Grammar Induction Strategy Using Genetic Algorithm: Case Study of Fifteen Toy Languages.

Nitin S. Choubey, Ph.D.[*1] and Madan U. Kharat, Ph.D.[2]

[1]Student. P.G. Department of Computer Science, S.G.B.A. University, Amravati, Maharastra, India.
[2]Professor, Department of Computer Engineering, Institute of Engineering, Bhujbal Knowledge City, Nashik, Maharastra, India

[*]E-mail: nschoubey@gmail.com

ABSTRACT

Grammar Induction (or Grammar Inference or Language Learning) is the process of learning of a grammar from training data of the positive and negative strings of the language. The paper discusses an extended approach of using stochastic mutation approach based on Adaptive Genetic Algorithm for the induction of the grammar for a set of fifteen different languages. In this approach, proportionate amount of the population is generated by crossover and mutation operators separately. The elite members from the resultant population and the original population are considered for inclusion in the next population.

(Keywords: evolutionary computation, genetic algorithm, automata, context free grammar, grammar induction)

INTRODUCTION

Genetic Algorithms (GAs) were invented by John Holland in the 1960s. Wyard [1] explored the impact of different grammar representations and experimental results show that an evolutionary algorithm using standard context-free grammars (BNF) outperformed other representations.

In the conventional grammatical induction, a language acceptor is constructed to accept all the positive examples. Learning from positive examples is called text learning. A more powerful technique uses negative samples as well. This is learning with an informant. In informant learning, the language acceptor is constructed so as to accept all the positive examples and reject all the negative examples. The field of evolutionary computing has been applying problem-solving techniques that are similar in intent to the Machine Learning recombination methods. Most evolutionary computing approaches hold in common that they try and find a solution to a particular problem, by recombining and mutating individuals in a society of possible solutions [2].

In formal language theory, a context-free grammar (CFG) is a grammar, in which every production rule is of the form [3],

$$V \rightarrow w \qquad \textbf{(1)}$$

where, V = single non-terminal symbol
w = string of terminals and/or non-terminals (possibly empty)

The term "context-free" expresses the fact that non-terminals can be rewritten without regard to the context in which they occur. A formal language is context-free if some context-free grammar generates it. These languages are all languages that can be recognized by a non-deterministic pushdown automata.

This paper discusses a brief overview of the Genetic Algorithm, a strategy adopted for CFG Induction with Genetic Algorithm, the details of the Languages used in the implementation undertaken by the authors for CFG induction with Genetic Algorithm, and a discussion on the results obtained, respectively.

GENETIC ALGORITHM

A simple GA works by creating a random initial population of fixed length chromosomes. Each iteration (generation), the population evolves by means of the use of selection, crossover and mutation, which are the main genetic operators in GAs. Individuals are chosen based on their fitness measure to act as parents of offspring which will constitute the new generation. This process is repeated until the termination criterion is satisfied.

GENETIC ALGORITHM METHOD USED

The Genetic Algorithm method used by the authors for the purpose of experiment creates the sub-section of the intermediate population by using the crossover and mutation method separately and merges them with the original population to get next population. The method is shown in Figure 1.

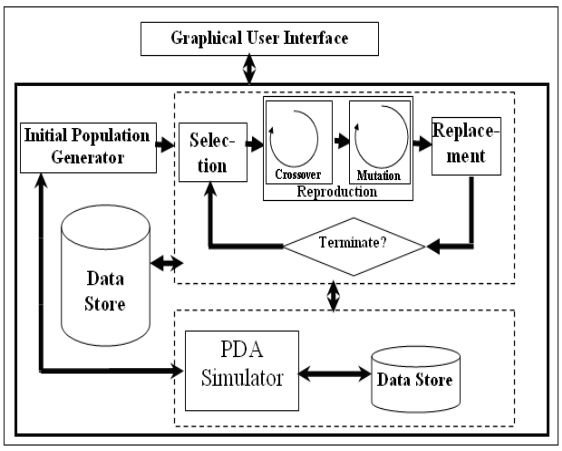

Figure 1: The Genetic Algorithm Method used [8].

