-hop neighbors of one another. In other words, these two nodes can directly communicate with each without intermediaries. If the distance between A and B nodes is in the way *Rs<dAB,=2Rs,* A and B nodes are regarded as the two-hop neighbors and they can communicate with one another via the help of one single-hop neighbor.

After cluster heads are selected and network clusters are formed, cluster heads specify a certain time for each of their members for transmitting data; hence, each member should transmit its data to the cluster head only within the specified time interval. The time duration for data transmission by each member is announced by means of message to each of the other members. Thus, scheduling and time planning among the members which is managed by the cluster heads prevents data collision while transmitting data to the cluster head. For maintaining a balance for network load, re-clustering is carried out at the beginning of each cycle.

3.4 Identifying Redundant Nodes

In this stage, the cluster head identifies redundant and unnecessary nodes with regard to covering an area and, if possible, inactivates them. The merit of this action is that in case some nodes are destroyed in an area and the respective area covered by them is destroyed, the redundant nodes can be transmitted to such areas so as to cover them. Different methods have been proposed for determining redundant nodes. As shown in Figure 1, in the proposed method, 3 n-polygons surrounded within a circle with r radius were used for detecting redundant nodes.

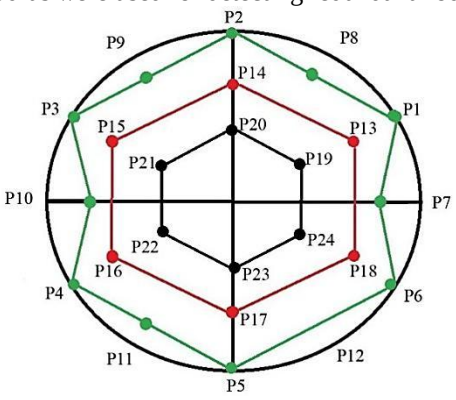


Fig. 1. 3 n-polygons surrounded within a circle with r radius were used for detecting redundant nodes.

Redundant nodes arte detected and identified throughout two stages in the following way:

A. Identifying single-hop redundant nodes by the cluster head:

1. In this step, the cluster head specifies certain nodes from among single-hop members which have *c(ni)* equal to or larger than k and stores their numbers in a temporary set $S(n_i)$. Then, their redundancies are determined through stages 2 and 3.
2. In this step, the cluster head selects a node like A from the $S(n_i)$ set with the lowest remaining energy as the redundant node.
3. After the selection of node as the redundant node, each active member of the *S(ni)* set with active members more than or equal to k will be also considered as a redundant node. this step will be repeated until no active node which is a member of *S(ni)* has the required conditions for being redundant. In case a redundant node is selected as the communication bridge between the cluster head and a node which is a member of a two-hop cluster, that node will be activated.

B. Identifying two-hop redundant nodes by the single-hop member nodes of the cluster

The steps for identifying two-hop redundant nodes are as follows:

1. In this step, each single-hop cluster member node specifies the nodes from among single-hop members with *C(ni)* sets which are greater or equal to k and stores them in the *S(ni)* temporary set. Then, it determines their redundancy through steps 2 and 3.
2. In this step, at first, single-hop member of the cluster head selects a node, namely A form the *S(ni)* set which has the lowest remaining energy as the redundant node.
3. Then, after the selection of *A* as the redundant node, if each active member of *S(ni)* has active *C(ni)* members which are greater than or equal to k, that node will also be selected as the redundant node. The third step is repeated until no active *S(ni)* member has the required conditions for being redundant.

After the cluster is produced, the cluster head uses the information obtained from the single-hop and two-hop members of the cluster to identify the available gaps in the cluster and covers them by using redundant nodes. In the following section, the way of identifying gaps and covering them are discussed.

