Symmetry in tensor contractions

Consider a contraction from the CCSD method

\[ Z_{i\bar{c}}^{a\bar{k}} = \sum_b \sum_j T_{ij}^{ab} \cdot V_{b\bar{c}}^{j\bar{k}} \]

where \( T \) is partially antisymmetric

\[ T_{ij}^{ab} = - T_{ij}^{ba} = - T_{ji}^{ab} = T_{ji}^{ba} \]

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 $A$ and vectors $b, c$
- $c = A \cdot b$ is usually done by computing a *nonsymmetric* intermediate matrix $W$,

\[
W_{ij} = A_{ij} \cdot b_j \\
c_i = \sum_{j=1}^{n} W_{ij}
\]

which requires $n^2$ multiplications and $n^2$ additions

- The *symmetry preserving algorithm* employs a *symmetric* intermediate matrix $Z$,

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

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 $\mathbf{C} = \mathbf{a} \cdot \mathbf{b}^T + \mathbf{b} \cdot \mathbf{a}^T$ is usually done by computing a *nonsymmetric* intermediate matrix $\mathbf{W}$,

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

  which requires $n^2$ multiplications and $n^2/2$ additions

- The *symmetry preserving algorithm* employs a *symmetric* intermediate matrix $\mathbf{Z}$,

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

  which requires $n^2/2$ multiplications and $2n^2$ additions
Symmetrized matrix multiplication

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

\[
W_{ijk} = A_{ik} \cdot B_{kj} \quad \tilde{W}_{ij} = \sum_k W_{ijk} \quad C_{ij} = W_{ij} + W_{ji}.
\]

which requires $n^3$ multiplications and $n^3$ additions.

- The symmetry preserving algorithm employs a symmetric intermediate tensor $Z$ using $n^3/6$ multiplications and $7n^3/6$ additions,

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

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

<table>
<thead>
<tr>
<th>$\omega$</th>
<th>$s$</th>
<th>$t$</th>
<th>$v$</th>
<th>$F_\Upsilon$</th>
<th>$F_\Psi$</th>
<th>$F_\Phi$</th>
<th>application cases</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>$n^2$</td>
<td>$n^2$</td>
<td>$n^2/2$</td>
<td>syr2, her2, (syr2k, her2k)</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>$n^2$</td>
<td>$n^2$</td>
<td>$n^2/2$</td>
<td>symv, hemv, (symm, hemm)</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>$n^3$</td>
<td>$n^3$</td>
<td>$n^3/6$</td>
<td>matrix (anti)commutator</td>
</tr>
<tr>
<td>$s+t+v$</td>
<td>$s$</td>
<td>$t$</td>
<td>$v$</td>
<td>$n^\omega$</td>
<td>$\binom{n}{s}$ (n) $\binom{n}{t}$ $\binom{n}{v}$</td>
<td>$\binom{n}{\omega}$</td>
<td>generally</td>
</tr>
</tbody>
</table>
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
- $A^2$ for symmetric or antisymmetric $A$ with $n^3/6$ multiplications
- $A^2$ for nonsymmetric $A$ (or $A \cdot B + B \cdot A$ for nonsymmetric $A$, $B$) with $2n^3/3$ products
- that CCSD contraction

$$Z_{i\bar{c}}^{a\bar{k}} = \sum_b \sum_j T_{ij}^{ab} \cdot V_{b\bar{c}}^{j\bar{k}}$$

in $n^6$ operations (2X fewer) via $\Phi^{(1,0,1)} \otimes \gamma^{(1,2,1)}$
A bilinear algorithm is defined by three matrices $F^{(A)}$, $F^{(B)}$, $F^{(C)}$

Given input vectors $a$ and $b$, it computes vector

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

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 $a$ and $b$) and outputs (dimension of $c$)
- 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:

\[ \Lambda_1 = (F^{(A)}_1, F^{(B)}_1, F^{(C)}_1) \]
\[ \Lambda_2 = (F^{(A)}_2, F^{(B)}_2, F^{(C)}_2) \]
\[ \Lambda_1 \otimes \Lambda_2 := (F^{(A)}_1 \otimes F^{(A)}_2, F^{(B)}_1 \otimes F^{(B)}_2, F^{(C)}_1 \otimes F^{(C)}_2) \]

\[ \text{rank}(\Lambda_1 \otimes \Lambda_2) = \text{rank}(\Lambda_1) \cdot \text{rank}(\Lambda_2) \]

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

\[ \Lambda_{\text{sub}} = (F^{(A)}P, F^{(B)}P, F^{(C)}P) \]
A bilinear algorithm $\Lambda$ has expansion bound $\mathcal{E}_\Lambda : \mathbb{N}^3 \rightarrow \mathbb{N}$, if for all 

