Communication lower bounds for numerical tensor algebra

Edgar Solomonik

ETH Zurich

DMML Workshop
Oct 24, 2015

Symmetry in tensor contractions

Consider a contraction from the CCSD method

$$Z^{a\bar{k}}_{i\bar{c}} = \sum_b \sum_j T^{ab}_{ij} \cdot V^{j\bar{k}}_{b\bar{c}}$$

where **T** is partially antisymmetric

$$T^{ab}_{ij} = -T^{ba}_{ij} = -T^{ab}_{ji} = T^{ba}_{ji}$$

When the tensors have dimensions $n \times n \times n \times n$, this contraction usually requires $2n^6$ total operations (to leading order).

Despite the symmetry in **T**, no scalar multiplications are equivalent.

Symmetric-matrix–vector multiplication

- Consider symmetric $n \times n$ matrix $\mathbf{A}$ and vectors $\mathbf{b}, \mathbf{c}$
- $\mathbf{c} = \mathbf{A} \cdot \mathbf{b}$ is usually done by computing a *nonsymmetric* intermediate matrix $\mathbf{W}$,

$$W_{ij} = A_{ij} \cdot b_j \qquad\qquad c_i = \sum_{j=1}^{n} W_{ij}$$

  which requires n^2 multiplications and n^2 additions
- The *symmetry preserving algorithm* employs a *symmetric* intermediate matrix $\mathbf{Z}$,

$$Z_{ij} = A_{ij} \cdot (b_i + b_j) \qquad\qquad c_i = \sum_{j=1}^{n} Z_{ij} - \left(\sum_{j=1}^{n} A_{ij} \right) \cdot b_i$$

  which requires $\frac{n^2}{2}$ multiplications and $\frac{5n^2}{2}$ additions

Symmetrized rank-two outer product

- Consider vectors $\mathbf{a}, \mathbf{b}$ of dimension n
- Symmetric matrix $\mathbf{C} = \mathbf{a} \cdot \mathbf{b}^\mathsf{T} + \mathbf{b} \cdot \mathbf{a}^\mathsf{T}$ is usually done by computing a *nonsymmetric* intermediate matrix $\mathbf{W}$,

$$W_{ij} = a_i \cdot b_j \qquad\qquad C_{ij} = W_{ij} + W_{ji}$$

  which requires n^2 multiplications and $n^2/2$ additions
- The *symmetry preserving algorithm* employs a *symmetric* intermediate matrix $\mathbf{Z}$,

$$Z_{ij} = (a_i + a_j) \cdot (b_i + b_j) \qquad C_{ij} = Z_{ij} - a_i \cdot b_i - a_j \cdot b_j$$

  which requires $\frac{n^2}{2}$ multiplications and $2n^2$ additions

Symmetrized matrix multiplication

- Consider symmetric $n \times n$ matrices $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$
- $\mathbf{C} = \mathbf{A} \cdot \mathbf{B} + \mathbf{B} \cdot \mathbf{A}$ is usually computed via a nonsymmetric intermediate order 3 tensor $\mathbf{W}$,

$$W_{ijk} = A_{ik} \cdot B_{kj} \qquad \bar{W}_{ij} = \sum_k W_{ijk} \qquad C_{ij} = W_{ij} + W_{ji}.$$

  which requires n^3 multiplications and n^3 additions.
- The *symmetry preserving algorithm* employs a *symmetric* intermediate tensor $\mathbf{Z}$ using $n^3/6$ multiplications and $7n^3/6$ additions,

$$Z_{ijk} = (A_{ij} + A_{ik} + A_{jk}) \cdot (B_{ij} + B_{ik} + B_{jk}) \qquad v_i = \sum_{k=1}^{n} A_{ik} \cdot B_{ik}$$

$$C_{ij} = \sum_{k=1}^{n} Z_{ijk} - n \cdot A_{ij} \cdot B_{ij} - v_i - v_j - \left(\sum_{k=1}^{n} A_{ik} \right) \cdot B_{ij} - A_{ij} \cdot \left(\sum_{k=1}^{n} B_{ik} \right)$$