3.5 Identifying Gaps and the Movement of the Redundant Nodes for Covering Available Gaps

In this stage, as shown in Figure 2, the cluster head surrounds a hexagon within the circle of its radio range and measures the coordinates of the specified points on the hexagon. Then, it investigates whether the obtained points are covered by the single-hop members or not? In case these points are not covered by the single-hop members, they will be considered as gaps. If *p* variable is assumed to denote the number of inactive single-hop members of the cluster head which have been selected as redundant nodes, then, the cluster head can cover $\frac{P}{2}$ of the gaps by means of the redundant inactive nodes. For instance, for covering two gaps, at least four redundant nodes should be used.

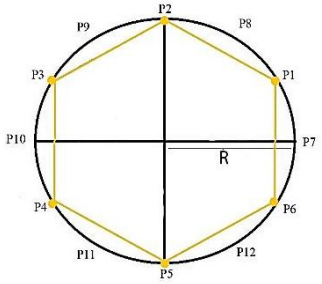


Fig. 2. the cluster head surrounds a hexagon within the circle of its radio range

3.6 Energy consumption model

Energy Consumption of the sensing device should be minimized and sensor nodes should be energy efficient since their limited energy resource determines their lifetime. The Energy consumption model should be measured based on equations (5), (6), (7)

$$E_{Tx} = k[E_{elec} + \varepsilon_{amp} d^2] \quad (5)$$

$$E_{Rx} = kE_{elec} \quad (6)$$

$$E_{Tot} = E_{Tx} + E_{Rx} = k\,[2E_{elec} + \varepsilon_{amp} d^2] \quad (7)$$

In above equations, *K* is packet size (bit), *d* is the distance between the sender and the receiver and E_{elec} is constant value.

4. Performance Evaluation

At first, a WSN in an environment the size of 100m×100m square meter including 50 to 250 sensor nodes which were randomly and steadily distributed with steady distribution was simulated by the software. Table 2 shown the parameters that used in the simulation.

Table 2. Parameters used in the simulation

Parameter	Value
Number of nodes	50-250
Reception range radius	10 m
Transmission range radius	20 m
Network area	100m×100m
The number of algorithm execution time	10
Sink location	50×50 m^2
	75×75 m^2
	25×25 m^2
α	0.18
K	2
E_{elec}	*50nj/bit*
E_{DA}	*5nj/bit/ message*
The size of data packets	4000 b
The size of control packets	200 b
ε_{fs}	100 Pj/bit/m^2
E_{elec}	0.0013 Pj/bit/m^4
initial energy	2 j

The efficiency of the proposed method was compared with that of the LAEEC [11] algorithm with respect to the following parameters: average number of active nodes,

average remaining energy of the sensors, the percentage of network coverage and the average number of control packets. The comparison of the evaluation parameters for the two algorithms are discussed below.

4.1 Comparing the Two Algorithms with Regard to the Average Number of Active Nodes

Regarding the average number of active nodes, the efficiency of the proposed method was compared with that of the LAEEC [11] algorithm under three different scenarios.

- **First scenario: sink in the [50 50] coordinate**

Figure 3 illustrates the average number of active nodes in relation to the number of network nodes within the range from 50 to 250 nodes. The proposed algorithm was executed within 1500 seconds. As shown in Figure 3, when the total number of network nodes was 50, the number of active nodes in the proposed method and the LAEEC protocol were about 38 and 43, respectively. In this way, when the total number of network nodes was 200, the number of active nodes in the proposed method and the LAEEC protocol were 113 and 142, respectively. As the average number of network nodes, active nodes of the network increases too which is illustrated in Figure 3. It should be noted that the degree of increase in the proposed method was less than that in the LAEEC protocol.

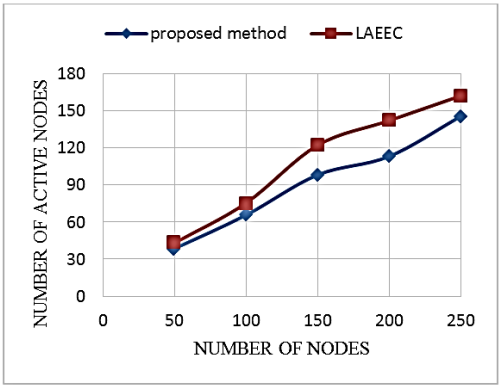


Fig. 3. The number of active nodes vs. the total number of network nodes.

