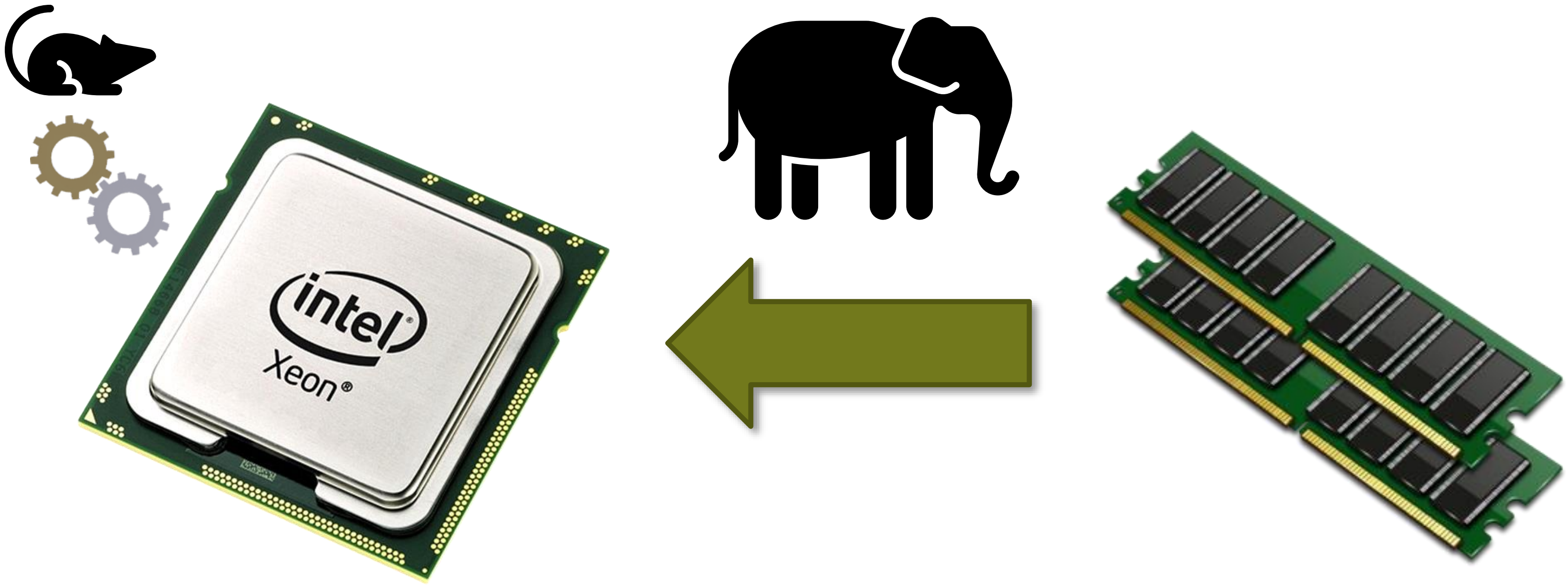
Boosting Performance Optimization with Interactive Data Movement Visualization

Philipp Schaad*, Tal Ben-Nun*, Torsten Hoefler*

* Scalable Parallel Computing Laboratory, ETH Zurich

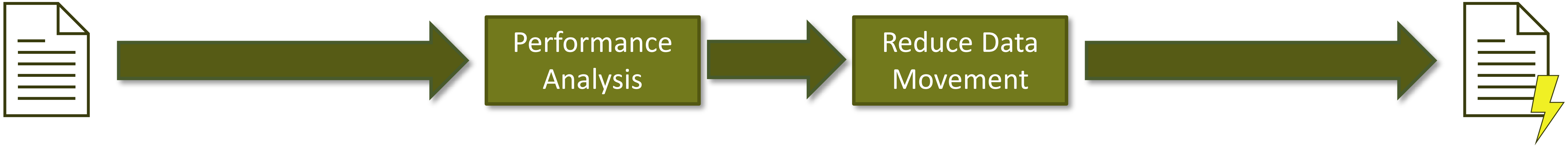
SC22 Dallas, TX | hpc accelerates.

The Cost of Data Movement

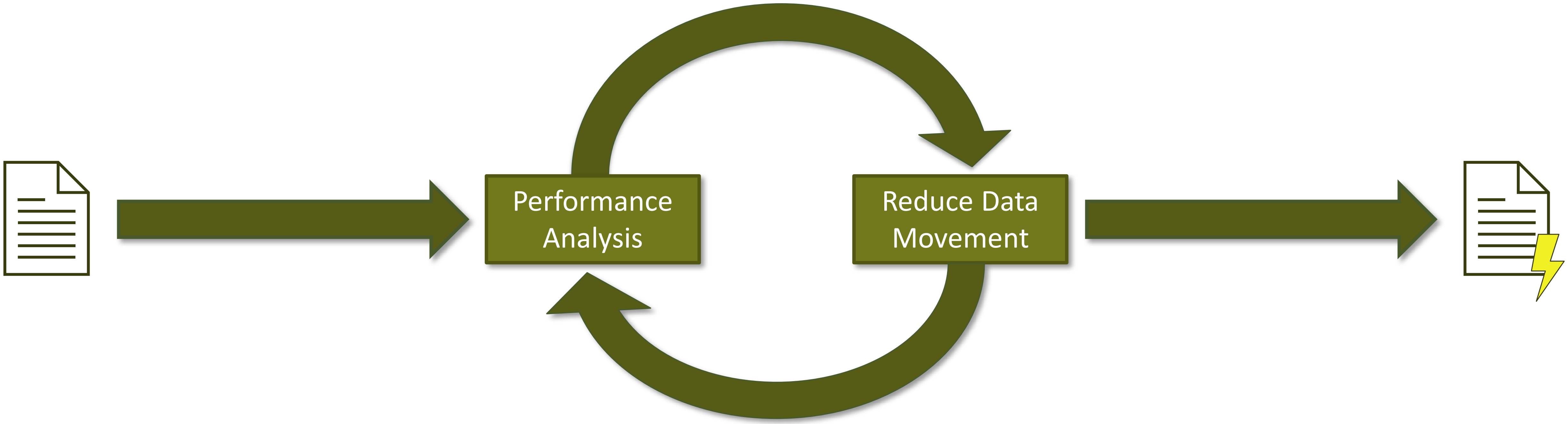


Exploit ***spatial locality***
and ***temporal locality***!

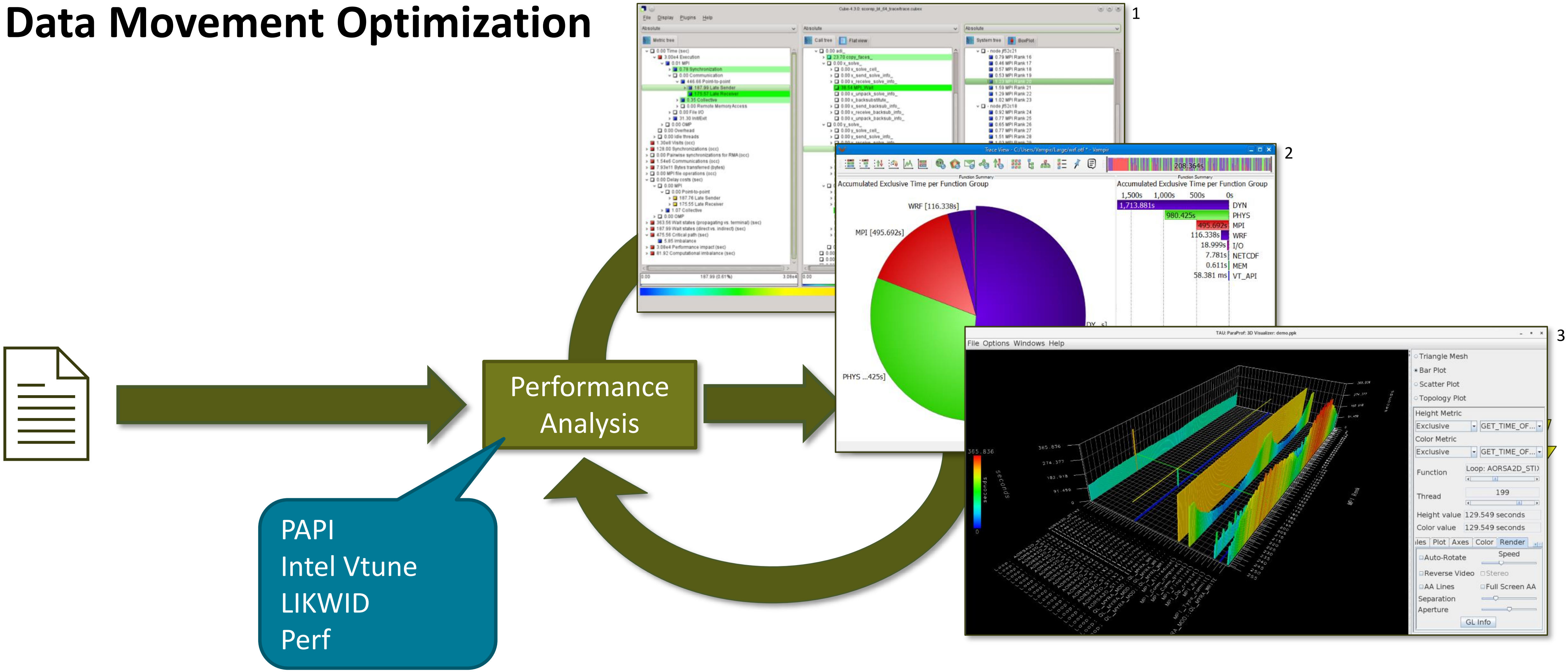
Data Movement Optimization



Data Movement Optimization

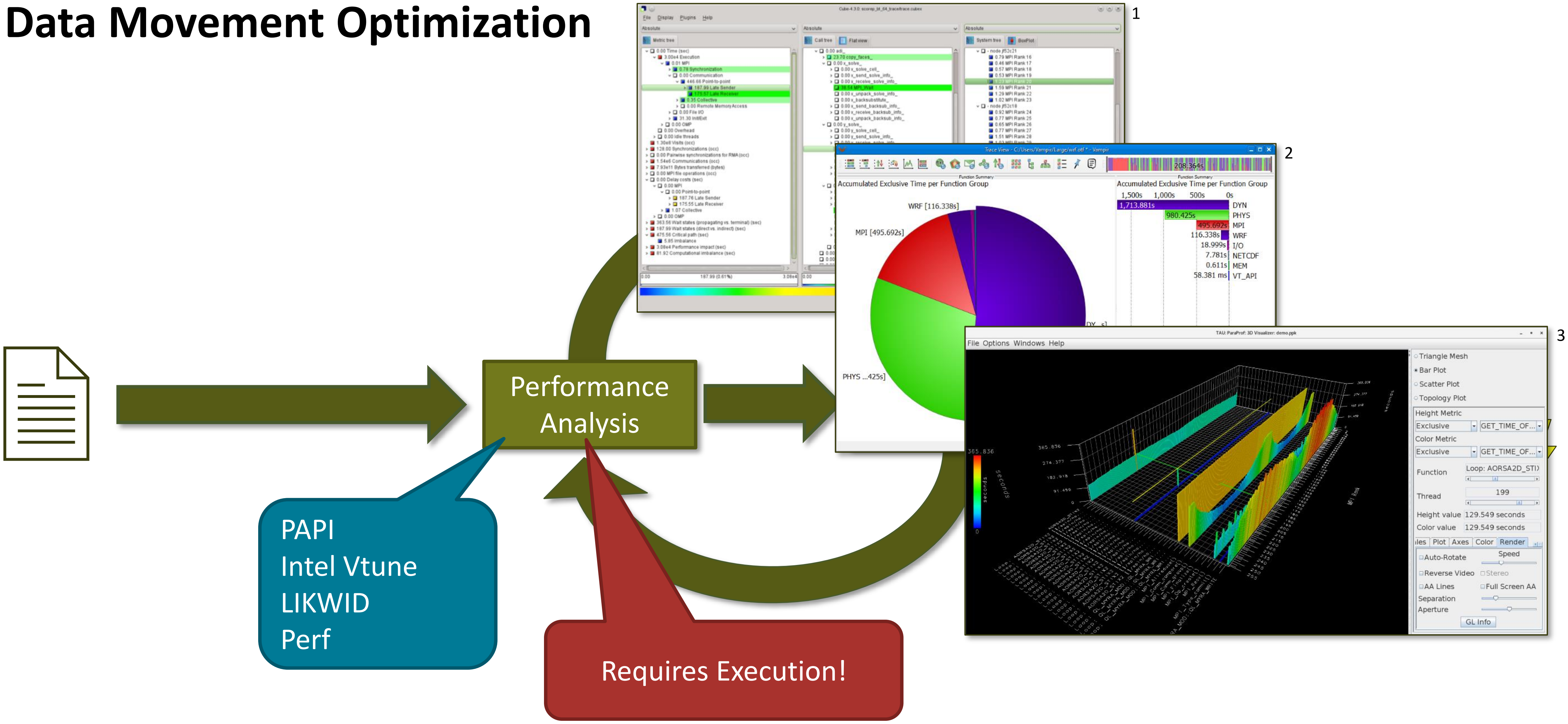


Data Movement Optimization



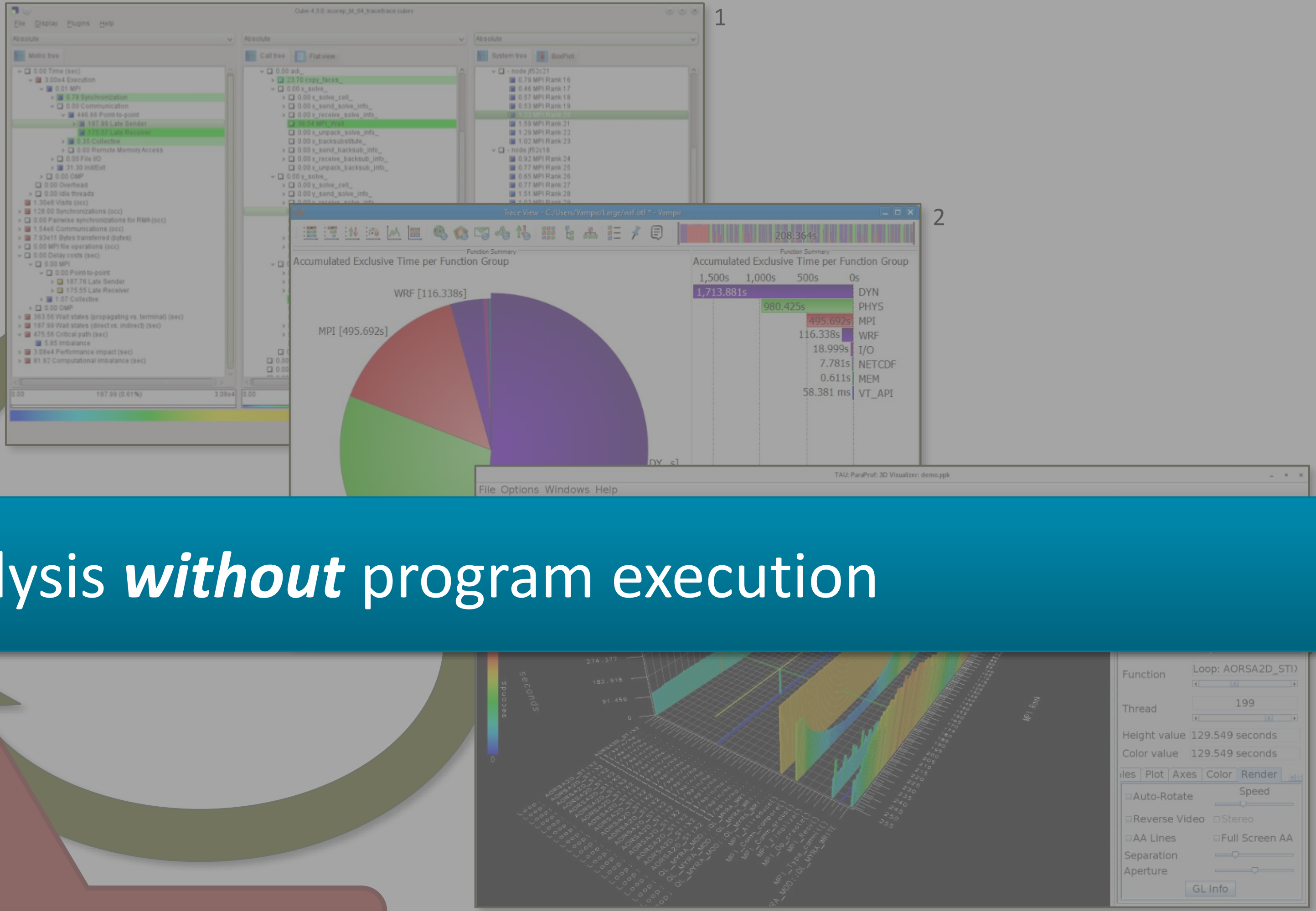
[1] Saviankou et al., Cube v4: From Performance Report Explorer to Performance Analysis Tool
[2] Nagel et al., VAMPIR: Visualization and Analysis of MPI Resources
[3] Bell et al., ParaProf: A Portable, Extensible, and Scalable Tool for Parallel Performance Profile Analysis

Data Movement Optimization



[1] Saviankou et al., Cube v4: From Performance Report Explorer to Performance Analysis Tool
[2] Nagel et al., VAMPIR: Visualization and Analysis of MPI Resources
[3] Bell et al., ParaProf: A Portable, Extensible, and Scalable Tool for Parallel Performance Profile Analysis

