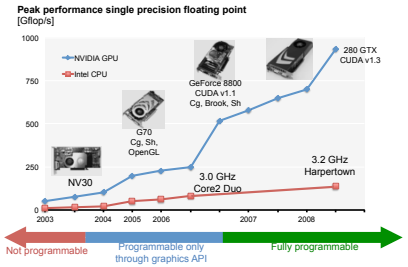
Automatic Generation of FFT Libraries for GPUs

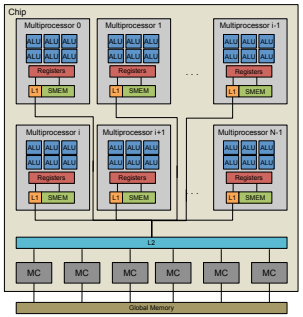
Christos Angelopoulos, Franz Franchetti and Markus Pueschel

Carnegie Mellon SPIRAL www.spiral.net

GPUs and Programmability



GPU Architecture Model



Shared Memory Optimized GPU DFT Algorithm

Original Stockham :

$$\mathrm{DFT}_{r^n} \rightarrow \prod_{i=0}^{n-1} (\mathrm{DFT}_r \otimes \mathrm{I}_{r^{n-1}}) \mathrm{T}^{r^n}_{l} (\mathrm{L}^{r^{n-1}}_{r} \otimes \mathrm{I}_{r^l})$$

Rewriting

Rules

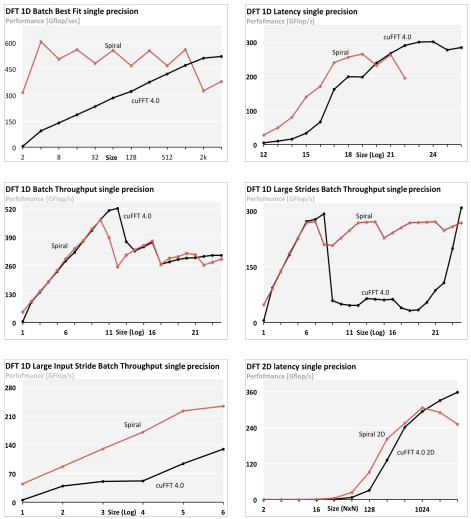
1. Loop Splitting
2. Loop Interchange
3. Loop Tiling for unit stride outputs
4. Cyclic Shift Property to avoid bank conflicts

GPU Stockham :

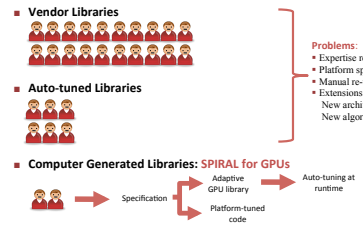
$$\mathrm{DFT}_{r^n} \rightarrow \prod_{i=0}^{n-1} (((\mathrm{D}^{l}_{r} \cdot \mathrm{DFT}_r \cdot Z^{l}_{A} \otimes \mathrm{I}_{\rho} \otimes_j \mathrm{I}_{\mu} \otimes \mathrm{I}_{\frac{r^n}{\rho\mu}}) \mathrm{L}^{r^{n-1}}_{r}) \otimes \mathrm{I}_{\rho} \otimes \mathrm{I}_{\mu} \otimes \mathrm{I}_{\frac{r^n}{\rho\mu}}) (\mathrm{I}_{\rho} \otimes \mathrm{L}^{\frac{r^n}{\rho}}_{\rho\mu}) (\mathrm{I}_{\frac{m}{p\mu}} \otimes \mathrm{L}^{p\mu}_{p}) \mathrm{T}^{r^{n-1}}_{r^{n-1}} (\mathrm{L}^{r^{n-1}}_{r} \otimes \mathrm{I}_{r^l})$$

- Inplace algorithm
- Minimize global memory transfers
- Use Shared memory
- Register to register computations

Results on the GTX 480



Philosophy



Architecture

- 15 Multiprocessors
- 32 cores per multiprocessor
- 32 K registers per multiprocessor
- 48 KB of shared memory
- 16 KB of L1 cache
- 768 KB of L2 cache
- 1.5 GB of GPU Memory

Restrictions

- Banked Shared Memory
  - 32 banks
    - Within one warp resolve bank conflicts
    - Every thread in the warp Reads/Writes at different bank
    - 32 threads in a warp to 32 banks
- Register pressure
  - Max registers per MP = 32K/# of threads per MP
- Uncommon Architectural Model
  - Size of registers > Size of caches
- Global Memory
  - Only block transfers, using caches
  - Double buffering

- Key problems
  - Parallelism, Vectorization, Memory Hierarchy
- Domain Specific Language

$$\mathrm{A} \otimes \mathrm{B} = [\alpha_{k,l}\mathrm{B}], \text{ for } \mathrm{A} = [\alpha_{k,l}]$$

- Address Hardware Characteristics
- Advantages
  - Efficient handling of complexity
  - Efficient porting to new platforms

