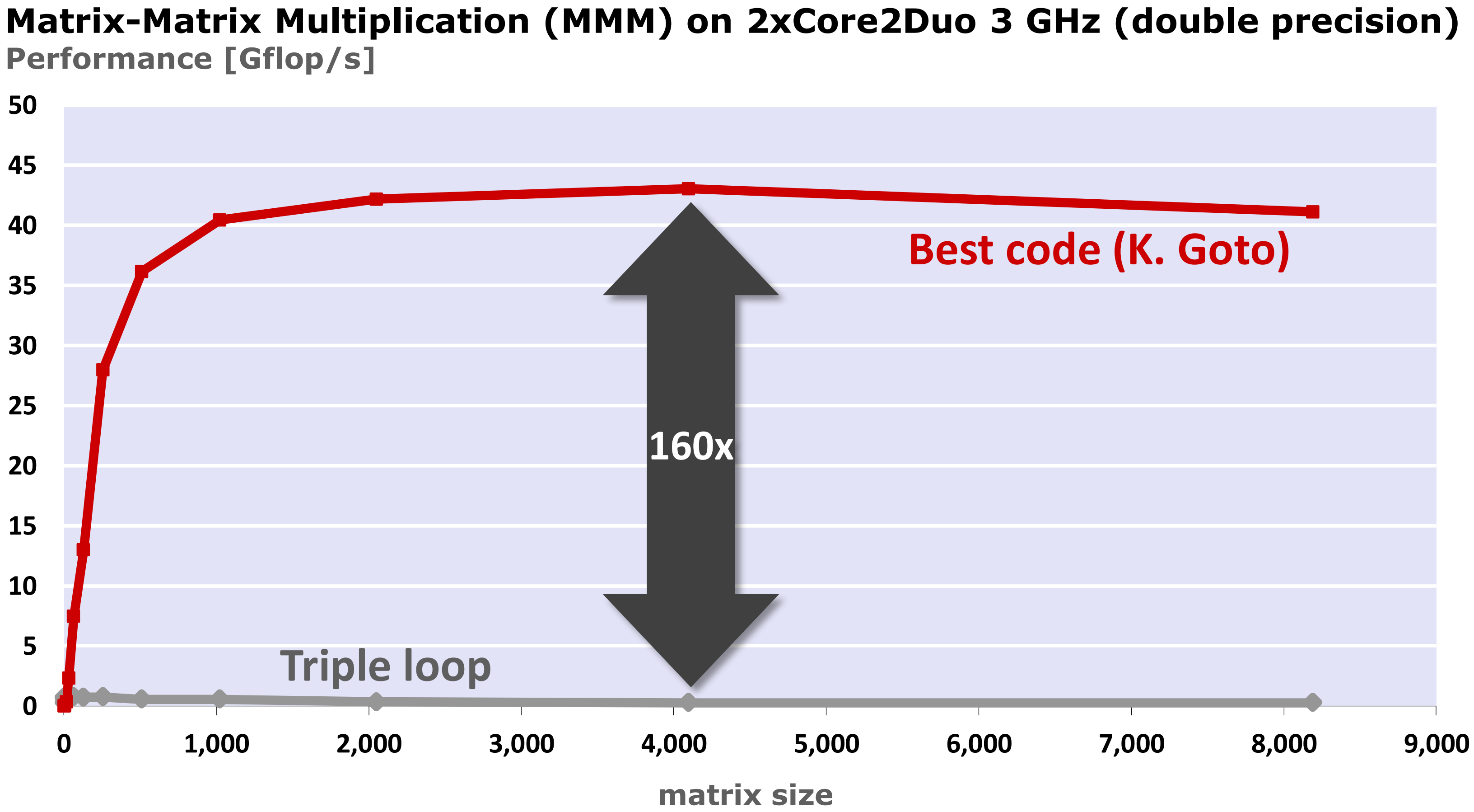
Operator Language: A Program Generation Framework for Fast Kernels

Franz Franchetti, Frédéric de Mesmay, Daniel McFarlin, Markus Püschel

Electrical and Computer Engineering
Carnegie Mellon University

Sponsors: DARPA DESA program, NSF-NGS/ITR, NSF-ACR, and Intel

The Problem: Example MMM



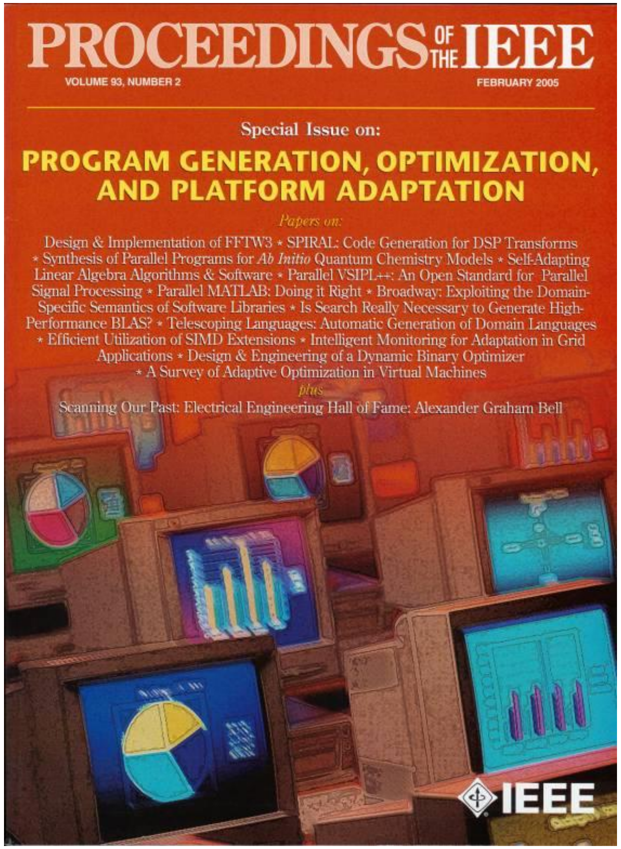
- Similar plots can be shown for all numerical kernels in linear algebra, signal processing, coding, crypto, …
- *What's going on? Hardware is becoming increasingly complex.*

Automatic Performance Tuning

- **Current vicious circle: Whenever a new platform comes out, the same functionality needs to be rewritten and reoptimized**

- **Automatic Performance Tuning**
  - BLAS: ATLAS, PHiPAC
  - Linear algebra: Sparsity/OSKI, Flame
  - Sorting
  - Fourier transform: FFTW
  - Linear transforms (and beyond): Spiral
  - …others

How to build an extensible system?
For more problem classes?
For yet un-invented platforms?



PROCEEDINGS OF THE IEEE

VOLUME 93, NUMBER 2 FEBRUARY 2005

Special Issue on:

PROGRAM GENERATION, OPTIMIZATION, AND PLATFORM ADAPTATION

Papers on:

Design & Implementation of FFTW3 ⋆ SPIRAL: Code Generation for DSP Transforms ⋆ Synthesis of Parallel Programs for *Ab Initio* Quantum Chemistry Models ⋆ Self-Adapting Linear Algebra Algorithms & Software ⋆ Parallel VSIPL++: An Open Standard for Parallel Signal Processing ⋆ Parallel MATLAB: Doing it Right ⋆ Broadway: Exploiting the Domain-Specific Semantics of Software Libraries ⋆ Is Search Really Necessary to Generate High-Performance BLAS? ⋆ Telescoping Languages: Automatic Generation of Domain Languages ⋆ Efficient Utilization of SIMD Extensions ⋆ Intelligent Monitoring for Adaptation in Grid Applications ⋆ Design & Engineering of a Dynamic Binary Optimizer ⋆ A Survey of Adaptive Optimization in Virtual Machines

