TALLER DE PROGRAMACIÓN SOBRE GPUS

Facultad de Informática – Universidad Nacional de La Plata

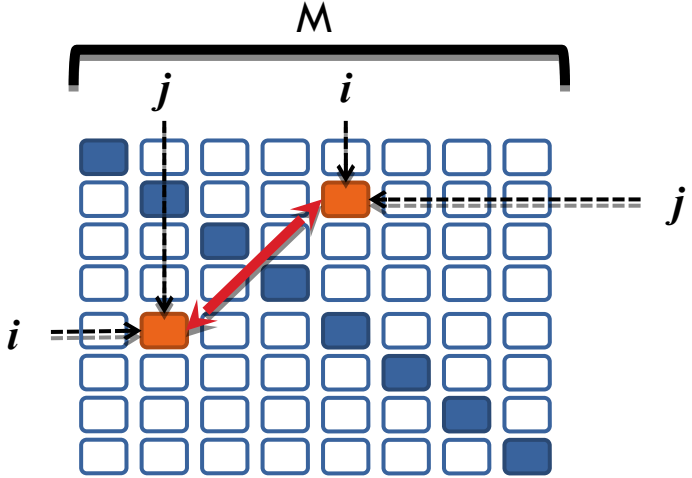
Dr. Adrián Pousa

Ejemplo: Calcular la matriz transpuesta

Transpuesta de una matriz de NxN

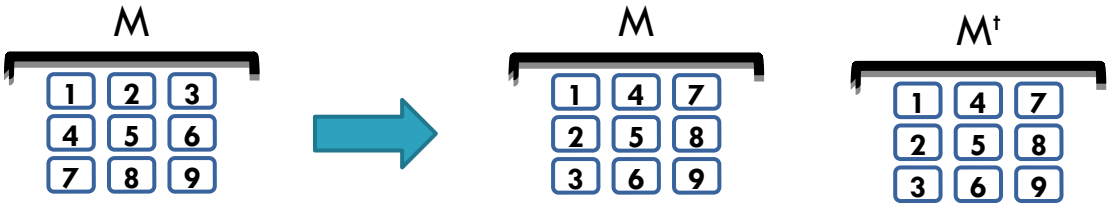
- Dada una matriz M de NxN elementos calcular su transpuesta.
- El elemento (i,j) se convertirá en el elemento (j,i) de

$$M(i,j)=M^t(j,i)$$



Transpuesta de una matriz de NxN

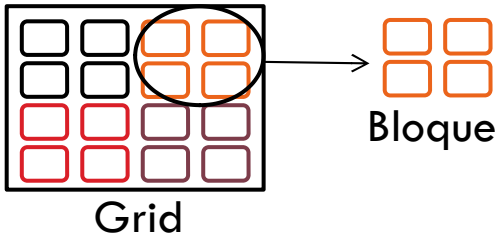
- El algoritmo que calcula la transpuesta puede ser:
  - Out- Place: se calcula sobre una matriz diferente sin alterar la matriz original



  - In-Place: se calcula sobre la misma matriz

Transpuesta de una matriz de NxN

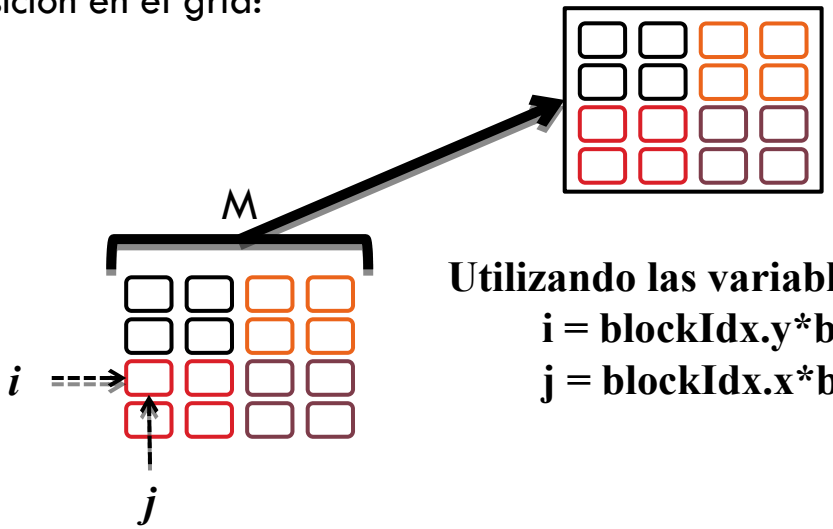
- Ejemplo de solución out-place:
- Se crea un Grid de 2x2 Bloques y cada Bloque de 2x2 Hilos:



Transpuesta de una matriz de NxN

- Ejemplo de solución out-place:

- Se mapea el grid a la matriz M de manera que cada hilo lee la posición correspondiente a su posición en el grid:

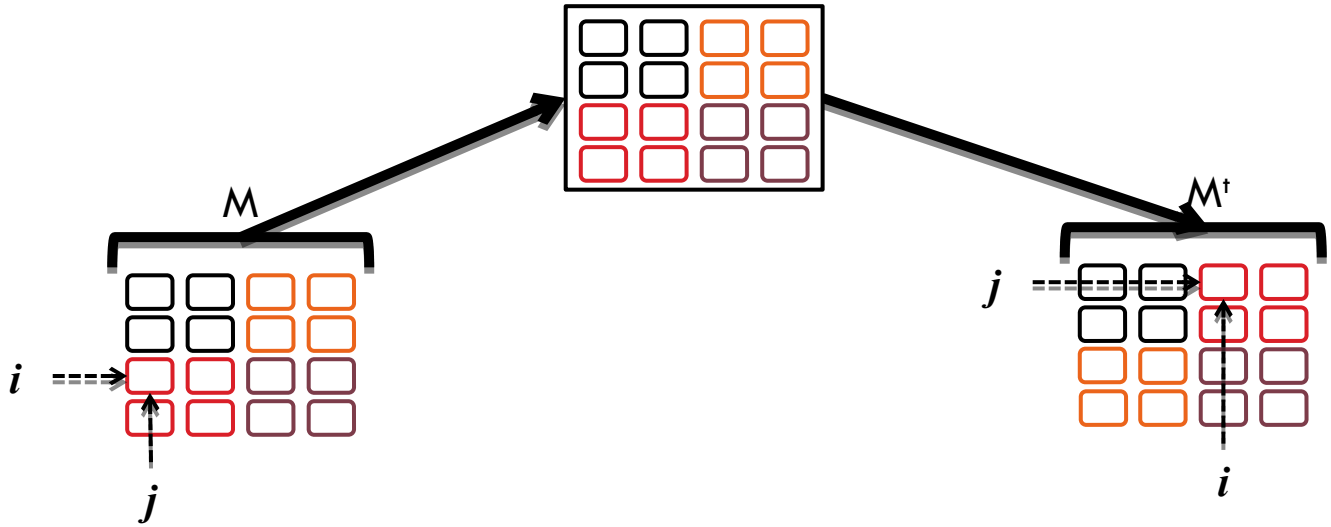


Utilizando las variables built-in de CUDA:

```
i = blockIdx.y*blockDim.y + threadIdx.y;
j = blockIdx.x*blockDim.x + threadIdx.x;
```

Transpuesta de una matriz de NxN

- Ejemplo de solución out-place:
- El hilo en la posición (i,j) escribe en la posición (j,i) de la matriz M^t:



Transpuesta de una matriz de NxN

- La solución out-place:

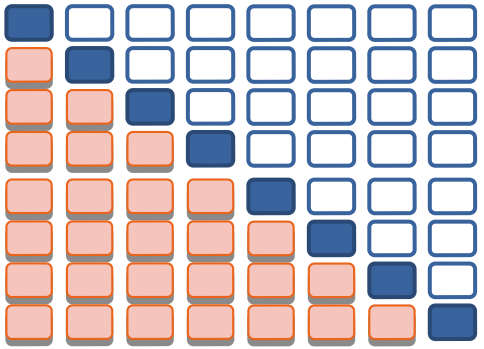
```
__global__ void transpuesta_out_place(float *mt, float *m, int N){
  int i = blockIdx.y*blockDim.y + threadIdx.y;
  int j = blockIdx.x*blockDim.x + threadIdx.x;

    if ( (i<N) && (j<N) )
        mt[j*N + i] = m[i*N + j];
}
```

Transpuesta de una matriz de NxN

- Ejemplo de solución in-place:
- Si se crean la misma cantidad de Hilos que elementos (idem out-place), existe el problema de sincronizar entre los hilos que transponen las mismas posiciones:
  - Es posible que se deban sincronizar dos hilos que pertenecen a bloques diferentes y esto lleva a una sincronización con un costo muy alto en el rendimiento.
- Para minimizar recursos la idea es crear una cantidad de hilos igual a la cantidad de intercambios:
  - Esto es igual a la cantidad de elementos en el triangulo inferior (o superior de la matriz).

