JCTC Journal of Chemical Theory and Computation

Article

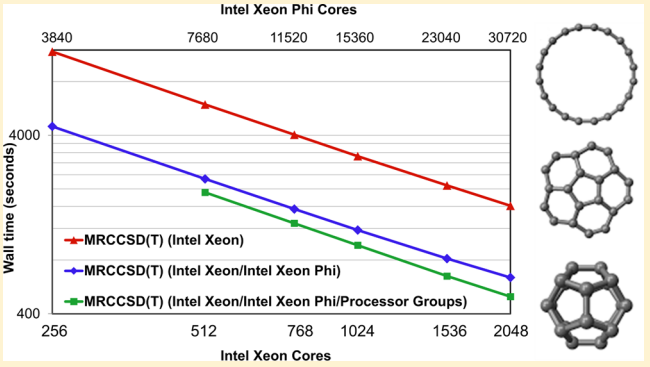
pubs.acs.org/JCTC

Implementation of High-Order Multireference Coupled-Cluster Methods on Intel Many Integrated Core Architecture

E. Aprà* and K. Kowalski*

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

ABSTRACT: In this paper we discuss the implementation of multireference coupled-cluster formalism with singles, doubles, and noniterative triples (MRCCSD(T)), which is capable of taking advantage of the processing power of the Intel Xeon Phi coprocessor. We discuss the integration of two levels of parallelism underlying the MRCCSD(T) implementation with computational kernels designed to offload the computationally intensive parts of the MRCCSD(T) formalism to Intel Xeon Phi coprocessors. Special attention is given to the enhancement of the parallel performance by task reordering that has improved load balancing in the noniterative part of the MRCCSD(T) calculations. We also discuss aspects regarding efficient optimization and vectorization strategies.



1. INTRODUCTION

The rapid development of computer technology and emergence of peta-scale hybrid architectures offer an enormous increase in computational power, which will have a transformative effect on the whole area of computational chemistry, including the shift in the system-size limit tractable by many-body methods and opportunity of using accurate many-body techniques which are characterized by steep numerical scaling. However, without thorough modernization of underlying codes toward efficient utilization of coprocessors or accelerators, a significant portion of the computational power offered by the current and emerging hybrid architecture will be forfeited. To address this challenge, in the past few years one could witness an intensive effort associated with the development of scientific high-performance software for the NVIDIA GPU and Intel Many Integrated Core architecture (Intel MIC).[1−3] In computational chemistry this effort embraces the development of a wide variety of methodologies ranging from molecular mechanics and molecular dynamics,[4−15] semiempirical formulations,[16,17] ab initio independent particle models (including Hartree−Fock (HF) and density functional theory (DFT) methods),[18−31] to post-HF methods.[32−41]

The main goal of this paper is to present new Intel MIC implementation for the canonical representation of the perturbative corrections due to triples to the multireference coupled cluster models with singles and doubles (the MRCCSD(T) approaches[42−47]). Similar development has already been reported for NVIDIA GPU architectures using the CUDA environment.[36] In developing the version of the code for Intel MIC architecture we have assumed different strategy—instead of a complete rewrite of the code in the CUDA programming model to efficiently run computing kernels on the GPU hardware, we have modified floating-point intensive portions of the MRCCSD(T) calculations employing a directive-based offload model (that keeps large parts of the existing Fortran code unmodified). This approach significantly reduces time needed to develop and validates the code and at the same time provides tangible performance speedups associated with the utilization of the Intel Xeon Phi coprocessors. We hope that the Intel MIC algorithms can be efficiently extended not only to other canonical nonperturbative formulations but can also be adopted in the reduced-cost MRCC formulations such as recently introduced local state-specific MRCC formulations.[48]

The MRCCSD(T) approach provides an efficient and economical way of introducing the effect of triple excitations to the multireference coupled cluster theories. In several recent studies we have demonstrated a clear need for introducing triple excitations in the context of ground- and excited-state calculations.[36,49] Unfortunately, due to steep scaling, which is proportional to $M \times N^7$ (M and N refer here to the dimensionality of the model space and size of the system, respectively), the widespread applicability is rather limited. However, the emergence of hybrid architectures offers a unique chance to develop numerical algorithms that would enable MRCCSD(T) studies for systems composed of several hundred to thousands molecular orbitals. The main goal of this paper is to provide an insight of how Intel MIC architecture can be utilized in conjunction with multiple level of parallelism that MRCC formulations naturally bring to the picture. The universality of this approach can be easily extended to other noniterative MRCC formulations including Methods of Moments MRCC energy corrections[50,51] and Universal State-Selective MRCC formulations.[52] The paper is organized as follows: in section 2, we discuss the basic tenets of the state-specific MRCC

Received: October 7, 2015
Published: January 25, 2016

DOI: 10.1021/acs.jctc.5b00957
J. Chem. Theory Comput. 2016, 12, 1129−1138

formulations. All implementations details are given in section 3, whereas performance tests are discussed in section 4. As benchmark systems we chose C_{20} and free-base porphyrin molecules.

2. STATE-SELECTIVE MRCCSD APPROACH

In this section we provide a brief overview of the state-selective MRCC formalisms utilizing the Jeziorski−Monkhorst Ansatz[53] for arbitrary electronic wave function $|\Psi\rangle$

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

where Slater determinants $\{|\Phi_\mu\rangle\}_{\mu=1}^{M}$ span the so-called model space ($\mathcal{M}_0$) providing an adequate approximation to the electronic state of interest. The $T^{(\mu)}$ operators are the reference-specific cluster oprtators. The performance of MRCC formulations strongly depends on the choice of the model space. In our considerations we will focus on the complete model spaces (CMSs) which are defined by Slater determinants corresponding to all possible distributions of active electrons among active orbitals. Although the size of CMS grows quickly with the number of active electrons/orbitals, the utilization of CMS provides great simplifications in the Hilbert-space MRCC formulations especially in the context of assuring size-consistency of the resulting energies. We will often utilize two projection operators: (1) projection operator onto the model space (P) and (2) projection operator onto its orthogonal complement (Q)

$$P = \sum_{\mu=1}^{M} |\Phi_\mu\rangle\langle\Phi_\mu| = \sum_{\mu=1}^{M} P_\mu \tag{2}$$

$$Q = 1 - P \tag{3}$$

where the P_μ operator is a projection operator onto a specific reference function $|\Phi_\mu\rangle$. The operator Ω

$$\Omega = \sum_{\mu=1}^{M} e^{T^{(\mu)}} P_\mu \tag{4}$$

is often referred to as the wave operator. The CMS based Hilbert-space state-specific formulations assume the intermediate normalization of the wave operator Ω, i.e.,

$$P\Omega = P \tag{5}$$

which is equivalent to the requirement that the $T^{(\mu)}$ cannot produce excitations within the model space when acting onto the corresponding reference function $|\Phi_\mu\rangle$.

In the approximate variants of the Hilbert-space MRCC approaches,[54−62] the expansions representing cluster operators $T^{(\mu)}$ are truncated at certain (numerically tractable) excitation level. For example in the ubiquitous variant of MRCC theory with singles and doubles (MRCCSD) approximation, the cluster operators are represented as sums of singly ($T_1^{(\mu)}$) and doubly ($T_2^{(\mu)}$) excited clusters

$$T^{(\mu)} \simeq T_1^{(\mu)} + T_2^{(\mu)} \tag{6}$$

where each component $T_i^{(\mu)}$ ($i = 1, 2$) is defined by the so-called cluster amplitudes $t_{a_{(\mu)}\cdots}^{i_{(\mu)}\cdots}(\mu)$

$$T_1^{(\mu)} = \sum_{i_\mu, a_\mu} t_{a_\mu}^{i_\mu}(\mu) X_{a_\mu}^{+} X_{i_\mu} \tag{7}$$

$$T_2^{(\mu)} = \sum_{i_\mu<j_\mu, a_\mu<b_\mu} t_{a_\mu b_\mu}^{i_\mu j_\mu}(\mu) X_{a_\mu}^{+} X_{b_\mu}^{+} X_{j_\mu} X_{i_\mu} \tag{8}$$

where indices i_μ, j_μ (a_μ, b_μ) represent occupied (unoccupied) spinorbitals in the reference $|\Phi_\mu\rangle$. The intermediate normalization condition prevent the situation where all indices correspond to active spinorbitals.

