61 THE CHIPS TO SYSTEMS CONFERENCE

SHAPING THE NEXT GENERATION OF ELECTRONICS

JUNE 23-27, 2024

MOSCONE WEST CENTER
SAN FRANCISCO, CA, USA

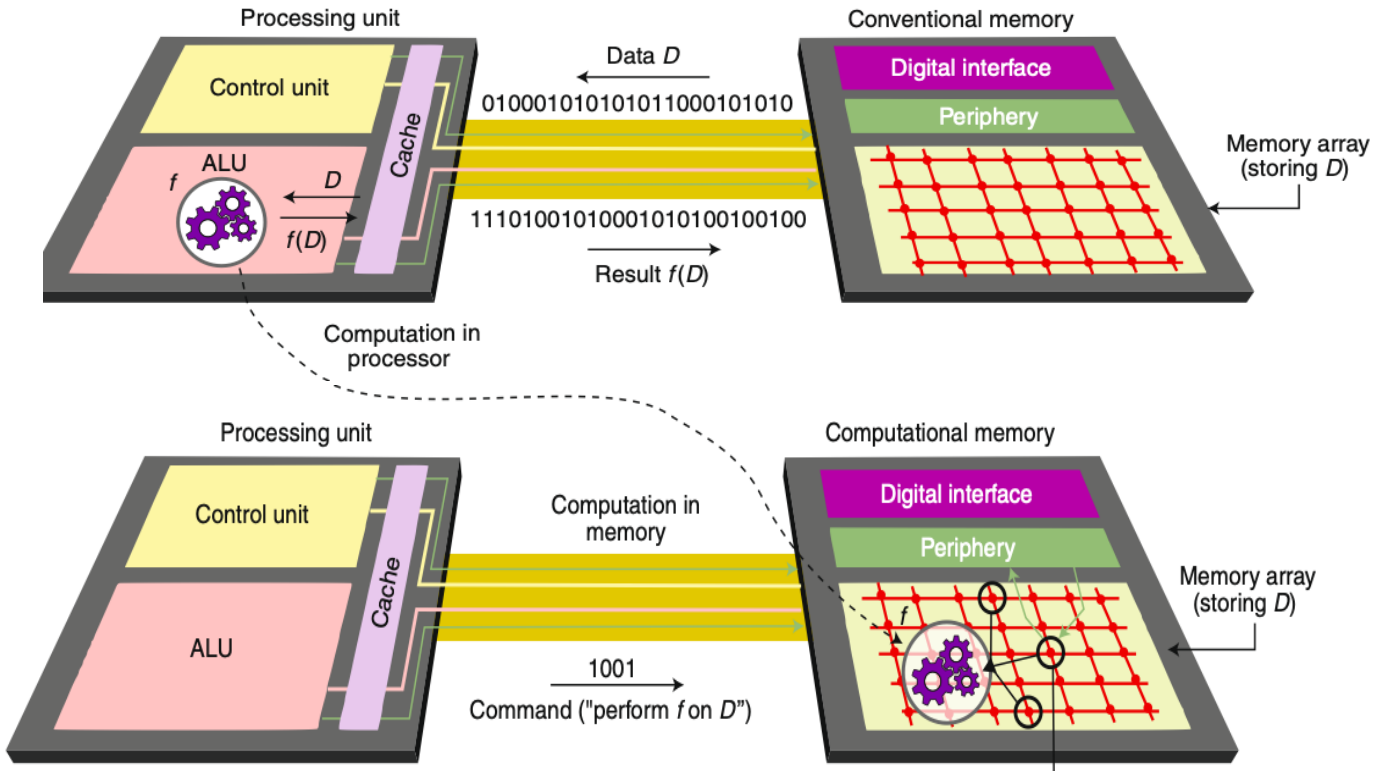
THE CHIPS TO SYSTEMS CONFERENCE
SHAPING THE NEXT GENERATION OF ELECTRONICS
JUNE 23-27, 2024
MOSCONE WEST CENTER
SAN FRANCISCO, CA, USA

SHERLOCK: Scheduling Efficient and Reliable Bulk Bitwise Operations in NVMs

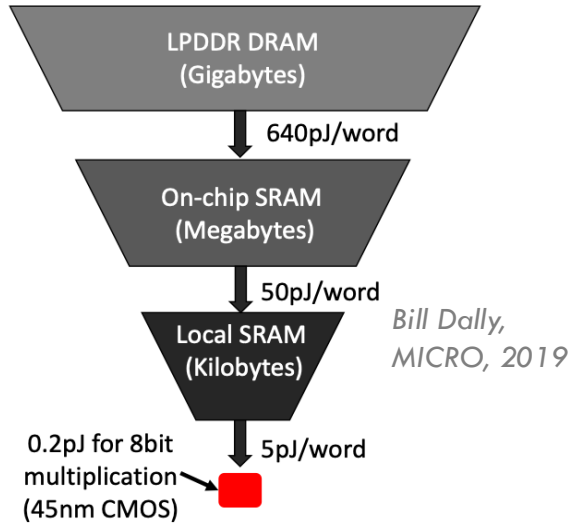
Hamid Farzaneh[1], João Paulo C. de Lima[12], Ali Nezhadi Khelejani[3], Asif Ali Khan[1], Mahta Mayahinia[3], Mehdi Tahoori[3] and Jeronimo Castrillon[12]

[1] Technical University of Dresden, [2] ScaDS.AI, [3] Karlsruhe Institute of Technology

Cost-driven accelerators design



Abu Sebastian et al, Nature Nanotechnology, 2020

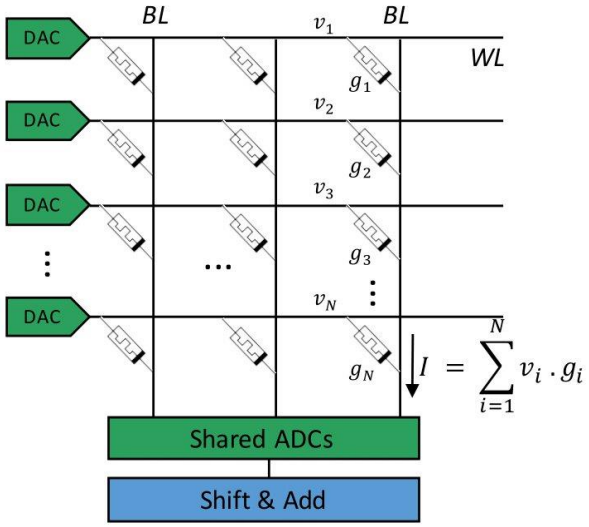


Bill Dally, MICRO, 2019

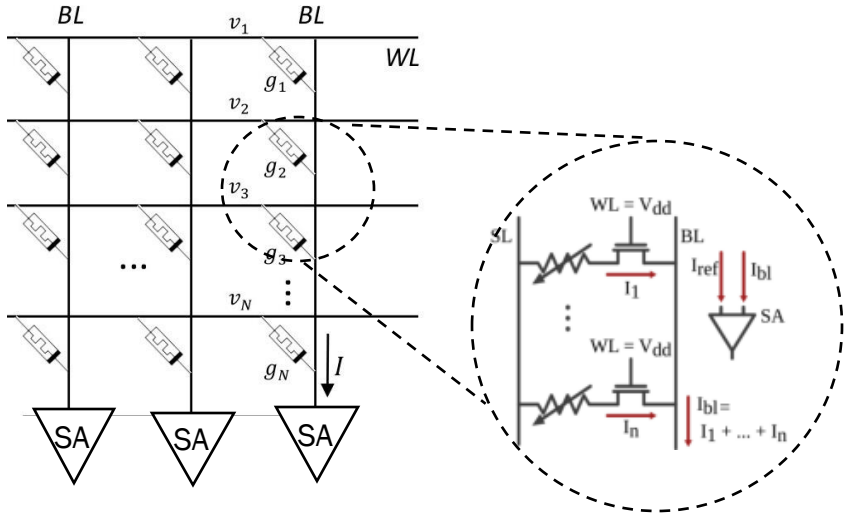
Computation is almost for free

Memristor-based computing-in-memory (CIM)

