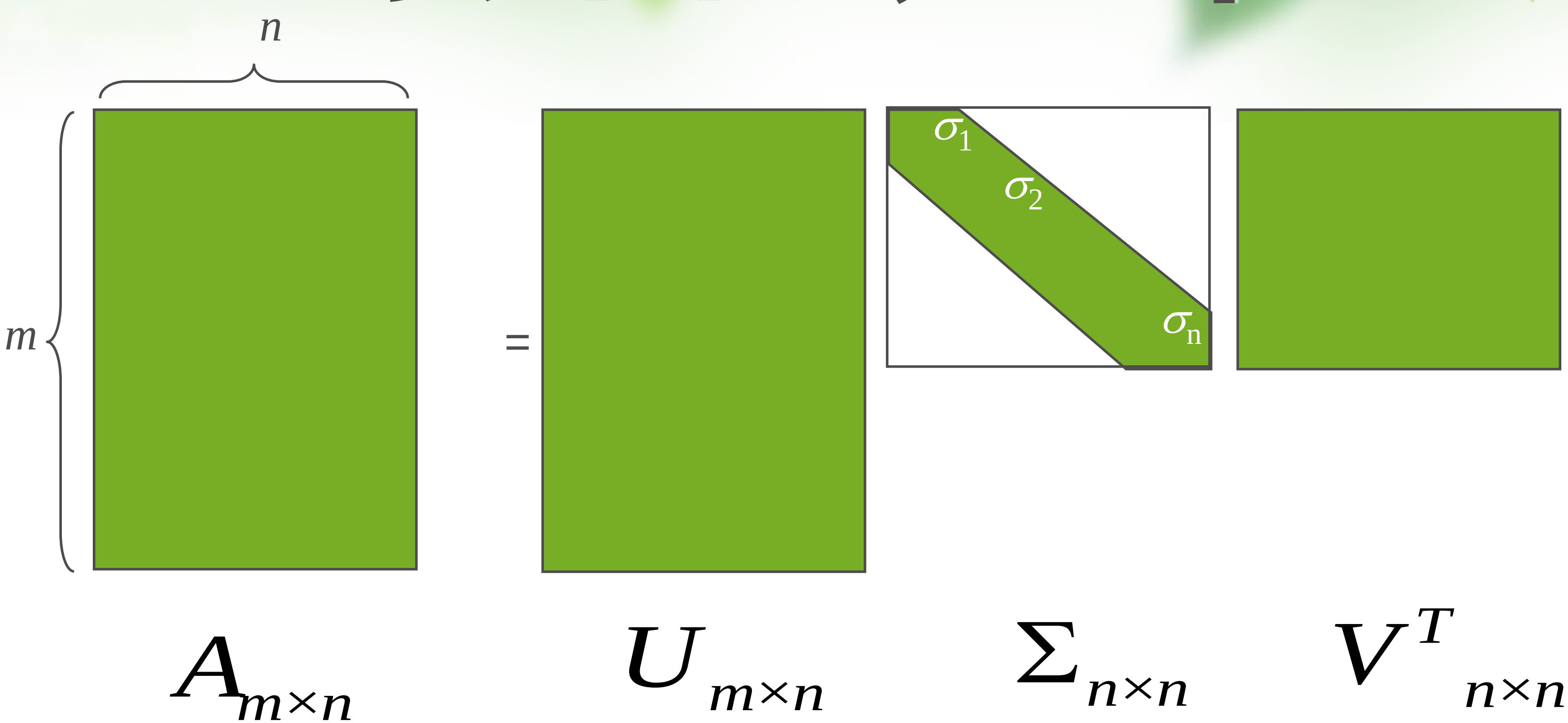
在CUDA中实现奇异值分解算法的提速

答辩人：　　赵　佳
指导教师：阮吉寿

目录

奇异值分解



$$A = U\Sigma V^T$$

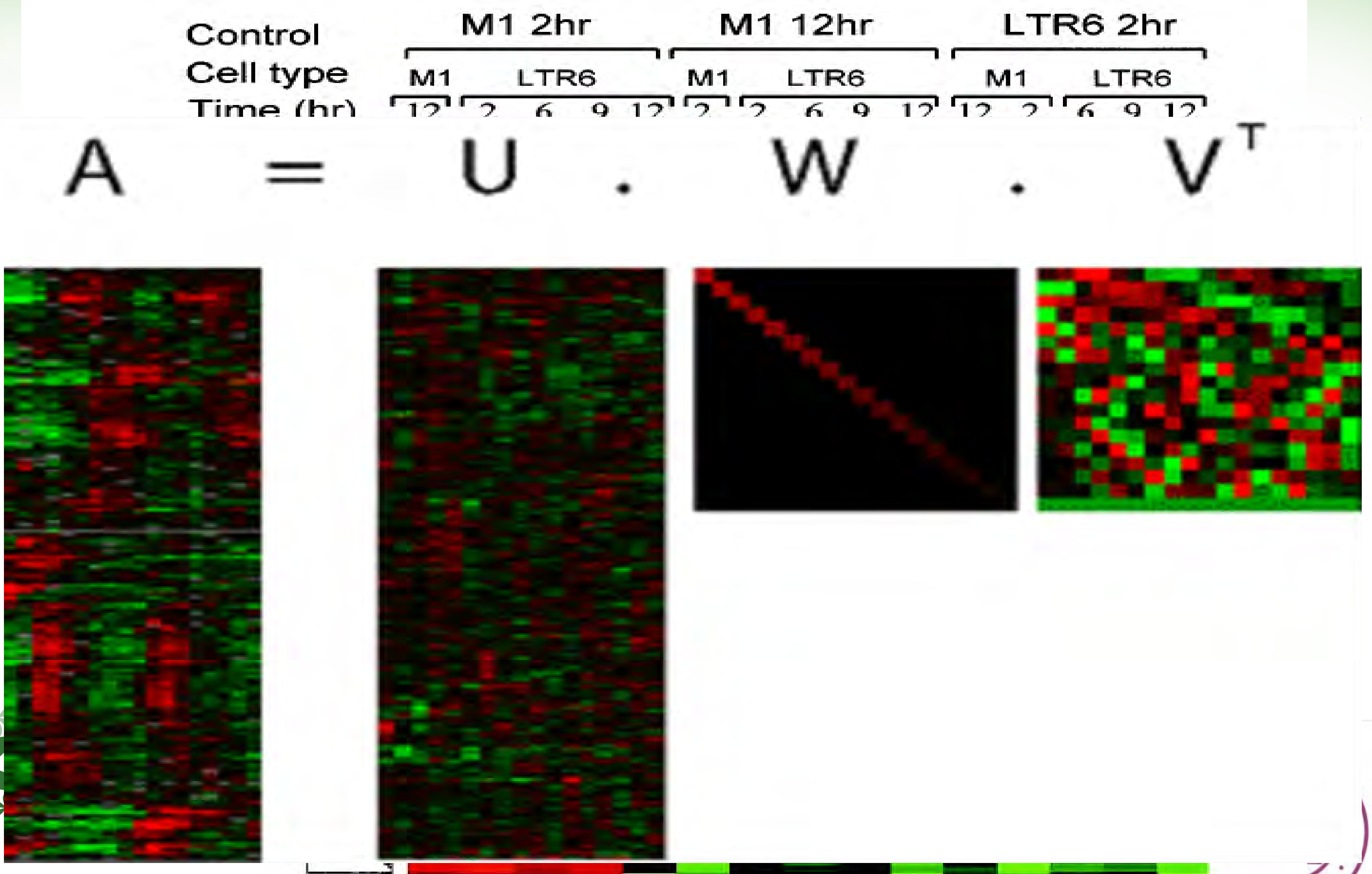
A： m×n 高矩阵
U： m×n 正交矩阵
V： n×n 正交矩阵
Σ： n×n 对角矩阵

