UNIÓN
REVISTA IBEROAMERICANA DE EDUCACIÓN MATEMÁTICA

Marzo de 2005, Número 1, páginas 37 - 44
ISSN: en trámite

Algoritmos en la enseñanza y el aprendizaje de las matemáticas

Antonio J. Pérez Jiménez

1. Revisión de los algoritmos tradicionales de la enseñanza elemental

La visión que socialmente se tiene de la enseñanza elemental de la Aritmética es principalmente la de un entrenamiento para la adquisición de habilidades de cálculo. Esta es también, en parte, la opinión del profesorado; y no sólo en mi país. Los algoritmos tradicionales de suma, resta, multiplicación y división han sido un hallazgo tan extraordinario y eficaz que, como todo aprendizaje consolidado, es un obstáculo (epistemológico) para su renovación.

Sin embargo esta posición preponderante de dichos algoritmos va perdiendo fuerza.

En primer lugar hay que destacar la cada vez más precaria ubicación en los programas del algoritmo usual de la raíz cuadrada. Son muy pocos, y cada vez menos, los profesores que enseñan a sus alumnos este algoritmo. ¿Por qué? Se dan múltiples razones, todas ellas loables; pero entonces la pregunta es: ¿no valen esos mismos argumentos para la progresiva eliminación en la enseñanza de los algoritmos usuales de suma, resta, multiplicación y división? En segundo lugar, el fuerte movimiento de renovación pedagógica que arranca en los años 50 (*Resolución de problemas*, de Polya, *Enseñanza dinámica*, del grupo interdisciplinar CIEAEM) y que resurge con una inspiración constructivista y en contra de los postulados de la llamada “matemática moderna”, cambian el centro de gravedad de la enseñanza y lo sitúan en el aprendizaje, es decir, el protagonismo pasa del profesor al alumno. Importa lo que el alumno pueda asimilar y para ello lo mejor es que sea él mismo quien construya sus conocimientos; y, con estos postulados, parece innegable que los algoritmos usuales pierden interés. En tercer lugar, la utilidad social de dichos algoritmos. Hoy no se ve a nadie, como en los tiempos de mi infancia y juventud, realizar las cuentas sobre un papel, o sobre la barra de un bar; las máquinas han ocupado este lugar. En cuarto lugar, los movimientos, proclamas (*Let's Abolish Pencil-and-Paper Arithmetic*, de Anthony Ralston; *Manifiesto en contra de los algoritmos tradicionales de las cuatro operaciones aritméticas y de la raíz cuadrada,* del Colegio Público Aguamansa de las Islas Canarias) a favor de la abolición de los algoritmos tradicionales. Estos movimientos sitúan como prioritarias las estrategias para el cálculo mental, frente a las habilidades que dichos algoritmos proporcionan; en lugar de la vieja tecnología del lápiz y papel proponen la utilización de la calculadora.

Parece pues claro que la enseñanza de los algoritmos básicos está en revisión. Pero no sólo eso, también en general se tiende a confundir un enfoque algorítmico con un tratamiento no significativo del aprendizaje. En esto ha tenido mucho que ver la oposición que las teorías constructivistas han opuesto a las doctrinas conductistas. En el conductismo la enseñanza se produce a través de una *enseñanza programada* en la que los algoritmos, que pueden ser implementados como programas en una *máquina de enseñar*, guían el aprendizaje del alumno. Estos métodos pretenden obviar el *error* y así, cuando un alumno yerra, el programa vuelve atrás (feed-back) automáticamente para que asimile los pasos anteriores –que se suponen mal aprendidos-. El constructivismo sostiene de manera radicalmente opuesta, que los errores no hay que evitarlos, sino más bien sacarlos a la luz, diagnosticarlos y proceder en consecuencia.

En mi país (y en otros muchos) los currículos oficiales son constructivistas y, en consecuencia, los algoritmos están devaluados. ¿Debe ser esto así?

2. ¿Qué es un algoritmo? Resolución algorítmica

La noción de algoritmo se ha manejado a lo largo de la historia de manera totalmente informal e intuitiva. La idea de algoritmo como secuencia de instrucciones elementales ha parecido siempre tan obvia que nadie se había planteado, hasta finales del siglo XIX, dar una definición formal del mismo. Es muy claro cuándo un problema se resuelve algorítmicamente: basta con encontrar un procedimiento mecánico que pueda ser considerado como tal. Sin embargo, para probar que un problema no es resoluble algorítmicamente se necesita saber con rigor qué es un algoritmo. La cuestión de si todo problema es resoluble algorítmicamente está implícita en el programa finistista de Hilbert y, en 1936, A. Turing da una respuesta negativa a la misma a través de un célebre problema conocido como *el problema de la parada*. Para ello, Turing define la noción de algoritmo a través de un artilugio teórico conocido como *máquinas de Turing*. Otros insignes matemáticos de la época –estamos hablando de los años 30- (Gödel, Church, Kleene) introducen por distintas vías la noción formal de modelo de computación tratando de capturar la idea intuitiva del concepto de algoritmo. Además se formula una hipótesis no refutada hasta hoy: la hipótesis de Church (o de Church-Turing) que afirma que dichos modelos (todos equivalentes) capturan completamente la noción intuitiva de computación.