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

we have

$$
\text{rank}(\Lambda_{\text{sub}}) \leq \mathcal{E}_\Lambda \left( \text{rank}(F^{(A)}_{\text{sub}}), \text{rank}(F^{(B)}_{\text{sub}}), \text{rank}(F^{(C)}_{\text{sub}}) \right)
$$
Any schedule on a sequential machine with a cache of size $H$ for
$\Lambda = (F(A), F(B), F(C))$ with expansion bound $E_{\Lambda}$ has vertical
communication cost

$$Q_{\Lambda} \geq \max \left[ \frac{2 \text{rank}(\Lambda) H}{E_{\Lambda}^{\text{max}}(H)}, \text{\#rows}(F(A)) + \text{\#rows}(F(B)) + \text{\#rows}(F(C)) \right]$$

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

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

further, we have

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

so we obtain the expected bound

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

$$
= \max \left[ \frac{2mkn}{\sqrt{H}}, mk + kn + mn \right]
$$
Horizontal communication in bilinear algorithms

Any load balanced schedule on a parallel machine with $p$ processes of
\( \Lambda = (F(A), F(B), F(C)) \) with expansion bound $E_\Lambda$ 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

\[
\frac{\text{rank}(\Lambda)}{p} \leq E_\Lambda \left( d^{(A)} + \frac{\# \text{rows}(F(A))}{p},
\; d^{(B)} + \frac{\# \text{rows}(F(B))}{p},
\; d^{(C)} + \frac{\# \text{rows}(F(C))}{p} \right)
\]
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

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

where

$$W_O(x, y, z, p) = \begin{cases} 
\left( \frac{xyz}{p} \right)^{2/3} & : p > \frac{yz}{x^2} \\
 x \left( \frac{yz}{p} \right)^{1/2} & : \frac{yz}{x^2} \geq p > \frac{z}{y} \\
 xy & : \frac{z}{y} \geq p 
\end{cases}$$
An expansion bound on $\Psi^{(s,t,v)}$ is

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

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

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

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

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

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

$$W_{\Psi} = \Omega \left(\left(\frac{n^\omega}{p}\right)^{1/2}\right)$$
Communication lower bounds for the symmetry preserving algorithm

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

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

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

$$Q_\Phi = \Omega \left( \frac{n^{\omega} H}{H^\omega / \kappa} + n^\kappa \right) \quad W_\Phi = \begin{cases} \Omega \left( \left( \frac{n^{\omega}}{p} \right)^{\kappa/\omega} \right) & : s, t, v > 0 \\ \Omega \left( \left( \frac{n^{\omega}}{p} \right)^{\max(s, t, v)/\omega} \right) & : \kappa = \omega \end{cases}$$
Conjecture: if bilinear algorithms $\lambda_1$ and $\lambda_2$ have expansion bounds $E_1$ and $E_2$, then $\lambda_1 \otimes \lambda_2$ has expansion bound $E_{12}(c^{(A)}, c^{(B)}, c^{(C)})$

$$
\begin{align*}
E_{12}(c^{(A)}, c^{(B)}, c^{(C)}) &= \max_{c_1^{(A)}, c_1^{(B)}, c_1^{(C)}, c_2^{(A)}, c_2^{(B)}, c_2^{(C)} \in \mathbb{N}, c_1^{(A)} c_2^{(A)} = c^{(A)}, c_1^{(B)} c_2^{(B)} = c^{(B)}, c_1^{(C)} c_2^{(C)} = c^{(C)}} \left[ E_1(c_1^{(A)}, c_1^{(B)}, c_1^{(C)}) E_2(c_2^{(A)}, c_2^{(B)}, c_2^{(C)}) \right]
\end{align*}
$$

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 $k^\alpha$
- any subset of $k$ columns of $B$ has rank at least $k^\beta$

then any subset of $k \in \mathbb{N}$ columns of $A \otimes B$ 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
Consider the Gaussian elimination algorithm computing $A = LU$

- it must compute the bilinear algorithm corresponding the matrix multiplication $LU$
- 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 $E$, satisfying

$$E_{\text{max}}(H) = H^d d^{-1},$$

then

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

Idea goes back to Papadimitriou and Ullman, 1987
Synchronization lower bounds as tradeoffs

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

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

For Cholesky of an $n \times n$ matrix

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

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

$$F_{\text{St}} \cdot S_{\text{St}}^d = \Omega \left( m^{2d} \cdot s^{d+1} \right) \quad W_{\text{St}} \cdot S_{\text{St}}^{d-1} = \Omega \left( m^d \cdot s^d \right)$$
What about memory bandwidth cost?

It's possible to lower memory bandwidth cost by $H_1^{1/d}$ without asymptotic increase in horizontal communication cost.
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!
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
Symmetry preserving algorithm vs Strassen’s algorithm

![Graph showing speed-up over classical direct evaluation alg. vs matrix dimension.

Strassen’s algorithm
Sym. preserving $\omega=6$
Sym. preserving $\omega=3$

(speed-up over classical direct evaluation alg.)

$\frac{[n^\omega/(s!t!v!)]}{\#\text{multiplications}}$

$n^S/s!$ (matrix dimension)