J. Basic. Appl. Sci. Res., 2(12)12124-12133, 2012

ISSN 2090-4304
Journal of Basic and Applied Scientific Research
www.textroad.com

Hardware Implementing of a New Decoding Algorithm HGSCA for HMM-Based Speech Recognition Systems

Mohammad Mosleh[1], Saeed Setayeshi[2], Mohammad Kheirandish[3]

[1,3]Department of Computer Engineering, Faculty of Postgraduate Studies, Dezful Branch, Islamic Azad University Dezful, Iran
[2]Faculty of Physics and Nuclear, Engineering Amirkabir University, Tehran, Iran

ABSTRACT

Since Hidden Markov Model (HMM) can model speech signal dynamicity, it has been used in speech recognition field successfully. The HMM-based speech recognition systems can compute the similarity between an input pattern and a reference model, with using decoding algorithms, Baum-Welch and Viterbi. Since such algorithms are based on dynamic programming (DP), they include many computations. In this paper, at first, a new decoding algorithm in terms of collaborating HMM and Genetic Algorithm (GA) has been introduced and then the steps of its hardware implementation will be presented. Hardware implementation of the proposed decoding algorithm on Field Programmable Gate Array (FPGA) leads to create fine grain architecture which can perform decoding operations in a parallel and distributed manner.
KEYWORDS: Automatic Speech Recognition (ASR), Hidden Markov Model (HMM), Genetic algorithm (GA), Field Programmable Gate Array (FPGA).

INTRODUCTION

In few past decades, the speech processing has been applied in a wide variety of applications. Automatic speech recognition (ASR), as the most important branch, has been paid attention by researchers. This topic relates to pattern classification problem, basically. The speech signal dynamicity has caused this problem not to be introduced as a simple classification problem. The block diagram of a statistic speech recognizing system has been shown in Fig. 1.

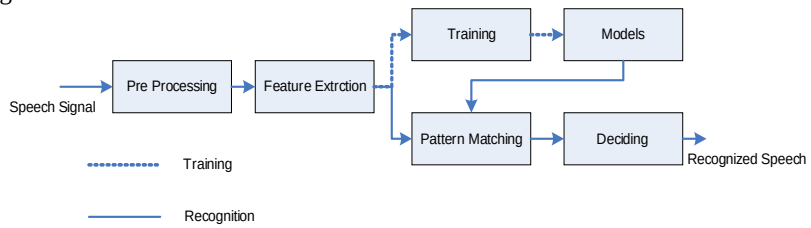


Fig. 1. Block diagram of a statistic speech recognizing system

As shown in Fig. 1, each speech recognizing system operates in two following modes: training mode and recognition mode. In these two modes, initially, preprocessing operations are performed on the speech signal and convert it to a set of frames. Afterward, the feature extraction operation on each extracted frame is done, and finally the speech patterns are formed.

In the training mode, the reference models are created by using learning algorithms and then are stored. In recognition mode, pattern matching is performed between stored reference patterns and input speech pattern, and then its class is identified. Up to now, several methods have been presented for speech recognition. For example, Dynamic Time Wrapping (DTW)[1], Hidden Markov Model (HMM)[2], Artificial Neural network-based methods[3, 4], Support Vector Machine-based methods (SVM)[5,6, 7], Fuzzy-based methods[8, 9] and hybrid methods[10, 11,12] can be mentioned. Among these methods, the speech recognition systems based on HMM have been known as the best systems; because this approach can model speech signal dynamicity, well. With using decoding algorithms such as Baum-Welch or Viterbi, the HMM can compute the similarity between input observation sequence and stored reference models. These algorithms are based on dynamic programming method, thus they impose massive computations. In order to execute such algorithms in a real-time manner, several different hardware architectures have been presented, up to now [10,11, 12, 13, 14, 15, 16, 17, 18, 19, 20]. For example, some hardware implementations for decoding algorithm, Baum-Welch, will be reviewed in the following. Elmisery et al. could implement this algorithm on FPGA, by assuming a fixed value for its recursive part[10]. Also, Surjuan Ke et al. used a soft core processor, Micro Blaze, for this purpose[11]. In [12],

