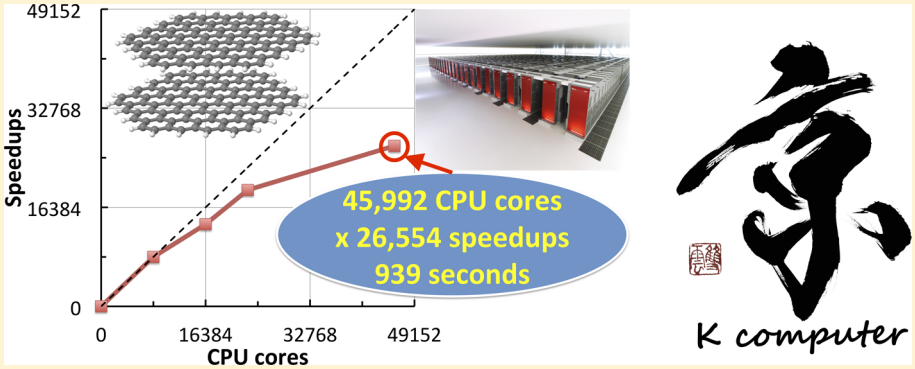
JCTC Journal of Chemical Theory and Computation
Article
pubs.acs.org/JCTC

MPI/OpenMP Hybrid Parallel Algorithm of Resolution of Identity Second-Order Møller–Plesset Perturbation Calculation for Massively Parallel Multicore Supercomputers

Michio Katouda and Takahito Nakajima*

Computational Molecular Science Research Team, RIKEN Advanced Institute for Computational Science, Kobe 657-0047, Japan



ABSTRACT: A new algorithm for massively parallel calculations of electron correlation energy of large molecules based on the resolution of identity second-order Møller–Plesset perturbation (RI-MP2) technique is developed and implemented into the quantum chemistry software NTChem. In this algorithm, a Message Passing Interface (MPI) and Open Multi-Processing (OpenMP) hybrid parallel programming model is applied to attain efficient parallel performance on massively parallel supercomputers. An in-core storage scheme of intermediate data of three-center electron repulsion integrals utilizing the distributed memory is developed to eliminate input/output (I/O) overhead. The parallel performance of the algorithm is tested on massively parallel supercomputers such as the K computer (using up to 45 992 central processing unit (CPU) cores) and a commodity Intel Xeon cluster (using up to 8192 CPU cores). The parallel RI-MP2/cc-pVTZ calculation of two-layer nanographene sheets $(C_{150}H_{30})_2$ (number of atomic orbitals is 9640) is performed using 8991 node and 71 288 CPU cores of the K computer.

1. INTRODUCTION

Electronic structure theory such as ab initio molecular orbital (MO) theory is a powerful tool for elucidating chemical phenomena of molecules such as electronic states, molecular structures, properties, and reaction mechanisms.[1] High-level ab initio MO calculations excellently reproduce the properties for small molecules. However, the computational costs drastically increase with respect to the size of molecules when high-level ab initio MO calculations are performed. To elucidate chemical phenomena of nanomolecules and biological molecules, the development of fast and robust theories and computational techniques of ab initio MO calculations is desired.

Second-order Møller–Plesset perturbation theory (MP2)[2,3] is the simplest method to account for the electron correlations among wave function approaches. It is often used for practical chemical applications. However, the computational cost of the MP2 calculation scales as $O(N^5)$ with respect to the size of molecules (N), and practical applications are limited to molecules of moderate size. To make the MP2 calculations applicable to the large molecular systems such as nanomolecules and biological molecules, development of efficient computational techniques is desired. Recently, several efficient computational methods of MP2 calculations have been developed (for a recent review, see ref 3 and references therein). The resolution of the identity (RI) MP2 (or the density fitting MP2)[4,5] is an efficient computational method for MP2 calculations. RI-MP2 is based on the RI approximation,[6] also termed as the density-fitting approximation,[7] where the four-center electron repulsion integrals (ERIs) are approximated by the product sum of two-center and three-center ERIs. The RI-MP2 techniques can reduce the computational costs as well as the required memory and disk sizes considerably while maintaining reliable accuracies for practical chemical applications.

Recently, the importance of massively parallel calculations has increased, being attributable to the emergence of massively parallel supercomputers that have more than 10 000 nodes and 100 000 central processing unit (CPU) cores. The K computer developed by RIKEN and Fujitsu as a Japanese national project is one of the largest massively parallel supercomputers in the world.[8] The system is a distributed memory supercomputer

Received: September 9, 2013
Published: October 23, 2013

system consisting of 88 128 nodes and 705 024 CPU cores. Each computer node consists of a SPARC64 VIIIfx CPU (8 CPU core per node, 128 giga FLoating-point Operations Per Second (FLOPS)) and a set of memory modules (16 gigabyte (GB) per node). The network between the computer nodes consists of a newly developed torus fusion network named the Tofu network that ensures very high data communication performance and fault tolerance. The network topology is a six-dimensional mesh/torus. Each node can simultaneously communicate to four directions, and the link throughput is 5 GB/s in each direction. Users can develop their application programs adapted to a one-, two-, or three-dimensional torus rank-mapping scheme to improve their usability, fault tolerance, and interoperability. By achieving a performance of 10.51 peta-FLOPS in the LINPACK benchmark, the K computer ranked first for two consecutive TOP500 lists in June and November of 2011.[9] The K computer is expected to be used for various scientific studies such as simulations of earthquakes and tsunamis to protect from disasters, atmospheric simulations for weather forecasting, drug discovery, and material design. Therefore, algorithms and software for massively parallel quantum chemical calculations of real macromolecules are desired for the practical chemical applications of drug discovery and material design on the K computer.

A three-level parallel programming model is recommended to attain high performance on the K computer. The first level is the single instruction multiple data (SIMD) processing at the instruction level on each core. The second is the thread programming on the compute node supported by automatic parallelization or OpenMP directives. The third is the distributed-memory parallel programming by the Message Passing Interface (MPI) library over multiple computing nodes. To maximize the parallel efficiency of MP2 calculations of large molecules on the K computer and other multicore based massively parallel supercomputers, it is important to develop new algorithms and codes applying the MPI/OpenMP hybrid parallel programming model.

