Análisis asintótico de relaciones de recurrencia en complejidad algorítmica mediante la teoría de ecuaciones en diferencia

Oscar Valero*
Departamento de Ciencias Matemáticas e Informática
Universidad de las Islas Baleares
Ctra. de Valldemossa km. 7.5, 07122, Palma de Mallorca, Spain
o.valero@uib.es

***Resumen*—La teoría de ecuaciones en diferencias permite modelar satisfactoriamente múltiples procesos que surgen de modo natural en diversas áreas de las ciencias aplicadas. En concreto, las ecuaciones en diferencias resultan de gran utilidad en Ciencia de la Computación. El objetivo del presente artículo es introducir a los estudiantes de las titulaciones de Grado en Matemáticas y Grado en Ingeniería Informática, así como a los futuros investigadores que se forman cursando el título de Máster en Tecnologías de la Información, en las técnicas de resolución de un tipo particular de ecuaciones en diferencias, las denominadas ecuaciones en diferencias finitas lineales, y mostrar su aplicabilidad al análisis de complejidad algorítmica.**

I. Motivación

En lo sucesivo las letras $\mathbb{N}$, $\mathbb{Z}^+$, $\mathbb{R}$, $\mathbb{R}^+$ y $\mathbb{R}_0^+$ denotarán el conjunto de los números enteros positivos, el conjunto de los números enteros no negativos, el conjunto de los números reales, el conjunto de los número reales no negativos y el conjunto de los número reales positivos, respectivamente.

En Ciencia de la Computación el análisis de la complejidad de un algoritmo se basa en determinar matemáticamente la cantidad de recursos (**coste computacional** o **coste de ejecución**) que dicho algoritmo necesita para resolver el problema para el que ha sido diseñado. Dos de los recursos típicos, los cuales juegan un papel fundamental en el análisis de la complejidad algorítmica, son el tiempo empleado por el algoritmo bajo estudio para resolver un problema (**tiempo de computación** o **tiempo de ejecución**), y la memoria (**espacio**) empleada por la computadora que ejecuta el algorítmo al resolver el problema bajo estudio.

Por otro lado, es habitual que existan varios algoritmos que resuelvan un problema dado. Por este motivo, uno de los objetivos principales del análisis de complejidad algorítmica es discernir cuál de todos los algoritmos que resuelven un problema lo hace con menos coste computacional. Este hecho motiva que, para averiguar qué algoritmo es más eficiente (en términos de coste), sea necesario comparar sus costes de computación. Para poder llevar a cabo la mencionada comparación se requiere el uso de herramientas matemáticas que permitan representar para cada algoritmo su coste computacional. Así, en el análisis de complejidad algorítmica, el coste de computación de un algoritmo es denotado por una función $T : \mathbb{N} \to [0, \infty)$ de modo que $T(n)$ representa el coste empleado por el algoritmo en resolver el problema bajo consideración cuando el dato de entrada del algoritmo tiene un tamaño n.

El método de representación introducido presenta un pequeño inconveniente, ya que el coste de ejecución de un algoritmo no depende exclusivamente del tamaño del dato de entrada n, sino que éste también depende de la distribución particular de los datos de entrada. Obviamente, y como ejemplo ilustrativo, si se considera un algoritmo de ordenación de modo que, dado un vector de n entradas numéricas, el algoritmo proporcione como resultado el vector con los datos almacenados en orden creciente (ejemplos de este tipo de algoritmos son Mergesort y Quicksort), entonces no tardará lo mismo en resolver el problema nuestro algoritmo si el vector está totalmente ordenado o si está completamente desordenado (los datos almacenados están en orden decreciente). Por tanto, el coste puede variar sustancialmente al procesar ejemplares diferentes (con distinta distribución) de los datos de entrada con un mismo tamaño. Con el objetivo de resolver este inconveniente, en el análisis de complejidad algorítmica se distinguen tres posibles comportamientos en cuanto a coste computacional se refiere. Estos son los denominados **mejor caso**, **peor caso** y **caso promedio** en coste de ejecución.