Corresponding Author: Mohammad Mosleh, Department of Computer Engineering, Dezful Branch, Islamic Azad University, Dezful, Iran. **Tel**:+ 986416260601, Fax:+986416260890, Email: Mosleh@iaud.ac.ir

we presented a linear systolic array based implementation for this algorithm. Furthermore, some attempts can be made for hardware implementing the Viterbi decoding algorithm. S. J. Melinkof et al. presented a hardware implementation on FPGA[13]. S. Yoshizawa et al. proposed an extendable architecture for parallel execution of this algorithm[14]. Kisun You et al. have used a soft core processor, Micro Blaze, for this purpose[15]. In [16], Gin DW and Kuo, KT have presented a Dual-ALU with RISC architecture. V. Amudha et al. implemented viterbi algorithm on soft core processor, NIOSII[17]. Liu, H et al. implemented an English speech recognition system on a chip, called speech system-on-chip (SoC)[18]. In 2011, an embedded speech recognition system was implemented to base on the Cyclone II of FPGA and the soft-core processor of NIOS-II by Zhang, G et al. [19]. Geng, H. et al. proposed a master-slave SoC structure composed of an ARM7-TDMI and a co-processor for Mahalanobis distance calculation. The SoC was implemented on an Actel ProASIC series M7A3P1000 FPGA [20].

As common problems for most of above-stated implementations, the inexistence of a regular hardware with parallel and pipeline capabilities, executing computation in a sequential manner and also inexistence of a pruning strategy for complexity reduction can be indexed. In this paper, at first, we introduce a novel methodology, called HGSCA, for decoding operations in the HMM-based speech recognition. This method collaborats HMM and GA on Stochastic Cellular Automata (SCA). Then the hardware implementation steps for the proposed algorithm HGSCA on FPGA will be presented. The hardware implementing the HGSCA algorithm leased to a highly regular structure, consisting of identical and simple processing elements. The design is very scalable, and also it is easily extensible. Such features provide optimum hardware resources causing lower area and power consumption.

The structure of this paper is as follow. In section 2, some backgrounds will be covered including HMM, GA and SCA. Section 3 has been assigned to introduce the proposed decoding algorithm, HGSCA. In section 4, the hardware implementation stages of HGSCA method on FPGA will be explained. The experiment results will be presented in section 5 and finally, section 6 will include conclusion.

2. Background

In this section, the principles and preliminaries for this research will be introduced, briefly.

2.1. Hidden Markov Model

A *N*-state HMM with *K* observations assigned to each state is defined by a triple $<\Pi , A , B>$, where $\Pi =\{\pi\}_{N\times 1}$ specifies initial state probabilities, $A=\{a_{ij}\}_{N\times N}$ is transition probabilities matrix and $B=\{b_{ik}\}_{N\times K}$ is observation probabilities matrix. The decoding procedure in Markov Chain is performed by Baum-Welch or Viterbi algorithms[2].

For an observation sequence $O= \{o_1, o_2, o_3 \ldots O_T\}$ and an HMM model defined with $\lambda=<\Pi, A, B>$, the Baum Welch and Viterbi algorithms are defined as follows:

The Baum-Welch decoding:

- Initialization

$$\alpha_1(j) = \pi(j).b_j(o_1) \qquad j = 1,2,..., N \tag{1}$$

- Recursion

$$\alpha_{t+1}(j) = \left(\sum_{i=1}^{N} \alpha_t(i).a_{ij}\right).b_j(O_{t+1}) \quad j = 1,2,...N, t = 1,2,,...T-1 \tag{2}$$

- Termination

$$P^{BW} = \sum_{i=1}^{N} \alpha_T(i) \tag{3}$$

The Viterbi decoding:

- Initialization

$$\varphi_1(j) = \pi(j).b_j(O_1) \quad j = 1,2,..., N \tag{4}$$

- Recursion

$$\varphi_{t+1}(j) = Max\left(\varphi_t(i).a_{ij}\right)b_j(O_{t+1}) \quad j = 1,2,..., N, t = 1,2,..., T-1 \tag{5}$$

- Termination

$$P^{V} = Max\left(\varphi_T(i)\right) \qquad i = 1,2,..., N \tag{6}$$

2.2 Genetic Algorithm

The genetic algorithm has been inspired from the nature and based on natural evolution principles, directs a population of individuals toward the evolution. Each individual in this population is a potential solution for optimization problem. In oppose to other techniques that focus on one solution, only, the genetic algorithm

operates on a set of possible solutions, concurrently. In each optimization problem, the optimization evaluation criterion for obtained solutions is expressed by a function, called quality function. The genetic algorithm is executed on an encoded representation of solutions, called chromosome, and assigns a quality score to each solution by the quality function. It is obvious that the individuals with higher quality scores have more chances for survival and reproduction and are expected to appear in next generations. By using selection, crossover and mutation operators on the current generation, the genetic algorithm can form the next generation[21].

