Francisco J. Almeida-Martínez
Jaime Urquiza-Fuentes

Building LL (1) grammars and parsers using VAST and ANTLR.

Number 2011-10

Serie de Informes Técnicos DLSI1-URJC
ISSN 1988-8074
Departamento de Lenguajes y Sistemas Informáticos I
Universidad Rey Juan Carlos

Index

Building LL (1) grammars and parsers using VAST and ANTLR

Francisco J. Almeida-Martínez and Jaime Urquiza-Fuentes

Universidad Rey Juan Carlos
Departamento de Lenguajes y Sistemas Informáticos I
c/ Tulipn s/n, 28933 Madrid, Spain
francisco.almeida@urjc.es, jaime.urquiza@urjc.es

Abstract. This evaluation tests the educational effectiveness after using VAST to design LL grammars and parsers with a punctual use. The results shows that with just one session there are not significant differences in using or not VAST. However, we can see how the students get used easily to the tool and likes it.

1 Introduction

Once we observed some educational positive effects after using VAST for LR parser design in a long term evaluation, we planned to re-evaluate the tool for LL grammars and parser design. The objective of the current evaluation was to observe if just using the tool in one session, students could improve their grammar design.

The version of the tool used in this evaluation is similar to previous evaluations. We have changed the icons of the tool bar, improved the user's help and added reserved words' detection in the grammar editor.

2 Description of the evaluation

In this section we describe the evaluation. We refer to the participants, the experiment's design, the tasks performed during the session and the protocol.

2.1 Subjects

In this evaluation participated **26 students** of the *Language Processor* subject of the Rey Juan Carlos Universitiy during the 2010-2011 course. The participation was voluntary and based incentive in a 1.25% over the final mark only if they passed the exam.

2.2 Experimental design

This evaluation was designed as an educational effectiveness plus an usability and observational study. Students were divided in two groups: control and treatment one. The control group used ANTLR parser generator tool and the treatment one VAST. In order to create the groups we used marks of a pretest of knowledge. Students were divided using these marks, but the assigment to the control or treatment group was random. The pretest included questions which referred to differents aspects of the Bloom's taxonomy [1]. In table 1 we show the results of the pretest's results. As we can see both groups are balanced, so there does not exist any significant difference in any of the Bloom's taxonomy levels. The independent variable was the tool used by each group; VAST for treatment one and ANTLR for control. The dependent variables were the educational effectiveness in terms of learning differences between a pretest and postest and the students' opinion about four usability aspects: ease of use, learning support, quality of the tool and satisfaction. This evaluation lasted 2 hours (one session).

Level	**Control**	**Treatment**	**Stats.**
Knowledge (K)	0.01	0.00	U=72.000, p=0.78
Understanding (U)	0.00	0.00	U=71.500, p=0.73
Application (Ap)	0.20	0.02	U=71.500, p=0.73
Synthesis (S)	0.07	0.00	U=72.000, p=0.77
Total	0.09	0.01	U=66.000, p=0.54

Table 1. Pretest of knowledge's results according to the Bloom's taxonomy

2.3 Tasks

The tasks performed by students had to be docummented at the end of the evaluation (see appendix 5) with both text explanations and visualizations using VAST in treatment group and any other software for control one. The tasks consisted in two exercises about the grammar design for LL (1) parsers. One week before the experiment we explained how to use ANTLR parser generator in class. Besides, at the beginning of the evaluation we explained to each group the tasks and how to work with the tools.

Treatment group. This group had to resolve the exercises using VAST's grammar editor and get the appropiate grammar to recognize all the inputs streams given the features of the grammar. The solutions for each exercise consisted in the ANTLR parser's specification and different input streams (correct and incorrect) together with visualizations and text explanations.

Control group. This group had to work just with the ANTLR parser generator. As in the treatment group, they were given the features of the grammar and they had to build a parser to accept the corresponding input streams.

In this case, the solutions to the exercises consisted in the ANTLR parser's specification and different input streams (correct and incorrect) together with visualizations and the corresponding text explanations.

2.4 Protocol

Three weeks before the evaluation we performed the pretest. One week before we dedicated one session to explain the generation tool ANTLR. In the evaluation we used about 15 minutes to introduced the tool VAST in the treatment group. Before the end of the session we asked the students for answering an usability-quality questionary about the tools. Two days after the evaluation the students had to send the solutions by email. The same day we performed the postest during a theory class. In table 2 we show a summary of the protocol followed in this evaluation.

