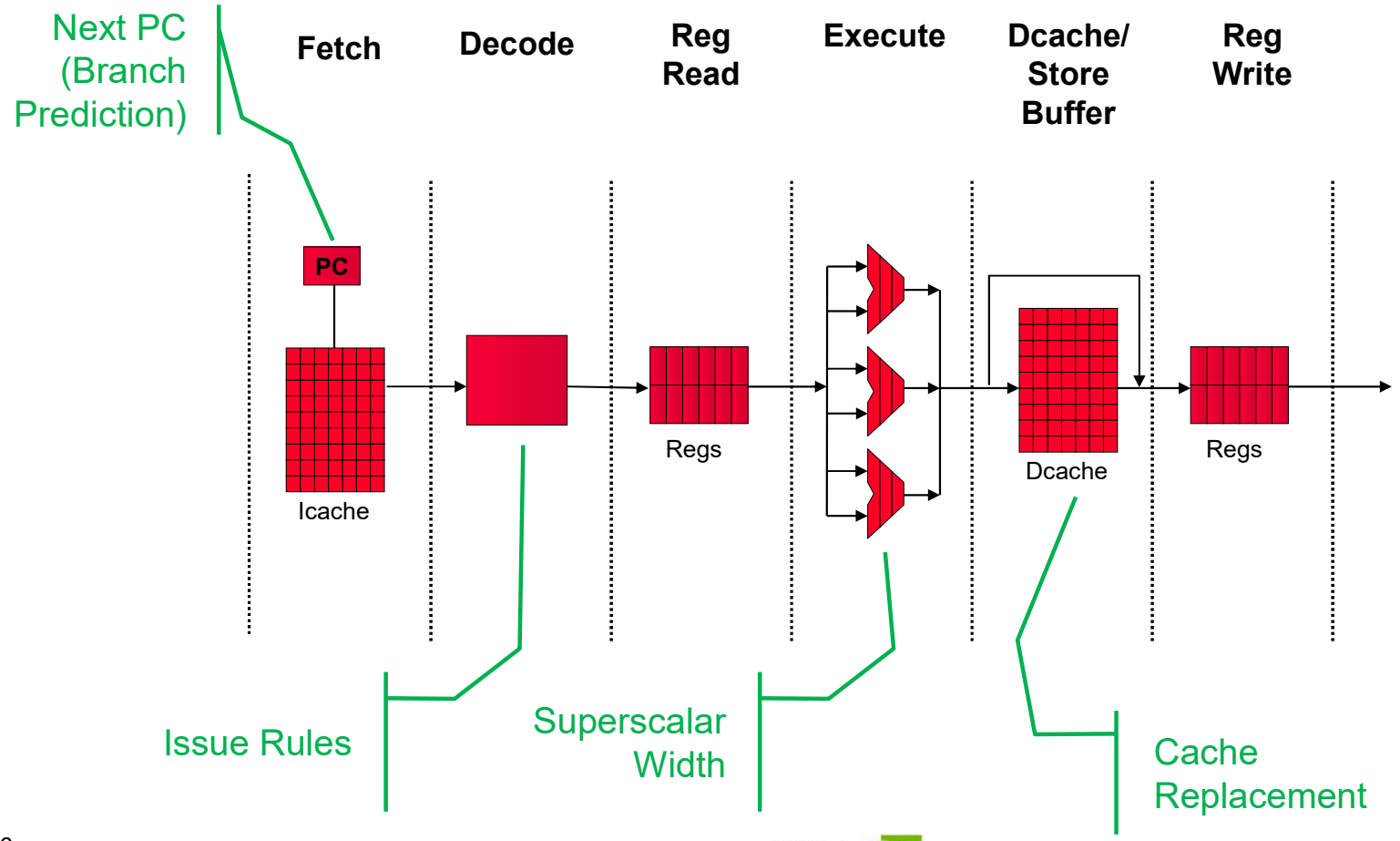
A Structured Approach to the Hardware/Software Co-Design of an AI-Accelerator

November 1, 2026

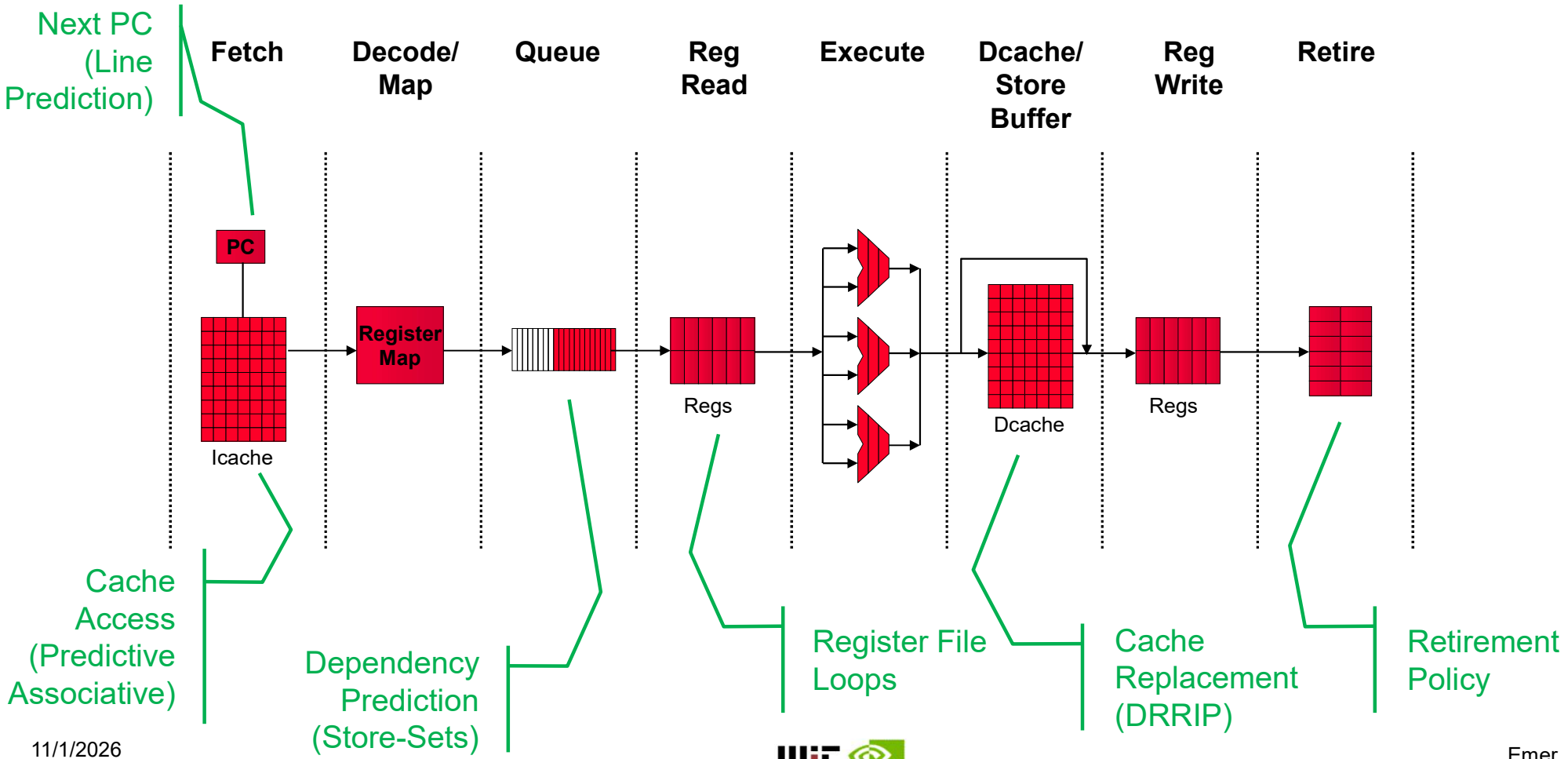
Joel S. Emer
MIT/Nvidia



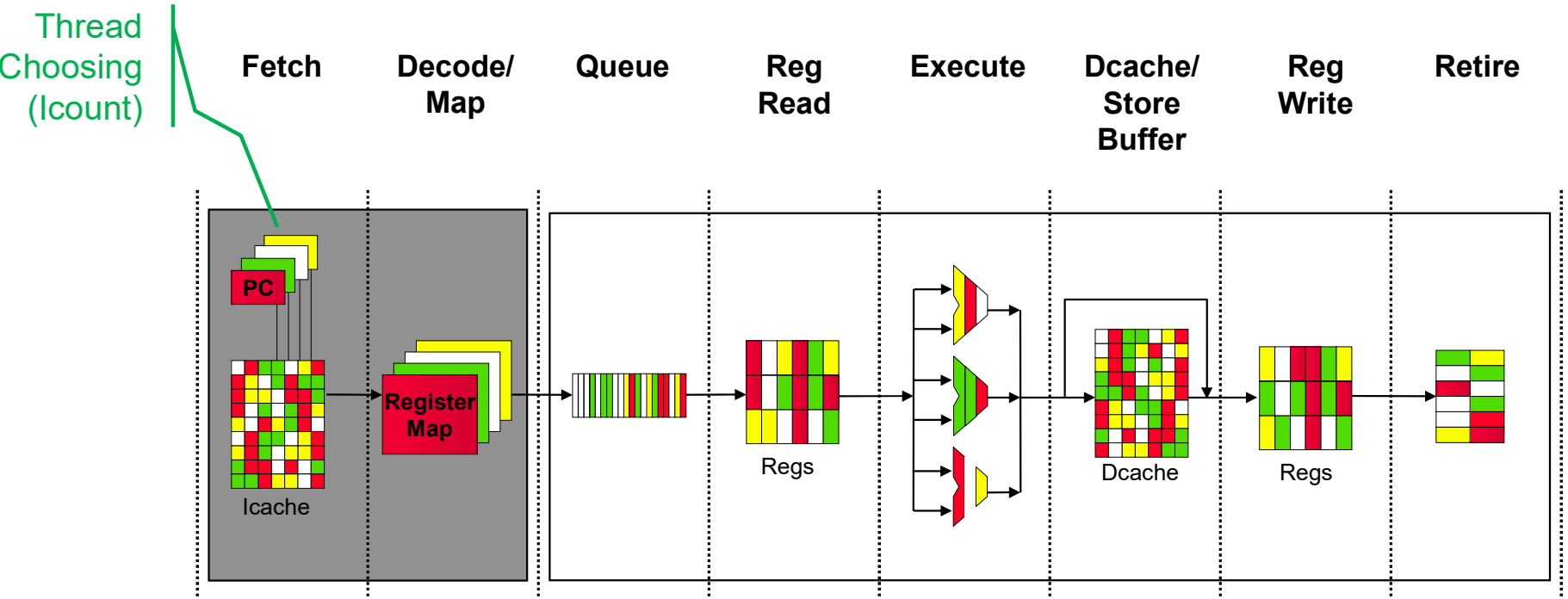
Simple In-Order Pipeline



Basic Out-of-order Pipeline

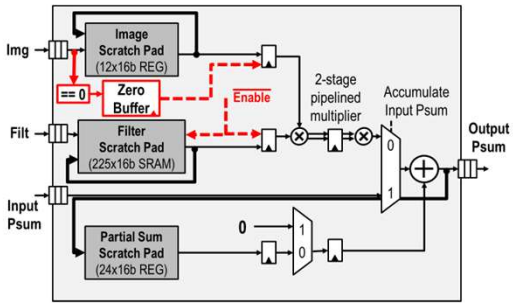


Out-of-Order SMT Pipeline

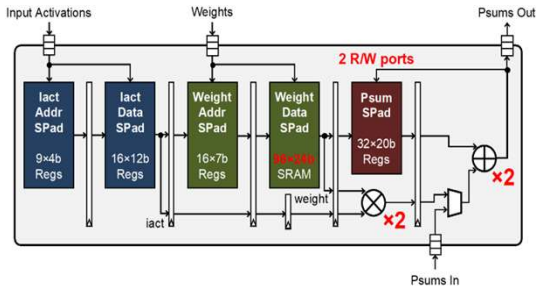


Every Accelerator is Unique

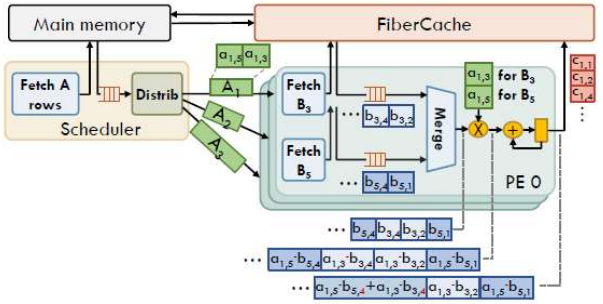
Selected tensor accelerator designs



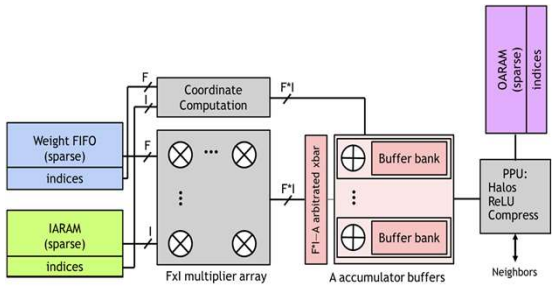
Eyeriss [JSSC2017]



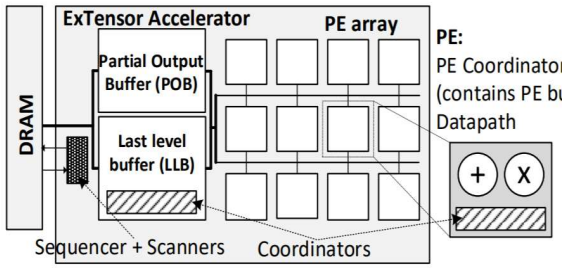
Eyeriss V2 [JETCAS2019]



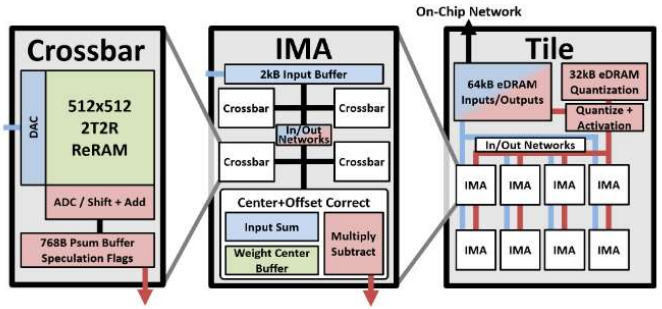
Gamma [ASPLOS2021]



SCNN [ISCA2017]



ExTensor [MICRO2019]



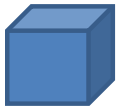
RAELLA [ISCA2023]

"Notation as a Tool for Thought"

- Ken Iverson, 1979

Tensors

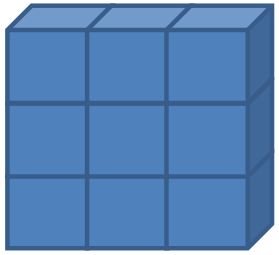
Rank-0: Scalar



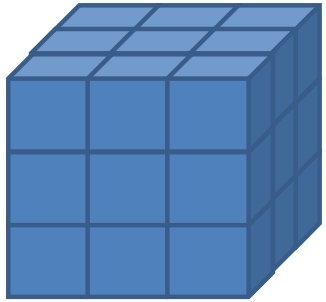
Rank-1: Vector



Rank-2: Matrix



Rank-3: Cube



$$A_{i,j,k}$$

Rank variables

Matrix Multiply – Loop Nest

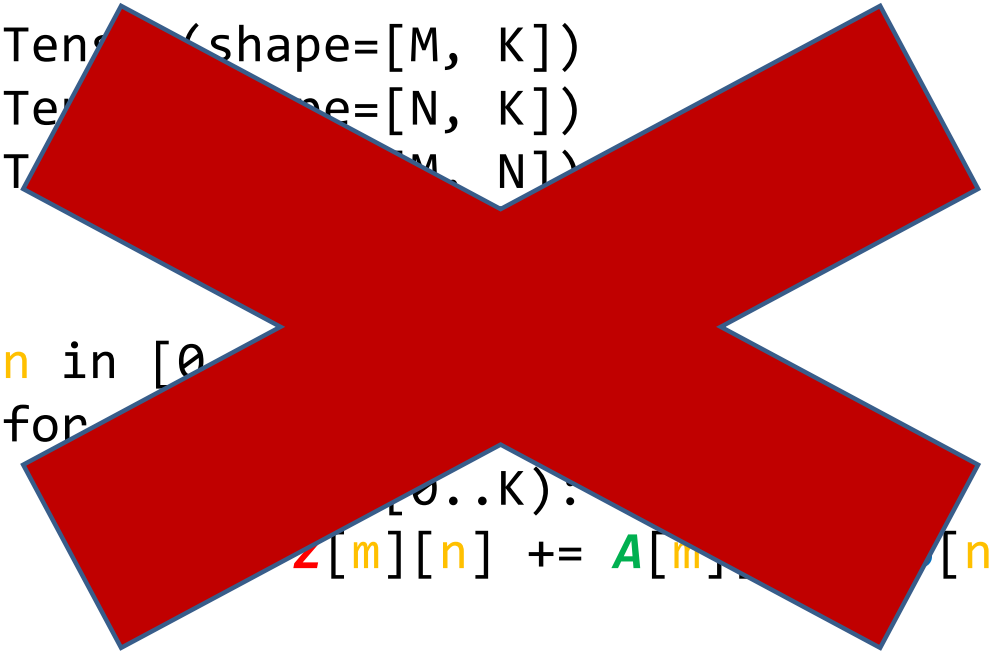
```
A = Tensor(shape=[M, K])
B = Tensor(shape=[N, K])
Z = Tensor(shape=[M, N])


for n in [0..N):
    for m in [0..M):
        for k in [0..K):
            Z[m][n] += A[m][k] × B[n][k]
```

