⑥ N=17, Block Size = 4

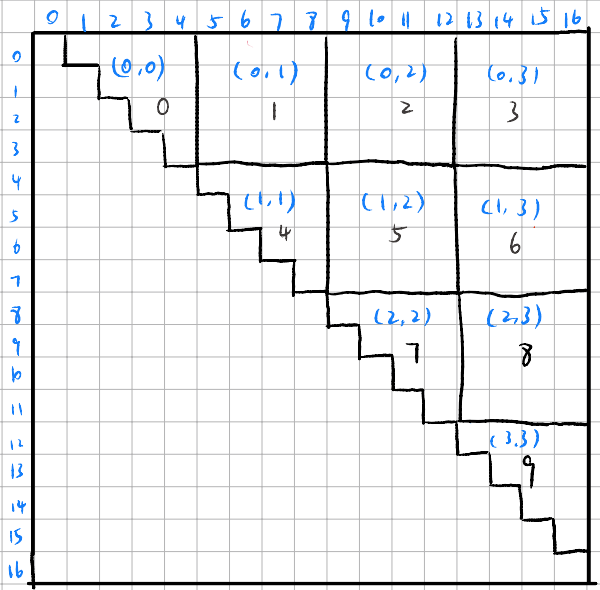
hori_group_num = (numbodies - 2) // Block Size + 1

= (17-2)//4+1 = 4 ✓

total_block_num = $\frac{(\text{hori\_group\_num} + 1)\,\text{hori\_group\_num}}{2}$

= $\frac{4\times5}{2}$ = 10 ✓

① Block Id ⟶ (group_id_x , group_id_y)



输入: BlockId (记为X)

len = hori_group_num; layer = 0 # 记录属于第几层(从0开始)

X = X+1 # 将从0开始的序号转化为从1开始的第几个

while X > 0:

    X = X - len

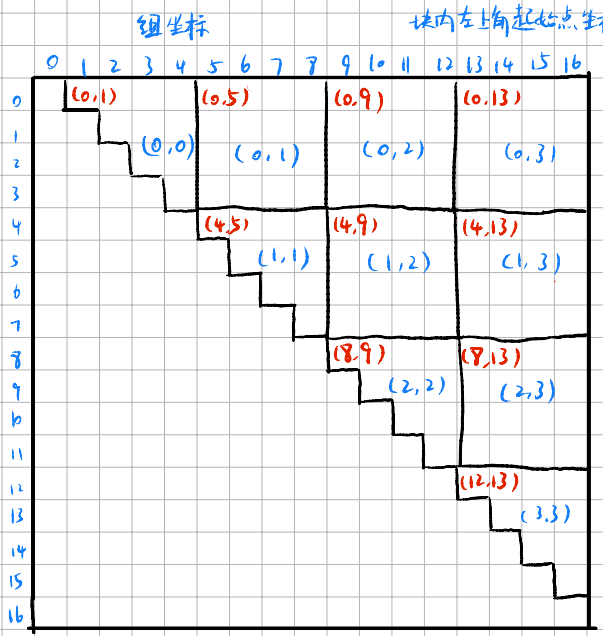
    if X > 0 : len -= 1 , layer += 1 ,

X = X + len # 最后一行剩的个数

group_id_x = layer

group_id_y = layer + x - 1

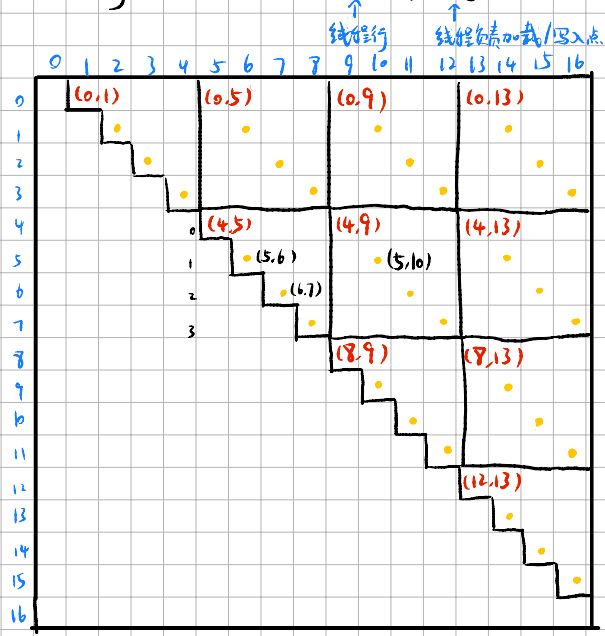
② (group_id_x, group_id_y) → (start_x, start_y)



