Theor Chem Acc (2011) 130:317–321
DOI 10.1007/s00214-011-0980-2

REGULAR ARTICLE

MPI/OpenMP hybrid parallel implementation of second-order Møller–Plesset perturbation theory using numerical quadratures

Kazuya Ishimura · Seiichiro Ten-no

Received: 29 March 2011 / Accepted: 14 June 2011 / Published online: 24 June 2011

Abstract An algorithm for massively parallel computers is developed for energy calculations of second-order Møller–Plesset (MP2) perturbation theory with numerical quadratures. Message Passing Interface (MPI) and Open Multi-Processing (OpenMP) technologies are utilized for inter-node and intra-node parallelization, respectively. Computational tasks and intermediates are distributed across nodes by dividing quadrature points, and the distributed data are stored in memory. Benchmark calculations were performed on 256–8,192 CPU cores, and we observed the speed-ups 4,534–6,266 for 8,192 cores. A large calculation for fullerene (C_{60}) with aug-cc-pCVTZ (3,540 basis functions) was completed in ca. 4.8 h on 8,192 cores without invoking molecular symmetry.

Keywords Second-order Møller–Plesset perturbation theory · MPI/OpenMP hybrid parallelization · Numerical quadratures

1 Introduction

Second-order Møller–Plesset (MP2) perturbation theory is the simplest ab initio correlation method [1]. Despite the moderate computational cost, which scales as the fifth power of the number of basis functions, more than 80% of the correlation effect is accounted for in the MP2 correlation energies. Various parallel algorithms have been proposed for canonical MP2 [2–9], density fitting (or Resolution of Identity, RI) MP2 [10–12], local MP2 [13, 14], Laplace-transformed MP2 [15, 16], pseudospectral MP2 [17], and explicitly correlated MP2 [18] methods. Most of those algorithms use single variable distributions of atomic orbitals (AOs), molecular orbitals (MOs), or auxiliary basis functions. Network communication increases with the number of processes in the AO-based parallelization, while repeated evaluations or broadcasts of AO integrals are necessary in the MO-based algorithm. Baker and Pulay [7] developed a disk-based two-step parallelization scheme, where AO and MO indices are distributed for the first and second half transformations, respectively. The total amount of network communication is independent of the number of processes, and AO integrals are calculated only once.

The number of central processing unit (CPU) cores per node continues to increase because of the power and heat problems. The latest high-performance supercomputers consist of more than 10,000 nodes and 100,000 CPU cores. It is necessary to develop a highly efficient parallelization scheme for distributing computational tasks and data in order to make the best use of such architectures. One approach for the data distribution is the shared-memory programming model for distributed-memory computers such as global arrays (GA) [19] and distributed data interface (DDI) [20], in which global memory is accessed without explicit cooperation by other processes. Another

---

Dedicated to Professor Shigeru Nagase on the occasion of his 65th birthday and published as part of the Nagase Festschrift Issue.

Electronic supplementary material The online version of this article (doi:10.1007/s00214-011-0980-2) contains supplementary material, which is available to authorized users.

K. Ishimura · S. Ten-no (✉)
Graduate School of System Informatics, Kobe University,
1-1 Rokkodai-cho, Nada-ku, Kobe 657-8501, Japan
e-mail: tenno@cs.kobe-u.ac.jp

S. Ten-no
CREST, Japan Science and Technology Agency (JST),
4-1-8 Honcho Kawaguchi, Saitama 332-0012, Japan

approach is the combined use of Message Passing Interface (MPI) and Open Multi-Processing (OpenMP) for inter-node and intra-node parallelization, respectively. Because OpenMP allows CPU cores to share data within a node, larger arrays per process are available compared with the flat MPI parallelization. Furthermore, computational tasks can be efficiently distributed to nodes and cores using different indices for MPI and OpenMP.

In this paper, an MPI/OpenMP hybrid parallel algorithm for large-scale MP2 energy calculations is presented. Two-electron MO integrals are evaluated using a numerical quadrature technique and two-step distribution of quadrature points and virtual MOs is introduced. A huge amount of data are distributed across nodes and stored in their memory.

