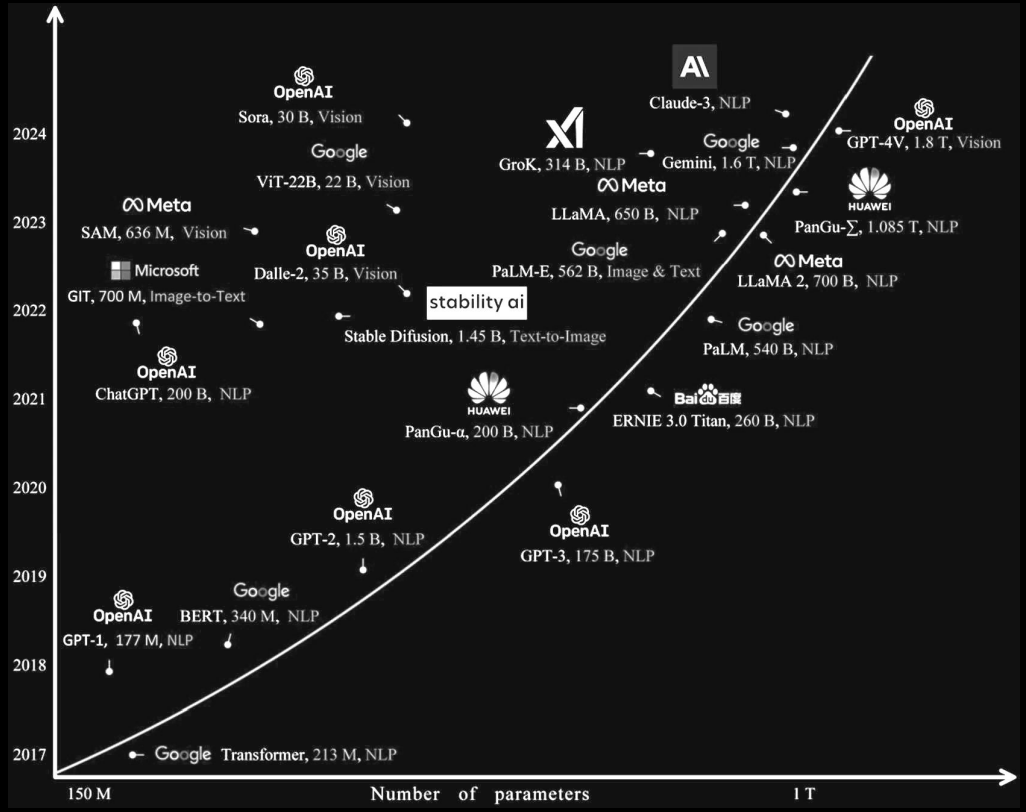
STyGIANet

Birkhoff Decompositions and Photonic Interconnects
Wait! Don't Forget the Compute!

Eliezer Amponsah, Vamsi Addanki

Emerging Trends: Increasing Size of AI Models

- Increasing size of AI models and workloads in DCN

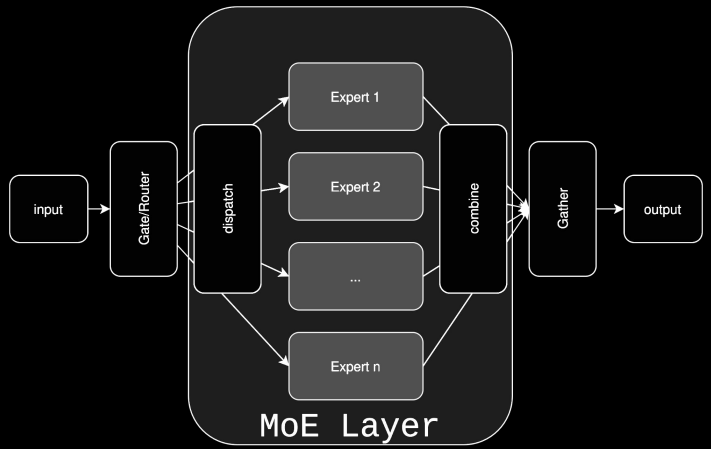


Tu, X., He, Z., Huang, Y. et al. An overview of large AI models and their applications. Vis. Intell. **2**, *34 (2024).*

STyGIANet

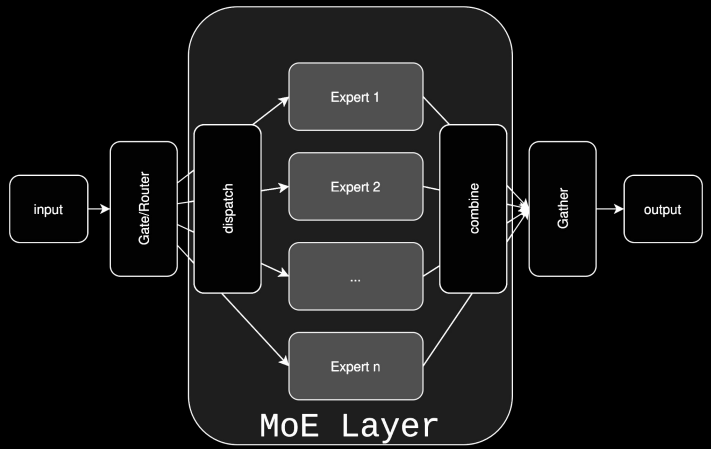
Emerging Trends: Increasing Size of AI Models

- MoE Innovation



Emerging Trends: Increasing Size of AI Models

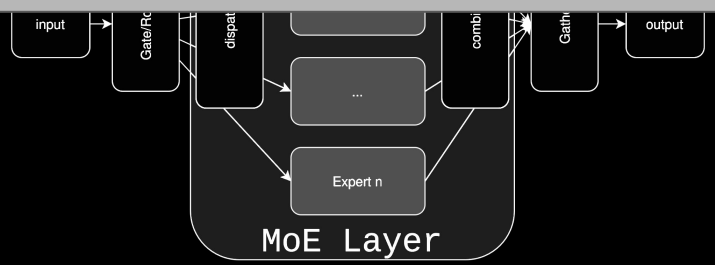
- MoE Innovation
- Sparse Activations
  - DeepSeek-V3: 671B - 37B activated
  - Qwen 3.5: 397B - 17B activated
  - Mixtral 8x22B: 141B - 39B activated
- Gate/Router



Emerging Trends: Increasing Size of AI Models

- MoE Innovation
- Sp
  -
  -
  -
- G

Compute no longer scales with model size



The Rise of Dynamic Communication Patterns

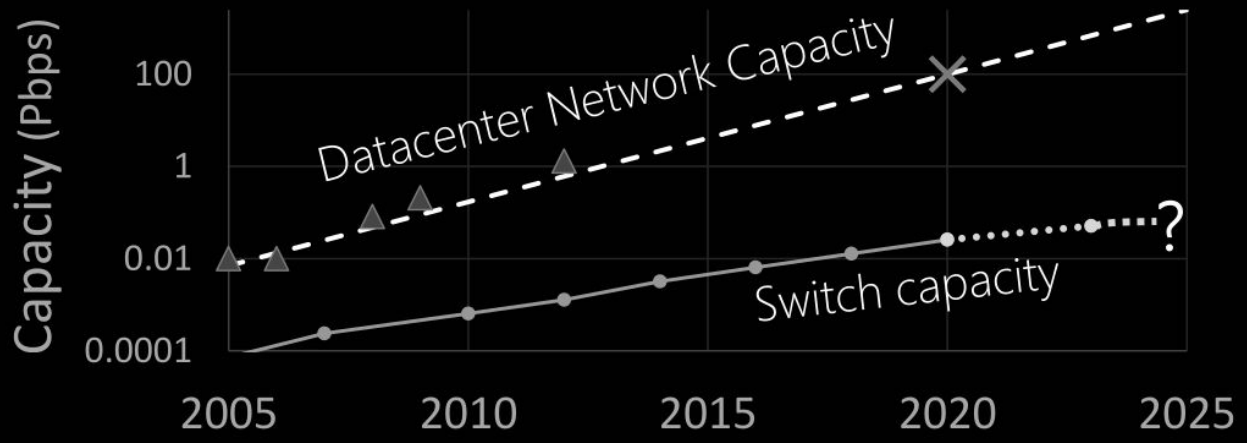
- All-to-All, unlike allreduce that allows static ring
- All to all structure changes often!

The Rise of Dynamic Communication Patterns

- All-to-All, unlike allreduce that allows static ring
- All to all structure changes often!
- Exacerbated in MoE
  - Varies on each input
  - Each layer

The Need for Silicon Photonic Interconnects

- Electrical interconnects are facing saturation with the near-end of moore's law...

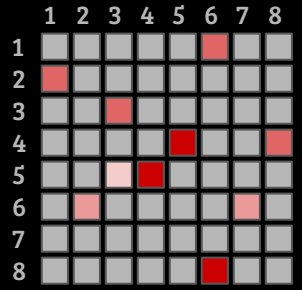


Ballani, Hitesh, et al. "Sirius: A flat datacenter network with nanosecond optical switching."

