INTRODUCCIÓN A LOS COMPUTADORES[1]

(5/9/2008; 2ª PARTE: ejercicios; 7 puntos)

---

1. Un documento de un texto se compone de 37.724 palabras (considerando también como palabras los signos de puntuación), y por término medio en número de caracteres por palabra es 4,5. El número total de palabras distintas que contiene el texto es 629. Hacer una estimación del factor de compresión si el texto se codificase con un diccionario adaptativo en lugar de UNICODE.

SOLUCCIÓN

Codificación en UNICODE: Cada carácter se codifica con 16 bits, con lo que la capacidad del fichero sin comprimir será:

$$C_a = N_{palabras} \cdot N_{caracteres/palabra} \cdot N_{Bytes/palabra} = 37.724 \cdot 4{,}5 \cdot 2\ Bytes = 339.516\ Bytes$$

Compresión con diccionario adaptativo:

Como el diccionario contiene todas las palabras, pero sin repetir, la ocupación del diccionario será:

$C_{diccionario}$= 629 palabras· 4,5 caracteres/palabra · 16 bits/carácter = 45.288 bits = 5.661 Bytes

Como hay 629 palabras, se necesitarán 10 bits (2^9<629<2^{10}) para identificar (codificar) cada una de ellas; con lo que el texto ocupará:

C_{texto} = 37.724 palabras · 10 bits/palabra = 337.240 bits = 47.155 Bytes

Es decir, la capacidad total que ocupará el fichero con diccionario adaptativo será:

C_{da} = $C_{diccionario}$ + C_{texto} = 5.661 + 47.155 = 52.816 Bytes

Con lo que el factor de compresión será:

$$f_c = \frac{C_a}{C_d} = \frac{C_a}{C_{da}} = \frac{339.516\ Bytes}{52.816\ Bytes} = 6{,}42827\ \ldots \approx 6{,}4\ ;$$

Es decir se obtiene un **factor de compresión de 6,4 a 1.**

2. Dada una tabla de números enteros positivos que se inicia en la posición *B000* y finaliza en la *BFFF*, hacer un programa para CODE-2 que indique por el puerto *OP1* el número de datos de la tabla que son múltiplos de *4*.

SOLUCIÓN

Algoritmo: Un número binario es múltiplo de cuatro cuando acaba en dos ceros. En consecuencia el algoritmo debe recorrer uno a uno los elementos de la tabla y contabilizar los

---

Puntuación: 1a = 1,5 puntos
a= 1,5; 2b= 0,5; 2c = 1
3a = 0,5 ; 3b = 0,5; 3c= 1
Presentación (legibilidad y limpieza), si se hacen los ejercicios: 0,5

que acaban en dos ceros. Podemos determinar si el bit menos significativo es cero con un desplazamiento a la derecha (SHR) y comprobando el bit de acarreo (*C*) resultante. Las variables y parámetros utilizados se muestran en la Tabla.