Data Movement Optimization

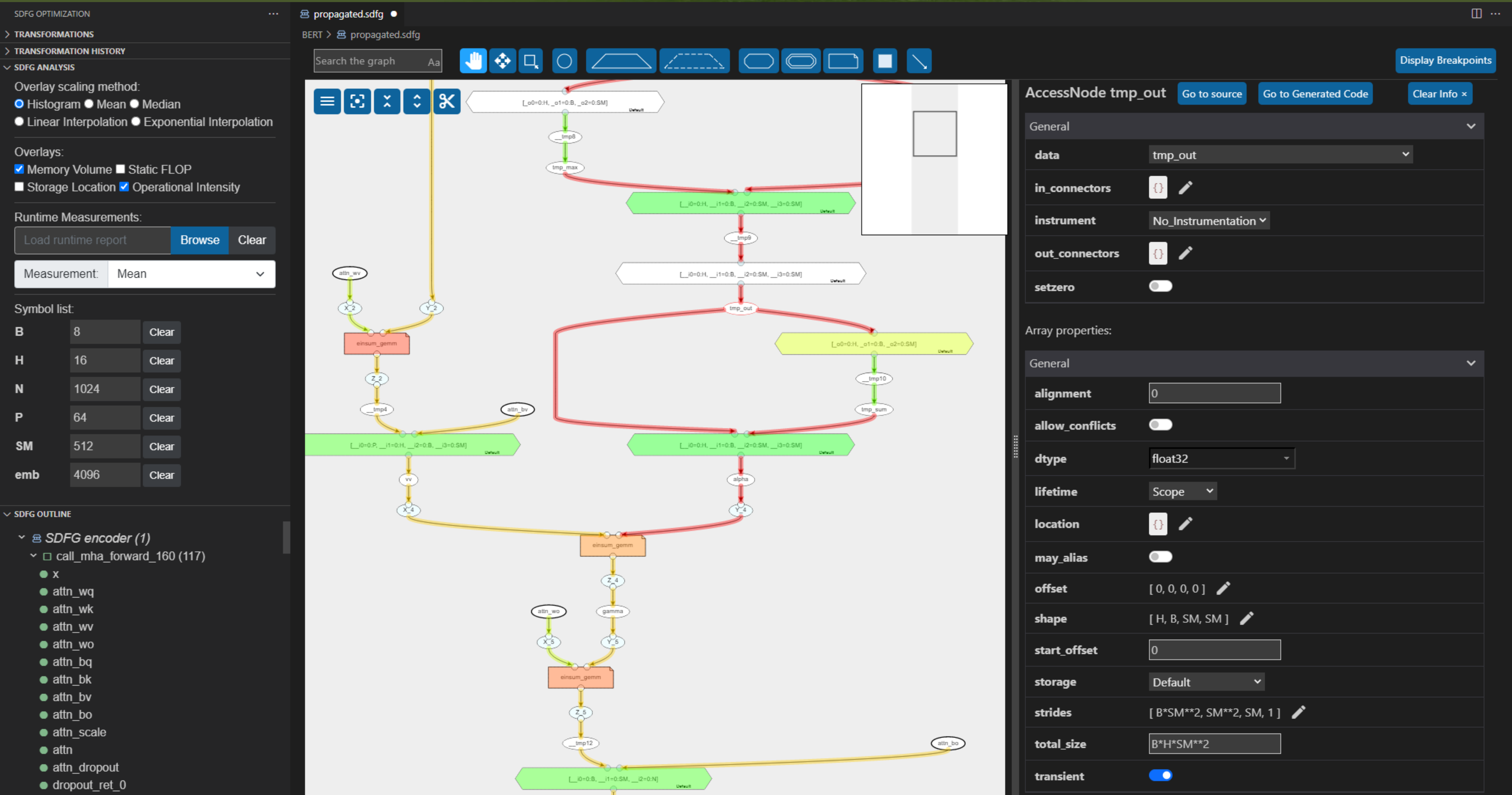


Performance analysis ***without*** program execution

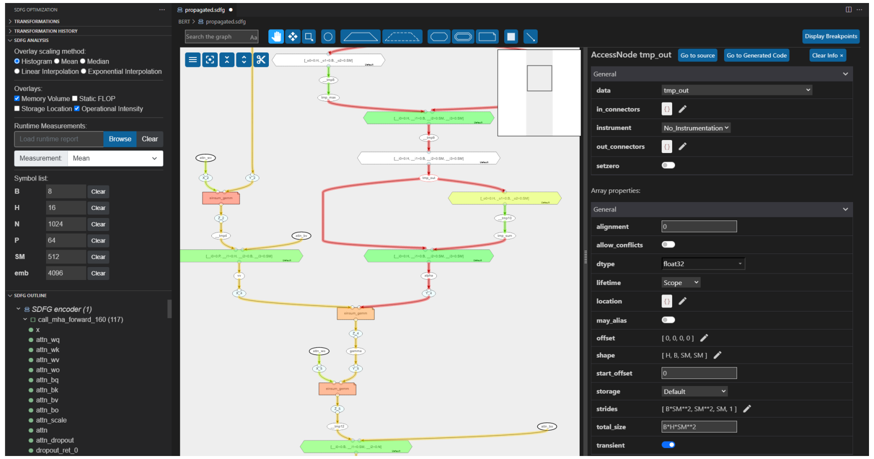
PAPI
Intel Vtune
LIKWID
Perf

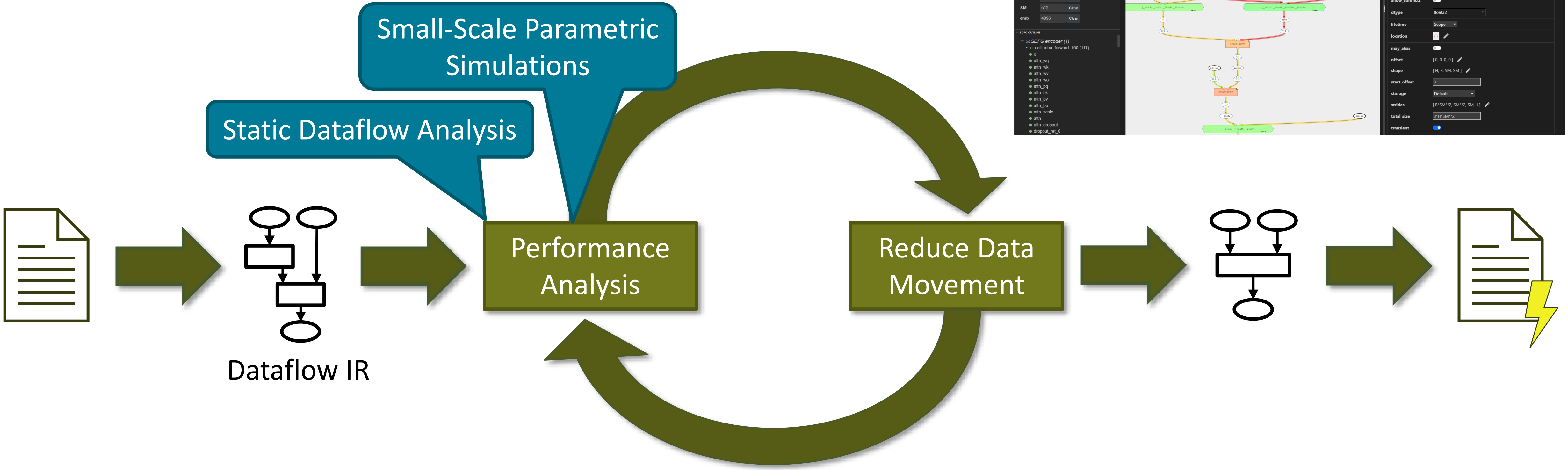
Requires Execution!

[1] Saviankou et al., Cube v4: From Performance Report Explorer to Performance Analysis Tool
[2] Nagel et al., VAMPIR: Visualization and Analysis of MPI Resources
[3] Bell et al., ParaProf: A Portable, Extensible, and Scalable Tool for Parallel Performance Profile Analysis

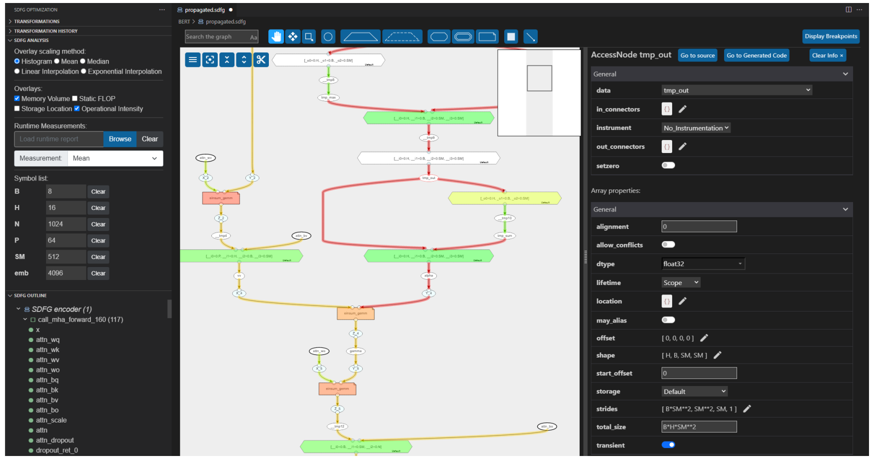

SDFG OPTIMIZATION
TRANSFORMATIONS
TRANSFORMATION HISTORY
SDFG ANALYSIS
Overlay scaling method:
Histogram
Mean
Median
Linear Interpolation
Exponential Interpolation
Overlays:
Memory Volume
Static FLOP
Storage Location
Operational Intensity
Runtime Measurements:
Load runtime report
Browse
Clear
Measurement:
Mean
Symbol list:
B 8 Clear
H 16 Clear
N 1024 Clear
P 64 Clear
SM 512 Clear
emb 4096 Clear
SDFG OUTLINE
SDFG encoder (1)
call_mha_forward_160 (117)
x
attn_wq
attn_wk
attn_wv
attn_wo
attn_bq
attn_bk
attn_bv
attn_bo
attn_scale
attn
attn_dropout
dropout_ret_0
propagated.sdfg
BERT > propagated.sdfg
Search the graph
Display Breakpoints
AccessNode tmp_out
Go to source
Go to Generated Code
Clear Info ×
General
data tmp_out
in_connectors
instrument No_Instrumentation
out_connectors
setzero
Array properties:
General
alignment 0
allow_conflicts
dtype float32
lifetime Scope
location
may_alias
offset [0, 0, 0, 0]
shape [H, B, SM, SM]
start_offset 0
storage Default
strides [B*SM**2, SM**2, SM, 1]
total_size B*H*SM**2
transient

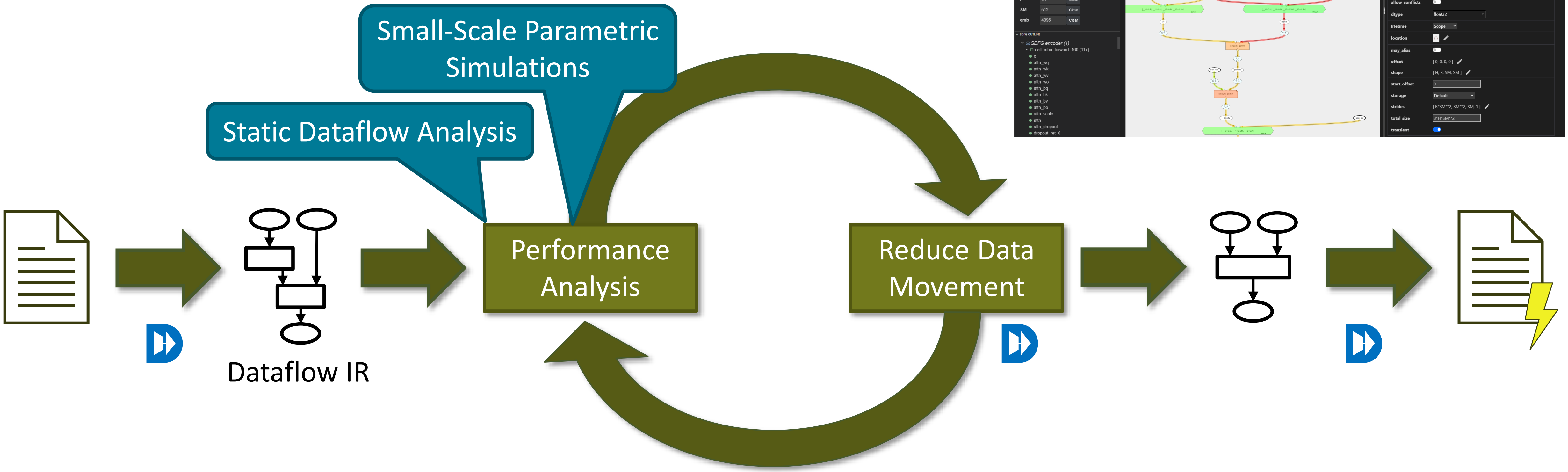
Data Movement Optimization





Data Movement Optimization

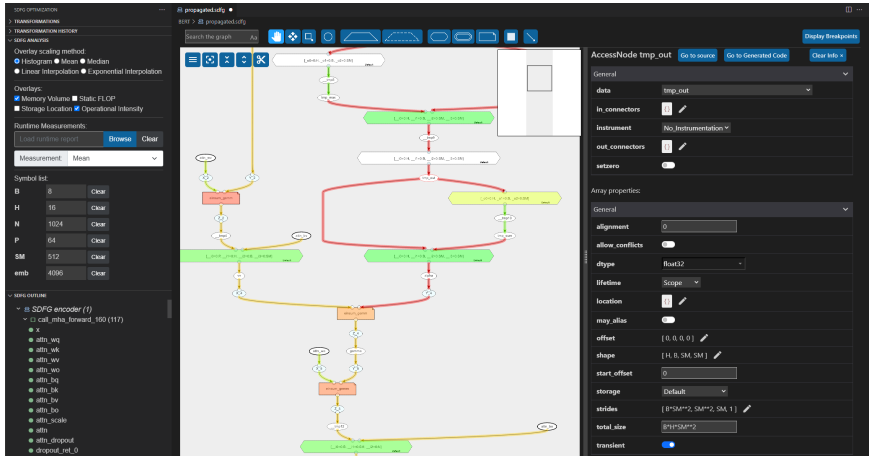


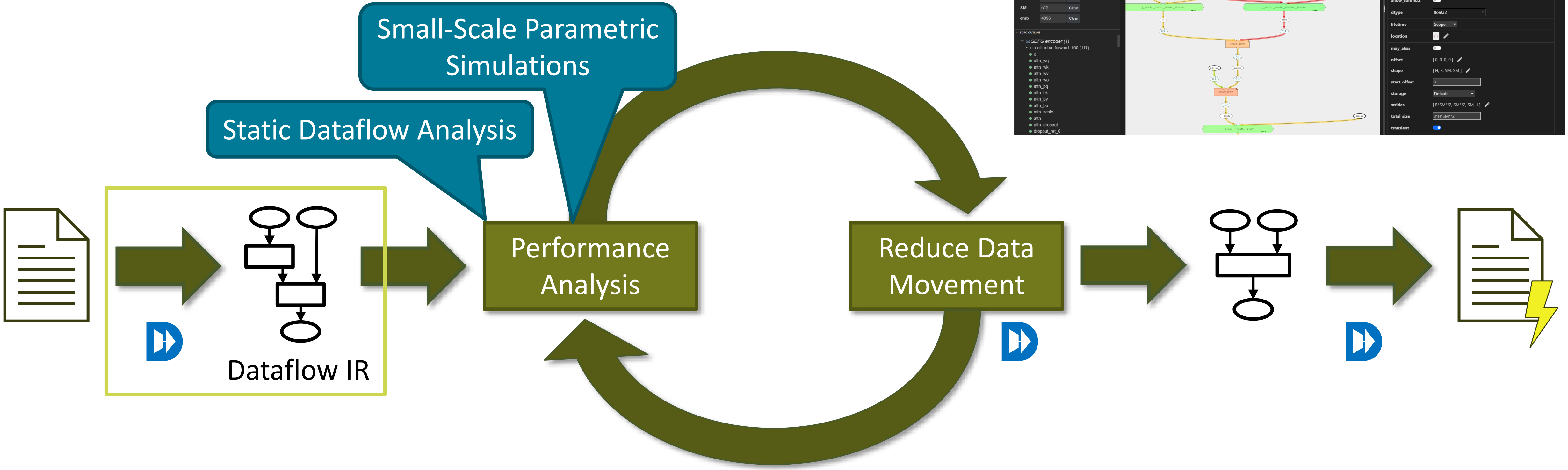


DaCe [1]
Stateful DataFlow multiGraphs (SDFGs)

[1] Ben-Nun et al., Stateful Dataflow Multigraphs: A Data-Centric Model for Performance Portability on Heterogeneous Architectures

Data Movement Optimization





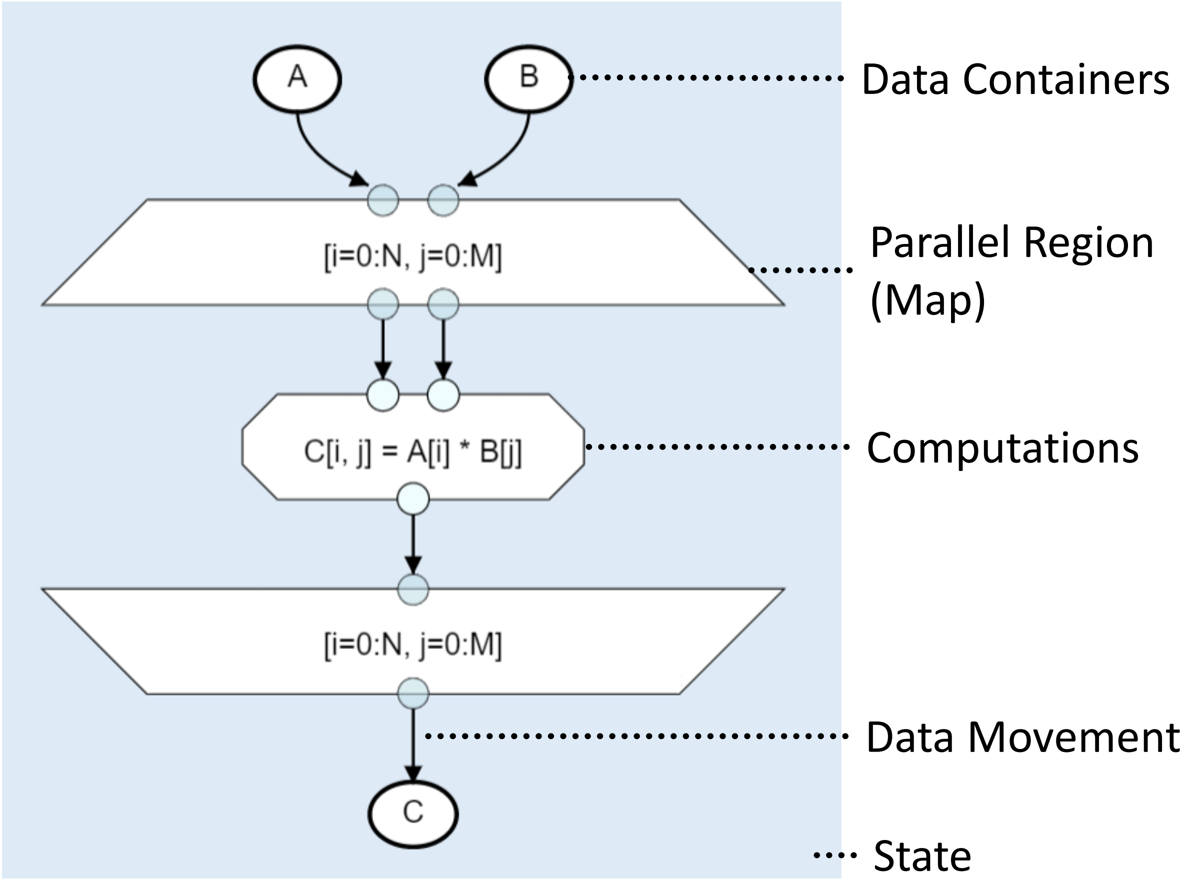
DaCe [1]
Stateful DataFlow multiGraphs (SDFGs)

[1] Ben-Nun et al., Stateful Dataflow Multigraphs: A Data-Centric Model for Performance Portability on Heterogeneous Architectures

Stateful DataFlow multiGraph (SDFG)

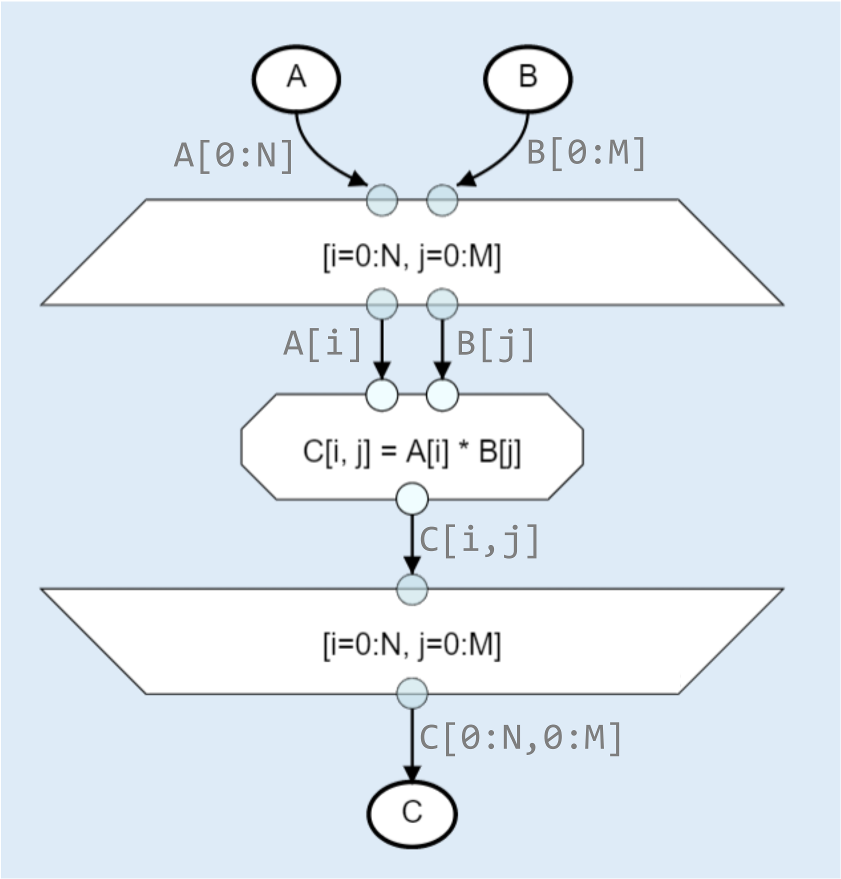
$C = A \otimes B \qquad A \in \mathbb{R}^N, B \in \mathbb{R}^M, C \in \mathbb{R}^{NxM}$

```
def outer_prod(A, B, C, N, M):
    for i in range(N):
        for j in range(M):
            C[i, j] = A[i] * B[j]
```