The crossover and mutation operators adapted for the purpose of experimentation are shown in Figure 2. A variant of the cyclic crossover is used. A random mask is used in the mutation operator. The mutation operator flips every bit corresponding to '1' bit in the mask. The method is based on Adaptive Genetic Algorithm (AGA) in which the parameters, such as the population size, the crossing over probability, or the mutation probability are varied while the GA is running [4]. The Mutation probability in the experiment is decided by the random nature of the mask which leads to the stochastic behavior of the mutation operator.

The chromosome is decoded from the binary chromosome by using the sequential chromosome method biased towards the generation of the variable on the left side of each production rule in the every grammar [5, 6, 7]. The process of grammar construction equivalent to the binary chromosome is shown is Figure 3.

The fitness function utilizes the number of valid parse in the corpus conducted by the PDA Simulator [8] and the length grammar generated [7, 8].

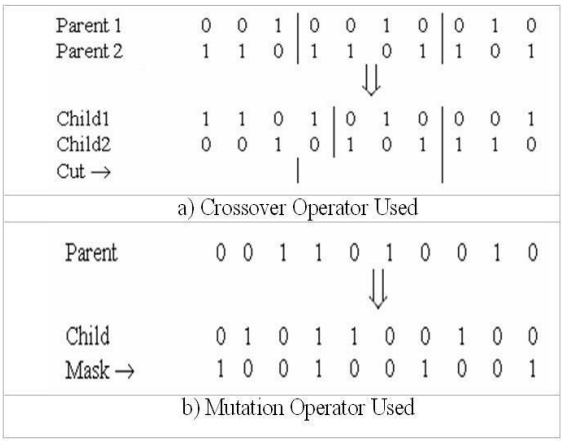

Figure 2: The Crossover and Mutation Operators Used [6].

BC	000 000 001 000 011	000 000 110 011 000	000 000 100 100 001	000 010 101 110 001
S C	SSASC	SS?CS	SS00A	S?1?A
V/I	I	I	I	I
ER	Discarded	Discarded	Discarded	Discarded

BC: Binary Chromosome; SC: Equivalent Symbolic Chromosome; V/I: Valid/Invalid; ER: Equivalent Rule

BC	[illegible]	[illegible]	000 000 011 000 001	011 000 001 111 011
S C	S????	?S1AA	SSCSA	CSABC
V/I	V	I	I	I
ER	S->?	Discarded	Discarded	Discarded

BC: Binary Chromosome; SC: Equivalent Symbolic Chromosome; V/I: Valid/Invalid; ER: Equivalent Rule

BC	000 011 010 000 010	000 010 101 100 000	[illegible]	011 001 010 100 001
S C	SC?S?	S?10S	ABASS	CA?0A
V/I	I	V	I	I
ER	Discarded	S->10S	Discarded	Discarded

BC: Binary Chromosome; SC: Equivalent Symbolic Chromosome; V/I: Valid/Invalid; ER: Equivalent Rule

BC	000 001 001 110 011	001 010 011 100 001	010 001 100 110 011	[illegible]
S C	SAA?C	A?C0A	?A0?C	?B0S?
V/I	I	I	I	I
ER	Discarded	Discarded	Discarded	Discarded

BC: Binary Chromosome; SC: Equivalent Symbolic Chromosome; V/I: Valid/Invalid; ER: Equivalent Rule

Figure 3: The Grammar Construction Equivalent to the Binary Chromosome [7, 8].

The fitness of an individual chromosome increases for every accepted positive sample and the rejected negative samples whereas it decreases in proportion to the every rejected positive sample and accepted negative sample. A factor inversely proportional to the number of rules available in the grammar also has an important role in the calculation of fitness. The Grammar which accepts all the positive samples and rejects the entire negative sample set from the corpus with minimum number of rules is considered to be the best grammar.

THE LANGUAGES SET USED

The Languages used for the purpose of experiment are listed in the Table 1:

Table 1: The Languages Used.

Language (L_i)	Language Description
L1	$\{ 0^n1^n, n\geq 0\}$.
L2	0* over (0+1)*.
L3	(10)*.
L4	Balanced Parentheses Problem.
L5	0*1 over (0+1)*.
L6	0(00)*1.
L7	Odd Binary number.
L8	All strings even number of 0 over (0+1)*.
L9	Even Binary number.
L10	Any String with even 0 and odd 1 over (0+1)*.
L11	$\{ 0^n1^{2n}n\geq 0\}$.
L12	(00)*10*.
L13	All string not containing '000' over (0+1)*.
L14	(00)*(111)*.
L15	Palindrome over {a, b}.