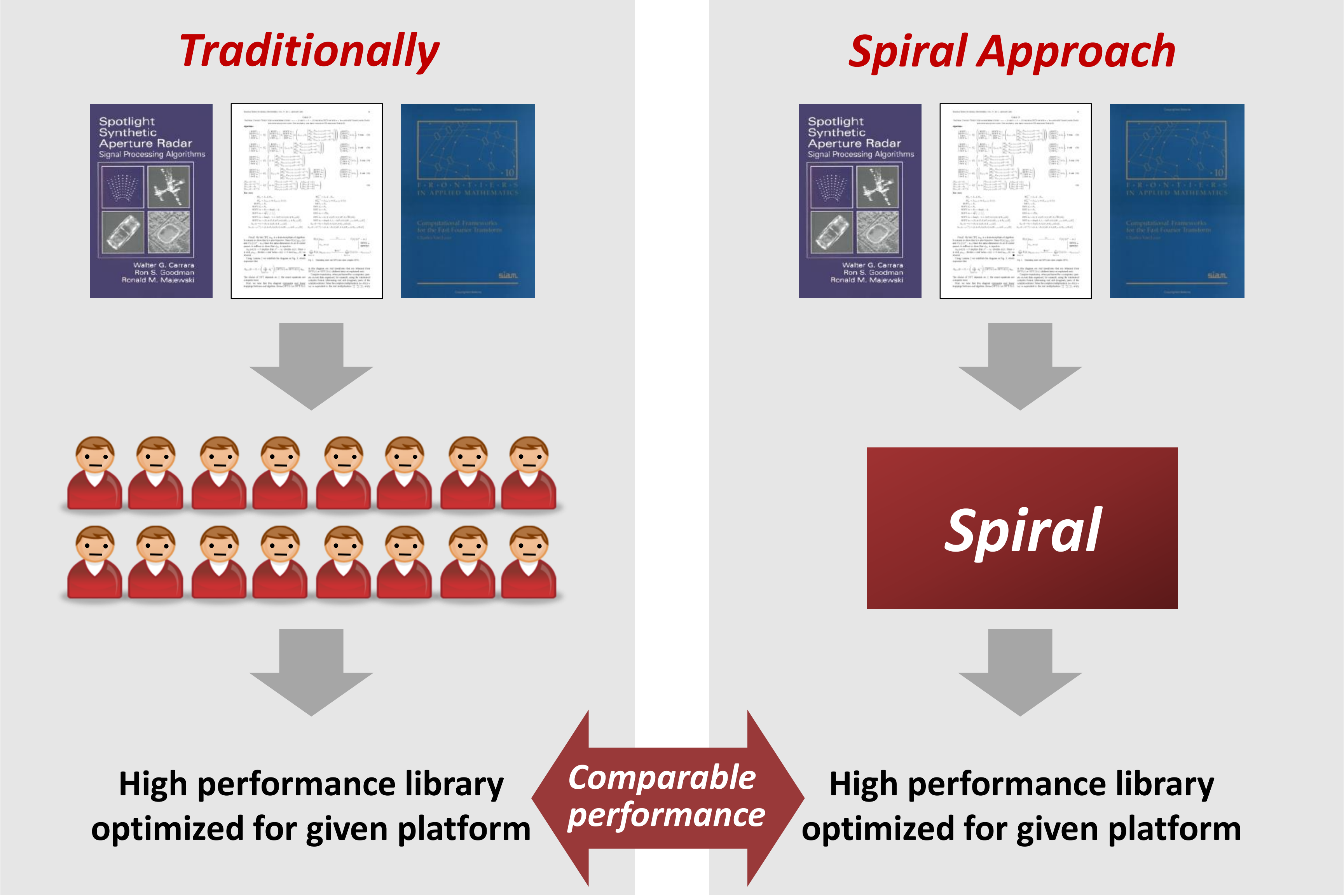
plus

Scanning Our Past: Electrical Engineering Hall of Fame: Alexander Graham Bell

IEEE

Proceedings of the IEEE special issue, Feb. 2005

What is Spiral?



Traditionally

High performance library optimized for given platform

Spiral Approach

Spiral

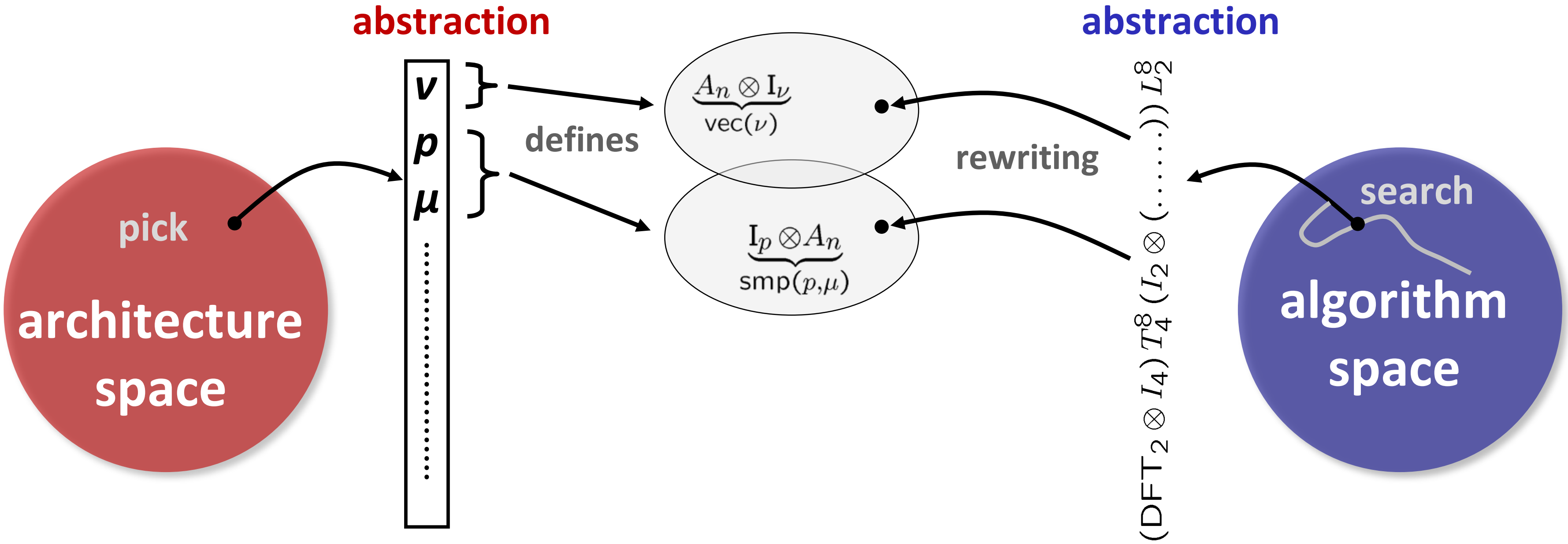
High performance library optimized for given platform

Comparable performance

Spotlight Synthetic Aperture Radar
Signal Processing Algorithms
Walter G. Carrara
Ron S. Goodman
Ronald M. Majewski
FRONTIERS IN APPLIED MATHEMATICS
Computational Frameworks for the Fast Fourier Transform
siam

Idea: Common Abstraction and Rewriting

Model: common abstraction
= spaces of matching formulas
= domain-specific language


abstraction
abstraction
ν
p
μ
defines
$\underbrace{A_n \otimes \mathrm{I}_\nu}_{\mathrm{vec}(\nu)}$
$\underbrace{\mathrm{I}_p \otimes A_n}_{\mathrm{smp}(p,\mu)}$
rewriting
$(\mathrm{DFT}_2 \otimes I_4)\, T^8_4 \,(I_2 \otimes (\ldots\ldots))\, L^8_2$
pick
architecture space
search
algorithm space

Architectural parameter:
Vector length,
#processors, …

optimization

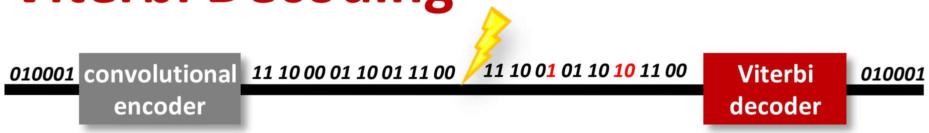
Kernel:
problem size,
algorithm choice

Some Kernels as OL Formulas.

Linear Transforms

$$
\begin{aligned}
\mathbf{DFT}_n &\to (\mathbf{DFT}_k \otimes \mathrm{I}_m)\, \mathrm{T}^n_m (\mathrm{I}_k \otimes \mathbf{DFT}_m)\, \mathrm{L}^n_k, \quad n = km \\
\mathbf{DFT}_n &\to P_n (\mathbf{DFT}_k \otimes \mathbf{DFT}_m) Q_n, \quad n = km,\ \mathrm{gcd}(k,m) = 1 \\
\mathbf{DFT}_p &\to R_p^T (\mathrm{I}_1 \oplus \mathbf{DFT}_{p-1}) D_p (\mathrm{I}_1 \oplus \mathbf{DFT}_{p-1}) R_p, \quad p \text{ prime} \\
\mathbf{DCT\text{-}3}_n &\to (\mathrm{I}_m \oplus \mathrm{J}_m)\, \mathrm{L}^n_m (\mathbf{DCT\text{-}3}_m(1/4) \oplus \mathbf{DCT\text{-}3}_m(3/4)) \\
&\qquad \cdot (\mathrm{F}_2 \otimes \mathrm{I}_m) \begin{bmatrix} \mathrm{I}_m & 0 \oplus -\mathrm{J}_{m-1} \\ & \frac{1}{\sqrt{2}} (\mathrm{I}_1 \oplus 2\, \mathrm{I}_m) \end{bmatrix}, \quad n = 2m \\
\mathbf{DCT\text{-}4}_n &\to S_n \mathbf{DCT\text{-}2}_n\, \mathrm{diag}_{0 \le k < n} (1/(2 \cos((2k+1)\pi/4n))) \\
\mathbf{IMDCT}_{2m} &\to (\mathrm{J}_m \oplus \mathrm{I}_m \oplus \mathrm{I}_m \oplus \mathrm{J}_m) \left(\left(\begin{bmatrix} 1 \\ -1 \end{bmatrix} \otimes \mathrm{I}_m \right) \oplus \left(\begin{bmatrix} -1 \\ -1 \end{bmatrix} \otimes \mathrm{I}_m \right) \right) \mathrm{J}_{2m}\, \mathbf{DCT\text{-}4}_{2m} \\
\mathbf{WHT}_{2^k} &\to \prod_{i=1}^{t} (\mathrm{I}_{2^{k_1 + \cdots + k_{i-1}}} \otimes \mathbf{WHT}_{2^{k_i}} \otimes \mathrm{I}_{2^{k_{i+1} + \cdots + k_t}}), \quad k = k_1 + \cdots + k_t \\
\mathbf{DFT}_2 &\to \mathrm{F}_2 \\
\mathbf{DCT\text{-}2}_2 &\to \mathrm{diag}(1, 1/\sqrt{2})\, \mathrm{F}_2 \\
\mathbf{DCT\text{-}4}_2 &\to \mathrm{J}_2\, \mathrm{R}_{13\pi/8}
\end{aligned}
$$

