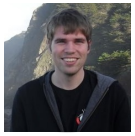
Communication-Avoiding QR Decomposition for GPUs

Michael Anderson, Grey Ballard, James Demmel, Kurt Keutzer
University of California, Berkeley

Parallel Computing Lab



Main Idea

• Reducing communication between GPU and DRAM can give us an order of magnitude speedup
  • Turn a bandwidth-bound problem into a compute-bound problem

• Communication-Avoiding QR[1] is a recent algorithm for solving a QR decomposition which is optimal with regard to the amount of communication performed

• This allows us to achieve higher computational intensity, requiring less memory traffic.

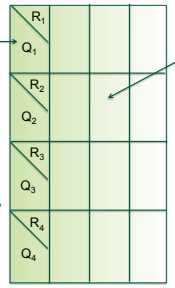
• CAQR performs exceptionally well on the GPU, especially for the challenging case of tall-skinny matrices.

1: J. Demmel, L. Grigori, M. Hoemmen, and J. Langou. Implementing Communication-Optimal Parallel and Sequential QR Factorizations. Arxiv preprint arXiv:0809.2407, 2008.

Communication-Avoiding QR

• Small QR decompositions fit in cache

• Eliminate triangles using a QR reduction tree



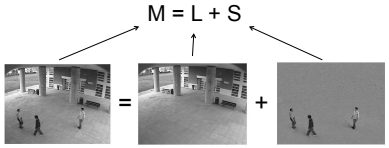
• Blocked trailing matrix updates also fit in cache

• Computational intensity
> 16 FLOPS/Byte

• Main GPU optimizations:
  • Avoiding shared memory and using the register file to store the matrix whenever possible
  • Tuning the block width to trade some extra work for a reduction in bandwidth
• Note: Q is stored differently than the standard approach

Example Application: Robust PCA

• Decompose a surveillance video into a **low rank** component and a **sparse** component:



• Video = tall-skinny matrix:

• Main computation is an SVD of the video matrix

• Use QR as a first step for SVD of a tall-skinny matrix

• *Quality of output is dependent on the number of QRs performed*

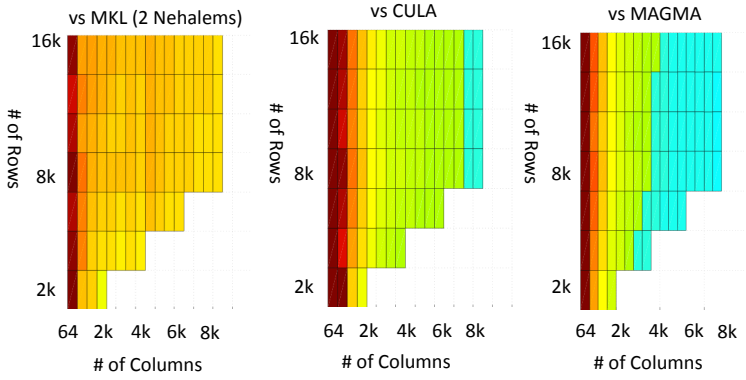
Traditional Householder QR

• From left to right, tall panel factorizations generate Householder vectors
• Matrix-multiply can be used to apply the Householder vectors to the rest of the matrix

• For wide matrices most of the time is spent in matrix-multiply
  • *FAST!*

• For skinny matrices, most time is spent in the BLAS2 panel factorization
  • *SLOW!*

Performance



• CAQR performs best for skinny matrices. For the square case, we are not able to use SGEMM so traditional approaches perform better.

• For very tall-skinny matrices, such as our video matrix, CAQR achieves an order of magnitude speedup.

"Communication-Avoiding QR for GPUs" Anderson, Ballard, Demmel, Keutzer IEEE International Parallel & Distributed Processing Symposium, 2011