Static Dataflow Analysis

Data Movement Volume



Static Dataflow Analysis

Data Movement Volume

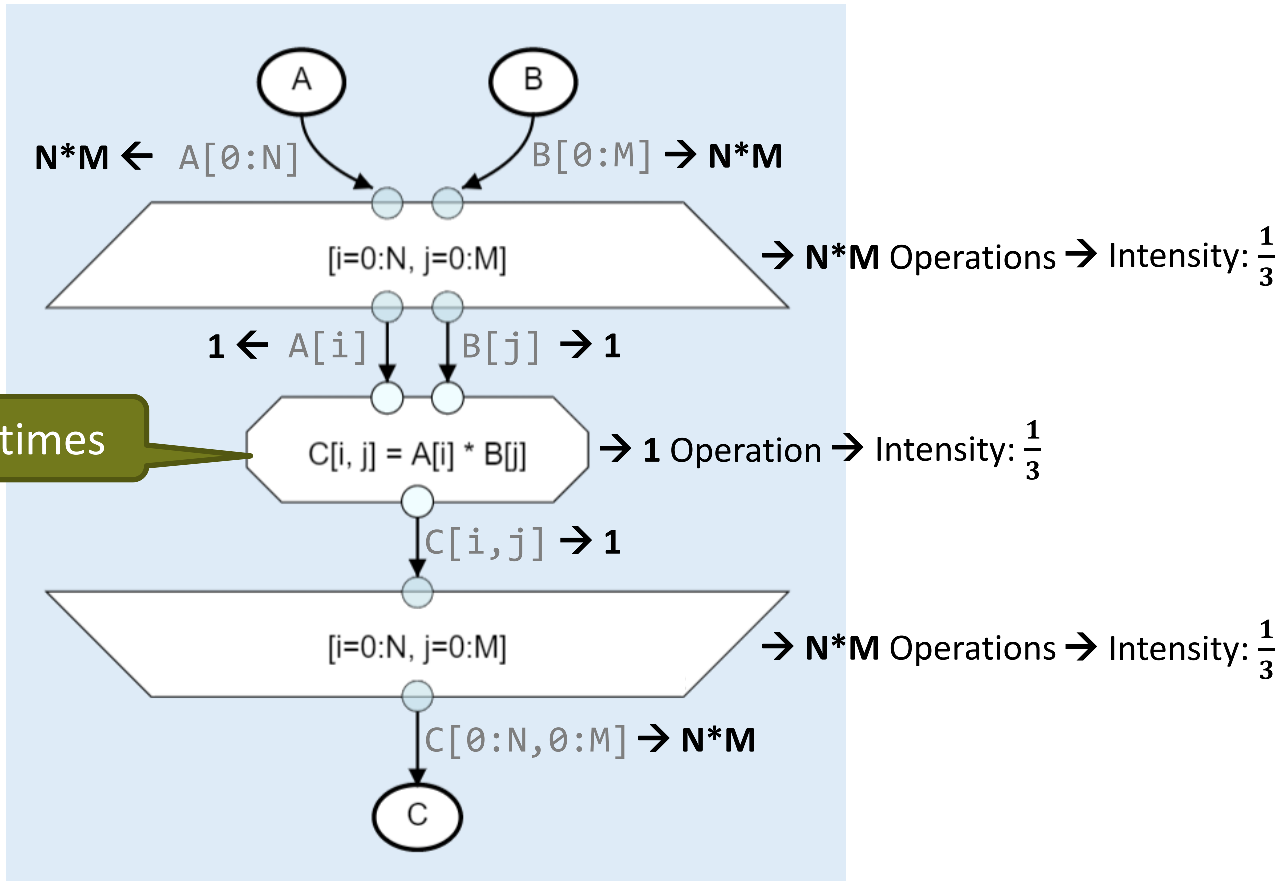
1. Derive volume for computations
2. Propagate through graph

Arithmetic Operations Count

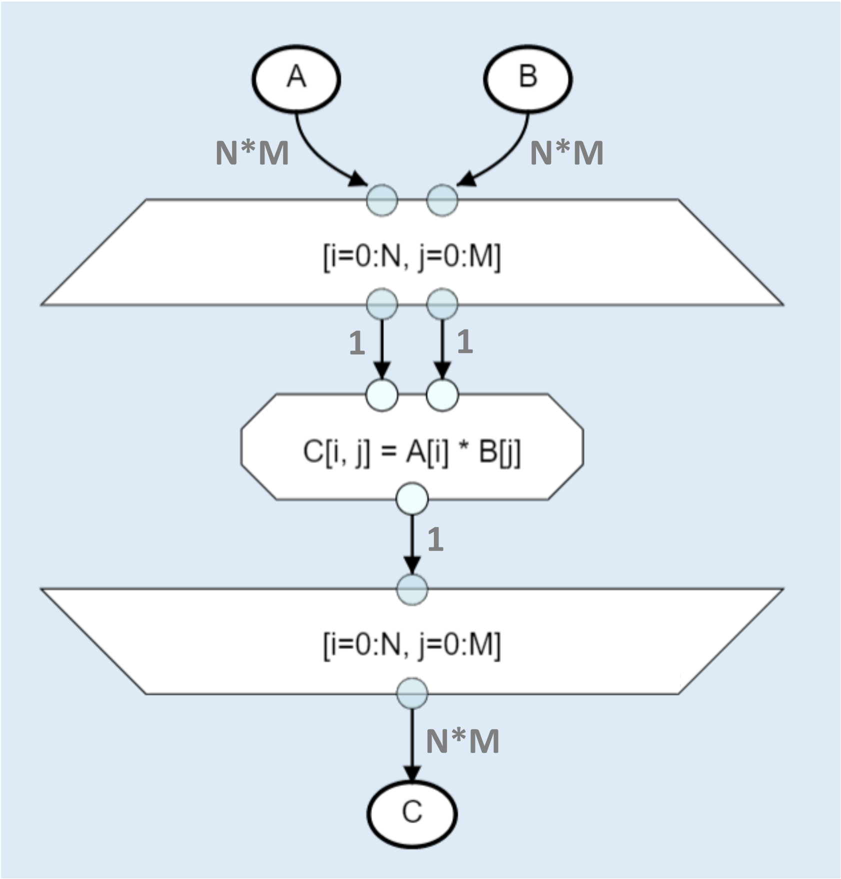
1. Count operations in AST of computations
2. Propagate through graph

