BURGRAM: Una herramienta interactiva para el estudio de los algoritmos de análisis sintáctico ascendente y descendente

César García Osorio, Carlos Gómez Palacios,
Jesús Maudes Raedo, Juan José Rodríguez Diez
Universidad de Burgos
Avda. Cantabría s/n, 09006, Burgos
cgosorio@ubu.es, carlosgomezpalacios@gmail.com, jmaudes@ubu.es, jjrodriguez@ubu.es

Resumen

"Cinco minutos interaccionando con la ejecución de un algoritmo vale más que media hora de clase magistral", esta podría ser la versión del conocidísimo "una imagen vale más que mil palabras" que ha inspirado el desarrollo del presente recurso docente. El contexto es el de la docencia en Procesadores de Lenguajes, en concreto la enseñanza de los algoritmos de análisis sintáctico ascendente y descendente. BURGRAM es una herramienta que permite la ejecución interactiva de este tipo de algoritmos.

Aunque existen otras herramientas, BURGRAM cubre sus deficiencias poniendo el énfasis en la interactividad y el diseño funcional de la interfaz gráfica. Es posible ejecutar los algoritmos paso a paso o deshacer el efecto de los pasos aplicados. Se visualiza simultáneamente la tabla de análisis, los contenidos de la pila y de la entrada, así como el árbol de análisis sintáctico. El docente o el estudiante pueden elegir los elementos en los que quieren centrase haciendo que éstos ocupen la totalidad de la ventana de la aplicación. También es posible generar un informe en formato PDF, RTF o HTML de la traza del algoritmo que el alumno puede utilizar para referencias posteriores, añadiéndolo a las notas tomadas en clase o para resolver dudas en tutorías.

1. Introducción

Uno de los principales problemas a la hora de ilustrar los algoritmos de análisis sintáctico es que los ejemplo utilizados o son muy simples o largos y tediosos de hacer a mano. El alumno tiende a estar más preocupado de copiar lo que el profesor escribe en la pizarra que de entender lo que esta haciendo y relacionarlo con los pasos del algoritmo explicado previamente. Además, no es infrecuente que el alumno cometa algún error al copiar el ejercicio, y que este en vez de suponer una ayuda para la comprensión del algoritmo, se convierta en un obstáculo adicional. Por otra parte, los alumnos están acostumbrados a usar el ordenador en otras asignaturas. Aquí presentamos BURGRAM, una herramienta de apoyo a la docencia de los métodos de análisis sintáctico que se suma a la, por fortuna, cada vez más numerosa herramientas de apoyo a la docencia de este tipo de asignaturas [4,9,10,3,5].

La siguiente sección presenta brevemente los conceptos teóricos sobre análisis sintáctico. La sección 3 presenta herramientas relacionadas. La aplicación desarrollada se presenta en la sección 4. Finalmente, se presentan las conclusiones y líneas de trabajo futuro.

2. Análisis Sintáctico

Un compilador traduce un lenguaje de programación de alto nivel a un lenguaje máquina, mientras que un interprete ejecuta directamente las instrucciones del lenguaje de programación.

En compilación se suelen considerar dos fases, el análisis y la síntesis. En la primera etapa se analiza el programa fuente mientras que en la segunda se genera el código para la máquina objeto. Dentro del análisis hay tres fases, el análisis lexicográfico, el sintáctico y el semántico. El primero descompone el programa fuente en sus elementos constituyentes (*tokens*). El segundo examina la estructura del programa y comprueba que respeta la sintaxis del lenguaje de programación. El semántico verifica la corrección de las construcciones sintácticas conforme a la semántica del lenguaje y recopila información necesaria para la síntesis. Este trabajo se enmarca dentro de la fase de análisis sintáctico.

El objetivo del análisis sintáctico es obtener a partir de una secuencia de tokens la estructura de los mismos, expresada mediante un árbol de análisis sintáctico.