2.3 Stochastic Cellular Automata

Cellular Automata (CA) was introduced by John Von Neumann for the first time and afterward was introduced by Stanislas Ulam as a model for investigating the complex systems behaviors. The CA is a mathematical model for systems in which several simple components cooperate to produce more complex patterns. The CAs are discrete dynamic systems that their behaviors are established based on local communications. In CA, there are a regular set of cells that each of them can be assigned multiple different values. These cells are updated in parallel discrete times and according to a local rule. In determining a new value for each cell, the neighbor cells values are considered, too. Selecting different rules for updating, has presented different types of CAs. Since these rules can be expressed in a determined or probabilistic manner, two types of CA named Deterministic Cellular Automata (DCA) and Stochastic Cellular Automata (SCA) have been defined[22].

3. Presenting the Proposed Decoding Method

The proposed HGSCA method can perform decoding operations in a parallel and distributed form, by taking advantages of a synergy between HMM and GA over SCA.

The basic manner in this method includes transforming decoding problem inside the Markov Chain to genetic algorithm domain, initially. In other words, the main purpose in this step includes applying genetic algorithm to find the best possible path, inside the grid formed by matching the inputted observation sequence $O= \{o_1, o_2, o_3 \ldots O_T\}$ with an N-state HMM model (as seen in Fig. 2)

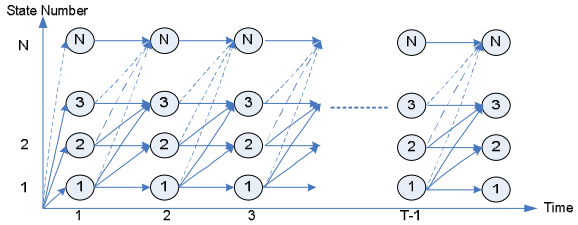


Fig. 2. The grid formed by matching an observation sequence with an N-state HMM model

For this purpose, the following encoding, selection, crossover and mutation mechanisms have been proposed.

A. Encoding Mechanism. In this step, each path represented in Fig.2 has been coded as a chromosome that starts in $t=1$ and terminates in $t=T$. The genes in each chromosome are defined as triplets $p_i = (t_i , s_i , f_i)$, where $1 \leq t_i \leq T$ and $1 \leq S_i \leq N$ denote the time step and the state number, respectively. As well as f_i denotes the score assigned to i'th gen. In order to impose the restrictions in the grid, the pair <time , state number> is defined as Eq. (7) and Eq. (8).

$$t_i = i \qquad i = 1,2,...,T \tag{7}$$

$$s_i = j, \qquad s_{i-1} \leq j \leq N, 1 \leq i \leq T \tag{8}$$

$$s_0 = 1$$

In order to evaluate the quality score for each sequence defined by a chromosome, fitness function is defined as follows:

$$f_i = f_{i1}(t_i, s_i) + f_{i2}\left((t_i, s_i)\middle|(t_{i-1}, s_{i-1})\right) \tag{9}$$

$$f_{i1}(t_i, s_i) = -\log\left(b_{s_i}(o_{t_i})\right) \qquad 1 \leq t \leq T,\ 1 \leq s_i \leq N$$

$$f_{i2}\left((t_i, s_i)\middle|(t_{i-1}, 0)\right) = -\log\left(\Pi(s_i)\right) \qquad t = 1$$

$$f_{i2}\left((t_i, s_i)\middle|(t_{i-1}, s_j)\right) = -\log\left(A(s_j, s_i)\right) \qquad 2 \leq t_i \leq T,\ 1 \leq s_j \leq N,\ s_j \leq s_i \leq N$$

Also, the total cost for each chromosome is determined as follows:

$$F_i = \sum_{t=1}^{T} f_t, \qquad 1 \leq i \leq P \tag{10}$$

where P is the total number of chromosomes.