Tabla: Asignación de registros y de memoria del programa de los múltiplos de 4

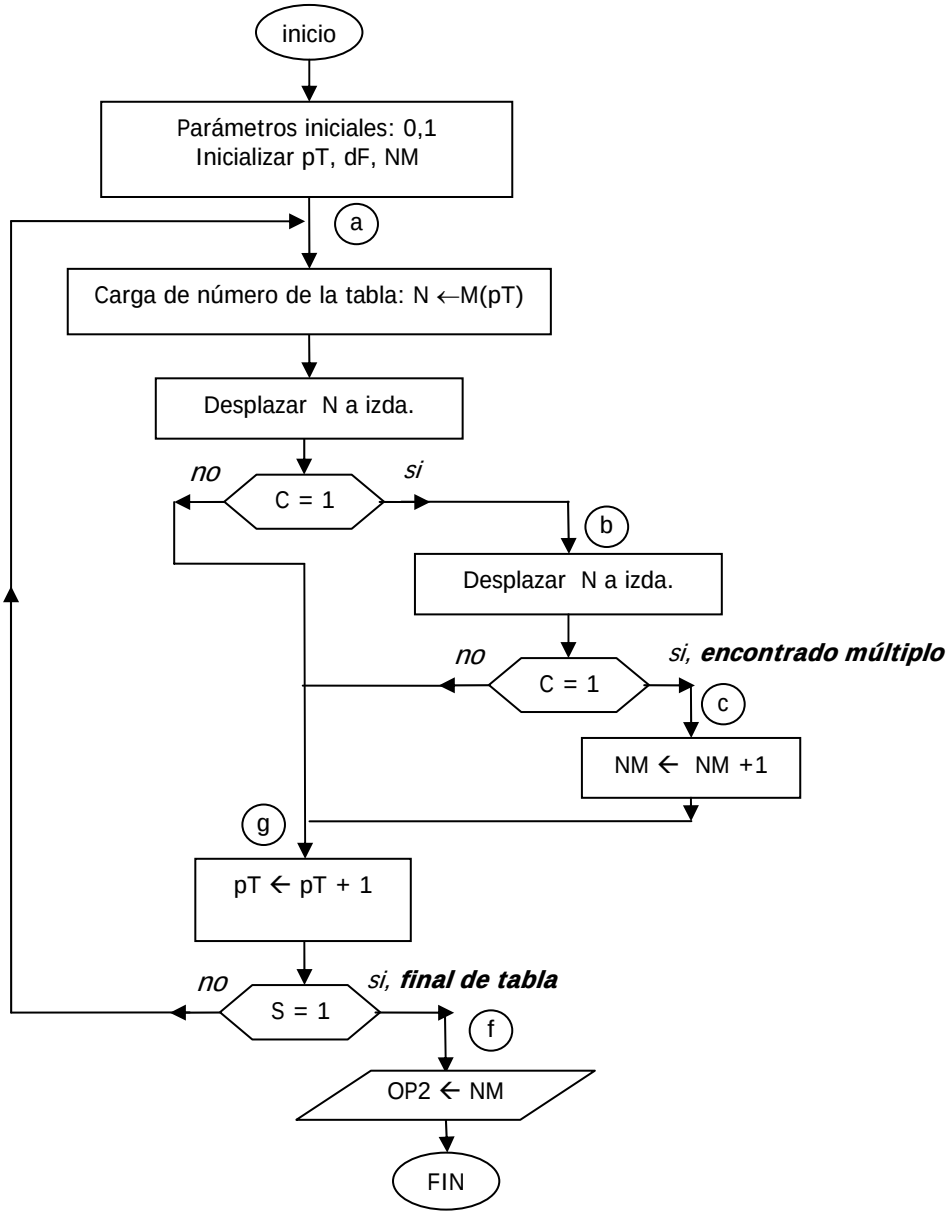
Parámetro o variable	***Comentario***
0000	Constante cero
0001	Para incrementar el puntero de la cadena
pT	Inicializar con la dirección de inicio de la cadena *B000*
N	Número en análisis
NM	Número de múltiplos (inicializar a 0000)
dF	Dirección final, *BFFF*

En la Figura se muestra el organigrama, que se explica por si mismo. Por otra parte, la asignación de registros y memoria se muestra en la Tabla.

Tabla: Asignación de registros y de memoria del programa de los múltiplos de 4

Parámetro o variable	***Registro***	***Posición de memoria***	***Comentario***
0000	r0		Constante cero
0001	r1		Para incrementar el puntero de la cadena
pT	r2		Inicializar con la dirección de inicio de la cadena *B000*
N	r3		Número en análisis
NM	r4		Número de múltiplos (inicializar a 0000)
dF	r5		Dirección final, *BFFF*
Programa		0000	Dirección de carga del programa

El programa en némonicos y en lenguaje máquina puede verse en la Tabla 4.27.



Organigrama del programa para detección de múltiplos de 4.

Programa detección de múltiplos de 4.