Para un algoritmo dado y un tamaño de dato fijado, se define el mejor caso (peor caso) en coste de ejecución como el coste de ejecución empleado por el algoritmo en cuestión para procesar aquel dato, del tamaño fijado, cuya distribución requiere menos (más) coste de procesado o, de manera equivalente,

$$C_M = \min_{|X|=n} C_X \qquad (C_P = \max_{|X|=n} C_X),$$

donde por C_X se ha representado el coste empleado por el algoritmo para resolver el problema con un dato de entrada con distribución X, y se ha indicado que el tamaño de los datos de entrada es n mediante la expresión $|X| = n$.

El análisis algorítmico basado en el comportamiento en el peor y mejor caso en coste computacional permite acotar la complejidad de un algoritmo, ya que para todos aquellos datos cuya distribución no se corresponda con la del peor (mejor) caso, el coste de ejecución será menor (mayor). Por este motivo el análisis basado en estos dos casos es el más extendido. Sin embargo, hay ocasiones en las que un algoritmo se emplea muchas veces sobre datos de entrada con distinta distribución pero mismo tamaño y, por tanto, resulta más útil

*Este trabajo ha sido elaborado en el marco del proyecto de investigación MTM2009-10962 que desarrolla actualmente el grupo Lógica Borrosa y Fusión de la Información (LOBFI) de la UIB, y que ha sido financiado por el Ministerio de Ciencia e Innovación.

conocer su coste de ejecución en media, es decir, el caso promedio en coste de ejecución. Así, para un algoritmo dado y un tamaño de dato fijado, se define el caso promedio en coste de ejecución como el promedio de los costes de ejecución empleados por el algoritmo al procesar todas las posibles distribuciones de los datos de entrada con el tamaño fijado, o equivalentemente

$$C_{Prom} = \sum_{|X|=n} C_X \cdot Pr(X),$$

donde por $Pr(X)$ se ha denotado la probabilidad de que los datos de entrada tengan la distribución X.

Con el fin de eliminar cualquier posible ambigüedad al representar por $T(n)$ el coste de ejecución de un algoritmo, debe ser siempre especificado el tipo de distribución de los datos de entrada que el algoritmo bajo estudio está procesando.

En general, determinar exactamente cuál es la función que describe el coste computacional de un algoritmo es una tarea ardua. Por este motivo en muchas ocasiones se pretende obtener información sobre dicha función de un modo "aproximado", es decir, se desconoce exactamente el valor de $T(n)$ para cada tamaño de dato n pero se saben ciertas propiedades respecto al comportamiento de T. Una de las técnicas más empleadas para realizar un análisis aproximado de la complejidad de un algoritmo se conoce bajo el nombre de **análisis de complejidad algorítmica asintótico**. Éste se basa fundamentalmente en el hecho siguiente: supongamos que dado un algoritmo y una distribución de los datos de entrada estamos interesados en conocer su coste computacional T pero, por otro lado, no tenemos ninguna técnica para poder explicitar la expresión de $T(n)$ para cada n. Sin embargo es posible deducir que existen $n_0 \in \mathbb{N}$, $c_1, c_2 \in \mathbb{R}^+$ y una función $U : \mathbb{N} \to [0, \infty)$ tales que

$$c_2 U(n) \leq T(n) \leq c_1 U(n)$$

para todo $n \geq n_0$ (habitualmente denotado por $T \in \Theta(U)$). Entonces la función U acota asintóticamente de modo superior e inferior a la función T y, así, nos proporciona una información aproximada del coste computacional, para la distribución de los datos de entrada considerada, del algoritmo que está siendo estudiado. De hecho si $T \in \Theta(U)$, se tiene que el coste computacional $T(n)$ será como máximo (mínimo) el valor proporcionado por la función $c_1 U(n)$ ($c_2 U(n)$) para cada $n \geq n_0$. Obsérvese que la acotación se verifica para tamaños de dato superiores a un umbral n_0, y que este hecho significa que los costes de ejecución de nuestro algoritmo al procesar todos aquellos datos de tamaño n con $n < n_0$ son conocidos. Por tanto, uno de los objetivos fundamentales del análisis de complejidad algorítmica consistirá en la obtención de la función U que proporciona la información asintótica (aproximada) de la función T y, por tanto, del coste computacional, fijada la distribución de los datos, de nuestro algoritmo bajo estudio.

