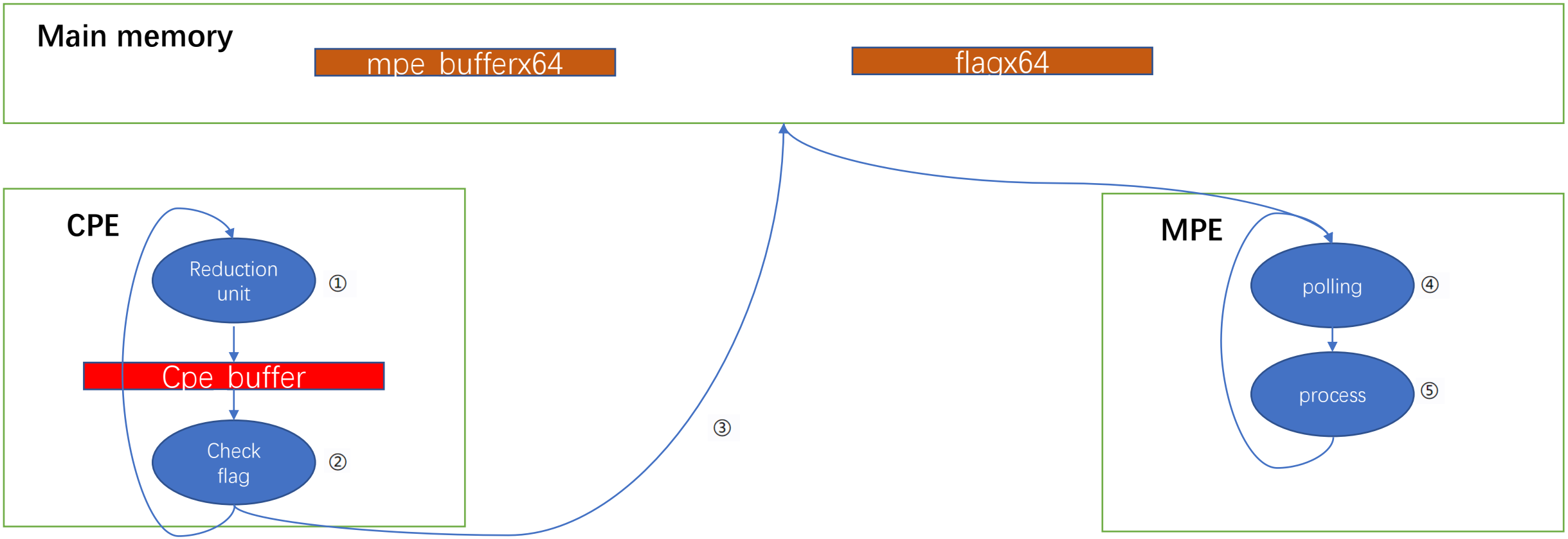
主核集中规约&小kernel DMA

罗海文

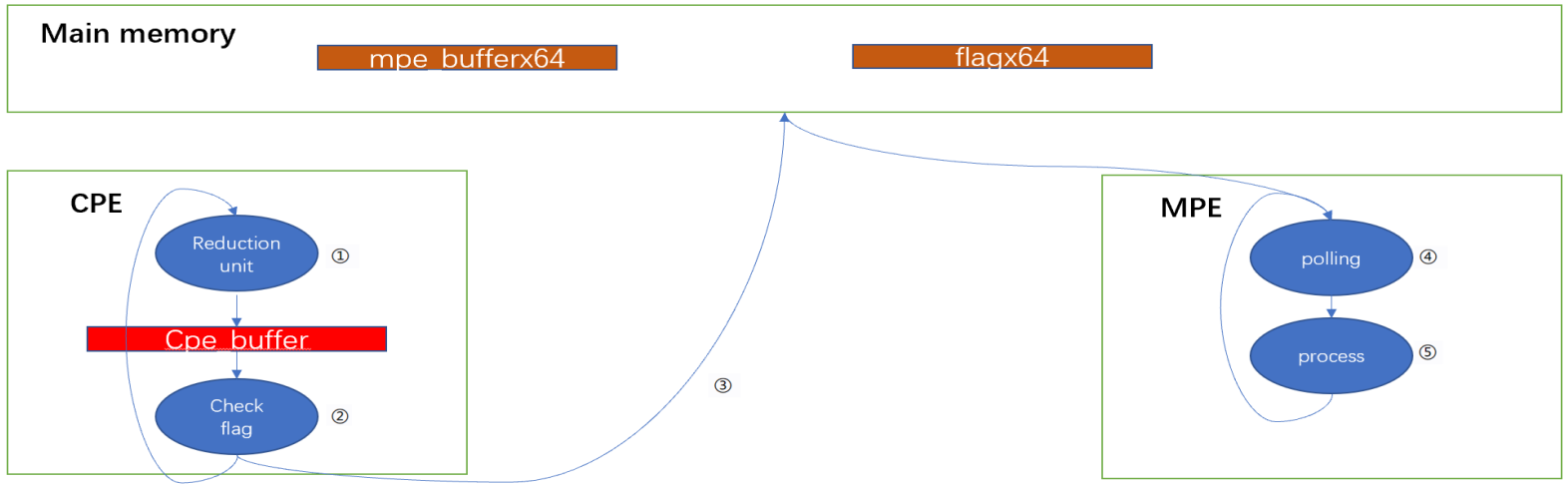
2021.10.18

主核集中规约-流程图



First_order_H[i_place]+=first_order_H_dense[i_compute][j_compute];

