JCTC Journal of Chemical Theory and Computation

Article

pubs.acs.org/JCTC

Parallel Implementation of Multireference Coupled-Cluster Theories Based on the Reference-Level Parallelism

Jiří Brabec,[†] Jiří Pittner,[†] Hubertus J. J. van Dam,[‡] Edoardo Aprà,[‡] and Karol Kowalski*[,‡]

[†]J. Heyrovský Institute of Physical Chemistry, Academy of Sciences of the Czech Republic, CZ-18223 Prague 8, Czech Republic

[‡]William R. Wiley Environmental Molecular Science Laboratory, Battelle, Pacific Northwest National Laboratory, K8-91, P.O. Box 999, Richland, Washington 99352, United States

Supporting Information

ABSTRACT: A novel algorithm for implementing a general type of multireference coupled-cluster (MRCC) theory based on the Jeziorski–Monkhorst exponential ansatz [Jeziorski, B.; Monkhorst, H. J. *Phys. Rev. A* **1981**, *24*, 1668] is introduced. The proposed algorithm utilizes processor groups to calculate the equations for the MRCC amplitudes. In the basic formulation, each processor group constructs the equations related to a specific subset of references. By flexible choice of processor groups and subset of reference-specific sufficiency conditions designated to a given group, one can ensure optimum utilization of available computing resources. The performance of this algorithm is illustrated on the examples of the Brillouin–Wigner and Mukherjee MRCC methods with singles and doubles (BW-MRCCSD and Mk-MRCCSD). A significant improvement in scalability and in reduction of time to solution is reported with respect to recently reported parallel implementation of the BW-MRCCSD formalism [Brabec, J.; van Dam, H. J. J.; Kowalski, K.; Pittner, J. *Chem. Phys. Lett.* **2011**, *514*, 347].

1. INTRODUCTION

The dawn of peta-scale computer architectures brings a unique chance of validating and applying accurate yet numerically expensive many-body theories to large molecular systems. The multireference Hilbert-space coupled-cluster methods (MRCC) fall into this category. These methods provide a means to treat properly static correlation effects for electronic states characterized by strong quasidegeneracy effects. Several problems such as the bond-breaking processes, polyradical species, transition metal compounds, low-spin open-shell states, and reactions involving potential energy surfaces crossing require by their very nature a multireference level of description.[1]

Formulations based on the Jeziorski–Monkhorst (JM) ansatz[2] belong to the mainstream Hilbert-space MRCC formalisms, where two major classes of approaches have emerged over the past few decades: State-Universal MRCC (SU-MRCC) approaches[2−11] and class of State-Specific (SS-MRCC) methods.[12−25] While the SU-MRCC approaches are embedded in the Bloch wave operator formalism,[26−29] which furnishes energies of M electronic states, where M is the dimension of the model space ($\mathcal{M}_0$) employed, the SS-MRCC methods, in order to avoid the so-called intruder-state problem,[30−32] were designed to target only one electronic state. The recent generalization of the SU-MRCC formalism to General Model Spaces (GMS-SU-MRCC) has been shown to alleviate significantly the intruder state problem in the context of the SU-MRCC theory.[33−35]

The SS-MRCC methods introduce directly the JM parametrization into the Schrödinger equation, which leads to well-known overcompleteness problems. Several SS-MRCC approaches such as the Brillouin–Wigner MRCC (BW-MRCC) approach,[12] Mk-MRCC methods by Mukherjee et al.,[16] and Hanrath's MRexpT approach[36] address this issue by defining the so-called sufficiency conditions (or equivalently the equations for cluster amplitudes). Generally, the SS-MRCC sufficiency conditions and Bloch equation for the SU-MRCC approaches share several common features: (1) The working equations are the aggregate of reference specific equations. (2) In each reference-specific equation, the so-called direct term, which defines peak numerical complexity, needs to be included, and (3) in each reference-specific equation, the so-called coupling terms (terms coupling the equations with respect to various references) need to be calculated. All three of these factors have to be properly addressed in the process of building efficient parallel implementation of the MRCC methods. While the direct terms do involve only cluster amplitudes specific to a given reference, the coupling terms generally require cluster amplitudes associated with all references, which may impose a significant communication bottleneck for corresponding numerical algorithm.

The SU-MRCCSD implementation of Piecuch and Landman[37] was historically the first parallel implementation of the MRCC theories. Although tests were performed on a relatively small—by today's standards—number of cores, the parallelization tests demonstrated that a significant speedup of the SU-MRCCSD calculations on shared-memory parallel systems having 16−32 CPU can be achieved. A more recent implementation of the Mk-MRCC method[38] allowed for the performance of MRCC calculations for larger systems but still for rather small model spaces. Recently, we reported a Tensor Contraction Engine based implementation of the BW-MRCCSD approach[39] where scalability tests have been performed across thousand of cores. We demonstrated that for simple model spaces involving open shell determinants, BW-MRCCSD calculations are possible for systems described by more than 400 correlated orbitals. Although the parallel performance was between 50 and 60%,

Received: November 11, 2011
Published: January 11, 2012

   dx.doi.org/10.1021/ct200809m | *J. Chem. TheoryComput.* 2012, 8, 487−497

it was shown that general purpose parallel spin-orbital single-reference codes can be used in the context of developing BW-MRCCSD[39] implementations for a general type of complete model spaces (CMS).

In this paper, we would like to discuss and illustrate the performance of the novel algorithm for performing MRCC calculations, which contains three essential components: (1) the use of processor groups to calculate reference specific MRCC equations (or equations corresponding to a subset of references), (2) on-the-fly recognition of the active spin-orbitals, and (3) new algorithms for task scheduling within the processor group. The performance of the new algorithm is illustrated on the examples of the BW-MRCCSD and Mk-MRCCSD approaches, where the two-level parallelism—the reference- and equation-level of parallelism—results in a very good parallel performance on computer architectures characterized by fast networks. The same principles can also be used to implement SU- and Mk-MRCC formalisms. The paper is organized as follows: In section 2, we discuss the basic theoretical threads. In section 3, we give the details of our formalism. Section 4 contains several benchmarks and parallelization tests for systems such as the N_2 molecule, 1,7-naphthyne isomer, and β-carotene.

2. THEORY

In the JM ansatz, one assumes the following expansion for the electronic wave functions (at this point, we will not make distinction between ground and excited states):

$$|\Psi\rangle = \sum_{\mu=1}^{M} c_\mu \, e^{T^{(\mu)}} |\Phi_\mu\rangle \tag{1}$$

where the $\{|\Phi_\mu\rangle\}_{\mu=1}^{M}$ are the reference functions (or determinants) that span the model space $\mathcal{M}_0$ which is used to approximate the essential static correlation effects. In this paper, we will entirely focus on complete model spaces. The operators $T^{(\mu)}$ (μ = 1, ..., M) are the reference-specific cluster operators, which produce a linear combination of the excited configurations from the orthogonal complement of $\mathcal{M}_0$ ($\mathcal{M}_0^\perp$) when acting on the corresponding reference function $|\Phi_\mu\rangle$ (μ = 1, ..., M). In all MRCC approaches, the intermediate normalization of the MRCC wave function is assumed, i.e.,

$$\langle \Phi_\nu | T^{(\mu)} | \Phi_\mu \rangle = 0 \tag{2}$$

which means that $T^{(\mu)}$ operators can produce only configurations from the ($\mathcal{M}_0^\perp$) space when acting on the corresponding $|\Phi_\mu\rangle$ reference function. The c_μ (μ = 1, ..., M) coefficients are components of the right eigenvector to the effective Hamiltonian H^{eff}, whose matrix elements are defined for the CMS as

$$H_{\nu\mu}^{\text{eff}} = \langle \Phi_\nu | (H \, e^{T^{(\mu)}})_C | \Phi_\mu \rangle \tag{3}$$

where subscript C designates a connected part of a given operator expression. The basic MRCCSD approximations (MRCC with single and double excitations) are defined by cluster operators including single ($T_1^{(\mu)}$) and double ($T_2^{(\mu)}$) excitations, i.e.,

$$T^{(\mu)} = T_1^{(\mu)} + T_2^{(\mu)} \tag{4}$$

In the SU- as well as the SS-MRCCSD case, the sufficiency conditions can be generally written as an aggregate of reference-specific equations, i.e.,

$$\mathbf{R}^{(\mu)} = \mathbf{F}^{(\mu)}(T^{(\mu)}) + \mathbf{G}^{(\mu)}(T^{(1)}, ..., T^{(\mu)}, ..., T^{(M)}) = \mathbf{0} \qquad \forall_{\mu=1,...,M} \tag{5}$$

where functional vectors $\mathbf{F}^{(\mu)}$ and $\mathbf{G}^{(\mu)}$ refer to direct and coupling terms, respectively. The $\mathbf{R}^{(\mu)}$ vector is commonly termed the residual vector. For all MRCCSD theories discussed here, the components $\mathbf{F}^{(\mu)}(\theta)$ of the direct term $\mathbf{F}^{(\mu)}$ are equal to

$$F_\theta^{(\mu)}(T^{(\mu)}) = \langle \Phi_\mu(\theta) | (H \, e^{T^{(\mu)}})_C | \Phi_\mu \rangle \tag{6}$$

where $|\Phi_\mu(\theta)\rangle$'s correspond to singly or doubly excited (with respect to $|\Phi_\mu\rangle$) configurations from $\mathcal{M}_0^\perp$ ($|\Phi_i^a(\mu)\rangle$, $|\Phi_{ij}^{ab}(\mu)\rangle$). The algebraic form of the coupling terms is method-specific. While the direct terms depend only on the cluster operator corresponding to a given reference, the coupling terms may involve all possible cluster operators. When the notational convention employed for the direct terms is used, the coupling terms can be expressed as follows:

