

Extra-P: Insightful Automatic Performance Modeling

Alexander Geiß¹, Gustavo de Moraes¹, Marcus Ritter¹,
Alexandru Calotoiu², Torsten Hoefler², and Felix Wolf¹



TECHNISCHE
UNIVERSITÄT
DARMSTADT

ETH zürich

¹ TU Darmstadt , ² ETH Zürich

Introduction

Extra-P

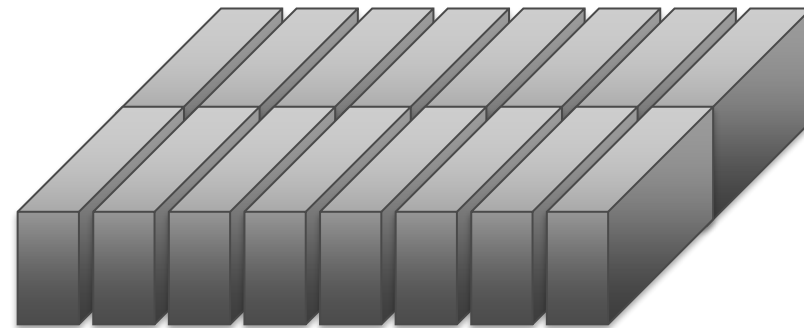
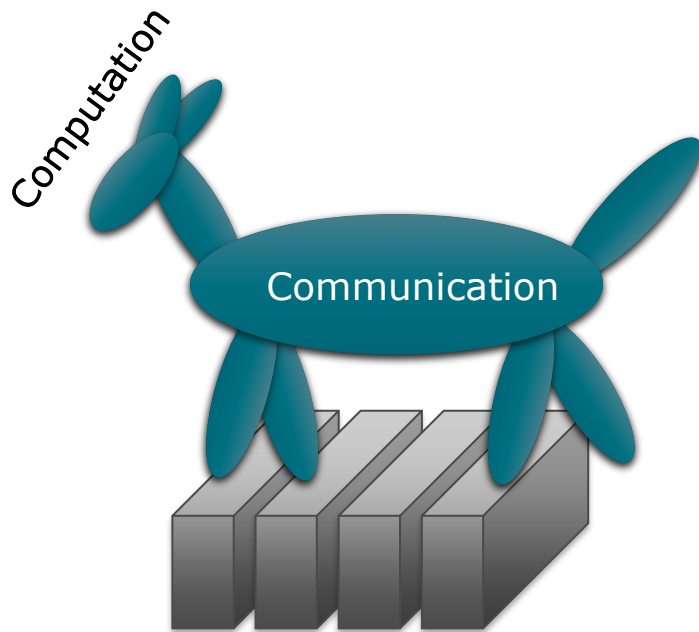


Watch Extra-P
overview video

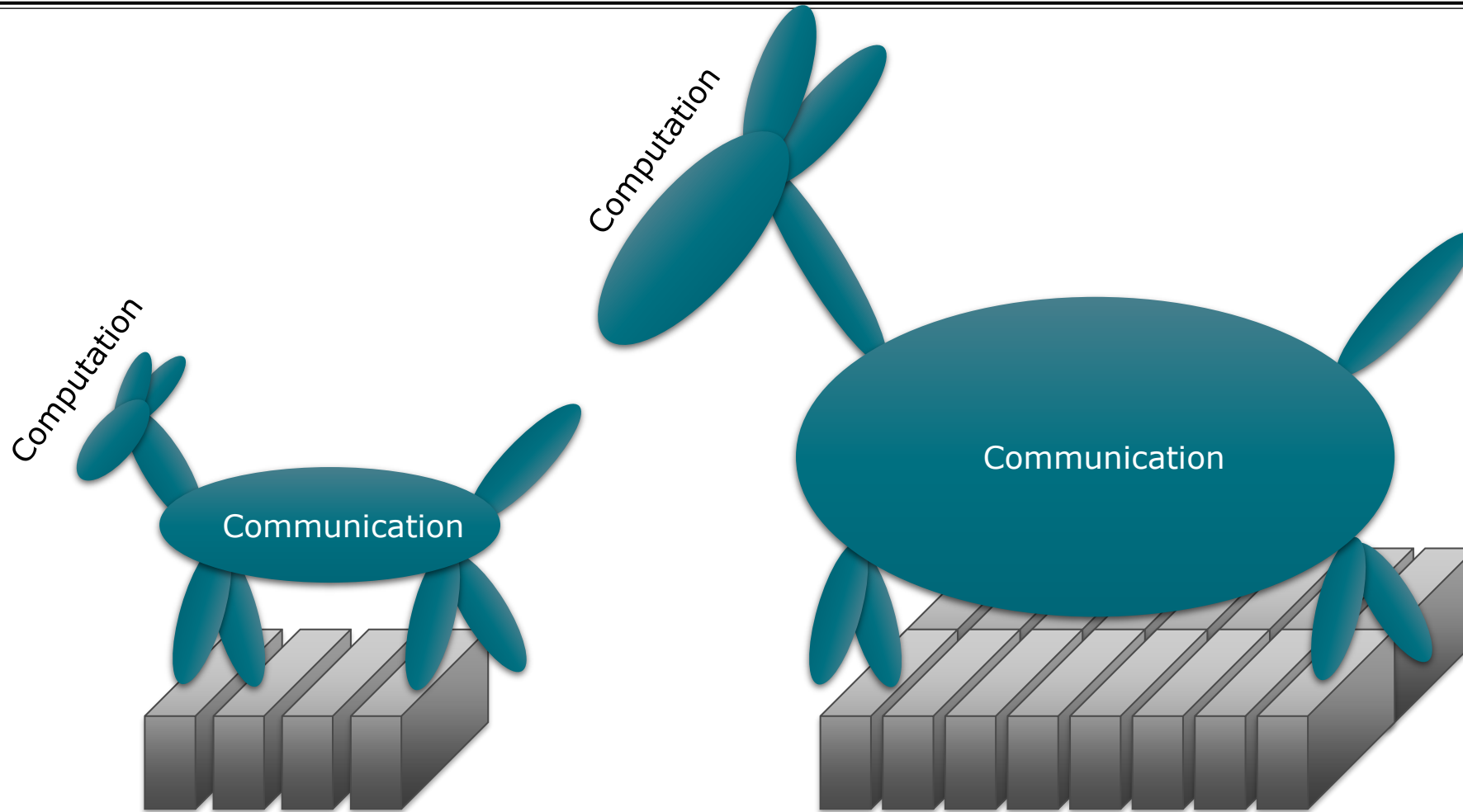


[https://www.youtube.com/
watch?v=Cv2YRCMWqBM](https://www.youtube.com/watch?v=Cv2YRCMWqBM)

Motivation - latent scalability bugs

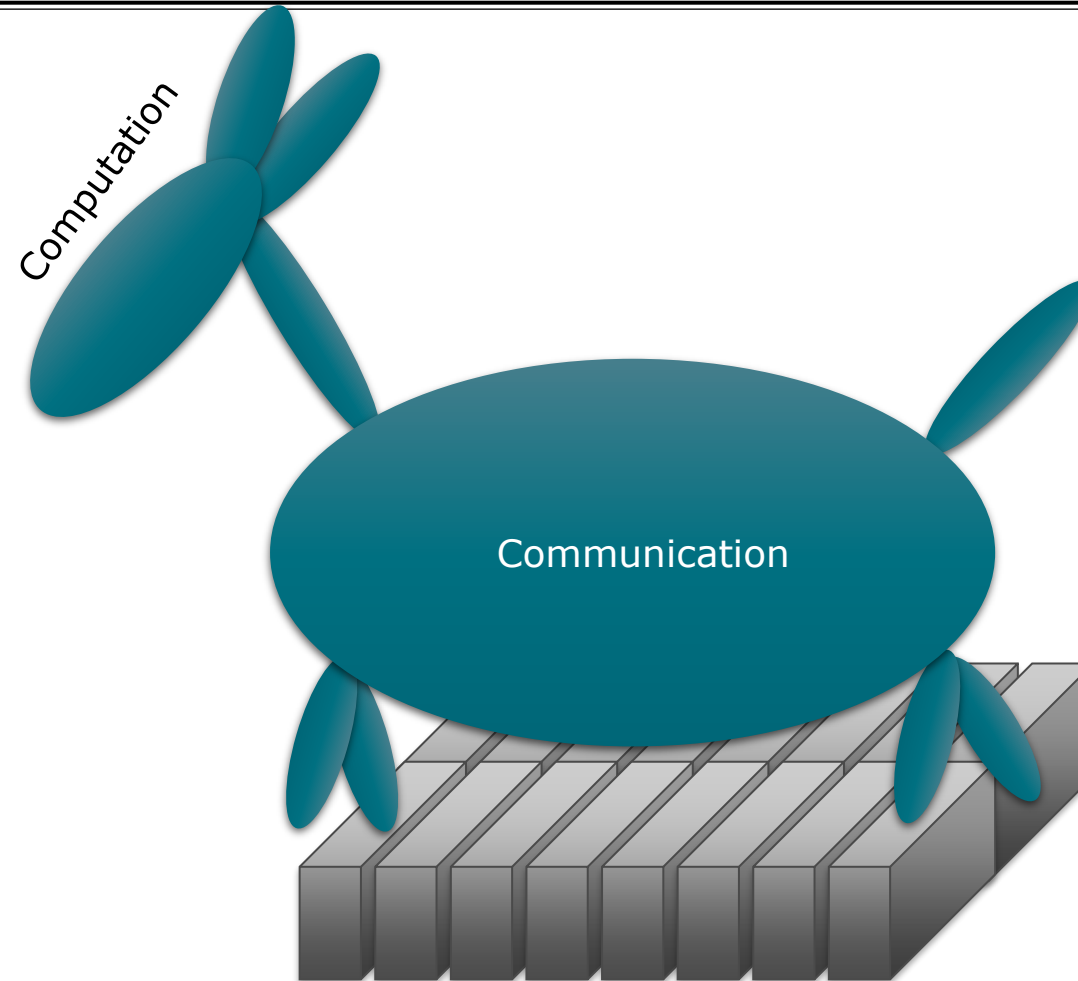


Scaling code to a bigger machine can unveil unpleasant surprises



Scaling code to a bigger machine can unveil unpleasant surprises

We need to find scaling issues before they occur



Spectrum of performance analysis methods

Benchmark

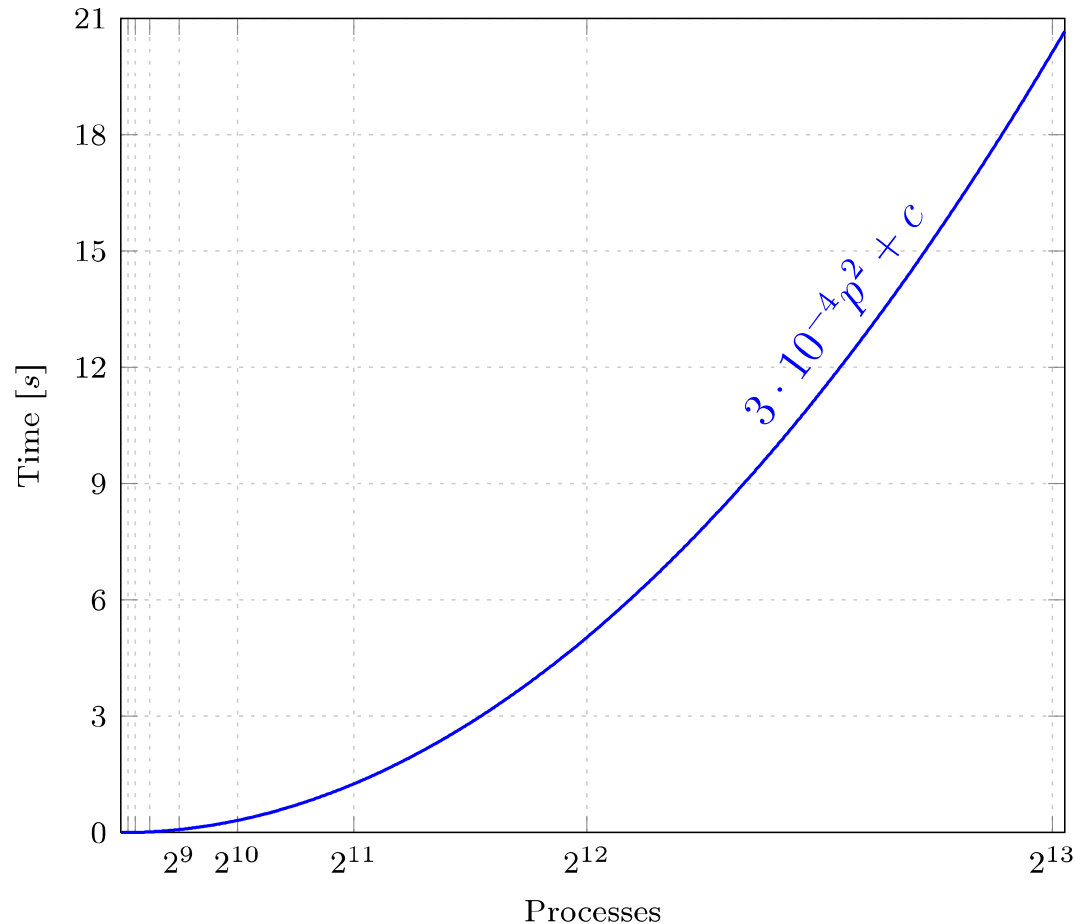
Full simulation

Model simulation

Model



Scaling model



- Represents performance metric as a function of the number of processes
- Provides insight into the program behavior at scale

Analytical performance modeling

Identify kernels

- Parts of the program that dominate its performance at larger scales
- Identified via small-scale tests and intuition

Create models

- Laborious process
- Still confined to a small community of skilled experts

Disadvantages:

- Time consuming
- Danger of overlooking unscalable code



Hoisie et al.: *Performance and scalability analysis of teraflop-scale parallel architectures using multi-dimensional wavefront applications*. International Journal of High Performance Computing Applications, 2000

Bauer et al.: *Analysis of the MILC Lattice QCD Application su3_rmd*. CCGrid, 2012

Automatic performance modeling

```
main() {  
  foo()  
  bar()  
  compute()  
}
```

Input

Output

Human-readable
performance models
of all functions
(e.g., $t(p) = c_1 \cdot \log(p) + c_2$)

