TEMA 1

REPRESENTACIÓN Y COMUNICACIÓN DE LA INFORMACIÓN

GUIÓN-ÍNDICE

1. INTRODUCCIÓN

Un ordenador es una máquina que procesa información. Más concretamente, la ejecución de un programa implica realizar unos tratamientos, según especifica un conjunto ordenado de instrucciones, con unos datos. Para que el ordenador ejecute un programa es necesario darle dos tipos de informaciones: las instrucciones que forman el programa y los datos con los que debe operar ese programa.

Dos de los aspectos más importantes que se presentan en Informática relacionados con la información son cómo representarla y cómo registrarla físicamente. Los dos problemas están íntimamente relacionados y el primero de ellos es el que trataremos en este tema. El segundo se resuelve con soportes de información: discos, cintas magnéticas, etc.

Normalmente, la información se da al ordenador en la forma usual escrita que utilizan los seres humanos; es decir, con la ayuda de un alfabeto o conjunto de símbolos, que denominaremos caracteres.

Los caracteres que constituyen el alfabeto suelen agruparse en cinco categorías:

Caracteres alfabéticos: Son las letras mayúsculas y minúsculas del abecedario inglés.

A, B, C, D, ..., X, Y, Z, a, b, c, d, ..., x, y, z

Caracteres numéricos: Están constituidos por las diez cifras decimales:

0, 1, 2, 3, 4, 5, 6, 7, 8, 9

Caracteres especiales: Son los símbolos no incluidos en los grupos anteriores, como los siguientes:

) (, * / ; : + Ñ ñ = ! ? . “ & > # < [@]

Caracteres de control: Representan órdenes de control, como el carácter indicador de fin de línea. Muchos de ellos son generados e insertados por el propio ordenador.

Caracteres gráficos: Son los símbolos con los que se pueden representar figuras elementales.

Nos centraremos en los tres primeros tipos de caracteres, a veces denominados **caracteres-texto**. A la agrupación de los dos primeros se le denomina conjunto de **caracteres alfanuméricos**.

Toda comunicación con el ordenador se ha de realizar según los caracteres que admitan sus dispositivos de entrada/salida (todo dato o instrucción se representará con el alfabeto definido para el sistema informático). El conjunto de caracteres codificable en un ordenador se denomina **juego de caracteres** de dicho ordenador.

El diseño y construcción de un ordenador se simplifica, aumentando su fiabilidad, si se utilizan sólo dos valores posibles para las variables físicas que representan los datos en el interior de la computadora. Estos valores se representan por 0 y 1, y corresponden a dos niveles de tensión, de corriente, etc. En definitiva, la información se mantiene utilizando dos valores de una magnitud física (bit) representable mediante ceros y unos.

Al tener que traducir toda la información suministrada por el ordenador a ceros y unos, es necesario establecer una **correspondencia** entre el conjunto de todos los caracteres y el conjunto binario $\{0, 1\}^n$ (n es el número de bits disponibles para representar los caracteres). En definitiva, es necesario llevar a cabo una codificación entre los elementos del primer conjunto mediante los del segundo. Estos códigos de transformación se denominan **códigos de E/S** y pueden definirse de forma arbitraria, aunque existen códigos de E/S normalizados.

Ejemplo, si queremos representar las cinco primeras letras mayúsculas del alfabeto, podemos establecer la siguiente correspondencia:

Carácter	Código
A	100
B	011
C	110
D	111
E	101

Como se ve en este ejemplo, se ha establecido una correspondencia entre el conjunto {A, B, C, D, E} y el conjunto $\{0, 1\}^3$. Aquí hemos utilizado el número mínimo de bits necesarios para representar estas 5 letras, es decir, tres (en este caso, n debe ser al menos 3).

Para realizar las operaciones aritméticas sobre datos numéricos, el propio ordenador efectúa una transformación de la representación en códigos de E/S a una representación basada en el sistema de numeración en base 2, que resulta más adecuada (una representación numérica posicional es muy apta para realizar operaciones aritméticas).

2. SISTEMAS DE NUMERACIÓN USUALES EN INFORMÁTICA

Los ordenadores suelen efectuar las operaciones aritméticas utilizando una representación para los datos numéricos basada en el **sistema de numeración en base dos** (*binario natural*). También se utilizan como **códigos intermedios** los sistemas octal y hexadecimal, ya que, el paso de un número en binario a estos sistemas es trivial, y se obtiene una representación que está más cerca del sistema decimal (se utilizan como paso intermedio en las conversiones binario-decimal y decimal-binario).