- SU-MRCCSD method:

$$G_\theta^{(\mu)}(T^{(1)}, ..., T^{(\mu)}, ..., T^{(M)}) = \sum_{\substack{\nu=1 \\ \nu\neq\mu}}^{M} \langle \Phi_\mu(\theta) | \, e^{-T^{(\mu)}} \, e^{T^{(\nu)}} | \Phi_\nu \rangle H_{\nu\mu}^{\text{eff}} \tag{7}$$

- Mk-MRCCSD method:

$$G_\theta^{(\mu)}(T^{(1)}, ..., T^{(\mu)}, ..., T^{(M)}) = \sum_{\substack{\nu=1 \\ \nu\neq\mu}}^{M} \langle \Phi_\mu(\theta) | e^{-T^{(\mu)}} \, e^{T^{(\nu)}} | \Phi_\mu \rangle H_{\mu\nu}^{\text{eff}} \frac{c_\nu}{c_\mu} \tag{8}$$

- BW-MRCCSD method:

$$G_\theta^{(\mu)}(T^{(1)}, ..., T^{(\mu)}, ..., T^{(M)}) = \langle \Phi_\mu(\theta) | (H \, e^{T^{(\mu)}})_{\text{DC,L+UL}} | \Phi_\mu \rangle - E \langle \Phi_\mu(\theta) | e^{T^{(\mu)}} | \Phi_\mu \rangle \tag{9}$$

In eq 9, E corresponds to the eigenvalue of the effective Hamiltonian, whereas "DC,L" and "UL" subscripts in eq 9 refer to disconnected linked and unlinked parts of a given operator expression. The explicit form of the coupling term in the Mk-MRCCSD approach has been derived by Evangelista et al.,[19] while in the SU-MRCCSD case this task is much more challenging. The diagrammatic form of the SU-MRCC coupling terms (eq 7) has been extensively discussed in the context of the general model space formulation of the SU-MRCC theory.[40] The simplest form of the coupling term is employed by the BW-MRCCSD approach, where the only quantity coupling the equations for various reference is energy E, which is also a function of all cluster operators. For the BW-MRCCSD and Mk-MRCCSD approaches, the numerical effort associated with forming residual vectors $\mathbf{R}^{(\mu)}$ is dominated by calculating the direct terms $\mathbf{F}^{(\mu)}(T^{(\mu)})$ with corresponding N^6 numerical complexity per each reference $|\Phi_\mu\rangle$, where N symbolically designates system size. For SU-MRCCSD, the same is true only if a simple approximation of coupling terms is assumed.

Given a large numerical overhead, proportional to $M \times N^6$, efficient parallel codes are needed in order to make the MRCCSD theories applicable to the large systems.

3. PARALLEL ALGORITHM

In this section, we will describe an algorithm which employs two-level parallelism: (1) reference-level parallelism, where each set of $\mathbf{R}^{(\mu)}$ equations (or their aggregate) is calculated on a separate processor group (PG); (2) task-level parallelism, used to calculate a given set of reference-specific equations ($\mathbf{R}^{(\mu)}$). In this paper, we will focus on the BW-MRCCSD and Mk-MRCCSD implementations, although the same ideas can be naturally extended to the SU-MRCCSD case. The use of processor groups has shown to be an efficient approach for the parallelization of electronic structure codes.[41−43] By adopting this coarse grain level of parallelism, it is possible to overcome the scaling limit of finer level parallelism that includes all of the processing elements. However, the use of processor groups is only feasible where the calculation can be easily divided into subunits, each one of them with similar computational cost; among the examples of methods amenable to this kind of parallelization scheme, we can mention methods that require the calculation of energy derivatives by numerical differentiation (needed for geometry optimization, transition state searches, frequencies calculation), reduced scaling methods that divide molecules into fragments, and linear algebra parts of electronic structure periodic codes where "images" are assigned to different groups of processors (e.g., k-point parallelism for diagonalization of a matrix represented in reciprocal space). In our opinion, MRCC methods with their coarse grain level parallelism corresponding to the reference-level parallelization are yet another candidate for an effective use of the processor groups. In the context of the MRCCSD theories, the use of PGs substantially reduces the amount of communication within a given subgroup. We will demonstrate in the remainder of the article that this approach is capable of efficient scaling at a large processor count, overcoming communication bottlenecks present in the task level parallelism.

The MRCC implementations discussed in this paper use in-core algorithms; i.e., all electron integrals, cluster amplitudes, and recursive intermediates are stored in Global Arrays (GA).[44] GA forms an abstraction layer that alleviates the task of the developer by isolating most of the complexities involved in parallelizing software that make use of dense matrices. It relies on several components: a message passing library, the ARMCI one-sided communication library, and a memory allocator (MA library). Most of the NWChem modules (including the ones used for this work) make very little use of MPI, since the majority of the communication is managed by the ARMCI one-sided communication library.

As mentioned earlier, task-level parallelism is associated with performing calculations within each PG and requires parallel implementations for the direct and coupling terms. To understand the details of the parallel MRCCSD implementations, several general remarks on the structure of our Tensor Contracted Engine (TCE)[45] generated codes are necessary. In TCE codes, the whole spin-orbital domains are partitioned into smaller subsets containing the spin-orbitals of the same spatial and spin symmetries. This partitioning into subsets (also called tiles) leads to a block-structure of all tensors representing one- and two-electron integrals as well as cluster amplitudes and recursive intermediates. In the single-reference theories, occupied (unoccupied) tiles are designated as [I], [J], [K], ... ([A], [B], [C], ...). In the multireference case, we will use separate tiling schemes for each reference, which will significantly improve the performance of the MRCCSD implementation. In order to distinguish between tiling schemes corresponding to various references, we will use the notation [I(μ)], [J(μ)], [K(μ)]... and [A(μ)], [B(μ)], [C(μ)]... for occupied and unoccupied tiles in the reference $|\Phi_\mu\rangle$. For example, the singly and doubly excited cluster amplitudes corresponding to this reference are stored in memory using the following block structure:

$$\mathbf{t}^{[\mathrm{I}(\mu)]}_{[\mathrm{A}(\mu)]}(\mu) \equiv \{t^i_a(\mu)\colon i \in [\mathrm{I}(\mu)],\ a \in [\mathrm{A}(\mu)]\} \tag{10}$$

$$\mathbf{t}^{[\mathrm{I}(\mu)][\mathrm{J}(\mu)]}_{[\mathrm{A}(\mu)][\mathrm{B}(\mu)]}(\mu) \equiv \{t^{ij}_{ab}(\mu)\colon i \in [\mathrm{I}(\mu)],\ j \in [\mathrm{J}(\mu)],\ a \in [\mathrm{A}(\mu)],\ b \in [\mathrm{B}(\mu)]\} \tag{11}$$

where the occupied i,j and unoccupied a,b indices (in the sense of reference $|\Phi_\mu\rangle$) belong to tiles [I(μ)], [J(μ)], [A(μ)], and [B(μ)], respectively, and $t^i_a(\mu)$ and $t^{ij}_{ab}(\mu)$ are the antisymmetric cluster amplitudes defining $T_1^{(\mu)}$ and $T_2^{(\mu)}$ operators. Amplitudes from subsets 10 and 11 corresponding to excitations within the model space are set equal to zero (see eq 2). The other tensors corresponding to the integrals and intermediates are represented in a similar way. This data representation provides the necessary granularity ideally suitable for dynamic load balancing. For example, the parallel calculation for the so-called four-particle diagram from the $|\Phi_\mu\rangle$

$$v^{ef}_{ab} t^{ij}_{ef}(\mu)(i < j,\ a < b,\ e < f) \tag{12}$$

is performed using the following scheme, where $\mathbf{v}$ (2-electron integrals) and $\mathbf{t}(\mu)$ tensors are assumed to be distributed on GA across all processes used in a given run

do [A(μ)] = noab(μ) + 1,noab(μ) + nvab(μ)

