IEEE International Symposium on Information Theory, Los Angeles (virtual) 2020

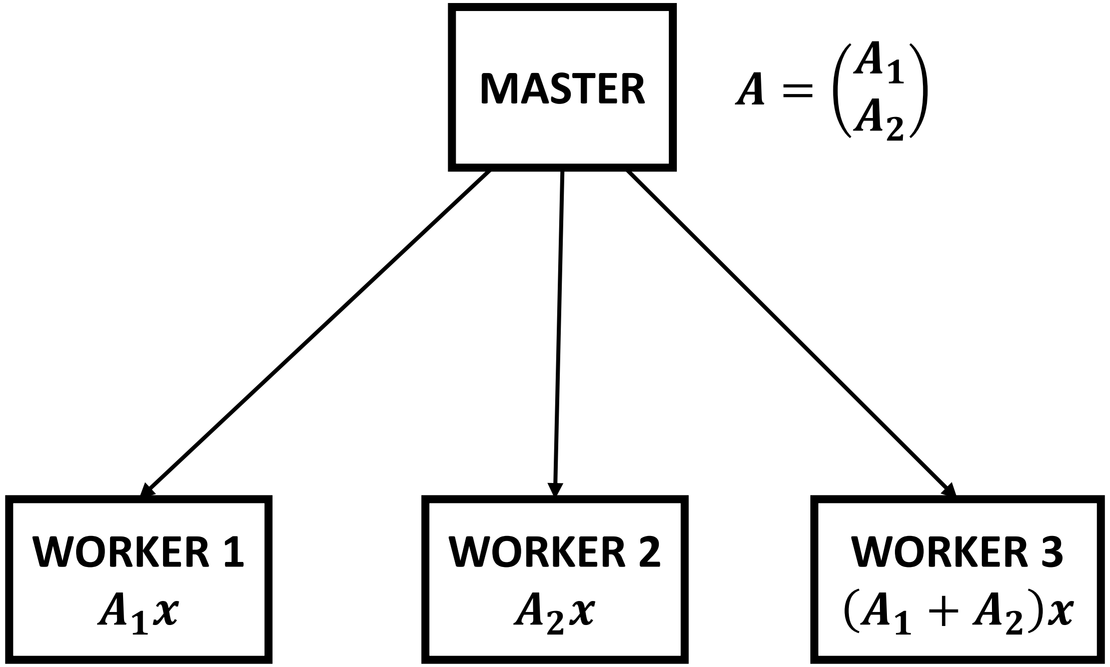
Optimizing the Transition Waste in Coded Elastic Computing

Hoang DAU

RMIT University (Melbourne)

Joint work with Ryan Gabrys, Yu-Chih Huang, Chen Feng, Quang-Hung Luu, Eidah Alzahrani, and Zahir Tari

