One-sided Communication Implementation in FMO Method


Computing and Communications Center, Kyushu University

†Fukuoka Industry, Science & Technology Foundation
Overview of FMO method (1)

- FMO (Fragment Molecular Orbital method) has been developed by Kitaura (AIST) and co-workers to calculate electronic states of a macromolecule such as a protein and nucleotide.
- The target molecule is divided into *fragments* (monomers).
- *Ab initio* molecular orbital (MO) calculation is carried out on each fragment and fragment pair (dimer).
- Usually executed with high parallel efficiency (>90%).
- The method can execute all-electron calculation on a protein molecule with 10,000 atoms.
Example of fragmentation
Overview of FMO method (2)

monomer calculation

Hamiltonian

\[ H^I = \sum_{i \in I} \left[ h_i + \sum_{J \neq I} V^J (r_i) \right] + \sum_{i > j \in I} \frac{1}{|r_i - r_j|} \]

environmental electrostatic potential

\[ V^J (r) = -\sum_{A \in K} \frac{Z_A^J}{|r - R_A|} + \sum_{i,j \in K} \int \frac{\rho^J (r')}{|r - r'|} dr' \]

(potential from other monomers)

\( \rho^J \) electron density of monomer \( J \)

monomer calculation depends on other monomers electron density

iterated until all electron densities unchanged (SCC procedure)
Overview of FMO method (3)

Schrödinger equation

\[ H^I \Psi^I = E^I \Psi^I \]

RHF (Restricted Hartree-Fock) SCF method

Molecular Orbital (MO)

\[ \phi^I_i(r) = \sum_j \chi^I_j(r) C^I_{ji} \]

\( \{C^I_{ij}\} \quad \text{MO coefficients} \)

\( \{\chi^I_j\} \quad \text{basis functions} \)

electron density

\[ \rho^I = 2 \sum_{jk} \chi^I_j \chi^I_k \sum_i C^I_{ji} C^I_{ki} = \sum_{jk} \chi^I_j \chi^I D^I_{jk} \]

\[ D^I_{jk} = 2 \sum_i C^I_{ji} C^I_{ki} \quad \text{density matrix elements} \quad N_b (N_b + 1)/2 \]

\( N_b \quad \text{number of basis functions} \)
Overview of FMO method (4)

dimer calculation

Hamiltonian

\[ H_{IJ} = \sum_{i \in IJ} [h_i + \sum_{K \neq I,J} V^K (r_i)] + \sum_{i > j \in IJ} \frac{1}{|r_i - r_j|} \]

environmental electrostatic potential

\[ V^K (r) = -\sum_{A \in K} \frac{Z^K_A}{|r - R_A|} + \sum_{i,j \in K} \int \rho^K (r') \frac{1}{|r - r'|} dr' \]

Schrödinger equation

\[ H_{IJ} \Psi^{IJ} = E^{IJ} \Psi^{IJ} \]

Total energy

\[ E = \sum_{I > J} E^{IJ} - (N - 2) \sum_{I} E^{I} \]

not all dimers are calculated by SCF
dimer-es approximation

- For distant dimers, SCF calculation are not carried out
- Energy is calculated by electrostatic approximation

\[ E^{IJ} \approx E^I + E^J - \int \frac{\rho^I(r_1) \rho^J(r_2)}{|r_1 - r_2|} \, dr_1 \, dr_2 \]
Flow chart of FMO calculation

1. Initial density calculation
   \[ \{D_i^0\} \]

2. Electronic structure calculation for monomer
   \[ H^I \Psi^I = E^I \Psi^I \]
   - Not yet converged
   - Convergence check of SCC procedure

3. Electronic structure calculation for SCF-dimer
   \[ H^{IJ} \Psi^{IJ} = E^{IJ} \Psi^{IJ} \]

4. ES dimer calculation
   \[ E^{ij} = E^i + E^j - \int \rho^i(r_1) \rho^j(r_2) \frac{dr_1 dr_2}{|r_1 - r_2|} \]

5. Total energy calculation
   \[ E = \sum_{i > j} E^{ij} - (N - 2) \sum_i E^i \]

Legend:
- Red: For all monomers
- Green: For all SCF-dimers
- Blue: For all separated dimers
Density requirements and update in FMO calculation