Viterbi Decoding



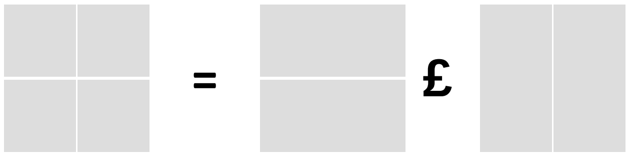
$$
\underbrace{\mathbf{Vit}}_{\mathtt{vec}(v)} \to \underbrace{\left(\prod (L \times I) \circ (I \otimes C) \right) \circ Id}_{\mathtt{vec}(v)}
$$

$$
\to \left(\prod \underbrace{(L \times I) \circ (I \otimes C)}_{\mathtt{vec}(v)} \right) \circ Id
$$

$$
\text{£} \quad \to \left(\prod (L \otimes I_v \times I) \circ (I \otimes C \otimes I_v) \circ (\vec{L} \times I) \right) \circ Id
$$

$$
\to \prod (L \otimes I_v \times I) \circ (I \otimes (B \otimes I_v)) \circ (\vec{L} \times I)
$$

Matrix-Matrix Multiplication



$$\mathsf{MMM}_{1,1,1} \to (\cdot)_1$$

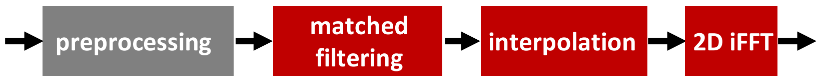
$$\mathsf{MMM}_{m,n,k} \to (\otimes)_{m/m_b \times 1} \otimes \mathsf{MMM}_{m_b,n,k}$$

$$\mathsf{MMM}_{m,n,k} \to \mathsf{MMM}_{m,nb,k} \otimes (\otimes)_{1 \times n/nb}$$

$$
\begin{aligned}
\mathsf{MMM}_{m,n,k} \to\ & ((\Sigma_{k/k_b} \circ (\cdot)_{k/k_b}) \otimes \mathsf{MMM}_{m,n,k_b}) \circ \\
& ((L^{mk/k_b}_{k/k_b} \otimes I_{k_b}) \times I_{kn})
\end{aligned}
$$

$$
\begin{aligned}
\mathsf{MMM}_{m,n,k} \to\ & (L^{mn/n_b}_m \otimes I_{n_b}) \circ \\
& ((\otimes)_{1 \times n/n_b} \otimes \mathsf{MMM}_{m,n_b,k}) \circ \\
& (I_{km} \times (L^{kn/n_b}_{n/n_b} \otimes I_{n_b}))
\end{aligned}
$$

Synthetic Aperture Radar (SAR)



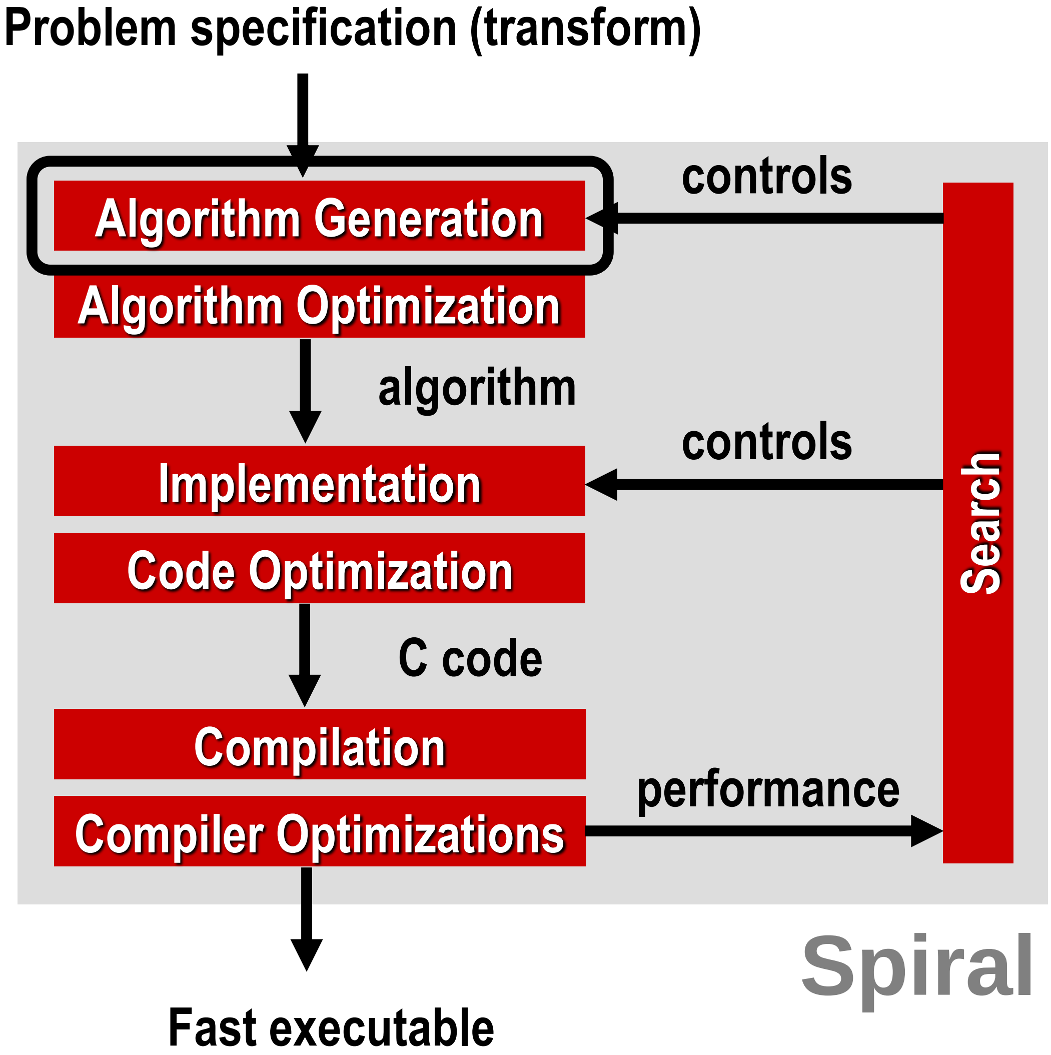
$$
\begin{aligned}
\mathsf{SAR}_{k \times m \to n \times n} &\to \mathsf{DFT}_{n \times n} \circ \mathsf{Interp}_{k \times m \to n \times n} \\
\mathsf{DFT}_{n \times n} &\to (\mathsf{DFT}_n \otimes \mathrm{I}_n) \circ (\mathrm{I}_n \otimes \mathsf{DFT}_n) \\
\mathsf{Interp}_{k \times m \to n \times n} &\to (\mathsf{Interp}_{k \to n} \otimes_i \mathrm{I}_n) \circ (\mathrm{I}_k \otimes_i \mathsf{Interp}_{m \to n}) \\
\mathsf{Interp}_{r \to s} &\to \left(\bigoplus_{i=0}^{n-2} \mathsf{InterpSeg}_k \right) \oplus \mathsf{InterpSegPruned}_{k,\ell} \\
\mathsf{InterpSeg}_k &\to \mathsf{G}_f^{u \cdot n \to k} \circ \mathsf{iPrunedDFT}_{n \to u \cdot n} \circ \left(\frac{1}{n} \right) \circ \mathsf{DFT}_n
\end{aligned}
$$

How Spiral Works

Spiral:
Complete automation of the implementation and optimization task

Basic ideas:
Declarative representation of algorithms

Rewriting systems to generate and optimize algorithms at a high level of abstraction



Markus Püschel, José M. F. Moura, Jeremy Johnson, David Padua, Manuela Veloso, Bryan Singer, Jianxin Xiong, Franz Franchetti, Aca Gacic, Yevgen Voronenko, Kang Chen, Robert W. Johnson, and Nick Rizzolo:
SPIRAL: Code Generation for DSP Transforms. Special issue, Proceedings of the IEEE 93(2), 2005

