PMPP 2015/16

TECHNISCHE UNIVERSITÄT DARMSTADT

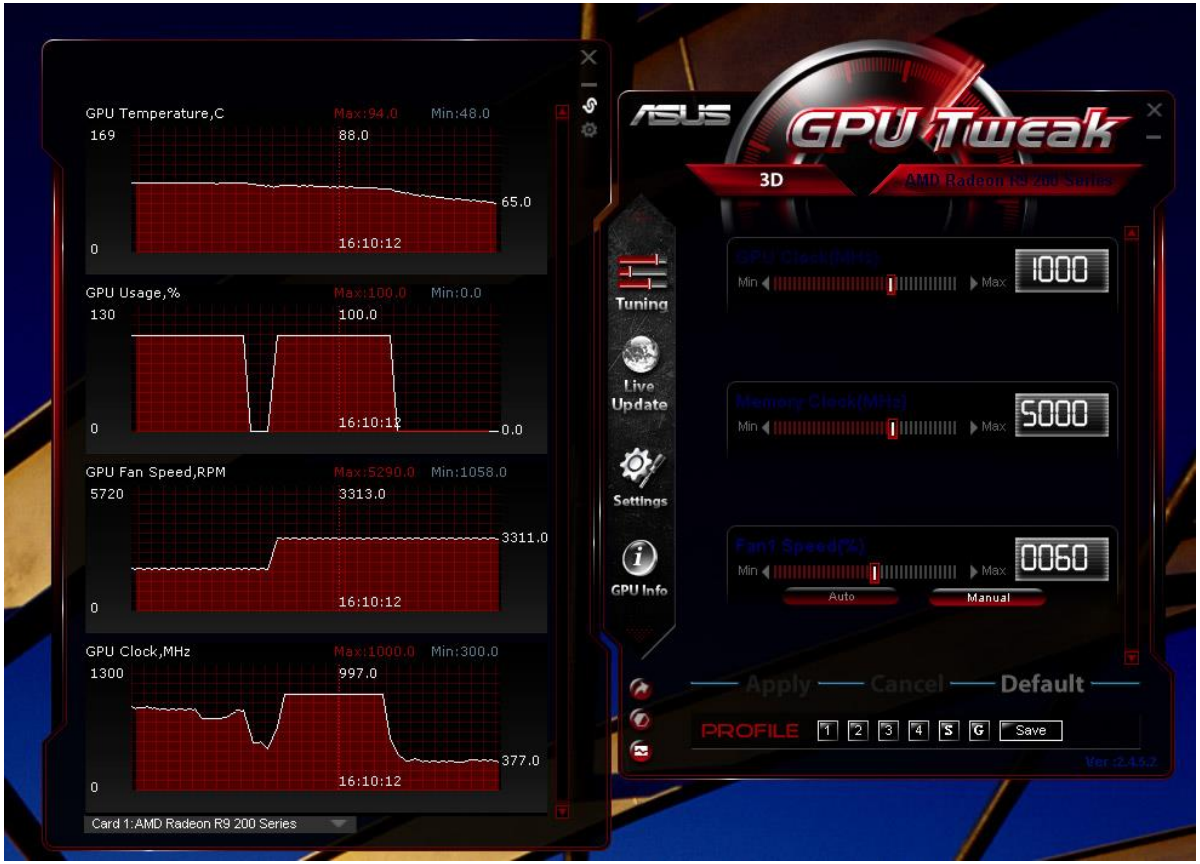
GPU Auto-Tuning

by Nicolas Weber

What is Auto-Tuning?

©2008 sshtroumpfy [https://c1.staticflickr.com/3/2179/2381555780_b41fa75386_z.jpg]

What is Auto-Tuning?



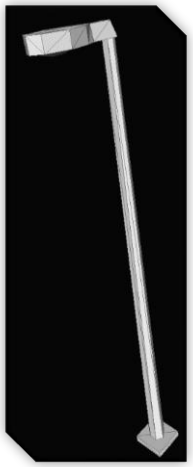
©Chip [http://www.chip.de/ii/2/2/1/8/5/9/4/1/gputweak_g_neu-bd586272175026bc.jpg]

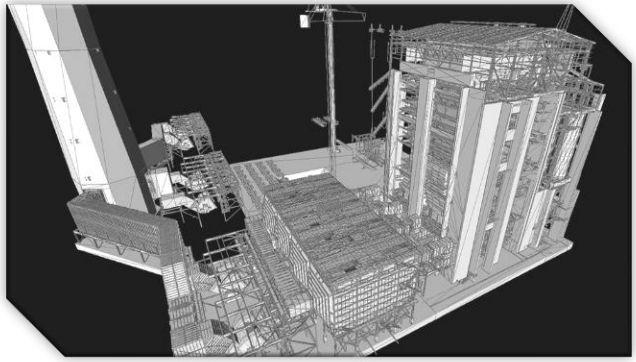
What is Auto-Tuning?

TECHNISCHE UNIVERSITÄT DARMSTADT

- Automated optimization of application performance
  - By optimizing the program code
  - Without changing the hardware or clock rate

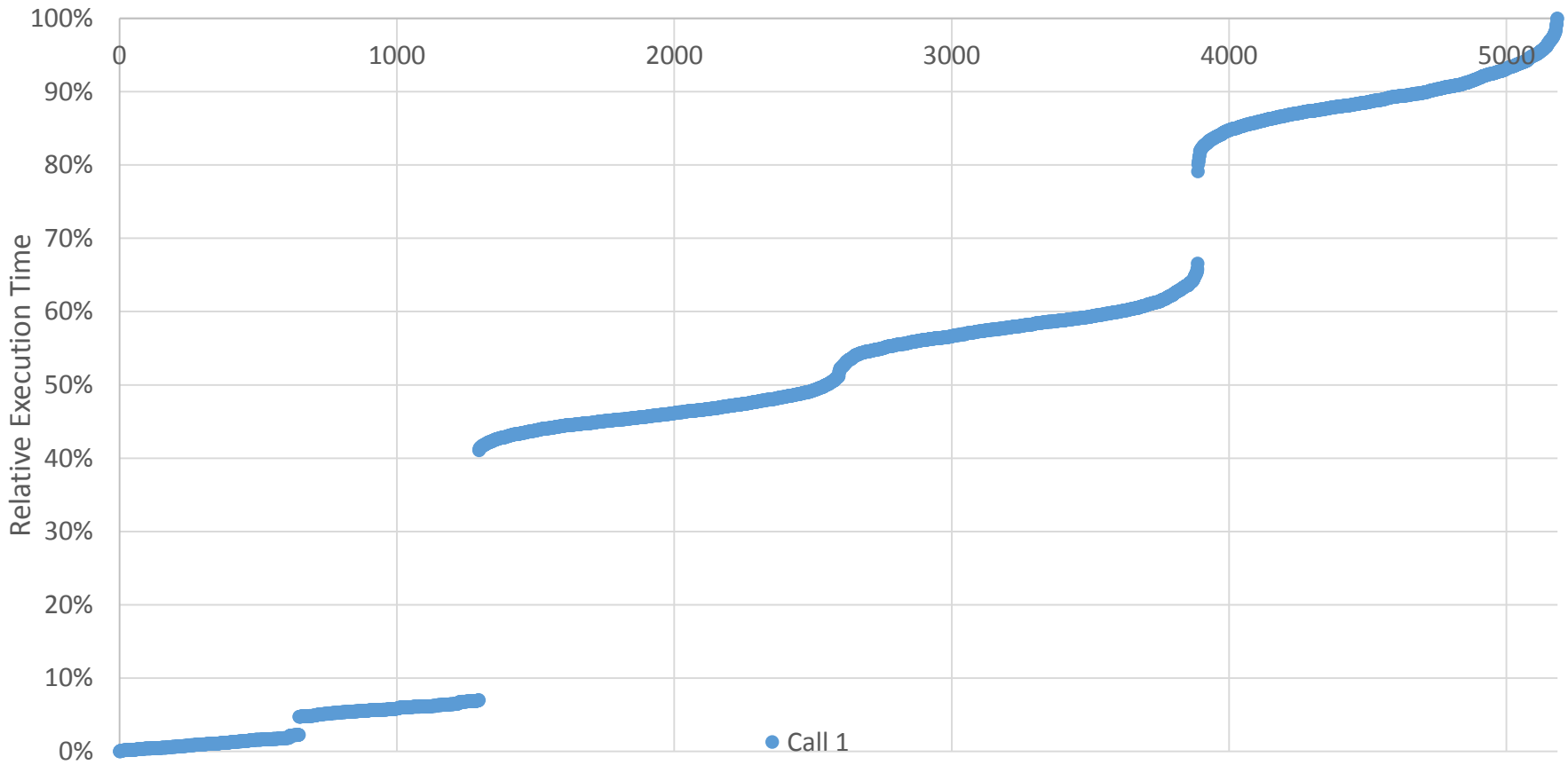
- Why should I use auto-tuning?
  - Changing GPU architectures every 1-2 years
    - requires to re-optimize every time a new GPU is released
    - also significant differences inside same architecture, e.g.:
      - Memory: DDR3 vs GDDR5
      - Number of multi-processors
  - Varying input data