Emerging Challenge

- Communication is dynamic
  - One topology cannot rule them all
  - Reconfiguration becomes necessary!
- Photonic Interconnects are circuit switched
  - Requires reconfiguration to satisfy all-to-all communication

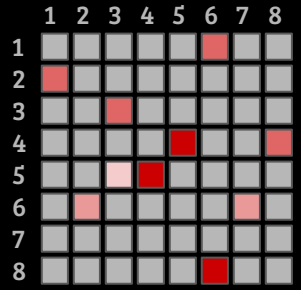
Traditional Approach: Birkhoff Decomposition



Demand Matrix

Row = sender
Column = receiver
$D_{i,j}$ = src i sends to recvr j

Traditional Approach: Birkhoff Decomposition



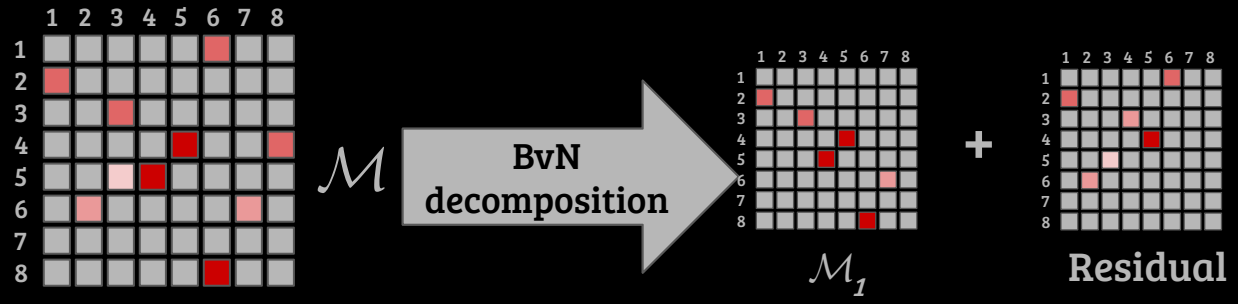
$\mathcal{M}$

BvN decomposition

Find the minimum non-zero entry and extract a permutation (matching)

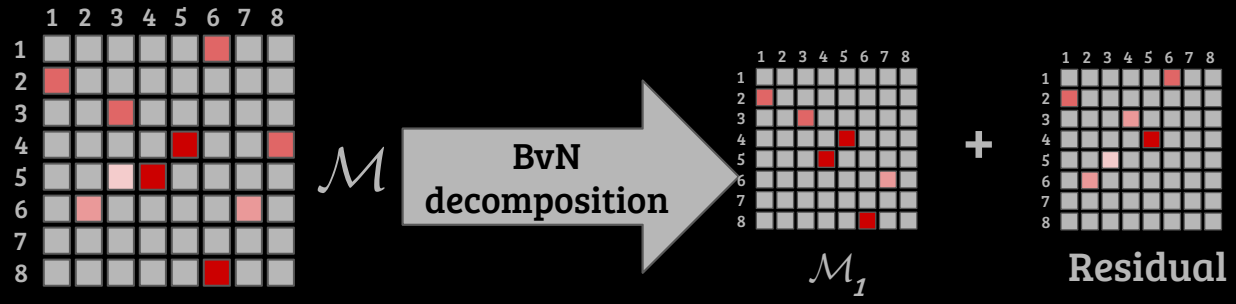
Demand Matrix

Traditional Approach: Birkhoff Decomposition



Demand Matrix

Traditional Approach: Birkhoff Decomposition



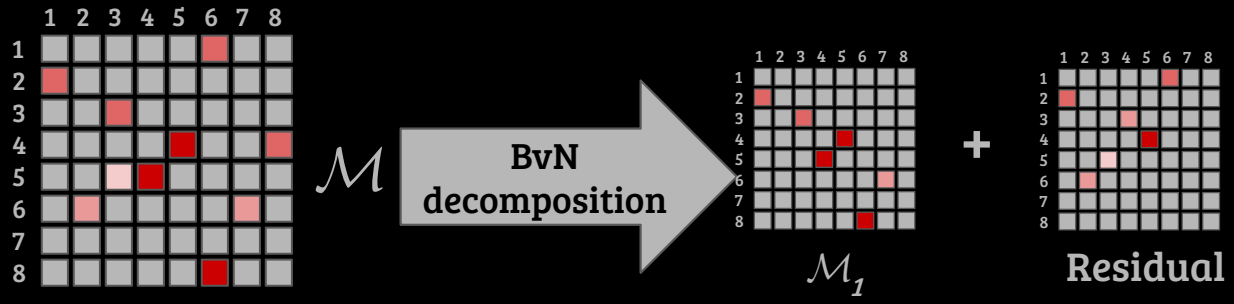
Demand Matrix

A requirement for Bvn

Doubly stochastic

All rows and column sum to the same value

Traditional Approach: Birkhoff Decomposition



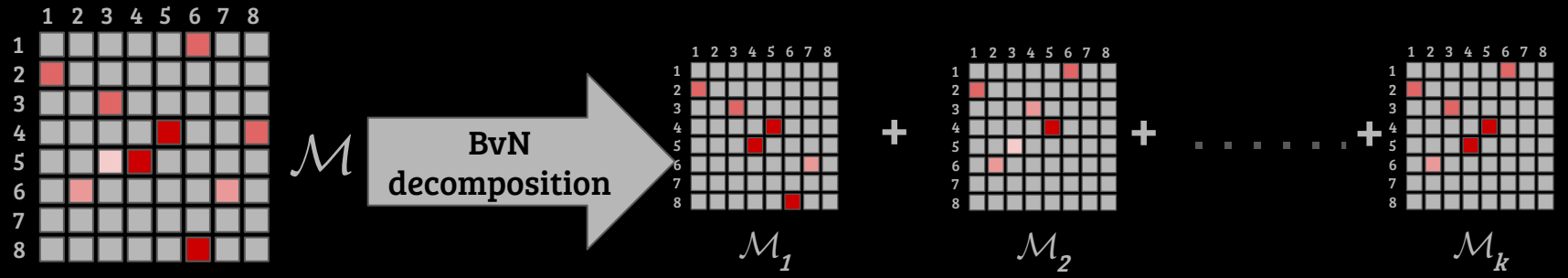
Demand Matrix

A property of Bvn

Doubly stochastic

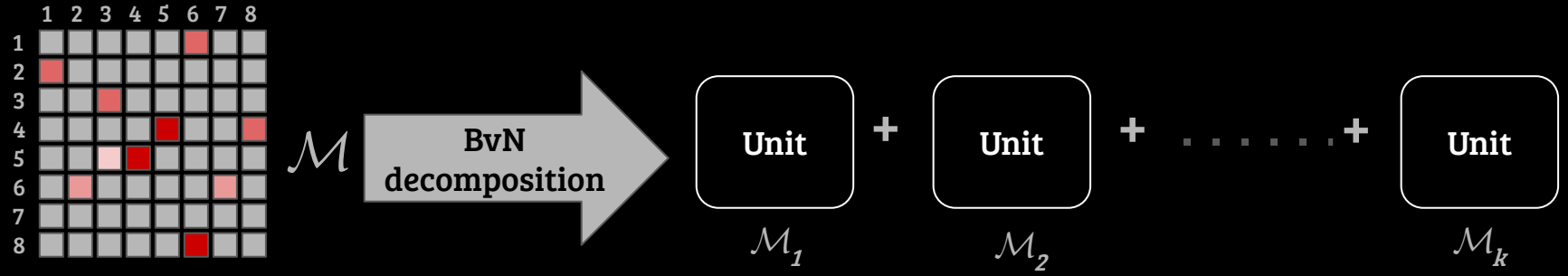
All rows and column sum to the same value

Traditional Approach: Birkhoff Decomposition



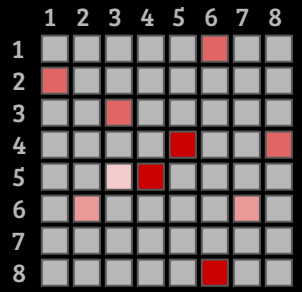
Demand Matrix

Traditional Approach: Birkhoff Decomposition



Demand Matrix

Traditional Approach: Birkhoff Decomposition



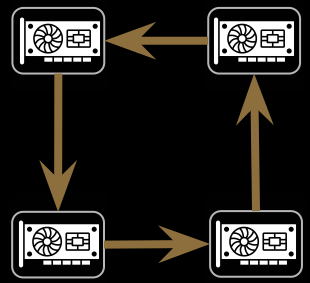
$\mathcal{M}$

BvN decomposition

Unit + Unit + ... + Unit

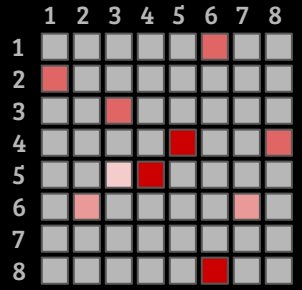
$\mathcal{M}_1$ $\mathcal{M}_2$ $\mathcal{M}_k$