A pesar de que en muchos casos tan sólo se puede obtener una estimación del tiempo de computación de un algoritmo mediante la técnica de acotación asintótica, existe un gran número de algoritmos cuyo coste puede ser determinado de modo exacto, es decir, dada una distribucón de los datos a procesar, para cada tamaño $n \in \mathbb{N}$ se puede determinar exactamente la expresión de $T(n)$ y, así, es posible obtener su comportamiento asintótico de un modo sencillo. Estos algoritmos son aquellos cuyo coste computacional satisface una **relación de recurrencia**.

Los algoritmos que siguen estrategias del tipo **Divide y Vencerás** son un ejemplo representativo de este tipo de situación. En efecto, en muchos casos el coste computacional de estos algoritmos recursivos satisface una relación de recurrencia del tipo siguiente:

$$T(n) = \begin{cases} bn^\alpha & \text{si } 1 \leq n \leq c \\ aT(\frac{n}{c}) + bn^\alpha & \text{si } n > c \end{cases}, \qquad (1)$$

donde $a, b \in \mathbb{R}_0^+$, $\alpha \in \mathbb{Z}^+$ y $c \in \mathbb{N}$ con $c > 1$. En términos generales, bn^α representa el coste de descomponer el problema inicial en a subproblemas, cada uno de ellos con datos de entrada de tamaño $\frac{n}{c}$, y el de componer las soluciones de tales subproblemas para producir la solución del problema original a resolver.

Ejemplos típicos de algoritmos, que siguen estrategias Divide y Vencerás, cuyo coste computacional verifica una relación de recurrencia de tipo (1) son Mergesort (en el mejor, peor y caso promedio) y Quicksort (en el mejor caso).

Algunas referencias básicas sobre algorítmica y análisis de complejidad computacional, donde puede encontrarse una discusión detallada sobre los conceptos aquí expuestos, son [1], [2] y [3].

Obviamente, para poder realizar el estudio del comportamiento asintótico del coste de ejecución empleado por los algoritmos que siguen una estrategia de tipo Divide y Vencerás se requiere demostrar que las relaciones de recurrencia (1) admiten una única solución, ya que si éstas admitiesen más de una solución habría que discernir cuál de todas las posibles soluciones obtenidas representa el coste de ejecución del algoritmo bajo estudio. Además, una vez garantizada la existencia y unicidad de solución, se requiere analizar el comportamiento asintótico de dicha solución. El próposito fundamental de este trabajo es ilustrar que ambos objetivos pueden ser alcanzados con técnicas sencillas mediante el uso de la teoría de las ecuaciones en diferencias. Así, en la Sección II se introducen los conceptos básicos de ecuaciones en diferencias y las técnicas de resolución de los problemas de valor inicial asociados a éstas. La Sección III está dedicada a aplicar los conceptos expuestos en la Sección II al análisis asintótico de las soluciones de las relaciones de recurrencia de tipo (1) que representan el coste computacional de algunos algoritmos recursivos que siguen estrategias de tipo Divide y Vencerás. Finalmente, en la Sección IV se presentarán algunas ventajas de las técnicas introducidas frente a otras que también pueden ser encontradas en la bibliografía.

II. UNA BREVE INTRODUCCIÓN A LAS ECUACIONES EN DIFERENCIAS FINITAS

Dada una sucesión $(x_n)_{n\in\mathbb{N}} \subset \mathbb{R}$, se dice que es **recurrente** si existen $k \in \mathbb{N}$ y una función $\Phi : \mathbb{R}^{k+1} \to \mathbb{R}$ tal que

$$x_n = \Phi(x_{n-1}, x_{n-2}, \ldots, x_{n-k}, n) \qquad (2)$$

para todo $n > k$. El número natural k recibe el nombre de **orden de recurrencia** de la sucesión $(x_n)_{n\in\mathbb{N}}$.

Una **ecuación en diferencias finitas**, o simplemente **ecuación en diferencias**, es una sucesión recurrente del tipo (2) de la cual se desea conocer la expresión de su término general. Al número natural k se le denomina **orden** de la ecuación en diferencias finitas.

En el siguiente ejemplo se ilustra la definición precedente.

Ejemplo 1.

1. $x_n = 4x_{n-1} + 2^n \ \forall n > 1$ (orden 1).
2. $x_n = x_{n-1} + x_{n-2} + 4 \ \forall n > 2$ (orden 2).
3. $x_n = 5x_{n-1} - 8x_{n-2} + 4x_{n-3} \ \forall n > 3$ (orden 3).

