El almacenaje de información numérica en una computadora digital

José Guerrero Grajeda

Facultad de Ciencias, UNAM

I. Presentación

Los sistemas numéricos con los que opera una computadora digital se conocen como sistemas de punto flotante, y su conocimiento constituye el punto de partida de todo estudio serio del trabajo numérico. Sin embargo, en los textos básicos sobre este tema casi nunca se trata la cuestión de cómo se almacenan los datos numéricos en la computadora.

Dado que en ocasiones este conocimiento resulta importante para quienes se dedican a las matemáticas computacionales, nos pareció pertinente presentar el asunto con cierta amplitud.

En nuestro tratamiento, adoptaremos como modelo un sistema de cómputo al que llamaremos B78, bastante aproximado a alguno de la realidad.

II. Preliminares

II.1. Representación en punto flotante

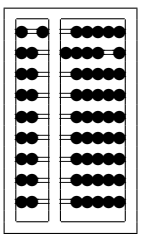
Sabemos que a cada punto $x \in R$ le está asociada una expresión decimal de la forma:

$$x = (\pm).d_1 d_2 \cdots d_t d_{t+1} \cdots \times 10^{e_x}, \tag{1}$$

con $d_1 \neq 0$; $\quad 0 \leq d_i \leq 9$, $i = 2, 3, \ldots$; $\quad e_x \in Z$.

También hemos visto que, en la práctica, en lugar de trabajar con la "cola" infinita de dígitos asociada a x, se trabaja sólo con un número finito de ellos, los cuales se obtienen de (1) aplicando la función:

$$Ft(x) = (\pm).d_1 d_2 \cdots d_t \times 10^{e_x}. \tag{2}$$



Para lo que aquí vamos a desarrollar es necesario que tengamos presentes las siguientes consideraciones adicionales:

1) El estudio sobre los sistemas numéricos de punto flotante se conserva sin modificaciones sustanciales si en lugar de trabajar en base 10 se trabaja en una base distinta. De hecho, en la práctica, las bases usuales son: $\beta = 2$, $\beta = 8$ y $\beta = 16$.

2) En función de limitaciones físicas, resulta necesario acotar el exponente que aparece en la expresión (2) para el número x redondeado a t dígitos. Esto es, debemos imponer la condición:

$$m \leq e_x \leq M.$$

Hechas estas aclaraciones, podemos definir un sistema de punto flotante $Fl(R)$ como:

$$Fl(R) = \left\{ y \in R \middle| y = (\pm)\Big(\sum_{k=1}^{t} \frac{d_k}{\beta^k}\Big) \times \beta^{e_x}; \qquad d_1 \neq 0; \right.$$
$$\left. 0 \leq d_i \leq \beta - 1, i = 2, \ldots, t; \quad m \leq e_y \leq M; \quad \text{o} \quad y = 0 \right\}.$$

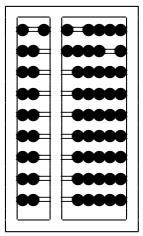
Es claro que si $y \in Fl(R)$, entonces,

$$y = (\pm).d_1 d_2 \ldots d_t \times \beta^{e_y}.$$

Terminaremos con esta cuestión añadiendo que, para el caso de la computadora B78 se tiene que:

$$t = 13 = (15)_8,$$
$$\beta = 8 = (10)_8.$$

II.2. La unidad de almacenamiento



Para nuestros propósitos, consideraremos que se trata de un dispositivo compuesto —entre otras cosas— de múltiples unidades capaces cada una de ellas de almacenar únicamente un dígito binario, esto es, un elemento del conjunto $\{0, 1\}$, y a esta información mínima se le llamará un *bit*.

Lo anterior nos da una posibilidad de entrar ya a una primera etapa de lo que será nuestro análisis, que se refiere a la presentación de una de las formas fundamentales relacionadas con el almacenaje de información: la

palabra, misma que está integrada por un conjunto de *bits* (48 en el caso de nuestra computadora), que a su vez se reagrupan de distintas formas, dependiendo del código de representación que se esté usando para los datos.

