Randomization Algorithm to Compute Low-Rank Approximation

HAN Ru

The Chinese University of Hong Kong

Outline

- Background
- Algorithm and Math Model
- Project Scheme
    - Done
    - To do

Background

- General SVD

$A = U\Sigma V^t$

matrices $U = [u1u2...um] \in Rm \times m$; $V = [v1v2...vn] \in Rn \times n$
$\Sigma = diag(\sigma1,...,\sigma v)$, where $\Sigma \in Rm \times n$, $v = min\{m,n\}$ and $\sigma1 \geq \sigma2 \geq ... \geq \sigma v \geq 0$.

- Low-Rank SVD Approximation

$A = U_k \Sigma_k V_k^t$

- LAPACK/MAGMA software framework

Algorithm--Power iteration:

Matlab Code “svd_rand./” SVD approximation

```
function [u,s,v] = svd_rand(A, k, l, max_iters)
q = randn(n,k+l);
[q,r] = qr(q,0);
    for iter=1:(max_iters-1)
        p = A*q;
        q = A'*p;
        [q,r] = qr(q,0);
    end
 p = A*q;
    [p,b] = qr(p,0);
end
[x,s,y] = svd(b);
```

u_k = p*x(:,1:k);

s = s(1:k,1:k);

v_k= q*y(:,1:k);

Algorithm and Math Model

Matrix	Size
A	M-by-N
Q	N-by-(K+L)
P	M-by-(K+L)
B	(K+L)-by-(K+L)
X	(K+L)-by-(K+L)
Y^T	(K+L)-by-(K+L)
SI	(K+L)-by-1
S	K-by-1
u_k	M-by-K
v_k	N-by-K

Error= $||A-U_K S_K V_K^T||_2$

= $(k+1)_{th}$ largest singular value of A

Project Scheme

1. Implementing the randomized algorithm using LAPACK on CPU

2. Implementing the randomized algorithm using MAGMA on GPU

3. Implementing the out-of-memory randomized algorithm on GPU

   -single queue

   -UMA

   -manual pipelining.

   -Multiple GPUs using CUBLAS-XT

4. set up tester to compare performance

5.Application

Done

1. Implementing the randomized algorithm using LAPACK on CPU

2. Implementing the randomized algorithm using MAGMA on GPU

3. Implementing the out-of-memory randomized algorithm on GPU with single queue

4. set up tester to compare the performance

   LAPACK &CPU&GPU&OUT-OF-MEMORY

```
 Error is ||A - Uk*Sk*Vk^T||_2, L=K, performs 10 iterations

%   M     N     K      LAPACK time (s)    Randomized time (s)   LAPACK error   R
andomized error
%==============================================================================
==============
     0.02,   0.00,   0.00,   0.00,   0.00,   0.00 ,   0.00
 NB=100, m=1022, n=1022
 1022  1022     2      0.91                 0.01,   0.03,   0.09          1.83e+01
       1.84e+01,1.84e+01,1.84e+01      (S[2]=1.83e+01)
```