□

Dada una ecuación en diferencias finitas de tipo (2) y un conjunto de valores $s_1, s_2, \ldots, s_{k-1}, s_k \in \mathbb{R}$, llamaremos **problema de valor inicial** asociado a (2) e inducido por $s_1, s_2, \ldots, s_{k-1}, s_k$ al problema de obtener todas aquellas soluciones de la ecuación en diferencias finitas (2) que satisfacen $x_1 = s_1, x_2 = s_2, \ldots, x_{k-1} = s_{k-1}$ y $x_k = s_k$.

A continuación se proporcionan algunos ejemplos de problemas de valor inicial.

Ejemplo 2.

1. $\begin{cases} x_n = 4x_{n-1} + 2^n & \forall n > 1 \\ x_1 = 4 \end{cases}$
2. $\begin{cases} x_n = x_{n-1} + x_{n-2} + 4 & \forall n > 2 \\ x_1 = 2 \\ x_2 = 10 \end{cases}$
3. $\begin{cases} x_n = 5x_{n-1} - 8x_{n-2} + 4x_{n-3} & \forall n > 3 \\ x_1 = 1 \\ x_2 = -8 \\ x_3 = 10 \end{cases}$

□

La primera cuestión que puede ser planteada de modo natural es, si dada una ecuación en diferencias finitas, cada problema de valor inicial asociado a dicha ecuación tiene una única solución o admite más de una en general. La respuesta a esta pregunta nos la proporciona el siguiente resultado cuya demostración puede ser encontrada en [4].

Teorema 1. *Sea una ecuación en diferencias finitas del tipo (2). Entonces cada problema de valor inicial asociado a (2) admite una única solución.* □

Existe una gran diversidad de ecuaciones en diferencias finitas, sin embargo nosotros, con el propósito de proporcionar herramientas que nos permitan determinar el comportamiento asintótico del coste computacional de un algoritmo de tipo Divide y Vencerás, nos centraremos en una familia concreta, las denominadas **ecuaciones en diferencias finitas lineales con coeficientes constantes** de orden 1. Este tipo de ecuaciones son aquellas para las que existen $\alpha \in \mathbb{R}_0^+$ y una sucesión $(y_n)_{n\in\mathbb{N}} \subset \mathbb{R}$ tales que la función Φ, en la expresión (2), verifica que:

$$\Phi(x_{n-1}, n) = \alpha x_{n-1} + y_n.$$

Así una ecuación en diferencias finitas lineal con coeficientes constantes de orden 1 es una ecuación en diferencias finitas que admite, para todo $n > 1$, la expresión general

$$x_n = \alpha x_{n-1} + y_n, \tag{3}$$

donde $(y_n)_{n\in\mathbb{N}} \subset \mathbb{R}$ y $\alpha \in \mathbb{R}_0^+$.

Un ejemplo de ecuación en diferencias finitas lineal con coeficientes constantes de orden 1 y un problema de valor inicial asociado a ella nos lo proporcionan el Ejemplo 1 y el Ejemplo 2, respectivamente.

Tras conocer, por el Teorema 1, que todo problema de valor inicial asociado a una ecuación en diferencias finitas lineal de orden 1 admite una única solución, tan sólo queda plantearse cómo puede ser obtenida dicha solución. El siguiente resultado nos proporciona una técnica para lograr ese fin (véase [4] y [1]).

Teorema 2. *Sean $p(n)$ un polinomio con $grado(p(n)) = d$, $\gamma, \beta \in \mathbb{R}$ y $\lambda \in \mathbb{R}_0^+$. El problema de valor inicial*

$$\begin{cases} x_n = \lambda x_{n-1} + \beta^n p(n) & \forall n > 1 \\ x_1 = \gamma \end{cases}$$

admite como única solución la sucesión $(x_n)_{n\in\mathbb{N}}$ cuyo término general viene dado por

$$x_n = \left(\frac{\gamma - \beta q(1)}{\lambda}\right)\lambda^n + \beta^n n^\delta q(n)$$

para todo $n \in \mathbb{N}$, siendo $q(n)$ un polinomio con $grado(q(n)) \leq d$ y

$$\delta = \begin{cases} 0 & si\ \beta \neq \lambda \\ 1 & si\ \beta = \lambda \end{cases}$$

□

Nota 1. Los coeficientes del polinomio $q(n)$ se obtienen exigiendo que la expresión $\beta^n n^\delta q(n)$ satisfaga la ecuación

$$x_n = \lambda x_{n-1} + \beta^n p(n).$$

El siguiente ejemplo clarifica el resultado anterior.