Demand Matrix



t_1

Traditional Approach: Birkhoff Decomposition



$\mathcal{M}$

BvN decomposition

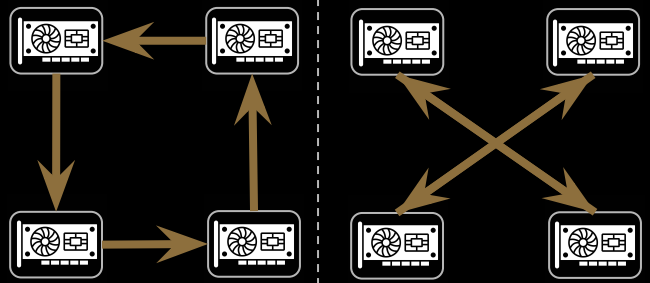
Unit + Unit + ... + Unit

$\mathcal{M}_1$ $\mathcal{M}_2$ $\mathcal{M}_k$

Demand Matrix

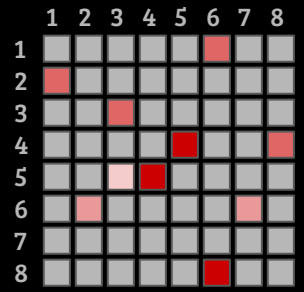
Reconfiguration

Circuit Switching



t_1 t_2

Traditional Approach: Birkhoff Decomposition



$\mathcal{M}$

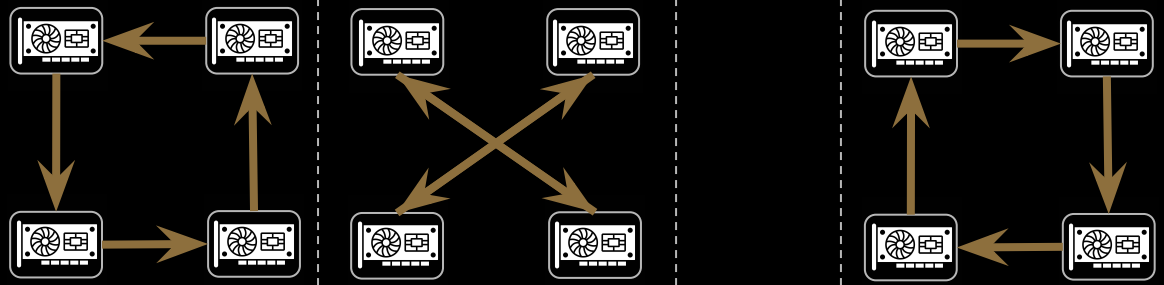
BvN decomposition

Unit + Unit + ... + Unit

$\mathcal{M}_1$ $\mathcal{M}_2$ $\mathcal{M}_k$

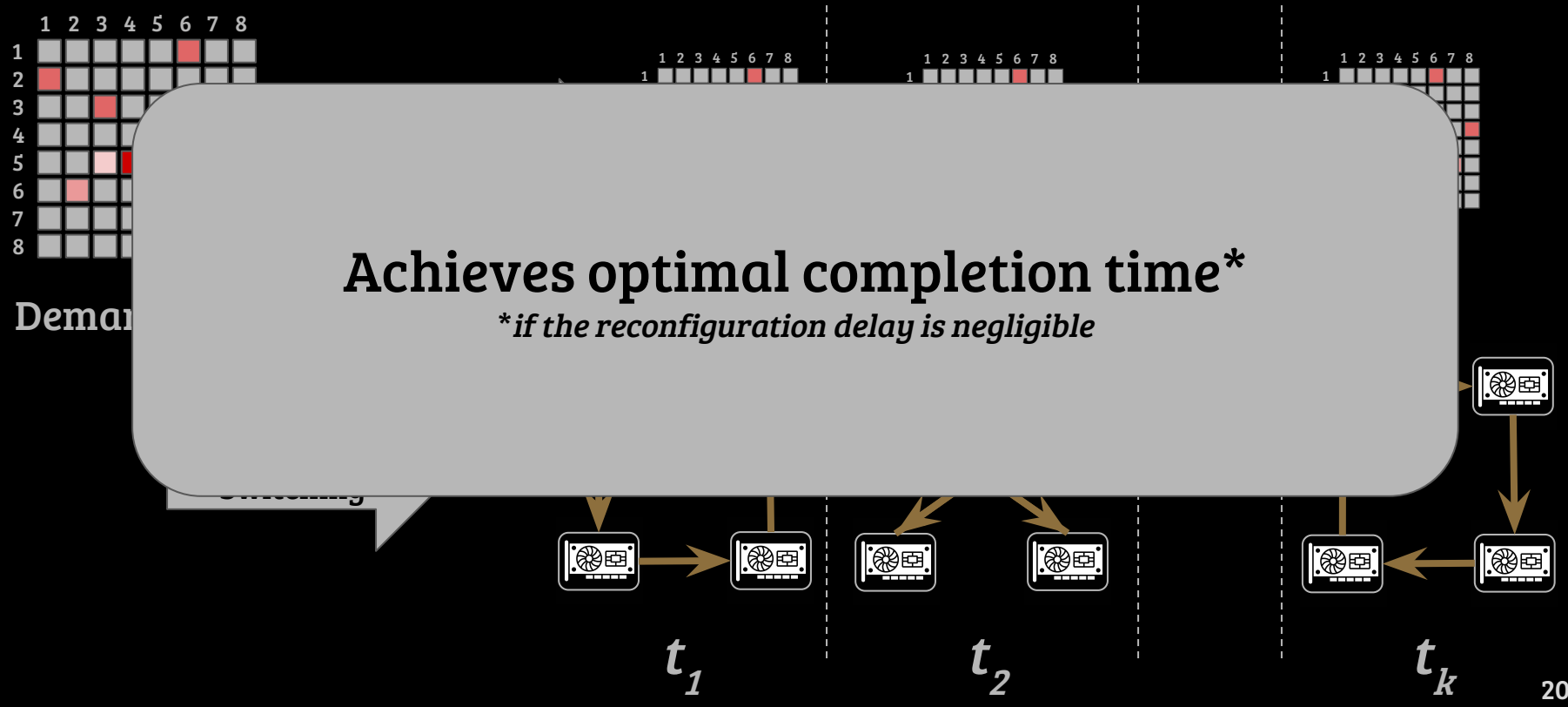
Demand Matrix

Circuit Switching



t_1 t_2 t_k

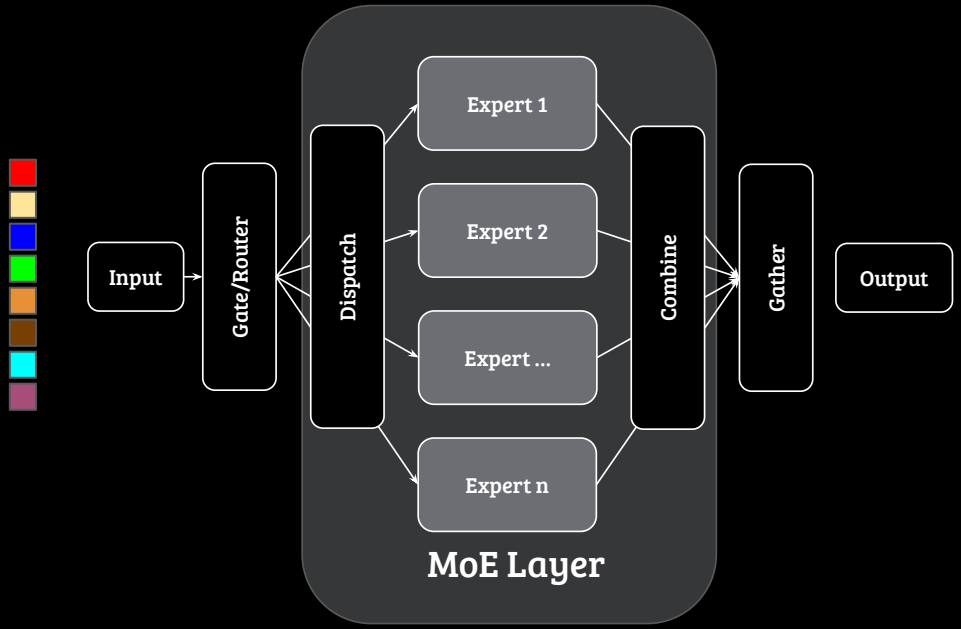
Traditional Approach: Birkhoff Decomposition



Birkhoff Decomposition is not Optimal for Makespan!

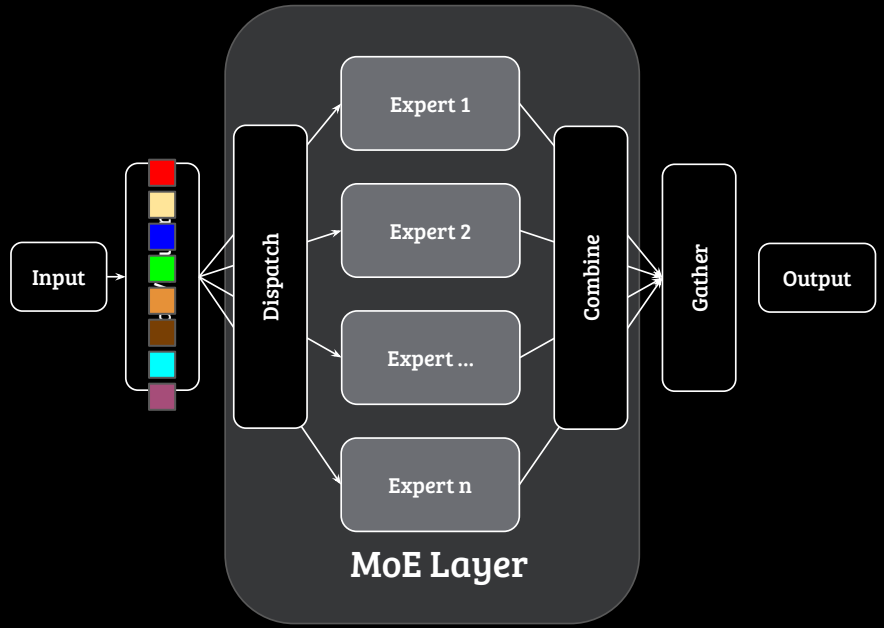
- MoE structure
- Doubly stochastic
  - Each row and column must sum to the same value
- Compute efficiency