1. **Initial Density Calculation**
   \[ \{D^0_i\} \]

2. **SCC Procedure**
   - **Not Yet Converged**
     - Electronic Structure Calculation for Monomer
     \[ H^I \Psi^I = E^I \Psi^I \]
   - Convergence Check of SCC Procedure
   - Update Monomer Density Matrices

3. **Already Converged**
   - Electronic Structure Calculation for SCF-Dimer
     \[ H^{IJ} \Psi^{IJ} = E^{IJ} \Psi^{IJ} \]
   - ES-Dimer Calculation
     \[ E^{IJ} \equiv E^I + E^J - \int \frac{\rho^I(r_1)\rho^J(r_2)}{|r_1 - r_2|} \, dr_1 \, dr_2 \]
   - Total Energy Calculation
     \[ E = \sum_{I>J} E^{IJ} - (N - 2) \sum_I E^I \]
   - Update Monomer Density Matrices
   - Refer Monomer Matrices Needed in Electronic Structure Calculation
   - Refer Two Monomer Matrices Needed in Dimer-ES Calculation
   - Refer Some Monomer Density Matrices
Approximation of $\sum_{J \neq I} V^J$ calculation (1)

matrix element $V^K_{ij}$ of $V^K (r)$ (needed in SCF calculation)

$$V^K_{ij} = \int \chi^I_i (r) V^K (r) \chi^I_j (r) \, dr$$

$$= \sum_{A \in K} \int \chi^I_i (r) \frac{-Z_A}{|r - R_A|} \chi^I_j (r) \, dr + \sum_{i,j \in K} \int \chi^I_i (r) \frac{\rho^K (r')}{|r - r'|} \chi^I_j (r) \, dr' \, dr$$

involves 4 center two electron integrals

$$\int \chi^I_i (r) \int \frac{\rho^K (r')}{|r - r'|} \, dr' \chi^I_j (r) \, dr$$

$$= \sum_{k,l \in K} \int \frac{\chi^I_i (r) \chi^I_j (r) \chi^K_k (r') \chi^K_l (r')}{|r - r'|} \, dr' \, dr \, D^K_{kl}$$

order of calculation costs $O(N_b^{3.5-4})$ $N_b$ number of basis functions

4-center two electron integral calculations for all environment monomers are prohibitive
Approximation of $\sum_{j\neq i} V^j$ calculation (2)

esp-aoc approximation

$$\rho^K (\mathbf{r}) \equiv \sum_{i \in K} \chi^K_i (\mathbf{r}) \chi^K_i (\mathbf{r}) (D^K S^K)_{ii}$$

$S^K_{ii}$ overlap matrix of fragment $K$

$$S^K_{ij} = \int \chi^K_i (\mathbf{r}) \chi^K_j (\mathbf{r}) d\mathbf{r}$$

$$(D^K S^K)_{ii}$ Mulliken AO population of $\chi^K_i (\mathbf{r})$

$$V^K (\mathbf{r}) \equiv - \sum_{A \in K} \frac{Z_A}{|\mathbf{r} - \mathbf{R}_A|} + \sum_{i \in K} \int \frac{\chi^K_i (\mathbf{r}') \chi^K_i (\mathbf{r}')}{|\mathbf{r}' - \mathbf{r}|} d\mathbf{r}' (D^K S^K)_{ii}$$
Approximation of $\sum_{J \neq I} V^J$ calculation (3)

**esp-ptc approximation**

$$\rho^K (r) \approx \sum_{A \in K} \delta (r - R_A) Q^K_A$$  point charge approx. of

$Q^K_A$  Mulliken atomic charge of the nucleus $A$

$$Q^K_A = \sum_{i \in A} \left( D^K S^K \right)_{ii} = \sum_{i \in A, j \in K} D^K_{ij} S^K_{ji}$$

$$V^K (r) \approx - \sum_{A \in K} \frac{Z^K_A}{|r - R_A|} + \sum_{A \in K} \frac{Q^K_A}{|r - R_A|}$$

$N_{4C}$  number of environment monomers for which 4-center two electron integral calculations are carried out
Hypothetical petascale computing environment

