A DETAILED ALGORITHM OF *TurboAttention*

Algorithm 1 *TurboAttention Prefill*

Require: Matrices $\mathbf{Q}, \mathbf{K}, \mathbf{V} \in \mathbb{R}^{N\times d}$ in HBM, compressed KV cache $\mathbf{K}^{q2}$, $\mathbf{V}^{q2}$,integer scale of KV cache $s^{int}_{\mathbf{K}^{q2}}$, floating scale s_K, s_V, $s^{int}_{\mathbf{V}^{q2}}$,zero point of KV cache $z^{int}_{\mathbf{K}^{q2}}$, $z^{int}_{\mathbf{V}^{q2}}$, block sizes B_c, B_r, Softmax approximate algorithm *SAS*.

Divide $\mathbf{Q}$ into $T_r = \left\lceil \frac{N}{B_r} \right\rceil$ blocks $\mathbf{Q}_1, \ldots, \mathbf{Q}_{T_r}$ of size $B_r \times d$ each, and divide $\mathbf{K}, \mathbf{V}$ in to $T_c = \left\lceil \frac{N}{B_c} \right\rceil$ blocks $\mathbf{K}_1, \ldots, \mathbf{K}_{T_c}$ and $\mathbf{V}_1, \ldots, \mathbf{V}_{T_c}$, of size $B_c \times d$ each.

Divide the output $\mathbf{O} \in \mathbb{R}^{N\times d}$ into T_r blocks $\mathbf{O}_i, \ldots, \mathbf{O}_{T_r}$ of size $B_r \times d$ each, and divide the logsumexp L into T_r blocks $L_i, \ldots, L_{T_r}$ of size B_r each.

for $1 \le i \le T_r$ **do**

Load $\mathbf{Q}_i$ from HBM to on-chip SRAM.

On chip, initialize $\mathbf{O}_i^{(0)} = (0)_{B_r\times d} \in \mathbb{R}^{B_r\times d}, \ell_i^{(0)} = (0)_{B_r} \in \mathbb{R}^{B_r}, m_i^{(0)} = (-\infty)_{B_r} \in \mathbb{R}^{B_r}$.

for $1 \le j \le T_c$ **do**

Load $\mathbf{K}_j, \mathbf{V}_j$ from HBM to on-chip SRAM.

On chip, compute $s_{\mathbf{Q}_i} = \frac{max(abs(\mathbf{Q}_i))}{119}, \mathbf{Q}_i^{q1} = \left\lceil \frac{\mathbf{Q}_i}{s_{\mathbf{Q}_i}} \right\rfloor, s_{\mathbf{K}_j} = \frac{max(abs(\mathbf{K}_j))}{119}, \mathbf{K}_j^{q1} = \left\lceil \frac{\mathbf{K}_j}{s_{\mathbf{K}_j}} \right\rfloor$, $s_{\mathbf{V}_j} = \frac{max(abs(\mathbf{V}_j))}{119}, \mathbf{V}_j^{q1} = \left\lceil \frac{\mathbf{V}_j}{s_{\mathbf{V}_j}} \right\rfloor$.

On chip, compute $\mathbf{S}_i^{(j)} = s_{\mathbf{Q}_i} \cdot s_{\mathbf{K}_j} \cdot \mathbf{Q}_i^{q1} \mathbf{K}_j^{q1} \in \mathbb{R}^{B_r\times B_c}$.

On chip, compute $m_i^{(j)} = \max(m_i^{(j-1)}, \mathrm{rowmax}(\mathbf{S}_i^{(j)})) \in \mathbb{R}^{B_r}$, $\tilde{\mathbf{P}}_i^{(j)} = SAS(\mathbf{S}_i^{(j)} - m_i^{(j)}) \in \mathbb{R}^{B_r\times B_c}$ (pointwise), $\ell_i^{(j)} = SAS(m_i^{j-1} - m_i^{(j)})\ell_i^{(j-1)} + \mathrm{rowsum}(\tilde{\mathbf{P}}_i^{(j)}) \in \mathbb{R}^{B_r}$.

On chip, compute $s_{\tilde{\mathbf{P}}_i^{(j)}} = \frac{max(abs(\tilde{\mathbf{P}}_i^{(j)})}{119}, Q(\tilde{\mathbf{P}}_i^{(j)}) = \left\lceil \frac{\tilde{\mathbf{P}}_i^{(j)}}{s_{\tilde{\mathbf{P}}_i^{(j)}}} \right\rfloor$,

On chip, compute $\mathbf{O}_i^{(j)} = \mathrm{diag}(SAS(m_i^{(j-1)} - m_i^{(j)}))^{-1}\mathbf{O}_i^{(j-1)} + s_{\tilde{\mathbf{P}}_i^{(j)}} \cdot s_{\mathbf{V}_j} \cdot Q(\tilde{\mathbf{P}}_i^{(j)})\mathbf{V}_j^{q1}$.