Matrix Multiply – Loop Nest

```
A = Ten(shape=[M, K])
B = Te  e=[N, K])
Z = T   M, N])

for n in [0
    for
        [0..K):
            Z[m][n] += A[m]  [n][k]
```



Einsum – Matrix Multiply

$$Z_{m,n} = A_{m,k} \times B_{n,k}$$

[Relativity, Einstein, Annelen de Physik, 1916]

Einsum – Matrix Multiply

$$Z_{m,n} = A_{m,k} \times B_{n,k}$$

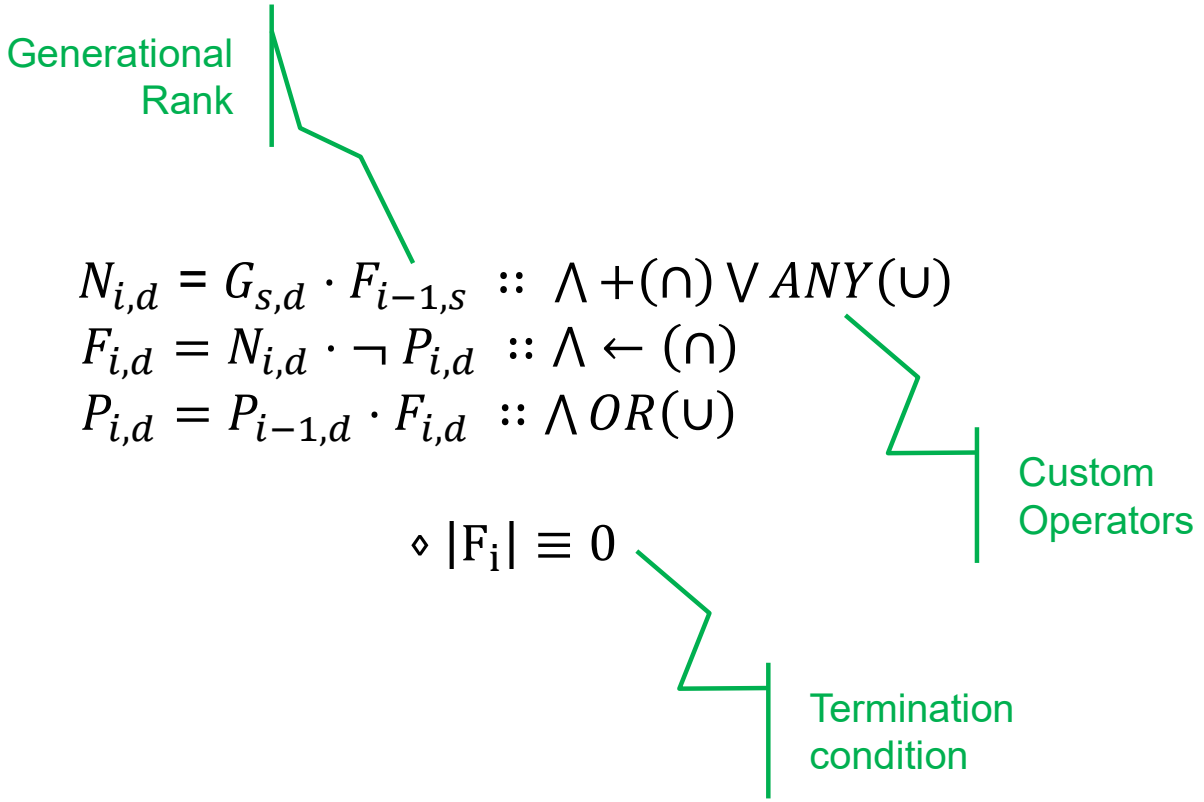
Operational Definition for Einsums (ODE):

- Traverse all points in space of all legal rank variable (index) values (iteration space)
- At each point in iteration space:
  - Calculate value on right hand at specified rank variable values for each operand
  - Assign value to operand at specified rank variable values on left hand side
  - Unless that operand is non-zero, then reduce value into it

[Relativity, Einstein, Annelen de Physik, 1916]
[Numpy/Einsum Python, ~2015]
[TACO, Kjolstad et.al., ASE 2017]
[Timeloop, Parashar et.al., ISPASS 2019]
[SAM, Hsu et.al., ASPLOS 2023]
[TeAAL, Nayak et.al., MICRO 2023]

[Orojenesis, Huang et.al, ISCA, 2024]
[FuseMAX, Nayak, et.al., MICRO, 2024]
[Looptree, Gilbert et.al., TCAI, 2025]
[Fast and Fusiest, Andrulis et.al., ArXiv, 2025]
[Insum, Won et.al, ASPLOS, 2026]
[Turbocharged, Andrulis et.al, MICRO, 2026]

Extensions for Breadth First Search (BFS)



[EDGE-BFS, Odemuyiwa et.al. ArXiv, 2026

More Generational Ranks - Mamba Cascade

▷Initializations:

$$H_{l,b,0,d,n} = 0 \tag{1}$$

$$LEX_{l=0,b,i,ed} = ET_{v,ed} \cdot Tokens_{b,i,v} :: \bigwedge \leftarrow (\cap) \bigvee \mathrm{ANY}_v \tag{2}$$

▷Extended Einsum:

$$\text{—Embeddings} \tag{3}$$

$$LEX_{l,b,i,ed} = EY_{l-1,b,i,ed} + LEX_{l-1,b,i,ed} \tag{4}$$

$$\text{—self.norm(x)} \tag{5}$$

$$PX_{l,b,i,ed} = LEX_{l,b,i,ed} * LEX_{l,b,i,ed} \tag{6}$$

$$NUM_{l,b,i} = PX_{l,b,i,ed} :: \bigvee +_{ed} \tag{7}$$

$$MEX_{l,b,i} = \frac{NUM_{l,b,i}}{ED} \tag{8}$$

$$SQEX_{l,b,i} = \frac{1}{sqrt(MEX_{l,b,i}+)} \tag{9}$$

$$NEX_{l,b,i,ed} = LEX_{l,b,i,ed} \cdot SQEX_{l,b,i} \cdot WEX_{ed} \tag{10}$$

$$\tag{11}$$

$$\text{—in-proj(x)} \tag{1}$$

$$TX_{l,b,i,d} = WTX_{l,ed,d} \cdot NEX_{l,b,i,ed} \tag{2}$$

$$RX_{l,b,i,d} = WRX_{l,ed,d} \cdot NEX_{l,b,i,ed} \tag{3}$$

$$\text{—conv1d(x)} \tag{4}$$

$$\text{Loop order: B, D, I} \tag{5}$$

$$TTX_{l,b,i,d} = WConv_{l,d,f} \cdot TX_{l,b,i-f,d} + BTX_{l,d} \tag{6}$$

$$\text{—silu(x)} \tag{7}$$

$$X_{l,b,i,d} = TTX_{l,b,i,d} \times \sigma(TTX_{l,b,i,d}) \tag{8}$$

$$\tag{9}$$

$$\text{—Discretization} \tag{1}$$

$$T\Delta_{l,b,i,r} = WT\Delta_{l,d,r} \cdot X_{l,b,i,d} \tag{2}$$

$$B_{l,b,i,n} = WB_{l,d,n} \cdot X_{l,b,i,d} \tag{3}$$

$$C_{l,b,i,n} = WC_{l,d,n} \cdot X_{l,b,i,d} \tag{4}$$

$$TT\Delta_{l,b,i,d} = W\Delta_{l,r,d} \cdot T\Delta_{l,b,i,r} + B\Delta_d \tag{5}$$

$$\text{—- softplus to get } \Delta \tag{6}$$

$$\Delta_{l,b,i,d} = \frac{1}{\beta} \times \log(1_{l,b,i,d} + exp(\beta \times TT\Delta_{l,b,i,d})) \tag{7}$$

$$\text{—- Main Recurrence —-} \tag{8}$$

$$\bar{A}_{l,b,i,d,n} = exp(\Delta_{l,b,i,d} \times A_{l,d,n}) \tag{9}$$

$$\bar{B}_{l,b,i,d,n} = \Delta_{l,b,i,d} \cdot B_{l,b,i,n} \tag{10}$$

$$HX_{l,b,i,d,n} = \bar{B}_{l,b,i,d,n} \cdot X_{l,b,i,d} \tag{11}$$

$$HH_{l,b,i,d,n} = \bar{A}_{l,b,i,d,n} \cdot H_{l,b,i-1,d,n} \tag{12}$$

$$H_{l,b,i,d,n} = HX_{l,b,i,d,n} + HH_{l,b,i,d,n} \tag{13}$$

$$S6Y_{l,b,i,d} = C_{l,b,i,n} \cdot H_{l,b,i,d,n} \tag{14}$$

$$LY_{l,b,i,d} = S6Y_{l,b,i,d} + D_d \cdot X_{l,b,i,d} \tag{15}$$

$$\text{— mix in part of the original input using silu} \tag{16}$$

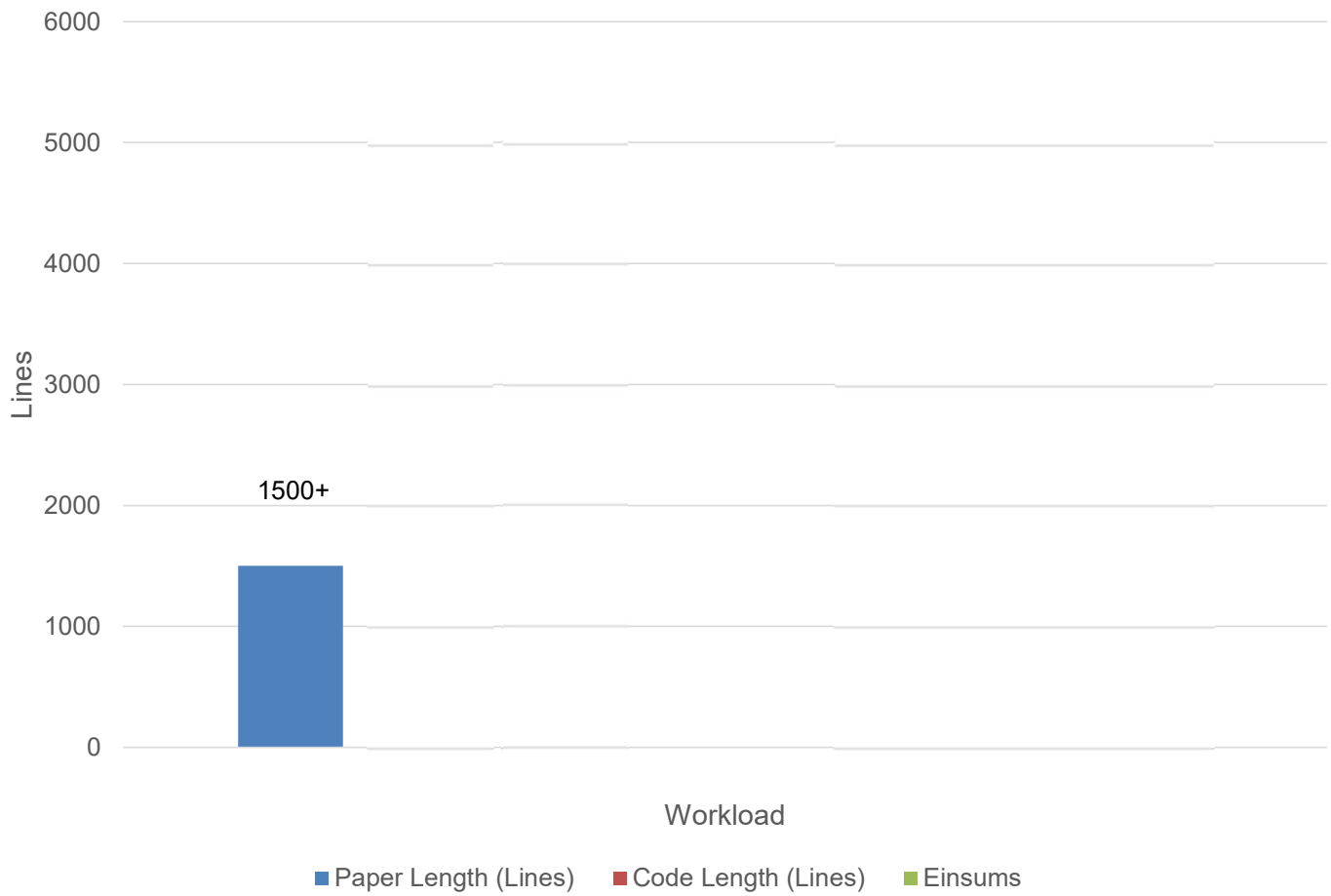
$$Y_{l,b,i,d} = LY_{l,b,i,d} \cdot (RX_{l,b,i,d} \times \sigma(RX_{b,i,d})) \tag{17}$$

$$\tag{18}$$

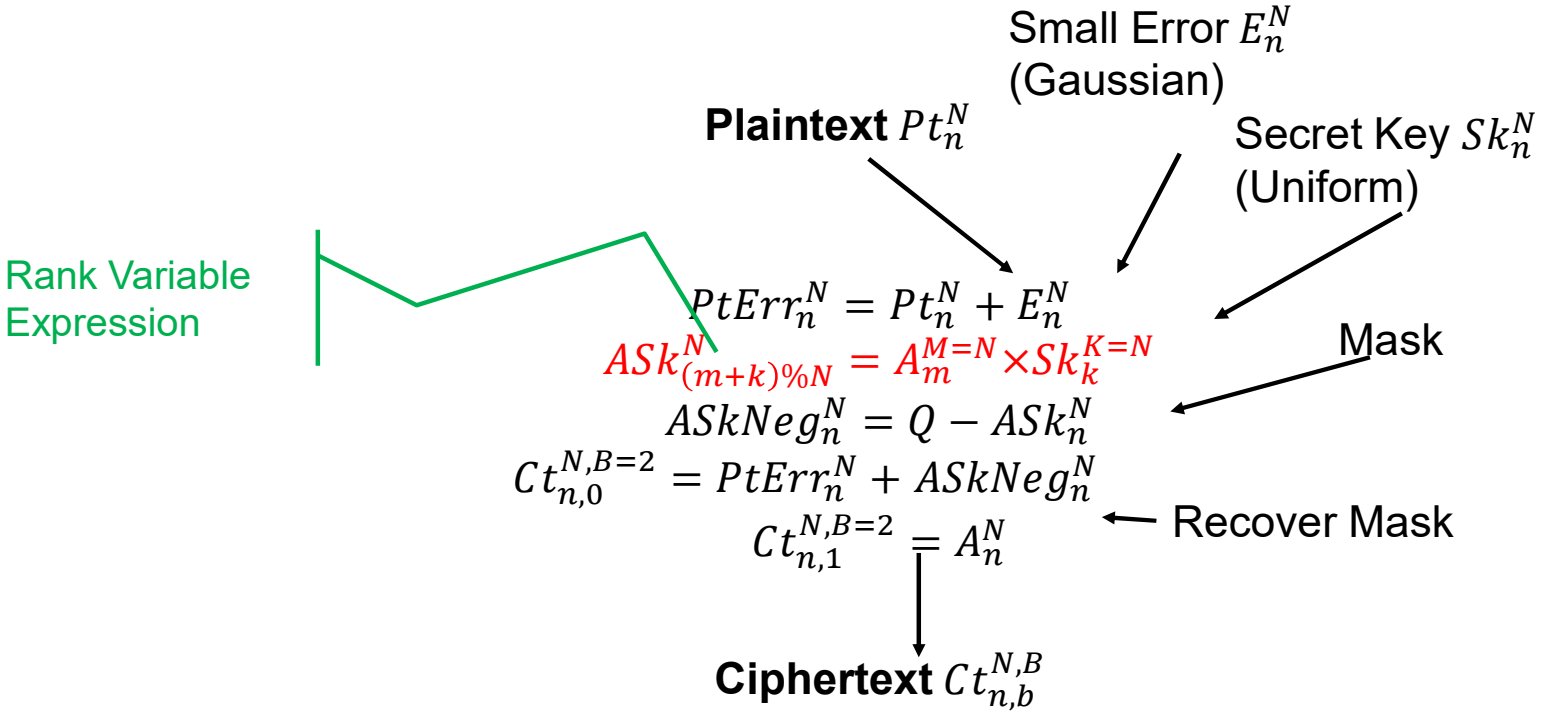
[Odemuyiwa, WIP]



Conciseness of Einsums



Rank Variable Expressions for FHE



Note: All $+$ and $\times$ are modular adds and modular multiplications

[Chen et.al., WIP]

Transformer Attention

Not a precise description of functionality.