B. The Selection Mechanism. Since the SCA will be applied for parallel executing GA, the nature of selection procedure is competitive. In order to avoid the local minima problem, the Simulated Annealing (SA) technique[23] has been used. The pseudo-code for proposed method has been presented as Fig. 3.

```
Procedure Selection (A chromosome; List of th 12126 romosomes; Initial temperature (T0); Iteration (n))
Begin
        Randomly choose B from the list of neighboring chromosomes;
        Δ= (Fitness value of B)-(Fitness value of A);
        If Δ≤0 then
                B is parent;
        Else
                T=0.95.T0;
                Y=Exp (-Δ/T);
                Randomly generate x between 0-1;
                If x < y then
                        B is parent;
                Else
                        A is parent;
                End if
        End if
End
```

Fig.3. the selection procedure pseudo-code

C. The Crossover Mechanism. The crossover rate is controlled by a probability factor, P_c, which controls the balance between the rate of exploration, the new recombined building block and disruption rate of qualified individuals. In this paper, single point crossover method has been used. The pseudo-code for crossover procedure has been shown in Fig. 4.

```
Procedure Crossover (A chromosome; List of the neighboring chromosomes)
Begin
        Randomly generate x between 0-1;
        If x≤ Pc then
                A is child;
        Else
                Randomly choose gene gcross from A chromosome
                For i=gcross+1 to end of the chromosome do
                   Selecting the best gene from list of the neighboring chromosomes where minimize the cost fi;
                End for
                Modify the cost A chromosome;
        End if
End
```

Fig.4. The pseudo-code for proposed crossover procedure

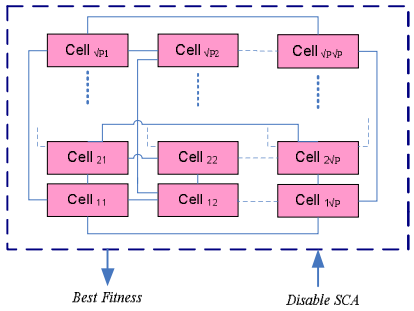
D. The Mutation Mechanism. This stage can speed up the algorithm convergence to optimal solution, by injecting optimal sub-paths into exist paths. The mutation rate is also controlled by a probability factor, P_m, controlling the balance between random searching and disruption rate of qualified individuals. Fig. 5 shows the correspond pseudo-code.

```
Procedure Mutation (A chromosome)
Begin
        Randomly generate x between 0-1;
        If x≤ PM then
                Randomly choose two genes gs and ge from A chromosome where gs<ge
                Generate a new sub path between these genes as following
                For t=gs to ge do
                   At(fit)=Min[-log(a(st-1,:))-log(b(:,Ot))];
                   At(st)=Argmin[-log(a(st-1,:))-log(b(:,Ot))];
                End for;
                Modify the cost A chromosome;
        End if
End
```

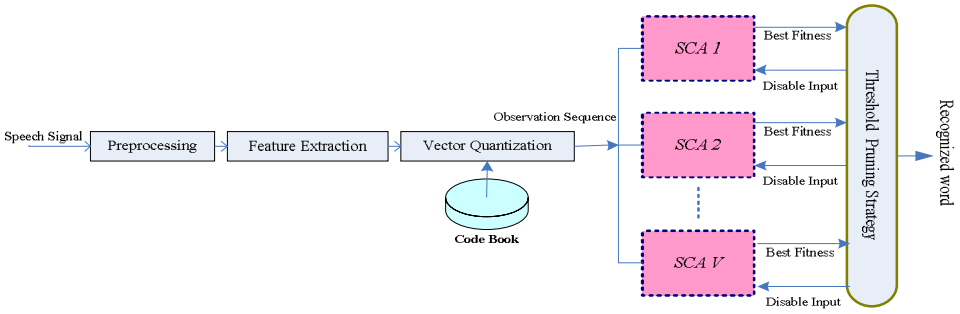
Fig. 5 The proposed mutation pseudo-code

In order to realize the parallel execution for proposed decoding procedure, the SCA presented in Fig. 6 (a) has been used. This SCA has *P* cells and follows Von Neumann neighborhood. One chromosome is put on each cell that is able to pursuit the optimal solution in parallel and by selection, crossover and mutation operations between itself and its neighbors. It must be no 12127 h SCA, an input named *Disable Input*, and an output named *Best Fitness* is considered. The Disable Input can inactivate the SCA and the *Best Fitness* can present the best chromosome cost in each time step.

In Fig. 6 (b), the block diagram of the proposed system for Isolated Speech-Recognition, with *V* reference words has been shown. As it can be seen, for each reference word, one SCA has been considered that can compute the similarity between input observation vector and each reference word HMM with using proposed described manner.