Analog dot-product engines



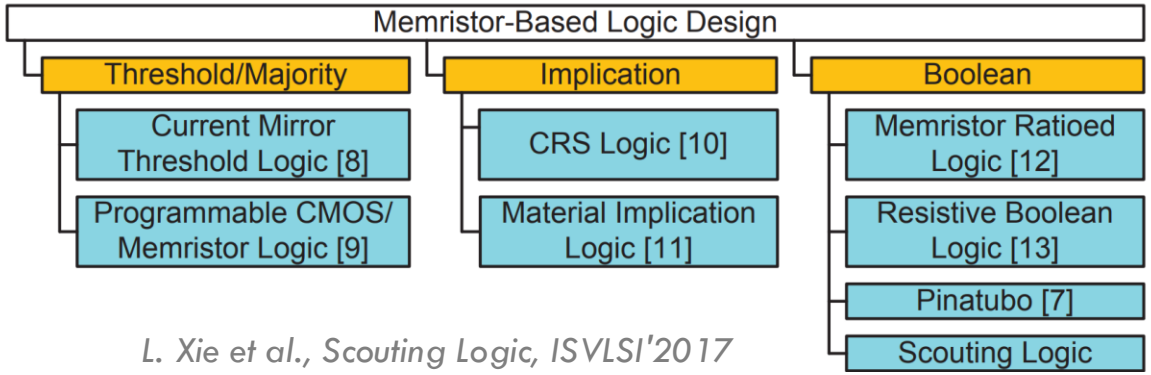
Bulk-bitwise logic



L. Xie et al., Scouting Logic, ISVLSI'2017

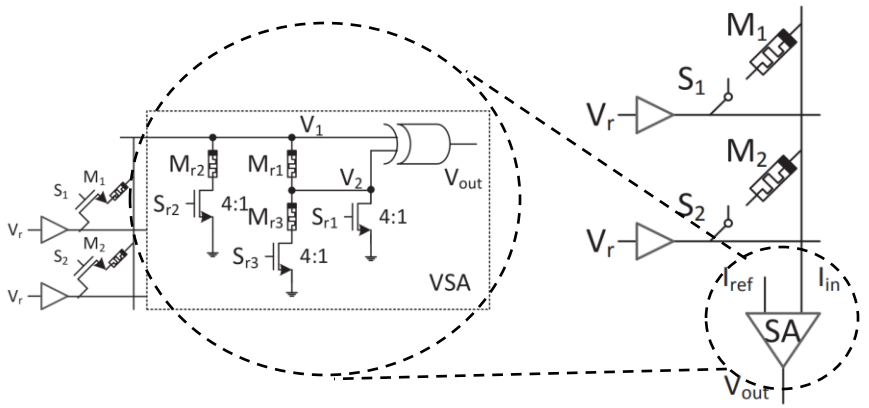
- CIM alleviates the data movement bottleneck
- Memristive crossbars enable:
  - Analog dot-product using DA/AD converters
  - Bitwise operations (OR, AND and NOT) using custom sense amplifiers

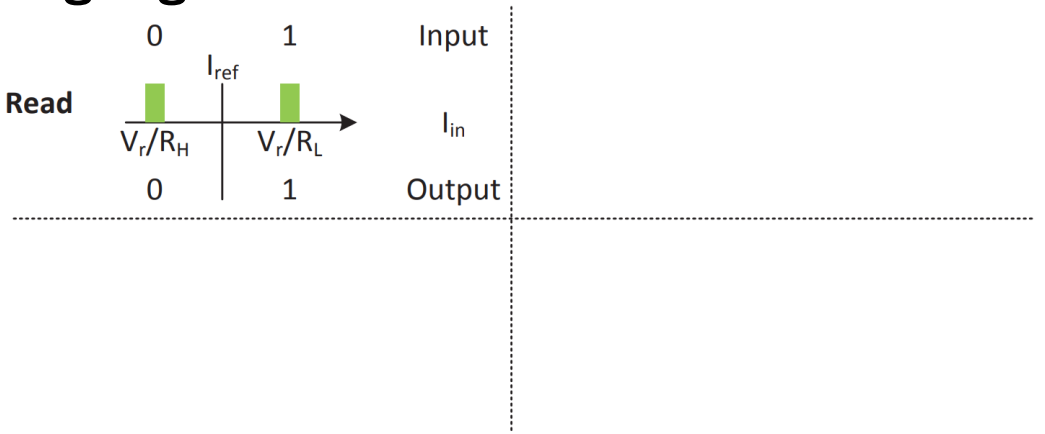
Memristor-based computing-in-memory (CIM)



L. Xie et al., Scouting Logic, ISVLSI'2017

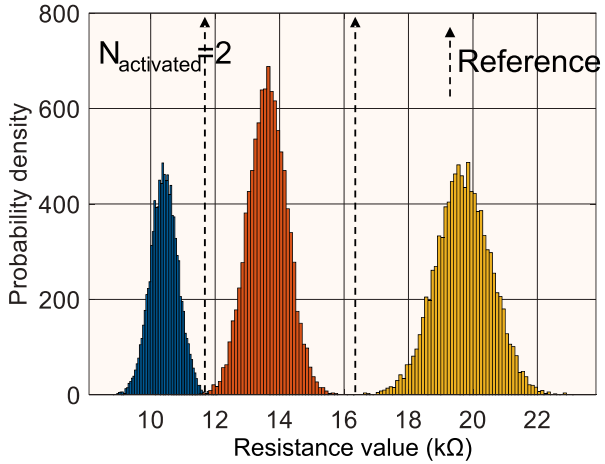
Scouting Logic

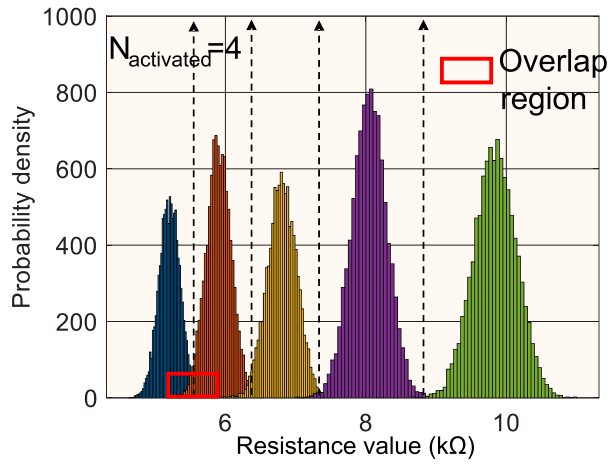




Reliability challenges of memristive CIM

- Decision failure probability
- HRS/LRS ratio
  - Larger HRS/LRS, lower decision failure
  - Technology-dependent parameter
- Number of inputs in logic operation
  - More inputs, smaller sense margin (↑ decision failure)
  - But also smaller standard deviation (↓ decision failure)



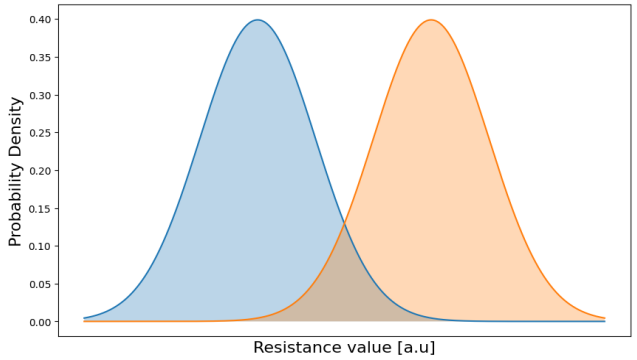


Sherlock at a glance

Problems

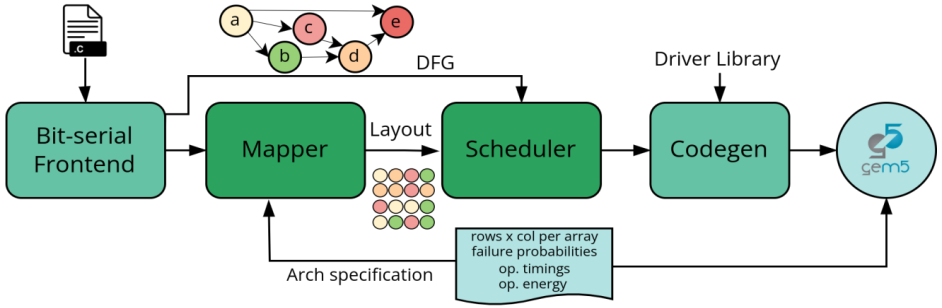
Lack of flexibility
- Row granularity

Performance-Reliability trade-off
- Multiple row activation



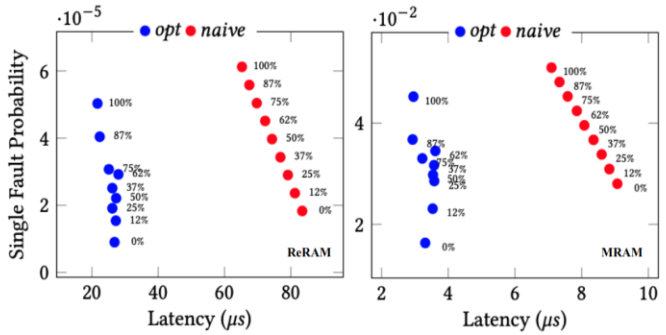
Goals

Mapping strategies for performance, energy and reliability
- Enable fine-grain control over operations
- Optimize data-flow graph mapping



Results

- Database, image processing and encryption algorithms
- Improves latency (~10×), energy consumption (~4.6×), and reliability (~1.5×) compared to the SotA



