Coded Computation over Heterogeneous Clusters
Amirhossein Reisizadeh, Saurav Prakash, Ramtin Pedarsani, Salman Avestimehr

Distributed Computing Systems

- Various types of server instances available
- Latency variability leads to stragglers [Dean et al., The tail at scale]

Homogeneous Clusters

- Use MDS codes to deal with stragglers

Heterogeneous Clusters

Two-step Alternative Optimization:

\[ P^{TLM}_2: T^{(2)}(\tau) = \arg \max \mathbb{E}[T_2] \]

\[ \text{subject to } P[X^{(2)}(\tau) < \tau] = \omega(1) \]

\[ X(\tau) = \sum_{j=1}^{\tau} X(j) = A_1 X(\tau, c) \]

- The alternative formulation is tractable!

We propose the Heterogeneous Coded Matrix Multiplication (HCMM) algorithm which is the solution to the alternative optimization.

\[ \lim_{n \to \infty} \mathbb{E}[T_{\text{HCMM}}] = \lim_{n \to \infty} \mathbb{E}[T_{\text{opt}}] = \tau^* = \Theta(1) \]

Further, compared to the computation time of the optimal uncoded algorithm, we get a speedup of \( \Theta(\log(n)) \)

Cost Optimization

Goal:

- Optimal load allocation under limited budget

Problem formulation: Minimize the expected runtime while satisfying the budget constraint \( C \):

\[ \text{Maximize } \mathbb{E}[T_{TLP}] \]

subject to \( \sum_{i=1}^{k} 1_{\{c_i > 0\}} \mathbb{E}[T_{TLP}] \leq C \)

This is an Integer Programming and is hard to solve

Model:

For a machine with processing rate \( \mu \) we assume that the cost per unit time is

\[ c = \kappa \mu^{\alpha}, \quad \alpha \geq 1 \]

Numerical Results

- We have 2 types of machines with rates 2 and 4, and 10 machines of each type.
- Maximum budget is \( C = 860 \)

Heuristic Algorithm

Lemma: Under such cost model and a set of machine types, the minimum cost is induced by using only the slowest machines.

Heuristic Algorithm for cost optimization:

- Run the HCMM allocation scheme for all machines.
- If the cost is higher than budget, decrease one of the fastest used machines.
- Repeat until budget constraint is satisfied.

Complexity of brute force search: \( O(\prod_{i=1}^{d} n_i) \)

Complexity of our algorithm: \( O(n) \)

References