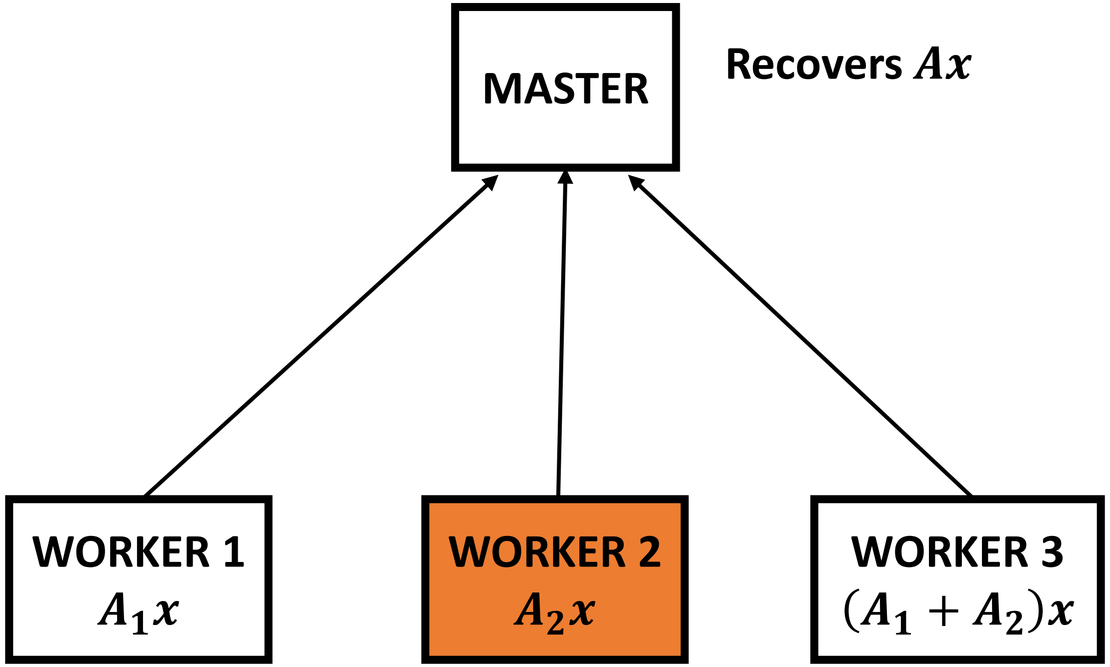
Coded Computing Toy Example



$$A = \begin{pmatrix} A_1 \\ A_2 \end{pmatrix}$$

Google

PageRank



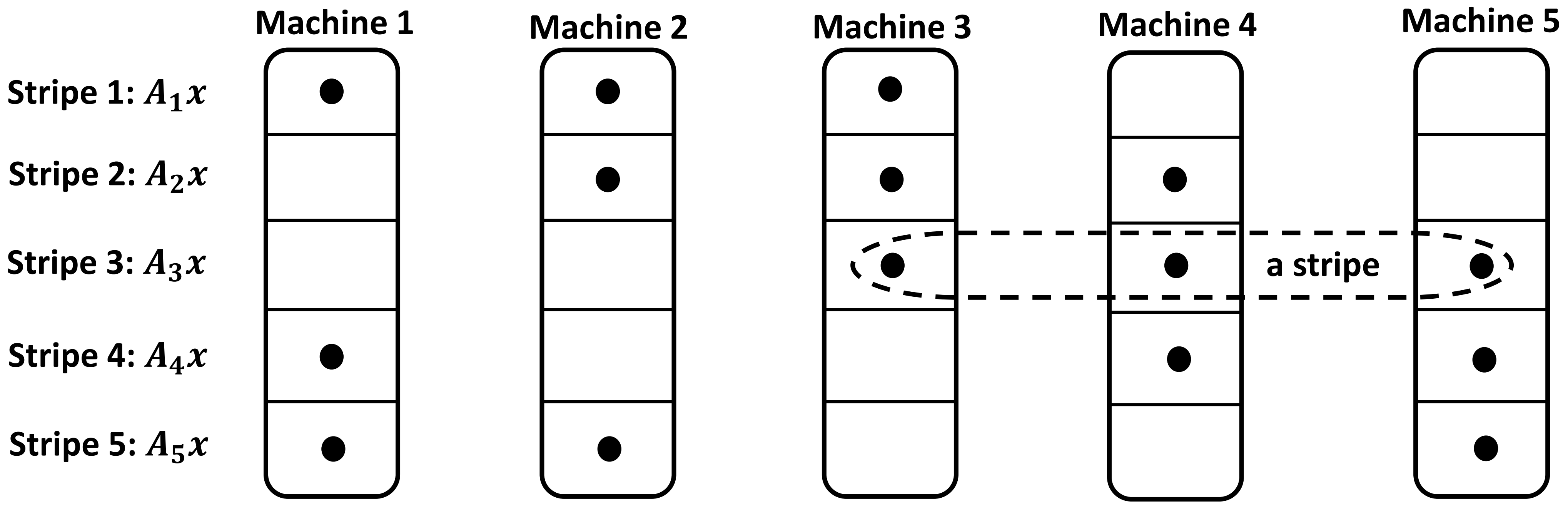
Recovers Ax

STRAGGLER

The master can recover $\boldsymbol{Ax}$ with one straggler

Coded Elastic Computing

- $[L = 3, K = 2]$ coded computing scheme (CCS) in this example
- If there are more than three machines, must we change the CCS? **NO**: **Striping**