This time, composition of many kernels.

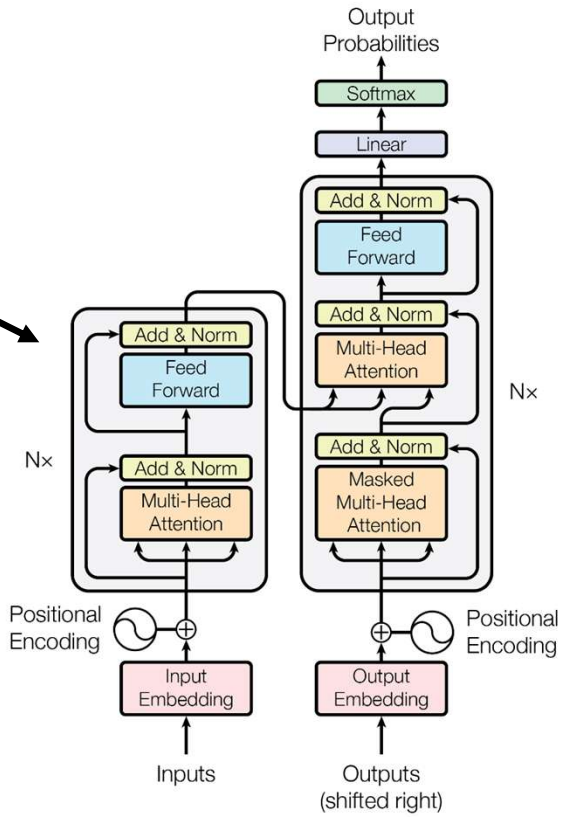


Figure 1: The Transformer - model architecture.

Transformer Attention - Einsums

Attention as a
cascade of Einsums

$$QK_{m,p} = Q_{e,p} \times K_{e,m}$$
$$GM_p = QK_{m,p} :: \bigvee_m \max(\cup)$$
$$SN_{m,p} = e^{QK_{m,p} - GM_p}$$
$$SD_p = SN_{m,p}$$
$$A_{m,p} = SN_{m,p} / SD_p$$
$$AV_{f,p} = A_{m,p} \times V_{f,m}$$

Custom
Operators

Transformer Attention – Linear Algebra

$$Q = X\,W^{Q}$$

$$K = X\,W^{K}$$

$$V = X\,W^{V}$$

$$A = softmax\left(\frac{Q\,K^{T}}{\sqrt{d_k}}\right)$$

$$Z = AV$$

Ideas for Sparse Attention in Transformers

Base transformer [Vaswani et al. 2016]

$$Q_{b,e,h,m} = I_{b,d,m} \times WQ_{d,e,h} : \bigvee_{d} +(\cup)$$

$$K_{b,e,h,m} = I_{b,d,m} \times WK_{d,e,h} : \bigvee_{d} +(\cup)$$

$$V_{b,f,h,m} = I_{b,d,m} \times WV_{d,f,h} : \bigvee_{d} +(\cup)$$

$$QK^{B,H,M,P=M}_{b,h,m,p} = \frac{1}{\sqrt{E}} \times Q^{B,E,H,M}_{b,e,h,p} \times K_{b,e,h,m} : \bigvee_{e} +(\cup)$$

$$SN^{B,H,M,P=M}_{b,h,m,p} = e^{QK_{b,h,m,p}}$$

$$SD^{B,H,P=M}_{b,h,p} = SN_{b,h,m,p} : \bigvee_{m} +(\cup)$$

$$A_{b,h,m,p} = SN_{b,h,m,p} / SD_{b,h,p}$$

$$AV_{b,f,h,p} = A_{b,h,m,p} \times V_{b,f,h,m} : \bigvee_{m} +(\cup)$$

$$C^{B,G=H\times F,P=M}_{b,h\times F+f,p} = AV_{b,f,h,p}$$

$$Z_{b,d,p} = C_{b,g,p} \times WZ_{d,g} : \bigvee_{g} +(\cup)$$

SpAtten [Wang et al. 2020]

$$IP_{i,b,d,m} = MT_{i,b,m} \times IKV_{b,d,m}$$

$$Q_{i,b,e,h,r} = MH_{i,b,h} \times IQ_{b,d,r} \times WQ_{d,e,h} : \bigvee_{d} +(\cup)$$

$$K_{i,b,e,h,m} = MH_{i,b,h} \times IP_{b,d,m} \times WK_{d,e,h} : \bigvee_{d} +(\cup)$$

$$V_{i,b,f,h,m} = MH_{i,b,h} \times IP_{b,d,m} \times WV_{d,f,h} : \bigvee_{d} +(\cup)$$

$$QK_{i,b,h,m,r} = \frac{1}{\sqrt{E}} \times Q_{i,b,e,h,r} \times K_{i,b,e,h,m} : \bigvee_{e} +(\cup)$$

$$SN_{i,b,h,m,r} = e^{QK_{i,b,h,m,r}}$$

$$SD_{i,b,h,r} = SN_{i,b,h,m,r} : \bigvee_{m} +(\cup)$$

$$A_{i,b,h,m,r} = SN_{i,b,h,m,r} / SD_{i,b,h,r}$$

$$AV_{i,b,f,h,r} = A_{i,b,h,m,r} \times V_{i,b,f,h,m} : \bigvee_{m} +(\cup)$$

$$C^{B,G=H\times F,R}_{i,b,h*F+f,r} = AV_{i,b,f,h,r}$$

$$Z_{i,b,d,r} = C_{i,b,g,r} \times WZ_{d,g} : \bigvee_{g} +(\cup)$$

$$ZA_{i,b,f,h,r} = Z_{i,b,h*F+f,r}$$

$$MT_{i+1,b,m} = prune(A_{i,b,h,m,r}) \times MT_{i,b,m}$$

$$MH_{i+1,b,h} = prune(ZA_{i,b,f,h,r}) \times MH_{i,b,h}$$

$$ST_{i+1,b,m} = ST_{i,b,m} + A_{i,b,h,m,r}$$

$$SH_{i+1,b,h} = SH_{i,b,h} + ZA_{i,b,f,h,r}$$



Ideas for Sparse Attention in Transformers

Base transformer [Vaswani et al. 2016]

$$Q_{b,e,h,m} = I_{b,d,m} \times WQ_{d,e,h} : \bigvee_{d} +(\cup)$$

$$K_{b,e,h,m} = I_{b,d,m} \times WK_{d,e,h} : \bigvee_{d} +(\cup)$$

$$V_{b,f,h,m} = I_{b,d,m} \times WV_{d,f,h} : \bigvee_{d} +(\cup)$$

$$QK^{B,H,M,P=M}_{b,h,m,p} = \frac{1}{\sqrt{E}} \times Q^{B,E,H,M}_{b,e,h,p} \times K_{b,e,h,m} : \bigvee_{e} +(\cup)$$

$$SN^{B,H,M,P=M}_{b,h,m,p} = e^{QK_{b,h,m,p}}$$

$$SD^{B,H,P=M}_{b,h,p} = SN_{b,h,m,p} : \bigvee_{m} +(\cup)$$

$$A_{b,h,m,p} = SN_{b,h,m,p} / SD_{b,h,p}$$

$$AV_{b,f,h,p} = A_{b,h,m,p} \times V_{b,f,h,m} : \bigvee_{m} +(\cup)$$

$$C^{B,G=H\times F,P=M}_{b,h\times F+f,p} = AV_{b,f,h,p}$$

$$Z_{b,d,p} = C_{b,g,p} \times WZ_{d,g} : \bigvee_{g} +(\cup)$$

Sanger [Lu et al. 2021]

$$Q_{b,e,h,m} = I_{b,d,m} \times WQ_{d,e,h} : \bigvee_{d} +(\cup)$$

$$K_{b,e,h,m} = I_{b,d,m} \times WK_{d,e,h} : \bigvee_{d} +(\cup)$$

$$V_{b,f,h,m} = I_{b,d,m} \times WV_{d,f,h} : \bigvee_{d} +(\cup)$$

$$\hat{Q}_{b,e,h,m} = Qt(Q_{b,e,h,m})$$

$$\hat{K}_{b,e,h,m} = Qt(K_{b,e,h,m})$$

$$\hat{QK}^{B,H,M,P=M}_{b,h,m,p} = \frac{1}{\sqrt{E}} \times \hat{Q}^{B,E,H,M}_{b,e,h,p} \times \hat{K}_{b,e,h,m} : \bigvee_{e} +(\cup)$$

$$\hat{SN}_{b,h,m,p} = e^{\hat{QK}_{b,h,m,p}}$$

$$\hat{SD}_{b,h,p} = \hat{SN}_{b,h,m,p} : \bigvee_{m} +(\cup)$$

$$\hat{A}_{b,h,m,p} = \hat{SN}_{b,h,m,p} / \hat{SD}_{b,h,p}$$

$$M_{b,h,m,p} = prune(\hat{A}_{b,h,m,p})$$

$$QK^{B,H,M,P=M}_{b,h,m,p} = \frac{1}{\sqrt{E}} \times M_{b,h,m,p} \times Q^{B,E,H,M}_{b,e,h,p} \times K_{b,e,h,m} : \bigvee_{e} +(\cup)$$

$$SN^{B,H,M,P=M}_{b,h,m,p} = e^{QK_{b,h,m,p}}$$

$$SD^{B,H,P=M}_{b,h,p} = SN_{b,h,m,p} : \bigvee_{m} +(\cup)$$

$$A^{B,H,M,P=M}_{b,h,m,p} = SN_{b,h,m,p} / SD_{b,h,p}$$

$$AV^{B,F,H,P=M}_{b,f,h,p} = A_{b,h,m,p} \times V_{b,f,h,m} : \bigvee_{m} +(\cup)$$

$$C^{B,G=H\times F,P=M}_{b,h*F+f,p} = AV_{b,f,h,p}$$

$$Z^{B,D,P=M}_{b,d,p} = C_{b,g,p} \times WZ_{d,g} : \bigvee_{g} +(\cup)$$

Ideas for Sparse Attention in Transformers

Base transformer [Vaswani et al. 2016]

$$Q_{b,e,h,m} = I_{b,d,m} \times WQ_{d,e,h} : \bigvee_{d} +(\cup)$$

$$K_{b,e,h,m} = I_{b,d,m} \times WK_{d,e,h} : \bigvee_{d} +(\cup)$$

$$V_{b,f,h,m} = I_{b,d,m} \times WV_{d,f,h} : \bigvee_{d} +(\cup)$$

$$QK^{B,H,M,P=M}_{b,h,m,p} = \frac{1}{\sqrt{E}} \times Q^{B,E,H,M}_{b,e,h,p} \times K_{b,e,h,m} : \bigvee_{e} +(\cup)$$

$$SN^{B,H,M,P=M}_{b,h,m,p} = e^{QK_{b,h,m,p}}$$

$$SD^{B,H,P=M}_{b,h,p} = SN_{b,h,m,p} : \bigvee_{m} +(\cup)$$

$$A_{b,h,m,p} = SN_{b,h,m,p} / SD_{b,h,p}$$

$$AV_{b,f,h,p} = A_{b,h,m,p} \times V_{b,f,h,m} : \bigvee_{m} +(\cup)$$

$$C^{B,G=H\times F,P=M}_{b,h\times F+f,p} = AV_{b,f,h,p}$$

$$Z_{b,d,p} = C_{b,g,p} \times WZ_{d,g} : \bigvee_{g} +(\cup)$$

EdgeBERT [Tambe et al. 2021]

$$MH_h = take(M_{h,m,p}, zeroes(m,p), 0)$$

$$Q_{b,e,h,m} = MH_h \times I_{b,d,m} \times WQ_{d,e,h} : \bigvee_{d} +(\cup)$$

$$K_{b,e,h,m} = MH_h \times I_{b,d,m} \times WK_{d,e,h} : \bigvee_{d} +(\cup)$$

$$V_{b,f,h,m} = MH_h \times I_{b,d,m} \times WV_{d,f,h} : \bigvee_{d} +(\cup)$$

$$QK^{B,H,M,P=M}_{b,h,m,p} = \frac{1}{\sqrt{E}} \times Q^{B,E,H,M}_{b,e,h,p} \times K_{b,e,h,m} : \bigvee_{e} +(\cup)$$

$$SN^{B,H,M,P=M}_{b,h,m,p} = e^{QK_{b,h,m,p}}$$

$$SD^{B,H,P=M}_{b,h,p} = SN_{b,h,m,p} : \bigvee_{m} +(\cup)$$

$$A_{b,h,m,p} = SN_{b,h,m,p} / SD_{b,h,p}$$

$$AV_{b,f,h,p} = M_{h,m,p} \times A_{b,h,m,p} \times V_{b,f,h,m} : \bigvee_{m} +(\cup)$$

$$C^{B,G=H\times F,P=M}_{b,h*F+f,p} = AV_{b,f,h,p}$$

$$Z^{B,D,P=M}_{b,d,p} = C_{b,g,p} \times WZ_{d,g} : \bigvee_{g} +(\cup)$$

Ideas for Sparse Attention in Transformers

Base transformer [Vaswani et al. 2016]

$$Q_{b,e,h,m} = I_{b,d,m} \times WQ_{d,e,h} : \bigvee_{d} +(\cup)$$

$$K_{b,e,h,m} = I_{b,d,m} \times WK_{d,e,h} : \bigvee_{d} +(\cup)$$

$$V_{b,f,h,m} = I_{b,d,m} \times WV_{d,f,h} : \bigvee_{d} +(\cup)$$

$$QK^{B,H,M,P=M}_{b,h,m,p} = \frac{1}{\sqrt{E}} \times Q^{B,E,H,M}_{b,e,h,p} \times K_{b,e,h,m} : \bigvee_{e} +(\cup)$$

$$SN^{B,H,M,P=M}_{b,h,m,p} = e^{QK_{b,h,m,p}}$$

$$SD^{B,H,P=M}_{b,h,p} = SN_{b,h,m,p} : \bigvee_{m} +(\cup)$$

$$A_{b,h,m,p} = SN_{b,h,m,p} / SD_{b,h,p}$$

$$AV_{b,f,h,p} = A_{b,h,m,p} \times V_{b,f,h,m} : \bigvee_{m} +(\cup)$$

$$C^{B,G=H\times F,P=M}_{b,h\times F+f,p} = AV_{b,f,h,p}$$

$$Z_{b,d,p} = C_{b,g,p} \times WZ_{d,g} : \bigvee_{g} +(\cup)$$

DOTA [Qu et al. 2022]

$$Q_{b,e,h,m} = I_{b,d,m} \times WQ_{d,e,h} : \bigvee_{d} +(\cup)$$

$$K_{b,e,h,m} = I_{b,d,m} \times WK_{d,e,h} : \bigvee_{d} +(\cup)$$

$$V_{b,f,h,m} = I_{b,d,m} \times WV_{d,f,h} : \bigvee_{d} +(\cup)$$

$$\tilde{Q}_{b,k,h,m} = I_{b,d,m} \times P^{D,K}_{d,j} \times \tilde{WQ}_{j,k,h} : \bigvee_{d,j} +(\cup)$$

$$\tilde{K}_{b,k,h,m} = I_{b,d,m} \times P^{D,K}_{d,j} \times \tilde{WK}_{j,k,h} : \bigvee_{d,j} +(\cup)$$

$$\tilde{QK}^{B,H,M,P=M}_{b,h,m,p} = \tilde{Q}^{B,K,H,M}_{b,k,h,p} \times \tilde{K}_{b,k,h,m} : \bigvee_{k} +(\cup)$$

$$M_{b,h,m,p} = prune(\tilde{QK}_{b,h,m,p})$$

Sparse Mask

$$QK^{B,H,M,P=M}_{b,h,m,p} = \frac{1}{\sqrt{E}} \times M_{b,h,m,p} \times Q^{B,E,H,M}_{b,e,h,p} \times K_{b,e,h,m} : \bigvee_{e} +(\cup)$$

$$SN^{B,H,M,P=M}_{b,h,m,p} = e^{QK_{b,h,m,p}}$$

$$SD^{B,H,P=M}_{b,h,p} = SN_{b,h,m,p} : \bigvee_{m} +(\cup)$$

$$A^{B,H,M,P=M}_{b,h,m,p} = SN_{b,h,m,p} / SD_{b,h,p}$$

$$AV^{B,F,H,P=M}_{b,f,h,p} = A_{b,h,m,p} \times V_{b,f,h,m} : \bigvee_{m} +(\cup)$$

$$C^{B,G=H\times F,P=M}_{b,h*F+f,p} = AV_{b,f,h,p}$$

$$Z^{B,D,P=M}_{b,d,p} = C_{b,g,p} \times WZ_{d,g} : \bigvee_{g} +(\cup)$$

Ideas for Sparse Attention in Transformers

Base transformer [Vaswani et al. 2016]