Several parallel algorithms for the MP2 calculation have been developed.[10−37] Parallel algorithms of the RI-MP2 method have been developed for calculations of large molecules.[38−40] In a previous study, the authors developed a parallel RI-MP2 algorithm for the calculations of large molecules on commodity personal computer (PC) clusters.[41] This algorithm was designed for efficient parallel calculations by reducing the I/O and network communication overheads and considering uniform task distribution for efficient load balancing. However, the RI-MP2 algorithm described in reports of previous studies was designed for parallel calculations on single-CPU core PC clusters. The parallel efficiency falls down on multiple CPU core PCs because multiple processes on a node access the same hard disk drives and network sockets simultaneously. Furthermore, the flat MPI parallelization requires replicated memory spaces and cannot use shared memory space efficiently. To overcome these problems and to achieve better parallel performance on massively parallel multicore supercomputers, MPI/OpenMP hybrid parallelization is necessary.

This article presents an efficient MPI/OpenMP hybrid parallel algorithm for the RI-MP2 calculations suitable for massively parallel computations on the K computer. This article is organized as follows. The MPI/OpenMP hybrid parallel RI-MP2 algorithm is presented in section 2. Benchmark results of MPI/OpenMP hybrid parallel algorithm RI-MP2 are presented and discussed in section 3. Finally, concluding remarks are presented in section 4.

2. MPI/OPENMP HYBRID PARALLEL RI-MP2 ALGORITHM

The RI approximation of atomic orbital (AO) products $\chi_\mu\chi_\nu$ is introduced by expanding atom-center auxiliary basis functions ξ_m as

$$\chi_\mu(\mathbf{r}_1)\,\chi_\nu(\mathbf{r}_1) \approx \sum_m \xi_m(\mathbf{r}_1)c_{m,\mu\nu} \tag{1}$$

where $c_{m,\mu\nu}$ is an expansion coefficient of auxiliary basis functions and is determined by the minimization of the norm

$$(R_{\mu\nu}|R_{\lambda\sigma}) = \int R_{\mu\nu}(\mathbf{r}_1)r_{12}^{-1}R_{\lambda\sigma}(\mathbf{r}_2)\,\mathrm{d}\mathbf{r}_1\,\mathrm{d}\mathbf{r}_2 \tag{2}$$

of the error vector

$$R_{\mu\nu}(\mathbf{r}_1) = \chi_\mu(\mathbf{r}_1)\,\chi_\nu(\mathbf{r}_1) - \sum_m \xi_m(\mathbf{r}_1)c_{m,\mu\nu} \tag{3}$$

The RI approximation of four-center AO ERIs is expressed by three-center and two-center AO ERIs as

$$(\mu\nu|\lambda\sigma) = \sum_{lm} (\mu\nu|l)(l|m)^{-1}(m|\lambda\sigma) \tag{4}$$

The closed-shell MP2 correlation energy in the MO representation is expressed as

$$E^{(2)} = \sum_{ij}^{\mathrm{occ.}}\sum_{ab}^{\mathrm{vir.}} \frac{(ia|jb)[2(ia|jb) - (ib|ja)]}{\varepsilon_i + \varepsilon_i - \varepsilon_a - \varepsilon_b} \tag{5}$$

where i and j are doubly occupied MOs, a and b are virtual MOs, and ε_i and ε_a are orbital energies. By virtue of the RI approximation of AO ERIs, the four-center MO ERIs are approximated as three-center MO ERIs:

$$(ia|jb) = \sum_n B_n^{ia}B_n^{jb} \tag{6}$$

The three-center MO ERIs are transformed from AO ERIs by multiplying with MO coefficients $C_{\mu i}$:

$$B_n^{ia} = \sum_{\mu\nu l} (\mu\nu|l)(l|n)^{-1/2}C_{\mu i}C_{\nu a} \tag{7}$$

This approximation considerably reduces the computational prefactor of MP2 calculations and the storage requirements. The computation cost of RI-MP2 calculations is several times smaller than that of full MP2 calculations and the storage requirement of RI-MP2 calculations is $O(N^3)$, whereas that of full MP2 calculations is $O(N^4)$. The computational bottleneck of RI-MP2 calculations is the generation of four-center MO ERIs by eq 6 requiring the $O(N^5)$ computational costs. Therefore, it is important to speed up the computation of eq 6 for the RI-MP2 calculations of large molecules.

Figure 1 presents the pseudocode of the new MPI/OpenMP hybrid parallel RI-MP2 algorithm. The computational cost, the required memory and disk, and the total amount of network communication are summarized in Table 1. The algorithm consists of three steps: (1) evaluation and 1/3 and 2/3 index transformation of three-center ERIs, (2) 3/3 index transformation of three-center ERIs, and (3) evaluation of four-center MO ERIs and MP2 correlation energy. The outer loops of step 1 (loop l), step 2 (loop A where A is the index of the

dx.doi.org/10.1021/ct400795v | J. Chem. Theory Comput. 2013, 9, 5373−5380

[Step 1]
Loop l (MPI parallelization)
  Loop μ (OpenMP parallelization with dynamic task scheduling)
    Evaluation of three-center AO ERIs $(\mu\nu|l)$
  End Loop μ
  1/3 index transformation of $(\mu\nu|l)$ by threaded DGEMM in BLAS (OpenMP parallelization)
  $(i\nu|l) = \sum_{\mu} C_{\mu i} (\mu\nu|l)$
  2/3 index transformation of $(i\nu|l)$ by threaded DGEMM in BLAS (OpenMP parallelization)
  $(ia|l) = \sum_{\nu} C_{\nu a} (i\nu|l)$
End Loop l

[Step 2]
Loop l (OpenMP parallelization with dynamic task scheduling)
  Evaluation of two-center AO ERIs $(l|m)$
End Loop l
(OpenMP parallelization)
Cholesky decomposition and inversion of $(l|m)$
by threaded LAPACK (OpenMP parallelization)
Loop batch of virtual orbital A (MPI parallelization)
  3/3 index transformation of $(ia|l)$ by threaded DGEMM in BLAS (OpenMP parallelization)
  $B_n^{ia} = \sum_{l} (ia|l)(l|n)^{-1/2} (a \in A)$
End Loop A

[Step 3]
Loop batch of virtual orbitals A (MPI parallelization)
  Read $B_n^{ia} (a \in A)$ in semi-direct algorithm
  Loop b_{Proc}
    Send B_n^{ia} to Myrank- b_{Proc} node
    Receive B_n^{jb} from Myrank+ b_{Proc} node
    Loop $a \in A$ (MPI parallelization)
      Loop $b \in \mathrm{Myrank} + b_{\mathrm{Proc}}$ with $a \geq b$ (MPI parallelization)
        Evaluation of four-center MO ERIs by threaded DGEMM in BLAS (OpenMP parallelization)
        $(ia \mid jb) = \sum_{n} B_n^{ia} B_n^{jb}$
        Evaluation of MP2 correlation energy $E^{(2)}$ (OpenMP parallelization)
      End Loop b
    End Loop a
  End Loop b_{Proc}
End Loop A

