稀疏矩阵与向量乘法的优化

实验环境

- OS：Linux version 3.10.0-862.9.1.el7.x86_64(CentOS)
- CPU：Intel(R) Xeon(R) CPU E5-2620 v4 @ 2.10GHz
- GPU：NVIDIA Corporation GP102 [GeForce GTX 1080 Ti] (rev a1)
- CUDA：Release 8.0, V8.0.61

问题描述

- 1、问题：稀疏矩阵（n/n）与向量（n/1）相乘
- 2、矩阵大小为256或1024
- 3、稀疏矩阵稀疏度约为80%，每行非0数据数目相同

优化步骤

一、CPU实现

- 直接采用CPU实现该该算法
- 不使用优化编译选项时，**1024阶矩阵重复1000次运算**（以下均在该条件下测试）总耗时为**4300ms**。
- 使用O3优化后，运算时间降低至**1240ms**。

二、GPU简单加速

- 初始想法为直接采用GPU加速，每个thread处理矩阵中的一行与向量相乘，如此实现的时间复杂度为O(N)。
- 为了实现合并访问，需在将矩阵转置后再进行运算，实现后大幅增加运算速度
- 直接使用GPU加速后的运算时间降低至**125.08ms**。

三、使用shared memory加速

- 由于对于向量的使用极为频繁，在整个计算中会重复使用N次，故考虑将向量从global memory取至shared memory中进行加速
- 使用shared memory加速后的运算时间降低至**70.59ms**。

四、使用CSR压缩稀疏矩阵加速

- 由于矩阵在条件中为一稀疏矩阵，故考虑采用稀疏矩阵的压缩表示方法CSR来对矩阵进行压缩，便于后面的矩阵乘运算。
- CSR压缩后得到稀疏矩阵的三元组，但由题设的特殊条件（每行非0数据数目相同）可知在知道稀疏度的情况下，可以直接省略行数组，只需要一个列数组即可通过固定的间隔（每行的非0元素数目）来得到所有稀疏矩阵的位置，故将三元组压缩为二元组。
- 对现在的二元组重复上述shared memory的优化并写成GPU程序，即得时间复杂度为O(S)的程序（S为每行非0元素数目）。
- 使用CSR压缩加速后的运算时间降低至**29.22ms**。

五、使用register存储压缩后的矩阵

- 当完成对稀疏矩阵的压缩后，在矩阵不大时，可考虑将矩阵直接移入register中存放（由于在

实际应用背景中，matrix多为一固定矩阵，而vector为一常变向量，故考虑将矩阵一次性放入每个thread的register中，在后续的计算中只改变vector的值），而从register中取值的速度远大于从主存中取值，因此在矩阵不大时，该优化的提升空间也很大。

- 由于该优化与矩阵的大小息息相关，在矩阵较小时有着良好的性能，其性能分析与性能对比图像在后面给出。

六、对比Cublas库实现结果

- 直接由cublas库实现的该问题时间为**0.0512ms**次（不统计*1000*次的原因是进行重复计算时该库函数的用时大幅减少导致单次运算的结果不准确）,与前者最佳的优化结果*0.0499ms*次相比几乎相似，但库内自主处理了矩阵转置等问题，故整体上可能cublas库仍小有优势。
- 加入了register的优化后，在register刚好能装下矩阵的情况下，优化结果明显好于cublas库；但在矩阵较大时，仍是cublas库更具有优势。