MoE Structure



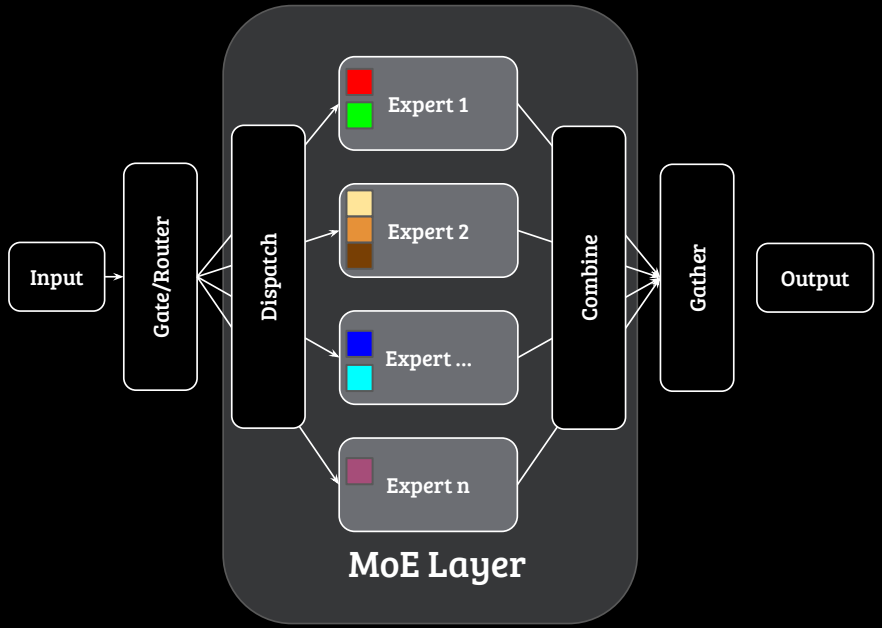
MoE Structure

Sparse computation: Router selects top-K experts



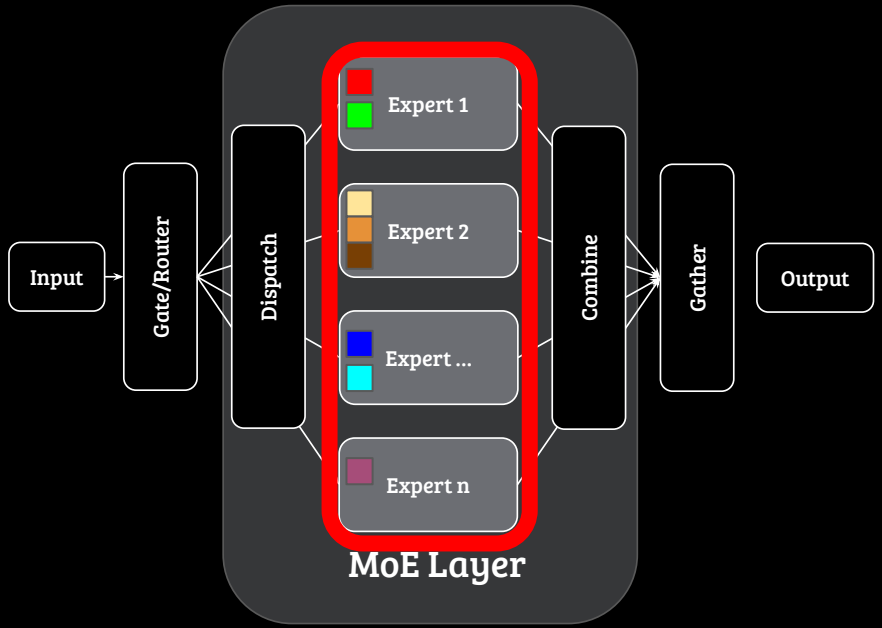
MoE Structure

Dispatch: All-to-All communication



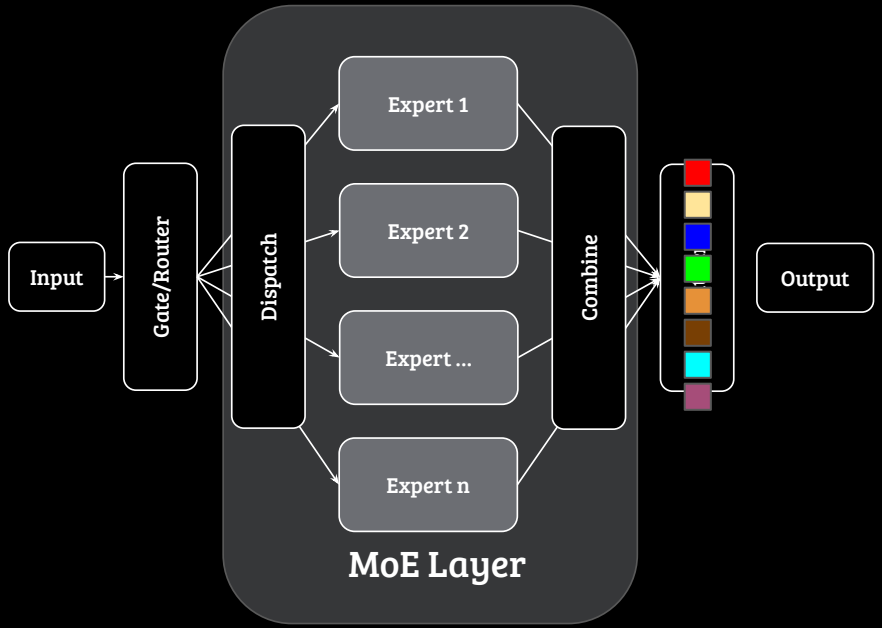
MoE Structure

Expert computation



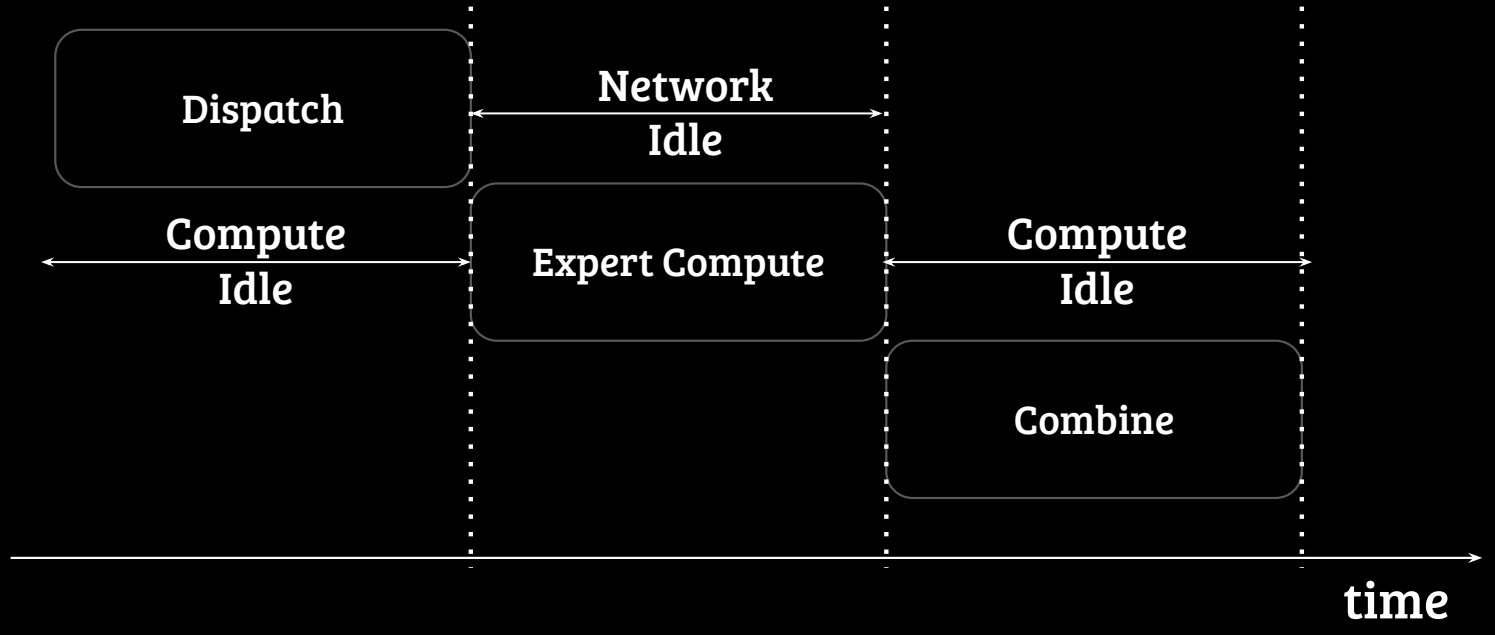
MoE Structure

Combine: All-to-All communication



MoE Structure

- Dispatch-compute-combine

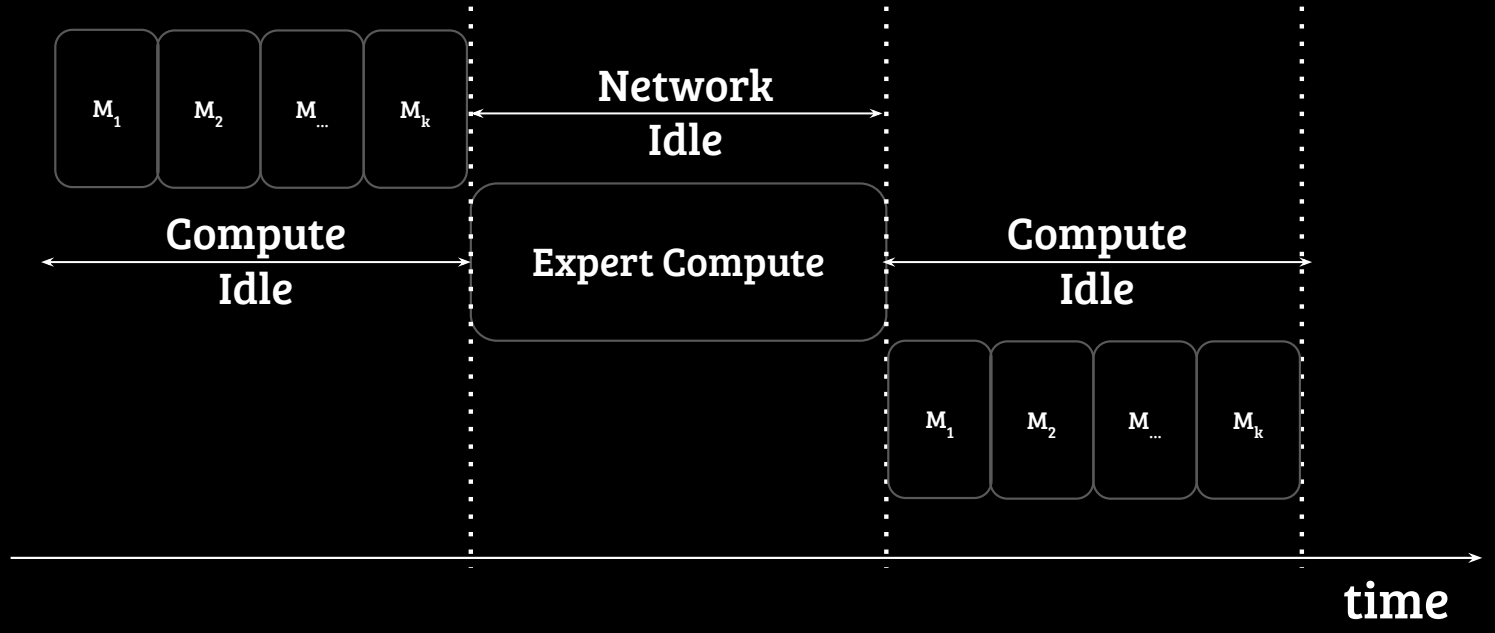


MoE Structure

- Dispatch-compute-combine

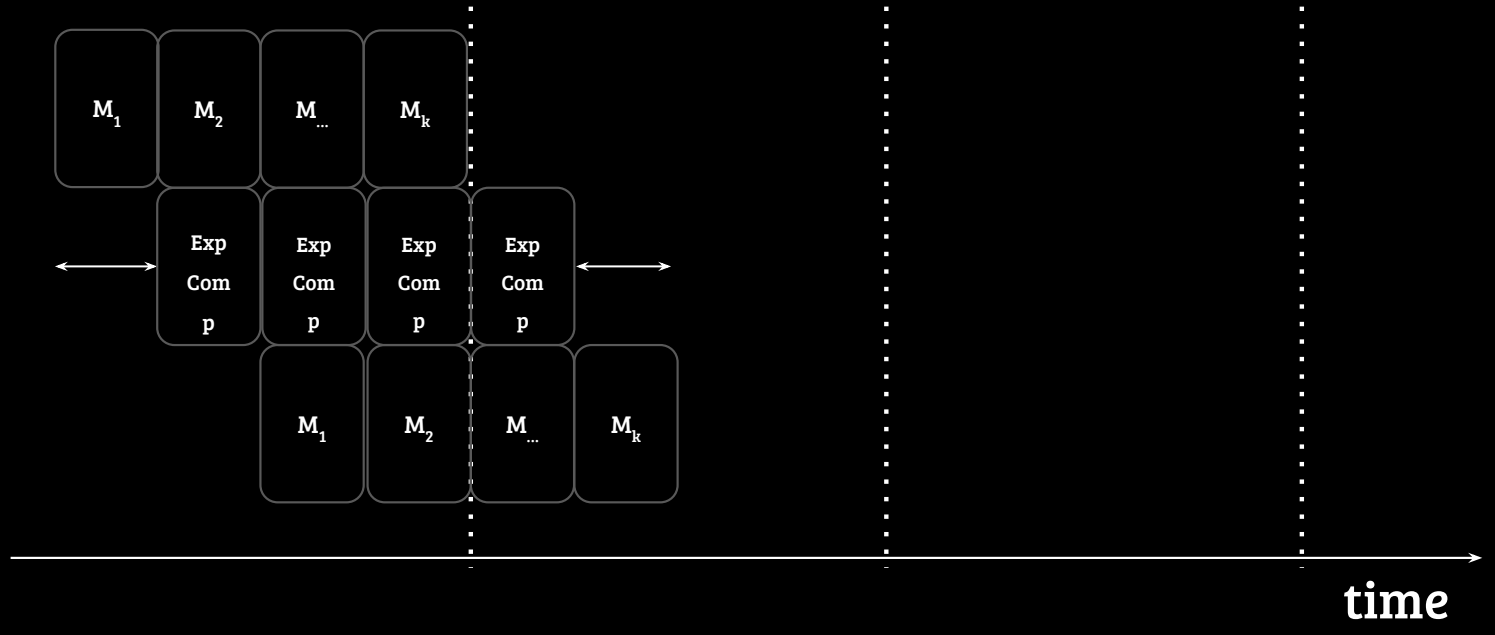
Sequential MoE Operations in OCS

- OCS decomposes the demand matrix



MoE Sequential Operation in OCS

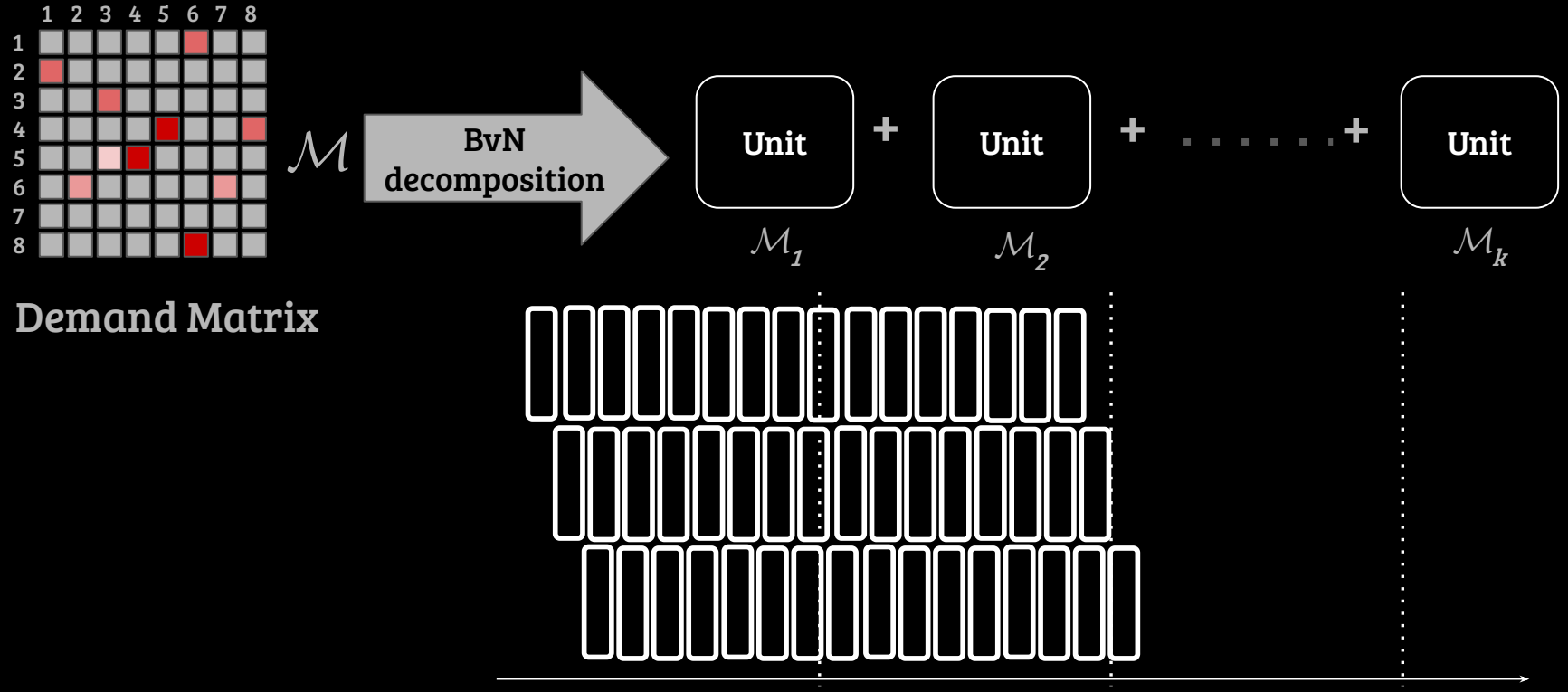