Instrumentation

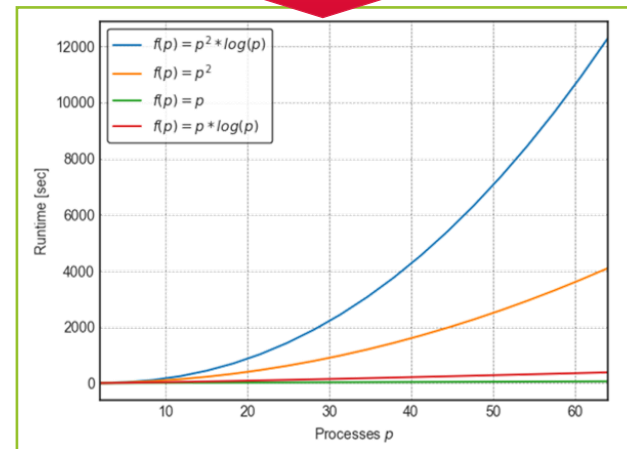
- All functions

Performance measurements

M_i

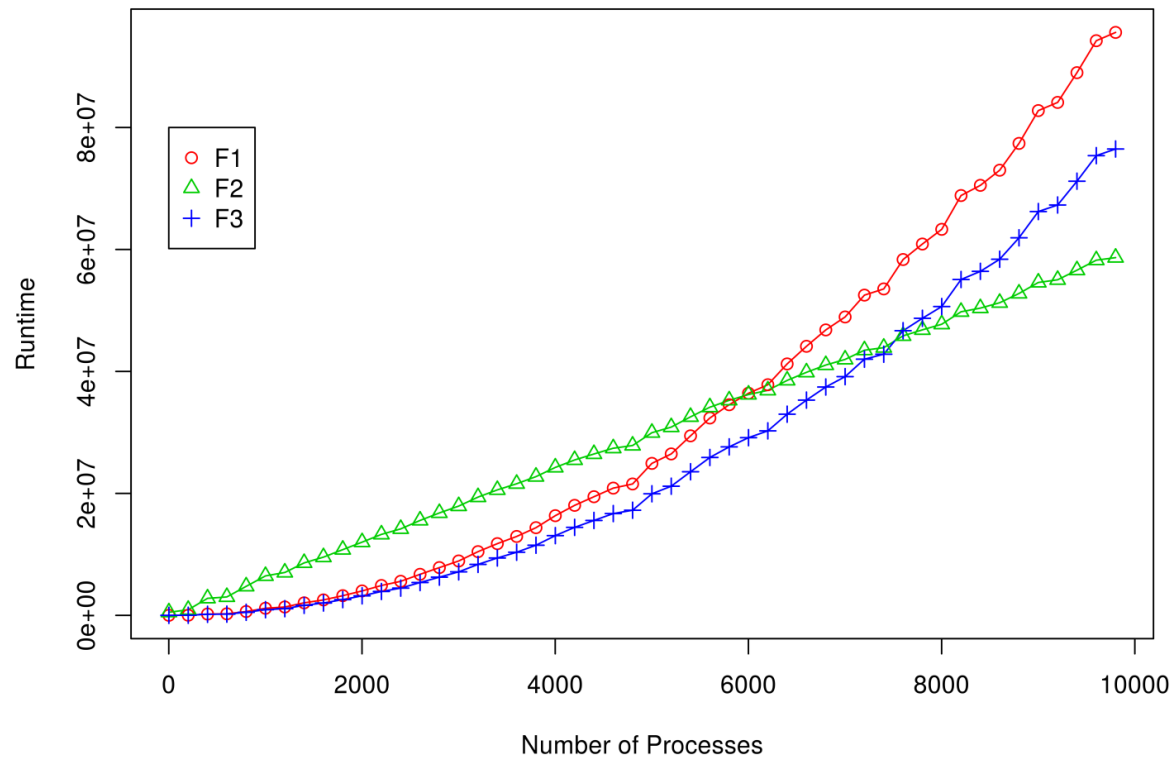
M_j

Extra-P

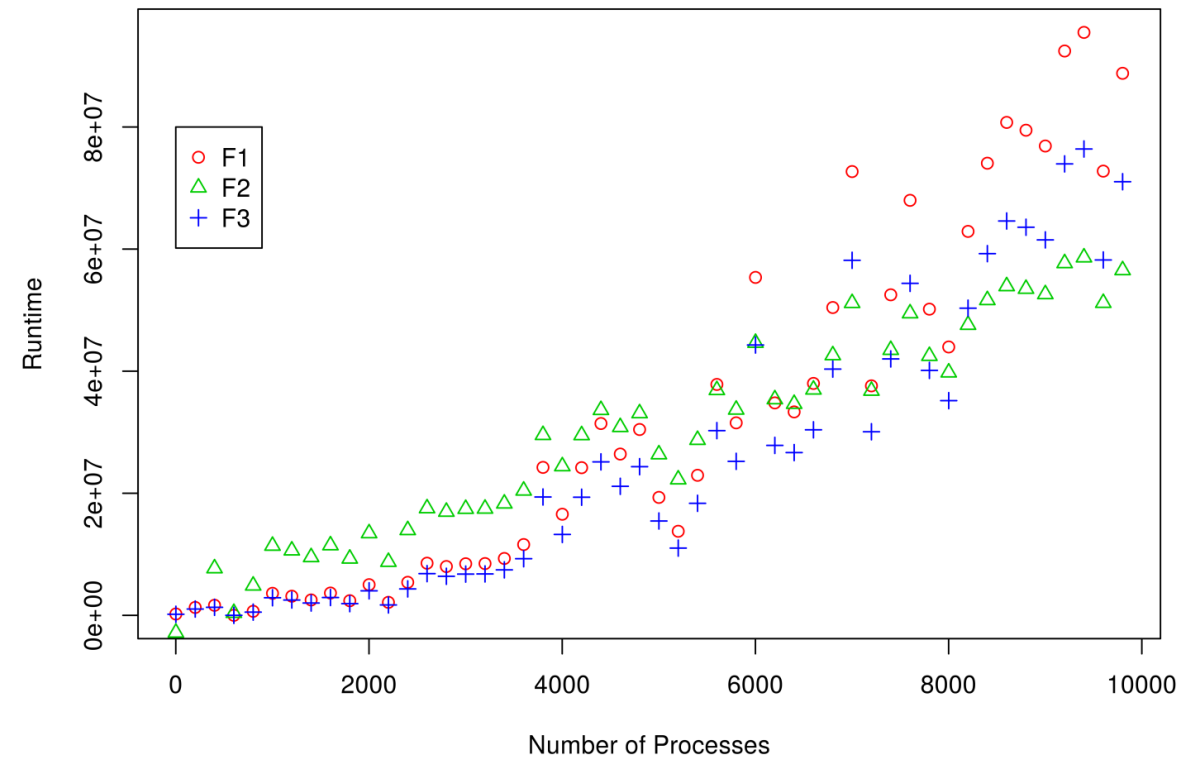


Primary focus on scaling trend

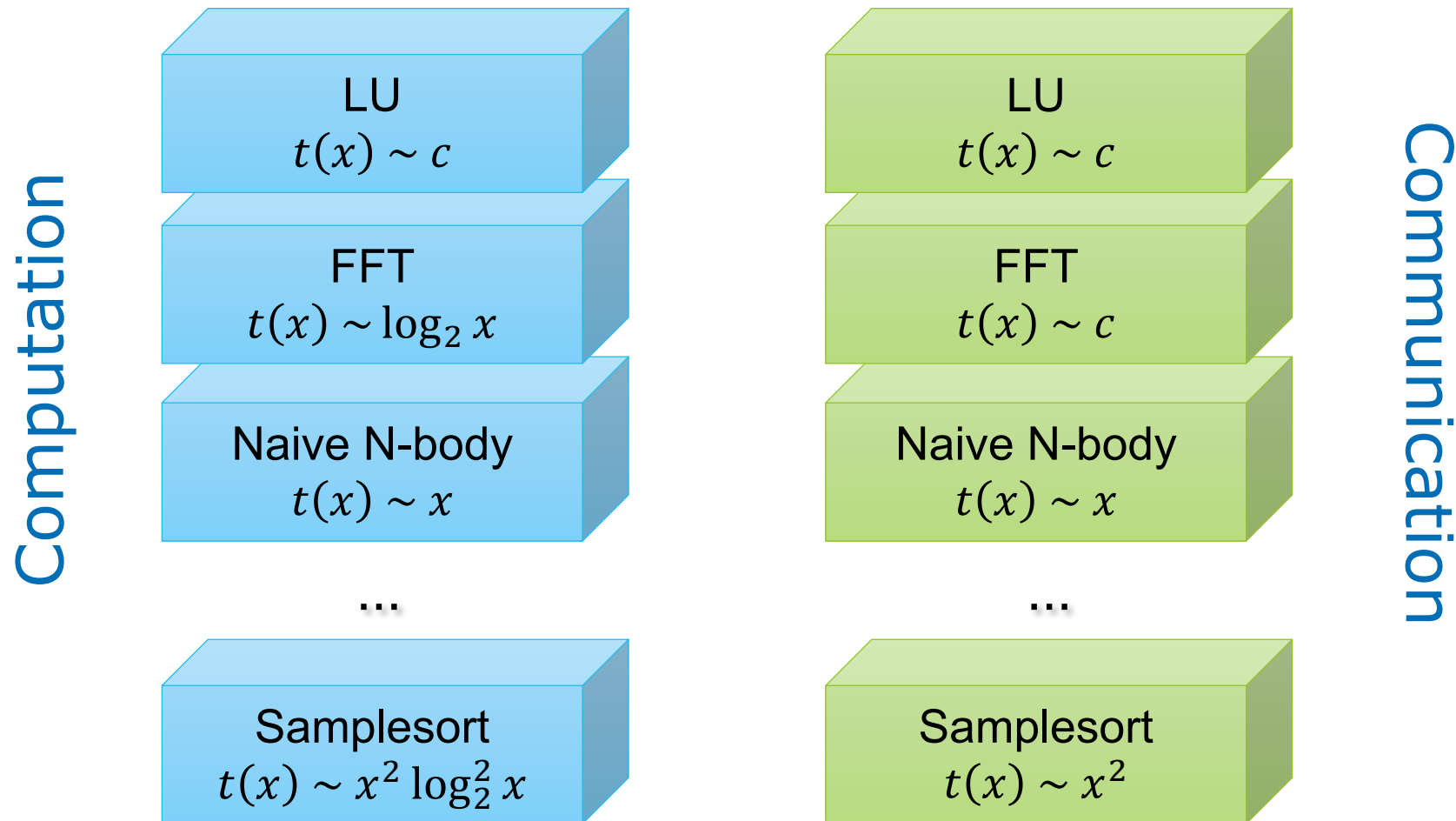
Common performance analysis chart in a paper



Production reality



Model building blocks



Performance model normal form

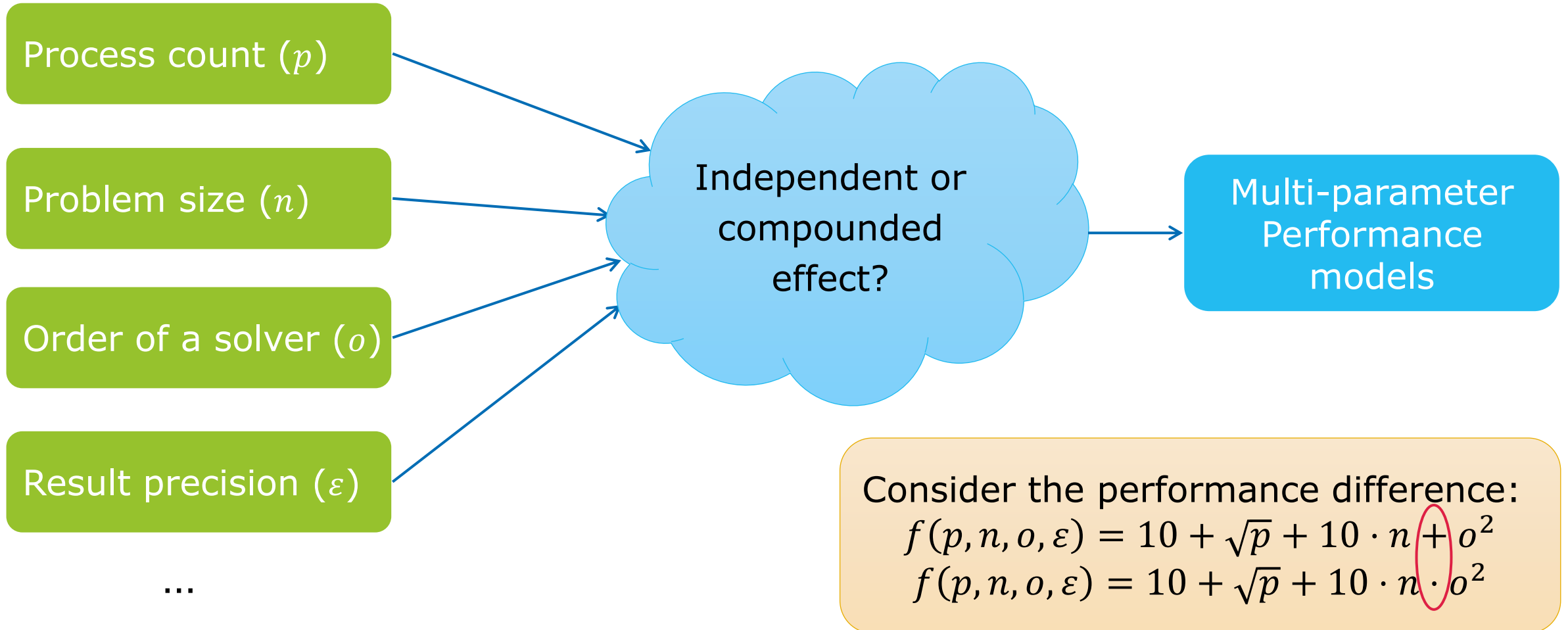
$$f(x) = \sum_{k=1}^n c_k \cdot x^{i_k} \cdot \log_2^{j_k}(x)$$

$$\begin{aligned} n &\in \mathbb{N} \\ i_k &\in I \\ j_k &\in J \\ I, J &\subset \mathbb{Q} \end{aligned}$$

$$\begin{aligned} n &= 1 \\ I &= \{0, 1, 2\} \\ J &= \{0, 1\} \end{aligned}$$

$$\begin{array}{ll} c_1 & c_1 \cdot \log x \\ c_1 \cdot x & c_1 \cdot x \cdot \log x \\ c_1 \cdot x^2 & c_1 \cdot x^2 \cdot \log x \end{array}$$

Fast multi-parameter performance modeling



Fast multi-parameter performance modeling

- Expanded performance model normal form

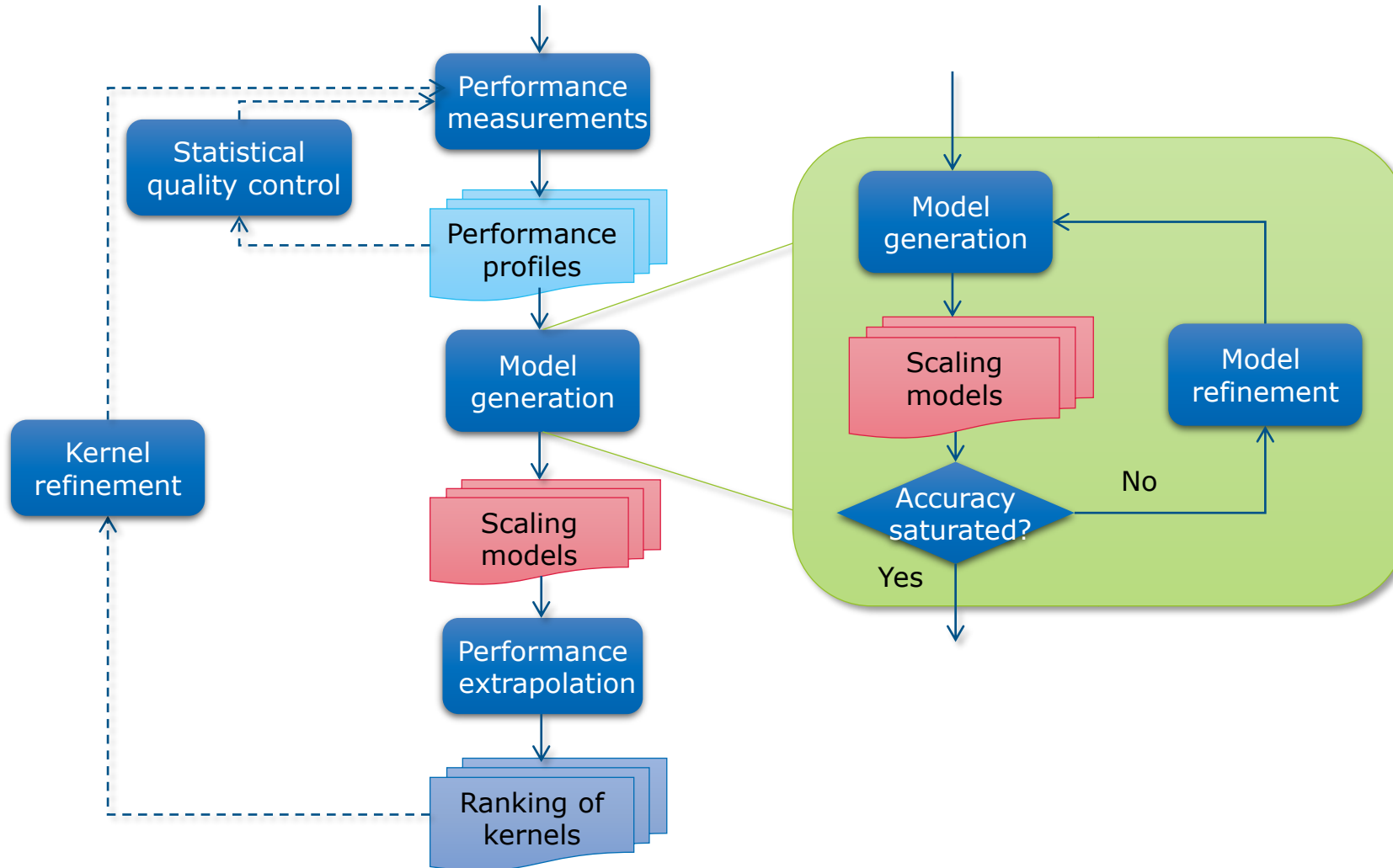
$$f(x_1, \dots, x_m) = \sum_{k=1}^n c_k \prod_{l=1}^m x_l^{i_{kl}} \cdot \log_2^{j_{kl}}(x_l)$$

$$\begin{aligned} m, n &\in \mathbb{N} \\ i_k &\in I \\ j_k &\in J \\ I, J &\subset \mathbb{Q} \end{aligned}$$

Model candidates

- Constant c_1
- Single parameter $c_1 + c_2 \cdot x_1$
- Multiple parameters \dots
 - Additive $c_1 + c_2 \cdot x_1 + c_3 \cdot x_2$
 - Multiplicative $c_1 + c_2 \cdot x_1 \cdot x_2$
 - Complex $c_1 + c_2 \cdot x_1 \cdot x_2 + c_3 \cdot \log x_2 \cdot x_2^3$