Figure 1. Pseudocode for MPI/OpenMP hybrid parallel RI-MP2 algorithm.

batch for virtual MOs a), and step 3 (loops A, a, and b) are parallelized by MPI. Computational procedures inside the MPI parallelized loops are parallelized by OpenMP. The inner loop μ for evaluation of three-center AO ERIs in step 1 is parallelized by OpenMP with dynamic task scheduling. Index transformations in steps 1 and 2 and evaluation of four-center MO ERIs in step 3 can be expressed as matrix−matrix multiplications, and efficient performance of SIMD floating-point operations and thread parallelization are attained by the highly optimized DGEMM subroutine in the basic linear algebra subprograms (BLAS) library.[42] Efficient thread parallelization of inversion and decomposition of the two-center ERI matrix are attained using the optimized Linear Algebra PACKage (LAPACK) library.[43] Because the highly optimized and threaded BLAS and LAPACK libraries are available on modern supercomputers and PC clusters, speedups of RI-MP2 calculations are achieved easily with high throughput.

To attain efficient load balancing for massively parallel calculations, the loops of virtual MOs in steps 2 and 3 are parallelized with MPI. This change makes a considerable improvement on load balancing compared with the flat MPI algorithm[41] developed by the authors previously, where the loops of occupied MOs in steps 2 and 3 are parallelized. Generally, the number of virtual MOs is more than 4 times larger than the number of occupied MOs when double-ζ plus polarization basis functions or a higher level of basis functions are employed.

We have developed a semidirect MPI/OpenMP hybrid parallel algorithm and an in-core MPI/OpenMP hybrid parallel algorithm to attain efficient performance on parallel computers of various kinds. In the semidirect algorithm, the three-center ERIs for which the data size is $O(N^3)$ are stored in the disk. This algorithm is suitable for parallel calculations on PC clusters that have limited memory size, but it has some I/O overheads. In the in-core algorithm, three-center ERIs are stored in the in-core memory. This in-core algorithm requires much more distributed memory space than the semidirect algorithm, but it can avoid the I/O overhead. This algorithm is especially suitable for massively parallel calculations on the K computer. The I/O to local scratch disks from each process considerably lowers the performance on the K computer because the local scratch disks are shared with eight nodes and a conflict of multiple I/O operations to the same disk occurs.

In our MPI/OpenMP hybrid parallel RI-MP2 algorithm, a multiple batch scheme developed in the previous study[41] is

Table 1. Floating Point Operation (FLOP) Count, Required Memory and Disk Sizes, and Total Amount of Network Communication in MPI/OpenMP Hybrid Parallel RI-MP2 Energy Algorithm[a]

step	computation procedure	FLOP count	replicated memory	distributed memory[b] or disk[c]	network
step 1	three-center AO ERI $(\mu\nu\|l)$ evaluation	$O(n^2 n_x)$	Sn^2		
	1/3 integral transformation of $(\mu\nu\|l)$	$O(on^2 n_x)$	Sn^2+n^2+on		
	2/3 integral transformation of $(i\nu\|l)$	$O(ovnn_x)$	$on+n^2+ov$	ovn_x/N_{Proc}	
step 2	two-center AO ERI $(l\|m)$ evaluation	$O(n_x^2)$	n_x^2		
	Cholesky decomposition and inversion of $(l\|m)$	$O(n_x^3)$	n_x^2		
	3/3 integral transformation of $(ia\|l)$	$O(ovn_x^2)$	$oVn_x+n_x^2+on_x$	$2ovn_x/N_{\mathrm{Proc}}$	ovn_x
step 3	four-center MO ERI $(ia\|jb)$ evaluation	$O(o^2v^2n_x)$	$2on_x+o^2$	ovn_x/N_{Proc}	$ovn_xN_{\mathrm{Proc}}/2$
	MP2 correlation energy evaluation	$O(o^2v^2)$	$2on_x+o^2$		

[a] n = the number of AOs, n_x = the number of auxiliary basis functions, o = the number of occupied MOs, v = the number of virtual MOs, S = the number of auxiliary basis functions in a shell, V = the number of virtual MOs in a batch, and N_{Proc} = the number of MPI processes. [b] In-core MPI/OpenMP hybrid parallel algorithm. [c] Semidirect MPI/OpenMP hybrid parallel algorithm.

dx.doi.org/10.1021/ct400795v | J. Chem. Theory Comput. 2013, 9, 5373−5380

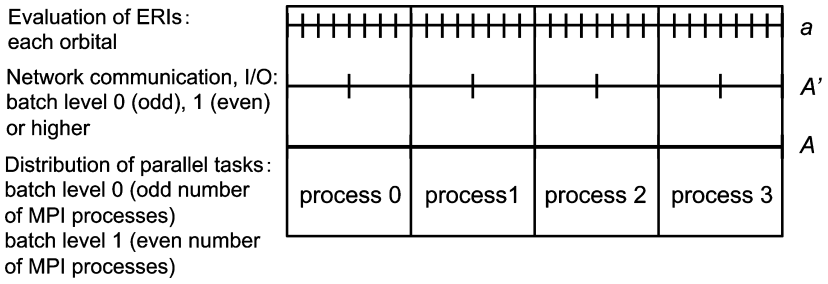


Figure 2. Distribution of batches for virtual MOs based on a multiple batch scheme in the MPI/OpenMP hybrid parallel RI-MP2 algorithm.

applied to attain the efficient load balancing of MPI parallel tasks and the reduction of network and I/O overhead in the computation of steps 2 and 3. Figure 2 presents the structure of batches for virtual MOs used in steps 2 and 3. The MPI parallel tasks are distributed to each node by the uniform sizes of the batch of virtual MOs, defined as batch level 0 or 1 for odd or even numbers of MPI processes, respectively, in Figure 2. In step 3, the batch pairs for virtual MOs (A and B) are statically distributed to each MPI process, as shown in Figure 3, attaining