Out-of-Memory GPU Implementation

Device: Tesla K80, 823.5 MHz clock, 11439.9 MiB memory, capability 3.7

1 MiB = 2^{20} bytes = 1024 kibibytes = 1048576bytes

11439.9Mib$*$1048576=1.1996e+10 bytes

Sqrt(12e9/8)=3.8730e+04

Out-of-Memory GPU Implementation

P=A*Q

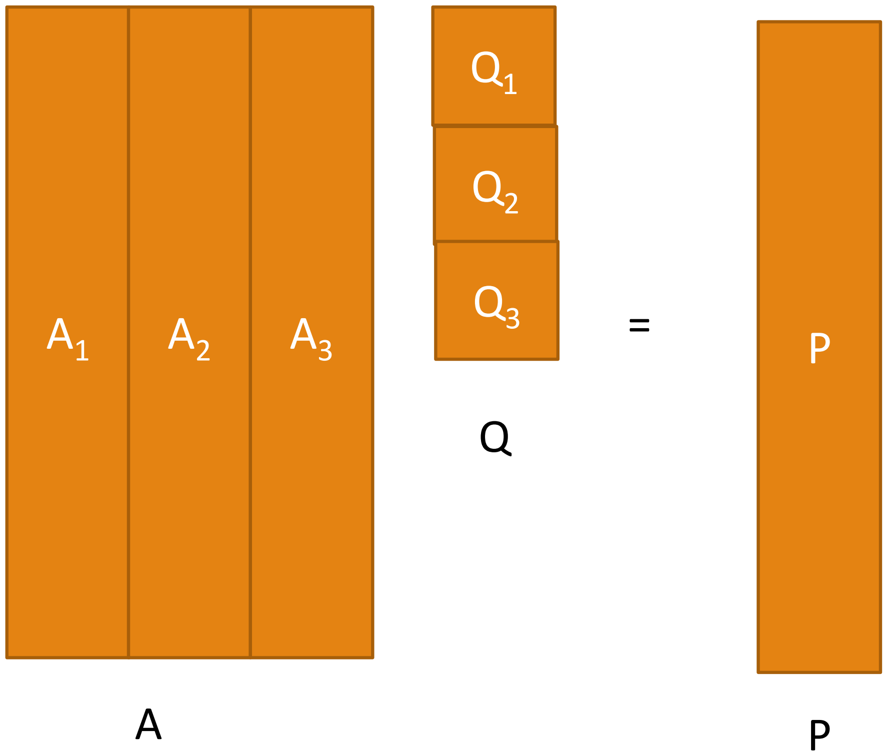
P=0;

For k=1,2,3…..

  set (A_k to dA);

  $P=P+A_kQ_k$;

end



Out-of-Memory GPU Implenmentation

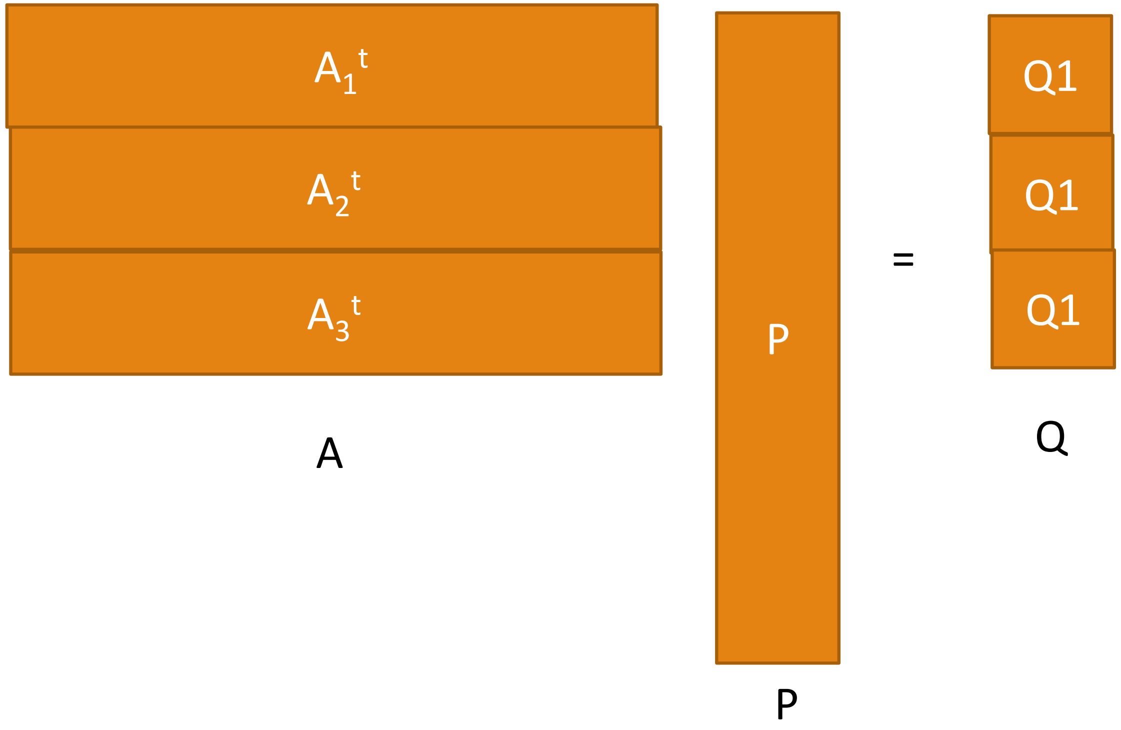
$Q=A^t*P$

For k=1,2,3…..