- Experts compute on received tokens immediately



MoE Sequential Operation in OCS

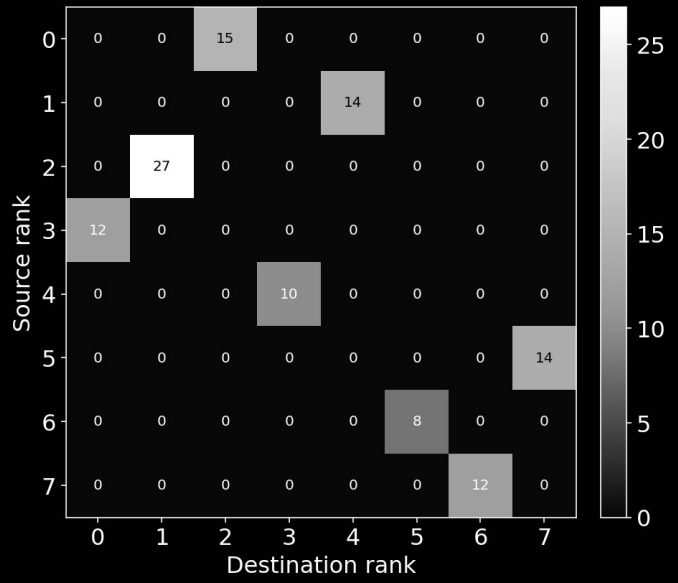
- Experts compute on received tokens immediately

BvN: Overlap and Theoretical Makespan Reduction



BvN Produces Small Batches Per Dispatch

- Many overlap opportunities
  - Up to N^2 matchings



Demand Matrix

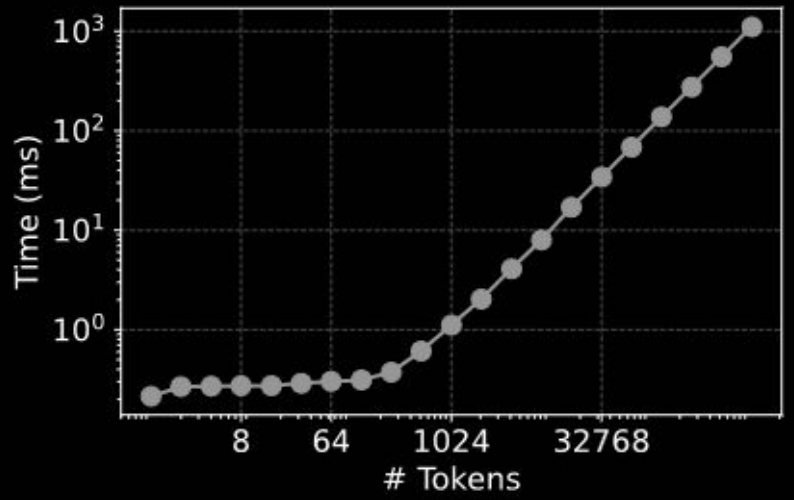
BVN

Unit + **Unit** + **Unit** + **Unit** + **Unit** . . . **Unit**

$\mathcal{M}_1$ $\mathcal{M}_2$ $\mathcal{M}_3$ $\mathcal{M}_4$ $\mathcal{M}_5$ $\mathcal{M}_k$

Problem 1: Compute Efficiency

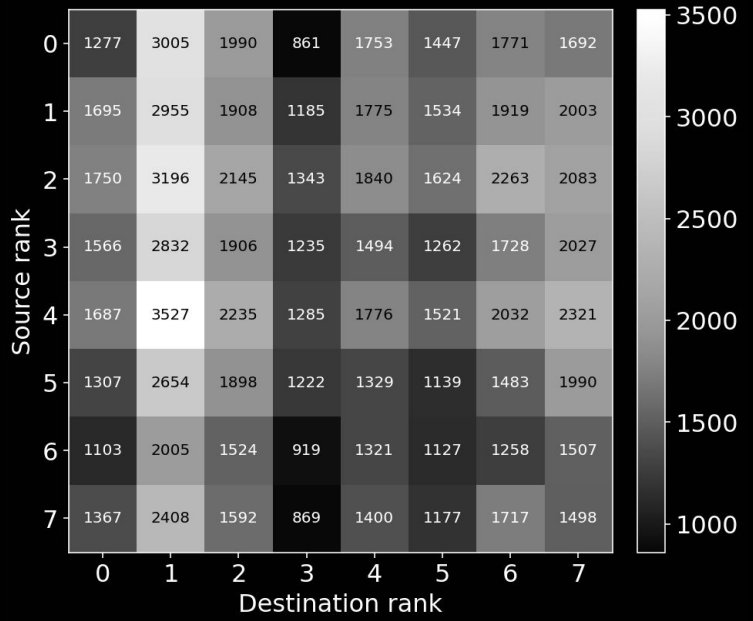
Smaller batches dominated by compute overheads (i.e. Kernel launches, synchronisation)