Operational Intensity

Executed N*M times



Static Dataflow Analysis



N*M Operations
Intensity: $\frac{1}{3}$

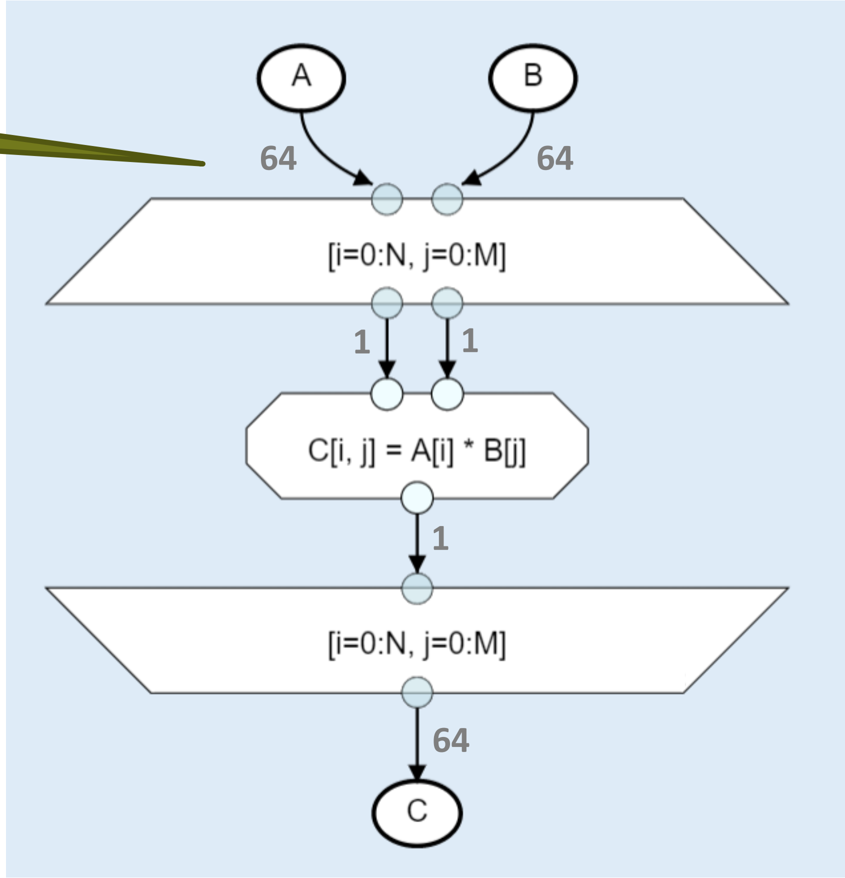
1 Operation
Intensity: $\frac{1}{3}$

N*M Operations
Intensity: $\frac{1}{3}$

Static Dataflow Analysis

$N = 8$
$M = 8$

Substitute symbols



64 Operations
Intensity: $\frac{1}{3}$

1 Operation
Intensity: $\frac{1}{3}$

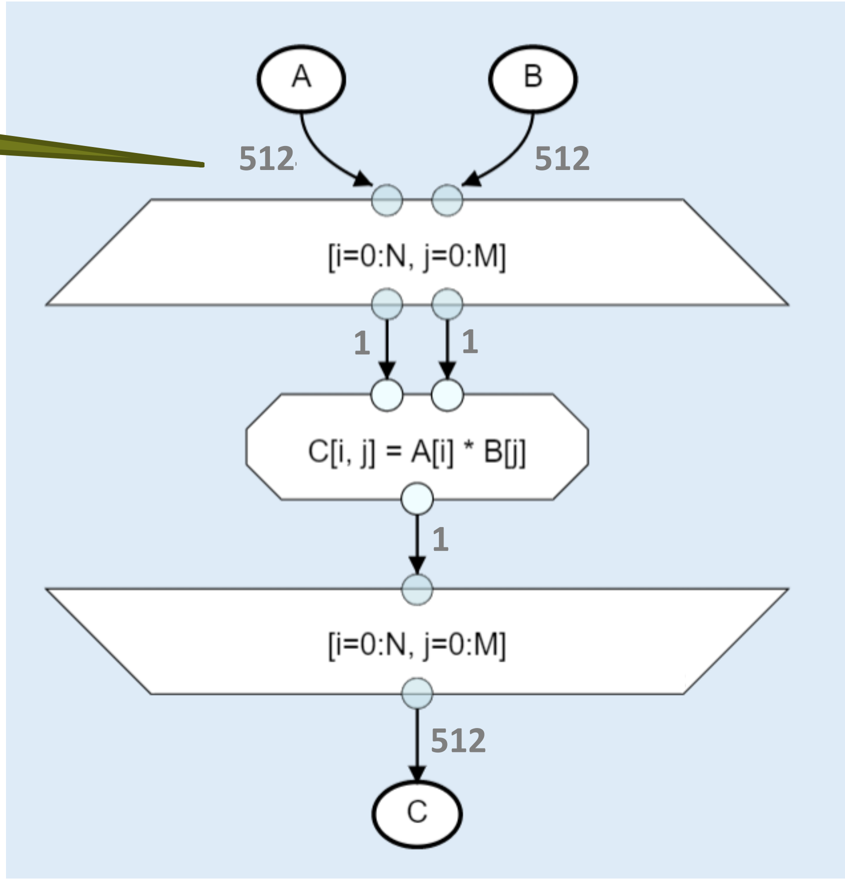
64 Operations
Intensity: $\frac{1}{3}$

Static Dataflow Analysis

$N = 8$
$M = 64$

Change symbol values to perform ***scaling analysis***

Substitute symbols



512 Operations
Intensity: $\frac{1}{3}$

1 Operation
Intensity: $\frac{1}{3}$

512 Operations
Intensity: $\frac{1}{3}$

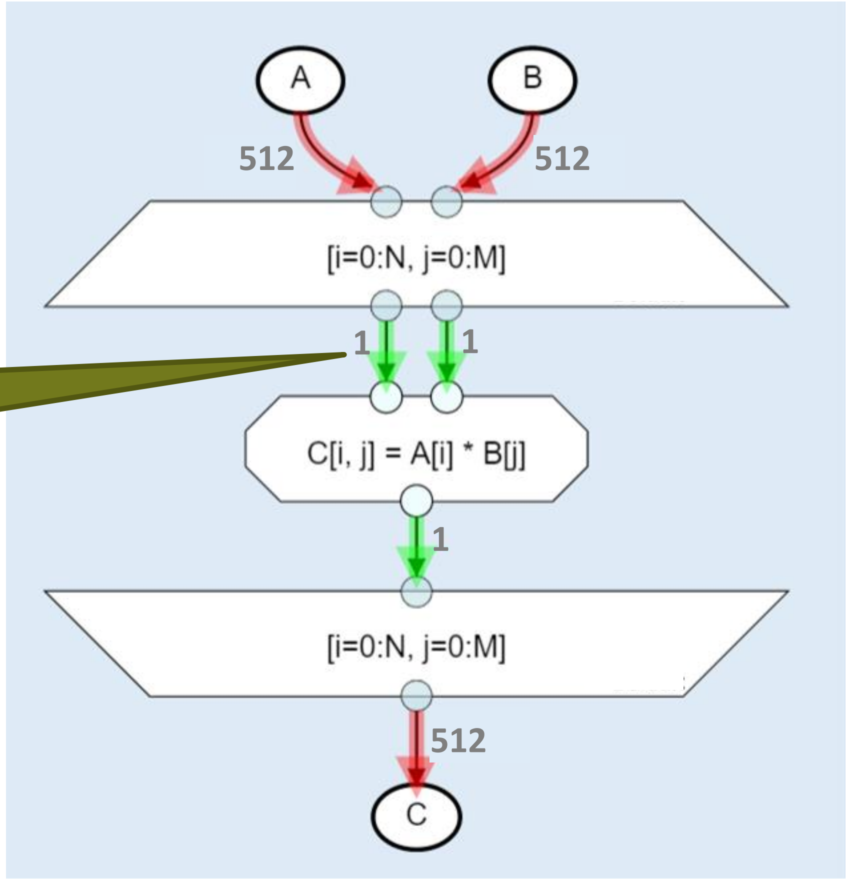
Visualization

$N = 8$
$M = 64$

Visualize data by overlaying a ***heatmap***

Low volume

High volume



512 Operations
Intensity: $\frac{1}{3}$

1 Operation
Intensity: $\frac{1}{3}$

512 Operations
Intensity: $\frac{1}{3}$

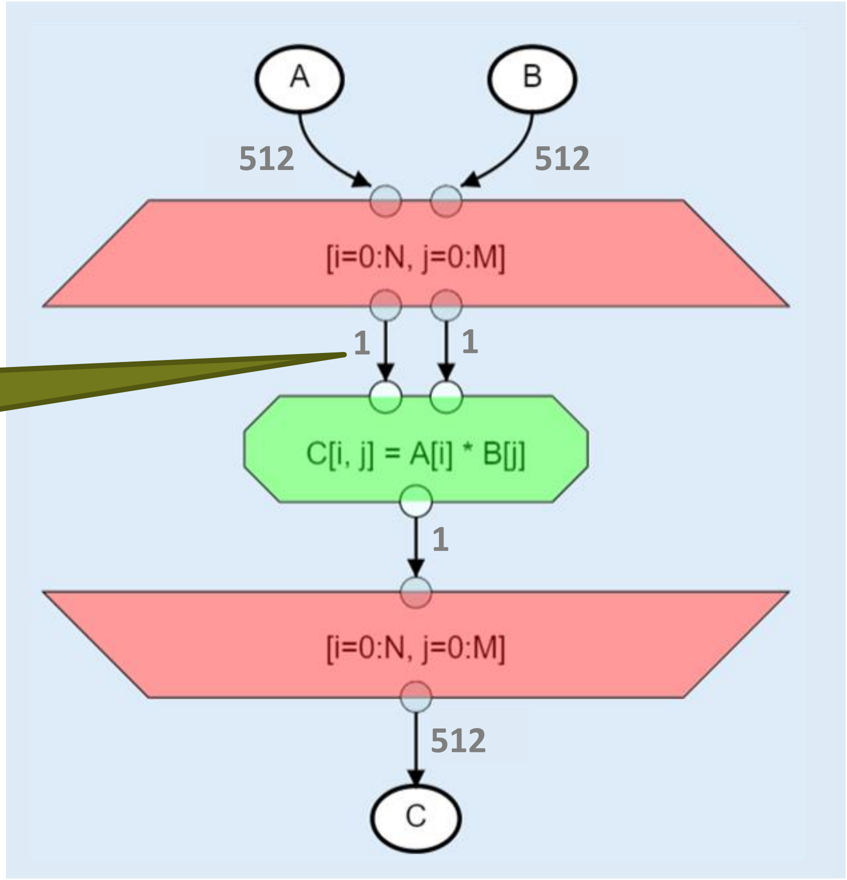
Visualization

$N = 8$
$M = 64$

Visualize data by overlaying a ***heatmap***

Low operation count

High operation count



512 Operations
Intensity: $\frac{1}{3}$

1 Operation
Intensity: $\frac{1}{3}$

512 Operations
Intensity: $\frac{1}{3}$

Visualization

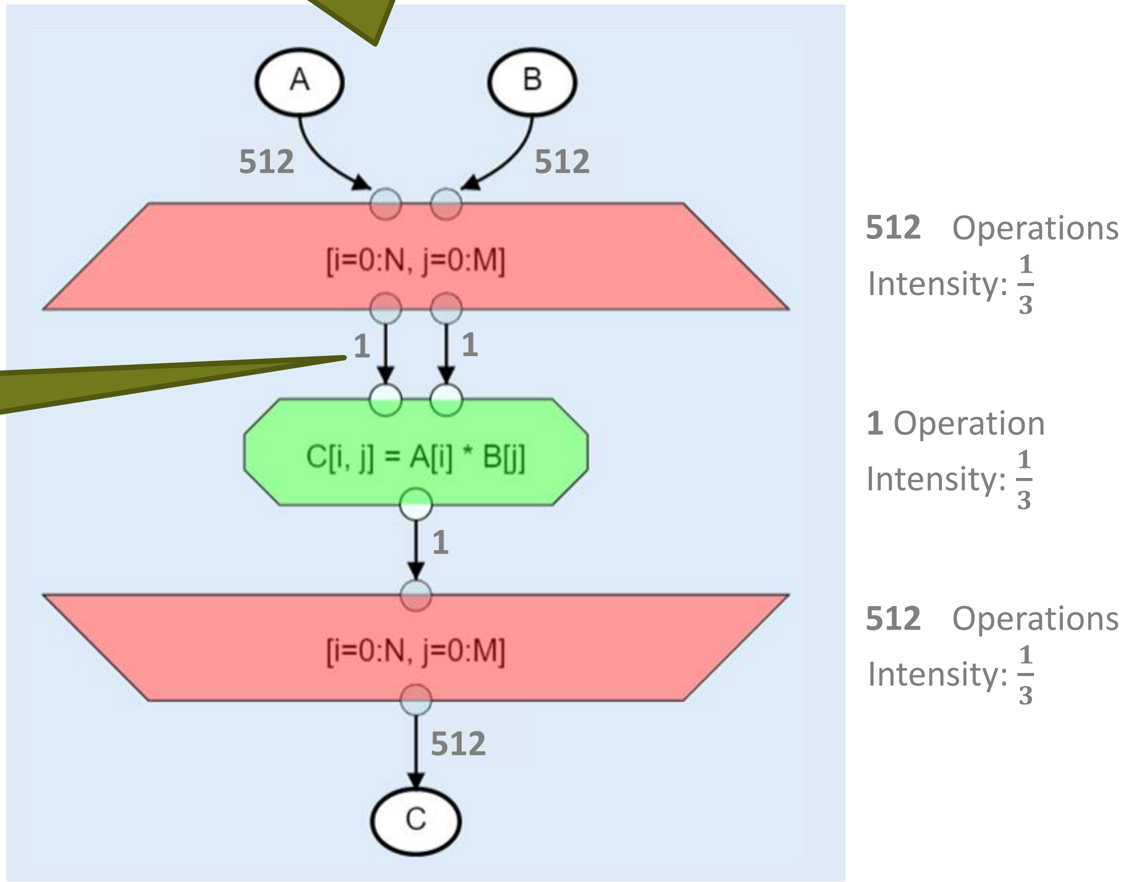
$N = 8$
$M = 64$

In-Situ performance data reduces context switching

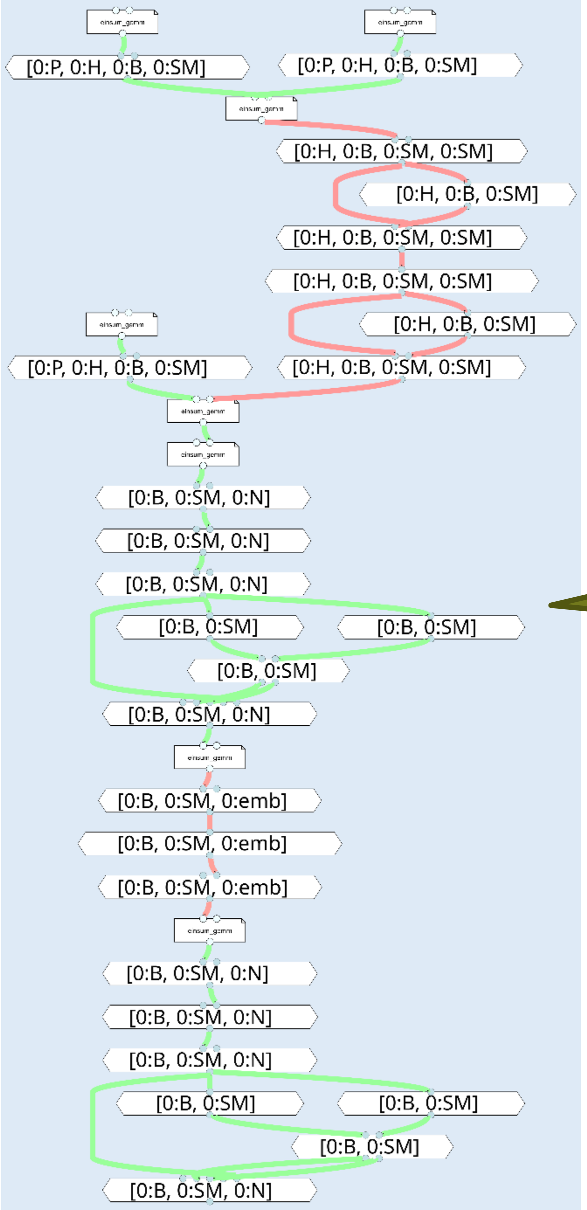
Visualize data by overlaying a ***heatmap***

