UF I5AISE51
Dimensionnement et évaluation des architectures
Introduction à la programmation massivement parallèle

—

Partie 5
Considérations sur les performances : memory coalescing

—

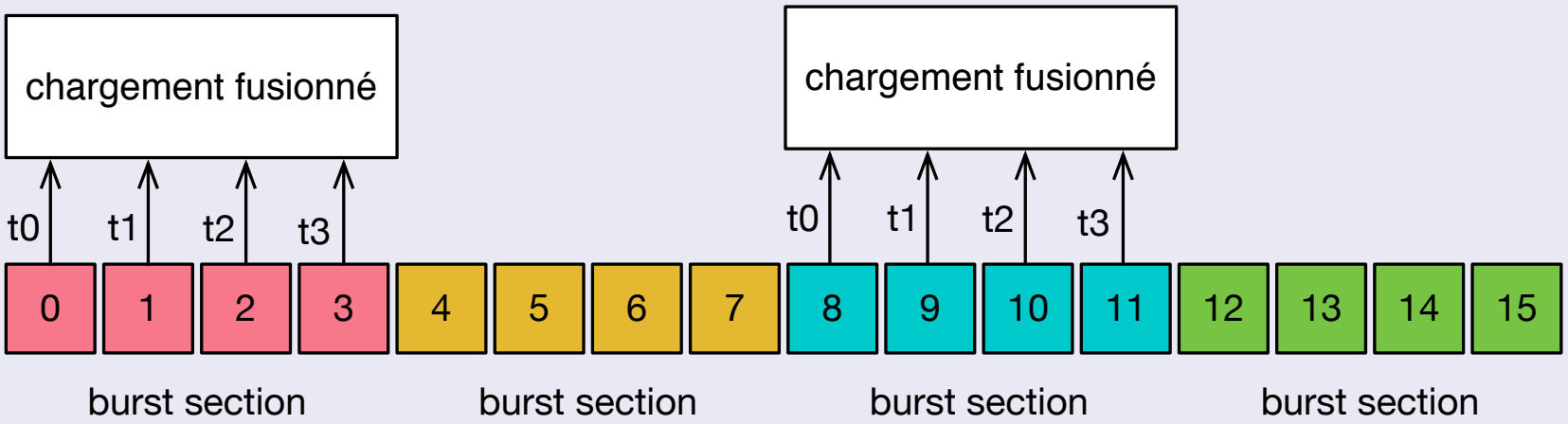
P.-E. Hladik

INSA Toulouse

11 janvier 2021

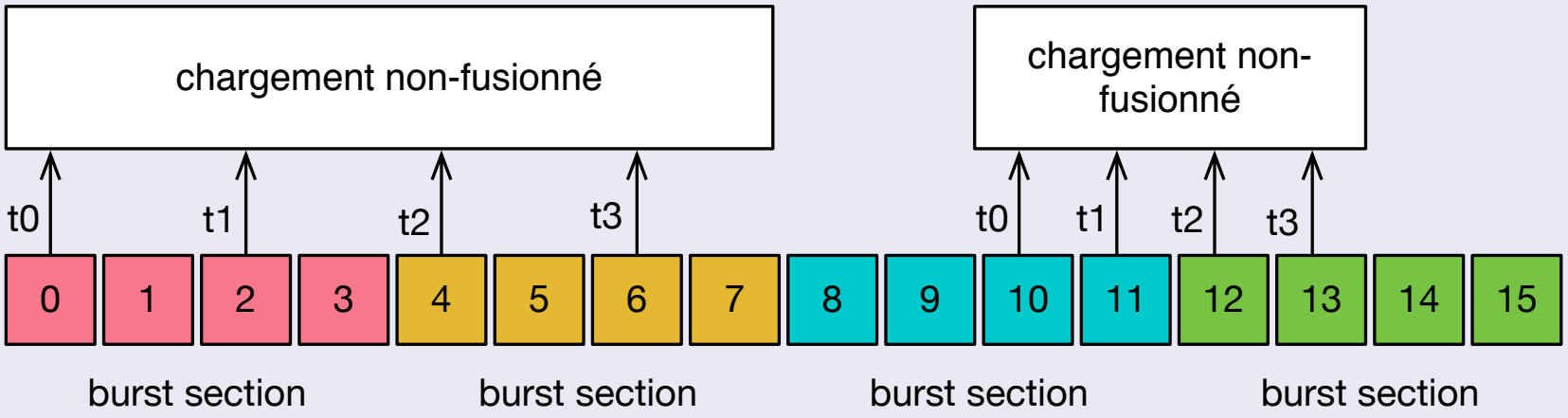
Plan

Memory Coalescing