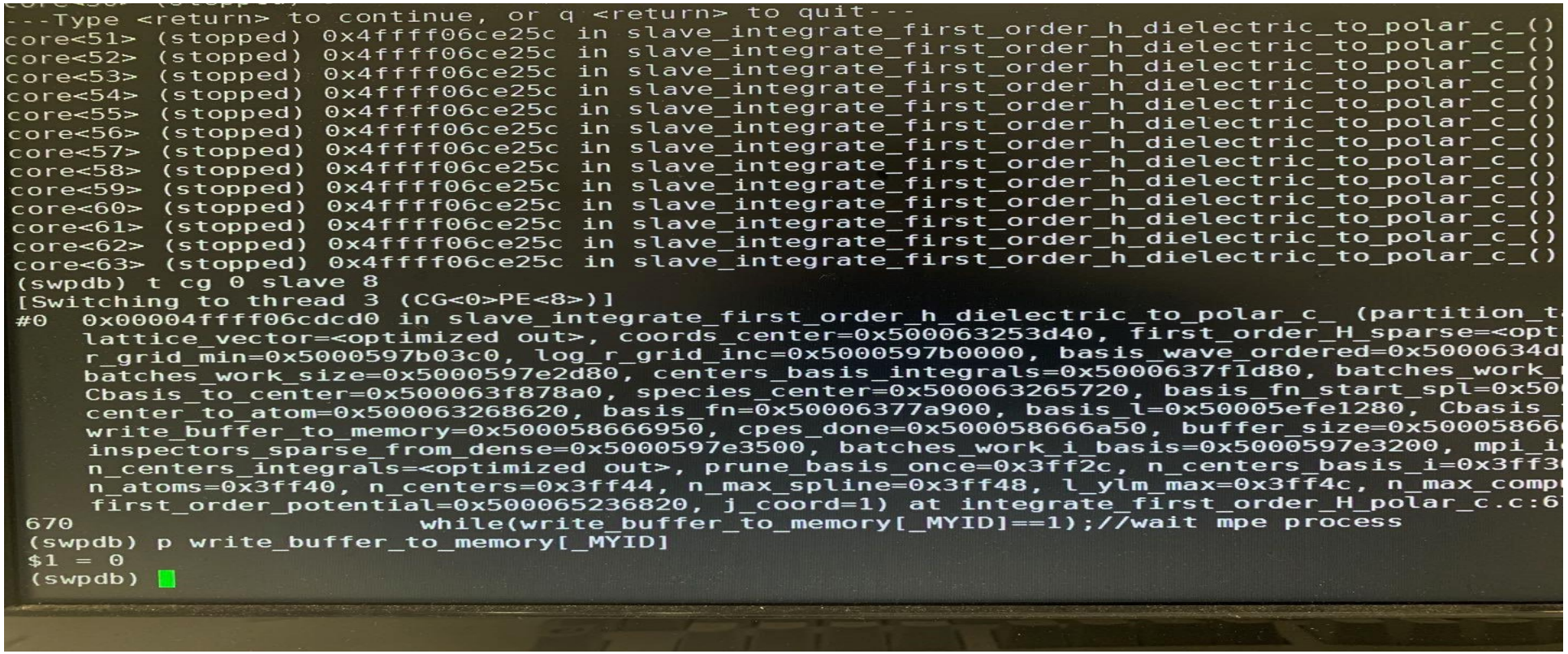
主核集中规约-文字描述



主存中申请的数组有两个，一个用于接收cpe buffer中的数据，由于有64个从核，所以大小是cpe_buffer的64倍；一个用于保存64个flag的值，flag用于从核与主核之间的同步。以cpe_id=0的从核为例，当flag[0]=0时，意味着主核已经处理完了该从核对应的buffer，因此该从核可以向主存中传输cpe_buffer中的数据，而主核在轮询时不进行规约处理；当flag[0]=1时，意味着该从核已经准备好了数据，主核可以进行规约处理。Cpe_buffer是ldm上的数组。