Así, pues, existen problemas para los que no hay ningún algoritmo que lo resuelva. El interés de esta afirmación, en el contexto de esta conferencia, es que si todos los problemas fuesen resolubles algorítmicamente se justificaría plenamente, desde el punto de vista matemático, una enseñanza de los algoritmos, dado que al fin y a la postre la matemática resuelve problemas. Pero puesto que no es así tendrá que utilizarse otros argumentos para justificar la enseñanza de los mismos.

Curiosamente los modelos de computación son establecidos antes de que aparecieran los primeros ordenadores (en el sentido actual). Es decir, antes de que éstos vieran la luz ya se había determinado qué podían hacer y qué no podían hacer. La aparición de los ordenadores (como máquinas de propósito general o

máquinas universales según el sueño de Leibnitz) supone una revolución en la que estamos inmersos. Tras los trabajos pioneros de von Neumann, los ordenadores son programados a través de *lenguajes de programación* y, como máquinas de propósito general que son, cualquier problema resoluble algorítmicamente puede ser resuelto por una de estas máquinas. Claro que no importa sólo esto pues entra en juego el problema de la eficiencia: no nos vale desde el punto de vista práctico un algoritmo que necesite una gran cantidad de tiempo para resolver instancias de un problema de tamaño medianamente grande. Es necesario, pues, analizar la cantidad de recursos necesarios (cuantificados en tiempo y/o memoria) para la ejecución de un algoritmo. Así nace la teoría de la complejidad computacional en cuyo marco aparece uno de los problemas abiertos más relevantes de la actualidad: el problema **P versus NP**. Una respuesta afirmativa a la conjetura **P** = **NP** significaría que determinados problemas podrían resolverse en un tiempo considerablemente menor (polinomial) y conllevaría algunas consecuencias muy importantes. Por ejemplo, los sistemas de computación carecerían de eficacia en tanto en cuanto cualquier texto cifrado podría desencriptarse mecánicamente y de forma rápida. Por ello habría que cambiar la filosofía de los sistemas de encriptación. Pero por ahora no hay que temer pues hoy por hoy todo apunta a una respuesta negativa de dicha conjetura.

3. Algoritmos y nuevas tecnologías

Todo algoritmo puede traducirse en *un programa* escrito en un lenguaje de programación y todo *programa* es un algoritmo. Todo problema resoluble algorítmicamente puede ser resuelto mecánicamente por un ordenador. Si tenemos en cuenta la revolución de las nuevas tecnologías de la información y si éstas han de tener una incidencia en la enseñanza, parece que los algoritmos vuelven a adquirir un papel relevante. Un ejemplo: el lenguaje *Logo* es una extraordinaria herramienta para la enseñanza de la geometría. Los alumnos han de escribir un programa muy sencillo para obtener como respuesta un dibujo que representa el objeto deseado. *Logo* fue toda una revolución en el panorama de la enseñanza de mediados de los años 80, pero muy rápidamente perdió parte de su empuje.

Logo es una aplicación educativa tal que, a través de programas formales (*procedimientos* en la terminología de este lenguaje) el alumno realiza sus tareas. El diseño de programas en un lenguaje de programación para resolver problemas ocupa hoy un lugar muy importante en el quehacer social; en el contexto escolar ese proceso de diseño tiene, entre otras, la virtud de conectar la realización de una tarea con la utilización explícita de un lenguaje, es decir, de conexión directa del significado con el significante.

Pero hay otras aplicaciones que tienen un fuerte carácter algorítmico aunque no se manejen con un lenguaje formal; un ejemplo es el *Cabri-Géomètre* o *Cabri*.

El *Cabri*, como el *Geometer's Sketpad* o *Cinderella*, es otra aplicación informática que está cambiando sensiblemente el enfoque y los métodos de la enseñanza de la geometría. Es una herramienta muy eficaz para la generalización geométrica (lo que permite la realización de pruebas matemáticas) y para la

elaboración de conjeturas (una de sus principales virtudes desde mi punto de vista); pero lo que quiero enfatizar ahora es su capacidad para el manejo y elaboración de algoritmos (como, por ejemplo, los de regla y compás).

