Journal of Engineering Education Transformations,
Volume No 36, January 2023, Special issue, eISSN 2394-1707

10.16920/jeet/2023/v36is2/23006

Use of FOSS for Teaching the Parser Types to Improve Problem Solving Ability A Case Study on Analysis of Gender based Discrimination in Education

Sunita M Dol
Assistant Professor, Computer Science and Engineering, Walchand Institute of Technology, Solapur P.B.No.634, Walchand Hirachand Marg, Ashok Chowk, Solapur - 413006

***Abstract*— Compiler Design is one of the core courses of Computer Science and Engineering. Syntax Analyzer or Parser is important phase in designing the compiler. So while designing this phase of the compiler, students must have thorough knowledge about the types of parser such as LL(1), SLR, Canonical LR and LALR. In the parser phase, the parsing table is built for given Context Free Grammar (CFG). While implementing this parsing phase in laboratory session, two Free Open Source Software (FOSS) - JFLAP and Parsing Emulator, are considered to explain the types of parser and building the parsing table for the CFG using these parser types.**

JFLAP is software for experimenting with formal languages topics including nondeterministic finite automata, nondeterministic pushdown automata, multi-tape Turing machines, several types of grammars, parsing, and L-systems. So this tool is considered for parsing phase of Compiler. JFLAP tool is used to explain the steps to be followed to solve the given problem using the parsing technique.

Parsing simulator is a software which implements the parsing table for the Context Free Grammars in tool. This simulator is used to generate parsing table (LL1, SLR, LR, and LALR). Parsing Emulator tool is used for practicing the problem statement given in the tool.

In current study, use of FOSS such as JFLAP and Parsing Emulator is considered for teaching parser types to improve problem solving ability. The Learning Objectives (LOs) of this study are: To build the parsing table for given CFG using LL(1) and SLR parser (LO1) and to parse the given string using LL(1) and SLR parser (LO2). The research questions for this study are – whether the use of FOSS improves the problem solving ability of the students? and whether gender bias is there in education?

Two groups post-test method is considered to check the effectiveness of this use of FOSS in learning the parser types. Also the students' perception about this method is also considered. The result shows that the students' problem solving ability of experimental group is improved as compared to the control group irrespective of gender.

Keywords- FOSS (Free Open Source Software) JFLAP, Parsing Emulator, Bloom's Taxonomy, Post-test, t-Test.

I. INTRODUCTION

In any engineering programme, there are some core courses and students should have the theoretical as well as practical knowledge of these core courses for a successful engineer. In Computer Science and Engineering programme, there are many core courses, out of which Compiler Design course is one of the core course. This course is about the detailed study of phases of compiler. Parser which is also called as Syntax Analysis or Hierarchical Analysis is the second phase of compiler.

In current study, use of FOSS such as JFLAP and Parsing Emulator for the topic – Construction of parsing table using LL(1) and SLR, is considered which is considered to be difficult to understand from students' point of view. Hence the FOSS is used to check whether students understand this topic and also to check students' problem solving ability. This method is also used to check whether there gender bias in education?

II. RELATED WORKS

There are various approaches to make the Compiler Design course interesting. The software tool LISA is described in this study which facilitates learning and conceptual understanding of compiler construction in an efficient, direct, and long-lasting way (Marjan Mernik, 2003).

The learning environment AtoCC as described by Michael Hielscher and Christian Wagenknecht is used in teaching automata and some of its applications in compiler construction which address a broad range of different learning activities forcing the students to actively interact with the subjects being taught.

Divya Kundra and Ashish Sureka (2016) implemented a Case-based and Project-based Learning environment for teaching important Compiler design concepts (CPLC)

A. Demaille, R. Levillain and B. Perrot (2008) introduced a set of tools especially designed or im- proved for compiler construction educative projects in C++

T. R. Henry presented the results of using a domain specific language in an upper division compiler course

The paper authored by M. Ruckert presented an unusual programming language, textttklx, illustrating the type of compiler construction projects the author uses

successfully to accompany a mostly traditional lecture on compilers while E. White presented a novel approach that enables students in graduate compiler courses to examine and experiment with a real compiler without becoming overwhelmed by complexity

Keshav Pingali and Gianfranco Bilardi presented a graphical representation of context-free grammars called the Grammar Flow Graph (GFG) which permits parsing problems to be phrased as path problems in graphs.

So there is no study which shows use of FOSS to teach Compiler Design course with analysis of Gender based Discrimination in Education.