Symmetry preserving algorithm

Consider contraction of symmetric tensors **A** of order $s+v$ and **B** of order $v+t$ that is symmetrized to produce a symmetric tensor **C** of order $s+t$

- Let $\omega = s+t+v$
- Let $\Upsilon^{(s,t,v)}$ be the nonsymmetric contraction algorithm
- Let $\Psi^{(s,t,v)}$ be the direct evaluation algorithm
- Let $\Phi^{(s,t,v)}$ be the symmetry preserving algorithm

ω	s	t	v	F_Υ	F_Ψ	F_Φ	application cases
2	1	1	0	n^2	n^2	$n^2/2$	syr2, her2, (syr2k, her2k)
2	1	0	1	n^2	n^2	$n^2/2$	symv, hemv, (symm, hemm)
3	1	1	1	n^3	n^3	$n^3/6$	matrix (anti)commutator
s+t+v	s	t	v	n^ω	$\binom{n}{s}\binom{n}{t}\binom{n}{v}$	$\binom{n}{\omega}$	generally

Antisymmetry and matrix powers

The symmetry preserving algorithm can compute

- symmetrized products of two symmetric or two antisymmetric tensors
- antisymmetrized products of a symmetric and an antisymmetric tensor
- Hermitian tensor contractions
- $\mathbf{A}^2$ for symmetric or antisymmetric $\mathbf{A}$ with $n^3/6$ multiplications
- $\mathbf{A}^2$ for nonsymmetric $\mathbf{A}$ (or $\mathbf{A} \cdot \mathbf{B} + \mathbf{B} \cdot \mathbf{A}$ for nonsymmetric $\mathbf{A}$, $\mathbf{B}$) with $2n^3/3$ products
- that CCSD contraction

$$Z^{a\bar{k}}_{i\bar{c}} = \sum_b \sum_j T^{ab}_{ij} \cdot V^{j\bar{k}}_{b\bar{c}}$$

in n^6 operations (2X fewer) via $\Phi^{(1,0,1)} \otimes \Upsilon^{(1,2,1)}$

Bilinear algorithms

A bilinear algorithm is defined by three matrices $\mathbf{F}^{(\mathbf{A})}$, $\mathbf{F}^{(\mathbf{B})}$, $\mathbf{F}^{(\mathbf{C})}$
Given input vectors $\mathbf{a}$ and $\mathbf{b}$, it computes vector

$$\mathbf{c} = \mathbf{F}^{(\mathbf{C})}[(\mathbf{F}^{(\mathbf{A})T}\mathbf{a}) \circ (\mathbf{F}^{(\mathbf{B})T}\mathbf{b})],$$

where $\circ$ is the Hadamard (pointwise) product

- the number of columns in the three matrices is equal and is the *bilinear algorithm rank*
- the number of rows in each matrix corresponds to the number of inputs (dimensions of $\mathbf{a}$ and $\mathbf{b}$) and outputs (dimension of $\mathbf{c}$)
- matrix multiplication and symmetric tensor contraction correspond to different bilinear algorithms (problems)
- the bilinear rank is the number of multiplications, for the symmetry preserving algorithm, it is $\binom{n}{\omega}$

Manipulation of bilinear algorithms

Given two bilinear algorithms:

$$\begin{aligned}\Lambda_1 =& (\mathbf{F_1^{(A)}}, \mathbf{F_1^{(B)}}, \mathbf{F_1^{(C)}})\\ \Lambda_2 =& (\mathbf{F_2^{(A)}}, \mathbf{F_2^{(B)}}, \mathbf{F_2^{(C)}})\\ \Lambda_1 \otimes \Lambda_2 :=& (\mathbf{F_1^{(A)}} \otimes \mathbf{F_2^{(A)}}, \mathbf{F_1^{(B)}} \otimes \mathbf{F_2^{(B)}}, \mathbf{F_1^{(C)}} \otimes \mathbf{F_2^{(C)}})\\ \mathrm{rank}(\Lambda_1 \otimes \Lambda_2) =& \mathrm{rank}(\Lambda_1) \cdot \mathrm{rank}(\Lambda_2)\end{aligned}$$