奇异值分解应用领域

奇异值分解

- 模式识别
- 信号处理
- 数据压缩
- 主因素分析
- 系统控制
- 最优化问题
- 最小二乘问题
- 特征值问题
- 广义逆
- 最佳近似矩阵

奇异值分解应用：基因芯片



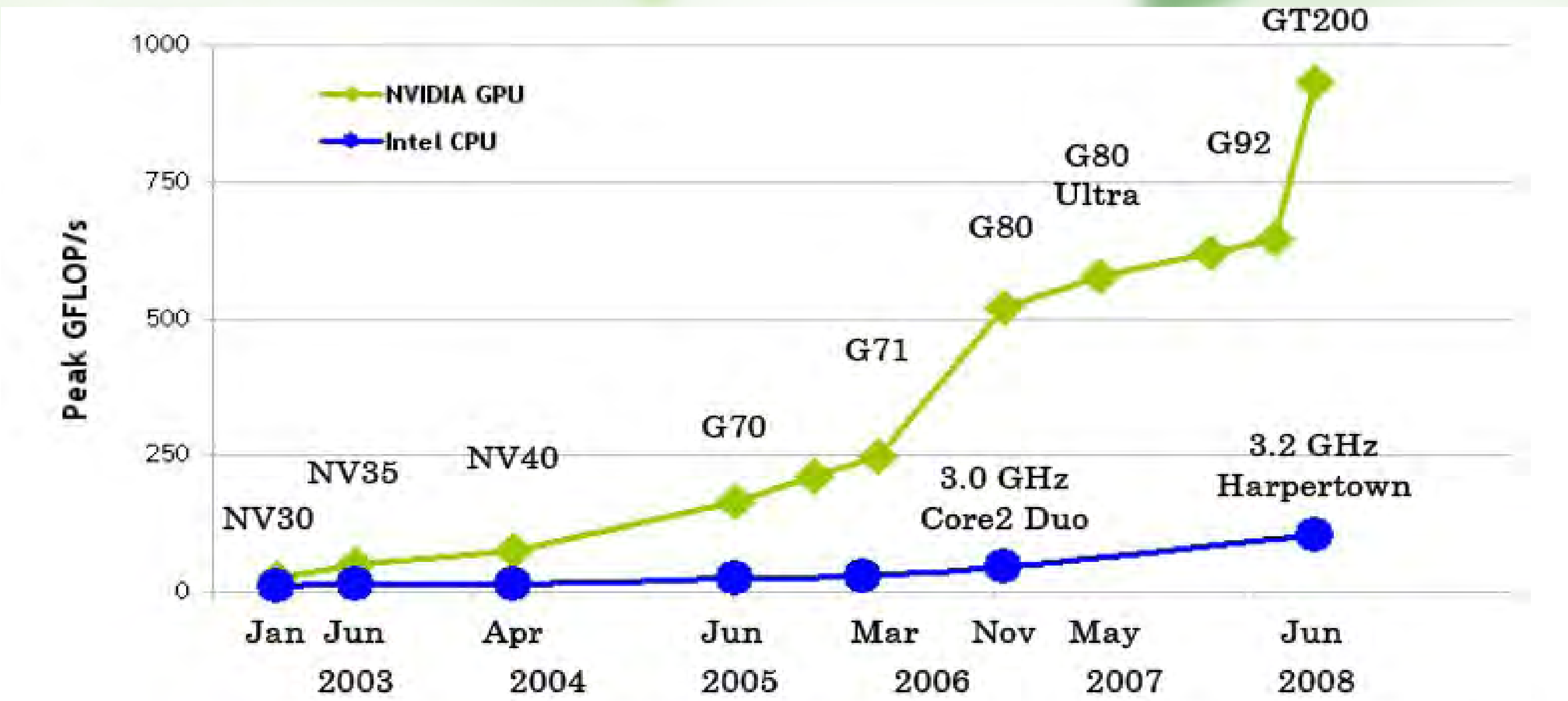
奇异值分解在实际应用中的瓶颈

即便是目前最快的分解算法，

算法复杂度 $O(m*n^2)+O(m^2*n)$

$$A_{m\times n}=U_{m\times n}\Sigma_{n\times n}V^T_{n\times n}$$

CPU与GPU浮点运算能力比较



GT200 = GeForce GTX 280

G92 = GeForce 9800 GTX

G80 = GeForce 8800 GTX

G71 = GeForce 7900 GTX

G70 = GeForce 7800 GTX

NV40 = GeForce 6800 Ultra

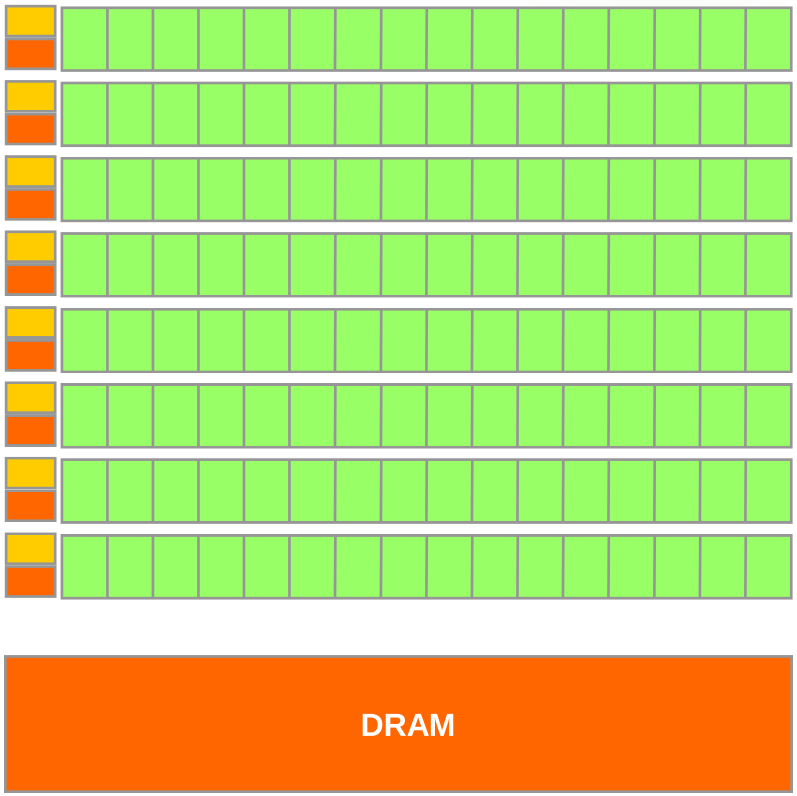
NV35 = GeForce FX 5950 Ultra

NV30 = GeForce FX 5800

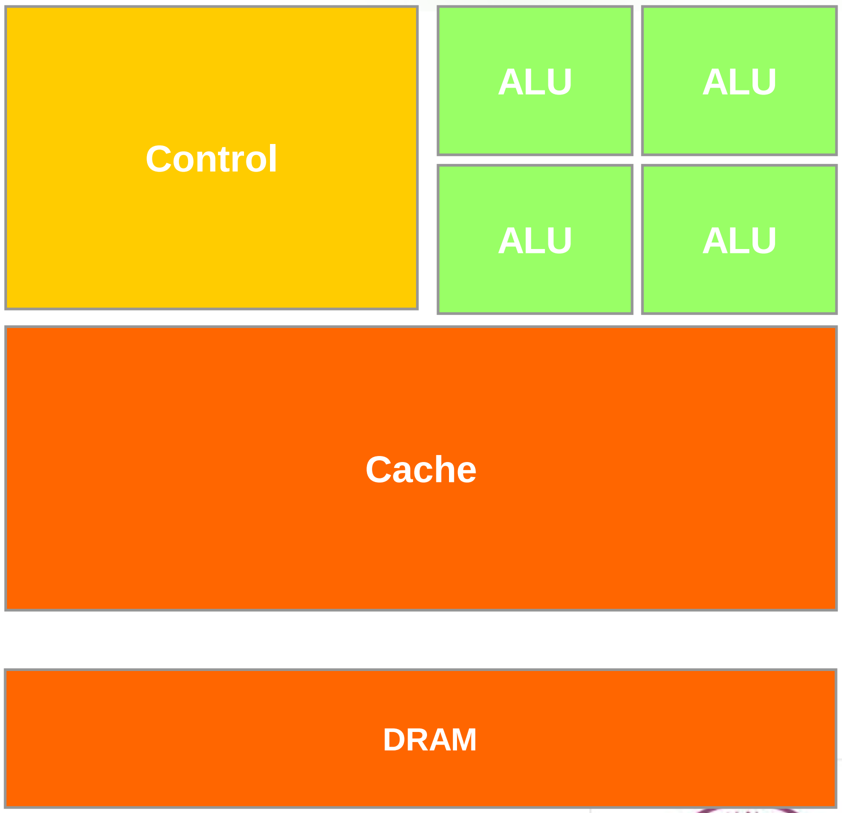
南开之星


TOP500
CERTIFICATE
3.06 GHz, Myrinet
IBM
China
is ranked
No. 42
among the world's TOP500 Supercomputers with
3.23 TFlop/s Linpack Performance
TOP500 list was published at the ISC2004 Conference in Heidelberg, Germany,
June 23rd, 2004
Congratulations from The TOP500 Editors
Hans Meuer
University of Mannheim
Erich Strohmaier
NERSC/Berkeley Lab
Jack Dongarra
University of Tennessee
Horst Simon
NERSC/Berkeley Lab