In the state-specific formulations methods, to determine cluster amplitudes ($t_{a_\mu}^{i_\mu}(\mu)$ and $t_{a_\mu b_\mu}^{i_\mu j_\mu}(\mu)$) one substitutes the Jeziorski−Monkhorst representation of the wave function (1) into the Schrödinger equation

$$H \sum_{\mu}^{M} c_\mu e^{T^{(\mu)}} |\Phi_\mu\rangle = E \sum_{\mu=1}^{M} c_\mu e^{T^{(\mu)}} |\Phi_\mu\rangle \tag{9}$$

However, due to the known overcompleteness problem of the state-selective approaches, several types of sufficiency conditions (or the equations used to determine cluster amplitudes) have been discussed in the literature.[63−76]

In this paper we will focus on the two classes of the SS-MRCCSD approaches: the Brillouin−Wigner (BW)[63] and Mukherjee (Mk) MRCCSD[66] formulations, where the sufficiency conditions are given in the following form

$$(E - H_{\mu\mu}^{\mathrm{eff}})\langle\Phi_\theta^{(\mu)}|e^{T^{(\mu)}}|\Phi_\mu\rangle - \langle\Phi_\theta^{(\mu)}|H_N(\mu)e^{T^{(\mu)}}|\Phi_\mu\rangle_{C+DC,L} = 0 \qquad \forall_\mu\ ,\ \text{(BW-MRCCSD)} \tag{10}$$

$$\langle\Phi_\theta^{(\mu)}|(He^{T^{(\mu)}})_C|\Phi_\mu\rangle c_\mu + \sum_{\nu\neq\mu} \langle\Phi_\theta^{(\mu)}|e^{-T^{(\mu)}} e^{T^{(\nu)}}|\Phi_\mu\rangle H_{\mu\nu}^{\mathrm{eff}} c_\nu = 0 \qquad \forall_\mu\ ,\ \text{(Mk-MRCCSD)} \tag{11}$$

where the cluster operators are given by eqs 6 and $\langle\Phi_\theta^{(\mu)}|$ are excited configurations corresponding to the excitations used to define cluster operators $T^{(\mu)}$. The subscript C+DC,L designates all connected diagrams and all linked, but disconnected diagrams. In contrast to the Mk-MRCC approach, the BW-MRCC formalism contains disconnected contributions. The diagonalization of the effective Hamiltonian matrix, defined for CMS by the matrix elements $H_{\nu\mu}^{\mathrm{eff}}$,

$$H_{\nu\mu}^{\mathrm{eff}} = \langle\Phi_\nu|(He^{T^{(\mu)}})_C|\Phi_\mu\rangle \tag{12}$$

leads to energies (eigenvalues) and c_μ coefficients.

As in the case of single-reference CC theories full inclusion of triple excitations leads to a significant increase in the computational cost. Instead, one may try to include the correlation effects due to triples in a noniterative (or perturbative) fashion as in the CCSD(T) approach. The multireference extension of the (T) corrections follows similar ideas. Various forms of the MRCC nonperturbative triples corrections have been discussed in the literature in the context of various MRCC formulations (for details see refs 42−47 and 77). Here, we will discuss the approach where the perturbative corrections are added to the matrix elements of the effective Hamiltonian stemming from the BW-MRCCSD or Mk-MRCCSD calculations.[78] The form of the diagonal elements is analogous to that of the single reference energy diagrams, calculated using cluster amplitudes corresponding to the μth reference

$$H_{\mu\mu}^{\mathrm{eff}}(\mathrm{T}) = H_{\mu\mu}^{\mathrm{eff}}(\mathrm{CCSD}) + E_{\mathrm{T}}^{[4]}(\mu) + E_{\mathrm{ST}}^{[5]}(\mu) + E_{\mathrm{ST}}^{[4]}(\mu) \tag{13}$$

DOI: 10.1021/acs.jctc.5b00957
J. Chem. Theory Comput. 2016, 12, 1129−1138

where $H_{\mu\mu}^{\mathrm{eff}}(\mathrm{CCSD})$ represents diagonal matrix elements of the MRCCSD effective Hamiltonian (either in the BW or Mk formulation). The $E_{\mathrm{T}}^{[4]}(\mu)$, $E_{\mathrm{ST}}^{[5]}(\mu)$, and $E_{\mathrm{ST}}^{[4]}(\mu)$ terms are the fourth and fifth order contributions, given by

$$E_{\mathrm{T}}^{[4]}(\mu) = \frac{1}{36} \sum_{a_\mu b_\mu c_\mu i_\mu j_\mu k_\mu} \langle (\Phi_\mu)_{i_\mu j_\mu k_\mu}^{a_\mu b_\mu c_\mu} | V_{\mathrm{N}}(\mu) T_2(\mu) | \Phi_\mu \rangle_{\mathrm{C}} t_{i_\mu j_\mu k_\mu}^{a_\mu b_\mu c_\mu}(\mu) \tag{14}$$

$$E_{\mathrm{ST}}^{[5]}(\mu) = \sum_{a_\mu i_\mu} s_{i_\mu}^{a_\mu}(\mu) t_{i_\mu}^{a_\mu}(\mu) \tag{15}$$

$$E_{\mathrm{ST}}^{[4]}(\mu) = \frac{1}{4} \sum_{a_\mu b_\mu c_\mu i_\mu j_\mu k_\mu} f_{k_\mu c_\mu}(\mu) t_{i_\mu j_\mu}^{a_\mu b_\mu}(\mu) t_{i_\mu j_\mu k_\mu}^{a_\mu b_\mu c_\mu}(\mu) \tag{16}$$

and the $s_{i_\mu}^{a_\mu}(\mu)$ intermediate is defined as

$$s_{i_\mu}^{a_\mu}(\mu) = \frac{1}{4} \sum_{b_\mu c_\mu j_\mu k_\mu} v_{b_\mu c_\mu}^{j_\mu k_\mu}(\mu) t_{i_\mu j_\mu k_\mu}^{a_\mu b_\mu c_\mu}(\mu) \tag{17}$$

where all $t_{i_\mu}^{a_\mu}(\mu)$, $t_{i_\mu j_\mu}^{a_\mu b_\mu}(\mu)$, and $t_{i_\mu\ j_\mu\ k_\mu}^{a_\mu\ b_\mu\ c_\mu}(\mu)$ are automatically equal zero if all spinorbital indices defining given amplitude are of the active character. The $V_{\mathrm{N}}(\mu)$ operator is the two body part of normal ordered form of the Hamiltonian operator defined with respect to the $|\Phi_\mu\rangle$ Fermi vacuum.

Several formulation for calculating perturbative $t_{i_\mu j_\mu k_\mu}^{a_\mu b_\mu c_\mu}(\mu)$ amplitudes have been extensively discussed in the literature (see ref 78 and references therein for details). In this paper we will follow the simplest choice defined by the formula

$$t_{i_\mu j_\mu k_\mu}^{a_\mu b_\mu c_\mu}(\mu) = \frac{\langle (\Phi_\mu)_{i_\mu j_\mu k_\mu}^{a_\mu b_\mu c_\mu} | V_{\mathrm{N}}(\mu) T_2^{(\mu)} | \Phi_\mu \rangle_{\mathrm{C}}}{D_{i_\mu j_\mu k_\mu}^{a_\mu b_\mu c_\mu}(\mu)} \tag{18}$$

where $D_{i_\mu j_\mu k_\mu}^{a_\mu b_\mu c_\mu}(\mu)$ represents perturbative denominator. The algebraic form of the above triply excited cluster operator is similar to the form of the perturbative T_3 operator utilized in single-reference CCSD(T) approach.

