Lecture 15 — Domain Specific Accelerators

Stanford CS343D (Winter 2024)
Fred Kjolstad

Domain-Specific Software Stack

Domain-Specific Language/Library

Domain-Specific Compiler

Domain-Specific Hardware

Why Domain-Specific Architectures

Chip type:

- Microprocessor
- Microprocessor + GPU
- General purpose DSP
- Dedicated design

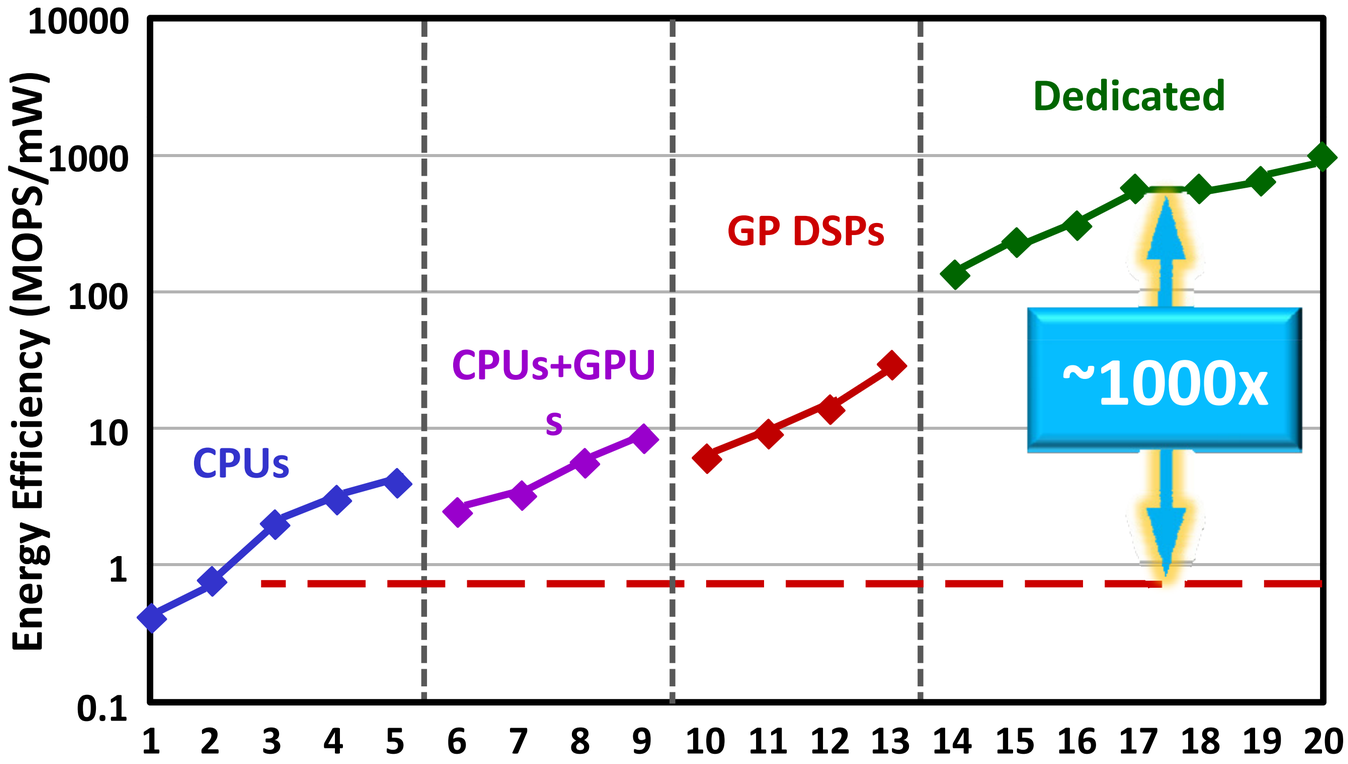
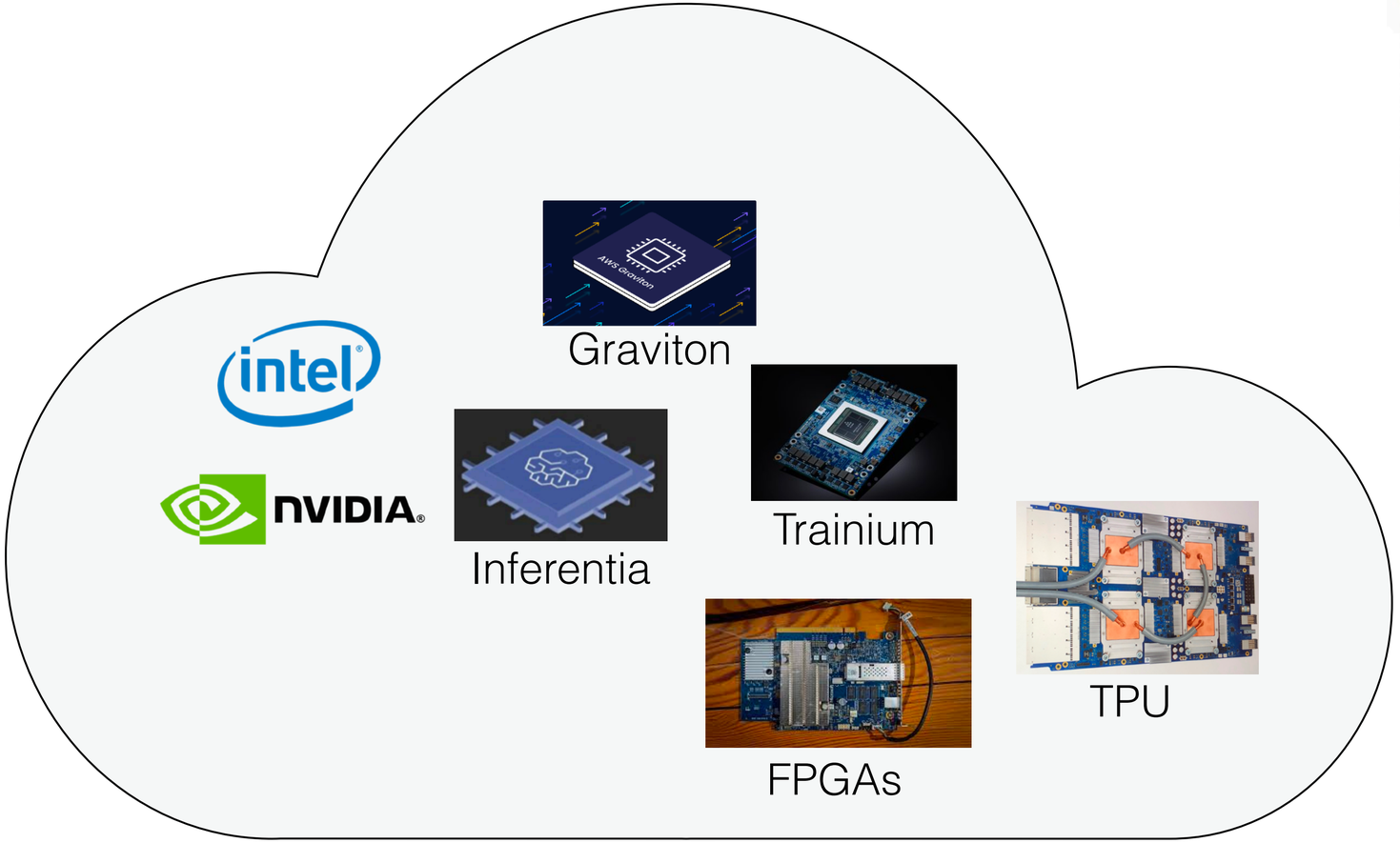


Image Credits: Dejan Markovic, EE292E Lecture Notes, Lecture 5, Stanford University, 2013