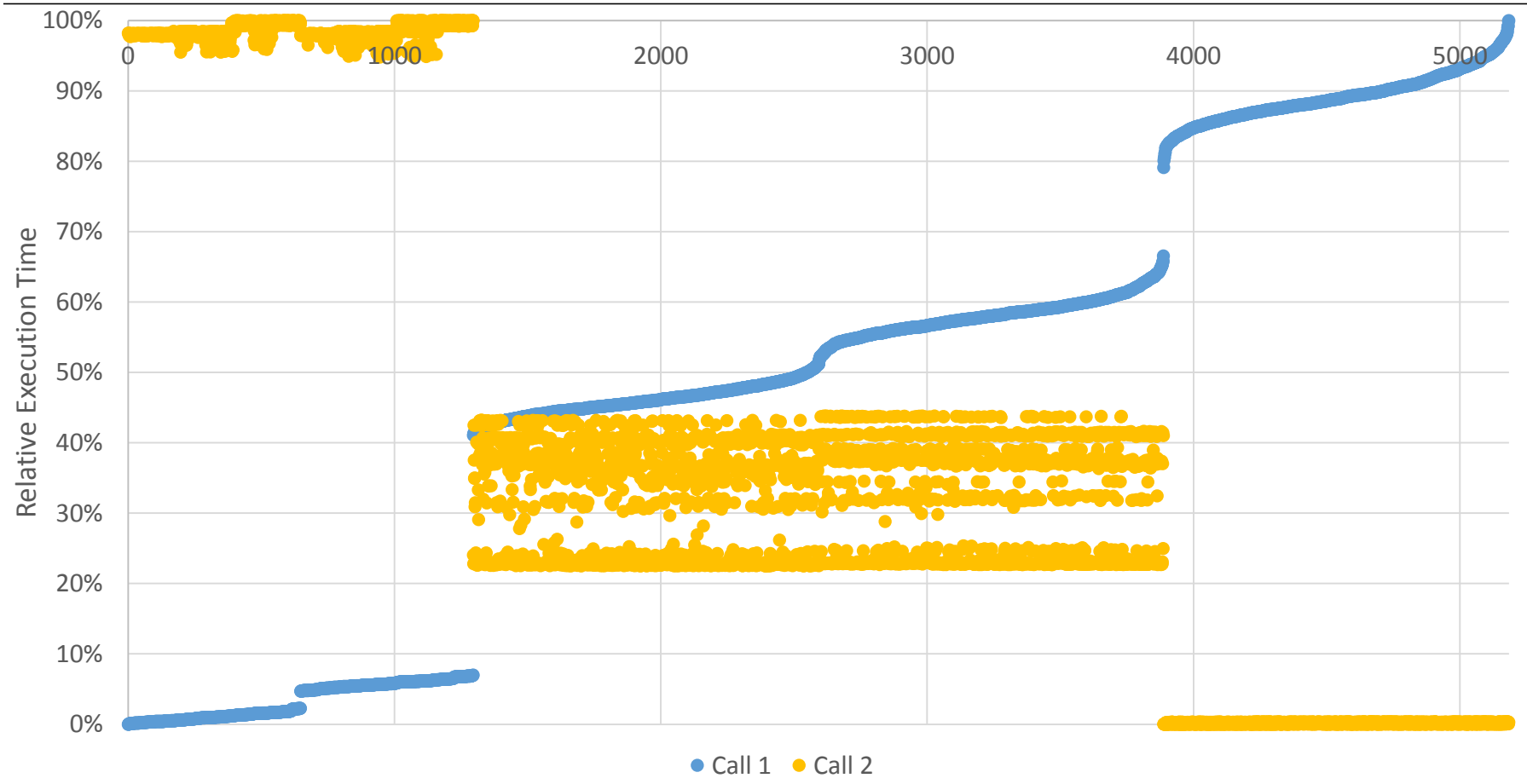
Input Sensitivity

TECHNISCHE UNIVERSITÄT DARMSTADT



GCC
Graphics, Capture and Massively Parallel Computing

Input Sensitivity



What is Auto-Tuning?

TECHNISCHE UNIVERSITÄT DARMSTADT

- Automated optimization of application performance
  - By optimizing the program code
  - Without changing the hardware or clock rate

- Why should I use auto-tuning?
  - Changing GPU architectures every 1-2 years
    - requires to re-optimize every time a new GPU is released
    - also significant differences inside same architecture, e.g.:
      - Memory: DDR3 vs GDDR5
      - Number of multi-processors
  - Varying input data
  - High Number of available optimizations

Optimizations

TECHNISCHE UNIVERSITÄT DARMSTADT

- Memory Access
  - Layouts: AoS, SoA or AoSoA
  - Matrix Transpositions
  - Memories (Global, Texture, Constant, Shared and Local)
  - Cache Usage
    - Using/Bypassing caches
    - L1 cache size (prefer SM, L1, EQ)
  - Shared Memory Bank width (32/64 Bit)

- Loop Transformations
  - e.g. unroll, reorder nested loops, ...

Optimizations

TECHNISCHE UNIVERSITÄT DARMSTADT

- API Function Usage
  - optimize overlap of computation and asynchronous memcopy
- Workload Distribution
  - e.g. in multi-gpu applications
- Kernel Launch Configuration
  - changing occupancy
- Algorithms
- Kernel Fusion/Splitting
  - When to place all code into one kernel? When to split it?
- Special Functions
  - e.g. __vadd4(int, int) → adds 4x Byte values
- ...

How do Auto-Tuners work?

TECHNISCHE UNIVERSITÄT DARMSTADT

1. How to integrate optimizations into the application?
2. How to determine optimal applications configurations?
3. How to apply the optimizations?

How to integrate optimizations into the application?

How to integrate optimizations into the application?

TECHNISCHE UNIVERSITÄT DARMSTADT

- High-Level Languages
  - Abstract description of the algorithm
  - Map language constructs onto parallel algorithms
    (e.g. min_reduce(data) / prefix_sum(data))
  - High-Level language is synthesized into low-level language
    (e.g. Python → CUDA)
  - Terra (Lua), Copperhead (Python), HiCUDA, ...

  - Require to learn a new language
  - Difficult to incorporate in big software products
  - Limited language construct support

How to integrate optimizations into the application?

TECHNISCHE UNIVERSITÄT DARMSTADT

- Compilers
  - Analyzes and adjusts code to be optimal
  - Can require code annotations
    (e.g. #pragma unroll)

  - Difficult to implement
  - Many optimizations impossible to implement this way
  - Depends on support of the compiler
    - new hardware architectures
    - operating system
    - CUDA Version
    - ...

How to integrate optimizations into the application?

TECHNISCHE UNIVERSITÄT DARMSTADT

- Application Specific Libraries
  - Hide operations by function calls
    (e.g. multiply(MatrixA, MatrixB))
  - Specifically designed to optimize problems of an application domain
  - Very easy to use

  - Limited to one application domain
  - Often incompatible to other libraries / auto-tuners

How to integrate optimizations into the application?

TECHNISCHE UNIVERSITÄT DARMSTADT

- Code Generation
  - Generates application specific data structures or kernels
  - Not necessarily bound to specific application domain
  - Not bound to a specific compiler

  - Application code has to be adjusted to use generated code
  - Can be incompatible to other libraries / auto-tuners

How to determine optimal applications configurations?

Static Analysis

TECHNISCHE UNIVERSITÄT DARMSTADT

- Analytically analyze application
  1. Analyzes code
  2. Builds models (Code and Hardware)
  3. Simulates code execution based on models

- Pros
  - Usually very fast
  - Repeatable results

- Cons
  - Relies on how good models represent code or hardware
  - Does not take input data into account

Empirical Profiling

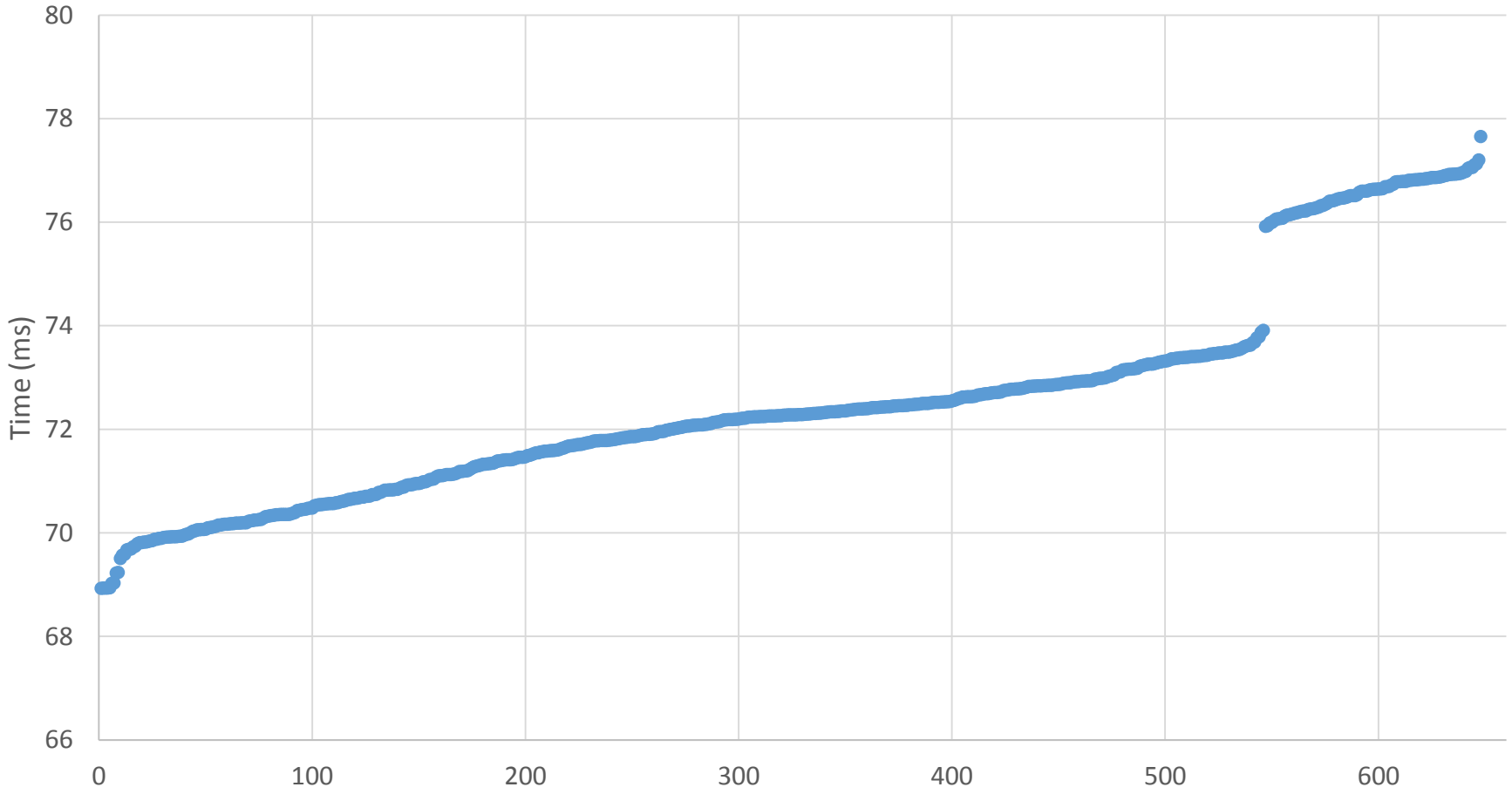
TECHNISCHE UNIVERSITÄT DARMSTADT

- Measures execution time
  1. Compile application in a specific configuration
  2. Run application with real data
  3. Measure time
  4. If not satisfied with result: Goto 1