III. METHODOLOGY

A. JFLAP Tool

JFLAP is software free open source software used for experimenting with formal languages and Compiler design topics such as finite automata, Mealy machine, pushdown automata, Turing machines, several types of grammars, parsing, etc. It also does the following conversion:

- Nondeterministic Finite Automata (NFA) → Deterministic Finite Automata (DFA) → Minimal DFA
- NFA ↔ regular expression
- NFA ↔ regular grammar
- Push-down Automata PDA → Context Free Grammar (CFG)
- CFG → PDA (LL parser)
- CFG → PDA (SLR parser)
- CFG → Chomsky Normal Form (CNF)
- CFG → LL parse table and parser
- CFG → SLR parse table and parser
- CFG → Brute force parser

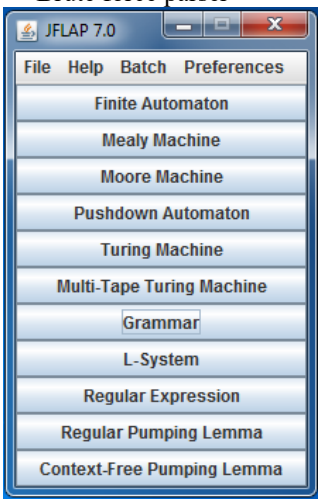

Fig.1. JFLAP Tool

B. Parsing Emulator

Parsing simulator is a software used for generating the parsing table for given CFG using LL1, SLR, LR, LALR technique.

Fig.2. Parsing Emulator

C. Problem Solving using JFLAP Tool

In this section, steps to construct the parsing table and parse the string of given CFG using SLR technique is considered.

Constructing the parsing table for given CFG using LL(1) technique consist of following steps

1. Compute the FIRST set for given CFG.
2. Compute the FOLLOW set for given CFG.
3. Construct the parsing table using LL(1) technique.

Constructing the parsing table for given CFG using SLR technique consist of following steps

4. Compute the FIRST set for given CFG.
5. Compute the FOLLOW set for given CFG.
6. Compute the LR(0) items for given CFG.
7. Draw the Deterministic Finite Automata (DFA) for LR(0) items.
8. Construct the parsing table using SLR method.
9. Parse the string of given CFG using parsing table.

The above steps for constructing the parsing table and parsing the string using SLR technique using JFLAP tool is given below:

Step 1: Go to JFLAP tool as shown in Figure 1.

Step 2: Click on the button ‘Grammar’.

Step 3: Enter the productions of CFG as shown in Figure 3: In this tool, the size of the font can be increased or decreased as per requirement using ‘Table Text Size’.

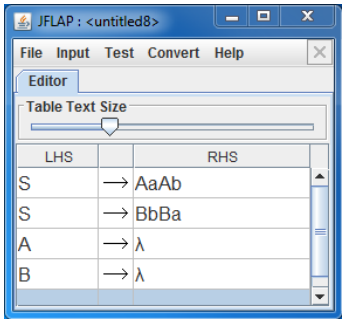

Fig.3. Entering the productions in JFLAP

Step 4: Click on 'Input' in main menu bar and select 'Build SLR(1) Parse Table' to construct the parsing table for given CFG as shown below in figure 4.

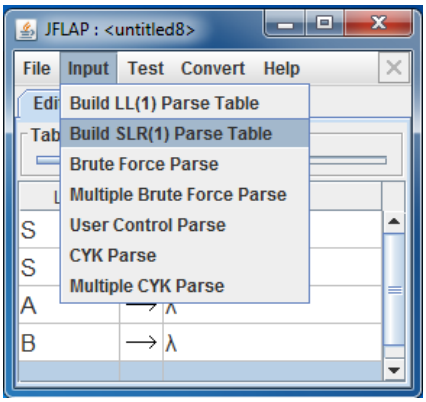

Fig.4. Step4

Step 5: Click on 'Do Step' for computing FIRST set for given CFG.

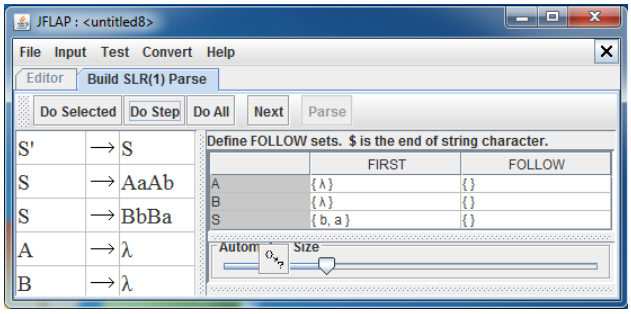

Fig.5. Computing FIRST set

Step 6: Click once again on 'Do Step' for computing FOLLOW set for given CFG.

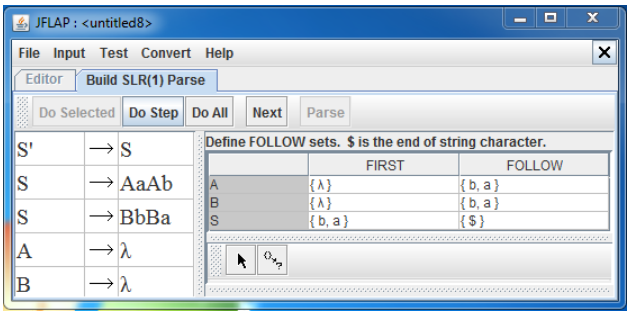

Fig.. Computing FOLLOW set

Step 7: Next step is to compute the LR(0) items for given CFG. To compute the LR(0) items, closure of all attribute is computed. As shown in Figure 7, the closure of each attribute can be check as shown in Figure 7.