CUDA架构下GPU与CPU的差异



CUDA架构下GPU



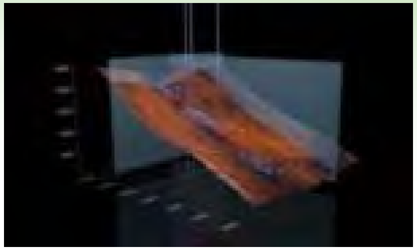
CPU

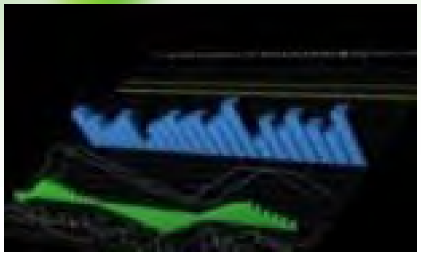
Over 200 Universities Teaching CUDA

UIUC
MIT
Harvard
Berkeley
Cambridge
Oxford
…

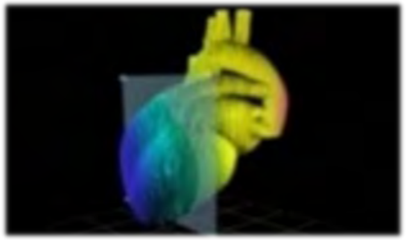
IIT Delhi
Tsinghua
Dortmundt
ETH Zurich
Moscow
NTU
…

CUDA Applications


Oil & Gas

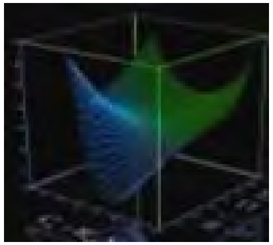

Finance


CFD

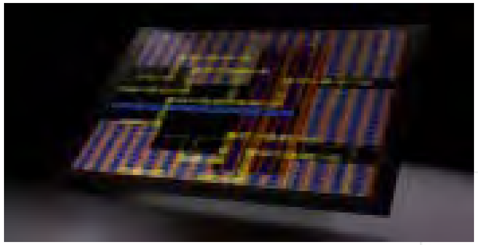

Medical


Biophysics


Imaging


Numerics


DSP


EDA

ISI Web of Knowledge℠ Take the next step

All Databases | Select a Database | Web of Science | Additional Resources

Results

Save History

2009.6

0	Topic=(svd) AND Topic=(cuda) *Timespan=All Years*
3,134	Topic=(cuda) OR Topic=(nvidia) OR Topic=(gpu) *Timespan=All Years*
12,586	Topic=(Singular value decomposition) OR Topic=(svd) *Timespan=All Years*

ISI Web of Knowledge℠

Take the next step

All Databases | Select a Database | Web of Science | Additional Resources

Results | Save History | Open Saved History

2010.5

2	Topic=(SVD) AND Topic=(cuda) *Timespan=All Years*
15,738	Topic=(Singular Value Decomposition) OR Topic=(SVD) *Timespan=All Years*
5,191	Topic=(cuda) OR Topic=(nvidia) OR Topic=(gpu) *Timespan=All Years*

ISI Web of Knowledge℠

Take the next step

All Databases | Select a Database | Web of Science | Additional Resources

Results Topic=(SVD CUDA)

Timespan=All Years.

2010.5

Results: **2**

Page 1 of 1 Go

Sort by: Publication Date

Print | E-mail | Add to Marked List | Save to EndNote Web

Save to EndNote, RefMan, ProCite | more options

Analyze Results

Refine Results

Search within results for

Search

General Categories Refine

SCIENCE & TECHNOLOGY (2)

Subject Areas Refine

COMPUTER SCIENCE (2)

MATHEMATICS (1)

more options / values...

Document Types

Authors

Source Titles

1. Title: Singular Value Decomposition on GPU using CUDA
Author(s): Lahabar, S; Narayanan, PJ
Conference Information: **23rd IEEE International Parallel and Distributed Processing Symposium**, Date: **MAY 23-29, 2009 Rome ITALY**
Source: **2009 IEEE INTERNATIONAL SYMPOSIUM ON PARALLEL & DISTRIBUTED PROCESSING, VOLS 1-5** Pages: **840-849** Published: **2009**
Times Cited: **0**

2. Title: Singular value decomposition on GPU using CUDA
Author(s): Lahabar, S.; Narayanan, P.J.
Conference Information: **2009 IEEE International Symposium on Parallel & Distributed Processing (IPDPS)**, Date: **Rome Italy**
Source: **2009 IEEE International Symposium on Parallel & Distributed Processing (IPDPS)** Pages: **10 pp.** Published: **2009**

相关的研究

- 2007年Nvidia的发布了特征值分解范例。
- 2008年 张舒实现基于CUDA架构512×512规模的SVD算法。
- 2009年Sheetal Lahabar 基于CUBLAS实现了QR－SVD算法。
- 2009年EM Photonics, Inc发布CULA1.0，比MKL加速3.2倍。
- 2009年赵佳百项工程实现了CUDA架构下SVD，比MKL加速3.5倍。

创新点

1. 思想层面 用新观点解决老问题
2. 学术层面 科学的最前沿
3. 技术层面 用最先进的技术

目录

单边Jacobi-SVD

单边雅可比算法采用逐次平面旋转的方法求 W 与 V。设 $A = A_0$ 逐次旋转方阵 R_k 使 A_k 的某两列正交化，反复进行旋转变换：

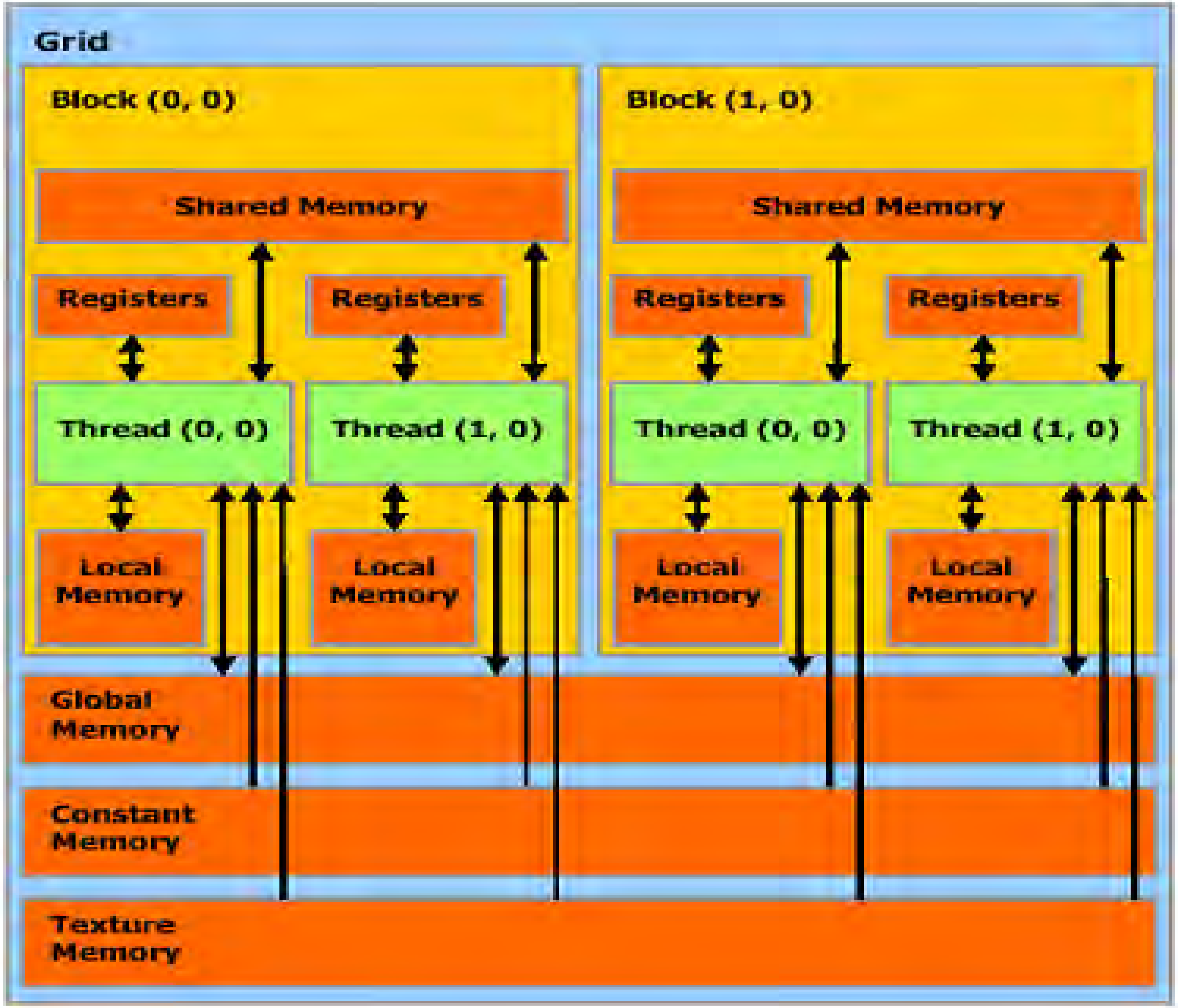
$$A_{k+1} = A_k R_k \quad (k = 0,1,\cdots)$$