3. IMPLEMENTATION DETAILS

The most expensive part of the MRCCSD(T) correction (scaling as $n_o(\mu)^3 n_u(\mu)^4$ and $n_o(\mu)^4 n_u(\mu)^3$) is associated with the presence of the $\langle (\Phi_\mu)_{i_\mu j_\mu k_\mu}^{a_\mu b_\mu c_\mu} | V_{\mathrm{N}}(\mu) T_2^{(\mu)} | \Phi_\mu \rangle$ term, which is defined by the expression:

$$\begin{aligned}
\langle (\Phi_\mu)_{i_\mu j_\mu k_\mu}^{a_\mu b_\mu c_\mu} | V_{\mathrm{N}}(\mu) T_2^{(\mu)} | \Phi_\mu \rangle &= v_{m_\mu a_\mu}^{i_\mu j_\mu}(\mu) t_{b_\mu c_\mu}^{m_\mu k_\mu}(\mu) \\
&- v_{m_\mu b_\mu}^{i_\mu j_\mu}(\mu) t_{a_\mu c_\mu}^{m_\mu k_\mu}(\mu) + v_{m_\mu c_\mu}^{i_\mu j_\mu}(\mu) t_{a_\mu b_\mu}^{m_\mu k_\mu}(\mu) - v_{m_\mu a_\mu}^{i_\mu k_\mu}(\mu) t_{b_\mu c_\mu}^{m_\mu j_\mu}(\mu) \\
&+ v_{m_\mu b_\mu}^{i_\mu k_\mu}(\mu) t_{a_\mu c_\mu}^{m_\mu j_\mu}(\mu) - v_{m_\mu c_\mu}^{i_\mu k_\mu}(\mu) t_{a_\mu b_\mu}^{m_\mu j_\mu}(\mu) + v_{m_\mu a_\mu}^{j_\mu k_\mu}(\mu) t_{b_\mu c_\mu}^{m_\mu i_\mu}(\mu) \\
&- v_{m_\mu b_\mu}^{j_\mu k_\mu}(\mu) t_{a_\mu c_\mu}^{m_\mu i_\mu}(\mu) + v_{m_\mu c_\mu}^{j_\mu k_\mu}(\mu) t_{a_\mu b_\mu}^{m_\mu i_\mu}(\mu) - v_{a_\mu b_\mu}^{e_\mu i_\mu}(\mu) t_{e_\mu c_\mu}^{j_\mu k_\mu}(\mu) \\
&+ v_{a_\mu c_\mu}^{e_\mu i_\mu}(\mu) t_{e_\mu b_\mu}^{j_\mu k_\mu}(\mu) - v_{b_\mu c_\mu}^{e_\mu i_\mu}(\mu) t_{e_\mu a_\mu}^{j_\mu k_\mu}(\mu) + v_{a_\mu b_\mu}^{e_\mu j_\mu}(\mu) t_{e_\mu c_\mu}^{i_\mu k_\mu}(\mu) \\
&- v_{a_\mu c_\mu}^{e_\mu j_\mu}(\mu) t_{e_\mu b_\mu}^{i_\mu k_\mu}(\mu) + v_{b_\mu c_\mu}^{e_\mu j_\mu}(\mu) t_{e_\mu a_\mu}^{i_\mu k_\mu}(\mu) - v_{a_\mu b_\mu}^{e_\mu k_\mu}(\mu) t_{e_\mu c_\mu}^{i_\mu j_\mu}(\mu) \\
&+ v_{a_\mu c_\mu}^{e_\mu k_\mu}(\mu) t_{e_\mu b_\mu}^{i_\mu j_\mu}(\mu) - v_{b_\mu c_\mu}^{e_\mu k_\mu}(\mu) t_{e_\mu a_\mu}^{i_\mu j_\mu}(\mu) \\
&(i_\mu < j_\mu < k_\mu,\ a_\mu < b_\mu < c_\mu)
\end{aligned} \tag{19}$$

where $v_{rs}^{pq}(\mu)$ is the tensor representing antisymmetrized two-electron integrals. Equation 19 can be separated into terms defined by contractions over occupied indices ($A_{i_\mu j_\mu k_\mu}^{a_\mu b_\mu c_\mu}(\mu)$; the first nine terms on the right-hand side (rhs) of eq 19) and terms corresponding to contraction over unoccupied indices ($B_{i_\mu j_\mu k_\mu}^{a_\mu b_\mu c_\mu}(\mu)$; the remaining nine terms on the rhs of eq 19), i.e.,

$$\langle (\Phi_\mu)_{i_\mu j_\mu k_\mu}^{a_\mu b_\mu c_\mu} | V_{\mathrm{N}}(\mu) T_2^{(\mu)} | \Phi_\mu \rangle = A_{i_\mu j_\mu k_\mu}^{a_\mu b_\mu c_\mu}(\mu) + B_{i_\mu j_\mu k_\mu}^{a_\mu b_\mu c_\mu}(\mu) \tag{20}$$

In implementing the MRCCSD(T) approach we employed Tensor Contraction Engine (TCE)[79] which uses spin−orbital representation of all tensors defining Hamiltonian and cluster operators.

In order to provide data granularity for the TCE generated codes, the whole spin−orbital domain is partitioned into smaller pieces called *tiles* which contain several spin−orbitals of the same spatial- and spin-symmetry. The maximum number of elements in the tile is often referred to as the `tilesize`. This partitioning induces the partitioning or the block structure of all tensors used in the CC calculations, including: amplitudes, recursive intermediates, integrals, and residual vectors. In contrast to the single-reference CC methods, the MRCC TCE implementations utilize reference-specific tiling scheme to provide the optimal layout from the point of view of building reference-specific contributions to the MRCCSD equations and MRCCSD(T) corrections.

In the parallel implementation of the (T)-parts of the MRCCSD(T)approach, each process takes care of a different set of projections defined by tiles: $[i_\mu]$, $[j_\mu]$, $[k_\mu]$, $[a_\mu]$, $[b_\mu]$, $[c_\mu]$, i.e., each process generates on-the-fly the set of $\langle (\Phi_\mu)_{i_\mu j_\mu k_\mu}^{a_\mu b_\mu c_\mu} | V_{\mathrm{N}}(\mu) T_2^{(\mu)} | \Phi_\mu \rangle$, $\langle (\Phi_\mu)_{i_\mu j_\mu k_\mu}^{a_\mu b_\mu c_\mu} | V_{\mathrm{N}}(\mu) T_1^{(\mu)} | \Phi_\mu \rangle$, and $\langle (\Phi_\mu)_{i_\mu j_\mu k_\mu}^{a_\mu b_\mu c_\mu} | F_{\mathrm{N}}(\mu) T_2^{(\mu)} | \Phi_\mu \rangle$ projections with $i_\mu \in [i_\mu]$, $j_\mu \in [j_\mu]$, $k_\mu \in [k_\mu]$, $a_\mu \in [a_\mu]$, $b_\mu \in [b_\mu]$, $c_\mu \in [c_\mu]$. These projections are stored in the six-dimensional matrices $P3(\mu)(\langle (\Phi_\mu)_{i_\mu j_\mu k_\mu}^{a_\mu b_\mu c_\mu} | V_{\mathrm{N}}(\mu) T_2^{(\mu)} | \Phi_\mu \rangle)$ and $R3(\mu)(\langle (\Phi_\mu)_{i_\mu j_\mu k_\mu}^{a_\mu b_\mu c_\mu} | V_{\mathrm{N}}(\mu) T_1^{(\mu)} | \Phi_\mu \rangle$ or $\langle (\Phi_\mu)_{i_\mu j_\mu k_\mu}^{a_\mu b_\mu c_\mu} | F_{\mathrm{N}} T_2^{(\mu)} | \Phi_\mu \rangle)$. For example, $P3(\mu)$ is defined as the following matrix

$$\begin{aligned}
P3(\mu) \equiv P3(\mu)(&\dim[a_\mu], \dim[b_\mu], \dim[c_\mu], \dim[i_\mu] \\
&, \dim[j_\mu], \dim[k_\mu])
\end{aligned} \tag{21}$$

Since the total number of floating-point operations on each process associated with forming a given $P3(\mu)$ tensor is proportional to $\mathtt{tilesize}^6 * n_o(\mu)$ ($A(\mu)$ terms) and $\mathtt{tilesize}^6 * n_u$ ($B(\mu)$ terms) (cost of forming $R3(\mu)$ is proportional to the $\mathtt{tilesize}^6$ only), this type of calculation is ideally suited to take advantage of the Xeon Phi coprocessor's computing capabilities.

The whole process of calculating (T) energy correction to the $H_{\mu\mu}^{\mathrm{eff}}$ element of effective Hamiltonian is split into a number of elementary tasks, where the summation is performed across the indices of a single tile, for example the 10th term on the rhs of eq 19

$$\begin{aligned}
P3(\mu)(a_\mu, b_\mu, c_\mu, i_\mu, j_\mu, k_\mu) - = &\sum_{e_\mu \in [e_\mu]} v_{a_\mu b_\mu}^{e_\mu i_\mu}(\mu) t_{e_\mu c_\mu}^{j_\mu k_\mu}(\mu) \\
&(i_\mu \in [i_\mu], j_\mu \in [j_\mu], k_\mu \in [k_\mu], a_\mu \in [a_\mu], b_\mu \in [b_\mu], c_\mu \\
&\in [c_\mu])
\end{aligned} \tag{22}$$

where summation over unoccupied indices is limited to unoccupied indices which belongs to a given unoccupied tile $[e_\mu]$.