do [B(μ)] = [A(μ)],noab(μ) + nvab(μ)
do [I(μ)] = 1,noab(μ)
do [J(μ)] = [I(μ)],noab(μ)
dynamic load balancing: each task is assigned to the first available process
create local memory region $\mathbf{r}^{[\mathrm{I}(\mu)][\mathrm{J}(\mu)]}_{[\mathrm{A}(\mu)][\mathrm{B}(\mu)]}$
do [E] = noab(μ) + 1,noab(μ) + nvab(μ)
do [F] = [E],noab(μ) + nvab(μ)
ga_get($\mathbf{t}^{[\mathrm{I}(\mu)][\mathrm{J}(\mu)]}_{[\mathrm{E}(\mu)][\mathrm{F}(\mu)]}$ (μ))
ga_get($\mathbf{v}^{[\mathrm{E}(\mu)][\mathrm{F}(\mu)]}_{[\mathrm{A}(\mu)][\mathrm{B}(\mu)]}$)
$\mathbf{r}^{[\mathrm{I}(\mu)][\mathrm{J}(\mu)]}_{[\mathrm{A}(\mu)][\mathrm{B}(\mu)]}$: =
+ multiply($\mathbf{v}^{[\mathrm{E}(\mu)][\mathrm{F}(\mu)]}_{[\mathrm{A}(\mu)][\mathrm{B}(\mu)]}$ $\mathbf{t}^{[\mathrm{I}(\mu)][\mathrm{J}(\mu)]}_{[\mathrm{E}(\mu)][\mathrm{F}(\mu)]}$ (μ))
enddo
enddo
ga_acc($\mathbf{r}^{[\mathrm{I}(\mu)][\mathrm{J}(\mu)]}_{[\mathrm{A}(\mu)][\mathrm{B}(\mu)]}$) → MRCCSD residual vector for μ-th reference (or appropriate elements of the effective Hamiltonian)
delete local memory region $\mathbf{r}^{[\mathrm{I}(\mu)][\mathrm{J}(\mu)]}_{[\mathrm{A}(\mu)][\mathrm{B}(\mu)]}$
end of load balance assigned task
enddo
enddo
enddo
enddo

dx.doi.org/10.1021/ct200809m | J. Chem. TheoryComput. 2012, 8, 487−497

where ga_get() and ga_acc() are typical one-sided GA calls which fetch from and accumulate to the corresponding global array. Other parts of the code make some use of ga_put() calls. The noab(μ) and nvab(μ) refer to the total number of occupied and unoccupied tiles in reference $|\Phi_\mu\rangle$. The accumulation to the effective Hamiltonian occurs when excitations from [I(μ)], [J(μ)], [A(μ)], [B(μ)] tiles correspond to the internal excitations (i.e., within the model space). If the maximum size of tiles is defined by a parameter tilesize, then the maximum amount of communication required to fetch the **v** and **t** blocks is proportional to 2 × tilesize^4, which in typical situation tilesize = 30 is equivalent to 12.4 MB. This communication upper bound is also valid for other diagrams in the direct and coupling terms.

An integral part of forming MRCCSD equations is related to calculating coupling terms (see eqs 7−9). While for the BW-MRCCSD approach, this is an inexpensive part of the whole calculation, in the Mk-MRCCSD case this constitutes a more challenging task due to the fact that Mk-MRCCSD coupling terms (eq 8) involve amplitudes corresponding to all references, which entails intensive communication. Our implementation utilizes the explicit form of the coupling terms derived for the Mk-MRCCSD approximation by Evangelista et al.,[19] with the singly and doubly excited terms $\langle\Phi_\mu(\theta)|e^{-T(\mu)}\, e^{T(\nu)}|\Phi_\mu\rangle$ given by the expressions:

$$\langle\Phi_i^a(\mu)|e^{-T^{(\mu)}}\, e^{T^{(\nu)}}|\Phi_\mu\rangle = t_a^i(\nu/\mu) - t_a^i(\mu) \tag{13}$$

$$\begin{aligned}\langle\Phi_{ij}^{ab}(\mu)|e^{-T^{(\mu)}}\, e^{T^{(\nu)}}|\Phi_\mu\rangle &= -t_{ab}^{ij}(\mu) + P(ij)\, t_a^i(\mu)\, t_b^j(\mu) \\ &\quad - P(ij)\, P(ab)\, t_a^i(\mu)\, t_b^j(\nu/\mu) + t_{ab}^{ij}(\nu/\mu) \\ &\quad + P(ij)\, t_a^i(\nu/\mu)\, t_b^j(\nu/\mu)\end{aligned} \tag{14}$$

where we employed the notation of ref 19 with the permutation operator $P(ij)$ defined as $P(ij)A^{ij} = A^{ij} - A^{ji}$ and where $t_a^i(\nu/\mu)$ and $t_{ab}^{ij}(\nu/\mu)$ refer to the so-called "common" amplitudes of operator $T^{(\nu)}$ with respect to operator $T^{(\mu)}$ (see ref 19 for details). In our implementation, for each $\mathbf{G}^{(\mu)}$ in the loop over ν references, we first calculate the auxiliary amplitudes $t_a^i(\nu/\mu)$ and $t_{ab}^{ij}(\nu/\mu)$, and then we use a simple TCE generated code to calculate contributions 13 and 14.

The essential part of the associated effort with the task-level parallelism has been recently reported in ref 39 where parallel TCE-based implementation of the BW-MRCCSD method was discussed. The main difference between the implementation reported here and the previous one from ref 39 is the fact that the active spin-orbitals are no longer grouped into small separate tiles, which contributed to significant problems with load balancing. Instead of using separate tiles for active spin-orbitals, these spin-orbitals are included in larger tiles, which make no distinction between active and inactive spin-orbitals. For example, in the BW-MRCCSD and Mk-MRCSSD codes, the components of the $\mathbf{F}_\mu$ vector form a subset of all components corresponding to a larger vector $\mathbf{E}_\mu$, which contains all elements

$$\langle\Phi_\nu|(H\, e^{T^{(\mu)}})_C|\Phi_\mu\rangle,\ \langle\Phi_i^a(\mu)\rangle|(H\, e^{T^{(\mu)}})_C|\Phi_\mu\rangle,\ \langle\Phi_{ij}^{ab}(\mu)\rangle|(H\, e^{T^{(\mu)}})_C|\Phi_\mu\rangle \tag{15}$$

aligned in some order defined by tiling scheme, where the $\langle\Phi_\nu|(H\, e^{T(\mu)})_C|\Phi_\mu\rangle$ corresponds to matrix elements of the effective Hamiltonian (eq 3). Although this new approach results in a more irregular distribution of the elements of the effective Hamiltonian (see Figure 1), the larger size of the tiles leads to a

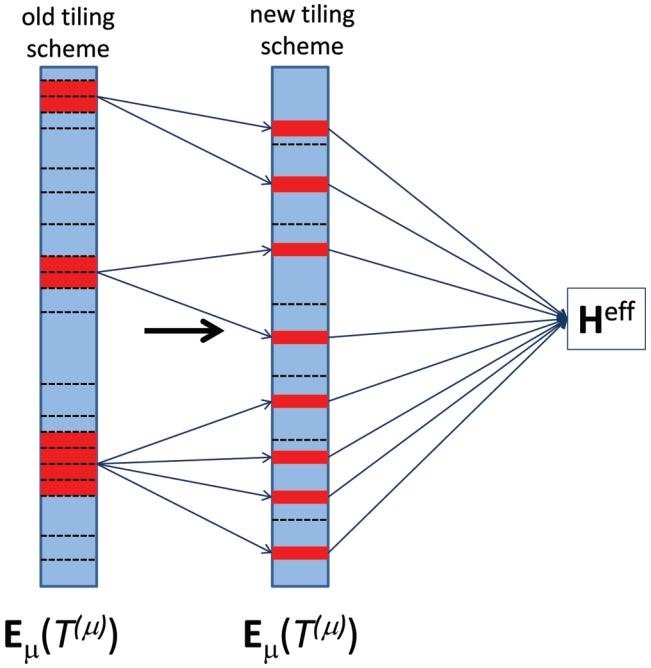

Figure 1. Schematic representation of the new tiling algorithms. The red blocks correspond to the elements of the effective Hamiltonian, while the blue ones correspond to the elements of the $\mathbf{F}_\mu(T^{(\mu)})$ vector. The dashed lines separate different blocks of the $\mathbf{E}_\mu(T^{(\mu)})$ vector induced by a given tiling scheme. The block structure of $\mathbf{E}_\mu(T^{(\mu)})$ in the old tiling scheme is more irregular. In the new tiling scheme, all $\mathbf{E}_\mu(T^{(\mu)})$ blocks are much less diversified sizewise.

better utilization of BLAS procedures[46−48] and substantially improves the load balancing. In analogy to ref 39, up to two-body contributions have been included in the effective Hamiltonian.