可得到矩阵序列 $\{A_k\}$。适当选择列的正交化顺序，则 A_k 的各列趋于两两正交，即 A_k 趋向于 W，而 $V = R_0 R_1 \ldots R_s$，这里 s 是旋转的总次数。

设 R_k 使 p、q 两列正交，则 A_k 与 A_{k+1} 仅在 p、q 两列不同。有：

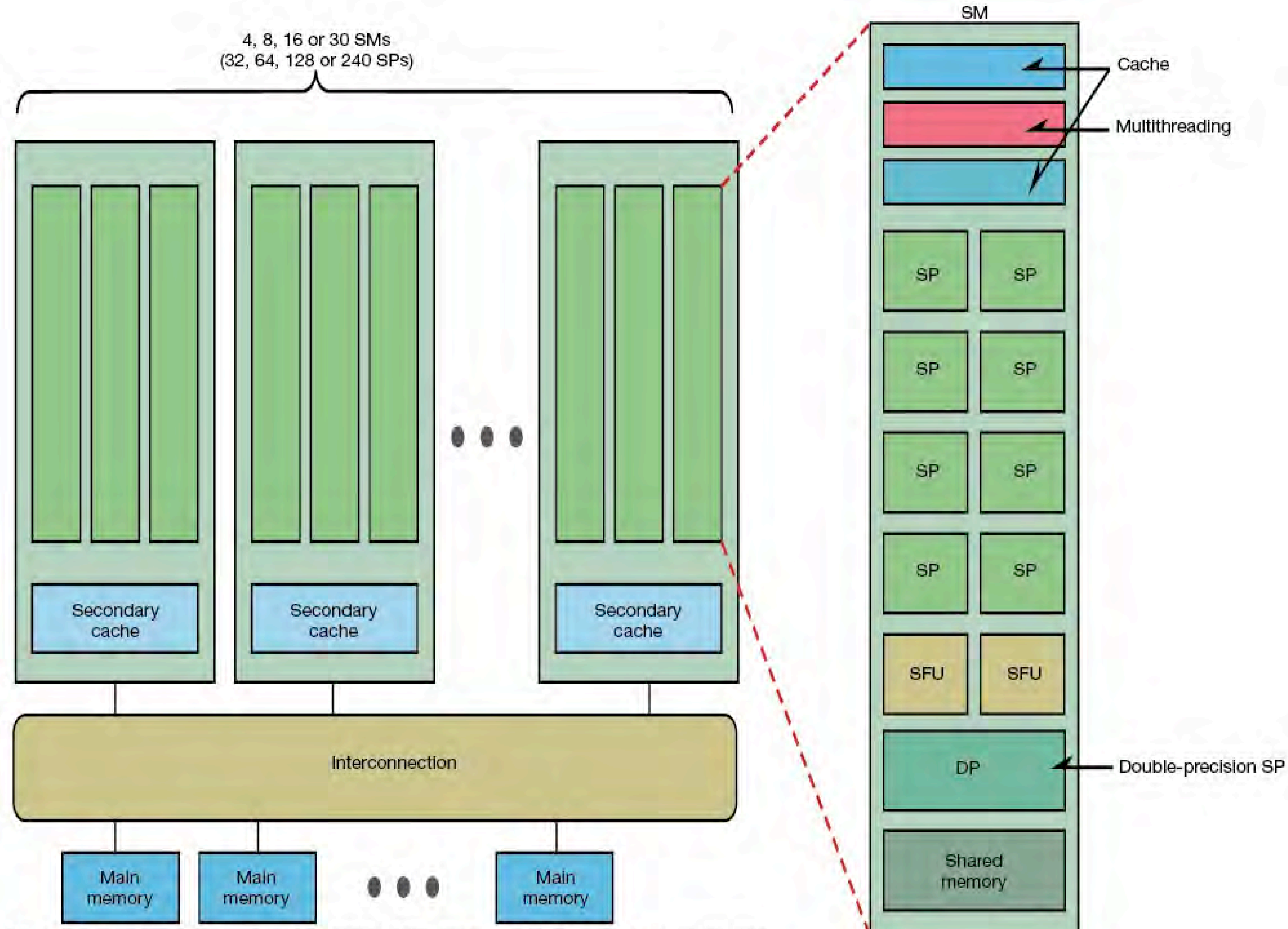
$$\begin{cases} a_p^{(k+1)} = a_p^{(k)} \cos\theta - a_q^{(k)} \sin\theta \\ a_q^{(k+1)} = a_p^{(k)} \sin\theta - a_q^{(k)} \cos\theta \end{cases}$$

难点：合理利用多级存储



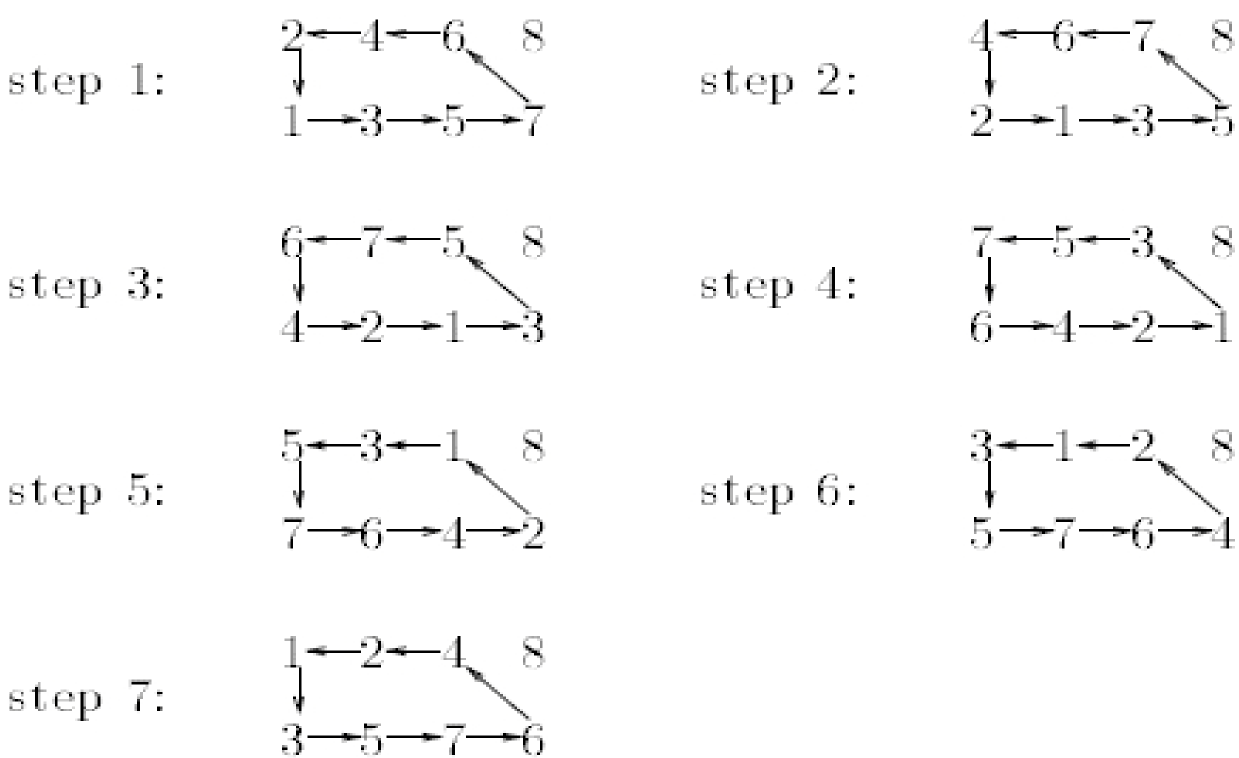
Grid
Block (0, 0)
Shared Memory
Registers
Registers
Thread (0, 0)
Thread (1, 0)
Local Memory
Local Memory
Block (1, 0)
Shared Memory
Registers
Registers
Thread (0, 0)
Thread (1, 0)
Local Memory
Local Memory
Global Memory
Constant Memory
Texture Memory