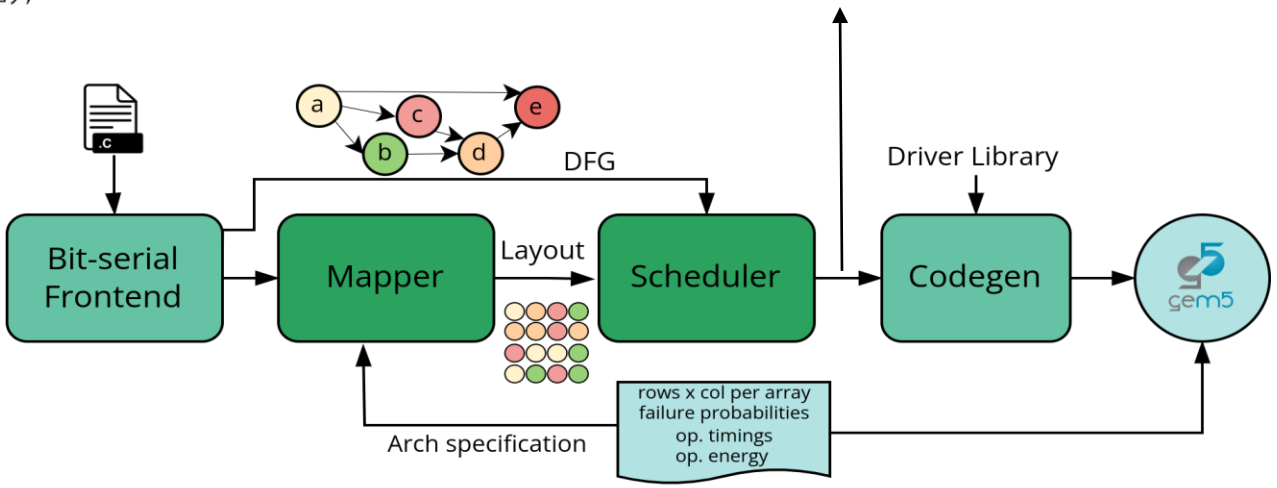
Sherlock flow

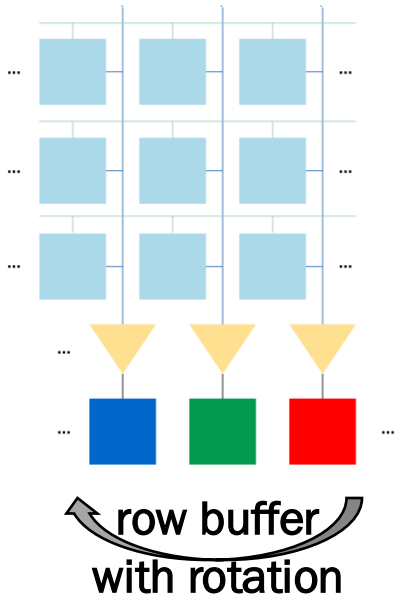
C/C++ code

```
for (i = 0 to K):
  t1 = NOT(C1[i]);
  T2 = AND(T1, data[i]);
  T3 = AND(T2, meq1);
  m_gt = OR(T3, mgt);
  ...
  meq1 = AND(meq1, T8);
  T9 = XOR(data[i], C2[i]);
  T10 = NOT(T9);
  meq2 = AND(meq2, T10);
```

MIMD representation

```
...
op address[arrayID][columns][rows] [cim-op]
write [0][4,8,12,16][932]
Read  [0][1,5,9, 13][5]
Shift [0] R[3]
Read  [0][4,8,12,16][933,934] [XOR,AND,OR,XOR]
...
```

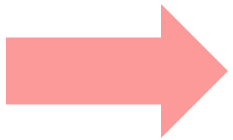


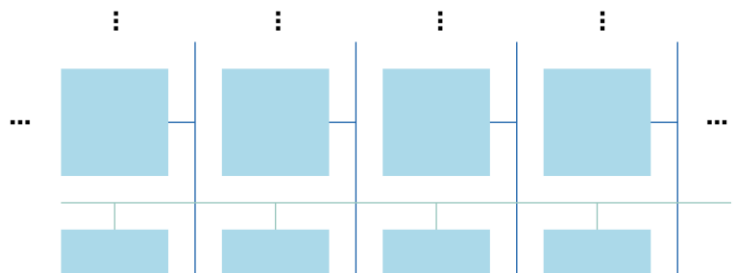


- Multiple row activation
- Rotation, row clone and NOT operation
- Output can be written back as voltage input for the next operation
- Fine-grain selection of columns

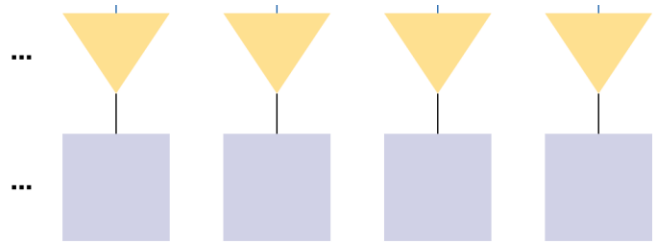
Problem statement with Bitweaving

```
for (i = 0 to K):
  t1 = NOT(C1[i]);
  T2 = AND(T1, data[i]);
  T3 = AND(T2, meq1);
  m_gt = OR(T3, mgt);
  T4 = NOT(data[i]);
  T5 = AND(T4, C2[i]);
  T6 = AND(T5, m_eq2);
  m_lt= OR(T6, mlt);
  T7 = XOR(data[i], C1[i]);
  T8 = NOT(T7);
  meq1 = AND(meq1, T8);
  T9 = XOR(data[i], C2[i]);
  T10 = NOT(T9);
  meq2 = AND(meq2, T10);
```





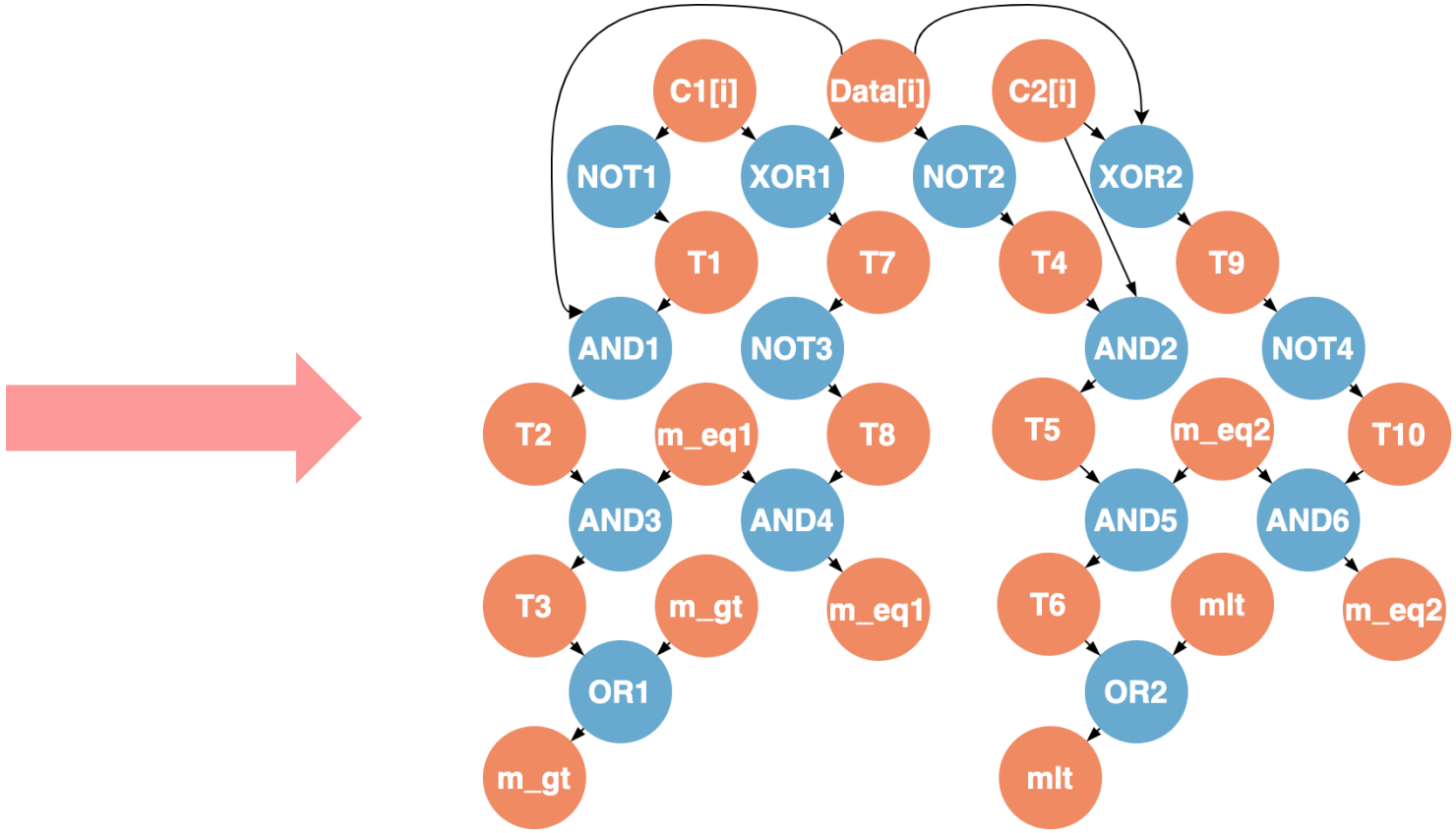
```
...
op address[arrayID][columns][rows] [cim-op]
write [0][4,8,12,16][932]
Read  [0][1,5,9, 13][5]
Shift [0] R[3]
Read  [0][4,8,12,16][933,934] [XOR,AND,OR,XOR]
...
```



Li and Patel, Bitweaving, SIGMOD'2013.