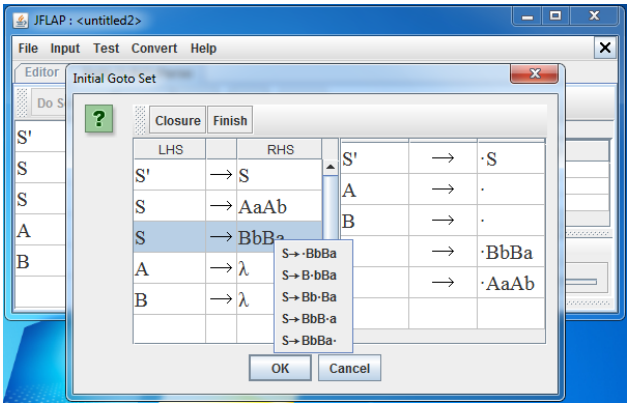

Fig.7. Computing LR(0) items

Step 8:The next step in construction of parsing table for given CFG is draw the DFA for LR(0) items. Once again after clicking on 'Do Step', the DFA is drawn step by step in JFLAP as shown in Figure 8.

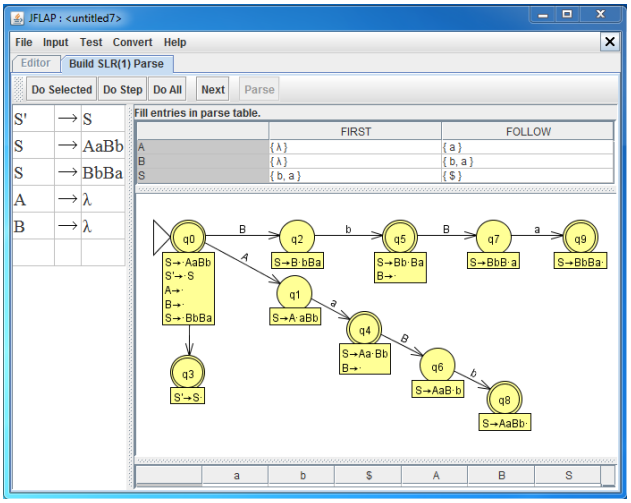

Fig.8. DFA for LR(0) items

Step 9: Next step is to construct the parsing table from DFA as shown in Figure 9.

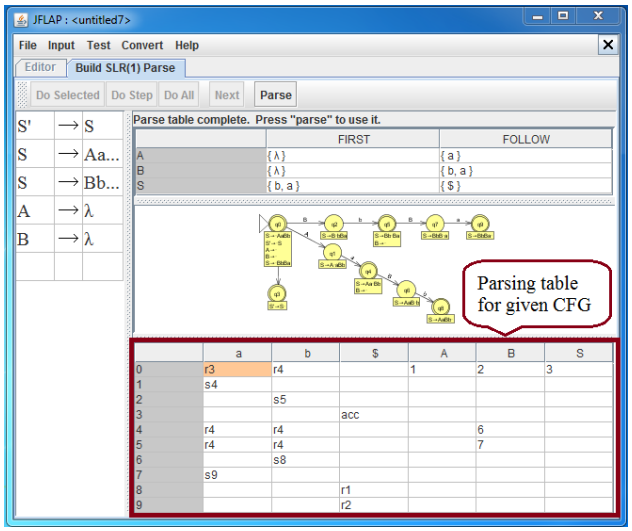

Fig.9. Constructing the Parsing Table

Step 10: Next step is to parse the string of the CFG using parsing table. There are following three ways to represent the parsing of the string using parsing table as shown in Figure 10, 11 and 12.