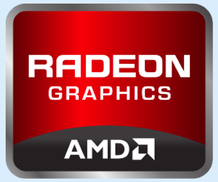
Hardware in Industry



Types of Hardware

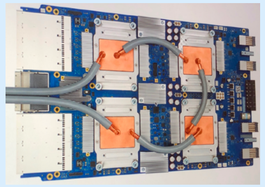
Multicore CPUs

SIMT GPUs



Fixed-Function ASICs

SIMD

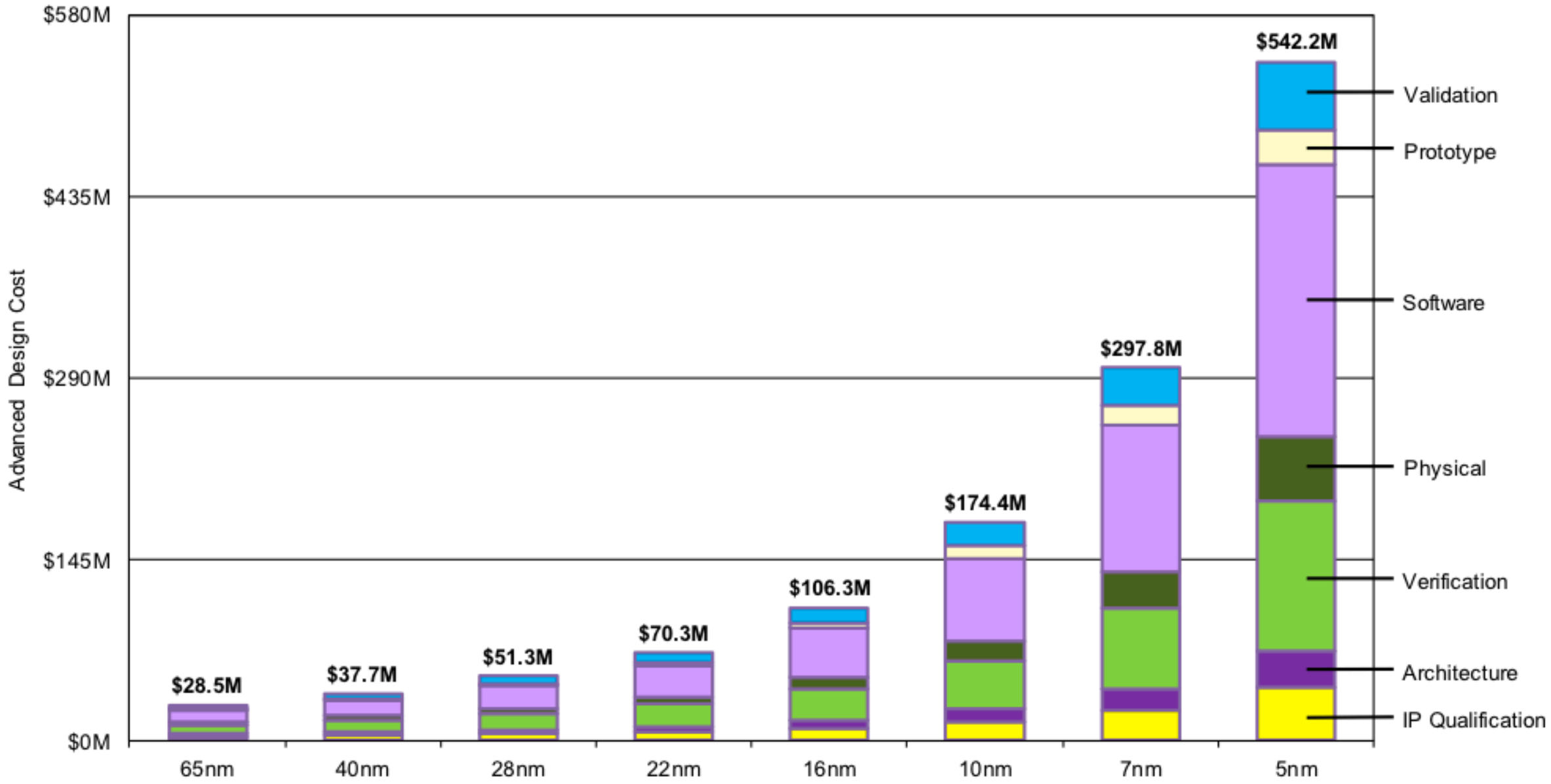


Programmable Streaming Dataflow

CGRA

RDA

Economics of Hardware



Source: IBS, see https://www.extremetech.com/computing/272096-3nm-process-node

Economics of Hardware

"That changed in 2013 when a projection showed people searching by voice for three minutes a day using speech recognition DNNs would double our datacenters' computation demands, which would be very expensive using conventional CPUs."

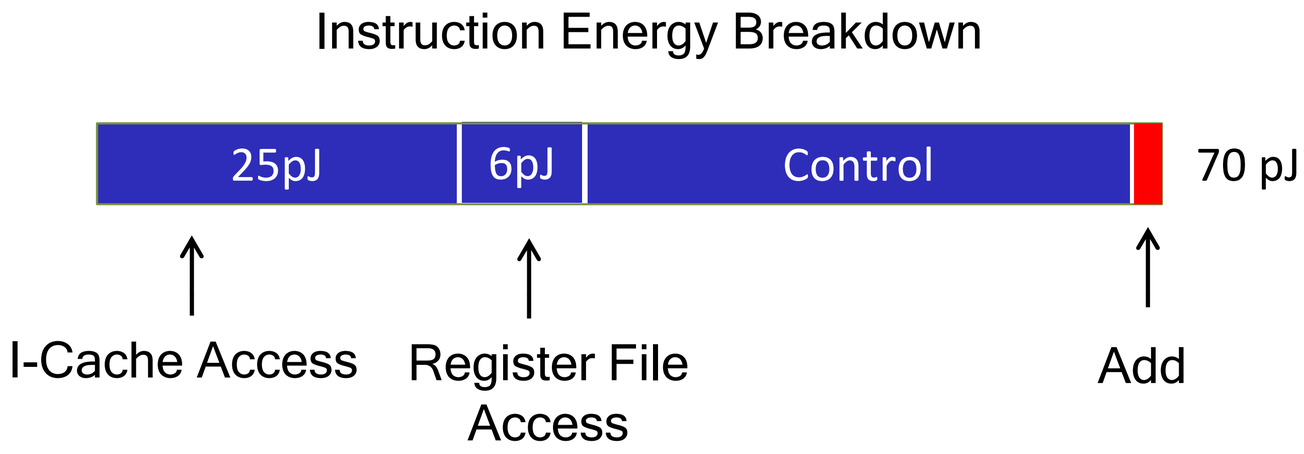
In-Datacenter Performance Analysis of a Tensor Processing Unit, Jouppi et al., Google, 2017

Hardware Design Considerations

Integer	
Add	
8 bit	0.03pJ
32 bit	0.1pJ
Mult	
8 bit	0.2pJ
32 bit	3.1pJ

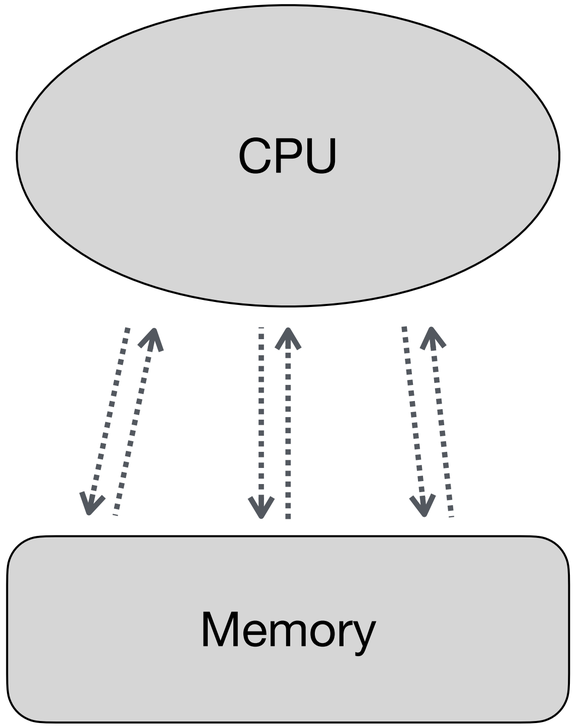
FP	
FAdd	
16 bit	0.4pJ
32 bit	0.9pJ
FMult	
16 bit	1.1pJ
32 bit	3.7pJ