Y. Yang et al., "Coded elastic computing", ISIT'19

Coded Elastic Computing

	Machine 1	Machine 2	Machine 3	Machine 4	Machine 5
Task 1: A_1x	$A_{1,1}x$	$(A_{1,1} + A_{1,2})x$	$(A_{1,1} + 2A_{1,2})x$	$(A_{1,1} + 3A_{1,2})x$	$(A_{1,1} + 4A_{1,2})x$
Task 2: A_2x	$A_{2,1}x$	$(A_{2,1} + A_{2,2})x$	$(A_{2,1} + 2A_{2,2})x$	$(A_{2,1} + 3A_{2,2})x$	$(A_{2,1} + 4A_{2,2})x$
Task 3: A_3x	$A_{3,1}x$	$(A_{3,1} + A_{3,2})x$	$(A_{3,1} + 2A_{3,2})x$	$(A_{3,1} + 3A_{3,2})x$	$(A_{3,1} + 4A_{3,2})x$
Task 4: A_4x	$A_{4,1}x$	$(A_{4,1} + A_{4,2})x$	$(A_{4,1} + 2A_{4,2})x$	$(A_{4,1} + 3A_{4,2})x$	$(A_{4,1} + 4A_{4,2})x$
Task 5: A_5x	$A_{5,1}x$	$(A_{5,1} + A_{5,2})x$	$(A_{5,1} + 2A_{5,2})x$	$(A_{5,1} + 3A_{5,2})x$	$(A_{5,1} + 4A_{5,2})x$

- Machines store all “tasks”
- Machines only execute shaded tasks according to a Task Allocation Scheme (TAS)
- When a machine leaves the system, all tasks (completed/uncompleted) on that machine are lost
- When a machine leaves the system, the set of tasks allocated to existing machines must be updated

Coded Elastic Computing

- Why do we consider a variable number of machines?
  - Industries are offering low-priority virtual machines at a reduced price (up to 90% down): Amazon EC2 Spot, Microsoft Azure Batch
  - Low priority: e.g., a Spot instance can be pre-empted on short notice (2 minutes)
  - E.g., we rent 7 VMs = 3 on-demand + 4 Spot, to speed up our computation at low cost, need to make our task assignment scheme ready for $N = 3,4,5,6,7$ machines
- Referred to as coded "elastic" computing (Yang et al. ISIT'2019): elasticity means the number of machines varies
- Another follow-up work: extends the cyclic task assignment to machines with varying computational speeds

N. Woolsey, R-R. Chen, and M. Ji, "*Heterogeneous Computation Assignments in Coded Elastic Computing*", 2020, https://arxiv.org/abs/2001.04005

Coded Elastic Computing

- In general, we divide $\boldsymbol{A}$ into F equal-sized submatrices $\boldsymbol{A}_0, \ldots, \boldsymbol{A}_{F-1}$
- The f-th *task/stripe* corresponds to the computation of $\boldsymbol{A}_f \boldsymbol{x}$ $(0 \leq f \leq F-1)$
- Any $[L, K]$ CCS can be used for each task f
- To assign tasks/stripes to machines, we need a Task Assignment Scheme (TAS)

Definition: An $[N, L, F]$-TAS is a list of sets $\mathcal{S}^N = (S_1^N, S_2^N, \ldots, S_N^N)$, where