Workflow

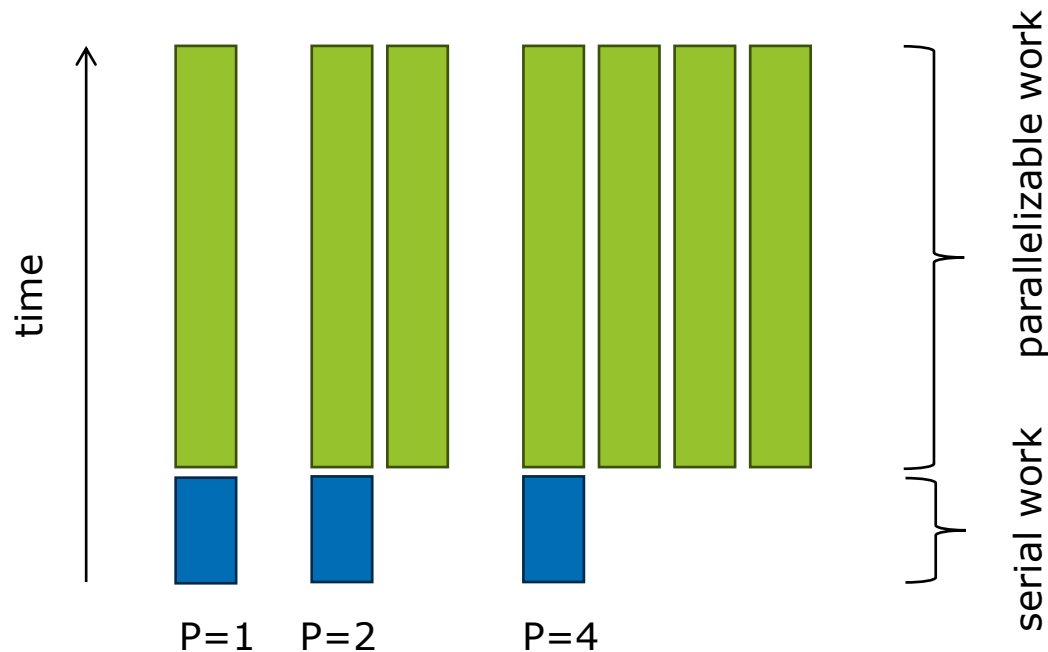


Scaling analysis: number of processes is increased

Preferred by
Extra-P

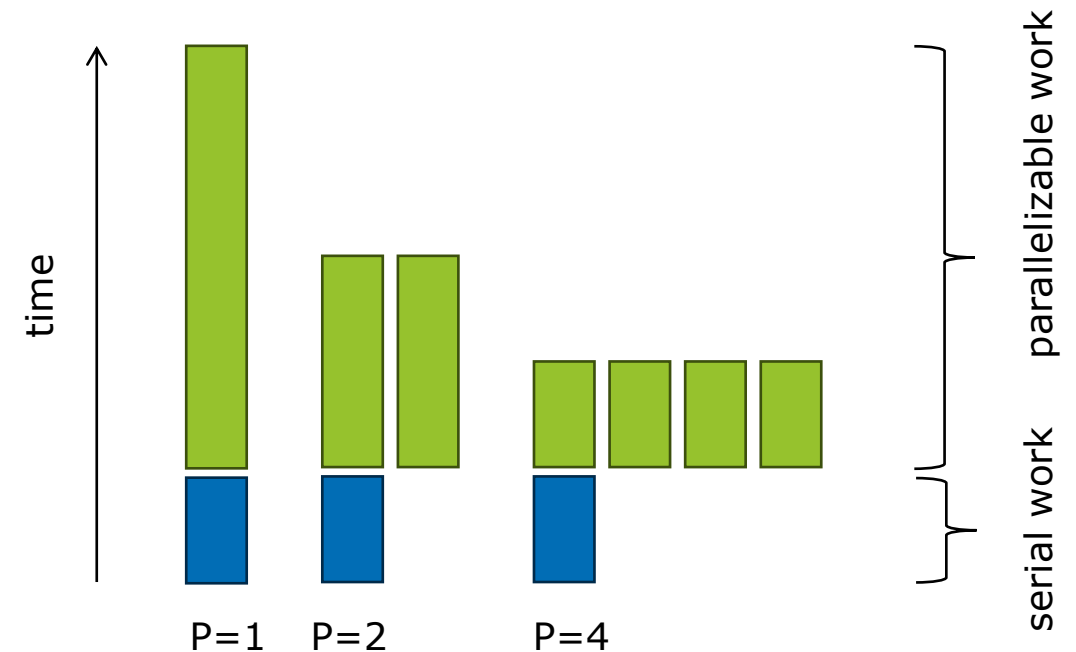
Weak scaling

- The problem size is increased alongside
- Law of Gustafson



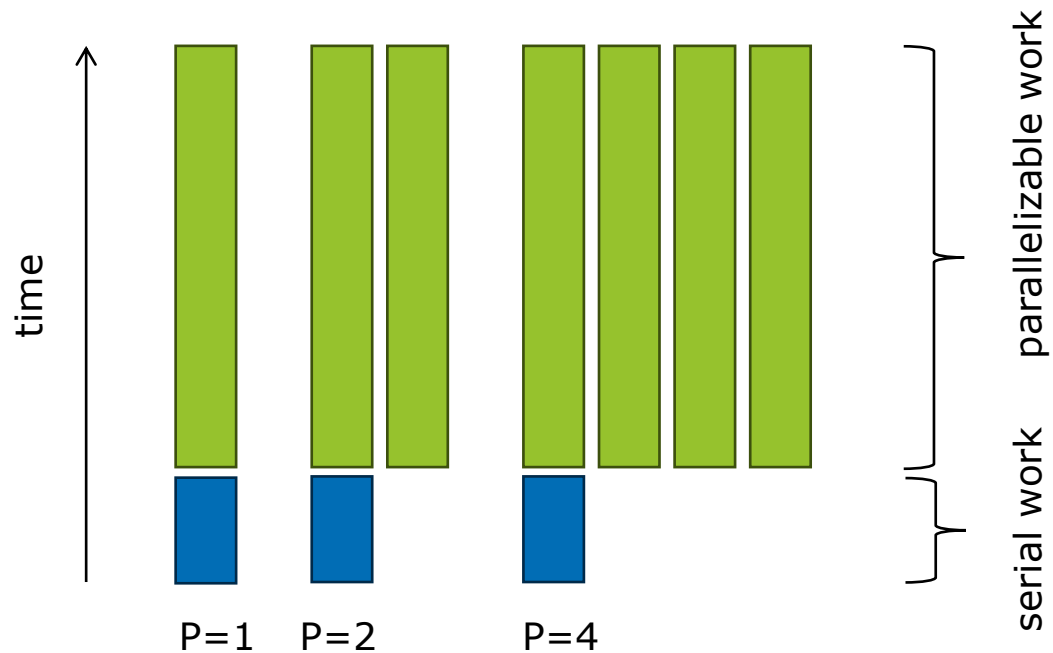
Strong scaling

- The problem size remains constant
- Amdahl's law

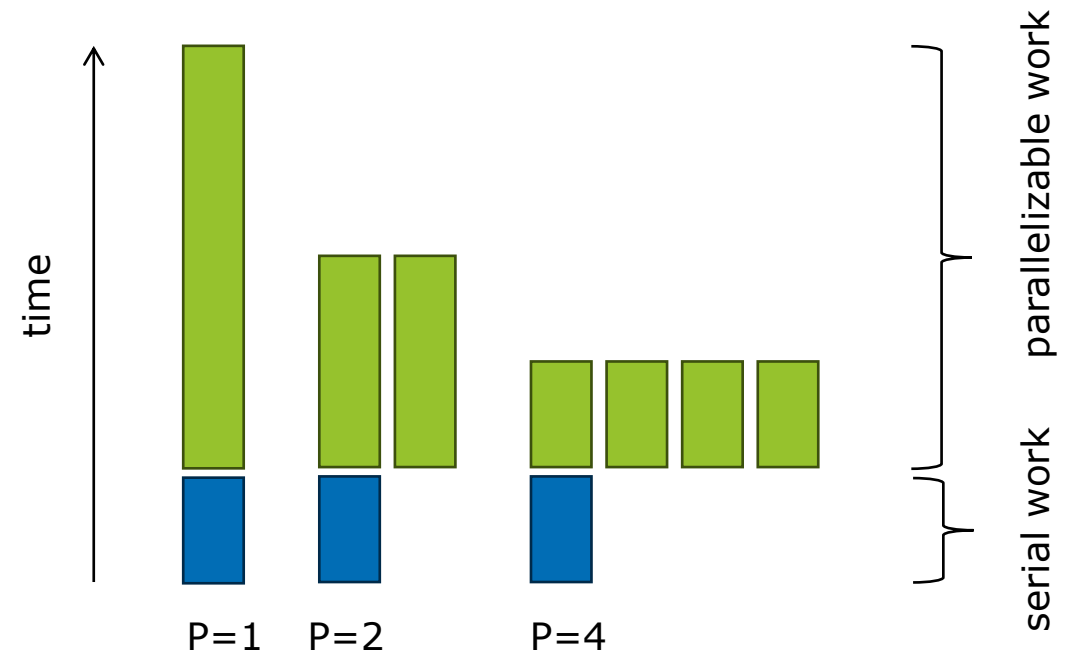


Scaling analysis with Extra-P

Weak scaling



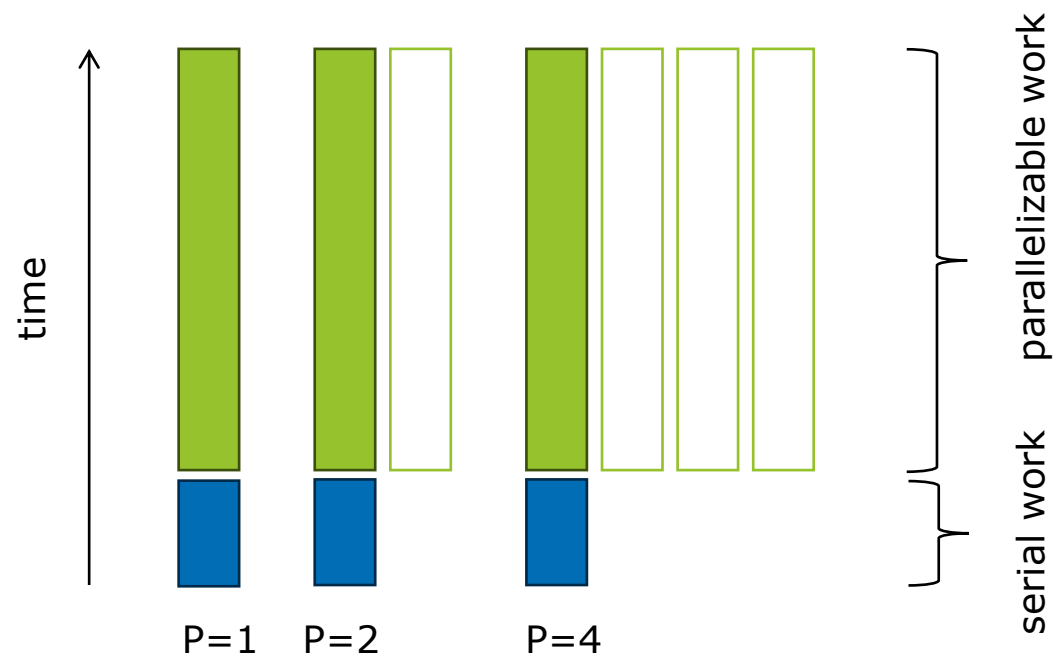
Strong scaling



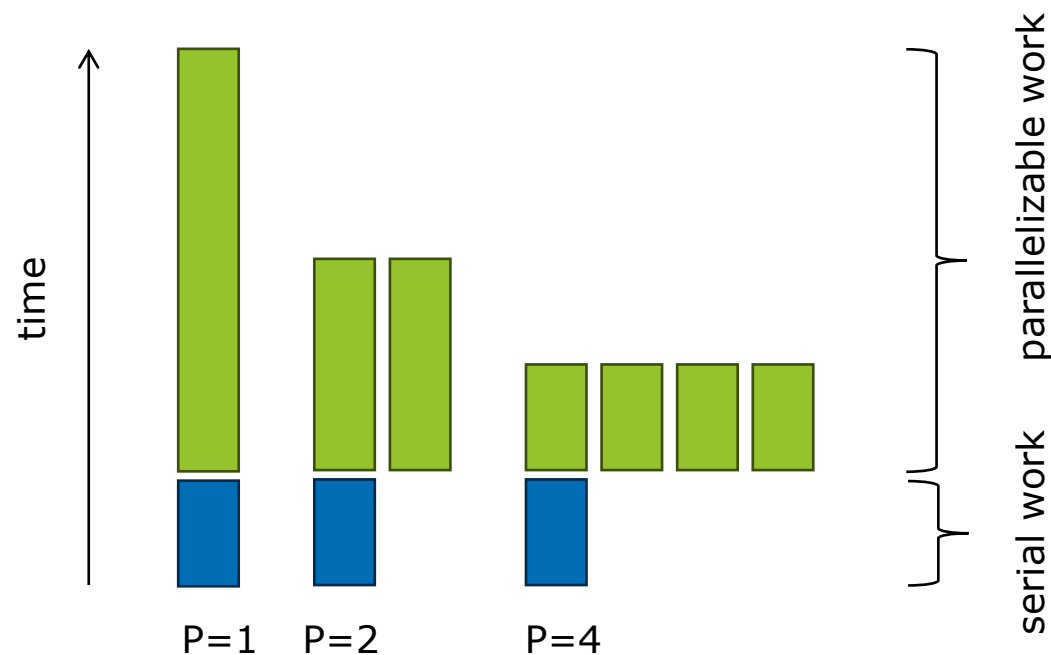
Scaling analysis with Extra-P

Weak scaling

- Extra-P models the runtime of one process



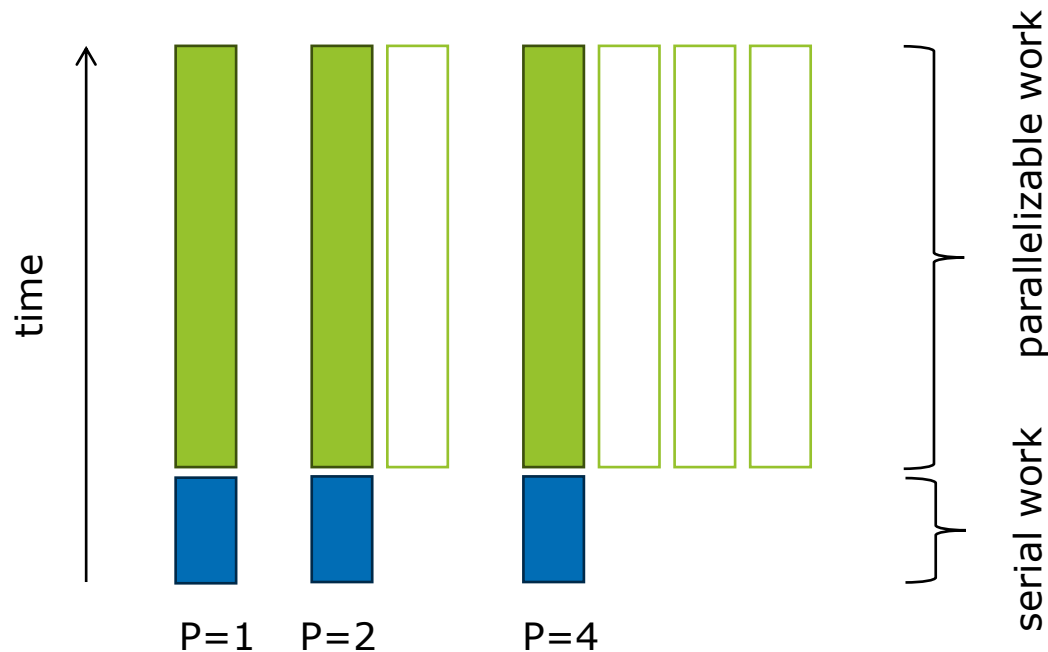
Strong scaling



Scaling analysis with Extra-P

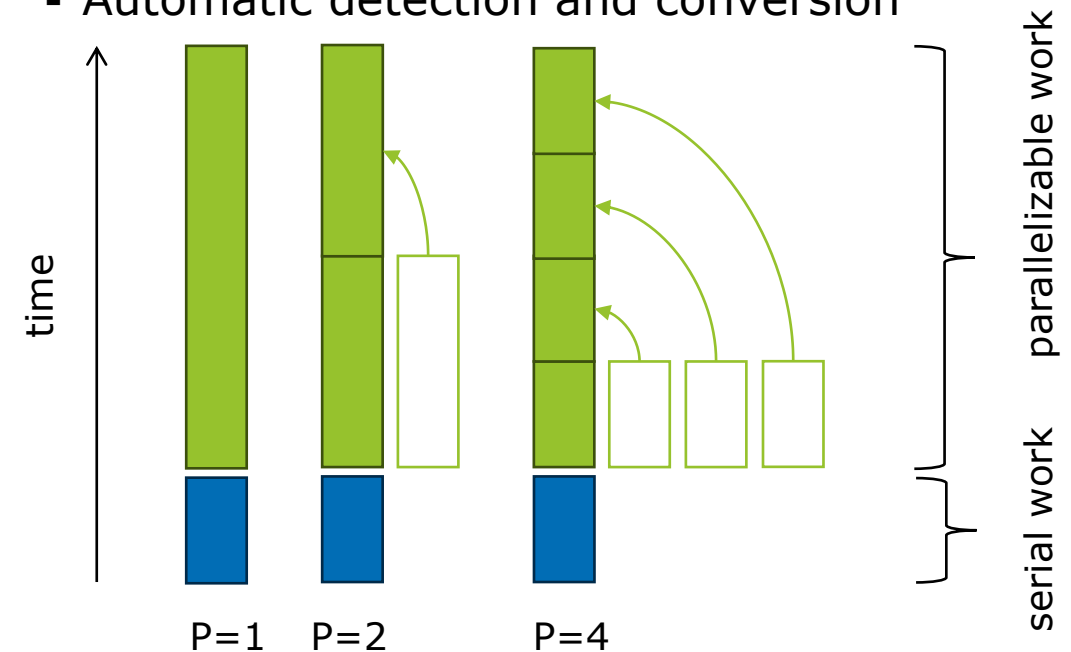
Weak scaling

- Extra-P models the runtime of one process



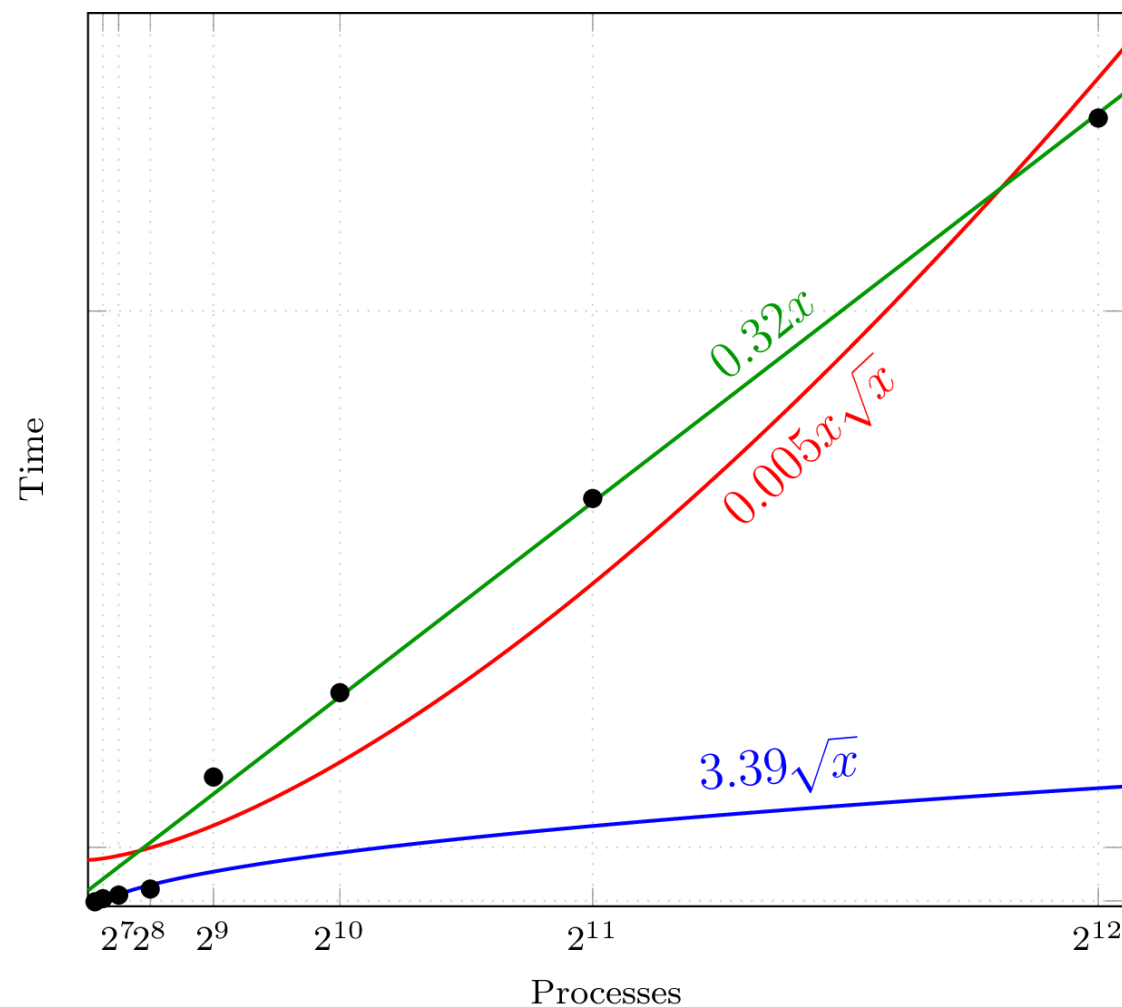
Strong scaling

- Extra-P models the resource consumption
 - Runtime of all processes combined
 - Equivalent to the number of core-hours
 - Automatic detection and conversion



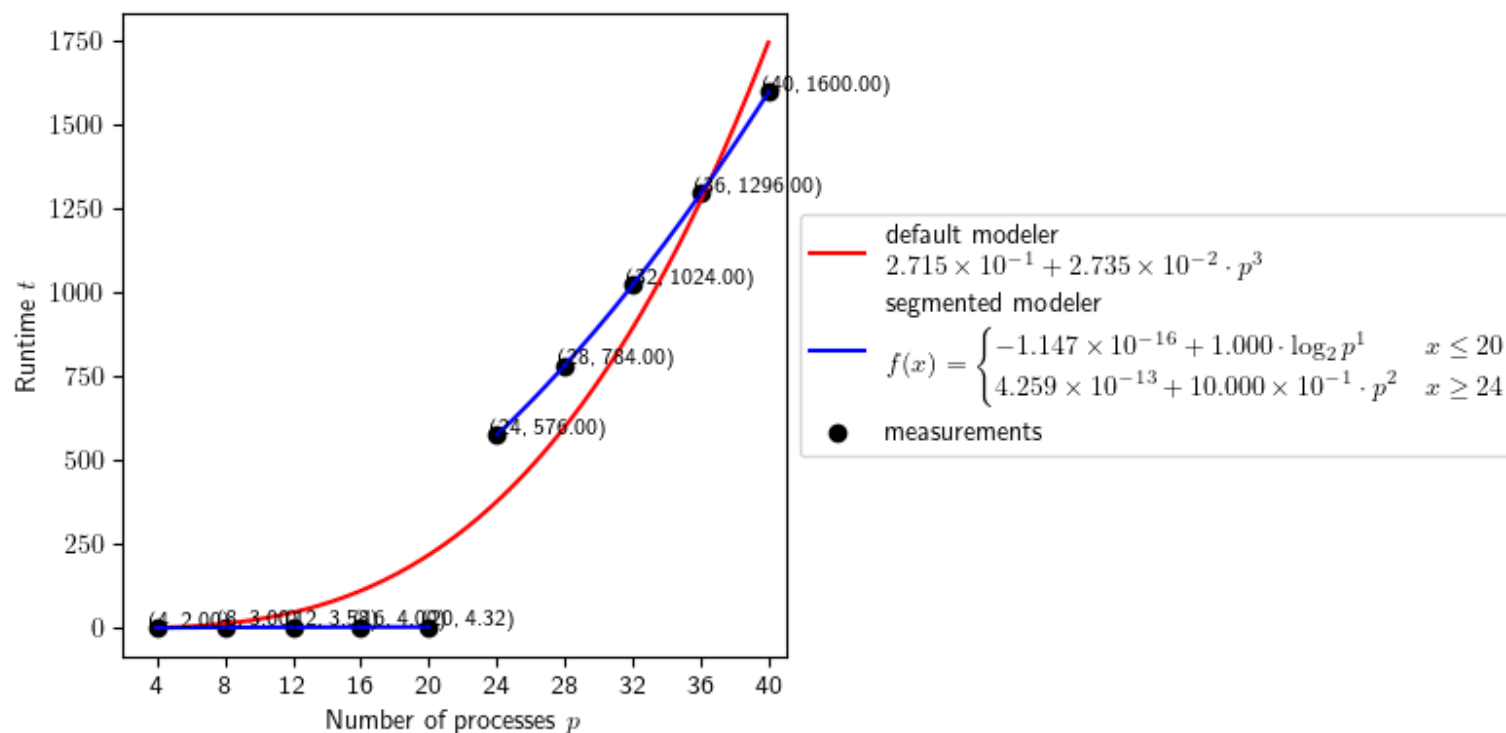
Assumptions & limitations

- Scaling behavior expressible with performance model normal form
- Only one scaling behavior for all the measurements; no jumps
- Some MPI collective operations switch their algorithm
 - results in bad models
- Example: **red model** tries to model measurements of different algorithms
 - First 4 points – one function
 - Last 4 points – another function (linear)
 - Adj. R2 = 0.95085 (!)