- Noninverted Tree
- Inverted String
- Derivation Table

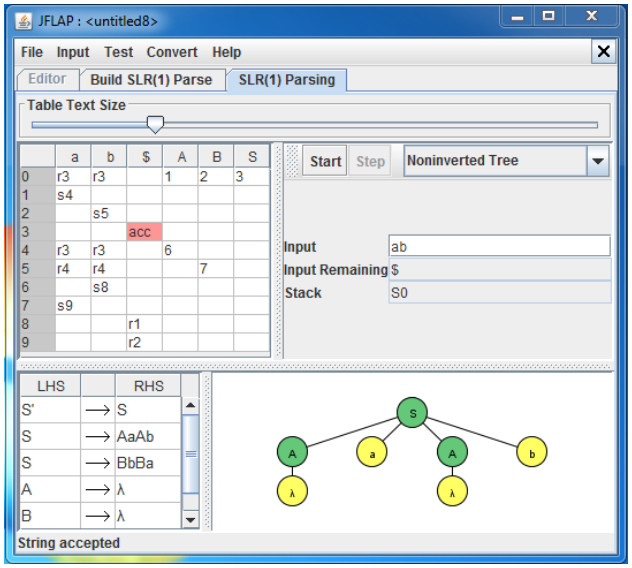

Fig.10. Noninverted Tree for string of CFG

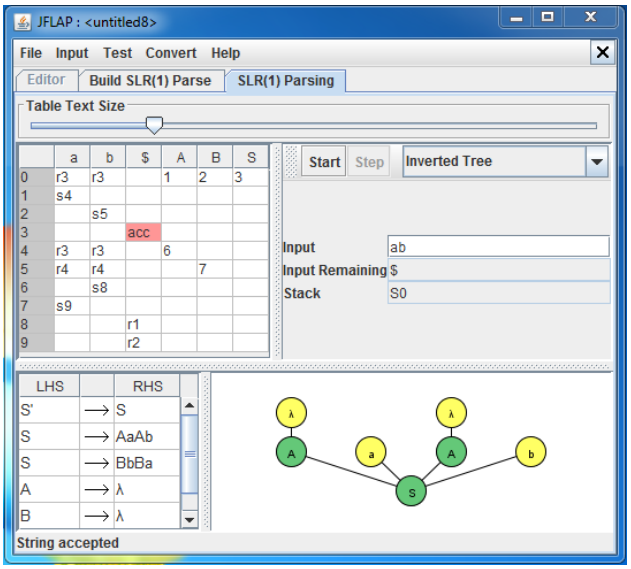

Fig.11. Inverted Tree for string of CFG

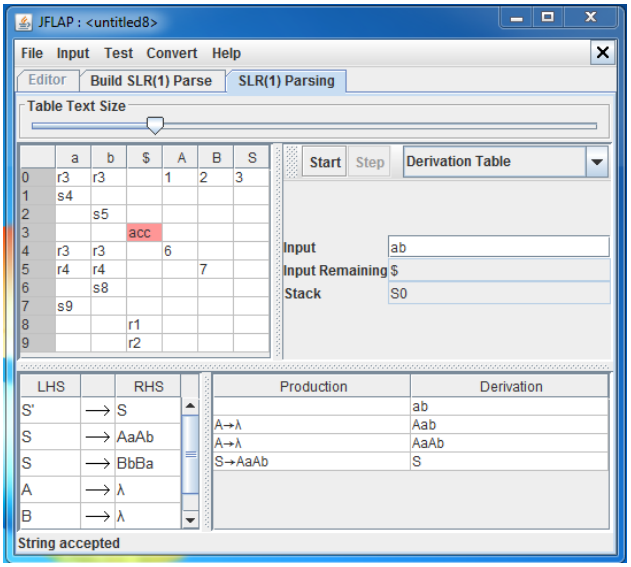

Fig.12. Derivation Table for string of CFG

D. Practice Problem using Parsing Emulator

Parsing emulator is free open source software which contains the solution for 12 CFGs. Students practice these problem statements on this tool.

Following are the steps to practice the construction of parsing table using SLR parsing technique:

Step 1: Open the parsing emulator software by clicking on ParsingEmu.exe file and load the example as shown in Figure 13.

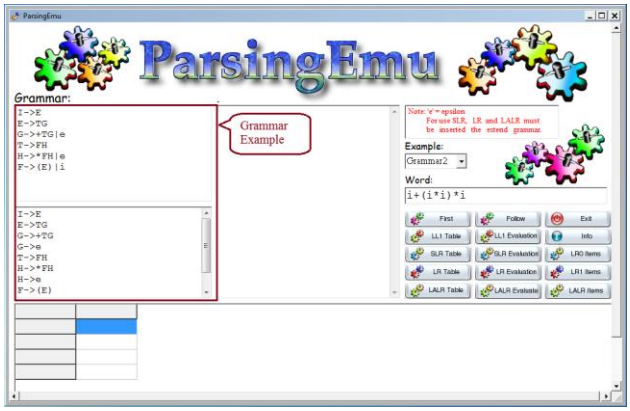

Fig.13. Parsing Emulator

Step 2: Compute the FIRST set, FOLLOW set and LR(0) items as shown in Figure 14.

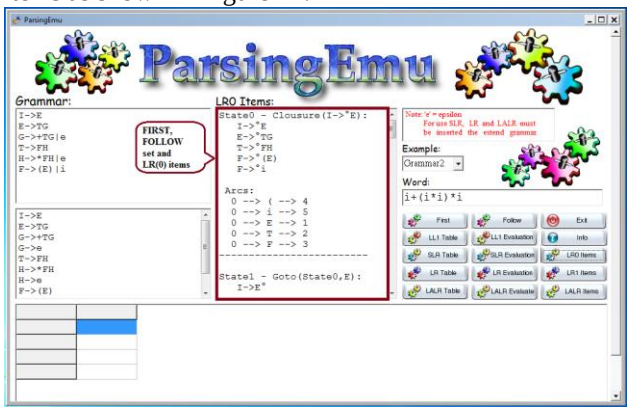

Fig.14. FIRST set, FOLLOW set and LR(0) items

Step 3: Construct the parsing table for given CFG as shown in Figure 15.