2.1. Representación posicional de los números

Un **sistema de numeración en base b** utiliza para representar los números un alfabeto compuesto por b símbolos o cifras. Así, todo número se expresa por un conjunto de cifras, contribuyendo cada una de ellas con un valor que depende de la cifra en sí y de la posición que ocupe dentro del número.

En el sistema de numeración decimal (o en base 10), se tiene b = 10 y el alfabeto está constituido por diez símbolos, denominados también cifras decimales {0, 1, 2, ..., 9}.

Ejemplo, se verifica que:

$$458.23)_{10} = 4*10^2 + 5*10^1 + 8*10^0 + 2*10^{-1} + 3*10^{-2}$$

Generalizando, si la expresión de un número N en base b es:

$$N)_b = \ldots n_3\ n_2\ n_1\ n_0\ n_{-1}\ n_{-2}\ n_{-3} \ldots$$

Entonces, su valor decimal viene dado por:

$$N)_{10} = \ldots + n_3 * b^3 + n_2 * b^2 + n_1 * b^1 + n_0 * b^0 + n_{-1} * b^{-1} + n_{-2} * b^{-2} + n_{-3} * b^{-3} + \ldots$$

Este resultado, conocido como el **teorema fundamental de la numeración**, relaciona una cantidad expresada en cualquier sistema de numeración con la misma cantidad expresada en el sistema decimal.

Ejemplo: El código octal está basado en un alfabeto de 8 símbolos (A = {0, 1, 2, 3, 4, 5, 6, 7} y b = 8). El número octal $165,4)_8$ tiene una representación decimal que viene dada por:

$$165,4)_8 = 1*8^2 + 6*8^1 + 5*8^0 + 4*8^{-1})_{10} = 117,5)_{10}$$

2.2. Sistema de numeración en base dos

Las operaciones aritméticas dentro de un ordenador se suelen llevar a cabo utilizando una representación para los datos basada en el código binario natural. Aunque el cambio de código de E/S a la representación en binario natural la realiza automáticamente el ordenador, veremos unas cuestiones relativas al sistema binario y a las transformaciones entre éste y el sistema decimal.

Definición de sistema binario

Se basa en un alfabeto de sólo dos símbolos, así que A = {0, 1} y b = 2. Los elementos de este alfabeto se denominan *cifras binarias* o *bits.*

Transformaciones entre bases binaria y decimal

Para transformar un número representado en el sistema binario a su representación en decimal, únicamente hay que aplicar el teorema fundamental de la numeración, visto en el apartado anterior.

Ejemplo:

$$110010,101)_2 = 1*2^5 + 1*2^4 + 0*2^3 + 0*2^2 + 1*2^1 + 1*2^0 + 1*2^{-1} + 0*2^{-2} + 1*2^{-3})_{10} = 50,625)_{10}$$

Transformaciones entre bases decimal y binaria

Para transformar un número de decimal a binario, se transforman de forma independiente la parte entera y la parte fraccionaria, y después se concatenan los resultados obtenidos.

Conversión de la parte entera: Basta con dividir por 2 la parte entera del número decimal, y después realizar divisiones por 2 de los cocientes sucesivos, hasta llegar a un cociente menor que dos. El número binario obtenido como resultado tendrá como bit más a la izquierda el último cociente obtenido en el proceso de divisiones sucesivas. El resto de los bits del número binario estará formado por los restos de las divisiones, comenzando por los últimos obtenidos.

Conversión de la parte fraccionaria: Basta con multiplicar por 2 la parte fraccionaria del número decimal de partida y después realizar la misma operación en cadena con las partes fraccionarias de los resultados obtenidos en los productos sucesivos. El proceso finaliza cuando se llega a un resultado con parte fraccionaria nula (suponiendo que el número binario tiene una parte fraccionaria finita). La parte fraccionaria en binario se forma con las partes enteras de los productos obtenidos, leídos en el mismo orden en que se obtienen.

