

## Lecture 3: January 30

*Lecturer: Horst Simon**Scribes: Eric Battenberg*

**Disclaimer:** *These notes have not been subjected to the usual scrutiny reserved for formal publications. They may be distributed outside this class only with the permission of the Instructor.*

### 3.1 Introduction

- Dense Linear Algebra
  - Achieves high machine efficiency
  - Large range of applications

### 3.2 Summary of last lecture

- Using simple memory model for optimizations
  - Intensity is key to algorithm efficiency:  $q = f/m$
  - Machine balance is key to machine efficiency:  $t_m/t_f$
- Matrix-vector multiplication
  - $q = 2$ , constant, see last lecture notes

### 3.3 Matrix-Matrix multiplication

- Naive MM multiply
  - $2n^3 = O(n^3)$  flops
  - $3n^2$  words of memory
  - $q$  as large as  $\frac{2n^3}{3n^2} = O(n)$ 
    - \* Can run well on every machine (for large matrices)
    - \* But actually do order  $n^3$  reads, major limitation of this implementation
    - \* In scientific computing
      - Memory references as or more important than floating point operations
    - \* MM mult: optimized in several ways, more later
  - Can exchange i,j,k loops: 6 different ways of organizing naive method
    - \* Changes number of memory references
  - Extra memory references add up as problem size increases, so performance doesn't increase
- Block (tiled) MM mult

- $N \times n$  elements,  $N \times N$  blocks with  $b \times b$  subblocks
- Read blocks into fast memory
- Blocking reduces number of memory operations
  - \*  $q \approx n/N = b$  for large  $n$
  - \* But little blocks need to fit in cache (next limitation)
- Fast memory has size  $M_{fast}$ ,
  - \*  $3b^2 \leq M_{fast}$
  - \*  $q \approx b \leq \sqrt{M_{fast}/3}$
  - \* Required cache size is reasonable for most machines (to achieve best performance)
    - This doesn't look at hierarchy of multiple caches
- Limits to optimizing MM mult
  - Using associativity, get slightly different answers from naive code due to roundoff
  - Blocked has computational intensity:  $\leq \sqrt{M_{fast}/3}$ 
    - \* Theorem: limit  $q = O(\sqrt{M_{fast}})$  to any reorganization to this algorithm
  - What about more levels of memory hierarchy?
- Recursion: Cache oblivious algorithms
  - Don't need to find a good block size
  - Treat  $n \times n$  mult as set of smaller problems
    - \*  $A(n \times m) \times B(m \times p)$ 
      - Case 1:  $m \geq \max(n, p)$ : split A horiz
      - Case 2:  $n \geq \max(m, p)$ : split A vert, B horiz
      - Case 3:  $p \geq \max(m, n)$ : split B vertically
    - \* ShoaiB said this implementation is relatively successful, but need to end recursion at some point
    - \* Implementing it isn't easy
  - Recursive data layouts
    - \* Blocking requires knowing cache size
    - \* Works well for every single cache size
    - \* But index calculations are expensive
    - \* Recursion covered in CS 170
- Strassen's MM Mult
  - Minimize number of floating point ops
  - Horst Simon is from Germany
    - \* He had a roommate who was a math major who couldn't sleep who stayed up trying to optimize the number of operations in an algorithm. Horst thought he was crazy, but he became a professor of mathematics.
  - $O(n^{2.81})$
  - 2x2 MM mult, takes 8 mults, 4 adds
    - \* Strassen: 7 mults, 18 adds
  - The key is recursion

- \* Then mults become MM mults which are much more complex than MM adds
- Memory got big enough in scientific computing to allow large enough MM mults to make Strassen's algorithm beneficial
- Strassen's algorithm beat vendor benchmark limit
- Asymptotically faster
- Error properties not as clear, purists are concerned
- Other fast MM mult algorithms
  - World record  $O(n^{2.376})$ , Coppersmith and Winograd
  - Hong/Kung theorem
  - Possibility of  $O(n^{2+\epsilon})$ , (Cohn, Umans, Kleinberg, 2003)

### 3.4 Short Break: Questions about the course?

- Group assignments are on website
- Guy from HPC consulting office hours, 777 soda, T 1-2pm.

### 3.5 Automatic Performance Tuning

- Search over block sizes
  - Number of rows/columns in a matrix block
    - \* Measure speed for various block sizes
    - \* There's no clear pattern, highly variable search space, so you actually have to search the space.
  - Tiling alone might not be enough
    - \* Even with best block size, isn't a clear gain in performance across board
- Optimizing in practice
  - Hand optimization (e.g. loop unrolling, named register variable) are done automatically in many software packages (ATLAS, PhiPack) based on machine
  - Removing false dependencies
    - \* Reorder operations to remove them, (instruction level parallelism)
  - Exploit multiple registers
    - \* Reduce demands on memory bandwidth by preloading into local variables (registers)
    - \* Example is convolution
  - Loop Unrolling
    - \* Expose instruction level parallelism
    - \* Multiple statements within one pass through loop
    - \* Reduce read/writes, reusing values
  - Expose Independent Operations

- Copy Optimizations
  - \* Copy input operands or blocks
  - \* Shoaib: Can do loop unrolling much easier
- ATLAS M mult, DGEMM  $n = 500$ 
  - \* Machine produced code can sometimes outperform the best expert implementations
- Search vs. Modeling
  - \* Model does well on most cases, but full force search outperforms
- Locality in Other Algorithms
  - Performance of any algorithm is limited by  $q$
  - In MM mult, increase  $q$  by changin computation order
    - \* Reuse data in cache (increase temporal locality)
  - Tuning is still an open problem for certain routines (e.g. sparse matrix, trees)
- Dense Linear algebra isn't just Matrix Mult
  - BLAS (Basic Linear Algebra Subroutines)
    - \* BLAS 1 vector operations 1970's
    - \* BLAS 2 matrix-vector operations mid 80's
    - \* BLAS 3 matrix-matrix operations late 80's
  - See LAPACK and ScaLAPACK
  - A lot of vendors need people to go through and optimize BLAS
  - BLAS 3 approaches peak while BLAS 1,2 do not.
  - Horst suggested being conscious of matrix size in your implementation of MM mult. (zero pad to faster, larger, more cache-conscious size may help)
- Other automatic tuning efforts
  - FFTW
    - \* Fastest fourier transform
  - Spiral
    - \* Signal processing, fft and beyond
  - See slides

## 3.6 Conclusion

- Summary
  - See slide 45
  - Horst will offer prizes for best team
    - \* But he needs to find something interesting
- Questions You Should be able to answer
  - See slide 47
- Will find out how to get on Jacquard at Friday's discussion
  - Accounts should start coming today