- N = # machines
- $S_n^N \subseteq [[F]] := \{0,1, \ldots, F-1\}$ is the set of task indices assigned to Machine n
- L-redundancy: each index in $[[F]]$ lies in L sets in $\mathcal{S}^N$ (each task is computed by L machines)
- Load Balancing: $|S_n^N| = \frac{LF}{N}$ (each machine computes $\frac{LF}{N}$ tasks)

Coded Elastic Computing

- $F = 20$ tasks (we have $A_1, A_2, \dots, A_{20}$), $N = 5$ machines, $L = 3$

S_1^5	S_2^5	S_3^5	S_4^5	S_5^5
$0 \rightarrow 3$			$0 \rightarrow 3$	$0 \rightarrow 3$
$4 \rightarrow 7$	$4 \rightarrow 7$			$4 \rightarrow 7$
$8 \rightarrow 11$	$8 \rightarrow 11$	$8 \rightarrow 11$		
	$12 \rightarrow 15$	$12 \rightarrow 15$	$12 \rightarrow 15$	
		$16 \rightarrow 19$	$16 \rightarrow 19$	$16 \rightarrow 19$

Cyclic Task Assignment when there are $N = 5$ machines (Yang *et al.* ISIT'19)

Coded Elastic Computing

- $F = 20$ tasks, $N = 4$ machines, $L = 3$

S_1^4	S_2^4	S_3^4	S_4^4
$0 \rightarrow 4$		$0 \rightarrow 4$	$0 \rightarrow 4$
$5 \rightarrow 9$	$5 \rightarrow 9$		$5 \rightarrow 9$
$10 \rightarrow 14$	$10 \rightarrow 14$	$10 \rightarrow 14$	
	$15 \rightarrow 19$	$15 \rightarrow 19$	$15 \rightarrow 19$

Cyclic Task Assignment when there are $N = 4$ machines (Yang *et al.* ISIT'19)

- When $N = 3 = L$, each machine computes ALL tasks

Transition Waste in Coded Elastic Computing

- Transition: when machines are removed or added
- Transition waste: Re-assigning tasks causes waste in computation

Definition

- TW at Machine n: W_n = #abandoned tasks + #new tasks - #necessary change

Example

- Machine 3 was working on tasks $\{8 \rightarrow 19\}$,
- After the removal of Machine $n^* = 5$, it now works on tasks $\{0 \rightarrow 4\} \cup \{10 \rightarrow 19\}$
- It must abandon two tasks $\{8, 9\}$ and takes over five new tasks $\{0 \rightarrow 4\}$ (symmetric difference)
- The transition waste at Machine 3: W_3 = 2 + 5 - 3 = 4 tasks (3 is the necessary increase in the #tasks each machine takes as Machine 5 leaves, so not counted as waste)

Transition Waste in Coded Elastic Computing

- TW of a transition $\mathcal{S}^N \rightarrow \mathcal{S}^{N-1}$ or $\mathcal{S}^N \rightarrow \mathcal{S}^{N+1}$ is the sum of TW at each machine
- $W(\mathcal{S}^5 \rightarrow \mathcal{S}^4) = (0+3-3)+(1+4-3)+(2+5-3)+(3+6-3) = 12$

S_1^5	S_2^5	S_3^5	S_4^5	S_5^5
$0 \to 3$			$0 \to 3$	$0 \to 3$
$4 \to 7$	$4 \to 7$			$4 \to 7$
$8 \to 11$	$8 \to 11$	$8 \to 11$		
	$12 \to 15$	$12 \to 15$	$12 \to 15$	
		$16 \to 19$	$16 \to 19$	$16 \to 19$

Transition $\mathcal{S}^5 \rightarrow \mathcal{S}^4$

S_1^4	S_2^4	S_3^4	S_4^4
$0 \to 4$		$0 \to 4$	$0 \to 4$
$5 \to 9$	$5 \to 9$		$5 \to 9$
$10 \to 14$	$10 \to 14$	$10 \to 14$	
	$15 \to 19$	$15 \to 19$	$15 \to 19$