Conversely given $\Lambda = (\mathbf{F^{(A)}}, \mathbf{F^{(B)}}, \mathbf{F^{(C)}})$, we say $\Lambda_{\mathrm{sub}} \subseteq \Lambda$ if there exists projection matrix $\mathbf{P}$ such that

$$\Lambda_{\mathrm{sub}} = (\mathbf{F^{(A)}}\mathbf{P}, \mathbf{F^{(B)}}\mathbf{P}, \mathbf{F^{(C)}}\mathbf{P})$$

Expansion in bilinear algorithms

A bilinear algorithm Λ has expansion bound $\mathcal{E}_\Lambda : \mathbb{N}^3 \to \mathbb{N}$, if for all

$$\Lambda_{\mathrm{sub}} := (\mathbf{F}^{(\mathbf{A})}_{\mathrm{sub}}, \mathbf{F}^{(\mathbf{B})}_{\mathrm{sub}}, \mathbf{F}^{(\mathbf{C})}_{\mathrm{sub}}) \subseteq \Lambda$$

we have

$$\mathrm{rank}(\Lambda_{\mathrm{sub}}) \leq \mathcal{E}_\Lambda \left(\mathrm{rank}(\mathbf{F}^{(\mathbf{A})}_{\mathrm{sub}}), \mathrm{rank}(\mathbf{F}^{(\mathbf{B})}_{\mathrm{sub}}), \mathrm{rank}(\mathbf{F}^{(\mathbf{C})}_{\mathrm{sub}}) \right)$$

Vertical communication in bilinear algorithms

Any schedule on a sequential machine with a cache of size H for $\Lambda = (\mathbf{F^{(A)}}, \mathbf{F^{(B)}}, \mathbf{F^{(C)}})$ with expansion bound $\mathcal{E}_\Lambda$ has vertical communication cost

$$Q_\Lambda \geq \max\left[\frac{2\,\mathrm{rank}(\Lambda)H}{\mathcal{E}_\Lambda^{\max}(H)}, \#\mathrm{rows}(\mathbf{F^{(A)}}) + \#\mathrm{rows}(\mathbf{F^{(B)}}) + \#\mathrm{rows}(\mathbf{F^{(C)}})\right]$$

where $\mathcal{E}_\Lambda^{\max}(H) := \max_{c^{(A)},c^{(B)},c^{(C)}\in\mathbb{N},\, c^{(A)}+c^{(B)}+c^{(C)}=3H} \mathcal{E}_\Lambda(c^{(A)}, c^{(B)}, c^{(C)})$

Vertical communication in matrix multiplication

For the classical (non-Strassen-like) matrix multiplication algorithm of m-by-k matrix **A** with k-by-n matrix **B** into m-by-n matrix C

$$\mathcal{E}_{\mathsf{MM}}(c^{(A)}, c^{(B)}, c^{(C)}) = (c^{(A)}c^{(B)}c^{(C)})^{1/2}$$

further, we have

$$\mathcal{E}^{\max}_{\mathrm{MM}}(H) = \max_{c^{(A)},c^{(B)},c^{(C)}\in\mathbb{N},c^{(A)}+c^{(B)}+c^{(C)}\leq 3H} (c^{(A)}c^{(B)}c^{(C)})^{1/2} = H^{3/2}$$

so we obtain the expected bound

$$\begin{aligned} Q_{\mathrm{MM}} &\geq \max\left[\frac{2\,\mathsf{rank}(\mathrm{MM})H}{\mathcal{E}^{\max}_{\mathrm{MM}}(H)}, \#\mathrm{rows}(\mathbf{F}^{(\mathbf{A})}) + \#\mathrm{rows}(\mathbf{F}^{(\mathbf{B})}) + \#\mathrm{rows}(\mathbf{F}^{(\mathbf{C})})\right] \\ &= \max\left[\frac{2mnk}{\sqrt{H}}, mk + kn + mn\right] \end{aligned}$$

