LU Decomposition on a GPU

Presented by Scott Shellenhammer
AGENDA

Project Overview & Goal

Results

Challenges & Learnings
The Project Goal is to factor a matrix into a LU Decomposition using parallel techniques on a graphics-processing unit (GPU) at an optimal speed.

Two techniques are used for the factorization and their performance results are compared with the corresponding sequential versions.

- **LU decomposition** without pivoting using a loop unrolling technique.
- **LU decomposition** with partial pivoting using a block algorithm.
What are the Benefits of LU Decomposition on a GPU?

It’s useful for many applications, such as solving Linear Equations.....and most importantly it needs to be fast!
**Project Approach**

- **Algorithms without Pivoting**
  - Sequential
  - Parallel

- **Algorithms with Partial Pivoting**
  - Sequential
  - Unblocked vs. Blocked
  - Parallel

- **Optimize Parallel Approaches**
  - Hybrid CPU/GPU
AGENDA

Project Overview & Goal

Results

Challenges & Learnings
Results without Pivoting

LU Algorithms w/o pivoting (gflops/s)

Matrix Size

Gflops/s

- Red: Doolittle
- Orange: Crout
- Blue: Right-Looking
- Green: GPU
Results with Partial Pivoting

LU Algorithms partial pivoting (gflops/s)

Matrix Size

Gflops/s

Scipy
C Unblocked
GPU blocked
AGENDA

Project Overview & Goal

Results

Challenges & Learnings
Summary of Challenges & Learnings

1. Review performance of individual components

2. Memory transfer from host to device

3. Working with external libraries

4. When in doubt...write/run the code
Review Individual Component Performance

GPU Algo Time Spent

- GPU Transfer
- Numpy Permutation
- Scipy LU
- GPU Calc

Matrix size

32  64  128  256  512  1024  2048  3072  4096  5120  10240
Memory Transfer from host to device

- Try to minimize transfer
- Sometimes better to do trivial work on GPU than to transfer back and forth
- CPU/GPU hybrid algorithm – difficult to reduce transfers
Working with External Libraries

• Can be excellent tools for quick development

• But...don’t take performance for granted!

• Analyze performance of external libraries as well as your own code
When in Doubt...Write/Run the Code

- GPUs kernels can be difficult to visualize
- Example: Memory access
- Write different access types and measure the performance gains
QUESTIONS?