Segmented models

- We can detect and model segmented behavior
- When enough measurement points are present
- Segmented modeler must be manually selected
- Limited to two segments



Performance measurements

Different ways of collecting measurements

- Score-P (<http://www.vi-hps.org/projects/score-p/>)
- Other profiling tools, e.g. HPCToolkit
- Manual ad-hoc measurements



Performance measurements (2)

- At least 5 different measurements recommended

Performance measurements (profiles)

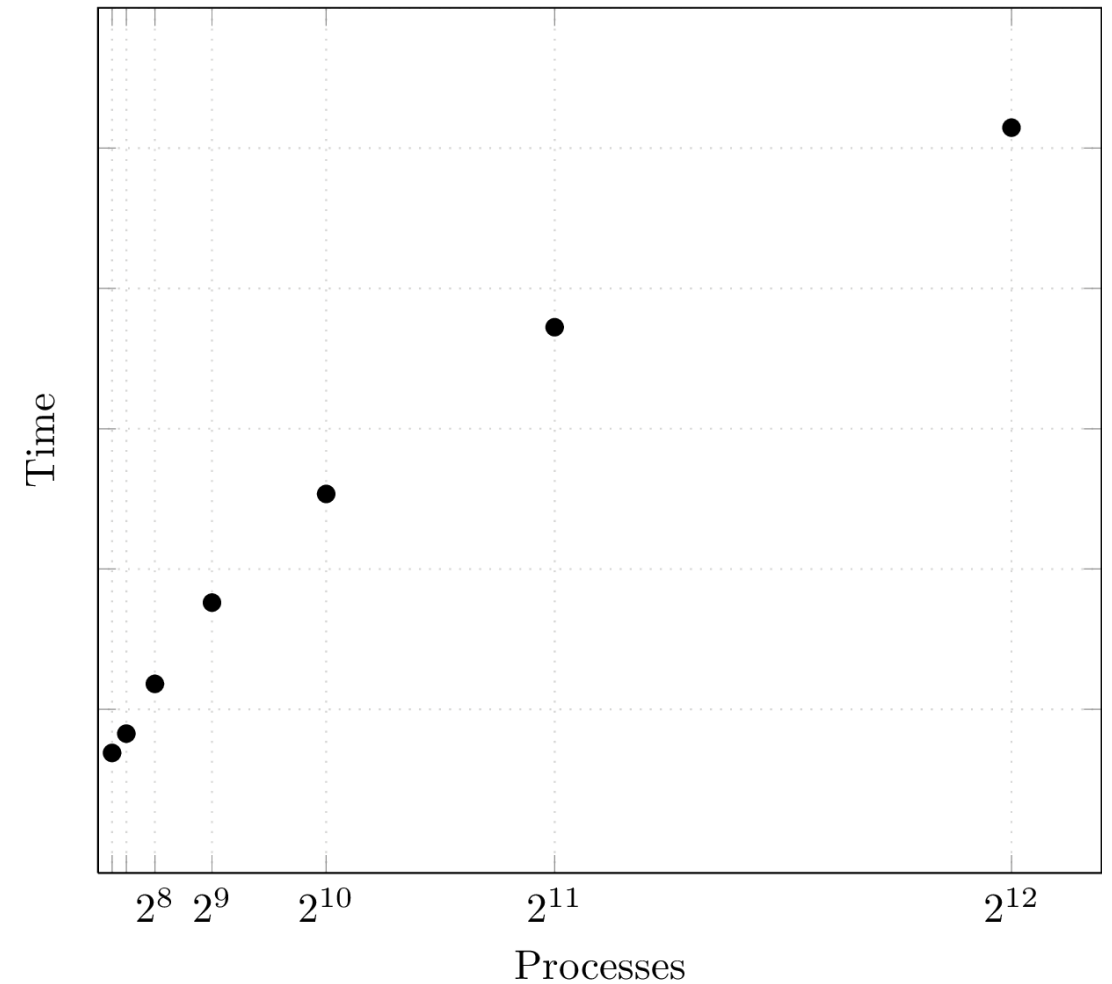
$$p_1 = 256$$

$$p_2 = 512$$

$$p_3 = 1024$$

$$p_4 = 2048$$

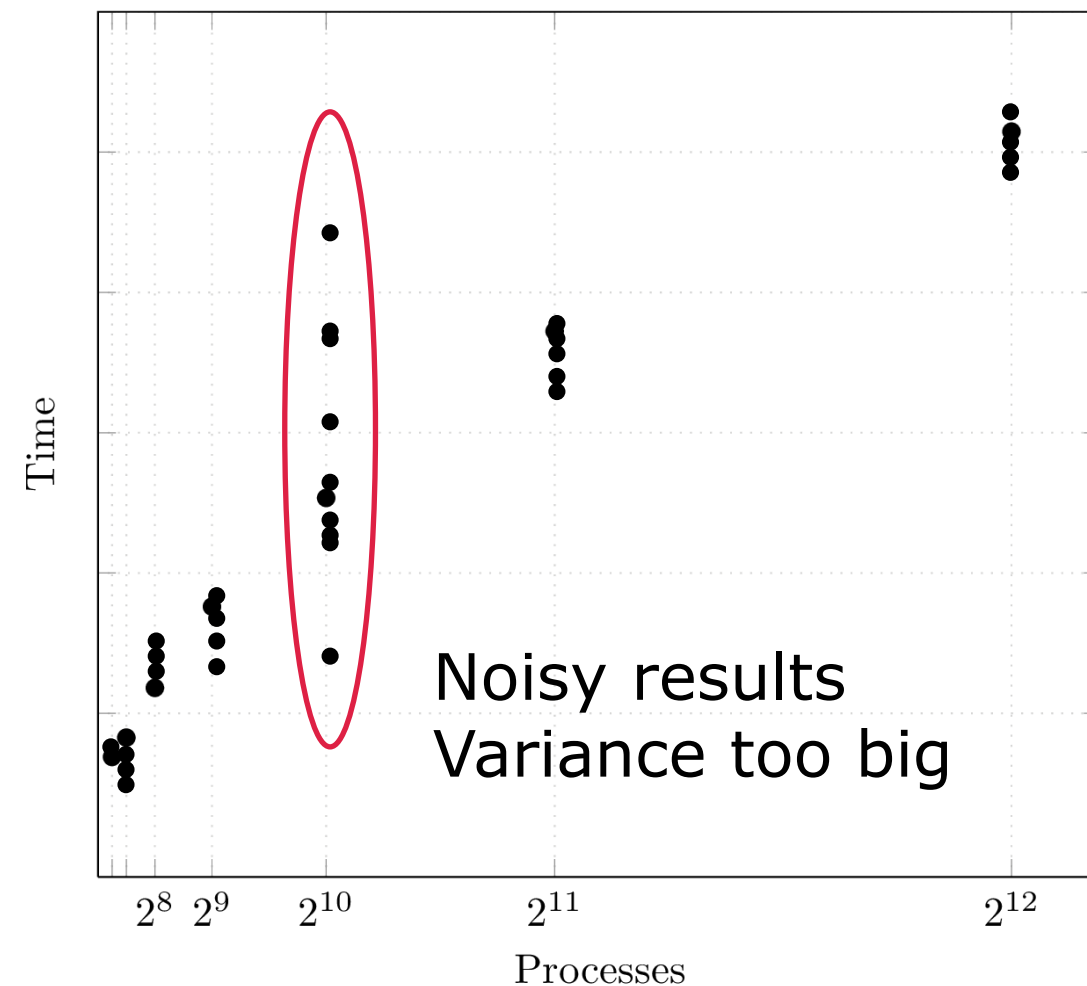
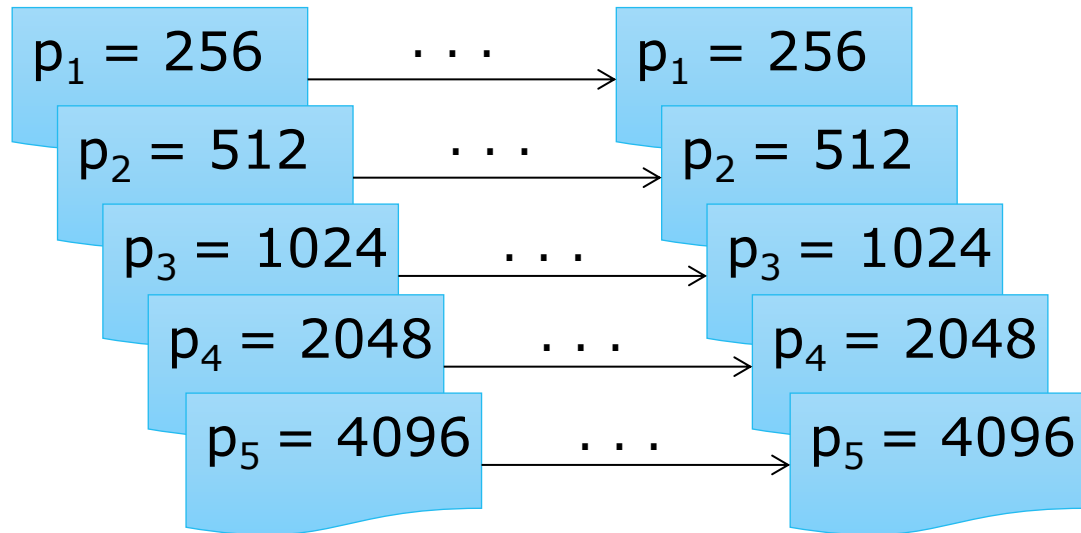
$$p_5 = 4096$$



Performance measurements (3)

- At least 5 different measurements recommended
- Each measurement repeated multiple times

Performance measurements (profiles)



Adjusted R^2

- R^2 represents how well the determined function fits the M available measurements
- Adjusted R^2 adjusts for N , the number of terms used
 - Adj. R^2 decreases \rightarrow more useless variables
 - Adj. R^2 increases \rightarrow more useful variables
- Rule of thumb: adj. $R^2 > 0.95$

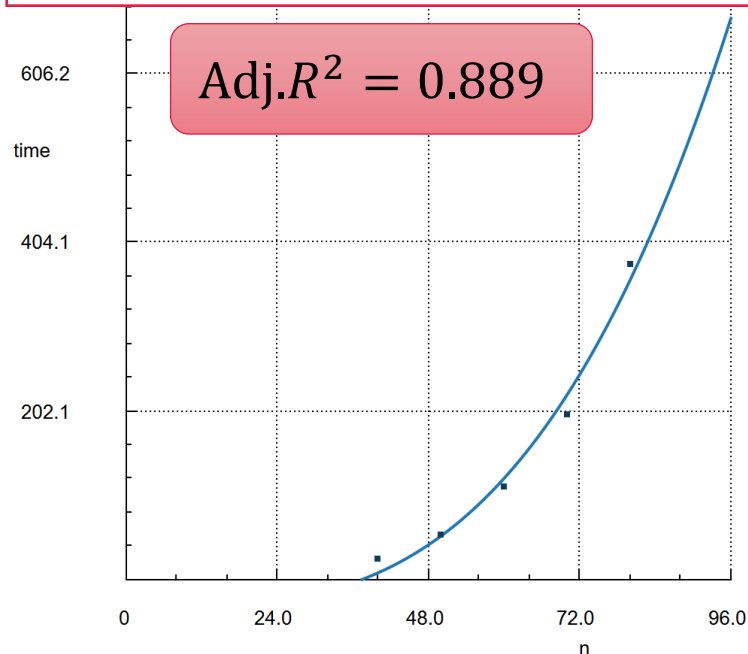
$$R^2 = 1 - \frac{\text{residualSumSquares}}{\text{totalSumSquares}}$$

$$\overline{R^2} = 1 - (1 - R^2) \cdot \frac{M - 1}{M - N - 2}$$

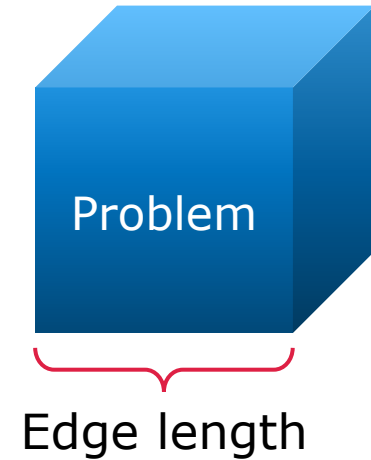
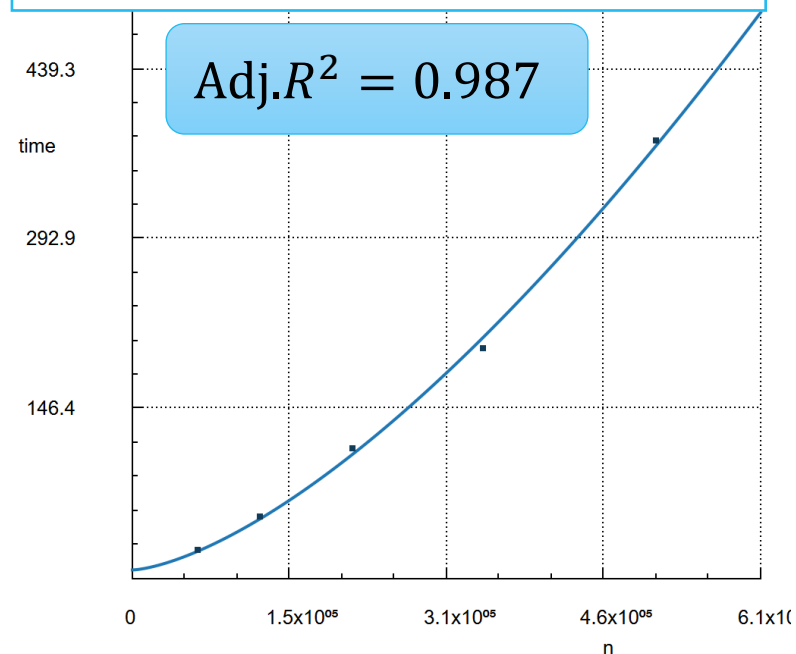
Quadratic and cubic problems