- **Second scenario: sink in the [75 75] coordinate**

Figure 4 illustrates the average number of active nodes in relation to the number of network nodes within the range from 50 to 250 nodes. The proposed algorithm was executed within 1500 seconds. As shown in figure 4, when the total number of network nodes was 50, the number of active nodes in the proposed method and the LAEEC [11] protocol were about 37 and 44, respectively. In this way, when the total number of network nodes was 200, the number of active nodes in the proposed method and the LAEEC [11] protocol were 137 and 148, respectively. As the average number of network nodes, active nodes of the network increases too which is illustrated in Figure 4. It should be noted that the degree of increase in the proposed method was less than that in the LAEEC [11] protocol.

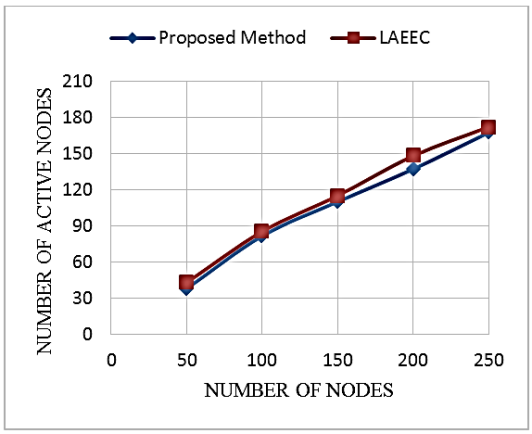


Fig. 4. The number of active nodes vs. the total number of network nodes.

4.2 Comparison of the Average Remaining Energy in Nodes

Regarding average remaining energy in the nodes, the efficiency of the proposed method was compared with the LAEEC method under three different scenarios.

- **First scenario: sink in the [50 50] coordinate**

Figure 5 shows the remaining energy in the nodes in relation to the number of network nodes within the interval from 50 to 250 nodes. After the execution of the algorithm for 1500 seconds, when the total number of network nodes is 50, the average remaining energy of the total nodes of the network in the proposed and LAEEC [11] methods were about 0.8 and 0.6 joules, respectively. In a similar vein, when the total number of network nodes was 200, the average remaining energy for the network nodes in the proposed and LAEEC [11] methods were about 1.3 and 1.2 joules, respectively. As shown in Figure 5, as the number of networks nodes increases, the average remaining energy of the network nodes in the proposed method was greater than that of the LAEEC [11]protocol. Consequently, it can be argued that network life time in the proposed method was greater than the network life time in the LAEEC [11] protocol.

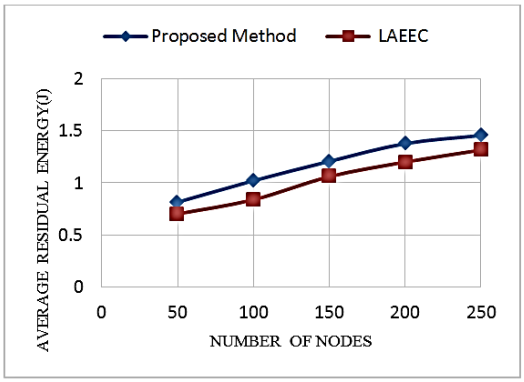


Fig. 5. The average residual energy of the active nodes as a function of the number of nodes.

- **Second scenario: sink in the [75 75] coordinate**

Figure 6 shows the remaining energy in the nodes in relation to the number of network nodes within the interval from 50 to 250 nodes. After the execution of the algorithm for 1500 seconds, when the total number of network nodes is 50, the average remaining energy of the total nodes of the network in the proposed and LAEEC [11] methods were about 0.6 and 0.8 joules, respectively. In a similar vein, when the total number of network nodes was 200, the average remaining energy for the network nodes in the proposed and LAEEC [11] methods were about 1.2 and 1.1 joules, respectively. As shown in Figure 6, as the number of networks nodes increases, the average remaining energy of the network nodes in the proposed method was greater than that of the LAEEC [11] protocol. Consequently, it can be argued that network life time in the proposed method was greater than the network life time in the LAEEC [11] protocol.