Para poder realizar el análisis de un programa en un determinado lenguaje se necesita una descripción precisa de éste. Normalmente se definen utilizando gramáticas libres de contexto. Están formadas por un conjunto de símbolos terminales, un conjunto de no terminales, un símbolo inicial y un conjunto de producciones que relacionan los terminales con los no terminales.

Para realizar el análisis es necesario calcular 3 tipos de conjuntos previamente: anulables, iniciales (*first)* y seguidores (*follow)*. El primero se corresponde a los no terminales que pueden generar la palabra vacía. El conjunto *first* de una cadena es el conjunto de terminales que pueden comenzar cualquier cadena obtenida a partir de la de partida. El conjunto *follow* de un no terminal es el conjunto de terminales que pueden aparecer inmediatamente después del no terminal.

Hay dos tipos de métodos de análisis sintáctico, los descendentes y los ascendentes. En el primer caso el árbol se construye desde la raíz hasta las hojas, mientras que en el segundo se construye desde las hojas a la raíz.

El método más conocido de análisis descendente es el denominado LL(1), que recibe ese nombre ya que se basa en gramáticas LL(1). Este tipo de gramáticas se caracterizan por no ser ni ambiguas ni recursivas a la izquierda. Un analizador de este tipo esta formado por el buffer de entrada, una pila, una tabla de análisis y el programa de control del analizador. La tabla indica, dados el símbolo de la entrada y el de la cima de la pila, la producción que se debe expandir.

En el análisis ascendente se parte de la cadena a analizar y se intenta llegar al símbolo inicial de la gramática. El más utilizado es el denominado LR. En este caso el analizador también dispone de un buffer de entrada, una pila y una tabla de análisis. En este caso la tabla tiene dos partes, la de "acción" y la de "ir a".

Este tipo de análisis se pude realizar con distintos tipos de tablas.: SLR(1), LR(1) y LALR(1). Las primeras son las más fáciles de generar, pero sólo son capaces de reconocer un número reducido de gramáticas. Las segundas permiten reconocer un mayor número de gramáticas. Las terceras se encuentran, desde el punto de vista de cuántas gramáticas son capaces de reconocer, entre las primeras y las terceras, pero tienen la ventaja de que las tablas son de tamaño más reducido que las de LR(1).

Para construir dichas tablas es necesario realizar ciertos cálculos: el del cierre o clausura, el de sucesores y el conjunto de estados.

Al realizar análisis ascendente pueden aparecer conflictos, siendo más frecuentes en el SLR. Un conflicto ocurre cuando en una entrada de la tabla hay más de una acción. Hay dos tipos de conflictos: desplazamiento/reducción y reducción/reducción.

Esta sección ha dado una descripción necesariamente somera de conceptos, los libros de Kelley [6], Brookshear [2] y Alfonseca [1] son excelentes para obtener descripciones más amplias.

3. Otras herramientas

En esta sección presentamos otras herramientas que pueden ser usadas para facilitar el aprendizaje de los métodos de análisis ascendente.

- *Anagra*[1] [7]: Esta aplicación está orientada a la simulación y análisis de gramáticas (independientes del contexto y regulares). Es capaz de realizar tanto el análisis ascendente como el descendente. También permite transformar gramáticas de un tipo a otro según la jerarquía establecida por Chomsky. Anagra es una herramienta muy completa, implementa gran cantidad de transformaciones que podemos aplicar sobre nuestras gramáticas. Sin embargo, desde nuestro punto de vista, esta limitada por la escasez de información, ya que da por hecho que el usuario conoce toda la información y no muestra los pasos intermedios de las transformaciones realizadas. Puede resultar muy útil para comprobar si un ejercicio se ha resuelto correctamente, pero no para aprender su funcionamiento.
- *Java formal language and automata processor* (*JFLAP*)[2]: Se trata de una herramienta muy completa y bien documentada [8]. La cualidad más destacable de esta aplicación es la gran cantidad de

[1] http : // webdiis . Unizar . Es / ~ezpeleta / COMPI / compiladoresI . htm
[2] http://www.jflap.org/