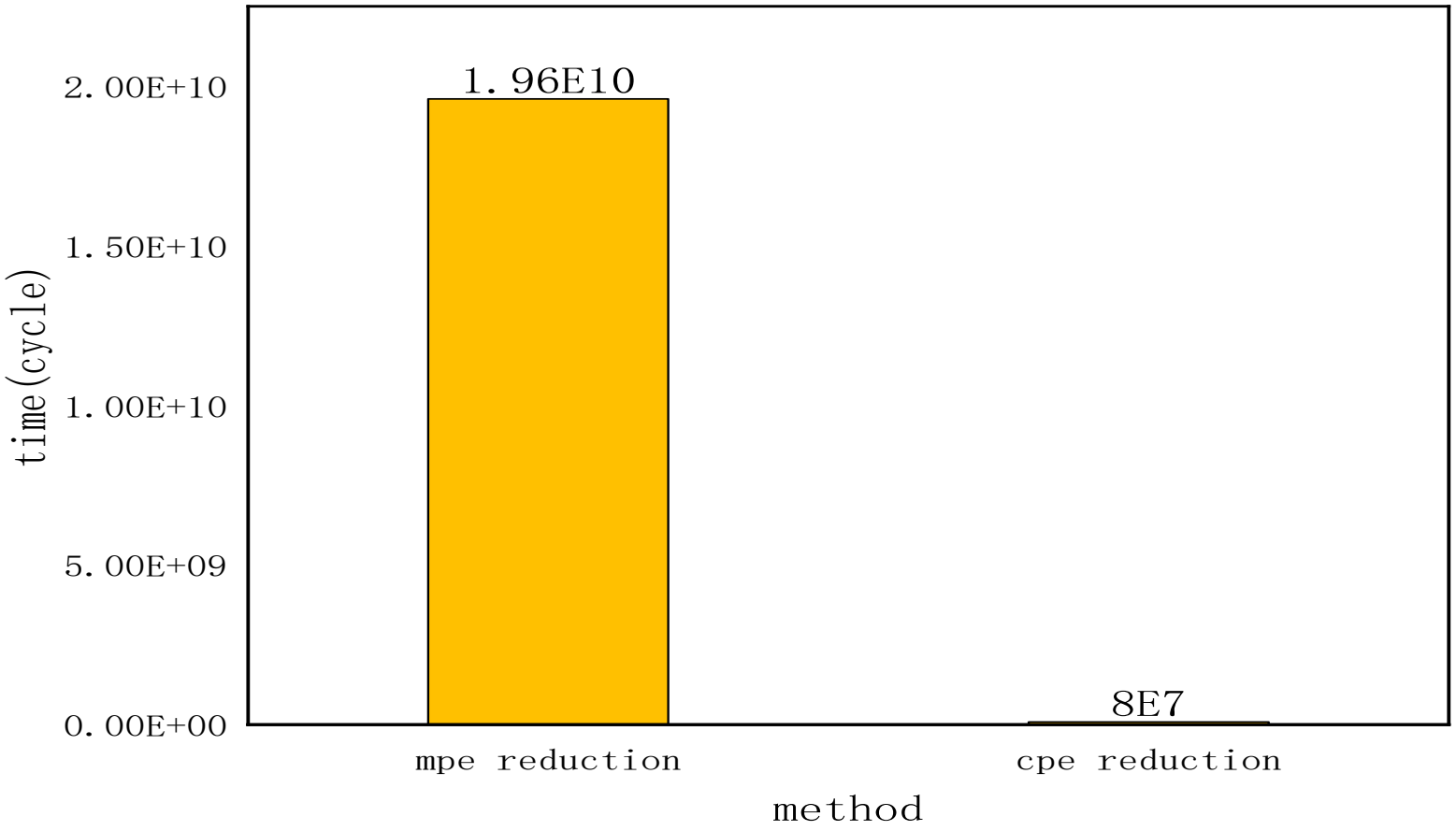
①从核的reduction unit获取i_place和first_ordre_H_dense[i_compute][j_compute]的值，暂存到cpe_buffer中
②当cpe_buffer满时，检查flag的值，如果flag=0，则③将cpe_buffer中的数据用dma的方式传输到主存，然后将flag置为1；如果flag=1，则等待flag的值变成0，然后执行③的操作。从核重复①②③操作。
④查看每个从核对应的flag的值，如果flag的值为1，则⑤执行规约操作，执行完成之后将flag置为0；如果flag的值为0，则不处理，进行在一轮的轮询。主核重复执行④⑤操作。