Ejemplo, pasar el número decimal $67{,}125)_{10}$ a binario.
Primero cogemos la parte entera del número decimal (67) y la dividimos sucesivamente por 2:

67/2 = 33 (r=1) /2 = 16 (r=1) /2 = 8 (r=0) /2 = 4 (r=0) /2 = 2 (r=0) /2 = 1 (r=0)

Con lo que la parte entera del número decimal tiene como representación en binario el número 1000011.
Para representar el resto del número, se toma la parte fraccionaria del número decimal y se multiplica sucesivamente por 2:

0’125 * 2 = 0,250 * 2 = 0,50 * 2 = 1,0

Por tanto, esta parte fraccionaria 0,125 tiene como representación en binario al número 0,001.
El número binario completo será:

$67{,}125)_{10} = 1000011{,}001)_2$

2.3. Operaciones aritméticas binarias

Las operaciones aritméticas básicas son la suma, resta, multiplicación y división. Estas operaciones resultan análogas a las realizadas en decimal pero basándonos en las tablas siguientes:

		SUMA	RESTA	PRODUCTO	DIVISIÓN
A	**B**	**A+B**	**A-B**	**A*B**	**A/B**
0	0	0	0	0	indeterminado
0	1	1	1 y debo 1	0	0
1	0	1	1	0	
1	1	0 y llevo 1	0	1	1

Multiplicar por $10)_2$ (es decir, por 2 en decimal) es equivalente a añadir un cero a la derecha, siendo esto similar a multiplicar por $10)_{10}$ a un número decimal. En realidad, esto ocurre en todos los sistemas de numeración cuando se multiplica un número por su base. De la misma forma, dividir por $2)_{10} = 10)_2$ se hace desplazando el punto decimal a la izquierda, o eliminando ceros a la derecha.

2.4. Códigos intermedios

Su uso se fundamenta en la facilidad de transformar un número en base 2 a otra base potencia de 2 y viceversa. Usualmente se utilizan como códigos intermedios los sistemas de numeración en base 8 (u octal) y en base 16 (o hexadecimal).

2.4.1. Base octal

Se tiene que b= 8 y el conjunto de símbolos utilizado es: {0, 1, 2, 3, 4, 5, 6, 7}.

Las conversiones octal-binario y binario-octal pueden hacerse fácilmente. El **paso de binario a octal** se efectúa formando grupos de tres cifras ($b=8=2^3$) a derecha e izquierda del punto decimal y efectuando la conversión a octal de cada grupo individual (basta con memorizar el equivalente binario de cada cifra octal). Para **pasar de octal a binario**, cada cifra octal se convierte individualmente a binario manteniendo el orden del número original.

Para **pasar de octal a decimal**, basta con aplicar el teorema fundamental de la numeración. Para **pasar de decimal a octal**, se pasan independientemente la parte entera (realizando divisiones sucesivas entre 8) y la parte fraccionaria (realizando sucesivas multiplicaciones por 8) y después se concatena el resultado (de forma similar al caso binario).

Ejemplo: Pasar el número octal $17352,16)_8$ a binario.

Se toma cada dígito octal y se sustituye por la expresión binaria correspondiente:

Octal:	1	7	3	5	2	,1	6
Binario:	001	111	011	101	010	,001	110

Con esto nos queda:

$$001111011101010,001110)_2$$

(los ceros a la izquierda y a la derecha se pueden eliminar)

Este código se utiliza cuando el número de bits a representar es múltiplo de 3, para aprovechar mejor la capacidad de almacenamiento.

2.4.2. Base hexadecimal

Se tiene que b= 16 y es necesario disponer de un alfabeto de 16 símbolos:

{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F}

Las **conversiones hexadecimal-binario y binario-hexadecimal** se realizan prácticamente de la misma manera que en el caso octal. La única diferencia estriba en que las agrupaciones deben ser en este caso de 4 bits, en lugar de grupos de 3 bits.

Las conversiones entre el sistema decimal y hexadecimal se realizan igual que en el sistema binario, sin más diferencia que tener en cuenta que, en este caso, la base es 16. Para **pasar de hexadecimal a decimal**, basta con aplicar el teorema fundamental de la numeración. Para **pasar de decimal a hexadecimal**, se pasan independientemente la parte entera (realizando sucesivas divisiones entre 16) y la parte fraccionaria (realizando sucesivas multiplicaciones por 16) y después se concatena el resultado.

Ejemplo: Pasar a binario el número hexadecimal $7B3,F)_{16}$.

Se toma cada dígito hexadecimal y se sustituye por la representación binaria correspondiente:

Hexadecimal:	7	B	3	,F
Binario:	0111	1011	0011	,1111

Con esto nos queda:

$$011110110011,1111)_2$$

(los ceros a la izquierda y a la derecha se pueden eliminar)

Este código se utiliza cuando el número de bits a representar es múltiplo de 4.