Control	**Treatment**
Pretest of knowledge	
ANTLR session	
ANTLR session	VAST session
ANTLR test	VAST test
Postest of knowledge	

Table 2. Protocol used in the evaluation

3 Results

In this section we describe the results of the evaluation. During the experiment the instructor observed how the students worked with the tool. The results are divided in three categories: instructor's observations, answers' to questionnaires and educational effectiveness results.

3.1 Instructor's observations

During the evaluation we observed in the treatment group that all students used VAST's grammar editor. Also we saw that some students had problems in their configuration with the MAC OS due to a certain execution permissions. After 30 minutes all students finished the exercise number 1. Before the end of the session 5 students had finished the exercises.

In the control group we observed that students had problems to configurate propertly the CLASSPATH variable and to compile their own specifications. After one hour of the session, none of the students had finished the exercise number 1. At the end of the session, only some students had finished the exercise number 1; none of the them had finished the exercise 2.

3.2 Answers to questionnaires

The VAST and ANTLR's opinion questionaries (see appendix 5) were designed to obtain as much information of the tool as possible. Students had to answer using a Likert scale with 5 values being 1 the lowest mark and 5 the highest one. Besides we included open questions. We performed an analysis of the avarage marks obtained in the questionaries, which allow to get information about the ease of use, quality (general and for each part), the learning support and satisfaction with the tool used. In table 3 we show the marks obtained for the general and specific aspects for both groups.

Aspect	**Control**	**Treatment**	**Stats.**
Ease of use			
General ease of use	3.36	3.17	t(24)=0.46, p=0.65
Parts' average	3.17	3.31	t(24)=-0.44, p=0.67
Learning support			
Syntax tree building	2.86	3.42	t(24)=-0.14, p=0.19
Stack view	2.57	3.25	t(24)=-0.05, p=0.17
Input stream processing	3.14	3.25	t(24)=-0.27, p=0.79
Technic quality			
General quality	3.00	3.58	t(24)=-0.14, p=0.16
Parts' average	2.93	3.38	t(24)=-0.14, p=0.16
Student satisfaction			
General satisfaction	2.71	3.42	t(24)=-016, p=0.12

Table 3. Usability-Quality opinions results

Apart from giving their mark for different aspects of the tools, the questionaries asked the students for their opinion about the difficult aspects, the parts with best/worst quality, things to include in the tools and finally the positive and negative aspects. In the control group students pointed out the execution from the command line, the first execution, the installation process and the difficulties to debug the grammar as difficult aspects. According to the best quality, in the control group students indicated that the tool was very simple. In treatment group they mentioned the input stream processing process, the syntax tree's animation, the ease of use, the possibility of trying and simulate the behaviour of a parser and the stack. As worst aspects, in control group students referred to the graphical user interface, the error messages of ANTLR and the lack of the parser's stack. In the treatment group they pointed out that it would be neccesary to explain the tool with more details. When asking about aspects to add to the tools, in control they mentioned the lack of a grammar editor and graphic interface. For positive aspects, in treatment group students pointed out the ease of use, "the tool is useful to study the subjec" and the grammar editor. As negative aspects, in control group they mentioned that it is very difficult to find the errors in the input streams.

3.3 Results of educational effectiveness

The results of educational effectiveness are divided in two parts. On the one hand, the differences between the postest-pretest. On the other hand, the marks of the evaluation's exercises. In table 4 we show the differences according to the Bloom's taxonomy [1] between postest-pretest. As we can see there does not exist significant differences in any of the Bloom's taxonomy levels.

Level	**Control**	**Treatment**	**Stats.**
Knowledge (K)	0.10	0.02	$X^2(2)$=2.39, p=0.30
Understanding (U)	0.40	0.14	$X^2(2)$=1.61, p=0.45
Application (Ap)	0.60	0.31	$X^2(2)$=1.50, p=0.47
Synthesis (S)	0.16	0.12	$X^2(2)$=1.17, p=0.56
Total	0.30	0.15	$X^2(2)$=1.45, p=0.49

Table 4. Educational effectiveness resutls according to the Bloom's taxonomy levels

According to the solutions to the practices, the average mark of the control group was 0.41 while in the treatment group was 0.58 (t(25)=-1.27, p>0.05). The analysis of data does not reveal any statistics significant differences in the solution to the exercises. However, the number of students which finished the practice was larger in the treatment group (8 in control and 11 in treatment).

