Hierarchical Coded Matrix Multiplication

Shahrzad Kiani, Nuwan Ferdinand, Stark C. Draper

Department of Electrical and Computer Engineering,
University of Toronto, Toronto, ON, Canada

The international HPC summer school 2019

Distributed matrix multiplication

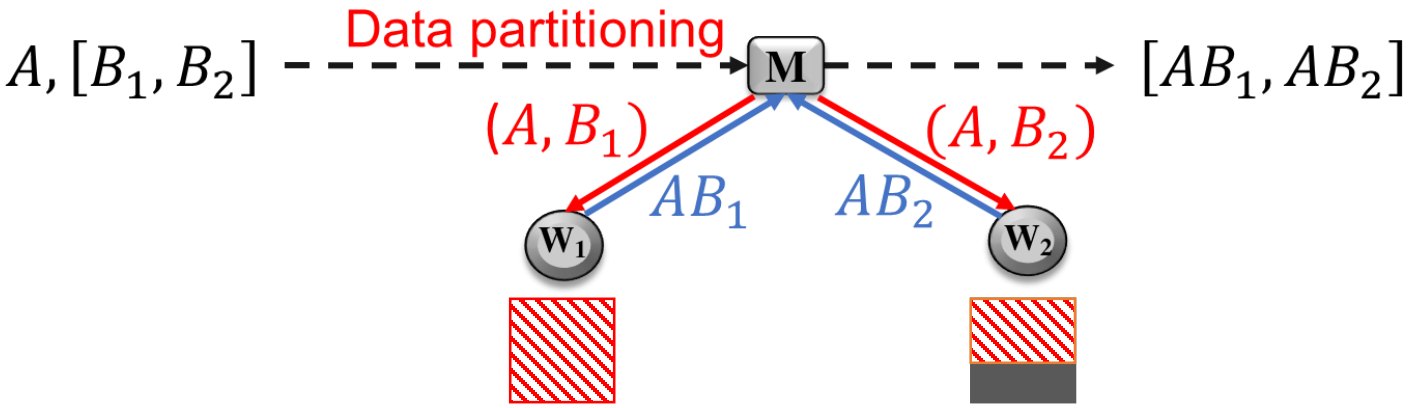
GOAL: Parallelize large-scale matrix multiplication $C = AB$ in a distributed system (A master and N workers)

Data partitioning at the master:

$$\text{If N} = 2, \quad (A, B) \rightarrow \{(A, B_1), (A, B_2)\} \text{ where } B = [B_1, B_2].$$

Distributed computation at workers: AB_1 and AB_2

Result concatenation at the master: $C = AB = [AB_1, AB_2]$



Challenge: Existence of stragglers (unpredictably slow nodes) which are observed in cloud computing systems, such as Amazon EC2

Solution: Distributed coded matrix multiplication

Coding to develop robustness to stragglers

Data partitioning and encoding at the master:

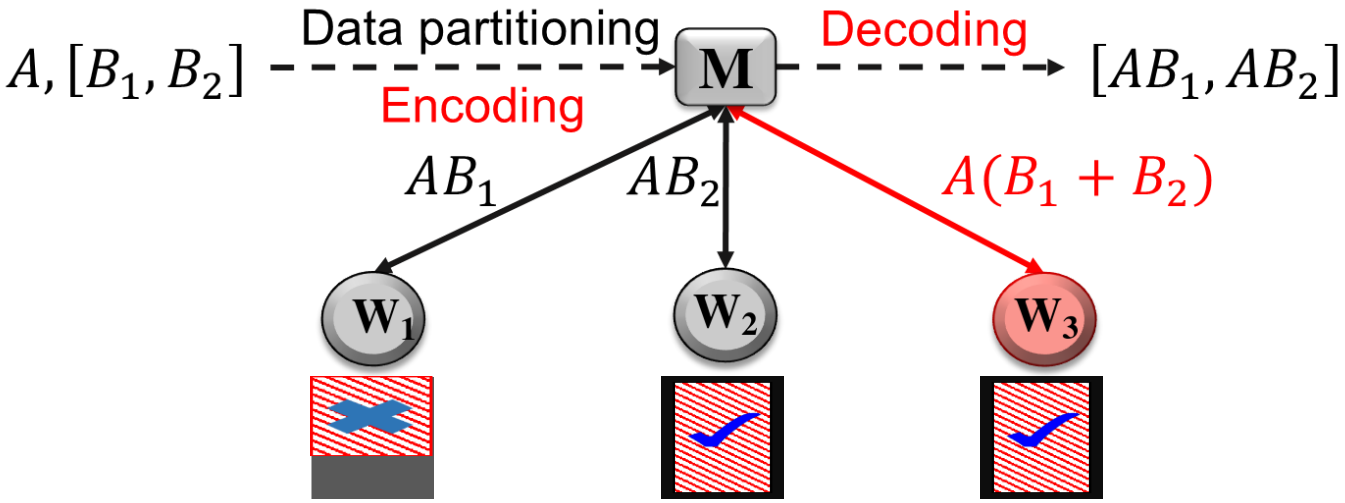
Idea: Encode a "parity" computation.