3. CÓDIGOS DE ENTRADA/SALIDA

Los códigos de entrada/salida (E/S) o códigos externos son códigos que asocian a cada carácter (alfabético, numérico o especial) una determinada combinación de bits. En otras palabras, un código de E/S es una correspondencia entre los conjuntos:

$$\{0, 1, 2, ..., 9, A, B, ..., Y, Z, a, b, ..., y, z, *,), (, /, \%, \$, ...\}$$
$$\{0, 1\}^n$$

El número de elementos m de depende del dispositivo o sistema informático que se esté utilizando.

Suponiendo un número n fijo de bits para codificar los m símbolos de , el valor mínimo de n dependerá del número m de elementos de . Así:

Con 2 bits (n=2) podemos hacer 4 combinaciones distintas (2^2), con lo que podremos codificar hasta 4 símbolos distintos (m=4).
Con 3 bits (n=3) podemos hacer 8 combinaciones distintas (2^3), con lo que podremos codificar hasta 8 símbolos distintos (m=8).
.....
Con n bits podremos codificar $m = 2^n$ símbolos distintos.

Dado un número m de símbolos a codificar, se necesita un código con n bits tal que $m = 2^n \Rightarrow n = \log_2 m$. Por consiguiente, n debe ser el menor número entero positivo que cumpla la relación:

$$n \quad \log_2 m = 3.32 \log(m)$$

Ejemplo. Para codificar las 10 cifras decimales se necesitarán $n = 3.32 \log(10) = 3.32$ bits
Es decir, 4 bits.
Por lo menos se necesitan 4 bits, pero pueden hacerse codificaciones con más bits de los necesarios.

Uno de los códigos usuales, el ASCII, suele utilizar unos 95 caracteres (es decir, m = 95), por lo que el número mínimo de bits para codificarlos es 7, ya que $2^6 < 95 < 2^7$.

Podríamos establecer códigos de E/S de forma totalmente arbitraria. No obstante, existen códigos de E/S normalizados que suelen ser utilizados por los fabricantes de ordenadores:

Código BCD de intercambio normalizado (*Standard Binary Coded Decimal Interchange Code*). Usualmente, este código utiliza 6 bits, con lo que puede representar $2^6 = 64$ caracteres. A veces se añade a su izquierda un bit adicional para verificar posibles errores en la transmisión o grabación (bit de paridad), con lo que cada carácter queda representado por 7 bits.

Las cuatro posiciones de la derecha se denominan bits de posición (para los caracteres numéricos del 1 al 9 coincide con la representación en binario natural de dichos números, por ello el BCD se denomina decimal codificado en binario). Los dos siguientes bits hacia la izquierda se denominan bits de zona, siendo éstos 00 para los caracteres numéricos. El último bit es opcional, de verificación.

Verificación	Bits de zona		Bits de posición			
6	5	4	3	2	1	0

Código EBCDIC (*Extended Binary Coded Decimal Interchange Code*). Utiliza 8 bits para representar cada carácter, por lo que puede codificar hasta 256 símbolos distintos. Esto posibilita representar una gran variedad de caracteres: incluye las letras minúsculas y muchos símbolos especiales. También es posible (se hace con las combinaciones que empiezan por 00) codificar caracteres que suministren órdenes o señales de control.

Código ASCII (*American Standard Code for Information Interchange*). Utiliza 7 bits (128 caracteres representables). Este código es de los más utilizados, se puede decir que la mayor parte de las transmisiones de datos entre dispositivos se realizan en esta codificación. Se suele incluir un octavo bit a la izquierda para detectar posibles errores de transmisión o grabación (bit de paridad). El código ASCII de 8 bits se denomina **código ASCII extendido**.

Existen otros códigos de uso más restringido que los anteriores (código de tarjeta, código **FIELDATA**: utilizado por los ordenadores Sperry-Univac serie 1100).