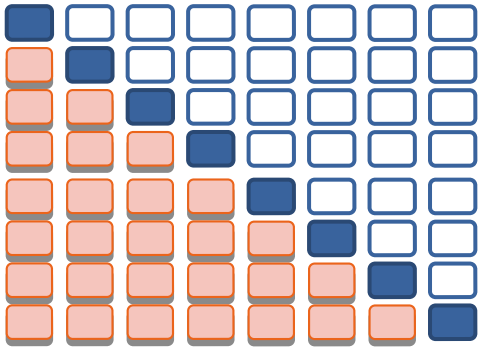
Transpuesta de una matriz de NxN

- El cálculo de la cantidad de elementos en el triangulo inferior se puede obtener de dos maneras:
  - Aritmética simple
  - Utilizando la ecuación de Gauss



Transpuesta de una matriz de NxN

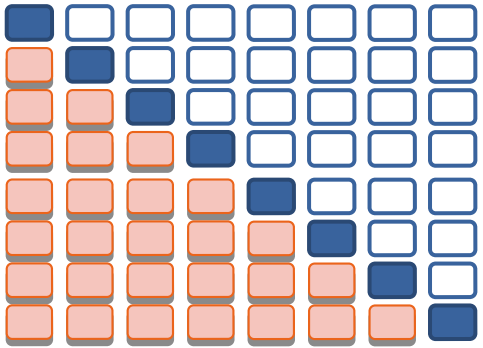
- Por aritmética simple hay que tener en cuenta:
  - NxN es la cantidad de elementos de la matriz
  - Hay N elementos en la diagonal
  - Si a NxN le restamos los elementos de la diagonal nos quedan los elementos de los dos triángulos. Es la mitad de este valor el que interesa.



$$t_{inf} = \frac{N \cdot N - N}{2} = \frac{N^2 - N}{2} = \frac{N \cdot (N-1)}{2}$$

Transpuesta de una matriz de NxN

- Utilizando la ecuación de Gauss miramos sólo el triángulo inferior (sin tener en cuenta la primera fila):
  - Transponemos **1** elemento en la primer fila
  - Transponemos **2** elementos de la segunda fila
  - Transponemos **3** elementos de la segunda fila
  - …
  - Transponemos **N-1** elementos de la fila N



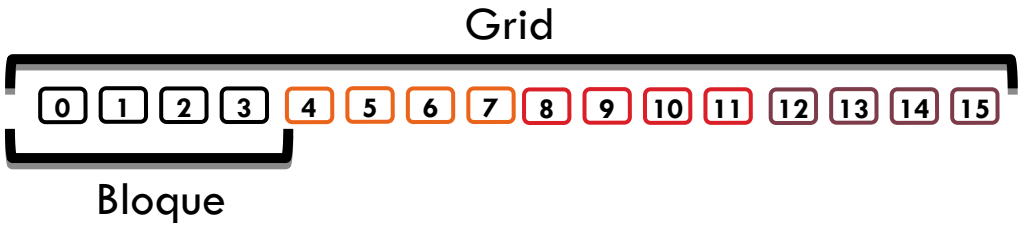
- Esto es la suma de N-1 elementos:

$$Gauss = \sum_{k=1}^{n} k = \frac{n \cdot (n+1)}{2} \quad \Rightarrow \quad t_{inf} = \sum_{k=1}^{n=N-1} k = \frac{(N-1) \cdot ((N-1)+1)}{2} = \frac{N \cdot (N-1)}{2}$$

Transpuesta de una matriz de NxN

- Debemos crear $t_{inf} = \frac{N \cdot (N-1)}{2}$ Hilos, cada uno asignado a una posición.
- Vamos a crearlos como un Grid unidimensional de Bloques unidimensionales.
- Además, utilizaremos el identificador único de cada Hilo calculado como:

tid = blockIdx.x*blockDim.x + threadIdx.x;