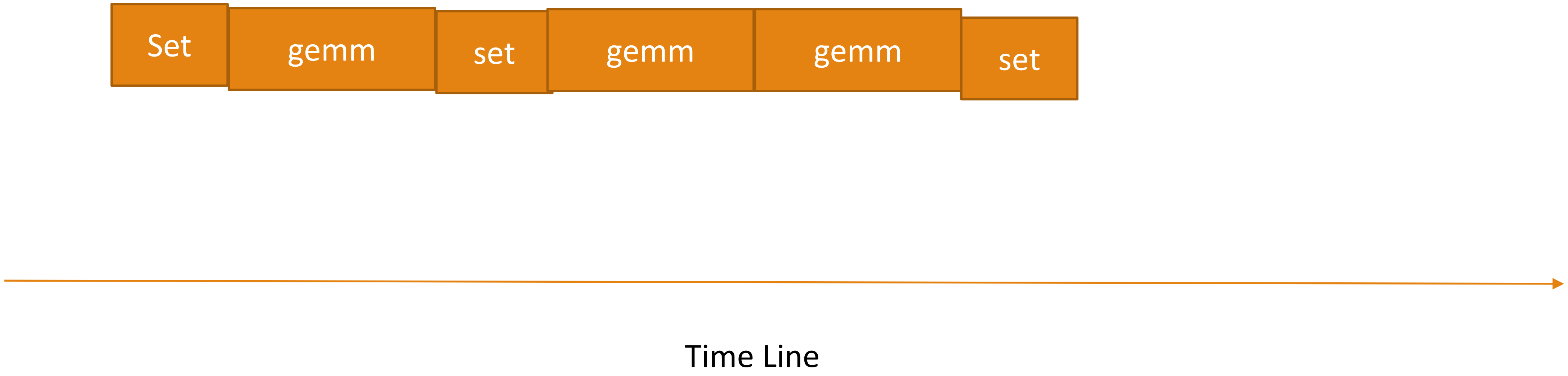
set (A_k to dA);

$Q_k=A_k^tP$;

end



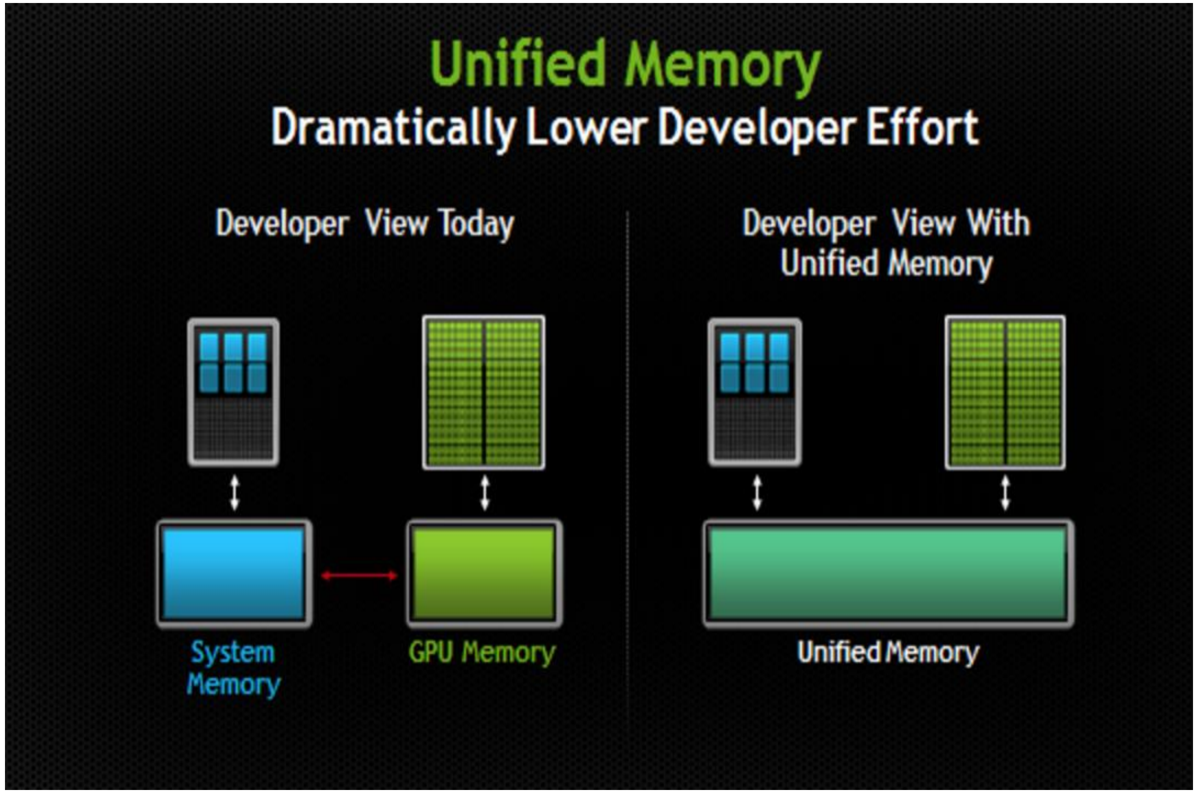
Out-of-Memory GPU Implementation using single queue