The languages chosen for the experiment are the collection of context free language as well as Regular Language.

RESULTS AND DISCUSSION

The experiment was done with JDK 1.4 on a Intel Core™2 CPU with 1.8 GHZ and 1 GB RAM. The Population size = 50, Chromosome size = 240, The Corpus size includes the set of 50 positive and negative strings for the language and the maximum number of generation are 400 for the experiment. The minimum length description principle (MLDP) [9] is used to generate the corpus of positive and negative samples. The training set and the test set required for the language learning is generated with the length 'L' (L = 0, 1, 2…) such that it covers all the possible valid strings with the length L till the sufficient number of valid string for corpus is generated. All the invalid strings generated during this procedure are considered as negative strings. The validity of the generated string is tested with the best known available grammar.

The Result set generated is given in the Table 2. The Grammars shown are the grammar, equivalent of the binary chromosome (Figure 3), with the fitness value (FV) accepting all the positive examples and rejecting the negative examples considered for the experiment. The Best Grammar is represented as <V, T, P, S>, where V is finite set of Variables, T is finite set of Terminals, P is finite set of Production rules and S is a starting Variable. The Generation Charts for the average of first ten successful run of the grammar induction process for the various languages used is given in Figure 4 and Figure 5.

Table 2: The Resultant Grammar and Its Fitness Value.

L_i	FV	The Equivalent Grammar
L1	1012	<{S, M}, {0, 1}, {S→M, S→0S1M, M→?}, S >.
L2	1013	< { S }, { 0, 1 }, { S→?, S→0S }, S>.
L3	1013	< { S}, { 0, 1}, { S→10S, S→? }, S >.
L4	1011	< { S, M }, { (,)}, { S→(M, M→S)M, M→?, M→)M }, S >.
L5	1013	< { S }, { 0, 1}, { S→1, S→0S }, S >.
L6	1011	<{S, L, C}, { 0, 1}, { S→0L, L→C, L→0S, C→1 }, S >.
L7	1011	< { S, M }, { 0, 1 }, { S→1M, S→0SM, M→SM, M→? }, S >.
L8	1010	<{S, M}, {0,1}, {S→M, S→1SSM, S→0S0M, M→?, M→1M }, S >.
L9	1011	< { S, L }, { 0, 1 }, { S→1S, S→0L, L→S, L→? }, S >.
L10	1008	< { S, M, K }, { 0, 1}, { S→1K, S→0SM0, M→?, M→0M0, K→1S1M, K→M, K→0M0 }, S >.
L11	1013	<{S, M}, {0, 1}, { S→?, S→0S11 }, S >.
L12	1011	<{S, M }, {0, 1 }, { S→1M, S→00S, M→?, M→ 0M }, S >.
L13	1009	< { S, M, K}, {0, 1}, { S→M, S→0K, M→?, M→1SM, K→M, K→0M }, S >.
L14	1011	< { S, M }, { 0, 1}, { S → M, S → 00SM , M → ?, M → 111M } , S >.
L15	1010	< {S, J}, {a, b}, { S→ bJ, S→ aSa, S→?, J→b, J→ Sb }, S >.

The method found to converge on the local optimum. Data about the Effective Grammar Induction (EGI), the Local Optimum Convergence (LOC), the Average time required per generation run and the number of Generations (Range, Mean and Standard Deviation) is shown in the Table 3.

