TORSTEN HOEFLER

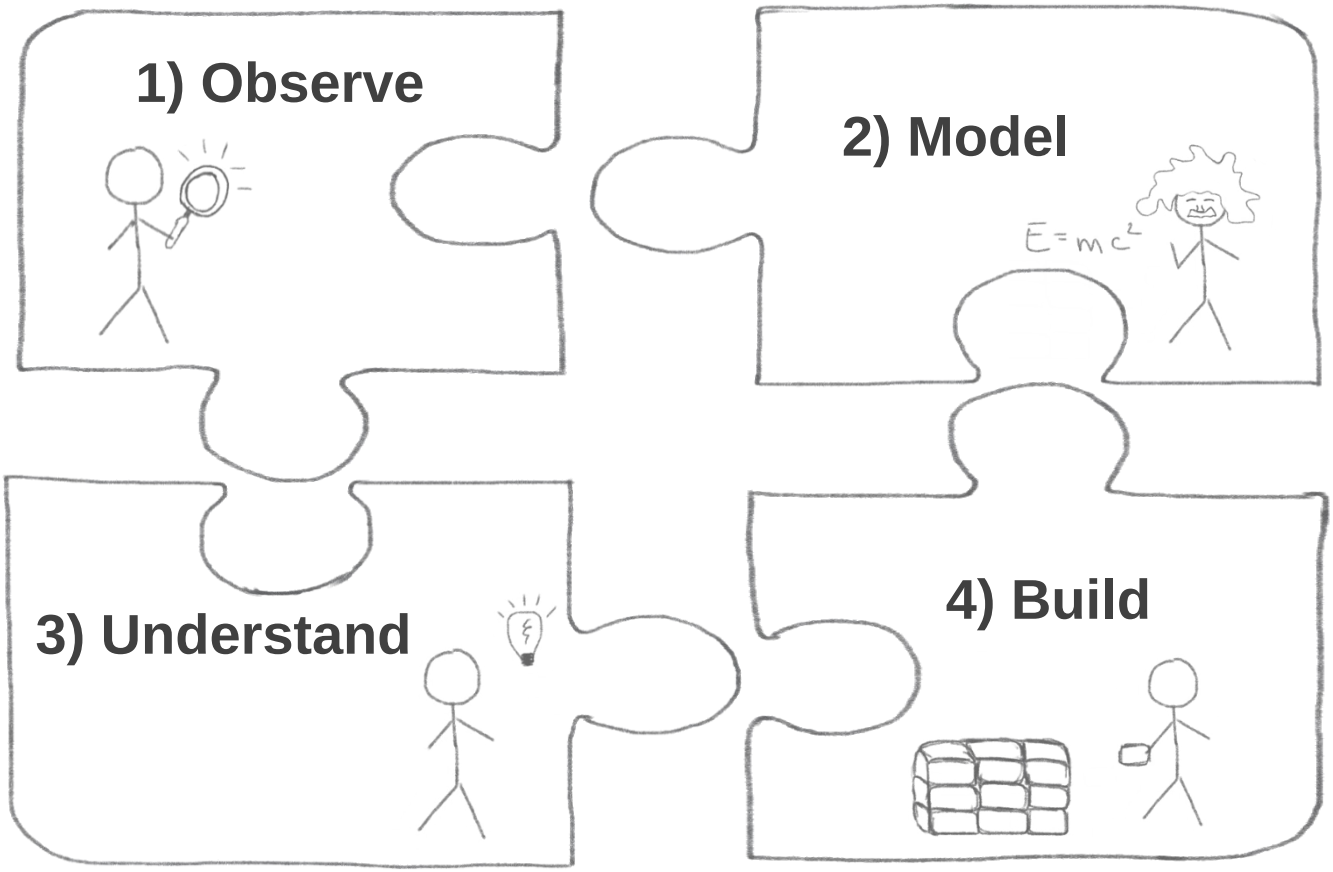
Accelerating weather and climate simulations on heterogeneous architectures

with support of Tobias Gysi, Tobias Grosser, Jeremiah Baer @ SPCL
presented at Co-Design, HPC China, Nov. 2016

PASC17
Platform for Advanced Scientific Computing Conference
Lugano Switzerland
26-28 June 2017

CLIMATE & WEATHER
SOLID EARTH
LIFE SCIENCE
CHEMISTRY & MATERIALS
PHYSICS
COMPUTER SCIENCE & MATHEMATICS
ENGINEERING
EMERGING DOMAINS

Scientific Performance Engineering



More at keynote at HPC China tomorrow morning 8:30am!

Stencil computations (oh

due to their low arithmetic intensity stencil computations are typically heavily memory bandwidth limited!

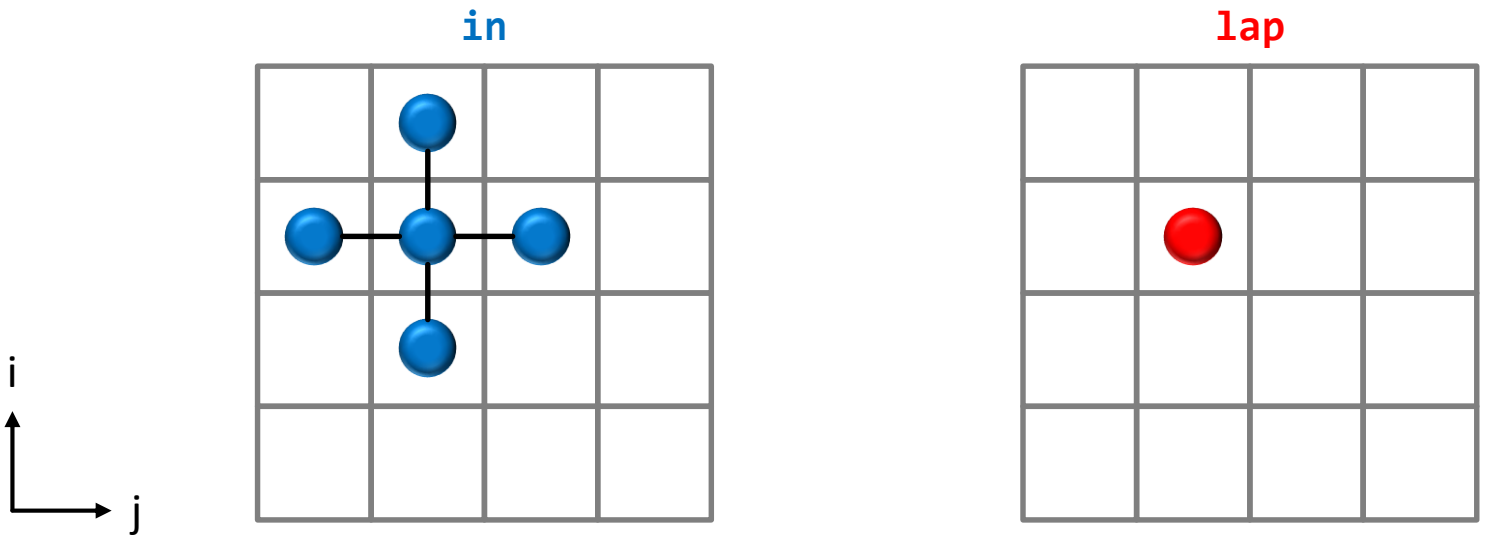
Motivation:

- **Important algorithmic motif (e.g., finite difference method)**

Definition:

- **Element-wise computation on a regular grid using a fixed neighborhood**
- **Typically working on multiple input fields and writing a single output field**

```
lap(i,j) = -4.0 * in(i,j) + in(i-1,j) + in(i+1,j) + in(i,j-1) + in(i,j+1)
```

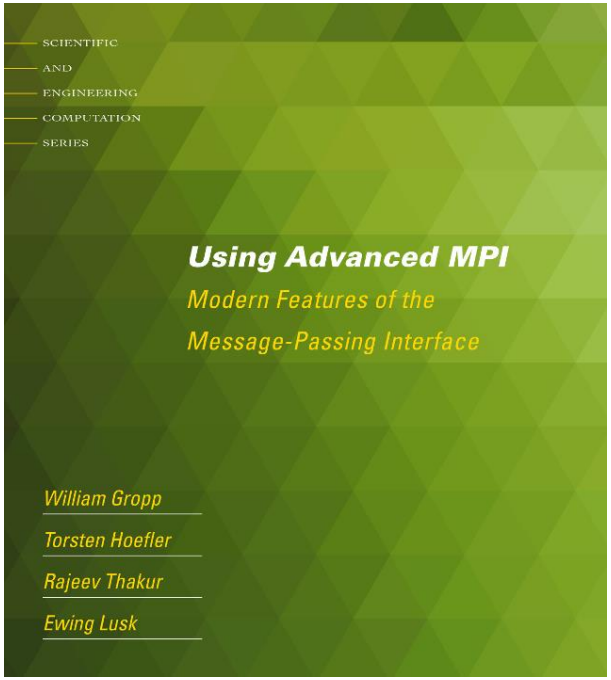


How to tune such stencils (most other stencil talks)

- **LOTS of related work!**
  - Compiler-based (e.g., Polyhedral such as PLUTO [1])
  - Auto-tuning (e.g., PATUS [2])
  - Manual model-based tuning (e.g., Datta et al. [3])
  - … essentially every micro-benchmark or tutorial, e.g.:

- **Common features**
  - Vectorization tricks (data layout)
  - Advanced communication (e.g., MPI neighbor colls)
  - Tiling in time, space (diamond etc.)
  - Pipelining

- **Much of that work DOES NOT compose well with practical complex <u>stencil programs</u>**



[1]: Uday Bondhugula, A. Hartono, J. Ramanujan, P. Sadayappan. *A Practical Automatic Polyhedral Parallelizer and Locality Optimizer* , PLDI'08
[2]: Matthias Christen, et al.: *PATUS: A Code Generation and Autotuning Framework for Parallel Iterative Stencil Computations …, IPDPS'11*
[3]: Kaushik Datta, et al., *Optimization and Performance Modeling of Stencil Computations on Modern Microprocessors,* SIAM review

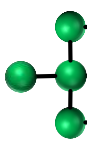
What is a “complex stencil program”? (this stencil talk)