Horizontal communication in bilinear algorithms

Any load balanced schedule on a parallel machine with p processes of $\Lambda = (\mathbf{F^{(A)}}, \mathbf{F^{(B)}}, \mathbf{F^{(C)}})$ with expansion bound $\mathcal{E}_\Lambda$ has horizontal communication cost

$$W_\Lambda \geq d^{(A)} + d^{(B)} + d^{(C)}$$

for some $d^{(A)}, d^{(B)}, d^{(C)} \in \mathbb{N}$ such that

$$\begin{aligned} \mathrm{rank}(\Lambda)/p \leq \mathcal{E}_\Lambda(&d^{(A)} + \#\mathrm{rows}(\mathbf{F^{(A)}})/p, \\ &d^{(B)} + \#\mathrm{rows}(\mathbf{F^{(B)}})/p, \\ &d^{(C)} + \#\mathrm{rows}(\mathbf{F^{(C)}})/p) \end{aligned}$$

Horizontal communication in matrix multiplication

For the classical (non-Strassen-like) matrix multiplication algorithm of m-by-k matrix **A** with k-by-n matrix **B** into m-by-n matrix **C** on a parallel machine of p processors

$$W_{\mathrm{MM}} = \Omega\left(W_{\mathrm{O}}(\min(m,n,k), \mathrm{median}(m,n,k), \max(m,n,k), p)\right)$$

where

$$W_{\mathrm{O}}(x,y,z,p) = \begin{cases} \left(\frac{xyz}{p}\right)^{2/3} & : p > yz/x^2 \\ x\left(\frac{yz}{p}\right)^{1/2} & : yz/x^2 \geq p > z/y \\ xy & : z/y \geq p \end{cases}$$

Communication lower bounds for direct evaluation of symmetric contractions

An expansion bound on $\Psi^{(s,t,v)}$ is

$$\mathcal{E}_{\Psi}^{(s,t,v)}(d^{(A)}, d^{(B)}, d^{(C)}) = q\left(d^{(A)} d^{(B)} d^{(C)}\right)^{1/2},$$

where $q = \left[\binom{s+v}{s}\binom{v+t}{v}\binom{s+t}{s}\right]^{1/2}$

Therefore, the same (asymptotically) horizontal and vertical communication lower bounds apply for $\Psi^{(s,t,v)}$ as for a matrix multiplication with dimensions $n^s \times n^t \times n^v$

Communication lower bounds for direct evaluation of symmetric contractions

Another expansion bound on $\Psi^{(s,t,0)}$ (when $v = 0$) is

$$\mathcal{E}_{\Psi}^{(s,t,0)}(d^{(A)}, d^{(B)}, d^{(C)}) = \left(\binom{\omega}{s} - 1\right) d^{(C)} + \min\left((d^{(A)})^{\omega/s}, (d^{(B)})^{\omega/t}, d^{(C)}\right)$$