Low operation count

High operation count

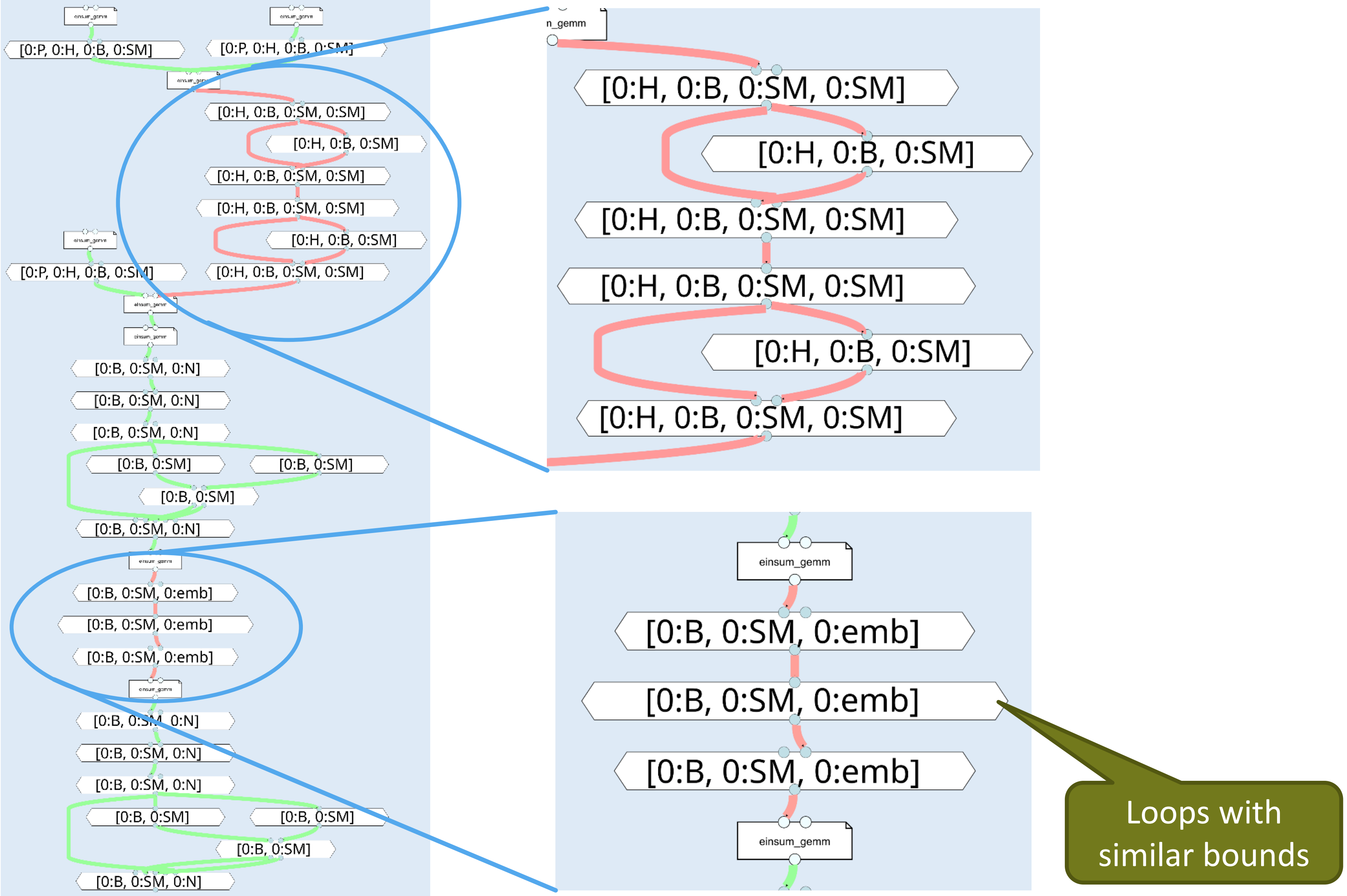


Optimizing BERT Transformer Encoder

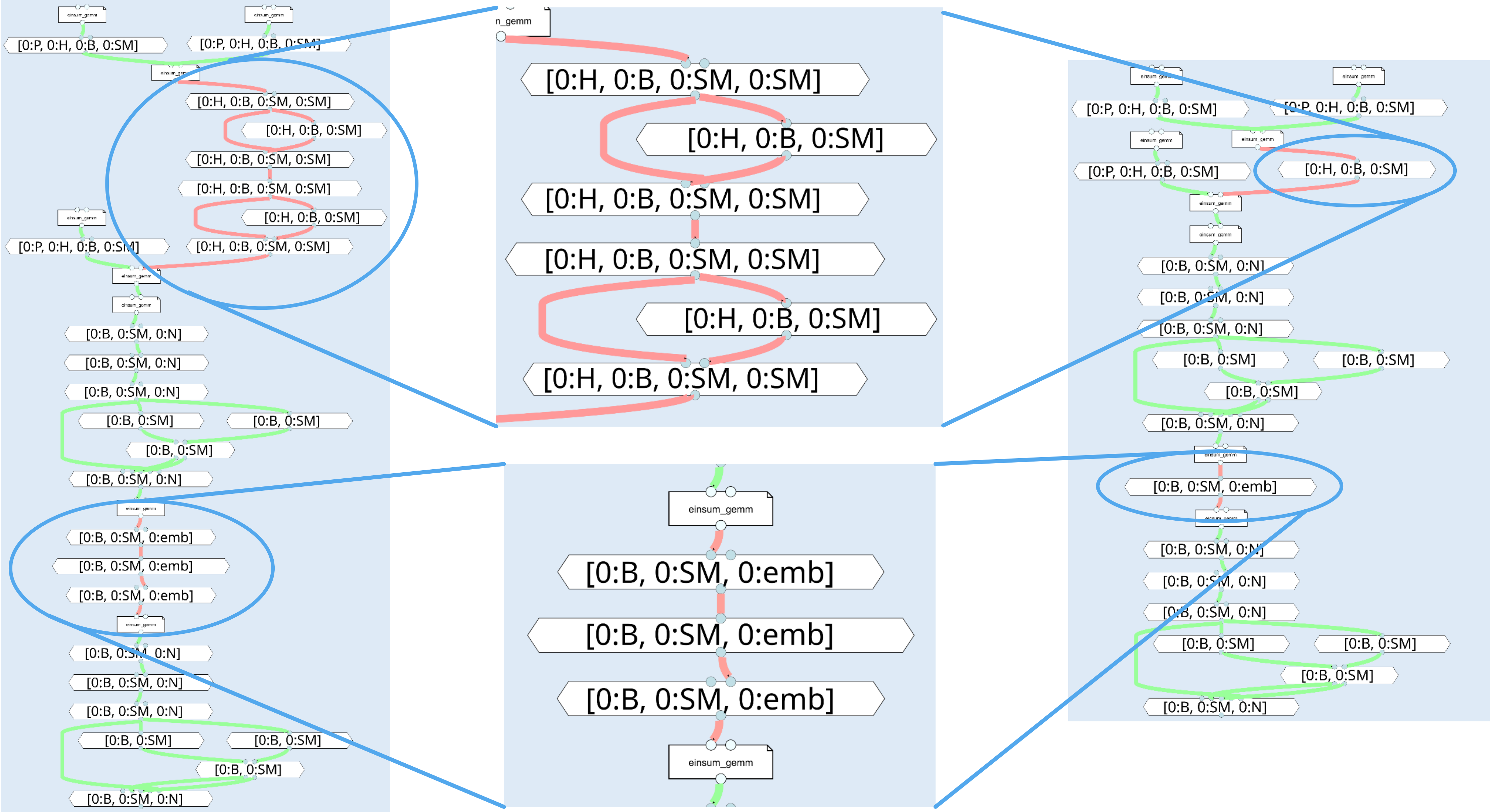


Data movement
heatmap

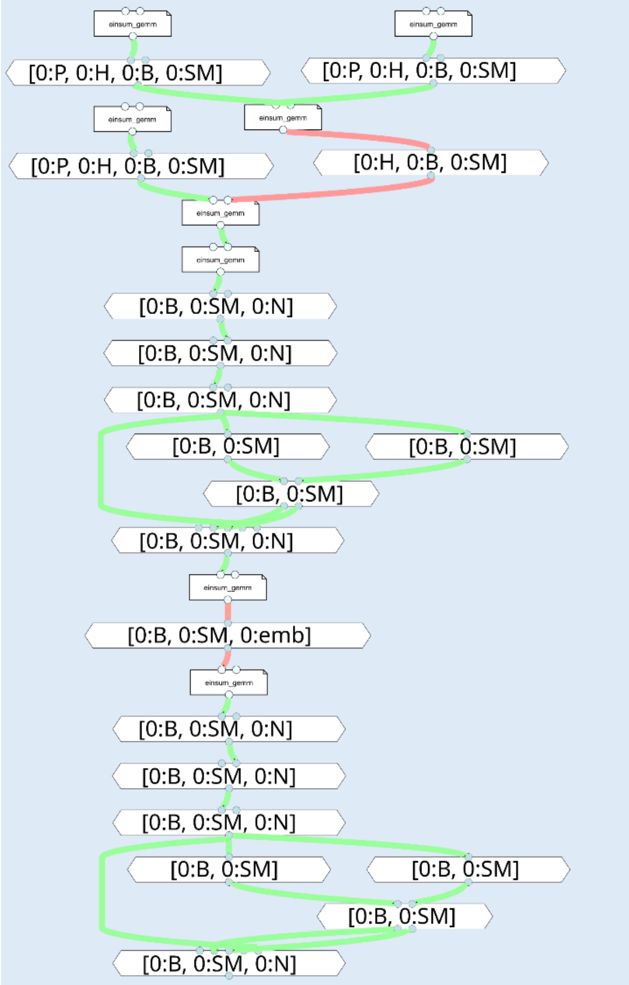
Optimizing BERT Transformer Encoder



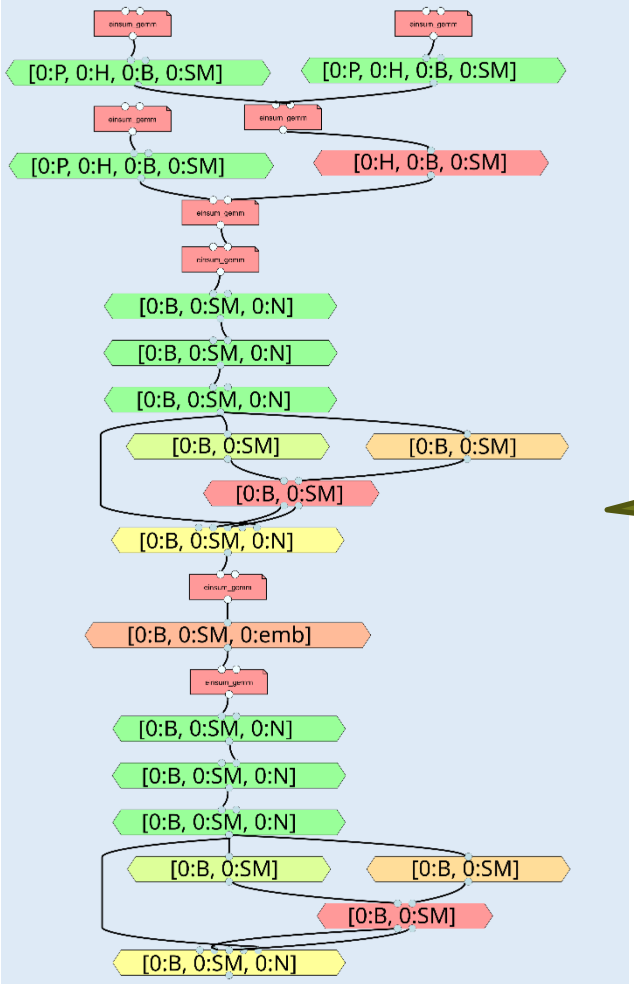
Optimizing BERT Transformer Encoder



Optimizing BERT Transformer Encoder

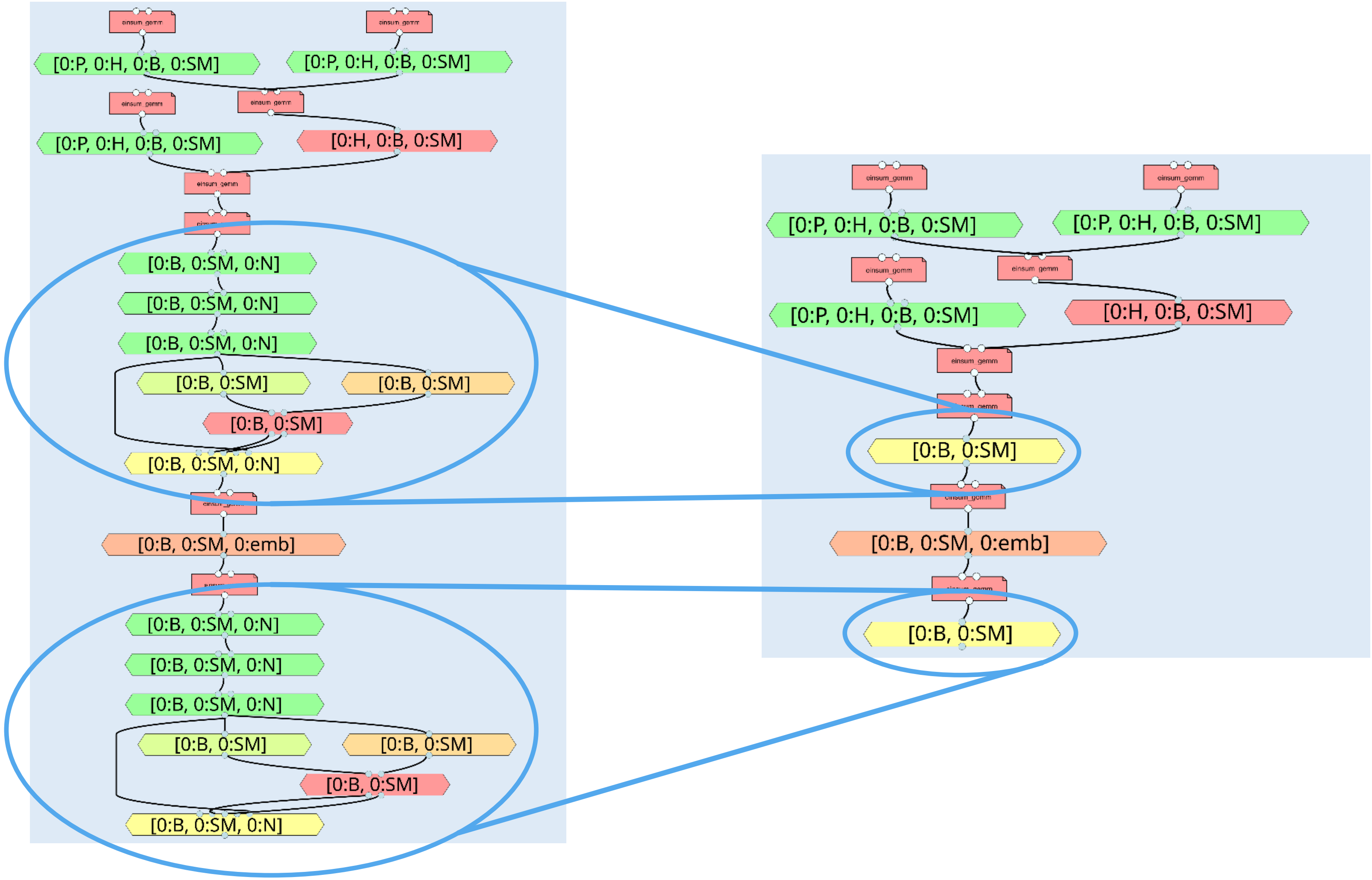


Optimizing BERT Transformer Encoder



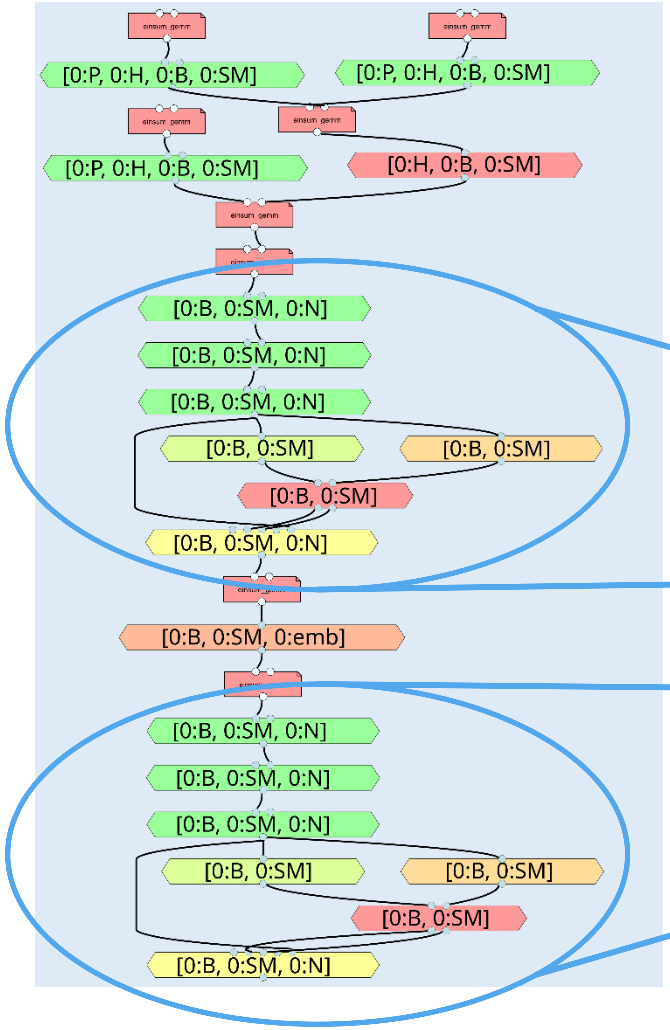
Operational intensity
heatmap

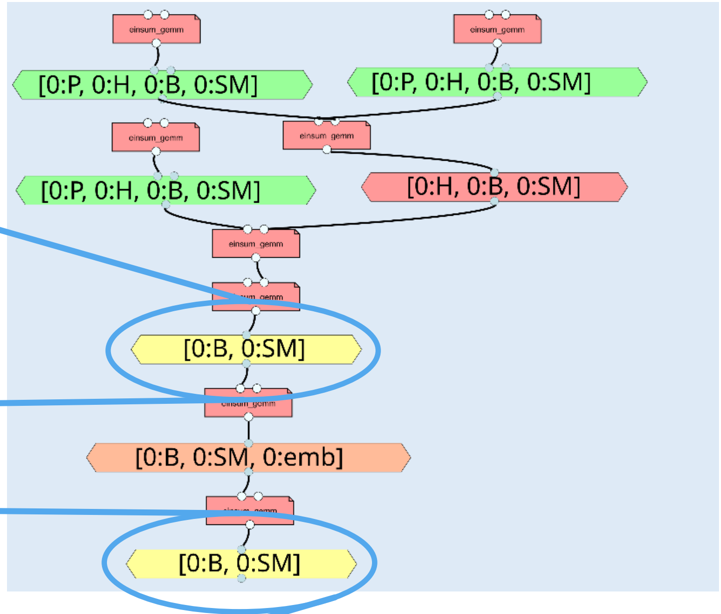
Optimizing BERT Transformer Encoder



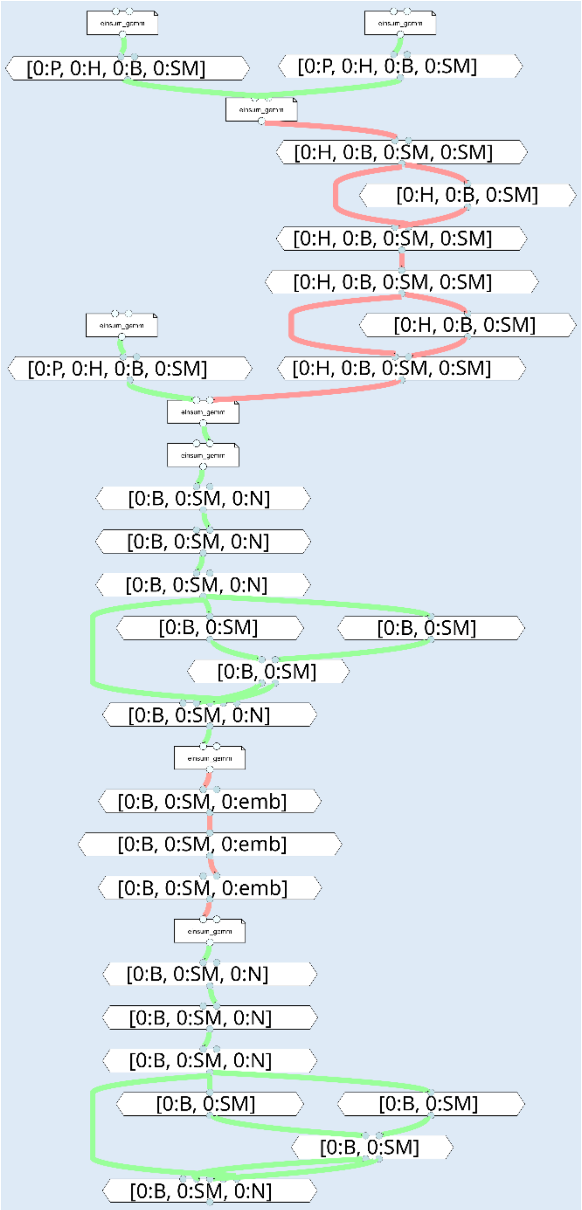
Optimizing BERT Transformer Encoder

16-core Intel Xeon Gold 6130 at 2.1 GHz, 1.5 TB RAM





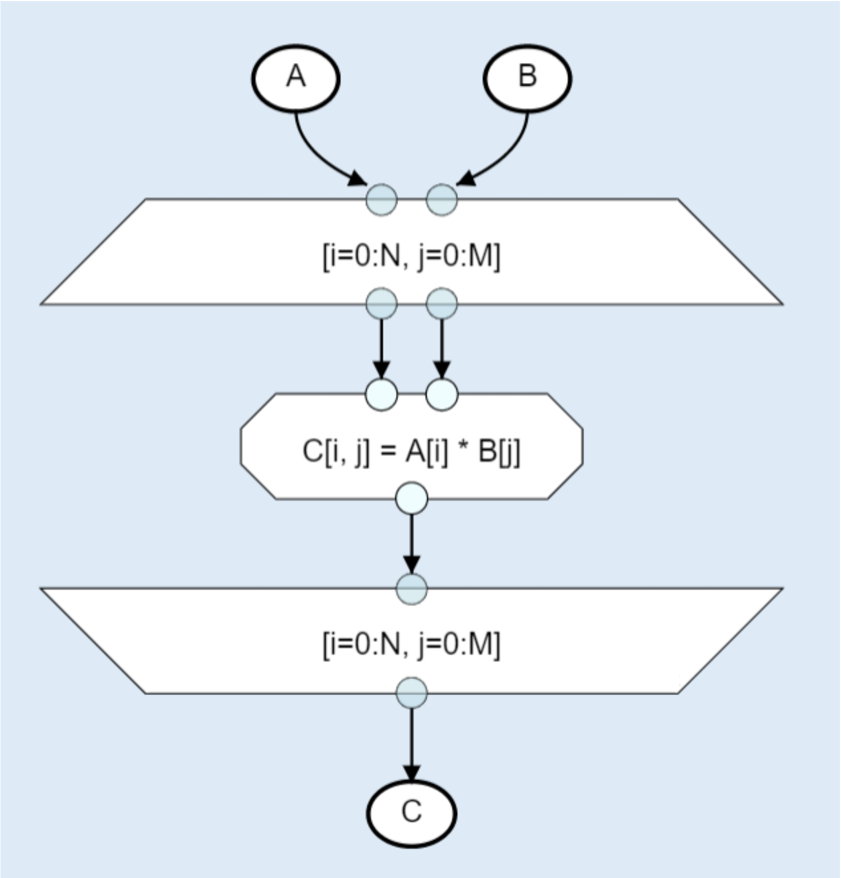
30.2x Speedup



Close-Up Reuse Analysis