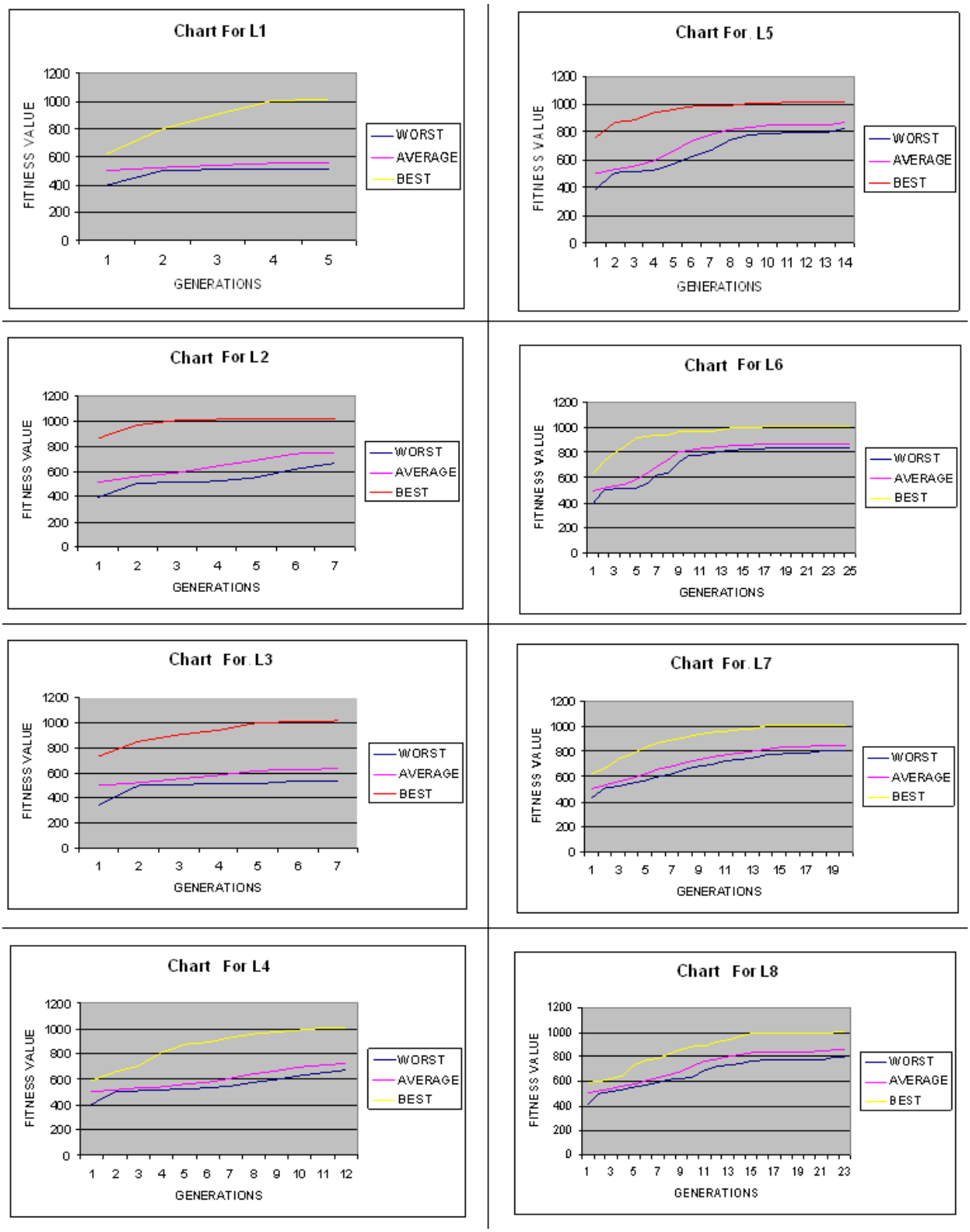

Figure 4: The Generation Charts for the Languages L1 through Language L8.