$$Q_{b,e,h,m} = I_{b,d,m} \times WQ_{d,e,h} : \bigvee_{d} +(\cup)$$

$$K_{b,e,h,m} = I_{b,d,m} \times WK_{d,e,h} : \bigvee_{d} +(\cup)$$

$$V_{b,f,h,m} = I_{b,d,m} \times WV_{d,f,h} : \bigvee_{d} +(\cup)$$

$$QK^{B,H,M,P=M}_{b,h,m,p} = \frac{1}{\sqrt{E}} \times Q^{B,E,H,M}_{b,e,h,p} \times K_{b,e,h,m} : \bigvee_{e} +(\cup)$$

$$SN^{B,H,M,P=M}_{b,h,m,p} = e^{QK_{b,h,m,p}}$$

$$SD^{B,H,P=M}_{b,h,p} = SN_{b,h,m,p} : \bigvee_{m} +(\cup)$$

$$A_{b,h,m,p} = SN_{b,h,m,p} / SD_{b,h,p}$$

$$AV_{b,f,h,p} = A_{b,h,m,p} \times V_{b,f,h,m} : \bigvee_{m} +(\cup)$$

$$C^{B,G=H\times F,P=M}_{b,h\times F+f,p} = AV_{b,f,h,p}$$

$$Z_{b,d,p} = C_{b,g,p} \times WZ_{d,g} : \bigvee_{g} +(\cup)$$

DOTA [Qu et al. 2022]

$$Q_{b,e,h,m} = I_{b,d,m} \times WQ_{d,e,h} : \bigvee_{d} +(\cup)$$

$$K_{b,e,h,m} = I_{b,d,m} \times WK_{d,e,h} : \bigvee_{d} +(\cup)$$

$$V_{b,f,h,m} = I_{b,d,m} \times WV_{d,f,h} : \bigvee_{d} +(\cup)$$

$$\tilde{Q}_{b,k,h,m} = I_{b,d,m} \times P^{D,K}_{d,j} \times \tilde{WQ}_{j,k,h} : \bigvee_{d,j} +(\cup)$$

$$\tilde{K}_{b,k,h,m} = I_{b,d,m} \times P^{D,K}_{d,j} \times \tilde{WK}_{j,k,h} : \bigvee_{d,j} +(\cup)$$

$$\tilde{QK}^{B,H,M,P=M}_{b,h,m,p} = \tilde{Q}^{B,K,H,M}_{b,k,h,p} \times \tilde{K}_{b,k,h,m} : \bigvee_{k} +(\cup)$$

$$M_{b,h,m,p} = prune(\tilde{QK}_{b,h,m,p})$$

$$QK^{B,H,M,P=M}_{b,h,m,p} = \frac{1}{\sqrt{E}} \times M_{b,h,m,p} \times Q^{B,E,H,M}_{b,e,h,p} \times K_{b,e,h,m} : \bigvee_{e} +(\cup)$$

$$SN^{B,H,M,P=M}_{b,h,m,p} = e^{QK_{b,h,m,p}}$$

$$SD^{B,H,P=M}_{b,h,p} = SN_{b,h,m,p} : \bigvee_{m} +(\cup)$$

$$A^{B,H,M,P=M}_{b,h,m,p} = SN_{b,h,m,p} / SD_{b,h,p}$$

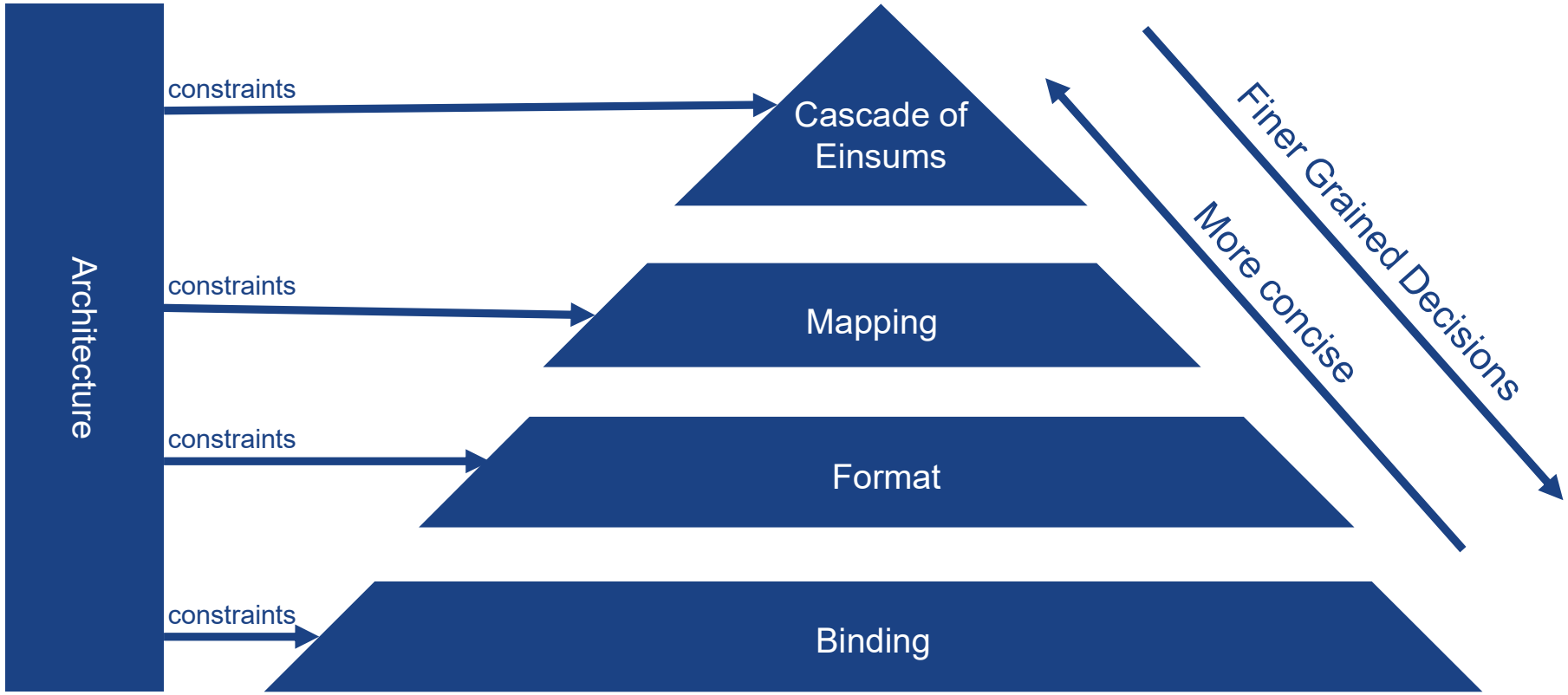
$$AV^{B,F,H,P=M}_{b,f,h,p} = A_{b,h,m,p} \times V_{b,f,h,m} : \bigvee_{m} +(\cup)$$

$$C^{B,G=H\times F,P=M}_{b,h*F+f,p} = AV_{b,f,h,p}$$

$$Z^{B,D,P=M}_{b,d,p} = C_{b,g,p} \times WZ_{d,g} : \bigvee_{g} +(\cup)$$

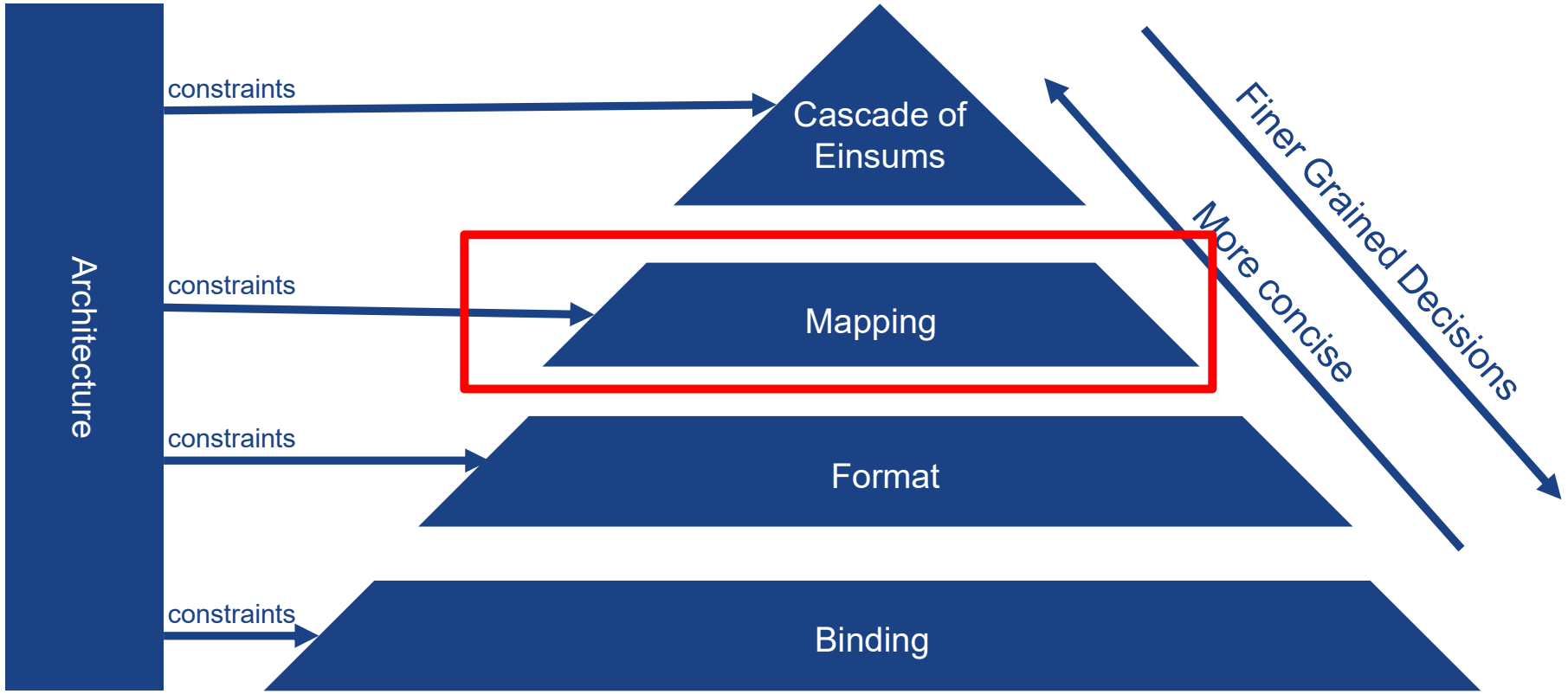


TeAAL Pyramid of Concerns



[TeAAL, Nayak et.al. MICRO 2023]

TeAAL Pyramid of Concerns



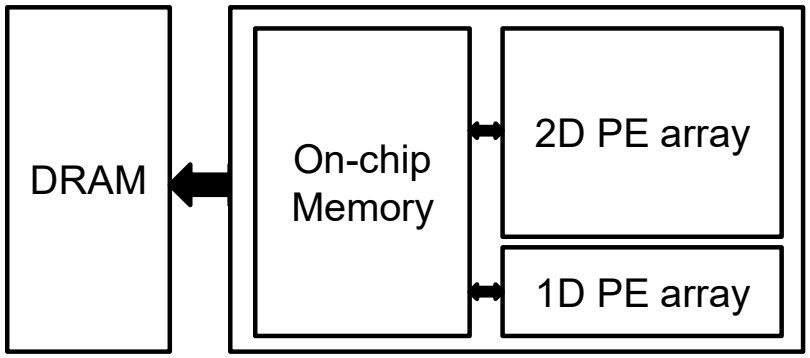
[TeAAL, Nayak et.al. MICRO 2023]

Aspects of Mapping

- How is processing ordered (including splitting)
  - In time, e.g., tiled
  - In space, e.g., parallelized
- How is data moved and stored hierarchically
  - Where are tiles of data stored
  - When is data reused without storing to memory, fusion

Accelerator Architecture

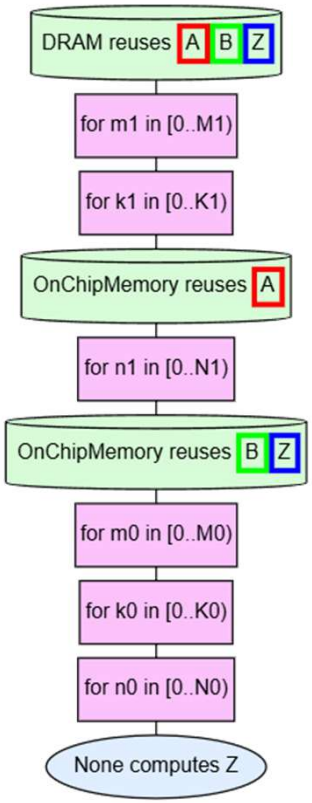
- An accelerator with on-chip memory, 2D PE array, and 1D PE array.
- **Connected to off-chip DRAM.**



Reading LoopTrees

- **Mappings can be represented as a LoopTree that consists of**

$$Z_{m,n} = A_{m,k} \times B_{k,n}$$

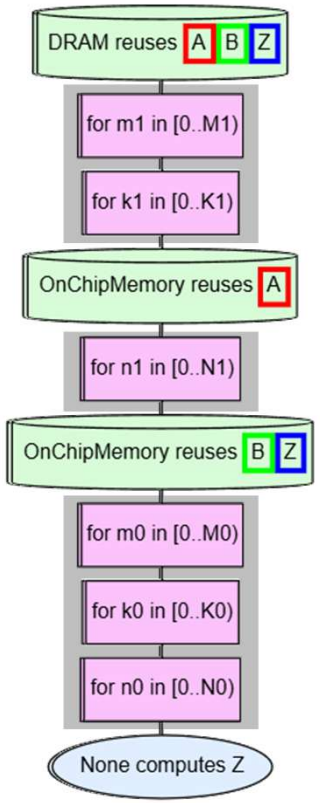


[LoopTree, Gilbert et.al. TCAI 2024]

Reading LoopTrees

- **Mappings can be represented as a LoopTree that consists of**
  - **Loop nodes, specifying traversal over ranks in the Einsum.**

$$Z_{m,n} = A_{m,k} \times B_{k,n}$$

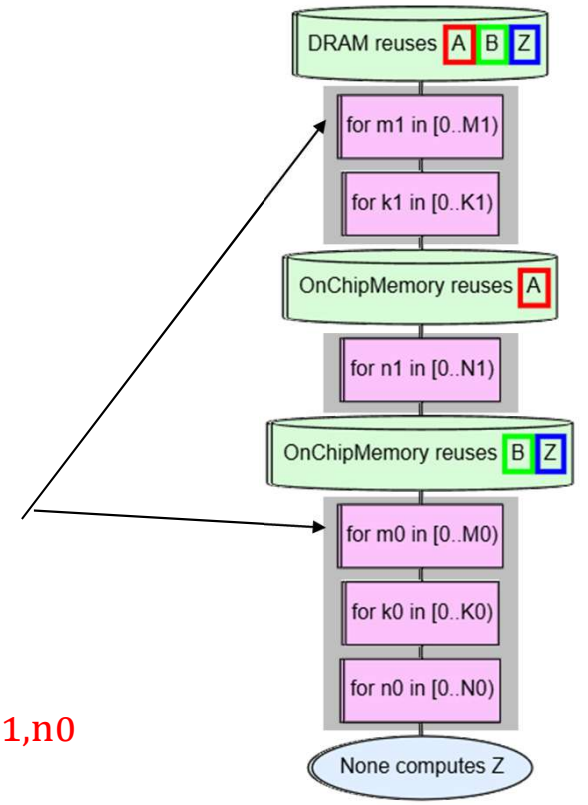


[LoopTree, Gilbert et.al. TCAI 2024]

Reading LoopTrees

- **Mappings can be represented as a LoopTree that consists of**
  - **Loop nodes, specifying traversal over ranks in the Einsum.**

$$Z_{m,n} = A_{m,k} \times B_{k,n}$$



Rank M has been *partitioned* into $M1 \times M0$.

These loops iterate over the created *tiles*.