2 Theory and algorithm

The closed-shell MP2 energy can be written as

$$E_{\mathrm{MP2}} = \sum_{ij}^{\mathrm{occ}} \sum_{ab}^{\mathrm{virt}} \frac{(ai|bj)\{2(ai|bj) - (aj|bi)\}}{\varepsilon_i - \varepsilon_j - \varepsilon_a - \varepsilon_b}, \tag{1}$$

where i and j are doubly occupied MOs, a and b are virtual MOs, and ε are the corresponding orbital energies. In the present implementation, we calculate two-electron MO integrals using numerical quadratures [21, 22], which have been utilized for three- and four-electron integrals in MP2-F12 calculations. The two-electron integrals are expanded into products of 2- and 3-index objects as,

$$(ai|bj) = \langle ab|r_{12}^{-1}|ij\rangle = \sum_g \overline{\phi_a}(\mathbf{r}_g)\phi_i(\mathbf{r}_g)\left\langle b\left|r_{1g}^{-1}\right|j\right\rangle, \tag{2}$$

where $\phi_i(\mathbf{r}_g)$ is an MO amplitude at the quadrature point, $\mathbf{r}_g$, and $\overline{\phi_a}(\mathbf{r}_g)$ denotes a weighted MO amplitude. The latter is obtained by

$$\begin{aligned}\overline{\phi_a}(\mathbf{r}_g) &= \omega(\mathbf{r}_g)\phi_a(\mathbf{r}_g)\\ &= \omega(\mathbf{r}_g)\sum_\mu C_{\mu a}\chi_\mu(\mathbf{r}_g)\end{aligned} \tag{3}$$

where $\omega(\mathbf{r})$ is a weight function, $\chi_\mu(\mathbf{r})$ are AOs, and $C_{\mu a}$ are MO coefficients. $\left\langle b|r_{1g}^{-1}|j\right\rangle$ are the following three-center electric field integrals,

$$\begin{aligned}\left\langle b|r_{1g}^{-1}|j\right\rangle &= \int d\mathbf{r}_1\phi_b(\mathbf{r}_1)\phi_j(\mathbf{r}_1)\left|\mathbf{r}_1 - \mathbf{r}_g\right|^{-1}\\ &= \sum_{\mu\nu} C_{\mu b}C_{\nu j}\int d\mathbf{r}_1\chi_\mu(\mathbf{r}_1)\chi_\nu(\mathbf{r}_1)\left|\mathbf{r}_1 - \mathbf{r}_g\right|^{-1}.\end{aligned} \tag{4}$$

The numbers of the MO amplitudes and three-center integrals scale as $(O+V)G$ and OVG, respectively, where O, V, and G are the numbers of occupied MOs, virtual MOs, and quadrature points, respectively. By the use of numerical quadratures, the scaling of intermediates is reduced from quartic to cubic. Moreover, the parallelization scheme is simplified since there is no need to transform the indices of quadrature points. Various numerical integration schemes have been developed in density functional theory. We use the Voronoi polyhedra [23] for spatial partitioning. Euler–Maclaurin and Gauss–Legendre schemes [24] are used for radial and angular (θ and ϕ) grid quadratures, respectively, in this particular work.

Figure 1 shows a schematic representation of our MPI/OpenMP parallel algorithm for MP2 energy calculations using numerical quadratures. The algorithm consists of 5 steps: (1) Grid data generation, (2) MO amplitude calculation, (3) Three-center integral calculation, (4) MO amplitude communication, and (5) Three-center integral communication and calculations of two-electron integrals and MP2 energy.