Rfcía. salto	Dircc. Memoria	Instrucción máquina		Explicación
		Nemónico	HEX	
	0000	LLI r0,00	2000	Inicializar r0 a 0000
	0001	LLI r1,01	2101	Inicializar r1 a 0001
	0002	LLI r2,3C	2D00	Dirección de inicio de tabla de números: *B000*
	0003	LHI r2,5F	32B0	
	0004	LLI r6,00	25FF	Dirección final de la tabla: *BFFF*
	0005	LHI r6,F0	36BF	
	0006	LLI r4,00	2400	NM← 0000
(a)	0007	ADDS rD,r2,r0	6D20	rD ← pT
	0008	LD r3,[00]	0300	Cargar número de la tabla en r3
	0009	SHR R3	A300	Desplazar a derecha el número
	000A	LLI rD,12	2D12	Dirección de salto (b)
	000B	BC	C300	Salta a (b) si acaba en cero

(g)	000C	ADDS r2,r2,r1	6221	pT ← pT +1
	000D	SUBS rF, r5, r2	7F52	dF – pT
	000E	LLI rD,1ª	2D1A	Dirección de salto (f)
	000F	BS	C200	Saltar a (f) si se llega a final de tabla
	0010	LLI rD,07	2D07	Dirección de salto (a)
	0011	BR	C000	Salto incondicional a (a)
(b)	0012	SHR r3	A300	Desplazar a derecha el número
	0013	LLI rD,17	2D17	Dirección de salto (c)
	0014	BC	C300	Salta a (c) si acaba en cero
	0015	LLI rD,0C	2D0C	Dirección de salto (g)
	0016	BR	C000	Salto incondicional a (g)
(c)	0017	ADDS r4,r4,r1	6441	NM ← NM +1
	0018	LLI rD,0C	2D0C	Dirección de salto (g)
	0019	BR	C000	Salto incondicional a (g)
(f)	001A	OUT OP2, r4	5402	Salida de máximo
	001B	HALT	FFFF	Fin

3. Su**ponga un computador que dispone de una memoria principal para ejecución de programas de CMP = 1 MBytes, y que en un momento dado se encuentran en la cola de espera un número ilimitado de procesos cada uno de los cuales ocupa en total CP=30 KBytes.**

Obtener el número de procesos, Np , que pueden ejecutarse concurrentemente (suponiendo que no hay intercambiabilidad con disco), y la capacidad total de memoria desaprovechada, Cf , suponiendo que el sistema operativo gestionase la memoria de las siguientes formas:

1a) Particiones estáticas de 64 KBytes
1b) Particiones dinámicas
1c) Paginación suponiendo que Cpág.= 4KBytes

SOLUCIÓN

1a) Particiones estáticas de 64 KBytes

El número de procesos coincidirá con el número de particiones:

$$N_P = \frac{C_{MP}}{C_{partición}} = \frac{1\ MB}{64\ KB} = \frac{1.024}{64} = 16$$

La capacidad de memoria desaprovechada será:

$$C_{fragmentación} = C_{MP} - 16 \cdot 30\ KB = 1.024\ KB - 480\ KB = 544\ KB$$

1b) Particiones dinámicas

Inicialmente se incluyen en la memoria uno detrás de otro todos los programas que quepan; es decir:

$$N_P = \frac{C_{MP}}{C_p} = \frac{1\ MB}{30\ KB} = \frac{1.024}{30} = 34{,}1333 \rightarrow 34\ procesos$$

La capacidad de memoria desaprovechada será:

$$C_{fragmentación} = C_{MP} - 34 \cdot 30\ KB = 1.024\ KB - 1.020\ KB = 4\ KB$$

1c) Paginación suponiendo que $C_{pág.}$= 4KBytes

El número de páginas que ocupa cada proceso es:

$$N_{páginas} = \frac{C_{proceso}}{C_{página}} = \frac{30\ KB}{4\ KB} = 7{,}5 \rightarrow 8\ páginas$$

Por otra parte, el número de marcos de página de la memoria principal es:

$$N_{marcos\ página} = \frac{C_{MP}}{C_{página}} = \frac{1\ MB}{4\ KB} = \frac{1.024}{4} = 256$$

Es decir, El número de procesos que caben es:

$$N_P = \frac{N_{mp}}{N_{páginas/proceso}} = \frac{256}{8} = 32$$

Con lo que la capacidad de memoria desaprovechada será:

$$C_{fragmentación} = C_{MP} - 32 \cdot 30\ KB = 1.024\ KB - 960\ KB = 64\ KB$$