难点：合理分配Block和Threads



Jacobi-SVD列调度方案:round-robin

以m=8为例说明列对构造过程：



Jacobi-SVD实现细节

合理利用零号线程

在实现过程中，主要通过参数传递方式确定旋转的次数，然后每个 Block 中第一个线程计算数对：

```
if( blockIdx.x == 0 && threadIdx.x == 0)
{
    calculate the row pairs;
}
if( blockIdx.x != 0 && threadIdx.x == 0)
{
    calculate the row pairs;
}
```

Jacobi-SVD实现细节

合理利用零号线程

If（ threadIdx.x == 0）

{

$$c = \frac{1}{\sqrt{1+t^2}}, \qquad s = tc.$$

$$t = \frac{sign(\tau)}{|\tau| + \sqrt{\tau^2+1}}$$

$$\tau = \frac{a_j^T a_j - a_i^T a_i}{2a_i^T a_j}$$

}

Jacobi-SVD实现细节

Blocks And Threads

- 虽然处理器规模小于n/2，Block被分配到SM内执行对用户透明，只需分配n/2个Block。

```
dim3 block_num (Row>>1, 1, 1); //Row stands for the row of the matrix
```

- 将矩阵转置，将列队转换成行对。
- 权衡每个Block共享16K的share memory等诸多限制，选取Block size为1×256。

```
dim3 thread_num (256, 1, 1);
```

Jacobi-SVD实现细节

归并算法

reduce bank-conflict and waiting time

```
if(THREADS_NUM>=512) {  if (tid <  256) { value1[tid] += value1[tid +  256];} __syncthreads();} \
          if(THREADS_NUM>=256) {  if (tid <  128) { value1[tid] += value1[tid +  128];} __syncthreads();} \
          if(THREADS_NUM>=128) {  if (tid <  64) { value1[tid]  += value1[tid +  64];}   __syncthreads();} \
          if(THREADS_NUM>=64)  {  if (tid <  32) { value1[tid]  += value1[tid +  32]; \
                                    value1[tid]  +=value1[tid  +  16]; \
                                    value1[tid]  += value1[tid +   8]; \
                                    value1[tid]  += value1[tid +   4]; \
                                    value1[tid]  += value1[tid +   2]; \
                                    value1[tid]  += value1[tid +   1]; } }
```

目录

测试平台简介

显卡处理单元（Graphic Processing Unit, GPU）

- Tesla T10，显存4G
- 单卡浮点处理峰值为1 Tflops
- 双精度浮点处理能力为87 Gflops

MKL环境

- Intel Core 2 Duo CPU E6750@2.66GHz
- 浮点运算能力22.4Gflops

Matlab环境

- Matlab版本为：Version 7.8.0.347 (R2009a) 64-bit
- 8核 Intel(R) Xeon(R) CPU E5405@2.00GHz处理器

实验方法

- 本文先通过随机生成0-100内随机矩阵，反复测试12次取平均值，去掉最大值、最小值，取剩余10次的平均值即为所得结果。避免由于某次时间过好或者过坏影响测试结果。
- 所有实验数据均采用单精度浮点运算数（single-precision），IEEE32位浮点数值的形式。
- 由于GPU的内存带宽可高达141GB/s，与单边Jacobi迭代消耗的时间相比可忽略不计，因此，下文中所测得T1070处理SVD的时间不包括对待处理矩阵拷贝至显存以及所得结果拷贝至内存时间。对矩阵的按列范数预处理在GPU中进行，时间计算在内。

实验方法

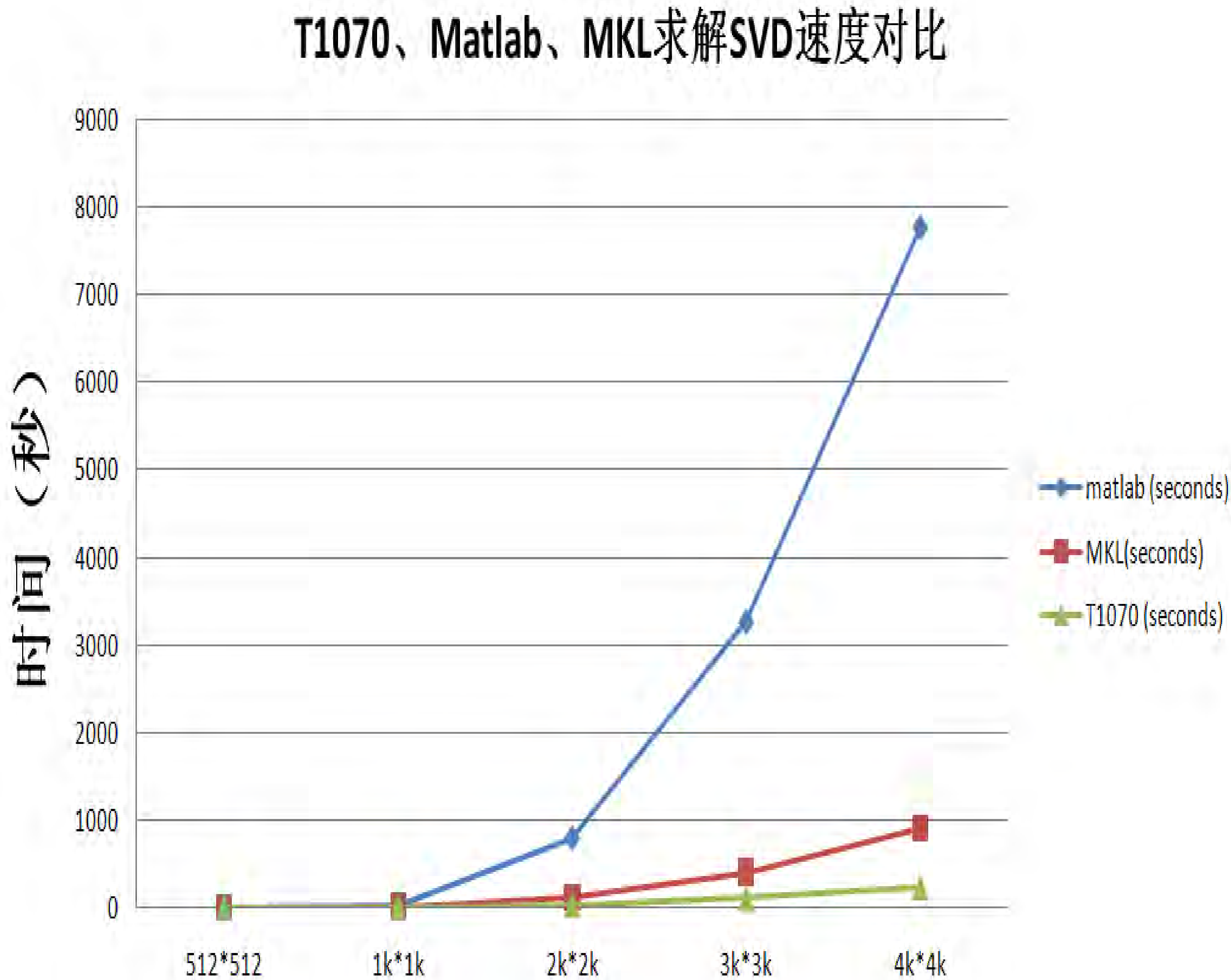
$$相对于Matlab加速比=\frac{Matlab\ 执行时间}{GPU执行时间}$$