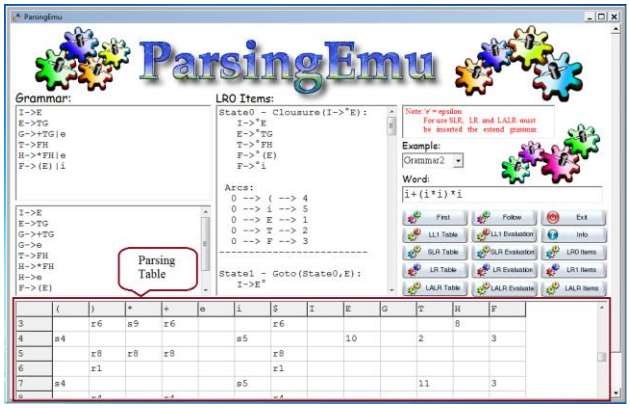

Fig.15. Construction of parsing table

Step 4: Parsing the string of CFG using parsing table is checked on this tool as shown in Figure 16.

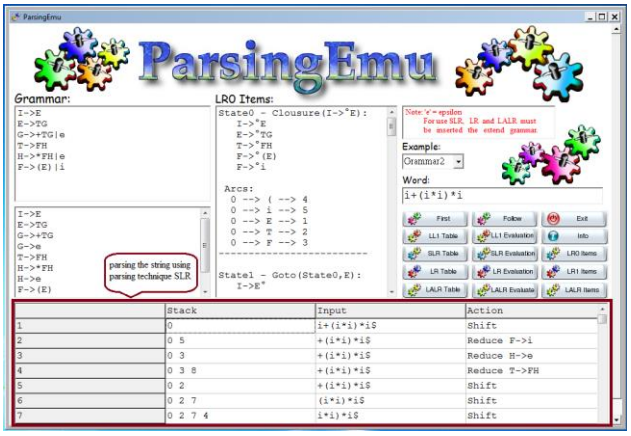

Figure 16: Parsing the string using SLR technique

IV. Experimental Details

A. Experimental Setup

The experimental details are given below:

- Technique used: Use of FOSS
- Course considered: Compiler Construction of Third Year Computer Science and Engineering
- Sample: Two groups of each 30 students of Third Year Computer Science and Engineering as shown in Table 1– each group contains
    - 15 boys
    - 15 girls
- Method: Two group post-test method
- Instruments used: Post-test and Feedback
- Learning domain used: Bloom's Taxonomy

TABLE I
EXPERIMENTAL AND CONTROL GROUP

Class↓	Experimental Group			Control Group		
	No. of boys	No. of girls	Total	No. of boys	No. of girls	Total
Distinction	4	4	8	4	4	8
First	6	6	12	6	6	12
Second	4	3	7	4	3	7
Pass	1	2	3	1	2	3
	Total		30	**Total**		30

B. Instruments used

1. Post-Test

Post-test were conducted for 30 marks. All questions considered in the post-test, covered Cognitive Level - Apply of Bloom's Taxonomy. The sample test question is given below.

Consider the following context free grammar:
S -> AaAb|BbBa
A->ε
A -> ε
a. Compute the FIRST set
b. Compute the FOLLOW set
c. Construct the parsing table using LL(1) technique
d. Parse the string ab using LL(1)
e. Compute the LR(0) items for given CFG
f. Draw the Deterministic Finite Automata (DFA) for LR(0) items.
g. Construct the parsing table using SLR method.
h. Parse the string ba using SLR

The post-test was considered for 60 marks and consist of 3 questions of above type, each of which was for 20 marks. Rubric used to check each question of the post-test is given below:

TABLE II
RUBRIC FOR EVALUATION OF POST-TEST

Sr. No.	Questions	Marks given	Blooms' Taxonomy Level - Cognitive
1	Compute the FIRST set	3	Apply Level
2	Compute the FOLLOW set	3	
3	Construct the parsing table using LL(1) technique	4	
4	Parse the string ab using LL(1)	2	
5	Compute the LR(0) items for given CFG	6	
6	Draw the Deterministic Finite Automata (DFA) for LR(0) items.	4	
7	Construct the parsing table using SLR method.	6	
8	Parse the string ba using SLR	2	
	Total	30	

C. Feedback

Feedback was conducted at the end of the activity to know about the perception of students related to the use of FOSS. Feedback is given in the Table 2.

TABLE III
FEEDBACK ABOUT USE OF FOSS

Sr. No.		Strongly Disagree	Disagree	Agree	Strongly Agree
FOSS - JFLAP					
1	The interface is easy to understand the steps involved in the construction of parsing table using LL(1) and SLR(1).	0%	1%	55%	44%
2	Various representation such as Noninverted tree, inverted tree and derivation table for parsing the string of given CFG also helped me to understand the working of the parser type.	1%	3%	39%	57%
3	Visual representation also made easy to understand the various parser type.	0%	3%	57%	40%
4	This tool is useful for understanding the step by step procedure involved in the construction of parsing table.	2%	4%	61%	33%
5	This tool is easy to understand.	0%	1%	60%	39%
FOSS - Parsing Emulator					
1	It helped me to clear the concept of parser technique by providing the solution to different grammar examples.	2%	1%	37%	60%
2	I solved almost all grammar examples and verified the result using this tool.	3%	0%	58%	39%
3	The interface is easy to understand the solution of the grammar.	0%	1%	68%	31%
4	It gives the solution to the grammar step by step.	4%	2%	49%	45%
5	Because of this tool, I understood all the parser types.	2%	3%	59%	46%
6	This tool is easy to understand.	0%	5%	58%	37%
7	Did you like this activity - use of FOSS during the laboratory session?	Yes =98% No =2%			

D. Research Design

This experiment was carried out for third year Computer Science and Engineering students. Two group post-test method was considered. Two groups were form randomly. Topic considered is – Construction of parsing table using LL(1) and SLR.