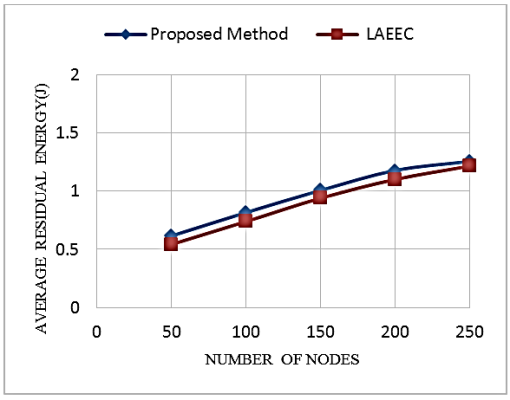


Fig. 6. The average residual energy of the active nodes as a function of the number of nodes.

4.3 Comparison of the Percentage of the Network Coverage

In this Section, the efficiency of the proposed method was compared with that of the LAEEC protocol in terms of network coverage under three different scenarios.

- **First scenario: sink in the [50 50] coordinate**

Figure 7 illustrates network coverage with respect to the number of network nodes within the range from 50 to250 nodes. Algorithm execution was completed after 1500 seconds. When the total number of network nodes is 50, 80% of the network was covered in the proposed but 71% of the network was covered in the LAEEC [11] protocol. Similarly, when the total number of network nodes was 200, 97% of the network in the proposed method and 93% of the network in the LAEEC [11] protocol were covered. In general, as the number of nodes increases, the percentage of network coverage increases too. Nevertheless, it should be highlighted that the percentage of network coverage in the proposed method was significantly greater than that of the LAEEC [11] method.

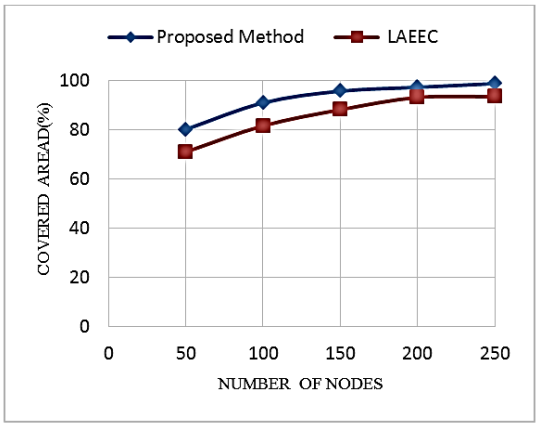


Fig. 7. The percentage of the covered area vs. the network size.

- **Second scenario: sink in the [75 75] coordinate**

The results of this scenario we saw no any difference with previous scenario on area coverage percent.

4.4 Comparison of the Average of Control Message Number

Figure 8 illustrates network average of control message number within the range from 50 to250 nodes. Algorithm execution was completed after 1500 seconds. When the total number of network nodes is 50, average of control message number in the proposed method and the LAEEC protocol were about 70 and 110, respectively. In this way, when the total number of network nodes was 200, average of control message number in the proposed method and the LAEEC protocol were 208 and 397, respectively. As the average number of network nodes, number of control message of the network increases too which is illustrated in Figure 8. It should be noted that the degree of increase in the proposed method was less than that in the LAEEC protocol.

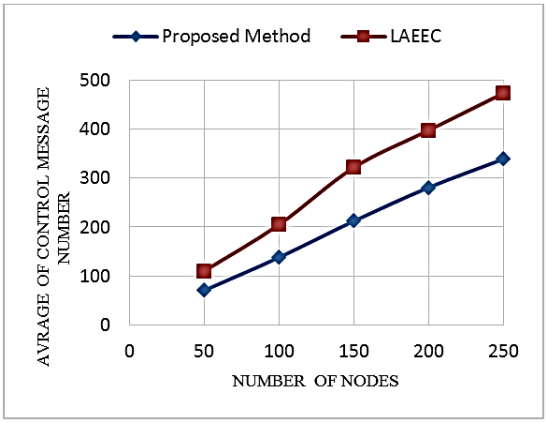


Fig. 8. The comparison of the average of control message number vs. the network size

- **Second scenario: sink in the [75 75] coordinate**

The results of this scenario we saw no any difference with previous scenario on average of control message number.