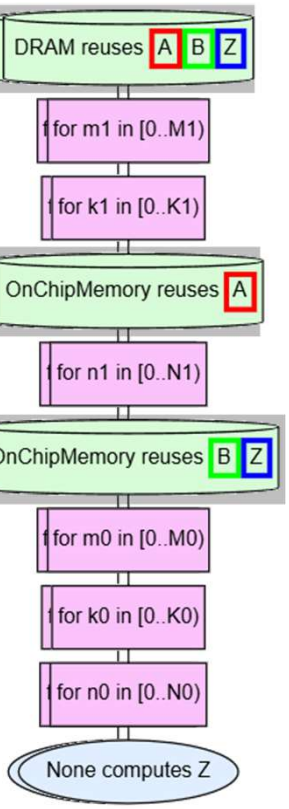
$$Z_{m1,\mathrm{m0},n1,\mathrm{n0}} = A_{m1,\mathrm{m0}k1,\mathrm{k0}} \times B_{k1,\mathrm{k0},n1,\mathrm{n0}}$$

[LoopTree, Gilbert et.al. TCAI 2024]

Reading LoopTrees

- **Mappings can be represented as a LoopTree that consists of**
  - **Loop nodes, specifying traversal over ranks in the Einsum.**
  - **Storage nodes, specifying keeping and reusing tensor tiles in memory.**

$$Z_{m,n} = A_{m,k} \times B_{k,n}$$

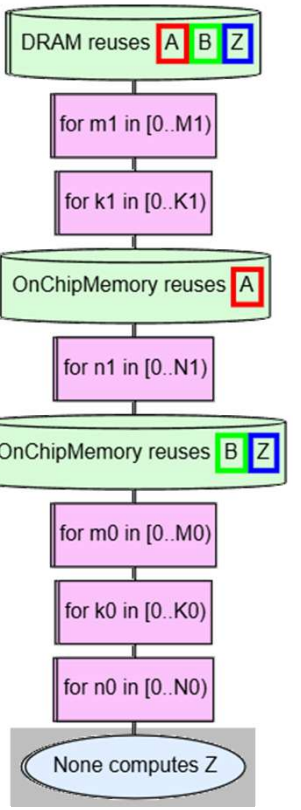


[LoopTree, Gilbert et.al. TCAI 2024]

Reading LoopTrees

- **Mappings can be represented as a LoopTree that consists of**
  - **Loop nodes, specifying traversal over ranks in the Einsum.**
  - **Storage nodes, specifying keeping and reusing tensor tiles in memory.**
  - **Compute nodes at the leaf, specifying processing operations at compute units.**

$$Z_{m,n} = A_{m,k} \times B_{k,n}$$

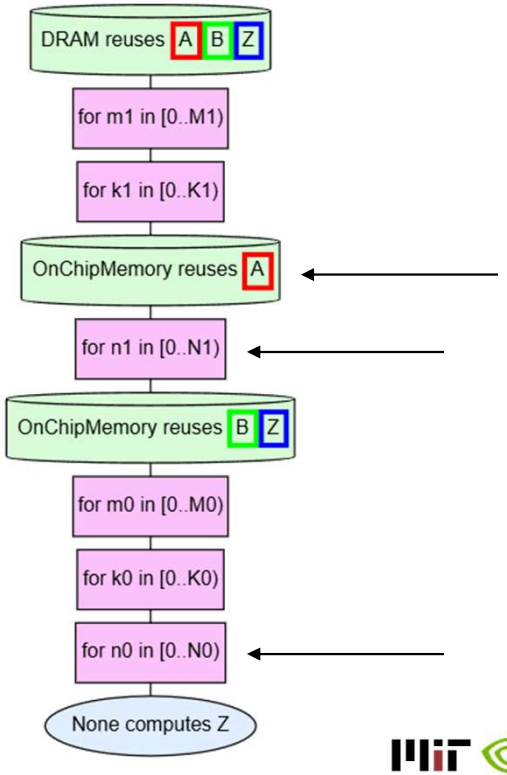


[LoopTree, Gilbert et.al. TCAI 2024]

LoopTrees Ease Interpretation of Mappings

- **LoopTrees have simple rules for analyzing reuse.**

$$Z_{m,n} = A_{m,k} \times B_{k,n}$$



Loops over rank variables *not* in the tensor *under* the storage node shows *reuse*.

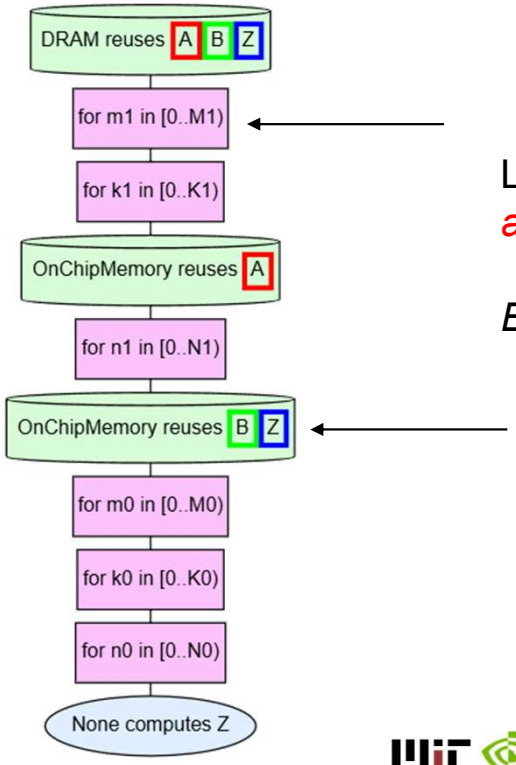
E.g., iterations over $n1$ and $n0$ reuse elements of A in OnChipMemory, reducing DRAM accesses.

[LoopTree, Gilbert et.al. TCAI 2024]

LoopTrees Ease Interpretation of Mappings

- **LoopTrees have simple rules for analyzing reuse.**

$$Z_{m,n} = A_{m,k} \times B_{k,n}$$



Loops over rank variables *not* in the tensor *above* the storage node shows *refetches*.

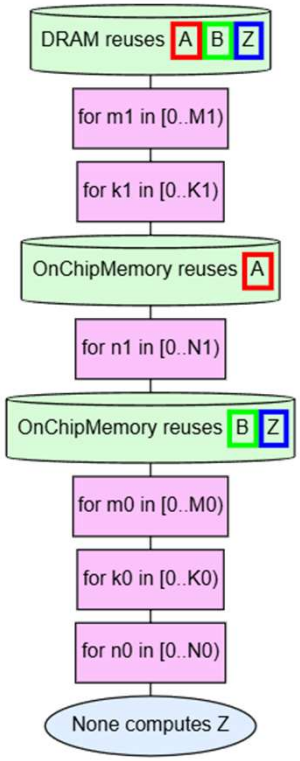
E.g., tensor B is refetched $M1$ times

[LoopTree, Gilbert et.al. TCAI 2024]

LoopTrees Ease Interpretation of Mappings

- **LoopTrees have simple rules for analyzing reuse.**

$$Z_{m,n} = A_{m,k} \times B_{k,n}$$



Loops over rank variables *not* in the tensor *above* the storage node shows *refetches*.

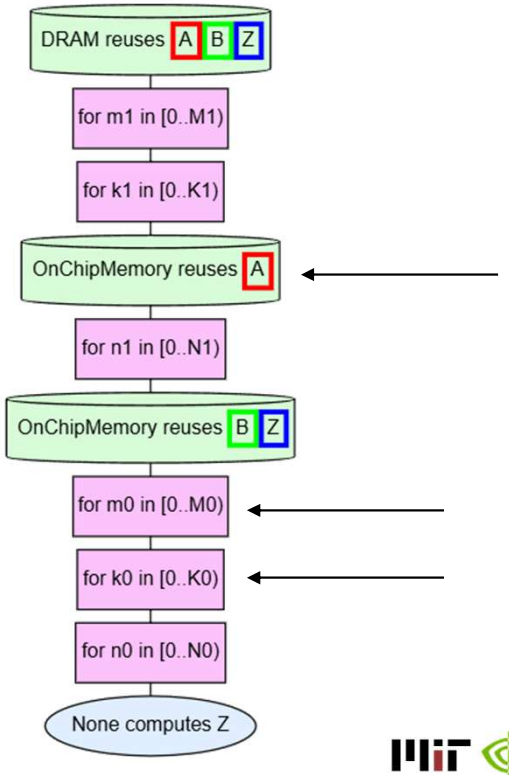
E.g., tensor A is never refetched.

[LoopTree, Gilbert et.al. TCAI 2024]

LoopTrees Ease Interpretation of Mappings

- **LoopTrees have simple rules for analyzing reuse.**

$$Z_{m,n} = A_{m,k} \times B_{k,n}$$



Loops over rank variables *in* the tensor *under* the storage node shows the *tile shape*.

E.g., tensor A has tile shape $(M0, K0)$.

[LoopTree, Gilbert et.al. TCAI 2024]

Basic Mapping as Baseline

- Here is a simplified attention Einsum cascade:

$$QK_{h,m,p} = Q_{h,m,e} \times K_{h,p,e}$$

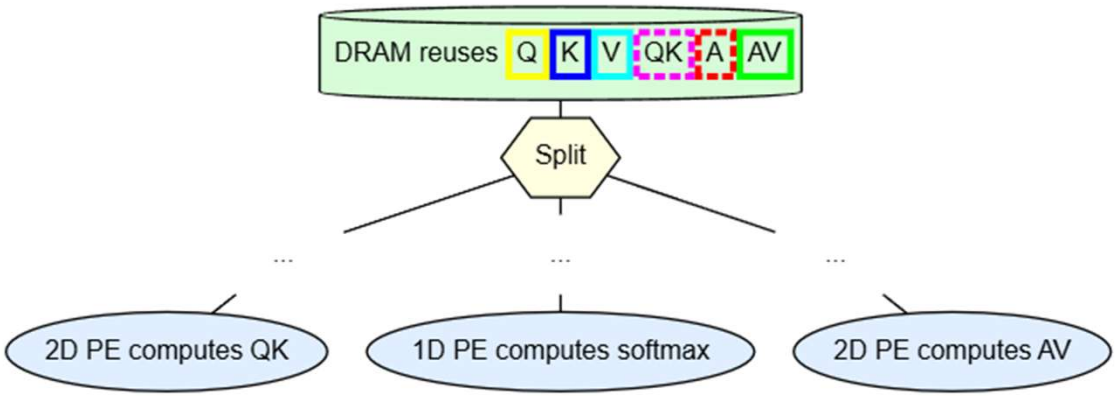
$$A_{h,m,p} = \text{softmax}(QK)$$

$$AV_{h,m,f} = A_{h,m,p} \times V_{p,f}$$

- **The baseline mapping processes each Einsum individually, using off-chip memory to keep any intermediate tensors.**

Basic Mapping as a LoopTree

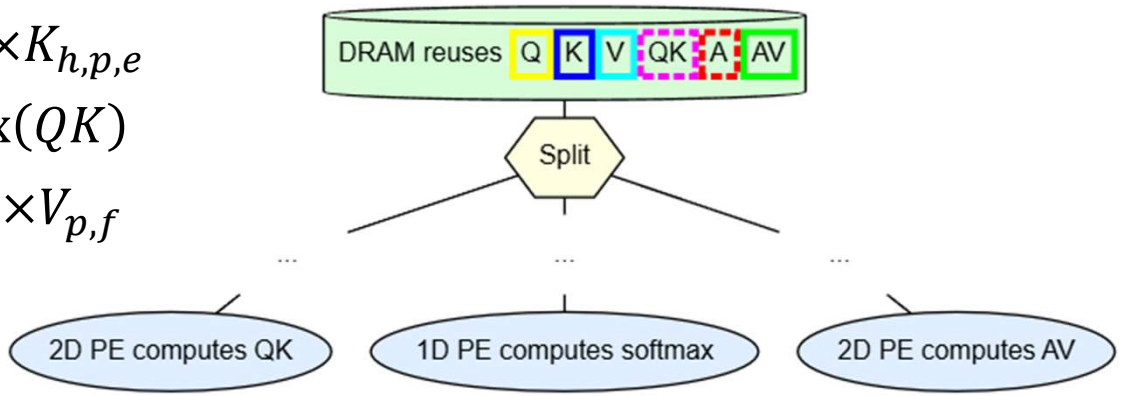
- **The basic mapping processes Einsums while keeping intermediate results (QK and A) in off-chip memory.**



Basic Mapping as a LoopTree

- **The basic mapping processes Einsums while keeping intermediate results (QK and A) in off-chip memory.**

$$QK_{h,m,p} = Q_{h,m,e} \times K_{h,p,e}$$
$$A_{h,m,p} = \text{softmax}(QK)$$
$$AV_{h,m,f} = A_{h,m,p} \times V_{p,f}$$

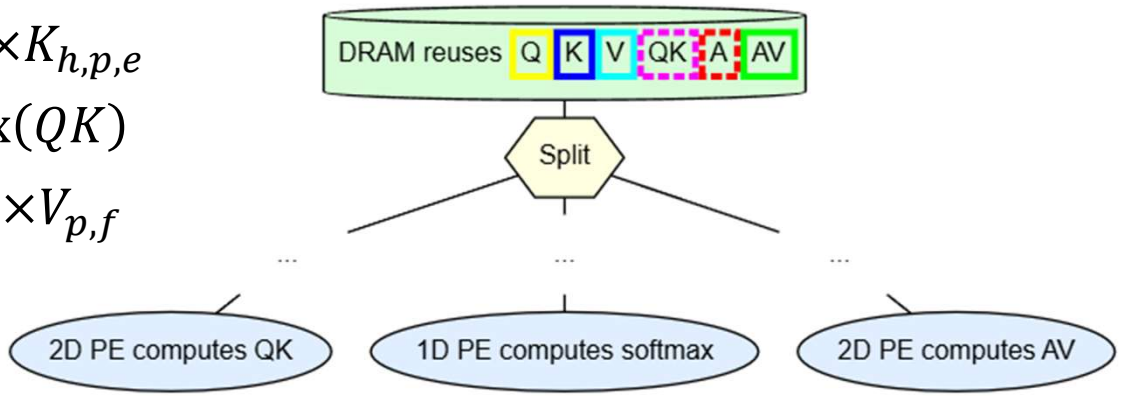


Basic Mapping as a LoopTree

- **The basic mapping processes Einsums while keeping intermediate results (QK and A) in off-chip memory.**

$$QK_{h,m,p} = Q_{h,m,e} \times K_{h,p,e}$$
$$A_{h,m,p} = \text{softmax}(QK)$$
$$AV_{h,m,f} = A_{h,m,p} \times V_{p,f}$$

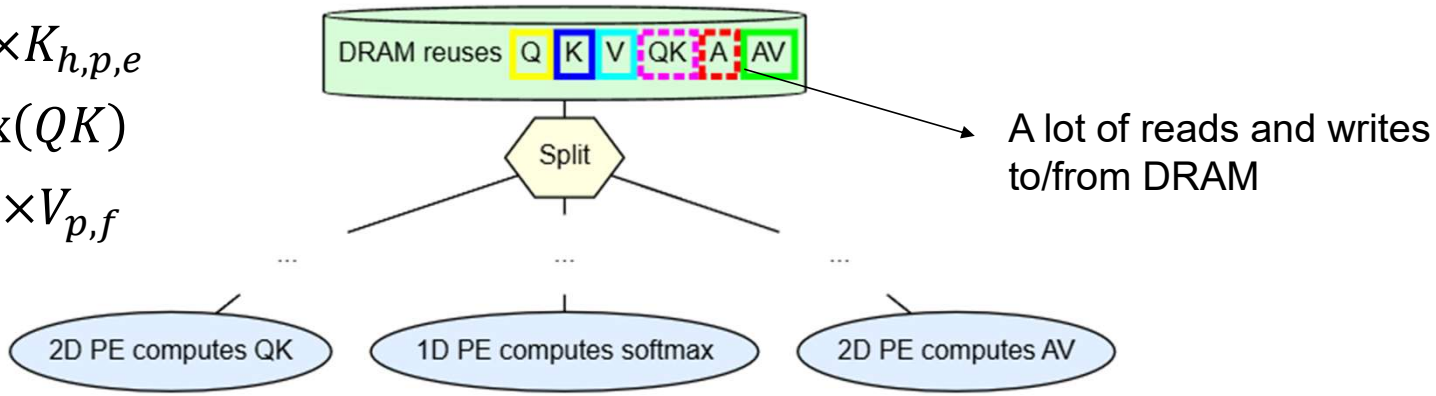
Scales quadratically with sequence length M and P



Unfused Mapping as a LoopTree

- **The unfused mapping processes Einsums while keeping intermediate results (QK and A) in off-chip memory.**

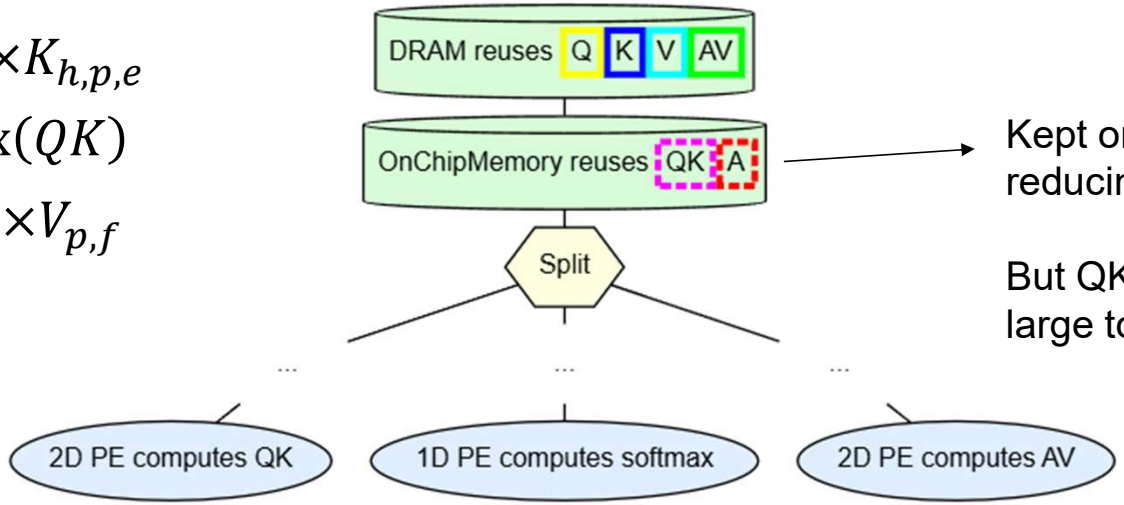
$$QK_{h,m,p} = Q_{h,m,e} \times K_{h,p,e}$$
$$A_{h,m,p} = \text{softmax}(QK)$$
$$AV_{h,m,f} = A_{h,m,p} \times V_{p,f}$$



