Logsim. Sistema de apoyo al aprendizaje de la programación lógica.

Mateo Lezcano Brito. *
Giraldo Valdés Pardo. **
* Facultad de Matemática, Física y Computación de la Universidad Central de Las Villas (UCLV), Carretera a Camajuaní Km. 5½ Santa Clara, 54830 .Cuba.
** Profesor de la Facultad de Ingeniería Eléctrica de la UCLV.

Resumen.

En el presente trabajo se describe el sistema LogSim de apoyo al aprendizaje de la programación lógica, el cual se viene utilizando en la carrera de Licenciatura en Ciencias de la Computación en la Universidad Central de Las Villas (UCLV, Cuba). Mediante la simulación del comportamiento del lenguaje Prolog se propicia que los alumnos lleguen a dominar sus mecanismos más importantes. Están disponible un conjunto de programas ilustrativos que, al mostrar su trayectoria de ejecución, ayudan a entender conceptos tales como los cortes, el "backtracking", el uso de las listas, etc. Adicionalmente este sistema tiende a desarrollar en los estudiantes un correcto estilo de programación lógica.

1. Introducción.

Cuando un docente emprende la tarea de enseñar un lenguaje de programación, debe ponerse al tanto sobre los conocimientos precedentes que poseen los estudiantes para poder hacer un enfoque adecuado del curso. Existen dos situaciones en general: el estudiante se enfrenta por primera vez a este aprendizaje o ya conoce algún otro lenguaje.

Si lo que se requiere enseñar es un nuevo lenguaje, asociado a un paradigma de programación ya conocido, la tarea será relativamente fácil. Sin embargo, si lo que se enseña es un nuevo paradigma de programación, puede que el conocimiento previo sea un factor adverso en el proceso de aprendizaje, debido a que los estudiantes tienden a pensar en la forma que lo han hecho hasta ese momento, lo cual provoca que, en ocasiones, realicen tareas que, si bien están codificadas con las sentencias del "nuevo lenguaje", responden al antiguo paradigma, el cual condiciona fuertemente su manera de pensar.

En cualquier caso, si el profesor dispone de una herramienta auxiliar apropiada, se puede lograr un aprendizaje productivo, sobre todo si el estudiante se involucra activamente en el proceso.

La asignatura Programación Lógica se imparte en el tercer año de la Licenciatura en Ciencias de la Computación en tres universidades de Cuba. Cuando el estudiante alcanza esta etapa ya debe saber programar en lenguaje Ensamblador, Pascal (Estructurado y Orientado a Objetos) y conocer otros lenguajes de acuerdo a su perfil de investigación. Hasta ese momento nunca se ha enfrentado al paradigma de la programación declarativa; o sea, está acostumbrado a "dictar órdenes" y la idea de declarar lo que se desea resolver y dejar que un mecanismo dado se encargue de resolverlo, le suena bastante rara. Por lo general el impacto cognoscitivo resulta violento

y a muchos estudiantes se les dificulta superar el esquema anterior, llegando incluso a rechazar el nuevo paradigma.

Por supuesto que al enseñarles un nuevo paradigma no se pretende, en modo alguno, que los estudiantes olviden lo que han aprendido antes, ya que cada herramienta de programación tiene su propio campo de acción y toca al especialista seleccionar el paradigma adecuado, de acuerdo al problema que desee resolver. Ahora bien, resulta claro que el estudiante debe ser capaz de razonar de acuerdo al medio de programación que esté usando y no mezclar ni extrapolar conceptos que pueden dar como resultado un código poco comprensible e ineficiente.

Con el fin de ayudar a resolver este problema se desarrolló el sistema LogSim (Logic Simulation), cuyo objetivo central es ayudar al estudiante a comprender el paradigma de la Programación Lógica.

2. Concepción general del sistema.

Para el diseño y la programación del sistema LogSim se utilizó el paradigma de la Programación Lógica, de manera que los propios programas fuentes sirvan como ejemplos de programación para aquellos estudiantes que están interesados en analizar el sistema "por dentro".

El sistema incluye un tutorial estructurado en forma de hipertexto, el cual explica teóricamente los conceptos fundamentales del lenguaje, a fin de que el estudiante pueda analizarlos tomando en cuenta su definición formal o más precisa, aunque consideramos que la barrera fundamental para aprender algo no está generalmente determinada por la mayor o menor dificultad para manejar la información, sino por la carencia de motivación del estudiante para hacer algo con ella. Varios estudios señalan que la información que se recibe pero no se usa durante el proceso de aprendizaje, difícilmente se recuerda cuando se necesita [Schank, 96].

El componente principal de LogSim es su herramienta de simulación, la que dispone de un conjunto de programas ejemplos, que facilitan el aprendizaje al mostrar los caminos de ejecución tomados y poder apreciar cómo influyen éstos sobre cada una de las estructuras de datos utilizadas.