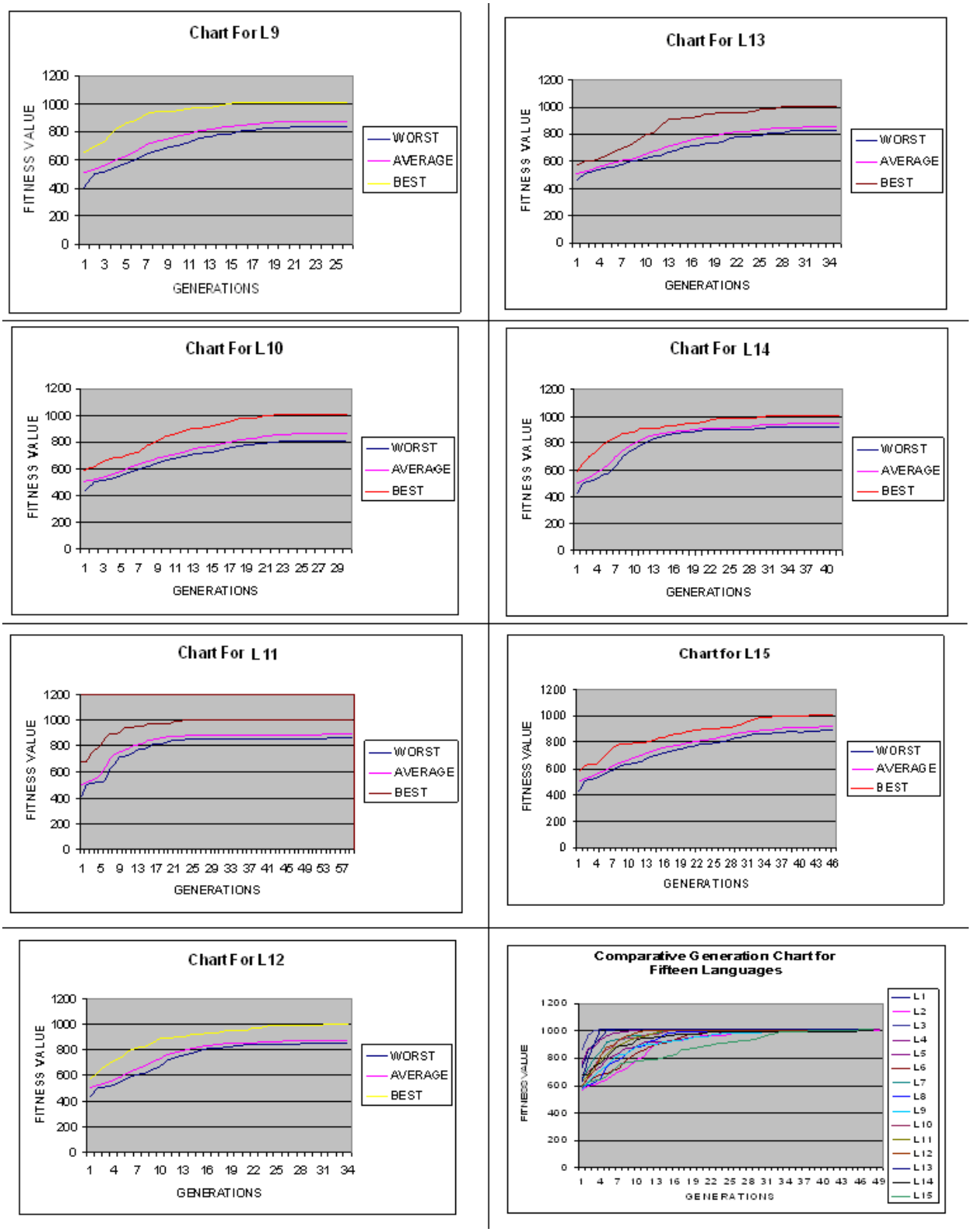

Figure 5: The Generation Charts for the Languages L9 through Language L15 and Comparison Chart for all the Languages.

Table 3: Statistical Data Analysis for the Experiment.

L_i	Total Runs	LOC	EGI %	LOC %	Time/ Generation*	Generations		
						Range	Mean	S.D.
L1	11	01	90.9%	9.1%	30340.35	4±1	3.2	1.03
L2	10	00	100%	00%	12081.24	5±2	4.4	2.17
L3	10	00	100%	00%	23364.91	5±2	4.8	1.135
L4	10	00	100%	00%	70310.04	8±5	8.3	3.83
L5	12	02	83.3%	16.6%	23268.87	8±6	8.7	3.83
L6	19	09	52.6%	47.4%	59746.92	12±10	11.9	5.54
L7	10	00	100%	00%	54465.83	12±9	11.9	5.10
L8	14	04	71.4%	28.6%	55957.27	15±7	14.6	4.5
L9	10	00	100%	00%	46751.62	16±10	15.1	7.14
L10	13	03	76.9%	23.1%	490693.5	20±8	19.5	5.77
L11	17	07	58.8%	41.2%	181977.4	31±25	20.54	14.62
L12	16	06	62.5%	37.5%	69389.5	19±13	22.5	8.91
L13	11	01	91.9%	9.09%	100621.5	23±12	22.8	8.85
L14	17	07	58.8%	41.1%	126158.9	27±15	27.3	8.93
L15	12	02	83.3%	16.6%	462196.2	29±18	32.7	10.07

**Average Time in milliseconds*

CONCLUSION

Experiment found to be working successfully and efficiently for the light weight grammar examples. There is further scope for adoption of the same method for more complex grammar sets. MLDP is found to be more effective in the selection of the corpus. The sample set used in the experiment is limited to the size with the MLDP. The selection of the good quality corpus (positive and negative string inputs) has resulted into induction of good quality grammar for the languages considered. There is further scope for adoption for larger length description of the corpus data set. Results have shown tendency towards the local optimum convergence which requires special attention in future work**.**

ACKNOWLEDGMENTS

We sincerely extend our acknowledgements to Dr. V. M. Thakare, P.G. Department of Computer Science, Sant Gadge Baba Amravati University, Amravati, Maharastra, India, for his kind support in providing Laboratory infrastructural facility required for carrying out the research work.