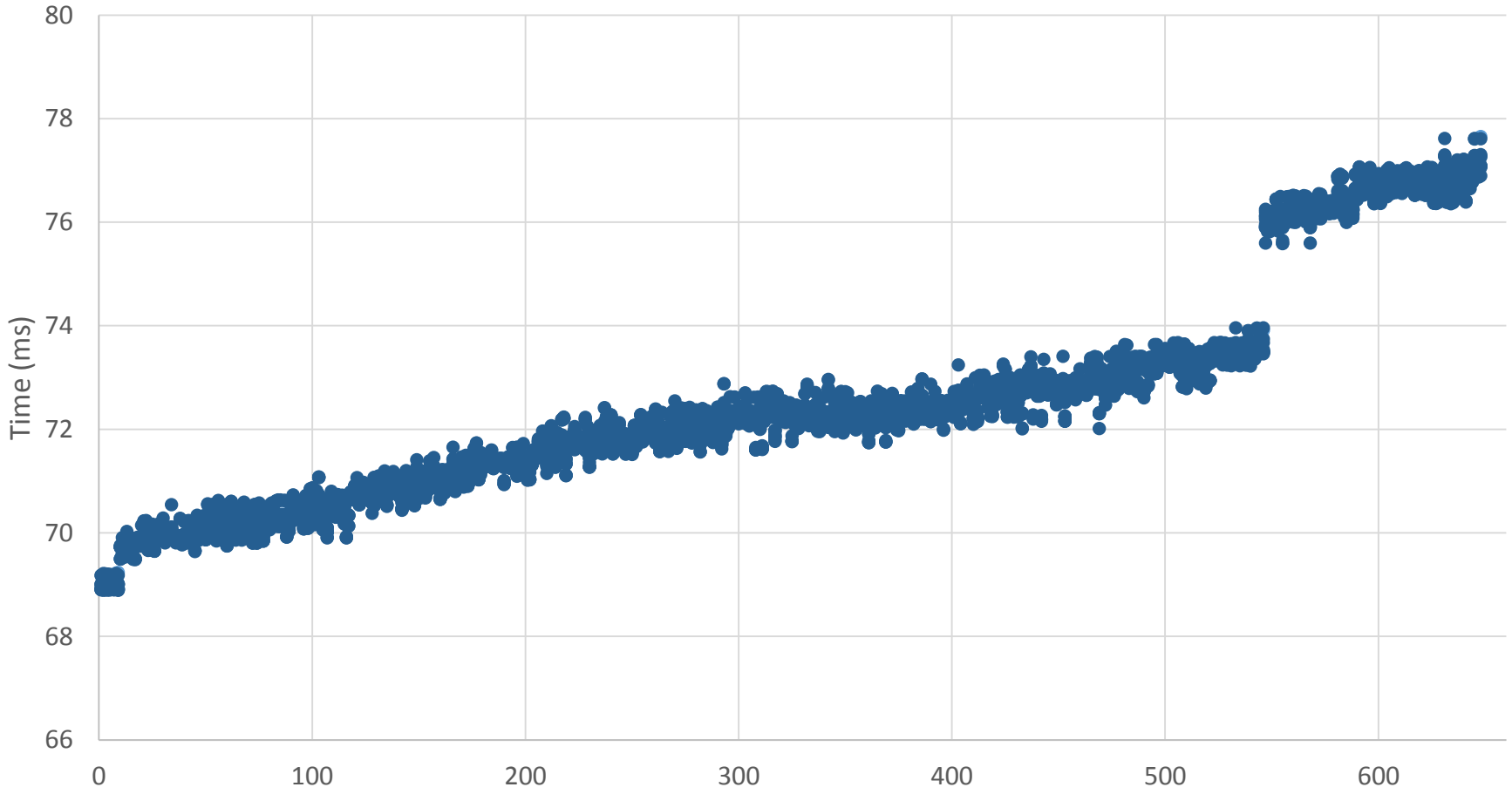
- Pros
  - Executes with real data and on target hardware
  - No new models required for future hardware

- Cons
  - Very time intensive (compilation + application execution)
  - Noise makes difficult to reproduce results

Noise



Noise



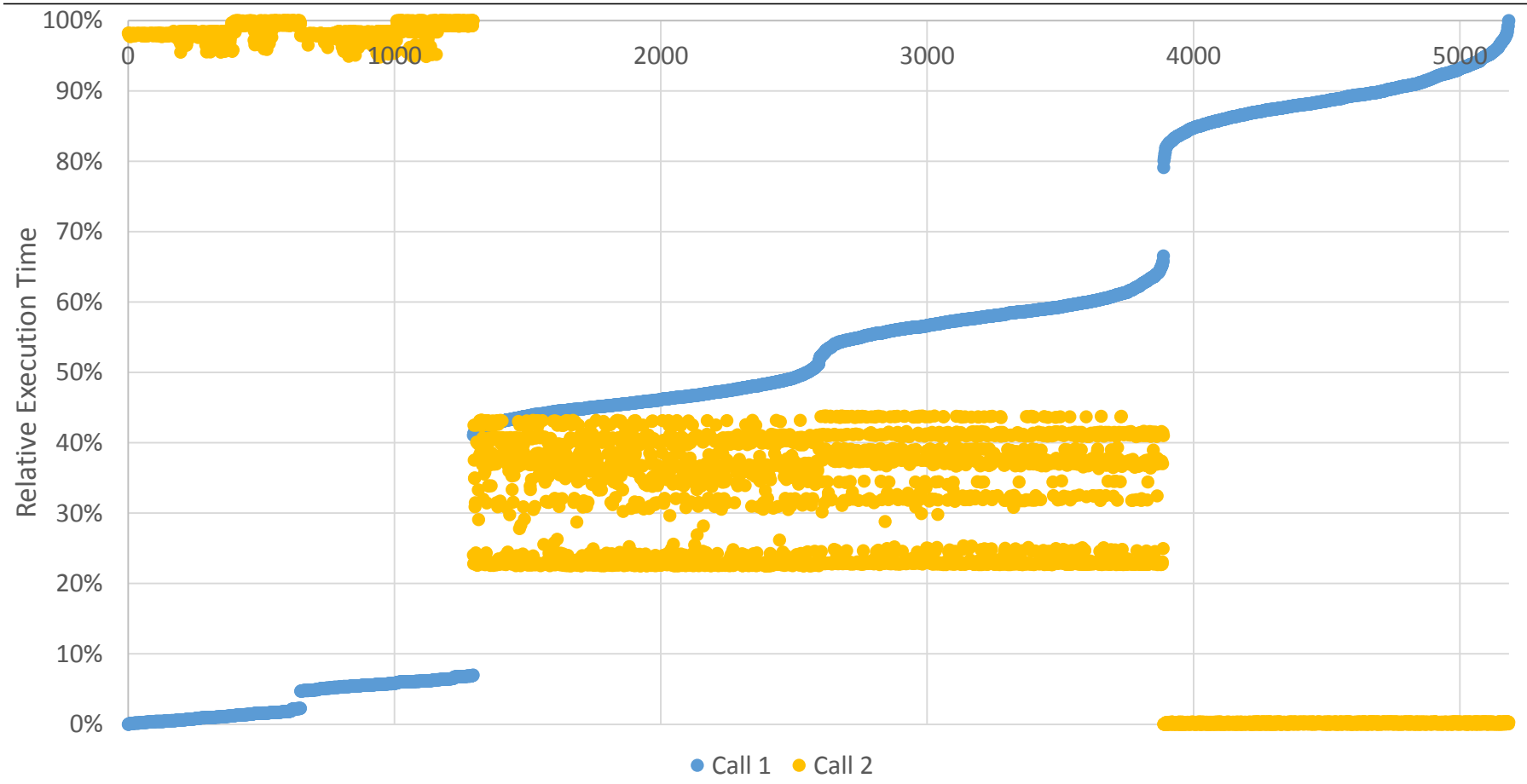
Empirical Profiling: Methods

TECHNISCHE UNIVERSITÄT DARMSTADT

- Static Methods
  - Exhaustive search
  - Greedy algorithm

- Adaptive Methods
  - Evolutionary algorithm
  - Downhill-Simplex-Method
  - “Learning” base algorithms
    (try configurations, which have worked in previous profilings)
  - ...

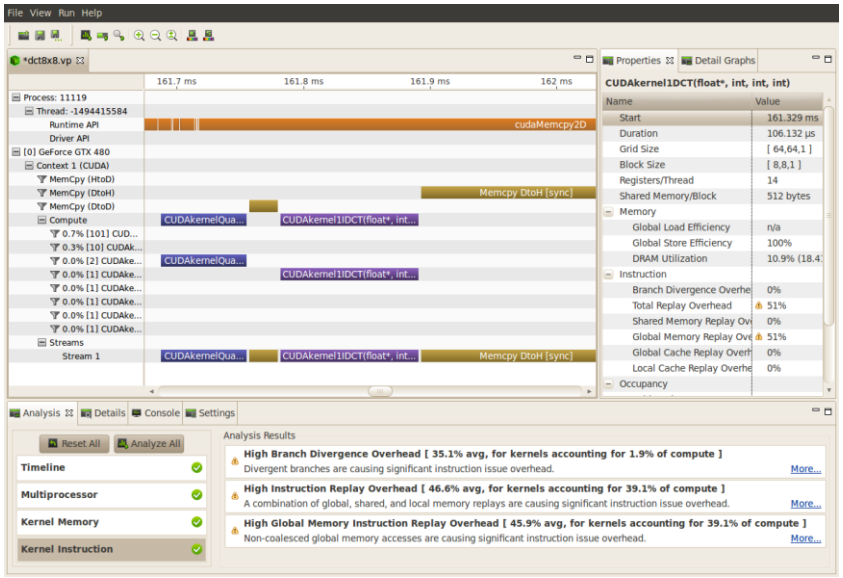
Recap: Input Sensitivity



Excurse: How to measure kernel times?

