Osayamen Jonathan Aimuyo, oja7@cornell.edu

Aristos & Grigora: Efficient MoE Collective Communication and Pipelining

Notation	Description
$H(x,\, y)$	Heaviside Function $H(x)$, where $H(0) = y$ following [Con23]
k	Number of experts selected during token routing, $k \in \mathbb{N}$ see *top_k* in [Sha+17; FZS22]
X	Set of all experts
N	Set of all nodes, $N \subset \mathbb{Z}^+$
J_n	Set of all ranks in node n, $J_n \subset \mathbb{Z}^+$
W	World of workers, note $\lvert N\rvert \cdot \lvert J_n\rvert = \lvert W\rvert > 0$
ε	Optimism factor, $\varepsilon \in \{0, \ldots,\, \lvert W\rvert - 1\} \subset \mathbb{Z}^+$, $\forall w \in W\ \varepsilon_w = \varepsilon$
G_j^n	Worker with rank j in node n
X_j^n	Set of experts hosted on G_j^n, $X_j^n \subseteq X$
ϕ	Tensor mapping G_j^n tokens T to experts $X' \subseteq X$, $\phi \in \mathbb{R}^{\lvert T\rvert \times k}$
S	Sharding specification tensor mapping X to $W' \subseteq W$

Table 1: Notation

Algorithm 1: *Aristos* executed by each G_j^n

Data: Q: a blocking task FIFO queue
Require: $X_j^n,\, S,\, \phi,\, \varepsilon,\, \lvert W\rvert$

begin
- $\xi \leftarrow \varepsilon$
- $G_\phi \leftarrow$ **GetWorkers**$(\phi,\, S)$
- $\mu \leftarrow \lvert G_\phi\rvert + H(\lvert X_j^n\rvert,\, 0) \cdot (\lvert W\rvert - 1)$ `// Upper bound for tasks`
- $flag \leftarrow$ **False**
- $W' \leftarrow S.W'$
- **if** $G_j^n \in G_\phi$ **then**
  - $\mu \leftarrow \mu + 1$
  - $\xi \leftarrow \xi + 1$
- **end if**
- `// Send token slices to corresponding workers`
- **broadcast**$(shouldProcess,\, \phi,\, G_\phi)$
- **while** $Q.timeout ==$ **False and** $\mu > 0$ **do**
  - **if** $flag ==$ **False and** $\xi == 0$ **then**
    - Q.**ActivateCountdown**$()$
    - $flag \leftarrow$ **True**
  - **end if**
  - $t \leftarrow Q$.**take**$()$
  - **if** $t.processed ==$ **True then**
    - **PostProcessing**$(\phi,\, t)$
    - $\xi \leftarrow \xi - 1$
    - $\mu \leftarrow \mu - 1$
  - **else if** $t.shouldProcess ==$ **True then**
    - $\theta =$ **ExpertProcessing**$(t.\tau,\, X_j^n)$
    - **Send**$^\dagger(processed,\, \theta,\, t.workers)$
    - $W' \leftarrow W' - t.workers$
    - $\mu \leftarrow \mu - 1$
- **end while**
- **if** $\epsilon > 0$ **and** $Q.timeout ==$ **True then**
  - **broadcast**$(processed,\, W')$
- **end if**

end

Notation	Description
W	Set of multi-node, distributed workers, $\lvert W\rvert > 0$
g	Subset of workers, $g \subseteq G$, $g = (V_g, E_g)$
$\mathcal{G}$	Set of all g, note $1 \leq \lvert\mathcal{G}\rvert \leq \lvert V\rvert$
$i,\ j$	Workers $i,\ j \in W$
α_{ij}	α cost between i and j, $\alpha \in \mathbb{R}^+$ See [Sol16] for an overview
β_{ij}	β cost between i and j, $\beta \in \mathbb{R}^+$
G	Graph adjacency matrix where $G[i, j] = (\alpha_{ij},\ \beta_{ij})$ and $G = (V, E)$, $V = W$
m_j	Memory capacity available for experts on worker j
$\mathcal{X}$	Set of all experts. For **valid** groups $g^* \in \mathcal{G}$, $\sum_{j\in V_{g^*}} m_j \geq \lvert\mathcal{X}\rvert$
$\pi(g)$	**Expert** compute cost of group g
f_x	Floating-point operations for computing x [Nar+21]
$\mathcal{F}_j$	*Actual* FLOPS of worker j
$\mathcal{C}_j$	Communication cost for worker j
$T_\rho(n)$	Time for all-reduce on n processes, see [Rab04; TRG05]

Table 2: Notation for *Grigora*

Grigora

Inputs

$W, \quad G, \quad F_j \quad \forall j \in W, \quad f_x \quad \forall x \in \mathcal{X}, \quad d_{ij} \quad \forall i, j \in W, \quad \eta$: frequency of using an edge

$$\gamma(\omega) = \frac{(2 + r) \cdot L \cdot B}{i \cdot \omega \cdot b},$$

where r: activation re-computation amount [Nar+21], L: num of layers, B: global batch, i: MoE layer frequency [Lie+24], b: mini-batch and ω: effective world size, fixed as W below but varies later.