On chip, compute $s^{int}_{\mathbf{K}_j^{q2}} = \left\lceil \frac{max(\mathbf{K}_j^{q2}) - min(\mathbf{K}_j^{q2})}{2^{bit}-1} \right\rfloor$, $z^{int}_{\mathbf{K}_j^{q2}} = \left\lfloor \frac{min(\mathbf{K}_j^{q2})}{s^{int}_{\mathbf{K}_j^{q2}}} \right\rceil$, $\mathbf{K}_j^{q2} = \left\lceil \frac{\mathbf{K}_j^{q1}}{s^{int}_{\mathbf{K}_j^{q2}}} - z^{int}_{\mathbf{K}_j^{q2}} \right\rfloor$ (channelwise)

On chip, compute $s^{int}_{\mathbf{V}_j^{q2}} = \left\lceil \frac{max(\mathbf{V}_j^{q2}) - min(\mathbf{V}_j^{q2})}{2^{bit}-1} \right\rfloor$, $z^{int}_{\mathbf{V}_j^{q2}} = \left\lfloor \frac{min(\mathbf{V}_j^{q2})}{s^{int}_{\mathbf{V}_j^{q2}}} \right\rceil$, $\mathbf{V}_j^{q2} = \left\lceil \frac{\mathbf{V}_j^{q1}}{s^{int}_{\mathbf{V}_j^{q2}}} - z^{int}_{\mathbf{V}_j^{q2}} \right\rfloor$ (channelwise)

Write $\mathbf{K}_j^{q2}$ to HBM as the j-th block of $\mathbf{K}^{q2}$.

Write $\mathbf{V}_j^{q2}$ to HBM as the j-th block of $\mathbf{V}^{q2}$.

Write $s^{int}_{\mathbf{K}_j^{q2}}$,$s^{int}_{\mathbf{V}_j^{q2}}$ to HBM as the j-th block of $s^{int}_{\mathbf{K}^{q2}}$, $s^{int}_{\mathbf{V}^{q2}}$.

Write $s_{\mathbf{K}_j}$,$s_{\mathbf{V}_j}$ to HBM as the j-th block of $s_{\mathbf{K}}$, $s_{\mathbf{V}}$.

Write $z^{int}_{\mathbf{K}_j^{q2}}$,$z^{int}_{\mathbf{V}_j^{q2}}$ to HBM as the j-th block of $z^{int}_{\mathbf{K}^{q2}}$, $z^{int}_{\mathbf{V}^{q2}}$.

end for

On chip, compute $\mathbf{O}_i = \mathrm{diag}(\ell_i^{(T_c)})^{-1}\mathbf{O}_i^{(T_c)}$.

On chip, compute $L_i = m_i^{(T_c)} + \log(\ell_i^{(T_c)})$.

Write $\mathbf{O}_i$ to HBM as the i-th block of $\mathbf{O}$.

Write L_i to HBM as the i-th block of L.

end for

Return the output $\mathbf{O}$ and the logsumexp L.

Algorithm 2 *TurboAttention Decode*

Require: Matrices $\mathbf{Q}, \in \mathbb{R}^{1\times d}$ in HBM, compressed KV cache $\mathbf{K}^{q2}, \mathbf{V}^{q2} \in \mathbb{R}^{N\times d}$, integer scale of KV cache $s^{int}_{\mathbf{K}^{q2}}$,$s^{int}_{\mathbf{V}^{q2}}$, floating scale s_K, s_V, zero point of KV cache $z^{int}_{\mathbf{K}^{q2}}$, $z^{int}_{\mathbf{V}^{q2}}$, block size B_c, Softmax approximate algorithm *SAS*.

Divide $\mathbf{K}^{q2}, \mathbf{V}^{q2}$ in to $T_c = \left\lceil \frac{N}{B_c} \right\rceil$ blocks $\mathbf{K}^{q2}_1, \ldots, \mathbf{K}^{q2}_{T_c}$ and $\mathbf{V}^{q2}_1, \ldots, \mathbf{V}^{q2}_{T_c}$, of size $B_c \times d$ each.

Divide $s^{int}_{\mathbf{K}^{q2}}$,$s^{int}_{\mathbf{V}^{q2}}$,$z^{int}_{\mathbf{V}^{q2}}$, block size $B_c$into $T_c = \left\lceil \frac{N}{B_c} \right\rceil$ blocks accordingly .

Divide the output $\mathbf{O} \in \mathbb{R}^{1\times d}$ into T_c blocks $\mathbf{O}_i, \ldots, \mathbf{O}_{T_c}$ of size $1 \times \frac{d}{T_c}$ each, and divide the logsumexp L into T_r blocks $L_i, \ldots, L_{T_c}$ of size 1 each.

Load $\mathbf{Q}$ from HBM to on-chip SRAM.