(a)



(b)

Fig. 6 Block diagram of the proposed method for Isolated Speech Recognition

In order to increase the performance, a threshold pruning strategy has been proposed. The basis for such a mechanism relies on this fact that the words with lower matching to input patterns are dropped from the competition, gradually. In each time step, all SCA units select the best chromosome and send their cost to *Threshold Pruning Strategy* unit, through the *Best Fitness* output. Then, this unit is able to form its dynamic threshold value, by computing the average of best fitness values, and the half value of their standard deviation. Finally, it will create a bit pattern by which it will be possible to deactivate all units that their best chromosome costs are higher than this threshold. Then these units will be out from competition. Therefore, such a mechanism can enhance the performance, by creating an intra-word competition. The pruning value will continue to P_p percent of reference words so that the algorithm can execute, correctly. P_p has been introduced as pruning ratio and is known as a parameter of proposed algorithm. The pseudo-code for proposed method has been shown in Fig. 7.

```
Procedure the proposed HGSCA method;
    P_C ( Crossover rate ) , P_M ( Mutation rate ), T_0 ( Initial temprature ), Templat_no ( Number of reference words ),
    Flag[1 : Templat_no ] = 1; P_p (pruning rate)
Begin
        For each word (i)  do in parallel form
            For each cell  in the SCA (i) do in parallel form
                Randomly generate a chromosome T_j = [(t1,s1,f1),(t2,s2,f2),…,(tT,sT,fT)];
            End for
        End for
        iter=1;
        While iter = Max_iter do
        Begin
            For each word (i) where Flag [i] =1 do in parallel form
                For each cell (j) in the SCA (i) do in parallel form
                    cell (j)= Select ( cell( j), neighboring cells, T, iter );
                    cell (j)=Cross over (cell( j), neighboring cells );
                    cell (j)=Mutation (cell( j));
                End for
                Result (i) =Fitness of the best Chromosome in the SCA (i);
            End for
            [Avg, S_D]= Computing average and standard deviation of the "Result" array;
            Index=Find ( Result ( : ) > Avg +  (S_D /2 ) );
            If  ( Templat_no - Length (Index) ) = ( P_p Templat_no)  then
                Flag (Index) =0;
            End if
            iter =iter+1;
        End while
End
```

Fig. 7 The pseudo-code for proposed method

4. The Implementation and Experiments Results

The software implementation – In order to verify the correctness of HGSCA algorithm operation, it has been implemented in a software framework. As well as it has been evaluated and compared with Baum-Welch and Viterbi decoding algorithms. For this purpose, the database FARSDAT has been used. Each frame is divided into a set of overlapping frames after preprocessing, and 13 MFCC features along with their energy and dynamic values have been extracted from each frame. The codebook used for the proposed algorithm has been computed by Kmeans algorithm with 256 clusters. In order to model the words, a right-to-left 6-state HMM has been used. For each reference word, 100 samples and 20 samples have been used for training and testing, respectively. In order to compute optimized parameters for the HGSCA method, several different experiments have been done from recognition accuracy and recognition time viewpoints. The results of evaluating eight word set for several algorithm execution have been shown in Table1.

Table1. Comparison of HGSCA, Baum Welch and Viterbi decoding algorithms in terms of recognition rate and recognition time

No. words	DECODING ALGORITHMS					
	Viterbi Decoding		**Baum Welch Decoding**		**HGSCA Decoding**	
	recognition rate (mean ± standard deviation)	recognition time (ms)	recognition rate (mean ± standard deviation)	recognition time (ms)	recognition rate (mean ± standard deviation)	recognition time (ms)
3	99.56 ± 0.38	0.65	98.11 ± 0.12	0.69	99.8 ± 0.20	0.20
6	95.85 ± 0.37	1.61	94.23 ± 0.28	1.63	95.15 ± 0.24	0.47
9	93.34 ± 0.04	2.47	91.87 ± 0.12	2.50	92.42 ± 0.14	0.74
12	91.26 ± 0.02	3.23	89.82 ± 0.15	3.23	90.37 ± 0.95	0.95
15	89.84 ± 0.28	3.93	87.12 ± 0.31	3.95	87.36 ± 0.63	1.20
18	89.45 ± 0.90	4.54	87.03 ± 0.42	4.54	87.17 ± 0.92	1.38
21	85.48 ± 0.13	5.15	83.17 ± 0.21	5.20	85.22 ± 0.06	1.50
24	84.68 ± 1.35	5.82	82.19 ± 0.93	5.85	83.92 ± 1.12	1.67

As can be seen in Table 1, the experimental results show that although the HGSCA decoding method is very close toViterbi and Baum-Welch decoding algorithms, regarding the recognition rate, it is so better than them.