Memory	
Cache	(64bit)
8KB	10pJ
32KB	20pJ
1MB	100pJ
DRAM	1.3-2.6nJ

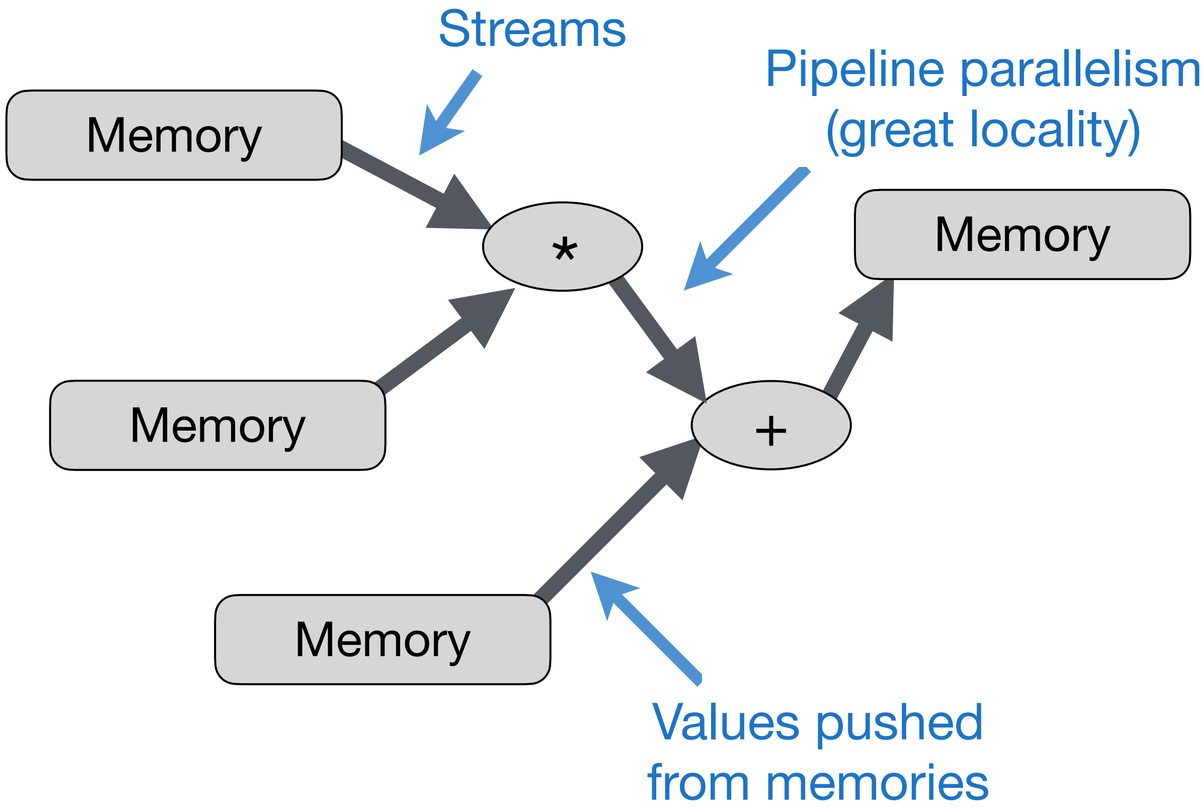


Spatial Hardware (a.k.a. streaming dataflow hardware)

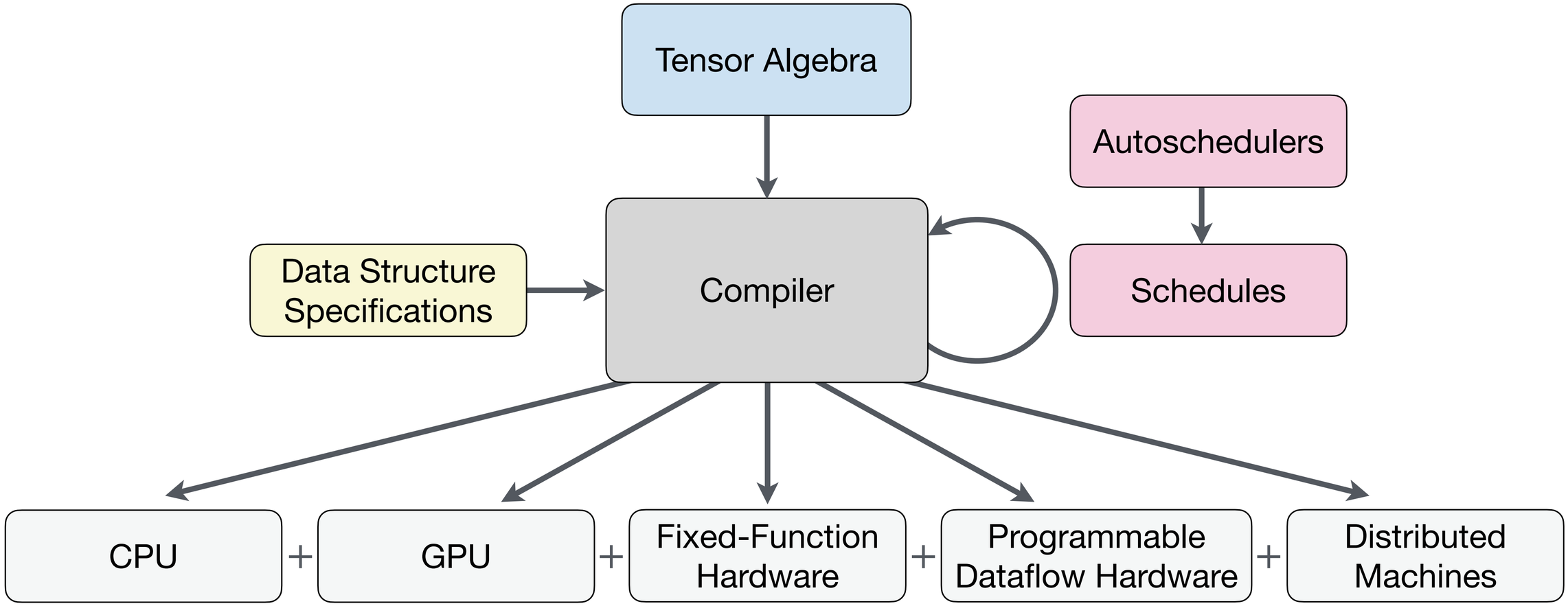
von Neumann architecture



Spatial architecture



Overview: Sparse Tensor Algebra Compilation

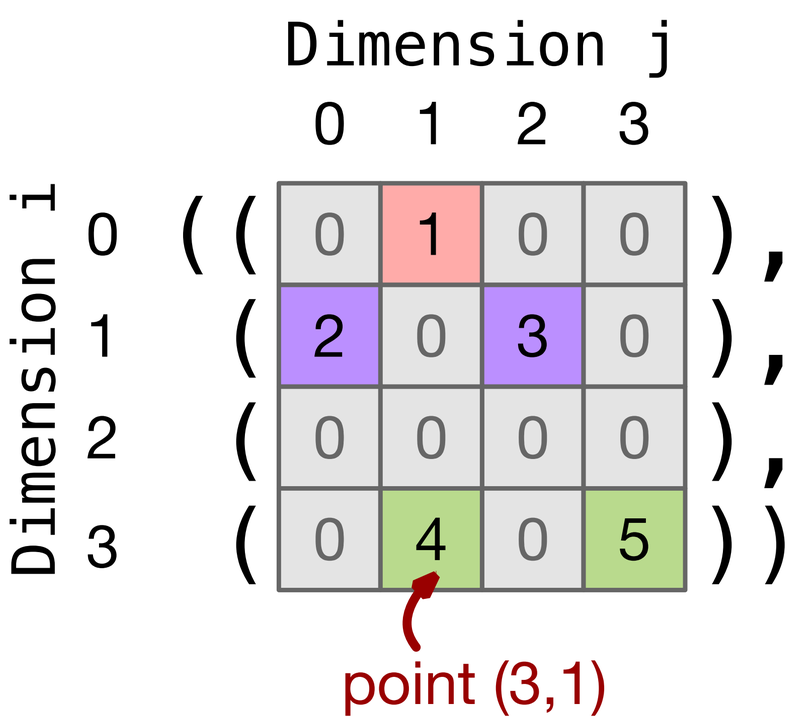


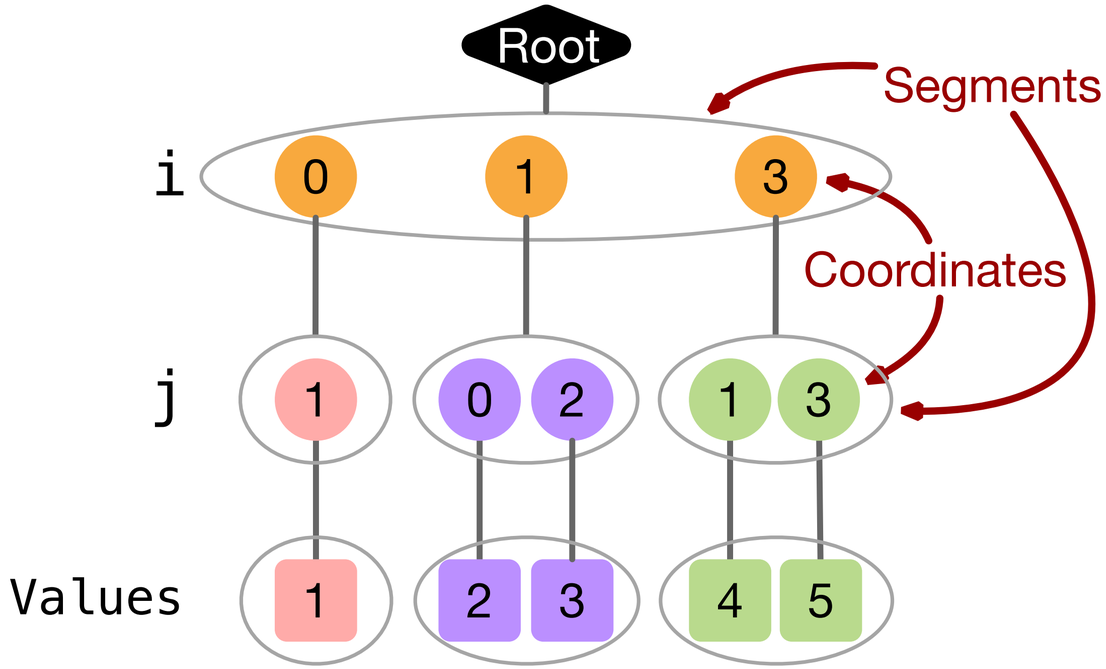
Hardware design for general sparse tensor operations