There are also symmetric bounds when $s = 0$ or $t = 0$

When exactly one of s, t, v is zero, any load balanced schedule of $\Psi^{(s,t,v)}$ on a parallel machine with p processors has horizontal communication cost

$$W_{\Psi} = \Omega\left((n^{\omega}/p)^{\max(s,t,v)/\omega}\right)$$

This can be stronger than the corresponding matrix-multiplication-like bound

$$W_{\Psi} = \Omega\left((n^{\omega}/p)^{1/2}\right)$$

Communication lower bounds for the symmetry preserving algorithm

An expansion bound on $\Phi^{(s,t,v)}$ is

$$\mathcal{E}_{\Phi}^{(s,t,v)}(d^{(A)}, d^{(B)}, d^{(C)}) = \min\left(\left(\binom{\omega}{t} d^{(A)}\right)^{\frac{\omega}{s+v}}, \left(\binom{\omega}{s} d^{(B)}\right)^{\frac{\omega}{v+t}}, \left(\binom{\omega}{v} d^{(C)}\right)^{\frac{\omega}{s+t}} \right)$$

This yields communication bounds with $\kappa := \max(s+v, v+t, s+t)$

$$Q_{\Phi} = \Omega\left(\frac{n^{\omega} H}{H^{\omega/\kappa}} + n^{\kappa}\right) \qquad W_{\Phi} = \begin{cases} \Omega\left((n^{\omega}/p)^{\kappa/\omega}\right) & : s,t,v > 0 \\ \Omega\left((n^{\omega}/p)^{\max(s,t,v)/\omega}\right) & : \kappa = \omega \end{cases}$$

Communication lower bounds for nested algorithms

Conjecture: if bilinear algorithms λ_1 and λ_2 have expansion bounds $\mathcal{E}_1$ and $\mathcal{E}_2$, then $\lambda_1 \otimes \lambda_2$ has expansion bound $\mathcal{E}_{12}(c^{(A)}, c^{(B)}, c^{(C)})$

$$= \max_{\substack{c_1^{(A)}, c_1^{(B)}, c_1^{(C)}, c_2^{(A)}, c_2^{(B)}, c_2^{(C)} \in \mathbb{N} \\ c_1^{(A)} c_2^{(A)} = c^{(A)}, c_1^{(B)} c_2^{(B)} = c^{(B)}, c_1^{(C)} c_2^{(C)} = c^{(C)}}} \left[\mathcal{E}_1(c_1^{(A)}, c_1^{(B)}, c_1^{(C)}) \mathcal{E}_2(c_2^{(A)}, c_2^{(B)}, c_2^{(C)}) \right]$$

Simpler conjecture: consider matrices **A** and **B**, such that for some $\alpha, \beta \in [0, 1]$ and any $k \in \mathbb{N}$

- any subset of k columns of **A** has rank at least k^α
- any subset of k columns of **B** has rank at least k^β

then any subset of $k \in \mathbb{N}$ columns of $\mathbf{A} \otimes \mathbf{B}$ has rank at least $k^{\min(\alpha,\beta)}$

The first conjecture would provide lower bounds for the nested algorithms we wish to use for partially-symmetric coupled-cluster contractions

Dependencies between bilinear forms

Consider the Gaussian elimination algorithm computing $\mathbf{A} = \mathbf{LU}$

- it must compute the bilinear algorithm corresponding the matrix multiplication $\mathbf{LU}$
- therefore, it has the same bilinear expansion bound and communication lower bounds as matrix multiplication
- but not all bilinear forms may be computed simultaneously
- a dependency DAG may be defined where the vertices are the bilinear forms
- this DAG defines a partial ordering on the bilinear forms

Dependency interval analysis

Consider a bilinear algorithm that computes a set of bilinear forms V with a partial ordering, we denote a dependency interval between $a, b \in V$ as

$$[a, b] = \{a, b\} \cup \{c : a < c < b, c \in V\}$$

If there exists $\{v_1, \ldots, v_n\} \in V$ with $v_i < v_{i+1}$ and $|[v_{i+1}, v_{i+k}]| = k^d$ for all $k \in \mathbb{N}$, then