Objective

$$\min_{g\in\mathcal{G}} \quad T_g, \quad \text{where } T_g = \gamma(\omega)\left(\pi(g) + \max_{j\in V_g} \mathcal{C}_j\right) + T_\rho(\lvert\mathcal{G}\rvert) \tag{1}$$

We derive local optima for Equation 1 using a novel *deterministic* graph partitioning algorithm reminiscent of Kruskal's classical MST algorithm [Wik24]. We do not yet have a tight bound, but a loose analysis shows that Grigora runs in $\mathcal{O}(n^3)$ time. We reiterate T_g below, clarifying $\pi(g)$, and $\mathcal{C}$. We define T_ρ per the worst case latency complexity of [Rab04]

$$T(g) = \gamma(\omega)\left(\frac{\sum_{x\in\mathcal{X}} f_x}{\sum_{j\in g} \mathcal{F}_j} + \max_{i\in V_g} \sum_{(i,j)\in E_g} \eta(\alpha_{ij} + \frac{d}{\lvert V_g\rvert}\beta_{ij})\right) + 2(\lvert\mathcal{G}\rvert - 1)\left(\alpha^* + d\frac{\beta^*}{\lvert\mathcal{G}\rvert}\right) \tag{2}$$

where $\forall a, b \in \mathcal{G}$, $\alpha^* = \max\{\alpha_{ab}\}$ and $\beta^* = \max\{\beta_{ab}\}$. Below is how we implement $T(g)$ in code.

$$\frac{T(g)}{\gamma(\omega)} = \frac{\sum_{X\in\mathcal{X}} f_x}{\sum_{j\in g} \mathcal{F}_j} + \max_{i\in V_g} \sum_{(i,j)\in E_g} \eta(\alpha_{ij} + \frac{d}{\lvert V_g\rvert}\beta_{ij}) + \frac{2(\lvert\mathcal{G}\rvert - 1)}{\gamma(\omega)}\left(\alpha^* + d\frac{\beta^*}{\lvert\mathcal{G}\rvert}\right) \tag{3}$$

References

[Rab04] Rolf Rabenseifner. "Optimization of Collective Reduction Operations". In: *Computational Science - ICCS 2004*. Ed. by Marian Bubak, Geert Dick van Albada, Peter M. A. Sloot, and Jack Dongarra. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 1–9. ISBN: 978-3-540-24685-5. URL: https://link.springer.com/content/pdf/10.1007/978-3-540-24685-5_1.pdf.

[TRG05] Rajeev Thakur, Rolf Rabenseifner, and William Gropp. "Optimization of Collective Communication Operations in MPICH". In: *The International Journal of High Performance Computing Applications* 19.1 (2005), pp. 49–66. DOI: 10.1177/1094342005051521. eprint: https://doi.org/10.1177/1094342005051521. URL: https://doi.org/10.1177/1094342005051521.

[Sol16] Edgar Solomonik. *CS 598: Communication Cost Analysis of Algorithms Lecture 1: Course motivation and overview; collective communication*. 2016. URL: https://solomonik.cs.illinois.edu/teaching/cs598%5C_fall2016/lectures/lec1.pdf.

[Sha+17] Noam Shazeer et al. "Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer". In: *CoRR* abs/1701.06538 (2017). arXiv: 1701.06538. URL: http://arxiv.org/abs/1701.06538.

[Nar+21] Deepak Narayanan et al. "Efficient large-scale language model training on GPU clusters using megatron-LM". In: *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*. SC '21. St. Louis, Missouri, Association for Computing Machinery, 2021. ISBN: 9781450384421. DOI: 10.1145/3458817.3476209. URL: https://doi.org/10.1145/3458817.3476209.

[FZS22] William Fedus, Barret Zoph, and Noam Shazeer. "Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity". In: *Journal of Machine Learning Research* 23.120 (2022), pp. 1–39. URL: http://jmlr.org/papers/v23/21-0998.html.

[Con23] PyTorch Contributors. *Torch.heaviside*. 2023. URL: https://pytorch.org/docs/stable/generated/torch.heaviside.html.

[Lie+24] Opher Lieber et al. *Jamba: A Hybrid Transformer-Mamba Language Model*. 2024. arXiv: 2403.19887 [cs.CL].

[Wik24] Wikipedia contributors. *Kruskal's algorithm — Wikipedia, The Free Encyclopedia*. [Online; accessed 6-May-2024]. 2024. URL: https://en.wikipedia.org/w/index.php?title=Kruskal%27s_algorithm&oldid=1221098161.