En cuanto a nuestro sistema B78, consideraremos varios códigos, de los cuales el más usual es el EBCDIC (*Extended Binary Coded Decimal Interchange Code*) que, como todo código, establece reglas para la traducción de un alfabeto a otro.

A los subgrupos de *bits* organizados en una palabra con relación a cierto código se les llama *bytes*, y cada *byte* constituye un caracter respecto al código en cuestión. Los *bytes* para el código EBCDIC y el sistema B78 están cmpuestos por 8 *bits*, con lo cual se tiene que una palabra EBCDIC consta de 6 caracteres (48 *bits*, 6 *bytes*). Una representación usual de esto es la siguiente:

caracter 0		1		2		3		4		caracter 5	
47	43			·	·	·				7	3
46	42			·	·	·				6	2
45	41			·	·	·				5	1
44	40			·	·	·				4	0

Palabra EBCDIC

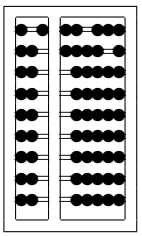
III. Sobre el almacenaje de los elementos de $Fl(R)$

III.1. Generalidades

De lo visto en la primera parte tenemos que, para el caso de la B78, todo número $y \in Fl(R)$ tiene la forma:

$$y = (\pm).d_1 d_2 \cdots d_{13} \times \beta^{e_y},$$

con $d_1 \neq 0$; $\quad 0 \leq d_i \leq \beta - 1$, $i = 2, \ldots, 13$; $\quad \beta = 8 = (10)_8$.



Para lo que sigue, tomaremos también en consideración que: dado $y \in Fl(R)$, éste es almacenado en la computadora haciendo uso de una palabra de memoria. Su distribución dentro de ésta es aproximadamente como sigue:

47	43	39	35	31	27	23	19	15	11	7	3
46	e_y 42	38	d_2 34	30	26	22	18	14	10	6	2
45	41	d_1 37	33	29	25	21	17	13	9	5	d_{13} 1
44	40	36	32	28	24	20	16	12	8	4	0

De la figura anterior se tiene que los *bits* 0–38 son usados para almacenar la mantisa y los *bits* 39–44 para almacenar el exponente. En cuanto a los signos se tiene que:

1) El *bit* 45 es usado para almacenar el signo del exponente, según la regla: $0 = +$, $1 = -$.
2) El *bit* 46 se usa análogamente para almacenar el signo de la mantisa.

Otra observación importante es que cuando una variable se halla almacenada en la forma anterior, el punto decimal se toma corrido hasta la derecha del último dígito (d_{13}).

Finalmente, una cuestión de la mayor importancia es la referente a que el contenido de una palabra en la que se ha almacenado una variable $y \in Fl(R)$ puede verse como una cadena entera octal (entero octal) de la forma:

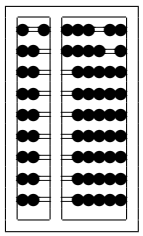
$$y = D_1 D_2 \cdots D_{16} \quad \text{(16 dígitos octales)},$$

donde en D_1 se tiene la información sobre el signo del número, en D_2 y D_3 se nos informa sobre el valor del exponente y en $D_4 \to D_{16}$ se tiene la mantisa. Veamos algunos ejemplos:

Ejemplo 1. Dar la representación interna del número 10.

De lo que hemos visto, la representación es:

0	0	0	0	0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	0	0	0	0	0



Escrito en forma de cadena queda como: 0000000000000012.

Análogamente, para el caso del número -10 se tiene:

0	0	0	0	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	0	0	0	0	0

o bien: 2000000000000012.