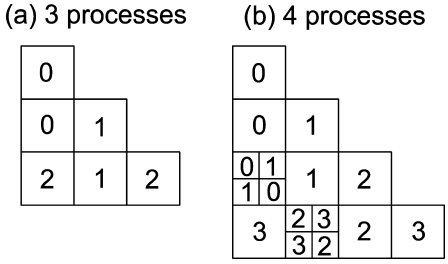


Figure 3. Scheme of distribution of batch pairs for virtual MOs to each MPI process in step 3 of the MPI/OpenMP hybrid parallel RI-MP2 algorithm: (a) three MPI processes, (b) four MPI processes.

uniform task distribution. Square boxes denote the sets of batch pairs assigned on each MPI process, and the number shown in square boxes denotes the index of the process to which the sets of batch pairs are assigned. The size of square boxes corresponds to the number of batches at batch level 0. The numbers of batches A and B in large square boxes are 1 when the number of MPI processes is odd. The numbers of batches A and B in large and small square boxes are 2 and 1, respectively, when the number of MPI processes is even. $N_{Proc}/2$ sets of batch pairs corresponding to large square boxes and two sets of batch pairs corresponding to small square boxes are assigned to each process when the number of MPI processes N_{Proc} is an even number. $(N_{Proc} + 1)/2$ sets of batch pairs corresponding to large square boxes are assigned to each process when the number of MPI processes is odd. The network communications and I/O operations are performed by the smaller size of the batch of virtual MOs defined as batch level p in Figure 2. Batch level p is determined to satisfy the following conditions: (1) $p \geq 0$ and $p \geq 1$ for odd and even numbers of MPI process, respectively; (2) p is determined to be as small as possible to satisfy the condition that the required memory size for the calculation in step 2 ($ovn_x/2^p + n_x^2 + on_x$ where o is the number of occupied MOs, v is the number of virtual MOs, and n_x is the number of auxiliary basis functions) does not exceed the available memory space of each node. By performing the network communications and I/O operations as batches of instructions, their overheads are greatly reduced, and high performance is attained in massively parallel calculations.

We have implemented our parallel RI-MP2 algorithm into the NTChem[44] quantum chemistry software developed by our research team. In this implementation, two-center and three-center AO ERIs are evaluated using the McMurchie–Davidson scheme.[45]

3. RESULTS AND DISCUSSION

3.1. Parallel Performance of MPI/OpenMP hybrid parallel algorithm on RICC PC cluster. First, we investigated the parallel performance of the MPI/OpenMP hybrid parallel algorithm on a commodity-grade Intel Xeon cluster. Benchmark calculations with up to 8192 CPU cores were performed on the massively parallel PC cluster at the RIKEN Integrated Cluster of Clusters (RICC) system. The

Table 2. Elapsed Times of MPI/OpenMP Hybrid and Flat MPI Parallel Calculations of Taxol at RI-MP2/cc-pVDZ Level (1123 AOs, 226 Occupied MOs, 897 Virtual MOs, and 4186 Auxiliary Basis Functions) on RICC PC Cluster (in s)[a]

	MPI/OpenMP hybrid parallel				flat MPI parallel			
number of CPU cores	virtual MO parallel		occupied MO parallel		virtual MO parallel		occupied MO parallel	
8	3012.4	(8.0)	2807.5	(8.0)	2901.3	(8.0)	3067.3	(8.0)
16	1521.3	(15.8)	1346.8	(16.7)	1246.9	(18.6)	1260.6	(19.5)
32	747.8	(32.2)	680.8	(33.0)	637.0	(36.4)	679.9	(36.1)
64	390.4	(61.7)	353.0	(63.6)	335.8	(69.1)	391.4	(62.7)
128	201.0	(119.9)	191.1	(117.5)	193.6	(119.9)	221.5	(110.8)
192	143.4	(168.0)	138.0	(162.8)	160.8	(144.3)	249.9	(98.2)
256	108.5	(222.1)	107.4	(209.2)	132.6	(175.0)	189.1	(129.8)
512	64.0	(376.4)	66.8	(336.0)	118.8	(195.3)	372.0	(66.0)
1024	39.5	(610.4)	47.7	(471.0)	208.1	(111.5)		
2048	36.7	(656.3)	44.1	(509.4)	556.9	(41.7)		

[a]Speedups are shown in parentheses.

dx.doi.org/10.1021/ct400795v | J. Chem. Theory Comput. 2013, 9, 5373−5380

system was a Fujitsu Primergy RX200S5 (1024 nodes, CPU: Intel Xeon 5570, 8 CPU cores/node, memory: 12 GB/node, HDD: 500 GB (RAID0, SAS), Network: InfiniBand ×4 DDR). The NTChem software was compiled using Intel Fortran compiler, version 11.1 and linked with the thread parallelized BLAS and LAPACK libraries supplied by Fujitsu. Eight OpenMP threads per node are generated in the MPI/OpenMP hybrid parallel calculations. cc-pVDZ and cc-pVTZ basis sets[46] and corresponding auxiliary basis sets optimized for electron correlation calculations[47] are used, respectively. All calculations were performed without using the molecular symmetry and the cutoff of three-center AO ERIs.

We tested the effects of MPI/OpenMP hybrid parallelization and changing the scheme of MPI task distribution from occupied MOs to virtual MOs. Table 2 presents the elapsed times and speedups of parallel calculations of a taxol molecule ($C_{47}H_{51}NO_{14}$, the structure is shown in Figure 4a) at the RI-MP2/cc-pVDZ level (1123 AOs, 226 occupied MOs, 897 virtual MOs, and 4186 auxiliary basis functions) using the new MPI/OpenMP hybrid parallel algorithm with MPI parallel task distribution for virtual MOs and the previous flat MPI parallel algorithm with parallel task distribution for occupied MOs. The MPI/OpenMP hybrid parallelization with MPI task distribution for virtual MOs attained the best parallel performance among all parallel algorithms. The MPI/OpenMP hybrid calculations with MPI task distribution for virtual MOs maintain good scaling up to 1024 CPU cores, whereas the flat MPI calculations slow for 256−1024 CPU cores. These results underscore the efficiency of new MPI/OpenMP hybrid parallel algorithms and implementations.

