大kernel和小kernel比较

罗海文

2021.09.27

小kernel

- 寻找热点函数->写kernel函数利用从核加速
- 一个大的程序段中可能有多个热点函数，便会有多个kernel

大kernel

- 相当于kernel fusion
- 将多个kernel和没上从核的程序段合并成一个kernel,如果多个kernel相距比较近并且有数据依赖的话。
- 优点：1.较少了反复启动从核的开销。2.较少了数据移动，一些频繁访问的数据可以一直保留在从核ldm中而不需要反复在从核ldm和主存之间移动。3. …

大kernel rho和小kernel rho的比较

rho

	大kernel	小kernel	原Fortran加xmath	原Fortran不加xmath
64核	15.11	17.61	27.35	239.60
512核	2.50	2.24	3.38	31.45

rho各个部分的比较（512核大kernel负载不均衡所以目前只比较64核）

	tab_atom	prune_density	i_index_loop	dgemm	matrix_ddot	collect_batch
64_小kernel	34亿	82亿	171亿	1.0亿	0.15亿	1.4亿
64_大kernel	25亿	64亿	211亿	3.6亿	0.24亿	4.8亿

1.512核RBD大kernel负载不均衡，运行时间最长的从核为2.50秒，运行时间最短的从核为1.8秒，所以512核是小kernel更好。
2.64核rho每个部分用时对比：各有优劣，整体上大kernel更好。

大kernel H和小kernel H的比较

H

	大kernel	小kernel	原Fortran加xmath	原Fortran不加xmath
64核	85.92	21.10	35.28	131.02
512核	13.63	3.15	4.86	17.47

H各个部分的比较

	tab_atom	update first_order_H	i_index_loop	dgemm+ddot	collect_batch
64_小kernel	34亿	9.8亿	263亿	9.8亿	1.4亿
64_大kernel	25亿	1372亿	90.8亿	3.7亿	2.4亿

1.大kernel中由于update first_order_H这个部分64个从核有数据冲突，所以用时明显高于小kernel的相应的部分。
2.除了update first_order_H，其他的部分加起来大kernel比小kernel好

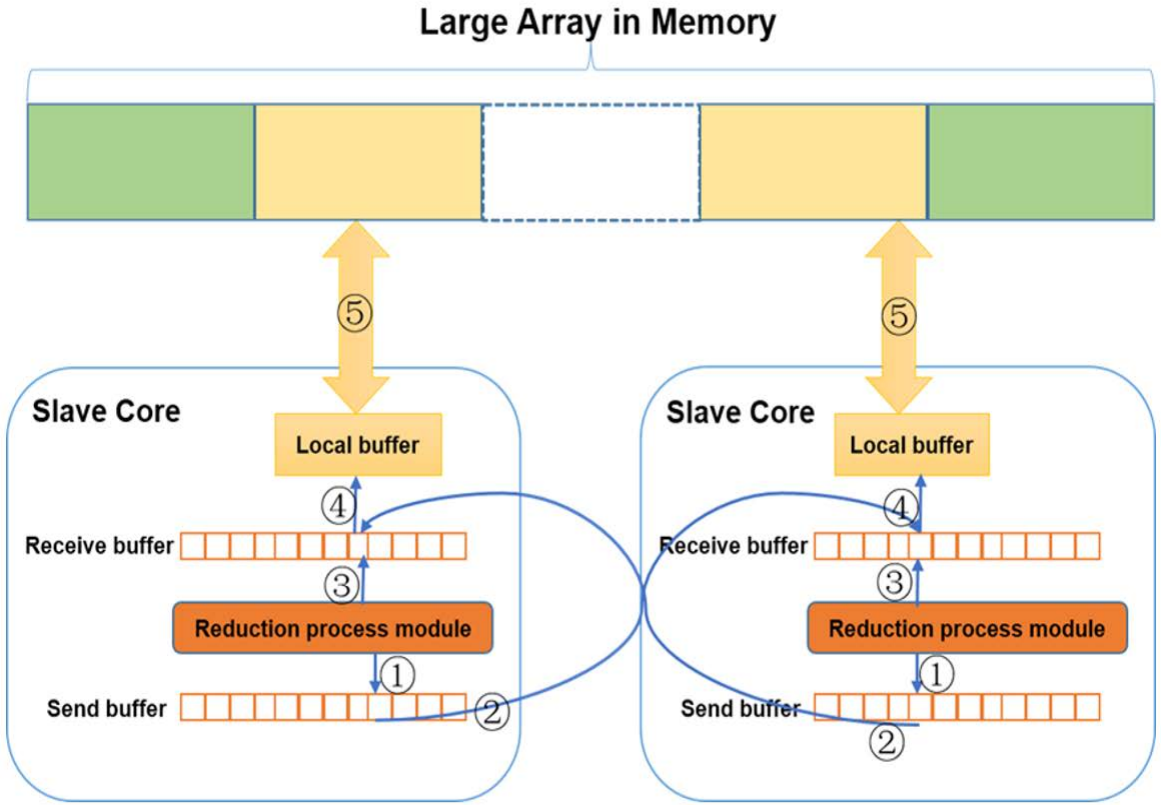
update first_order_H

```
for(int i_copy = 0; i_copy < n_sparse;i_copy++)
{
   int i_place=inspectors_sparse_from_dense(3*i_copy,i_my_batch)-1;
   int i_compute =inspectors_sparse_from_dense(3*i_copy+1,i_my_batch)-1;
   int j_compute = inspectors_sparse_from_dense(3*i_copy+2,i_my_batch)-1;

   first_order_H_sparse(i_place) = first_order_H_sparse(i_place) + first_order_H_dense(i_compute,j_compute);
}
```