主核集中规约-遇到的bug



- 64个从核有两个卡住了，使用swpdb调试打印出flag的值是0，但是程序却卡在了while(flag==1);这条语句。

主核集中规约-单个从核性能提升

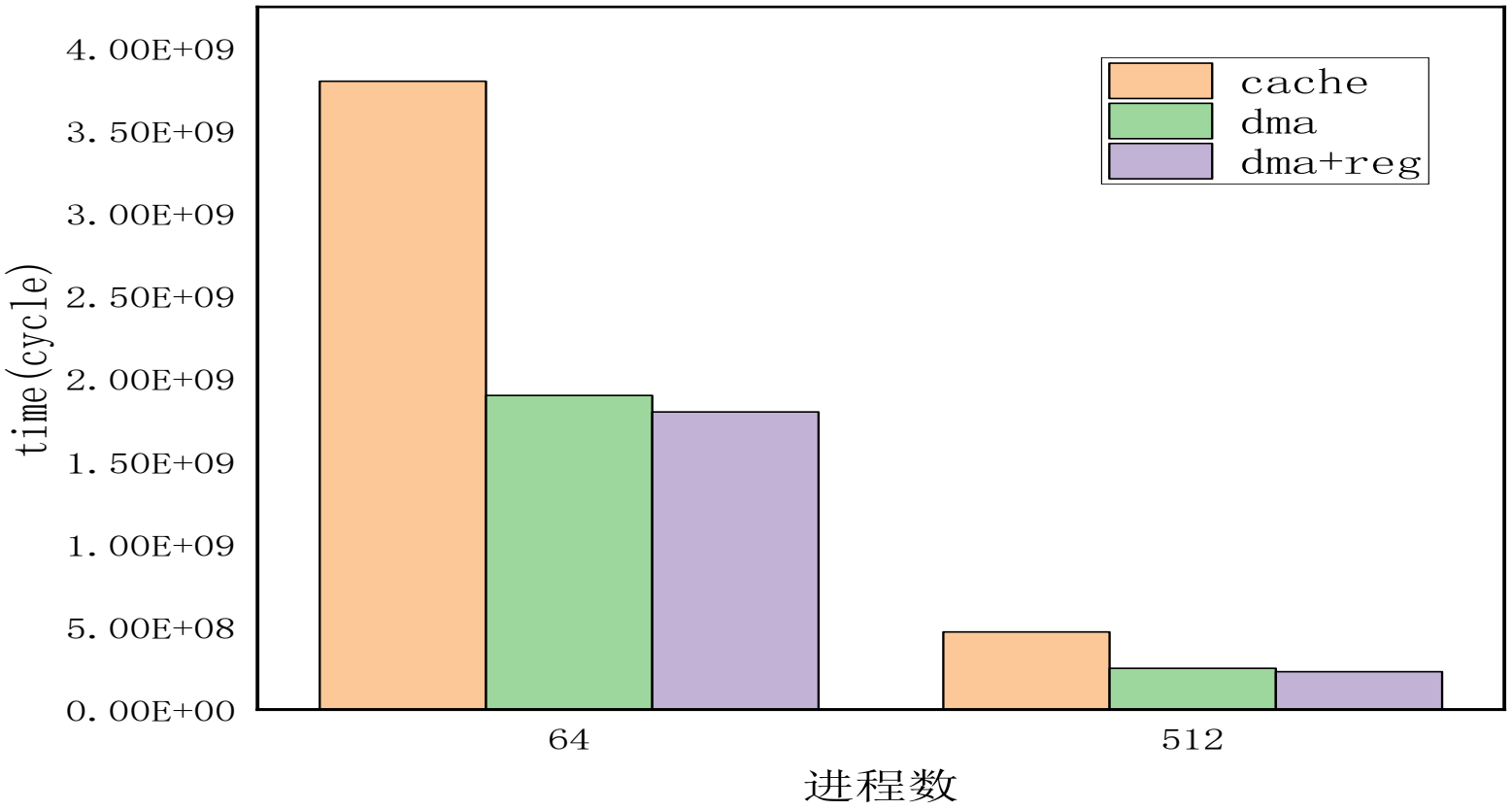


小kernel cache换DMA-tab_atom_centered_coords

```
for(int i_point=0;i_point<n_point;i_point++)
{
    for(int i_center=0;i_center<n_centers_integrals;i_center++)
    {
        dir_tab(0,i_center,i_point)=coord_current(0)-dir_tab_index(0,i_center);
        dir_tab(1,i_center,i_point)=coord_current(1)-dir_tab_index(1,i_center);
        dir_tab(2,i_center,i_point)=coord_current(2)-dir_tab_index(2,i_center);
        dist_tab_sq(i_center,i_point)=
                        dir_tab(2,i_center,i_point)* dir_tab(2,i_center,i_point)
                         +dir_tab(2,i_center,i_point)*dir_tab(2,i_center,i_point)
                         +dir_tab(2,i_center,i_point)*dir_tab(2,i_center,i_point);
    }
}
```

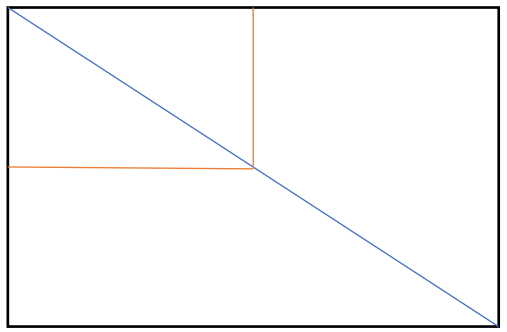
- Loop tiling
  - ■外层循环通过调度把任务分配给从核
  - ■对内层循环进行分块，由于内层循环中需要访问数组dir_tab,dir_tab_index,dist_tab_sq的大小都是n_centers_integrals，所以分块的大小为MAX_ALLOCATABLE/5