5. Conclusion and Future Work

The chief objective of the present study was to enhance network coverage by optimizing and improving clustering and reducing energy consumption of the nodes in WSNs. The protocols which were reviewed earlier in the paper were all aimed at improving significant parameters such as coverage and network life time. However, although many studies have been conducted in this domain, there are still many open problems and gaps. Hence, the present study was conducted to address these issues and improve the parameters. In this study, sensors with adaptable and dynamic reception range were used and an attempt was made to optimize clustering. Indeed, redundant nodes were selected to cover the probable gaps of the network, enhance network life time and better cover the network.

In this paper, the proposed method was compared with the LAEEC protocol. The results of the simulation indicated that the proposed method was better than the LAEEC protocol with respect to the following evaluative parameters: average number of active nodes, average remaining energy of the nodes, percentage of network coverage and the average number of control packets. In future, we will focus on providing new scheme based on soft computing. Also some new parameters must be considered to further improvement of this algorithm and extending the network lifetime.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," Computer networks, vol. 38, no. 4, pp. 393-422, 2002.

[2] W. Li and Y. Wu, "Tree-based coverage hole detection and healing method in wireless sensor networks," Computer Networks, vol. 103, pp. 33-43, 2016.

[3] B. Wang, H. B. Lim, and D. Ma, "A coverage-aware clustering protocol for wireless sensor networks," Computer Networks, vol. 56, no. 5, pp. 1599-1611, 2012.

[4] G. Wang, G. Cao, P. Berman, and T. F. La Porta, "Bidding protocols for deploying mobile sensors," IEEE Transactions on Mobile Computing, vol. 6, no. 5, pp. 563-576, 2007.

[5] B. Wang, H. B. Lim, and D. Ma, "A survey of movement strategies for improving network coverage in wireless sensor networks," Computer Communications, vol. 32, no. 13, pp. 1427-1436, 2009.

[6] A. Ghaffari, "Congestion control mechanisms in wireless sensor networks: A survey," Journal of network and computer applications, vol. 52, pp. 101-115, 2015.

[7] A. Ghaffari, "An energy efficient routing protocol for wireless sensor networks using A-star algorithm," Journal of applied research and technology, vol. 12, no. 4, pp. 815-822, 2014.

[8] A. Ghaffari and S. Nobahary, "FDMG: Fault detection method by using genetic algorithm in clustered wireless sensor networks," Journal of AI and Data Mining, vol. 3, no. 1, pp. 47-57, 2015.

[9] A. Ghaffari, "Designing a Wireless Sensor Network for ocean status notification system," Indian Journal of Science and Technology, vol. 7, no. 6, pp. 809-814, 2014.

[10] P. Cheng, X. Cao, J. Bai, and Y. Sun, "On optimizing sensing quality with guaranteed coverage in autonomous mobile sensor networks," Computer Communications, vol. 35, no. 9, pp. 1107-1114, 2012.Torkestani, J. A. (2013). An adaptive energy-efficient area coverage algorithm for wireless sensor networks. Ad hoc networks, 11(6), 1655-1666.

[11] G. Fan and S. Jin, "Coverage problem in wireless sensor network: A survey," JNW, vol. 5, no. 9, pp. 1033-1040, 2010 .

[12] A. Ghaffari, "Real-time routing algorithm for mobile ad hoc networks using reinforcement learning and heuristic algorithms," Wireless Networks, pp. 1-12, 2016.

[13] A. Ghosh, "Estimating coverage holes and enhancing coverage in mixed sensor networks," in Local Computer Networks, 2004. 29th Annual IEEE International Conference on, 2004, pp. 68-76: IEEE.

