A partitioning problem for load balancing and reducing communication from the field of quantum chemistry

Edmond Chow
School of Computational Science and Engineering
Georgia Institute of Technology, USA

SIAM Conference on Parallel Processing for Scientific Computing
Paris, April 12-15, 2016
Setting for this talk

- Fock matrix construction in the Hartree-Fock method of quantum chemistry
- Distributed Fock matrix construction: partitioning problem to balance load and reduce communication

This talk will present:

- a simplified version of the problem in an abstract way
- one possible solution approach using hypergraph partitioning
- the realistic version of the problem
- a solution to the realistic problem, but the problem is still open!

Quantum chemistry: many applications, many parallel codes, much work on distributed Fock matrix construction.
Simplified abstract problem

The Fock matrix is partitioned by blocks, one block for each compute node.

A task is represented by a shell quartet \((MN|PQ)\), where \(M, N, P, Q\) are shell indices, \(M = 1, \ldots, n_{\text{shells}}\), etc.

The set of tasks is sparse.

Communication required by task \((MN|PQ)\): a task updates the following elements of the Fock matrix, called shell pairs: \((M, N), (P, Q), (M, P), (N, P), (M, Q), (N, Q)\). Note: the Fock matrix is symmetric.

Problem: find how the tasks should be assigned to nodes in a balanced way while minimizing communication.
Solution approach using hypergraph partitioning

Solution approach using hypergraph partitioning

(Assume a fixed partitioning of the Fock matrix.)
Solution approach using hypergraph partitioning

(Assume a fixed partitioning of the Fock matrix.)
Issues with the hypergraph approach

Realistic problems have too many shell quartets for hypergraph partitioning. (Some grouping of shell quartets is possible.)

Not possible to list all the shell quartets. Cannot represent an arbitrary list of shell quartets assigned to a node.

But, the approach can be used to help design other more efficient approaches.

Assuming the hypergraph approach gives a “best” partitioning, what is the communication cost, and what do the partitions look like?
Solution approaches utilizing geometric information and sparsity structure

Geometric information can be used to help reduce partitioning cost.

A shell is associated with a geometric coordinate (an atomic center).

A shell quartet has magnitude related by the integrals

\[(ij|kl) = \int \phi_i(x_1)\phi_j(x_1)r_{12}^{-1}\phi_k(x_2)\phi_l(x_2)dx_1 \, dx_2\]

where \(x_1\) and \(x_2\) are coordinates in \(\mathbb{R}^3\), and \(r_{12} = \|x_1 - x_2\|\).

Symmetry

\[(ij|kl) = (ji|kl) = (ij|lk) = (kl|ij) = \cdots\]

Schwarz Screening

\[(ij|kl) \leq \sqrt{(ij|ij)\sqrt{(kl|kl)}} \leq \tau\]
Numerically nonzero shell quartets

Basis functions are grouped into shells. A shell quartet is the set of integrals:

\[(MN|PQ) \equiv \{(ij|kl) \text{ s.t. } i \in \text{shell } M, j \in \text{shell } N, k \in \text{shell } P, l \in \text{shell } Q\}\]

System: \( \text{C}_{24}\text{H}_{50} \) with cc-pVDZ (294 shells, 586 functions)
Fock matrix structure

\[ F_{ij} = H_{ij}^{\text{core}} + \sum_{kl} D_{kl} (2(ij|kl) - (ik|jl)) \]

where \((ij|kl)\) denotes an element of a 4-index electron repulsion integral (ERI) tensor.
Coulomb and exchange matrices

\[
F_{ij} = H_{ij}^{\text{core}} + 2 \sum_{kl} D_{kl}(ij|kl) - \sum_{kl} D_{kl}(ik|jl)
\]

\[
F = H^{\text{core}} + 2J - K
\]

Suppose we have computed the shell quartet \((MN|PQ)\), then

\[
J_{MN} = \sum D_{PQ}(MN|PQ) \quad K_{NQ} = \sum D_{MP}(MN|PQ)
\]
\[
J_{NM} = \sum D_{PQ}(NM|PQ) \quad K_{MQ} = \sum D_{NP}(NM|PQ)
\]
\[
J_{MN} = \sum D_{QP}(MN|QP) \quad K_{NP} = \sum D_{MQ}(MN|QP)
\]
\[
J_{NM} = \sum D_{QP}(NM|QP) \quad K_{MP} = \sum D_{QM}(MN|QP)
\]
\[
J_{PQ} = \sum D_{MN}(PQ|MN) \quad K_{QN} = \sum D_{PM}(PQ|MN)
\]
\[
J_{QP} = \sum D_{MN}(QP|MN) \quad K_{PN} = \sum D_{QM}(QP|MN)
\]
\[
J_{PQ} = \sum D_{NM}(PQ|NM) \quad K_{QM} = \sum D_{PN}(PQ|NM)
\]
\[
J_{QP} = \sum D_{NM}(QP|NM) \quad K_{PM} = \sum D_{QN}(QP|NM)
\]
Coulomb and exchange matrices

\[
F_{ij} = H^{\text{core}}_{ij} + 2 \sum_{kl} D_{kl}(ij|kl) - \sum_{kl} D_{kl}(ik|jl)
\]

\[
F = H^{\text{core}} + 2J - K
\]