To facilitate discussion on the reference-level parallelism let us introduce basic terminology and notation. By processor group (G_i), we will understand a partitioning of the processor domain (D) into smaller pieces, which can be symbolically expressed

$$D = \bigcup_{i=1,\ldots,I} G_i \tag{16}$$

where I is the total number of the processor groups. We will assume that the number of processors in each group is the same and is equal to S, i.e.,

$$S_i = S = \frac{N_p}{I}\ (i = 1, \ldots, I) \tag{17}$$

where S_i is the size of the ith group. In the above equation N_p stands for the total number of processors, which is a multiple of PGs number I. The key idea of the discussed algorithm is to distribute processes of forming $\mathbf{R}^{(\mu)}$ equations over various processor groups. In the simplest case, the work organization chart (symbolically designated by W) corresponds to the situation when a single PG is delegated to calculate a single set of equations $\mathbf{R}^{(\mu)}$. In this case, the number of PGs coincides

dx.doi.org/10.1021/ct200809m | J. Chem. TheoryComput. 2012, 8, 487−497

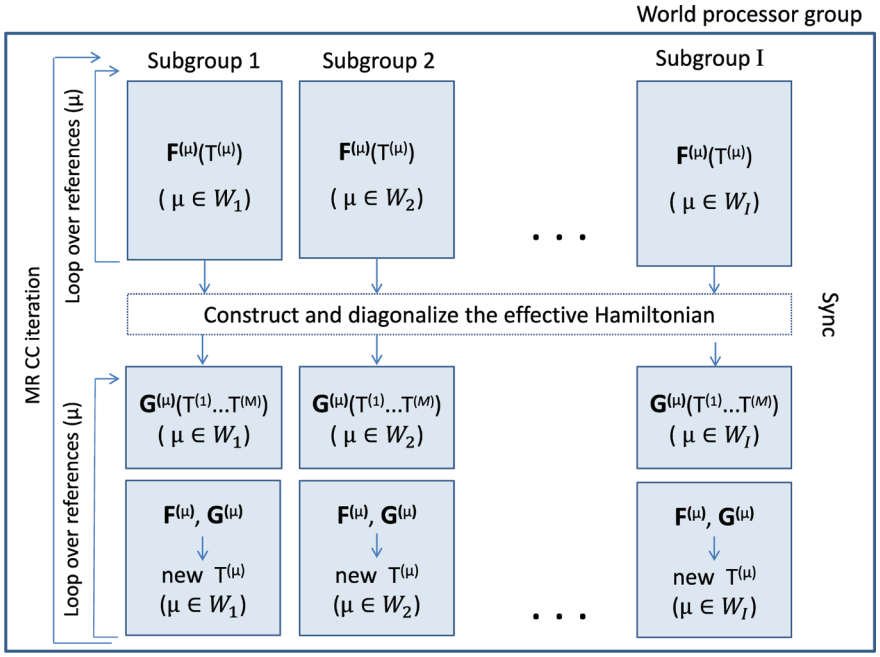

Figure 2. Simplified representation of the processor-group-based MRCC algorithm for the BW-MRCCSD and Mk-MRCCSD formulations.

with the size of the model space (i.e., $I = M$). A more general situation corresponds to the case when each PG (G_i) forms several ($n_r(i)$) residual vectors $\mathbf{R}^{(\mu)}$. This can be symbolically denoted as

$$W = \bigcup_{i=1,\ldots,I} W_i(n_r(i)) \tag{18}$$

where W_i refers to work-load on the corresponding processor group G_i. In order to provide best load balancing between the workloads on each PG, it is natural to assume that

$$n_r = n_r(i) = \frac{M}{I} \quad (i = 1, \ldots, I) \tag{19}$$

In the following, in order to report specific configurations of the processor groups used in the calculation, we will use the $PG(n_r,I,S)$ notation.

The schematic representation of our algorithm is shown in Figure 2. In the first step, the domain D (or the world processor group) is partitioned into I groups $\{G_i\}_{i=1}^{I}$. Each group calculates the corresponding subset of $\mathbf{F}^{(\mu)}$ vectors and the subset of matrix elements of the effective Hamiltonian. In the second step, the effective Hamiltonian is formed and diagonalized to obtain the energy estimate in a given iteration cycle. In the third step, the processor groups are used to calculate coupling terms and to build the updates for the cluster amplitudes. In this process, each processor group works on the cluster amplitudes for the subset of references delegated to a given PG. As a result of the different structure of the sufficiency conditions and the presence of communication-intensive coupling terms in the Mk-MRCCSD approaches, the utilization of the PG and the data organization are slightly different in the BW-MRCCSD and Mk-MRCCSD approaches:

- In BW-MRCCSD, all sufficiency conditions involve only cluster amplitudes relevant to a given reference. Therefore, each PG stores corresponding copies of cluster amplitudes, one- and two-electron integrals, residual vectors, and all recursive intermediates required to calculate the direct term.
- Since the reference-specific Mk-MRCCSD equations require all types of cluster amplitudes, in the Mk-MRCCSD approach, all cluster amplitudes are stored in global arrays belonging to the so-called world group; i.e., they can be accessible by all PGs. Other tensors including one- and two-body integrals, residual vectors, and recursive intermediates of the direct terms are stored only on a corresponding processor group.

For relatively small model spaces discussed here, these schemes seem to provide the most effective solution. The data organization described above determines the communication and memory requirements of our code. One should mention that while the most memory consuming contribution (in the sense of a given PG) is related to the two-electron integrals (of the N^4 size, where N stands for the number of spin-orbitals), the size of the doubly excited cluster amplitudes file and the size of the maximum recursive intermediate used in the direct terms are proportional to $n_o^2 n_u^2$ (where n_o and n_u refer to the numbers of occupied and unoccupied spin-orbitals).

We have two types of synchronization steps: one is internal to PG (internal synchronization), and the other is executed by all processes (global synchronization). While the PG internal synchronization steps are required in calculating direct and coupling terms, the global synchronization steps are required before and after calculating the effective Hamiltonian and before and after calling DIIS solver ("after" synchronizations are required to broadcast the results to all processes). The improved implementation of the direct term (the so-call A2 algorithm of ref 39) requires only four internal PG synchronization steps. In the more laborious coupling terms of the Mk-MRCCSD approach, there are several PG internal synchronization steps corresponding to each reference contributing to a given coupling term.

Several techniques can be used to find the solution of highly nonlinear BW-MRCCSD and Mk-MRCCSD equations. For the BW-MRCCSD method, we used a simple Jacobi style solver, which can easily be extended to the DIIS algorithm.[49,50] In general, the BW-MRCCSD equations are much easier to converge than

dx.doi.org/10.1021/ct200809m | J. Chem. TheoryComput. 2012, 8, 487–497

their Mk-MRCCSD counterparts. Achieving the same accuracy with the Mk-MRCCSD approach poses a more challenging task. In our calculations, we employed a simple iterative scheme: We performed several (usually around 10) Jacobi iterations with the BW-MRCCSD method followed by the DIIS algorithm for the Mk-MRCCSD equations. Usually, this hybrid solver is capable of reducing the number of iterations required to converge the Mk-MRCCSD equations (in situations where the convergence is possible to achieve). In general, however, it is hard to design an efficient solver to the Mk-MRCCSD equations capable of reducing the number of required iterations to the level of single reference CC methods or even of the BW-MRCCSD approach. Moreover, performing more BW-MRCCSD iterations does not always lead to a systematic reduction in the iteration count of the subsequent Mk-MRCCSD solver. We believe that more sophisticated Newton−Raphson-type forms of the preconditioners used in the amplitudes updates may be more efficient in the reduction of total number of iterations. However, this will introduce additional numerical effort associated with approximating the Jacobian matrix in each iteration.

4. COMPUTATIONAL DETAILS

We performed numerical tests for several benchmark systems: from the small N_2 system to larger molecules such as 1,7-naphthyne and β-carotene. The MRCC calculations for N_2, 1,7-naphthyne, and β-carotene have been performed using aug-cc-pVTZ,[51] cc-pVTZ,[51] and 6-31G basis sets,[52] respectively. For the 1,7 naphthyne isomer, we used the geometry of ref 39, while the geometry of the β-carotene was optimized with the DFT B3LYP approach[53] using the cc-pVTZ basis set (the geometries of 1,7-naphthyne and β-carotene and the corresponding Hartree−Fock energies can be found in the Supporting Information). For the N_2 systems, the calculations have been performed for stretched internuclear geometry (4.136 Bohr). All calculations were based on the Hartree−Fock orbitals using C_{2v}, C_s, and C_i symmetry groups for the N_2 molecule, 1,7-naphthyne isomer, and β-carotene, respectively. The following complete model spaces have been used for these systems: CMS(6,6) (N_2), CMS(2,2) (1,7-naphthyne isomer), and CMS(4,4) (β-carotene). These model spaces include the following orbitals: $5a_1$ (HOMO−2), $1b_1$ (HOMO−1), $1b_2$ (HOMO), $2b_2$ (LUMO), $2b_1$ (LUMO+1), and $6a_1$ (LUMO+2) for the N_2 molecule; $28a'$ (HOMO) and $29a'$ (LUMO) for the 1,7-naphthyne isomer; and $74a_g$ (HOMO−1), $74a_u$ (HOMO), $75a_g$ (LUMO), and $75a_u$ (LUMO+1) for β-carotene. The corresponding complete model spaces include 112 (M = 112), 4 (M = 4), and 20 (M = 20) reference functions, respectively. For N_2 and β-carotene, all core orbitals have been kept frozen. Additionally, for the N_2 system, the highest two virtual orbitals were kept frozen.