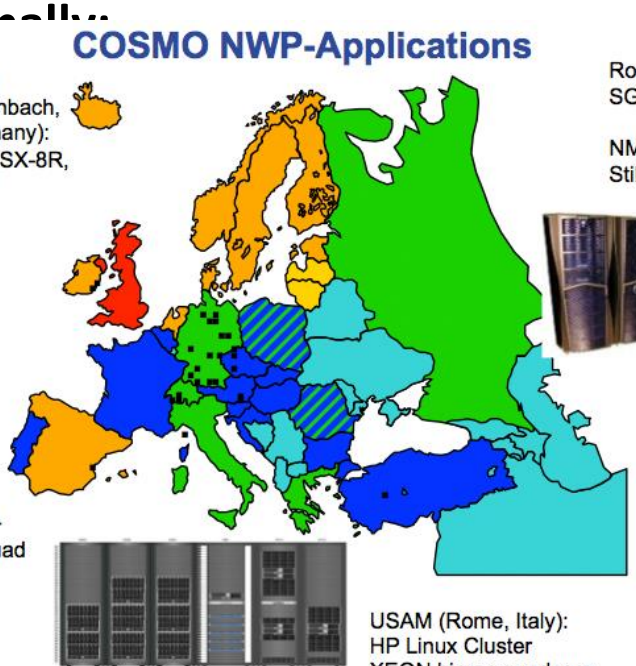
E.g., the COSMO weather code

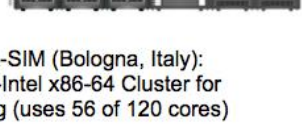
- **is a regional climate model used by 7 national weather services**
- **contains hundreds of different complex stencils**

Modelin~~g stencils formally:

- **Repr**
  - M~~o~~







DWD (Offenbach, Germany): NEC SX-8R, SX-9

Roshydromet (Moscow, Russia), SGI

NMA (Bucharest, Romania): Still in planning / procurement phase

IMGW (Warsawa, Poland): SGI Origin 3800: uses 88 of 100 nodes

MeteoSwiss: Cray XT4: COSMO-7 and COSMO-2 use 980+4 MPI-Tasks on 246 out of 260 quad core AMD nodes

USAM (Rome, Italy): HP Linux Cluster XEON biproc quadcore System in preparation

ARPA-SIM (Bologna, Italy): Linux-Intel x86-64 Cluster for testing (uses 56 of 120 cores)

ARPA-SIM (Bologna, Italy): IBM pwr5: up to 160 of 512 nodes at CINECA

COSMO-LEPS (at ECMWF): running on ECMWF pwr6 as member-state time-critical application

HNMS (Athens, Greece): IBM pwr4: 120 of 256 nodes

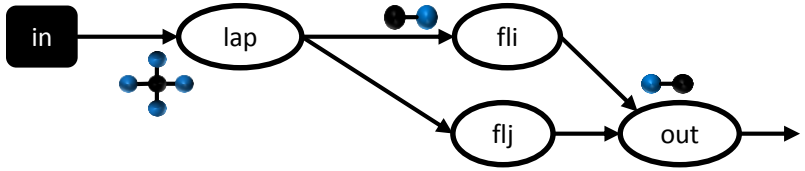
in

out

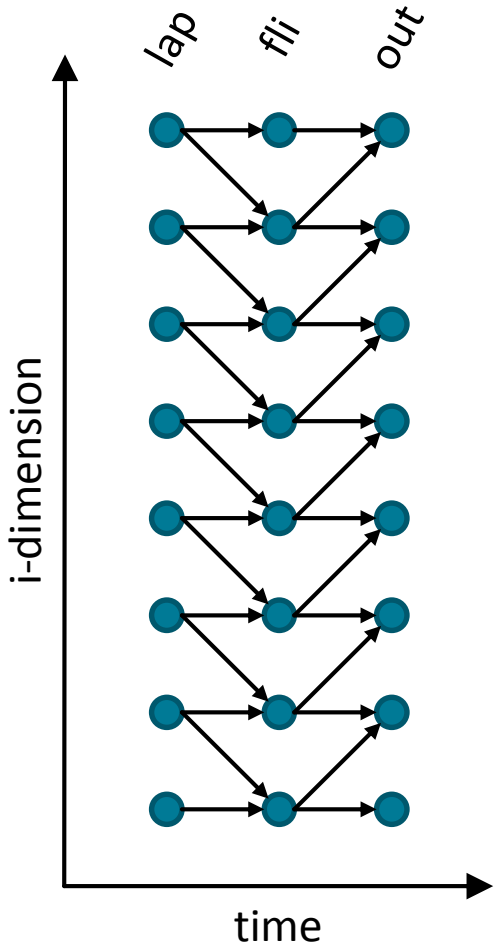
$$a \oplus b = \{a' + b' | a' \in a, b' \in b\}$$

T. Gysi, T. Grosser, TH: MODESTO: Data-centric Analytic Optimization of Complex Stencil Programs on Heterogeneous Architectures, ACM ICS’15

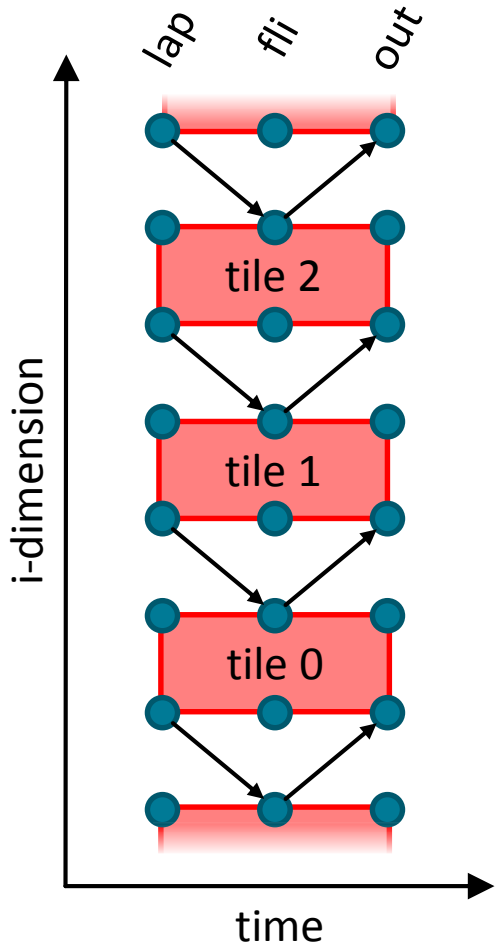
Data-locality Transformations



- **Consider the horizontal diffusion lap-fli-out dependency chain (i-dimension)**

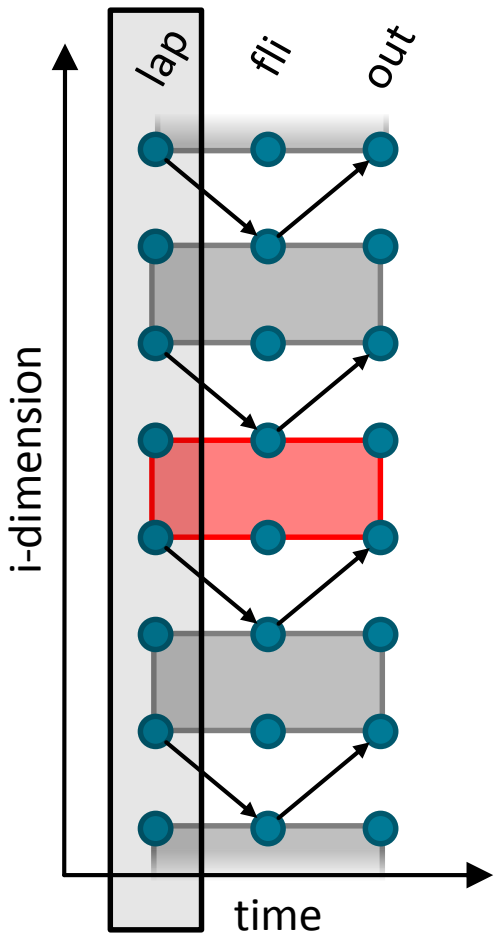


Loop Tiling & Loop Fusion



How to Deal with Data Dependencies?

- **Consider the horizontal diffusion lap-fli-out dependency chain (i-dimension)**



Halo Exchange Parallel (hp):

- Update tiles in parallel
- Perform halo exchange communication

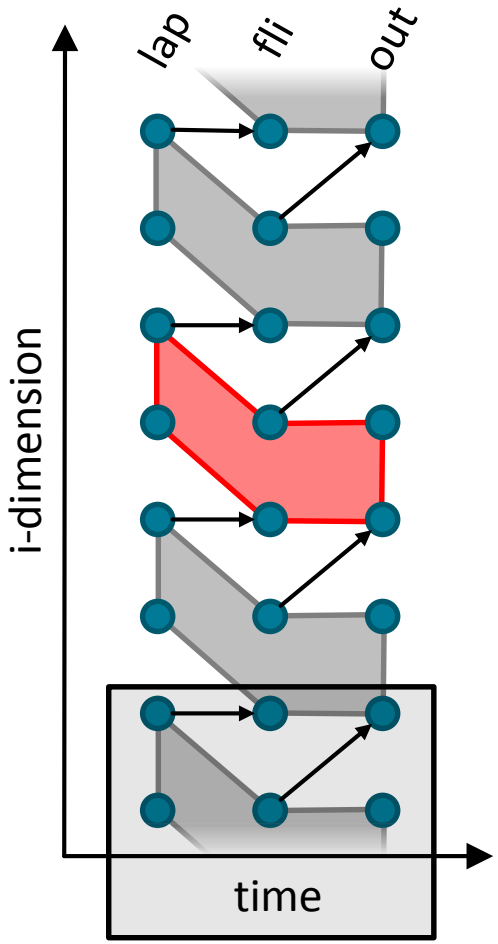
Pros and Cons:

- Avoid redundant computation
- At the cost of additional synchronization

T. Gysi, T. Grosser, TH: MODESTO: Data-centric Analytic Optimization of Complex Stencil Programs on Heterogeneous Architectures, ACM ICS'15

How to Deal with Data Dependencies?

- **Consider the horizontal diffusion lap-fli-out dependency chain (i-dimension)**



Halo Exchange Sequential (hs):

- Update tiles sequentially
- Innermost loop updates tile-by-tile

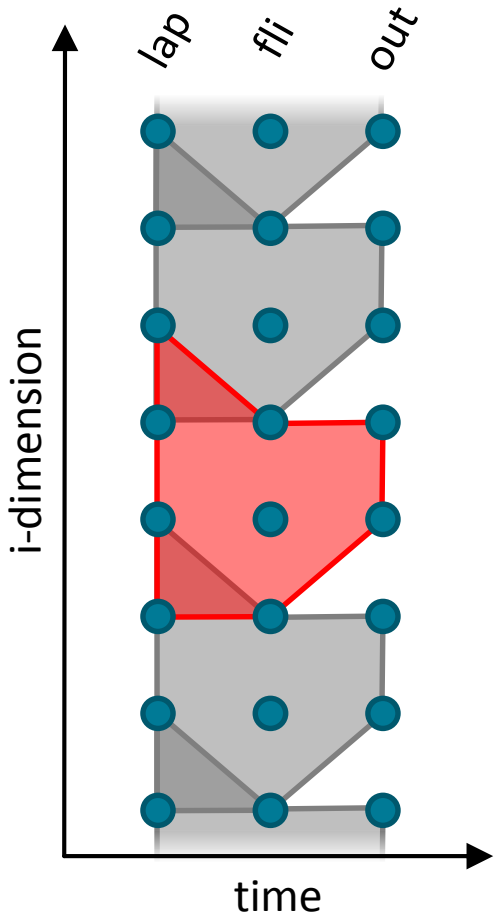
Pros and Cons:

- Avoid redundant computation
- At cost of being sequential

T. Gysi, T. Grosser, TH: MODESTO: Data-centric Analytic Optimization of Complex Stencil Programs on Heterogeneous Architectures, ACM ICS'15

How to Deal with Data Dependencies?