Fusing the Einsums

- **A *fused* mapping keeps and reuses intermediate tensors QK and A in on-chip memory:**

$$QK_{h,m,p} = Q_{h,m,e} \times K_{h,p,e}$$
$$A_{h,m,p} = \text{softmax}(QK)$$
$$AV_{h,m,f} = A_{h,m,p} \times V_{p,f}$$

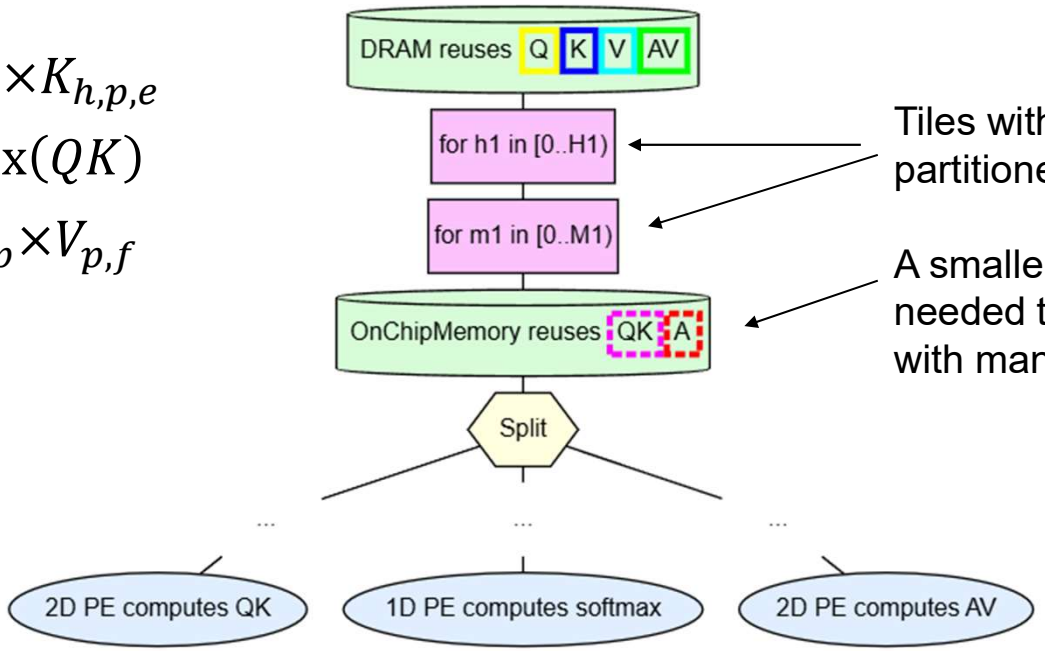


Kept only in OnChipMemory, reducing DRAM accesses.

But QK and A may still be too large to fit on chip.

The Mapping of FLAT

$$QK_{h,m,p} = Q_{h,m,e} \times K_{h,p,e}$$
$$A_{h,m,p} = \text{softmax}(QK)$$
$$AV_{h,m,f} = A_{h,m,p} \times V_{p,f}$$



Tiles with ranks H and M partitioned!

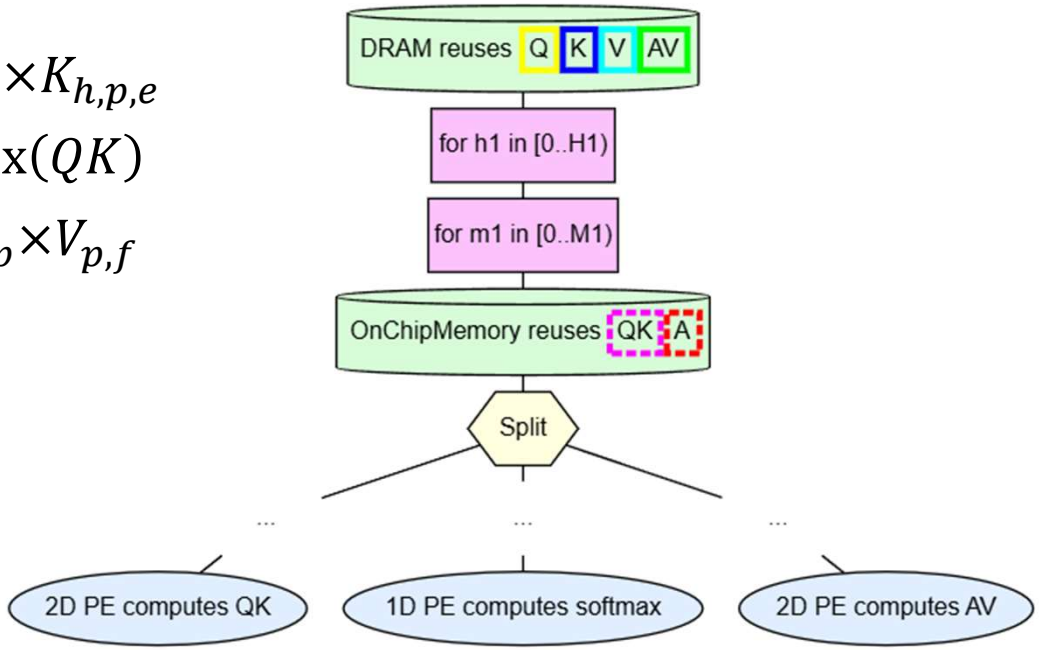
A smaller on-chip memory is needed to keep QK and A, but with many refetches.

[FLAT, Kao et. al., ASPLOS 2023]

The Mapping of FLAT

$$QK_{h,m,p} = Q_{h,m,e} \times K_{h,p,e}$$
$$A_{h,m,p} = \text{softmax}(QK)$$
$$AV_{h,m,f} = A_{h,m,p} \times V_{p,f}$$

Scales quadratically with sequence length M and P



[FLAT, Kao et. al., ASPLOS 2023]

Can We Reduce Memory Requirement More?

Einsum from before

$$QK_{h,m,p} = Q_{h,m,e} \times K_{h,p,e}$$
$$A_{h,m,p} = \text{softmax}(QK)$$
$$AV_{h,m,f} = A_{h,m,p} \times V_{p,f}$$

Can We Reduce Memory Requirement More?

Einsum from before

$$QK_{h,m,p} = Q_{h,m,e} \times K_{h,p,e}$$
$$A_{h,m,p} = \text{softmax}(QK)$$
$$AV_{h,m,f} = A_{h,m,p} \times V_{p,f}$$

With softmax expanded

$$QK_{h,m,p} = Q_{h,m,e} \times K_{h,p,e}$$
$$GM_m = QK_{m,p} :: \bigvee_p \max(\cup)$$
$$SN_{h,m,p} = \exp\left(QK_{h,m,p} - GM_m\right)$$
$$A_{h,m,p} = SN_{h,m,p} / SD_m$$
$$AV_{h,m,f} = A_{h,m,p} \times V_{p,f}$$

Can We Reduce Memory Requirement More?

Einsum from before

$$QK_{h,m,p} = Q_{h,m,e} \times K_{h,p,e}$$
$$A_{h,m,p} = \text{softmax}(QK)$$
$$AV_{h,m,f} = A_{h,m,p} \times V_{p,f}$$

With softmax expanded

$$QK_{h,m,p} = Q_{h,m,e} \times K_{h,p,e}$$
$$GM_m = QK_{m,p} :: \bigvee_p \max(\cup)$$
$$SN_{h,m,p} = \exp\left(QK_{h,m,p} - GM_m\right)$$
$$A_{h,m,p} = SN_{h,m,p} / SD_m$$
$$AV_{h,m,f} = A_{h,m,p} \times V_{p,f}$$

Reduction over P prevents partitioning rank P

Can We Reduce Memory Requirement More?

Einsum from before

$$QK_{h,m,p} = Q_{h,m,e} \times K_{h,p,e}$$
$$A_{h,m,p} = \text{softmax}(QK)$$
$$AV_{h,m,f} = A_{h,m,p} \times V_{p,f}$$

With softmax expanded

$$QK_{h,m,p} = Q_{h,m,e} \times K_{h,p,e}$$
$$GM_m = QK_{m,p} :: \bigvee_p \max(\cup)$$
$$SN_{h,m,p} = \exp\left(QK_{h,m,p} - GM_m\right)$$
$$A_{h,m,p} = SN_{h,m,p}/SD_m$$
$$AV_{h,m,f} = A_{h,m,p} \times V_{p,f}$$

Reduction over P prevents partitioning rank P

With Eingebraic manipulations from FuseMax[]

Initialization:

$$BK_{e,m1,m0} = K_{e,m1 \times M0+m0}$$
$$BV_{f,m1,m0} = V_{f,m1 \times M0+m0}$$
$$RM_{m1:m1=0,p} = -\infty$$
$$RD_{m1:m1=0,p} = 0$$
$$RNV_{m1:m1=0,p} = 0$$

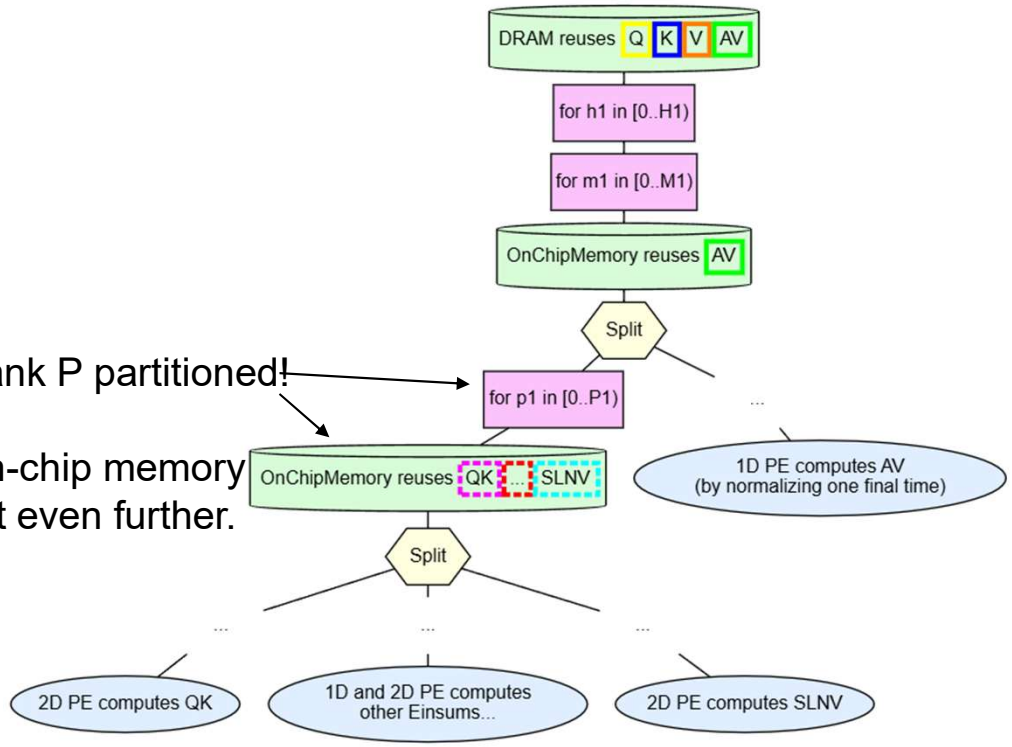
Extended Einsums:

$$BQK_{m1,m0,p} = Q_{e,p} \times BK_{e,m1,m0}$$
$$LM_{m1,p} = BQK_{m1,m0,p} :: \bigvee_{m0} \max(\cup)$$
$$RM_{m1+1,p} = max(RM_{m1,p}, LM_{m1,p})$$
$$SLN_{m1,m0,p} = e^{BQK_{m1,m0,p} - RM_{m1+1,p}}$$
$$SLD_{m1,p} = SLN_{m1,m0,p}$$
$$SLNV_{f,m1,p} = SLN_{m1,m0,p} \times BV_{f,m1,m0}$$
$$PRM_{m1,p} = e^{RM_{m1,p} - RM_{m1+1,p}}$$
$$SPD_{m1,p} = RD_{m1,p} \times PRM_{m1,p}$$
$$RD_{m1+1,p} = SLD_{m1,p} + SPD_{m1,p}$$
$$SPNV_{f,m1,p} = RNV_{f,m1,p} \times PRM_{m1,p}$$
$$RNV_{f,m1+1,p} = SLNV_{f,m1,p} + SPNV_{f,m1,p}$$
$$AV_{f,p} = RNV_{f,M1,p}/RD_{M1,p}$$
$$\diamond : m1 \geq M1$$

The Mapping of FuseMax

Tiles with rank P partitioned!

Reduces on-chip memory requirement even further.



[FuseMax, Nayak et. al., MICRO 2024]

Can We Fuse More Einsums?

- **We looked at the following original Einsums, …**

$$QK_{h,m,p} = Q_{h,m,e} \times K_{h,p,e}$$
$$A_{h,m,p} = \text{softmax}(QK)$$
$$AV_{h,m,f} = A_{h,m,p} \times V_{p,f}$$

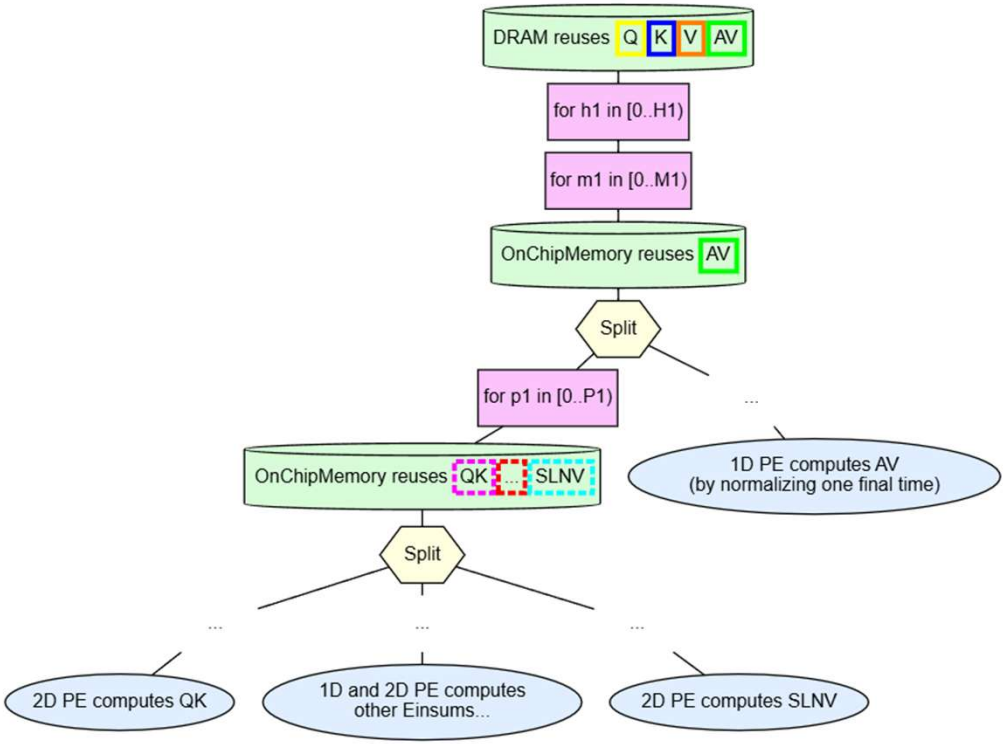
Can We Fuse More Einsums?

- **We looked at the following original Einsums, but transformers have even more:**

$$Q_{h,m,e} = I_{m,d} \times WQ_{d,h,e}$$
$$K_{h,m,f} = I_{m,d} \times WK_{d,h,f}$$
$$V_{h,m,f} = I_{m,d} \times WV_{d,h,f}$$
$$QK_{h,m,p} = Q_{h,m,e} \times K_{h,p,e}$$
$$A_{h,m,p} = \mathrm{softmax}(QK)$$
$$AV_{h,m,f} = A_{h,m,p} \times V_{p,f}$$
$$Z_{m,g} = AV_{h,m,f} \times WZ_{h,f,g}$$
$$FFA_{m,c} = Z_{m,g} \times WFFA_{g,c}$$
$$FFB_{m,j} = FFA_{m,c} \times WFFB_{c,j}$$

Can We Fuse More Einsums?