Simulate data reuse behavior

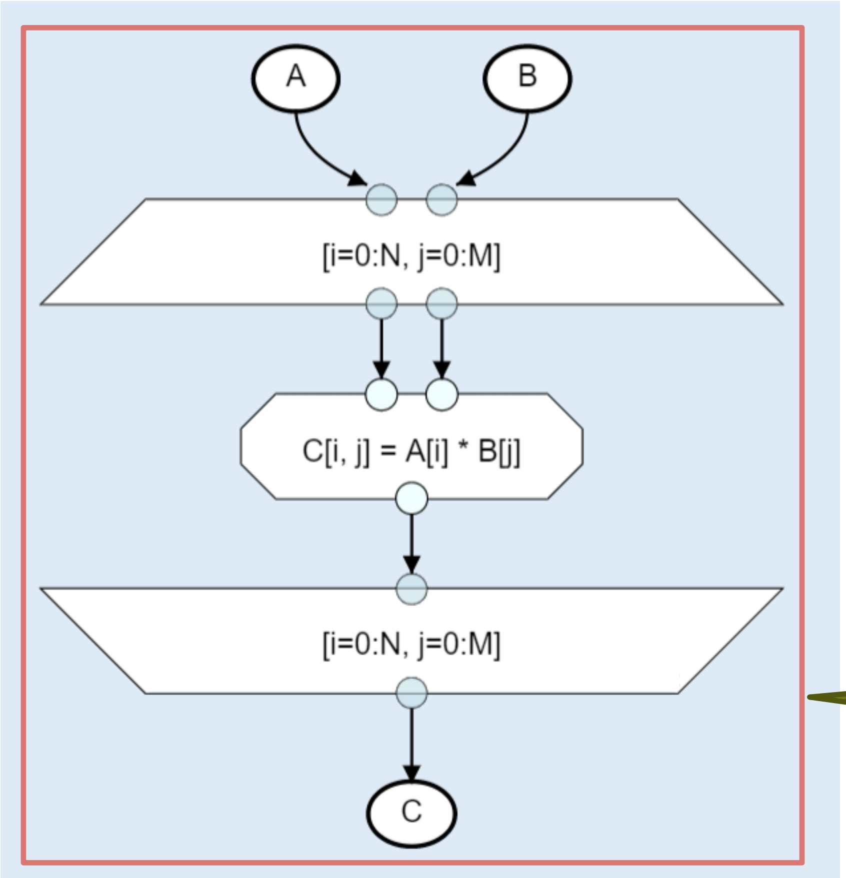
$C = A \otimes B \qquad A \in \mathbb{R}^N, B \in \mathbb{R}^M, C \in \mathbb{R}^{NxM}$



Close-Up Reuse Analysis

Simulate data reuse behavior

$C = A \otimes B \qquad A \in \mathbb{R}^N, B \in \mathbb{R}^M, C \in \mathbb{R}^{NxM}$



Specify program region

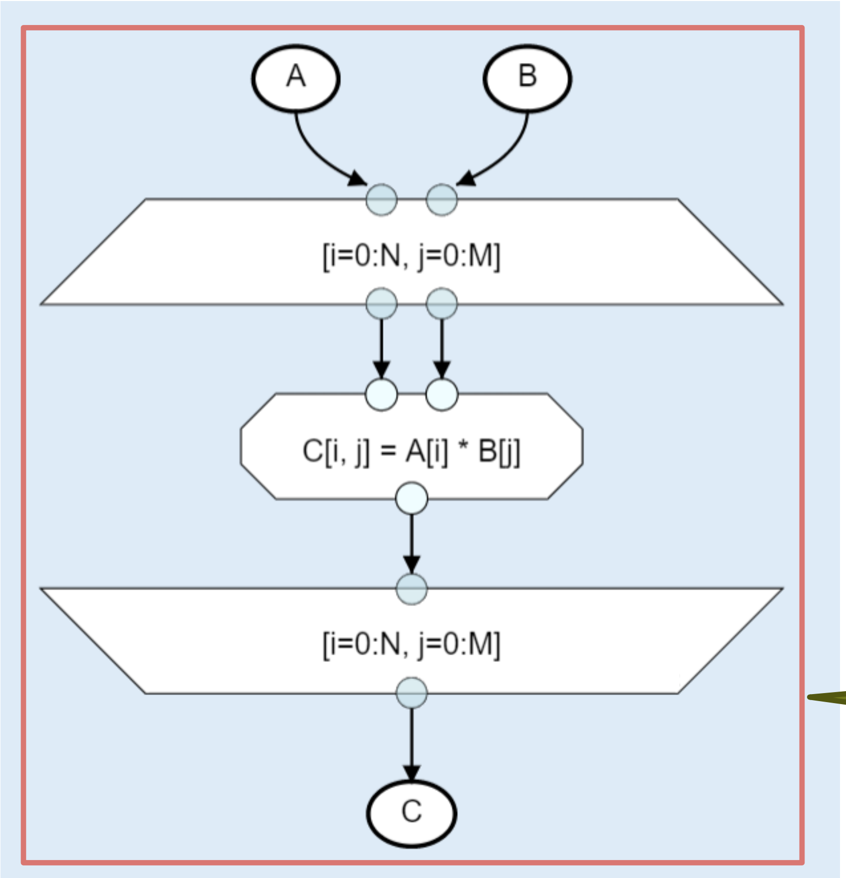
Close-Up Reuse Analysis

Simulate data reuse behavior

$C = A \otimes B \qquad A \in \mathbb{R}^3, B \in \mathbb{R}^4, C \in \mathbb{R}^{3x4}$

Specify small example input parameters

$N = 3$
$M = 4$



Specify program region

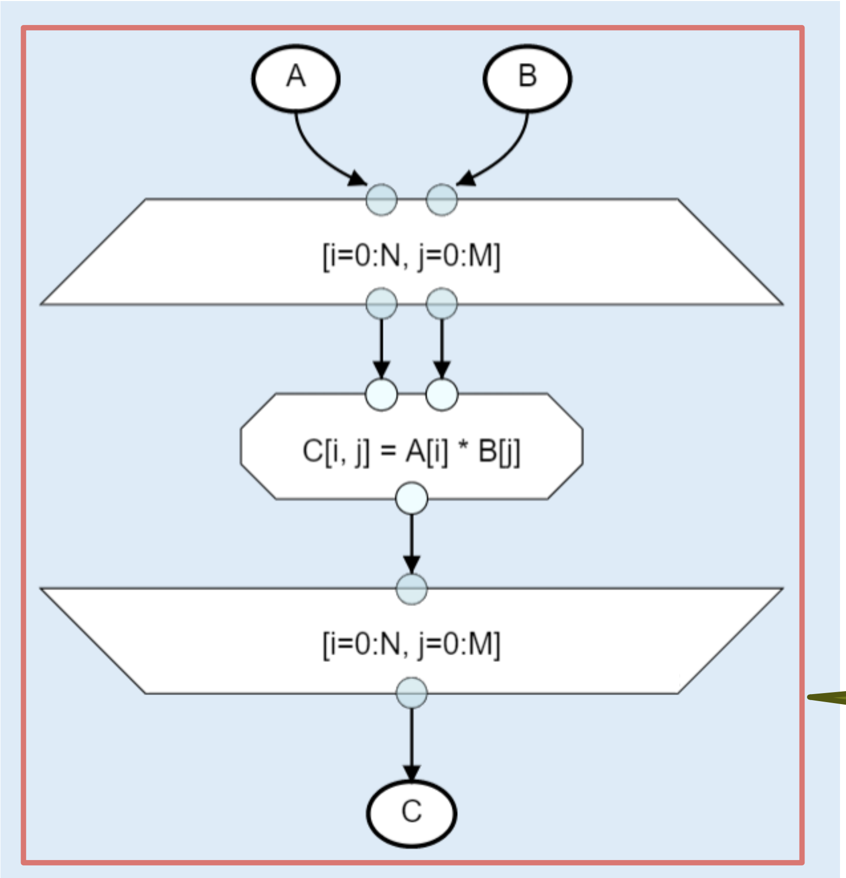
Close-Up Reuse Analysis

Simulate data reuse behavior

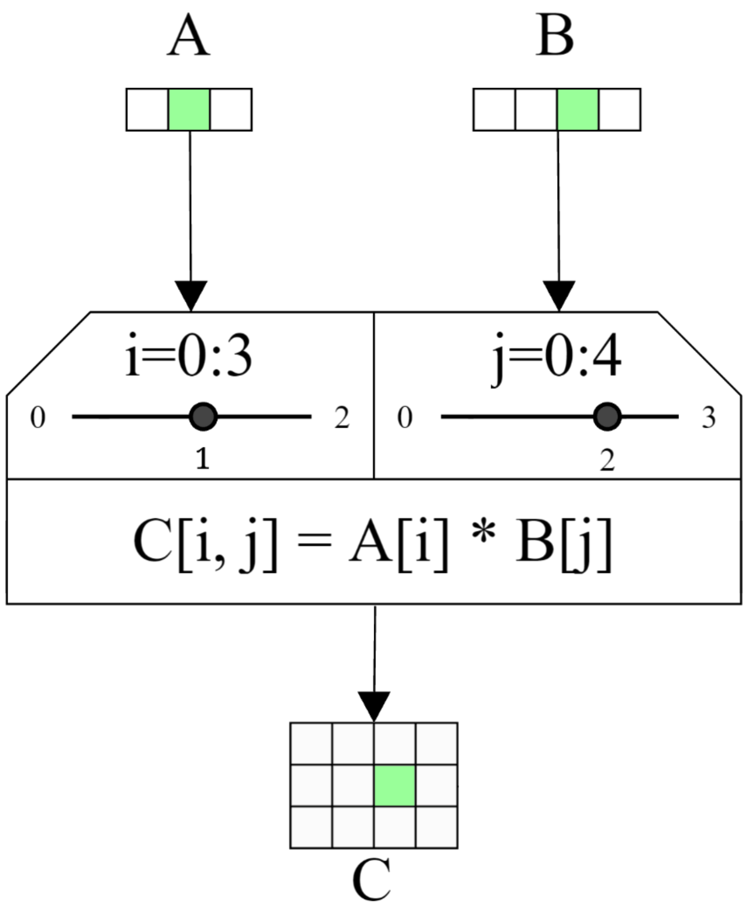
$C = A \otimes B$ $\quad A \in \mathbb{R}^3, B \in \mathbb{R}^4, C \in \mathbb{R}^{3x4}$