Organization

- Operator language and algorithms
- Optimizing algorithms for platforms
- Performance results
- Summary

Organization

- **Operator language and algorithms**
- Optimizing algorithms for platforms
- Performance results
- Summary

Operators

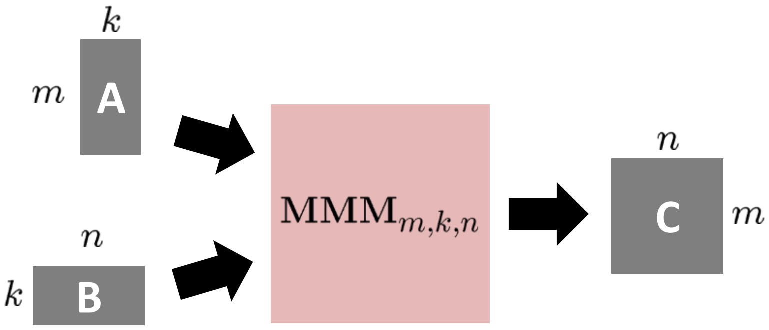
Definition

- Operator: Multiple complex vectors ! multiple complex vectors
- Higher-dimensional data is linearized
- Operators are potentially nonlinear

$$\mathsf{M} : \begin{cases} \mathbb{C}^{n_0} \times \cdots \times \mathbb{C}^{n_{k-1}} \to \mathbb{C}^{N_0} \times \cdots \times \mathbb{C}^{N_{\ell-1}} \\ (\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{k-1}) \mapsto \mathsf{M}(\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{k-1}) \end{cases}$$

Example: Matrix-matrix-multiplication (MMM)

$$\begin{aligned} \mathrm{MMM}_{m,k,n} : \mathbb{R}^{mk} \times \mathbb{R}^{kn} &\to \mathbb{R}^{mn} \\ (\mathrm{A}, \mathrm{B}) &\mapsto \mathrm{AB} \end{aligned}$$



Operator Language

name	definition
Linear, arity (1,1)	
identity	$I_n : \mathbb{C}^n \rightarrow \mathbb{C}^n;\ \mathbf{x} \mapsto \mathbf{x}$
vector flip	$J_n : \mathbb{C}^n \rightarrow \mathbb{C}^n;\ (x_i) \mapsto (x_{n-i})$
transposition of an $m \times n$ matrix	$L^{mn}_m : \mathbb{C}^{mn} \rightarrow \mathbb{C}^{mn};\ \mathbf{A} \mapsto \mathbf{A}^T$
matrix $M \in \mathbb{C}^{m \times n}$	$\mathrm{M} : \mathbb{C}^n \rightarrow \mathbb{C}^m; \mathbf{x} \mapsto M\mathbf{x}$
Multilinear, arity (2,1)	
Point-wise product	$\mathrm{P}_n : \mathbb{C}^n \times \mathbb{C}^n \rightarrow \mathbb{C}^n;\ ((x_i),(y_i)) \mapsto (x_i y_i)$
Scalar product	$\mathrm{S}_n : \mathbb{C}^n \times \mathbb{C}^n \rightarrow \mathbb{C};\ ((x_i),(y_i)) \mapsto \Sigma(x_i y_i)$
Kronecker product	$\mathrm{K}_{m \times n} : \mathbb{C}^m \times \mathbb{C}^n \rightarrow \mathbb{C}^{mn};\ ((x_i),\mathbf{y})) \mapsto (x_i \mathbf{y})$
Others	
Fork	$\mathrm{Fork}_n : \mathbb{C}^n \rightarrow \mathbb{C}^n \times \mathbb{C}^n;\ \mathbf{x} \mapsto (\mathbf{x},\mathbf{x})$
Split	$\mathrm{Split}_n : \mathbb{C}^n \rightarrow \mathbb{C}^{n/2} \times \mathbb{C}^{n/2};\ \mathbf{x} \mapsto (\mathbf{x}^U, \mathbf{x}^L)$
Concatenate	$\oplus_n : \mathbb{C}^n \times \mathbb{C}^m \rightarrow \mathbb{C}^{n+m};\ (\mathbf{x}, \mathbf{y}) \mapsto \mathbf{x} \oplus \mathbf{y}$
Duplication	$\mathrm{dup}^m_n : \mathbb{C}^n \rightarrow \mathbb{C}^{nm};\ (\mathbf{x} \mapsto \mathbf{x} \otimes I_m$
Min	$\mathrm{min}_n : \mathbb{C}^n \times \mathbb{C}^n \rightarrow \mathbb{C}^n;\ (\mathbf{x}, \mathbf{y}) \mapsto (\min(x_i, y_i))$
Max	$\mathrm{max}_n : \mathbb{C}^n \times \mathbb{C}^n \rightarrow \mathbb{C}^n;\ (\mathbf{x}, \mathbf{y}) \mapsto (\max(x_i, y_i))$

OL Tensor Product: Repetitive Structure

Kronecker product

(structured matrices)

$$\mathrm{I}_m \otimes A_n = \begin{bmatrix} A_n & & \\ & \ddots & \\ & & A_n \end{bmatrix}$$

OL Tensor product

(structured operators)

$$\mathrm{I}_{M/M_B \times 1 \rightarrow M/M_B} \otimes \mathsf{MMM}_{M_B,N,K} = \left((A,B) \mapsto \left[\begin{array}{c} A_0 \\ \hline \vdots \\ \hline A_{M/M_B-1} \end{array} \right] B \right)$$

Definition

(extension to non-linear)

$$(\mathrm{I}_{m\times n \rightarrow mn} \otimes \mathsf{A}) \left(\sum_{i=0}^{m-1} \mathbf{e}_i^m \otimes \mathbf{x}, \sum_{i=0}^{n-1} \mathbf{e}_i^n \otimes \mathbf{y} \right) = \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} \mathbf{e}_i^m \otimes \mathbf{e}_j^n \otimes \mathsf{A}(\mathbf{x}, \mathbf{y})$$

$$(\mathsf{A} \otimes \mathrm{I}_{m\times n \rightarrow mn}) \left(\sum_{i=0}^{m-1} \mathbf{x} \otimes \mathbf{e}_i^m, \sum_{i=0}^{n-1} \mathbf{y} \otimes \mathbf{e}_i^n \right) = \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} \mathsf{A}(\mathbf{x}, \mathbf{y}) \otimes \mathbf{e}_i^m \otimes \mathbf{e}_j^n$$

Translating OL Formulas Into Programs

operator formula	code

$\mathbf{r} = (\mathsf{A}_{K\times M\to N} \circ \mathsf{B}_{k\times m\to K\times M})(\mathbf{x},\mathbf{y})$	`(t, u) = B(x, y);`
	`r = A(t, u);`

$(\mathbf{r},\mathbf{s}) = (\mathsf{A}_{m\to M} \times \mathsf{B}_{n\to N})(\mathbf{x},\mathbf{y})$	`r = A(x);`
	`s = B(y);`

$\mathbf{r} = (\mathrm{I}_{m\times n\to mn} \otimes A_{M\times N\to K})(\mathbf{x},\mathbf{y})$	`for (i=0;i<m;i++)`
	`    for (j=0;j<m;j++)`
	`        r[(i*m+j)*K:1:(i*m+j+1)*K-1] =`
	`            A(x[i*M:1:i*M+M-1], y[j*N:1:j*N+N-1]);`

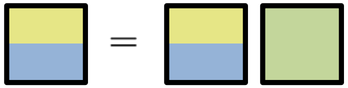
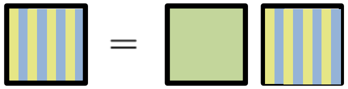
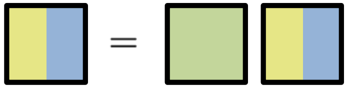
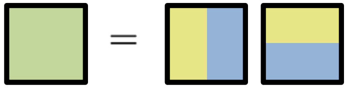
$\mathbf{r} = (A_{M\times N\to K} \otimes \mathrm{I}_{m\times n\to mn})(\mathbf{x},\mathbf{y})$	`for (i=0;i<m;i++)`
	`    for (j=0;j<n;j++)`
	`        r[i*m+j:m*n:i*m+j+m*n*(K-1))] =`
	`            A(x[i:m:i+m*(M-1))], y[j:n:j+n*(N-1)]);`

$$\mathrm{I}_{M/M_B\times 1\to M/M_B} \otimes \mathsf{MMM}_{M_B,N,K} = \left((A,B) \mapsto \left[\begin{array}{c} A_0 \\ \hline \vdots \\ \hline A_{M/M_B-1} \end{array}\right] B\right)$$