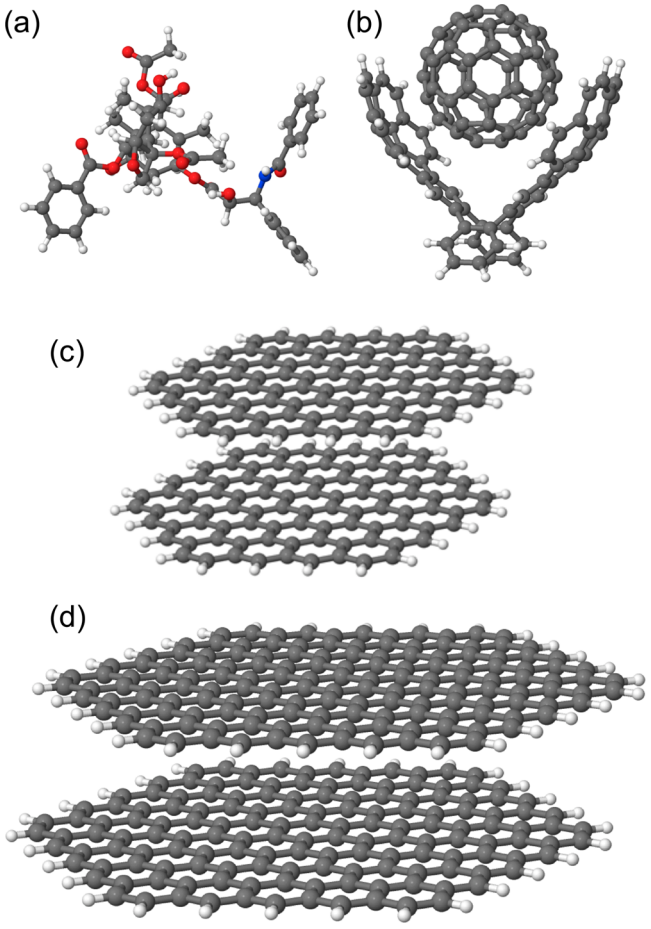


Figure 4. Molecules used for RI-MP2 parallel test calculations: (a) taxol $C_{47}H_{51}NO_{14}$, (b) buckycatcher $C_{60}@C_{60}H_{28}$, (c) two-layer nanographene dimer $(C_{96}H_{24})_2$, and (d) two-layer nanographene dimer $(C_{150}H_{30})_2$.

Table 3 presents the speedups of MPI/OpenMP hybrid parallel calculations of buckycatcher $C_{60}@C_{60}H_{28}$ (the structure is shown in Figure 4b) at the RI-MP2/cc-pVTZ level (3992 AOs, 374 occupied MOs, 3618 virtual MOs, and 10 560 auxiliary basis functions). Virtual MOs are used for the distribution of MPI parallel tasks. The parallel performance is good up to 8192 CPU cores. In fact, a 3343.5 times speedup was attained using 8192 cores. Calculations were finished in 11.9 min using 8192 cores. This result demonstrates that RI-MP2 calculations up to about 4000 AOs are now possible on massively parallel supercomputers based on the commodity Intel architecture for routine applications.

Table 3. Elapsed Times of MPI/OpenMP Hybrid Parallel Calculations of Buckycatcher $C_{60}@C_{60}H_{28}$ at RI-MP2/cc-pVTZ Level (3992 AOs, 374 Occupied MOs, 3618 Virtual MOs, and 10 560 Auxiliary Basis Functions) on RICC PC Cluster (in s)

number of CPU cores	elapsed times[a]	
64	37020.4	(64.0)
128	18423.3	(128.6)
256	8849.7	(267.7)
512	4464.8	(530.7)
1024	2427.7	(975.9)
2048	1389.7	(1704.9)
4096	904.0	(2620.9)
8192	708.6	(3343.5)

[a]Speedups are shown in parentheses.

Table 4 presents the CPU times and the network communication times for each step of RI-MP2/cc-pVTZ calculations of buckycatcher $C_{60}@C_{60}H_{28}$. The most time-consuming step in RI-MP2 calculations is the construction of four-center MO ERIs in step 3, where 71−93% of total computation time was spent. Steps 1 and 2 required less CPU time than step 3:1−3% for step 1 and 2−5% for step 2. CPU times in step 2 (the evaluation of two-center AO ERIs and their Cholesky decomposition and inversion) are not significant (less than 4%) and do not increase with respect to the number of CPU cores because this step is parallelized by LAPACK with OpenMP. The network communication times to transfer three-center ERIs in step 3 increased with respect to the number of CPU cores and became 38% of the total computational time on 8192 cores because the data size of network communications increases with respect to the number of computing processes. This increase of network communications becomes the bottleneck in massively parallel calculations using numerous MPI processes. High network throughput is required to maintain efficient parallel scalability.

3.2. Parallel Performance of MPI/OpenMP Hybrid Parallel RI-MP2 Algorithm on a K Computer. Next, we investigated the parallel performance of the MPI/OpenMP hybrid parallel algorithm on the K computer. Benchmark calculations with up to 71 288 CPU cores were performed on the K computer (CPU, SPARC64 VIIIfx; 8 CPU cores/node; memory, 16 GB/node; Network, Tofu network). NTChem software was compiled with a Fujitsu Fortran compiler and was

dx.doi.org/10.1021/ct400795v | J. Chem. Theory Comput. 2013, 9, 5373−5380

Table 4. CPU Times and Network Communication Times for Each Computational Step in MPI/OpenMP Hybrid Parallel Calculations of Buckycatcher $C_{60}@C_{60}H_{28}$ at RI-MP2/cc-pVTZ Level (3992 AOs, 374 Occupied MOs, 3618 Virtual MOs, and 10 560 Auxiliary Basis Functions) on RICC PC Cluster (in s)[a]

		number of CPU cores											
		256		512		1024		2048		4096		8192	
		computation											
step 1	three-center AO ERI evaluation	194.8	(256.0)	100.3	(497.3)	44.5	(1120.6)	19.6	(2546.1)	7.4	(6722.2)	2.8	(18137.8)
	1/3 integral transformation	48.3	(256.0)	24.2	(511.6)	10.9	(1133.9)	4.7	(2646.6)	2.0	(6273.9)	0.6	(19935.0)
	2/3 integral transformation	42.0	(256.0)	20.7	(519.8)	9.2	(1174.6)	3.9	(2766.0)	1.4	(7796.9)	0.5	(21097.4)
step 2	two-center AO ERI evaluation	18.6	(256.0)	9.3	(510.6)	9.4	(508.4)	9.4	(504.7)	9.3	(510.1)	9.5	(499.9)
	inversion of two-center AO ERI	18.6	(256.0)	18.4	(258.6)	18.4	(258.5)	18.5	(257.2)	18.6	(256.0)	18.6	(256.7)
	3/3 integral transformation	117.3	(256.0)	59.1	(508.1)	30.1	(997.7)	15.6	(1928.8)	8.3	(3631.4)	4.1	(7289.2)
step 3	four-center MO ERI evaluation	7820.0	(256.0)	3937.8	(508.4)	2005.6	(998.2)	1038.5	(1927.8)	554.0	(3613.9)	311.3	(6431.3)
	MP2 correlation energy evaluation	35.5	(256.0)	17.8	(512.4)	9.1	(1003.9)	4.7	(1935.3)	2.6	(3553.0)	1.4	(6316.4)
		network communication											
step 2	2/3 transformed three-center ERIs	7.1	(256.0)	4.9	(373.0)	3.1	(591.0)	3.3	(560.0)	2.2	(842.7)	2.1	(870.9)
step 3	3/3 transformed three-center ERIs	47.2	(256.0)	102.5	(117.9)	121.4	(99.6)	127.4	(94.9)	151.7	(79.7)	227.7	(53.1)