- **Consider the horizontal diffusion lap-fli-out dependency chain (i-dimension)**



Computation on-the-fly (of):

- Compute all dependencies on-the-fly
- Overlapped tiling

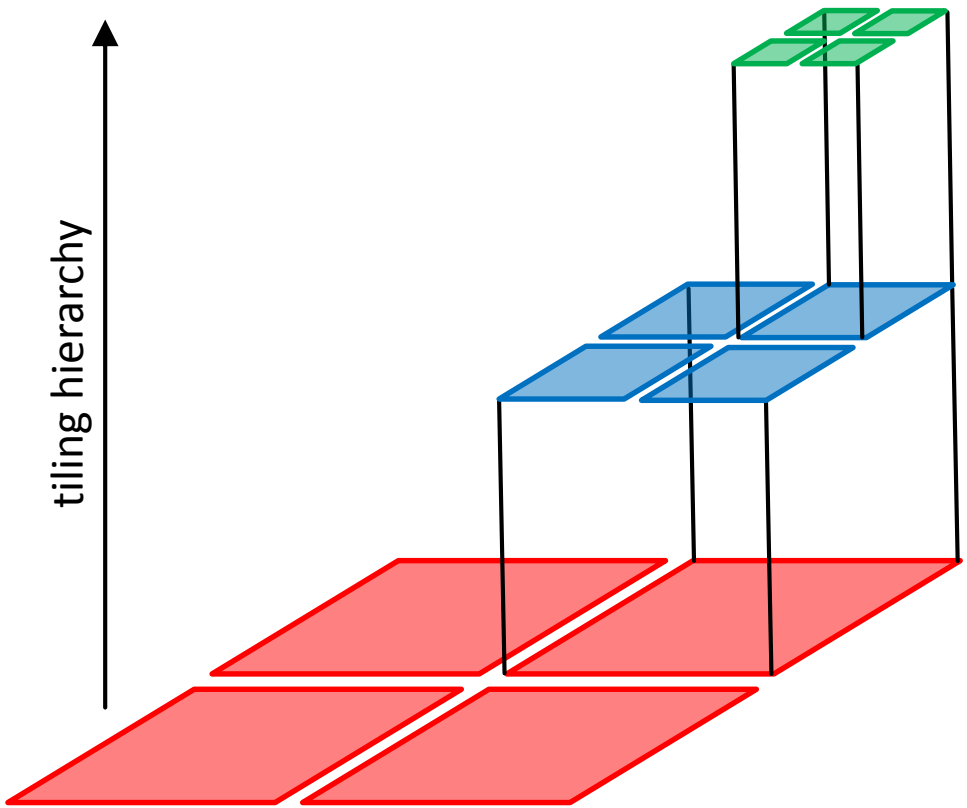
Pros and Cons:

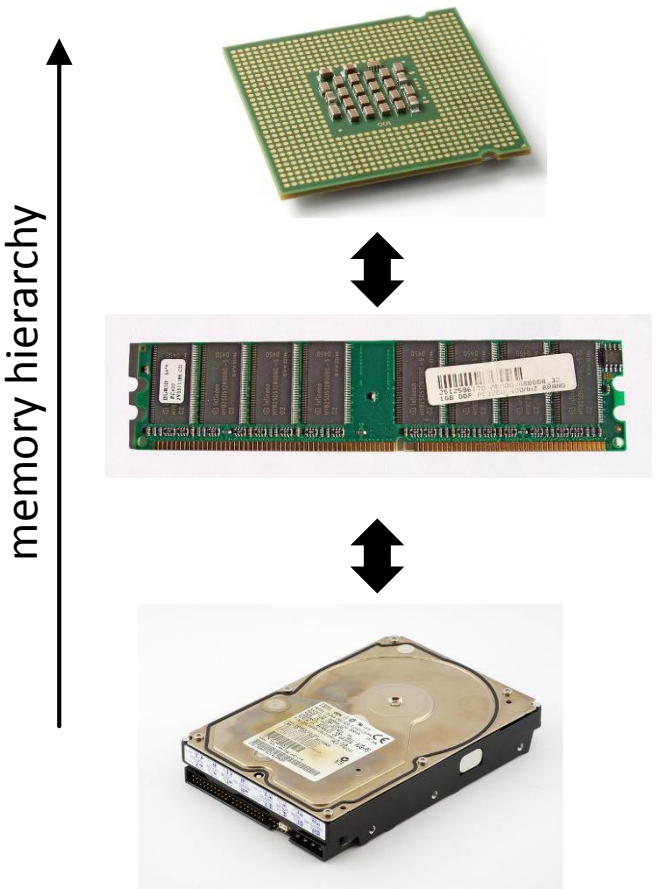
- Avoid synchronization
- At the cost of redundant computation

T. Gysi, T. Grosser, TH: MODESTO: Data-centric Analytic Optimization of Complex Stencil Programs on Heterogeneous Architectures, ACM ICS'15

Hierarchical Tiling

- By tiling the domain repeatedly we target multiple memory hierarchy levels





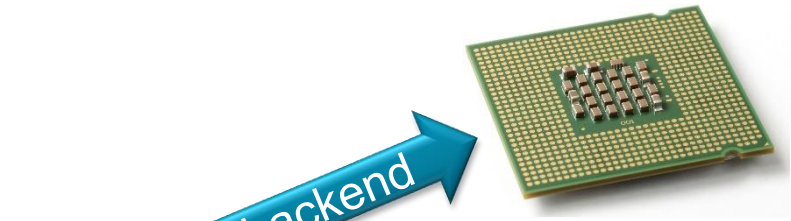
T. Gysi, T. Grosser, TH: MODESTO: Data-centric Analytic Optimization of Complex Stencil Programs on Heterogeneous Architectures, ACM ICS'15

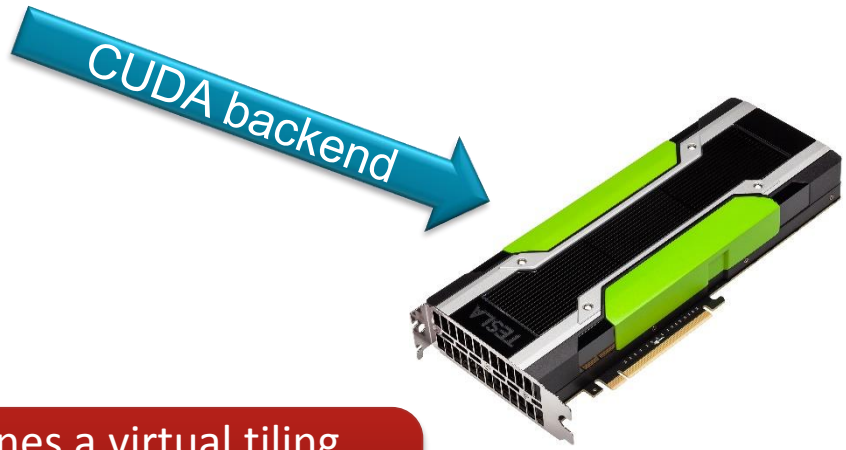
Case Study: STELLA (STEncil Loop LAnguage)

- **STELLA is a C++ stencil DS(e)L of COSMO's dynamical core (50k LOC, 60% RT)**

```
// define stencil functors
struct Lap { ... };
struct Fli { ... };
...
// stencil assembly
Stencil stencil;
StencilCompiler::Build(
 stencil,
 pack_parameters( ... ),
 define_temporaries(
  StencilBuffer<lap, double>(),
  StencilBuffer<fli, double>(),
  ...
 ),
 define_loops(
  define_sweep(
   StencilStage<Lap, IJRange<-1,1,-1,1> >(),
   StencilStage<Fli, IJRange<-1,0,0,0> >(),
   ...
)));
// stencil execution
stencil.Apply();
```

using C++ template metaprogramming:



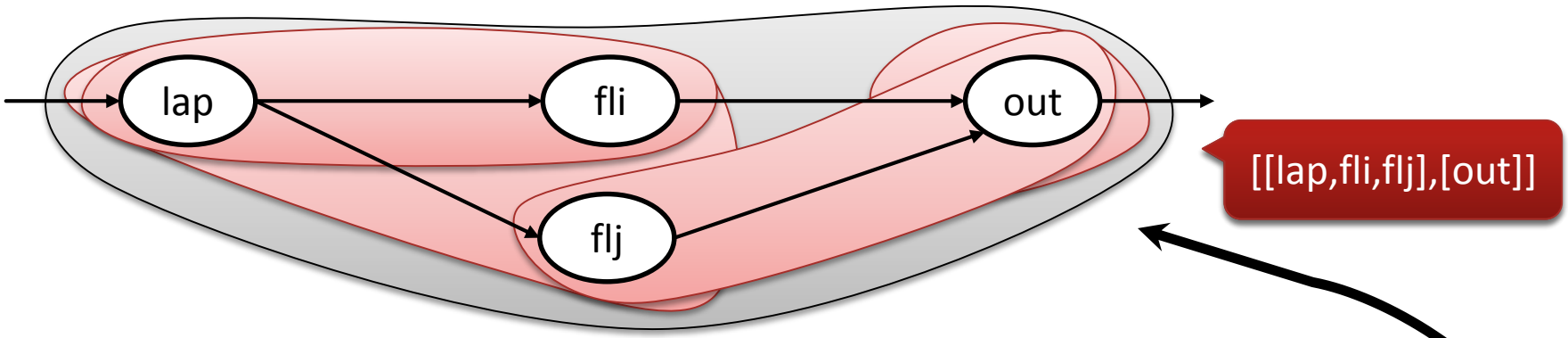


STELLA defines a virtual tiling hierarchy that facilitates platform independent code generation

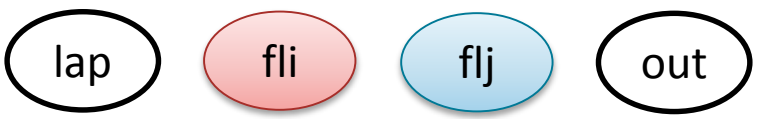
T. Gysi, T. Grosser, TH: MODESTO: Data-centric Analytic Optimization of Complex Stencil Programs on Heterogeneous Architectures, ACM ICS'15