Sparse tensor algebra accelerators must support:

1. **Generality:** arbitrary tensor algebra operations
2. **Data Structures:** dense and sparse data structures
3. **Fusion:** Fusion across operations
4. **Reordering:** Changing the order they process tensor dimensions

Abstract tensor data model



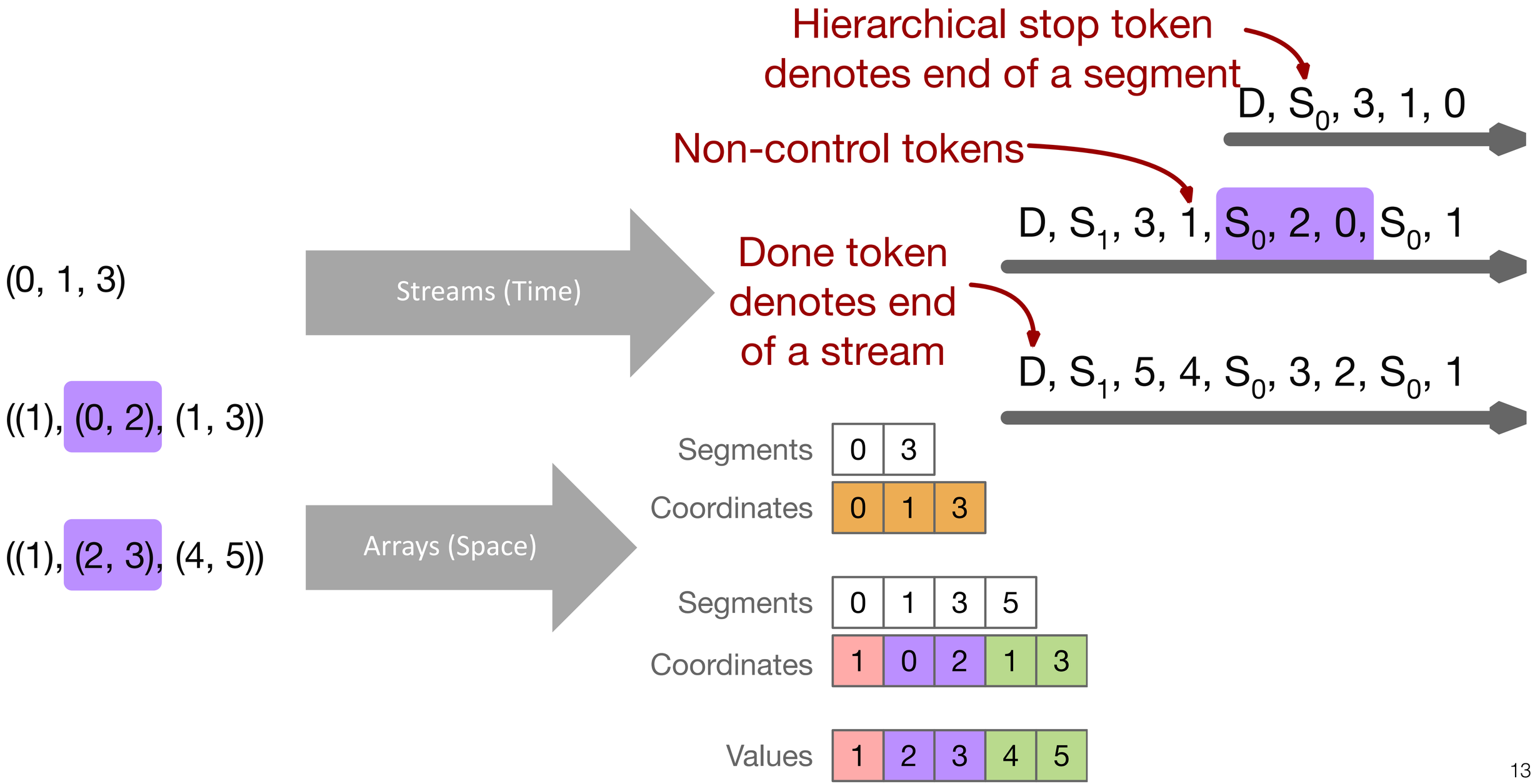


(0, 1, 3)

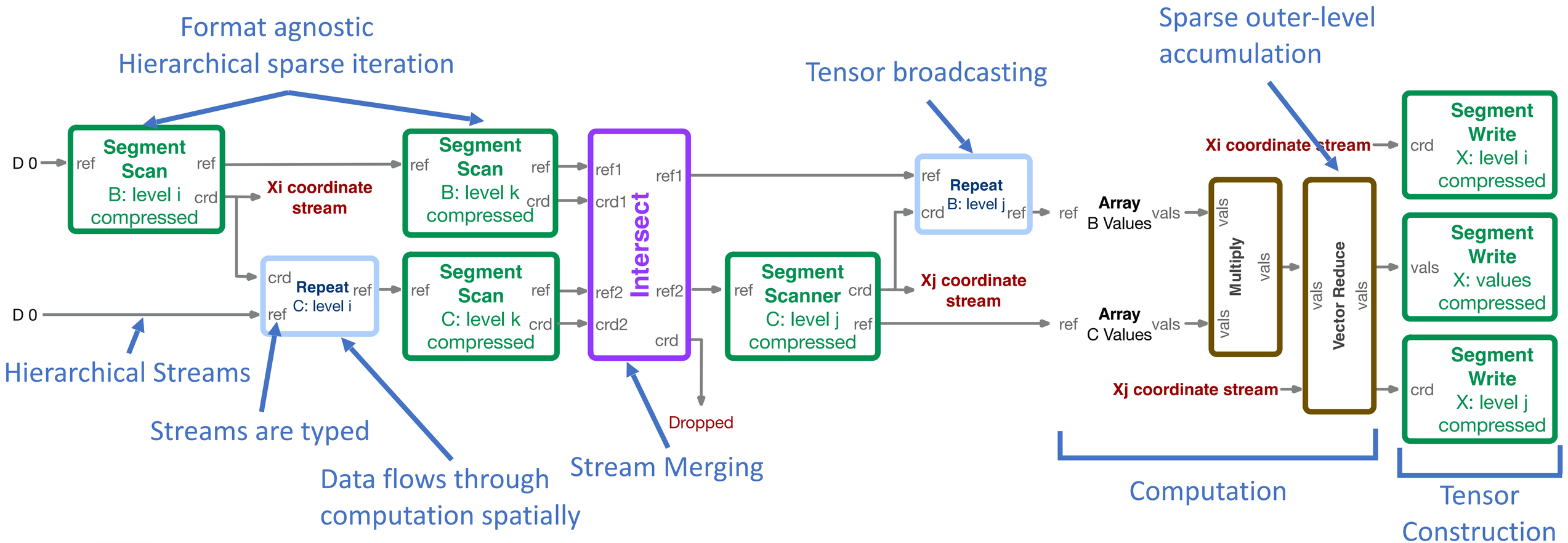
((1), (0, 2), (1, 3))

((1), (2, 3), (4, 5))

Tensors on Wires



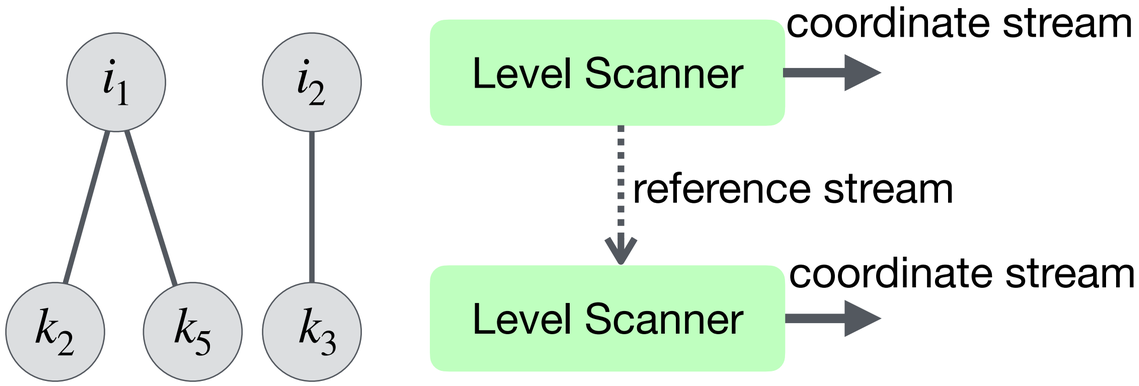
Sparse-matrix sparse-matrix multiplication