start_x = BlockSize · group_id_x

start_y = BlockSize · group_id_y + 1

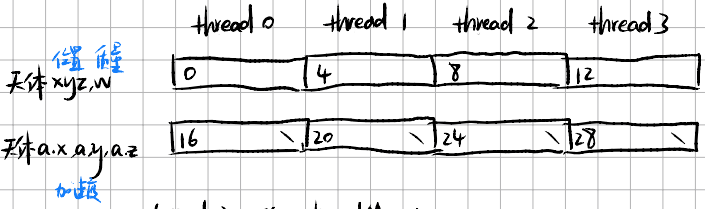
③ (start_x, start_y) & threadIdx ⇒ (point_x, point_y)



point_x = start_x + threadIdx.x

point_y = start_y + threadIdx.x

④ sharedMem: 2 × BlockSize 个元素位置，每个元素由4个float组成



(real*) mass = sharedMem;

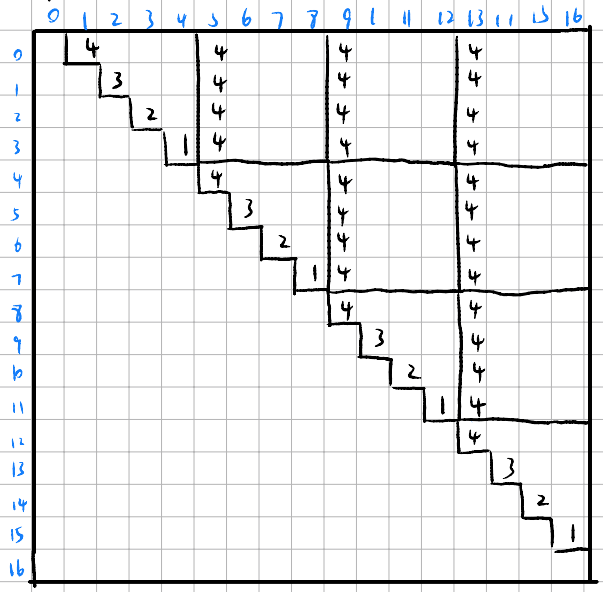
(real*) acc_sum_list = sharedMem + 4 * BlockSize;

对 thread i:

(real*) mass_pointer = mass + 4*i

(real*) acc_pointer = acc_sum_list + 4*i

⑤ 每个线程计算的格数量：



i) 不是对角块 ⇒ BlockSize

ii) 是对角块 ⇒ BlockSize − threadIdx

判断对角块：

start_y == start_x − 1

⑥ 每个线程起始计算点的 index

index = threadIdx

⑦ 循环 index 公式：

index' = (index + 1) % BlockSize

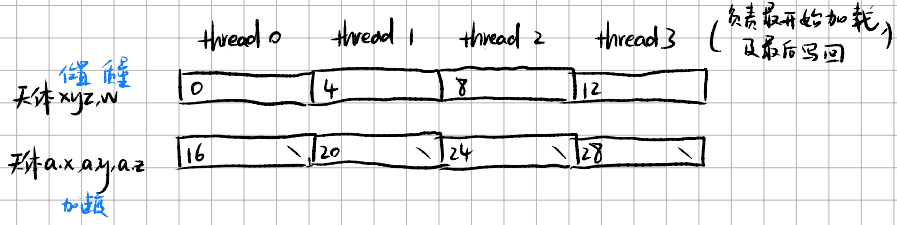
0 → 1
1 → 2
2 → 3
3 → 0

⑧ 函数内数据对应

threadId → 局部变量存 point_x 号天体的 xyzw

current_indx → mass + current_index ×4 对应的 xyzw

计算的序号

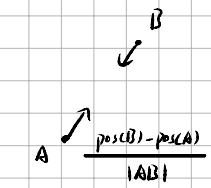


⑨ coff 计算：

$$\vec{F}_{(A受到来自B的加速度)} = \frac{G\frac{M_A M_B}{|AB|^2}\cdot\frac{\vec{AB}}{|AB|}}{M_A} = \frac{\vec{AB}}{|AB|^3}\cdot M_B$$

$$\vec{F}_{(B受到来自A的加速度)} = \frac{\vec{BA}}{|AB|^3}\cdot M_A = \frac{\vec{AB}}{|AB|^3}\cdot M_A\cdot(-1)$$

只算一次



$\vec{AB} = pos(B) - pos(A)$

⑩ coff 精度补偿：

$$coff = \frac{\vec{r}}{r^3} = \frac{1000.\vec{r}}{(1000.r)^3}\cdot 1000^2$$