STyGIANet

Problem 2 : MoE All-to-All is **Rarely** Doubly Stochastic

- Communication imbalance
- Directly applying BvN leaves scheduling bubbles



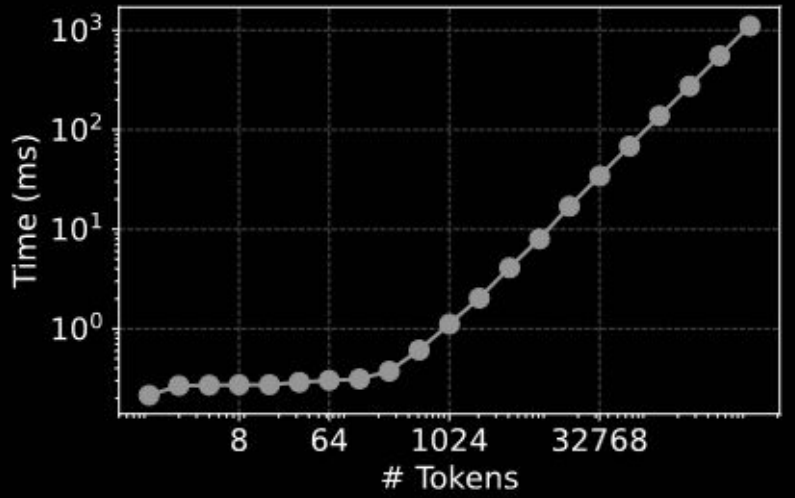
Problem 3: BvN Produces Small Batches Per Dispatch

Excessive number of matchings*

**Up to $O(N^2$ matchings). Costly if reconfiguration is not negligible*

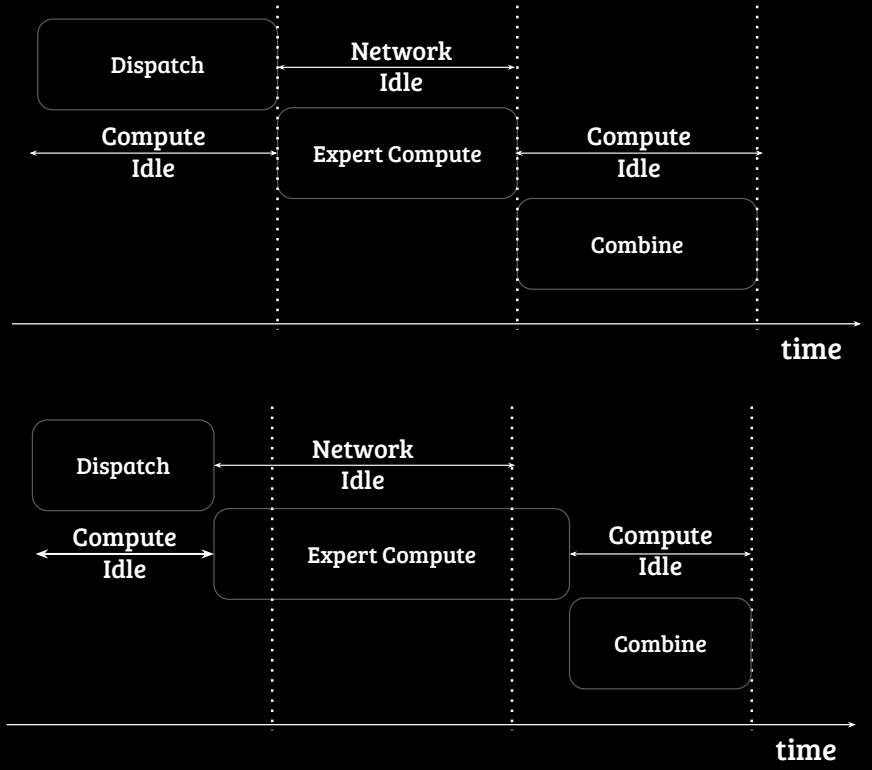
Together: BvN Decomposition is not Optimal for Makespan!

- Fragmented Compute
  - Wasted cycles on compute overheads
- Can increase batch size for same compute cost



Directive: Joint Optimize Network and Compute

- Efficiency in one operation does not equate to downstream efficiency



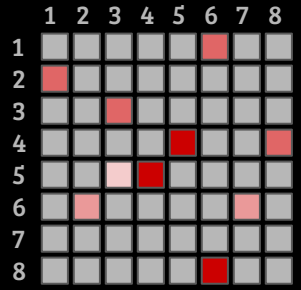
Decompose Differently

- Preserve large batches for compute efficiency
- Maintain beneficial properties of BvN
  - Incast/Outcast free
- Reduce number of matchings

Maxweight Decomposition

- Large Batches
- Matrix skew agnostic
- $\Theta(N-1)$ matchings

Maxweight Decomposition



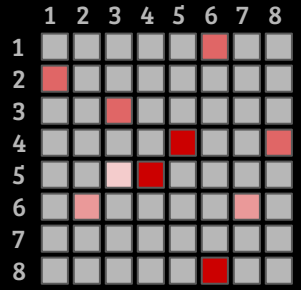
$\mathcal{M}$

MaxWeight decomposition

Find the minimum non-zero entry and extract a permutation (matching)

Demand Matrix

Maxweight Decomposition



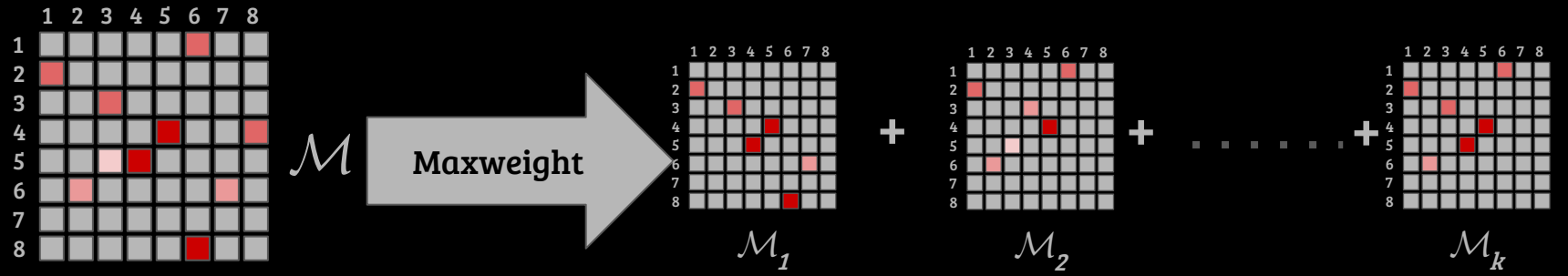
Demand Matrix

$\mathcal{M}$

MaxWeight decomposition

Find the maximum weight permutation (matching)

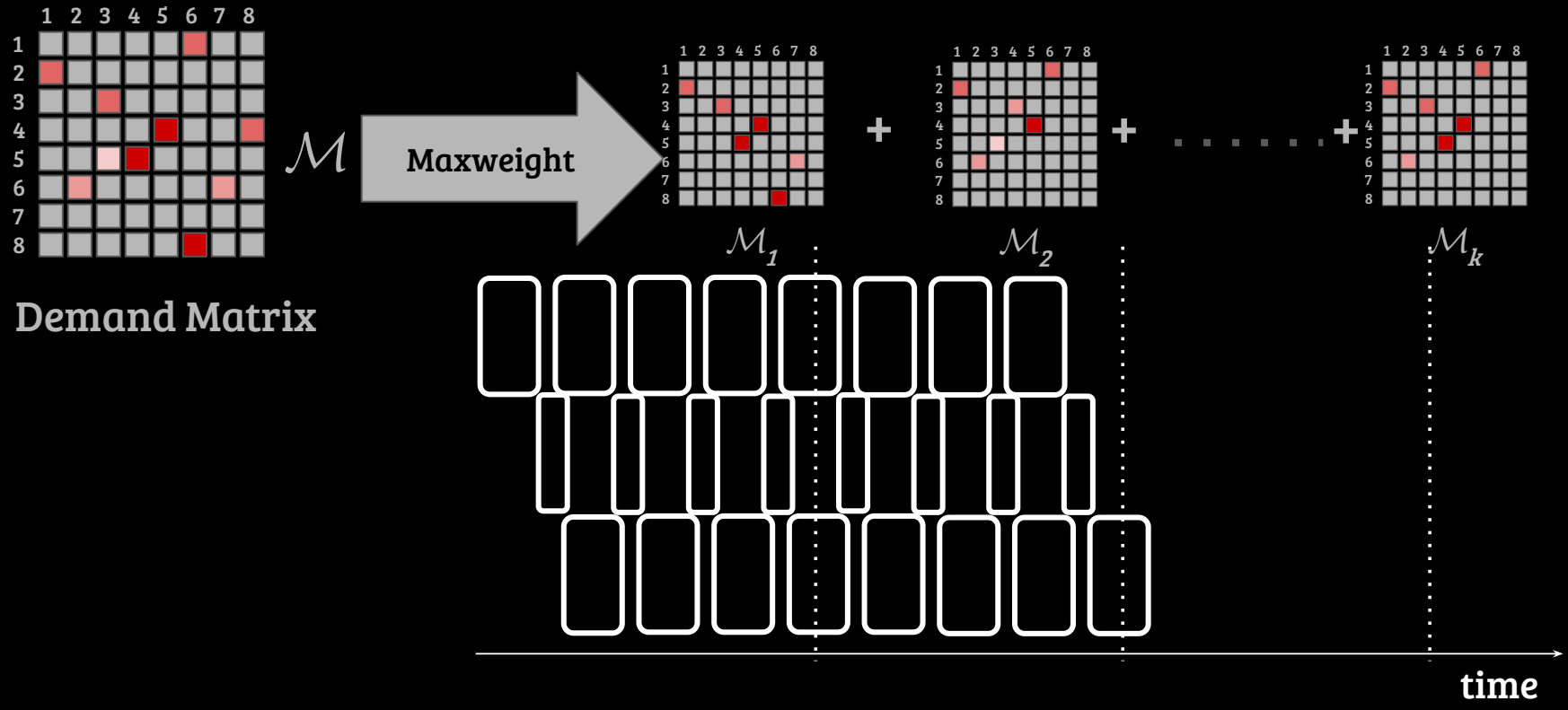
Maintain Large Batches for Compute Efficiency