Lorsque tous les threads exécutent une instruction de chargement, si tous les emplacements accédés tombent dans la même burst section, une seule demande de DRAM sera faite et l'accès est entièrement fusionné (*coalesced*).

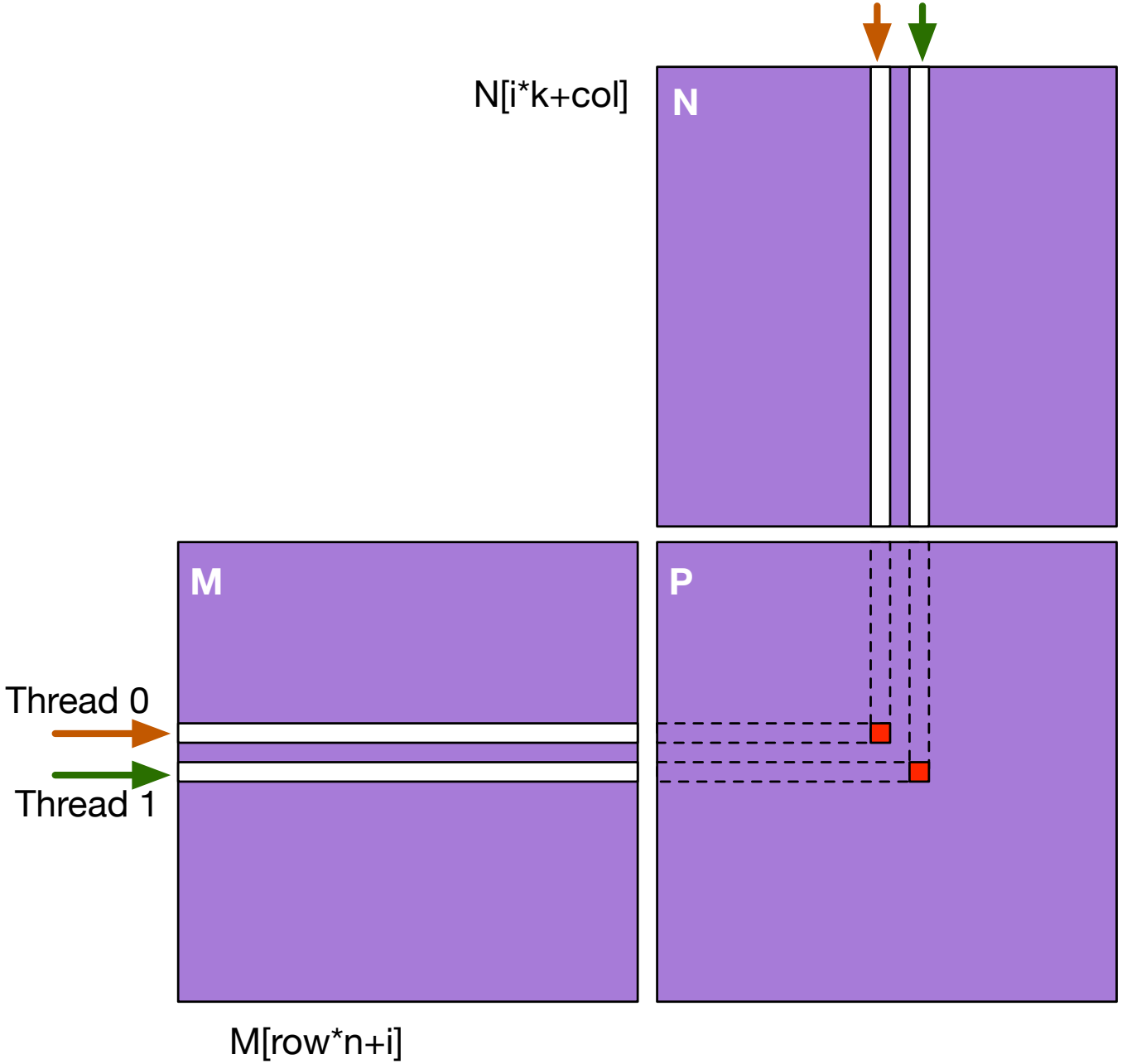
Accès Un-coalesced



Lorsque les adresses accessibles sont sur plusieurs burst sections :