Stencil Program Algebra

- **Map stencils to the tiling hierarchy using a bracket expression**



- **Enumerate the stencil execution orders that respect the dependencies**



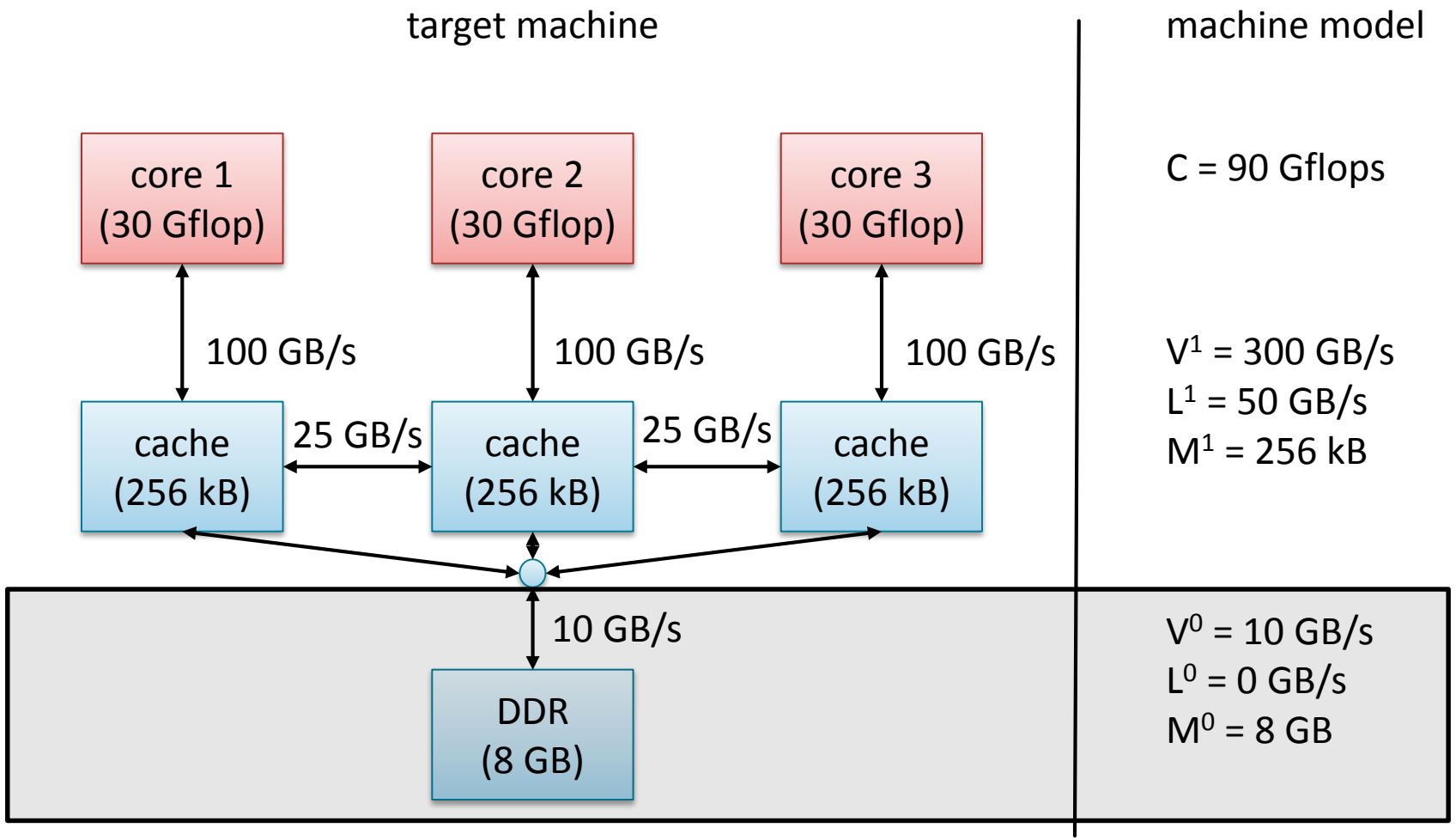
- **Enumerate implementation variants by adding/removing brackets**

...,lap,fli **]],[[** flj,out,...

Machine Performance Mo

lateral and vertical communication refer to communication within one respectively between different tiling hierarchy levels

- **Our model considers peak computation and communication throughputs**



T. Gysi, T. Grosser, TH: MODESTO: Data-centric Analytic Optimization of Complex Stencil Programs on Heterogeneous Architectures, ACM ICS'15

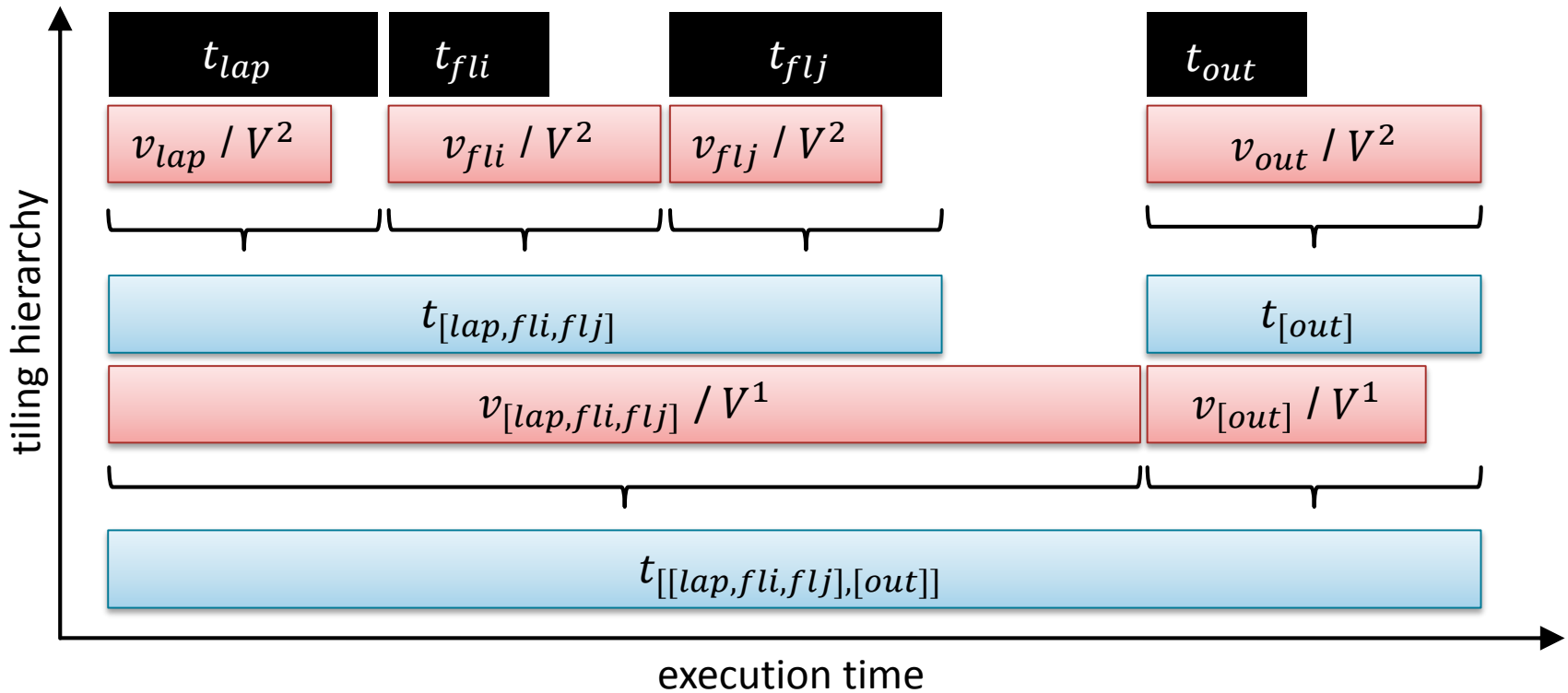
Stencil Performance Model - Overview

- **Given a stencil s given and the amount of computation c_s**

$$t_s = c_s / C$$

- **Given a group g and the vertical and lateral communication v_c and $l_c^1, \dots, l_c^m$**

$$t_g = \sum_{c \in g.child} \max(t_c, v_c/V^m, l_c^1/L^1, \dots, l_c^m/L^m)$$



T. Gysi, T. Grosser, TH: MODESTO: Data-centric Analytic Optimization of Complex Stencil Programs on Heterogeneous Architectures, ACM ICS'15

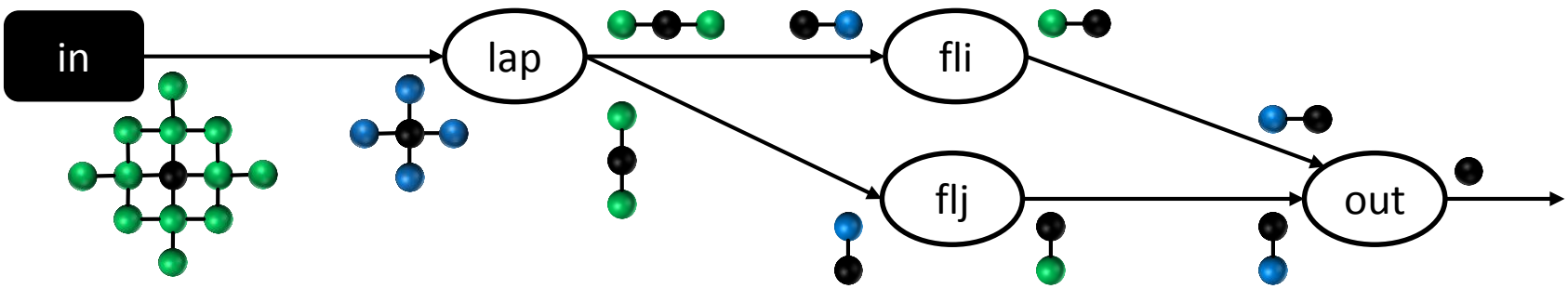
Stencil Performance Model - Affine Sets and Maps

- **The stencil program analysis is based on (quasi-) affine sets and maps**

$$S = \{\vec{\imath} \mid \vec{\imath} \in \mathbb{Z}^n \wedge (0, \dots, 0) < \vec{\imath} < (10, \dots, 10)\}$$
$$M = \{\vec{\imath} \to \vec{\jmath} \mid \vec{\imath} \in \mathbb{Z}^n, \vec{\jmath} \in \mathbb{Z}^n \wedge \vec{\jmath} = 2 \cdot \vec{\imath}\}$$