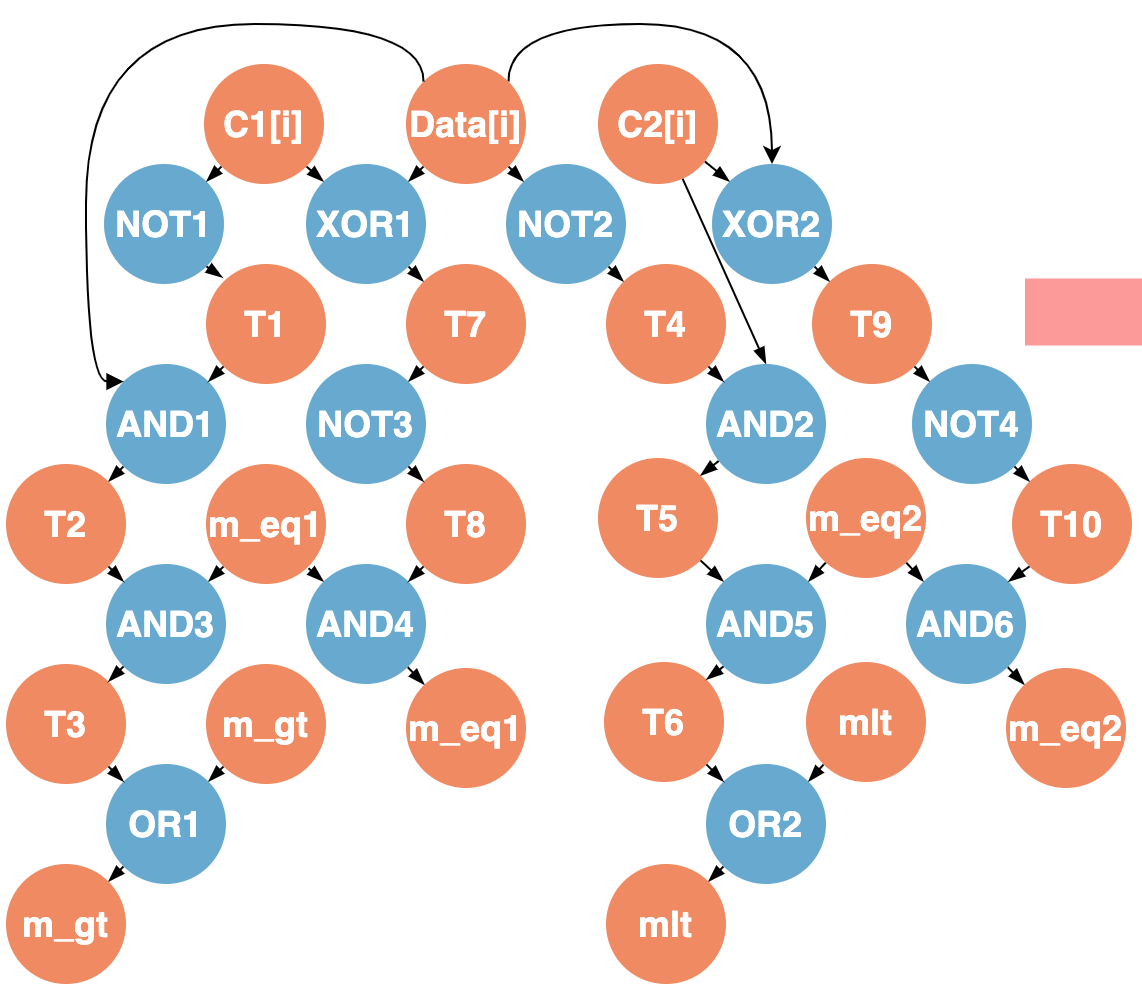
DFG Generation

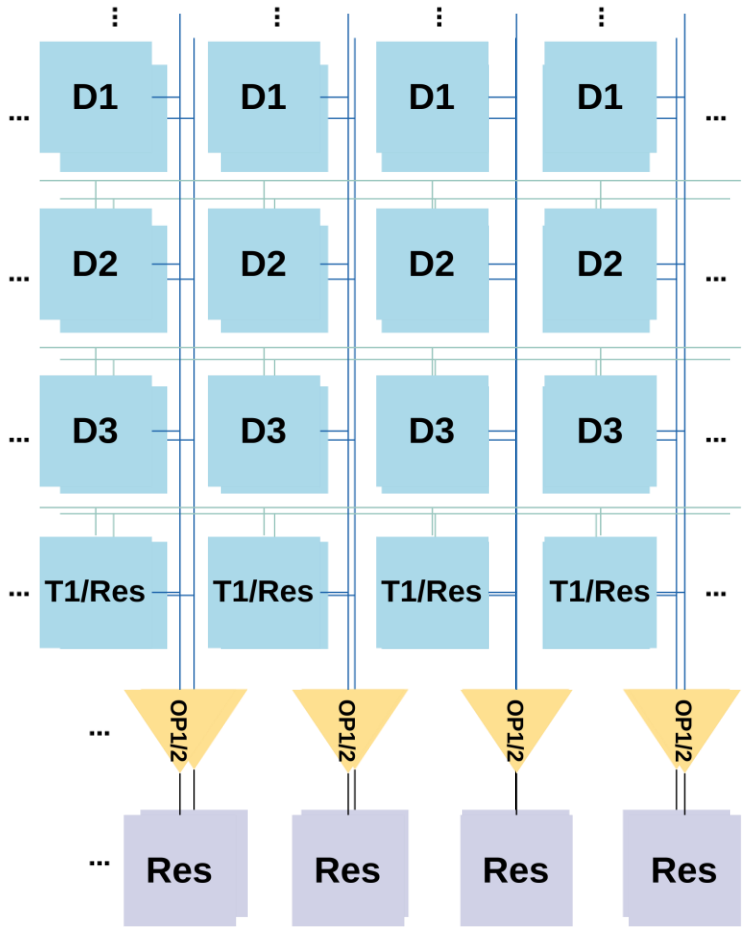
```
for (i = 0 to K):
  t1 = NOT(C1[i]);
  T2 = AND(T1, data[i]);
  T3 = AND(T2, meq1);
  m_gt = OR(T3, mgt);
  T4 = NOT(data[i]);
  T5 = AND(T4, C2[i]);
  T6 = AND(T5, m_eq2);
  m_lt= OR(T6, mlt);
  T7 = XOR(data[i], C1[i]);
  T8 = NOT(T7);
  meq1 = AND(meq1, T8);
  T9 = XOR(data[i], C2[i]);
  T10 = NOT(T9);
  meq2 = AND(meq2, T10);
```



Li and Patel, Bitweaving, SIGMOD'2013.