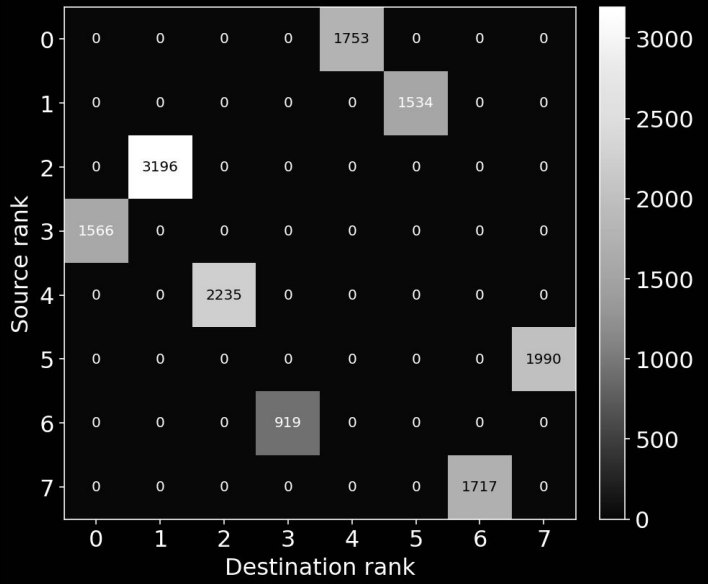
Demand Matrix

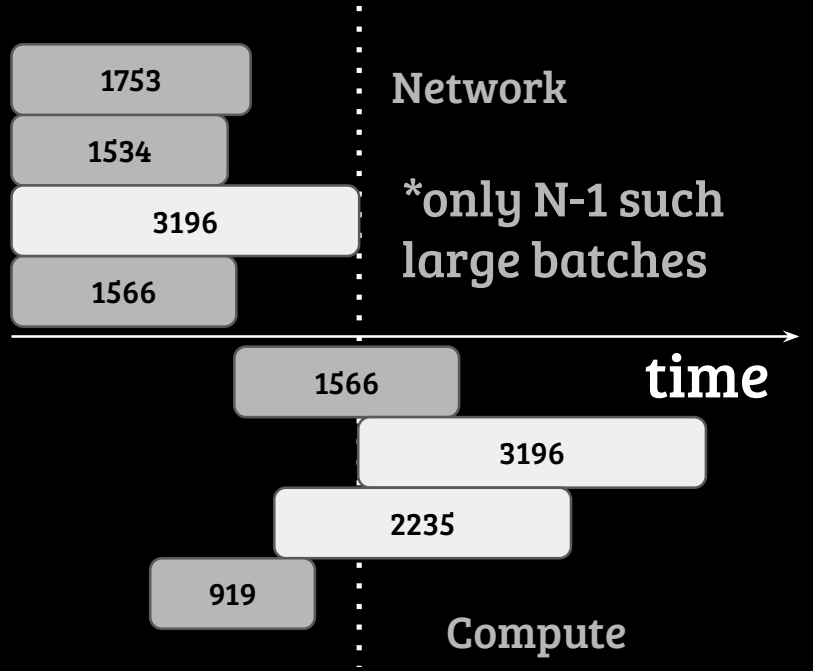
Only *n-1* matchings
&
Large batches

Maintain Large Batches for Compute Efficiency



Network Gaps but Computationally Efficient

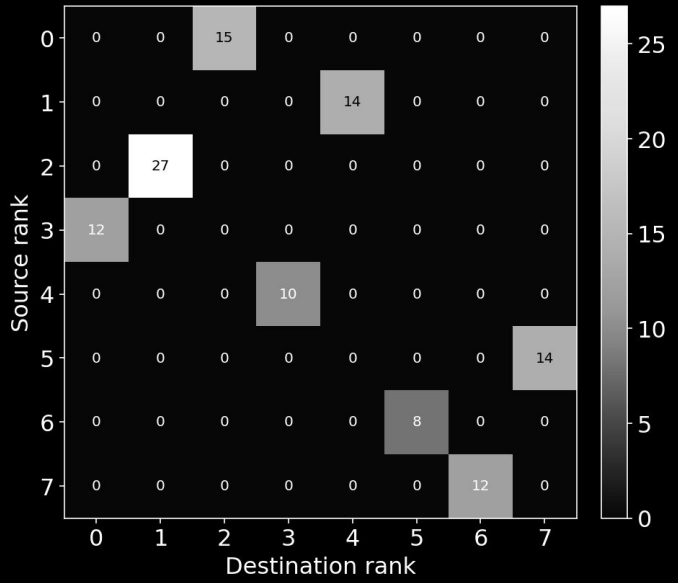




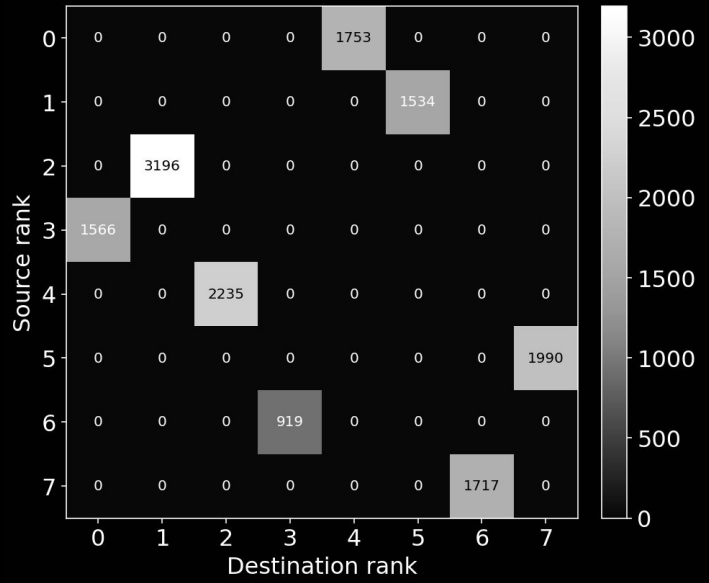
STyGIANet

Putting it All Together: Better Compute Efficiency

- Enough compute to hide communication

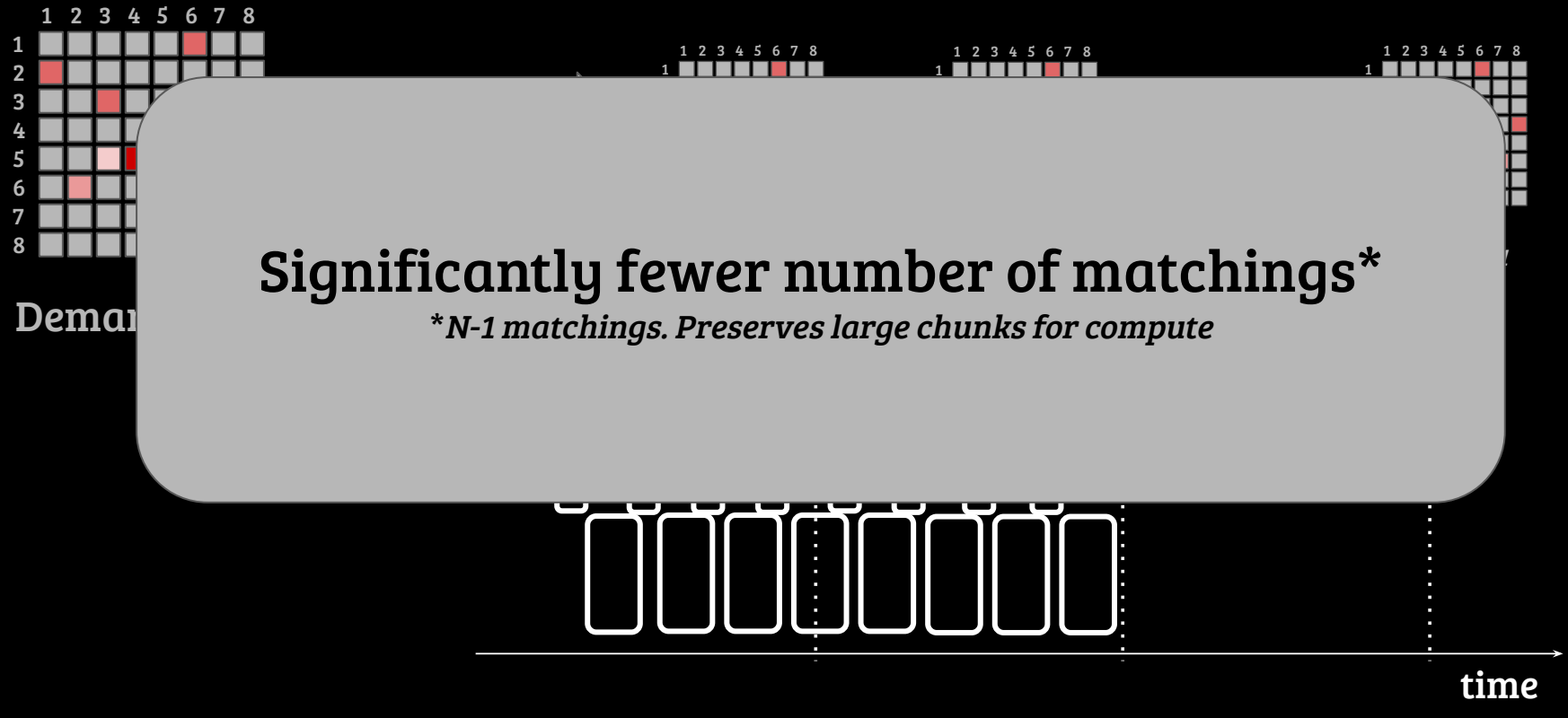


BvN



Maxweight

Putting it All Together: Efficient Compute on Large Batch

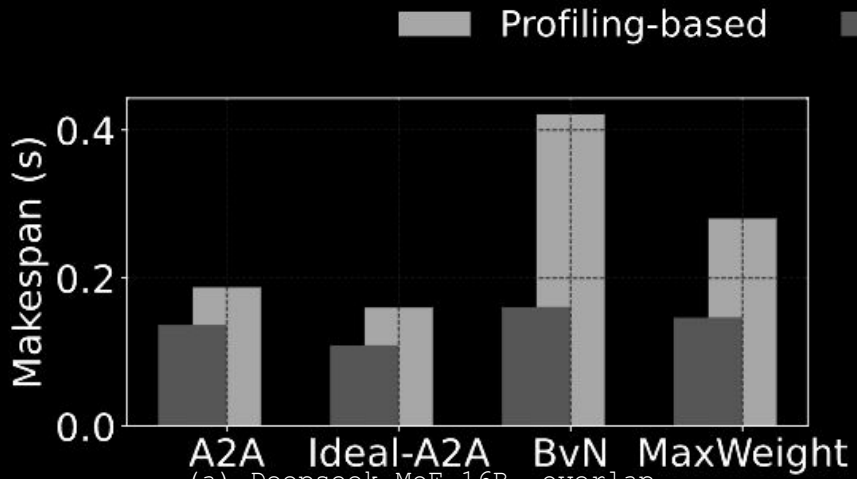


Evaluation

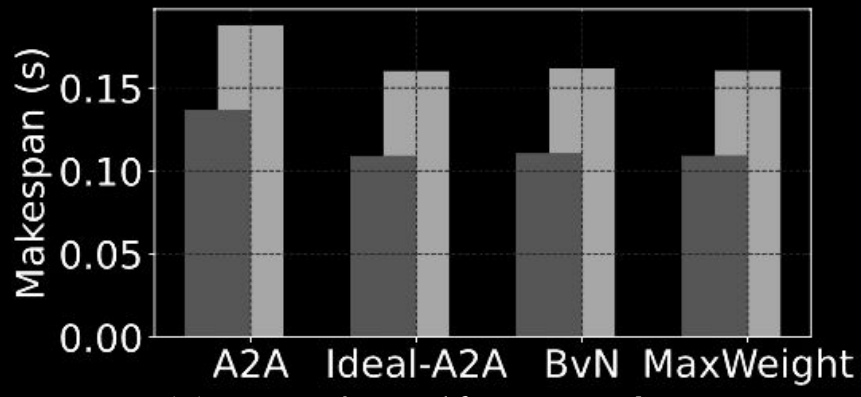
- Real MoE traffic matrices
- 8 RTX 6000 PRO
- Sequential vs Overlap
- Ideal All-to-all (congestion free solved in Gurobi)