- **For example, data dependencies can be expressed using named maps**

$$D_{fli} = \{(fli, \vec{\imath}) \to (lap, \vec{\imath} + \vec{\jmath}) \mid \vec{\imath} \in \mathbb{Z}^2, \vec{\jmath} \in \{(0,0), (1,0)\}\}$$



$$D = D_{lap} \cup D_{fli} \cup D_{flj} \cup D_{out}$$

$$E = \mathrm{D}^{+}(\{\,(out, \vec{0})\,\})$$

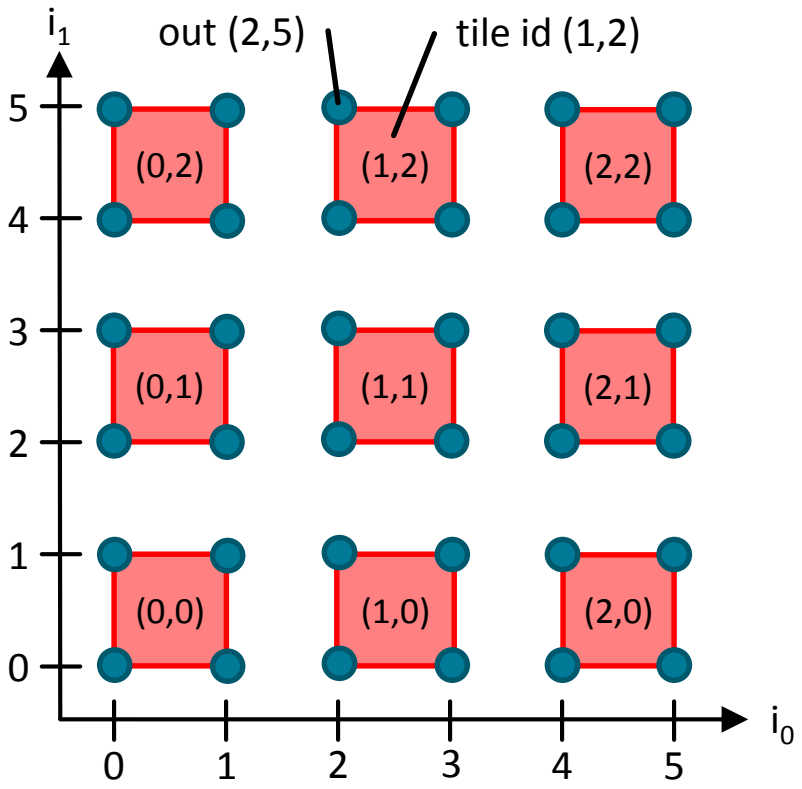
apply the out origin vector to the transitive closure of all dependencies

T. Gysi, T. Grosser, TH: MODESTO: Data-centric Analytic Optimization of Complex Stencil Programs on Heterogeneous Architectures, ACM ICS'15

Stencil Performance Model - Tiling Transformations

- **Define a tiling using a map that associates stencil evaluations to tile ids**

$$T_{out} = \{(\text{out}, (i_0, i_1)) \rightarrow (\lfloor i_0/2 \rfloor, \lfloor i_1/2 \rfloor)\}$$



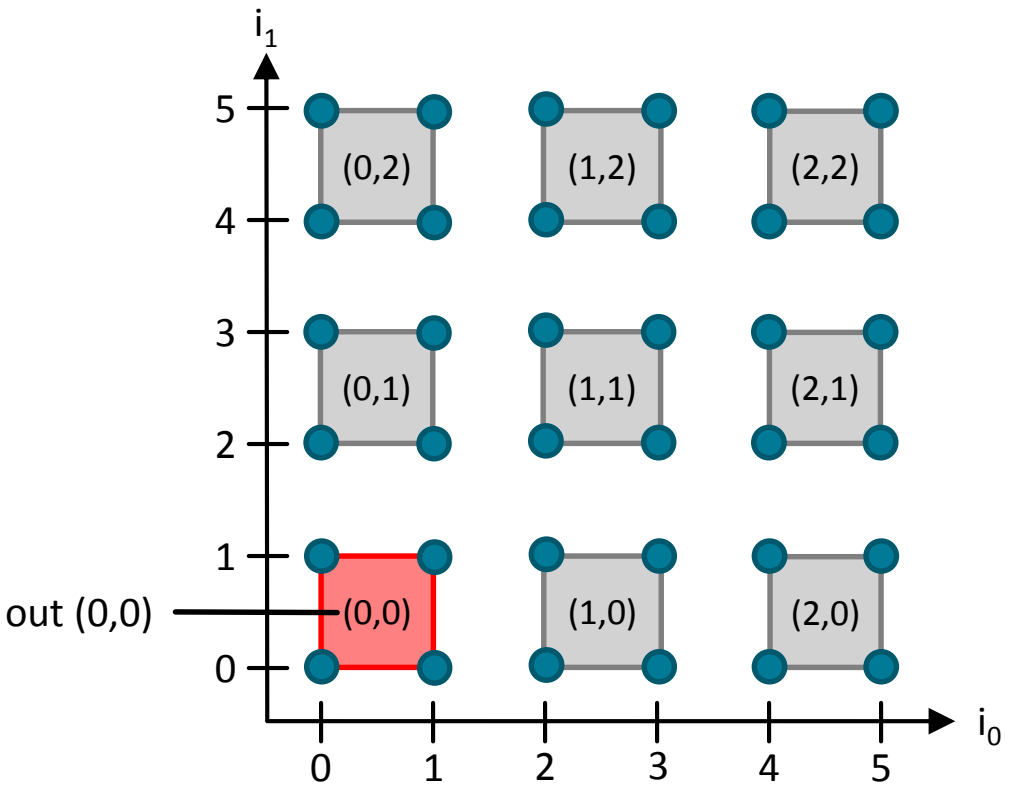
Stencil Performance Model – Comp & Comm

- **Count floating point operations necessary to update tile (0,0)**

$$c_{out} = |T_{out} \cap_{ran} \{(0,0)\}| \cdot \#flops$$

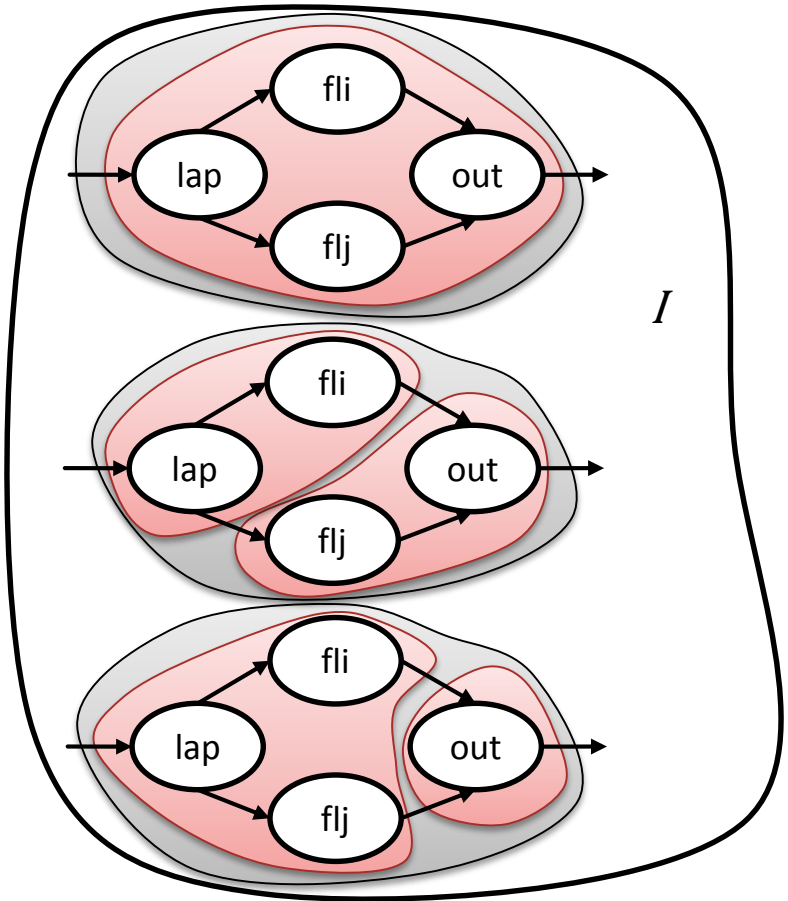
- **Count the number of loads necessary to update tile (0,0)**

$$l_{out} = |(T_{out} \circ D_{out}^{-1}) \cap_{ran} \{(0,0)\}|$$



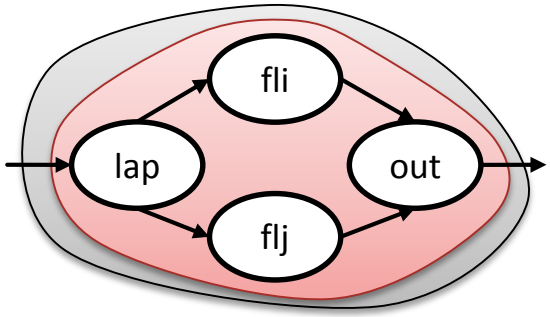
Analytic Stencil Program Optimization

- **Put it all together (stencil algebra, performance model, stencil analysis)**
  1. Optimize the stencil execution order (brute force search)
  2. Optimize the stencil grouping (dynamic programming / brute force search)



$$\underset{x \in I}{\text{minimize}}\, t(x)$$

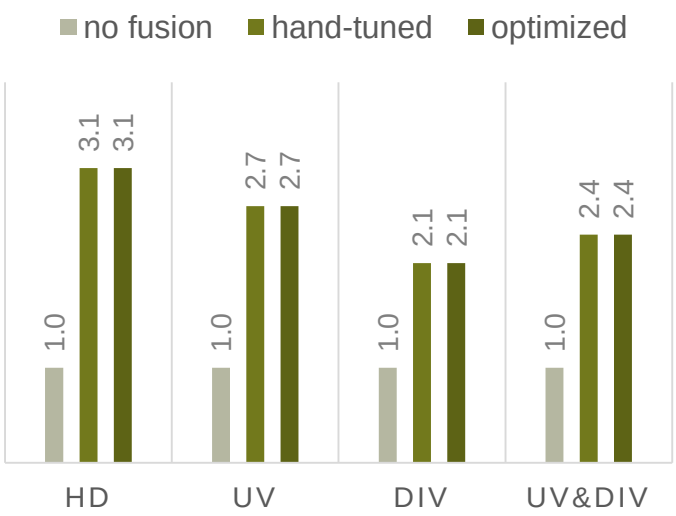
$$\text{subject to } m(x) \leq M$$

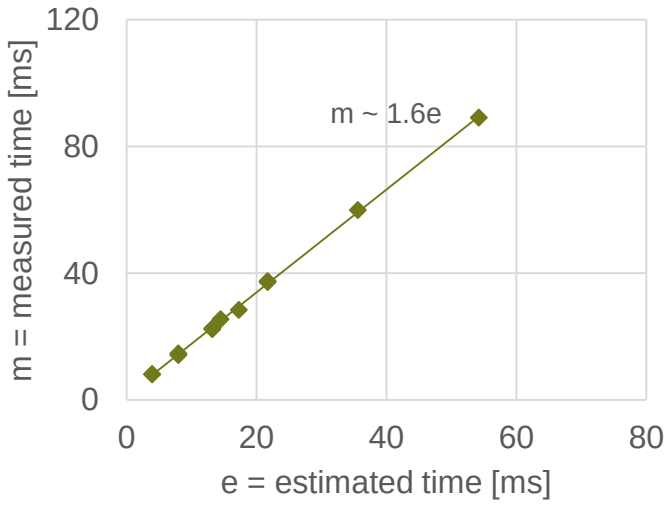


T. Gysi, T. Grosser, TH: MODESTO: Data-centric Analytic Optimization of Complex Stencil Programs on Heterogeneous Architectures, ACM ICS'15