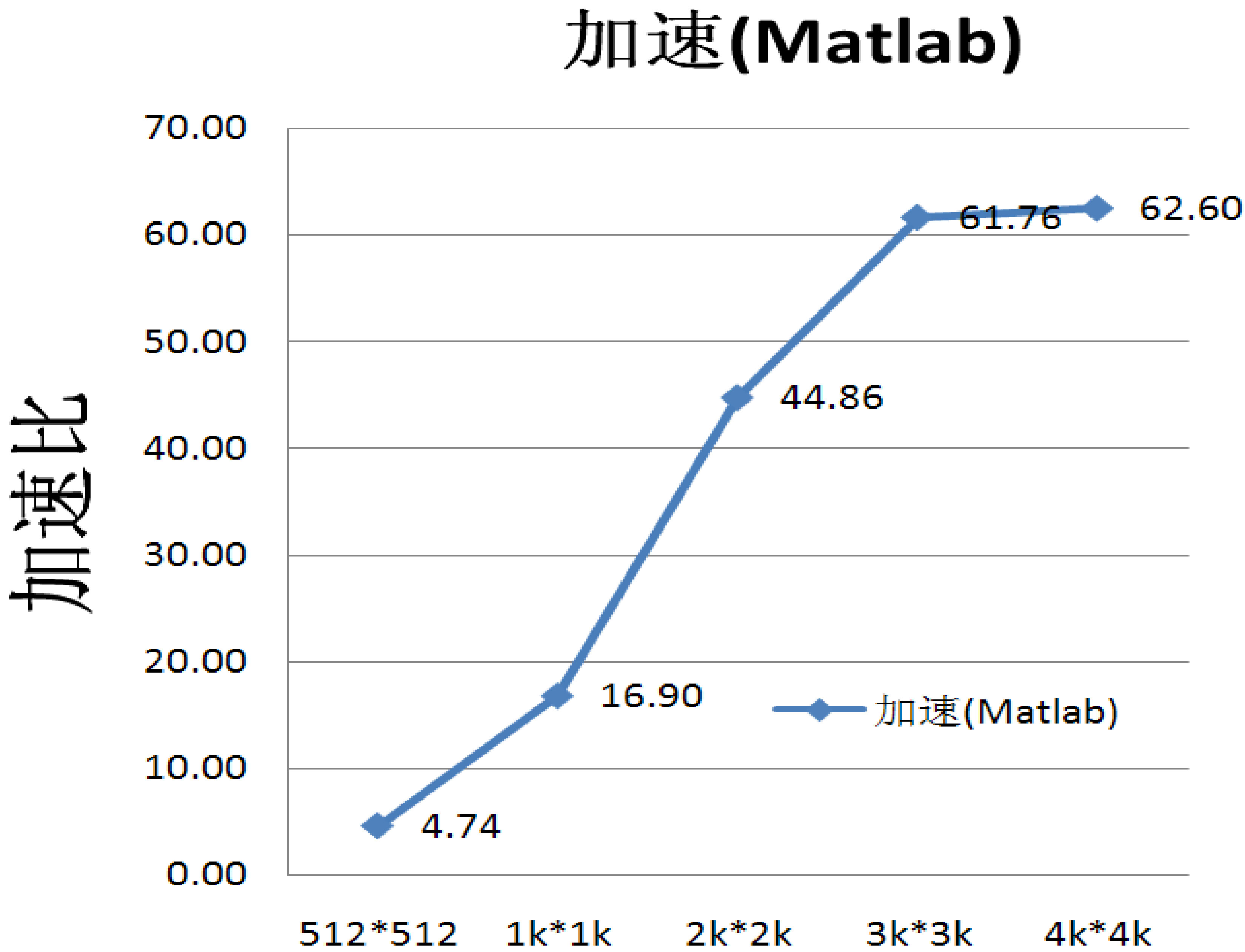
$$相对于MKL加速比=\frac{MKL执行时间}{GPU执行时间}$$

$$误差=\frac{|计算值-精确值|}{精确值}*100$$

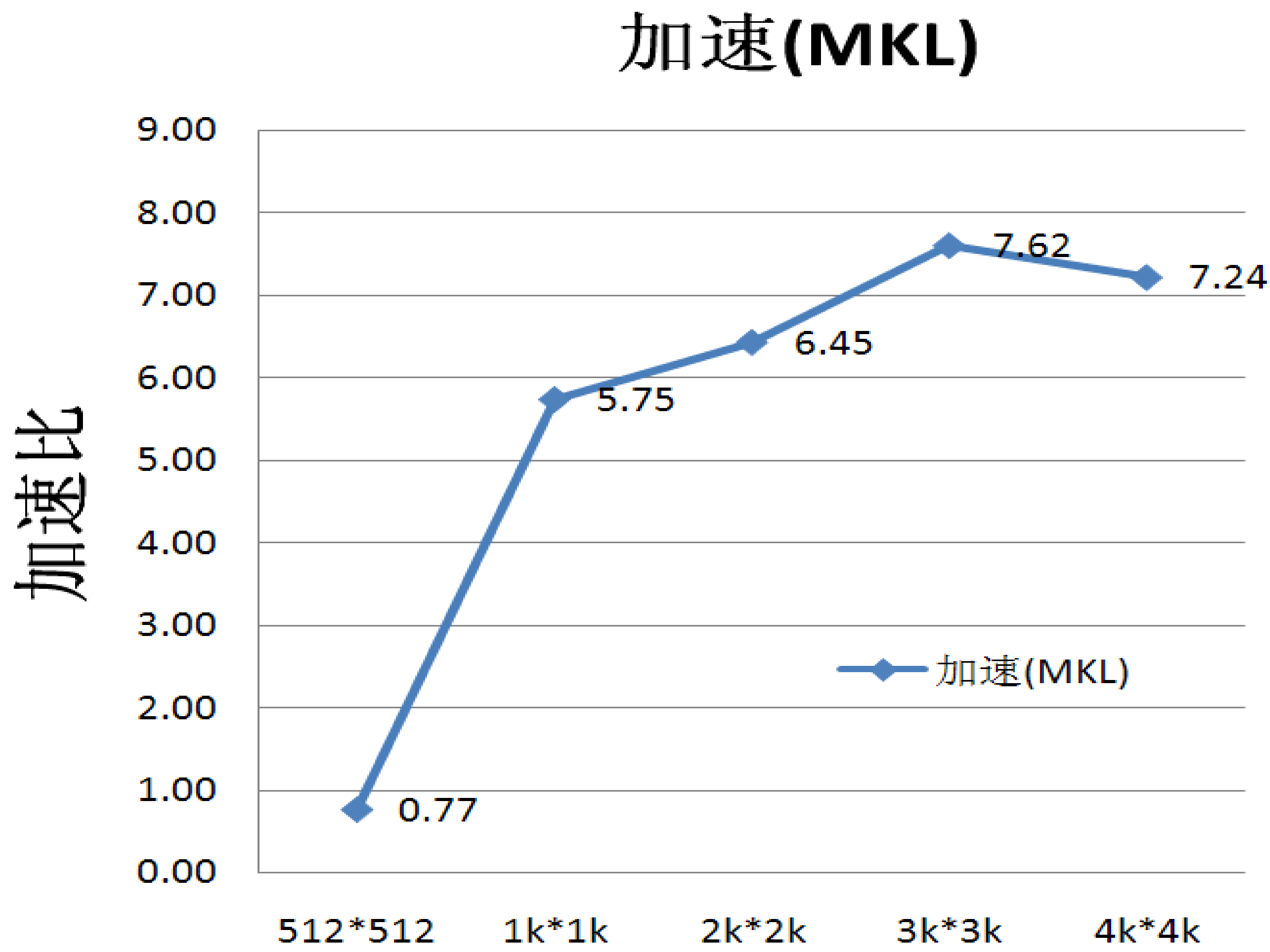
优化算法与Matlab以及MKL比较

length	matlab (s)	MKL(s)	T1070 (s)	加速(Matlab)	加速(MKL)
512*512	3.60	0.58	0.76	4.74	0.77
1k*1k	33.06	11.26	1.96	16.90	5.75
2k*2k	797.51	114.63	17.78	44.86	6.45
3k*3k	3262.65	402.70	52.83	61.76	7.62
4k*4k	7770.88	898.23	124.13	62.60	7.24


T1070、Matlab、MKL求解SVD速度对比
时间（秒）
9000
8000
7000
6000
5000
4000
3000
2000
1000
0
512*512
1k*1k
2k*2k
3k*3k
4k*4k
matlab (seconds)
MKL(seconds)
T1070 (seconds)