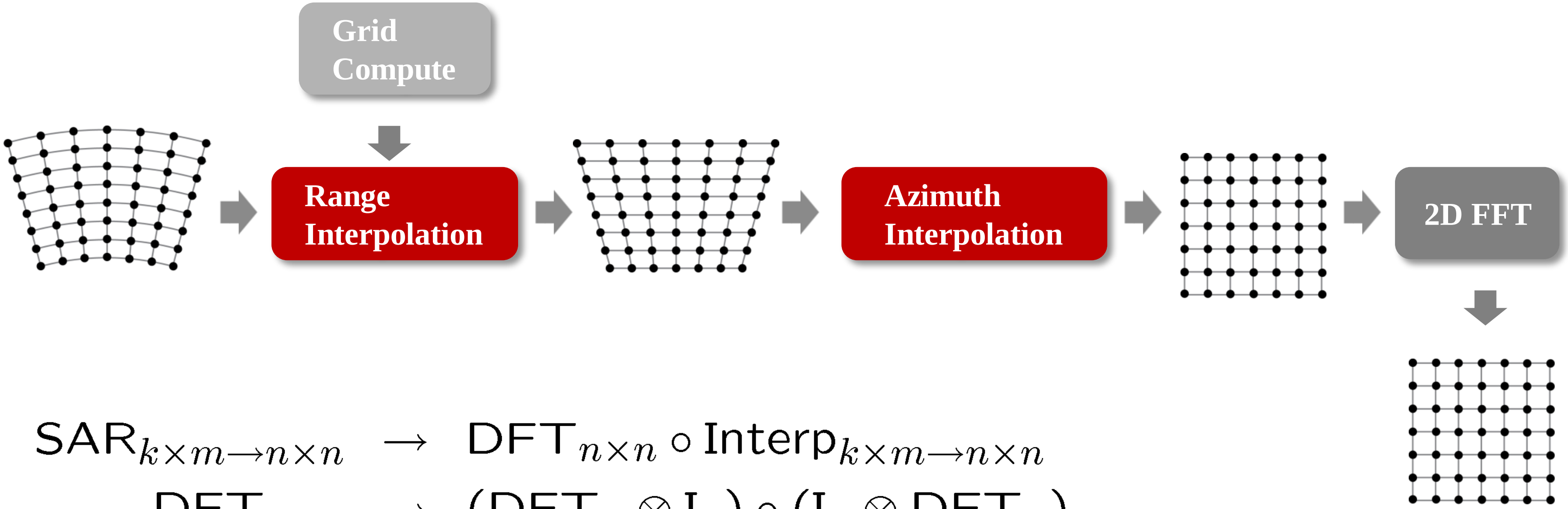
Example: Matrix Multiplication (MMM)

Breakdown rules:

capture various forms of blocking

breakdown rule	description
$\mathrm{MMM}_{1,1,1} \rightarrow (\cdot)_1$	base case
$\mathrm{MMM}_{m,n,k} \rightarrow (\otimes)_{m/m_b \times 1} \otimes \mathrm{MMM}_{m_b,n,k}$	horizontal blocking 
$\mathrm{MMM}_{m,n,k} \rightarrow \mathrm{MMM}_{m,nb,k} \otimes (\otimes)_{1 \times n/nb}$	interleaved blocking 
$\mathrm{MMM}_{m,n,k} \rightarrow ((\Sigma_{k/k_b} \circ (\cdot)_{k/k_b}) \otimes \mathrm{MMM}_{m,n,k_b}) \circ ((L^{mk/k_b}_{k/k_b} \otimes I_{k_b}) \times I_{kn})$	accumulative blocking 
$\mathrm{MMM}_{m,n,k} \rightarrow (L^{mn/n_b}_{m} \otimes I_{n_b}) \circ ((\otimes)_{1 \times n/n_b} \otimes \mathrm{MMM}_{m,n_b,k}) \circ (I_{km} \times (L^{kn/n_b}_{n/n_b} \otimes I_{n_b}))$	vertical blocking 

Example: SAR Computation as OL Rules



Grid Compute

Range Interpolation

Azimuth Interpolation

2D FFT

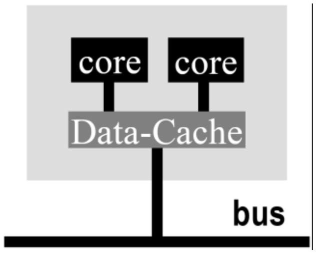
$$
\begin{aligned}
\mathrm{SAR}_{k\times m\to n\times n} &\to \mathrm{DFT}_{n\times n}\circ \mathrm{Interp}_{k\times m\to n\times n}\\
\mathrm{DFT}_{n\times n} &\to (\mathrm{DFT}_n \otimes \mathrm{I}_n)\circ(\mathrm{I}_n\otimes \mathrm{DFT}_n)\\
\mathrm{Interp}_{k\times m\to n\times n} &\to (\mathrm{Interp}_{k\to n}\otimes_i \mathrm{I}_n)\circ(\mathrm{I}_k\otimes_i \mathrm{Interp}_{m\to n})\\
\mathrm{Interp}_{r\to s} &\to \left(\bigoplus_{i=0}^{n-2}\mathrm{InterpSeg}_k\right)\oplus \mathrm{InterpSegPruned}_{k,\ell}\\
\mathrm{InterpSeg}_k &\to \mathrm{G}_f^{u\cdot n\to k}\circ \mathrm{iPrunedDFT}_{n\to u\cdot n}\circ\left(\frac{1}{n}\right)\circ \mathrm{DFT}_n
\end{aligned}
$$

Organization

- Operator language and algorithms
- **Optimizing algorithms for platforms**
- Performance results
- Summary

Modeling Multicore: Base Cases

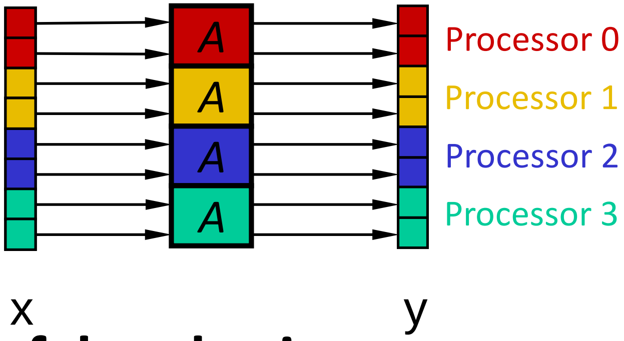
- Hardware abstraction: shared cache with cache lines



$$\underbrace{A_{k\times m\to n}}_{\mathsf{smp}(p,\mu)}$$

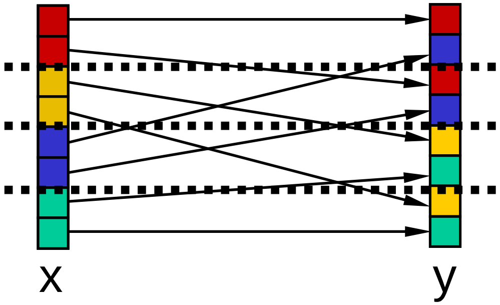
- Tensor product: embarrassingly parallel operator

$$y = \big(\mathrm{I}_p \otimes A\big)(x)$$



- Permutation: problematic; may produce false sharing

$$y = \mathrm{L}^8_4(x)$$



Parallelization: OL Rewriting Rules

- Tags encode hardware constraints
- Rules are algorithm-independent
- Rules encode program transformations

$$\underbrace{\left(\mathrm{I}_k \otimes \mathrm{L}^{mn}_{n}\right)}_{\mathrm{smp}(p,\mu)} \circ \underbrace{\mathrm{L}^{kmn}_{km}}_{\mathrm{smp}(p,\mu)} \to \left(\mathrm{L}^{kn}_{k} \otimes \mathrm{I}_{m/\mu}\right) \bar{\otimes}\, \mathrm{I}_{\mu}$$

$$\underbrace{\mathrm{L}^{kmn}_{n}}_{\mathrm{smp}(p,\mu)} \circ \underbrace{\left(\mathrm{I}_k \otimes \mathrm{L}^{mn}_{m}\right)}_{\mathrm{smp}(p,\mu)} \to \left(\mathrm{L}^{kn}_{n} \otimes \mathrm{I}_{m/\mu}\right) \bar{\otimes}\, \mathrm{I}_{\mu}$$

$$\underbrace{\mathrm{A} \circ \mathrm{B}}_{\mathrm{smp}(p,\mu)} \to \underbrace{\mathrm{A}}_{\mathrm{smp}(p,\mu)} \circ \underbrace{\mathrm{B}}_{\mathrm{smp}(p,\mu)}$$

$$\underbrace{\mathrm{A}^{k\times m\to n} \otimes \mathrm{I}^{1\times p\to p}}_{\mathrm{smp}(p,\mu)} \to \underbrace{\mathrm{L}^{pn}_{n}}_{\mathrm{smp}(p,\mu)} \circ \left(\mathrm{I}_{1\times p\to p} \otimes_{\|} \mathrm{A}^{k\times m\to n}\right) \circ \underbrace{\left(\mathrm{I}_k \times \mathrm{L}^{pm}_{p}\right)}_{\mathrm{smp}(p,\mu)}$$

$$\underbrace{\left(\mathrm{A} \times \mathrm{B}\right)}_{\mathrm{smp}(p,\mu)} \circ \underbrace{\left(\mathrm{C} \times \mathrm{D}\right)}_{\mathrm{smp}(p,\mu)} \to \underbrace{\left(\mathrm{A} \circ \mathrm{C}\right)}_{\mathrm{smp}(p,\mu)} \times \underbrace{\left(\mathrm{B} \circ \mathrm{D}\right)}_{\mathrm{smp}(p,\mu)}$$