Our codes have been integrated with the NWChem software.[54] The scalability tests were performed on two computer architectures: for the N_2 and 1,7-naphthyne systems we used the Chinook system at PNNL consisting of 2310 HP DL185 nodes with dual socket, 64-bit, Quad-core AMD 2.2 GHz Opteron processors. Each node has 32 GB of memory, i.e., 4 GB per core. The nodes are connected with a single rail Infiniband interconnect using Voltaire switches and Mellanox network interface cards, whereas for β-carotene we used the Jaguar system at ORNL, which contains 18 688 compute nodes in addition to dedicated login/service nodes. Each XT5 compute node contains dual hex-core AMD Opteron 2435 (Istanbul) processors and 16 GB of memory and uses the Cray proprietary Seastar network.

5. RESULTS AND DISCUSSION

We start our analysis of the parallel performance of the MRCCSD codes with a discussion of the BW-MRCCSD implementation for the N_2 molecule. Although the size of the molecule hardly allows one to categorize it as a "large-molecular system", several interesting performance-related issues can be illustrated with this example. In particular, we can illustrate the performance of the two-level parallelism. In Figure 3, we showed the timings

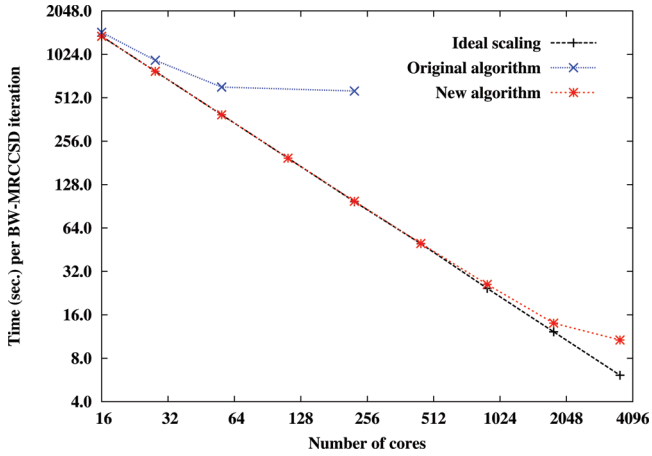

Figure 3. The scalability of the BW-MRCCSD/CMS(6,6) (112 reference functions; M = 112) calculations for the N_2 molecule in the aug-cc-pVTZ basis set (two lowest occupied and two highest unoccupied orbitals were kept frozen in the calculations). The calculations were performed using 16, 28, 56, 112, 224, 448, 896, 1792, and 3584 cores using the following processor groups: PG(7,16,1), PG(4,28,1), PG(2,56,1), PG(1,112,1), PG(1,112,2), PG(1,112,4), PG(1,112,8), PG(1,112,16), and PG(1,112,32), respectively (see text for details).

of the BW-MRCCSD iterations as a function of the number of processes used. In these tests, we used our first version of the BW-MRCCSD approach (version A1 of ref 39). The calculations were performed for several configurations corresponding to different uses of processor groups: PG(7,16,1), PG(4,28,1), PG(2,56,1), PG(1,112,1), PG(1,112,2), PG(1,112,4), PG(1,112,8), PG(1,112,16), PG(1,112,32), where 16, 28, 56, 112, 224, 448, 896, 1792, 3584 cores have been used, respectively. As explained earlier, we use the $P(n_p,I,S)$ specification of the computational setup where n_p, I, and S designate the number of equations per processor group, total number of processor groups, and number of processors in each group, respectively. One can notice that the performance of the original version of the code quickly saturates around 56 cores, which is a consequence of the small size of the system and relatively small task pool when the standard way of executing the BW-MRCCSD sequence of equations is invoked (i.e., serial execution of parallel routines corresponding to direct and coupling terms for various reference functions). Compared to the original algorithm, the use of the processor groups can significantly enlarge the processes' pool that can be effectively utilized in parallel BW-MRCCSD calculations for this system, allowing for efficient calculations up to 3584 cores. This was possible thanks to the flexible use of two-level parallelism. While up to 112 cores mostly reference-level parallelism is

employed, which is associated with calculating various number of reference-specific BW-MRCCSD equations per processor group (see the definition of the n_r parameter), for calculations involving a larger number of cores, the task-level parallelism (or the parallelism of the particular procedures used to calculate the $\mathbf{E}_\mu$ vector) plays the leading role. From 16 to 112 cores, the calculations were performed with one core per processor group, which means that the number of processor groups is equivalent to the total number of cores employed. From 224 cores, the number of cores per processor group starts to increase from two cores per processor group (224 processors run) to 36 cores per processor group (3584 cores run). The good performance from 112 to 3584 should be mostly attributed to the good scalability of the original code in the region of 2−32 cores. This example gives us a general idea how these two levels of parallelism can be intertwined to provide an optimum use of large number of processors.

Next, we analyze the code performance for the larger system, the 1,7-naphthyne isomer. In contrast to the N_2 benchmark, the 1,7-naphthyne isomer is a much larger molecule, while the model space considered is much smaller (M = 4 vs M = 112). By having two σ-radical centers at different atoms, the naphthyne isomers are very interesting systems from the viewpoint of MRCC calculations. Depending on the mutual distance of these radical centers, the strength of the quasi-degeneracy varies and should grow with the distance. The 1,7 isomer (Figure 4) belongs to those which have strong

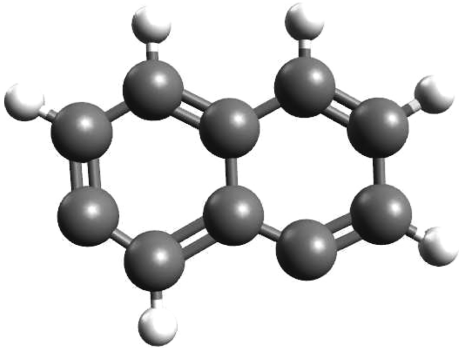

Figure 4. The structure of the 1,7-naphthyne isomer (see text for more details).

multireference character.[55] For the 1,7 isomer, the CMS(2,2) contains four reference functions including the RHF determinant $|\Phi\rangle$, two singly excited determinants $|\Phi_{\mathrm{H}\alpha}^{\mathrm{L}\alpha}\rangle$ and $|\Phi_{\mathrm{H}\beta}^{\mathrm{L}\beta}\rangle$, and a doubly excited configuration $|\Phi_{\mathrm{H}\alpha\mathrm{H}\beta}^{\mathrm{L}\alpha\mathrm{L}\beta}\rangle$, where H and L correspond to the highest occupied and lowest unoccupied orbitals, respectively. In ref 39, we showed that the $|\Phi\rangle$ and $|\Phi_{\mathrm{H}\alpha\mathrm{H}\beta}^{\mathrm{L}\alpha\mathrm{L}\beta}\rangle$ determinants are nearly degenerate in the expansion for the ground state. The scalability tests for the 1,7-naphthyne isomer are collected in Tables 1 and 2 and in Figures 5, 6, and 7. The BW-MRCCSD and Mk-MRCCSD scalability tests for the cc-pVTZ basis set (in spherical representation) have been performed on 1024, 2048, and 4096 cores of the Chinook PNNL system using the PG(4,1,1024), PG(2,2,1024), and PG(1,4,1024) configurations, respectively. In these tests, eight cores of the eight-core node have been used (see Table 1). These results indicate a very good performance of the iterative BW-MRCCSD calculations when moving from 1024 to 4096 cores and using reference-level parallelism. For the same basis set and for the same computational setup, one can compare the

Table 1. BW-MRCCSD and Mk-MRCCSD Timings (per BW-MRCCSD/Mk-MRCCSD iteration) for the 1,7-Naphthyne As Described by the cc-pVTZ Basis Set (In Spherical Representation)[a]

number of cores	PG configuration	time (s)
	BW-MRCCSD	
1024	PG(4,1,1024)	525
2048	PG(2,2,1024)	235
4096	PG(1,4,1024)	86
	Mk-MRCCSD	
1024	PG(4,1,1024)	580
2048	PG(2,2,1024)	290
4096	PG(1,4,1024)	140

[a]We used (2,2) CMS (four reference functions); all electrons were correlated in the course of calculations. The A2 algorithm (improved task scheduling described in ref 39) and improved tiling scheme were used. The scalability tests were performed on the Chinook system at PNNL using eight cores per eight-core node. The calculated BW-MRCCSD energy with a posteriori correction of refs 60 and 61 is equal to −383.84033 Hartree, while the Mk-MRCCSD energy is equal to −383.82863 Hartree.