Como se observa del ejemplo anterior, en el caso de números enteros, éstos son almacenados en los *bits* correspondientes a los últimos dígitos, de manera que si una vez obtenida la cadena octal de un número, nos interesa conocer su forma normalizada, lo único que hay que tomar en cuenta es el signo (almacenado en D_1) y el punto (que como ya se ha dicho, siempre se considerará situado hasta la derecha del número). Por ejemplo, la forma normalizada de 0000000000000012 es $(.12 \times 10^2)_8$; y y la correspondiente a 2000000000000012 es $-(.12 \times 10^2)_8$.

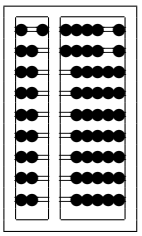
La situación cambia un poco cuando se trata de números reales. En este caso, el número se almacena a partir de los *bits* correspondientes al primer dígito de la mantisa. Así por ejemplo, la representación interna del número $(1.25)_{10} = (1.2)_8$ es como sigue:

0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	1	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0	0	0

o bien 1141200000000000, que puede expresarse en forma normalizada como:

$$-(.12 \times 10^{15-14})_8 = -(.12 \times 10^1)_8.$$

III.2. Cálculo de algunos valores significativos de $Fl(R)$



Comenzaremos esta parte con el cálculo de E_n (EPSMAQ), que se define como:

$$E_n = S_n(\eta_n),$$

el "siguiente" de η_n en $Fl(R)$ con

$$\eta_n = \sup\{y \in Fn(R) \mid 1 \oplus y = 1\},$$

donde $\oplus$ representa la suma en $Fl(R)$; esto es, se tiene que E_n es el número más pequeño tal que:

$$1 \oplus E_n > 1,$$

o bien

$$1 \oplus E_n = S(1).$$

Debido al contexto en el que hemos trabajado en esta parte, usaremos E_t en lugar de E_n para simbolizar a EPSMAQ.

Tenemos entonces que para el caso de la B78, las representaciones para 1 y $S(1)$ son:

$$1 \rightarrow 1141000000000000$$

y

$$S(1) \rightarrow 1141000000000001,$$

de donde:

$$S(1) - 1 = 1301000000000000,$$

lo cual nos da finalmente:

$$E_t = \frac{S(1) - 1}{2} = 1314000000000000,$$

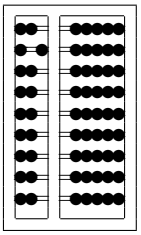
o bien, en forma normalizada,

$$E_t = (.4 \times 10^{-14})_8.$$

Nota: El hecho de haber tomado $E_t = [S(1) - 1]/2$ se debe a que la B78 redondea tomando en consideración el dígito siguiente (catorceavo en este caso) a aquél en el que trunca la expresión decimal del número que va ha ser redondeado.

Gráficamente, se tiene para E_t,

0	0	1	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	0	0



Procederemos ahora al cálculo del número más pequeño (en valor absoluto) representable en la B78.

Es claro que tal número, al que daremos el nombre de ETA, deberá construirse a partir del menor exponente y la menor mantisa, por lo que gráficamente deberá tener la siguiente configuración:

0	1	1	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0	0
1	1	1	0	0	0	0	0	0	0	0	0

En forma de cadena octal se tiene:

$$\text{ETA} = 1771000000000000,$$

o bien,

$$\text{ETA} = (.1 \times 10^{-62})_8.$$

Para finalizar, tenemos que el número más grande (en valor absoluto) que es posible almacenar en una estructura como la que aquí hemos presentado, es aquél cuya configuración interna es como sigue:

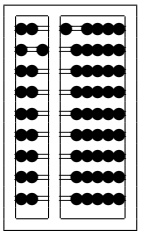
0	1	1	1	1	1	1	1	1	1	1	1
0	1	1	1	1	1	1	1	1	1	1	1
0	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1

que, como entero octal puede escribirse:

$$0777777777777777,$$

o bien, en forma normalizada,

$$(.7777777777777 \times 10^{77})_8.$$



De estos últimos cálculos se tiene que las cotas m y M para el exponente e_y —de los que ya hemos hablado antes— son:

$$m = -(62)_8 = -(50))_{10},$$
$$M = (77)_8 = (63)_{10}.$$