Firstly, the topic – LL(1) and SLR, is taught using traditional teaching method i.e. blackboard teaching to the experimental group as well as control group. Six sessions of each one hour were conducted for teaching the topic.

After blackboard teaching, the treatment for both groups was different. Two sessions of each two hours were given to both the group to solve the examples in the laboratory session.

- For experimental group, FOSS based learning was considered during the laboratory session. JFLAP tool is used to explain the step by step procedure to construct the parsing table for given CFG using

LL(1) and SLR technique. Parsing Emulator FOSS is given to the students to practice these parser types' examples. After that students implemented the LL(1) parser type in laboratory session for particular grammar.

- For control group, discussion was conducted related to the topic- parser type and examples were given to them to solve. After that students implemented the LL(1) parser type in laboratory session for particular grammar.

After the different treatment given to both the group, post-test was conducted on the topic to check the effectiveness of use of FOSS.

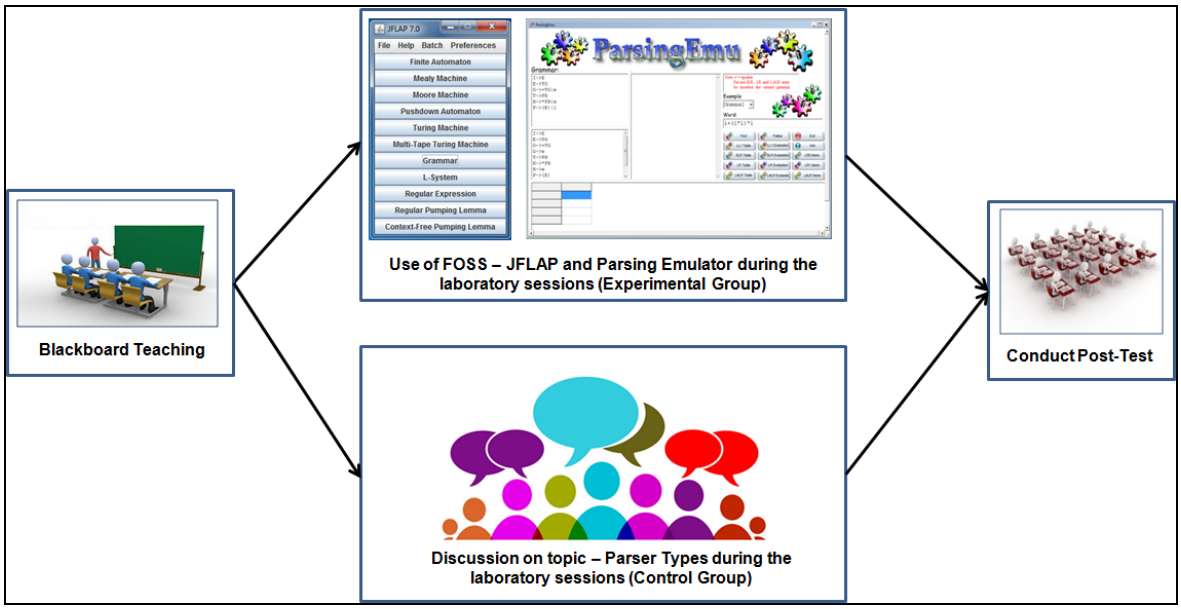

Fig. 14. Experimental Setup

E. Learning Objectives (LOs), Research Question (RQ) and Hypothesis of Study

Learning Objectives (LOs) of this study is to teach problem solving skill. These LOs are:

- To build the parsing table for given CFG using LL(1) and SLR parser (LO1) and
- To parse the given string using LL(1) and SLR parser (LO2).

The research questions for this study are

- Whether the use of FOSS improves the problem solving ability of the students? and
- Whether gender bias is there in education?

Hypothesis of our study are

H1: Students' post test scores of experimental group for LO1 are higher than students' post test scores of control group irrespective of gender.

H2: Students' post test scores of experimental group for LO2 are higher than students' post test scores of control group irrespective of gender.

V. Result Analysis

Students' problem solving ability was analyzed using post test marks. Result analysis is done in five ways

A. Experimental and control group - Mixed (Boys + Girls)
B. Experimental and control group - Only Girls
C. Experimental and control group - Only Boys
D. Experimental Group (Girls) and Control Group (Boys)
E. Experimental Group (Boys) and Control Group (Girls)

A. Experimental and control group - Mixed (Boys + Girls)

Students' problem solving ability was analyzed using post test marks as shown in figure 15. The graph shows the significant improvement in the post-test performance of experimental group as compared to control group.