Suppose we have computed the \textit{shell quartet} \((MN|PQ)\), then

\[
J_{MN} = \sum D_{PQ}(MN|PQ)
\]
\[
J_{NM} = \sum D_{PQ}(NM|PQ)
\]
\[
J_{MN} = \sum D_{QP}(MN|QP)
\]
\[
J_{NM} = \sum D_{QP}(NM|QP)
\]
\[
J_{PQ} = \sum D_{MN}(PQ|MN)
\]
\[
J_{QP} = \sum D_{MN}(QP|MN)
\]
\[
J_{PQ} = \sum D_{NM}(PQ|NM)
\]
\[
J_{QP} = \sum D_{NM}(QP|NM)
\]

\[
K_{NQ} = \sum D_{MP}(MN|PQ)
\]
\[
K_{MQ} = \sum D_{NP}(NM|PQ)
\]
\[
K_{NP} = \sum D_{MQ}(MN|QP)
\]
\[
K_{MP} = \sum D_{NQ}(NM|QP)
\]
\[
K_{QN} = \sum D_{PM}(PQ|MN)
\]
\[
K_{PN} = \sum D_{QM}(QP|MN)
\]
\[
K_{QM} = \sum D_{PN}(PQ|NM)
\]
\[
K_{PM} = \sum D_{QN}(QP|NM)
\]
Distributed computation of the Fock matrix

$F$ and $D$ are distributed; each node stores a part of $F$ and $D$

1. for unique shell quartets $(MN|PQ)$ do
   2. if $(MN|PQ)$ is not screened out then
      3. Compute shell quartet $(MN|PQ)$
      4. Receive submatrices $D_{MN}$, $D_{PQ}$, $D_{NP}$, $D_{MQ}$, $D_{NQ}$, $D_{MP}$
      5. Compute submatrices $F_{MN}$, $F_{PQ}$, $F_{NP}$, $F_{MQ}$, $F_{NQ}$, $F_{MP}$
      6. Send submatrices of $F$ to their owners
   7. end
8. end

Two basic options for distributed parallelization:
   - Dynamic scheduling of tasks, where a task is a set of shell quartets
   - Static partitioning of the shell quartets, maintaining load balance, and minimizing communication of $D$ and $F$
Static partitioning problem

Balance the number of non-screened shell quartets in each partition.

Each partition should use a minimal set of $D$ and $F$ submatrices.
Partitioning framework

**1D partitioning for \( p \) nodes**

\[(\mathcal{M}_i, : | : , :) \equiv \{(MN|PQ), \text{ s.t. } M \in \mathcal{M}_i, \text{ for all } N, P, Q\}, \quad i \in \{1, \ldots, p\}\]

**2D partitioning for \( p_r \times p_c \) nodes**

\[(\mathcal{M}_i, \mathcal{N}_j | : , :) \equiv \{(MN|PQ), \text{ s.t. } M \in \mathcal{M}_i, N \in \mathcal{N}_j, \text{ for all } P, Q\}, \quad i \in \{1, \ldots, p_r\}, j \in \{1, \ldots, p_c\}\]

\[(\mathcal{M}_i, : | \mathcal{P}_k, :) \equiv \{(MN|PQ), \text{ s.t. } M \in \mathcal{M}_i, P \in \mathcal{P}_k, \text{ for all } N, Q\}, \quad i \in \{1, \ldots, p_r\}, j \in \{1, \ldots, p_c\}\]

Two “types” of 2D partitioning.
One type each of 1D, 3D, and 4D partitioning.

We use 2D partitions of the form \((\mathcal{M}_i, : | \mathcal{P}_k, :).\)
2D partitioning

Define a *slice* or *task* as $T_{MP} \equiv (M, : | P, : )$. This leads to a *task matrix*.