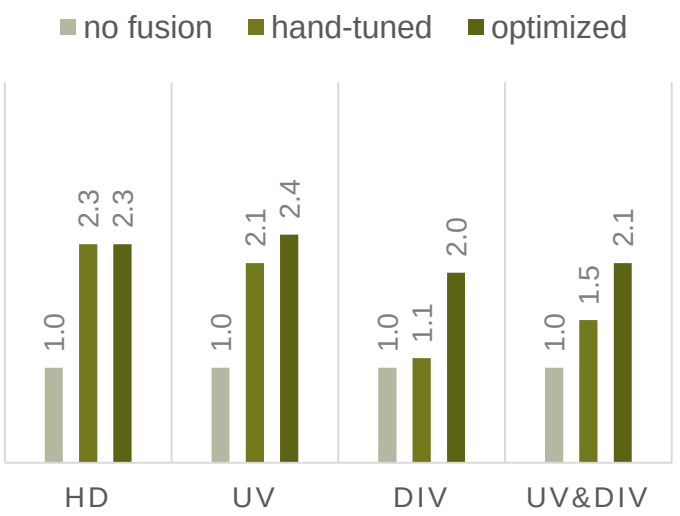
Evaluation – single CPU/GPU

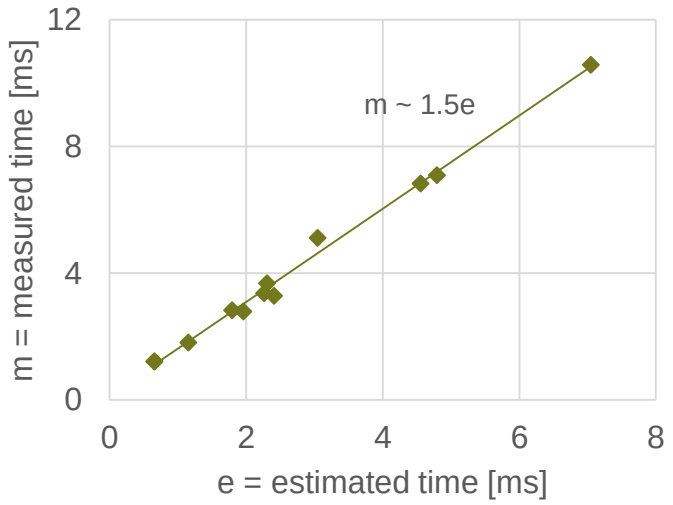
CPU Experiments (i5-3330):





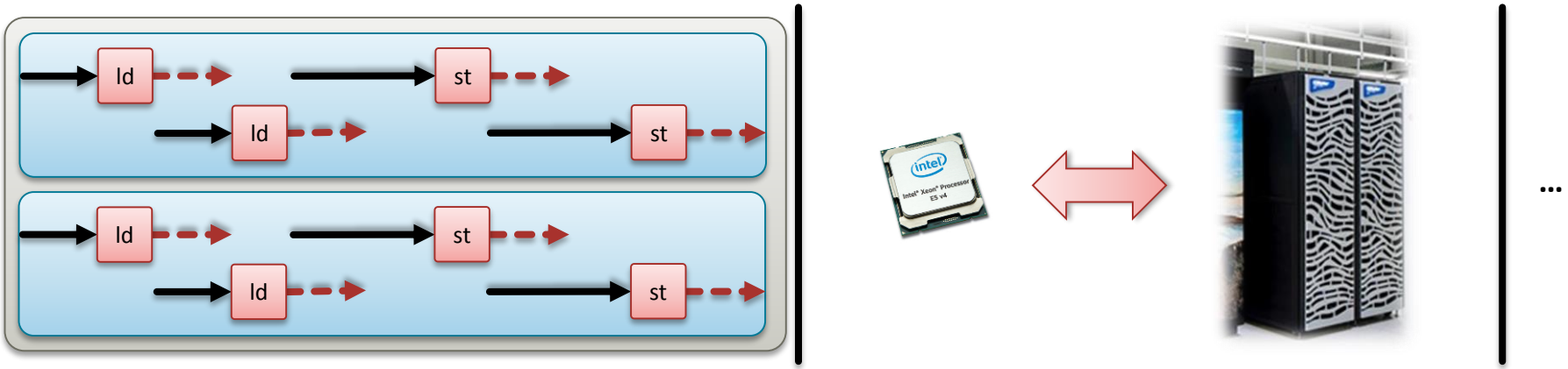
GPU Experiments (Tesla K20c):





T. Gysi, T. Grosser, TH: MODESTO: Data-centric Analytic Optimization of Complex Stencil Programs on Heterogeneous Architectures, ACM ICS'15

From GPUs to the cluster!

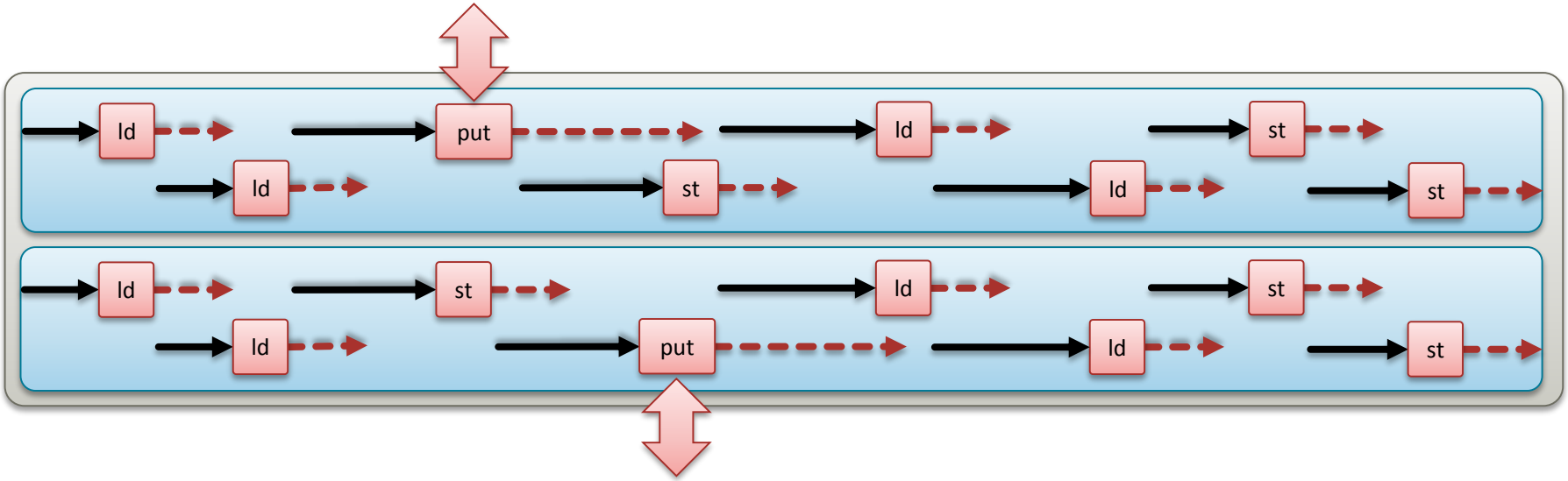


CUDA
- over-subscribe hardware
- use spare parallel slack for latency hiding

MPI
- host controlled
- full device synchronization

T. Gysi, J. Baer, TH: dCUDA: Hardware Supported Overlap of Computation and Communication, ACM/IEEE SC16 (preprint at SPCL page)

Latency hiding at the cluster level?



dCUDA (distributed CUDA)

- unified programming model for GPU clusters
- avoid unnecessary device synchronization to enable system wide latency hiding

T. Gysi, J. Baer, TH: dCUDA: Hardware Supported Overlap of Computation and Communication, ACM/IEEE SC16 (preprint at SPCL page)

dCUDA extends CUDA with MPI-3 RMA and notifications

```
for (int i = 0; i < steps; ++i) {
  for (int idx = from; idx < to; idx += jstride)
    out[idx] = -4.0 * in[idx] +
      in[idx + 1] + in[idx - 1] +
      in[idx + jstride] + in[idx - jstride];

  if (lsend)
    dcuda_put_notify(ctx, wout, rank - 1,
      len + jstride, jstride, &out[jstride], tag);
  if (rsend)
    dcuda_put_notify(ctx, wout, rank + 1,
      0, jstride, &out[len], tag);

  dcuda_wait_notifications(ctx, wout,
    DCUDA_ANY_SOURCE, tag, lsend + rsend);

  swap(in, out); swap(win, wout);
}
```

computation

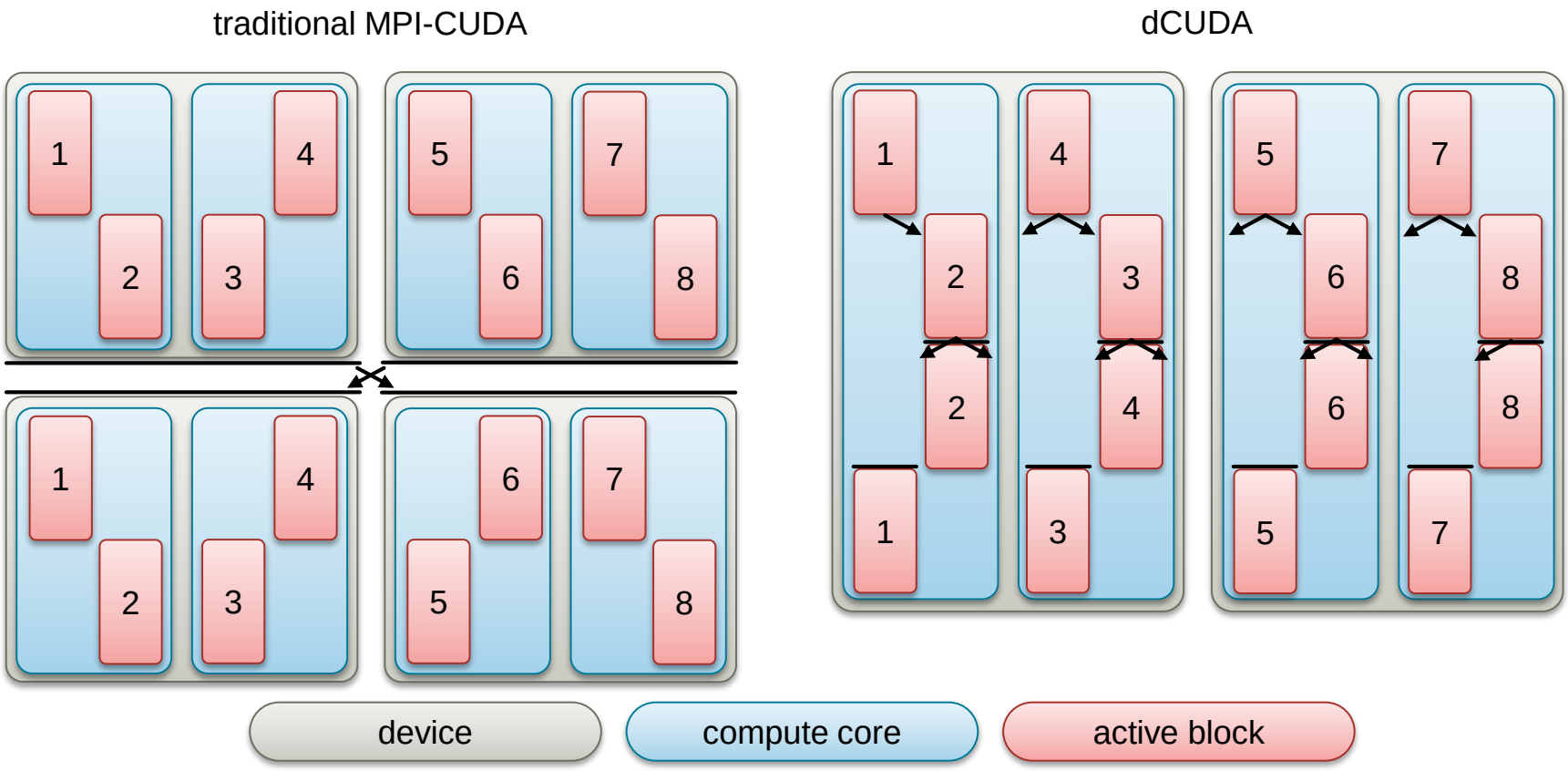
communication

- iterative stencil kernel
- thread specific idx

- map ranks to blocks
- device-side put/get operations
- notifications for synchronization
- shared and distributed memory