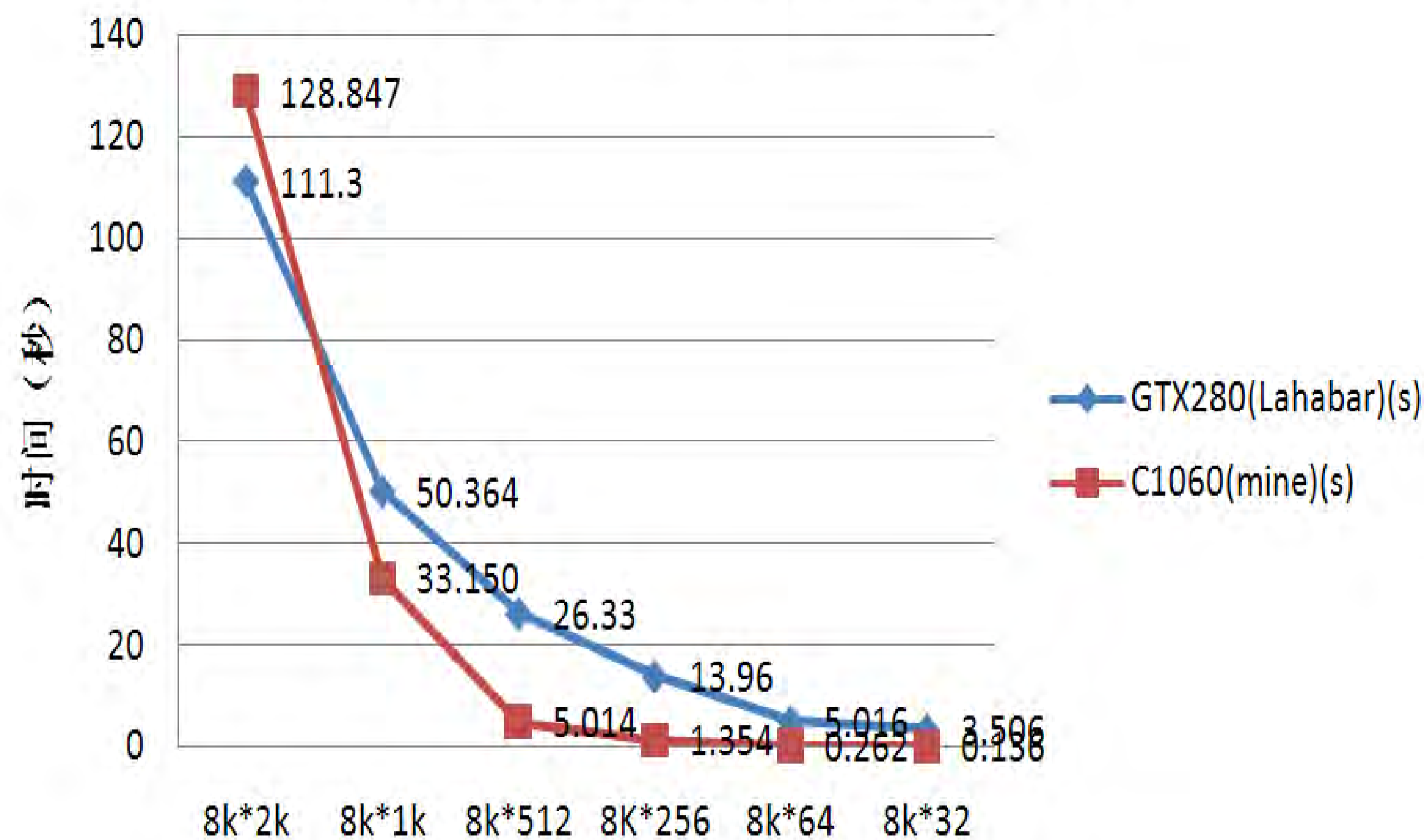
加速(Matlab)
加速比
70.00
60.00
50.00
40.00
30.00
20.00
10.00
0.00
4.74
16.90
44.86
61.76
62.60
加速(Matlab)
512*512
1k*1k
2k*2k
3k*3k
4k*4k


加速(MKL)
加速比
9.00
8.00
7.00
6.00
5.00
4.00
3.00
2.00
1.00
0.00
0.77
5.75
6.45
7.62
7.24
加速(MKL)
512*512
1k*1k
2k*2k
3k*3k
4k*4k

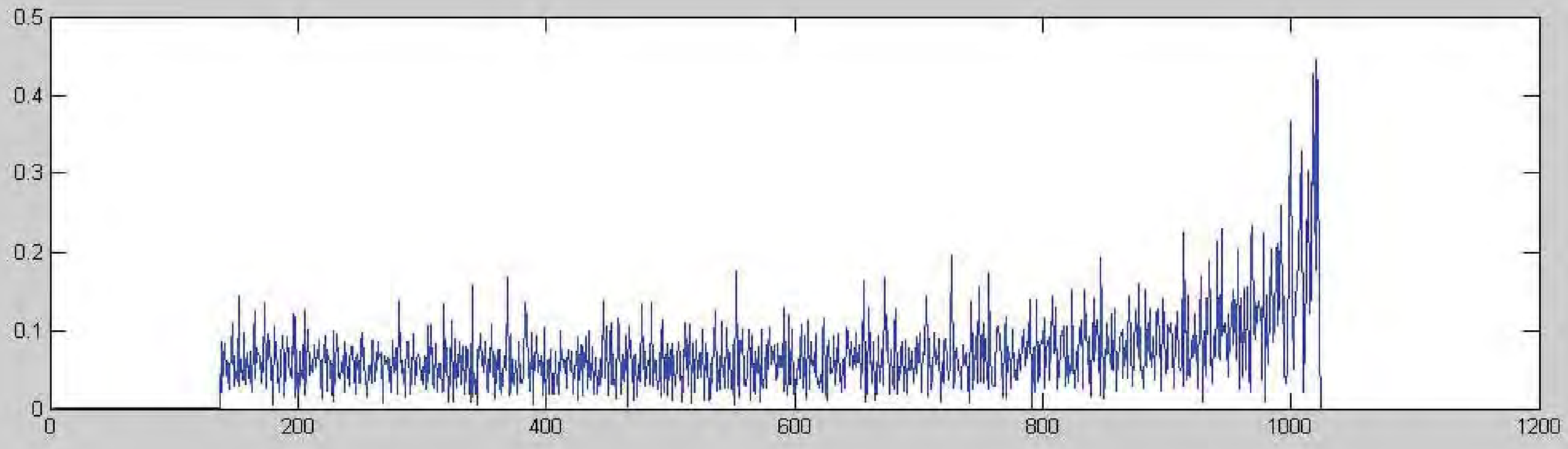
优化算法与Lahabar 文献比较

length	GTX280(Lahabar)(s)	C1060(mine)(s)	length	加速比
8k*2k	111.3	128.847	8k*2048	0.86
8k*1k	50.364	33.150	8k*1024	1.52
8k*512	26.33	5.014	8k*512	5.25
8K*256	13.96	1.354	8K*256	10.31
8k*64	5.016	0.262	8k*64	19.12
8k*32	3.506	0.136	8k*32	25.71