A similar algebraic structure of elementary tasks is characterized for the $A_{ijk}^{abc}(\mu)$ term where the elementary tasks

DOI: 10.1021/acs.jctc.5b00957
J. Chem. Theory Comput. 2016, 12, 1129−1138

contain the summation over indices that belong to a given occupied tile [m]. For each of 18 elementary tasks the flop count is proportional to tilesize[7], which for tilesize = 20 translates into 1.2 GFLOP. We expect, that the utilization of the coprocessor should lead to considerable speedups in the case of a large numerical load, which is implied by the use of larger tiles. Both types of contractions defining elementary tasks can be performed by calling dgemm BLAS procedures.[80] However, this approach leads to increased local memory requirements from 2 × (tilesize)[6] to 3 × (tilesize)[6] and to the necessity to perform index permutations in 6-dimensional (tilesize)[6] objects. To avoid these problems one can replace the calls to dgemm by simple procedures containing 7-nested do loops (see Figure 1 for an example). A similar algebraic structure of

```
do p4m=1,p4m_d
 do p5m=1,p5m_d
  do p6m=1,p6m_d
   do h1m=1,h1m_d
    do h7m=1,h7m_d
     do h2m=1,h2m_d
      do h3m=1,h3m_d
       triplesx_m(h3m,h2m,h1m,p6m,p5m,p4m) =
        triplesx_m(h3m,h2m,h1m,p6m,p5m,p4m) -
        t2sub_m(h7m,p4m,p5m,h1m)*v2sub_m(h3m,h2m,p6m,h7m)
       enddo
      enddo
     enddo
    enddo
   enddo
  enddo
 enddo
enddo
```

Figure 1. Example of dgemm replacement kernel.

elementary tasks holds for the $A_{ijk}^{abc}(\mu)$ term where the elementary tasks contain the summation over indices that belong to a given occupied tile [m_μ]. In this paper we will consider local-memory efficient algorithm where expensive from the point of view of local memory utilization transpositions of 6-dimensional $P3(\mu)$ tensors are eliminated by replacing dgemm procedures by 7-nested do loops (see Figure 1) In Figure 1, h1m, h2m, h3m, and h7m are the occupied spin−orbital indices (i_μ, j_μ, k_μ, m_μ) in the μth reference $|\Phi_\mu\rangle$. The loops p4m, p5m, and p6m are over unoccupied (in μth reference) spin−orbital indices (a_μ, b_μ, c_μ). In the above code snapshot the triplesx_m tensor corresponds to the $P3(\mu)$ tensor of eq 22, while t2sub_m and v2sub_m are sub-blocks of cluster amplitudes and two-electron integrals corresponding to the μth reference, respectively. This approach has been implemented several years ago in the NWChem TCE CCSD(T) module and has also been reported in the context of excited-state noniterative methods.[81] It has also been used as a platform for the GPU and Intel MIC single-reference CCSD(T) implementations.[35,40]

In analogy to the CCSD(T) formalism, the implementation of the MRCCSD(T) approach for the Intel Xeon Phi coprocessor requires two distinct steps: (1) offloading of the compute-intensive kernels to the coprocessors and (2) optimization of these kernels to take full advantage of the Xeon Phi hardware, which requires a thread-parallel and single-instruction-multiple-data (SIMD) parallel approach on the target device. In Figure 2, we show some of the most important directives we have used to move data between host and coprocessor. Similarly to the single-reference CCSD(T) approach the triplesx_m 6-dimensional array, main output quantity of the procedure, is first created and initialized to zero on the coprocessor (therefore requiring no data movement up to this point). Next, before the

```
#define ALLOC alloc_if(.true.) free_if(.false.)
#define FREE  alloc_if(.false.) free_if(.true.)
#define REUSE alloc_if(.false.) free_if(.false.)

CDIR$ OFFLOAD TARGET(mic:mic_device)
     N     IN(triplesx_m:length(0) REUSE)
     I     IN(h3m_d,h2m_d,h1m_d)
     I     IN(p6m_d,p5m_d,p4m_d,h7m_d)
     I     IN(t2sub:length(l_t2sub_m) REUSE)
     R     IN(v2sub:length(l_v2sub_m) REUSE)
```

Figure 2. Example of offload directives.

call of each kernel that multiplies together the 4-dimensional array v2sub_m and t2sub_m, the directives shown in Figure 2 are issued. As far as the triplesx_m array is concerned, once again no data movement is necessary at this stage; this is possible since during the offload phase we are accumulating the contributions resulting from multiplying together the v2sub_m and t2sub_m quantities directly on the coprocessor.

In each task the number of executions of a particular kernels is proportional to the total number of occupied (for a given reference) tiles noab_m (A term in eq 20) or to the total number of unoccupied (for a given reference) tiles nvab_m (B term in eq 20) or equivalently the contributions with contractions over virtual indices (see eq 22).

For kernel parallelization we used OpenMP for multithreading and the autovectorizer of the Intel Composer for SIMD parallelism. In Figure 3 we show the OpenMP directive added

```
!$omp parallel do &
!$omp     private(p4m,p5m,p6m,h2m,h1m,h3m,h7m) &
!$omp     collapse(OMPCOLLAPSE)
do p4m=1,p4m_d
 do p5m=1,p5m_d
  do p6m=1,p6m_d
   do h1m=1,h1m_d
    do h7m=1,h7m_d
!dec$ loop count max=1000, min=20
     do h3h2m=1,h2m_d*h3m_d
      triplesx_m(h3h2m,h1m,p6m,p5m,p4m) =
       triplesx_m(h3h2m,h1m,p6m,p5m,p4m) -
       t2sub_m(h7m,p4m,p5m,h1m)*v2sub_m(h3h2m,p6m,h7m)
     enddo
    enddo
   enddo
  enddo
 enddo
enddo
!$omp end parallel do
```

Figure 3. Example kernel of Figure 1 with OpenMP and loop collapsing.

to the kernel of Figure 1. Since the trip count of the outer loops p4m to p6m is low (cf. tilesize), they do not exhibit enough parallelism to make use of all coprocessor cores. Hence, the collapse clause is used to instruct the compiler to collapse the outer loops into a single product loop for parallelization. The OMPCOLLAPSE parameter can be defined during compile time of NWChem (and is set to 3 by default), effectively collapsing the loop nest of p4m, p5m, and p6m. This results in a product loop with about 1 or 2 order of magnitude more iterations than the number of threads (depending on tilesize).

The innermost loop is also a good candidate for autovectorization. The triplesx_m 6-dimensional array is an invariant and t2sub_m and v2sub_m are both accessed with stride-1 along the column-major dimension. However, due to the small trip

DOI: 10.1021/acs.jctc.5b00957
J. Chem. Theory Comput. 2016, 12, 1129−1138

count of the innermost loop, vectorization will not be effective. A `tilesize` of 19 will lead to an average vector utilization of 6.3 elements for double precision vectors (vectorization efficiency 80%). The efficency can be increased by (manually) collapsing the `h3m` and `h2m` loop in Figure 3. For the example `tilesize` of 19, the product loop will have a trip count of 361, which leads to a vectorization efficiency of 98% or an average utilization of 7.8 DP elements per vector. For loops with a trip count that is a multiple of 8, this transformation lowers the total loop overhead.

In all kernels we augmented loops with a directive to inform the compiler about the minimum and maximum trip counts of the loops (see Figure 3). Without the directive, the compiler assumes too high trip counts and thus trigger counterproductive high-level optimizations that inhibit optimal vectorization. Avoiding these transformations improves performance by a rough 10%.