REFERENCES

1. Wyard, P. 1994. "Representational Issues for Context-Free Grammar Induction Using Genetic Algorithm". *Proceedings of the 2nd International Colloquium on Grammatical Inference and Applications, Lecture Notes in Artificial Intelligence*. 862:222-235.

2. Guy De Pauw. 2003. "Evolutionary Computing as a Tool for Grammar Development". CNTS – Language Technology Group, UIA, University of Antwerp: Antwerp, Belgium. Springer-Verlag Berlin Heidelberg.

3. Hopcroft, J.E., Motwani, R., and Ullman, J.D. 2007. *Introduction to Automata Theory, Languages, and Computation. 3/E.* Addison-Wesley: New York, NY.

4. Sivanandam and Deepa. 2008. *Introduction to Genetic Algorithm*. Springer: Berlin, Germany.

5. Rodrigues, E. and Lopes H.S. 2007. "Genetic Programming for Induction of Context-free Grammars". *Seventh International Conference on Intelligent Systems Design and Applications*. IEEE.

6. Choubey N.S. and Kharat M.U. 2009. "Grammar Induction and Genetic Algorithms- An Overview". *Pacific Journal of Science and Technology*. 10(2):884-889.

7. Choubey, N.S. and Kharat, M.U. 2010, "Sequential Structuring Element for CFG Induction Using Genetic Algorithm". *International Journal of Futuristic Computer Application*. 1(1): 12-16, February 2010. Foundation of Computer Science.

8. Choubey, N.S. and Kharat M.U. 2010. "PDA Simulator for CFG Induction Using Genetic Algorithm". *International Conference on Simulation*

and Modelling. UKSIM-2010, Cambridge, U.K. Unpublished.

9. Kellller, B. and R. Lutz. 1997. "Evolving Stochastic Context Free Grammars from Examples using Minimum Description Length Principle". *Workshop on Automata Induction Grammatical Inference and Language Acquisition*, ICML-97, 1997.

ABOUT THE AUTHORS

N. S. Choubey, BE, ME, MBA, Ph.D. (Management) was educated at Sant Gadge Baba Amravati (SGBA) University, Amravati India and also holds a Diploma in TQM & ISO 9000. He is pursuing a Ph.D. program in faculty of Computer Science & Engineering from SGBA University, Amravati, Maharastra, India. Presently he is working at Mukesh Patel School of Technology Management and Engineering at S.V.K.M.'s Mukesh Patel Technology Park, Shirpur, Dhule, Maharastra, India, as an Associate Professor and Head of the Computer Engineering Department He has presented papers at National/International conferences and also published papers in National/International Journals on various issues of Computer Engineering and Management. To his credit, he has published books on various topics in Computer Science and Management subjects. His areas of interest include Algorithms, Theoretical Computer Science, and Computer Networks and Internet.

M. U. Kharat, BE, MS, Ph.D. was educated at Sant Gadge Baba Amravati (SGBA) University, Amravati, India. Presently he is working at the Institute of Engineering, Bhujbal Knowledge City, Nashik, Maharastra, India, as Professor and Head of the Computer Engineering Department. He has presented papers at National and International conferences and also published papers in National and International Journals on various aspects of Computer Engineering and Networks. He has worked in various capacities in academic institutions at the level of Professor, Head of Computer Engineering Department, and Principal. His areas of interest include Digital Signal Processing, Computer Networks, and Internet.

SUGGESTED CITATION

Choubey N.S. and Kharat M.U. 2010. "Grammar Induction Strategy Using Genetic Algorithm: Case Study of Fifteen Toy Languages". *Pacific Journal of Science and Technology*. 11(1):294-300.

Pacific Journal of Science and Technology