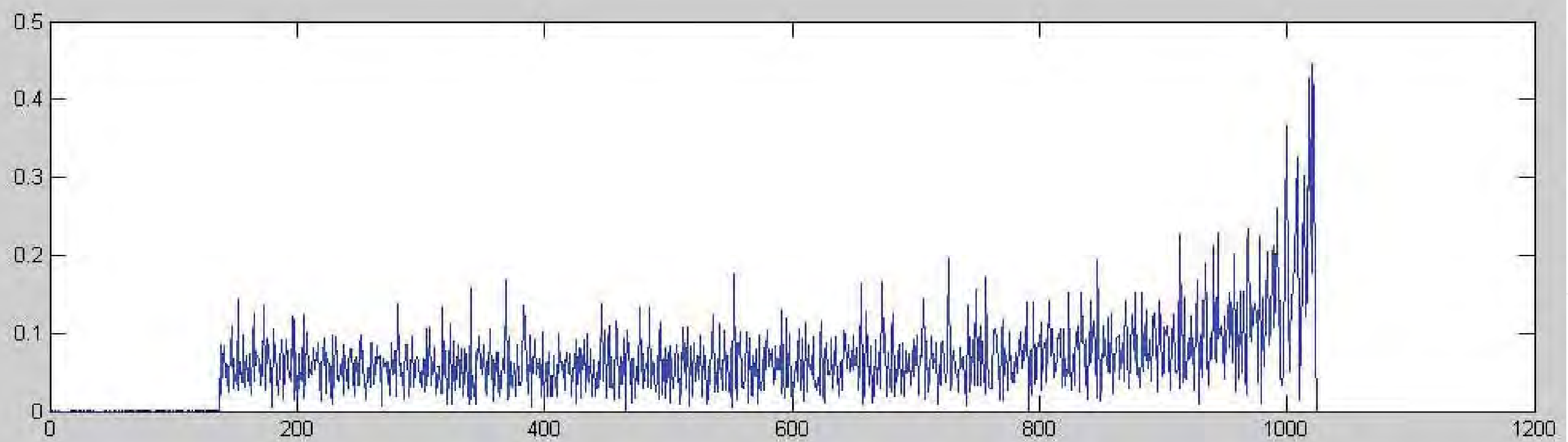
优化算法与Lahabar 文献比较



优化算法误差分析

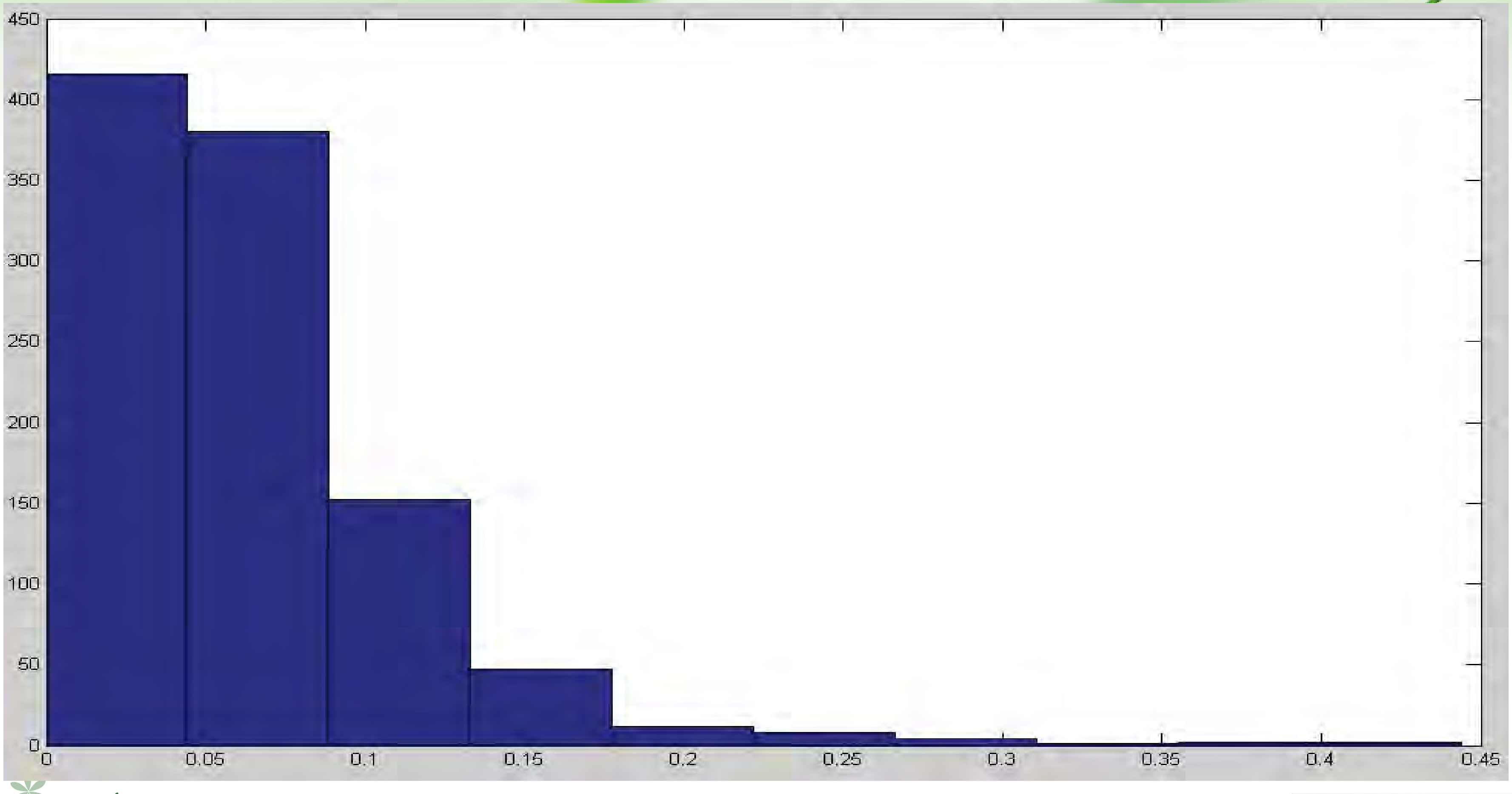


迭代7轮后奇异值误差分布



迭代10轮后奇异值误差分布

优化算法误差分析



奇异值误差分布条形图

目录

结　论

- 在CUDA环境中实现了Jacobi-SVD算法。
- 当处理的矩阵规模达到4096*4096时，得到相对于Matlab了60倍以上的提速,相对于MKL7倍的加速。
- 通过对实验数据分析，当处理长宽比例大于16时，算法加速明显。

http://www.nku-isc.org/~zhaojia/

欢迎光临 我爱南开 BBS | 学习 - 南开BT站 | ZhaoJia's - Home

http://www.nku-isc.org/~zhaojia/svd/

Customize Links | Free Hotmail | Windows Marketplace | Windows Media | Windows | 从 IE 导入

ZhaoJia's Homepage

School of Mathematical Sciences, Nankai University

- **Home**
- Personal Information
- Research Topics
- Courses
- Photo album
- Download page
- Usefull Links

Welcome

Welcome to ZhaoJia's homepage.Leaving me message

Some Information

College of Math., Nankai U, Tianjin 300071,P.R. China

Email : benbenwo1091@163.com

Blog : http://benbenwo1091.blog.163.com

Singular Value Decomposition (SVD) is wildly used and an important tool for reducing rank. The fastest method dealing with SVD of matrix has the complexity of , which affects the speed of SVD and the scale dealt with. In this paper, by using the powerful capacity of floating point operation in GPU, we have implemented one-sided Jacobi based SVD in CUDA, which could deal with Large scale dense matrix. When dealing with the matrix larger than , the program in this paper is 60 times faster than the program in Matlab, and 7 times faster than MKL, which is distributed by Inter Company. The program in CUDA has improve the speed and scale of the problems dealing with SVD.

Keyword : Singular Value Decomposition (SVD); one-sided Jacobi Rotation; CUDA; GPU

My Pictures

Download

感　　想

- 英语很重要
- 可做的东西还是很多

  基于Multi-GPU的SVD算法。

  基于GPU的Tensor-SVD算法。

参考文献

[1] Shu,Z. One Sided Jacobi Method on CUDA for SVD.Journal of Application Research of computers. 2007,Vol24 No.6.
[2] S. Lahabar, P.J. Narayanan, Singular value decomposition on GPU using CUDA, In Proceedings of 23rd IEEE International Parallel and Distributed Processing Sympo-sium(IPDPS), May 2009.
[3] G.H. Golub and C. Van Loan. Matrix Computations. Johns Hopkins, second edition edition,1989.
[4] Nvidia CUDA oficial website
[5] G.H. Golub and W. Kahan. Calculating the singular values and pseudo-inverse of a matrix.SIAM J.Numer. Anal.,, (3):205-224, 1965.
[6] Golub G H,Luk F T. Singular value decomposition: application and computations. Trans22nd Conf Army Mathematians, ARO Report.1977. 77（1）（577-605）
[7] Golub G H,Reinsch C. Singular Value Decomposition and Least Squares Solutions. NumerMath.1970,14:403 420
[8] Jordan C. Memoire sur les formes bilineaires. J Math Pures Appl, Deuxieme Serie,1874,19:35 54
[9] Demmel J,Veselic K, Jacobi's methrod is more accurate than QR. SIAM J Matrix AnalAppl,1992,13(4):1204 1245
[10] Erik Lindholm, John Nickolls, Stuart Oberman. NVIDIA Tesla: a uni¯ed graphics andcomputing architecture. Hot chips 2008 March-April, Page（s）39-55
[11] John D. Owens. A Surey of General-Purpose Computation on Graphics Hardware, EURO-GRAPHICS 2005
[12] Ian Buck, Tim Foley, Daniel Horn, Jeremy Sugerman etc. Brook for GPUs: Stream Com-puting on Graphics Hardware.SIGGRAPH 2004
[13] LESSIG, C. Eigenvalue Computation with CUDA. Tech. rep., NVIDIA Corporation, August2007.
[14] Brent R P, Luk F T.The solution of singular value and symmetric eigenproblems on multi-processor arrays. Sct Stat Comput. 1985,6:69-84
[15] J.C. Nash. A one-sided transformation method for the singular value decomposition and algebraiceigenproblems. Comput. J., 18(1):74 76, 1975.
[16] Zhou, B., Brent, R.: A Parallel Ring Ordering Algorithm for E±cient One-sided SVD Computations. Journal of Parallel and Distributed Computing 42, 1–10 (1997)
[17] Rajasekaran, S., Song, M.: A Novel Scheme for the Parallel Computation of SVDs. In: Proc. High Performance Computing and Communications, pp. 129–137 (2006)

Thank You!