Ejemplo 3. Consideremos el problema de valor inicial siguiente:

$$\begin{cases} x_n = 4x_{n-1} + 2^n & \forall n > 1 \\ x_1 = 4 \end{cases}$$

La solución a dicho problema la obtendremos aplicando el Teorema 2 con $p(n) = 1 \ \forall n \in \mathbb{N}$, $\lambda = 4$, $\beta = 2$, $\gamma = 4$ y $\delta = 0$. Por tanto, el mencionado teorema nos garantiza que la solución al problema de valor inicial propuesto viene dada por la sucesión $(x_n)_{n\in\mathbb{N}}$ cuyo término general satisface la expresión

$$x_n = \left(\frac{4 - 2D}{4}\right)4^n + D2^n \ \forall n \in \mathbb{N},$$

donde $q(n) = D\ \forall n \in \mathbb{N}$. Para determinar el valor de la constante D debemos exigir, siguiendo la Nota 1, que la expresión $D2^n$ verifique la ecuación en diferencias

$$x_n = 4x_{n-1} + 2^n,$$

es decir exigiendo que

$$D2^n = 4D2^{n-1} + 2^n.$$

De esta última expresión se deduce inmediatamente que $D = -1$. En consecuencia se tiene que

$$x_n = \frac{3}{2}4^n - 2^n\ \forall n \in \mathbb{N}.$$

□

Finalizamos esta sección mediante el estudio de un problema de valor inicial que jugará un papel fundamental en el análisis asintótico de las relaciones de recurrencia de tipo (1) en la Sección III.

Sean $\gamma, \lambda, b \in \mathbb{R}$, $c \in \mathbb{N}$ $(c > 1)$ y $\alpha \in \mathbb{Z}^+$. Veamos como obtener, aplicando el Teorema 2, la solución del siguiente problema de valor inicial:

$$\begin{cases} x_n = \lambda x_{n-1} + b(c^\alpha)^n & \forall n > 1 \\ x_1 = \gamma \end{cases}. \tag{4}$$

Para ello distinguiremos dos casos posibles y tendremos en cuenta que $p(n) = b\ \forall n \in \mathbb{N}$:

Caso 1. $c^\alpha \neq \lambda$. En este caso $\delta = 0$ y, así, la expresión de la solución del problema (4) que nos proporciona el Teorema 2 viene dada por

$$x_n = \left(\frac{\gamma - c^\alpha D}{\lambda}\right)\lambda^n + D\,(c^\alpha)^n$$

(obsérvese que $q(n) = D\ \forall n \in \mathbb{N}$). Exigiendo, tal y como nos indica la Nota 1, que la expresión $D\,(c^\alpha)^n$ verifique la ecuación en diferencias que induce el problema bajo estudio se obtiene que

$$D = \frac{bc^\alpha}{c^\alpha - \lambda}.$$

De este modo la expresión de la solución se reduce a

$$x_n = \left(\frac{\gamma - \frac{bc^{2\alpha}}{c^\alpha - \lambda}}{\lambda}\right)\lambda^n + \frac{bc^\alpha}{c^\alpha - \lambda}\,(c^\alpha)^n.$$

Caso 2. $c^\alpha = \lambda$. En este caso $\delta = 1$ y, así, la expresión de la solución del problema (4) que nos proporciona el Teorema 2 viene dada por

$$\begin{aligned} x_n &= \left(\tfrac{\gamma - c^\alpha D}{\lambda}\right)\lambda^n + Dn\,(c^\alpha)^n \\ &= \left(\tfrac{\gamma - c^\alpha D}{c^\alpha}\right)(c^\alpha)^n + Dn\,(c^\alpha)^n \\ &= \left[\left(\tfrac{\gamma}{c^\alpha} - D\right) + Dn\right](c^\alpha)^n \end{aligned}$$