TECHNISCHE UNIVERSITÄT DARMSTADT

- clock();
  - very unreliable as only records wall-time
- std::chrono
  - better, but also measures OS/Driver overhead
- Nvidia Profiler
  - executes time measurement inside GPU driver



Excurse: CUPTI

TECHNISCHE UNIVERSITÄT DARMSTADT

- CUDA Profiling Tools Interface (CUPTI)
  - allows to access CUDA profiler inside applications
  - APIs:
    - Activity: Record CUDA activities (e.g. kernel calls)
    - Callback: Pre/Post-Callback for all CUDA functions
    - Event: React on certain Events (e.g. in streams)
    - Metric: Gathers metric data of kernel (e.g. Cache hit/miss rates)

- Additional Reading: http://docs.nvidia.com/cuda/cupti/index.html

How to apply the optimizations?

How to apply the optimizations?

TECHNISCHE UNIVERSITÄT DARMSTADT

- Only one optimal solution
  - compile application in optimal configuration
- Multiple optimal solutions

How to apply the optimizations?

TECHNISCHE UNIVERSITÄT DARMSTADT

- How to distinguish multiple optimization scenarios?
  - Use decision models to determine optimal configurations
    - Decision Trees, SVM, Neural Nets, ...
  - Meta data for decision models
    - Automatically gathered
      - Array sizes, launch configurations, application parameters, ...
    - Data analysis functions
      - requires programmer to implement a such function, to analyze and categorize the input data

How to apply the optimizations?

TECHNISCHE UNIVERSITÄT DARMSTADT

- Application wrapper
  - Decide prior the application start which implementation works best for given data / parameters.
  - Pros: no overhead during runtime
  - Cons: cannot react on changing effects

- Runtime Optimization
  - Decide during runtime, which implementation works best.
  - Pros: can react on changing effects
  - Cons: additional auto-tuning overhead during runtime

Summary

TECHNISCHE UNIVERSITÄT DARMSTADT

- Motivation
- Optimizations
- Auto-Tuning Challenges
  - How to integrate the optimizations?
    - High-Level languages, Libs, Compiler and Code-Generation
  - How to analyze applications?
    - Static Analysis
    - Empirical Profiling
      - Static / Adaptive methods
  - How to apply optimizations?
    - One / Multiple optimal configurations
    - Decision Models
    - Prior application start / during runtime

MATOG Auto-Tuner

MATOG Features

TECHNISCHE UNIVERSITÄT DARMSTADT

- Auto-Tuner for CUDA array access
  - Array of Struct layouts: AoS, SoA or AoSoA (32-tiles)
  - Transposition of multi-dimensional arrays
  - Adjustment of L1 cache size (Kepler GPUs)
  - Usage of Texture Memory
  - Preprocessor Optimizations, e.g. #if MyDefine == 5 ...
  - ...

How to integrate optimizations into the application?

Programming Interface

TECHNISCHE UNIVERSITÄT DARMSTADT

- Normal C++ array access
  - `array[x + x_width * y + x_width * y_width * z].field = 5.0f;`
  - difficult to read
  - difficult to optimize

- MATOG
  - Multi-dimensional array access
    `array[x][y][z].field = 5.0f;`
  - easy to read
  - easy to interface

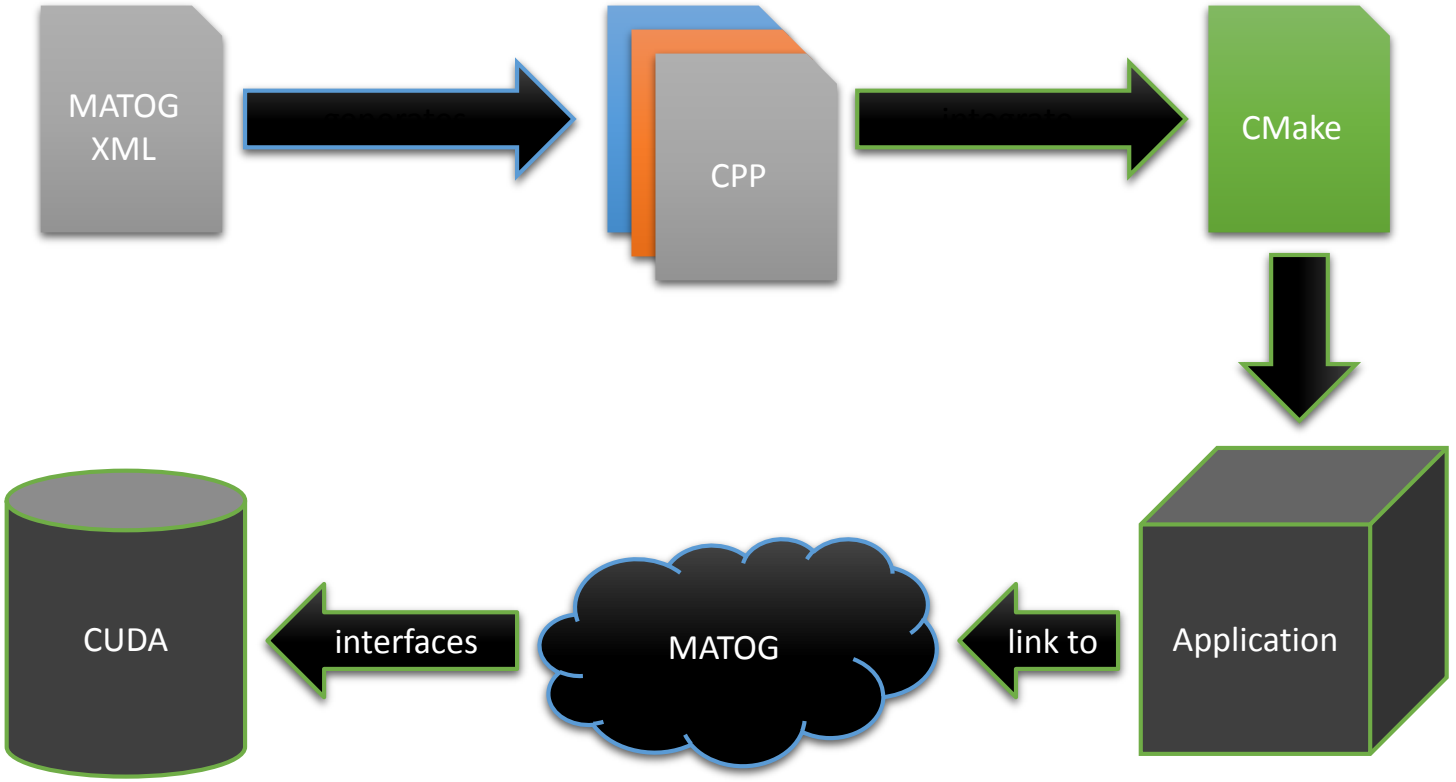
Workflow


```
// MATOG XML
<array name="MyArray" dimensions="2">
  <field name="x" type="float" />
  <field name="y" type="float" />
</array>

// CPP
struct MyArray {
  float x;
  float y;
};

MyArray array[X][Y];
```

Workflow


MATOG
XML
generate
CPP
include
CMake
CUDA
interfaces
MATOG
link to
Application

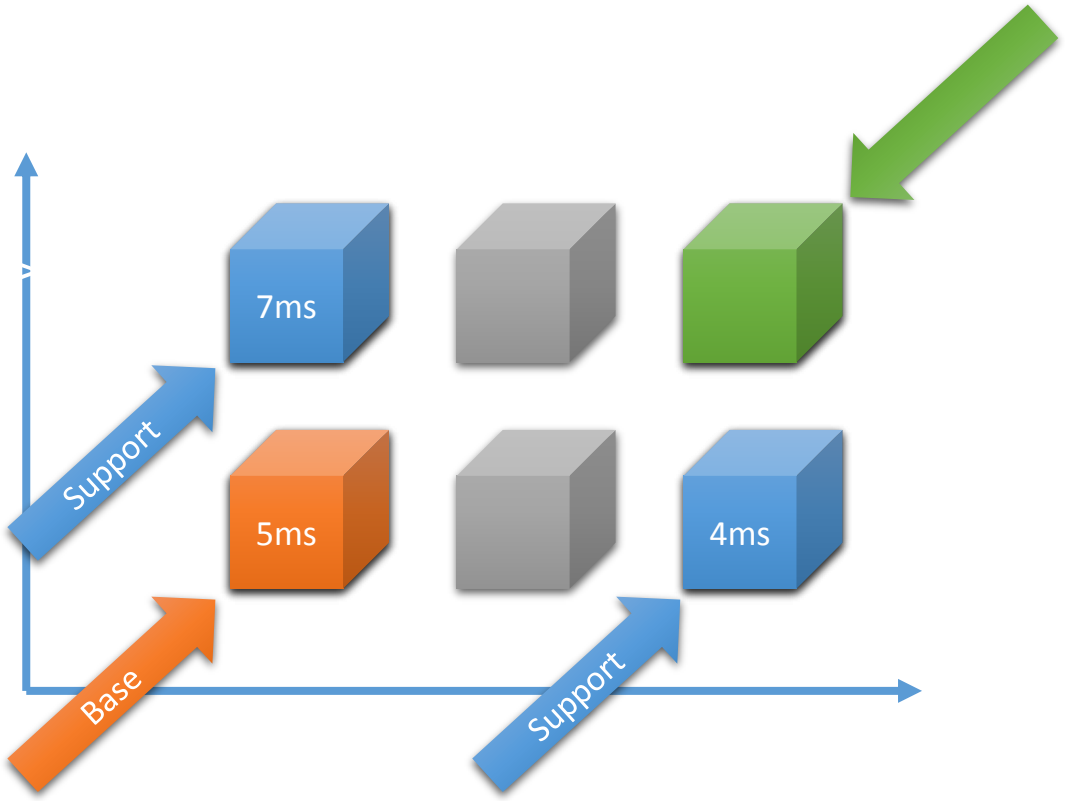
How to determine optimal applications configurations?