$$(A, B) \rightarrow \{(A, B_1), (A, B_2), (A, B_1 + B_2)\}$$

Distributed computation at workers: $AB_1,\ AB_2, A(B_1 + B_2)$

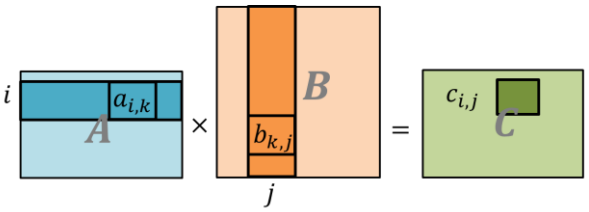
Concatenating and decoding results at the master:

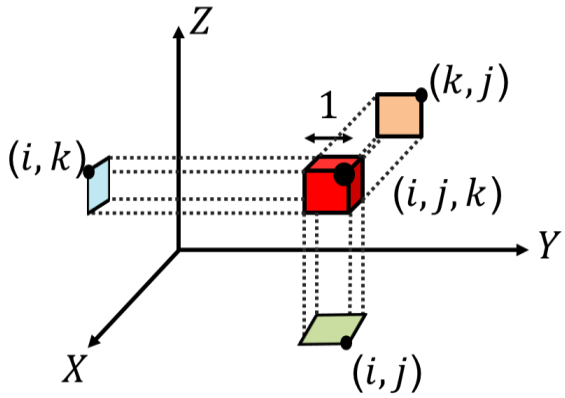
Idea: Any two completed workers can recover AB.