(nótese que $q(n) = D\ \forall n \in \mathbb{N}$). Exigiendo, tal y como nos indica la Nota 1, que la expresión $Dn\,(c^\alpha)^n$ verifique la ecuación en diferencias que induce el problema bajo estudio se obtiene que $D = b$. De este modo la expresión de la solución se reduce a

$$x_n = \left[\left(\frac{\gamma}{c^\alpha} - b\right) + bn\right](c^\alpha)^n$$

III. ANÁLISIS DE COMPLEJIDAD ALGORÍTMICA Y ECUACIONES EN DIFERENCIAS FINITAS

En lo que sigue nos centraremos en obtener el comportamiento asintótico de la solución de la relación de recurrencia (1). Para ello aplicaremos la técnica de resolución de problemas de valor inicial inducidos por ecuaciones en diferencias finitas lineales con coeficientes constantes de orden 1 descrita en el Teorema 2.

En primer lugar debemos observar el hecho de que la relación de recurrencia (1) no satisface la expresión (2) y, así, la definición de ecuación en diferencias. Por este motivo es necesario manipular la expresión (1), realizando un cambio de variable, para poder aplicar la técnica de resolución expuesta en la Sección II.

En lo que sigue consideraremos la relación de recurrencia

$$T(n) = \begin{cases} bn^\alpha & \text{si } 1 \leq n < c \\ aT(\frac{n}{c}) + bn^\alpha & \text{si } n \in \mathbb{N}_c \end{cases}, \tag{5}$$

donde $a, b \in \mathbb{R}_0^+$, $\alpha \in \mathbb{Z}^+$, $c \in \mathbb{N}$ con $c > 1$ y $\mathbb{N}_c = \{c^k : k \in \mathbb{N}\}$.

Obsérvese que en la relación de recurrencia (5) se ha considerado que el tamaño de los datos de entrada n recorre el conjunto $\mathbb{N}_c$, esto es debido a que el comportamiento asintótico en el caso más general, en el que $n \in \mathbb{N}$, puede ser analizado utilizando la información deducida de este caso particular. En primer lugar estudiaremos la relación de recurrencia (5) y posteriormente abordaremos el análisis asintótico de la relación de recurrencia (1).

Con el objetivo de abordar el estudio de la relación de recurrencia (5), y teniendo en cuenta que $n \in \mathbb{N}_c$, realizamos la siguiente identificación para todo $k \in \mathbb{N}$:

$$x_k = T(c^k).$$

De donde se tiene que

$$T(\frac{n}{c}) = T(c^{k-1}) = x_{k-1}.$$

A continuación planteamos el siguiente problema de valor inicial:

$$\begin{cases} x_k = ax_{k-1} + b(c^\alpha)^k & \forall n > 1 \\ x_1 = ab + bc^\alpha \end{cases}. \tag{6}$$

Nótese que $ab + bc^\alpha = T(c)$.

Obervemos que el problema de valor inicial precedente ha sido estudiado en la Sección II con $\lambda = a$ y $\gamma = ab + bc^\alpha$. Siguiendo el esquema de resolución planteado en la sección mencionada se obtienen dos casos posibles que nos conducen a la expresión de la solución:

Caso 1. $c^\alpha \neq a$. En este caso la expresión de la solución del problema (6) viene dada por

$$\begin{aligned} x_k &= \left(\frac{ab+bc^\alpha-\frac{c^{2\alpha}b}{c^\alpha-a}}{a}\right)a^k+\frac{bc^\alpha}{c^\alpha-a}\left(c^\alpha\right)^k \\ &= \frac{ab}{a-c^\alpha}a^k+\frac{bc^\alpha}{c^\alpha-a}\left(c^\alpha\right)^k. \end{aligned}$$

Caso 2. $c^\alpha = a$. En este caso la expresión de la solución del problema (6) viene dada por

$$\begin{aligned} x_k &= \left(\frac{ab+bc^\alpha}{c^\alpha}-b+bk\right)\left(c^\alpha\right)^k \\ &= \left(\frac{ab}{c^\alpha}+bk\right)\left(c^\alpha\right)^k. \end{aligned}$$

Una vez resuelto el problema de valor inicial (6), se obtiene inmediatamente la solución de la relación de recurrencia (5). En efecto, teniendo en cuenta que $n \in \mathbb{N}_c$ ($k = \log_c(n)$) y que $x_k = T(c^k)$ se tiene lo siguiente:

Caso 1. ($c^\alpha \neq a$). En este caso la solución se corresponde con la expresión

$$T(n) = \begin{cases} bn^\alpha & \text{si } 1 \leq n < c \\ \frac{ab}{a-c^\alpha}a^{\log_c(n)}+\frac{bc^\alpha}{c^\alpha-a}n^\alpha & \text{si } n \in \mathbb{N}_c \end{cases}. \quad (7)$$

Nótese que $a^{\log_c(n)} = n^{\log_c(a)}$ y, por tanto, la expresión (7) puede ser reescrita del modo siguiente:

$$T(n) = \begin{cases} bn^\alpha & \text{si } 1 \leq n < c \\ \frac{ab}{a-c^\alpha}n^{\log_c(a)}+\frac{bc^\alpha}{c^\alpha-a}n^\alpha & \text{si } n \in \mathbb{N}_c \end{cases}. \quad (8)$$

Caso 2. ($c^\alpha = a$). En este caso la solución se corresponde con la expresión

$$T(n) = \begin{cases} bn^\alpha & \text{si } 1 \leq n < c \\ \frac{ab}{c^\alpha}n^\alpha+bn^\alpha\log_c(n) & \text{si } n \in \mathbb{N}_c \end{cases}. \quad (9)$$

Tras resolver la relación de recurrencia (5) estamos en condiciones de plantearnos el estudio de su comportamiento asintótico. Para ello, necesitaremos introducir la notación siguiente: dadas dos funciones $f, g : \mathbb{N} \to [0,\infty)$ y $c \in \mathbb{N}$ con $c > 1$, diremos que $f \in \Theta_{\mathbb{N}_c}(g)$ si existen $c_1, c_2 \in \mathbb{R}^+$ y $n_0 \in \mathbb{N}$ tales que

$$c_2 g(n) \leq f(n) \leq c_1 g(n)$$

para todo $n \in \mathbb{N}_c$ con $n \geq n_0$.

Teniendo en cuenta que

$$f \in \Theta_{\mathbb{N}_c}(g) \Leftrightarrow 0 < \lim_{n\to\infty}\frac{f(n)}{g(n)} < \infty,$$

el estudio asintótico se reduce a los discusión de los siguientes casos posibles:

Caso 1. ($c^\alpha \neq a$). En este caso debemos, a su vez, distinguir dos nuevos casos:

Caso 1.1. $c^\alpha > a$. En este caso, de la expresión (8) y del hecho de que $\alpha > \log_c(a)$ se deduce que

$$\lim_{n\to\infty}\frac{T(n)}{n^\alpha} = \frac{bc^\alpha}{c^\alpha-a} > 0.$$

Con lo cual se tiene que

$$T \in \Theta_{\mathbb{N}_c}(n^\alpha).$$

Caso 1.2 $c^\alpha < a$. En este caso, de la expresión (8) y del hecho de que $\alpha < \log_c(a)$ se deduce que

$$\lim_{n\to\infty}\frac{T(n)}{n^{\log_c(a)}} = \frac{ab}{a-c^\alpha} > 0.$$

Con lo cual se tiene que

$$T \in \Theta_{\mathbb{N}_c}(n^{\log_c(a)}).$$

Caso 2. $a = c^\alpha$. En este caso, de la expresión (9) y del hecho de que $\log_c(n) \geq 1$ se deduce que

$$\lim_{n\to\infty}\frac{T(n)}{n^\alpha\log_c(n)} = b > 0.$$

En consecuencia $T \in \Theta_{\mathbb{N}_c}(n^\alpha\log_c(n))$.

El análisis realizado del comportamiento asintótico de la relación de recurrencia (5) nos va a permitir estudiar dicho comportamiento para la relación de recurrencia general (1). Para este fin será necesario emplear los siguientes conceptos.

Dada una función $f : \mathbb{N} \to [0,\infty)$, diremos que es **asintóticamente no decreciente** si existe $n_0 \in \mathbb{N}$ tal que para todo $n \geq n_0$ se verifica $f(n) \leq f(n+1)$. Además, una función asintóticamente no decreciente f es **r-armónica** ($r \in \mathbb{N}$ con $r \geq 2$) si la función $f_r : \mathbb{N} \to [0,\infty)$ satisface la condición $f_r \in \Theta(f)$, donde $f_r(n) = f(rn)$.