- The whole problem size should be used as parameter
 - Not just the edge length

Edge length: n
 $-32.98 + 0.000121 * n^3 * \log_2(n)$

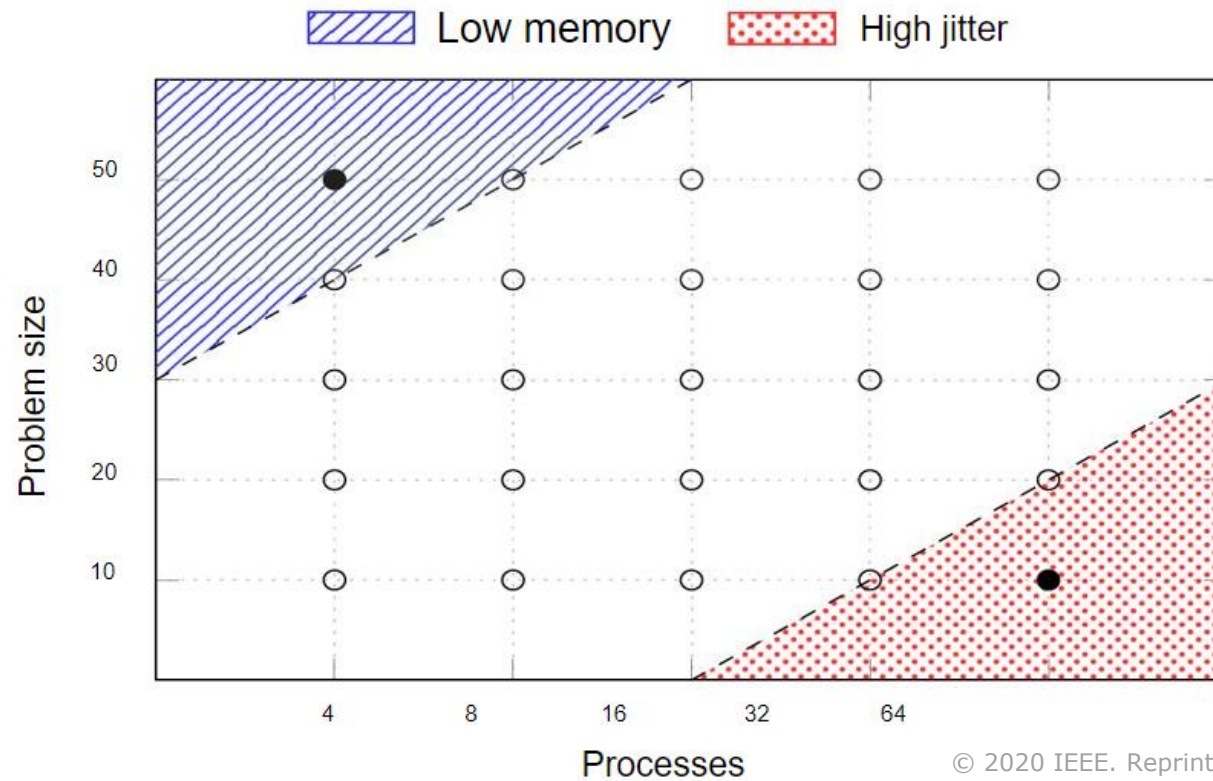


Volume: n
 $7.53 + 9.98 \cdot 10^{-7} * n^{1.5}$



Sparse modeling

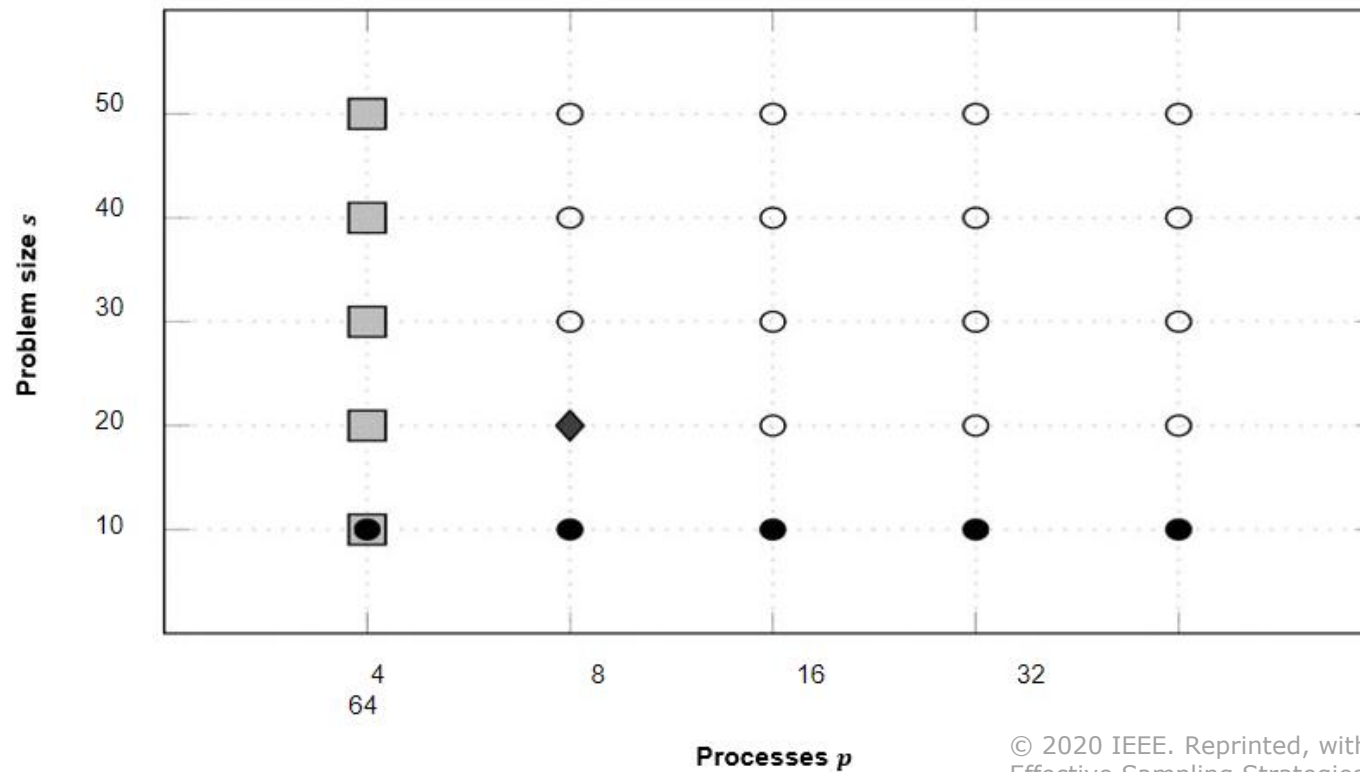
- Experiments can be expensive
- So far we needed 5×5^m experiments, m =number of parameters



© 2020 IEEE. Reprinted, with permission, from M. Ritter et al. "Learning Cost-Effective Sampling Strategies for Empirical Performance Modeling," IPDPS 2020.

Sparse modeling

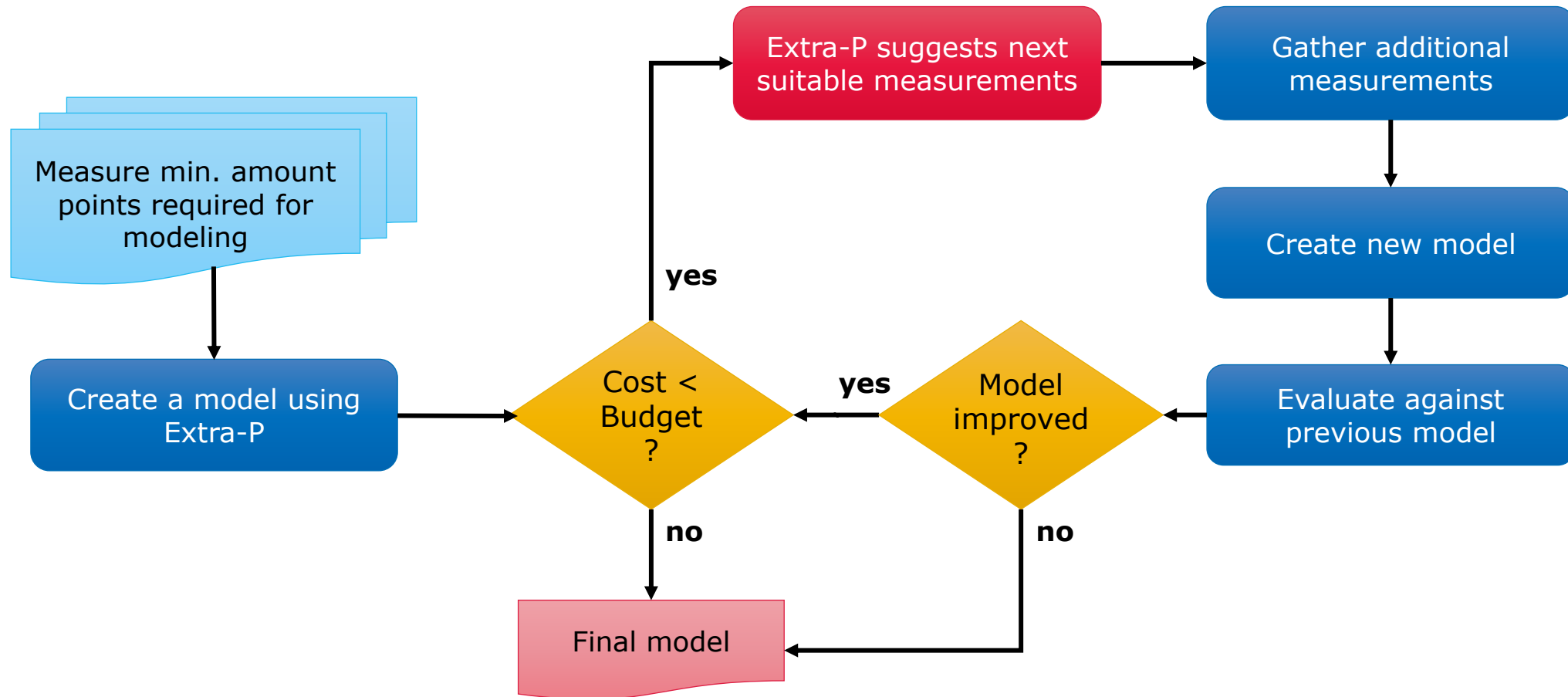
- Using our new sparse modeling approach we can model with less points!
- We only need $5 \times 5 \cdot m$ experiments to start, m =number of parameters



© 2020 IEEE. Reprinted, with permission, from M. Ritter et al. "Learning Cost-Effective Sampling Strategies for Empirical Performance Modeling," IPDPS 2020.

Sparse modeling with measurement point suggestions

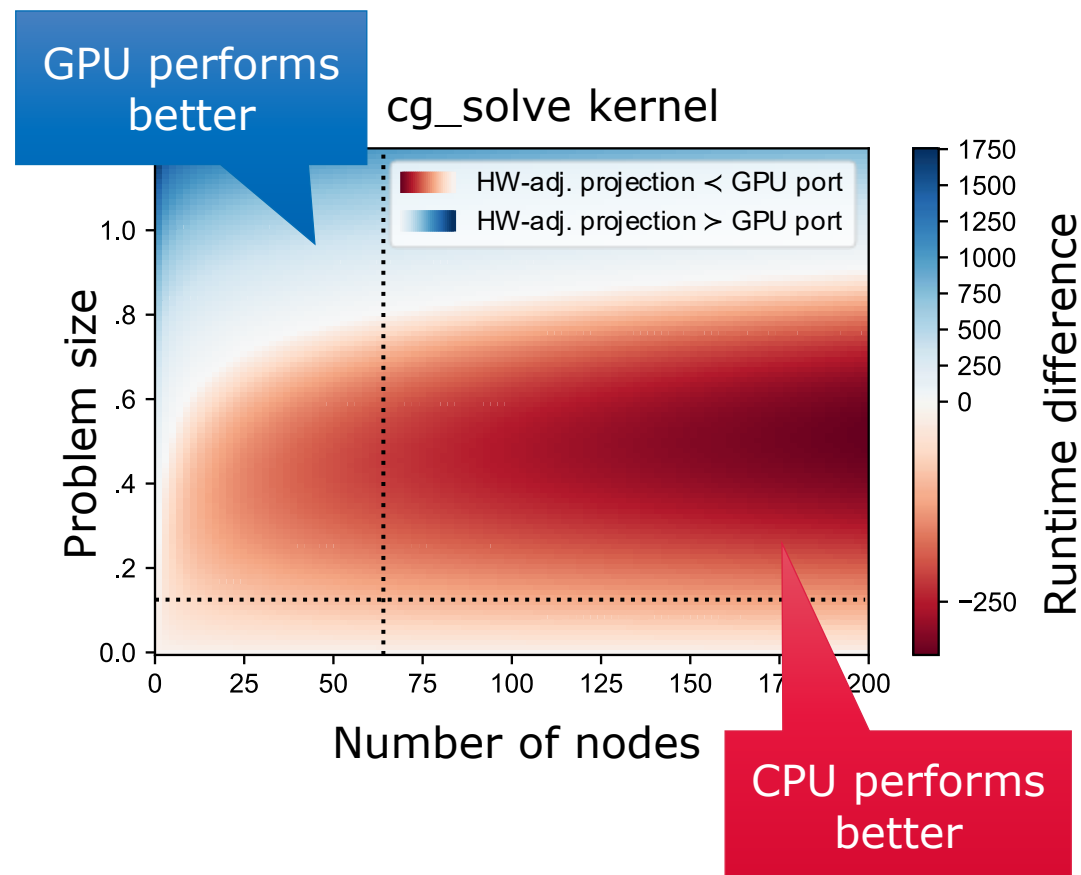
- Recommends experiment configuration strategy using our heuristics



Does a GPU port meet scaling expectations?

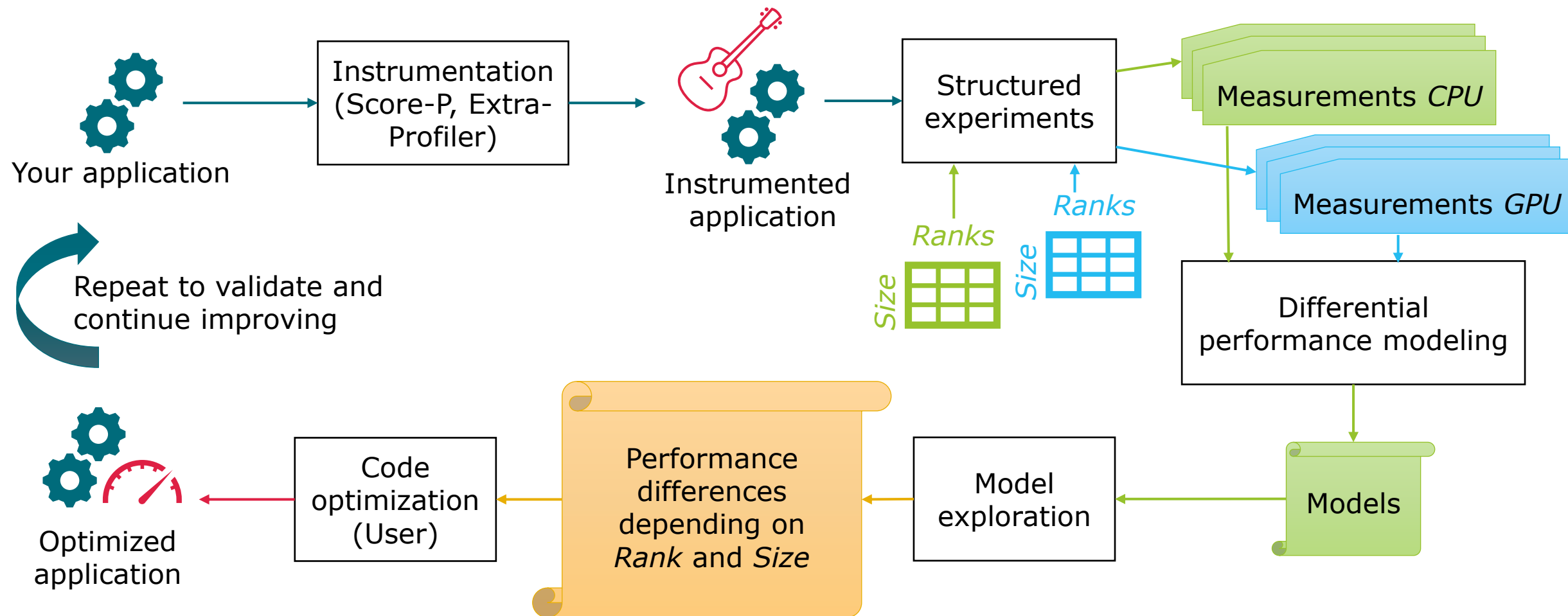
Beta

- GPUs are a prime vehicle to improve performance and energy efficiency
- However, porting to GPUs is hard
- Challenges of comparison
 - Re-structured source code
 - Asynchronous execution on GPU
 - Defining expectations
- Extra-P can help you with that!
 - Comparison assistant
 - Automatic call-tree mapping



Validating the performance of GPU ports

Beta



Using Extra-P

Installing Extra-P

- Easy to install via pip
- Just run: `python -m pip install extrap -upgrade --pre`
 - The `--upgrade` forces the installation of a new version
- All dependencies (packages) will be installed automatically

Use `--pre` to get the beta version

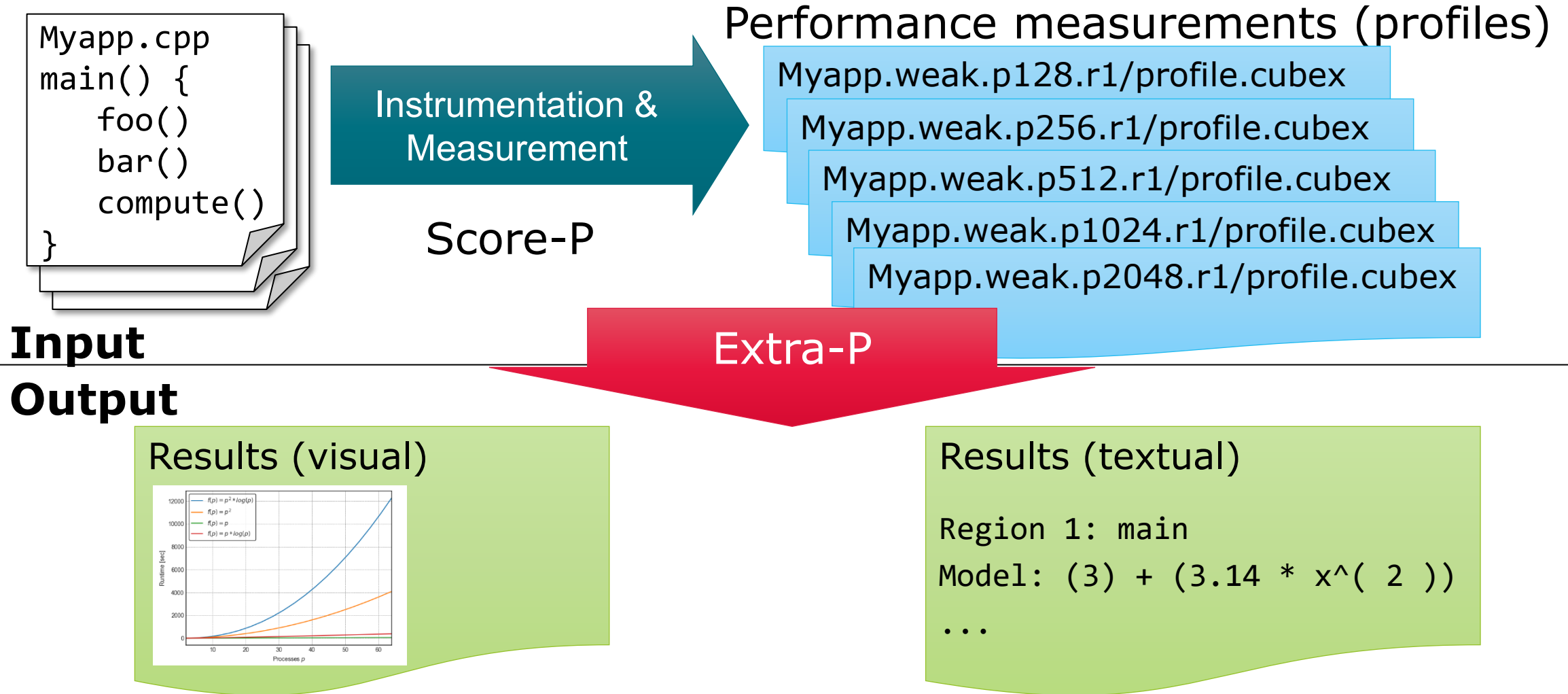
Extra-P in the tuning workshop

- Available at: <https://github.com/extra-p/extrap>
- When installed on the system simply run:
 - `extrap` – for the command line version
 - `extrap-gui` – for the graphical user interface version
- The GUI version is not intended to be used on the cluster



Already installed on
COSMA, just do
module load extrap

Automatic performance modeling with Extra-P



HANDS ON

Using the command line tool

Extra-P command line tool

Already installed on COSMA, just do
`module load extrap`

- Provides the same functionality, without visualization for use on cluster
- Usage guideline and command can be found at: <https://github.com/extra-p/extrap>
- 1.) Run: `extrap`
- Command Format: `extrap OPTIONS (--cube | --text | --talpas | --json | --extra-p-3) FILEPATH`
- 2.) Select input type: `extrap --text /cosma/apps/do009/shared/Extra-P/examples/one_parameter.txt`

Extra-P command line tool

3.) Output:

Callpath: compute

Metric: time

Measurement point: (2.00E+01) Mean: 8.19E+01 Median: 8.20E+01

Measurement point: (3.00E+01) Mean: 1.79E+02 Median: 1.78E+02

Measurement point: (4.00E+01) Mean: 3.19E+02 Median: 3.19E+02

Measurement point: (5.00E+01) Mean: 5.05E+02 Median: 5.06E+02

Measurement point: (6.00E+01) Mean: 7.25E+02 Median: 7.26E+02

Model: $-0.8897934098062804 + 0.20168243826499183 * x^{(2)}$

RSS: 3.43E+01

Adjusted R²: 1.00E+00

Callpath, kernel of the application that was measured

Metric name; either Score-P metrics (time, bytes, etc.) or custom metrics

Measurements for each input element (e.g., #processes)

Best-fit model

RSS: Residual sum of squares

Adjusted R² (explained previously)

Modeling sets of CUBE experiments

Extra-P Cube input description

Modeling tool expects CUBE files in the following format:

- `<DIR>/<PREFIX>.<PARAMETERS>.r<REPETITION>/<FILENAME>.cubex`
 - DIR, PREFIX, FILENAME – are just names, no meaning for Extra-P
 - REPETITION – number of the repetition of the experiments with same parameter values
- `<PARAMETERS>:=<PARAM1><VALUE1>...<PARAMn><VALUEn>`
 - List of parameter-value-pairs separated by “.”
 - PARAM – varied parameter e.g. number of processes
 - VALUE – value of the varied parameter

Extra-P Cube input description – example

- `app.processes2.size8000.r1`
- `app.processes2.size8000.r2`
- `app.processes2.size8000.r3`
- `app.processes2.size8000.r4`
- `app.processes2.size16000.r1`
- ...
- `app.processes2.size40000.r1`
- `app.processes4.size8000.r1`
- ...
- `app.processes4.size40000.r4`
- `app.processes8.size8000.r1`
- ...
- `app.processes8.size40000.r4`
- `app.processes16.size8000.r1`
- ...
- `app.processes16.size40000.r4`
- `app.processes32.size8000.r1`
- ...
- `app.processes32.size40000.r4`

Collecting a measurement series: sbatch script

```
#!/bin/bash
#SBATCH --job-name="BTMZ_B"           # Job name
##SBATCH --output=slurm-%A.out       # Job output file
#SBATCH --ntasks=2                   # Total number of MPI processes
#SBATCH --ntasks-per-node=4          # MPI processes per node
#SBATCH --cpus-per-task=4            # OMP threads per process
#SBATCH --time=00:05:00              # Maximum run time (hh:mm:ss)
#SBATCH --exclusive                   # No sharing of compute nodes
#SBATCH --partition=bluefield1       # DINE compute nodes
#SBATCH --account=do009
#SBATCH --array=1-5

# set up environment
module purge
module load gnu_comp openmpi
module load scorep

# measurement configuration
export SCOREP_EXPERIMENT_DIRECTORY=scorep_bt-mz_sum.p${SLURM_NTASKS}.r${SLURM_ARRAY_TASK_ID}
export SCOREP_FILTERING_FILE=../config/scorep.filt

# benchmark configuration
set -x
export OMP_NUM_THREADS=4
time -p mpiexec ./bt-mz_B.x
```

Used here to control repetitions

We want to vary the number of processes

Experiment directory name follows naming scheme

Copy file from:
/cosma/apps/do009/shared/
Extra-P/scorep-scaling.sbatch

Collecting a measurement series: start script

- We want to vary the number of processes

```
#!/bin/bash
for ntasks in 2 4 6 8 10; do
    sbatch --ntasks=$ntasks scorep-scaling.sbatch
done
```

Arguments take precedence
over #SBATCH directives

Copy file from:
/cosma/apps/do009/shared/
Extra-P/start-scaling.sh

Choose input file

Open set of CUBE files

Open Extra-P 3 experiment

Open JSON input

Open Talpas input

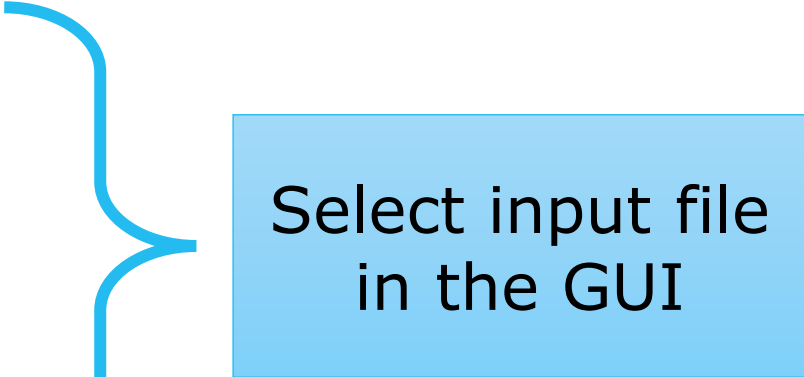
Open text input

Open experiment Ctrl+O

Save experiment Ctrl+S

Screenshot Ctrl+I

Exit



Select input file
in the GUI

Extra-P Cube input

Open set of CUBE files

Open Extra-P Experiment

Open JSON input

Open Talpas input

Open text input

Open experiment Ctrl+O

Save experiment Ctrl+S

Screenshot Ctrl+I

Exit



Import Options

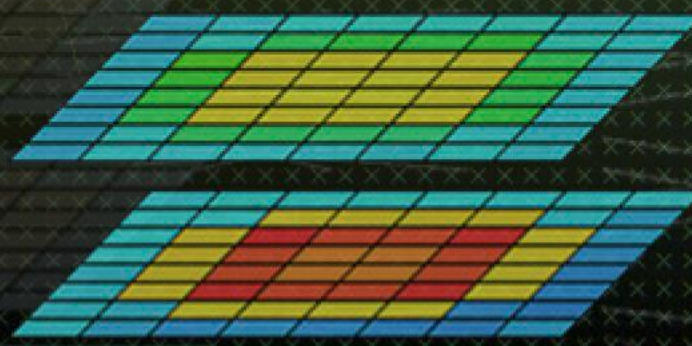
Scaling type: weak

Select the type of scaling analysis.
Use **strong** scaling if the problem size remains unchanged while adding more computational resources (e.g., nodes, processes, cores, threads) are added.
If the problem size was scaled alongside the computational resources, choose either **weak** scaling or **weak threaded** scaling when your application uses multithreading.

Keep values:

Keeps the individual measurement values of each repetition.
Can significantly slow down modeling and processing.

Restore Defaults OK Cancel



Visualization with Extra-P

Extra-P user interface

The screenshot displays the Extra-P user interface, which is divided into several main sections:

- Selection Panel:** Located at the top left, it includes a menu bar (File, View, Plots, Model, Help), a "Model:" dropdown set to "Default Model", and a "Metric:" dropdown set to "time".
- Call Tree:** A tree view showing the execution path. The "Eval_dv_dt" node is highlighted in blue, indicating it is the selected kernel(s). The tree includes nodes like PARALLEL, MPI_Init, MPI_Comm_size, MPI_Comm_rank, MPI_Bcast, MPI_Isend, MPI_Waitall, MPI_Allgather, MPI_Reduce, MPI_Allreduce, MPI_Irecv, MPI_Allgather, ComputeCornerF..., main, MPI_Comm_r..., MPI_Comm_si..., MPI_Irecv, MPI_Waitall, MPI_Isend, ComputeCorn..., MPI_Red..., and MPI Allreduce.
- Line Graph:** A plot titled "Line graph" showing the relationship between "time" (y-axis) and "p" (x-axis). The y-axis ranges from 6.41 to 19.2, and the x-axis ranges from 0 to 2.0x10⁴. A blue line labeled "Eval_dv_dt" shows a curve that rises sharply and then levels off. A callout points to this plot with the text "Plot of the model".
- Modeler Panel:** Located on the right, it includes a "Model name:" field set to "New Model", a radio button for "Model mean" (selected) and a dropdown for "Model Median", a "Model generator:" dropdown set to "Default", an "Advanced options" dropdown, and a "Generate models" button.
- Graph Limits:** A panel at the bottom right showing "X-axis" set to "p" and "max." set to "19660,80".
- Color Info:** A color scale at the bottom left ranging from 2.513x10⁻⁰⁴ (blue) to 17.295 (red).

Callouts highlight key features:

- "Selected kernel(s)" points to the "Eval_dv_dt" node in the call tree.
- "Call tree exploration" points to the call tree structure.
- "Plot of the model" points to the line graph.

Extra-P call tree view

Metric selection

Model selection

Call tree exploration

Model

Quality of fit metrics:
Residual sum of squares
and Adjusted R²

Impact of each kernel on
the metric at the
selected process count
compared to the other
kernels

Model: Default Model

Metric: time

Sev	Call tree	Anr	Value	RSS	Adj. R ²	SMAPE	RE
	PARALLEL		$0.958 + 1.837 \times 10^{-05} * p * \log_2(p)$	0.092	0.990	2.903	0.029
	MPI_Init		$0.457 + 2.393 \times 10^{-10} * p^{9/4}$	4.871x10...	0.998	0.829	8.259×10^{-03}
	MPI_Comm_size		$9.282 \times 10^{-04} + 1.232 \times 10^{-09} * p * \log_2(p)$	8.406x10...	0.981	1.096	0.011
	MPI_Comm_rank		$8.203 \times 10^{-04} + 1.796 \times 10^{-10} * p * \log_2(p)$	1.241x10...	0.987	0.155	1.552×10^{-03}
	MPI_Bcast		$0.155 + 8.580 \times 10^{-11} * p^{7/3}$	6.011x10...	0.997	3.815	0.038
	MPI_Isend		$3.157 \times 10^{-03} + 3.410 \times 10^{-05} * \log_2(p)$	1.641x10...	0.568	1.365	0.014
	MPI_Waitall		0.157	1.788x10...	1	2.934	3.950×10^{-03}
	MPI_Allgather		$0.018 + 1.218 \times 10^{-09} * p^{7/4}$	9.760x10...	0.980	5.835	0.058
	MPI_Reduce		2.513×10^{-04}	1.119x10...	1	13.626	0.261
	MPI_Allreduce		$0.170 + 2.478 \times 10^{-04} * p^{1/3} * \log_2(p)$	7.463x10...	0.790	5.390	0.054
	MPI_Irecv		$2.105 \times 10^{-03} + 5.098 \times 10^{-05} * \log_2(p)$	2.552x10...	0.693	2.447	0.025
	MPI_Allgatherv		$3.555 \times 10^{-04} + 2.418 \times 10^{-07} * p^{5/4}$	2.974x10...	0.997	6.479	0.068
	▶ ComputeCornerForces		$0.089 - 1.645 \times 10^{-04} * \log_2(p)$	1.201x10...	-0.124	0.469	4.685×10^{-03}
	▼ main		2.140	4.007x10...	1	1.264	6.078×10^{-03}
	MPI_Comm_rank		$0.013 - 1.971 \times 10^{-05} * \log_2(p)$	9.469x10...	0.319	0.280	2.799×10^{-03}
	MPI_Comm_size		0.016	1.771x10...	1	0.341	5.654×10^{-04}
	MPI_Finalize		0.072	9.185x10...	1	1.582	0.012
	MPI_Waitall		0.203	0.019	1	22.235	0.239
	MPI_Isend		$0.079 + 7.508 \times 10^{-04} * \log_2(p)$	5.459x10...	0.695	1.033	0.010
	▶ Eval_dv_dt		$13.301 + 0.190 * \log_2(p)$	0.803	0.392	2.211	0.022
	▼ ComputeCornerForces		17.295	0.038	1	0.340	7.466×10^{-04}
	MPI_Reduce		$0.090 + 5.990 \times 10^{-09} * p^{5/4} * \log_2(p)$	4.877x10...	0.684	2.470	0.025
	MPI_Allreduce		1.597	0.051	1	4.829	0.084
	MPI_Finalize		2.734x10...	0.051	1	28.617	0.235
	MPI_Recv		1.304×10^{-03}	0.051	0.829	39.132	0.529

All Show model Show parameters

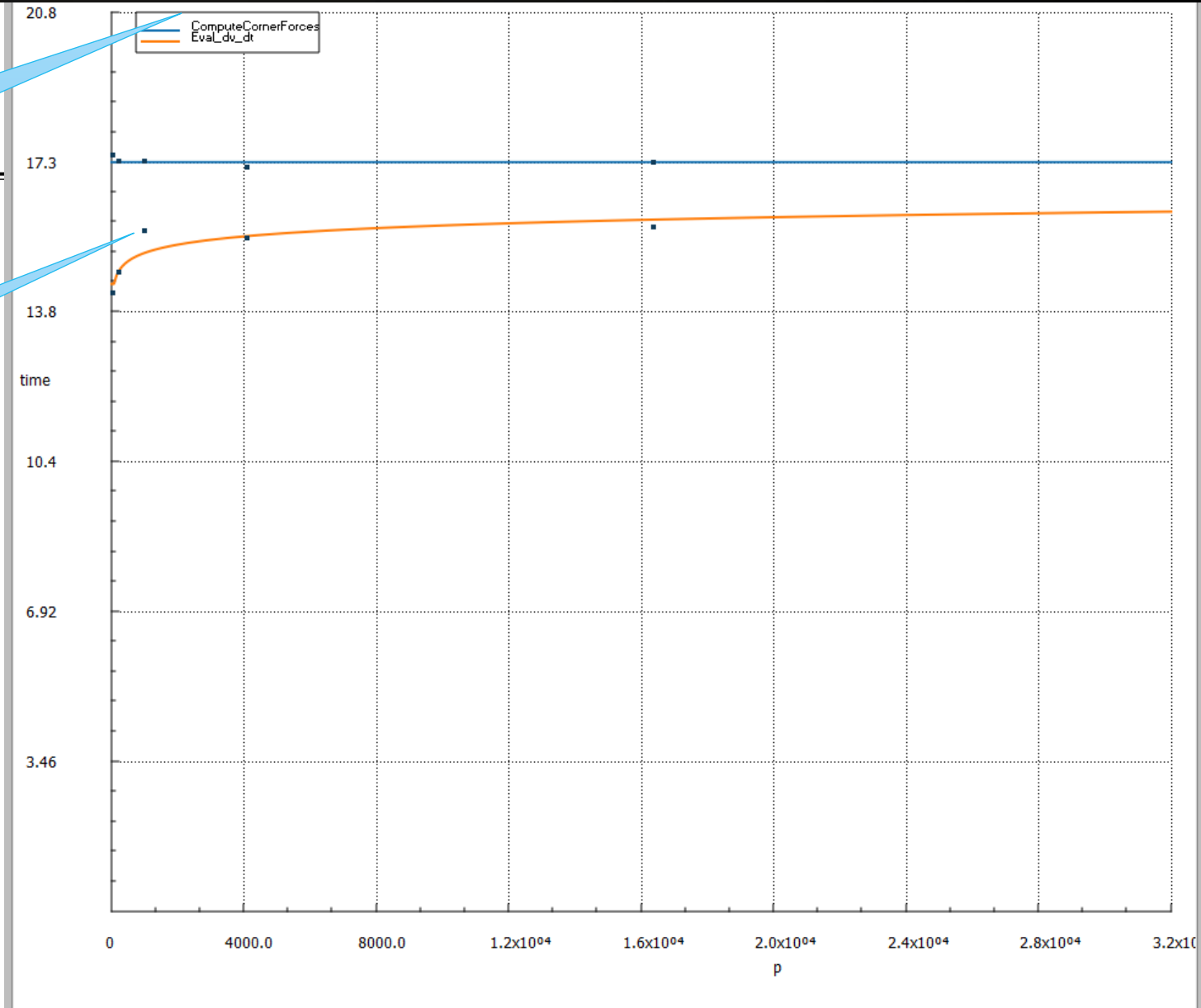
Asymptotic behavior

Extra-P model view

Models selected in the Call path view

Measurement values

X axis scale control for prediction of behavior at other process counts



Graph Limits

X-axis

p

max.

32000,00

52

Modeling measurements from a text file

Extra-P input in text form

- Useful when no CUBE files are available or when modeling a small data set

```
PARAMETER p  
POINTS 1000 2000 4000 8000 16000  
METRIC metric1  
REGION region1  
DATA 1 1 1 1 1  
DATA 4 4 4 3.99 4.01  
DATA 16 15.999 16.01 16.01 15.99  
DATA 64 64 64 64.01 63.99  
DATA 256.01 255.99 256 256
```

Parameter name

This name will be used in the GUI as well as in the textual output

Measurement points

Use at least 5, but in general the more the better

Metric name

Region name

Both used to determine the output Cube file hierarchical structure and identify separate data sets

Data points

Each row corresponds to a point; all values in a row are considered repeat measurements of the same experiment

Extra-P input as JSON lines

- Useful when you do not want to use CUBE files
 - Easy to generate with your own scripts
- Each line of the file is a JSON object
 - Describes one measurement value

```
{"params": {"<parameter1>": 0, "...": "..."}, "value": 0.0,  
"callpath": "<callpath>", "metric": "<metric>" } ↵
```



One line

Example

```
{"params":{"x":1,"y":1}, "metric":"metr", "callpath":"test", "value":2.03}  
{"params":{"x":1,"y":2}, "metric":"metr", "callpath":"test", "value":3.02}  
{"params":{"x":1,"y":3}, "metric":"metr", "callpath":"test", "value":4.01}  
{"params":{"x":1,"y":4}, "metric":"metr", "callpath":"test", "value":5.02}  
{"params":{"x":1,"y":5}, "metric":"metr", "callpath":"test", "value":6.01}  
[...]
```



Comparing models between experiments

Open/import first experiment

Open set of <u>C</u> UBE files	
Open set of <u>E</u> xtra-Prof files	
Open Extra-P <u>3</u> experiment	
Open <u>J</u> SON input	
Open set of <u>N</u> sight files	
Open Talpas input	
Open <u>t</u> ext input	
<hr/>	
<u>O</u> pen experiment	Ctrl+O
<u>S</u> ave experiment	Ctrl+S
<u>C</u> ompare with experiment	
<hr/>	
<u>S</u> creenshot	Ctrl+I
<hr/>	
<u>E</u> xit	

Import
measurements

Open stored
experiment

Example files are in:

/cosma/apps/do009/shared/Extra-P/examples/CPU-GPU-comparison

Start comparison

Open set of CUBE files
Open set of Extra-Prof files
Open Extra-P 3 experiment
Open JSON input
Open set of Nsight files
Open Talpas input
Open text input

Open experiment Ctrl+O

Save experiment Ctrl+S

Compare with experiment

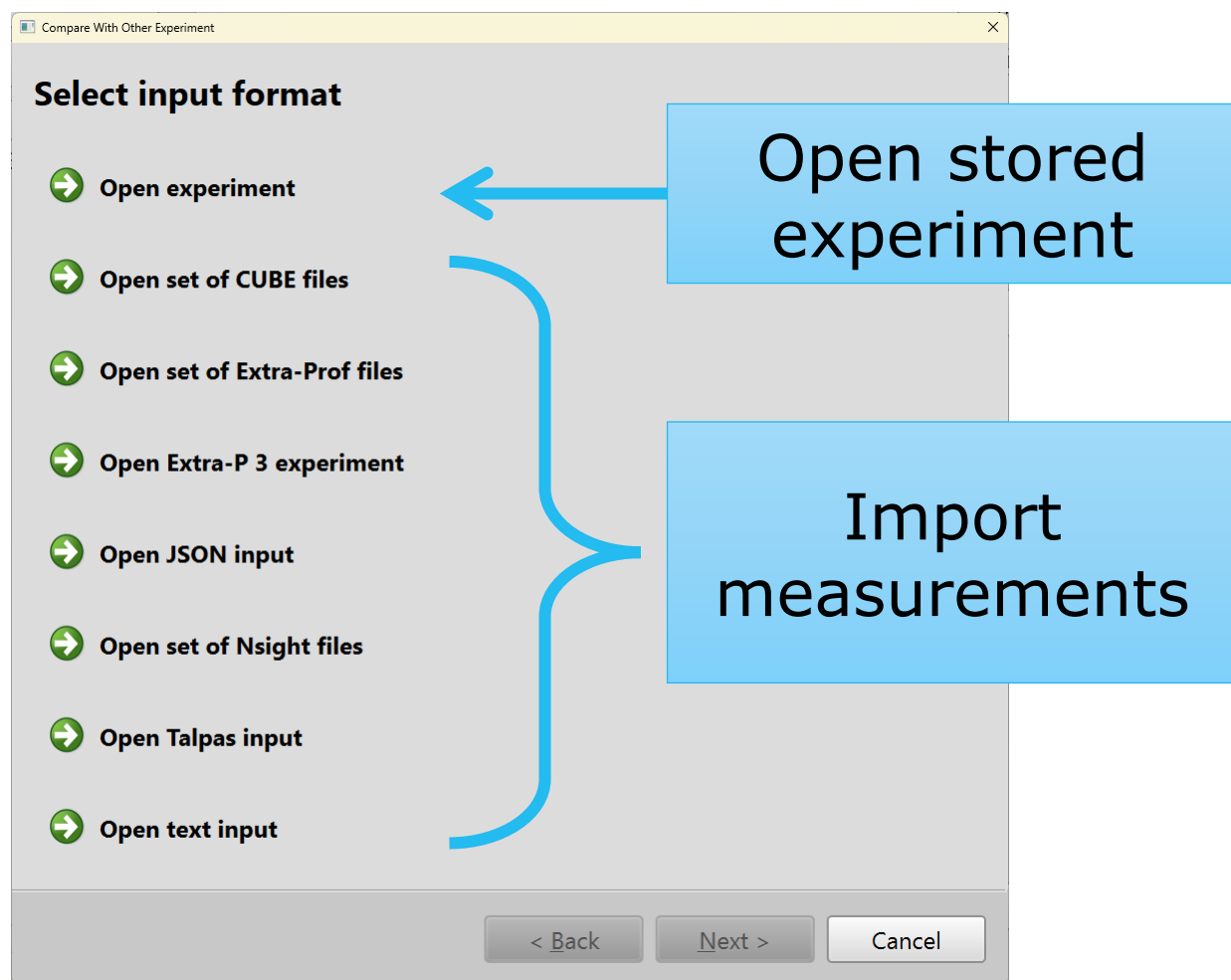
Screenshot Ctrl+I

Exit

Start comparison
assistant



Open/import second experiment

Beta

The screenshot shows a dialog box titled "Compare With Other Experiment" with a list of input format options. A blue box labeled "Open stored experiment" has an arrow pointing to the "Open experiment" option. A larger blue box labeled "Import measurements" has a bracket pointing to the "Open set of CUBE files", "Open set of Extra-Prof files", "Open Extra-P 3 experiment", "Open JSON input", "Open set of Nsight files", "Open Talpas input", and "Open text input" options. At the bottom of the dialog are buttons for "< Back", "Next >", and "Cancel".