Prediction Guided Profiling

TECHNISCHE UNIVERSITÄT DARMSTADT

- MATOG uses its own prediction guided profiling method
  - Profiles limited set of configurations
  - Uses linear model to predict performance

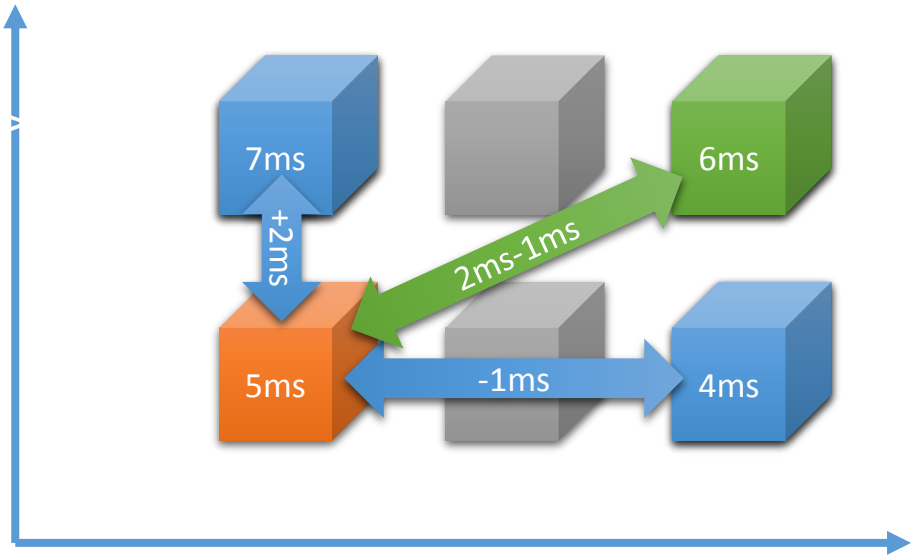
Toy Example: Performance Estimation 2D



Toy Example: Performance Estimation 2D

TECHNISCHE UNIVERSITÄT DARMSTADT

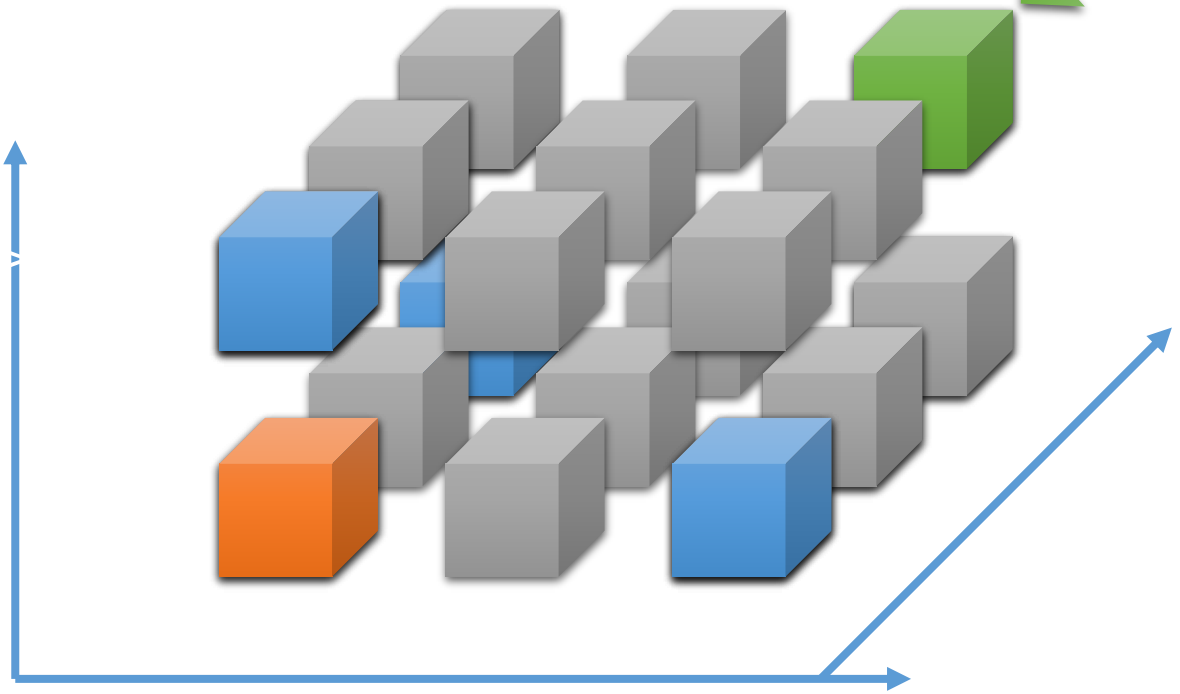
- Predicted Execution Time
  - Execution Time of Base + Δ(Base, Support Configurations)



GCC
Graphics, Capture and Massively Parallel Computing

Toy Example: Performance Estimation 3D

TECHNISCHE UNIVERSITÄT DARMSTADT

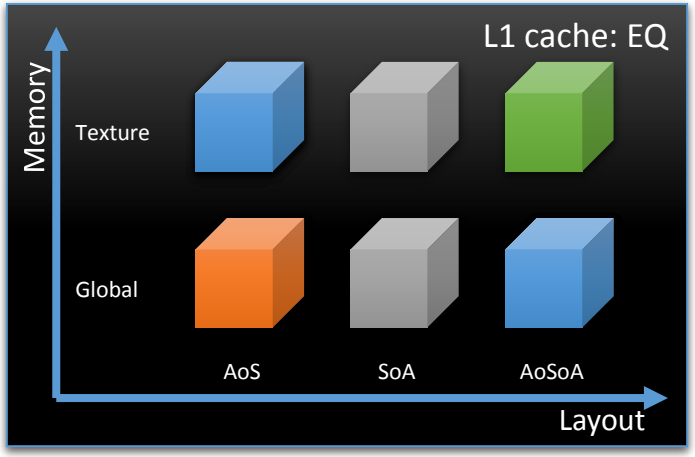


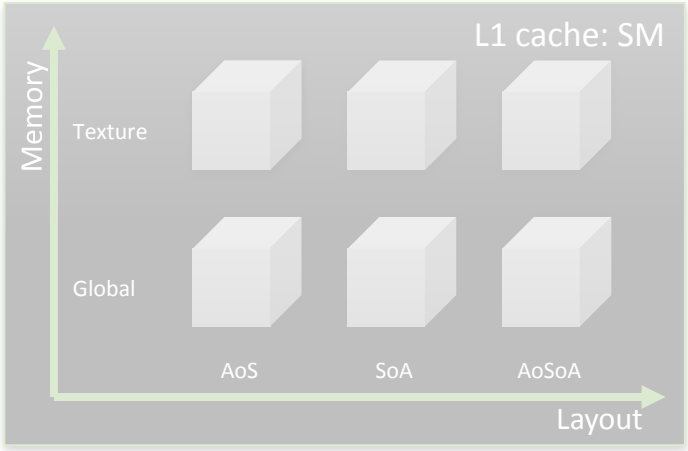
Non-Linear Relationship

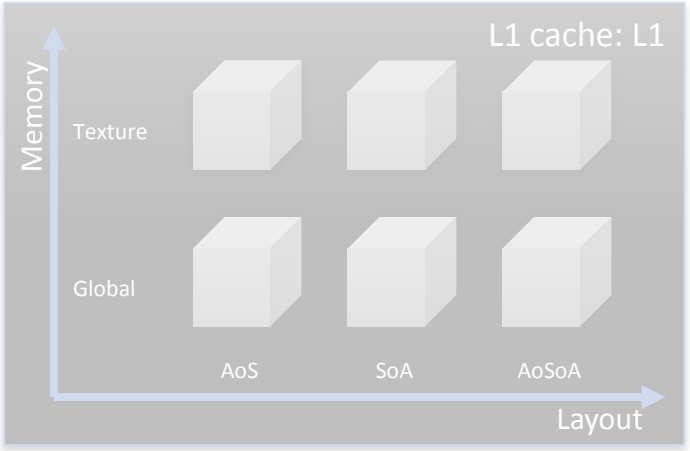
TECHNISCHE UNIVERSITÄT DARMSTADT

- Not all configurations are linearly related to each other
- Independent dimensions
  - Only affect one array
  - Layout, memory and transposition
- Shared dimensions
  - Affect all arrays
  - L1 Cache size