In Steps from 1 to 3, quadrature points are distributed, while virtual MO pairs are distributed in Step 5. In Steps 2 and 3, MO amplitudes and three-center integrals are calculated and stored for the grid points distributed to each process. The MO amplitudes are gathered to the master process and then broadcasted in Step 4. The three-center integrals for all occupied MOs, quadrature points, and single virtual MO are gathered to appropriate processes in Step 5. The communication cost is $\mathrm{O}((O+V)G\log_2 N_{\mathrm{proc}})$ and $\mathrm{O}(OVG)$ for MO amplitudes and three-center integrals, respectively, where N_{proc} is the number of processes. The amount of communication is reduced from the fourth power of the numbers of basis functions and MOs in conventional four-index transformation schemes. To simplify the communication, MPI calls are outside the OpenMP regions. By switching the index of the MPI parallelization from g in Steps 1 to 3 to ab pairs in Step 5, repeated calculations of objects are avoided. When the numbers of quadrature points and virtual MO pairs are much larger than the number of processes, computational tasks and data are almost equally distributed. The computational cost for two-electron integrals is $\mathrm{O}(O^2V^2G)$. This method is advantageous compared with the $\mathrm{O}(ON^4)$ first integral transformation of the conventional method when a large basis set, roughly $N^2 > OG$, is used.

OpenMP parallelization is introduced for the distributions of the quadrature points in Step 1, AO indices in Steps 2 and 3, and occupied MO pair indices in Step 5. By sharing two- and three-dimensional arrays with all CPU cores in a node, a large amount of data can be stored in memory.

In the RI-MP2 method, two-electron integrals are evaluated by a sum of products of 2-index and 3-index objects

```
Step 1: Loop over $g$ (MPI and OpenMP parallelization)
    Calculate coordinates and weights for grid points
  End of $g$ loop
  Broadcast the number of the grid points of each process
Step 2: Loop over $\mu$ (OpenMP parallelization)
    Loop over $g$ (MPI parallelization)
      Calculate $\phi_\mu(\mathbf{r}_g)$
    End of $g$ loop
    Transform to $\phi_p(\mathbf{r}_g)$
  End of $\mu$ loop
Step 3: Loop over $g$ (MPI parallelization)
    Loop over $\mu$ (OpenMP parallelization)
      Loop over $\nu$
        Calculate $\langle \mu | r_{1g}^{-1} | \nu \rangle$
      End of $\nu$ loop
    End of $\mu$ loop
    Transform to $\langle b | r_{1g}^{-1} | j \rangle$
  End of $g$ loop
Step 4: Gather partial $\phi_p(\mathbf{r}_g)$ to master process
  Broadcast $\phi_p(\mathbf{r}_g)$ to all processes
Step 5: Loop over $b$ (MPI parallelization)
    Send and receive $\langle b | r_{1g}^{-1} | j \rangle$ for all $j$ and $g$, and single $b$
    Loop over $a$ (MPI parallelization)
      Loop over $ij$-batch (OpenMP parallelization)
        Calculate $(ai|bj)$
      End of $ij$ loop
      Calculate partial MP2 energy
    End of $a$ loop
  End of $b$ loop
  Accumulate MP2 energy to master process
```

Fig. 1 Outline of MPI/OpenMP hybrid parallel algorithm for MP2 energy calculation

including auxiliary basis functions. This approach makes it possible to obtain accurate energies at reasonable computational costs. Parallel algorithms [10–12] include the network communication for at most 3-index objects, and they will be also suitable for parallel computing. The parallel implementation of the pseudospectral method [25] resembles the present algorithm. The weighted MO amplitudes in Eq. 2 are replaced by least-square fitting operators in this case.

3 Results and discussion

The algorithm was implemented into the quantum chemistry program package GELLAN [26]. Benchmark calculations with up to 8,192 CPU cores were performed using the facilities of the Supercomputer Center, Institute for Solid State Physics, University of Tokyo. The computer system is SGI Altix ICE 8400EX (CPU: Intel Xeon X5570 2.93GH, 8 CPU cores/node, Memory: 24 GB/node, Network: Enhanced Hypercube with 4x QDR InfiniBand) with Intel Fortran Compiler version 11.1 and Intel Math Kernel Library version 10.2. Eight threads per process are generated in the MPI/OpenMP hybrid parallel calculations.