Cuando el alumno se enfrenta a una tarea con el *Cabri* es usual que tenga que elaborar, implícitamente en muchos casos, un algoritmo informal para resolver el problema implicado en esa tarea. El *Cabri* proporciona, en los menús, las instrucciones elementales que el alumno ha de utilizar para obtener un algoritmo que resuelva el problema. El profesor, si lo desea, puede modificar (suprimir, añadir) esas instrucciones elementales lo que permite por una parte el aprendizaje y asimilación de determinadas instrucciones (no consideradas elementales por el profesor) y, por otra, la agilización en las tareas mediante la introducción de macro-instrucciones.

El manejo de las nuevas herramientas que proporcionan las actuales tecnologías posee una carga algorítmica muy importante (por supuesto, va mucho más allá al adentrase en el terreno de la conjetura y de la prueba matemática). Concretando en el objeto de esta conferencia, podemos concluir que la nueva perspectiva tecnológica favorece la utilización de algoritmos tanto en la enseñanza como en el aprendizaje de las matemáticas.

4. ¿Qué algoritmos?

Si un problema es resoluble algorítmicamente existen distintos algoritmos que lo resuelven (es más, existen infinitos algoritmos). La utilización de ábacos (la yupana incaica, el suan-pan chino, el soroban japonés, etc) condicionan un determinado tratamiento algorítmico de las operaciones elementales alejado (independiente) de la notación numérica. La multiplicación mediante celosías, el ábaco de arena, suponen ya un cálculo vinculado a la notación. La multiplicación rusa se basa en el conocimiento de la suma, del doble y de la mitad (por defecto). Las regletas de Neper, que reducen la multiplicación a la suma, introducen un elemento más alto de automatización en el cálculo. Las máquinas de Pascal y Leibnitz son el primer paso hacia la automatización del cálculo de las operaciones elementales. Las actuales calculadoras electrónicas permiten la utilización de "algoritmos ingenuos" (pronto veremos un ejemplo) en lugar de los algoritmos usuales, y las calculadoras gráficas proporcionan una fuerte capacidad de visualización en los problemas de estudio de funciones. Todo ello sea dicho como argumentos para concluir que los algoritmos dependen, entre otros factores, de la tecnología disponible en cada momento cultural. Los algoritmos de lápiz y papel, vinculados además al enorme hallazgo del cero, han supuesto un gran éxito en la historia del cálculo al ser fuertemente automatizados y depender sólo de una tecnología tan usual y barata como el lápiz y papel (o similares).

Pero tras el formidable invento de nuestras actuales máquinas de calcular, ¿no van a ser relegados el lápiz y papel? Estarán conmigo en que la pregunta es muy pertinente y que los intentos de abolicionistas están fuertemente basados.

Pongamos un ejemplo. Para calcular la raíz cuadrada entera de un número podemos proceder con un *algoritmo ingenuo*: mediante ensayo y error buscamos un número tal que su cuadrado sea menor que el número dado y, sin embargo, el cuadrado de su siguiente lo sobrepase. Este método, implementado en lápiz y papel tiene una enorme desventaja frente al tradicional: no es tan mecánico y es más lento; tan lento que salvo en cálculos de números pequeños se hace inviable (mucho más si se quiere obtener una solución con decimales). Pero tiene unas enormes ventajas desde el punto de vista conceptual pues en cada paso se está manejando la definición de raíz cuadrada como el inverso (y la reversibilidad importa mucho desde el punto de vista de la psicología del aprendizaje); además, para dar el mínimo número de pasos el alumno debe efectuar *cálculos mentales* aproximados. Y, por abundar, hemos de decir que puede extenderse fácilmente al cálculo de otras potencias.

Pues bien, si a las ventajas indicadas unimos que las desventajas pueden obviarse con la utilización de la calculadora, comprenderán ustedes que yo me decante por este “algoritmo” de ensayo y error. La enseñanza tiene la misión de elegir en cada etapa histórica los algoritmos más pertinentes.