On chip, initialize $\mathbf{O}_j = (0)_{1\times \frac{d}{T_c}} \in \mathbb{R}^{1\times d}, \ell^{(0)}_i = (0)_{B_c} \in \nVdash^{B_c}, m^{(0)}_i = (-\infty)_{B_c} \in \mathbb{R}^{B_c}$.

for $1 \leq j \leq T_c$ **do**

  Load $\mathbf{K}^{q2}_j, \mathbf{V}^{q2}_j$ from HBM to on-chip SRAM.

  On chip, compute $s_{\mathbf{Q}} = \frac{max(abs(\mathbf{Q}))}{119}, \mathbf{Q}^{q1} = \left\lceil \frac{\mathbf{Q}}{s_{\mathbf{Q}}} \right\rfloor \mathbf{K}^{q1}_j = \mathbf{K}^{q2}_j \cdot s^{int}_{\mathbf{K}^{q2}_j} + z_{\mathbf{K}^{q2}_j}, \mathbf{V}^{q1}_j = \mathbf{V}^{q2}_j \cdot s^{int}_{\mathbf{V}^{q2}_j} + z_{\mathbf{V}^{q2}_j}$

  On chip, compute $\mathbf{S}_j = s_{\mathbf{Q}} \cdot s_{\mathbf{K}_j} \cdot \mathbf{Q}^{q1}\mathbf{K}^{q1}_j \in \mathbb{R}^{1\times B_c}$.

  On chip, compute $m_j = \max(m_{(j-1)}, \mathrm{rowmax}(\mathbf{S}_j)) \in \mathbb{R}^1$, $\tilde{\mathbf{P}}^{(j)} = SAS(\mathbf{S}_j - m_j) \in \mathbb{R}^{1\times B_c}$ (pointwise), $\ell^{(j)} = SAS(m^{j-1} - m^{(j)})\ell^{(j-1)} + \mathrm{rowsum}(\tilde{\mathbf{P}}^{(j)}) \in \mathbb{R}^1$.

  On chip, compute $s_{\tilde{\mathbf{P}}^{(j)}} = \frac{max(abs(\tilde{\mathbf{P}}^{(j)})}{119}, Q(\tilde{\mathbf{P}}^{(j)}) = \left\lceil \frac{\tilde{\mathbf{P}}^{(j)}}{s_{\tilde{\mathbf{P}}^{(j)}}} \right\rfloor$,

  On chip, compute $\mathbf{O}^{(j)} = \mathrm{diag}(SAS(m^{(j-1)} - m^{(j)}))^{-1}\mathbf{O}^{(j-1)} + s_{\tilde{\mathbf{P}}^{(j)}} \cdot s_{\mathbf{V}_j} \cdot Q(\tilde{\mathbf{P}}^{(j)})\mathbf{V}^{q1}_j$.

end for

On chip, compute $\mathbf{O} = \mathrm{diag}(\ell^{(T_c)})^{-1}\mathbf{O}^{(T_c)}$.

On chip, compute $L = m^{(T_c)} + \log(\ell^{(T_c)})$.

Write $\mathbf{O}$ to HBM.

Return the output $\mathbf{O}$ and the logsumexp L.

B DETAILED ALGORITHM OF *SAS*

Algorithm 3 Sparsity-based Softmax Approximation

Require: Input matrix $X \in \mathbb{R}^{N\times d}$, integer threshold n_r, look-up table T with $T[n_r + 1]$ set to 0, polynomial approximator function $POLY$.

Step 1: Normalize input values

$x_{max} = \mathrm{rowmax}(X)$ *// Max value per row*

$X = X - x_{max}$

Step 2: Apply sparsity threshold

$X[X < n_r] = n_r + 1$

Step 3: Separate integer and decimal parts

$X = X_{int} + X_{decimal}$

Step 4: Approximate exponential values

$X_{lut} = T[X_{int}]$ *// Lookup for integer part*

$X_{approx} = POLY(X_{decimal})$ *// Polynomial approx. for decimal part*

$X = X_{lut} \times X_{approx}$

Step 5: Normalize with row-wise sum

$X_{sum} = \mathrm{rowsum}(X)$

$X = \frac{X}{X_{sum}}$

Return X

C DETAILED ILLUSTRATION OF *FlashQ* AND *TurboAttention*

In this section, we separately evaluate the accuracy degradation caused by *FlashQ* and *SAS*. Results are shown in Table 4. It demonstrated that both techniques are near loss-less.

Table 4. Separate discussion about accuracy degradation of FlashQ and SAS

Model	Dataset	Method	Acc
LLaMA3-8B-inst	AQuA	FP16	50.79
LLaMA3-8B-inst	AQuA	FlashQ-4bit	49.60
LLaMA3-8B-inst	AQuA	SAS	50.12
LLaMA3-8B-inst	AQuA	FlashQ-4bit + SAS	48.03