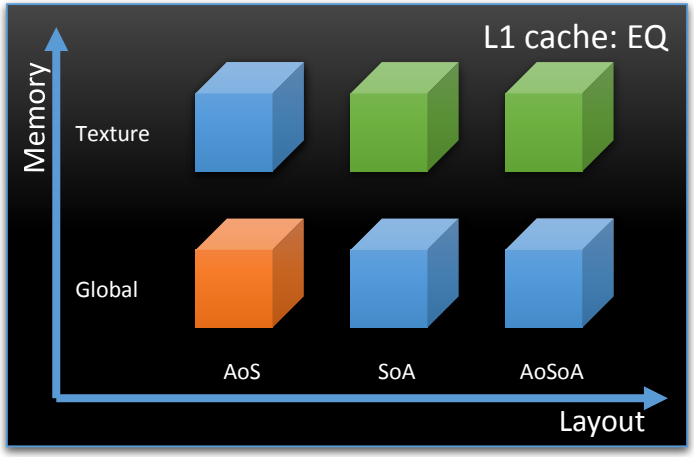
Toy Example: Prediction Domains

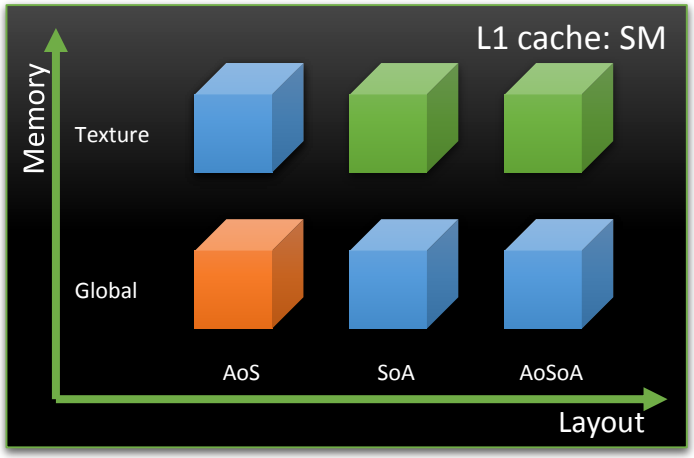


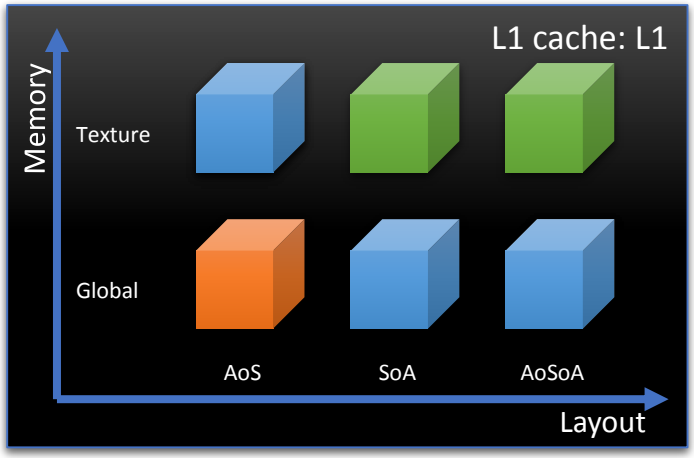




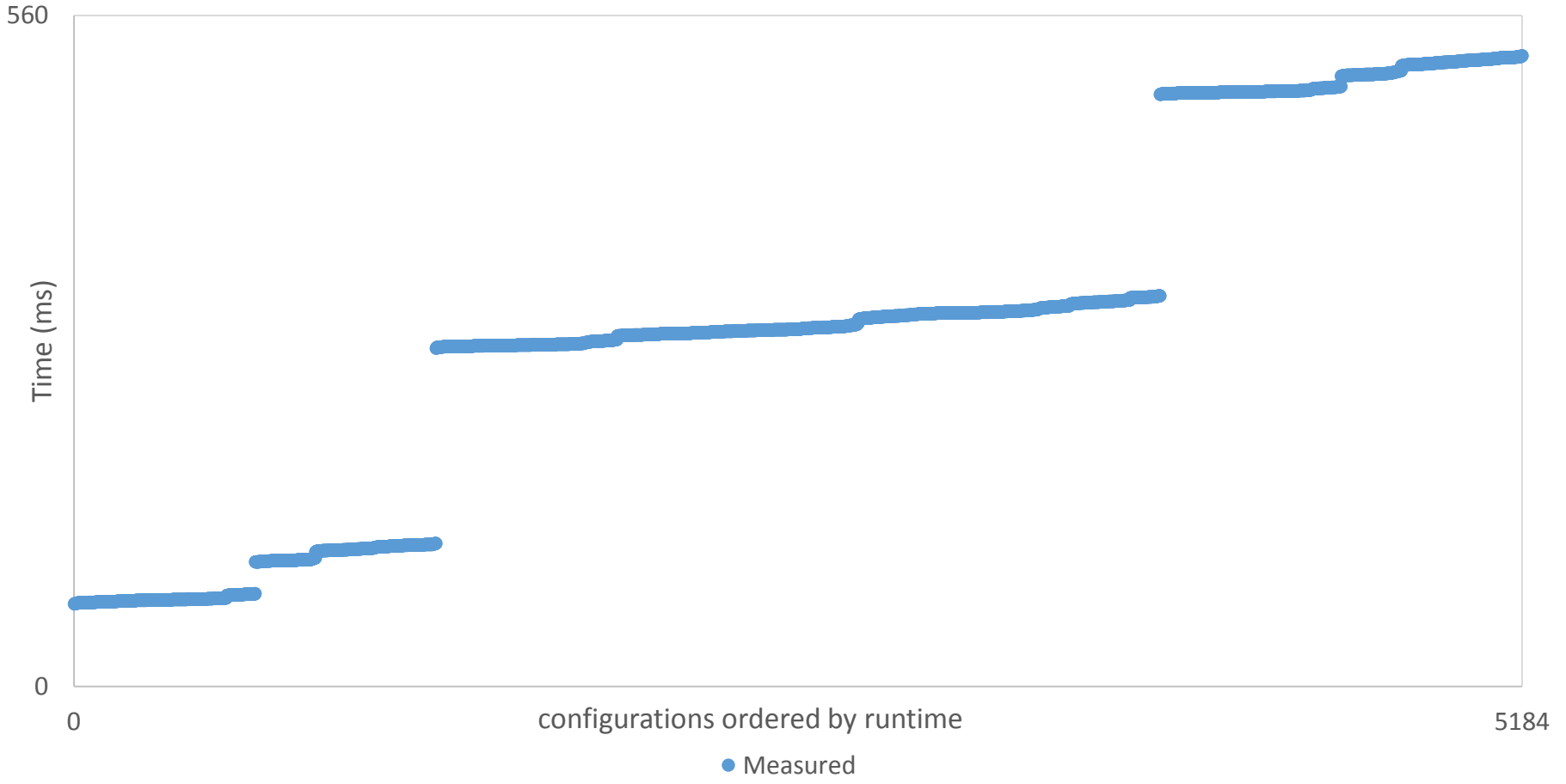
Toy Example: Prediction Domains





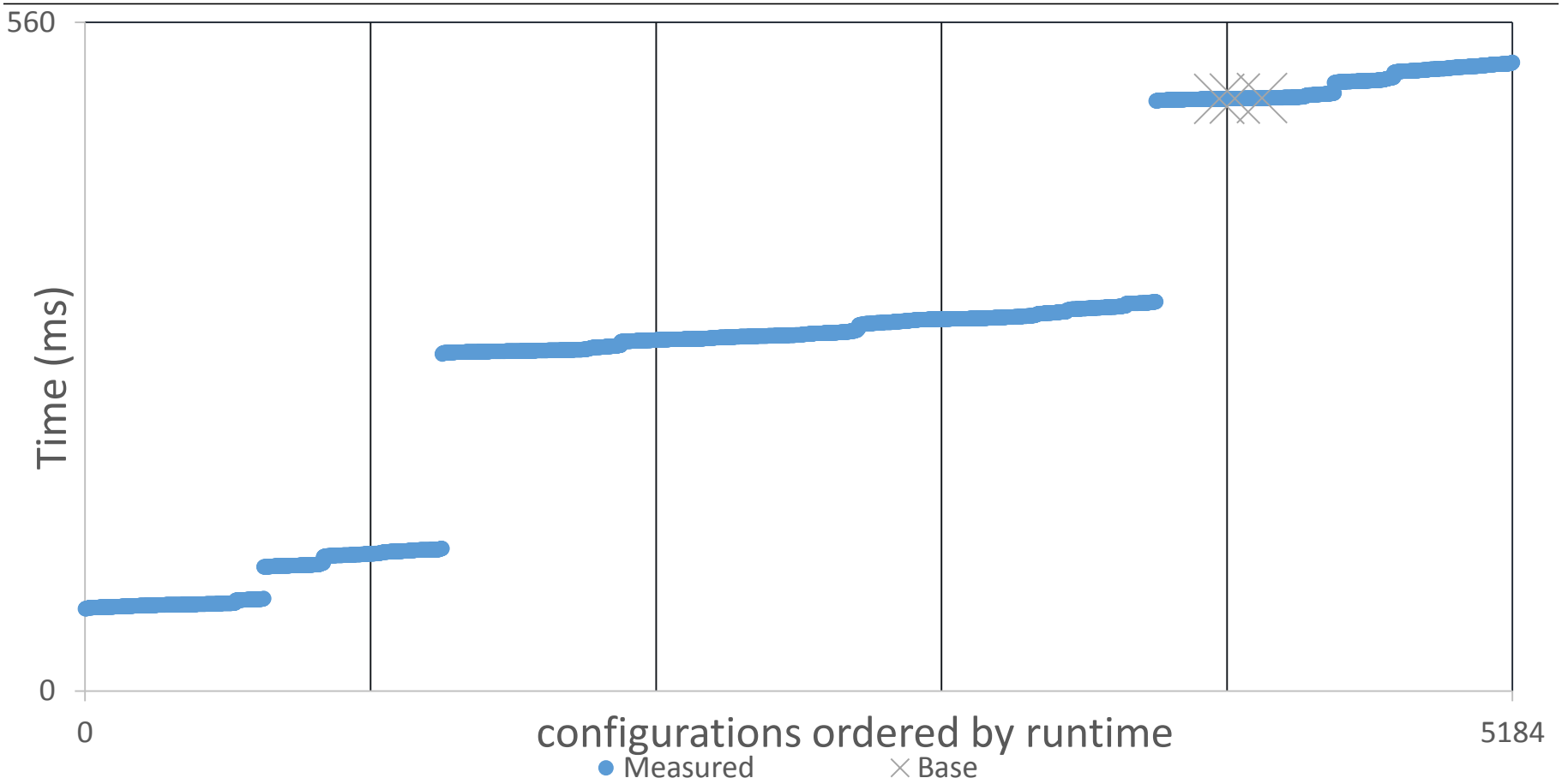


Real Example: Measured Time



Real Example: Base Configurations

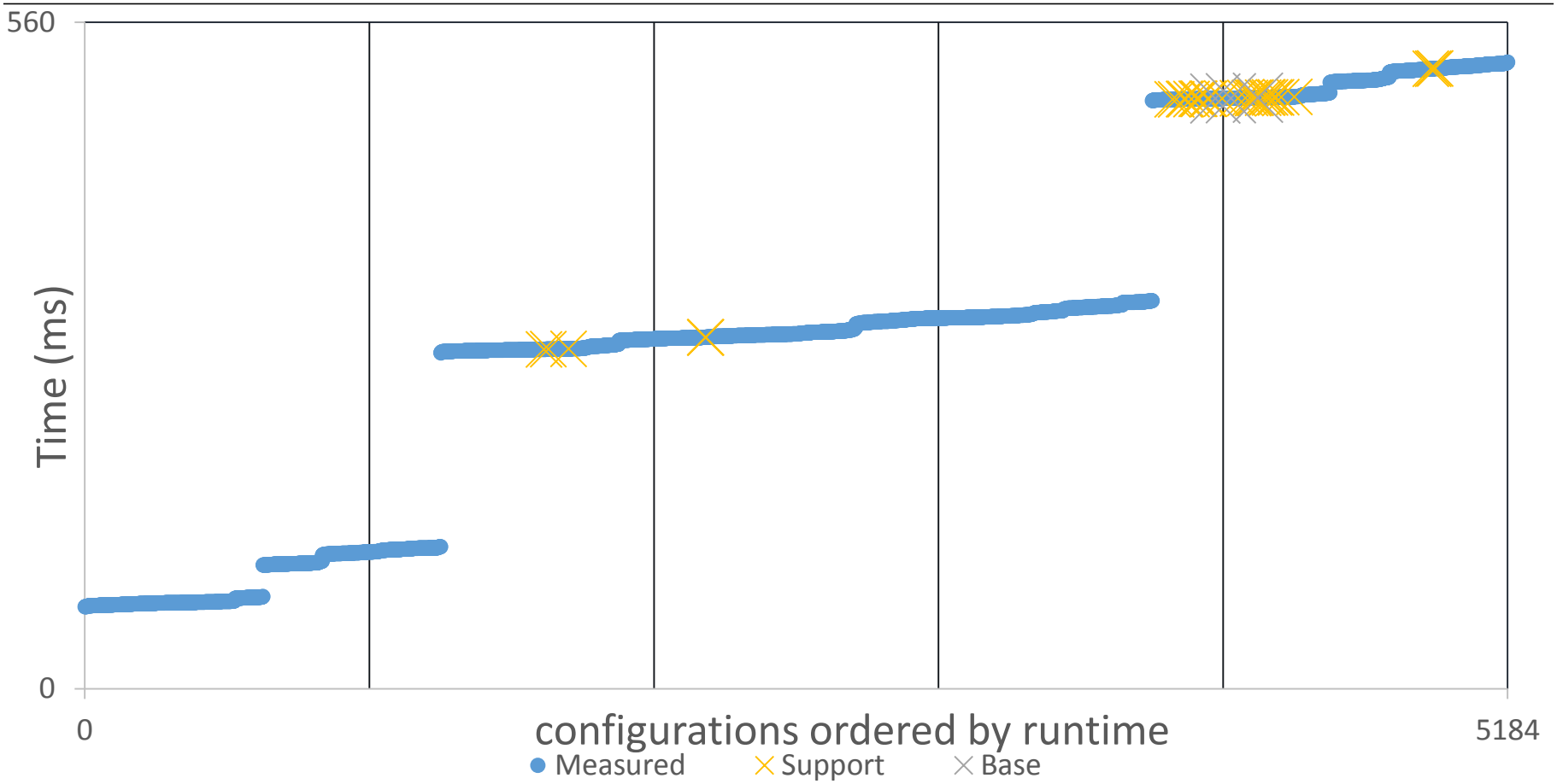
TECHNISCHE UNIVERSITÄT DARMSTADT



GCC
Graphics, Capture and Massively Parallel Computing

Real Example: Support Configurations

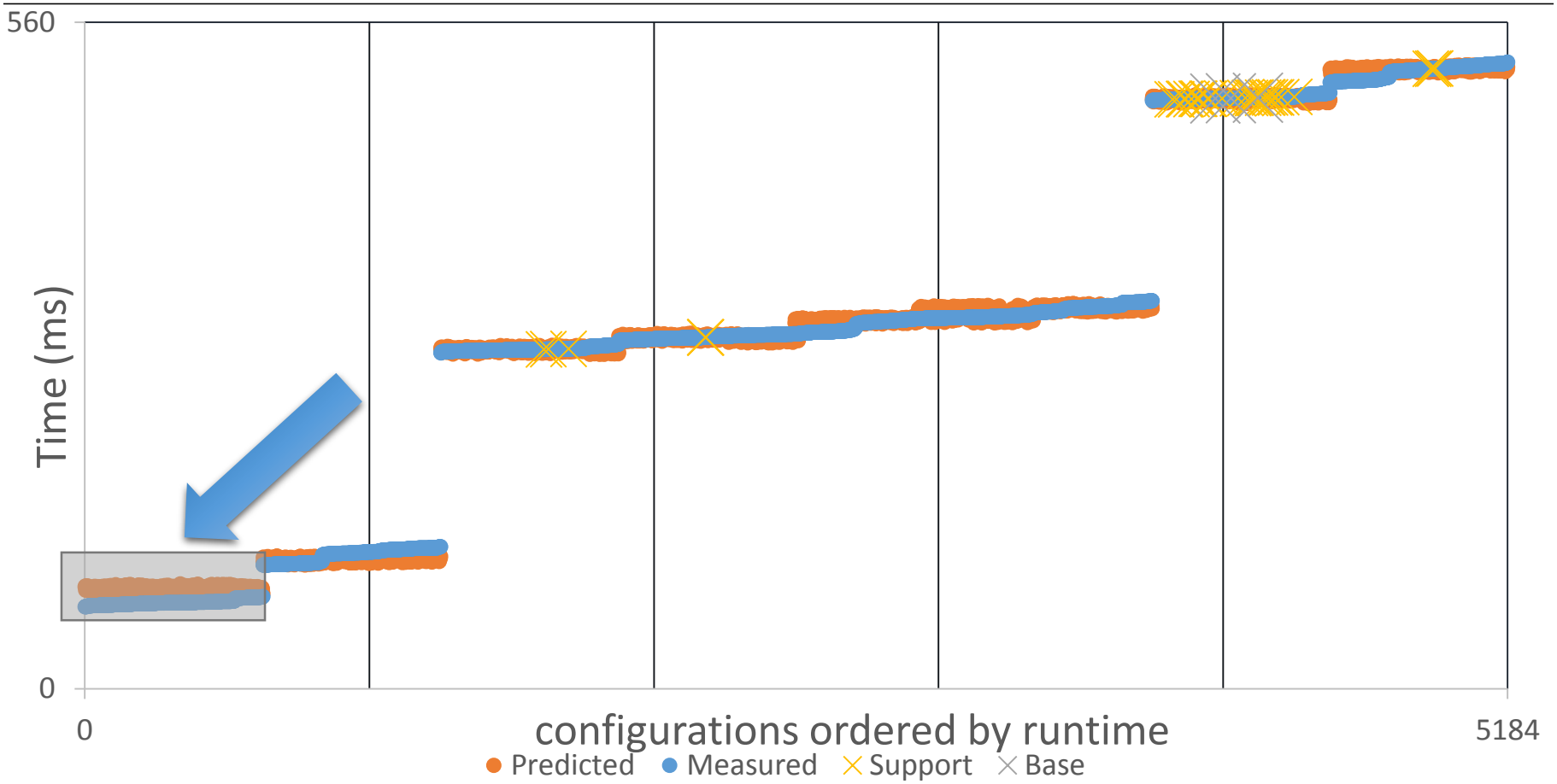