La siguiente tabla muestra la codificación de los dígitos numéricos según distintos códigos:

Carácter	BCD (octal)	EBCDIC (hexadecimal)	ASCII (octal)
0	112	F0	060
1	001	F1	061
2	002	F2	062
3	103	F3	063
4	104	F4	064
5	105	F5	065
6	106	F6	066
7	007	F7	067
8	010	F8	070
9	111	F9	071

4. REPRESENTACIÓN INTERNA DE LA INFORMACIÓN

En la memoria y el procesador central, la información se transmite y procesa en unidades denominadas **palabras**. Esto quiere decir que toda la información, a la hora de ser procesada por la unidad aritmético-lógica o ser transferida a la memoria principal, debe estructurarse en palabras. Debido a ello, y para hacer un buen aprovechamiento de la memoria, la longitud de la palabra debe ser un múltiplo entero del número de bits utilizados para representar un carácter. Así, por ejemplo, en los ordenadores de 8, 16, 32 o 64 bits se utilizan códigos de E/S de 8 bits (EBCDIC o ASCII, con bit de paridad), mientras que los de 36 o 60 bit utilizarán un código de E/S de 6 bits (por ejemplo, Sperry-Univac serie 1100 utilizaba el código FIELDATA).

Los datos se introducen inicialmente en el ordenador según un código de E/S, tanto si son de tipo alfabético como numérico. Los datos de tipo numérico se utilizan normalmente para operar aritméticamente con ellos, y la representación simbólica obtenida con el código de E/S no resulta adecuada para realizar este tipo de operaciones.

Imaginemos, por ejemplo, que deseamos operar con el número 253, y supongamos que está representado en notación ASCII con bit de paridad par, es decir:

$$253)_{10} = 1011\ 0010\ 0011\ 0101\ 0011\ 0011)_{ASCII}$$

Este mismo número representado en binario natural es:

$$253)_{10} = 11111101)_{2}$$

Resulta más adecuado operar con una notación fundamentada en el sistema matemático de numeración convencional (que precisamente se ideó para facilitar los cálculos aritméticos), que en un código de E/S. Por otra parte, la representación con el código de E/S utiliza un número excesivo de bits frente a la representación en binario natural: en el ejemplo, 24 bits, frente a 8 bits.

Por los motivos anteriores, y teniendo en cuenta que la unidad aritmético-lógica opera con palabras, se realiza una conversión de notaciones, pasando de la representación simbólica de E/S a otra notación que denominaremos **representación interna**. Esta representación interna dependerá del ordenador o del lenguaje de programación utilizado y del uso que el programador desee hacer con los datos. Es decir, el usuario, según las reglas del lenguaje de programación que utilice, puede optar entre varias representaciones posibles.

5. COMUNICACIÓN DE LA INFORMACIÓN

La **comunicación humana** consiste en un acto mediante el cual una persona establece contacto con otra, lo que le permite transmitirle una información.

El **emisor** es la persona que comunica la información, y el **receptor** es la que recibe la información.

Información será el conjunto de conocimientos o datos que permiten ampliar o precisar los que ya se poseen.

Si los conocimientos ya existían en la memoria, no habrá verdadera información sino confirmación, rememoración o actualización. Para que se dé verdadera información, la memoria o conciencia del receptor ha de incrementarse en sus contenidos con la nueva notificación recibida.

Mensaje es el conjunto de señales, signos o símbolos que son objeto de una comunicación, o bien, el contenido de esta misma comunicación.

En la comunicación intervienen varios **elementos fundamentales**.

Así, gestionando las posibles *relaciones entre usuario y ordenador* tendremos:

El usuario introduce datos en el ordenador para que sean procesados y produzcan unos resultados de salida.

El usuario debe codificar sus ideas mediante el código de un lenguaje reconocido por el ordenador.

El usuario debe utilizar un dispositivo adecuado o canal de transmisión para transmitir el mensaje al ordenador.

Para que el mensaje llegue íntegro a la memoria del ordenador debe transmitirse sin perturbaciones. Esto hace necesario utilizar un adaptador periférico-ordenador adecuado al dispositivo transmisor empleado.

Para poder "enterarse" de la idea del usuario, subyacente en el mensaje, el ordenador debe traducirlo a lenguaje máquina.

Por último, en la *comunicación usuario-ordenador* tenemos:

El sistema operativo del ordenador hace que se procese la información recibida en el sentido esperado por el usuario, siempre que su arquitectura y filosofía le confieran capacidad para realizarlo, y que no hayan entrado elementos perturbadores junto con el mensaje.

El ordenador debe transmitir, utilizar o almacenar los resultados obtenidos en la forma que le haya sido indicada, para lo que tendrá que utilizar un adaptador adecuado.

6. BIBLIOGRAFÍA

Alberto Prieto
Introducción a la Informática
Mc Graw-Hill, 2ª edición, 1997

Alfonso Ureña López
Fundamentos de Informática
Ra-ma, 1997

Pascual Laporta, G.
Estructura de la Información
Mc Graw-Hill, 1992