# Paralelización de operaciones con matrices

El requerimiento de paralelización se puede presentar de varias formas. Se puede requerir la ejecución de differentes instrucciones sobre diferentes datos (MIMD), differentes instrucciones sobre los mismos datos (MISD), mismas instrucciones sobre diferentes datos (SIMD) o mismas instrucciones sobre los mismos datos (SISD). En la actualidad, con la popularidad de los algoritmos de machine learning la opción SIMD (Single Instruction Multiple Data) se ha convertido en una de las formas de paralelismo más requeridas. Las GPUs son especialmente idóneas para dicho tipo de paralelismo y han sido en parte las responsables de ofrecer el poder de cómputo requerido por los algoritmos de machine learning actuales.

En la base del aprovechamiento de las GPUs para el entrenamiento de algoritmos de inteligencia artifial se encuentran las operaciones matriciales. En esta práctica se les pide paralelizar un conjunto de operaciones que se ejecutan cientos de millones de veces durante el entrenamiento de las redes neuronales (NN). Como ustedes saben, Python es el lenguaje por excelencia para definir la estructura de una NN y normalmente las librerías más usadas en Python para NN tienen un backend que esta escrito en lenguaje C. Esta práctica entonces es además una invitación a conocer lo que sucede detrás de la escena cuando usted entrena una red neuronal.

Cabe mencionar que las GPUs ofrecen un paralelismo a nivel de hardware (aprovechable por ejemplo a través de frameworks como CUDA) mientras que en esta práctica se espera que los estudiantes hagan uso de la librería PThread de C para paralelizar las operaciones pedidas.

En esta práctica se les pide paralelizar las siguientes operaciones:

1. Calcular la media de cada columna de una matriz
2. Calcular la varianza de cada columna de una matriz
3. Calcular la desviacion estandar de cada columna de una matriz
4. Calcular el valor mínimo y el valor máximo de cada columna de una matriz
5. Sumar dos matrices
6. Realizar el producto punto de dos matrices (Tenga en cuenta que los vectores pueden verse como casos particulares de las matrices)
7. Multiplicar un escalar por una matriz
8. Normalizar una matriz columna por columna de acuerdo con la siguiente formula:

$$x' = (x - x_{min})/(x_{max} - x_{min})$$

   *x'* es el nuevo valor que tomara cada elemento de la matriz
   *xmin* es el valor mínimo de cada columna
   *xmax* es el valor máximo de cada columna

9. Normalizar una matriz columna por columna de acuerdo con la siguiente formula:

$$x' = (x - \mu)/\sigma$$

*x'* es el nuevo valor que tomara cada elemento de la matriz
*u* es la media de cada columna
*r* es la desviacion estandar de cada columna

El programa principal debe llamarse ***matrices*** y debe recibir una serie de parámetros indicados a través del uso de flags (véase el uso de la función *getopt* de C, en el ejemplo de calentamiento para la práctica se muestra su uso) de la siguiente manera:

- *-o:* Para indicar el número correspondiente a la operación a realizar (único flag obligatorio).
- *-f*: el número de filas del primer operando de la operación a realizar
- *-c*: el número de columnas del primer operando de la operación a realizar
- *-r*: el número de filas del segundo operando de la operación a realizar
- -s: el número de columnas del segundo operando de la operación a realizar
- -p: el path del archivo con los operandos de la operación a realizar (Si se indica esta opción, no es necesario indicar los tamaños de los operandos y estos deben deducirse del archivo)

Cuando no se indica el flag *-p* se deben requerir los flags *-f, -c, -r,-s* y todos los valores de todos los operandos deben inicializarse de forma random.

Cuando los operandos se indican a través de un archivo (uso del flag *-p*) el formato de dicho archivo obedece las siguientes reglas:

- Si se quiere indicar un valor escalar para la operación este debe indcarse como primera entrada. Se debe empezar una línea con el caracter '*s'* seguido por un espacio en blanco e inmediatamente después se debe indicar el correspondiente valor escalar.
- Para indicar el primer operando se debe empezar una línea con los caracteres *'op1'* seguido por dos valores enteros, el primero indicando las filas del operando (*f*) y el segundo indicando las columnas (*c*) del operando. Cada valor debe estar separado de la entrada siguiente por un espacio en blanco. Despues de esto se deben indicar *f* lineas cada una conteniendo *c* elementos. Usted debe validar que las filas y columnas sean consistentes con los valores indicador en *f* y *c.*
- La indicación del segundo operando es igual que la del primero pero utilizando *'op2'* en lugar de *'op1'*.

Ejemplo:

```
s -0.007412489037960767745971679688
op1 3 2
-0.007412489037960767745971679688 -0.007412489037960767745971679688
-0.007412489037960767745971679688 -0.007412489037960767745971679688
-0.007412489037960767745971679688 -0.007412489037960767745971679688
```

Su programa debe validar que los argumentos indicados en el archivo de entrada se correspondan con los requeridos por la operacion pedida. Esto incluye que se validen los tamaños de los operandos y por lo tanto el orden en el que los operandos son indicados es importante ya que las operaciones con matrices no son conmutativas. Una matriz 4x3 (filas por columnas) se puede multiplicar con un vector columna de tres posiciones pero un vector columna de tres posiciones no se puede multiplicar con una matriz de 4x3 a menos que se transponga (transpose) la matriz o el vector.

En esta práctica se asume que todos los operandos corresponden a valores double. Por lo tanto, estos pueden tener signo y las posiciones decimales se indican con el uso del caracter '.'.

Cada vez que se ejecute una operación usted debe presentar en la salida estandar el tiempo que demoró la ejecución secuencial en la primera línea y el tiempo que demoró la ejecución paralela en la segunda línea seguido por el resultado de la operacion (No pongo ninguna descripción en la salida estandar, solo los valores). Tenga en cuenta que el resultado debe ser consistente con la operacion; es decir; si multiplico una matriz 4x3 por un vector 3x1, el resultado es un vector 4x1 lo que significa que el resultado requiere 4 lineas cada una con un solo valor mientras que si multiplico un vector 1x3 por una matriz 3x4 el resultado es un vector 1x4 lo que significa que el resultado require una sola linea con 4 valores separados por un espacio en blanco.

Como punto de partida para la practica se les hace entrega de un archivo .c y .h con unas estructuras (Vector y Matrix) y unas operaciones ya implementadas sobre dichas estructuras. Tenga en cuanta que este es solo un punto de partida y son ustedes quienes deben definir si requieren algo diferente.

La práctica se debe entregar el dia 28 de Abril, todos los miembros del equipo deben participar y estar preparados para responder las preguntas del profesor con respeto a su código. Deben enviar el código de su práctica en un archivo .zip con el siguiente formato de nombramiento: Prac3_Presencial_Apellido1_Apellido2….zip. Tenga en cuenta las políticas de entrega tardía definidas para el curso. La fecha y hora de entrega límites son el 28 de Abril a las 23:59.