TECHNISCHE UNIVERSITÄT DARMSTADT



GCC
Graphics, Capture and Massively Parallel Computing

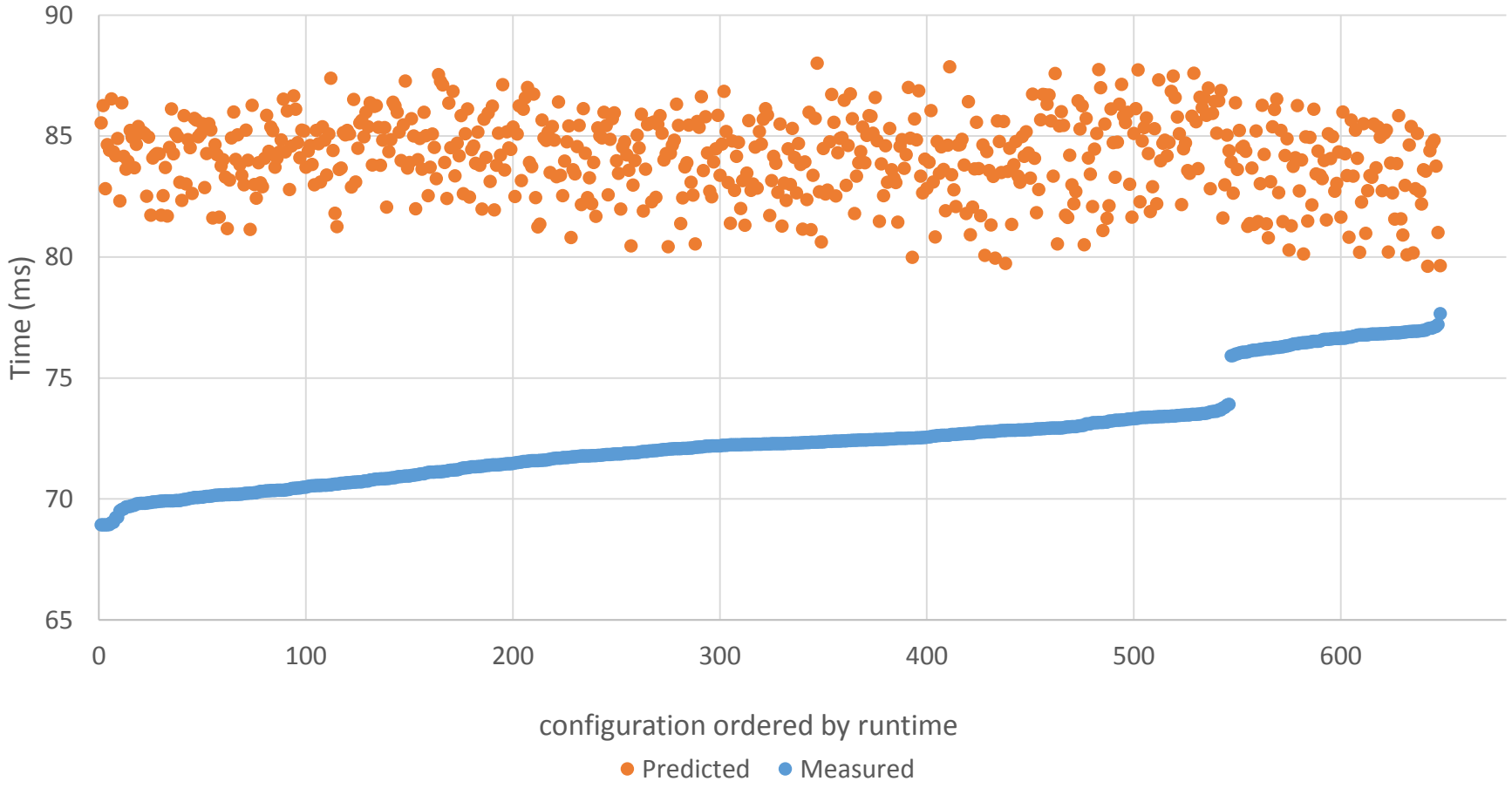
Real Example: Prediction

TECHNISCHE UNIVERSITÄT DARMSTADT

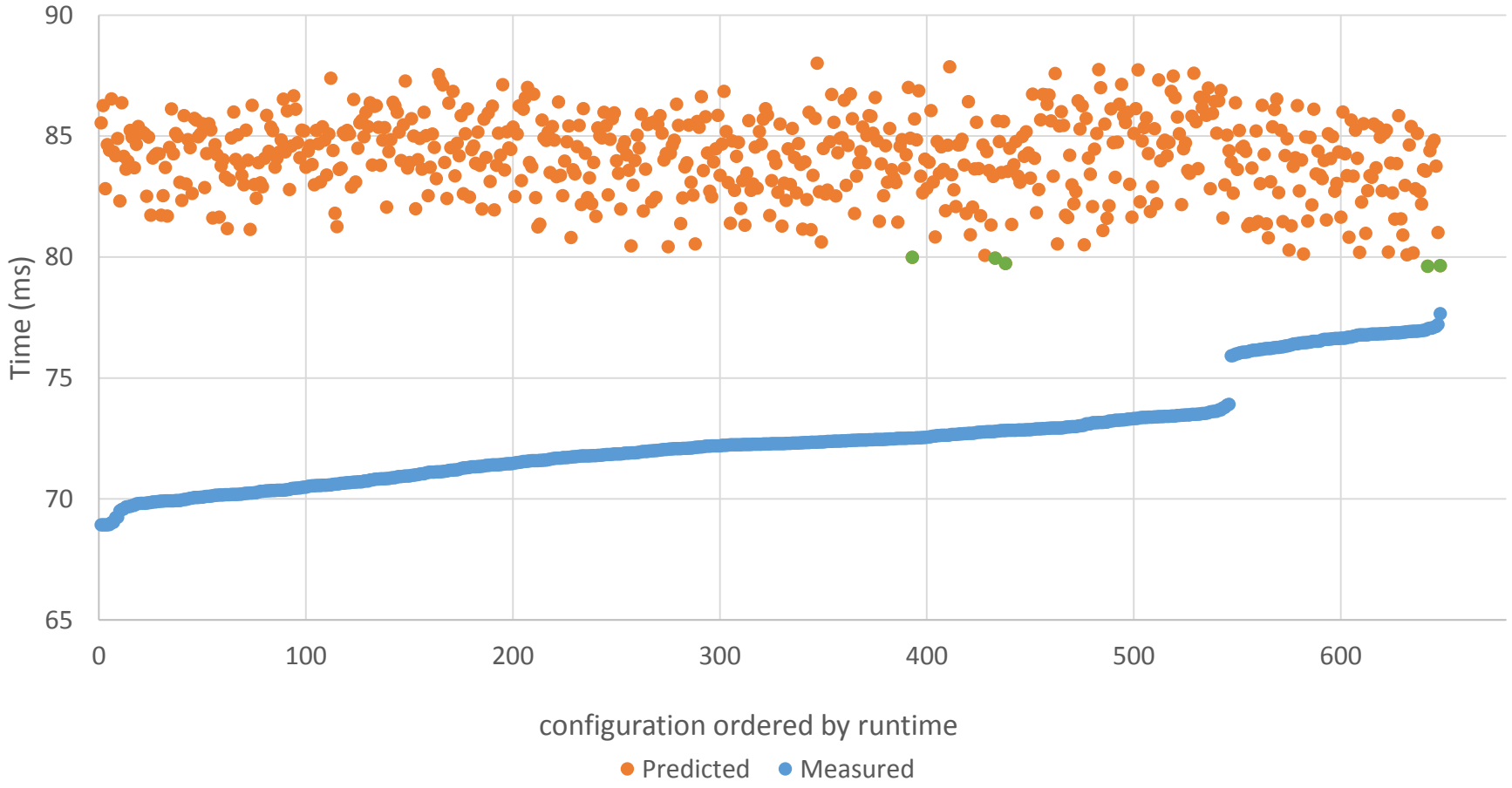


GCC
Graphics, Capture and Massively Parallel Computing

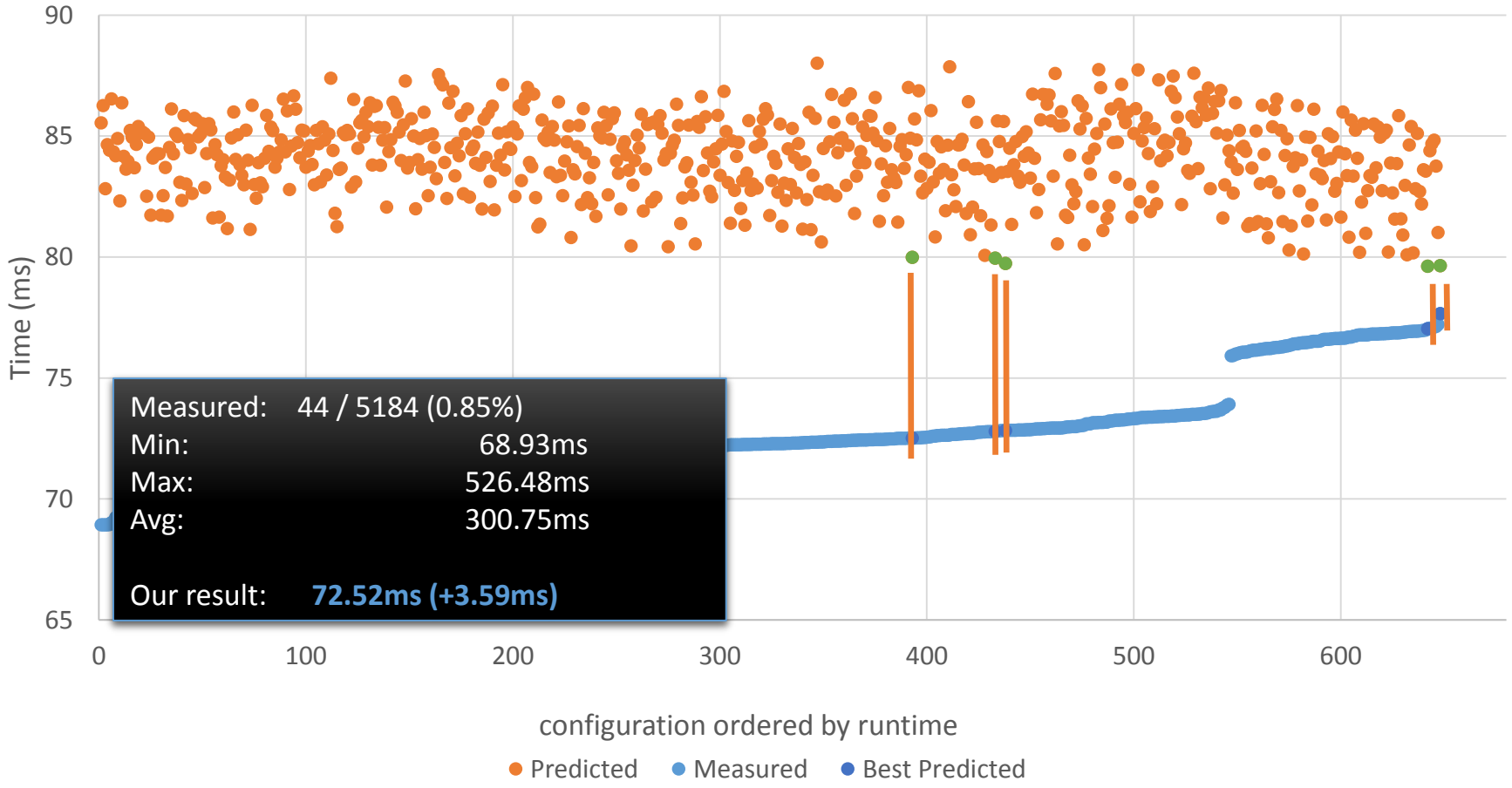
Real Example: Prediction (zoom in)



Real Example: Prediction (zoom in)



Real Example: Prediction (zoom in)



Algorithm can also be used to hand optimize code

TECHNISCHE UNIVERSITÄT DARMSTADT

- Mostly it suffices to optimize each data structure independently
- Only the value for L1 cache has to be optimized for all other optimizations

- Additional reading:
  Guided Profiling for Auto-Tuning Array Layouts on GPUs
  N.Weber, S. C. Amend and M. Goesele
  [http://tinyurl.com/matog]

How to apply the optimizations?

Future Work

TECHNISCHE UNIVERSITÄT DARMSTADT

- Coming soon...

MATOG

TECHNISCHE UNIVERSITÄT DARMSTADT

- Open Source (New BSD License)
- Code available: http://tinyurl.com/matog
- Documentation + Example code included

- Questions? → nicolas.weber@gris.tu-darmstadt.de

THE END

TECHNISCHE UNIVERSITÄT DARMSTADT