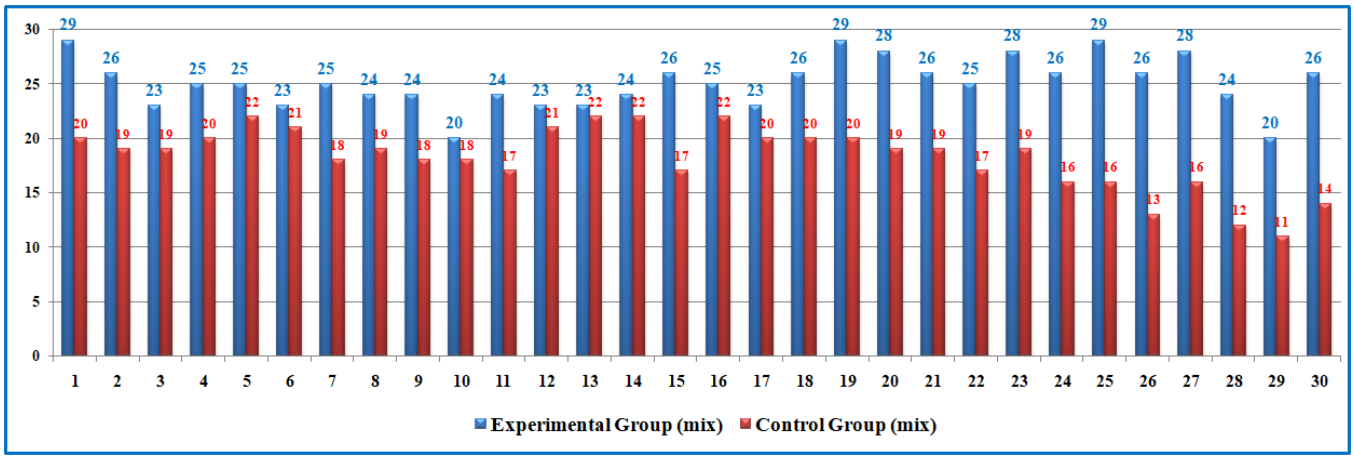

Fig. 15: Post-test marks comparison of control group and experimental group consisting of equal number of boys and girls

t-Test is used to test if two grops differed significantly from each other. If the p-value is less than 0.05 then the t-test is significant. t-Test result of Post-Test for experimental group and control group is shown table 3.

TABLE IV
T-TEST RESULT OF POST-TEST FOR FIGURE 15

t value	p value
10.08	< .00001

B. Experimental and control group - Only Girls

Figure 16 shows the graph for the post-test marks of control and experimental group consisting of only girls. The graph as well as t-Test in Table 5 shows that experimental group performs better than control group.

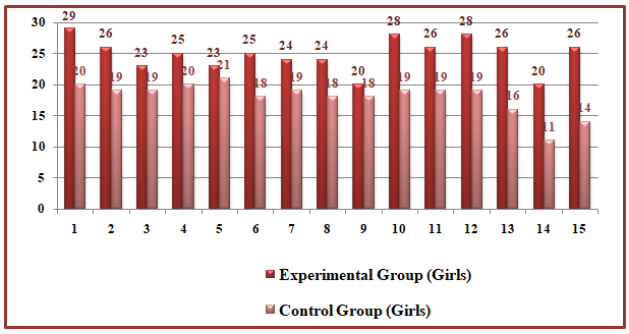

Fig. 16. Post-test marks comparison of control group and experimental group consisting of only girls

TABLE V
T-TEST RESULT OF POST-TEST FOR FIGURE 16

t value	p value
7.22	< .00001

C. Experimental and control group - Only Boys

Figure 17 shows the graph for the post-test marks of control and experimental group consisting of only boys. The graph as well as t-Test in Table 6 shows that experimental group performs better than control group.

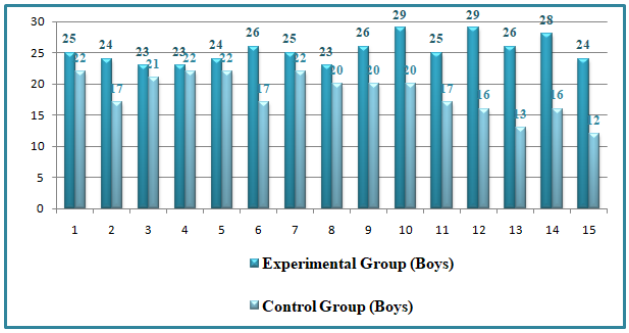

Fig. 17. Post-test marks comparison of control group and experimental group consisting of only boys

TABLE VI
T-TEST RESULT OF POST-TEST FOR FIGURE 17

t value	p value
6.85	< .00001

D. Experimental Group (Girls) and Control Group (Boys)

Figure 18 shows the graph for the post-test marks of control group consisting of only boys and experimental group consisting of only girls. The graph as well as t-Test in Table 7 shows that experimental group performs better than control group.