[a]Speedups are shown in parentheses.

linked with the thread-parallelized BLAS and LAPACK libraries supplied by Fujitsu. Eight OpenMP threads per node were generated in the MPI/OpenMP hybrid parallel calculations. The cc-pVDZ and cc-pVTZ basis sets[46] and corresponding auxiliary basis sets optimized for electron correlation calculations[47] were used. All calculations were performed without using the molecular symmetry and the cutoff of three-center AO ERIs. We only investigated the performance of the MPI/OpenMP hybrid parallel algorithm with the MPI task distribution by virtual MOs because this algorithm attained the best performance in the results of section 3.1. We investigated the effects for changing the storage of three-center ERIs from the disk to the in-core memory. FLOPS and the ratio of SIMD instructions were measured using the performance profiler supplied by Fujitsu.

Table 5 presents the speedups of MPI/OpenMP hybrid parallel calculations of taxol at the RI-MP2/cc-pVTZ level (2574 AOs, 226 occupied MOs, 2348 virtual MOs and 6552 auxiliary basis functions). The elapsed times of the in-core storage scheme were 1.8−5.2 times shorter than those of the disk storage scheme because the local scratch disks are shared with eight nodes, and the conflict of I/O operation decreases I/O performance on the K computer. Furthermore, the parallel performance of the in-core storage scheme is better than that of the disk storage scheme.

Table 5. Elapsed Times of MPI/OpenMP Hybrid Parallel Calculations of Taxol at RI-MP2/cc-pVTZ Level (2574 AOs, 226 Occupied MOs, 2348 Virtual MOs, 6552 Auxiliary Basis Functions) on K Computer (in s)[a]

number of CPU cores	in-core algorithm		semidirect algorithm	
256	905.1	(256.0)	1644.6	(256.0)
512	468.9	(494.2)	1308.1	(321.8)
1024	257.8	(898.7)	1347.0	(312.6)
2048	150.3	(1542.1)	673.1	(625.4)
4096	96.0	(2414.4)	414.6	(1015.5)
8192	86.0	(2694.9)	187.2	(2249.1)

[a]Speedups are shown in parentheses.

Table 6 presents the speedups of MPI/OpenMP hybrid parallel calculations of buckycatcher $C_{60}@C_{60}H_{28}$ at the RI-MP2/cc-pVTZ level (3992 AOs, 374 occupied MOs, 3618 virtual MOs, and 10 560 auxiliary basis functions). The parallel performances are good up to 28 952 CPU cores; a 12 436.2 times speedup was attained using 28 952 cores. The RI-MP2/cc-pVTZ calculation of $C_{60}@C_{60}H_{28}$ was finished in 3.4 min using 28 952 CPU cores.

Table 6. Elapsed Times of MPI/OpenMP Hybrid Parallel Calculations of Buckycatcher $C_{60}@C_{60}H_{28}$ at RI-MP2/cc-pVTZ (3992 AOs, 374 Occupied MOs, 3618 Virtual MOs, and 10 560 Auxiliary Basis Functions) on K Computer (in s)

number of CPU cores	elapsed times[a]	
1024	2427.7	(1024.0)
2048	1273.3	(1952.5)
4096	728.6	(3412.0)
8192	449.5	(5531.1)
16 384	291.0	(8543.6)
28 952	199.9	(12436.2)

[a]Speedups are shown in parentheses.

Table 7 presents the CPU times and the network communication times for each step of the RI-MP2/cc-pVTZ calculation of buckycatcher $C_{60}@C_{60}H_{28}$. The most time-consuming step in the RI-MP2 calculation is the construction of four-center MO ERIs in step 3, where 34−84% of total computation time was spent. Steps 1 and 2 spent less CPU time than step 3: 0−5% for step 1 and 2−17% for step 2. CPU times in step 2, which is parallelized only by OpenMP, do not increase with respect to the number of CPU cores, but the ratio

dx.doi.org/10.1021/ct400795v | J. Chem. Theory Comput. 2013, 9, 5373−5380

Table 7. CPU Times and Network Communication Times for Each Computational Step in MPI/OpenMP Hybrid Parallel Calculations of Buckycatcher $C_{60}@C_{60}H_{28}$ at RI-MP2/cc-pVTZ Level (3992 AOs, 374 Occupied MOs, 3618 Virtual MOs, and 10 560 Auxiliary Basis Functions) on K Computer (in s)[a]

		number of CPU cores									
		1024		2048		4096		8192		16 384		28 952	
		computation											
step 1	three-center AO ERI evaluation	108.0	(1024.0)	47.6	(2325.5)	18.2	(6083.7)	6.4	(17335.8)	3.1	(35223.6)	0.8	(131669.3)
	1/3 integral transformation	9.2	(1024.0)	3.9	(2408.6)	1.4	(6740.7)	0.5	(18371.8)	0.3	(32309.0)	0.0[b]	(−)[b]
	2/3 integral transformation	8.2	(1024.0)	3.5	(2411.8)	1.2	(6788.1)	0.5	(18705.1)	0.1	(76520.7)	0.0[b]	(−)[b]
step 2	two-center AO ERI evaluation	23.0	(1024.0)	22.3	(1055.3)	22.6	(1040.8)	22.5	(1044.9)	22.6	(1039.9)	21.7	(1082.5)
	inversion of two-center AO ERI	11.2	(1024.0)	11.2	(1026.7)	11.2	(1027.7)	11.2	(1029.5)	11.4	(1011.4)	11.1	(1035.1)
	3/3 integral transformation	27.4	(1024.0)	14.2	(1980.0)	7.6	(3713.5)	3.8	(7417.2)	1.9	(14834.5)	0.9	(30147.4)
step 3	four-center MO ERI evaluation	2049.3	(1024.0)	1067.4	(1965.9)	571.3	(3673.2)	317.3	(6614.1)	158.9	(13209.5)	68.0	(30855.2)
	MP2 correlation energy evaluation	7.9	(1024.0)	4.2	(1923.4)	2.2	(3613.8)	1.2	(6660.2)	0.6	(13211.3)	0.3	(30995.7)
		network communication											
step 2	2/3 transformed three-center ERIs	1.1	(1024.0)	2.1	(546.5)	1.8	(628.0)	1.9	(611.1)	2.3	(494.2)	2.8	(401.6)
step 3	3/3 transformed three-center ERIs	48.4	(1024.0)	41.2	(1201.6)	48.1	(1030.6)	51.8	(957.1)	47.4	(1045.8)	72.2	(686.3)