Table 2. BW-MRCCSD Timings (per BW-MRCCSD Iteration) for the 1,7-Naphthyne As Described by the cc-pVTZ Basis Set (In Spherical Representation)[a]

number of cores	PG configuration	time (s)
1024	PG(4,1,1024)	231
2048	PG(2,2,1024)	118
4096	PG(1,4,1024)	62

[a]We used (2,2) CMS (four reference functions); all electrons were correlated in the course of calculations. The A2 algorithm (improved task scheduling described in ref 39) and improved tiling scheme were used. The scalability tests were performed on the Chinook system at PNNL using four cores per eight-core node. The calculated BW-MRCCSD energy with a posteriori correction of refs 60 and 61 is equal to −383.84033 hartree.

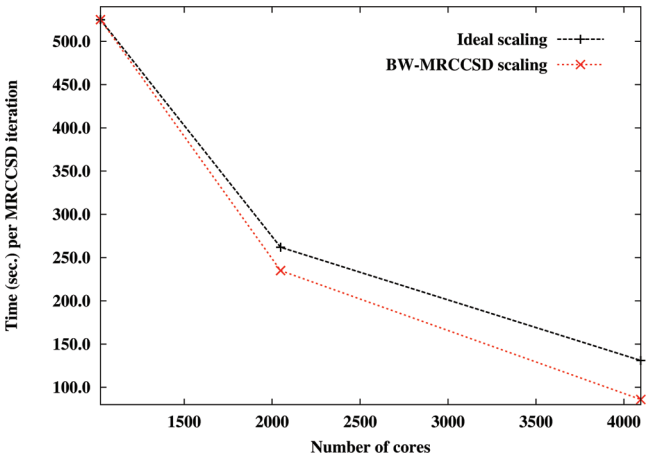

Figure 5. The scalability of the BW-MRCCSD/CMS(2,2) (four reference functions; M = 4) calculations for the 1,7-naphthyne isomer in the spherical cc-pVTZ basis set (all orbitals were correlated in the calculations). The calculations were performed using 1024, 2048, and 4096 cores employing PG(4,1,1024), PG(2,2,1024), and PG(1,4,1024) configurations, respectively (see Table 1 for the corresponding timings). In these tests, we used eight cores per eight-core node of the Chinook system at PNNL.

timings of the BW-MRCCSD and Mk-MRCCSD approaches (see Table 1). The additional cost associated with the inclusion

dx.doi.org/10.1021/ct200809m | J. Chem. TheoryComput. 2012, 8, 487−497

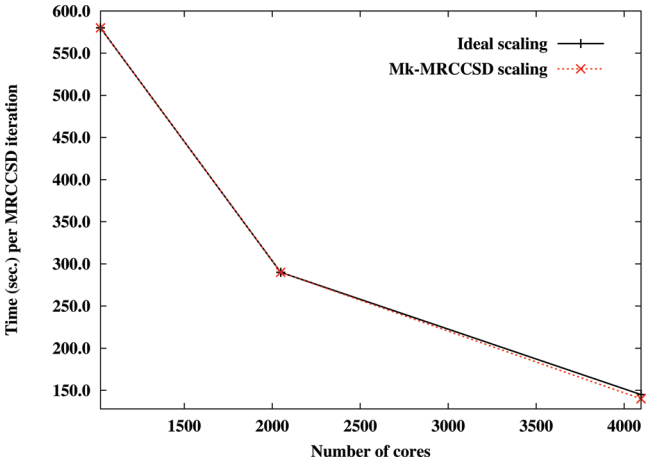

Figure 6. The scalability of the Mk-MRCCSD/CMS(2,2) (four reference functions; $M = 4$) calculations for the 1,7-naphthyne isomer in the spherical cc-pVTZ basis set (all orbitals were correlated in the calculations). The calculations were performed using 1024, 2048, and 4096 cores employing $PG(4,1,1024)$, $PG(2,2,1024)$, and $PG(1,4,1024)$ configurations, respectively (see Table 1 for the corresponding timings). In these tests, we used eight cores per eight-core node of the Chinook system at PNNL.

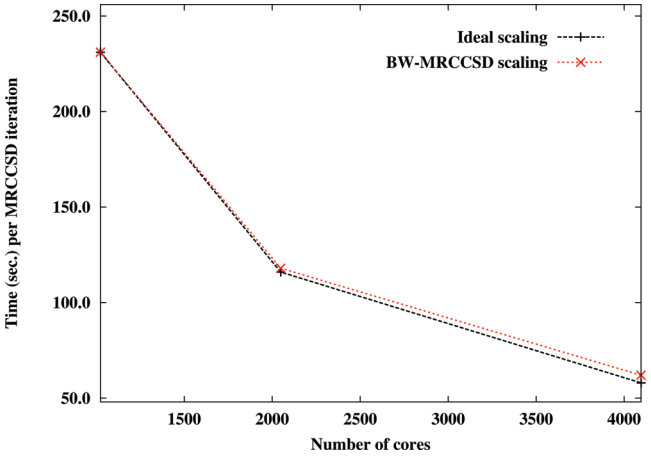

Figure 7. The scalability of the BW-MRCCSD/CMS(2,2) (four reference functions; $M = 4$) calculations for the 1,7-naphthyne isomer in the spherical cc-pVTZ basis set (all orbitals were correlated in the calculations). The calculations were performed using 1024, 2048, and 4096 cores employing $PG(4,1,1024)$, $PG(2,2,1024)$, and $PG(1,4,1024)$ configurations, respectively (see Table 2 for the corresponding timings). In these tests, we used four cores per eight-core node of the Chinook system at PNNL.

of more demanding (from the point of view of communication) coupling terms in the Mk-MRCCSD equation (see eq 8) is a function of the number of cores used. It is interesting to notice that for each case (1024, 2048, and 4096 cores) the timing per Mk-MRCCSD iteration is by ≃55 s bigger compared to the corresponding timings of the BW-MRCCSD iterations. At the same point, the scalability of the Mk-MRCCSD code for the 1,7-naphthyne isomer remains still very good. A different computational setup—four cores per eight-core node on the Chinook system—resulted in a significant reduction of the time per BW-MRCCSD iteration. For 1024 and 2048 cores, one can observe a 2-fold reduction of time per BW-MRCCSD iteration (see Table 2 and Figure 7). This improvement in performances by using a reduced number of cores per node can be attributed to the following: (i) ARMCI (the low level communication library used by GA) is implemented on several high-performance networks (including Infiniband and Seastar) using a communication helper thread that burns resources.[56] (ii) The limited amount of memory available on each core (e.g., 1.3 GB on Jaguar Cray XT5); therefore using fewer cores one obtains a larger amount of memory per core. (iii) When network resources are shared among multiple cores, communication performance might see a degradation, more likely seen in latency (see Figure 4b of ref 57). However, timings of the BW-MRCCSD runs shown in Tables 1 and 2 suggest that the use of eight cores per node is preferable from the point of view of efficient utilization of the computational resources since the time to solution is reduced. For example, the 86 s per BW-MRCCSD iteration when 512 nodes are used in an eight-core per node run (for a total of 4096 processors, see Table 1) should be compared with the 118 s per BW-MRCCSD iteration when the same number of nodes is used, but only four cores are utilized on each node (2048 cores, see Table 2).

The last remark on the 1,7-naphthyne isomer concerns the performance of the iterative solvers for the BW-MRCCSD and Mk-MRCCSD approaches. A typical situation is shown in Figure 8 where the discrepancies between exact MRCCSD

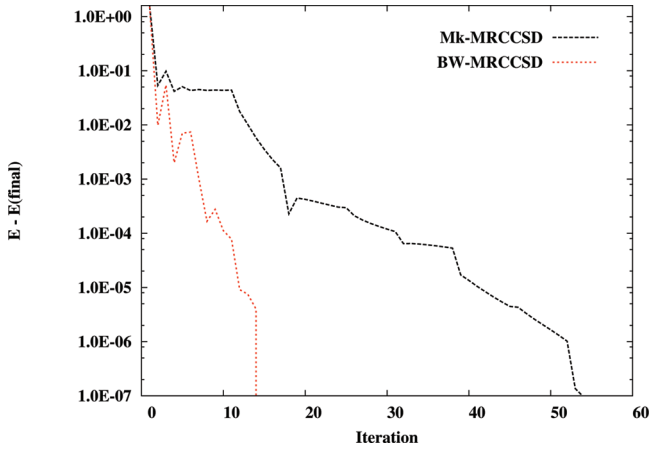

Figure 8. Comparison of the energy convergence characteristic during the BW-MRCCSD and Mk-MRCCSD iterations for the 1,7-naphthyne isomer in the cc-pVTZ basis set (in the spherical representation). All core electrons were kept frozen in the calculations.