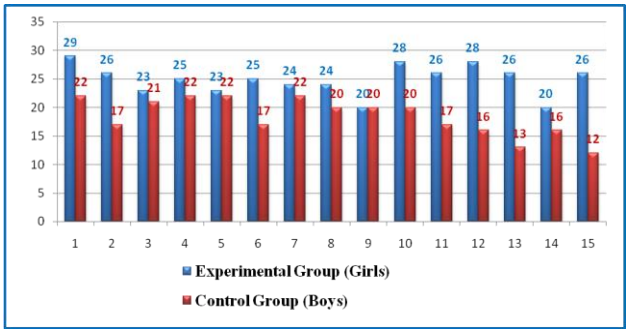

Fig. 18. Post-test marks comparison of control group consisting of only boys and experimental group consisting of only girls

TABLE VII
T-TEST RESULT OF POST-TEST FOR FIGURE 18

t value	p value
5.85	< 0.00001

E. Experimental Group (Boys) and Control Group (Girls)

Figure 19 shows the graph for the post-test marks of control group consisting of only girls and experimental group consisting of only boys. The graph as well as t-Test in Table 8 shows that experimental group performs better than control group.

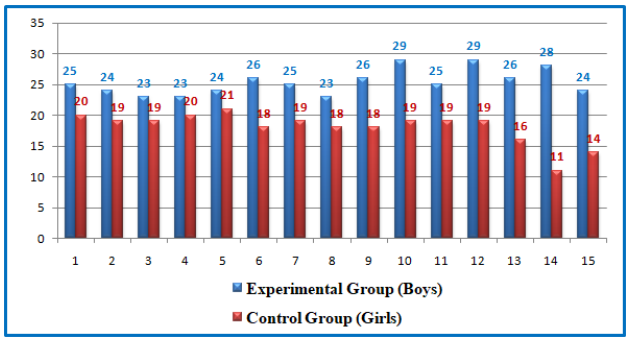

Fig. 19. Post-test marks comparison of control group consisting of only girls and experimental group consisting of only boys

TABLE VIII
T-TEST RESULT OF POST-TEST FOR FIGURE 19

t value	p value
6.10	< .00001

Table 9 shows overall analysis of the experimental and control group.

TABLE IX
EXPERIMENTAL AND CONTROL GROUP ANALYSIS

	Experimental Group	Control Group	t-Test
Mixed (Boys + Girls)	30	30	< 0.00001
Only Girls	15	15	< 0.00001
Only Boys	15	15	< 0.00001
Experimental Group (Girls) and Control Group (Boys)	15	15	< 0.00001
Experimental Group (Boys) and Control Group (Girls)	15	15	< 0.00001

VI. CONCLUSION

In this paper, the use of FOSS is considered for the course, Compiler Construction of Third Year Computer Science and Engineering. Two tools were considered – JFLAP and Parsing Emulator. JFLAP tool is considered for explaining the step-by-step procedure to construct the parsing table while Parsing Emulator tool is considered for practicing the examples. The result shows that the students' problem solving ability of experimental group is improved as compared to the control group. Also the performance of students is analyzed in five ways

- Experimental and control group - Mixed (Boys + Girls)
- Experimental and control group - Only Girls
- Experimental and control group - Only Boys
- Experimental Group (Girls) and Control Group (Boys)
- Experimental Group (Boys) and Control Group (Girls)

In all cases, students' problem solving ability of experimental group is improved as compared to the control group irrespective of gender. It means there is no gender bias in Education.

REFERENCES

Marjan Mernik (2003). An educational tool for teaching compiler construction. *IEEE Transactions on Education 46(1):*61 – 68, *DOI: 10.1109/TE.2002.808277*

T. R. Henry (2005). Teaching compiler construction using a domain specific language. *SIGCSE Bull 37(1),* 7-11.

E. White et.al. (2005). Hide and show: Using real compiler code for teaching. *SIGCSE Bull 37(1),* 12-16.

Michael Hielscher and Christian Wagenknecht (2006). AtoCC: learning environment for teaching theory of automata and formal languages, *ACM SIGCSE Bulletin June 2006* https://doi.org/10.1145/1140123.1140211.

M. Ruckert (2007). Teaching compiler construction and language design: Making the case for unusual compiler projects with postscript as the target language. *SIGCSE Bull 39(1),* 435-439.

A. Demaille, R. Levillain and B. Perrot (2008). A set of tools to teach compiler construction. *Proceedings of the 13th Annual Conference on Innovation and Technology in Computer Science Education ser. ITiCSE'08*, 68-72.

Keshav Pingali and Gianfranco Bilardi (2015). A Graphical Model for Context-Free Grammar Parsing. . *International Conference on Compiler Construction*, DOI: 10.1007/978-3-662-46663-6_1.

Divya Kundra and Ashish Sureka (2016). Teaching Compiler Design Concepts Using Case-Based and Project-Based Learning Approaches. *IEEE Eighth International Conference on Technology for Education (T4E)*, DOI: 10.1109/T4E.2016.052

Sunita M. Dol and Dr. S. A. Halkude (2016). An Active Learning Strategy Think-PairFree Open Source Software-Share to Teach Engineering Courses. *Journal of Engineering Education Transformation*, ISSN 2349-2473 (Print), eISSN 2394-1707 (Online).