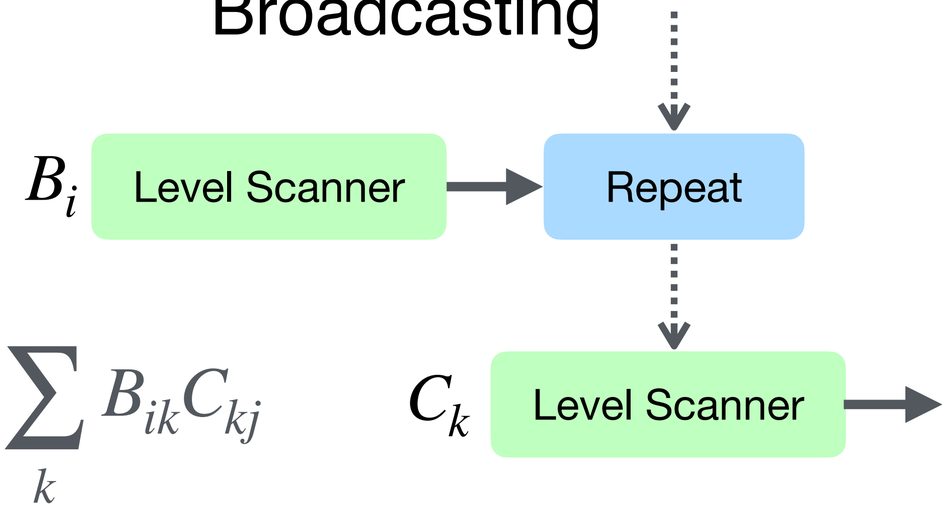
$$\forall_i \forall_k \forall_j \quad X_{ij} += B_{ik} * C_{kj}$$

Streaming dataflow abstract machine (for sparse tensor algebra)

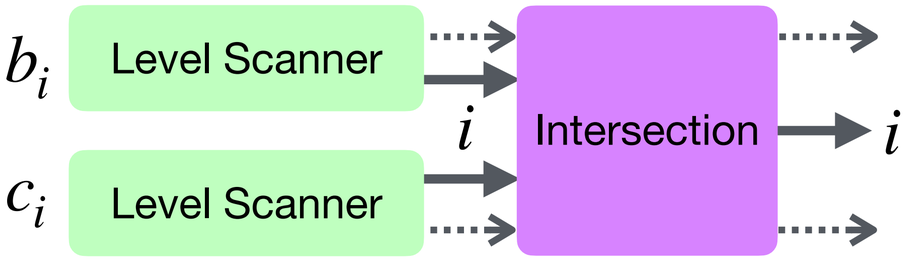
Hierarchical Iteration

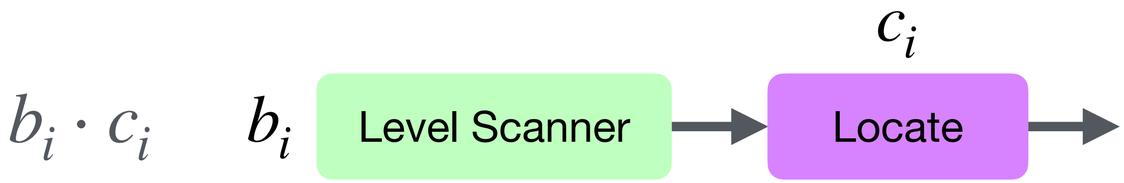


Broadcasting

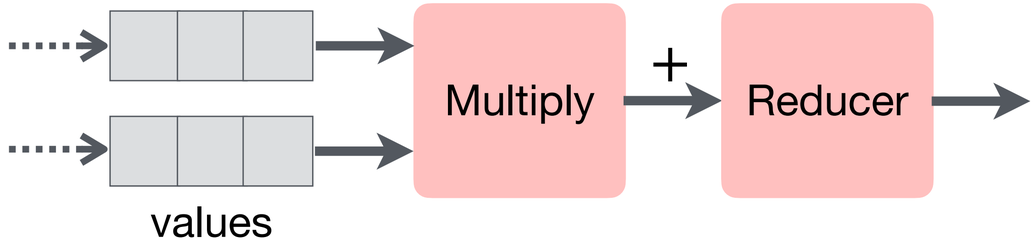


Coiteration / Locate





Computation / Reduction



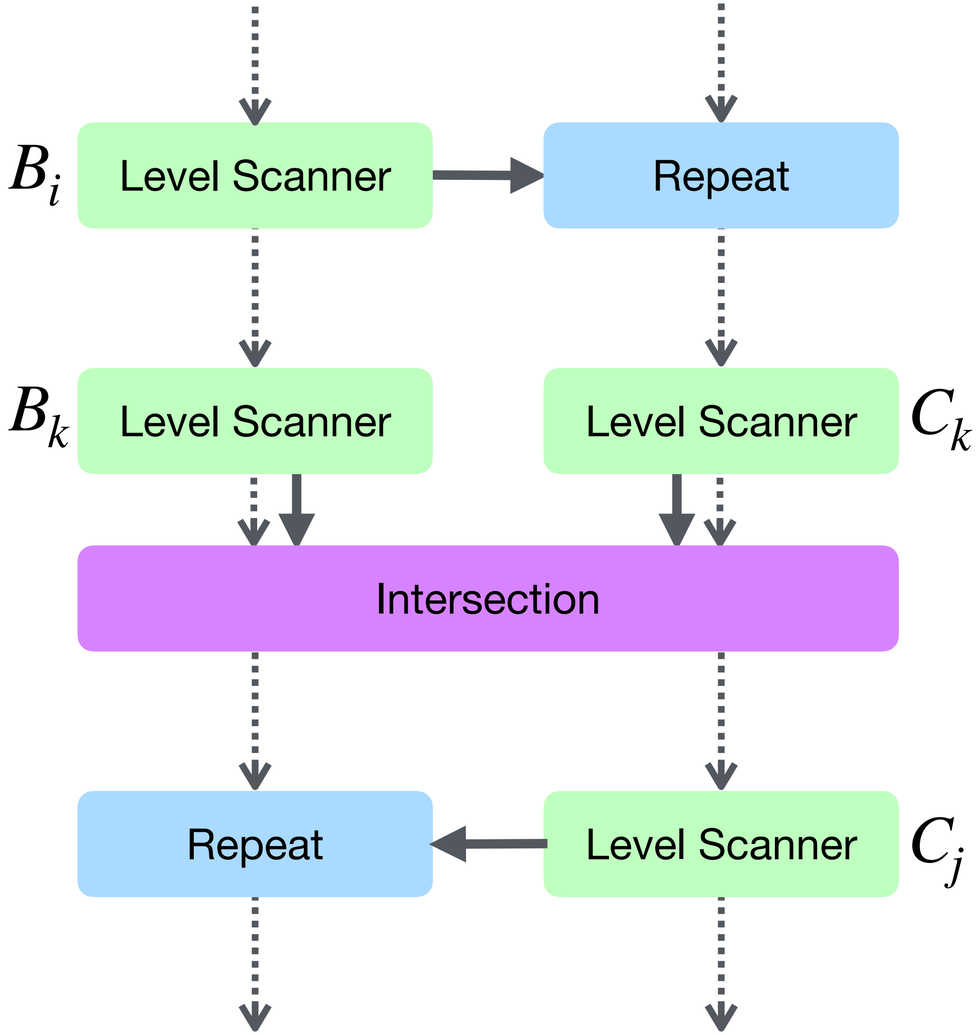
Compiling to the streaming dataflow abstract machine