- La fusion échoue
- Plusieurs demandes de DRAM sont faites
- L'accès n'est pas totalement fusionné
- Certains des octets auxquels on accède et qui sont transférés ne sont pas utilisés par les threads

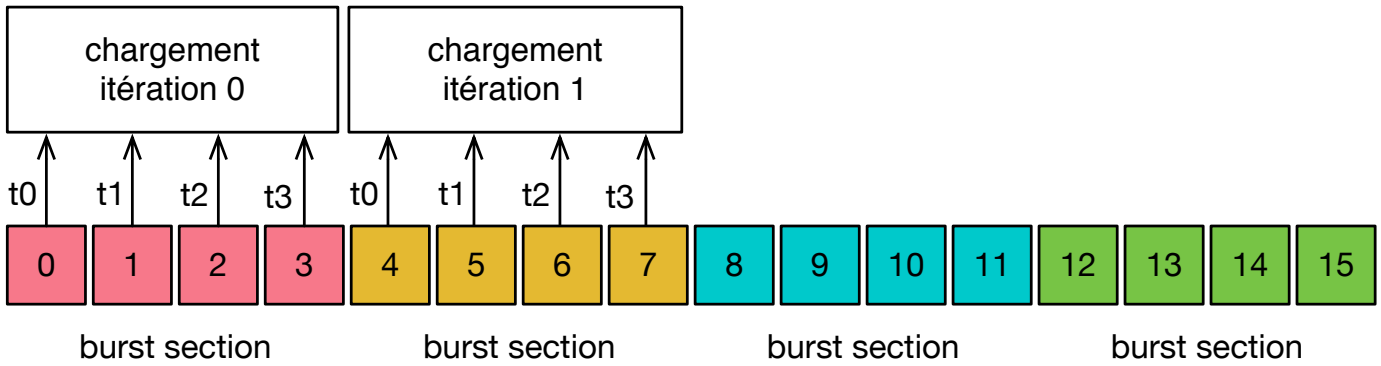
Exemple : produit matriciel

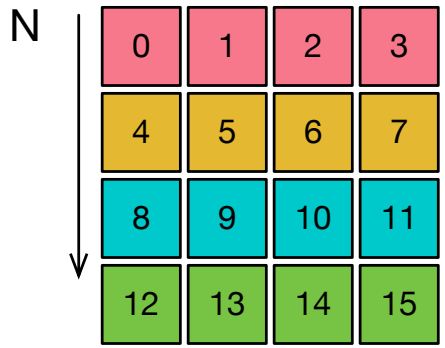


Exemple avec la multiplication matricielle : kernel

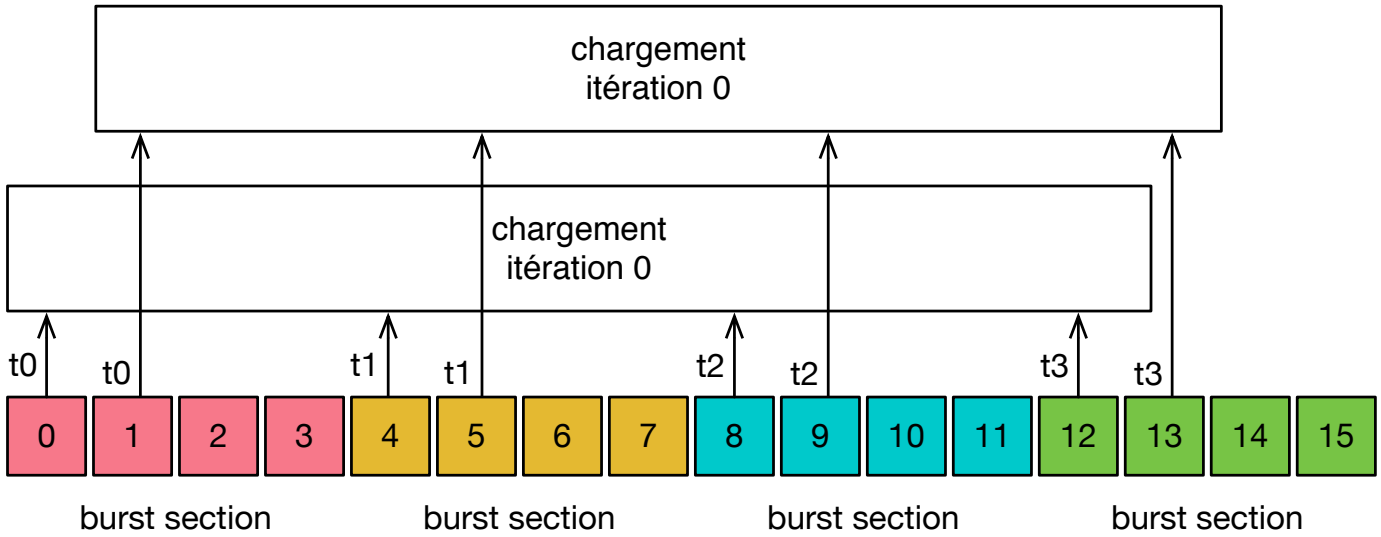
```
__global__
void MatrixMulKernel(float* M, float* N, float* P, int Width) {

  // Calculate the row index of the P element and M
  int Row = blockIdx.y*blockDim.y+threadIdx.y;

  // Calculate the column index of P and N
  int Col = blockIdx.x*blockDim.x+threadIdx.x;

  if ((Row < Width) && (Col < Width)) {
    float Pvalue = 0;
    // each thread computes one element of the block sub-matrix
    for (int k = 0; k < Width; ++k) {
      Pvalue += M[Row*Width+k]*N[k*Width+Col];
    }
    P[Row*Width+Col] = Pvalue;
  }
}
```

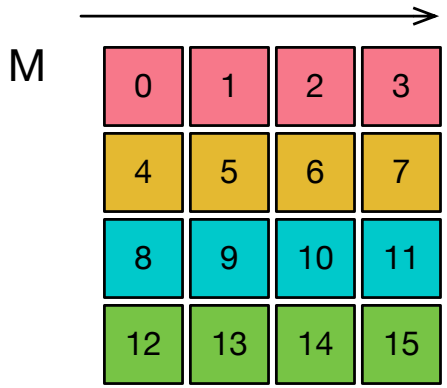