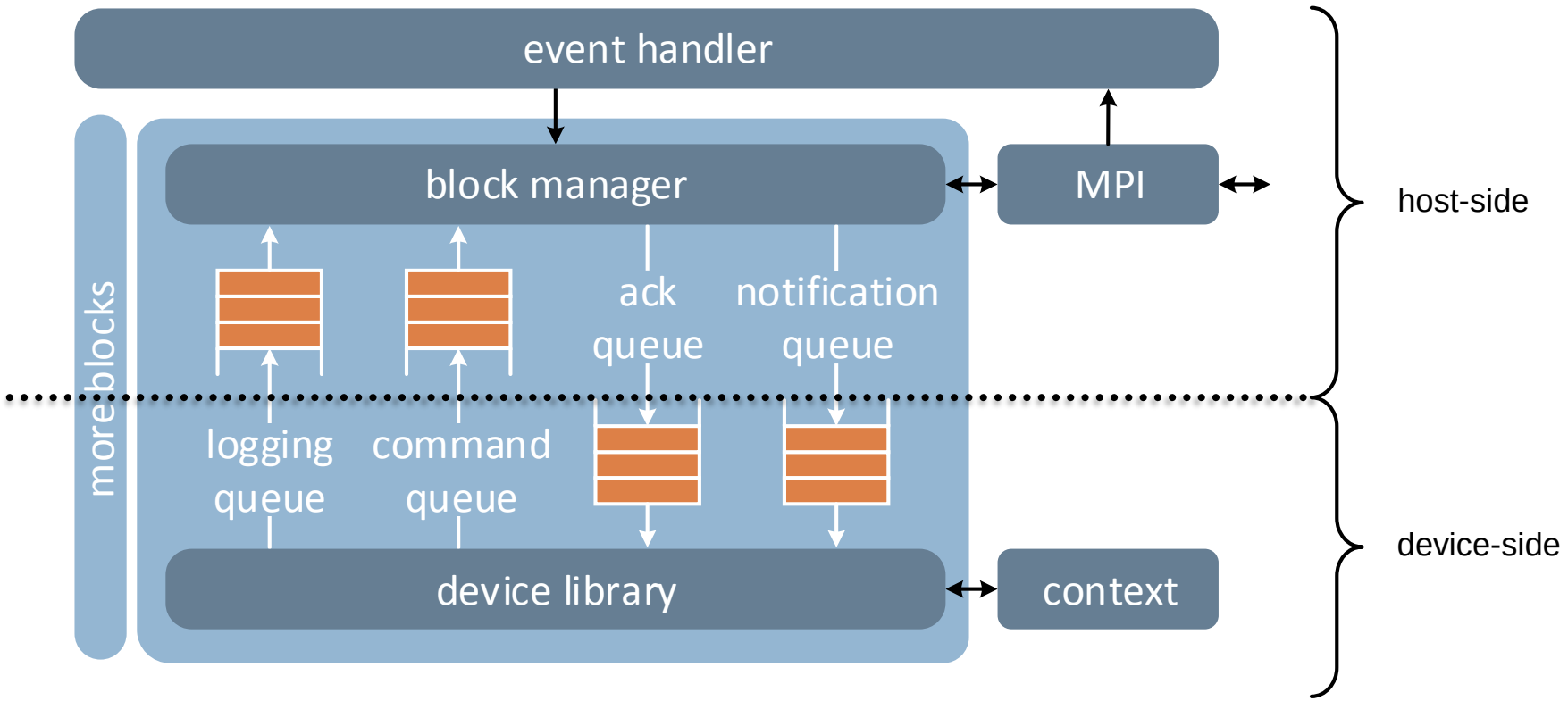
T. Gysi, J. Baer, TH: dCUDA: Hardware Supported Overlap of Computation and Communication, ACM/IEEE SC16 (preprint at SPCL page)

Hardware supported communication overlap



T. Gysi, J. Baer, TH: dCUDA: Hardware Supported Overlap of Computation and Communication, ACM/IEEE SC16 (preprint at SPCL page)

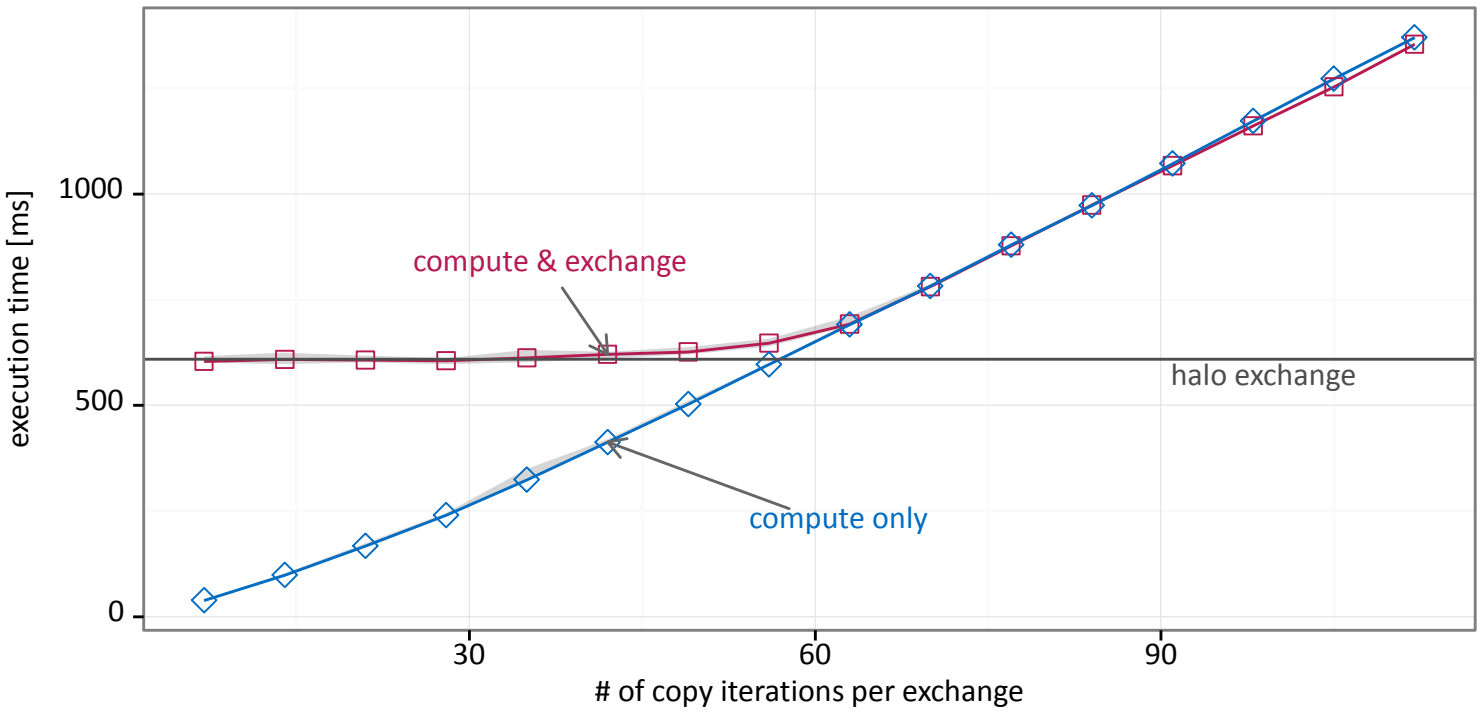
Implementation of the dCUDA runtime system



T. Gysi, J. Baer, TH: dCUDA: Hardware Supported Overlap of Computation and Communication, ACM/IEEE SC16 (preprint at SPCL page)

Overlap of a copy kernel with halo exchange communication

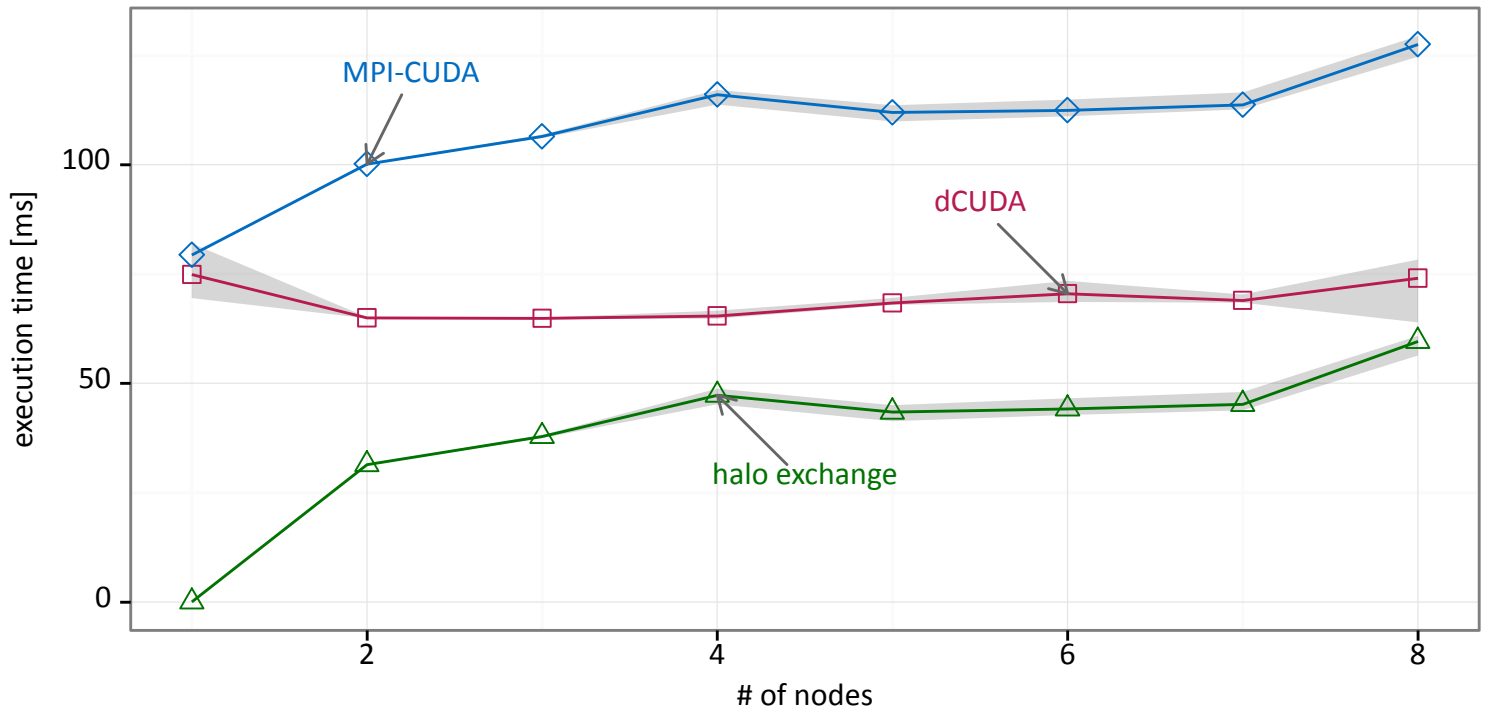
- Benchmarked on Greina (8 Haswell nodes with 1x Tesla K80 per node)