Transition Waste in Coded Elastic Computing

Our 1st contributions

- Explicit formulas for transition wastes in the cyclic TAS proposed in Yang *et al.*'19

Theorem 1. *The transition waste when transitioning from a cyclic (N, L, F)-TAS $\mathcal{S}^{N}_{\text{cyc}}$ to a cyclic $(N+1, L, F)$-TAS $\mathcal{S}^{N+1}_{\text{cyc}}$ (defined in (2)) is given below (assuming $N > L$).*

$$W(\mathcal{S}^{N}_{\text{cyc}} \to \mathcal{S}^{N+1}_{\text{cyc}}) = \frac{N-1}{N+1}F.$$

Theorem 2. *The transition waste when Machine $n^* \in [N]$ leaves and the system transitions from a cyclic (N, L, F)-TAS $\mathcal{S}^{N}_{\text{cyc}}$ to a cyclic $(N-1, L, F)$-TAS $\mathcal{S}^{N-1}_{\text{cyc}}$ (defined in (2)) is given as follows (assuming $N > L+1$).*

If $n^* < N-L$, $W_{n^*}(\mathcal{S}^{N}_{\text{cyc}} \to \mathcal{S}^{N-1}_{\text{cyc}})$ is

$$\left((n^*-1)(n^*-2)+(N-L-n^*)(N-L-n^*+1)\right)\frac{F}{N(N-1)}.$$

If $n^* \geq N-L$, $W_{n^*}(\mathcal{S}^{N}_{\text{cyc}} \to \mathcal{S}^{N-1}_{\text{cyc}})$ is

$$(n^*-1)(n^*-2)\frac{F}{N(N-1)}.$$

Transition Waste in Coded Elastic Computing

Our 2nd contributions

- Propose "shifted" cyclic TAS, which reduces the transition waste significantly

Ordinary cyclic TAS
Start from index 0

S_1^4	S_2^4	S_3^4	S_4^4
$0 \to 4$		$0 \to 4$	$0 \to 4$
$5 \to 9$	$5 \to 9$		$5 \to 9$
$10 \to 14$	$10 \to 14$	$10 \to 14$	
	$15 \to 19$	$15 \to 19$	$15 \to 19$

Shifted cyclic TAS
Start from index i

S_1^4	S_2^4	S_3^4	S_4^4
$17 \to 1$		$17 \to 1$	$17 \to 1$
$2 \to 6$	$2 \to 6$		$2 \to 6$
$7 \to 11$	$7 \to 11$	$7 \to 11$	
	$12 \to 16$	$12 \to 16$	$12 \to 16$

Transition Waste in Coded Elastic Computing

Our 2nd contributions

- Propose "shifted" cyclic TAS, which reduces the transition waste significantly
- Identify the shifted cyclic TAS with a minimal TW (under a divisibility condition)

Theorem 3. *The transition waste when transitioning from a δ'-shifted cyclic (N,L,F)-TAS $\mathcal{S}^{N}_{\delta'\text{-cyc}}$ to a δ-shifted cyclic $(N+1,L,F)$-TAS $\mathcal{S}^{N+1}_{\delta\text{-cyc}}$ with $\delta=\delta'+\lfloor\frac{N+L-1}{2}\rfloor\frac{F}{N(N+1)}$ is*

$$W(\mathcal{S}^{N}_{\delta'\text{-cyc}}\to\mathcal{S}^{N+1}_{\delta\text{-cyc}})=\begin{cases}\frac{(N-L-1)(N-L+1)F}{2N(N+1)}, & \text{for odd } N-L,\\ \frac{(N-L)^2F}{2N(N+1)}, & \text{for even } N-L.\end{cases}$$