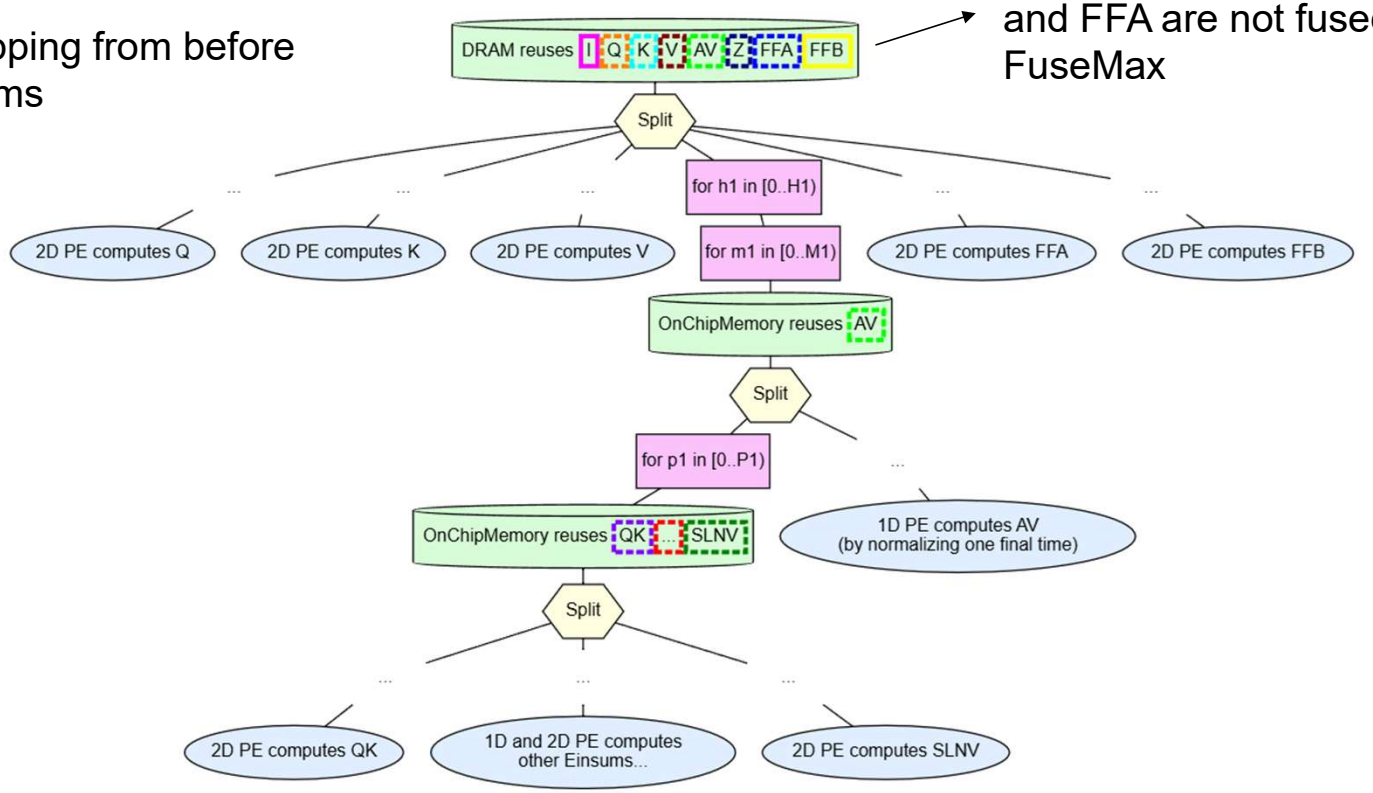
FuseMax mapping from before…



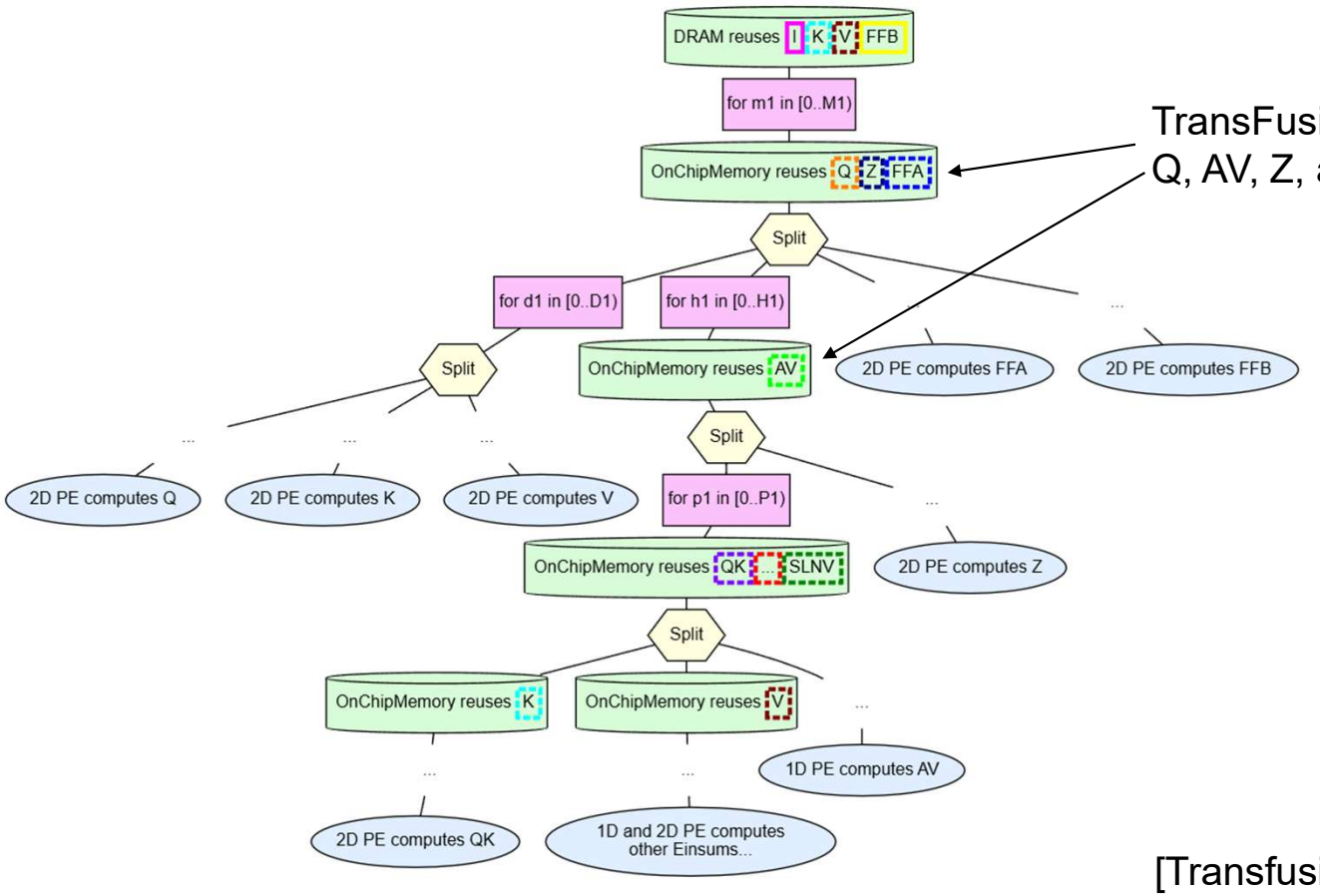
Can We Fuse More Einsums?

FuseMax mapping from before
+ other Einsums

Intermediate tensors Q, K, V, Z, and FFA are not fused by FuseMax



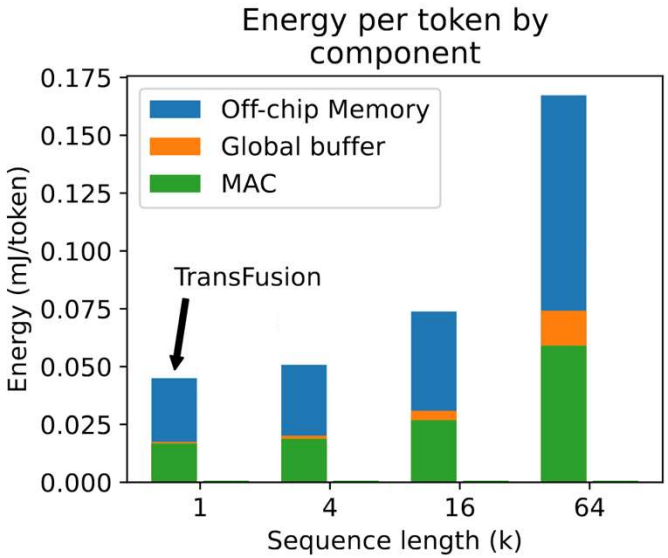
Fuse All Einsums in a Transformer Block



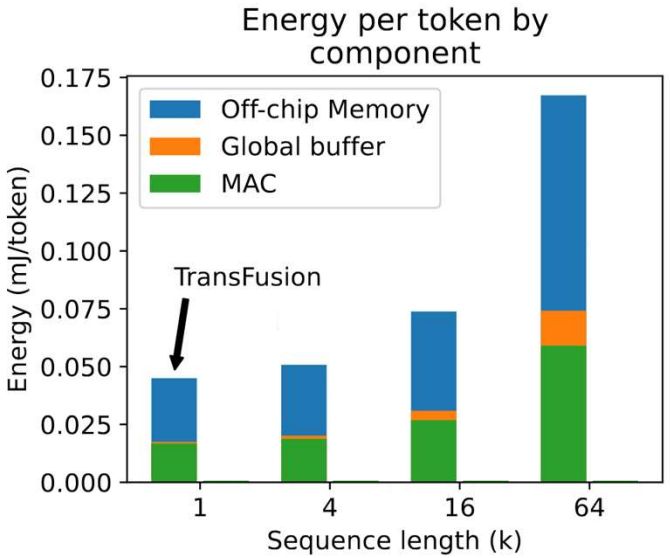
TransFusion:
Q, AV, Z, and FFA are also fused.

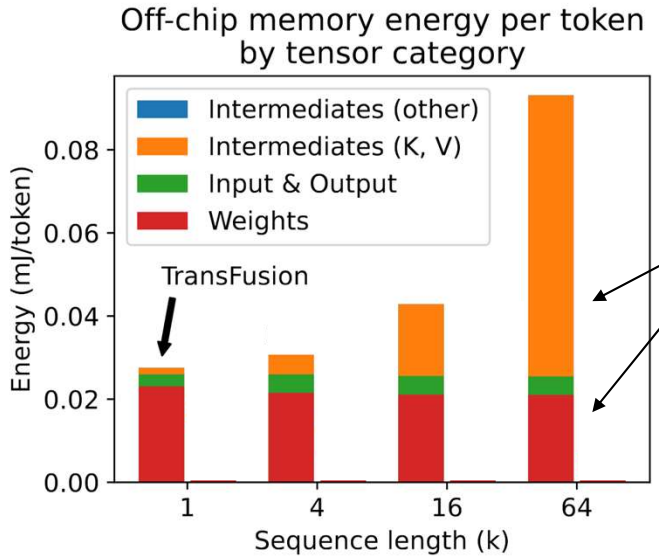
[Transfusion, Zhang et. Al., MICRO 2025]

What Happens With Lots of Fusion



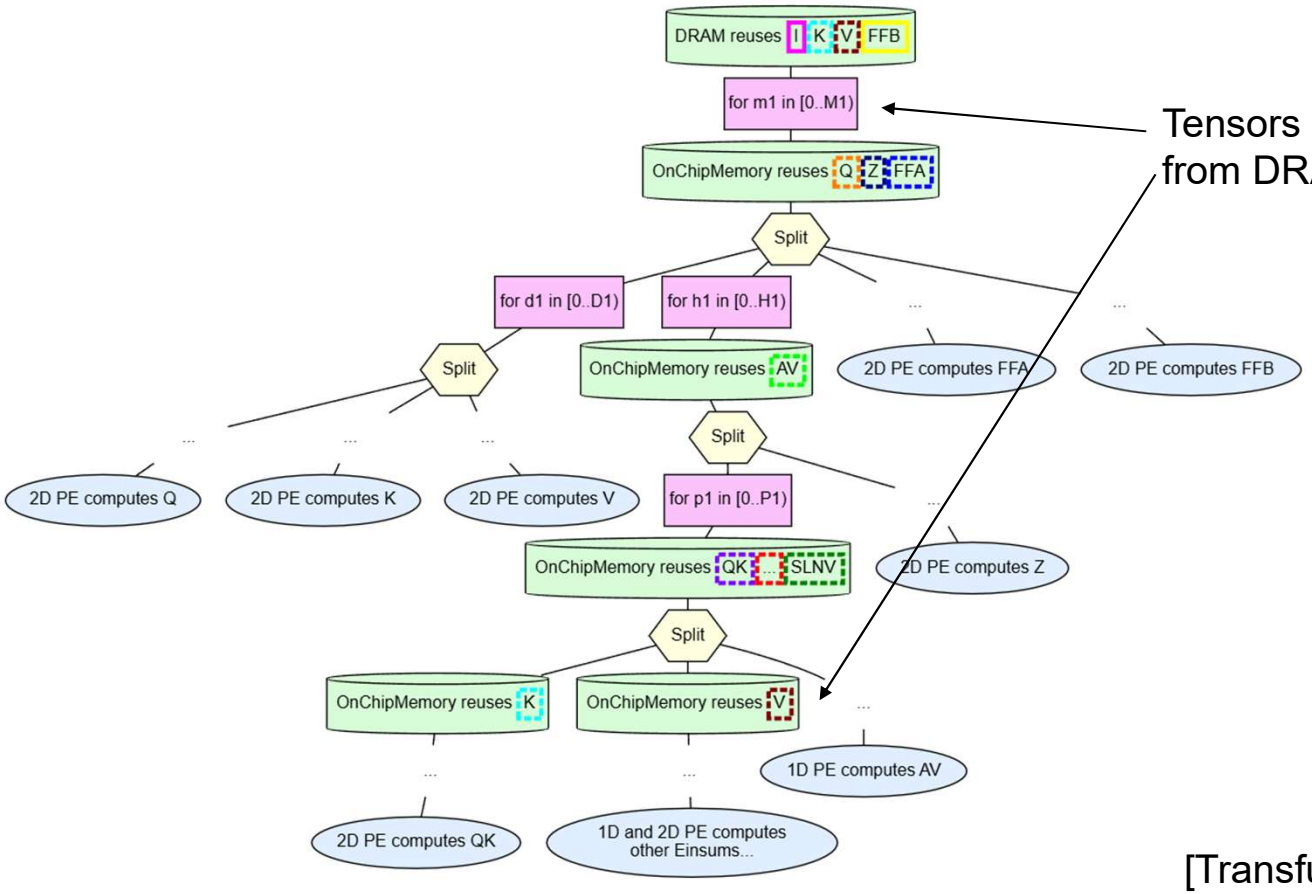
More Fusion is Not Always Better!





Data movement of K, V, and weights dominates energy usage!