We have tested the present numerical MP2 algorithm for anthracene ($C_{14}H_{10}$), tetracene ($C_{18}H_{12}$), and coronene ($C_{24}H_{12}$) molecules using the medium, fine, and ultrafine grids in [22], the numbers of points in which are 9,216, 24,576, and 51,200, respectively. The geometrical parameters are optimized at B3LYP [27, 28]/6-31G(d) [29, 30], all electrons are correlated, and molecular symmetries are not utilized for electron correlation in this particular work. We use the aug-cc-pCVTZ basis set for coronene, aug-cc-pCVQZ for tetracene, and aug-cc-pCV5Z for anthracene [31–33]. Linearly-dependent vectors are removed from basis set with the threshold 10^{-6} for eigenvalues of the overlap matrix. The conventional MP2 energies are obtained using the GAMESS program [34]. The system sizes, SCF, and MP2 energies are summarized in Table 1. The largest error of the medium grid is 1.7 mE_h for coronene, while the ultrafine grid leads to correlation energies accurate to a few 10 μE_h. The use of the Lebedev quadrature [35] and partitioning an atom into layers for angular grid [36], which are not employed in this work, will increase the efficiency. Furthermore, the error of numerical integration can be reduced by ca. one order of magnitude by renormalizing orbital amplitudes with respect to the overlap metric (Ten-no S, unpublished).

The elapsed times and speed-ups relative to the 256 CPU core time are shown in Table 2. The speed-ups are almost linear up to 2,048 cores and the parallel efficiency retained even at the maximum 8,192 cores. In addition, the calculation for fullerene (C_{60}) with aug-cc-pCVTZ (3,540 basis functions, 180 occupied MOs) was completed in only 17,179 s (4.8 h) on 8,192 cores with the total amount of the three-center integrals ca. 1.9 TB. By distributing large data across nodes, MP2 calculations for such large molecules become feasible in spite of the limited amount of memory per node.

The elapsed times and speed-ups for computational steps are summarized in Table 3. The speed-ups for all steps are roughly linear up to 8,192 cores, indicating that the computational tasks are almost evenly distributed among the processes. The most time-consuming step is $O(O^2V^2G)$ for the calculation of two-electron integrals using MO amplitudes and three-center integrals.

Table 4 shows the communication times for MO amplitudes and three-center integrals. The duration for MO

Table 1 System sizes, SCF, and MP2 energies with different grids (E_h)

	Occupied MOs	Basis functions	SCF	MP2			
				Medium	Fine	Ultrafine	Exact
Anthracene	47	3,334	−536.200 418	−3.142 972	−3.143 208	−3.143 289	
Tetracene	60	2,514	−688.889 241	−3.957 214 (0.000 399)	−3.957 518 (0.000 095)	−3.957 641 (−0.000 028)	−3.957 613
Coronene	78	1,692	−916.227 106	−4.941 021 (0.001 704)	−4.942 623 (0.000 102)	−4.942 747 (−0.000 022)	−4.942 725

Deviations from the exact energies are given in parentheses

Table 2 Elapsed times for MP2 calculations (seconds)

Molecule	Number of CPU cores				
	256	1,024	2,048	4,096	8,192
Anthracene	15,180.0 (256.0)	3,869.3 (1,004.3)	1,977.3 (1,965.3)	1,095.9 (3,545.9)	857.2 (4,533.7)
Tetracene	15,133.8 (256.0)	3,845.7 (1,007.4)	1,973.6 (1,963.1)	1,063.1 (3,644.2)	656.7 (5,900.0)
Coronene	14,212.9 (256.0)	3,601.8 (1,010.2)	1,850.4 (1,966.3)	975.6 (3,729.6)	580.7 (6,266.1)

Speed-ups relative to the 256 CPU core time are given in parentheses

Table 3 Elapsed times for computational steps (seconds)