The Joint Rule Set: MMM

- **Algorithm rules: breakdown rules**

$$\mathsf{MMM}_{m,n,k} \rightarrow (\otimes)_{m/m_b\times 1} \otimes \mathsf{MMM}_{m_b,n,k}$$

$$\mathsf{MMM}_{m,n,k} \rightarrow \mathsf{MMM}_{m,nb,k} \otimes (\otimes)_{1\times n/nb}$$

$$\begin{aligned}\mathsf{MMM}_{m,n,k} \rightarrow ((\Sigma_{k/k_b} \circ (\cdot)_{k/k_b}) \otimes \mathsf{MMM}_{m,n,k_b}) \circ \\ ((L^{mk/k_b}_{k/k_b} \otimes I_{k_b}) \times I_{kn})\end{aligned}$$

- **Hardware constraints: base cases**

$$\mathrm{I}_p \otimes_{\|} A_{m\mu\rightarrow n\mu}, \quad \mathsf{P}_p \otimes_{\|} A_{k\mu\times m\mu\rightarrow n\mu}, \quad \mathsf{K}_{q\times r} \otimes_{\|} A_{k\mu\times m\mu\rightarrow n\mu},$$

$$M \bar{\otimes} \mu, \quad (M \bar{\otimes} \mathrm{I}_\mu) \otimes (N \bar{\otimes} \mathrm{I}_\mu)$$

- **Program transformations: manipulation rules**

$$\underbrace{\left(\mathrm{I}_k \otimes \mathsf{L}^{mn}_n\right)}_{\mathsf{smp}(p,\mu)} \circ \underbrace{\mathsf{L}^{kmn}_{km}}_{\mathsf{smp}(p,\mu)} \rightarrow \left(\mathsf{L}^{kn}_k \otimes \mathrm{I}_{m/\mu}\right) \bar{\otimes} \mathrm{I}_\mu$$

$$\underbrace{\mathsf{L}^{kmn}_{n}}_{\mathsf{smp}(p,\mu)} \circ \underbrace{\left(\mathrm{I}_k \otimes \mathsf{L}^{mn}_m\right)}_{\mathsf{smp}(p,\mu)} \rightarrow \left(\mathsf{L}^{kn}_n \otimes \mathrm{I}_{m/\mu}\right) \bar{\otimes} \mathrm{I}_\mu$$

Combined rule set spans search space for empirical optimization

Parallelization Through Rewriting: MMM

$$\underbrace{\mathsf{MMM}_{m,n,k}}_{\mathsf{smp}(p,\mu)}$$

$$\rightarrow \underbrace{\left(\mathrm{I}_m \otimes \mathsf{L}^n_p\right) \circ \left(\mathsf{MMM}_{m,n/p,k} \otimes (\otimes)_{1\times p \rightarrow p}\right) \circ \left(\mathrm{I}_{km} \times (\mathrm{I}_k \otimes \mathsf{L}^n_{n/p})\right)}_{\mathsf{smp}(p,\mu)}$$

$$\rightarrow \underbrace{\left(\mathrm{I}_m \otimes \mathsf{L}^n_p\right)}_{\mathsf{smp}(p,\mu)} \circ \underbrace{\left(\mathsf{MMM}_{m,n/p,k} \otimes (\otimes)_{1\times p \rightarrow p}\right)}_{\mathsf{smp}(p,\mu)} \circ \underbrace{\left(\mathrm{I}_{km} \times (\mathrm{I}_k \otimes \mathsf{L}^n_{n/p})\right)}_{\mathsf{smp}(p,\mu)}$$

$$\rightarrow \underbrace{\left(\mathrm{I}_m \otimes \mathsf{L}^n_p\right)}_{\mathsf{smp}(p,\mu)} \circ \underbrace{\mathsf{L}^{mn}_{m/pn}}_{\mathsf{smp}(p,\mu)} \circ \left((\otimes)_{1\times p \rightarrow p} \otimes_{\|} \mathsf{MMM}_{m/p,n,k}\right) \circ \underbrace{\left(\mathrm{I}_{km} \times \mathsf{L}^{kn}_p\right)}_{\mathsf{smp}(p,\mu)} \circ \underbrace{\left(\mathrm{I}_{km} \times (\mathrm{I}_k \otimes \mathsf{L}^n_{n/p})\right)}_{\mathsf{smp}(p,\mu)}$$

$$\rightarrow \underline{\left((\mathsf{L}^{mp}_m \otimes \mathrm{I}_{n/(p\mu)}) \bar{\otimes} \mathrm{I}_\mu\right)} \circ \underline{\left((\otimes)_{1\times p \rightarrow p} \otimes_{\|} \mathsf{MMM}_{m,n/p,k}\right)} \circ \underline{\left((\mathrm{I}_{km/\mu} \bar{\otimes} \mathrm{I}_\mu) \times ((\mathsf{L}^{kp}_p \otimes \mathrm{I}_{n/(p\mu)}) \bar{\otimes} \mathrm{I}_\mu)\right)}$$

Load-balanced

No false sharing

Same Approach for Different Paradigms

Threading:

$$\underbrace{\mathbf{DFT}_{mn}}_{\mathsf{smp}(p,\mu)} \rightarrow \underbrace{\left((\mathbf{DFT}_m \otimes \mathrm{I}_n)\mathsf{T}_n^{mn}(\mathrm{I}_m \otimes \mathbf{DFT}_n)\,\mathsf{L}_m^{mn}\right)}_{\mathsf{smp}(p,\mu)}$$

$$\ldots$$

$$\rightarrow \underbrace{\left(\mathbf{DFT}_m \otimes \mathrm{I}_n\right)}_{\mathsf{smp}(p,\mu)} \underbrace{\mathsf{T}_n^{mn}}_{\mathsf{smp}(p,\mu)} \underbrace{\left(\mathrm{I}_m \otimes \mathbf{DFT}_n\right)}_{\mathsf{smp}(p,\mu)} \underbrace{\mathsf{L}_m^{nm}}_{\mathsf{smp}(p,\mu)}$$

$$\ldots$$

$$\rightarrow \left((\mathsf{L}_m^{mp} \otimes \mathrm{I}_{n/p\mu}) \otimes_\mu \mathrm{I}_\mu\right)\left(\mathrm{I}_p \otimes_\parallel (\mathbf{DFT}_m \otimes \mathrm{I}_{n/p})\right)\left((\mathsf{L}_p^{mp} \otimes \mathrm{I}_{n/p\mu}) \otimes_\mu \mathrm{I}_\mu\right)$$

$$\left(\bigoplus_{i=0}^{p-1}{}_{\parallel}\, \mathsf{T}_n^{mn,i}\right)\left(\mathrm{I}_p \otimes_\parallel (\mathrm{I}_{m/p} \otimes \mathbf{DFT}_n)\right)\left(\mathrm{I}_p \otimes_\parallel \mathsf{L}_{m/p}^{mn/p}\right)\left((\mathsf{L}_p^{pn} \otimes \mathrm{I}_{m/p\mu}) \otimes_\mu \mathrm{I}_\mu\right)$$

Vectorization:

$$\underbrace{\left(\overline{\mathbf{DFT}_{mn}}\right)}_{\mathsf{vec}(\nu)} \rightarrow \underbrace{\overline{\left((\mathbf{DFT}_m \otimes \mathrm{I}_n)\mathsf{T}_n^{mn}(\mathrm{I}_m \otimes \mathbf{DFT}_n)\,\mathsf{L}_m^{mn}\right)}}_{\mathsf{vec}(\nu)}$$

$$\ldots$$

$$\rightarrow \overleftarrow{(\underbrace{\mathsf{DFT}_m \otimes \mathrm{I}_n}_{\mathsf{vec}(\nu)})}^{\nu}\, \overline{(\underbrace{\mathsf{T}_n^{mn}}_{\mathsf{vec}(\nu)})}^{\nu}\, \overrightarrow{\underbrace{(\mathrm{I}_m \otimes \mathsf{DFT}_n)\,\mathsf{L}_m^{mn}}_{\mathsf{vec}(\nu)}}^{\nu}$$

$$\ldots$$

$$\rightarrow (\mathrm{I}_{mn/\nu} \otimes \underbrace{\mathsf{L}_\nu^{2\nu}}_{\mathsf{sse}})(\overline{\mathsf{DFT}_m \otimes \mathrm{I}_{n/\nu}} \vec{\otimes}\, \mathrm{I}_\nu)\overline{(\underbrace{\mathsf{T}_n^{mn}}_{\mathsf{sse}})}^{\nu}$$

$$\left(\mathrm{I}_{m/\nu} \otimes (\overline{\mathsf{L}_\nu^n} \vec{\otimes}\, \mathrm{I}_\nu)(\mathrm{I}_{n/\nu} \otimes (\mathsf{L}_\nu^{2\nu} \vec{\otimes}\, \mathrm{I}_\nu)(\mathrm{I}_2 \otimes \underbrace{\mathsf{L}_\nu^{\nu^2}}_{\mathsf{sse}})(\mathsf{L}_2^{2\nu} \vec{\otimes}\, \mathrm{I}_\nu))(\overline{\mathsf{DFT}_n} \vec{\otimes}\, \mathrm{I}_\nu)\right)$$

$$\left((\mathsf{L}_m^{mn} \otimes \mathrm{I}_2) \vec{\otimes}\, \mathrm{I}_\nu\right)(\mathrm{I}_{mn/\nu} \otimes \underbrace{\mathsf{L}_2^{2\nu}}_{\mathsf{sse}})$$