[14] W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," IEEE Transactions on wireless communications, vol. 1, no. 4, pp. 660-670, 2002.

[15] H. Karl and A. Willig, Protocols and architectures for wireless sensor networks. John Wiley & Sons, 2005.

[16] J. Li, L. L. Andrew, C. H. Foh, M. Zukerman, and H.-H. Chen, "Connectivity, coverage and placement in wireless sensor networks," Sensors, vol. 9, no. 10, pp. 7664-7693, 2009.

[17] M. Liu, J.-N. Cao, G.-H. Chen, L.-J. Chen, X.-M. Wang, and H.-G. Gong, "EADEEG: An energy-aware data gathering protocol for wireless sensor networks," Ruan Jian Xue Bao/Journal of Software, 2007.

[18] R. Mulligan and H. M. Ammari, "Coverage in wireless sensor networks: A survey," Network Protocols and Algorithms, vol. 2, no. 2, pp. 27-53, 2010.

[19] O. Younis and S. Fahmy, "HEED: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks," IEEE Transactions on mobile computing, vol. 3, no. 4, pp. 366-379, 2004.

[20] Z. Mottaghinia and A. Ghaffari, "A Unicast Tree-Based Data Gathering Protocol for Delay Tolerant Mobile Sensor Networks," Information Systems & Telecommunication, vol. 59, no. 25, pp. 1-12, 2016.

[21] G. Xing, X. Wang, Y. Zhang, C. Lu, R. Pless, and C. Gill, "Integrated coverage and connectivity configuration for energy conservation in sensor networks," ACM Transactions on Sensor Networks (TOSN), vol. 1, no. 1, pp. 36-72, 2005.

[22] Z. Xinlian, W. Min, and X. Jianbo, "BPEC: an energy-aware distributed clustering algorithm in WSNs," J. Comput. Res. Dev, vol. 46, no. 5, pp. 723-730, 2009.

[23] R. Masoudi and A. Ghaffari, "Software defined networks: A survey," Journal of Network and Computer Applications, vol. 67, pp. 1-25, 2016.

[24] Z. Liu, Q. Zheng, L. Xue, and X. Guan, "A distributed energy-efficient clustering algorithm with improved

coverage in wireless sensor networks," Future Generation Computer Systems, vol. 28, no. 5, pp. 780-790, 2012.
[25] J. Yu, Y. Chen, L. Ma, B. Huang, and X. Cheng, "On connected target k-coverage in heterogeneous wireless sensor networks," Sensors, vol. 16, no. 1, p. 104, 2016.
[26] M. Akhlaq, T. R. Sheltami, and E. M. Shakshuki, "C3: an energy-efficient protocol for coverage, connectivity and communication in WSNs," Personal and Ubiquitous Computing, vol. 18, no. 5, pp. 1117-1133, 2014.
[27] N.-T. Le and Y. M. Jang, "Energy-efficient coverage guarantees scheduling and routing strategy for wireless sensor networks," International Journal of Distributed Sensor Networks, 2015.

Amir Pakmehr received the M.Sc degree in Computer Engineering from Tabriz Azad University, Iran in 2014, respectively. He is currently a Ph.D student of Computer Engineering in Qazvin Islamic Azad University (QIAU), he researches about Wireless Sensor Networks, His research interests include Wireless Networks, and Cover Energy Saving in Wireless Sensor Networks.

Ali Ghaffari received his B.Sc M.Sc and Ph.D degrees in computer engineering from the University of Tehran and IAUT (Islamic Azad University), Tehran, Iran in 1994, 2002 and 2011 respectively. As an assistant Professor of computer engineering at Islamic Azad University, Tabriz branch, IRAN, his research interests are mainly in the field of wired and wireless networks, Wireless Sensor Networks (WSNs), Mobile Ad Hoc Networks (MANETs), Vehicular Ad Hoc Networks (VANETs), Software Defined Networks (SDNs), networks security and Quality of Service (QoS). He has published more than 60 international conference and reviewed journal papers.