Number of CPU cores	256	1,024	2,048	4,096	8,192
Anthracene					
MO amplitude	42.70 (256.0)	18.46 (592.2)	5.48 (1,993.4)	2.84 (3,846.5)	1.68 (6,518.6)
Three-center integral	1,507.43 (256.0)	382.14 (1,009.8)	191.38 (2,016.5)	99.83 (3,865.7)	53.17 (7,257.7)
Two-electron integral	13,468.00 (256.0)	3,369.00 (1,023.4)	1,669.69 (2,064.9)	844.08 (4,084.7)	427.88 (8,057.8)
MP2 energy	3.36 (256.0)	0.84 (1,020.1)	0.42 (2,037.7)	0.21 (4,018.2)	0.10 (8,774.5)
Tetracene					
MO amplitude	23.44 (256.0)	5.88 (1,020.5)	3.00 (2,001.9)	1.61 (3,739.4)	0.81 (7,437.0)
Three-center integral	1,061.55 (256.0)	269.65 (1,007.8)	138.06 (1,968.4)	73.34 (3,705.3)	38.45 (7,068.4)
Two-electron integral	13,899.99 (256.0)	3,477.44 (1,023.3)	1,754.19 (2,028.5)	859.29 (4,141.1)	445.21 (7,992.7)
MP2 energy	2.85 (256.0)	0.72 (1,016.2)	0.36 (2,029.5)	0.18 (4,059.0)	0.09 (8,398.0)
Coronene					
MO amplitude	13.62 (256.0)	3.41 (1,021.4)	1.79 (1,952.9)	0.90 (3,877.6)	0.44 (7,995.3)
Three-center integral	592.22 (256.0)	152.31 (995.4)	75.76 (2,001.3)	41.11 (3,688.3)	21.84 (6,941.1)
Two-electron integral	13,392.32 (256.0)	3,340.35 (1,026.4)	1,671.24 (2,051.4)	844.31 (4,060.6)	430.62 (7,961.6)
MP2 energy	2.39 (256.0)	0.60 (1,016.3)	0.30 (2,032.7)	0.15 (4,190.7)	0.07 (8,268.1)

Speed-ups relative to the 256 CPU core time are given in parentheses

amplitudes increases logarithmically with the number of cores, owing to the broadcast in this step. On the other hand, the time for three-center integrals decreases up to 2,048 or 4,096 cores because the size of the communicated data per process is reduced with the number of processes. The communication time on 8,192 cores, however, increases from that on 4,096 cores, especially for anthracene that has the largest number of virtual MOs. The network latency increases with the number of MPI communications, which is proportional to the numbers of virtual MOs and processes. Thus, the latency becomes the bottleneck in communicating three-center integrals for a large number of processes.

The communication time makes up 22–43% of the total MP2 time on 8,192 cores. The ratio of the communication to the computation will decrease with the sizes of molecules and basis sets, since the communications are of $\mathrm{O}((O+V)G\log_2 N_{\mathrm{proc}})$ and $\mathrm{O}(OVG)$ for MO amplitudes and three-center integrals, respectively, while the computational cost scales as $\mathrm{O}(O^2V^2G)$. High parallel efficiency can

Table 4 Elapsed times for communication steps (seconds)

Number of CPU cores	256	1,024	2,048	4,096	8,192
Anthracene					
MO amplitude	31.5	55.5	68.5	80.5	99.9
Three-center integral	125.0	40.4	38.5	64.5	270.3
Tetracene					
MO amplitude	29.3	51.3	61.9	77.9	91.8
Three-center integral	115.6	39.1	14.3	48.6	77.5
Coronene					
MO amplitude	24.4	42.4	51.2	63.8	75.8
Three-center integral	187.3	61.9	49.2	24.2	50.6

thus be attained for large systems even with more than 10,000 CPU cores.

4 Conclusions

We have developed an MPI/OpenMP hybrid parallel algorithm for MP2 calculations using numerical quadrature. The scaling of intermediates is reduced from quartic to cubic, and the distributions of the computational task and data are simplified by utilizing numerical quadratures.