Hardware Implementation – In this section, hardware implementing the HGSCA algorithm by using VHDL language on FPGA will be considered. For this purpose, the Register Transmission Language (RTL) for all parts of HGSCA decoding method should be extracted, initially. Fig.8 shows the processing unit, RTL. The relevant RTLs to Selection, Crossover and Mutation units have been shown in Fig 9. Also, the RTL for Threshold Pruning unit has been presented in Fig.10.

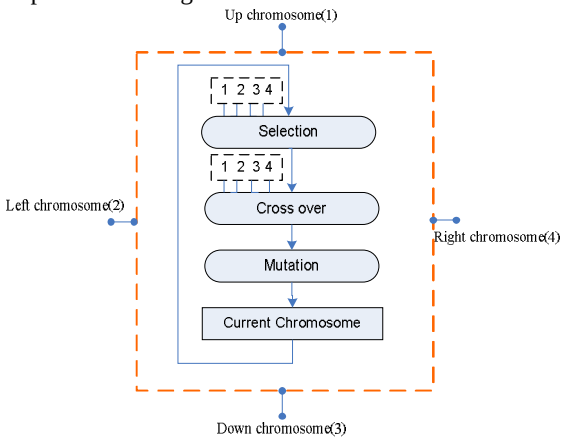
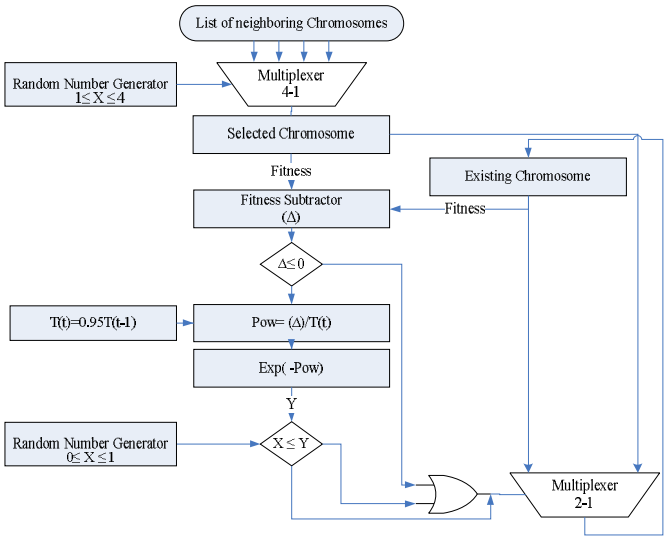


Fig 8. The processing element RTL



(a)