$$\forall_i \in B_i \cap \mathbb{U}_i$$

$$\forall_k \in B_k \cap C_k$$

$$\forall_j \in \mathbb{U}_j \cap C_j$$

$$A_{ij} \mathrel{+}= B_{ik} C_{kj}$$

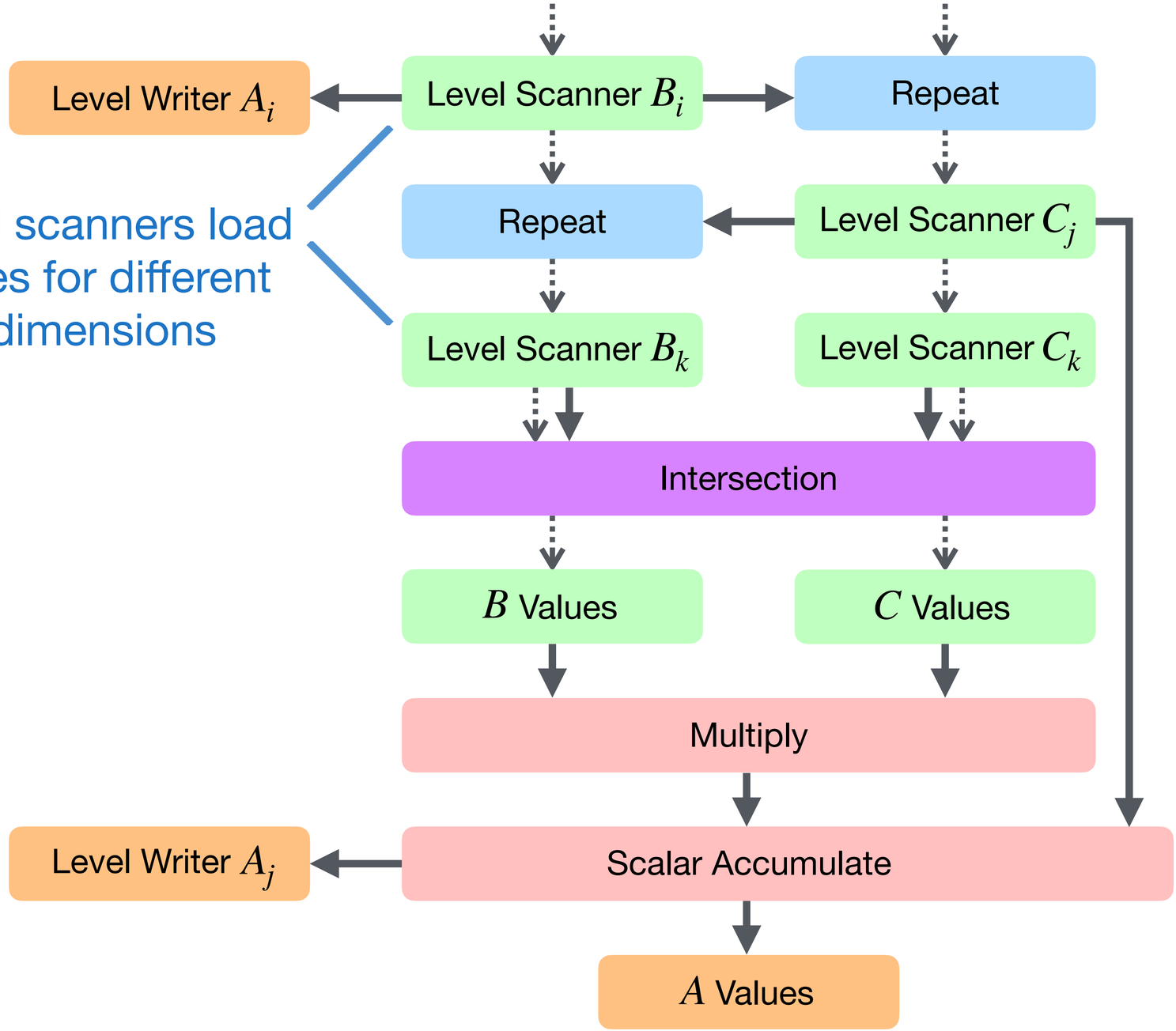


computation graph
for scalar expression

Inner-product sparse matrix multiplication

$$A_{ij} = \sum_k B_{ik} C_{kj}$$

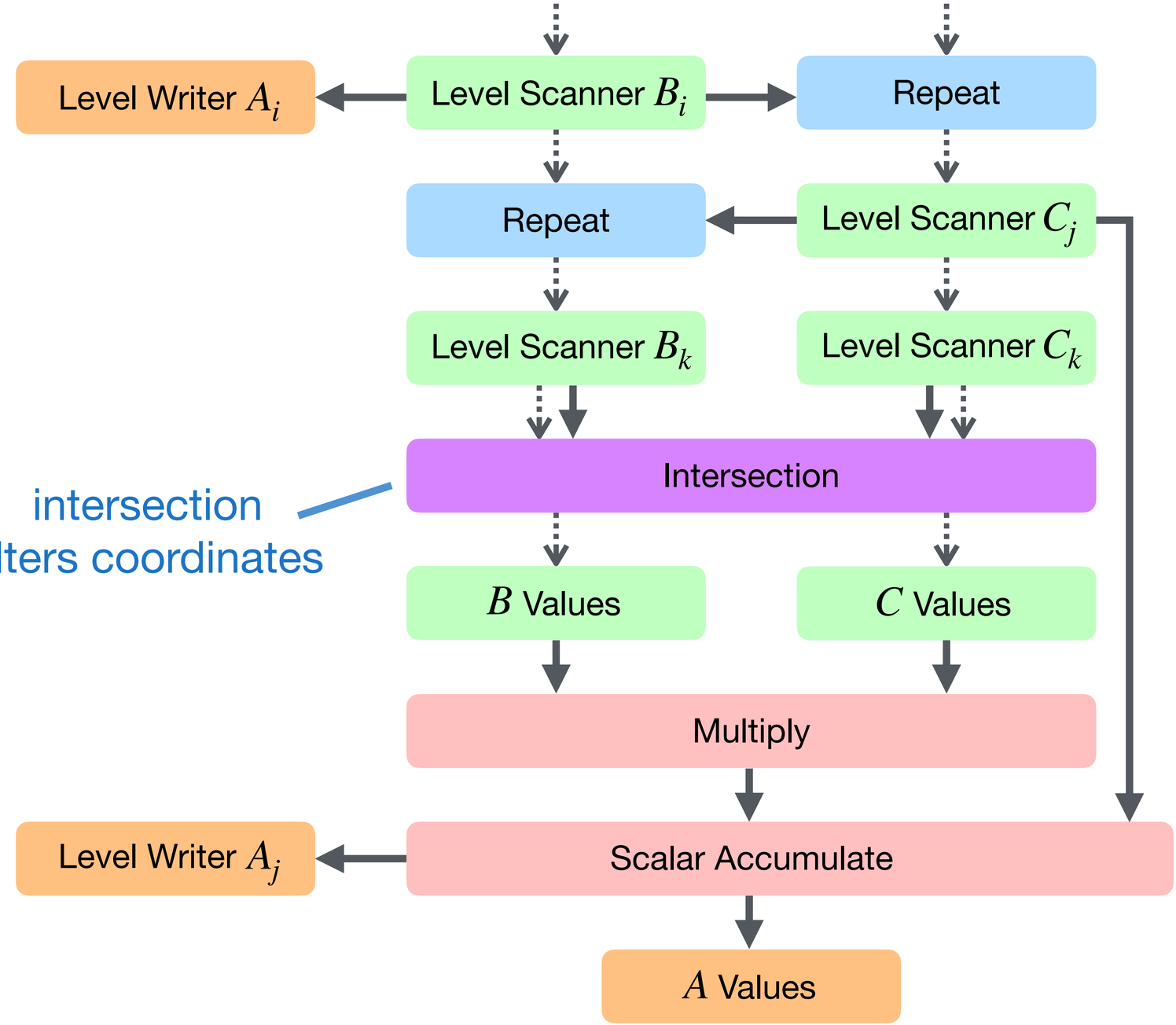
hierarchical scanners load coordinates for different tensor dimensions