**Even grouping of shell indices** – assumes number of non-screened shell quartets will average out

\[ \begin{bmatrix}
T_{11} & T_{12} & T_{13} & T_{14} & T_{15} & T_{16} \\
T_{21} & T_{22} & T_{23} & T_{24} & T_{25} & T_{26} \\
T_{31} & T_{32} & T_{33} & T_{34} & T_{35} & T_{36} \\
T_{41} & T_{42} & T_{43} & T_{44} & T_{45} & T_{46} \\
T_{51} & T_{52} & T_{53} & T_{54} & T_{55} & T_{56} \\
T_{61} & T_{62} & T_{63} & T_{64} & T_{65} & T_{66}
\end{bmatrix} \]

**Uneven grouping of shell indices** – adjust grouping to balance number of non-screened shell quartets in each partition

\[ \begin{bmatrix}
T_{11} & T_{12} & T_{13} & T_{14} & T_{15} & T_{16} \\
T_{21} & T_{22} & T_{23} & T_{24} & T_{25} & T_{26} \\
T_{31} & T_{32} & T_{33} & T_{34} & T_{35} & T_{36} \\
T_{41} & T_{42} & T_{43} & T_{44} & T_{45} & T_{46} \\
T_{51} & T_{52} & T_{53} & T_{54} & T_{55} & T_{56} \\
T_{61} & T_{62} & T_{63} & T_{64} & T_{65} & T_{66}
\end{bmatrix} \]
Estimating the number of non-screened shell quartets in a task \((M, : | P, : )\)

Define \(\Phi(M)\) to be the set of indices of shells that have nearby centers to shell \(M\), i.e., shell \(M\) and shells in \(\Phi(M)\) are “close.”

Set of non-screened shell quartets in \((M, : | P, : )\) is contained in

\[
\{ (MN|PQ) \text{ s.t. } N \in \Phi(M), Q \in \Phi(P) \}.
\]

Good upper bound on number of non-screened shell quartets is \(\eta(M)\eta(P)\), where \(\eta(M)\) is the number of elements in \(\Phi(M)\).
Estimating the number of non-screened shell quartets in a task (100, : |200, :)

nz = 10526
Number of non-screened shell quartets in \((M, : | P, : )\)

System: \(C_{24}H_{50}\) with cc-pVDZ (294 shells, 586 basis functions)
Uneven grouping of shell indices

Group the indices $M$ into sets $G_i$ such that the sum of the $\eta(M)$ for each subset is about the same,

$$\sum_{M \in G_i} \eta(M) \approx \eta^*.$$ 

Task $(M, : | P, :)$ is assigned to partition $(i, j)$ if $M \in G_i$ and $P \in G_j$.

Then for partition $(i, j)$, the estimated number of non-screened shell quartets is

$$\sum_{M \in G_i} \eta(M) \eta(P) = \sum_{M \in G_i} \eta(M) \times \sum_{P \in G_j} \eta(P) \approx (\eta^*)^2$$

which is approximately balanced with other partitions.
Choose partitions to balance cost

\[
\begin{bmatrix}
\eta(1)\eta(1) & \eta(1)\eta(2) & \eta(1)\eta(3) & \eta(1)\eta(4) & \eta(1)\eta(5) & \eta(1)\eta(6) \\
\eta(2)\eta(1) & \eta(2)\eta(2) & \eta(2)\eta(3) & \eta(2)\eta(4) & \eta(2)\eta(5) & \eta(2)\eta(6) \\
\eta(3)\eta(1) & \eta(3)\eta(2) & \eta(3)\eta(3) & \eta(3)\eta(4) & \eta(3)\eta(5) & \eta(3)\eta(6) \\
\eta(4)\eta(1) & \eta(4)\eta(2) & \eta(4)\eta(3) & \eta(4)\eta(4) & \eta(4)\eta(5) & \eta(4)\eta(6) \\
\eta(5)\eta(1) & \eta(5)\eta(2) & \eta(5)\eta(3) & \eta(5)\eta(4) & \eta(5)\eta(5) & \eta(5)\eta(6) \\
\eta(6)\eta(1) & \eta(6)\eta(2) & \eta(6)\eta(3) & \eta(6)\eta(4) & \eta(6)\eta(5) & \eta(6)\eta(6) \\
\end{bmatrix}
\]

Bottom-middle block is

\[
[\eta(5) + \eta(6)] [\eta(2) + \eta(3) + \eta(4)]
\]
What communication is required for a task?

After computing $(M, : | P, :)$, what are the submatrices of $D$ and $F$ that are needed?

Submatrices of $D$ that are needed are contained in

$$(M, \Phi(M)), \quad (P, \Phi(P)), \quad (\Phi(M), \Phi(P))$$

i.e., $D_{MN}, D_{PQ}, D_{NP}, D_{MQ}, D_{NQ}, D_{MP}$ are all contained in one of the above sets. Similarly for $F$.