Specify small example input parameters

$N = 3$
$M = 4$



Specify program region



Visualizing High-Dimensional Data

$w \in \mathbb{R}^{C_{out} \times C_{in} \times K_y \times K_X}$

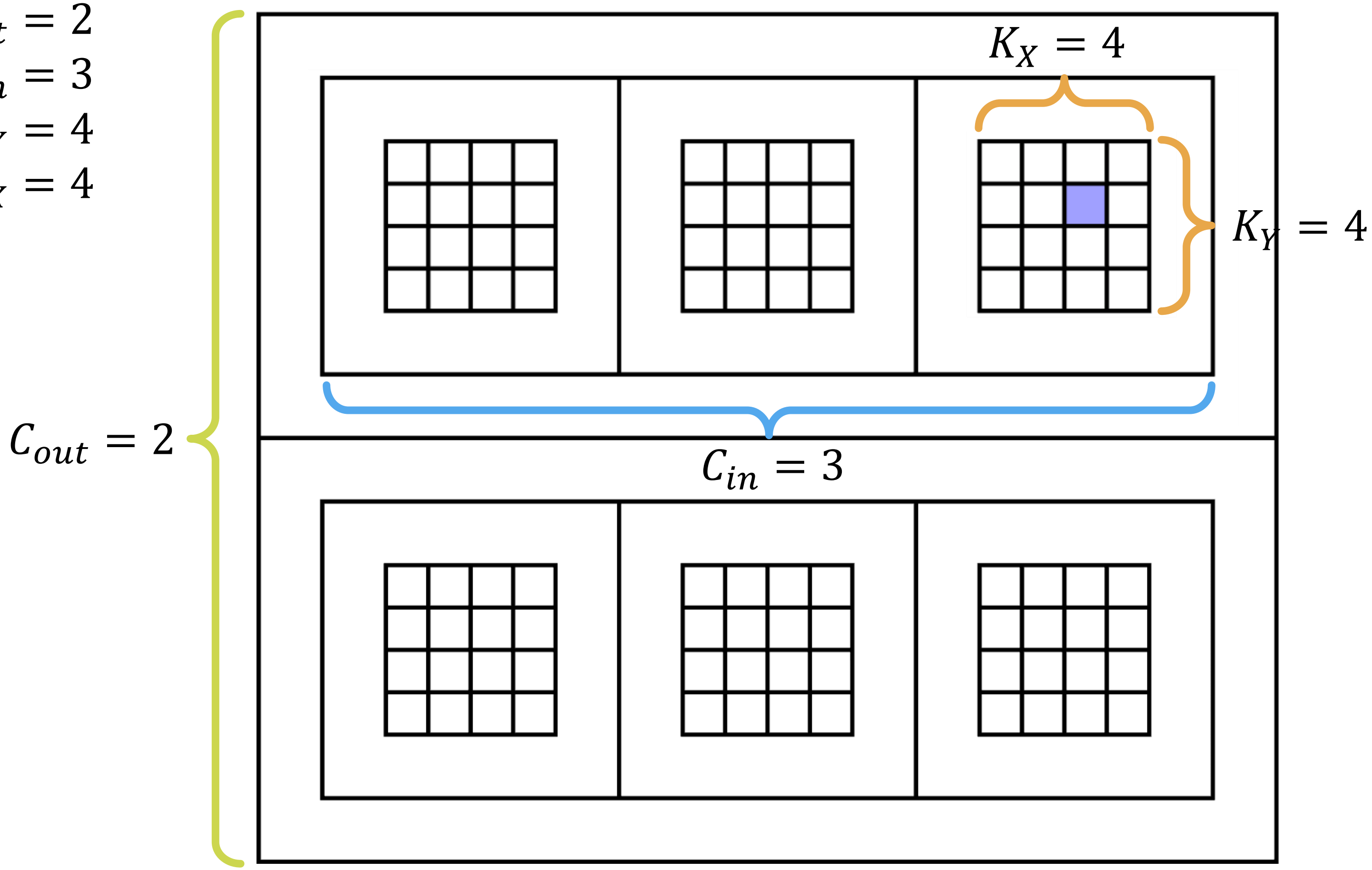
4D weights of a convolution

$C_{out} = 2$
$C_{in} = 3$
$K_Y = 4$
$K_X = 4$

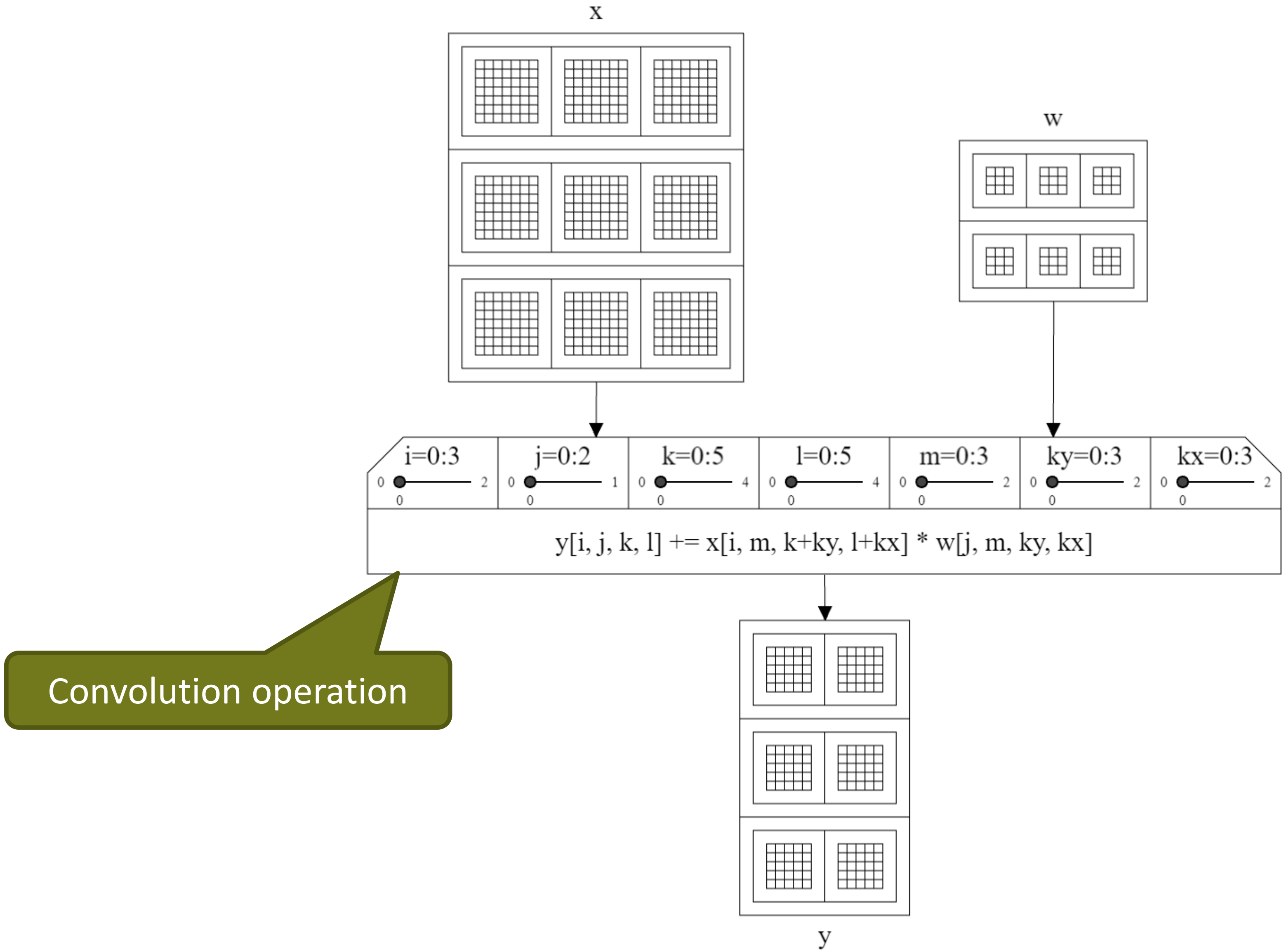
Visualizing High-Dimensional Data

$w \in \mathbb{R}^{C_{out} \times C_{in} \times K_y \times K_X}$

$C_{out} = 2$
$C_{in} = 3$
$K_Y = 4$
$K_X = 4$

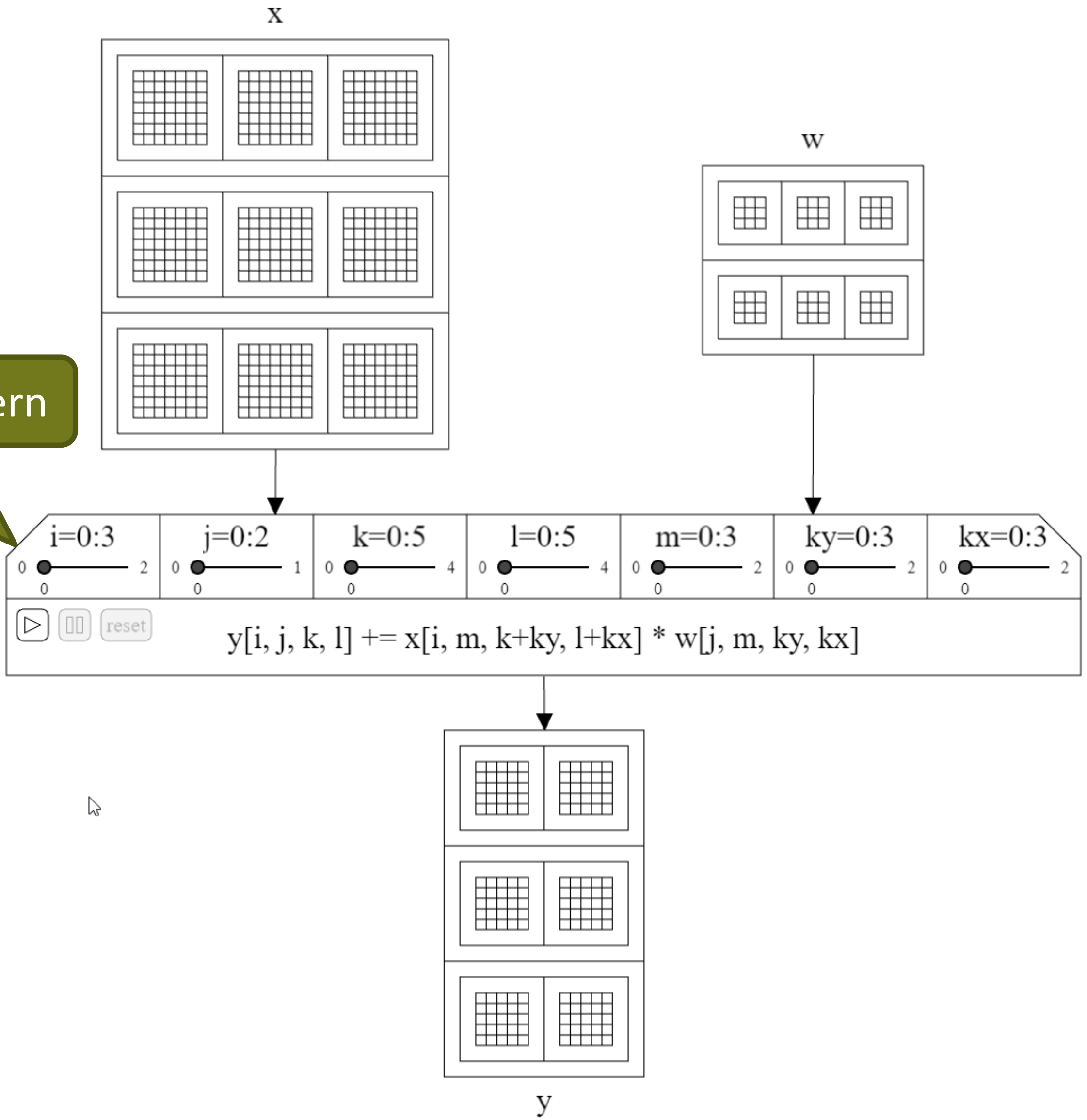


Access Pattern Simulation

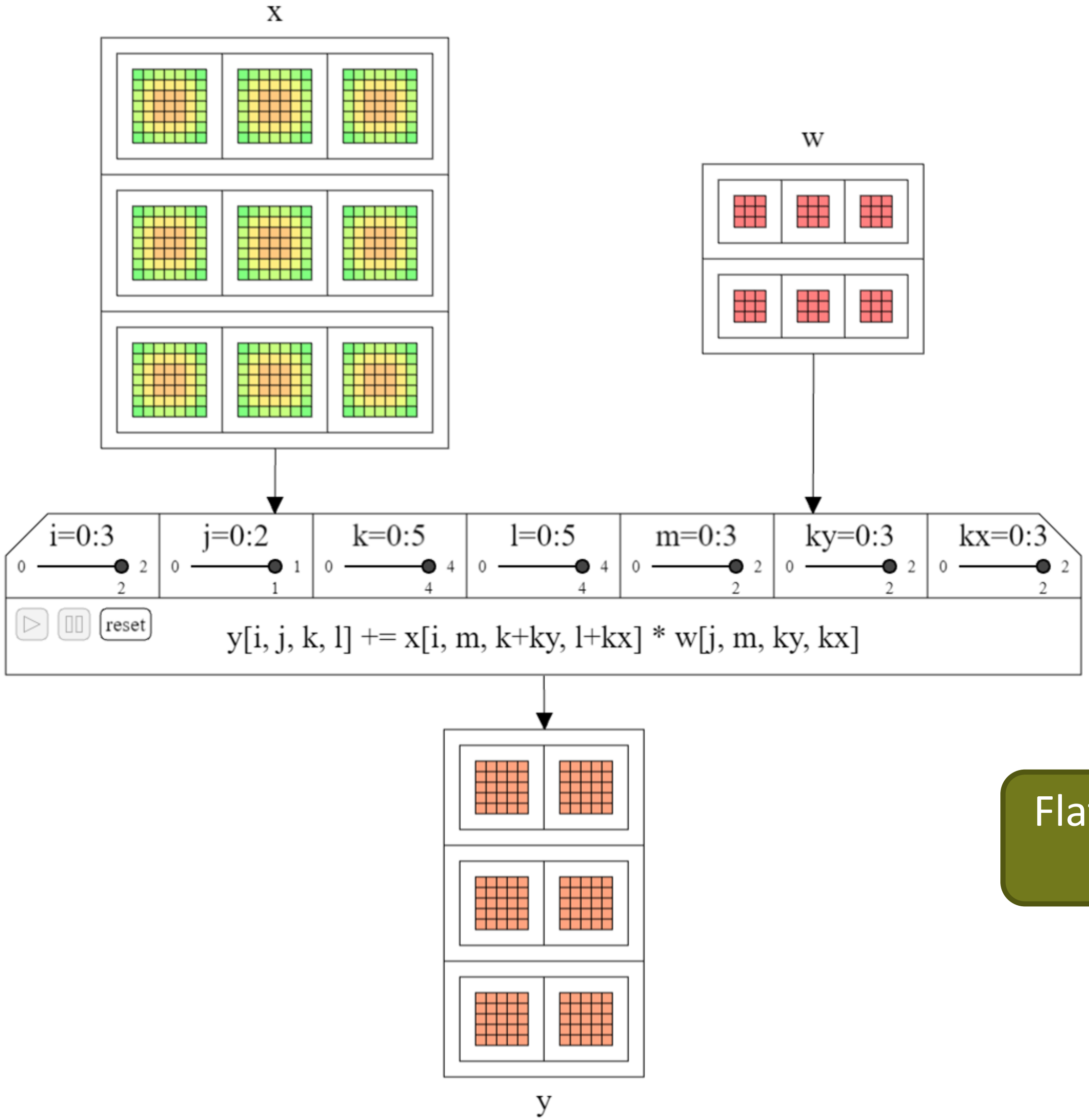


Access Pattern Simulation

Visually play back access pattern

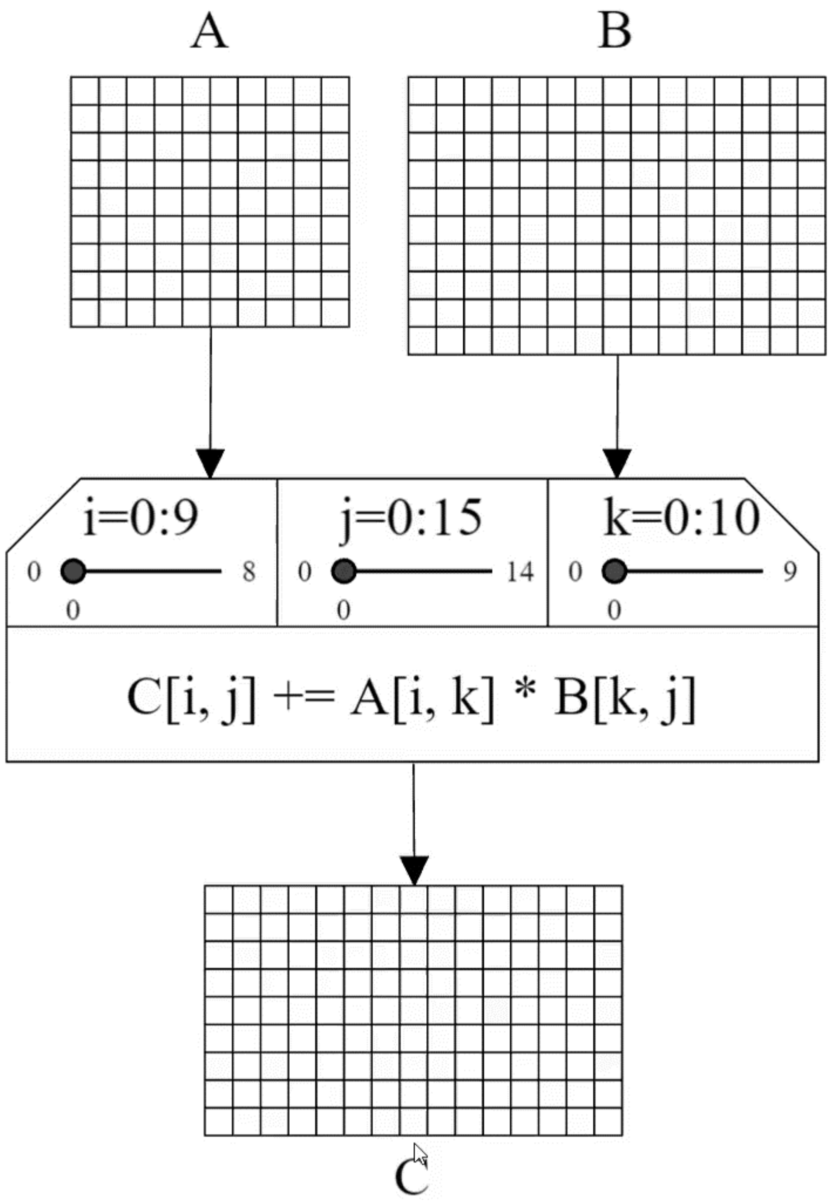


Access Pattern Simulation



Flatten time dimension with heatmap

Access Pattern Simulation

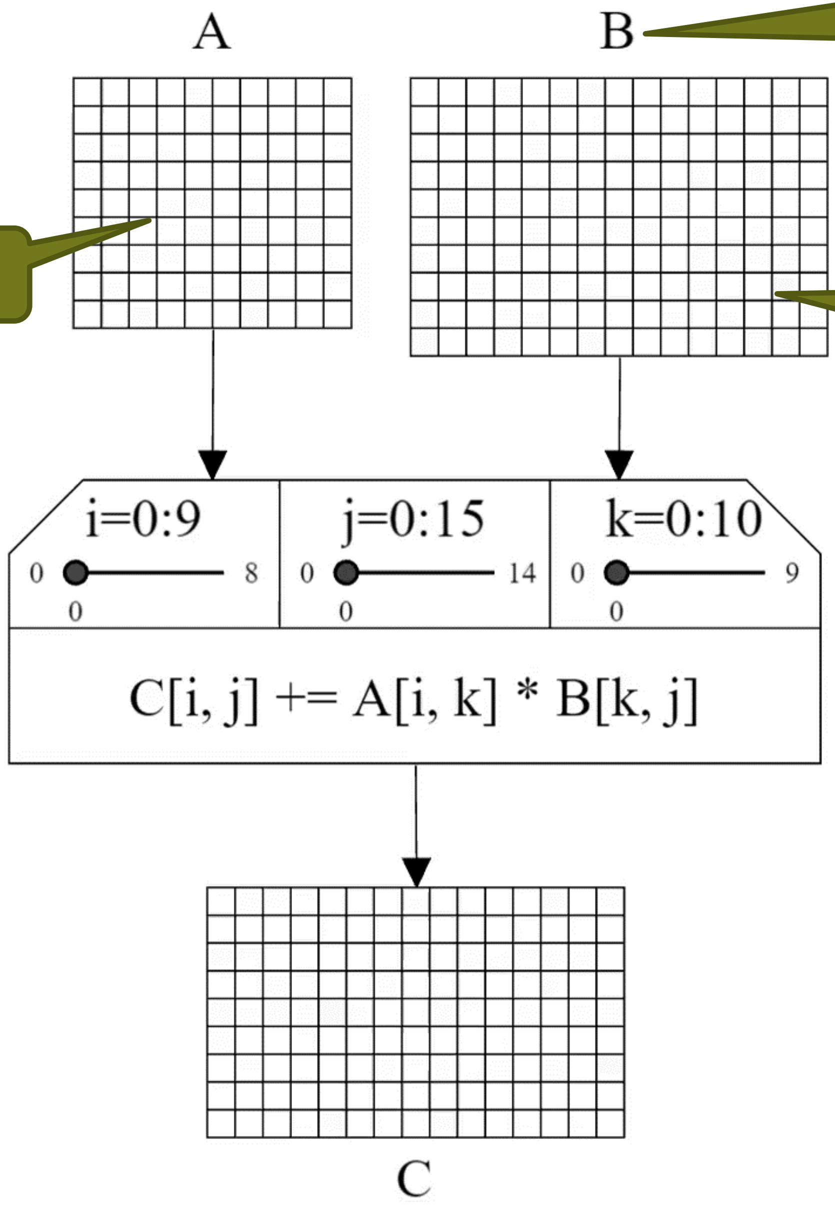


Data Layout Visualization

float64 / double → element size = 8 bytes

Exposes data layout

Determine cache line using strides, line size, and element size

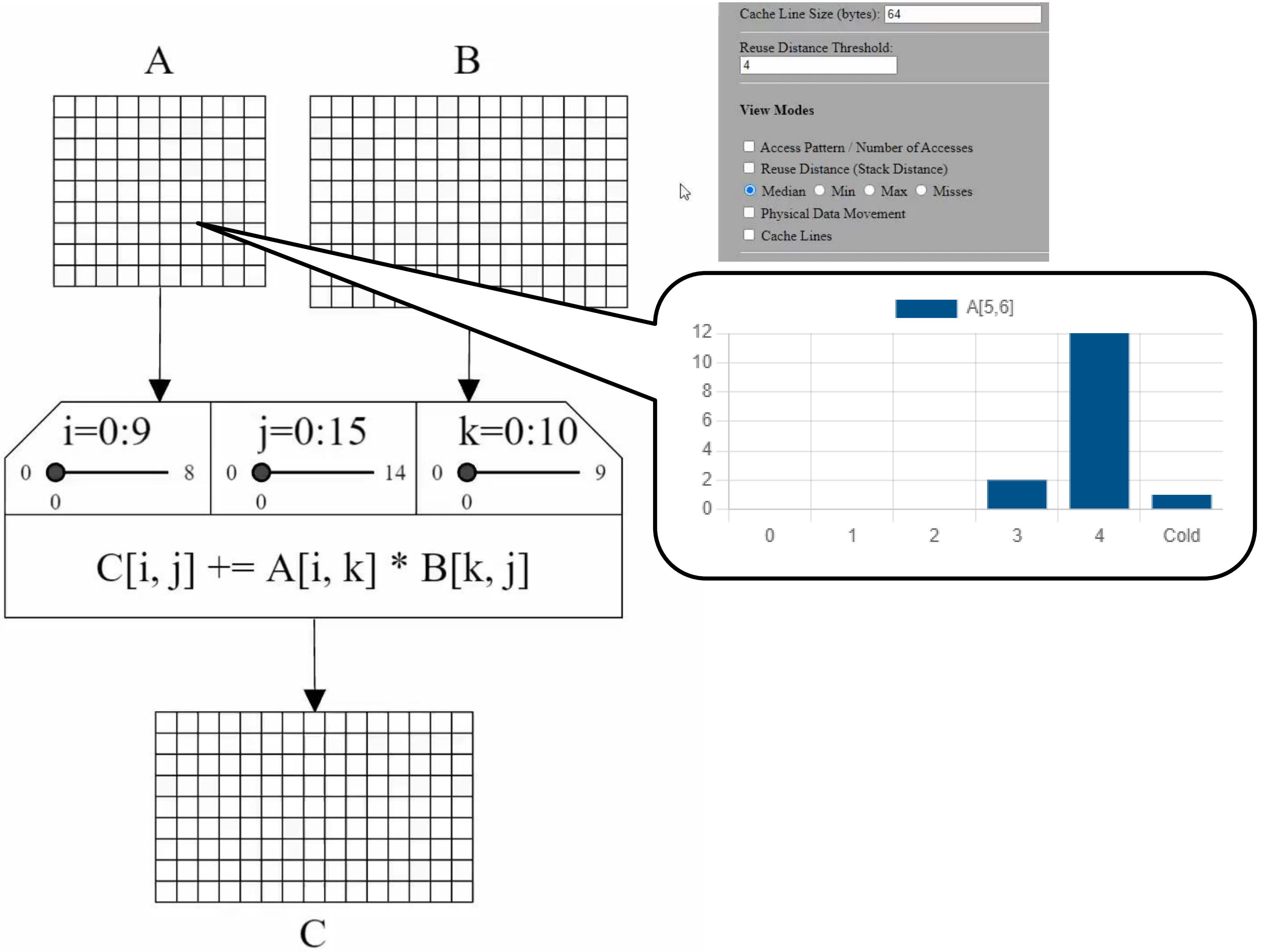


Data layout & access pattern → **spatial locality**

Temporal Locality

Stack distance, cache line granularity

Accesses to *unique* addresses since last reference



Cache Misses

1. Cold miss

   Access with stack distance = ∞

2. Capacity miss

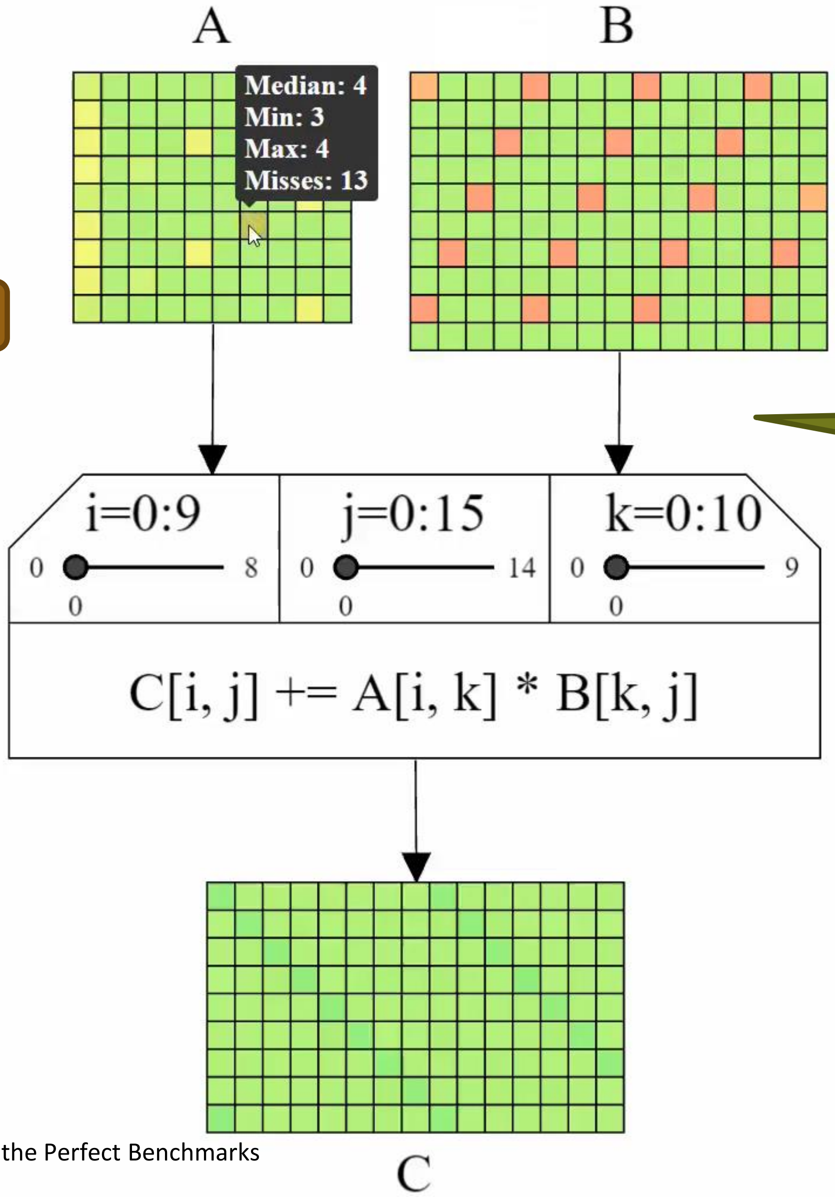
   Assuming LRU

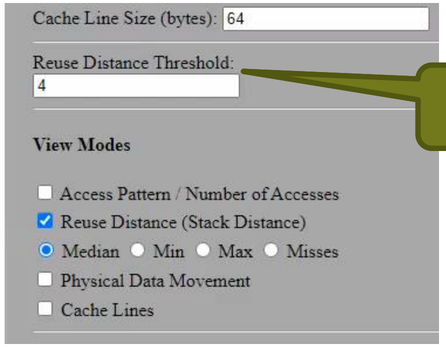
   Access with stack distance ≥ t_d stack distance threshold

3. Conflict miss

   Not counted in fully-associative cache

   Calculations generalizeable [1][2]





$t_d = 5$

Physical data movement = #misses x cache line size

[1] McKinley and Temam, Quantifying Loop Nest Locality Using SPEC'95 and the Perfect Benchmarks
[2] Beyls and D'Hollander, Reuse distance as a metric for cache behavior