Number of CPUs : 1PFlops = 100,000 CPU
  (current one CPU peak performance ~10GFlops)

Number of fragments $N_f : 10,000$-$100,000$
  (the current largest $N_f \sim 1,000$)
Memory requirement in FMO

Number of fragments \( N_f \)

The size of one density matrix \( D^I \) \( \equiv \) 350KB

Interfragment distance data \( R^{IJ} \) \( \equiv \) \( N_f^2 / 2 \)

<table>
<thead>
<tr>
<th>( N_f )</th>
<th>( D^I )</th>
<th>( R^{IJ} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,000</td>
<td>350MB</td>
<td>4MB</td>
</tr>
<tr>
<td>5,000</td>
<td>1.7GB</td>
<td>95MB</td>
</tr>
<tr>
<td>10,000</td>
<td>3.4GB</td>
<td>380MB</td>
</tr>
<tr>
<td>50,000</td>
<td>17GB</td>
<td>9.3GB</td>
</tr>
<tr>
<td>100,000</td>
<td>34GB</td>
<td>37GB</td>
</tr>
</tbody>
</table>

\[ \times \] Memory per one CPU core \( > 10 \) GB

\[ \downarrow \]

It is difficult for each process to store all necessary data
OSC implementation in OpenFMO (1)

- In its execution, only the process of rank 0 is used as a server process for dynamic load balancing. All the other processes are worker processes and are divided into groups and the two-level parallelization is used as the other implementations.
OSC implementation in OpenFMO (2)

Group00

node00

Group01

node01
node02
node03

Group02

node04

memory window created by MPI_Win

density matrix data

MPI_Get
Estimation of communication cost (1)

Assumptions

1. The sizes of all density matrices are the same. All groups have the same number of worker processes.

2. The time to put or get one density matrix is equal to the average point-to-point communication time.

3. MPI_Bcast implements a binomial tree algorithm so that the time to broadcast one density matrix over $N$ processes is obtained by $t_{p2p} \log_2 N$.

4. And for OSC scheme, dynamic load balancing works well and the delay due to competing put or get requests is ignorable. Therefore all groups execute the same number of monomer and dimer jobs and this means that all groups have the same $N_{\text{proc}} / N_{\text{group}}$.
Estimation of communication cost (2)

\[
\begin{array}{|c|c|}
\hline
N_f & 10,000-100,000 \\
N_{\text{proc}} & 96,000 \\
N_{\text{proc}} / N_{\text{group}} & 32 \\
\hline
\end{array}
\]

| \(N_{\text{SCC}}\) | SCC iterations | 20 |
| \(\overline{N}_{\text{mon}}^{4C}\) | neighboring monomers (average) | 7 |
| \(\overline{N}_{\text{dim}}^{4C}\) | neighboring monomers (average) | 12 |
| \(\overline{N}_{\text{dim}}^{\text{SCF}}\) | monomers with which a monomer forms a SCF dimer | 17 |
| \(N_f \overline{N}_{\text{dim}}^{\text{SCF}} / 2\) | SCF dimers |
| \(\overline{t}_{p2p}\) | point-to-point communication time to transfer one density matrix |
Estimation of communication cost (3)

OSC scheme

- number of monomer get requests: $N_{\text{get}}^\text{mon} = N_{\text{SCC}} N_f \left( \bar{N}_{4C}^\text{mon} + 1 \right)$
- total number of dimers: $N_f \left( N_f - 1 \right)/2$
- number of SCF dimers: $N_f \bar{N}_{\text{SCF}}^\text{dim} / 2$
- number of SCF dimer get requests: $N_{\text{get}}^\text{dim} = N_f \bar{N}_{\text{SCF}}^\text{dim} \left( \bar{N}_{4C}^\text{dim} + 2 \right)/2$
- number of ES dimers: $N_f \left( N_f - 1 \right)/2 - N_f \bar{N}_{\text{SCF}}^\text{dim} / 2$
- number of ES dimer get requests: $N_{\text{get}}^\text{es} = \left[ N_f \left( N_f - 1 \right)/2 - N_f \bar{N}_{\text{SCF}}^\text{dim} / 2 \right] \times 2$
- total number of get requests: $N_{\text{get}} = N_{\text{get}}^\text{mon} + N_{\text{get}}^\text{dim} + N_{\text{get}}^\text{es}$

$$= N_{\text{SCC}} N_f \left( \bar{N}_{4C}^\text{mon} + 1 \right) + N_f \bar{N}_{\text{SCF}}^\text{dim} \bar{N}_{4C}^\text{dim} / 2 + N_f \left( N_f - 1 \right)$$
Estimation of communication cost (4)