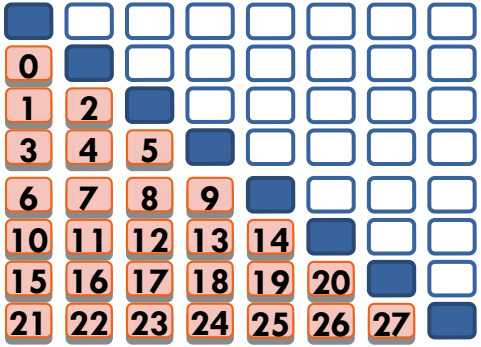


Transpuesta de una matriz de NxN

- Dado un Hilo con un identificador *tid* el problema es:

¿Cómo obtener los índices *i* y *j* del elemento que este hilo debe intercambiar?

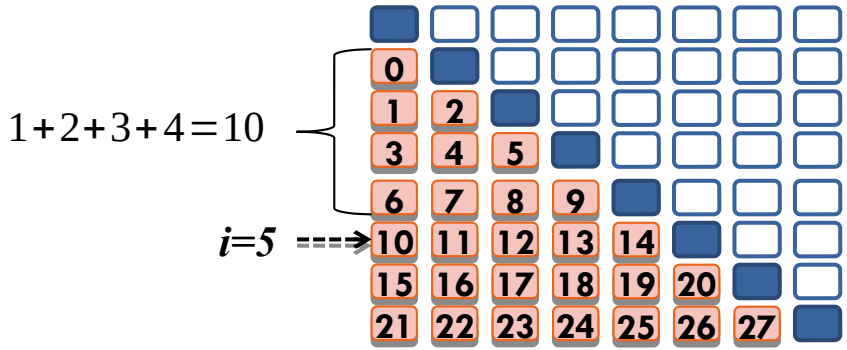
- Vamos a asignar los hilos (según *tid*) a los elementos de la matriz de la siguiente manera:



Transpuesta de una matriz de NxN

¿ Cómo calculamos la coordenada *i* ?

- Si tomamos un elemento de la primera columna, vemos que la cantidad de elementos hasta ese punto coincide con la suma de Gauss:



- Es decir, el *tid* del Hilo en la fila *i* columna *0* coincide con la fórmula de Gauss:

$$tid_{columna\,0} = \sum_{k=1}^{i-1} k = \frac{(i-1)\cdot((i-1)+1)}{2} = \frac{(i-1)\cdot i}{2} = \frac{i^2-i}{2}$$

$$i=5 \rightarrow \frac{i^2-i}{2} = \frac{5^2-5}{2} = \frac{25-5}{2} = \frac{20}{2} = 10$$

Transpuesta de una matriz de NxN

- Nos interesa despejar i de la fórmula:

$$tid_{columna\,0} = \frac{i^2 - i}{2} \quad \Rightarrow \quad i^2 - i - 2 \cdot tid_{columna\,0} = 0$$

- Nos queda una ecuación cuadrática que resolvemos aplicando la ecuación:

$$\boxed{\frac{-b \pm \sqrt{b^2 - 4ac}}{2a}}$$

- Sólo nos interesa la parte positiva de la raíz cuadrada.

Transpuesta de una matriz de NxN

$$i^2 - i - 2\,tid_{columna\,0} = 0 \quad \begin{cases} a = 1 \\ b = -1 \\ c = -2 \cdot tid_{columna\,0} \end{cases}$$

$$\frac{-b + \sqrt{b^2 - 4ac}}{2a} = \frac{-(-1) + \sqrt{(-1)^2 - 4 \cdot (-2 \cdot tid_{columna\,0})}}{2} = \frac{1 + \sqrt{1 + 8 \cdot tid_{columna\,0}}}{2}$$

$$\boxed{i = \frac{1 + \sqrt{1 + 8 \cdot tid_{columna\,0}}}{2}}$$

Transpuesta de una matriz de NxN

- La ecuación puede utilizarse para cualquier i, no sólo en la columna 0:

$$i = \frac{1 + \sqrt{1 + 8 \cdot tid}}{2}$$

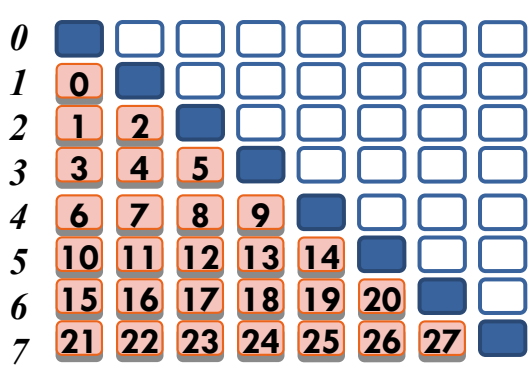
Si el tid de un Hilo se corresponde con un valor de la ecuación de Gauss (columna 0) la ecuación da un número entero que representa la coordenada i que ese Hilo deberá intercambiar.

Si el tid de un Hilo NO se corresponde con Gauss (NO está en la columna 0) la ecuación NO da un número entero, pero la parte entera se corresponde con la coordenada i que ese hilo deberá intercambiar.

Transpuesta de una matriz de NxN

- Ejemplo:

$i =$



tid	$\frac{1+\sqrt{1+8 \cdot tid}}{2}$	i
0	1	1
2	2,56155281	2
4	3,37228132	3
8	4,53112887	4
10	5	5
15	6	6
18	6,52079729	6
23	7,30073525	7

Transpuesta de una matriz de NxN

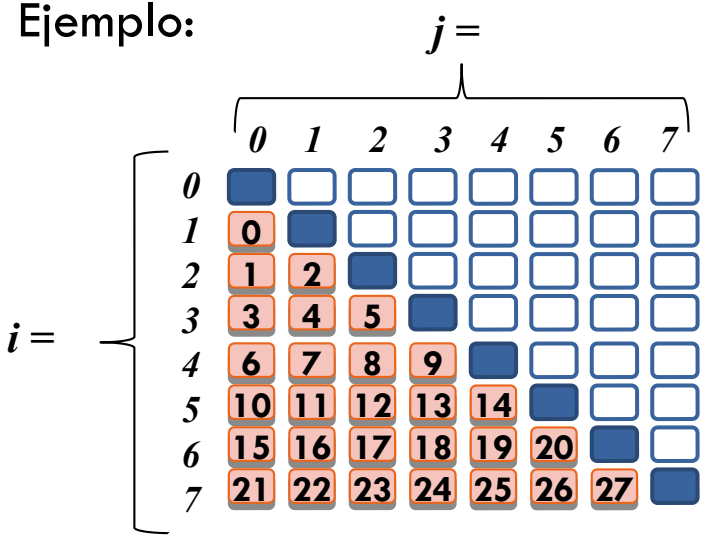
¿ Cómo calculamos la coordenada *j* ?

- Debemos obtener la coordenada j que el Hilo tid va a intercambiar.
- Si conocemos fila en la que está el elemento que debe trasponer el Hilo tid, es decir i, la columna puede calcularse restando tid al tid del Hilo que está en la misma fila pero columna 0:

$$j = tid - tid_{columna\,0} = tid - \frac{i^2 - i}{2}$$

Transpuesta de una matriz de NxN

- Ejemplo:



tid	$\frac{1+\sqrt{1+8 \cdot tid}}{2}$	i	$j = tid - \frac{i^2 - i}{2}$
0	1	1	0
2	2,56155281	2	1
4	3,37228132	3	1
8	4,53112887	4	2
10	5	5	0
15	6	6	0
18	6,52079729	6	3
23	7,30073525	7	2

Transpuesta de una matriz de NxN

- La solución in-place:

```
__global__ void transpuesta_in_place(float *m, int N){
 int tid = blockIdx.x*blockDim.x + threadIdx.x;
 int i = int((1 + sqrtf(1 + 8*tid)) / 2);
 int j = tid - (i*(i-1)/2);  int aux;

    if ( (i<N) && (j<N) ){
        aux = m[i*N + j] ;
        m[i*N + j] = m[j*N + i];
        m[j*N + i] = aux;
    }
}
```

Transpuesta de una matriz de NxN

- Out-Place vs In-Place:
  - Out-place: requiere más espacio de memoria pero realiza menos operaciones
  - In-place: requiere menos espacio de memoria pero realiza más operaciones

- Generalmente, la solución Out-Place alcanza mayor rendimiento que la solución In-Place.
- El número de operaciones que realiza la solución In-Place impacta negativamente en el rendimiento.