Avoiding Communication in Machine Learning

Aditya Devarakonda, Kimon Fountoulakis, James Demmel and Michael Mahoney

Motivation

- Least Squares (LS) is an important technique used in data mining and big data analytics for prediction.
- Existing parallel SDCA algorithms communicate at every iteration.
- We present a communication-avoiding SDCA algorithm which communicates every s iterations with the same convergence rate, in exact arithmetic.

Cost Model

Algorithms have two costs: Computation (FLOPS) and Communication (moving data).
- Sequential: Between levels of memory hierarchy.
- Parallel: Between processors over a network.

Communication is the dominant cost.

Stochastic Dual Coordinate Ascent

1: Input: $H > 1$, $x^{(0)} \in \mathbb{R}^n$
2: Data: $\{(x_i, y_i)\}_{i=1}^n$
3: Initialize: $w^{(0)} = \frac{1}{\sqrt{n}} X x^{(0)}$
4: for $h = 1, 2, \cdots, H$ do
5: choose $i \in \{1, 2, \cdots, n\}$ uniformly at random
6: $\Delta x_i = e_i^T r^{(h-1)}$
7: $x^{(h)} = x^{(h-1)} + \frac{\Delta x_i}{\lambda}$
8: $w^{(h)} = w^{(h-1)} + \frac{\Delta x_i e_i}{\lambda}$
9: $r^{(h)} = (I + \gamma_i A e_i e_i^T) r^{(h-1)}$
10: Output $x^{(H)}$ and $w^{(H)}$

- Implementation and extension to L1-Regularization.
- Reorganize other iterative machine learning and convex optimization algorithms.
- Reorganize Newton-type algorithms.

Communication-avoiding SDCA

We can use the recursion for the residual to build a Krylov-like basis for the step sizes:

$$\Lambda_{\phi} = \left[ e_i^T(I + \gamma_i A e_i e_i^T) r^{(s)} e_i^T(I + \gamma_i A e_i e_i^T) r^{(s)}, \cdots, e_i^T(I + \gamma_i A e_i e_i^T) r^{(s)} e_i^T(I + \gamma_i A e_i e_i^T) r^{(s)} \right]$$

- Choosing $s$ indices to update gives us a polynomial expansion of terms. Dependency graph of the coefficients is given below.

Ongoing Work

- Communication cost of parallel SDCA

<table>
<thead>
<tr>
<th>Computation</th>
<th>Latency</th>
<th>Bandwidth</th>
</tr>
</thead>
<tbody>
<tr>
<td>$O \left( \frac{H h s}{P} \right)$</td>
<td>$O \left( \frac{H \log P}{h s} \right)$</td>
<td>$O \left( \frac{H h s \log P}{P} \right)$</td>
</tr>
</tbody>
</table>

- Communication cost of parallel CA-SDDCA with 1D row partitioning

<table>
<thead>
<tr>
<th>Computation</th>
<th>Latency</th>
<th>Bandwidth</th>
</tr>
</thead>
<tbody>
<tr>
<td>$O \left( \frac{H h s}{P} \right)$</td>
<td>$O \left( \frac{H \log P}{h s} \right)$</td>
<td>$O \left( \frac{H h s \log P}{P} \right)$</td>
</tr>
</tbody>
</table>

- Communication cost of parallel CA-SDDCA with 1D column partitioning

<table>
<thead>
<tr>
<th>Computation</th>
<th>Latency</th>
<th>Bandwidth</th>
</tr>
</thead>
<tbody>
<tr>
<td>$O \left( \frac{H h s}{P} \right)$</td>
<td>$O \left( \frac{H \log P}{h s} \right)$</td>
<td>$O \left( \frac{H h s \log P}{P} \right)$</td>
</tr>
</tbody>
</table>

Time = $\alpha \times (\# messages) + \beta \times (\# words moved) + \gamma \times (\# flops)$