[a]Speedups are shown in parentheses. [b]Elapsed times were too short for time measurement.

of CPU times increases with respect to the number of CPU cores: for example, 11% of CPU time is spent for the evaluation of two-center AO ERIs in the case using 28 952 CPU cores. To improve parallel performance, the memory distribution of the two-center ERI matrix to each node and the MPI parallelization with Scalable Linear Algebra PACKage (ScaLAPACK)[48] may be required. Compared with the results of the RICC PC cluster, the ratios of network communication times increase with respect to the number of CPU cores but are smaller than those in Table 4: 12% and 38% of the total computational time on 8192 cores of the K computer and the RICC PC cluster, respectively. This relates to the better performance of the network on the K computer than that on the RICC PC cluster.

Table 8 presents the speedups of the MPI/OpenMP hybrid parallel calculation of the buckycatcher $(C_{96}H_{24})_2$ (the structure is shown in Figure 4c) at the RI-MP2/cc-pVTZ level (6432 AOs, 600 occupied MOs, 5749 virtual MOs, and 16 992 auxiliary basis functions where the linear dependent 83 MOs were removed during the canonical orthogonalization) on the K computer. The parallel performance is good up to 45 992 CPU cores. In fact, a 26 554.3 times speedup was attained using 45 992 cores. The RI-MP2/cc-pVTZ calculation of $(C_{96}H_{24})_2$ was finished in 15.7 min using 45 992 CPU cores.

Table 8. Elapsed Times of MPI/OpenMP Hybrid Parallel Calculations of Nanographene Dimer $(C_{96}H_{24})_2$ at RI-MP2/cc-pVTZ Level (6432 AOs, 600 Occupied MOs, 5749 Virtual MOs, and 16 992 Auxiliary Basis Functions Where 83 Linear Dependent MOs Were Removed during Canonical orthogonalization) on K computer (in s)

number of CPU cores	elapsed times[a]	
8192	3043.7	(8192.0)
16 384	1836.7	(13575.3)
23 000	1295.3	(19250.4)
45 992	939.0	(26554.3)

[a]Speedups are shown in parentheses.

As an example of RI-MP2 parallel calculations of large molecules, we performed a calculation of two-layer stacked nanographene $(C_{150}H_{30})_2$ (the structure is shown in Figure 4d) using 8911 nodes and 71 288 cores of the K computer without symmetry using the cc-pVTZ basis and auxiliary basis functions (9840 AOs, 930 occupied MOs, 8910 virtual MOs, and 26 100 auxiliary basis functions). This calculation is the largest RI-MP2 energy calculation in the world, as we know. The parallel calculation was finished within 64.6 min. This result demonstrates that RI-MP2 calculations of molecules that have about 360 atoms and 10 000 basis functions can be performed routinely on the K computer. A total 43.4% of peak performance (494.2 tera-FLOPS) was attained, and the ratio of SIMD instructions is 75.8% attributable to exploitation of the efficient DGEMM subroutine in BLAS. The weak scaling parallel efficiency obtained with the results of RI-MP2/cc-pVTZ calculations of $(C_{150}H_{30})_2$ using 71 288 cores and $(C_{96}H_{24})_2$ using 8520 cores (elapsed time: 51.7 min) is 0.80. This result demonstrates that the present MPI/Open MP hybrid parallel algorithm attains good parallel performance in the case of massively parallel computation on the next-generation exa-FLOPS supercomputers.

4. CONCLUDING REMARKS

For this study, we developed an MPI/OpenMP hybrid parallel RI-MP2 algorithm that is suitable for massively parallel calculations of large molecular systems on supercomputers. The MPI/OpenMP hybrid parallelization, the scheme of MPI parallel task distribution using virtual MOs, and the in-core storage scheme for three-center ERIs enhance the parallel performance on massively parallel computers. Test calculations using up to 45 992 CPU cores on the K computer and 8192 CPU cores on the RICC Intel Xeon cluster demonstrate the efficiency of the new parallel algorithm. We performed the RI-MP2/cc-pVTZ calculation of a nanographene dimer $(C_{150}H_{30})_2$ containing 360 atoms and 9840 AOs in 65 min on the K computer using 71 288 CPU cores. This result demonstrates that RI-MP2 calculations of molecules that have about 360 atoms and 10 000 basis functions can be performed routinely on peta-FLOPS supercomputers such as the K computer.

The RI-MP2 code developed in this study has been supplied to users of the K computer as a part of NTChem software.[44] It is expected that the MP2 calculation using our code on the K computer will become an effective approach for nanomaterial design and biological application.

AUTHOR INFORMATION

Corresponding Author

*E-mail: nakajima@riken.jp.

Notes

The authors declare no competing financial interest.

ACKNOWLEDGMENTS

This work was supported by the MEXT, Japan, Next-Generation Supercomputer project and the Murata Science Foundation, Overseas study program grant. Calculations were performed on the K computer (RIKEN, Kobe, Japan), the RICC system (RIKEN, Wako, Japan), and the supercomputer system of Research Center for Computational Science (Institute for Molecular Science, Okazaki, Japan).