3. Aspectos pedagógicos.

Se dice, no sin razón, que el uso de un código existente permite a los estudiantes terminar un proyecto en un lapso de tiempo razonable [Walker, 94]. Consideramos que se logra el éxito si se emplea un código bien hecho, que permita la asimilación de un estilo de programación adecuado. Esta es la idea medular del sistema LogSim: enseñar a programar observando el comportamiento de ejemplos significativos que ilustran los mecanismos fundamentales del lenguaje, mostrando cómo dichos mecanismos resuelven el problema planteado.

Se ha procurado crear un banco de problemas que sean conocidos por los estudiantes; algunos de ellos son tomados de la vida real y otros resultan clásicos pero requieren de cierta dosis de ingenio para resolverlos.

Entre los primeros se puede mencionar el programa para obtener las derivadas de una función. Al estudiante se le pide que presente las ideas generales para elaborar un programa con ese fin. En este punto conviene que se le permita justificar la elección de un lenguaje X para enfrentar la tarea y después mostrarle las ventajas que brinda el Prolog cuando se hace procesamiento simbólico. Este tipo de ejemplo sirve, además,

para establecer un vínculo entre la Programación Lógica y otras asignaturas como el Análisis Matemático en este caso.

En el segundo grupo de problemas se destacan, entre otros, el de ubicar ocho reinas sobre un tablero de ajedrez sin que se ataquen mutuamente, el de las torres de Hanoi, etc. Dichos problemas contribuyen notablemente al desarrollo del pensamiento lógico, y en este caso se logra vincular la Programación Lógica con aspectos que se abordan en otros cursos, particularmente en el de Matemática Discreta.

Un tercer grupo de problemas, [1] que resulta de la combinación de los dos anteriores, está formado por ejercicios propios de alguna asignatura de la especialidad; por ejemplo, el problema de programar un autómata de pila no determinístico, que se vincula a la asignatura Compiladores. El
autómata es una necesidad real en un compilador, pero la simplificación que se hace del mismo, en función de la enseñanza, sólo abarca la idea de reconocer una cadena de entrada simple, para no abrumar a los estudiantes con muchos conceptos que compliquen el proceso de aprendizaje.

4. Forma de utilización.

Como se mencionó anteriormente, el enfoque que instrumenta el sistema como recurso fundamental consiste en visualizar la trayectoria de ejecución de los programas y los efectos que se producen sobre sus datos.

Tomemos por ejemplo el caso de un autómata de pila no determinístico. Cuando el estudiante selecciona este problema, se le presenta una definición formal:
"Un autómata de pila es un 7-tuplo $P = (Q, \Sigma, \delta, \Gamma, q_0, z_0, F)$, ..."
el enunciado permite recordar los conceptos abordados en otra asignatura de la especialidad.
Seguidamente el sistema pedirá al alumno que defina:

- el conjunto de estados (Q).
- el conjunto de símbolos del alfabeto (Σ).
- el conjunto de transiciones entre cada uno de los estados (δ).
- el conjunto de símbolos permitidos en la pila (Γ).
- el estado inicial (q_0).
- el símbolo inicial en la pila z_0.
- el conjunto de estados finales (F).

De manera sencilla, el estudiante participa en la construcción de su conocimiento, ya que a partir de los datos que él suministra se generan algunas de las cláusulas Prolog que formarán el programa. Otras cláusulas conforman la parte invariable del problema a resolver y son generadas sin intervención del estudiante. Por último, se solicita la cadena de entrada que deberá ser analizada por el autómata para decidir si la acepta o no.

---

[1] Según [Reeves, 1991] en un aula el conocimiento se presenta muchas veces como algo acabado; sin embargo, sería más adecuado considerar el conocimiento como un algo para resolver problemas o interpretar eventos. De acuerdo a la concepción del sistema LogSim, algunos programas son generados a partir de las ideas elaboradas por los estudiantes, otros son parcialmente generados y el sistema en general se puede modificar sobre la base de lo que se aprende con su ayuda.

Posteriormente se le presentará al estudiante un programa formado por las cláusulas derivadas del diálogo, más una parte invariable. Cuando LogSim se aplica en clases, el profesor debe destacar por qué existe esa parte invariable y en el caso que el alumno lo esté usando de forma individual, se le debe haber orientado algún ejercicio para que analice el programa correspondiente y justifique cada una de sus cláusulas.

En el caso del autómata, la pila y la cadena de entrada se simulan mediante listas Prolog, Estas y otras estructuras se presentan en ventanas aparte, de manera que se pueda apreciar en cada instante los valores con que están instanciadas y cómo reciben su valor.

Seguidamente LogSim permite ejecutar paso a paso el programa, pudiendo apreciarse los mecanismos típicos del lenguaje, tales como: la unificación de patrones, el backtracking, la recursividad y otros.

La gama de ejemplos disponibles abarca todos los mecanismos fundamentales de la programación lógica y, en virtud de la concepción del sistema, se puede plantear como ejercicio de culminación del curso el diseño y programación de pequeños programas que bs propios estudiantes consideren adecuados para ayudar a otros a aprender lo que ellos han asimilado. Estos nuevos ejemplos son incorporados al sistema por los propios estudiantes. Nuestra experiencia muestra que a los alumnos les resulta altamente motivante el hecho de darles la oportunidad de apoyar la labor docente.