OSC scheme

- cost of \( \text{MPI}_\text{Bcast} \) \( \bar{t}_{p2p} \log_2 N \)
- cost for one get request \( \bar{t}_{p2p} \left(1 + \log_2 \frac{N_{\text{proc}}}{N_{\text{group}}} \right) \)
- total cost of put requests \( N_{\text{SCC}} N_f \bar{t}_{p2p} / N_{\text{group}} \)
- total cost per one group \( T_{\text{OSC}} = N_{\text{SCC}} N_f \bar{t}_{p2p} / N_{\text{group}} \)
  \[ + \bar{t}_{p2p} \left(1 + \log_2 \frac{N_{\text{proc}}}{N_{\text{group}}} \right) N_{\text{get}} / N_{\text{group}} \]

Bcast scheme

- total cost per one group \( T_{\text{Bcast}} = N_{\text{SCC}} N_f \bar{t}_{p2p} \log_2 N_{\text{proc}} \)
Estimation of communication cost (5)
Simulation using skelton code (1)

- FMO calculation was simulated using skelton code
  - quantum calculation parts are removed
- Skelton code accumulates estimated time of:
  - molecular integral calculation
  - Fock matrix building, SCF procedure
- Time of each communication are estimated and accumulated
  - MPI_Send, MPI_recv, MPI_get, MPI_put estimated from measurements
  - MPI_Bcast, MPI_Allreduce estimated from point to point communication time by assuming binomial tree, butterfly algorithm
Simulation using skelton code (2)

- Integral calculation time estimation formula
  Integrals required for environmental potential
  1 electron
  \[ f_{1e}^{\text{env}}(N_b) = AN_b + BN_b^2 \]
  \[ A = 1.12072 \times 10^{-5} \quad B = 6.03297 \times 10^{-8} \]
  2 electron (3center)
  \[ f_{2e3C}^{\text{env}}(N_b) = AN_b + BN_b^2 \]
  \[ A = 5.28555 \times 10^{-5} \quad B = 2.60718 \times 10^{-7} \]
  2 electron (4center)
  \[ f_{2e4C}^{\text{env}}(N_b) = AN_b + BN_b^2 \]
  \[ A = 0.00050 \quad B = 3.26399 \times 10^{-6} \]
  Integrals required for SCF
  \[ f_{2e}(N_b) = AN_b^2 + BN_b^3 \]
  \[ A = 7.57123 \times 10^{-5} \quad B = 9.33117 \times 10^{-7} \]
- Fock building time estimation
  \[ f_{\text{Fock}}(N_b) = AN_b^2 + BN_b^3 \]
  \[ A = 5.84313 \times 10^{-5} \quad B = 1.12603 \times 10^{-6} \]
simulation using skelton codes (3)

Aquaporin protein

PDBID 2F2B,

\[ N_f = 492, 14,492 \text{ atoms} \]

6-31G* basis set

\[ N_{\text{proc}} / N_{\text{group}} = 16 \]

\[ N_{\text{group}} = 4, 8, 16, 32, 64 \]

OSC cost < Bcast cost

with \[ N_{\text{proc}} > 512 \]

\( (N_{\text{group}} > 32) \)

Linux cluster in RIKEN Super Combined Cluster (RSCC) was used
Summary

- OSC of MPI-2 standard has been implemented in a new FMO code to reduce the memory requirement per process.

- OSC implementation also makes the exchange of $D^I$ by broadcast unnecessary.

- Evaluation of communication costs shows that OSC scheme has an advantage over the Bcast scheme for $N_f = 10,000-100,000$, $N_{proc} = 96,000$.

- Simulations using skelton code also show that OSC scheme has an advantage in communication cost with large $N_{group}$.

- These results show OSC scheme is prospective for the petascale computing environment.