energy and energies obtained for particular iterations are shown for calculations for the 1,7-naphthyne isomer in the cc-pVTZ basis set (spherical representation). The application of the DIIS procedure to the BW-MRCCSD iterations (DIIS cycle length was set equal to 5) resulted in 13 iterations required to converge energy to the 10^{-7} Hartree threshold. For the Mk-MRCCSD approach, which is notorious for convergence problems, we used a slightly different strategy. First, we let the code perform 10 BW-MRCCSD iterations without DIIS support. Then, the full form of the Mk-MRCCSD equations was switched back on, and the remaining iterations were performed with the help of the DIIS algorithm. In the Mk-MRCCSD case, we found that the use of a longer DIIS cycle may be beneficial for accelerating the convergence. In the discussed calculations, it took 53 iterations to converge the Mk-MRCCSD energy to within 10^{-7} Hartree. When discussing the iterative processes, it is also instructive to compare the structure

dx.doi.org/10.1021/ct200809m | J. Chem. TheoryComput. 2012, 8, 487–497

of the right eigenvectors of the effective Hamiltonians obtained with the BW-MRCCSD (with a posteriori correction) and Mk-MRCCSD methods. Both approaches yield similar eigenvectors (0.774, −0.624, −0.076, −0.076) for BW-MRCCSD and (0.796, −0.593, −0.090, −0.090 for Mk-MRCCSD) where the components corresponds to the {|20|, |02|, |αβ|, |βα|} reference determinants (where we show occupancies of active orbitals only; for example, |20| corresponds to the determinant with doubly occupied active orbital 28a' while |αβ| corresponds to the determinant with α and β electrons on active orbitals 28a' and 29a', respectively). The structure of these eigenvectors shows the importance of the "open-shell" configurations |αβ| and |βα| in the wave function expansion.

Our last example is related to the scalability tests for the β-carotene molecule (see Figure 9). The choice of this system

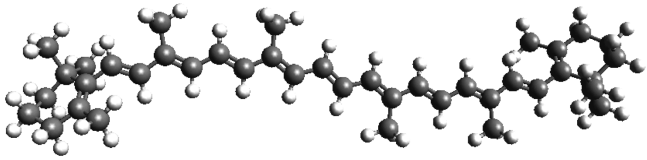

Figure 9. The structure of the β-carotene. The geometry optimization was performed using DFT B3LYP approach using cc-pVTZ basis set.

is dictated by the fact that the proper description of excited states of carotenoids poses a significant challenge for theoretical approaches and requires multireference methods to describe complicated low-lying excited states.[58,59] However, for systems of this size, the MRCC calculations are very expensive. In the present paper, we would like to demonstrate that the MRCC calculations (in the BW-MRCCSD variant) are possible at least for the ground state (in the future, we are planning to perform excited-state studies using the SU-MRCCSD method, which is characterized by a similar numerical complexity as the BW-MRCCSD method). In all of our numerical tests, we used Processor Groups including 1200 processors (S = 1200); therefore, these tests, in contrast to the N_2 tests, mainly reflect the reference-level parallelism. The results of our simulations are shown in Table 3 and in Figure 10, where we included

Table 3. Timings for β-Carotene in 6-31G Basis Set[a]

number of cores	PG configuration	time (s)
6000	PG(4,5,1200)	2710
12000	PG(2,10,1200)	1216
24000	PG(1,20,1200)	660

[a]The (4,4) CMS was used (20 reference functions), and all core electrons were kept frozen in calculations. In our tests, we used version A2 of ref 39 based on the improved tiling scheme described in the Parallel Algorithm section. All calculations were performed on Cray XT5 system at ORNL. The calculated BW-MRCCSD energy with a posteriori correction of refs 60 and 61 is equal to −1550.6189 hartree.

timings for 6000, 12 000, and 24 000 core runs. In these calculations, four (n_r = 4), two (n_r = 2), and one (n_r = 1) reference-specific equations were generated on each group, respectively. Moreover, these timings were obtained with the A2 algorithm based on a new tiling scheme (differences in timings between the A2 algorithm using old and new tiling schemes are significant; for example, the old tiling scheme is characterized by ≃1900 s per iteration on 32 000 cores when 20 processor groups are used). From Table 3, one can notice that parallel

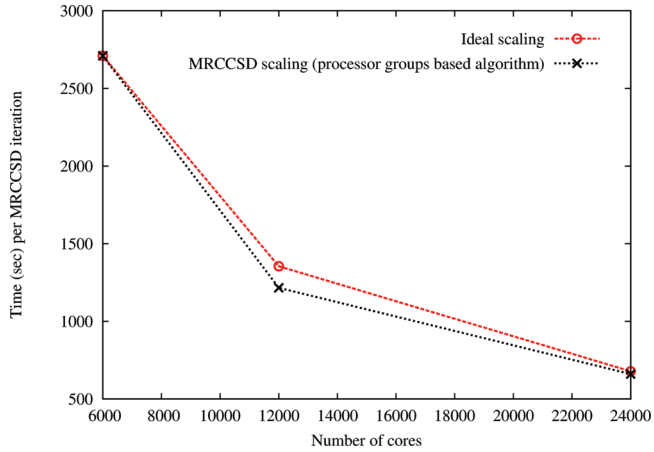

Figure 10. The scalability of the BW-MRCCSD/CMS(4,4) (20 reference functions, M = 20) calculations for the 1,7-naphthyne isomer in the cc-pVTZ basis set (all core orbitals were kept frozen in the calculations). The calculations were performed using 6000, 12 000, and 24 000 cores employing PG(4,5,1200), PG(2,10,1200), and PG(1,20,1200) configurations, respectively. In these tests, we used six cores per 12-core node of the Jaguar system at ORNL.

efficiency reaches superlinear 102% when 6000 and 24 000 jobs are compared. The results of Table 3 suggest that this efficiency is reduced when larger jobs are compared. For example, the efficiency of 24 000 and 12 000 core jobs drops down to 92%, which is a more realistic estimate of parallel performance of the BW-MRCCSD method when a large number of cores is used. We believe that these results provide convincing arguments for using the processor groups as an efficient tool for building efficient MRCCSD implementations. The structure of the right eigenvector of the BW-MRCCSD effective Hamiltonian (again, the BW-MRCCSD amplitudes include the a posteriori correction) shows that the dominant contributions stem from the following reference determinants: |2200| (−0.925), |2002| (0.194), |0202| (0.157), |0220| (0.120), |αβαβ| (−0.117), and |βαβα| (−0.117), which demonstrated mildly multireference character of the ground state of the β-carotene molecule.

6. CONCLUSIONS

The development of efficient parallel implementations of the MRCCSD methods remains one of the most important factors in enabling high-precision calculations for strongly correlated systems. The utilization of novel parallel tools such as the processor groups provides a means to significantly extend the area of application of two MRCCSD approaches: the Brillouin−Wigner and Mukherjee MRCCSD methods. Using two-level parallelism we demonstrated that when two degrees of freedom (reference-level parallelism and task-level parallelism) are combined, one can perform large-scale MRCCSD calculations with efficiency exceeding 90%. In particular, for the β-carotene molecule described by the 6-31G basis set (472 orbitals, 216 correlated electrons, 20 reference functions), the time per single BW-MRCCSD can be reduced to 660 s when 24 000 cores are used. The preliminary tests also show a very good scalability of the Mk-MRCCSD methods, which due to the presence of coupling terms introduce a significant amount of interprocessor communication compared to the BW-MRCCSD approach. Despite this fact, in tests employing our processor-groups-based implementation of the Mk-MRCCSD approach, no deterioration of the parallel performance has been

observed. We believe that our algorithm will pave the way for even larger MRCC simulations employing several hundred reference functions. In the future, special attention will be given to the development of the SU-MRCCSD approaches, enabling accurate excited-state calculations for complicated excited states. In this aspect, the development of the SU-MRCCSD implementations for the General Model Space utilizing the C-condition emerges as one of the most pressing tasks.

■ ASSOCIATED CONTENT

Supporting Information

Geometries of the β-carotene and 1,7-naphthyne isomer. This material is available free of charge via the Internet at http://pubs.acs.org.

■ AUTHOR INFORMATION

Corresponding Author

*E-mail: karol.kowalski@pnnl.gov.

Notes

The authors declare no competing financial interest.

■ ACKNOWLEDGMENTS

This work has been supported by the Extreme Scale Computing Initiative (J.B., H.J.J.v.D., K.K.), a Laboratory Directed Research and Development Program at Pacific Northwest National Laboratory. A large portion of the calculations have been performed using EMSL, a national scientific user facility sponsored by the Department of Energy's Office of Biological and Environmental Research and located at Pacific Northwest National Laboratory. The Pacific Northwest National Laboratory is operated for the U.S. Department of Energy by the Battelle Memorial Institute under Contract DE-AC06.76RLO-1830. This work has also been supported by the Grant Agency of the Czech Republic (GACR Project No. 208/11/2222) and the Grant Agency of Charles University (GAUK 43.252494). The largest scalability tests of the BW-MRCCSD implementation in NWChem have been performed on the Jaguar Cray-XT5 computer system of the National Center for Computational Sciences at Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.