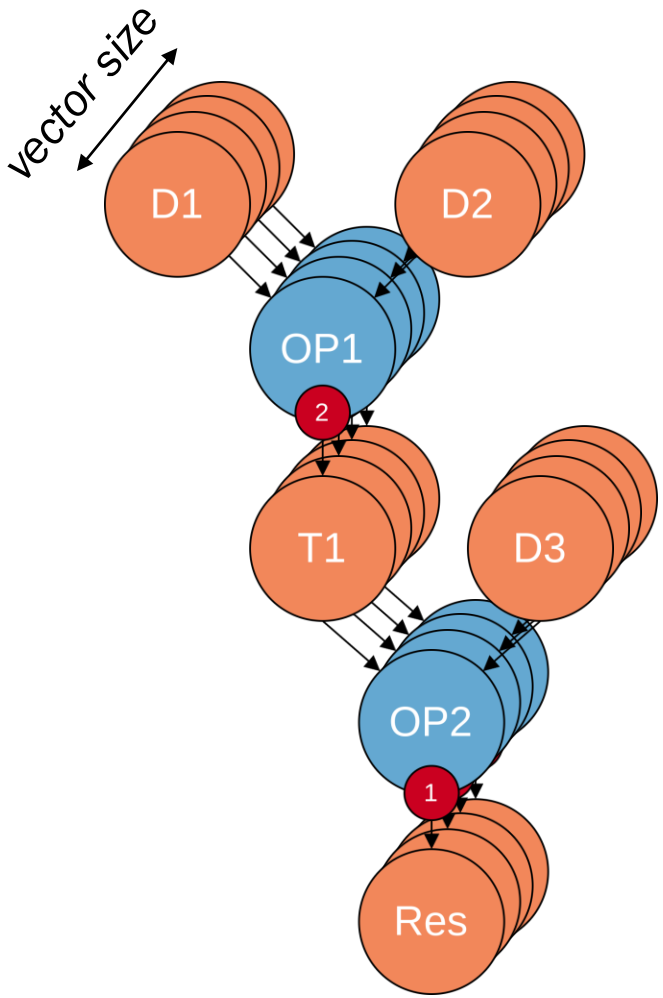
Map operands and operations

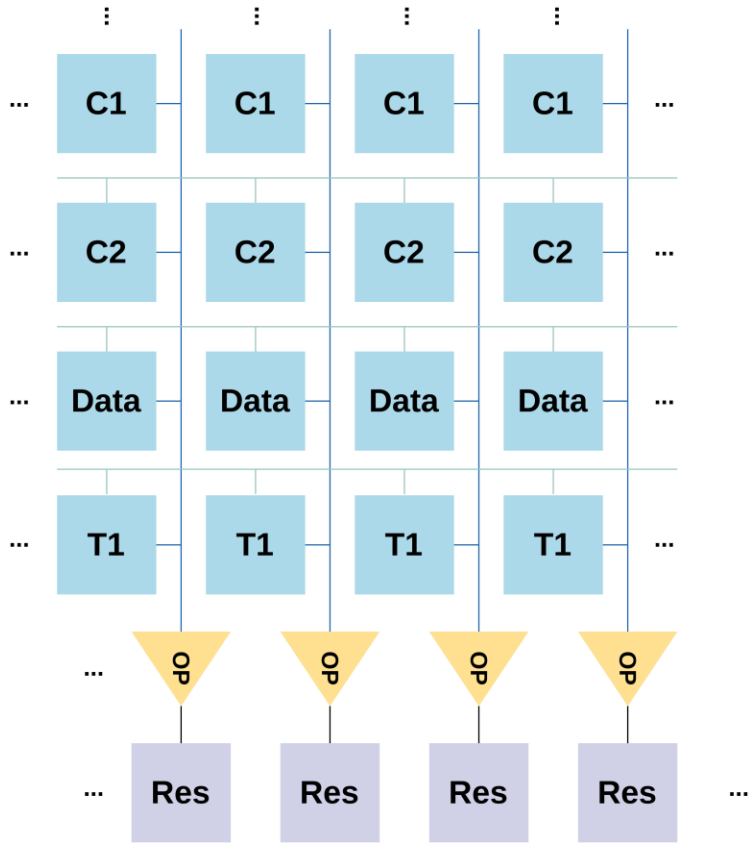




Pros of naive mapping

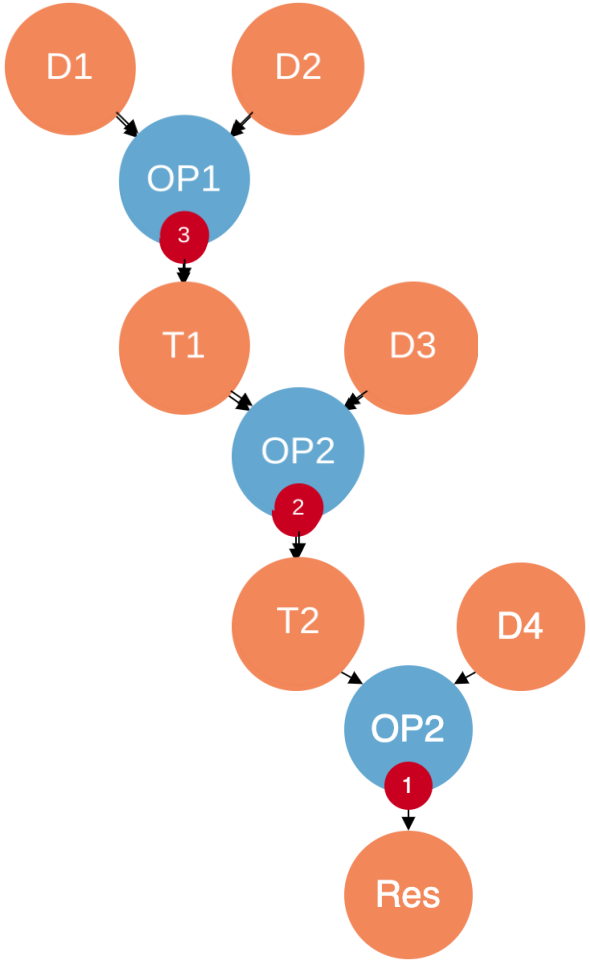
- Example
  - Step 1: Priority
  - Step 2: Map sequentially
- Pros
  - Simple
  - Extends SIMD parallelism

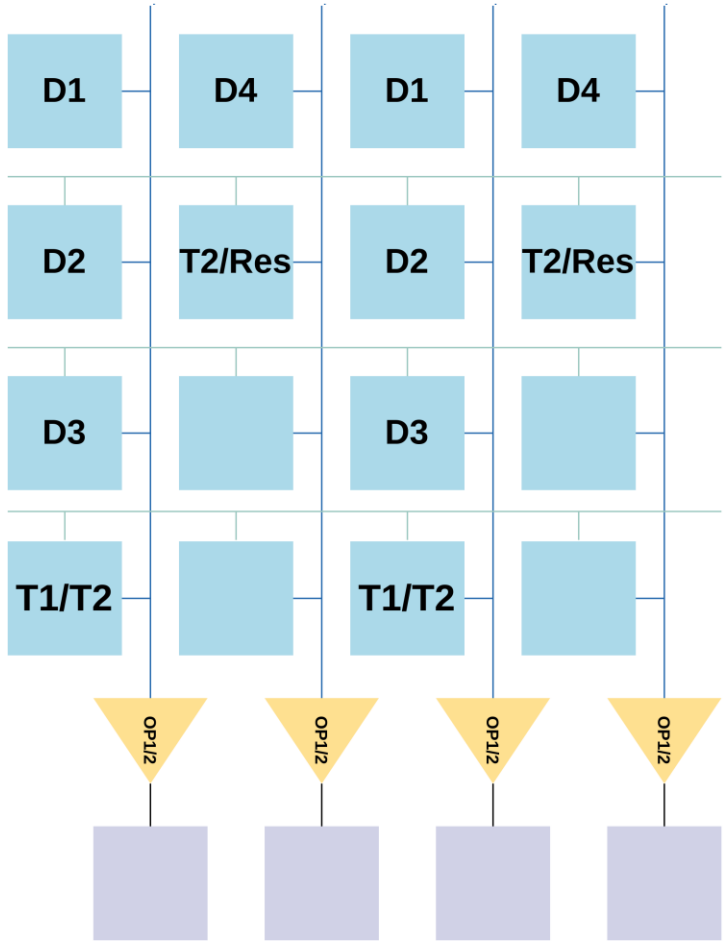




Cons of naive mapping

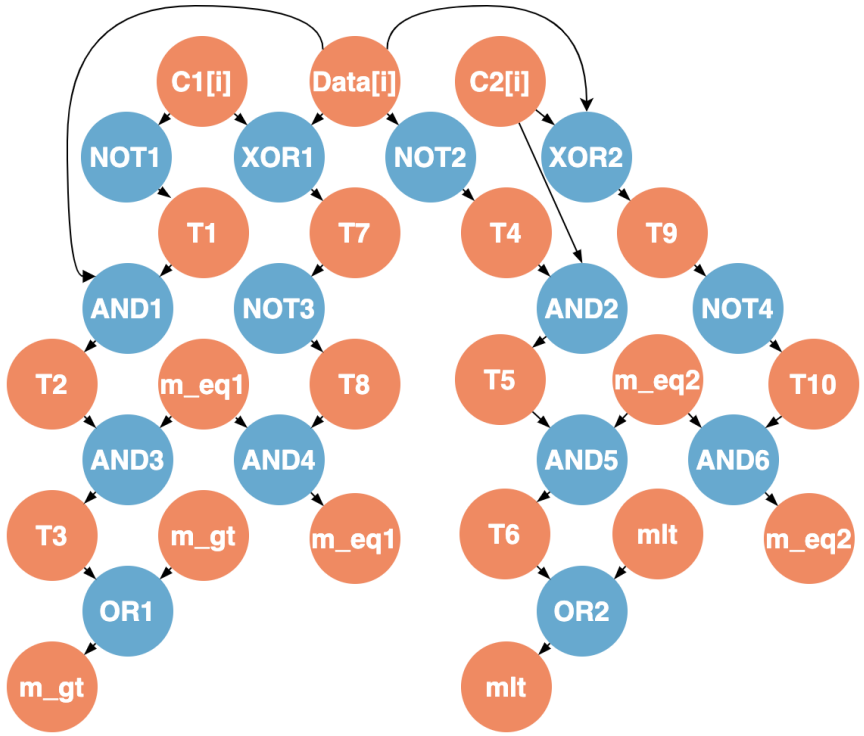
- DFGs may not fit into a single array
- Additional memory copies across columns are required in larger DFGs
- Resource utilization is poor when *vector size* < *column size*