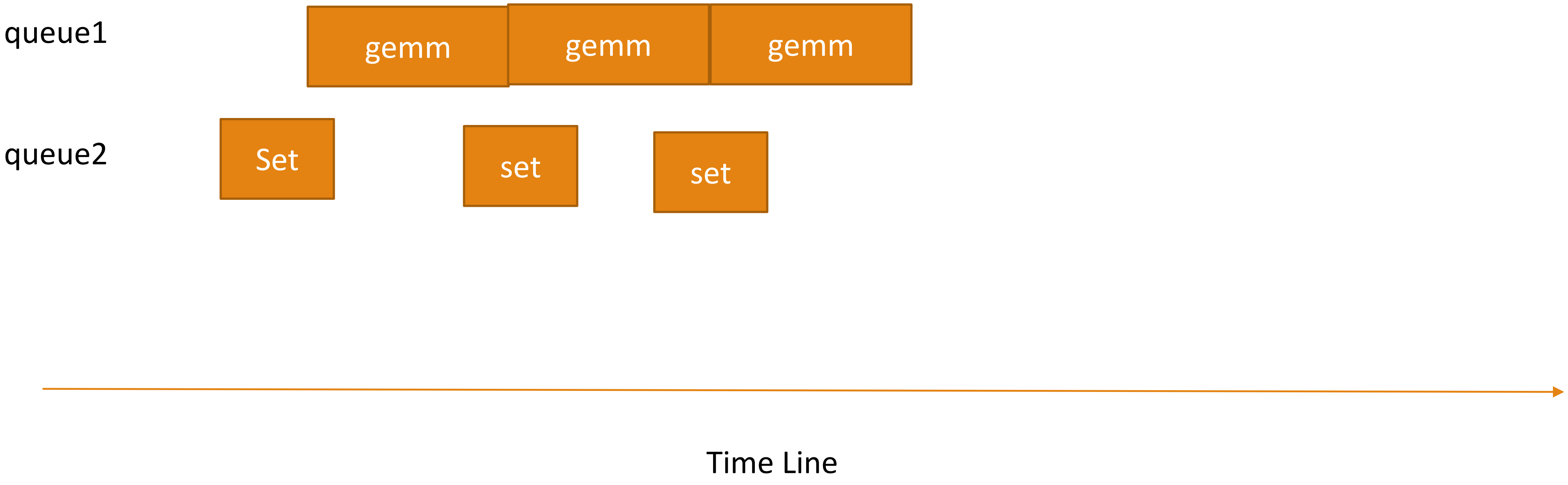
To do

- Implementation on single GPU using UMA(Unified Memory Access)
- Implementation on single GPU using manually pipelining
- Implementation on multiple GPUs using CUBLAS-XT
- Application—image processing