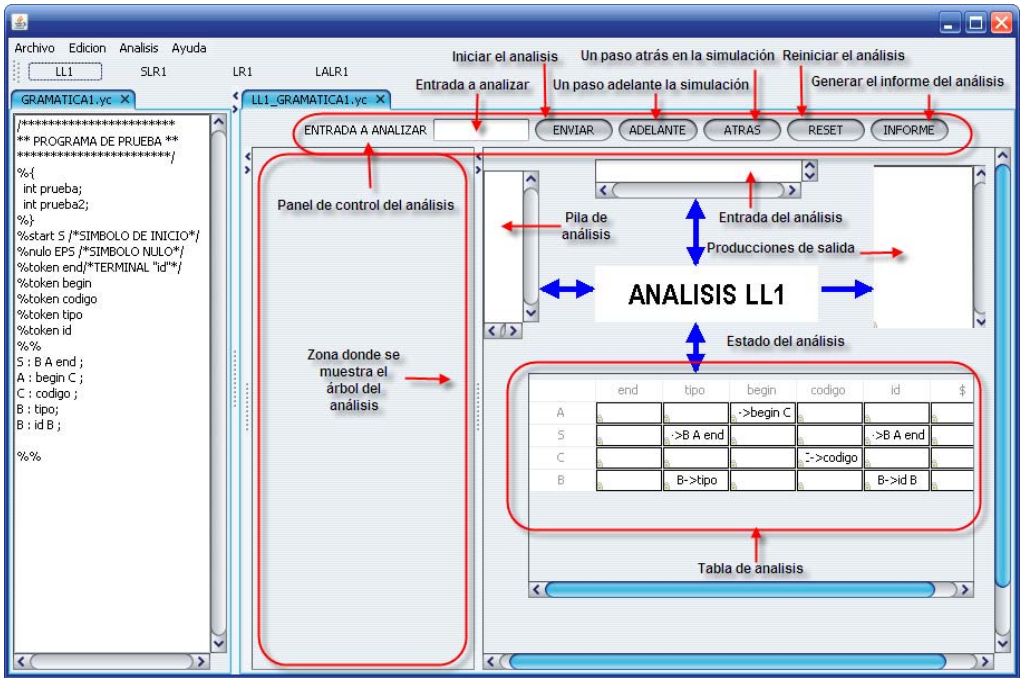

Figura 1. Ejemplo: análisis LL1.

operaciones que se pueden llevar a cabo. Aunque la mayoría de ellas están más orientadas a una asignatura de Autómatas y Lenguajes Formales previa a Procesadores de Lenguajes. Dentro del análisis ascendente sólo considera el método SLR(1) para la generación de tablas.

- Sefalas[3]: Se trata de una herramienta con el mismo objetivo que la presentada en este artículo, la docencia de los algoritmos de análisis. Implementa algoritmos de análisis, como el de precedencia de operadores, que no están presentes en BURGRAM. Pero en cambio la simulación del análisis no es tan conveniente. En BURGRAM se muestran al mismo tiempo todos los elemento relevantes, como las tablas, el árbol de análisis, el contenido de la pila, la entrada por leer y las acciones de análisis; en cambio en Sefalas sólo se muestran simultáneamente los tres últimos. Otra característica de BURGRAM, no presente en Sefalas, es la visualización del autómata para los métodos ascendentes.

Una de las diferencias principales con respecto a la aplicación presentada es la capacidad de generar informes. Por otro lado, el uso de Grappa [4] permite la obtención de autómatas con una visualización más atractiva.

4. Descripción de la aplicación

La aplicación está desarrollada en Java, por lo que es multiplaforma. No obstante, se dispone de un programa de instalación para Windows. La aplicación (figura 1), se organiza, en un primer nivel, en dos zonas. En la izquierda se encuentra la zona de texto, que es donde se introducen las gramáticas. En la parte derecha, se encuentra la zona de análisis, que será donde el usuario verá el