Figure 4 shows another candidate kernel for vectorization that exhibits poor performance due to the permuted array subscripts.

```
!$omp parallel do &
!$omp     private(p4m,p5m,p6m,h2m,h1m,h3m,h7m) &
!$omp     collapse(OMPCOLLAPSE)
do p4m=1,p4d_m
 do p5m=1,p5d_m
  do p6m=1,p6d_m
   do h2m=1,h2d_m
    do h3m=1,h3d_m
     do h7m=1,h7d_m
      do h1m=1,h1d_m
       triplesx_m(h1m,h3m,h2m,p6m,p5m,p4m) =
     *   triplesx_m(h1m,h3m,h2m,p6m,p5m,p4m) -
     *   t2sub_m(h7m,p4m,p5m,h1m)*v2sub_m(h3m,h2m,p6m,h7m)
      enddo
     enddo
    enddo
   enddo
  enddo
 enddo
enddo
!$omp end parallel do
```

Figure 4. Example `dgemm` replacement kernel without stride-1 access.

Any permutation of the loop nest leads to gather/scatter operations for at least one array. While the coprocessor supports gather/scatter operations,[82] these are high-latency instructions and impose non-negligible performance overheads. A detailed analysis of the permutations of the array subscripts and loop nests reveals that the `t2sub_m` array can be transposed, effectively reversing its array subscripts. With this transformation all array accesses become stride-1 accesses, while other kernels are not affected performancewise: either the transformation is beneficial or the access to `t2sub_m` is invariant for the innermost loops. The transposition is performed on the host after the array has been received and before it is transferred to the coprocessor for processing. The performance overhead of the transposition is negligible compared to the kernel runtime on the coprocessor and is hidden in the measurement noise.

The Intel MIC version of the MRCCSD(T) code can be easily integrated with the parallel MRCC algorithms based on the utilization of the reference-level parallelism and processor groups (PG). By processor group (G_i) we mean a partitioning of the processor domain (D) into smaller groups of processors, which can be symbolically expressed as

$$D = \bigcup_{i=1,...,I} G_i \tag{23}$$

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

$$S_i = S = \frac{N_p}{I} \qquad (i = 1, ..., I) \tag{24}$$

In the above equation N_p stands for the total number of processors, which is a multiple of I. The key idea is to distribute the formation of reference-specific MRCCSD equations or reference-specific (T) corrections over various processor groups. This approach employs two-level parallelism: (1) reference-level parallelism, where each set of reference-specific equations/energy contributions is calculated on separate PG, (2) task-level parallelism used to calculate given set of reference-specific equations/(T)-energy-contributions. 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 reference-specific equations/(T)-energy contributions. In this case the number of PGs coincides with the size of the model space (i.e., $I = M$).

4. COMPUTE ARCHITECTURE

The Intel Xeon Phi coprocessor is a massively parallel computing device housed in a double-wide PCIExpress (gen 2) extension card. It resembles and extends the key properties of the well-known Intel Architecture (IA). The 5110P model of the coprocessor, used to collected the perfomance data of this work, offers 60 general-purpose cores with in-order execution and a base frequency of 1053 MHz. The card is equipped with 8 GB of on-card GDDR5 memory with 16 memory channels. The cores are based on a modified Intel Pentium (P54C) processor design. Each core can execute 64-bit instructions and offers a 512-bit wide vector processing unit (VPU) for SIMD processing. Each of the cores supports hyper-threading with a four-way round-robin scheduling of hardware threads. The peak floating-point performance for double precision (DP) is 1010.88 GFLOPS or 2021.76 GFLOPS for single-precision (SP) computation.

Each core of the coprocessor owns a private L1 and L2 cache with 32 KB and 512 KB, respectively. With 60 cores, the overall size of the L2 cache is 31 MB. The cache is fully coherent and implements the standard IA coherency semantics across all caches and the card's memory. An on-die interconnection network connects all cores and their L2 caches in a bidirectional ring topology for fast on-chip communication.

Apart from the SIMD instructions, there are several key differences between the coprocessor architecture and the architecture of a Intel Xeon E5 series processor. The wider SIMD vector registers require a higher degree of vectorization than the Xeon processor. In addition, the larger number of cores demands a much higher degrees of parallelism and thus workloads must not have a too high sequential fraction or parallelization overhead. There are also differences in cache structure. In contrast to the Xeon processor, the coprocessor does not have a Last Level Cache that can be shared arbitrarily between the cores. Each coprocessor core can only load data into its local portion of the L2 cache; another core's L2 cache can only be accessed for remote cache hits.

5. PERFORMANCE TESTS

Before discussing the performance of the Intel MIC implementation of the MRCCSD(T) approaches it is instructive

DOI: 10.1021/acs.jctc.5b00957
J. Chem. Theory Comput. 2016, 12, 1129−1138

to estimate (at least approximately) the performance of the noniterative (T)-type implementations on various heterogeneous architectures. For this purpose, we chose to compare individual timings of Intel MIC and NVIDIA GPU implementations of noniterative triples of single-reference CCSD(T) approach—both available in the official NWChem release. Also, both implementations use identical representation of data structure, the identical form of the algebraic equations, and similar design principles. In the test runs we used CASCADE (Intel Xeon Phi 5110P) and Titan (NVIDIA K20X) platforms, which are characterized by similar performance parameters. The obtained timings for the pentacene molecule (in C_1 symmetry) described by the cc-pVDZ basis set[83] are shown in Table 1. For

Table 1. Wall Time to Solution (s) of Noniterative Triples Part of the Single-Reference CCSD(T) Approach for the Pentacene Molecule Using Intel MIC and NVIDIA GPU Implementations of References 40 and 35, Respectively[a]

tilesize	Intel MIC	NVIDIA GPU
18	1806.4	1824.9
21	1652.2	1699.3
24	1453.3	1554.4

[a]Tests were performed using 96 computing nodes of the Cascade system at EMSL (Intel MIC) and the Titan system at ORNL (NVIDIA GPU).

example, for `tilesize` = 18 the difference between Intel MIC and NVIDIA GPU timings is smaller than 20 s, which constitute a small fraction of total time to solution (1806.4 s with Intel MIC implementation). Because of these small disagreements between the Intel MIC and NVIDIA GPU timings, which we believe should to a large extent be attributed to minor differences in corresponding implementations (see ref 35 for NVIDIA GPU implementation details) and system specifications, one should view these timings as comparable. Therefore, the obtained results provide a good motivation for developing a unified "platform-agnostic" programming model for hybrid architectures.

We have chosen as benchmark systems to test the performance of the Intel MIC based implementation of the MRCCSD(T) due to triples corrections two medium size systems: C_{20} in the cc-pVDZ basis set[83] and free base porphyrin (FBP) in the 6-31G basis set.[84] In all calculations we kept core electron frozen. For both benchmark systems C_1 symmetry was used. For the purpose of this paper we adopted the molecular geometries of the FBP and C_{20} systems from refs.,[85,86] respectively. The complete model spaces used in our calculations are denoted using (N_{ae}, N_{ao}) convention, where N_{ae} and N_{ao} stand for the number of active electrons and orbitals, respectively. All performance data discussed in this section correspond to the speedups achieved in tests MRCCSD(T) runs based on the low-memory algorithms for calculating the most expensive components of the MRCCSD(T) corrections described by eq 20. In our tests we utilized the CASCADE system at EMSL, which consists of dual-socket Intel Xeon E5-2670 processors with 2.6 GHz and 128 GB of memory. Each node houses two Intel Xeon Phi 5110P coprocessors with 60 cores at 1053 MHz and 8 GB on-card memory.

We will start our analysis with the results obtained for the C_{20} system. For test purposes we used its cage configuration and 4-dimensional (2, 2) model space. As shown in Figure 5 the wall time for the MRCC noniterative triples is almost `tilesize` independent for all `tilesize` values considered (15, 20, and 24). This is especially true for the MRCCSD(T) implementation

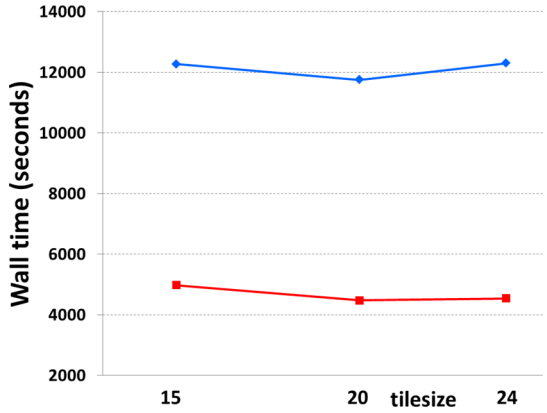


Figure 5. Wall time to solution (s) for the MRCCSD(T) perturbative triples correction to the MRCCSD correlation energy of the C_{20} system in the cc-pVDZ[83] basis set. A 4-dimensional (2, 2) model space was employed in the calculations. In all calculations core electrons were kept frozen and C_1 symmetry was used.