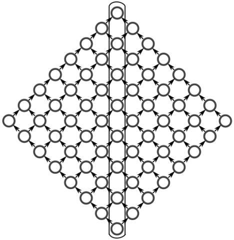
$$F \cdot S^{d-1} = \Omega(n^d)$$

where F is the computation cost and S is the synchronization cost

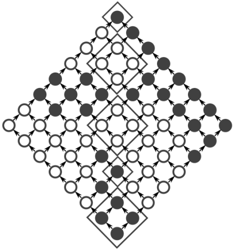
Further, if the algorithm has bilinear expansion $\mathcal{E}$, satisfying $\mathcal{E}^{\max}(H) = H^{\frac{d}{d-1}}$, then

$$W \cdot S^{d-2} = \Omega(n^{d-1})$$

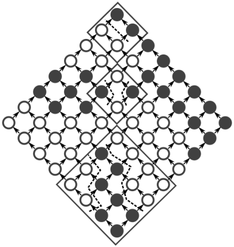
What just happened?



Dependency chain P



Monochrome dependency intervals



Multicolored dependency intervals

Idea goes back to Papadimitriou and Ullman, 1987

Synchronization lower bounds as tradeoffs

For triangular solve with an $n \times n$ matrix

$$F_{\mathrm{TRSV}} \cdot S_{\mathrm{TRSV}} = \Omega\left(n^2\right)$$

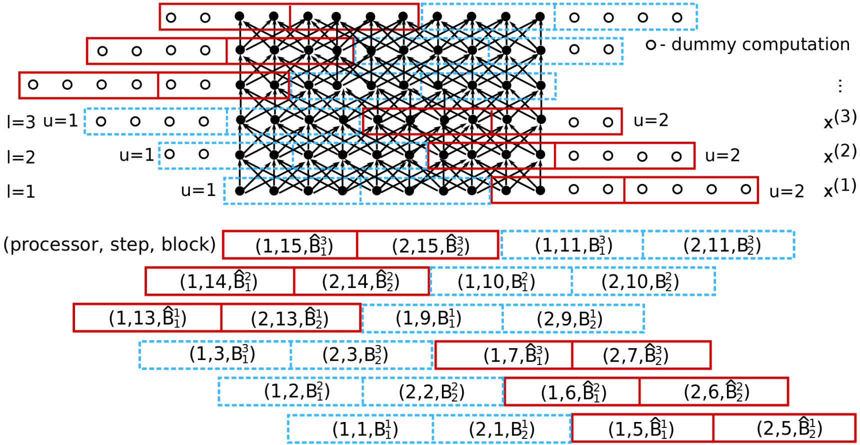
For Cholesky of an $n \times n$ matrix

$$F_{\mathrm{CHOL}} \cdot S_{\mathrm{CHOL}}^2 = \Omega\left(n^3\right) \qquad W_{\mathrm{CHOL}} \cdot S_{\mathrm{CHOL}} = \Omega\left(n^2\right)$$

For computing s applications of a $(2m+1)^d$-point stencil

$$F_{\mathrm{St}} \cdot S_{\mathrm{St}}^d = \Omega\left(m^{2d} \cdot s^{d+1}\right) \qquad W_{\mathrm{St}} \cdot S_{\mathrm{St}}^{d-1} = \Omega\left(m^d \cdot s^d\right)$$

What about memory bandwidth cost?



Its possible to lower memory bandwidth cost by $H^{1/d}$ without asymptotic increase in horizontal communication cost

Conclusion

- exploiting symmetry raises communication cost
- dense matrix factorizations cannot scale
- iterative solvers also cannot scale
- but there are also some good news...
- Happy Birthday Jim!

Self-references

For more information see

- ES and James Demmel; Contracting symmetric tensors using fewer multiplications
- ES, James Demmel, and Torsten Hoefler; Communication lower bounds for tensor contraction algorithms
- ES, Erin Carson, Nicholas Knight, and James Demmel; Tradeoffs between synchronization, communication, and work in parallel linear algebra computations

Backup slides

Symmetry preserving algorithm vs Strassen's algorithm