UNIVERSIDAD DE LA LAGUNA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

Estructura de Computadores, curso 2010/11

Aclaraciones sobre la práctica de programación

Profesor: Juan Julian Merino Rubio

Sobre el cálculo de las distancias

En el programa propuesto se requiere calcular la distancia desde el origen de coordenadas hasta un punto (x, y). La distancia exacta se calcularía por medio del Teorema de Pitágoras:

$$d = \sqrt{x^2 + y^2}$$

y nos encontramos con el problema de calcular la raiz cuadrada (aproximada al entero más próximo) con las instrucciones que conocemos en ensamblador de números enteros. No está permitido usar las instrucciones de la unidad de coma flotante que sí tiene la instrucción `fsqrt`, pues esa unidad (la **FPU**) se estudiará en el 2º cuatrimestre.

El cálculo de la raiz cuadrada de un número se puede hacer probando sucesivos números crecientes hasta que lleguemos al que buscamos, algo así:

$k = 1$
while $k^2 < n$ **do**
    $k = k + 1$
endw
$k = k - 1$; es la aprox. por abajo

Al final es

$$k < \sqrt{n} \leq k + 1$$

Por ejemplo, para calcular $\sqrt{23}$, se iteraría así:

$$\begin{array}{lll} k = 1 & ; & 1 < 23 \\ k = 2 & ; & 4 < 23 \\ k = 3 & ; & 9 < 23 \\ k = 4 & ; & 16 < 23 \\ k = 5 & ; & 25 \geq 23 \Rightarrow k = 4 \text{ es la raiz entera de } 23 \end{array}$$

$$4 < \sqrt{23} \leq 5$$

¿nos quedamos con $k = 4$ o con $k = 5$?. Bueno, hay dos soluciones. La más fácil es truncar simplemente y siempre nos quedamos con la menor ($k = 4$). La otra solución es tomar el valor más aproximado.

Si $(n - k^2)$ es menor o igual que $((k+1)^2 - n)$ entonces tomamos k (el de la izquierda) y si no $(k+1)$ (el de la derecha).

Además la iteración se puede hacer más rápida evitando la multiplicación para calcular el cuadrado, pues conocido k^2 se puede calcular $(k+1)^2$ así:

$$(k+1)^2 = k^2 + 2 \cdot k + 1 \quad ,$$

y el algoritmo, sin multiplicación queda:

$k = 1$
$k2 = 1$
while $k2 < n$ **do**
    $k2 = k2 + 2 \cdot k + 1$
    $k = k + 1$
endw
$k = k - 1$

o bien, completándolo con el redondeo al más próximo:

$k = 1$
$k2 = 1$
while $k2 < n$ **do**
    $sk = k << 1$
    $k2 = k2 + sk + 1$
    $k = k + 1$
endw
$dde = k2 - n$
$k2 = k2 - sk - 1$
$diz = n - k2$
if $(dde < diz)$
    then $k = k - 1$

Distancias aproximadas

Por otro lado hay expresiones sencillas para calcular de forma aproximada la distancia desde el origen hasta un punto, que se pueden usar para iniciar las iteraciones en el algoritmo anterior y no tener que empezar siempre desde 1. El método más usado es el siguiente:

$a = \text{máx}(|x|, |y|)$
$b = \text{mín}(|x|, |y|)$
$d \approx \text{máx}(a, (\lfloor 7a/8 \rfloor + \lfloor b/2 \rfloor))$

que nos proporciona d con una aproximación sorprendente a base solo de sumas y desplazamientos.

Usando números con signo de 32 bits como tipo de dato básico en esta práctica, les recuerdo la forma rápida de hallar el valor absoluto de un número contenido en el acumulador:

```
ABSD    MACRO
        cdq
        xor     eax,edx
        sub     eax,edx
ABSD    ENDM
```