Fuse All Einsums in a Transformer Block

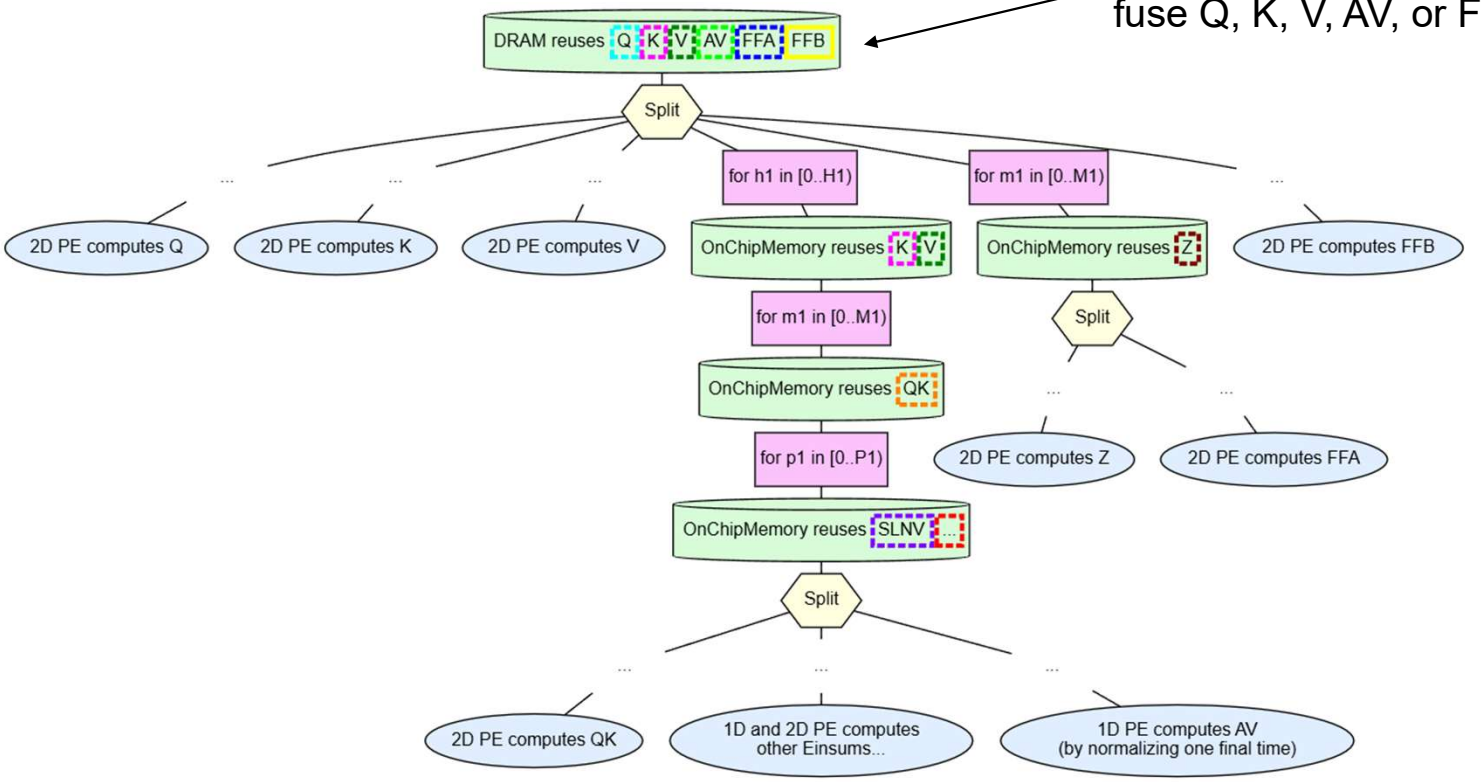


Tensors K and V are read M1 times from DRAM

[Transfusion, Zhang et. Al., MICRO 2025]

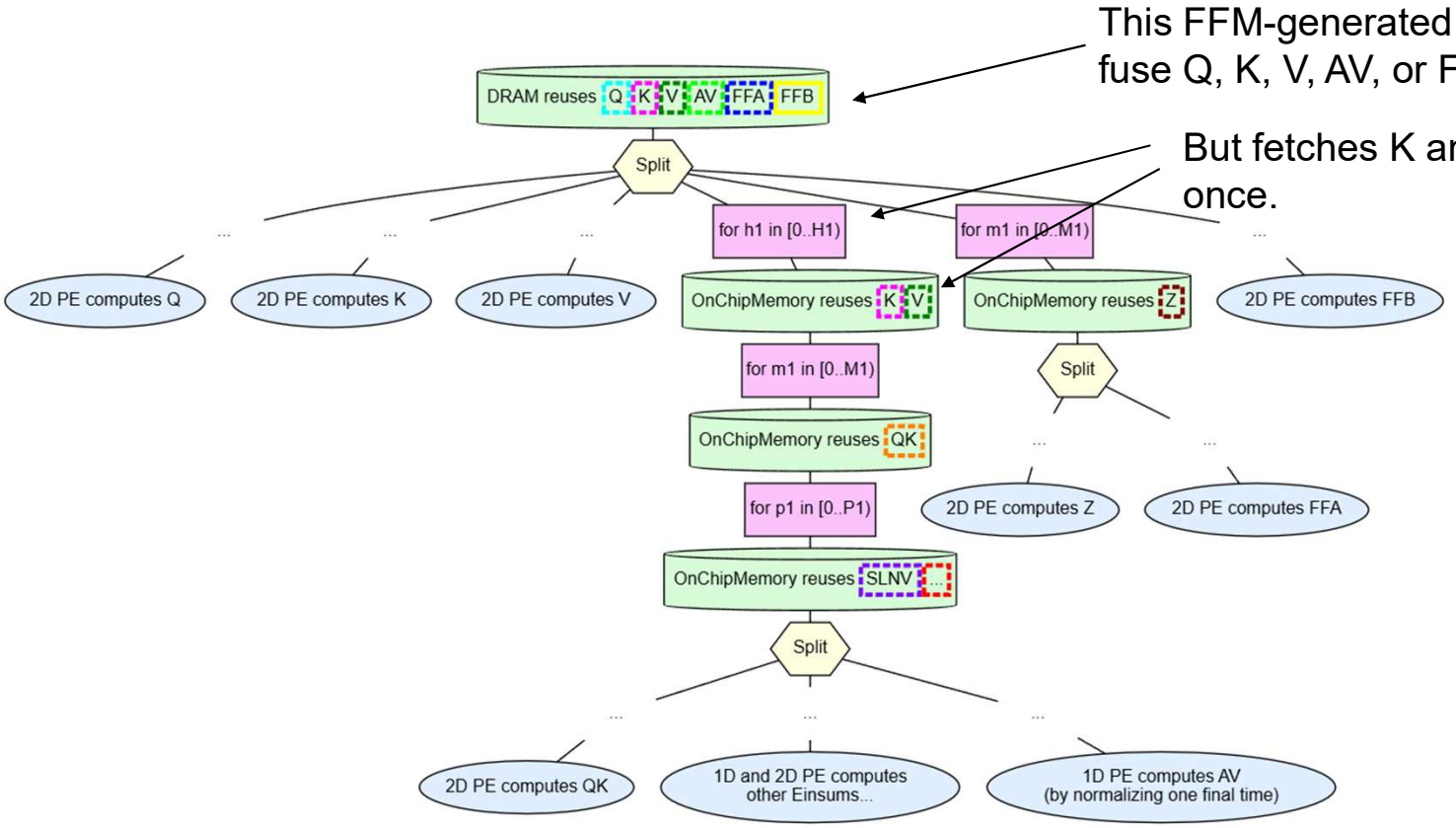
Mapping Generated by FFM for GPT-3 6.7B

This FFM-generated mapping does not fuse Q, K, V, AV, or FFA.



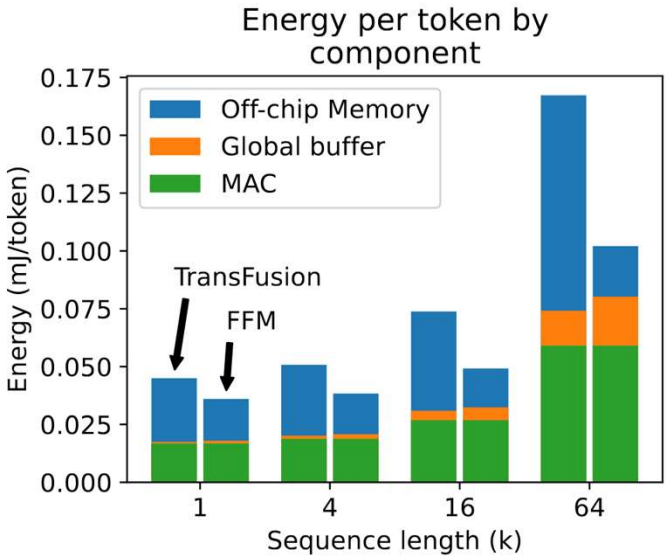
[Fast and Fusiest , Andrulis et. al., ArXiv 2026]

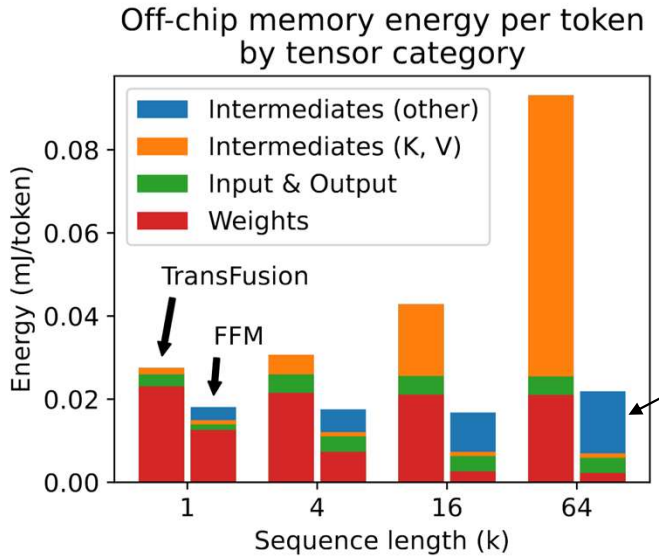
Mapping Generated by FFM for GPT-3 6.7B



[Fast and Fusiest , Andrulis et. al., ArXiv 2026]

Selective Fusion is Better

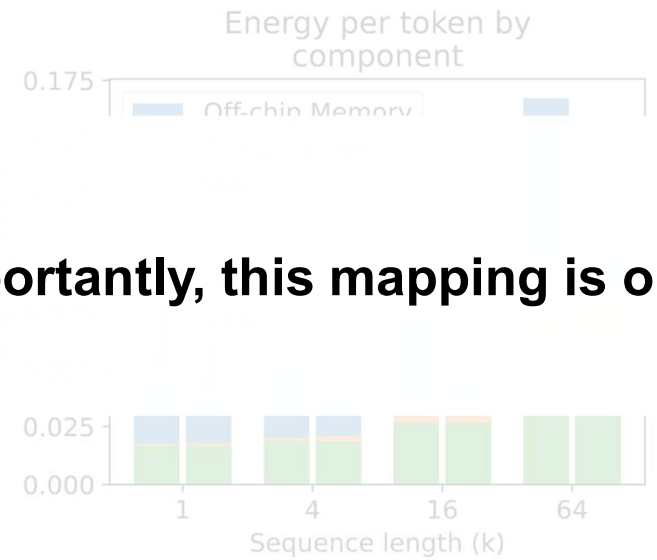




Higher data movement for unfused intermediates, but much lower for K, V, and weights.

[Fast and Fusiest , Andrulis et. al., ArXiv 2026]

More Fusion is Not Always Better!

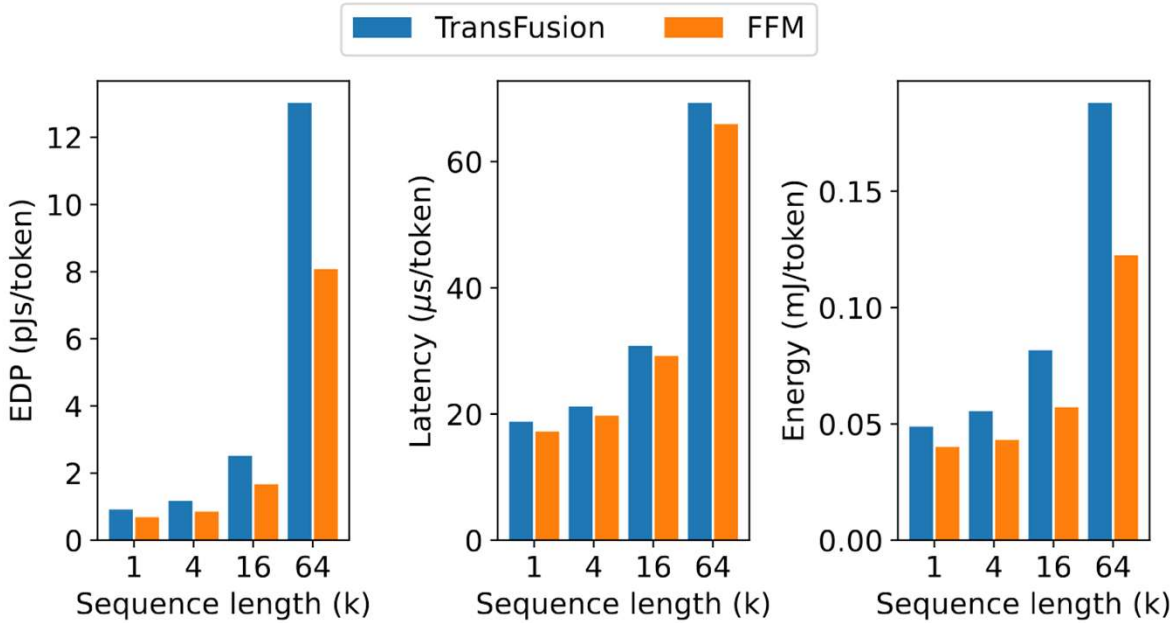




Importantly, this mapping is only better for *this* workload-architecture pairing!

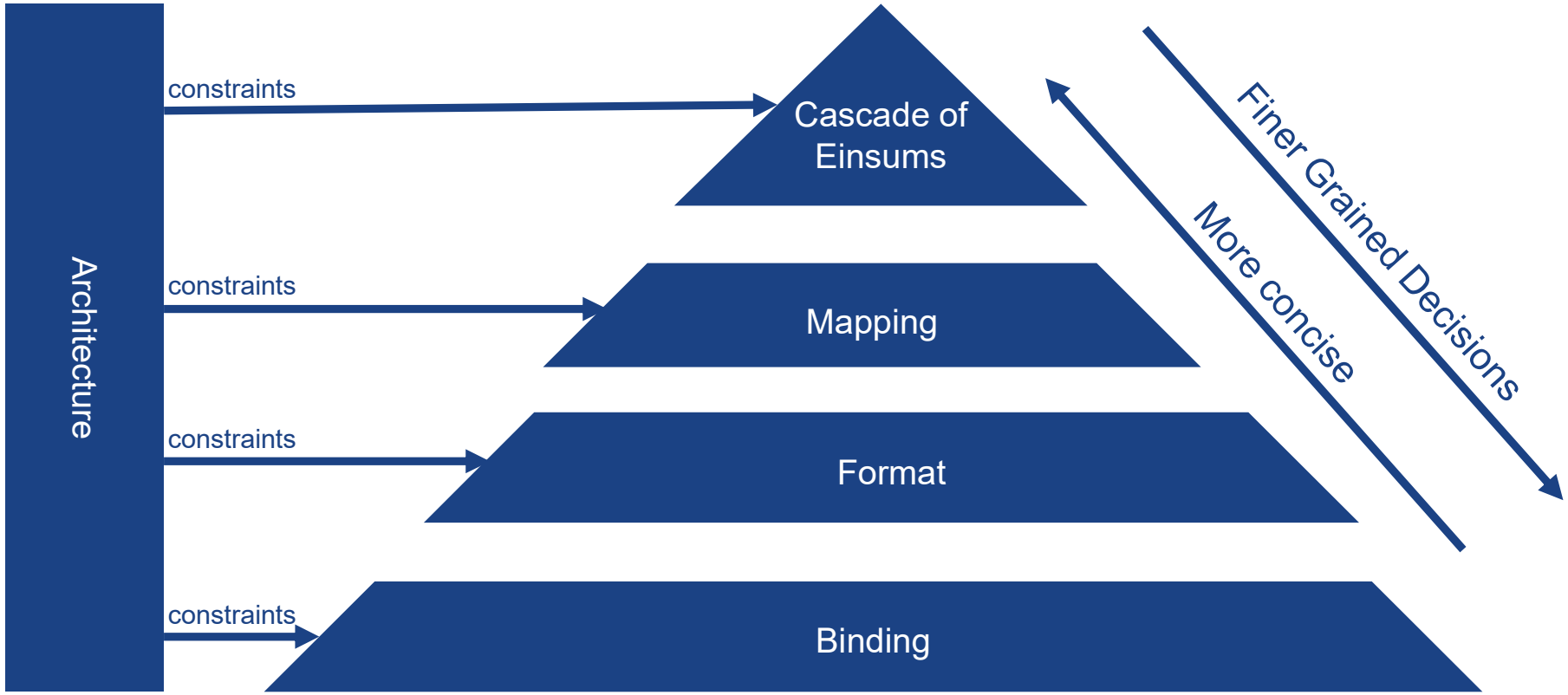
Mappers Help Navigate Mapping Trade-offs

- **The end result is lower EDP, latency, and energy simultaneously:**



[Fast and Fusiest , Andrulis et. al., ArXiv 2026]

TeAAL Pyramid of Concerns



[TeAAL, Nayak et.al. MICRO 2023]

Acknowledgements

- Some of the concepts presented in this lecture are the fruits of a collaborative effort that has included inputs from:

  - Vivienne Sze – MIT
  - Yu-Hsin Chen – MIT (Encharge)
  - Nellie Wu – MIT (Meta)
  - Michael Gilbert - MIT
  - Tanner Andrulis – MIT
  - Fisher Xue – MIT
  - Saman Amarasinghe - MIT
  - Jaeyeon Won – MIT
  - Willow Ahrens – MIT (GaTech)
  - Teo Colins – MIT
  - Angshuman Parashar – NVIDIA
  - Michael Pellauer – NVIDIA
  - Po-An Tsai – NVIDIA
  - Jenny Huang - NVIDIA
  - Daniel Sanchez – MIT
  - Yifan Yang – MIT
  - Guowei Zhang – MIT
  - Courney Golden – MIT
  - Maggie Du – MIT
  - Toluwa Odemuyiwa – UC Davis
  - Fred Kjolstad – MIT (Stanford)
  - Mark Horowitz – Stanford
  - Olivia Hsu – Stanford
  - Max Strange - Stanford
  - Chris Fletcher – UC Berkeley
  - Kartik Hegde – UC Berkeley
  - Nandeeka Nayak – UC Berkeley
  - Severn Bochem – UC Berkeley
  - Yan Zhu – UC Berkeley
  - Xinrui Wu - UCLA

Some slides from MIT class 6.5930

Questions?