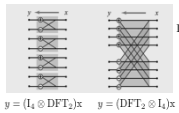
Identify hardware parameters $vector(v)$

Identify formulas $\mathrm{A} \otimes \mathrm{I}_v, \quad D_n, \quad \mathrm{L}^{v^2}_{v}$

Derive algorithm $\mathrm{DFT}_{mn} = (\mathrm{DFT}_k \otimes \mathrm{I}_m)\mathrm{T}^{mn}_{n}\mathrm{L}^{n}_{k}(\mathrm{DFT}_m \otimes \mathrm{I}_k)$

Generate platform oriented algorithm $\mathrm{DFT}_{mn} = ((\mathrm{DFT} \otimes \mathrm{I}_{n/v} \otimes \mathrm{I}_v)\mathrm{T}^{mn}_{n}(\mathrm{I}_{m/v} \otimes (\mathrm{I}_{n/v} \otimes \mathrm{L}^{v^2}_{v}) (\mathrm{L}^{n}_{n/v} \otimes \mathrm{I}_v)(\mathrm{DFT}_n \otimes \mathrm{I}_v))(\mathrm{L}^{mn/v}_{m/v} \otimes \mathrm{I}_v)$

Generate Platform Tuned Code

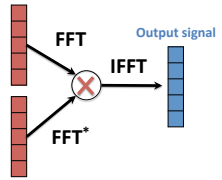


$y = (\mathrm{I}_4 \otimes \mathrm{DFT}_2)x \qquad y = (\mathrm{DFT}_2 \otimes \mathrm{I}_4)x$

Next Step

Correlation (Frequency Domain)

- Code generator
  - Only one data transfer from CPU DRAM to GPU
  - Minimize GPU DRAM memory roundtrips
- Application Scenario
  - PDE solvers
    - Huge correlations



Future Work

- Fast PDE solvers on GPUs

Forward Problem: Match Algorithm to Architecture

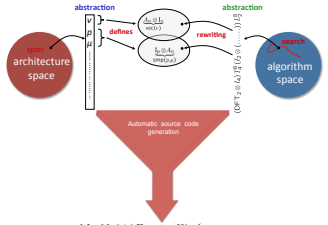
Platform specification

Architectural parameter:
Vector length, #processors, ...

Problem specification

Model: common abstraction
= spaces of matching formulas

Transform:
problem size, kernel choice

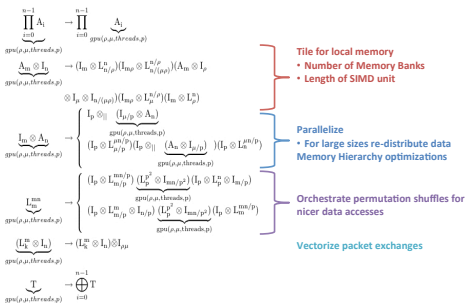



```
void dft64(float* X) {
    a283 = ((20*i10) + i16);
    s97 = s_data[a283];
    ...
    t137 = (s101 + s103);
    t138 = (s102 + s104);
    t139 = (s101 - s103);
    t140 = (s102 - s104);
    ...
    a296 = ((4*i10) + i16);
    s_data[a296] = (t137 + t141);
}
```

Customized, high-performance code for GPU platform
Mapped to target architecture by construction

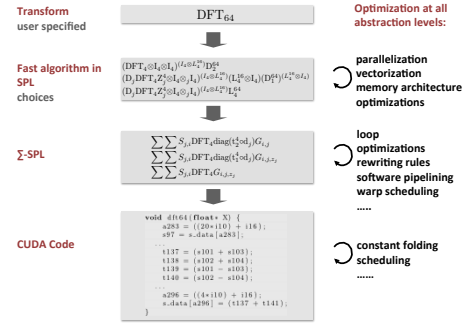
Automatic Library Generation With Spiral

GPU Architectural Constrains in Formulas



Algorithm & Program Generation

GPU Code Through Formula Rewriting



Iteration of this process to search for the fastest

This work was supported by DARPA DESA Program and Nvidia

References:

1. F. Franchetti, M. Püschel, Y. Voronenko, Sr. Chellappa and J. M. F. Moura
   Discrete Fourier Transform on Multicore
   IEEE Signal Processing Magazine, special issue on ``Signal Processing on Platforms with Multiple Cores'', Vol. 26, No. 6, pp. 90-102, 2009
2. M. Püschel, J. M. F. Moura, J. Johnson, D. Padua, M. Veloso, B. Singer, J. Xiong, F. Franchetti, A. Gacic, Y. Voronenko, K. Chen, R. W. Johnson and N. Rizzolo
   SPIRAL: Code Generation for DSP Transforms Proceedings of the IEEE, special issue on "Program Generation, Optimization, and Adaptation", Vol. 93, No. 2, pp. 232- 275, 2005
3. F. Franchetti, Y. Voronenko and M. Püschel
   FFT Program Generation for Shared Memory: SMP and Multicore Proc. Supercomputing (SC), 2006