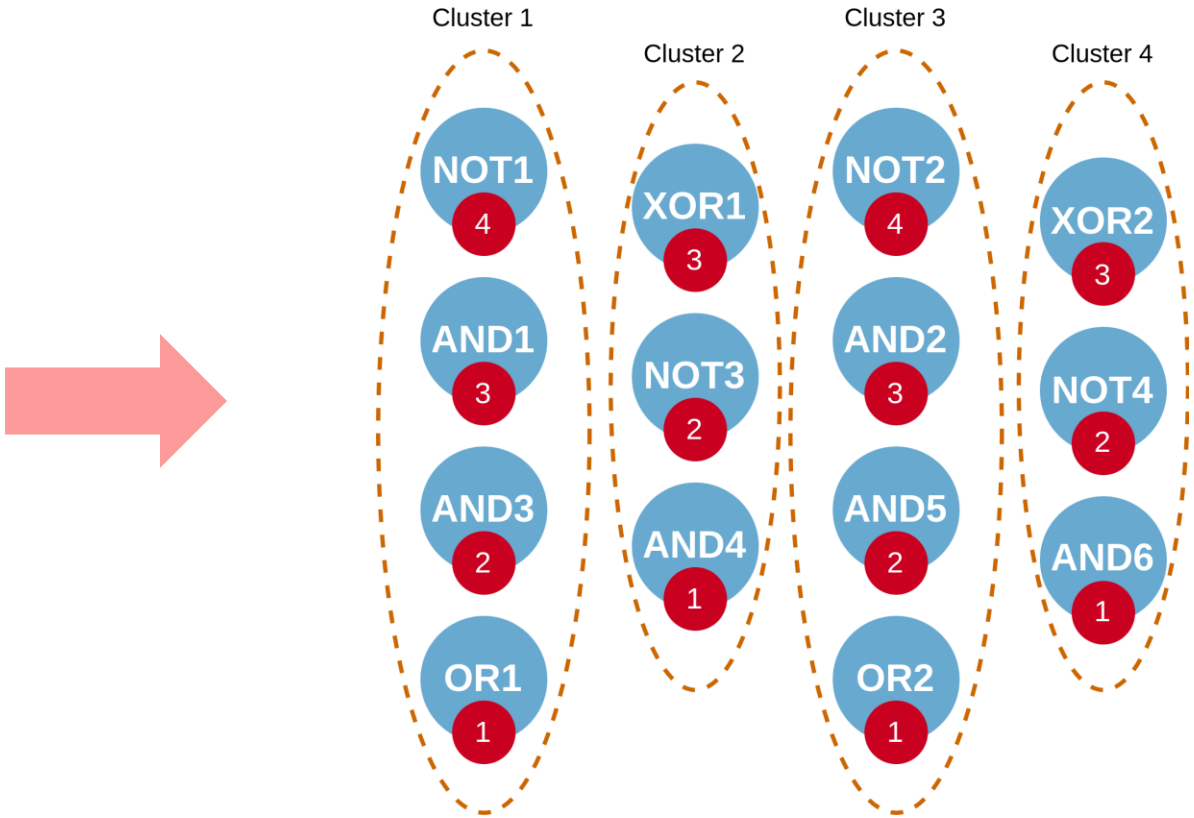




Sherlock mapping

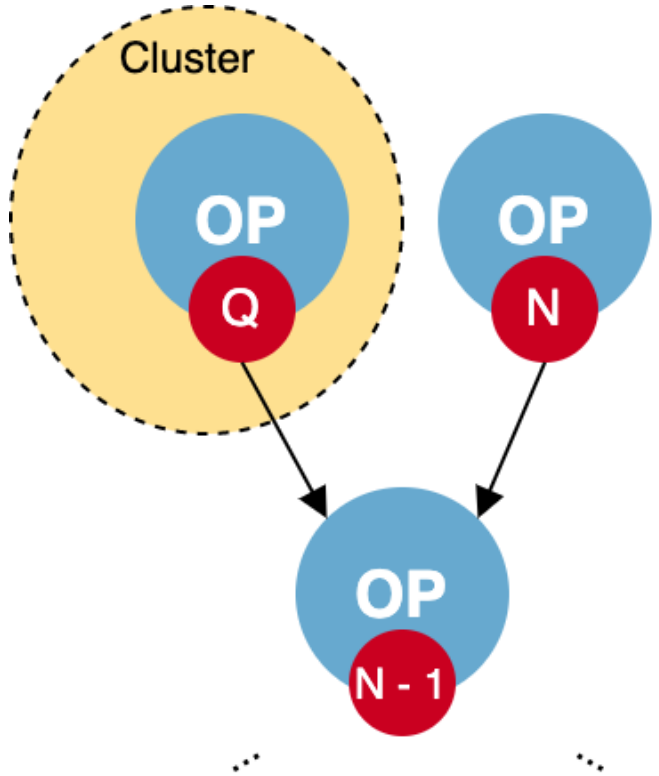
- Goal: Find K Clusters of operations for K columns of the memory





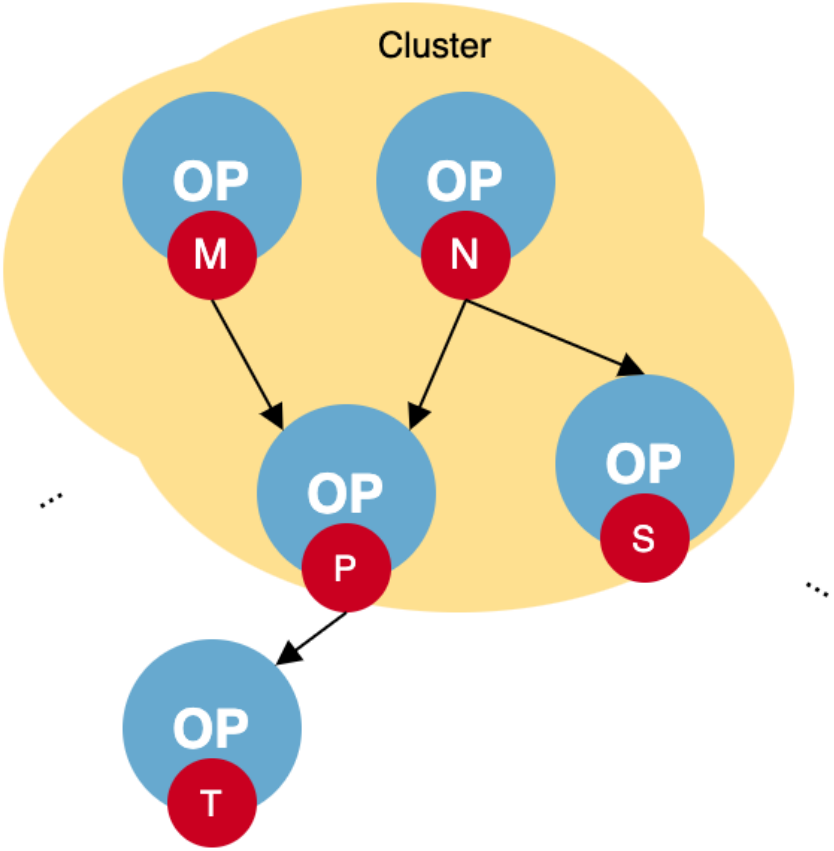
Clustering operation nodes

1) Assign priority values to OP nodes
2) Create initial cluster C
3) Start from the highest priority
   - Add Q (highest priority) to C
   - Add more nodes to C or create a new cluster based on different metrics



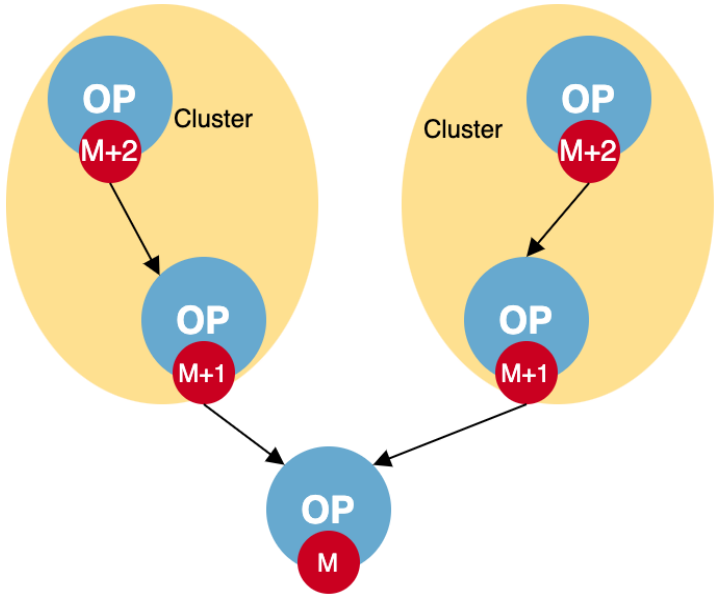
Clustering operation nodes

- 5 cases were identified
- Case 1:
  - A node only has one predecessor
  - If $C_{maxsize}$ > $Cluster_{size}$:
    - Node is added to the new cluster
  - Else
    - New cluster is created

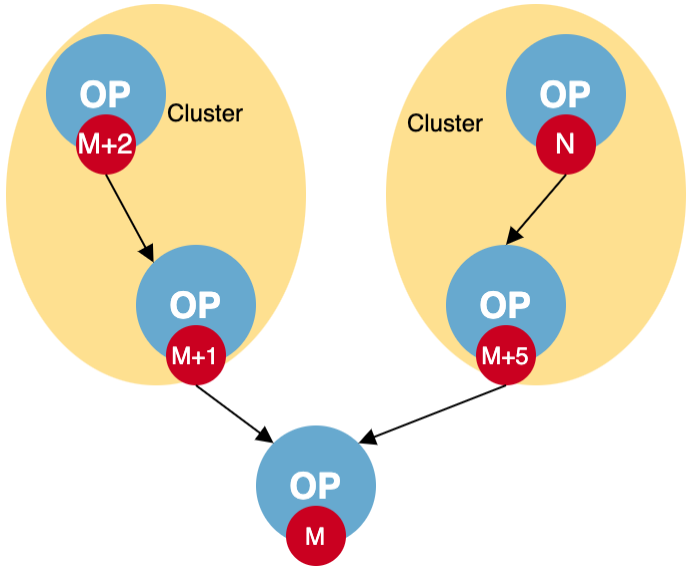


Clustering operation nodes

- Case 2:
  - Clusters are similar
  - Dependency relation is similar
  - $(Cluster_{size} + 1) < C_{maxsize}$
  - Randomly assignment

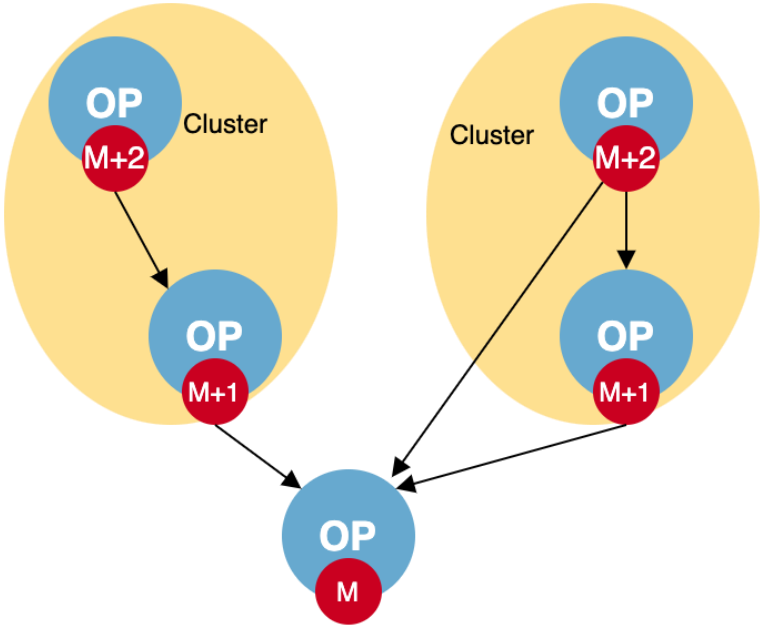


- Case 3:
  - Clusters are similar
  - Dependency relation is different
  - Put the node in the cluster with less difference in the priority: *Critical path*