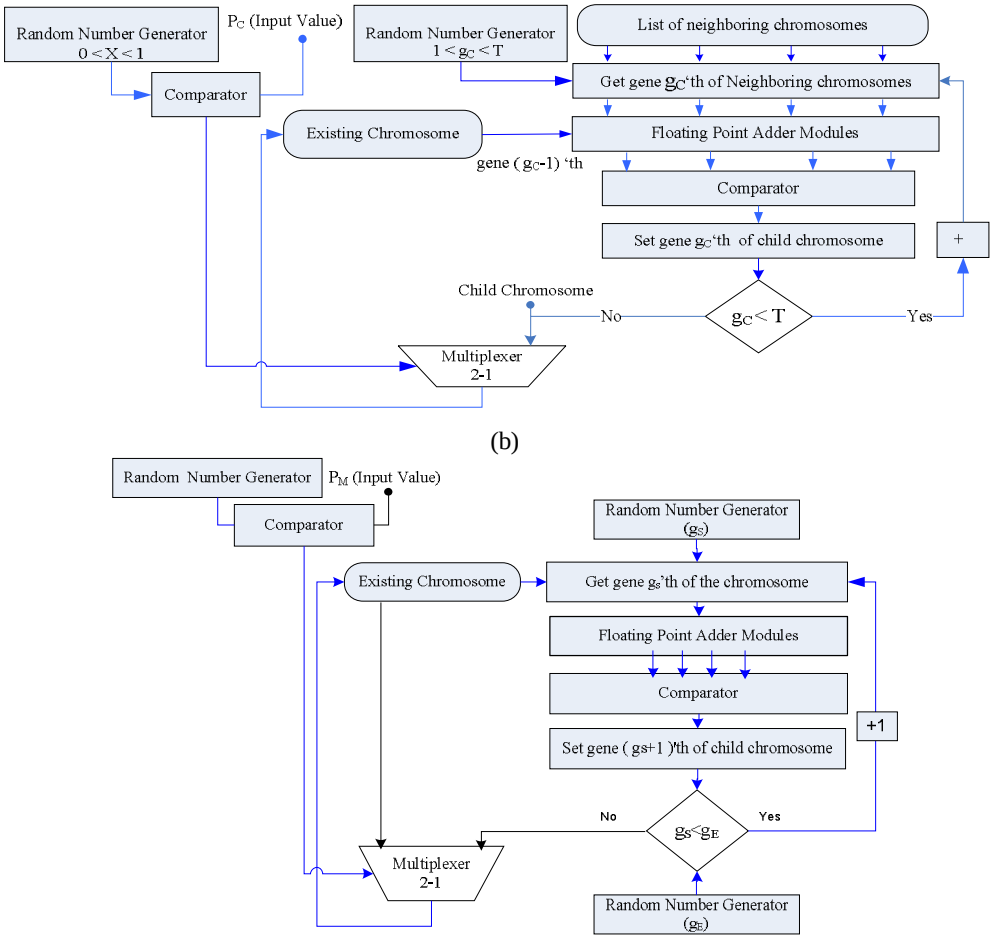


Fig 9. (a) The Selection unit RTL (b) The Cross over unit RTL (c) The Mutation unit RTL

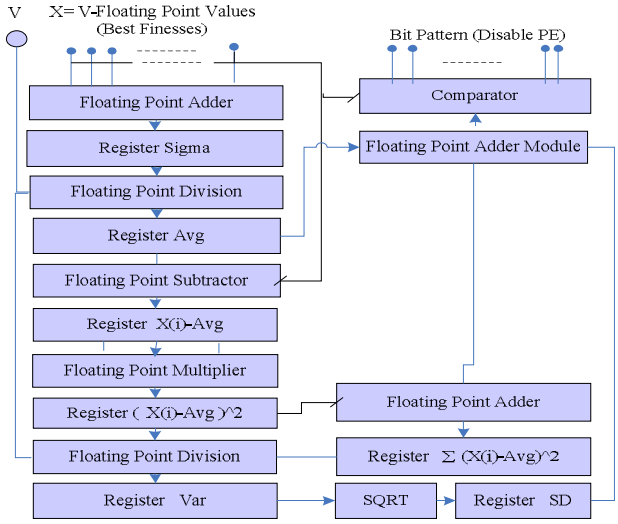


Fig 10. The Threshold Pruning unit RTL

All computations are done in floating point form and based on standard IEEE754. The floating point Adder / Subtractor, Multiplier and Divider modules have been implemented based on register transmission language, proposed in[24] . Random Number Generating (RNG) is done by LFSRs that are shift registers that their inputs are linear functions of two or more register bits. In order to compute the exponential function in selection unit,

and also the square root function in Threshold Pruning unit, the approximations presented in Eq. 11 and Eq. 12 have been used, respectively.

$$e^x = 1 - x + 0.5x^2 \quad (11)$$

$$\sqrt{x} = 1 - 0.5y - 0.25y^2 \quad \text{where } y = 1 - x \quad (12)$$

The proposed system has been designed and simulated with VHDL and in ModelSim6.4 software framework, fully tested and error free. For synthesizing it, the software ISE8.2 has been used. The results of synthesizing this algorithm over FPGA XC4VLX15 from Vertix4 family have been reported in Table 2.

Table 2. Synthesizing results of the HGSCA decoding method

Module	Number of slices	Number of LUTs	Delay(ns)
Floating Point Adder	139	241	6.96
Floating Point Multiplier	22	39	5.42
Floating Point Divider	23	39	5.42
RNG	62	108	8.12
SQRT	23	40	5.12
Exponent	277	483	15.85
Selection	3351	6088	260.36
Cross over	8648	52	155.36
Mutation	8486	6	132.88
Threshold Pruning	62	120	3.64
Processing Element	17308	34360	480.22

5. Conclusion

In this paper, a new decoding approach by collaborating HMM and GA based on SCA, here called HGSCA, has been presented. This approach is able to perform decoding operations in speech recognizer systems, in parallel and distributed forms. The HGSCA method is able to accelerate recognizing process with creating competition among reference words, by applying threshold pruning strategy. Hardware implementation of the HGSCA method has caused highly regular and fine grain architecture that employing it in embedded systems can bring out low area and power. For future works, the HGSCA decoding method can be implemented on soft core processors such as Micro Blaze or NIOSII and is used on the real-time speech recognition systems well.