Theorem 4. *The transition waste when transitioning from a δ'-shifted cyclic (N,L,F)-TAS $\mathcal{S}^{N}_{\delta'\text{-cyc}}$ to a δ-shifted cyclic $(N-1,L,F)$-TAS $\mathcal{S}^{N-1}_{\delta\text{-cyc}}$ with $\delta=\delta'+(N-n^*)-\lfloor\frac{(N+L-2)}{2}\rfloor\frac{F}{N(N-1)}$, where Machine n^* leaves, is*

$$W(\mathcal{S}^{N}_{\delta'\text{-cyc}}\to\mathcal{S}^{N-1}_{\delta\text{-cyc}})=\begin{cases}\frac{(N-L-1)^2F}{2N(N-1)}, & \text{for odd } N-L,\\ \frac{(N-L)(N-L-2)F}{2N(N-1)}, & \text{for even } N-L.\end{cases}$$

Transition Waste in Coded Elastic Computing

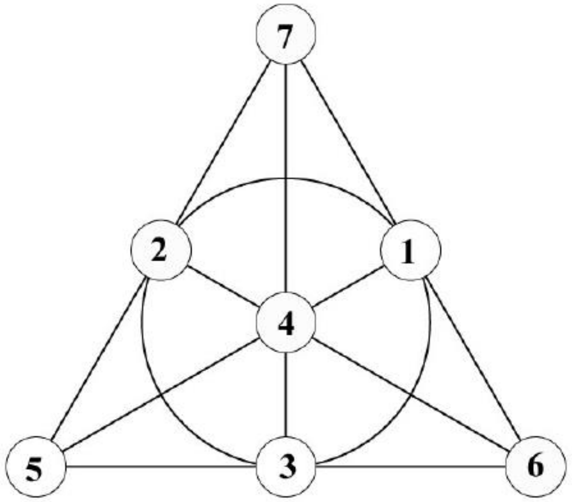
Our 3rd contributions

- How about zero-waste transitions?
- Theorem 6: $\exists$ a zero-waste transition $\Leftrightarrow$ $\exists$ a perfect matching in transition graph
- Intuition from Theorem 6: intersections of sets in the TAS should be small
- This led to the idea of using a configuration from combinatorial designs

Definition: A (v, k) (symmetric) configuration is an incident structure of v points and v lines such that

- Each line contains k points
- Each point lies on k lines
- Two different points lie on at most one line

Example: The Fano plane is a $(7, 7)$ configuration.



Transition Waste in Coded Elastic Computing

Our 3rd contributions

- How about zero-waste transitions?

S_1^7	S_2^7	S_3^7	S_4^7	S_5^7	S_6^7	S_7^7
$\{0,1\}$	$\{0,1\}$	$\{0,1\}$				
$\{2,3\}$			$\{2,3\}$	$\{2,3\}$		
$\{4,5\}$					$\{4,5\}$	$\{4,5\}$
	$\{6,7\}$		$\{6,7\}$			$\{6,7\}$
	$\{8,9\}$			$\{8,9\}$	$\{8,9\}$	
		$\{10,11\}$	$\{10,11\}$		$\{10,11\}$	
		$\{12,13\}$		$\{12,13\}$		$\{12,13\}$

Example: using Fano plane, we can construct a TAS with 7 machines, which allows zero-waste transitions within 5, 6, and 7 machines.

Transition Waste in Coded Elastic Computing

Our 3rd contributions

- **Theorem 7**: $\exists$ an $(N_{\max}, L)$ configuration $\Longrightarrow$ $\exists$ zero-waste transitions between $N_{\min}$ and $N_{\max}$ machines, where $N_{\min} \approx \frac{N_{\max}}{2}$ from standard constructions of configurations $\left(L \approx \sqrt{N_{\max}}\right)$

Open problem: does there exist zero-waste transitions for arbitrary $L, N_{\max}, N_{\min}$?

Open problem: transition waste taking into account the sets of completed tasks?

Open problem: min-max transition waste?