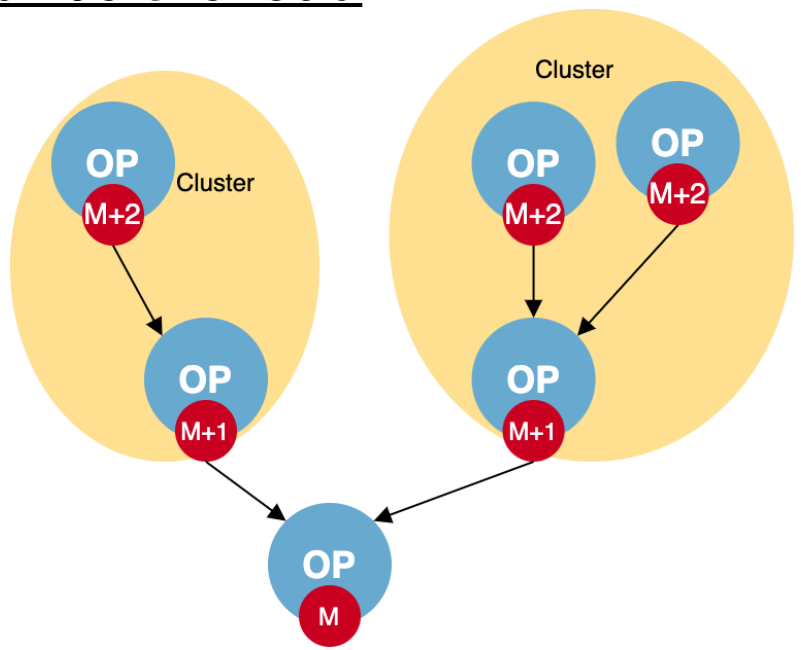


Clustering operation nodes

- Case 4:
  - The clusters are similar
  - The dependency relation is different
  - Put the node in the cluster with greater dependency

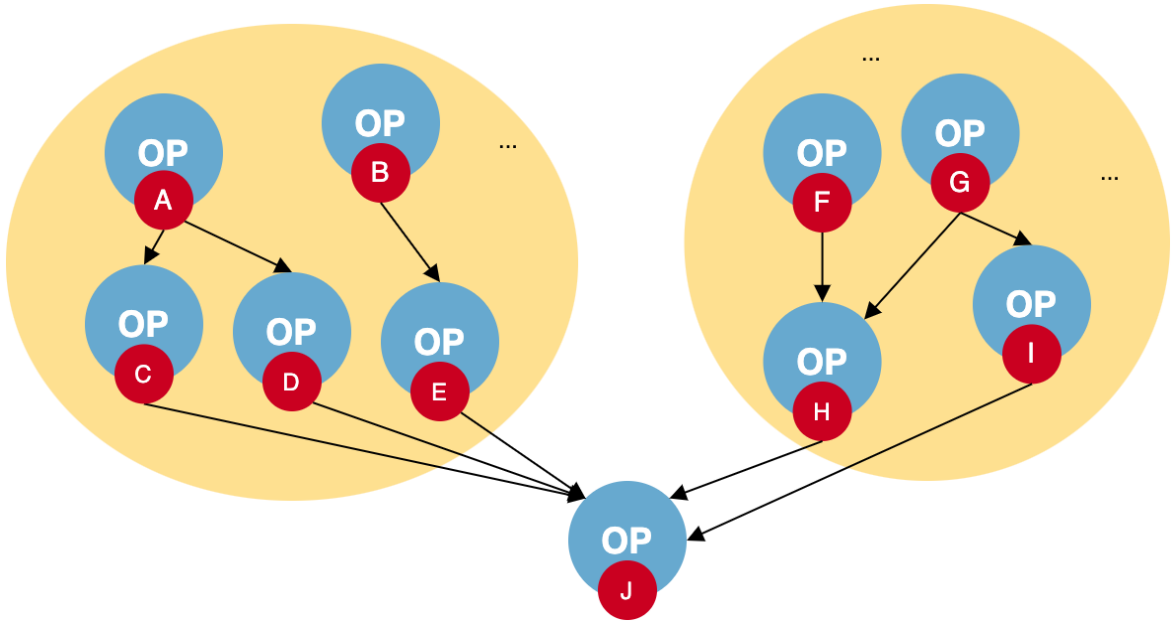


- Case 5:
  - The clusters are different
  - The dependency relation is similar
  - Put the node in the smaller cluster
  - *Balance the load*



Clustering operation nodes

- Generalizing for all cases



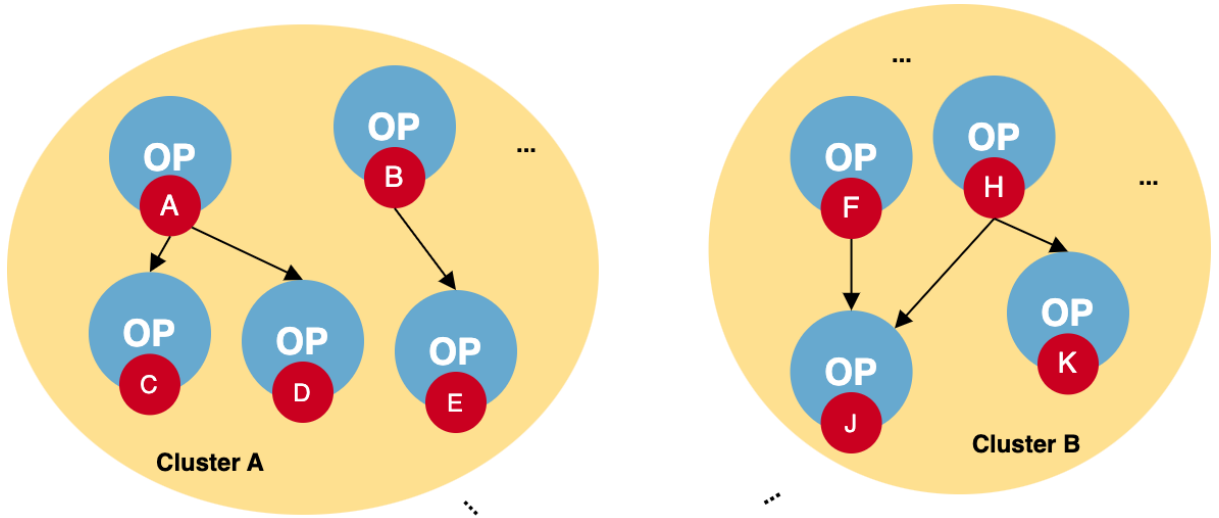
$$\text{score}(d, C) = \beta \cdot |C| + \alpha \cdot \sum_{q \in \text{pred}(d) \cap C} \rho(d, q)$$

α and β control the effect of cluster size and priority measure

ρ is the priority value difference between d and q

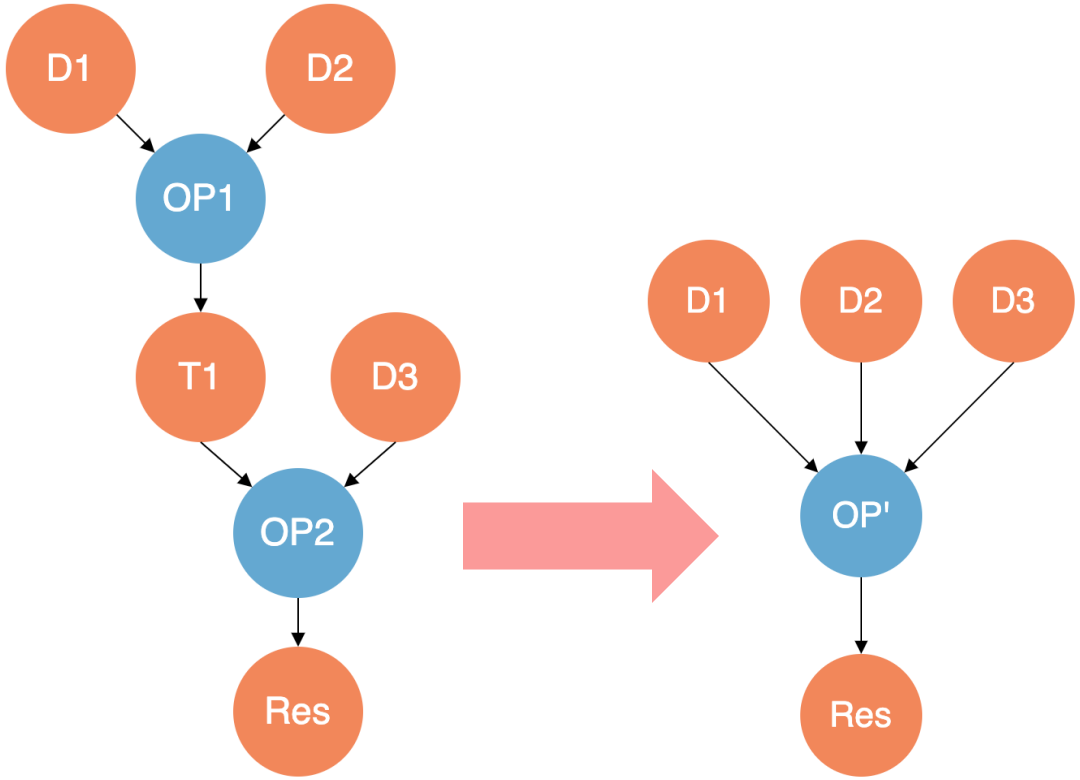
Mapping clusters to memory

- The size of a cluster < $C_{maxsize}$
- Two clusters can be merged
  - To better utilize memory
  - if ($Cluster\ A_{size}$ + $Cluster\ B_{size}$) < $C_{maxsize}$
  - Maximize dependency