Exemple : produit matriciel – accès fusionné





Exemple : produit matriciel – accès fusionné





Exemple avec la multiplication matricielle : kernel

```
__global__
void MatrixMulKernel(float* M, float* N, float* P, IntWidth)
{
  __shared__ float ds_M[TILE_WIDTH][TILE_WIDTH];
  __shared__ float ds_N[TILE_WIDTH][TILE_WIDTH];

  int bx = blockIdx.x; int by  = blockIdx.y;
  int tx = threadIdx.x; int ty = threadIdx.y;

  int Row = by * blockDim.y+ ty;
  int Col = bx* blockDim.x+ tx;
  float Pvalue= 0;

  // Loop over the M and N tiles required to compute the P element
  for (int p = 0; p < n/TILE_WIDTH; ++p) {
    // Collaborative loading of M and N tiles into shared memory
    ds_M[ty][tx] = M[Row*Width + p*TILE_WIDTH+tx];
    ds_N[ty][tx] = N[(t*TILE_WIDTH+ty)*Width + Col];
    __syncthreads();

    // Usage
    for (int i = 0; i < TILE_WIDTH; ++i) Pvalue+= ds_M[ty][i] * ds_N[i][tx];
    __syncthreads();
  }
  P[Row*Width+Col] = Pvalue;
}
```

Exemple : produit matriciel – accès fusionné