REFERENCES

1. Rabiner, LR, and BH Juang, 1993. Fundamentals of speech recognition: Prentice Hall.

2. Rabiner, L.R, 1989. A tutorial on hidden Markov models and selected applications in speech recognition. Proceedings of the IEEE 7(2): pp 257-286.

3. Lippmann, RP, 1989. Review of neural networks for speech recognition. Neural computation 1 (1): 1-38.

4. Lang, KJ, AH Waibel, and GE Hinton, 1990. A time-delay neural network architecture for isolated word recognition. Neural Networks 3 (1):23-43.

5. Smith, N, and M Gales, 2002. Speech recognition using SVMs. Advances in Neural Information Processing Systems 2:1197-1204.

6. Moniri, A, F Razzazi, and E Kabir, 2005. An SVM Based Acoustic Model for Isolated Word Speech Recognition. International Symposium on Telecommunications at Shiraz,Iran.

7. Solera-Urena, R., J. Padrell-Sendra, D. Martin-Iglesias, A. Gallardo-Antol n, C. Pel ez-Moreno, and F. Diaz-de-Maria, 2007. SVMs for automatic speech recognition: a survey. Progress in nonlinear speech processing: pp190-216.

8. Su, MC, CT Hsieh, and CC Chin, 1998. A neuro-fuzzy approach to speech recognition without time alignment. Fuzzy Sets and Systems 98 (1):33-41.

9. Halavati, R., S.B. Shouraki, and S.H. Zadeh, 2007. Recognition of human speech phonemes using a novel fuzzy approach. Applied Soft Computing 7 (3):828-839.

10. Elmisery, FA, AH Khalil, AE Salama, and HF Hammed, 2003. A FPGA-based HMM for a discrete Arabic speech recognition system. ICM2003, pp:322-325

11. Sujuan Ke, Yibin Hou, Zhangqin Huang, and Hui Li, 2008. A HMM Speech Recognition System Based on FPGA. Congress on Image and Signal Processing, pp:305-309

12. Mosleh, M., S. Setayeshi, M Lotfinejad, and A Mirshekari, 2010. FPGA Implementation of a Linear Systolic Array for Speech Recognition Based on HMM. ICCAE,pp:75-78

13. Melnikoff, SJ, SF Quigley, and MJ Russell, 2002. Implementing a simple continuous speech recognition system on an FPGA. Proceedings Field-Programmable Custom Computing Machines, pp: 275 – 276

14. Yoshizawa, S, N Wada, N Hayasaka, and Y Miyanaga, 2006. Scalable architecture for word HMM-based speech recognition. IEEE/SMC International Conference on System of Systems Engineering, pp: 62-74

15. Kisun You, Hyunjin Lim, and and Wonyong Sung, 2006. Architectural Design and Implementation af an FPGA Softcore Based Speech recognition. The 6th International Workshop on System on Chip for Real Time Applications, pp:50-55

16. Gin DW, Kuo, KT, 2006. Dual-ALU Structure Processor for Speech Recognition, IEEE/SMC International Conference on System of Systems Engineering,pp:194-196

17. Amudha, V, and B Venkataramani, 2009. Software/Hardware Co-design of HMM Based Isolated Digit Recognition System. Journal of Computers 4 (2):154.

18. Liu, H., Y. Qian, and J. Liu, 2011. English Speech Recognition System on Chip. Tsinghua Science & Technology 16 (1):95-99.

19. Zhang, G., J. Yin, Q. Liu, and C. Yang, 2011. A real-time speech recognition system based on the Implementation of FPGA. Cross Strait Quad-Regional Radio Science and Wireless Technology Conference (CSQRWC), pp:1375-1378

20. Geng, H., Y. Shi, M. Dong, and R. Liu, 2012. A master-slave SoC structure for HMM based speech recognition. International Symposium on VLSI Design, Automation, and Test (VLSI-DAT), pp:1-4

21. Haupt, SE, 2004. Practical genetic algorithms: Wiley-Interscience.

22. Bar Yam, Y, 2003. Dynamics of complex systems: Westview Pr.

23. Kirkpatrick, S, 1984. Optimization by simulated annealing: Quantitative studies. Journal of Statistical Physics 34 (5):975-986.

24. Koren, Israel, ed, 1993. Computer Arithmatic Algorithms: Prentice-Hall, Inc. Upper Saddle River, NJ, USA