Optimizing with many-row activation

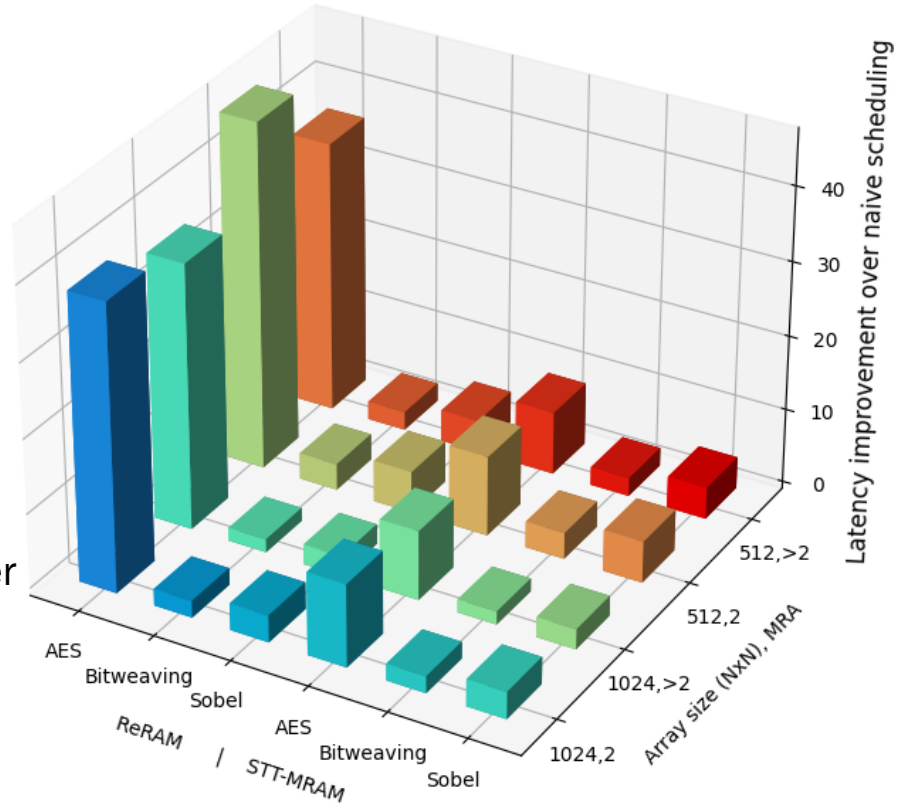
- Include multiple row activation in the scheduling
- Merge the operations with multiple operands in the final clusters
- Requirements
  - Operations are of the same type
  - Operands are mapped to the same column

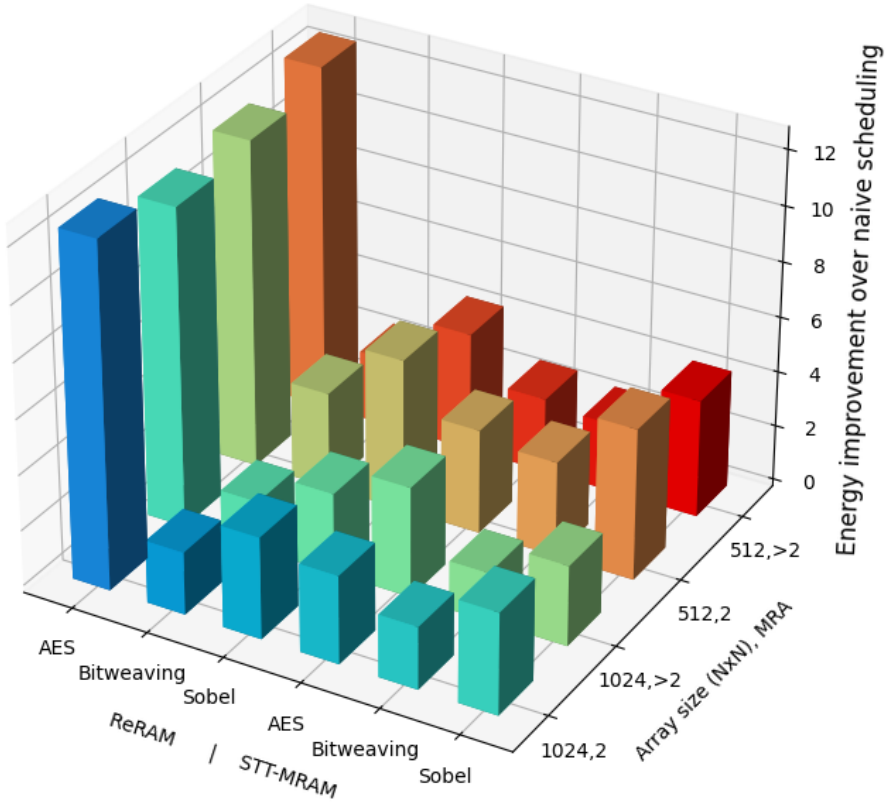


Latency and energy comparison

- Parameters to vary include technology (ReRAM, STT-MRAM), array size (512, 1024), and multi-row activation (2-operand only, >2 operands)

Higher is better

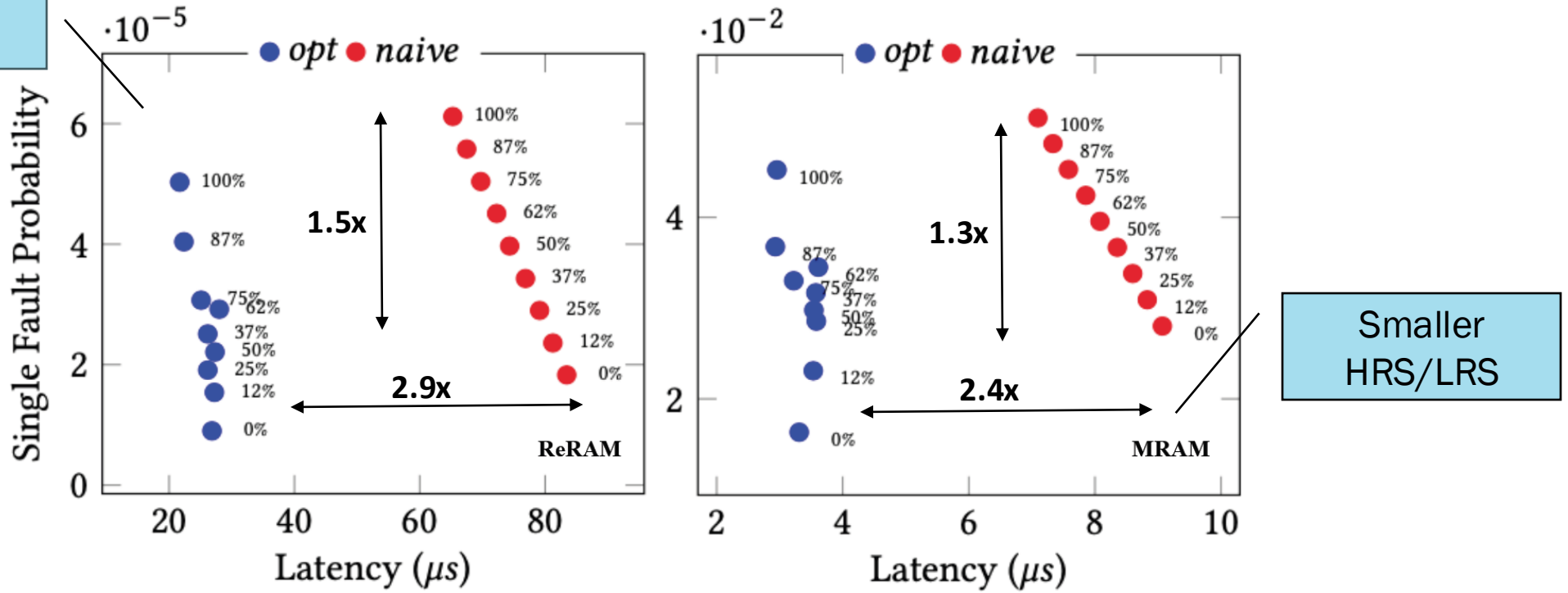




Impact of MRA on application reliability

- Reliability of Bitweaving output varying the allowed percentage of MRA (> 2 operands)

$P_{app} < 10^{-4}$ is highly reliable



Smaller HRS/LRS

suitable for applications tolerating some result inaccuracy

Main takeaways

- Scouting logic to alleviate the memory wall issue
- Mapping larger applications on the target CIM is not straightforward
- More flexibility by a retargetable scheduler
- Latency (~10×), energy consumption (~4.6×), and reliability (~1.5×) improvement compared to the SotA
- Sherlock aims to maximize the performance of CIM-logic

THE CHIPS TO SYSTEMS CONFERENCE
SHAPING THE NEXT GENERATION OF ELECTRONICS
JUNE 23-27, 2024
MOSCONE WEST CENTER
SAN FRANCISCO, CA, USA

Thank you for listening

hamid.farzaneh@tu-dresden.de

joao.lima@tu-dresden.de

61 THE CHIPS TO SYSTEMS CONFERENCE

SHAPING THE NEXT GENERATION OF ELECTRONICS

JUNE 23-27, 2024

MOSCONE WEST CENTER
SAN FRANCISCO, CA, USA