T. Gysi, J. Baer, TH: dCUDA: Hardware Supported Overlap of Computation and Communication, ACM/IEEE SC16 (preprint at SPCL page)

Weak scaling of MPI-CUDA and dCUDA for a stencil program

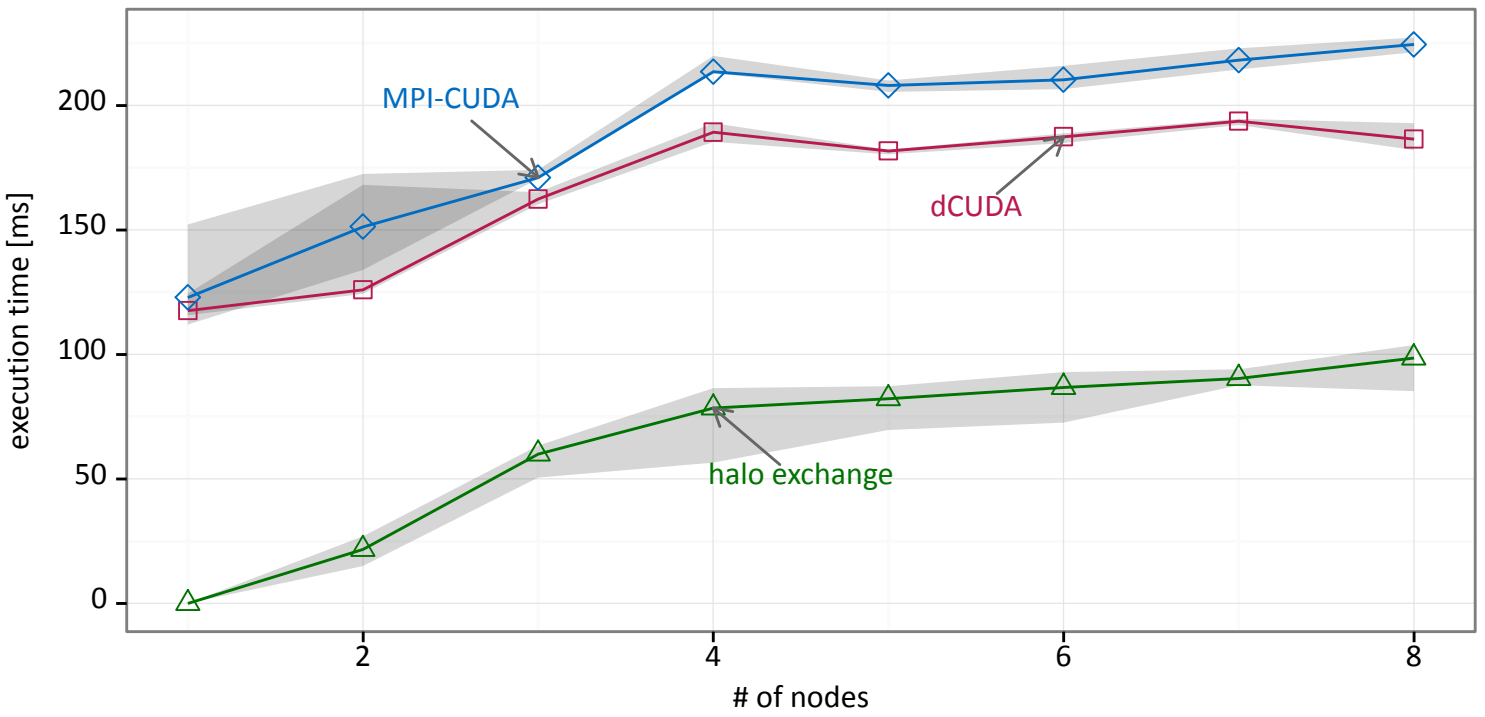
- **Benchmarked on Greina (8 Haswell nodes with 1x Tesla K80 per node)**



T. Gysi, J. Baer, TH: dCUDA: Hardware Supported Overlap of Computation and Communication, ACM/IEEE SC16 (preprint at SPCL page)

Weak scaling of MPI-CUDA and dCUDA for a particle simulation

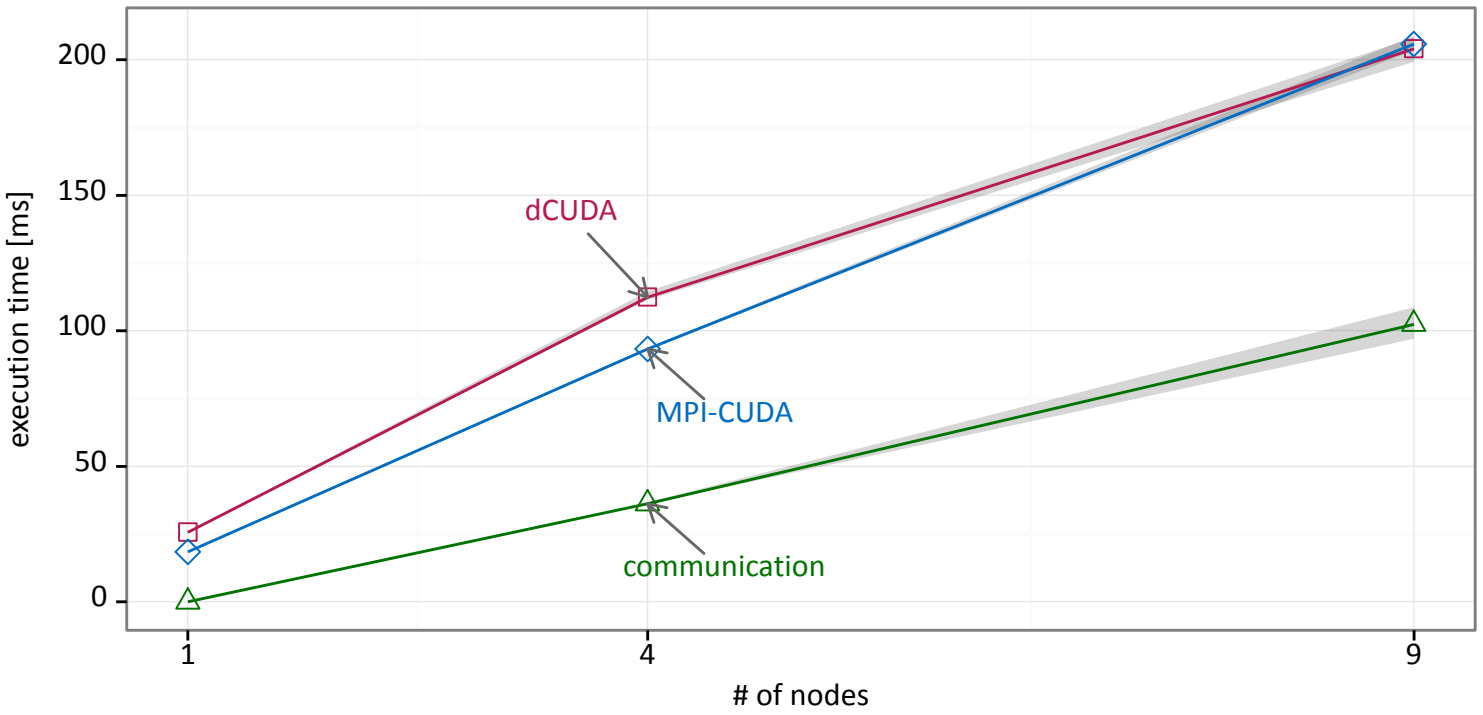
- Benchmarked on Greina (8 Haswell nodes with 1x Tesla K80 per node)



T. Gysi, J. Baer, TH: dCUDA: Hardware Supported Overlap of Computation and Communication, ACM/IEEE SC16 (preprint at SPCL page)

Weak scaling of MPI-CUDA and dCUDA for sparse-matrix vector multiplication

- Benchmarked on Greina (8 Haswell nodes with 1x Tesla K80 per node)



T. Gysi, J. Baer, TH: dCUDA: Hardware Supported Overlap of Computation and Communication, ACM/IEEE SC16 (preprint at SPCL page)

Not just your basic, average, everyday, ordinary, run-of-the-mill, ho-hum stencil optimizer

- **Complete performance models for:**
  - Computation (very simple)
  - Communication (somewhat tricky, using sets and Minkowski sums, parts of the PM)
- **Established a stencil algebra**
  - Complete enumeration of **all** program variants
- **Analytic tuning of stencil programs (using STELLA)**
  - 2.0-3.1x speedup against naive implementations
  - 1.0-1.8x speedup against expert tuned implementations
- **dCUDA enables overlap of communication and computation**
  - Similar to the throughput computing/CUDA idea, just distributed memory
  - Also simplifies programming (no kernel/host code separation)

Sponsors: ETH-RAT, swissuniversities Schweizerische Universitätskonferenz

PASC17 Platform for Advanced Scientific Computing Conference
Lugano Switzerland 26-28 June 2017

T. Gysi, T. Grosser, TH: MODESTO: Data-centric Analytic Optimization of Complex Stencil Programs on Heterogeneous Architectures, ACM ICS'15
T. Gysi, J. Baer, TH: dCUDA: Hardware Supported Overlap of Computation and Communication, ACM/IEEE SC16 (preprint at SPCL page)

Backup Slides