High parallel efficiency is attained by distributing quadrature points and virtual MO pairs. The speed-up for the computation is nearly linear, while the communication time increases because of the broadcast and the network latency when thousands of cores are used. However, the parallel performance of the present scheme will remain high for large systems even with more than 10,000 CPU cores, since the ratio of the computation to the communication increases with the system size.

The parallel scheme using numerical quadratures can be applied to higher-order correlated methods. The implementation for explicitly correlated methods is also in progress.

Acknowledgments The authors thank the Supercomputer Center, Institute for Solid State Physics, University of Tokyo and Computational Materials Science Initiative (CMSI) for the use of the facilities.

References

1. Møller C, Plesset MS (1934) Phys Rev 46:618
2. Watts JD, Dupuis M (1988) J Comput Chem 9:158
3. Limaye AC, Gadre SR (1994) J Chem Phys 100:1303
4. Wong AT, Harrison RJ, Rendell AP (1996) Theor Chim Acta 93:317
5. Schutz M, Lindh R (1997) Theor Chim Acta 95:13
6. Nielsen IMB, Janssen CL (2000) Comput Phys Comm 128:238
7. Baker J, Pulay P (2002) J Comput Chem 23:1150
8. Ishimura K, Pulay P, Nagase S (2006) J Comput Chem 27:407
9. Lotrich V, Flocke N, Ponton M, Yau AD, Perera A, Deumens E, Bartlett RJ (2008) J Chem Phys 128:194104
10. Bernholdt DE, Harrison RJ (1996) Chem Phys Lett 250:477
11. Hattig C, Hellweg A, Kohn A (2006) Phys Chem Chem Phys 8:1159
12. Katouda M, Nagase S (2009) Int J Quantum Chem 109:2121
13. Bernholdt DE, Harrison RJ (1995) J Chem Phys 102:9582
14. Nielsen IMB, Janssen CL (2007) J Chem Theory Comput 3:71
15. Häser M (1993) Theor Chim Acta 87:147
16. Doser B, Lambrecht DS, Ochsenfeld C (2008) Phys Chem Chem Phys 10:3335
17. Beachy MD, Chasman D, Friesner RA, Murphy RB (1998) J Comput Chem 19:1030
18. Valeev EF, Janssen CL (2004) J Chem Phys 121:1214
19. Nieplocha J, Harrison RJ, Littlefield RJ (1996) J Supercomput 10:169
20. Fletcher GD, Schmidt MW, Bode BM, Gordon MS (2000) Comput Phys Commun 128:190
21. Ten-no S (2004) J Chem Phys 121:117
22. Ten-no S (2007) J Chem Phys 126:14108
23. Becke AD (1988) J Chem Phys 88:2547
24. Murray CW, Handy NC, Laming GJ (1993) Mol Phys 78:997
25. Martinez TJ, Carter EA (1994) J Chem Phys 100:3631
26. GELLAN (2010) A hierarchical quantum chemistry program, Kobe University
27. Becke AD (1993) J Chem Phys 98:5648
28. Lee C, Yang W, Parr RG (1988) Phys Rev B 37:785
29. Hehre WJ, Ditchfield R, Pople JA (1972) J Chem Phys 56:2257
30. Hariharan PC, Pople JA (1973) Theoret Chimica Acta 28:213
31. Dunning TH Jr (1989) J Chem Phys 90:1007
32. Kendall RA, Dunning TH Jr, Harrison RJ (1992) J Chem Phys 96:6796
33. Woon DE, Dunning TH Jr (1995) J Chem Phys 103:4572
34. Schmidt MW, Baldridge KK, Boatz JA, Elbert ST, Gordon MS, Jensen JH, Koseki S, Matsunaga N, Nguyen KA, Su S, Windus TL, Dupuis M, Montgomery JA (1993) J Comput Chem 14:1347
35. Lebedev VI, Laikov DN (1999) Doklady Mathematics 59:477
36. Gill PMW, Johnson BG, Pople JA (1993) Chem Phys Lett 209:506