utilizing Intel Xeon processors and Intel Xeon Phi coprocessors (red line in Figure 5). One can also notice a consistent 2.7 times speed-ups of the MRCCSD(T) runs utilizing processors and coprocessors compared to the host-only execution (Intel Xeon processors). These tests were performed using 256 host cores and 256 host cores plus 3840 Xeon Phi cores, which totals to 4096 aggregate cores for the heterogeneous runs. The achieved speedups are in line with the speedups reported in the context of the single-reference CCSD(T) formulation.[40]

In order to illustrate the parallel performance of the MRCCSD(T) method we performed several scalability tests for the C_{20} system using the EMSL CASCADE computer. The results of our tests are shown in Figure 6. In particular, we are

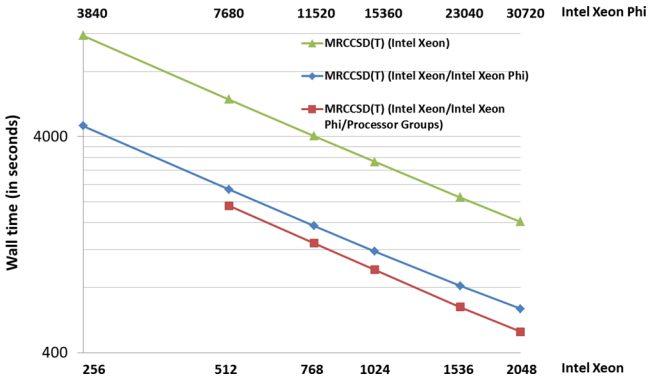


Figure 6. Wall time to solution (s) for the MRCCSD(T) perturbative triples correction to the MRCCSD correlation energy of the C_{20} system in the cc-pVDZ[83] basis set. A 4-dimensional (2, 2) model space was employed in the calculations. In all calculations core electrons were kept frozen and C_1 symmetry was used.

interested in estimating the parallel performance of the MRCCSD(T) code based on the concurrent utilization of processors, coprocessors, and processor groups. The standard MRCCSD(T) Intel Xeon implementation (green line) performs quite satisfactory providing 87% parallel efficiency in runs ranging from 256 to 2048 Intel Xeon cores. The offloading not only reduces time to solution (consistently resulting in 2.7 time speedups) but also positively impact the parallel performance—one can observe slightly better 91% parallel efficiency in runs involving Intel Xeon processors and Intel Xeon Phi coprocessors.

DOI: 10.1021/acs.jctc.5b00957
J. Chem. Theory Comput. 2016, 12, 1129−1138

The additional utilization of processor groups further enhances the parallel performance (96% in runs ranging from 512 host cores and 7680 Xeon Phi cores to 2048 host cores and 30720 Xeon Phi cores). It should also be stressed that the combined utilization of processors, coprocessors, and PGs results in more than 3 times acceleration of the original Intel Xeon based code.

As an indicator of the improvements stemming from the utilization of the processor groups, in Figure 6 we compare the performance of the MRCCSD(T) version utilizing processor groups and Xeon/Xeon Phi cores (T_{PG}) and non-PG based MRCCSD(T) variant still utilizing Xeon/Xeon Phi cores ($T_{non\text{-}PG}$) using the following speedup factor

$$\eta = 1 - \frac{T_{PG}}{T_{non\text{-}PG}} \tag{25}$$

Again, we used C_{20} benchmark system employed in previous paragraphs. In our tests we employed 512, 768, 1024, 1536, 2048 Intel Xeon processors and 7680, 11 520, 15 360, 23 040, 30 720 Intel Xeon Phi cores, respectively. In the PG-based runs it translates into 128, 192, 256, 384, and 512 Intel Xeon cores (1920, 2880, 3840, 5760, 7680 Intel Xeon Phi cores) per reference in the PG-based algorithm. From Figure 6 (see also Table 11) one can see that the parallel execution of the PG-based algorithm is consistently better than the non-PG implementation. The superior performance of the PG-algorithm over the non-PG variant can be observed for larger number of cores. For example, for test runs utilizing 2048 CPU and 30720 Xeon Phi threads the PG-algorithm is 22% faster than its non-PG version. Figure 7 also suggests that performance discrepancies increase with the number of CPU/Xeon Phi threads.

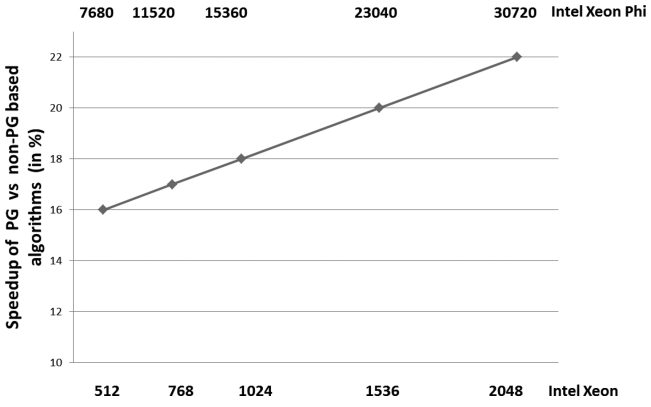


Figure 7. Speedup of the PG-based algorithm with respect to the non-PG MRCCSD(T) implementation. Test runs were performed for the C_{20} system in the cc-pVDZ[83] basis set. A 4-dimensional (2, 2) model space was employed in the calculations. In all calculations core electrons were kept frozen and C_1 symmetry was used.

The relative energies of the three C_{20} isomers (bowl, cage, ring) have been a subject of intensive theoretical studies.[86−100] To our knowledge the MRCC methods have never been used to study the energy ordering of these isomers. Our MRCC results obtained with the cc-pVDZ basis set (assuming all core electrons to be frozen) are summarized in Tables 2−3. In all MRCC calculations we employed restricted Hartree−Fock orbitals. For all C_{20} configurations we observed a dominant role played by the RHF Slater determinant in the ground-state right eigenvectors of the effective Hamiltonian. Also, we have not observed a significant effect of the model space employed (as shown in

Table 2. Timings (s) for the MRCCSD(T) Perturbative Triples Correction to the MRCCSD Correlation Energy of the C_{20} System in the cc-pVDZ[83] Basis Set[a]

MRCCSD(T) (Intel Xeon cores)	MRCCSD(T) (Intel Xeon/Intel Xeon Phi cores)	MRCCSD(T)/Processor Group (Intel Xeon/Intel Xeon Phi cores)
11754	4479	
(256)	(256/3840)	
5948	2278	1914
(512)	(512/7680)	(512/7680)
4027	1544	1285
(768)	(768/11520)	(768/11520)
3053	1177	968
(1024)	(1024/15360)	(1024/15360)
2094	815	650
(1536)	(1536/23040)	(1536/23040)
1611	639	499
(2048)	(2048/30720)	(2048/30720)

[a]A 4-dimensional (2, 2) model space was employed in the calculations. In all calculations core electrons were kept frozen and C_1 symmetry was used (see also Figure 6).

Table 3. Total Electronic Energies for C_{20} (Fullerene) Obtained with the cc-pVDZ Basis Set for Various Model Spaces[a]

model space	BW-MRCCSD	BW-MRCCSD(T)	Mk-MRCCSD	Mk-MRCCSD(T)
(2, 2)	−759.2012	−759.3949	−759.2012	−759.3940
(2, 3)	−759.2007	−759.3996	−759.2008	−759.3934
(2, 4)	−759.2004	−759.4006		

[a]All core electrons were kept frozen in calculations.

Table 2, where several MRCC energies obtained for various model spaces are provided) on the MRCC energies. The relative energies of bowl, fullerene, and ring isomers are collected in Table 3. Although all results (except for the CCSD(T) obtained for the LDA optimized geometries) predict the energy ordering in (ascending) order bowl-fullerene-ring, one can notice significant differences between bowl-fullerene and bowl-ring energetics. While Mk-MRCCSD methods predicts 0.98 eV energy separation between bowl an fullerene structures, adding noniterative triples (Mk-MRCCSD(T)) narrows this gap to 0.24 eV (which is reminiscent of almost degenerate CCSD(T) energies obtained for LDA optimized geometries of ref.[93]). The bowl−fullerene energy separation obtained in the most recent CCSD(T) calculations based on the cc-pVTZ basis set (Jin et al.[100]) corresponds to 0.69 eV, which falls between Mk-MRCCSD and Mk-MRCCSD(T) predictions. At the same point the corresponding cc-pVTZ CCSD calculations,[100] obtained within frozen core approximation predict bowl−fullerene separation to amount to 1.36 eV. The Mk-MRCCSD and Mk-MRCCSD(T) bowl-ring energy separations are much closer to each other, 2.13 vs 2.27 eV, respectively, than compared to the bowl−fullerene case. Slightly higher values have been reported by An et al.[97] and by Bylaska et al.[93] The utilization of the cc-pVTZ basis set leads to smaller bowl−ring separation of 1.37 eV.