Select input format

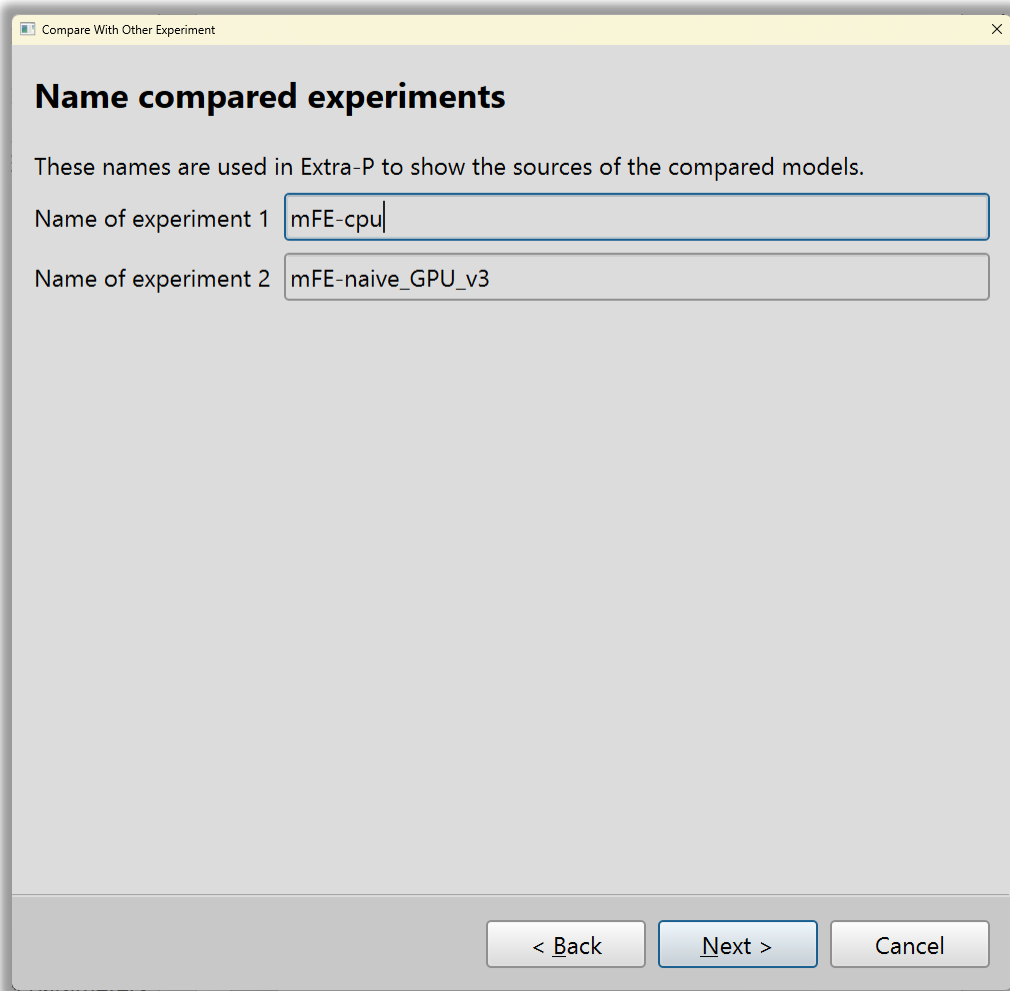
- Open experiment
- Open set of CUBE files
- Open set of Extra-Prof files
- Open Extra-P 3 experiment
- Open JSON input
- Open set of Nsight files
- Open Talpas input
- Open text input

< Back Next > Cancel

Follow the comparison assistant



Beta



Compare With Other Experiment

Name compared experiments

These names are used in Extra-P to show the sources of the compared models.

Name of experiment 1

Name of experiment 2

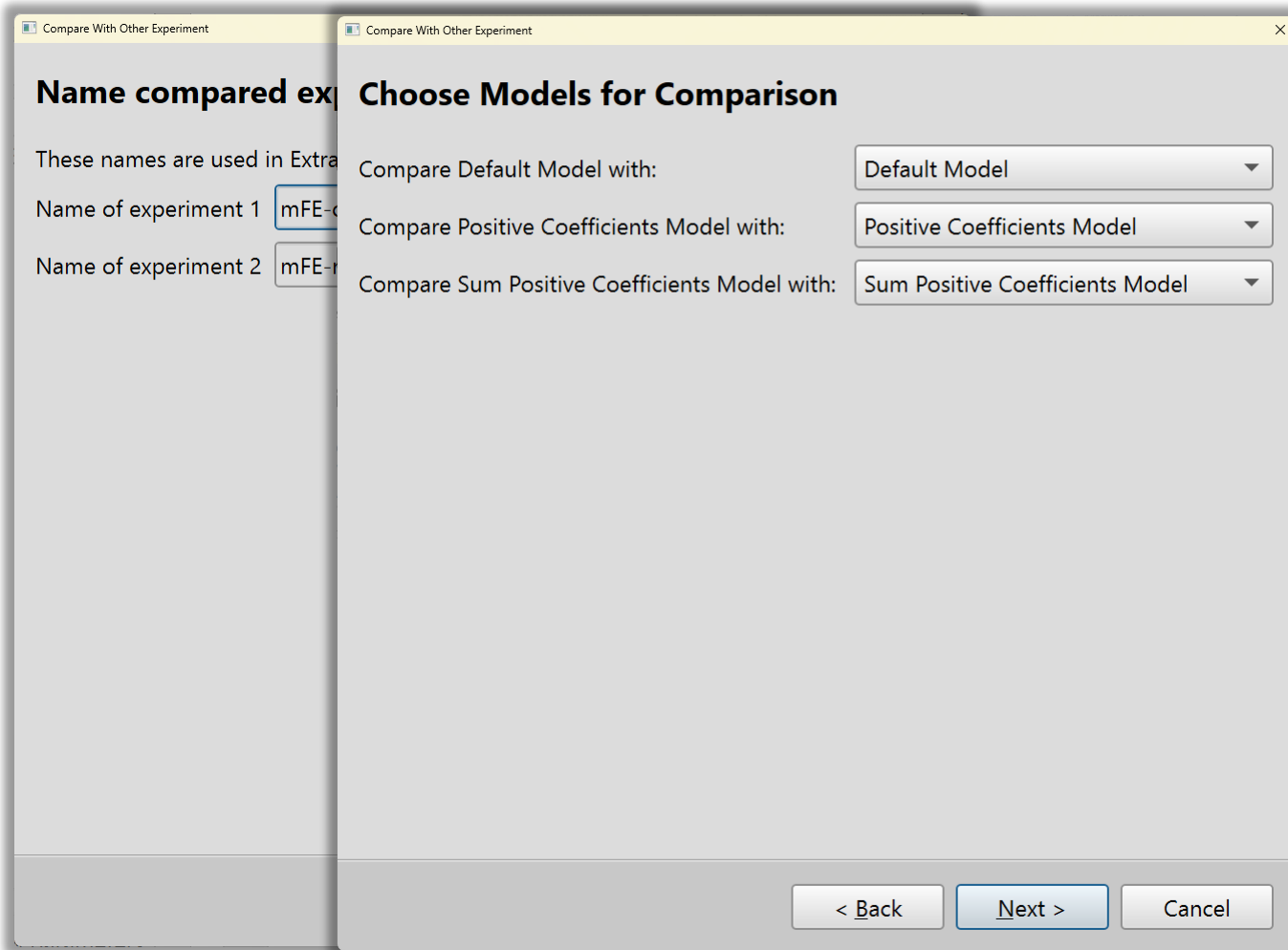
< Back Next > Cancel

Tip: short names
improve readability

Follow the comparison assistant



Beta



Compare With Other Experiment

Name compared ex

These names are used in Extra

Name of experiment 1 mFE-c

Name of experiment 2 mFE-c

Choose Models for Comparison

Compare Default Model with: Default Model

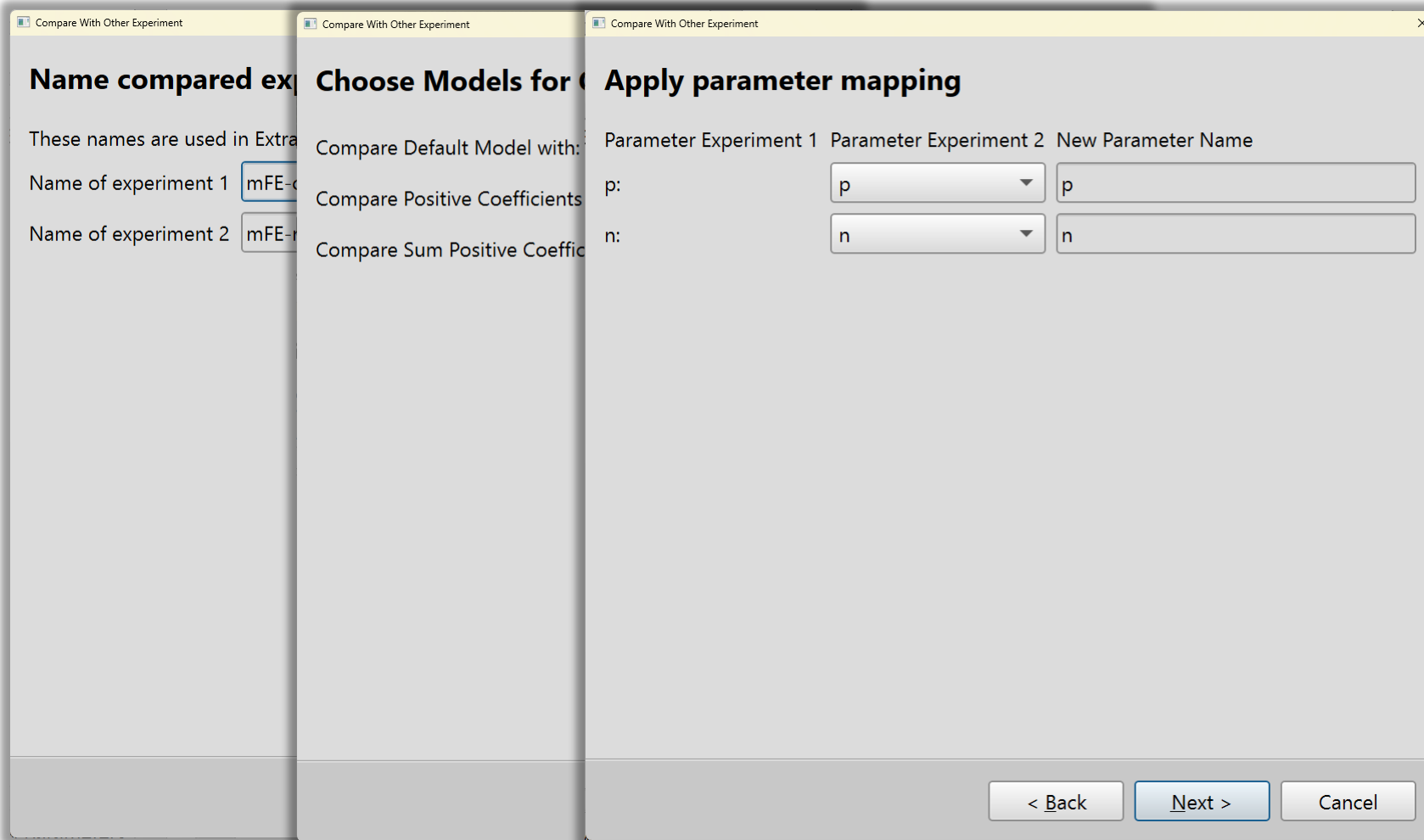
Compare Positive Coefficients Model with: Positive Coefficients Model

Compare Sum Positive Coefficients Model with: Sum Positive Coefficients Model

< Back Next > Cancel

Model sets with identical name are matched automatically

Follow the comparison assistant

Beta

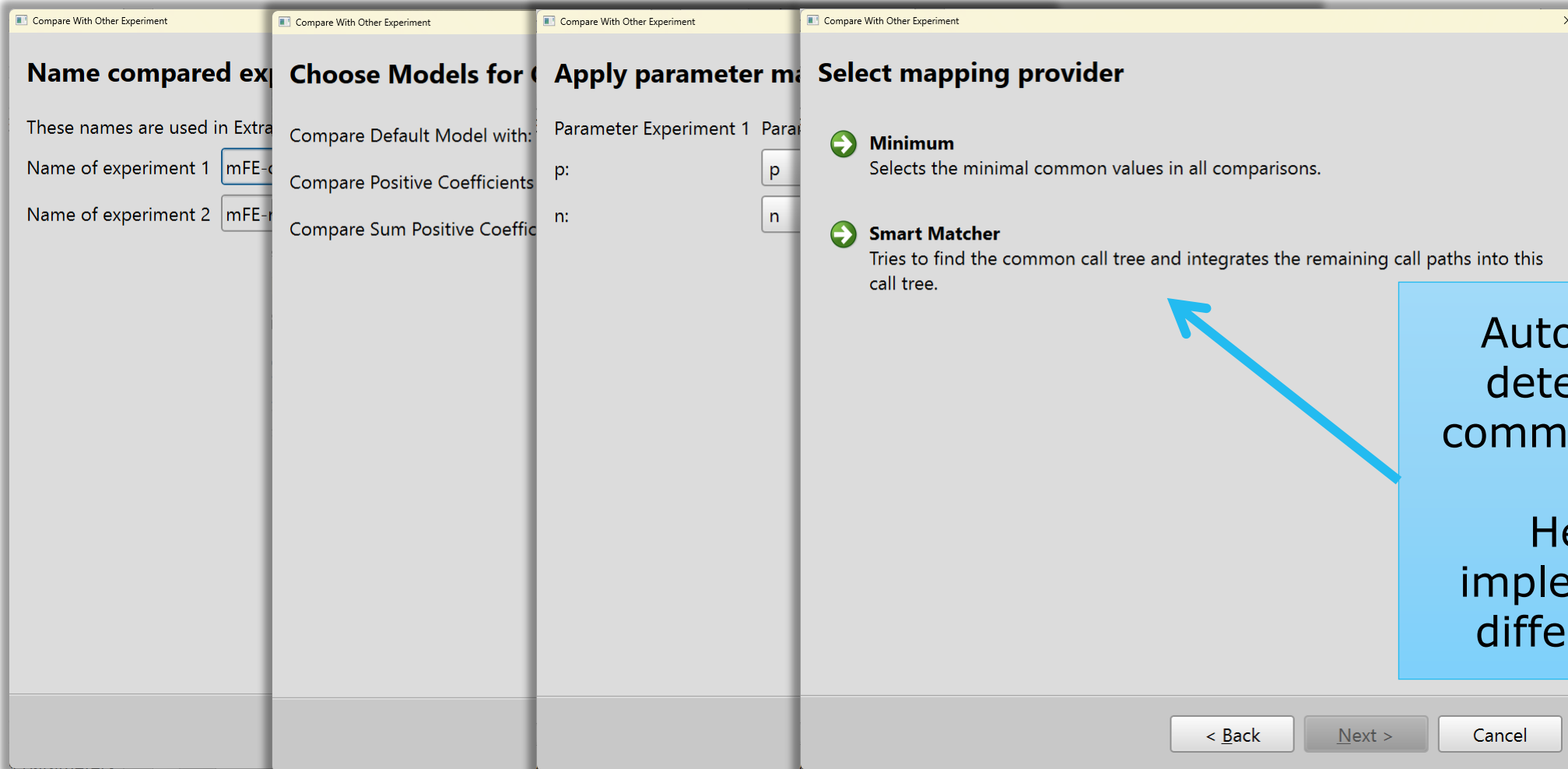
The screenshot shows three overlapping windows of the 'Compare With Other Experiment' wizard. The first window is titled 'Name compared experiment' and contains the text 'These names are used in Extra' and two input fields for 'Name of experiment 1' and 'Name of experiment 2', both containing 'mFE-'. The second window is titled 'Choose Models for Comparison' and contains three radio button options: 'Compare Default Model with', 'Compare Positive Coefficients', and 'Compare Sum Positive Coefficients'. The third window is titled 'Apply parameter mapping' and contains a table with three columns: 'Parameter Experiment 1', 'Parameter Experiment 2', and 'New Parameter Name'. The table has two rows: 'p:' with a dropdown menu showing 'p' and an input field containing 'p'; and 'n:' with a dropdown menu showing 'n' and an input field containing 'n'. At the bottom of the third window are three buttons: '< Back', 'Next >', and 'Cancel'.