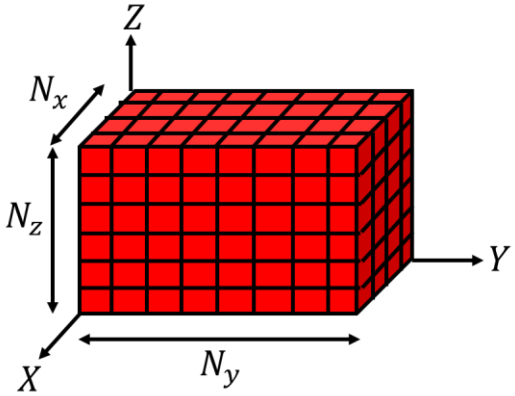


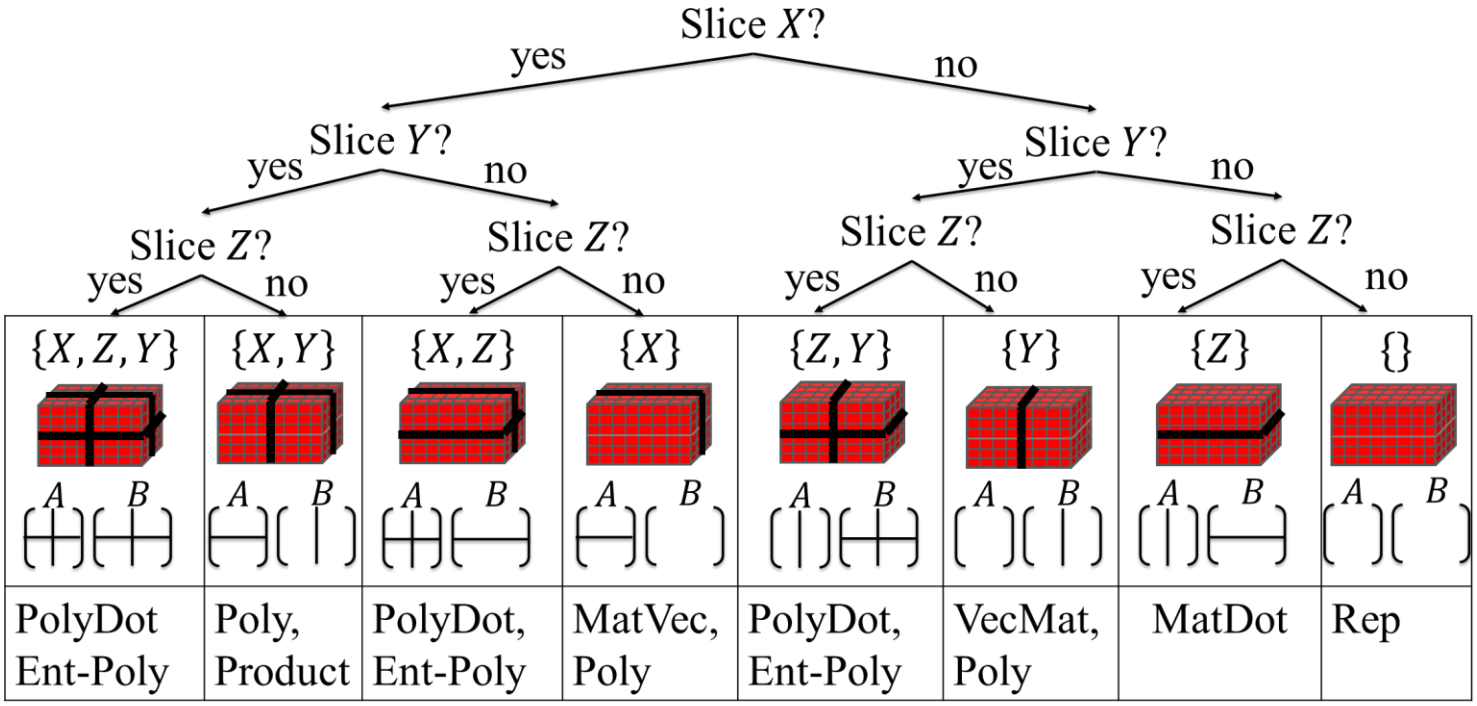
Challenge: Throw away half-completed work by worker 1; No ability to exploit partial work from stragglers.