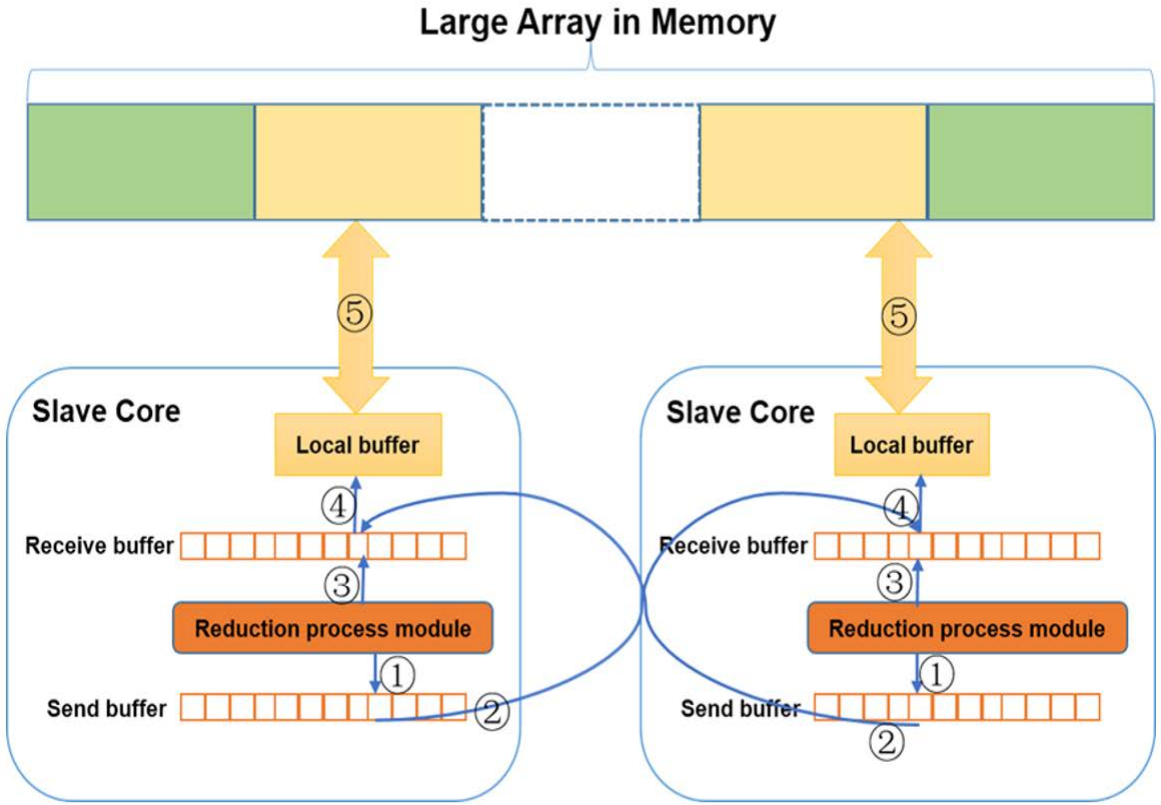
64个从核在执行这段代码时i_place可能会相同从而有数据冲突

Openacc-分布式规约



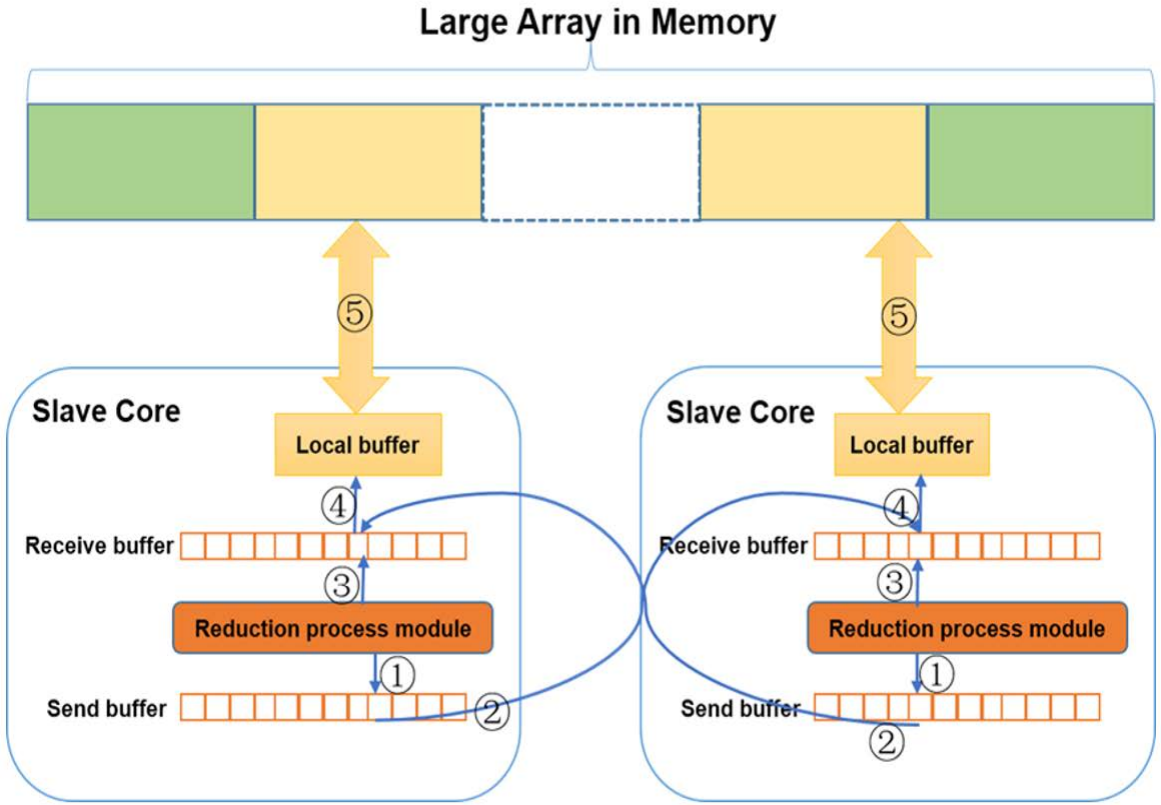
Step1：根据当前的i_place值，计算出当前要更新的元素位置落在哪个从核管理的数据区间（假定目的从核ID为M），然后先将规约值及规约操作符缓存到本地对应从核的发送缓冲单元SM。