GPUs:

$$\underbrace{\left(\mathbf{DFT}_{r^k}\right)}_{\mathsf{gpu}(t,c)} \rightarrow \underbrace{\left(\prod_{i=0}^{k-1} \mathsf{L}_r^{r^k}\left(\mathrm{I}_{r^{k-1}} \otimes \mathbf{DFT}_r\right)\left(\mathsf{L}_{r^{k-i-1}}^{r^k}(\mathrm{I}_{r^i} \otimes \mathsf{T}_{r^{k-i-1}}^{r^{k-i}})\,\underbrace{\mathsf{L}_{r^{i+1}}^{r^k}}_{\mathsf{vec}(c)}\right)\right)\mathsf{R}_r^{r^k}}_{\mathsf{gpu}(t,c)}$$

$$\ldots$$

$$\rightarrow \left(\prod_{i=0}^{k-1}(\mathsf{L}_r^{r^n/2} \vec{\otimes}\, \mathrm{I}_2)\left(\mathrm{I}_{r^{n-1}/2} \otimes_\times \underbrace{(\mathbf{DFT}_r \vec{\otimes}\, \mathrm{I}_2)\,\mathsf{L}_r^{2r}}_{\mathsf{shd}(t,c)}\right)\mathsf{T}_i\right)$$

$$(\mathsf{L}_r^{r^n/2} \vec{\otimes}\, \mathrm{I}_2)(\mathrm{I}_{r^{n-1}/2} \otimes_\times \underbrace{\mathsf{L}_r^{2r}}_{\mathsf{shd}(t,c)})(\mathsf{R}_r^{r^{n-1}} \vec{\otimes}\, \mathrm{I}_r)$$

Verilog for FPGAs:

$$\underbrace{\left(\mathbf{DFT}_{r^k}\right)}_{\mathsf{stream}(r^s)} \rightarrow \underbrace{\left[\prod_{i=0}^{k-1} \mathsf{L}_r^{r^k}\left(\mathrm{I}_{r^{k-1}} \otimes \mathbf{DFT}_r\right)\left(\mathsf{L}_{r^{k-i-1}}^{r^k}(\mathrm{I}_{r^i} \otimes \mathsf{T}_{r^{k-i-1}}^{r^{k-i}})\,\mathsf{L}_{r^{i+1}}^{r^k}\right)\right]\mathsf{R}_r^{r^k}}_{\mathsf{stream}(r^s)}$$

$$\ldots$$

$$\rightarrow \left[\prod_{i=0}^{k-1} \underbrace{\mathsf{L}_r^{r^k}}_{\mathsf{stream}(r^s)} \underbrace{\left(\mathrm{I}_{r^{k-1}} \otimes \mathbf{DFT}_r\right)}_{\mathsf{stream}(r^s)} \underbrace{\left(\mathsf{L}_{r^{k-i-1}}^{r^k}(\mathrm{I}_{r^i} \otimes \mathsf{T}_{r^{k-i-1}}^{r^{k-i}})\,\mathsf{L}_{r^{i+1}}^{r^k}\right)}_{\mathsf{stream}(r^s)}\right] \underbrace{\mathsf{R}_r^{r^k}}_{\mathsf{stream}(r^s)}$$

$$\ldots$$

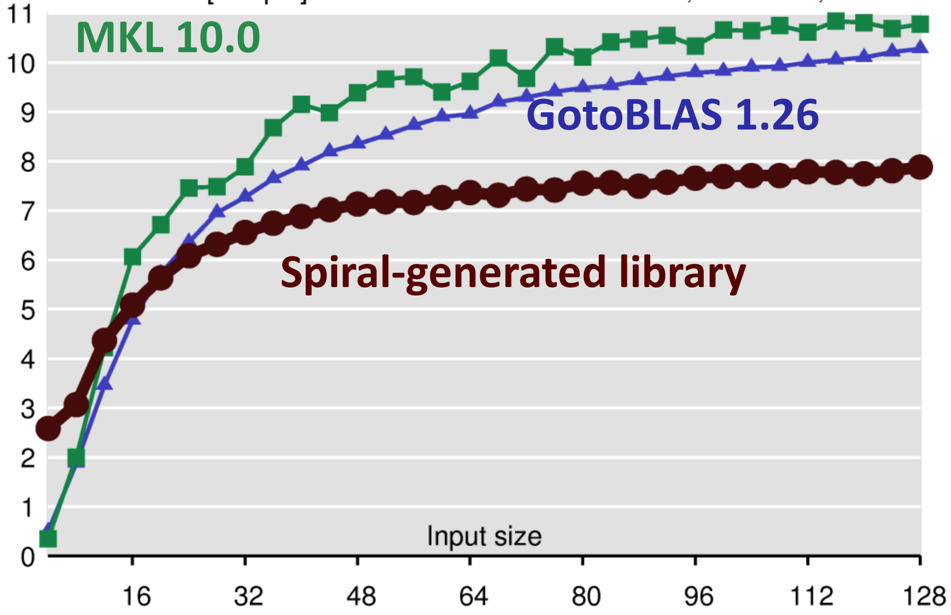
$$\rightarrow \left[\prod_{i=0}^{k-1} \underbrace{\mathsf{L}_r^{r^k}}_{\mathsf{stream}(r^s)} \left(\mathrm{I}_{r^{k-s-1}} \otimes_s (\mathrm{I}_{r^{s-1}} \otimes \mathbf{DFT}_r)\right) \underbrace{\mathsf{T}_i'}_{\mathsf{stream}(r^s)}\right] \underbrace{\mathsf{R}_r^{r^k}}_{\mathsf{stream}(r^s)}$$

Organization

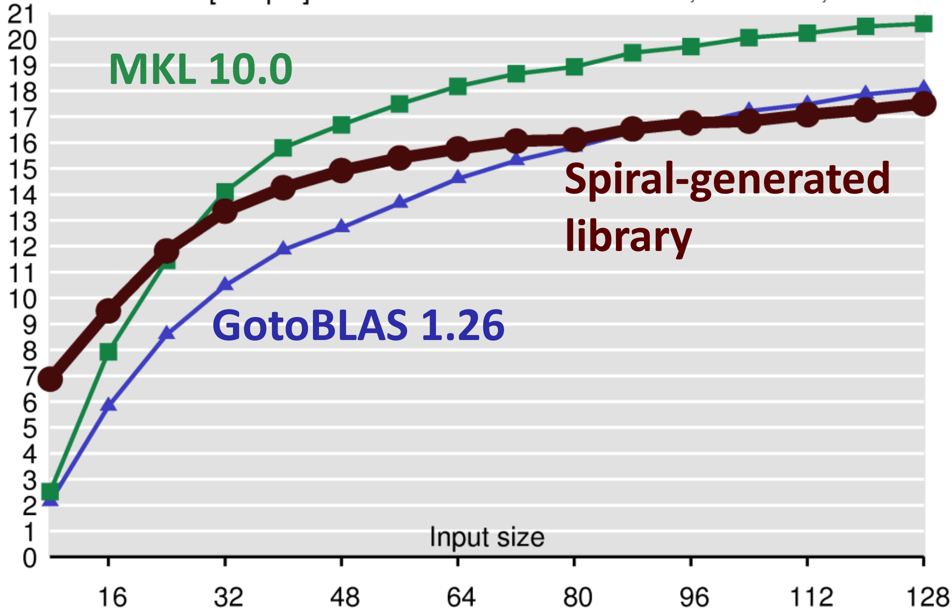
- Operator language and algorithms
- Optimizing algorithms for platforms
- **Performance results**
- Summary