To do-Implementation on GPU using UMA

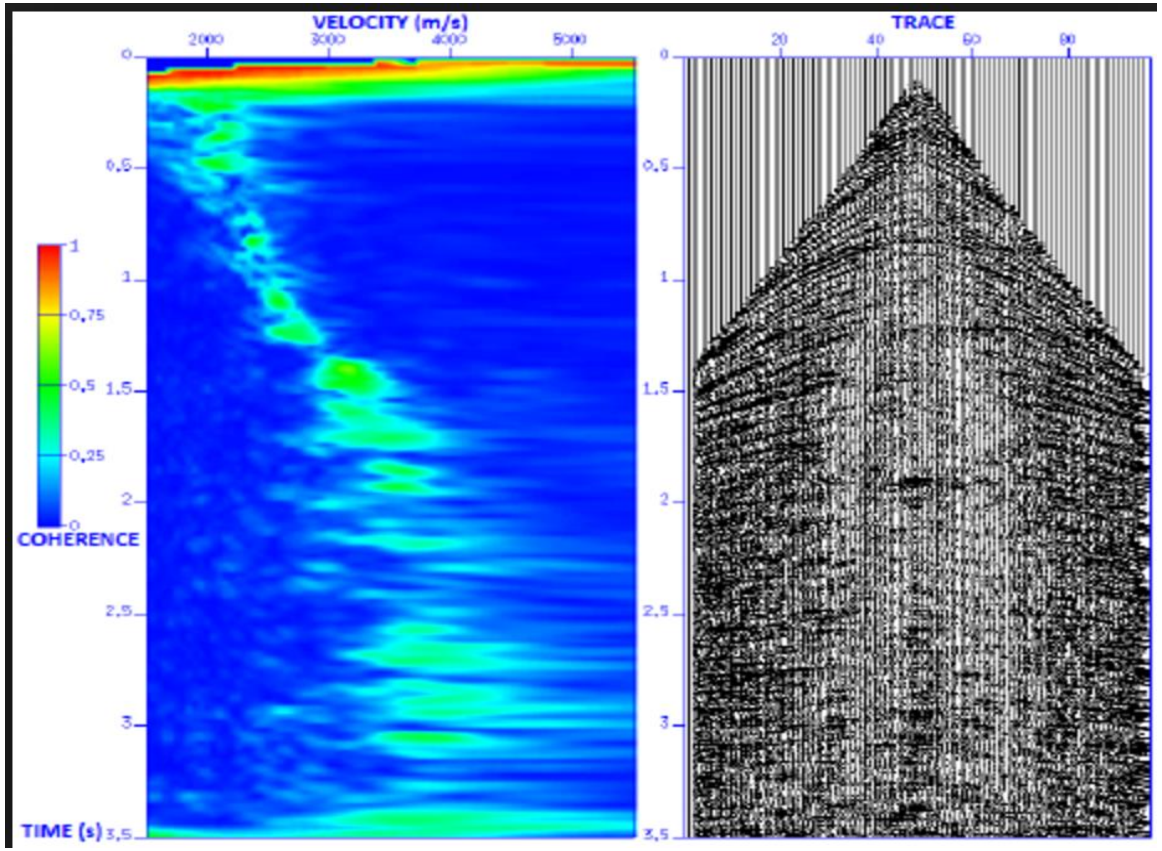


To do-Implementation on GPU using manually pipelining



To Do-Application

- Latent Semantic Indexing (LSI)
- Genetic clustering
- subspace tracking
- image processing



Reference

[1]Harris, M. and →, V. (2017). *Unified Memory in CUDA 6*. [online] Parallel Forall. Available at: https://devblogs.nvidia.com/parallelforall/unified-memory-in-cuda-6/ [Accessed 21 Jun. 2017].

[2]Mahoney, M. (2011). *Randomized algorithms for matrices and data*. Hanover, Mass.: Now Publishers.

[3]Drinea, E., Drineas, P. and Huggins2, P. (2017). *A Randomized Singular Value Decomposition Algorithm for Image Processing Applications*. [ebook] 1 Computer Science Department, Harvard University Cambridge, MA 02138, USA 2 Computer Science Department, Yale University New Haven, CT 06520, USA. Available at: http://ai2-s2-pdfs.s3.amazonaws.com/e881/439705f383468b276415b9d01d0059c1d3e5.pdf [Accessed 26 Jun. 2017].

[4] En.wikipedia.org. (2017). *Latent semantic analysis*. [online] Available at: https://en.wikipedia.org/wiki/Latent_semantic_analysis [Accessed 29 Jun. 2017].