Openacc-分布式规约



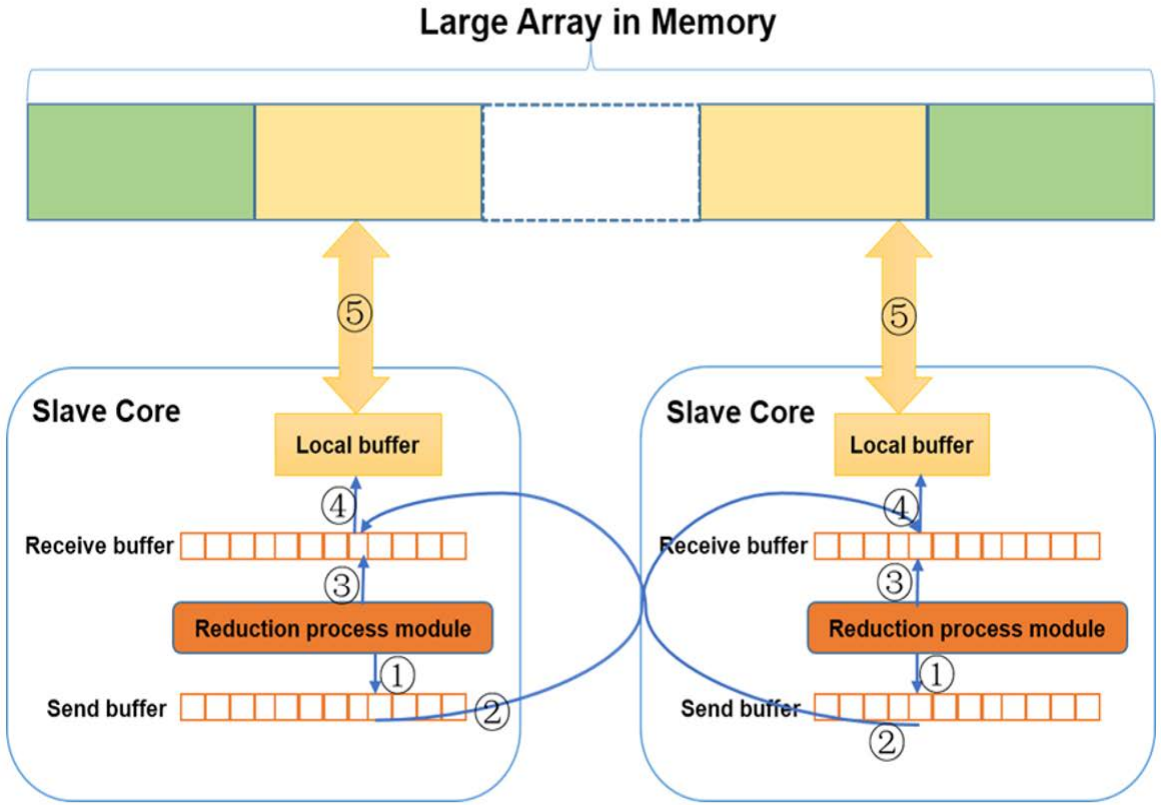
Step2：检查发送缓冲单元SM是否已满，如果已满，则将SM的内容使用RMA发送给从核M，同时重置该发送缓冲单元SM。

Openacc-分布式规约



Step3：轮询处理自己的接收缓冲单元，如果发现接收缓冲单元RN存在有效数据，则解析出该接收缓冲单元的规约值、规约位置等信息。

Openacc-分布式规约



Step4和step5：如果发现该规约位置的原数据已经缓冲到buf中，则直接执行更新操作，否则先将缓冲buf数据刷回主存，然后使用DMA将需要的数据片段缓冲到buf中。

Openacc结果

- 512RBD处理30万个数据用了200亿个时钟周期，平均处理一个数据需要6666个时钟周期，远高于gld操作所需要的时钟周期数。

可能的解决办法

- 用更细粒度的锁：512核中平均每个不同的i_place只有4个从核有冲突，可以在first_order_H中每个元素都加锁。
- 主核集中式规约，可以比从核规约少很多rma和dma操作。