Inner-product sparse matrix multiplication

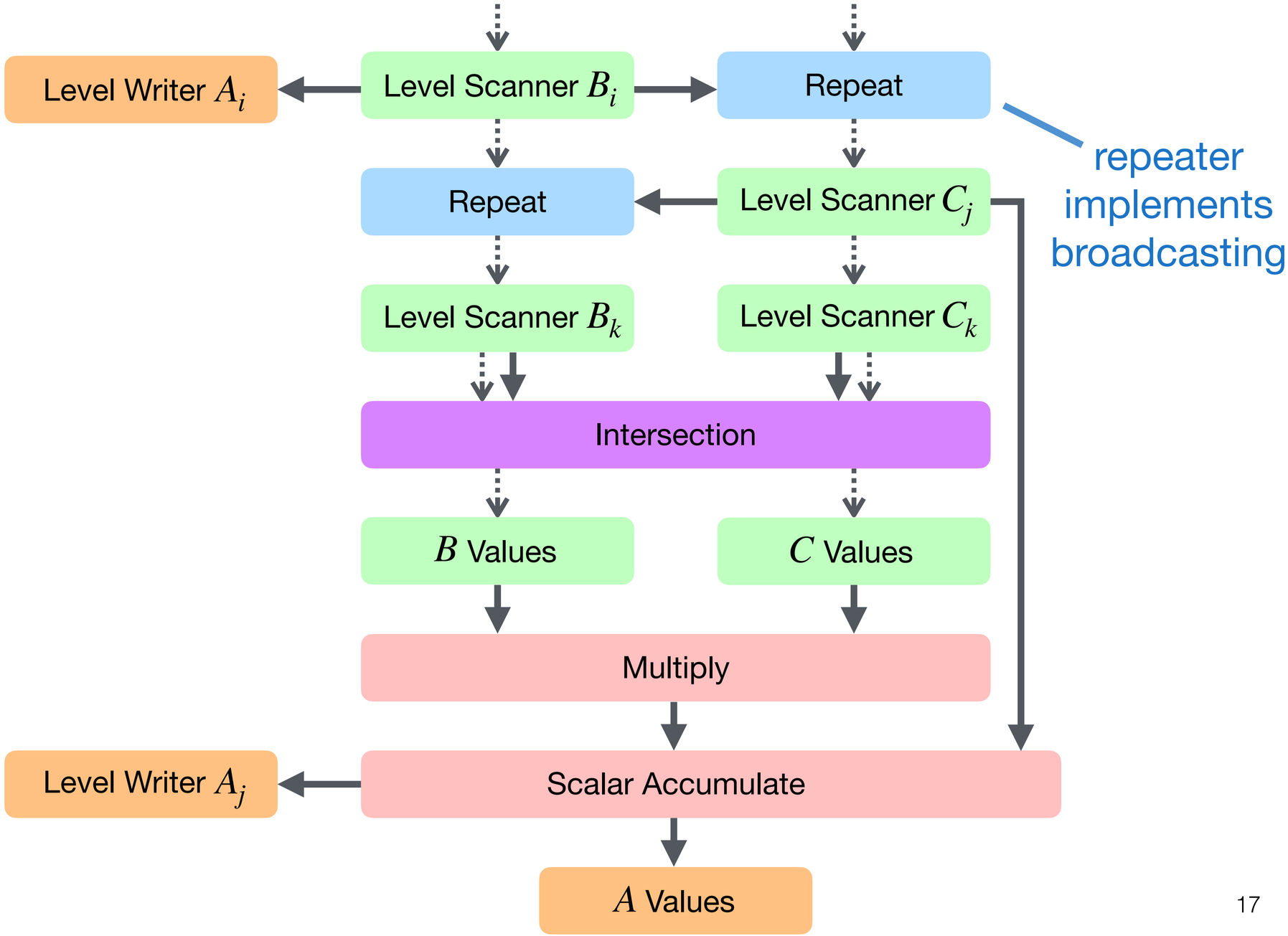
$$A_{ij} = \sum_k B_{ik} C_{kj}$$

intersection filters coordinates



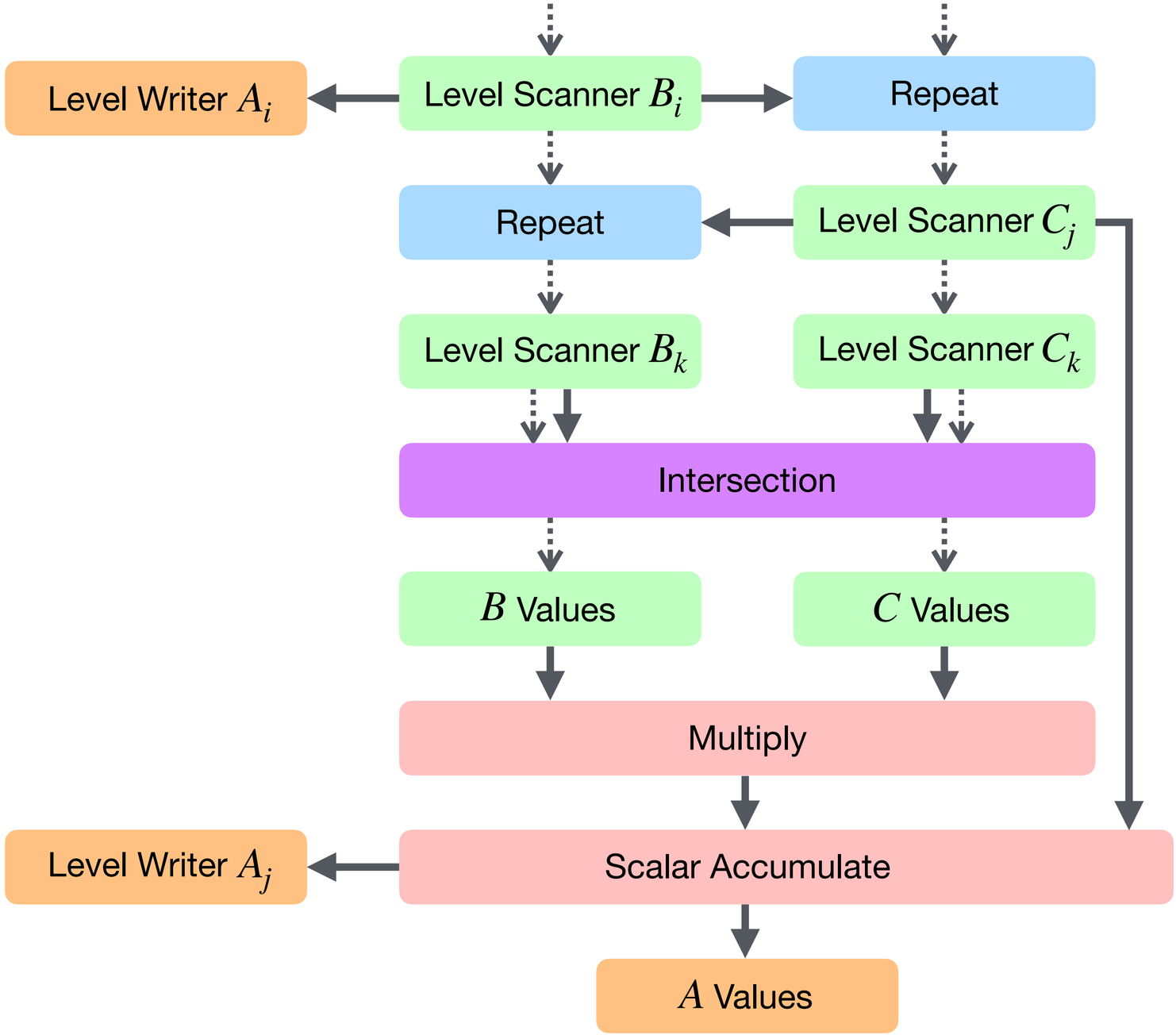
Inner-product sparse matrix multiplication

$$A_{ij} = \sum_k B_{ik} C_{kj}$$



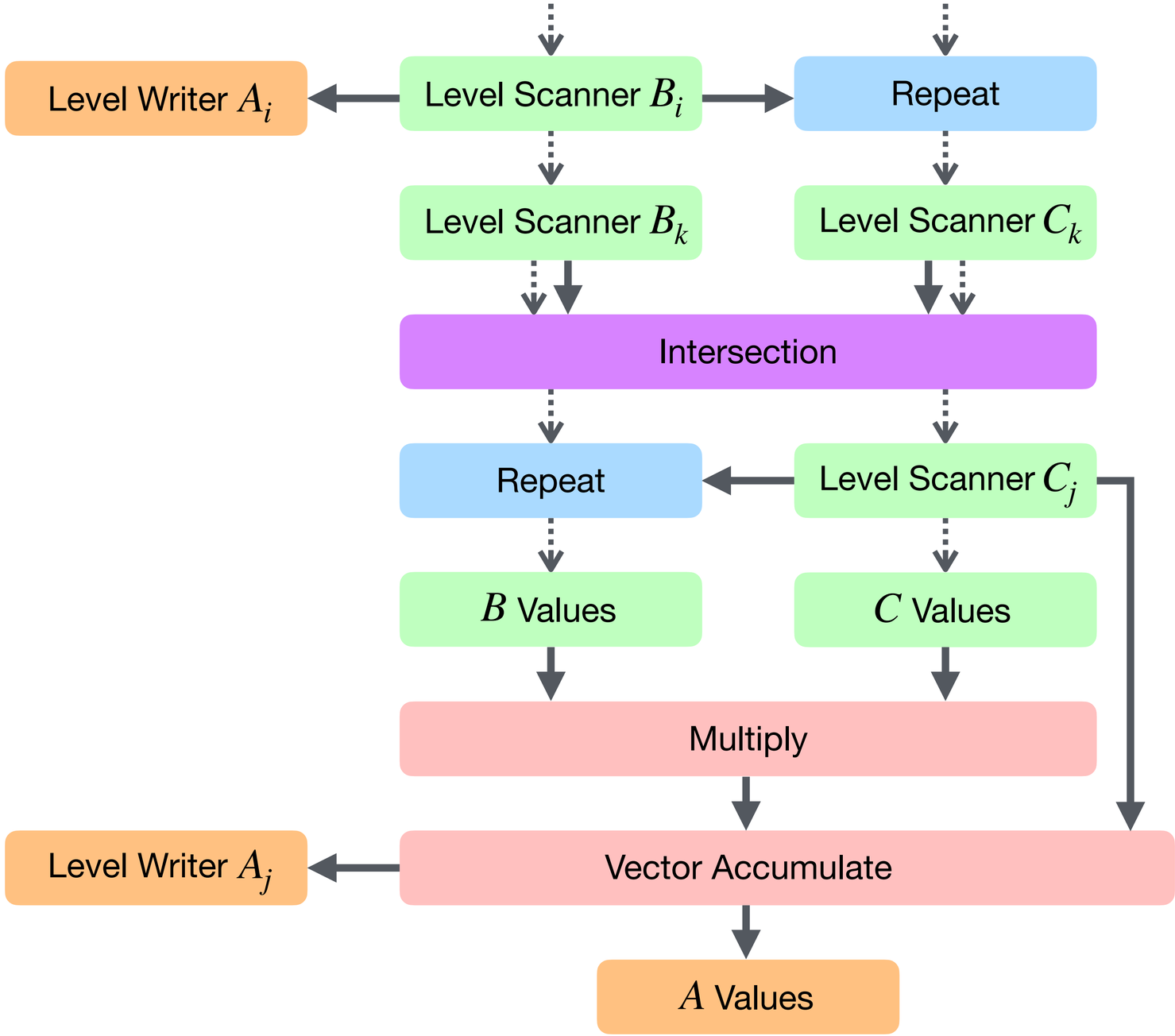
Inner-product sparse matrix multiplication