[3] http://lsi.ugr.es/%7Epl/software.php

[4] http://www.research.att.com/~john/Grappa/

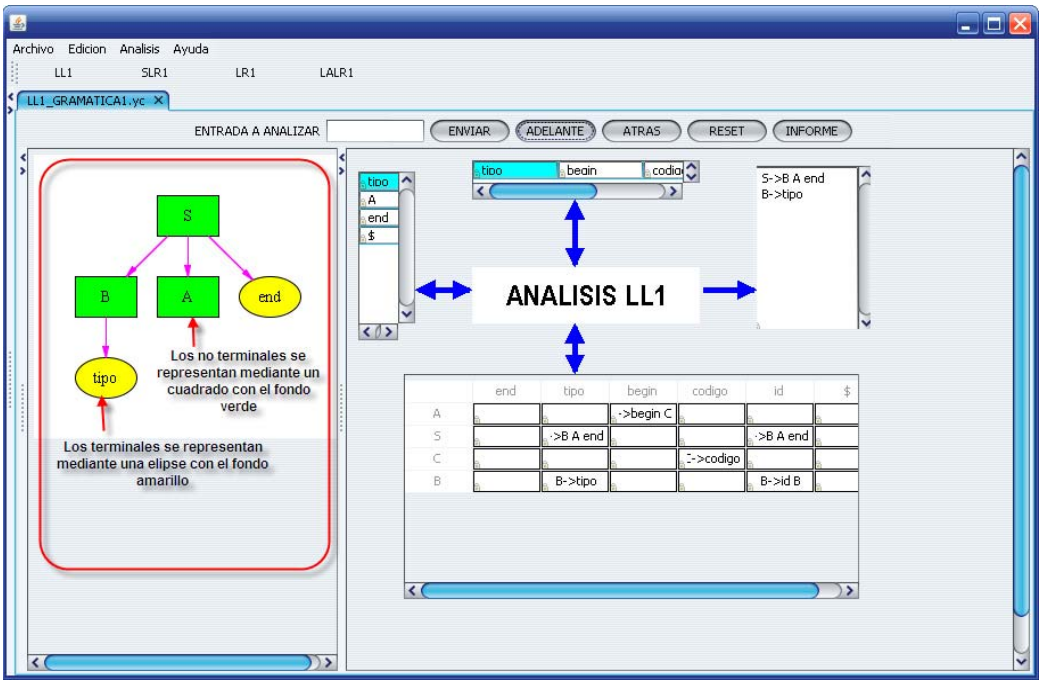

Figura 2. Pestaña con el panel de análisis.

análisis sintáctico realizado, y donde podrá interactuar con él realizando simulaciones.

En ambas zonas se dispone de pestañas. Es decir, se pueden tener simultáneamente varias gramáticas abiertas y para cada una de ellas se pueden tener asociados distintos tipos de análisis.

En el menú de archivo se dispone de las típicas opciones: nuevo, abrir, guardar, guardar como, cerrar y salir. El menú de edición incluye opciones útiles para la edición, como texto, de las gramáticas: cortar, copiar, pegar y seleccionar todo.

En el menú de análisis aparecen los análisis que se pueden realizar: LL1, SLR1, LR1, LALR1. Por otro lado, el usuario también puede realizar estos análisis utilizando un conjunto de botones situados debajo del menú.

Al seleccionar cualquiera de estas opciones, se revisa la gramática, informando al usuario de los posibles errores.

Una vez que se ha realizado el análisis, se pueden realizar simulaciones del mismo. La zona de la aplicación dedicada a la simulación del análisis está formada por varias componentes. En la parte superior se encuentra el panel de control. Dicho panel está formada por un entrada de texto, donde el usuario puede indicar la cadena a analizar, y de varios botones: el envío de la entrada para comenzar la simulación, realizar un paso hacia delante en la simulación, retroceder un paso hacia atrás, resetear la simulación y generar el informe.

En la parte izquierda de la zona de análisis se encuentra el árbol de análisis (figura 2). Los componentes de dicho árbol se van generando a medida que se realizan los pasos de la simulación. Se utilizan distintos tipos de nodos para indicar si los nodos son o no terminales.