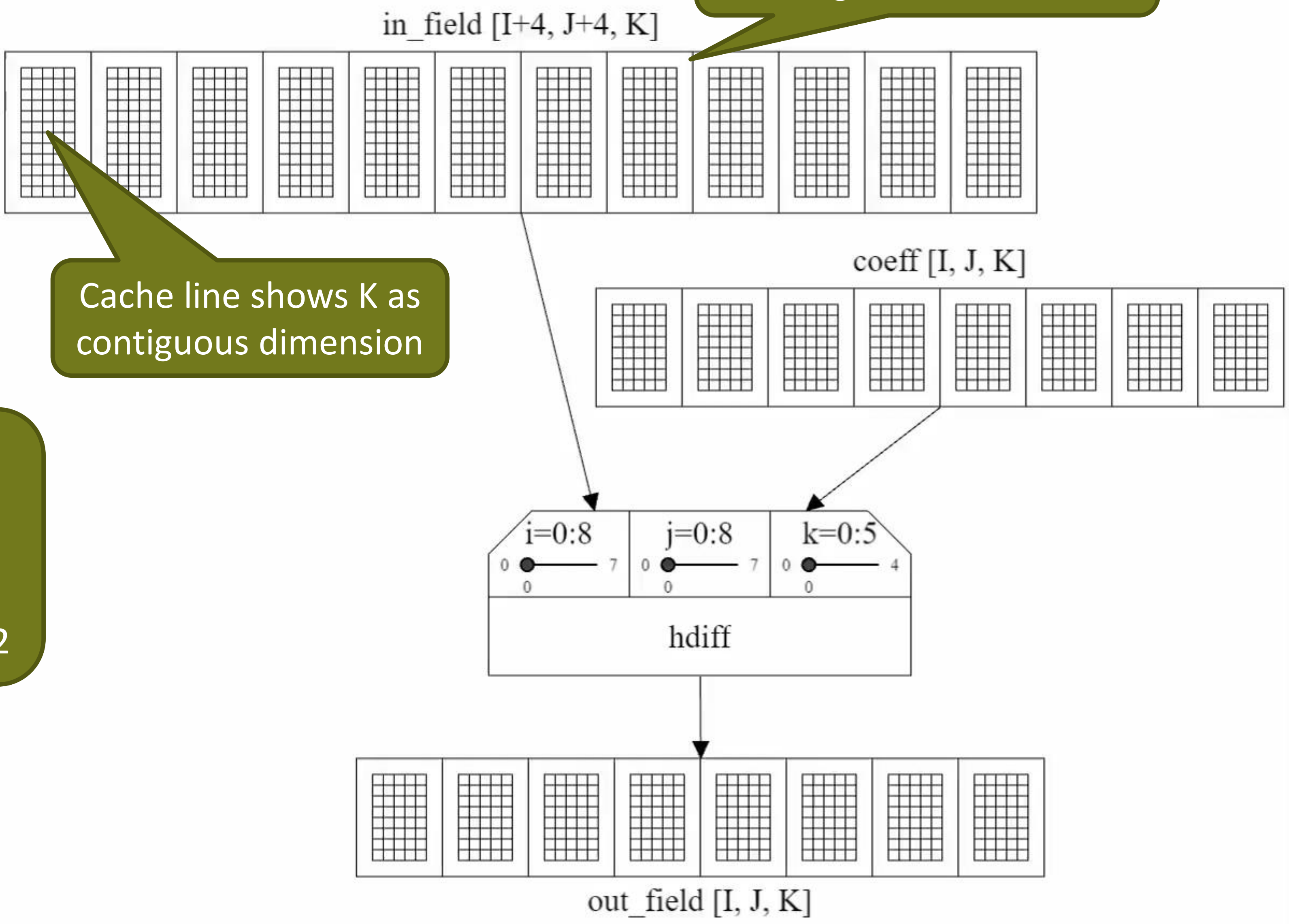
Stencil Optimization

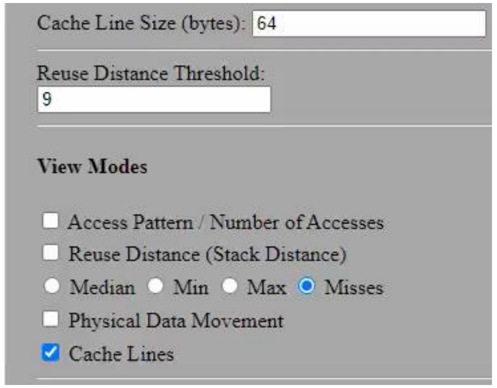
$I = 8$
$J = 8$
$K = 5$

Original sizes:
$I = 256$
$J = 256$
$K = 160$
Scaling Factor x32

Accesses spread over non-contiguous dimension

Cache line shows K as contiguous dimension





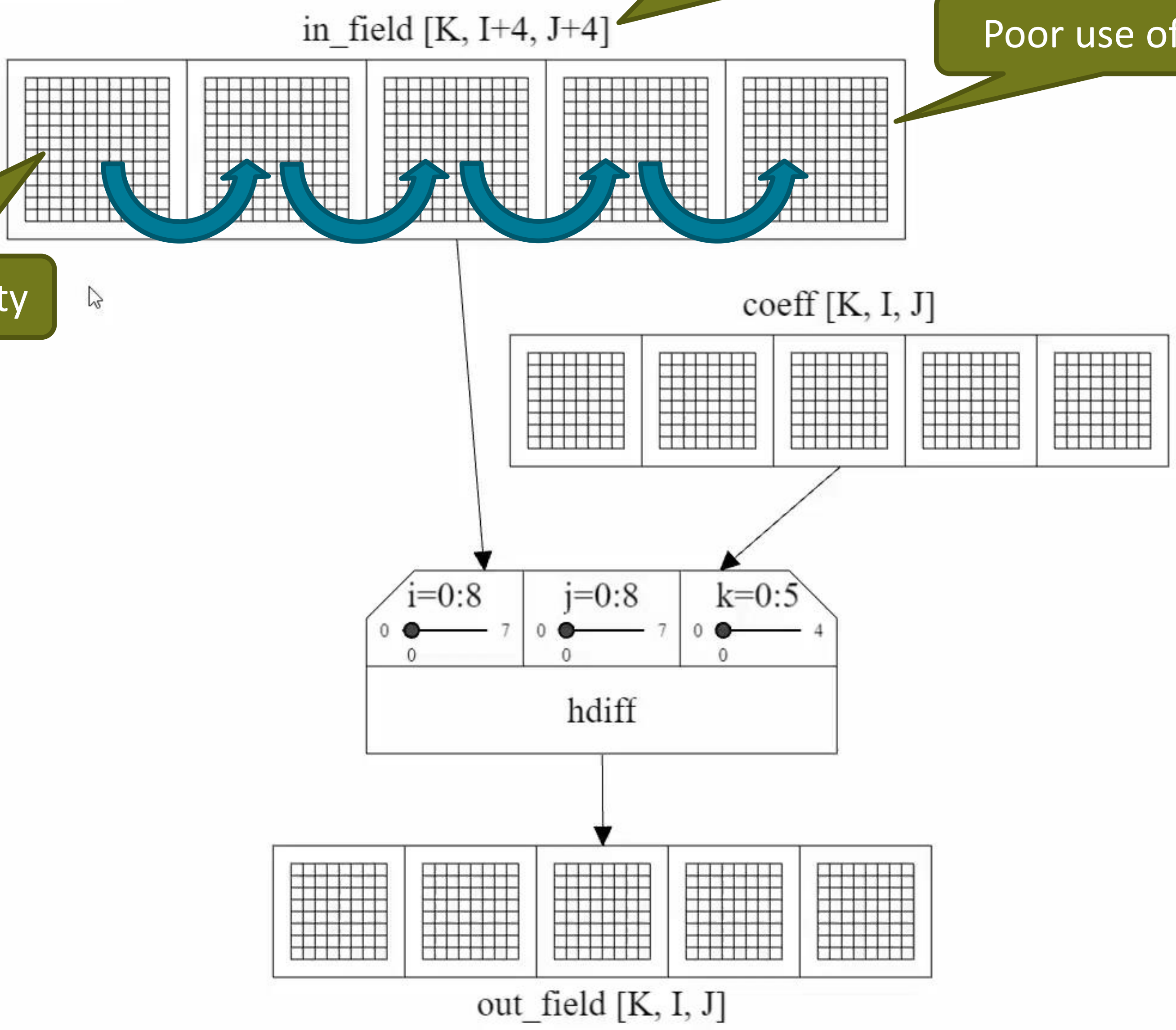
Stencil Optimization

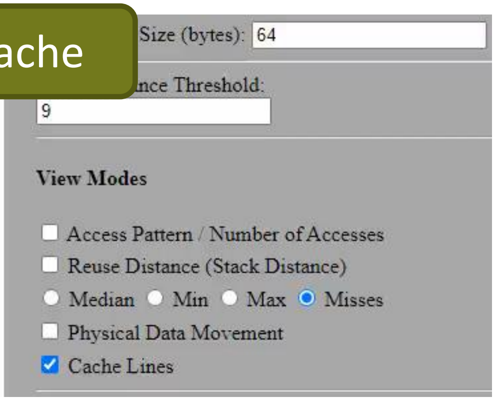
$I = 8$
$J = 8$
$K = 5$

Reshape data containers

Poor use of cache

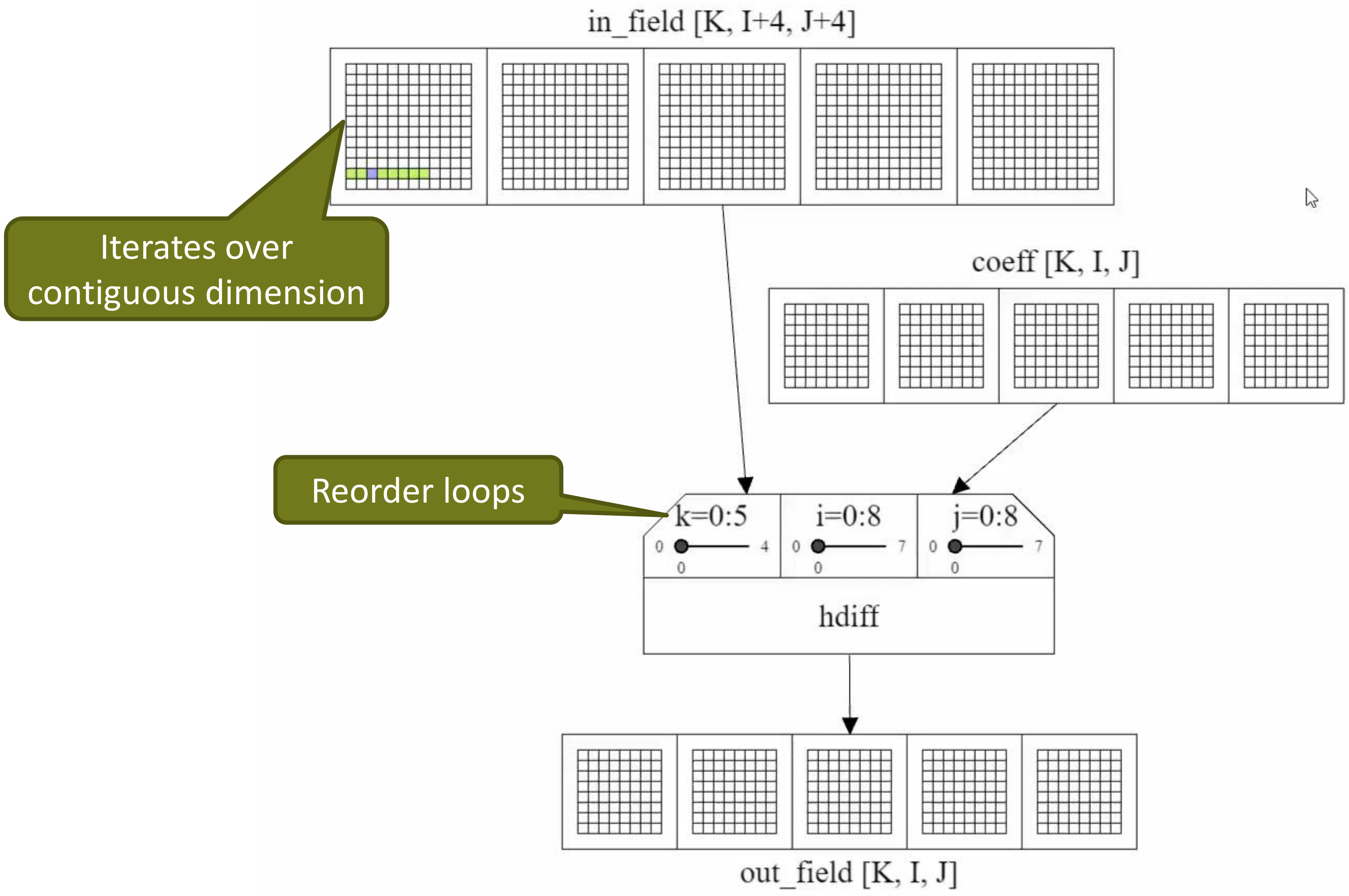
Better use of spatial locality

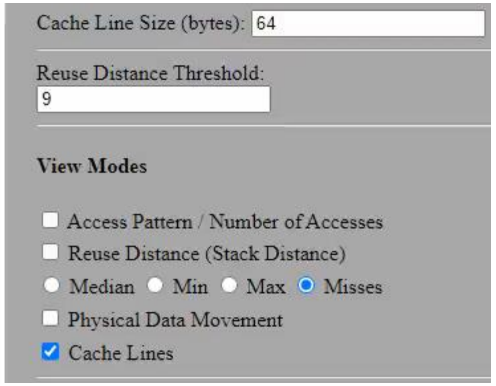




Stencil Optimization

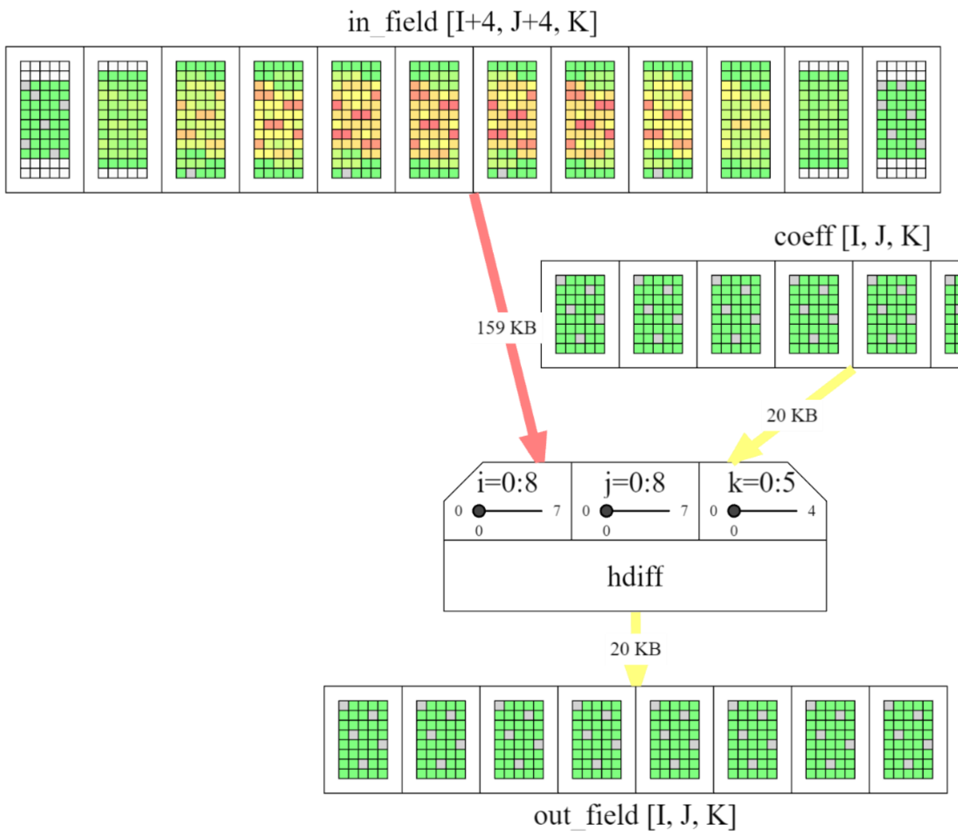
$I = 8$
$J = 8$
$K = 5$



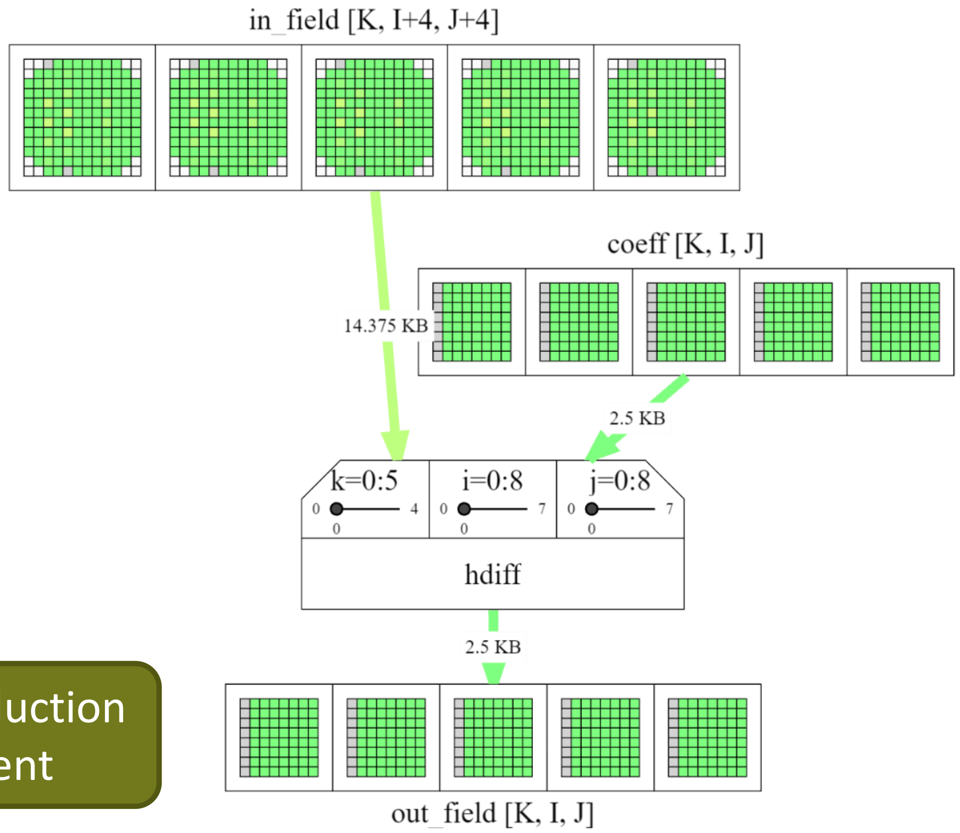


Stencil Optimization

16-core Intel Xeon Gold 6130 at 2.1 GHz, 1.5 TB RAM



Predicted 10.3x reduction in data movement

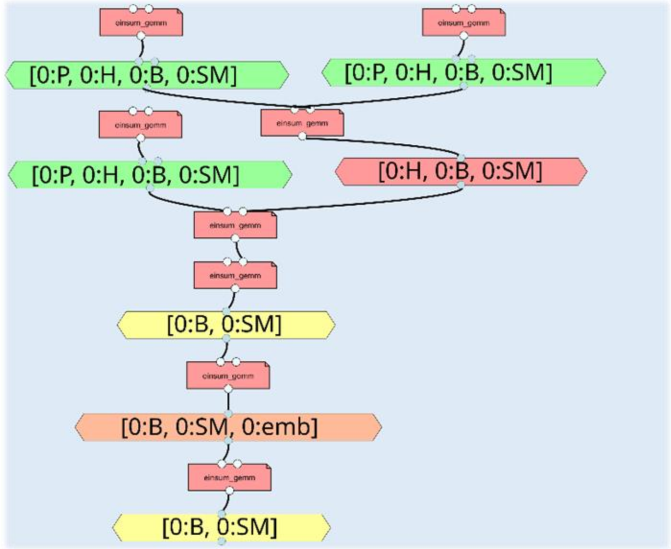


138x Speedup

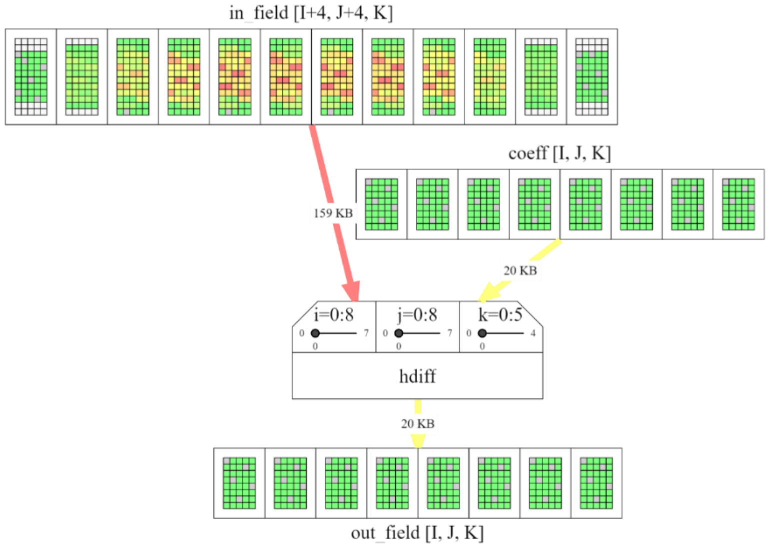
9.6x Reduction in cache misses

Conclusion

Global Data Movement

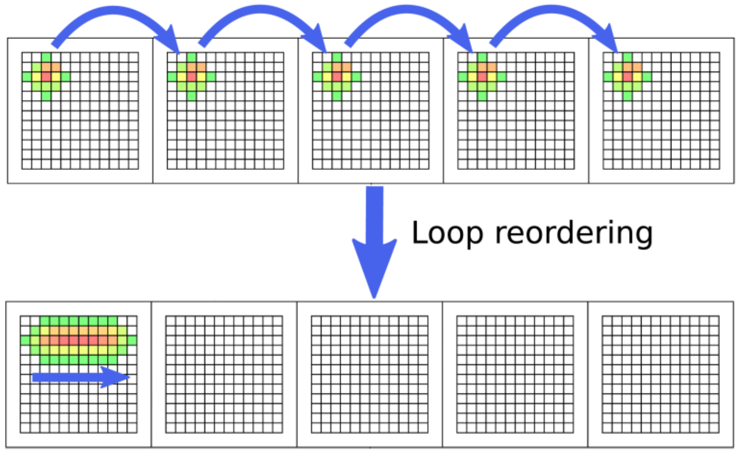


Fine-Grained Data Reuse

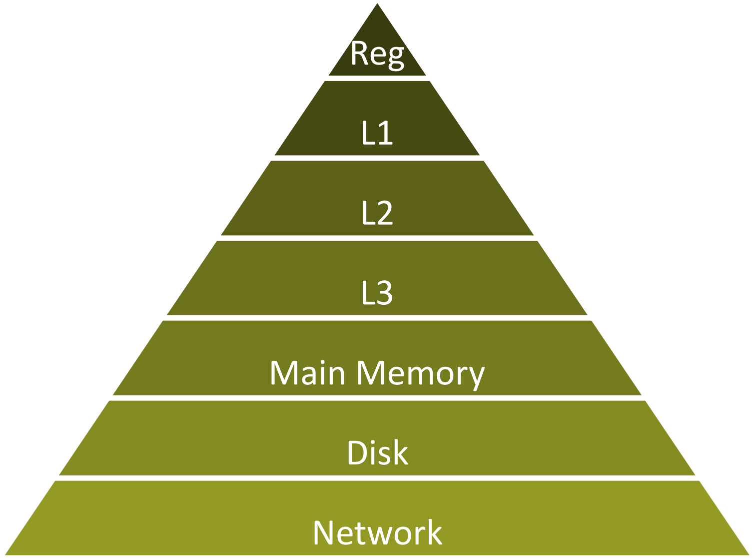


Where Next?

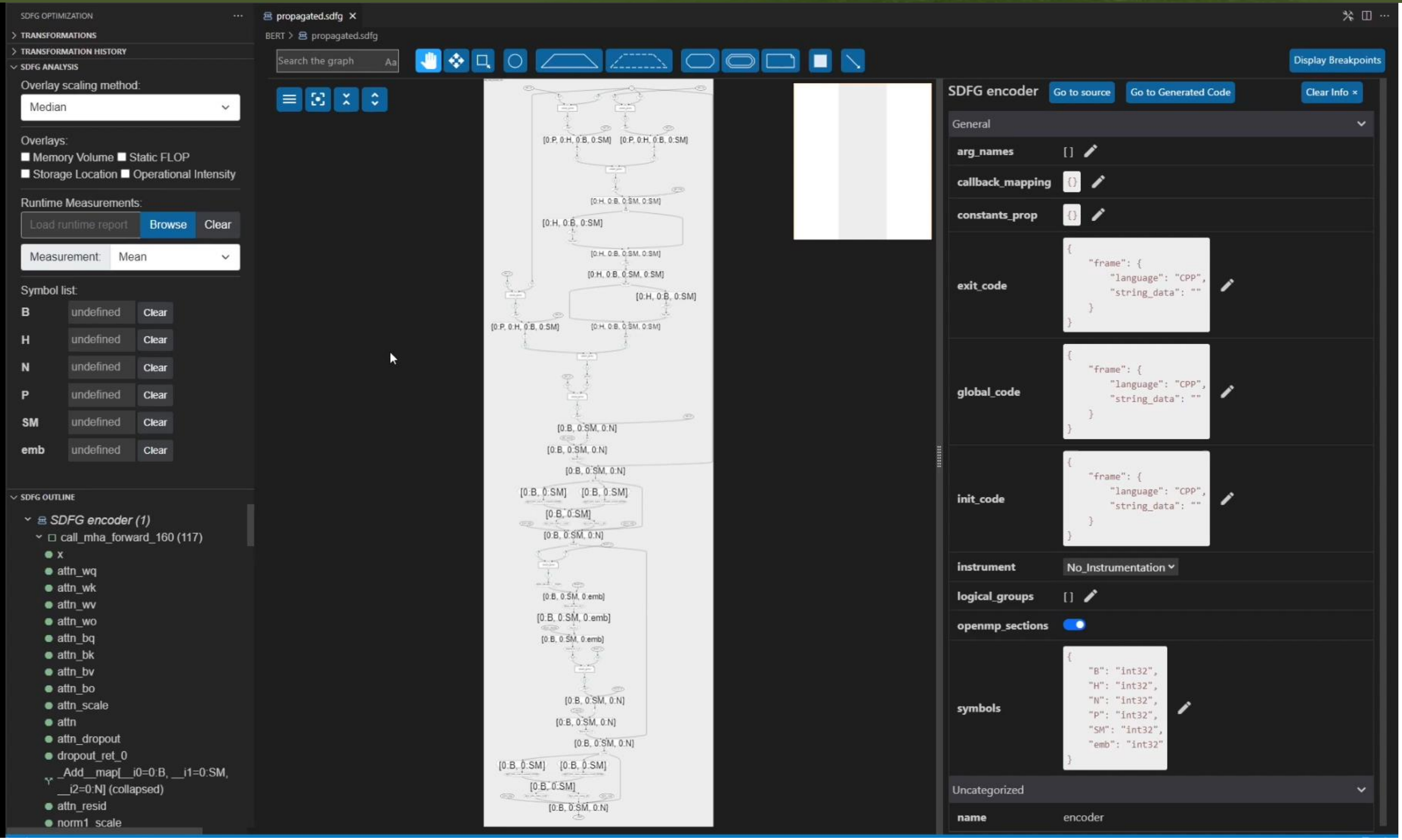
Automatic Optimization



Hardware Modelling

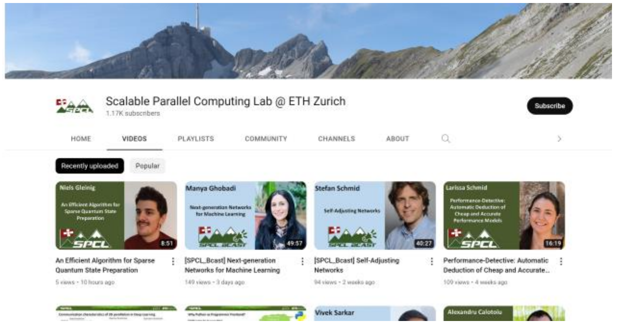


Educational Tool



Thank you!

youtube.com/@spcl



twitter.com/spcl_eth

spcl.inf.ethz.ch

github.com/spcl

https://marketplace.visualstudio.com/items?itemName=phschaad.sdfv

https://github.com/spcl/dace-vscode