Tip: use identical names to avoid confusion

Follow the comparison assistant



Beta



The screenshot shows a four-step wizard for comparing experiments. The steps are:

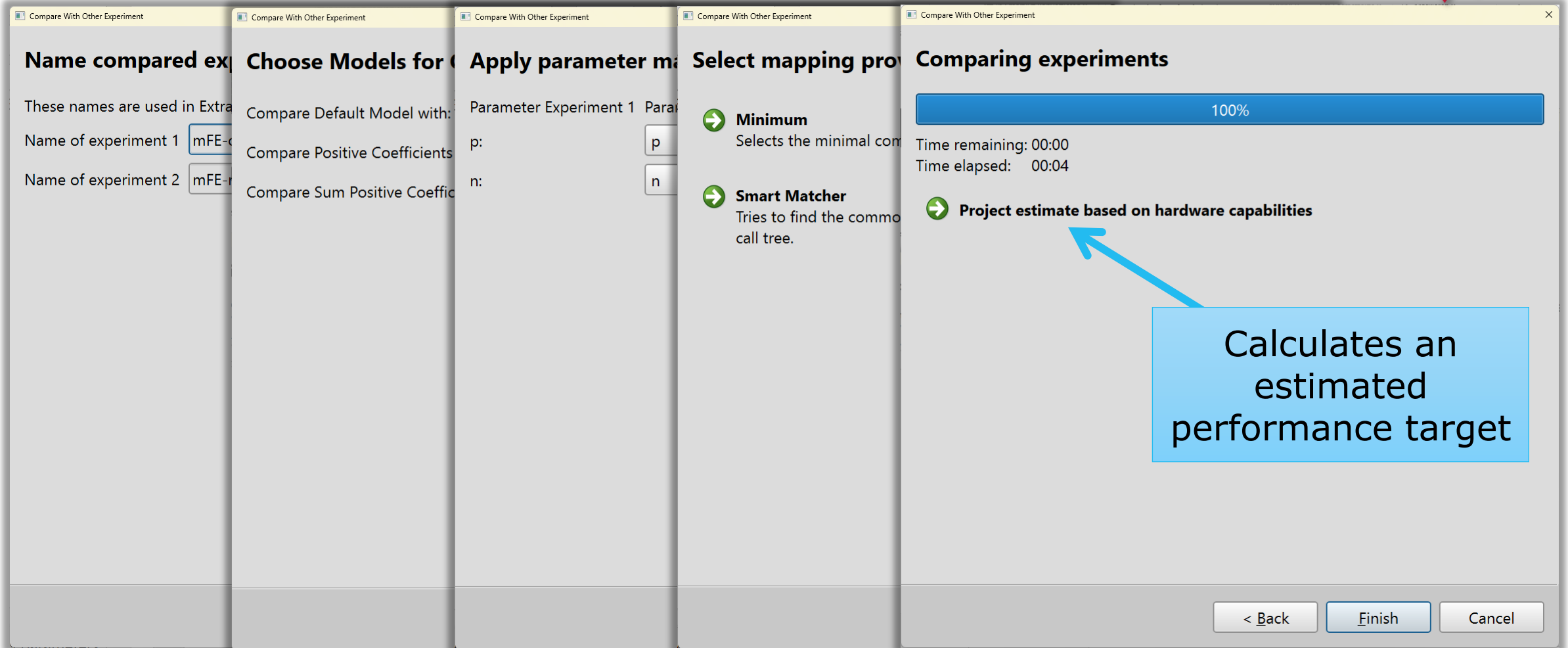
- Name compared experiments:** "These names are used in Extra... Name of experiment 1 [mFE-c] Name of experiment 2 [mFE-r]"
- Choose Models for Comparison:** "Compare Default Model with: [p] Compare Positive Coefficients: [n] Compare Sum Positive Coefficients: [n]"
- Apply parameter mapping:** "Parameter Experiment 1 [p] Parameter Experiment 2 [n]"
- Select mapping provider:** "Minimum: Selects the minimal common values in all comparisons. Smart Matcher: Tries to find the common call tree and integrates the remaining call paths into this call tree." A blue arrow points from a text box to the "Smart Matcher" option.

Navigation buttons at the bottom: < Back, Next >, Cancel

Automatically determines a common call tree

Helpful if implementation differs slightly

Follow the comparison assistant

Beta

The image displays a sequence of five screenshots from a software interface, illustrating the steps of a comparison assistant. The screenshots are arranged horizontally, showing the progression from naming experiments to the final comparison phase.

- Step 1: Name compared experiments**

These names are used in Extra...
Name of experiment 1: mFE-c
Name of experiment 2: mFE-r
- Step 2: Choose Models for Comparison**

Compare Default Model with:
Compare Positive Coefficients:
Compare Sum Positive Coefficients:
- Step 3: Apply parameter mapping**

Parameter Experiment 1: p
Parameter Experiment 2: n
- Step 4: Select mapping procedure**
 - Minimum**
Selects the minimal common...
 - Smart Matcher**
Tries to find the common call tree.
- Step 5: Comparing experiments**

Progress bar: 100%
Time remaining: 00:00
Time elapsed: 00:04

 - Project estimate based on hardware capabilities**

Annotation: Calculates an estimated performance target

Navigation buttons: < Back, Finish, Cancel

Project expected improvement based on hardware capability

Beta

Compare With Other Experiment

Hardware-adjusted projection

Target experiment mFE-cpu mFE-GPU

Metrics to project

- visits
- time

Roofline model information


	Memory bandwidth	Peak performance	
mFE-cpu	196,67 GBytes/s	601,80 GFlops/s	<input type="button" value="Load ERT JSON file..."/>
mFE-GPU	785,04 GBytes/s	6420,94 GFlops/s	<input type="button" value="Load ERT JSON file..."/>

Use arithmetic intensity (operation intensity) to improve accuracy

Floating point operations (double precision)

Number of memory transfers

Bytes per memory transfer



Uses roofline models to estimate improvement

The roofline model info can be determined using the empirical roofline toolkit

Interactive comparison of CPU and GPU variants



Comparison indicator

Selection

Model: Default Model

Metric: time

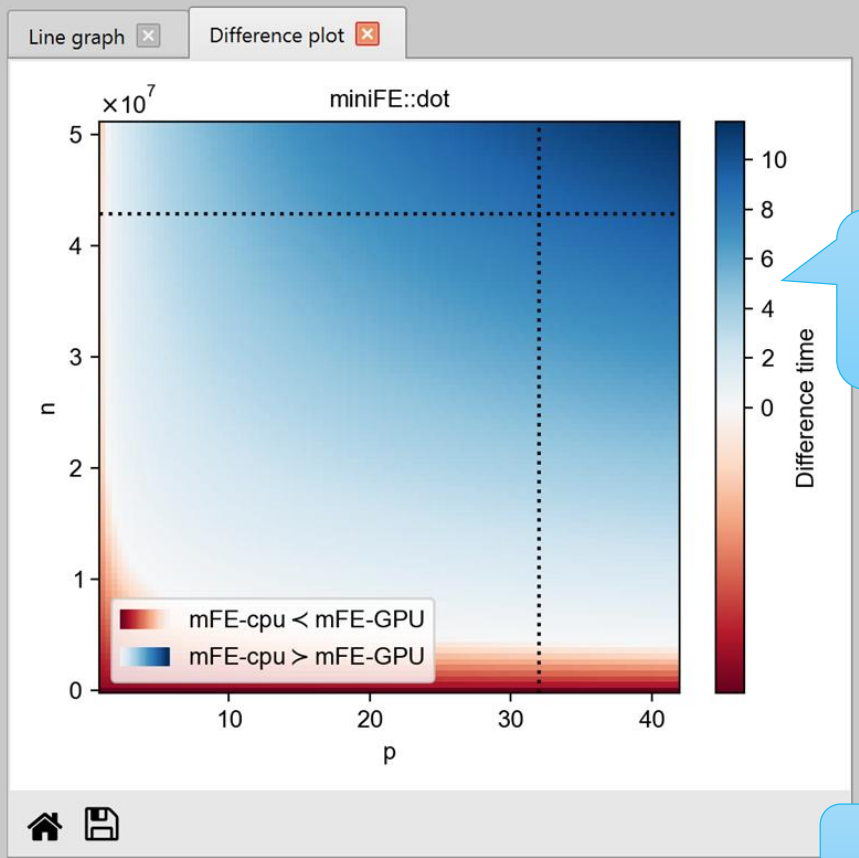
	Anr	Value	RSS
▶ miniFE::get_id		$[1.072 \times 10^{-8} * n^{1/2} * \log_2(n)^2 \dots]$	0.001
▶ miniFE::find_row_for_id		$[-0.003 * p^{1/4} + 2.784 \times 10^{-7} \dots]$	0.002
▶ miniFE::generate_matr...		$[1.798 \times 10^{-20} * n^{7/4} + 4.276 \dots]$	81.416
▶ miniFE::assemble_FE...		$[2.505 \times 10^{-6} \downarrow 1.473 \times 10^{-6}]$	1.426 x 1
▶ YAML_Element::add		$[-8.965 \times 10^{-10} * p^{1/2} * \log_2(p) \dots]$	1.826 x 1
▶ YAML_Element::add		$[-8.965 \times 10^{-10} * p^{1/2} * \log_2(p) \dots]$	1.826 x 1
▶ YAML_Element::add		$[-8.965 \times 10^{-10} * p^{1/2} * \log_2(p) \dots]$	1.826 x 1
▶ YAML_Element::get		$[-3.154 \times 10^{-6} * \log_2(p) + 3.2 \dots]$	1.050 x 1
▶ YAML_Element::add		$[-8.965 \times 10^{-10} * p^{1/2} * \log_2(p) \dots]$	1.826 x 1
▶ YAML_Element::add		$[-8.965 \times 10^{-10} * p^{1/2} * \log_2(p) \dots]$	1.826 x 1
▶ miniFE::impose_dirichl...		$[1.691 \times 10^{-14} * n^{3/2} * \log_2(n) \dots]$	5.653
▶ miniFE::make_local_m...		$[1.323 \times 10^{-5} * n^{1/2} * p^{5/4} + 1 \dots]$	4.387
▶ miniFE::compute_matr...		$[3.623 \times 10^{-21} * n^{7/4} * p^{1/2} + \ell \dots]$	9.590 x 1
▶ miniFE::cg_solve		$[2.566 \times 10^{-9} * n^{3/4} * p^{1/2} * \log_2(p) \dots]$	614.146
▶ [Comparison]			
▶ miniFE::mytimer		$[-5.456 \times 10^{-7} * p^{3/4} * \log_2(p) \dots]$	1.374 x 1
▶ miniFE::waxpby		$[7.517 \times 10^{-11} * p^{1/3} * \log_2(n) \dots]$	1.969 x 1
▶ miniFE::dot_r2		$[6.303 \times 10^{-10} * n^{3/4} * p^{1/4} * \log_2(p) \dots]$	8.680
▶ miniFE::daxpby		$[2.051 \times 10^{-15} * n^{3/2} + 3.943 \dots]$	0.002
▶ miniFE::dot		$[-5.824 \times 10^{-8} * p^{4/3} * \log_2(p) \dots]$	88.257
▶ YAML_Element::add		$[-8.965 \times 10^{-10} * p^{1/2} * \log_2(p) \dots]$	1.826 x 1

Color Info: 0 to 389.563

Ranking

mFE-cpu >= mFE-GPU		mFE-cpu <= mFE-GPU	
Difference	Callpath	Difference	Callpath
386.172	main->miniFE::driver...	75.000	main->miniFE::driver...
39.648	main->miniFE::driver...	34.364	main->miniFE::driver...
8.754	main->miniFE::driver...	9.171	main->miniFE::driver...

Select comparison point



Difference of behavior for selected call path

Ranking best and worst call paths

Interactive comparison of CPU and GPU variants with projected target



Projected runtime

Selection

Model: Default Model

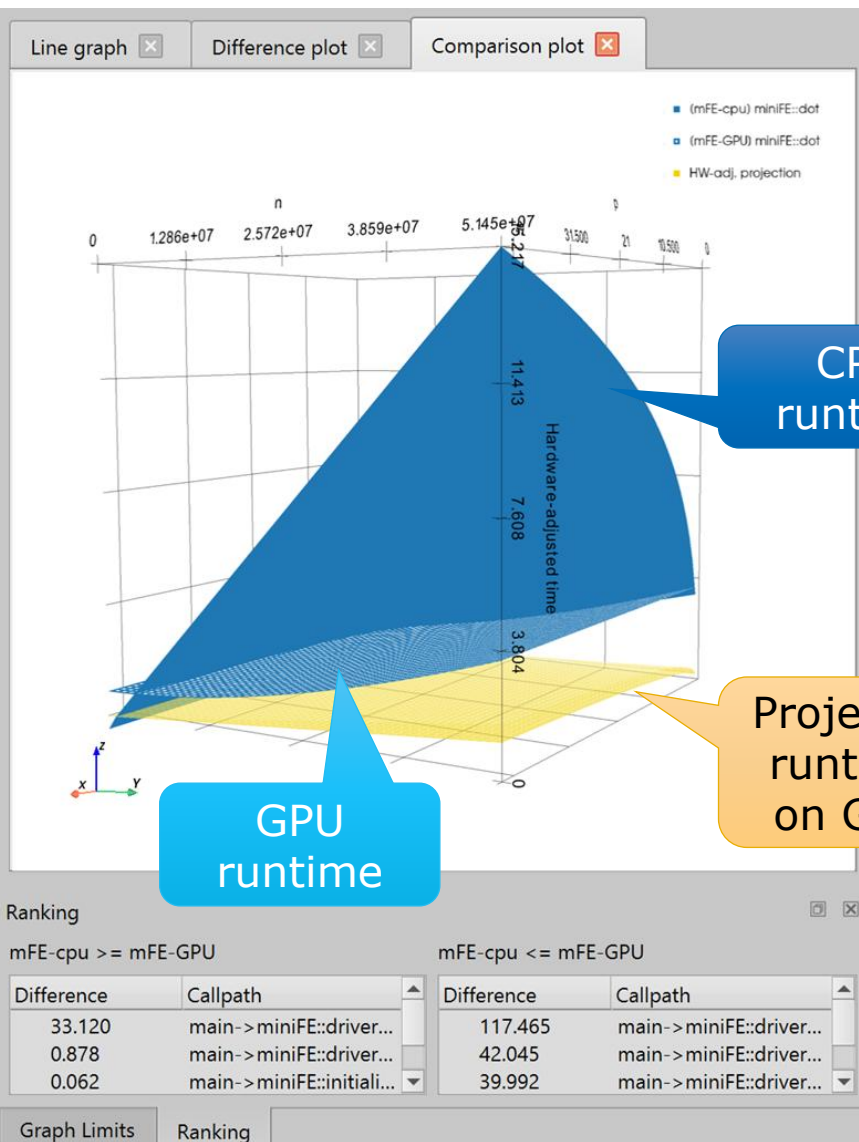
Metric: Hardware-adjusted time

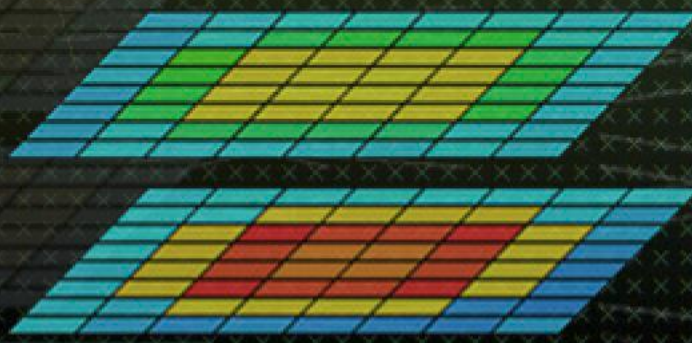
Sev	Callpath	Anr	Value	RSS
	▶ YAML_Element::add		$[-8.402 \times 10^{-11} * p^{1/2} * \log_2(p... 1.826 \times 10^7]$	
	▶ YAML_Element::get		$[-2.956 \times 10^{-7} * \log_2(p)^1 + 3. (... 1.050 \times 10^7]$	
	▶ YAML_Element::add		$[-8.402 \times 10^{-11} * p^{1/2} * \log_2(p... 1.826 \times 10^7]$	
	▶ YAML_Element::add		$[-8.402 \times 10^{-11} * p^{1/2} * \log_2(p... 1.826 \times 10^7]$	
	▶ YAML_Element::add		$[-8.402 \times 10^{-11} * p^{1/2} * \log_2(p... 1.826 \times 10^7]$	
	miniFE::impose_dirichl...		$[1.585 \times 10^{-15} * n^{3/2} * \log_2(n) ... 5.653]$	
	miniFE::make_local_m...		$[1.24 \times 10^{-6} * n^{1/2} * p^{5/4} + 1.3... 4.387]$	
	▶ miniFE::compute_matr...		$[3.395 \times 10^{-22} * n^{7/4} * p^{1/2} + \dots 9.590 \times 10^7]$	
	▼ miniFE::cg_solve		$[2.405 \times 10^{-10} * n^{3/4} * p^{1/2} * \text{lc... 614.146]$	
	miniFE::mytimer		$[-5.114 \times 10^{-8} * p^{3/4} * \log_2(p) ... 1.374 \times 10^7]$	
	miniFE::waxpby		$[7.045 \times 10^{-12} * p^{1/3} * \log_2(n) ... 1.969 \times 10^7]$	
	miniFE::dot_r2		$[5.908 \times 10^{-11} * n^{3/4} * p^{1/4} * \text{lc... 8.680]$	
	miniFE::daxpby		$[1.923 \times 10^{-16} * n^{3/2} + 3.695 \times ... 0.002]$	
	miniFE::dot		$[-5.459 \times 10^{-9} * p^{4/3} * \log_2(p) ... 88.257]$	
	▶ YAML_Element::add		$[-8.402 \times 10^{-11} * p^{1/2} * \log_2(p... 1.826 \times 10^7]$	
	▶ YAML_Element::add		$[-8.402 \times 10^{-11} * p^{1/2} * \log_2(p... 1.826 \times 10^7]$	
	▶ YAML_Element::add		$[-8.402 \times 10^{-11} * p^{1/2} * \log_2(p... 1.826 \times 10^7]$	
	▶ YAML_Element::add		$[5.903 \times 10^{-18} * n^{2/3} * p^{2/3} * \text{lc... 4.861 \times 10^7]$	
	▶ YAML_Doc::generateYAML		$[0.01457 7.712 \times 10^{-19} * n^2 ... 3.598]$	
	miniFE::finalize_mpi		$[2.972 \times 10^{-11} * n^{1/3} * \log_2(n) ... 3.028]$	
	▶ YAML_Doc::~YAML_Doc		$[2.043 \times 10^{-7} 5.943 \times 10^{-17} * r... 9.542 \times 10^7]$	

Color Info: 0 to 121.856

Ranking

mFE-cpu >= mFE-GPU		mFE-cpu <= mFE-GPU	
Difference	Callpath	Difference	Callpath
33.120	main->miniFE::driver...	117.465	main->miniFE::driver...
0.878	main->miniFE::driver...	42.045	main->miniFE::driver...
0.062	main->miniFE::initiali...	39.992	main->miniFE::driver...





- What additional features would you like to see?
- Did you find any bugs?

You can contact us via email: extra-p-support@lists.parallel.informatik.tu-darmstadt.de

Or on GitHub using the issues tool: <https://github.com/extra-p/extrap>