En la parte derecha aparece el estado de la simulación. La información que aquí aparece es la tabla de análisis sintáctico generada, el estado de la pila, el estado de la entrada y las producciones de salida emitidas en el análisis. Los posibles estados del análisis (en curso, error en la entrada, finalización correcta) se indican con un código de colores.

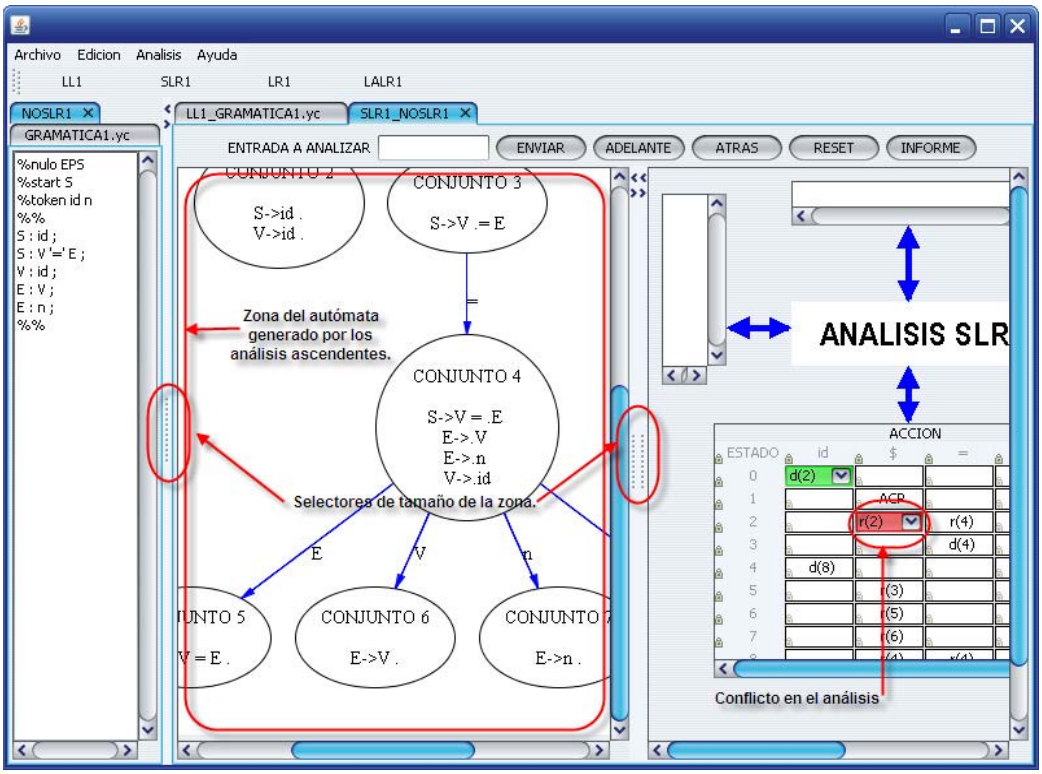

Figura 3. Análisis SLR1.

En el caso de los análisis ascendentes (i.e., LR1, SLR1 o LALR1) la aplicación muestra el autómata generado por dicho análisis. En la figura 3 se muestra la aplicación con un autómata generado para un análisis SLR1.

Una vez realizado un análisis se puede generar un informe del mismo. Los formatos actualmente soportados son html, rtf y pdf. El contenido del informe es el siguiente: la gramática, los conjuntos de terminales y no terminales, el símbolo de inicio, las producciones de la gramática y los conjuntos *FIRST* y *FOLLOW*. Si se ha introducido alguna entrada para analizar, se incluirá también el resultado del análisis y el árbol de análisis generado. Además, dependiendo del tipo de análisis, los informes incluyen otros elementos. Para los análisis descendentes se incluye la tabla de análisis sintáctico descendente. Para los ascendentes, la gramática ampliada, la tabla de análisis sintáctico ascendente y el autómata generado por el análisis.

Las figuras 4 y 5 muestran el autómata y el árbol de análisis generados como parte de un informe.