Obsérvese que toda función verificando la relación de recurrencia (1) es asintóticamente no decreciente.

La conexión entre los dos últimos conceptos introducidos y el análisis asintótico nos la proporciona el siguiente resultado (véase [1]), el cual será clave para poder culminar nuestro estudio.

Proposición 1. *Sea $r \in \mathbb{N}$ con $r \geq 2$. Si $g : \mathbb{N} \to [0,\infty)$ es una función r-armónica y $f : \mathbb{N} \to [0,\infty)$ es una función asintóticamente no decreciente tal que $f \in \Theta_{\mathbb{N}_r}(g)$, entonces $f \in \Theta(g)$.* □

En virtud de la proposición anterior se concluye nuestro estudio mediante la siguiente discusión:

Caso 1. $c^\alpha > a$. En este caso se tiene que $T \in \Theta_{\mathbb{N}_c}(n^\alpha)$. Del hecho de que la función n^α es c-armónica concluimos, por la Proposición 1, que $T \in \Theta(n^\alpha)$.

Caso 2. $c^\alpha < a$. En este caso se tiene que $T \in \Theta_{\mathbb{N}_c}(n^{\log_c(a)})$. Como la función $n^{\log_c(a)}$ es c-armónica concluimos, por la Proposición 1, que $T \in \Theta(n^{\log_c(a)})$.

Caso 3. $a = c^\alpha$. En este caso se tiene que $T \in \Theta_{\mathbb{N}_c}(n^\alpha \log_c(n))$. Puesto que la función $n^\alpha \log_c(n)$ es c-armónica concluimos, por la Proposición 1, que $T \in \Theta(n^\alpha \log_c(n))$.

IV. Conclusiones

En Ciencia de la Computación se requiere determinar matemáticamente el coste de ejecución de los algoritmos que se emplean para resolver los problemas para los que han sido diseñados. Para poder llevar a cabo dicho objetivo, el coste de computación de un algoritmo es denotado por una función $T : \mathbb{N} \to [0, \infty)$ tal que $T(n)$ representa el coste empleado por el algoritmo en resolver el problema bajo consideración cuando el dato de entrada del algoritmo tiene un tamaño n. En general, dado un algoritmo, determinar exactamente la expresión de la función que representa su coste computacional es una tarea ardua y, por ese motivo, en la práctica tan sólo se requiere obtener una información aproximada del coste mediante la acotación de los valores $T(n)$ para cada tamaño de dato de entrada n. Dichas cotas pueden ser obtenidas mediante técnicas matemáticas de naturaleza muy distinta. En particular, en este trabajo hemos mostrado que la teoría de ecuaciones en diferencias finitas constituye una herramienta útil para ese fin en aquellos casos en los que el coste computacional del algoritmo satisface una relación de recurrencia. Concretamente se han aplicado las técnicas de resolución de problemas de valor inicial asociados a ecuaciones en diferencias finitas lineales con coeficientes constantes para analizar el comportamiento asintótico de algoritmos recursivos que siguen estrategias de tipo Divide y Vencerás. Los resultados asintóticos obtenidos para este tipo de algoritmos coinciden con los que se obtienen mediante el uso de otros métodos conocidos y ampliamente extendidos en la literatura, como son el método de substitución, el método del árbol de recursión y el método maestro (véase [3]). Sin embargo, las técnicas basadas en el uso de las ecuaciones en diferencias presentan la ventaja, con respecto a las ya mencionadas, de permitir llevar a cabo un razonamiento estructurado y mecanizado al analizar la complejidad algorítmica que, además, destaca por su simplicidad y elegancia.

Referencias

[1] G. Brassard, P. Bratley, *Algorítmica: Concepción y Análisis*, Editorial Masson, Barcelona, 1990.
[2] G. Brassard, P. Bratley, *Fundamentos de algoritmia*, Prentice-Hall, Madrid, 1997.
[3] T.H. Cornen, C.E. Leiserson, L.R. Rivest, C. Stein, *Introduction to Algorithms*, MIT Press, Massachusetts, 2001.
[4] P. Cull, M. Flahive, R. Robson, *Difference Equations: From Rabbits to Chaos*, Springer, New York, 2005.