To estimate the performance of the PG-based algorithm for more realistic MRCCSD(T) simulations employing larger number of reference functions we used free base porphyrin in the 6-31G basis set as a benchmark system. In our calculations we used C_1 symmetry and employed (4, 4) complete model space

DOI: 10.1021/acs.jctc.5b00957
J. Chem. Theory Comput. 2016, 12, 1129−1138

Table 4. Calculated Relative Energies (eV) Obtained with Single- and Multireference Coupled-Cluster Methods[a]

C_{20} configuration (model space)	MkCCSD	MkCCSD(T)	CCSD(T)[b]	CCSD(T)[c]	CCSD(T)[d]	CCSD(T)[e]
fullerene (2, 3)	0.98	0.24	0.3	0.0	0.39	0.69
ring (4, 4)	2.13	2.27	2.5	1.7	2.45	1.73
bowl (4, 4)	0.00	0.00	0.00	0.0	0.00	0.0

[a]The Mk-MRCCSD (MkCCSD column) and Mk-MRCCSD(T) (MkCCSD(T) column) results were obtained with cc-pVDZ basis set using RHF geometries of ref 86 for several model spaces defined in the first column). Frozen core approximation has been used in the MRCC calculations. [b]The cc-pVDZ MP2/CCSD(T) results for SCF geometries were taken from ref 93. [c]The cc-pVDZ CCSD(T) results for the LDA geometries were taken from ref 93. [d]The cc-pVDZ CCSD(T) result of ref 97. [e]The cc-pVTZ CCSD(T) (drop core) results of ref 100.

which contains 36 Slater determinants. In analogy to the C_{20} case we have performed two series of tests of non-PG and PG implementations of Xeon/Xeon Phi version of the MRCCSD(T) code. We performed two series of calculations: (1) employing the PG-based algorithm with 32 and 64 cores per subgroup (which totaled to 1152 and 2304 Intel Xeon cores, respectively) and (2) utilizing the non-PG based MRCCSD(T) algorithm for the same count of Intel Xeon cores. The results are shown in Table 4. The parallel efficiencies of the PG and non-PG codes (calculated based on two runs involving 1152 and 2304 Intel Xeon cores) correspond to 99.6% and 94.2%, respectively. In analogy, the C_{20} case, the PG algorithm performs better than the non-PG implementation offering 8.75% and 13.69% speed-up for 1152 and 2304 Intel Xeon core runs.

Table 5. Wall Time to Solution (s) for the MRCCSD(T) Perturbative Triples Correction for the Free Base Porphyrin in the 6-31G Basis Set and Model Space Containing 36 Slater Determinants

computational configuration	PG used	PG not used	η
1152 Intel Xeon processors 17280 Intel Xeon Phi coprocessors	8712.8	9548.6	8.75%
2304 Intel Xeon processors 34560 Intel Xeon Phi coprocessors	4372.0	5065.7	13.69%

6. CONCLUSIONS

In this paper we performed several tests for the MRCCSD(T) implementations capable of taking advantage of (1) Intel Xeon Phi cores and (2) multiple levels of parallelism associated with the reference-level parallelism and task level parallelism within each processor group. We have demonstrated that adapting the MRCCSD(T) code for the Intel MIC architecture results in a sizable 3 time speed-up of the code compared to the version which does not take advantage of Intel Xeon Phi cores. This is in line which our earlier results reported in the context of the single-reference CCSD(T) calculations (especially for larger values of `tilesize` parameter). We have also shown that the performance of processor group algorithm is consistently better compare to the non-PG algorithms. The obtained results suggest that the PG-based algorithm is especially effective for larger number of cores contributing to a given processor group (9% vs 22% of speed-up for the PG-based MRCCSD(T) for 32 and 512 Intel Xeon cores per processor group, respectively). These results may rise the hopes for very efficient utilization of hybrid computer architectures for multireference theories, which can utilize multiple levels of parallelism. Currently, an intensive effort is made to extend the implementations of universal state-selective MRCC methods[52,101−103] to take advantage of Intel MIC technology.

AUTHOR INFORMATION

Corresponding Authors

*E-mail: edoardo.apra@pnnl.gov

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

Notes

The authors declare no competing financial interest.

ACKNOWLEDGMENTS

This work has been supported by the Intel Parallel Computational Centers program. All 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 US Department of Energy by the Battelle Memorial Institute under Contract No. DE-AC06.76RLO-1830.

REFERENCES

(1) Owens, J. D.; Luebke, D.; Govindaraju, N.; Harris, M.; Krüger, J.; Lefohn, A. E.; Purcell, T. J. *Comput, Graph. Forum* **2007**, *26*, 80−113.

(2) Jeffers, J.; Reinders, J. In *Intel Xeon Phi Coprocessor High Performance Programming*; Reinders, J., Jeffers, J., Eds.; Morgan Kaufmann: Boston, 2013; pp 1−22.

(3) Reinders, J. In *High Performance Parallelism Pearls*; Jeffers, J., Reinders, J., Eds.; Morgan Kaufmann: Boston, 2015; pp 1−5.

(4) Stone, J. E.; Phillips, J. C.; Freddolino, P. L.; Hardy, D. J.; Trabuco, L. G.; Schulten, K. *J. Comput. Chem.* **2007**, *28*, 2618−2640.

(5) Hardy, D. J.; Stone, J. E.; Schulten, K. Revolutionary Technologies for Acceleration of Emerging Petascale Applications. *Parallel Comput.* **2009**, *35*, 164−177.

(6) Stone, J. E.; Hardy, D. J.; Ufimtsev, I. S.; Schulten, K. *J. Mol. Graphics Modell.* **2010**, *29*, 116−125.

(7) Ufimtsev, I. S.; Martinez, T. J. *J. Chem. Theory Comput.* **2009**, *5*, 2619−2628.

(8) Anderson, J. A.; Lorenz, C. D.; Travesset, A. *J. Comput. Phys.* **2008**, *227*, 5342−5359.

(9) Friedrichs, M. S.; Eastman, P.; Vaidyanathan, V.; Houston, M.; Legrand, S.; Beberg, A. L.; Ensign, D. L.; Bruns, C. M.; Pande, V. S. *J. Comput. Chem.* **2009**, *30*, 864−872.

(10) van Meel, J.; Arnold, A.; Frenkel, D.; Zwart, S. P.; Belleman, R. *Mol. Simul.* **2008**, *34*, 259−266.

(11) Eastman, P.; Pande, V. S. *J. Comput. Chem.* **2010**, *31*, 1268−1272.

(12) Levine, B. G.; LeBard, D. N.; DeVane, R.; Shinoda, W.; Kohlmeyer, A.; Klein, M. L. *J. Chem. Theory Comput.* **2011**, *7*, 4135−4145.

(13) Maruyama, Y.; Hirata, F. *J. Chem. Theory Comput.* **2012**, *8*, 3015−3021.

(14) Götz, A. W.; Williamson, M. J.; Xu, D.; Poole, D.; Le Grand, S.; Walker, R. C. *J. Chem. Theory Comput.* **2012**, *8*, 1542−1555.

(15) Le Grand, S.; Götz, A. W.; Walker, R. C. *Comput. Phys. Commun.* **2013**, *184*, 374−380.

(16) Wu, X.; Koslowski, A.; Thiel, W. *J. Chem. Theory Comput.* **2012**, *8*, 2272−2281.

(17) Nam, K. *J. Chem. Theory Comput.* **2013**, *9*, 3393−3403.

(18) Yasuda, K. *J. Comput. Chem.* **2008**, *29*, 334−342.