5. Conclusiones y trabajos futuros

Hemos presentado una herramienta que pretende facilitar al alumno la interacción con algoritmos que siempre es más visual y menos pesada que las trazas con lápiz y papel, permitiendo la experimentación y observando la evolución paso a paso de los algoritmos. Esto permite por un lado que el profesor pueda realizar ejercicios en clase, apoyado por un cañón de proyección, de un modo más rápido y ameno que usando la pizarra. Por otro lado, los alumnos pueden realizar tantos ejercicios como sean necesarios, de modo que puedan estimar sus progresos.

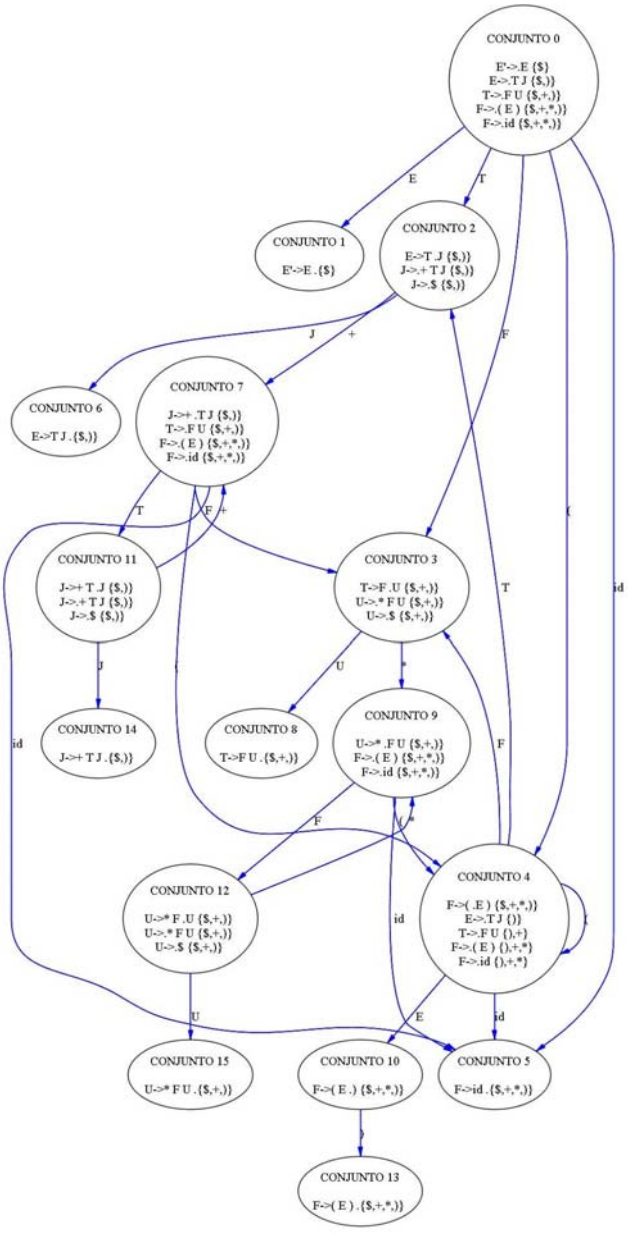

Figura 4. Autómata de análisis LALR1.