A rectangular cuboid of multiply-and-accumulate operations









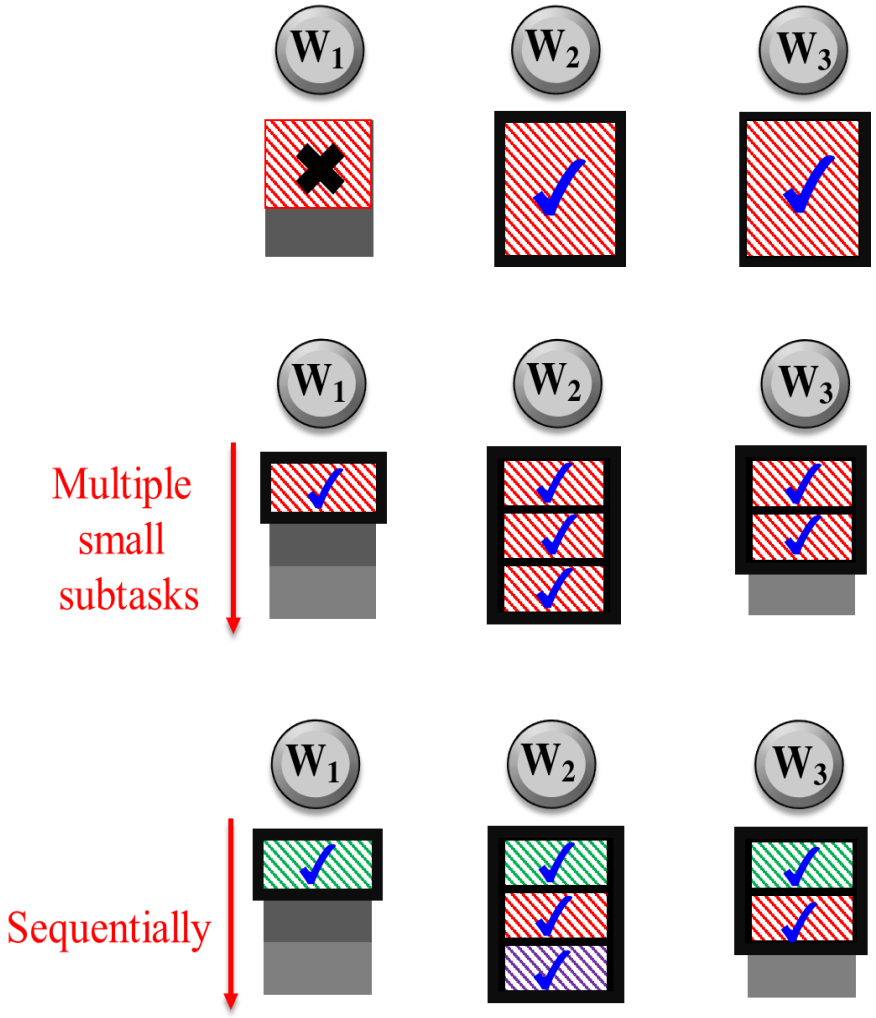
Novel strategies to exploit stragglers

Standard approach: Throw away half-completed work.

Idea 1: BICC [Kiani et al. ISIT'18] (i) Split into smaller subtasks, and (ii) apply one (long!) code across all possible computations.

the master can decode from any 6 completed subtasks of all 9.

Idea 2: MLCC [Kiani et al. CODML workshop, ICML'19] (i) Split into smaller subtasks, (ii) collect into layers of subtasks, and (iii) code with each layer.



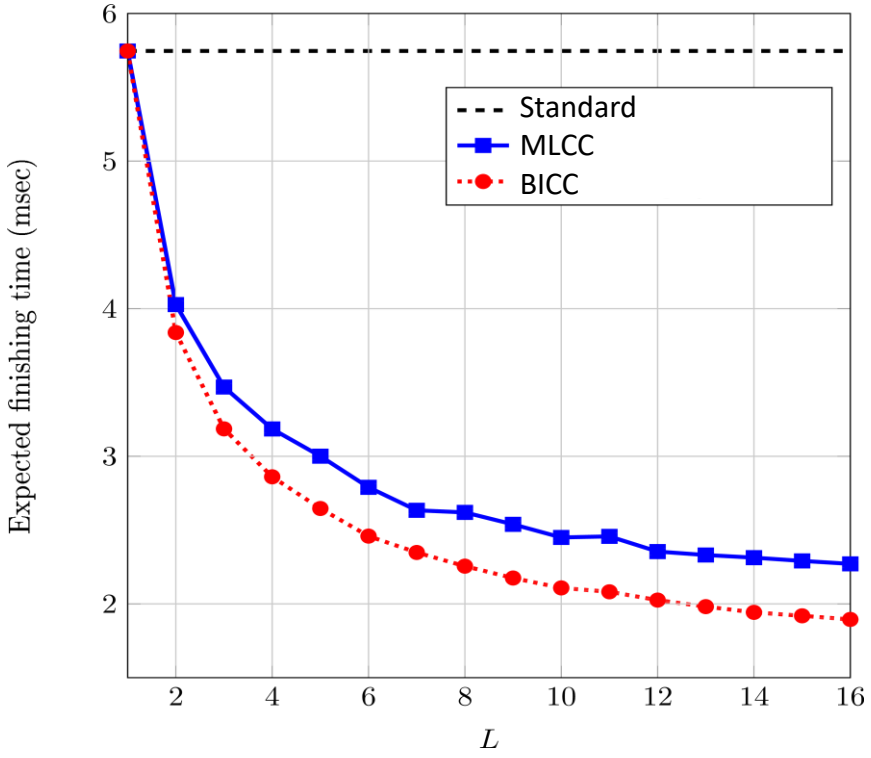
Results and conclusion

MLCC and BICC exploits work done by all workers, including stragglers.

MLCC and BICC realizes more than 60% improvement in expected finishing time compared to standard approach.

MLCC has lower decoding time compared to BICC. BICC has lower computation time compared to MLCC.

Shifted-exponential distribution, N=200



EC2 experiment, $N_x = N_z = N_y = 1000$