$$A_{ij} = \sum_k B_{ik} C_{kj}$$



Gustafson sparse matrix multiplication

$$A_{ij} = \sum_k B_{ik} C_{kj}$$

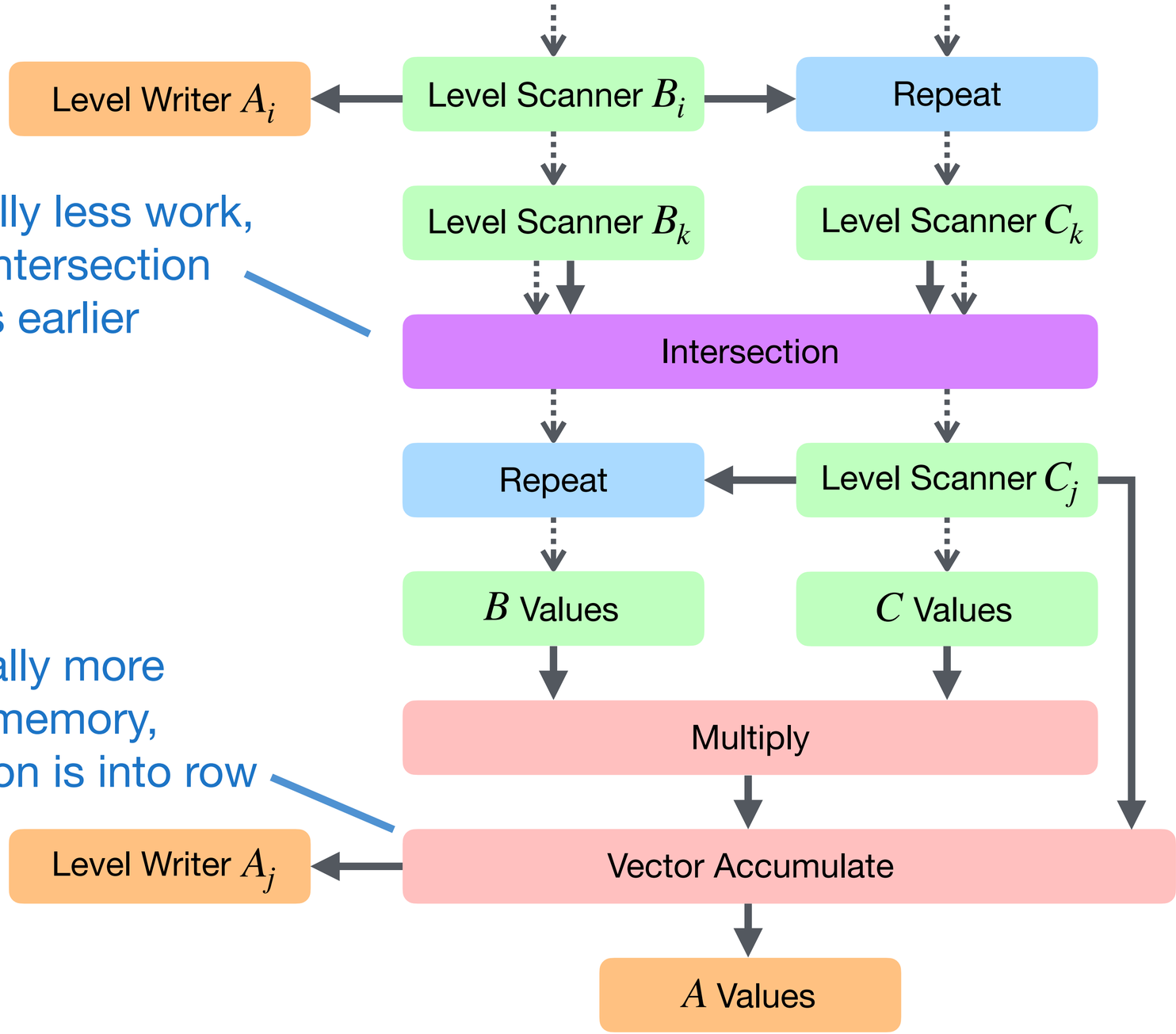


Gustafson sparse matrix multiplication

$$A_{ij} = \sum_k B_{ik} C_{kj}$$

asymptotically less work, because intersection occurs earlier

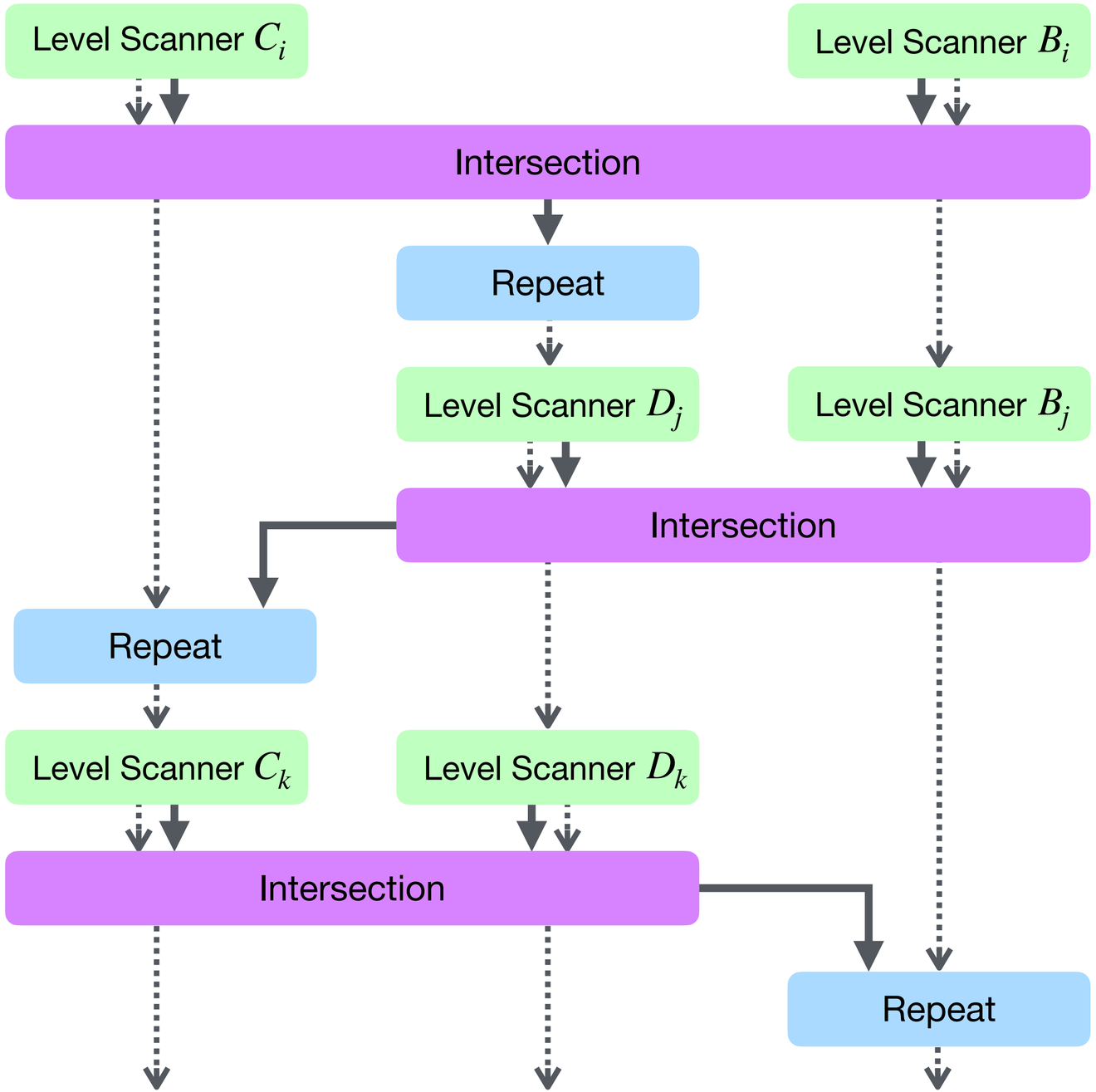
asymptotically more temporary memory, because reduction is into row



Fused SDDMM

$$A = B \odot (CD)$$

$$O(\text{NNZ}_B \cdot K)$$



Sparse Dataflow Compiler Overview