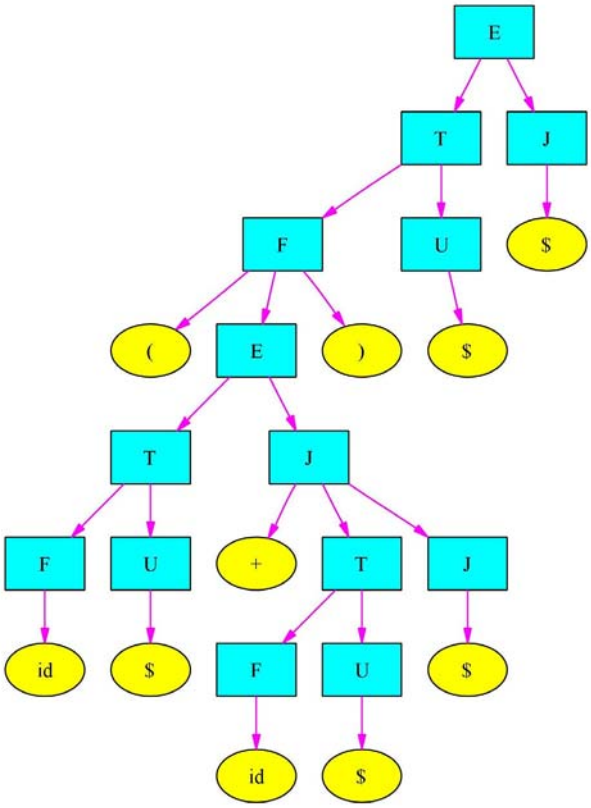

Figura 5. Árbol de análisis.

Una de las características diferenciales de esta aplicación frente a otras similares es la generación de informes. Por un lado se evita el tiempo necesario para copiar los ejercicios realizados por el profesor en clase, por otro los alumnos pueden llevar los informes generados al profesor para la aclaración de posibles dudas.

Otro tipo de análisis que se estudia en la asignatura de Procesadores de Lenguajes es el análisis ascendente por precedencia. Se está trabajando en la incorporación de este tipo de análisis en la herramienta.

Lamentablemente, todavía no se dispone de una evaluación formal por parte de los alumnos, la asignatura de "Procesadores de Lenguajes" en Burgos es una asignatura del primer cuatrimestre y la versión estable de la aplicación no estuvo disponible a tiempo como para poder haber sido utilizada intensivamente. Este es otro trabajo pendiente, junto con el de las mejoras previamente comentadas.

La versión comentada en este artículo se encuentra disponible para su descarga en: http://pisuerga.inf.ubu.es/cgosorio/BURGRAM/ y cualquier comentario sobre la misma será bienvenido.

6. Referencias

[1] Alfonseca, M., Alfonseca, E., Roberto M. *Teoría de Autómatas y Lenguajes Formales*, McGraw Hill, 2007.

[2] Brookshear, J.G. *Theory of Computation: Formal Languages, Automata, and Complexity.* Benjamin-Cummings Publishing Company, 1989.

[3] Castro-Schez, J.J., Castillo, E., Hortolano. J. Una herramienta para la enseñanza y aprendizaje de lenguajes formales y teoría de autómatas. *XIII Jornadas de Enseñanza Universitaria de la Informática*, 379-386, Teruel, 2007.

[4] Gallego Carrillo, M., Gortázar Bellas, F., Urquiza Fuentes, J., Velázquez Iturbide, J.A. SOTA, una herramienta educativa para la enseñanza de la tabla de símbolos. *XI Jornadas de Enseñanza Universitaria de la Informática*, 339-346, Madrid, 2005.

[5] García Osorio, C., Arnáiz González, A., Arnáiz Moreno, A. THOTH: A new tool for automata theory learning. *International Technology, Education and Development Conference (INTED2007)*, Valencia, 2007.

[6] Kelley, D. *Automata and Formal Languages: An Introduction.* Prentice Hall, 1998.

[7] Novoa Mínguez, R., Ezpeleta Mateo, J. Anagra 2.0. Un entorno para el estudio de las fases de análisis en el desarrollo de traductores. http://webdiis.unizar.es/~ezpeleta/ANAGRA. 2003

[8] Rodger, S.H., Finley, T.W. *JFLAP: An Interactive Formal Languages and Automata Package.* Jones & Bartlett Publishers, 2006.

[9] Santos, P.A., Castro-Schez, J.J. Una herramienta para la enseñanza y aprendizaje de la asignatura Procesadores de Lenguajes. *XII Jornadas de Enseñanza Universitaria de la Informática*, 499-506, Bilbao, 2006.

[10] Troyano, J.A., Galán, F.J., Carrillo, V., Enríquez, F., García, E.J. Prácticas de la asignatura Procesadores de Lenguaje con la herramienta ANTLR. *XI Jornadas de Enseñanza Universitaria de la Informática*, 323-330, Madrid, 2005.