REFERENCES

(1) Jensen, F. *Introduction to Computational Chemistry*, 2nd ed.; Wiley: New York, 2006.
(2) Møller, C.; Plesset, M. *Phys. Rev.* **1934**, *46*, 618−622.
(3) Cremer, D. *WIREs Comput. Mol. Sci.* **2011**, *1*, 509−530.
(4) Feyereisen, M.; Fitzgerald, G.; Komornicki, A. *Chem. Phys. Lett.* **1993**, *208*, 359−363.
(5) Bernholdt, D. E.; Harrison, R. J. *Chem. Phys. Lett.* **1996**, *250*, 477−484.
(6) Vahtras, O.; Almlöf, J.; Feyereisen, M. W. *Chem. Phys. Lett.* **1993**, *213*, 514−518.
(7) Dunlap, B. I.; Connolly, J. W. D.; Sabin, J. R. *J. Chem. Phys.* **1979**, *71*, 3396−3402.
(8) Yokokawa, M.; Shoji, F.; Uno, A.; Kurokawa, M.; Watanabe, T. *ISLPED 2011* **2011**, 371−372.
(9) TOP500. http://www.top500.org/ (accessed September 8, 2013).
(10) de Jong, W. A.; Bylaska, E.; Govind, N.; Janssen, C. L.; Kowalski, K.; Müller, T.; Nielsen, I. M. B.; van Dam, H. J. J.; Veryazov, V.; Lindh, R. *Phys. Chem. Chem. Phys.* **2010**, *12*, 6896−6920.
(11) Harrison, R. J.; R. Shepard, R. *Annu. Rev. Phys. Chem.* **1994**, *45*, 623−658.
(12) Watts, J. D.; Dupuis, M. *J. Comput. Chem.* **1988**, *9*, 158−170.
(13) Limaye, A. C.; Gadre, S. R. *J. Chem. Phys.* **1994**, *100*, 1303−1307.
(14) Nielsen, I. M. B.; Seidl, E. T. *J. Comput. Chem.* **1995**, *16*, 1301−1313.
(15) Bernholdt, D. E.; Harrison, R. J. *J. Chem. Phys.* **1995**, *102*, 9582−9589.
(16) Márquez, A. M.; Dupuis, M. *J. Comput. Chem.* **1995**, *16*, 395−404.
(17) Nielsen, I. M. B. *Chem. Phys. Lett.* **1996**, *255*, 210−216.
(18) Wong, A. T.; Harrison, R. J.; Rendell, A. P. *Theor. Chim. Acta* **1996**, *93*, 317−331.
(19) Fletcher, G. D.; Rendell, A. P.; Sherwood, P. *Mol. Phys.* **1997**, *91*, 431−438.
(20) Limaye, A. C. *J. Comput. Chem.* **1997**, *18*, 552−561.
(21) Schütz, M.; Lindh, R. *Theor. Chim. Acta* **1997**, *95*, 13−34.
(22) Sosa, C. P.; Ochterski, J.; Carpenter, J.; Frisch, M. J. *J. Comput. Chem.* **1998**, *19*, 1053−1063.
(23) Fletcher, G. D.; Schmidt, M. W.; Gordon, M. S. In *Advances in Chemical Physics*; Prigogine, I., Rice, S. A., Eds.; Wiley: New York, 1999; Vol. *110*, pp 267−294.
(24) Baker, J.; Pulay, P. *J. Comput. Chem.* **2002**, *23*, 1150−1156.
(25) Nakao, Y.; Hirao, K. *J. Chem. Phys.* **2004**, *120*, 6375−6380.
(26) Aikens, C. M.; Gordon, M. S. *J. Phys. Chem. A* **2004**, *108*, 3103−3110.
(27) Ishimura, K.; Pulay, P.; Nagase, S. *J. Comput. Chem.* **2006**, *27*, 407−413.
(28) Ishimura, K.; Pulay, P.; Nagase, S. *J. Comput. Chem.* **2007**, *28*, 2034−42.
(29) Nielsen, I. M. B.; Janssen, C. L. *J. Chem. Theory Comput.* **2007**, *3*, 71−79.
(30) Ford, A. R.; Janowski, T.; Pulay, P. *J. Comput. Chem.* **2007**, *28*, 1215−20.
(31) Lotrich, V.; Flocke, N.; Ponton, M.; Yau, A. D.; Perera, A.; Deumens, E.; Bartlett, R. J. *J. Chem. Phys.* **2008**, *128*, 194104.
(32) Doser, B.; Lambrecht, D. S.; Ochsenfeld, C. *Phys. Chem. Chem. Phys.* **2008**, *10*, 3335−44.
(33) Baker, J.; Wolinski, K. *J. Comput. Chem.* **2011**, *32*, 3304−3312.
(34) Ishimura, K.; Ten-no, S. *Theor. Chem. Acc.* **2011**, *130*, 317−321.
(35) Vogt, L.; Olivares-Amaya, R.; Kermes, S.; Shao, Y.; Amador-Bedolla, C.; Aspuru-Guzik, A. *J. Phys. Chem. A* **2008**, *112*, 2049−2057.
(36) Olivares-Amaya, R.; Watson, M. A.; Edgar, R. G.; Vogt, L.; Shao, Y.; Aspuru-Guzik, A. *J. Chem. Theory Comput.* **2010**, *6*, 135−144.
(37) Watson, M. A.; Olivares-Amaya, R.; Edgar, R. G.; Aspuru-Guzik, A. *Comput. Sci. Eng.* **2010**, *12*, 40−51.
(38) Bernholdt, D. E.; Harrison, R. J. *Chem. Phys. Lett.* **1996**, *250*, 477−484.
(39) Bernholdt, D. E. *Parallel Comput.* **2000**, *26*, 945−963.
(40) Hättig, C.; Hellweg, A.; Köhn, A. *Phys. Chem. Chem. Phys.* **2006**, *8*, 1159−1169.
(41) Katouda, M.; Nagase, S. *Int. J. Quantum Chem.* **2009**, *109*, 2121−2130.
(42) Basic Linear Algebra Subprograms. http://www.netlib.org/blas/ (accessed September 8, 2013).
(43) Linear Algebra PACKage. http://www.netlib.org/lapack/ (accessed September 8, 2013).
(44) NTChem 2013. http://labs.aics.riken.jp/nakajimat_top/ntchem_e.html (accessed October 7, 2013).
(45) McMurchie, L. E.; Davidson, E. R. *J. Comput. Phys.* **1978**, *26*, 218−231.
(46) Dunning, T. H., Jr. *J. Chem. Phys.* **1989**, *90*, 1007−1022.
(47) Weigend, F.; Köhn, A.; Hättig, C. *J. Chem. Phys.* **2002**, *116*, 3175−3183.
(48) Scalable Linear Algebra PACKage. http://www.netlib.org/scalapack/ (accessed October 7, 2013).