优化结果

1、优化程序的输出结果

- 运行矩阵（1000次，不包含cublas对比）：

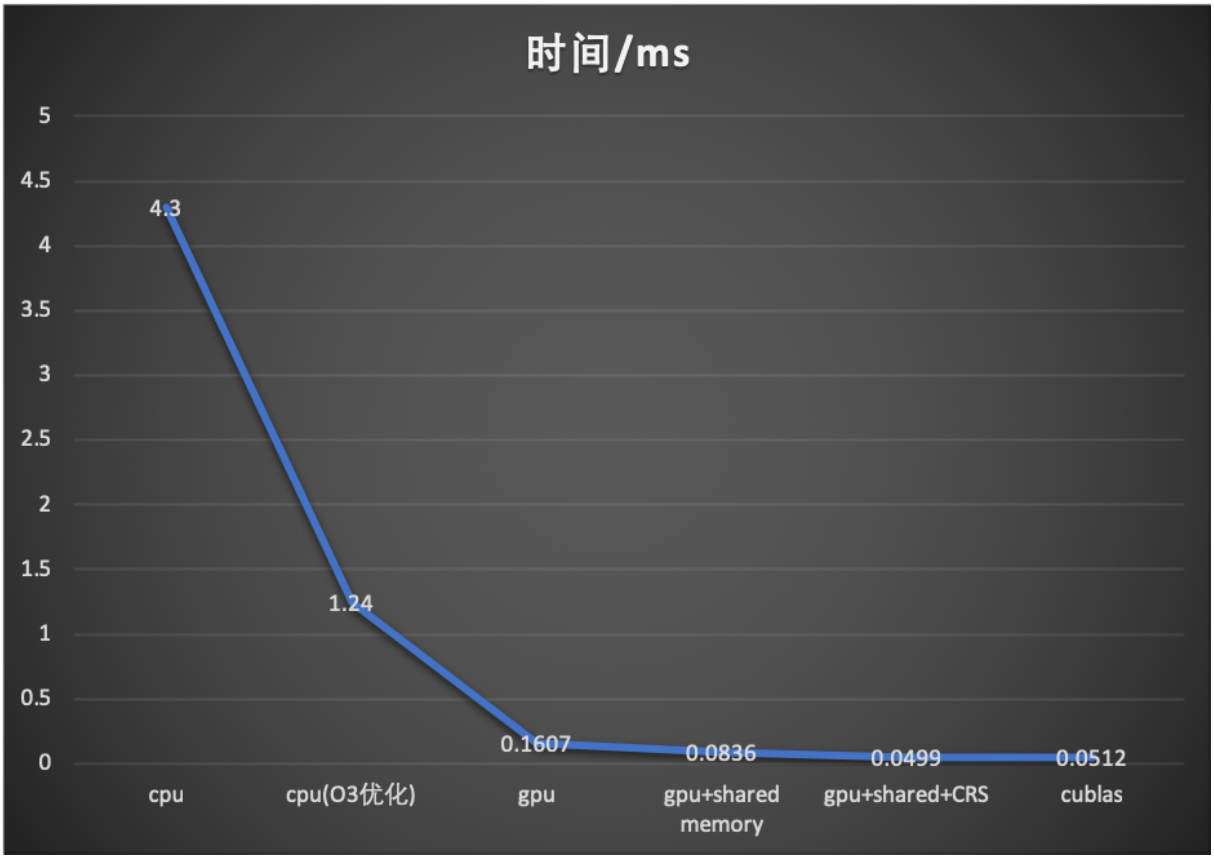
```
[vgpu@gpu8 matrixmul]$ ./fa
The total cpu run time is 1240.000000 ms.
The total gpu run time is 125.076477 ms.
The total gpu (use shared memory) run time is 70.594208 ms.
The total gpu (using csr and shared memory) run time is 29.224993 ms.
1PASSED!
2PASSED!
3PASSED!
```

- 执行一次（1次，包含cublas的对比）：

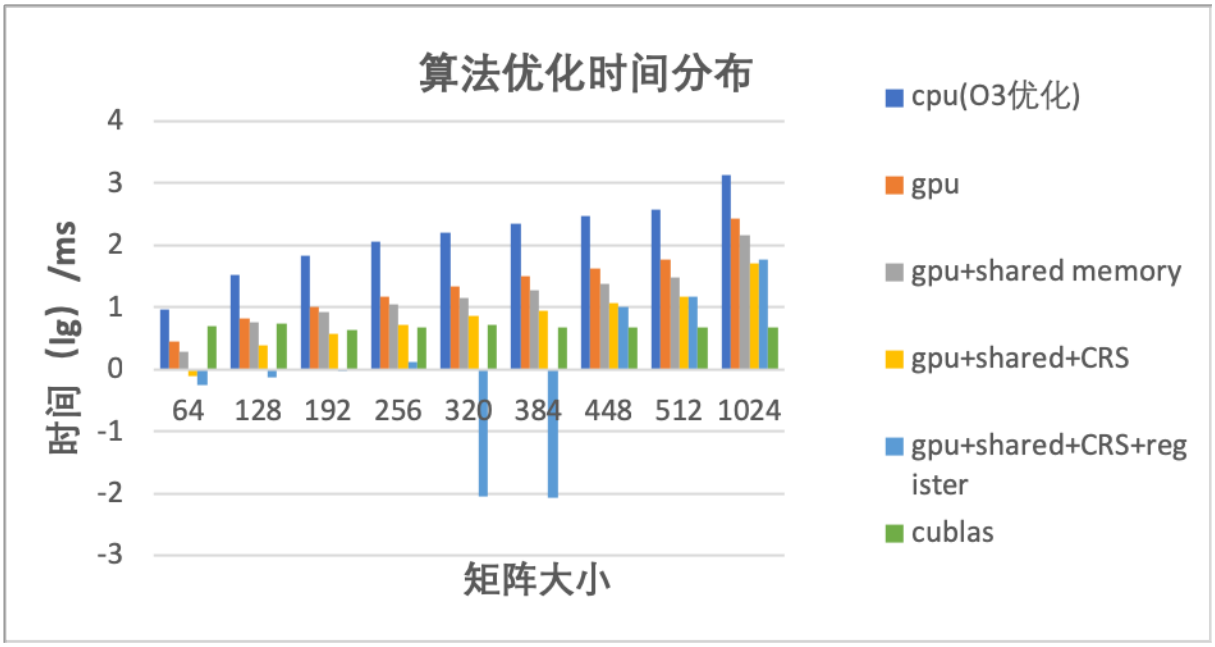
```
[vgpu@gpu8 matrixmul]$ ./f
The total cpu run time is 0.000000 ms.
The total gpu run time is 0.160736 ms.
The total gpu (use shared memory) run time is 0.083616 ms.
The total gpu (using csr and shared memory) run time is 0.049952 ms.
 ** On entry to SGEMM  parameter number 8 had an illegal value
The total gpu (using cublas) run time is 0.051200 ms.
Test correct:
1PASSED!
2PASSED!
3PASSED!
4PASSED!
```