Ya he comentado que hay quienes pretenden que la calculadora se emplee desde la más tierna infancia y desaparezcan los algoritmos de lápiz y papel. Creo que los algoritmos de suma, resta, multiplicación y división pueden ser sustituidos por otros que utilicen la calculadora (combinados posiblemente con técnicas no algorítmicas) pues, como en el *algoritmo ingenuo* de la raíz cuadrada, se favorece claramente el cálculo mental y, de camino, se elimina un elemento memorístico que consume gran parte del tiempo del alumno en un momento en que, además, se supone que está ávido de conocimientos. Pero he de reconocer que en el caso de las cuatro operaciones básicas es bastante más difícil pues aparte de la raigambre tan profunda que poseen dichos algoritmos y, por qué no decirlo, de su belleza, la resistencia social y del profesorado tiene su razón: la relación del enseñante con el pupilo y con su entorno es, en términos de Guy Brosseau, un *contrato didáctico* (implícito); en esa relación contractual los algoritmos usuales cumplen el papel de unas reglas fijas y muy claras. Su “eliminación” ha de venir por la vía de la sustitución de ese contrato con otras reglas aceptadas por la generalidad y eso, pienso yo, va a llevar su tiempo. El contrato didáctico es más fuerte (o mejor, más conservador) que el contrato social.

5. Algoritmos en la enseñanza y en el aprendizaje de las matemáticas

Las teorías sobre el aprendizaje significativo han de tomarse (al igual que tantas teorías sobre la enseñanza) como una tendencia. Me explico: la enseñanza de determinadas habilidades como la del algoritmo usual de la raíz cuadrada es memorística, no significativa. ¿Quiero ello decir que no debieron enseñarme dicho algoritmo? Yo no estoy de acuerdo y la razón es bien simple: con esa habilidad yo podía resolver de manera efectiva los problemas elementales de áreas, de semejanza cuadrática. A su vez, el manejo de éstos me ayudó a adquirir la noción

de raíz cuadrada. Pero hoy, con la poderosa calculadora en nuestras manos, yo defiendo el *algoritmo ingenuo*, es decir, un aprendizaje enmarcado en una enseñanza significativa.

En la enseñanza de una operación básica se hace un *paréntesis* para enseñar una rutina necesaria para su cálculo: el algoritmo tradicional. Pero, ¿había otra elección cuando a mí, a finales de los años 50, me enseñaron a sumar? Y aunque la hubiera (Felix Klein, ya en 1910, propugnaba la utilización de la Brunsviga -calculadora mecánica- como instrumento de formalización de las operaciones básicas), ¿se daban las condiciones culturales, de divulgación tecnológica y de actualización del profesorado adecuadas? La respuesta a ambas preguntas es negativa. (Una precisión: las regletas de Cuisenaire se han utilizado, y se utilizan, para la conceptualización de las operaciones básicas, pero nunca como una alternativa al cálculo).

Una pregunta crucial en el caso de las operaciones aritméticas básicas es la siguiente: ¿puede evitarse ese *paréntesis* o hacerlo lo más corto posible? Dicho de otra forma, ¿puede integrase un algoritmo de calculo en el aprendizaje del concepto correspondiente? Cuando esto ocurre la enseñanza gana en significatividad; el algoritmo correspondiente es, entonces, un "*algoritmo ingenuo*" que, como en el caso de la raíz cuadrada, irá acompañado normalmente de diversas estrategias heurísticas.

Para calcular la fórmula de la distancia de un punto a una recta el profesor utiliza el siguiente algoritmo en su demostración; tras señalar los datos, una recta r y un punto P genéricos, realiza los siguientes pasos: 1) Calcula la ecuación de la recta que pasa por P y es perpendicular a r; 2) Calcula el punto de intersección de r con la recta obtenida en el paso anterior; 3) Calcula la distancia entre dos puntos.

A partir de este momento el alumno utilizará la fórmula obtenida.

En la enseñanza tradicional este algoritmo es frecuentemente "ocultado" por el profesor. En una enseñanza significativa el manejo de dicho algoritmo por parte del alumno a través de ejercicios con distintos grados de concreción, funciona como un todo en el reforzamiento del conocimiento implícito contenido en las instrucciones y en la consecución del objetivo deseado, que puede llegar, si así se ha propuesto, a la obtención de la fórmula.

Tanto el algoritmo anterior como los *algoritmos ingenuos* citados son utilizados como *medio* para la conceptualización y para el cálculo (de las operaciones básicas, de las fórmulas).

Pero en muchos casos la situación planteada puede llevar a la obtención de algoritmos; es decir, los algoritmos se convierten así en *objetivos* a conseguir. Cuando se propone un ejercicio a resolver con un lenguaje de programación el alumno ha de elaborar los algoritmos correspondientes (y traducirlos a dicho lenguaje). El trabajo con *Logo* o con *Cabri* aboca en muchos casos a convertir los algoritmos en objetivos.