These submatrices of $D$ are *prefetched* before computations are started. Submatrices of $F$ are used to update the distributed, global $F$. 
Reorder shells for locality using a space filling curve

Elements of $D$ required

- for $(100, : | 200, :)$
  - 1 task
- for $(100 : 109, : | 200 : 209, :)$
  - 100 tasks

100 tasks require only $10 \times$ communication of single task
System: $C_{24}H_{50}$ with cc-pVDZ (294 shells, 586 functions)
### 64 node partitioning

#### Protein with 617 basis functions

<table>
<thead>
<tr>
<th>Submatrix Description</th>
<th>Shell Quartet Balance</th>
<th>Communication Balance</th>
<th>Ave Comm/node (Submatrices)</th>
<th>Ave Comm/node (kB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>((M_i, N_j</td>
<td>:::))</td>
<td>8x8</td>
<td>3.81</td>
<td>40731.3</td>
</tr>
<tr>
<td>((M_i, :</td>
<td>P_k,:))</td>
<td>8x8</td>
<td>2.01</td>
<td>42789.4</td>
</tr>
<tr>
<td>((M_i, N_j</td>
<td>P_k,:))</td>
<td>4x4x4</td>
<td>7.71</td>
<td>18093.6</td>
</tr>
<tr>
<td>((M_i, N_j</td>
<td>P_k,Q_l))</td>
<td>4x4x2x2</td>
<td>11.38</td>
<td>16629.7</td>
</tr>
<tr>
<td>((M_i, :</td>
<td>P_k,:))</td>
<td>8x8</td>
<td>1.15</td>
<td>35674.4</td>
</tr>
</tbody>
</table>

#### Alkane with 1930 basis functions

<table>
<thead>
<tr>
<th>Submatrix Description</th>
<th>Shell Quartet Balance</th>
<th>Communication Balance</th>
<th>Ave Comm/node (Submatrices)</th>
<th>Ave Comm/node (kB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>((M_i, N_j</td>
<td>:::))</td>
<td>8x8</td>
<td>6.37</td>
<td>82484.9</td>
</tr>
<tr>
<td>((M_i, :</td>
<td>P_k,:))</td>
<td>8x8</td>
<td>1.07</td>
<td>84447.5</td>
</tr>
<tr>
<td>((M_i, N_j</td>
<td>P_k,:))</td>
<td>4x4x4</td>
<td>4.26</td>
<td>40707.0</td>
</tr>
<tr>
<td>((M_i, N_j</td>
<td>P_k,Q_l))</td>
<td>4x4x2x2</td>
<td>8.01</td>
<td>37533.6</td>
</tr>
<tr>
<td>((M_i, :</td>
<td>P_k,:))</td>
<td>8x8</td>
<td>1.03</td>
<td>80027.0</td>
</tr>
</tbody>
</table>
Additional complication

Find the partitioning of the Fock matrix at the same time.

So far, we use geometric information for partitioning the Fock matrix.
Additional complication

Shell quartets are not created equal. Need “weights” for the tasks.

Cost of computing a shell quartet varies widely. Cost depends on: basis functions comprising the integral, total angular momentum of the basis functions, degree of contraction of the basis functions.

If the cost can be modeled as the product of four factors, then can use previous approach to partition the total work.
Primitives and Contractions

Contracted function is a linear combination of primitive Gaussians:

$$\phi_A = \sum_i c_{Ai} \chi_{Ai}$$

Contracted ERI as sum of primitive ERI:

$$\langle \phi_A \phi_B | \phi_C \phi_D \rangle = \sum_{\mu} \sum_{\nu} \sum_{\lambda} \sum_{\sigma} c_{A\mu} c_{B\nu} c_{C\lambda} c_{D\sigma} [\chi_{A\mu} \chi_{B\nu} | \chi_{C\lambda} \chi_{D\sigma}]$$

Figure: Jensen 2007
Improved scalability of Hartree-Fock calculations

1hsg_38 with 3555 basis functions on Stampede (16 cores per node)
Concluding Thoughts

Open problem: partitioning shell quartets for distributed construction of Fock matrices.

- Hypergraph partitioning could help evaluate and find new approaches for partitioning shell quartets.
- How to best use geometric information for partitioning shell quartets?
- Hierarchically combine geometric partitioning (for grouping shell quartets, e.g., 4-D partitions) with hypergraph partitioning?

Acknowledgments

Xing Liu (now at IBM Research)
Intel Parallel Computing Lab
Intel Parallel Computing Center grant