4 Conclusions

The results obtained can be divided in two parts: usability-quality results and educational effectiveness results. According to usability and quality we can see that VAST is as easy to use as ANTLR taking into account the students opinion. Refering to quality, students also think that it is similar for both tools.

The educatinal effectiveness has been measured taking into account two aspects: the differences between postest-pretest and the practice results. The differences between the tests do not show any significant difference between any group. The same occurs in the solutions to the exercises. However, we have found differences in the number of deliveries of the exercises, so although the control group (ANTLR) was more numerous, the number of students which delivered the exercise was larger in treatment group. The percentage of the practices delivered in the control group was 77.14% (8/14) while in treatment group was 92.67% (11/12).

From the instructors' point of view, the results of the evaluation were satisfactory, in treatment group students did not have any problem in the use of the tool. However in the control group they had many difficulties to compile propertly the parsers built by ANTLR. Besides, the students who worked with VAST were able to finished the exercises faster during the session.

Given the results of this evaluation, we plan another one to design grammars for LR (1) parsers following a similar protocol.

5 Acknowledgment

This project is supported by project TIN2008- 04103/TSI of the Spanish Ministry of Science and Innovation.

A Exercise used for the pre-postest

1. What are the main features of the top-down syntax analysis?
2. What is the fundament of the top-down-predective-recursive syntax analysis?
3. What is the main difference between the top-down predictive-recursive parser and not recursive?
4. Which are the top-down parser conditions?
5. Given the following table of a LL (1) parser:

	id	+	*	(	)	**$**
E	E::=TE’			E::=TE’		
E’		E’::=+TE’			E’::=	E’::=
T	T::=FT’			T::=FT’		
T’		T::=	T’::=*FT’		T’::=	T’::=
F	F::=id			T::=(E)		

   Given the input stream *id+id*id$*, simulate its processing, indicating in each step the state of the stack and remarking the part processed.
6. Given the grammar:

```
E::=E+E
E::=-E
E::=i
```

   Is it possible to implement a LL (1) parser? Why?
7. Design a LL (1) grammar to recognize input streams with declarations of **for-endfor** blocks. Each **for** sentence has a declaration of a counter, a condition in **parenthesis** and the step. This step can be modified using **increment/decretment or any maths expression**. Each one of the parts has to be separated using **;**. Each **for** sentence allows to declare sentences in its body and they can be nested. A valid example could be:

```
for (int i=0; i<=(i+3); i++)
   var1=var2+var3;
   for (var1; var1<=30; var*2)
      var4=var1-2
   endfor
endfor
```

B Exercises used during the evaluation

1. **Problem 1.**We want to know if a input stream has a certain structure. In particular the stream has to follow this scheme:

   ```
   STREET_TYPE NAME, NUMBER, STAIR, DOOR, ZIP
   ```

   Where:
   - STREET_TYPE: can take the folling values: Street, Avenue or Passage.
   - NAME: it is a text stream. Blanks are not allowed.
   - NUMBER: integer possitive number.
   - STAIR: can take values 0 (for right) and 1 (for left).
   - DOOR: values from A to D.
   - ZIP: number between 28000-28999.

   For example:
   Street Example, 20, 0, A, 28909
2. **Problem 2.** We want to build a parser to recognize a mini-assambler language. An example of the input streams could be:

   ```
   etiq1: ADD op1, op2, res1#
   ADDI res1, val#
   JUMP etiq1#
   SUB op1, op2#
   BEQ op1, op2, etiq1#
   ```

   The language admits adds, immediate adds, inconditional jumps, subs and conditional jumps. The reserved words are ADD, ADDI, JUMP, SUB and BEQ. Sentences have to finished with the # character. We allow the use of labels, operands and variables to save the results. Requirements:
   - OPERANDS: they are the word *op* follows by number from 1-9
   - LABELS: word *etiq* follows by a positive integer number. They must be followed by *:*.
   - RESULTS VARIABLES: word *res* followed by numbers from 0-9.

For each exercise:

Document the resut with visualizations (1 or more). To do this you have to create a Word document with one or more examples with the following elements: input stream, resulting tree and the grammar used for that tree.

References

1. B. Bloom, E. Furst, W. Hill, and D.R. Krathwohl. *Taxonomy of Educational Objetives: Handbook I, The Cognitive Domain.* Addison-Wesley, 1959.