High Performance Computing
Departamento de Ingeniería en Informática
LAB1 : SIMD - paralelismo con SSE

1 Objetivo

El objetivo de este laboratorio es conocer, usar y apreciar las características de paralelismo a nivel de instrucción de un procesador moderno. Muchas veces, desconocemos por completo las capacidades de cómputo SIMD que los procesadores poseen, y rara vez nos preocupamos de explotarlas.

En este laboratorio usted implementará un programa mono-hebreado de procesamiento de imágenes.

2 Operadores morfológicos

En el ámbito del procesamiento de imágenes, la morfología consiste de un conjunto de operadores matemáticos que se aplican sobre imágenes. En morfología estos operadores se conocen por operadores morfológicos y generalmente se especifican como un patrón bidimensional binario o **elemento de estructuración (ES)**. Por ejemplo, la figura 1 muestra dos ES comunes de 3×3 píxeles:

0	1	0
1	1	1
0	1	0

1	1	1
1	1	1
1	1	1

Figure 1: Dos elementos de estructuración.

Todo ES tiene un centro. En las figuras anteriores, el centro es indicado con el pixel en gris. Como el ES se aplica a otra imagen, el centro del ES, indica sobre qué pixel de la imagen se está aplicando el ES, y por lo tanto el correspondiente pixel de la imagen resultante. Como se puede ver, un ES define una vecindad (o conjunto de píxeles) de la imagen original a los cuales se les aplicará alguna operación.

Los operadores morfilógicos comunes son:

1. Erosion: adelgaza el foreground
2. Dilation: expande el foreground
3. Closing: elimina hoyos en el foreground
4. Opening: elimina pixeles ruido del foreground

En este trabajo sólo implementaremos el operador **Dilation** ($\oplus$) para imágenes en escala de grises (gray-scale). Este operador se define matemáticamente como sigue. Sea A una imagen binaria y B un ES, entonces

$$C = A \oplus B \tag{1}$$

$$C = \bigcup_{p \in A} B_p \tag{2}$$

Una forma equivalente de definición es:

$$C = A \oplus B \quad (3)$$

$$C = \bigcup_{p \in B} A_p \quad (4)$$

Esta última definición es más apropiada para entender cómo implementar dilation en SIMD. El resultado se puede interpretar como la unión (suma) de los valores de los píxeles de la imagen original (A) que "están debajo de los píxeles con valor en 1 del ES" (B), para todos los pixeles del ES. Por ejemplo, considere el ES que define la vecindad de 4 elementos (arriba, abajo, izquierda y derecha), aplicada a una imagen binaria, como lo muestra la figura 2. El centro del ES es justamente el pixel central de la cruz. En la imagen se ha representado con dos tonalidades de gris el ES y la imagen. En el primer ejemplo, la unión da como resultado un valor vacío (o cero); en cambio en el segundo ejemplo, da como resultado un conjunto no vacío, que incluye a dos píxeles de la imagen original.

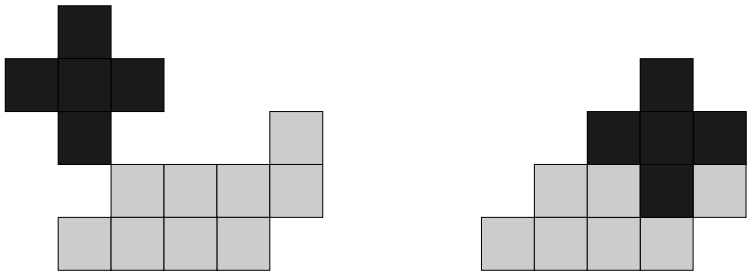

Figure 2: Aplicación del ES a dos posiciones distintas de la imagen.

3 Estrategia SIMD

Para este lab implementaremos dilation sólo con el ES que define la vecindad de 4 vecinos para imágenes. La estrategia de paralelización consistirá en usar 5 registros SIMD R_i, tal que $R_i[0]$ almacena los valores de la imagen correspondientes a una operación de dilatación. La imagen 3 muestra cómo debe cargarse los registros con los valores de la imagen.

Es decir, $R_0[0]$ contiene el valor del pixel superior; $R_1[0]$ el valor del pixel de la izquierda; $R_2[0]$ el valor del pixel de la derecha; $R_3[0]$ el valor del pixel inferior; y $R_4[0]$ el valor del pixel central. Luego, usando una red de funciones `mm_max()` es posible obtener el pixel resultante en 4 posiciones de la imagen. Sólo use `mm_max()` para obtener el valor resultante.

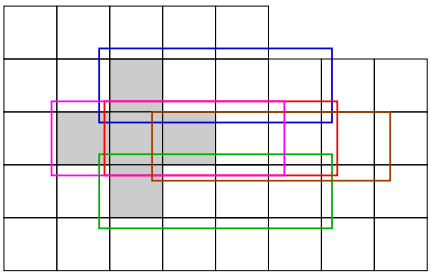

Figure 3: Asignación de registros a pixeles en base el ES.

4 El programa y la imagen

Diseñe un programa en C que implemente la operación de dilatación en forma secuencial y en forma paralela SIMD. El programa deber ser invocado de la siguiente forma:

```
$ ./dilation -i imagen_entrada.pgm -s imagen_salida1.pgm -p imagen_salida2.pgm
                  -N ancho_imagen
```

Es requerimiento de este lab que para manejar argumentos utilice `getopt()`. Documéntese!

La opción `-s` indica el nombre del archivo de la imagen resultante del proceso secuencial, y la opción `-p` indica la imagen resultante del proceso SIMD. Las imágenes estarán en formato binario PGM, es decir formato P5:

1. En la primera línea: el string "P5"
2. En la segunda línea: el número de filas y columnas de la imagen, en ASCII
3. En la tercera línea: el valor máximo de la imagen, en ASCII
4. Luego vienen los bytes que representan los píxeles de la imagen, en binario

Por ejemplo:

```
P5
512 512
255
@
...
```

Los píxeles de la imagen pueden ser leídos con `read()` o `fread()`. De la misma forma, los píxeles de la imagen resultante pueden ser escritos con `write()` o `fwrite()`.

Al finalizar el programa, se debe imprimir por stdout el tiempo de ejecución secuencial y el tiempo de ejecución SIMD, de sólo de la operación de dilatación. No incluir I/O.

5 Entregables

Tarree, comprima y envíe a `fernando.rannou@usach.cl` al menos los siguientes archivos:

1. `Makefile`: archivo para make que compila ambos programas
2. `dilation.c`: archivo con el código. Puede incluir otros archivos fuentes.

Fecha de entrega (hard-deadline): 01 de octubre, 2023 antes de las 23:59 hrs.

.