2、整体优化结果曲线图

- （1）未加入Register优化前（取1024阶矩阵重复1000次运算）可得性能为：



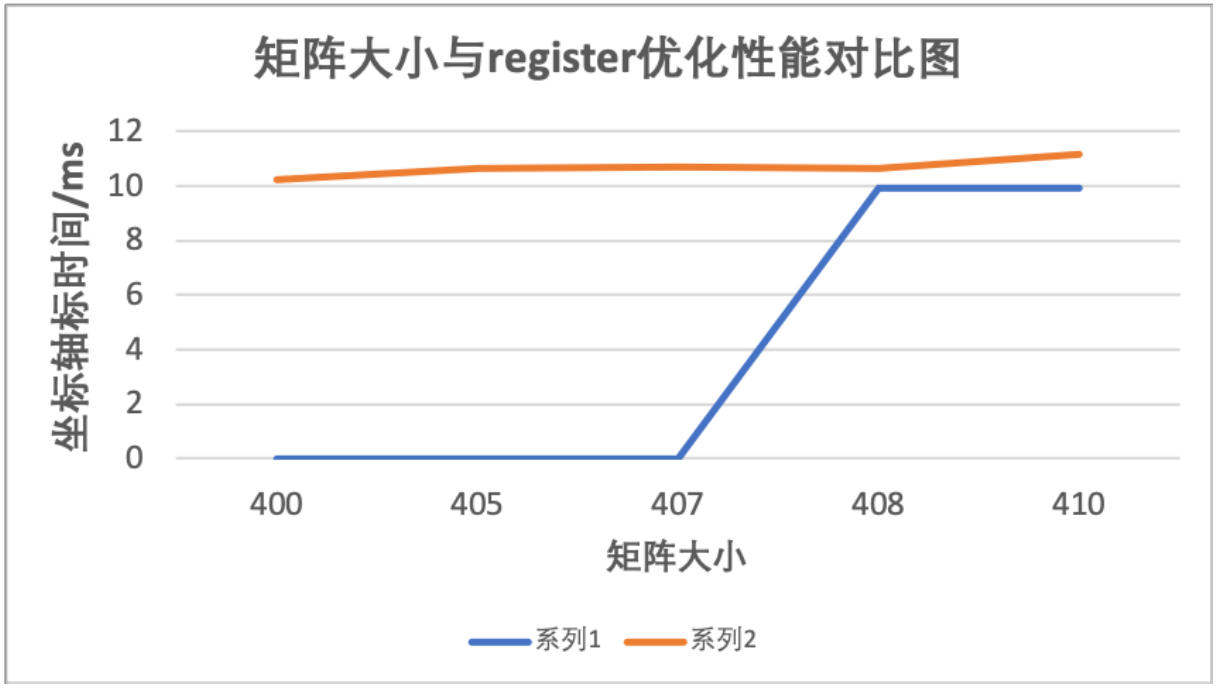
- （2）加入Register后，由于该优化与矩阵大小关系较大，因此取矩阵大小为64~1024的矩阵分别测试性能，最终得到以下性能图：（时间已取log10）



性能分析

- 对于本次优化的性能分析主要对于Register优化给出，由于register的数量有限，故在实际矩阵较大时，可能压缩过的CSR矩阵不完全能放入register中，此时会该部分矩阵仍会存储在主存中，因而使得优化力度减小。
- 由优化结果中的曲线图，对矩阵大小为384~448的运行结果观察不难得知，register数量的最大值即出现在该范围内，可以观察到在448及更大的矩阵运算时，register优化的效果逐渐降

低甚至几乎为0。因此对384~448段的矩阵做更为精确的分析，可得到以下图像，其中红色线代表只做到CSR优化时的时间消耗，蓝色代表加上register优化后的时间消耗：



- 由蓝色线条在408处的迅速上升可知，407即为当前GPU下使用该方法能给单个thread分配最多register的矩阵大小，超过407大小的矩阵部分可能被移交主存（甚至在移交后不再大部分只用register存储，因而导致性能的提升微乎其微甚至下降）。
- 更多对测试数据的观察与编译过程中cuda使用register的记录可在test.xlsx文件中查看。