Together: Severe Compute Inefficiency

- MMLU Dataset: Naturally small prompts



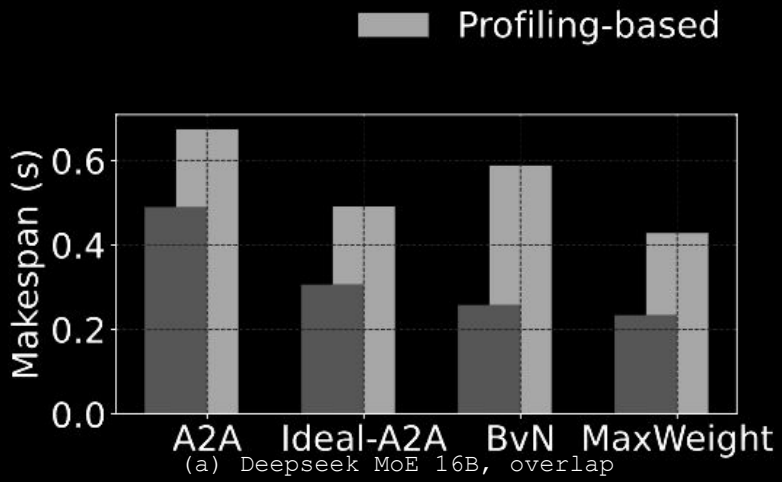
(a) Deepseek MoE 16B, overlap



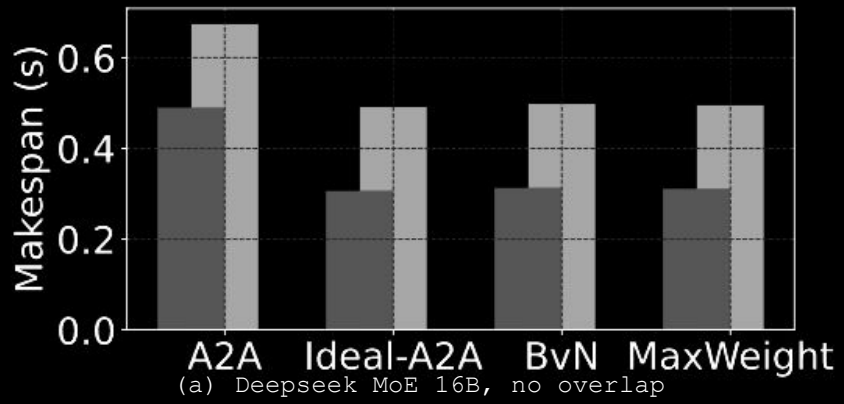
(a) Deepseek MoE 16B, no overlap

Together: Efficient Compute on Large Batch

- SpeedBench Dataset: Naturally large prompts (~2K tokens per prompt)



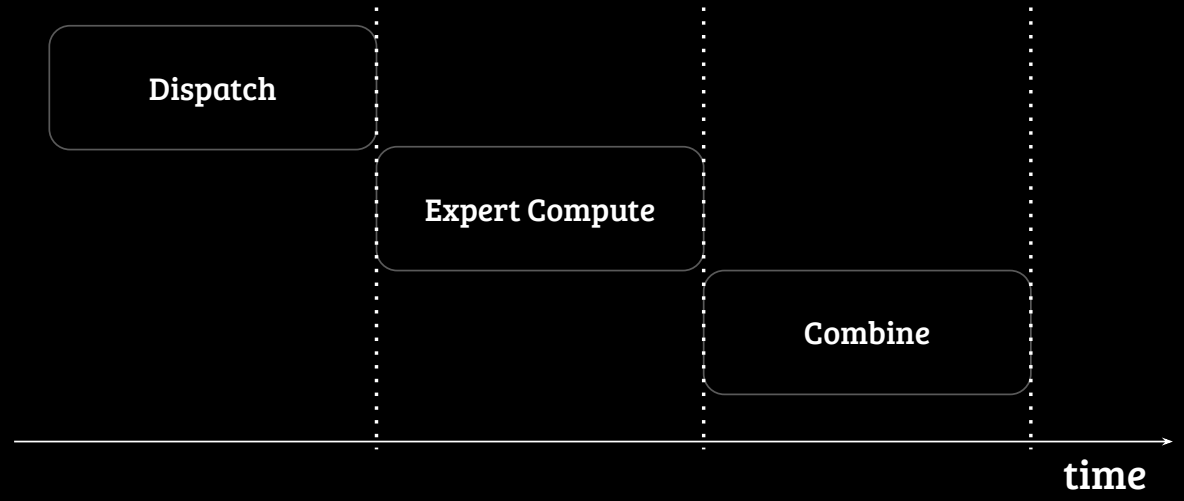
(a) Deepseek MoE 16B, overlap



(a) Deepseek MoE 16B, no overlap

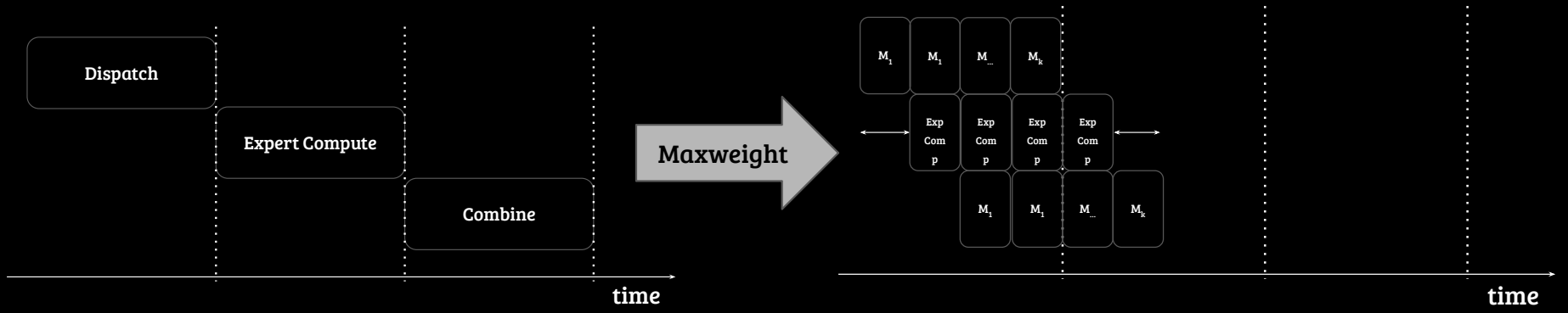
Takeaway: Jointly Optimize Network and Compute

- Makespan vs Completion Time
  - Data dependency
  - Dispatch-Compute-Combine



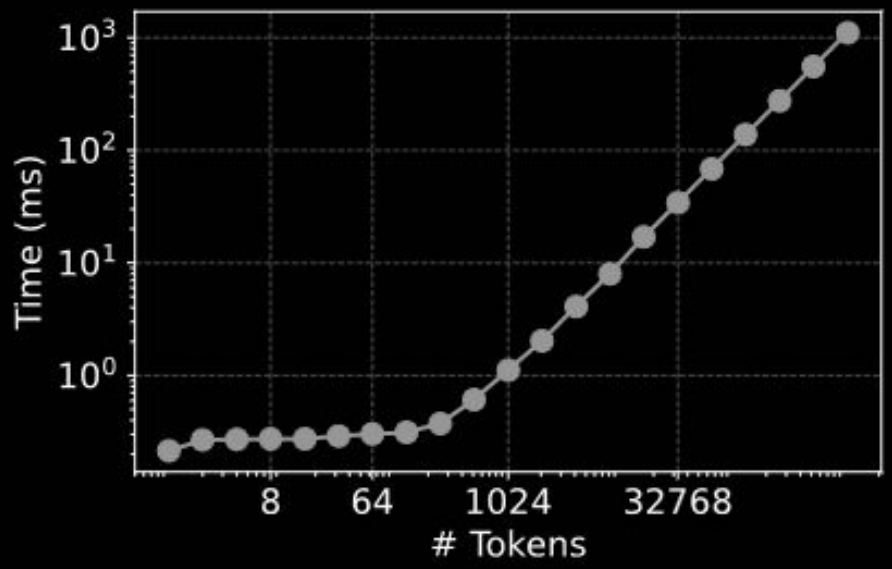
Takeaway: Overlap Opportunities

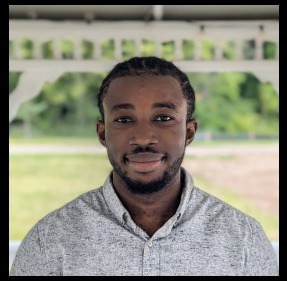
- Matrix decomposition strategies in OCS naturally provides overlap
  - More matchings, more overlap opportunities



Finally: Decomposition Quality

- Not just number of matchings
  - Structure of decomposition
  - Transmission Imbalance
  - Compute Fragmentation





Eliezer Amponsah
eampons@purdue.edu

Vamsi Addanki
vaddank@purdue.edu

Thank You