Prolog es un lenguaje declarativo y sus usuarios deben seguir el principio de centrarse en el "qué hacer", declararlo y dejar que el lenguaje resuelva el problema, y no preocuparse por cómo lo resolverá. Pudiera pensarse que una herramienta que presenta el funcionamiento interno del lenguaje no sea didácticamente la más adecuada, pero la experiencia de aplicación de LogSim ha demostrado que los estudiantes no aprovechan las posibilidades del lenguaje hasta que logran comprender la potencia de sus mecanismos. El ejemplo antes mencionado resulta típico, ya que la solución se basa, ante todo, en lo conceptos de unificación de patrones y "backtracking" y cuando éstos no se dominan bien, la tendencia es a resolver el problema usando estrategias imperativas tradicionales despreciando los recursos lógicos fundamentales del lenguaje declarativo.

Lo primero que debe enseñarse en un curso sobre programación lógica debe ser el paradigma de programación, sin entrar en los detalles propios de una implementación particular. En consecuencia hemos utilizado LogSim de la siguiente forma:

- El hipertexto se ha empleado para apoyar el estudio de los contenidos teóricos impartidos en clase. Esta parte del sistema solo hace referencia a conceptos generales y puede usarse desde el inicio del curso de forma independiente, en el laboratorio.
- Los programas que simulan el comportamiento del lenguaje han servido para introducir nuevos conceptos. Las primeras actividades de este tipo deben ser dirigidas por el profesor, pero el estudiante debe asumir seguidamente un papel activo. Se puede hacer, por ejemplo, la ejecución paso a paso del programa y explicar sólo los detalles necesarios para que los estudiantes puedan deducir por sí mismos cómo funciona el mecanismo objeto de análisis.
- Se han orientado actividades independientes con objetivos claramente establecidos. Por ejemplo: "analizar la forma en que el programa deriva una función dada". Los resultados de esta actividad se discutieron en un seminario y se extrajeron conclusiones a partir de los diversos criterios planteados.
- Cuando el curso hubo avanzado lo suficiente y los estudiantes tenían los conocimientos necesarios, se les explicó cómo está construido el sistema que le sirve de apoyo al estudio.

- En la etapa final del curso se entiende que el dominio alcanzado por los estudiantes con relación a los conceptos estudiados y las propias dificultades a las que se han enfrentado los facultan para proponer modificaciones a los programas disponibles o nuevos programas que permitan ayudar a comprender mejor los conceptos estudiados. Se solicitaron propuestas a los estudiantes, las que fueron discutidas en el aula y, una vez obtenido el consenso necesario se planteó como un proyecto extra clase, la implementación de las ideas analizadas.

5. Conclusiones.

El uso del sistema LogSim y otras herramientas informáticas de aplicación docente ha permitido que los estudiantes alcancen un mayor dominio del lenguaje Prolog.

Los estudiantes participan activamente en la construcción de sus conocimientos, pudiendo incluso modificar y enriquecer los medios de enseñanza utilizados, a fin de ayudar a otros estudiantes que utilizarán el sistema en cursos posteriores.

En nuestra opinión la enseñanza de la programación debe iniciarse por la forma declarativa y no por la forma imperativa. Se conoce de experiencias sobre la enseñanza de la Programación Lógica a niños pequeños, con resultados sorprendentes. Esto se debe a que la forma de pensar del ser humano se acerca más a la lógica y cuando se enseña a programar en forma imperativa realmente se trata de imponer una vía de razonamiento que no es propio de los humanos, la cual está determinado, en última instancia, por la arquitectura secuencial de las computadoras y no por nuestros mecanismos de razonamiento. Creemos que la Programación Lógica y los enfoques que se orienten más a la forma de actuar humana, deben ser objeto de una mayor atención, presentamos este trabajo como un modesto aporte en esta dirección.

6. Bibliografía.

- Lezcano, M. Prolog y Sistemas Expertos (1995). Ediciones Universidad. Central de Las Villas. Santa Clara, Cuba.
- Reeves, T (1991). Designing CAL To support learning: The case of multimedia in Higher Education Nordic Conference on Computer Aided Higher Education. Helsinki University of Technology, Finland.
- Rich, E. y Knight, K. (1994). Inteligencia Artificial Segunda edición McGraw-Hill.
- Roth, A. (April 1993). The practical application of Prolog. Artificial Intelligence Expert.
- Roth, A. Spencer Clive (1993). The benefits of Prolog. Software Development.
- Schank, R. and Kass, A. (1996). A Goal-Based Scenario for High School Students. Communications of the ACM. Vol. 39 No. 4.
- Spencer, C. (July 1996). LPA-flex 1.2 Technical Product Information.

- Spencer C. (1995). The beginning of Prolog. Prime marketing publications LTD.
- Thomas, S. R. (1994). A consideration of some approaches to course organization. AAAI Fall Symposium on Improving Instruction of Introductory AI (IIIA).
- Walker, E. (1994). The use of Computers for teaching Artificial Intelligence. AAAI Fall Symposium on Improving Instruction of Introductory AI (IIIA).