Esta clasificación de los algoritmos como *medio* o como *objetivo* no es disjunta; depende del tratamiento, de la situación que planteemos. Por ejemplo, el algoritmo que señalábamos para la distancia de un punto a una recta puede ser solicitado de una manera natural en el contexto del Cabri. Basta que, en dicho contexto, planteemos al alumno que calcule la distancia de un punto a una recta concretos.

6. La recursión

Con la introducción de las nuevas tecnologías nuevos contenidos y nuevos métodos solicitan un lugar en la enseñanza. La teoría de grafos, la de fractales, las técnicas de matemática discreta, la recursión como método, son ejemplos poderosos que están en la mente de todo profesor renovador. Vamos a centrarnos en la recursión. Para fijar ideas pondremos dos ejemplos.

El primero va a consistir en un ejercicio dedicado a la obtención de la suma de *n* números naturales consecutivos. Tradicionalmente este ejercicio ha sido enfocado desde el tratamiento clásico de las progresiones geométricas. Pero veamos otra manera de enfocarlo. Supongamos que queremos sumar de 1 a 10. De manera simbólica podemos expresar el problema así:

Sumar_hasta 10 = 1+2+3+4+5+6+7+8+9+10 = 10 + Sumar_hasta 9.

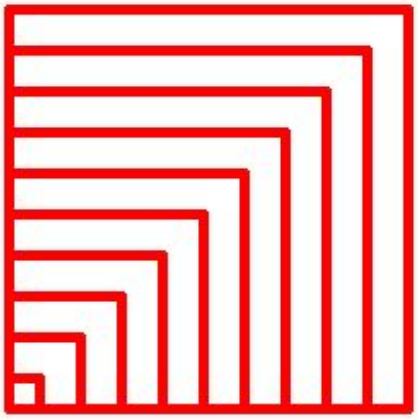
Conceptualmente hemos hecho algo evidente: para sumar hasta 10 sumamos a 10 el resultado de sumar hasta 9. Este es un ejemplo de recursión donde para obtener un cálculo definido utilizamos la propia definición (*ley de recursión*). Si seguimos avanzando en el cálculo según la ley de recursión, tendríamos que efectuar, a su vez, **Sumar_hasta 9,** para la que utilizaremos **Sumar_hasta 8** y así sucesivamente. ¿Hasta donde? Parece claro que pararemos (*caso base*) cuando lleguemos a 1, así: **Sumar_hasta 1 = 1**. Un programa informal que calcule la suma hasta n sería:

Sumar_hasta n

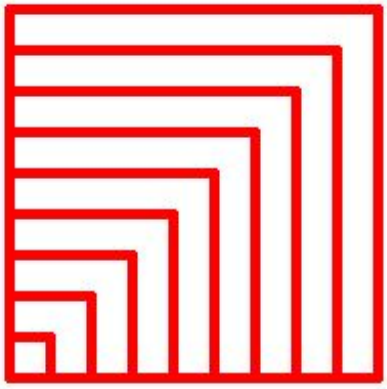
Si n=1 devuelve 1 y Termina

n + Sumar_hasta n-1

El segundo ejemplo lo vamos a extraer de Logo y es análogo al anterior. Observemos los cuadrados encajados que aparecen en la figura. Simbólicamente expresaremos esta construcción con el nombre **Cuadrados.encajados 10** (donde 10 representa el lado del cuadrado mayor).



cuadrados.encajados 10



cuadrados.encajados 9

La visión recursiva de esa figura es la siguiente: si dibujamos el cuadrado exterior lo que nos queda **es lo mismo** pero de lado 9 (menor). Así pues, podemos decir a una máquina que dibuje los cuadrados encajados así:

Cuadrados.encajados n

Si n=0 Termina

Cuadrado n

Encajar

Cuadrados.encajados n-1

La realización de algoritmos recursivos requieren una gran cantidad de espacio (memoria) y tradicionalmente se han sustituidos por algoritmos iterativos que realizan la misma tarea global (otros, como la construcción de árboles o el clásico Copo de nieve, tienen un tratamiento natural desde el punto de vista de la recursión mientras que de forma iterativa son mucho más complejos y artificiales). El mismo Pascal, quien inventase la primera máquina de calcular, abordó de forma recursiva el famoso *problema de los repartos*. Fermat trató este mismo problema explicitando las posibilidades; ambos inventaron el moderno cálculo de probabilidades, pero las técnicas de Pascal no se impusieron porque no existía la tecnología adecuada.

Hoy existen máquinas poderosas y, por supuesto, lenguajes recursivos como *Logo* que permiten su tratamiento de una manera natural. Es quizás el momento de introducir esta poderosa técnica conceptual.