■ REFERENCES

(1) Paldus, J.; Pittner, J.; Čársky, P. In *Recent Progress in Coupled Cluster Methods*; Čársky, P.; Paldus, J.; Pittner, J., Eds.; Springer: Berlin, 2010; pp 455−490.
(2) Jeziorski, B.; Monkhorst, H. J. *Phys. Rev. A* **1981**, *24*, 1668−1681.
(3) Jeziorski, B.; Paldus, J. *J. Chem. Phys.* **1988**, *88*, 5673−5687.
(4) Paldus, J.; Piecuch, P.; Pylypow, L.; Jeziorski, B. *Phys. Rev. A* **1993**, *47*, 2738−2782.
(5) Piecuch, P.; Paldus, J. *Phys. Rev. A* **1994**, *49*, 3479−3514.
(6) Piecuch, P.; Paldus, J. *J. Chem. Phys.* **1994**, *101*, 5875−5890.
(7) Kucharski, S. A.; Bartlett, R. J. *J. Chem. Phys.* **1991**, *95*, 8227−8238.
(8) Balková, A.; Kucharski, S. A.; Meissner, L.; Bartlett, R. J. *Theor. Chim. Acta* **1991**, *80*, 335−348.
(9) Balková, A.; Kucharski, S. A.; Meissner, L.; Bartlett, R. J. *J. Chem. Phys.* **1991**, *95*, 4311−4316.
(10) Kowalski, K.; Piecuch, P. *Mol. Phys.* **2004**, *102*, 2425−2449.
(11) Pittner, J.; Piecuch, P. *Mol. Phys.* **2009**, *107*, 1362−3028.
(12) Hubač, I. In *New Methods in Quantum Theory*; Tsipis, A., Popov, V. S., Herschbach, D. R., Avery, J. S., Eds.; NATO ASI Series 3: High Technology; Kluwer: Dordrecht, The Netherlands, 1996; Vol. *8*, pp 183−202.
(13) Mášik, J.; Hubač, I. *Adv. Quantum Chem.* **1998**, *31*, 75−104.
(14) Pittner, J.; Nachtigall, P.; Čársky, P.; Mášik, J.; Hubač, I. *J. Chem. Phys.* **1999**, *110*, 10275−10282.
(15) Pittner, J. *J. Chem. Phys.* **2003**, *118*, 10876−10889.
(16) Mahapatra, U. S.; Datta, B.; Mukherjee, D. *J. Chem. Phys.* **1999**, *110*, 6171−6188.
(17) Mahapatra, U. S.; Datta, B.; Mukherjee, D. *Chem. Phys. Lett.* **1999**, *299*, 42−50.
(18) Evangelista, F. A.; Allen, W. D.; Schaefer, H. F. III. *J. Chem. Phys.* **2006**, *125*, 154113.
(19) Evangelista, F. A.; Allen, W. D.; Schaefer, H. F. III. *J. Chem. Phys.* **2007**, *127*, 024102.
(20) Bhaskaran-Nair, K.; Demel, O.; Pittner, J. *J. Chem. Phys.* **2008**, *129*, 184105.
(21) Bhaskaran-Nair, K.; Demel, O.; Pittner, J. *J. Chem. Phys.* **2010**, *132*, 154105.
(22) Demel, O.; Bhaskaran-Nair, K.; Pittner, J. *J. Chem. Phys.* **2010**, *133*, 134106.
(23) Bhaskaran-Nair, K.; Demel, O.; Šmydke, J.; Pittner, J. *J. Chem. Phys.* **2011**, *134*, 154106.
(24) Das, S.; Mukherjee, D.; Kállay, M. *J. Chem. Phys.* **2010**, *132*, 074103.
(25) Mahapatra, U. S.; Chattopadhyay, S. *J. Chem. Phys.* **2010**, *133*, 074102.
(26) Bloch, C. *Nucl. Phys.* **1958**, *6*, 329−347.
(27) Jorgensen, F. *Mol. Phys.* **1975**, *29*, 1137−1164.
(28) Soliverez, C. E. *Phys. Rev. A* **1981**, *24*, 4−9.
(29) Durand, P. *Phys. Rev. A* **1983**, *28*, 3184−3192.
(30) Schucan, T. H.; Weidenmüller, H. A. *Ann. Phys.* **1972**, *73*, 108−135.
(31) Schucan, T. H.; Weidenmüller, H. A. *Ann. Phys.* **1973**, *76*, 483−509.
(32) Kowalski, K.; Piecuch, P. *Phys. Rev. A* **2000**, *61*, 052506.
(33) Li, X.; Paldus, J. *J. Chem. Phys.* **2003**, *119*, 5320−5333.
(34) Li, X.; Paldus, J. *J. Chem. Phys.* **2003**, *119*, 5334−5345.
(35) Li, X.; Paldus, J. *J. Chem. Phys.* **2003**, *119*, 5346−5357.
(36) Hanrath, M. *J. Chem. Phys.* **2005**, *123*, 084102.
(37) Piecuch, P.; Landman, J. I. *Parallel Comput.* **2000**, *26*, 913−943.
(38) Prochnow, E.; Harding, M. E.; Gauss, J. *J. Chem. Theory Comput.* **2010**, *6*, 2339−2347.
(39) Brabec, J.; Krishnamoorthy, S.; van Dam, H. J. J.; Kowalski, K.; Pittner, J. *Chem. Phys. Lett.* **2011**, *514*, 347−351.
(40) Paldus, J.; Li, X.; Petraco, N. D. *J. Math. Chem.* **2004**, *35*, 215−251.
(41) Fedorov, D. G.; Olson, R. M.; Kitaura, K.; Gordon, M. S.; Koseki, S. *J. Chem. Phys.* **2004**, *25*, 872−880.
(42) Hutter, J.; Curioni, A. *ChemPhysChem* **2005**, *6*, 1788−1793.
(43) Giannozzi, P.; et al. *J. Physics: Condens. Matter* **2009**, *21*, 395502.
(44) Nieplocha, J.; Palmer, B.; Tipparaju, V.; Krishnan, M.; Trease, H.; Apra, E. *Int. J. High Perform. Comput. Appl.* **2006**, *20*, 203−231.
(45) Hirata, S. *J. Phys. Chem. A* **2003**, *107*, 9887−9897.
(46) Dongarra, J. *International Journal of High Performance Computing Applications* **2002**, *16*, 115−199.
(47) Dongarra, J. *Int. J. High Perform. Comput. Appl.* **2002**, *16*, 1−111.
(48) Blackford, L. S.; Demmel, J.; Dongarra, J.; Duff, I.; Hammarling, S.; Henry, G.; Heroux, M.; Kaufman, L.; Lumsdaine, A.; Petitet, A.; Pozo, R.; Remington, K.; Whaley, R. C. *ACM Trans. Math. Softw.* **2002**, *28*, 135−151.
(49) Pulay, P. *Chem. Phys. Lett.* **1980**, *73*, 393−398.
(50) Pulay, P. *J. Comput. Chem.* **1982**, *3*, 556−560.
(51) Dunning, T. Jr. *J. Chem. Phys.* **1989**, *90*, 1007−1023.
(52) Hehre, W. J.; Ditchfield, R.; Pople, J. A. *J. Chem. Phys.* **1972**, *56*, 2257−2261.
(53) Stephens, P. J.; Devlin, F. J.; Chabalowski, C. F.; Frisch, M. J. *J. Phys. Chem.* **1994**, *98*, 11623−11627.

(54) Valiev, M.; Bylaska, E. J.; Govind, N.; Kowalski, K.; Straatsma, T. P.; Van Dam, H. J. J.; Wang, D.; Nieplocha, J.; Apra, E.; Windus, T. L.; de Jong, W. *Comput. Phys. Commun.* **2010**, *181*, 1477−1489.
(55) Li, X. Z.; Paldus, J. *Can. J. Chem.—Rev. Can. Chim.* **2009**, *87*, 917−926.
(56) Tipparaju, V.; Aprá, E.; Yu, W.; Vetter, J. S. *Proceedings of the 7th ACM International Conference on Computing Frontiers*; CF '10; ACM: New York, 2010; pp 207−216.
(57) Alam, S.; Barrett, R.; Kuehn, J.; Poole, S. *Parallel Distributed Processing, 2009. IPDPS 2009. IEEE International Symposium*; IEEE: New York, 2009; pp 1−8.
(58) Kleinschmidt, M.; Marian, C. M.; Waletzke, M.; Grimme, S. *J. Chem. Phys.* **2009**, *130*, 044708.
(59) Ceron-Carrasco, J. P.; Requena, A.; Marian, C. M. *Chem. Phys.* **2010**, *373*, 98−103.
(60) Hubač, I.; Wilson, S. *Brillouin-Wigner Methods for Many-Body Systems*; Springer: Berlin, 2010; Vol. *21*.
(61) Hubač, I.; Pittner, J.; Čársky, P. *J. Chem. Phys.* **2000**, *112*, 8779−8784.