Matrix Multiplication Library

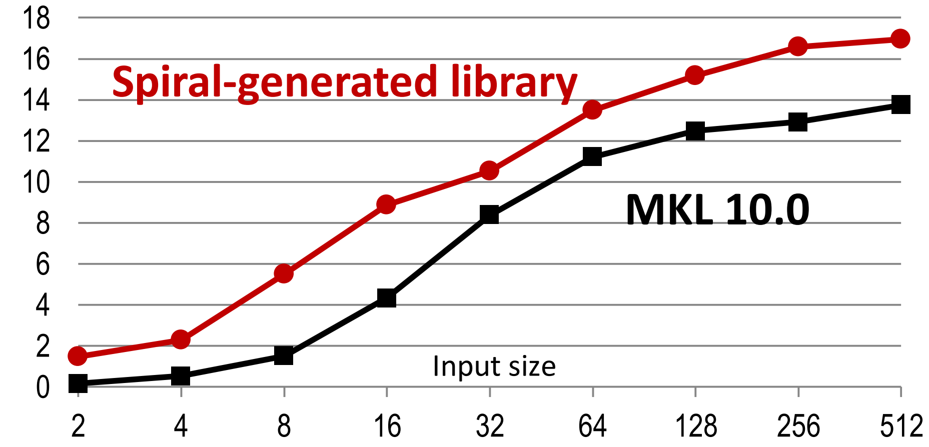
GEMM (acc. packed sq. NN), single-threaded, double precision
Performance [Gflop/s] Dual Intel Xeon 5160, 3000 MHz, icc 10.1



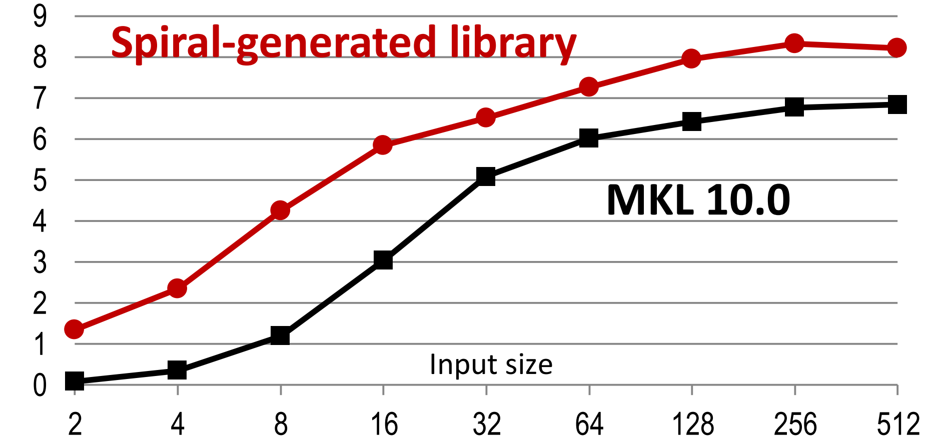
GEMM (acc. packed sq. NN), single-threaded, single precision
Performance [Gflop/s] Dual Intel Xeon 5160, 3000 MHz, icc 10.1



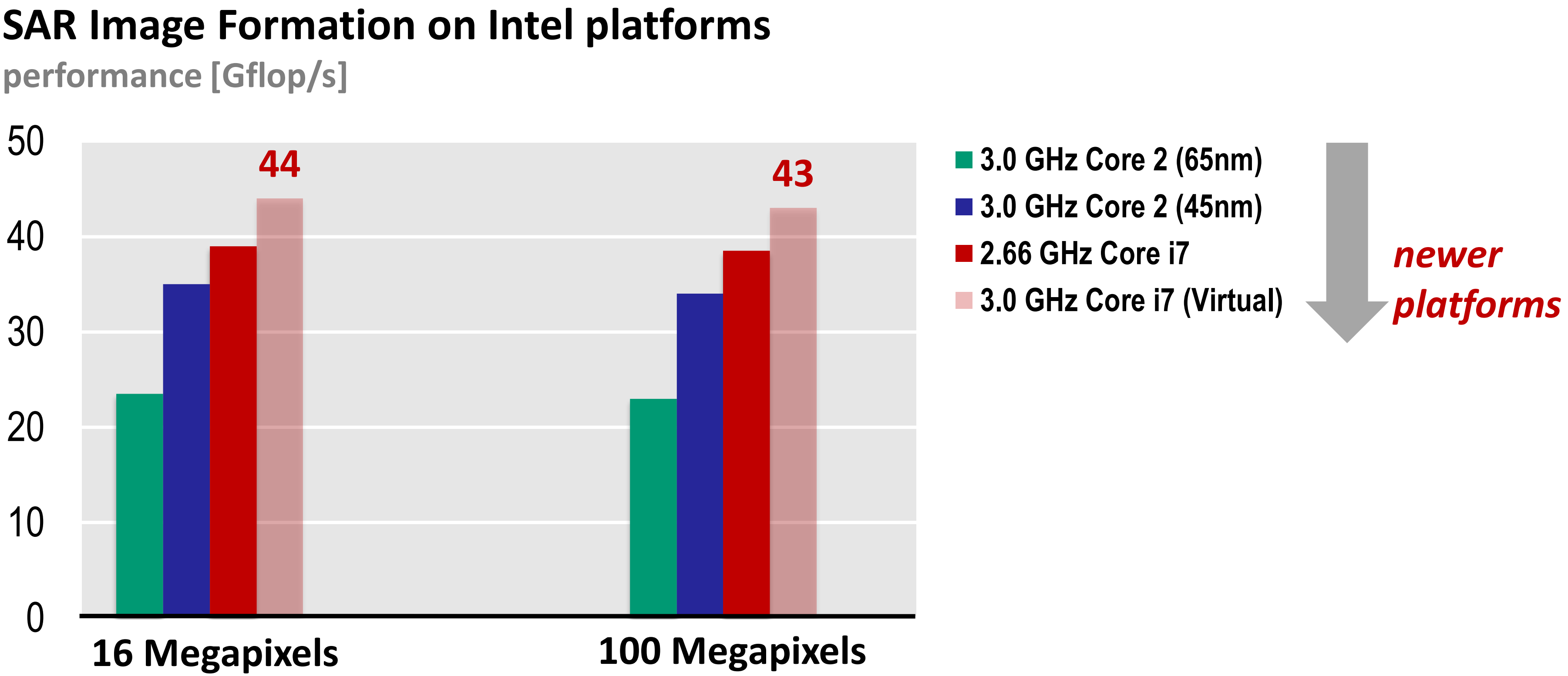
Rank-k Update, single precision, k=4
performance [Gflop/s] Dual Intel Xeon 5160, 3Ghz



Rank-k Update, double precision, k=4
performance [Gflop/s] Dual Intel Xeon 5160, 3Ghz



Result: Spiral-Generated PFA SAR on Core2 Quad



- Algorithm by J. Rudin (best paper award, HPEC 2007): 30 Gflop/s on Cell
- Each implementation: vectorized, threaded, cache tuned, ~13 MB of code

Organization

- Operator language and algorithms
- Optimizing algorithms for platforms
- Performance results
- **Summary**

Summary

- **Platforms are powerful yet complicated**
  optimization will stay a hard problem

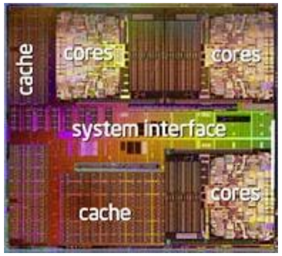
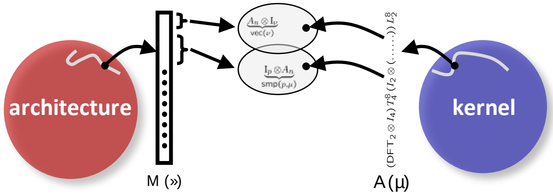


Image: Intel

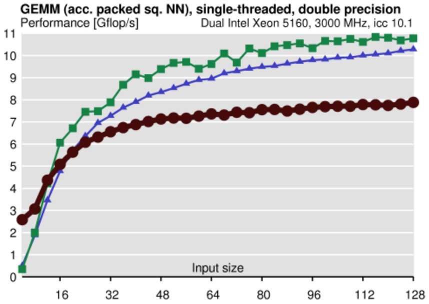
- **OL: unified mathematical framework**
  captures platforms and algorithms

$$\underbrace{\mathrm{I}_p \otimes A_n}_{\mathrm{smp}(p,\mu)}$$

- **Spiral: program generation and autotuning**
  can provide full automation


architecture
M (»)
$\underbrace{A_n \otimes \mathrm{I}_\nu}_{\mathrm{vec}(\nu)}$
$\underbrace{\mathrm{I}_p \otimes A_n}_{\mathrm{smp}(p,\mu)}$
$(\mathrm{DFT}_2 \otimes I_4)\, T^8_4\, (I_2 \otimes (\ldots\ldots))\, L^8_2$
A (μ)
kernel

- **Performance of supported kernels**
  is competitive with expert tuning


GEMM (acc. packed sq. NN), single-threaded, double precision
Performance [Gflop/s]
Dual Intel Xeon 5160, 3000 MHz, icc 10.1
Input size