- 充分利用寄存器
  - ■每一次内层循环dir_tab都需要写入一次和读入一次，可以先把dir_tab的值存在寄存器中，写入dir_tab之后需要读入时可以直接读寄存器中的值

小kernel cache换DMA-tab_atom_centered_coord性能提升



小kernel cache换DMA-prune_density_matrix_sparse

- **loop tiling**
  - 第一层循环通过调度把任务分配给从核
  - 每一个任务中i_basis_index可以一直保留在ldm中
  - 对column_index和density_matrix_sparse进行分块，分块大小为i_start-i_end。
- **对称矩阵的写入**
  - 先把结果暂存在ldm中，然后通过dma写入内存，列使用跨步dma，行使用连续dma



```
for(i=0;i<n_compute_c;i++)
{
   i_basis=i_basis_index(i);
   i_start=index_Hamiltonian(1,1,i_basis);
   i_end=index_Hamiltonian(2,1,i_basis);

   for(j=0;j<i;j++)
   {
      j_basis=i_basis_index(j);
      for(k=i_start;k<i_end;k++)
      {
         if(column_index(k)==j_basis)
         {
            density_matrix_con(i,j)=
                  density_matrix_sparse(k);
            density_matrix_con(j,i)=
                  density_matrix_sparse(k);
         }
      }
   }
}
```

小kernel cache换DMA-prune_density_matrix_sparse性能提升