DOI: 10.1021/acs.jctc.5b00957
J. Chem. Theory Comput. 2016, 12, 1129−1138

(19) Yasuda, K. *J. Chem. Theory Comput.* **2008**, *4*, 1230−1236.
(20) Ufimtsev, I. S.; Martinez, T. J. *J. Chem. Theory Comput.* **2008**, *4*, 222−231.
(21) Ufimtsev, I. S.; Martinez, T. J. *J. Chem. Theory Comput.* **2009**, *5*, 1004−1015.
(22) Isborn, C. M.; Luehr, N.; Ufimtsev, I. S.; Martinez, T. J. *J. Chem. Theory Comput.* **2011**, *7*, 1814−1823.
(23) Hohenstein, E. G.; Luehr, N.; Ufimtsev, I. S.; Martinez, T. J. *J. Chem. Phys.* **2015**, *142*, 224103.
(24) Asadchev, A.; Allada, V.; Felder, J.; Bode, B. M.; Gordon, M. S.; Windus, T. L. *J. Chem. Theory Comput.* **2010**, *6*, 696−704.
(25) Wilkinson, K.; Skylaris, C.-K. *J. Comput. Chem.* **2013**, *34*, 2446−2459.
(26) Carsky, P.; Curik, R. *Theor. Chem. Acc.* **2014**, *133*, 1466.
(27) Yoshikawa, T.; Nakai, H. *J. Comput. Chem.* **2015**, *36*, 164−170.
(28) Maia, J. D. C.; Urquiza Carvalho, G. A.; Mangueira, C. P.; Santana, S. R.; Cabral, L. A. F.; Rocha, G. B. *J. Chem. Theory Comput.* **2012**, *8*, 3072−3081.
(29) Kussmann, J.; Ochsenfeld, C. *J. Chem. Theory Comput.* **2015**, *11*, 918−922.
(30) Miao, Y.; Merz, K. M. *J. Chem. Theory Comput.* **2015**, *11*, 1449−1462.
(31) Chow, E.; Liu, X.; Smelyanskiy, M.; Hammond, J. R. *J. Chem. Phys.* **2015**, *142*, 104103.
(32) Vogt, L.; Olivares-Amaya, R.; Kermes, S.; Shao, Y.; Amador-Bedolla, C.; Aspuru-Guzik, A. *J. Phys. Chem. A* **2008**, *112*, 2049−2057.
(33) Olivares-Amaya, R.; Watson, M. A.; Edgar, R. G.; Vogt, L.; Shao, Y.; Aspuru-Guzik, A. *J. Chem. Theory Comput.* **2010**, *6*, 135−144.
(34) DePrince, A. E.; Hammond, J. R. *J. Chem. Theory Comput.* **2011**, *7*, 1287−1295.
(35) Ma, W.; Krishnamoorthy, S.; Villa, O.; Kowalski, K. *J. Chem. Theory Comput.* **2011**, *7*, 1316−1327.
(36) Bhaskaran-Nair, K.; Ma, W.; Krishnamoorthy, S.; Villa, O.; van Dam, H. J.; Aprá, E.; Kowalski, K. *J. Chem. Theory Comput.* **2013**, *9*, 1949−1957.
(37) DePrince, A. E.; Kennedy, M. R.; Sumpter, B. G.; Sherrill, C. D. *Mol. Phys.* **2014**, *112*, 844−852.
(38) Maurer, S. A.; Kussmann, J.; Ochsenfeld, C. *J. Chem. Phys.* **2014**, *141*, 051106.
(39) Jindal, N.; Lotrich, V.; Deumens, E.; Sanders, B. *Int. J. Parallel Prog.* **2014**, 1−16.
(40) Aprà, E.; Klemm, M.; Kowalski, K. Efficient Implementation of Many-body Quantum Chemical Methods on the Intel® Xeon Phi™ Coprocessor. *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, Piscataway, NJ, USA, 2014; pp 674−684.
(41) Fales, B. S.; Levine, B. G. *J. Chem. Theory Comput.* **2015**, *11*, 4708−4716. PMID: 26574260.
(42) Balkova, A.; Bartlett, R. *J. Chem. Phys.* **1994**, *101*, 8972−8987.
(43) Li, X. Z.; Paldus, J. *J. Chem. Phys.* **2006**, *124*, 034112.
(44) Demel, O.; Pittner, J. *J. Chem. Phys.* **2006**, *124*, 144112.
(45) Bhaskaran-Nair, K.; Demel, O.; Pittner, J. *J. Chem. Phys.* **2008**, *129*, 184105.
(46) Bhaskaran-Nair, K.; Demel, O.; Smydke, J.; Pittner, J. *J. Chem. Phys.* **2011**, *134*, 154106.
(47) Evangelista, F. A.; Prochnow, E.; Gauss, J.; Schaefer, H. F., III *J. Chem. Phys.* **2010**, *132*, 074107.
(48) Demel, O.; Pittner, J.; Neese, F. *J. Chem. Theory Comput.* **2015**, *11*, 3104−3114.
(49) Bhaskaran-Nair, K.; Kowalski, K. *J. Chem. Phys.* **2012**, *137*, 216101.
(50) Kowalski, K.; Piecuch, P. *J. Mol. Struct.: THEOCHEM* **2001**, *547*, 191−208.
(51) Pittner, J.; Piecuch, P. *Mol. Phys.* **2009**, *107*, 1209−1221.
(52) Kowalski, K. *J. Chem. Phys.* **2011**, *134*, 194107.
(53) Jeziorski, B.; Monkhorst, H. *Phys. Rev. A: At., Mol., Opt. Phys.* **1981**, *24*, 1668−1681.
(54) Meissner, L.; Jankowski, K.; Wasilewski, J. *Int. J. Quantum Chem.* **1988**, *34*, 535−557.
(55) Jeziorski, B.; Paldus, J. *J. Chem. Phys.* **1988**, *88*, 5673−5687.
(56) Piecuch, P.; Paldus, J. *Theor. Chim. Acta* **1992**, *83*, 69−103.
(57) Paldus, J.; Piecuch, P.; Pylypow, L.; Jeziorski, B. *Phys. Rev. A: At., Mol., Opt. Phys.* **1993**, *47*, 2738−2782.
(58) Piecuch, P.; Paldus, J. *Phys. Rev. A: At., Mol., Opt. Phys.* **1994**, *49*, 3479−3514.
(59) Kucharski, S.; Bartlett, R. *J. Chem. Phys.* **1991**, *95*, 8227−8238.
(60) Balkova, A.; Kucharski, S.; Meissner, L.; Bartlett, R. *Theor. Chim. Acta* **1991**, *80*, 335−348.
(61) Balkova, A.; Kucharski, S.; Bartlett, R. *Chem. Phys. Lett.* **1991**, *182*, 511−518.
(62) Kucharski, S.; Balkova, A.; Szalay, P.; Bartlett, R. *J. Chem. Phys.* **1992**, *97*, 4289−4300.
(63) Másik, J.; Hubac, I. *Adv. Quantum Chem.* **1998**, *31*, 75−104.
(64) Pittner, J.; Nachtigall, P.; Carsky, P.; Masik, J.; Hubac, I. *J. Chem. Phys.* **1999**, *110*, 10275−10282.
(65) Pittner, J. *J. Chem. Phys.* **2003**, *118*, 10876−10889.
(66) Mahapatra, U. S.; Datta, B.; Mukherjee, D. *Mol. Phys.* **1998**, *94*, 157−171.
(67) Mahapatra, U. S.; Datta, B.; Bandyopadhyay, B.; Mukherjee, D. *Adv. Quantum Chem.* **1998**, *30*, 163−193.
(68) Mahapatra, U. S.; Datta, B.; Mukherjee, D. *J. Chem. Phys.* **1999**, *110*, 6171−6188.
(69) Evangelista, F. A.; Allen, W. D.; Schaefer, H. F. *J. Chem. Phys.* **2007**, *127*, 024102.
(70) Evangelista, F. A.; Simmonett, A. C.; Allen, W. D.; Schaefer, H. F., III; Gauss, J. *J. Chem. Phys.* **2008**, *128*, 124104.
(71) Das, S.; Mukherjee, D.; Kallay, M. *J. Chem. Phys.* **2010**, *132*, 074103.
(72) Das, S.; Kallay, M.; Mukherjee, D. *J. Chem. Phys.* **2010**, *133*, 234110.
(73) Hanrath, M. *J. Chem. Phys.* **2005**, *123*, 084102.
(74) Hanrath, M. *Chem. Phys. Lett.* **2006**, *420*, 426−431.
(75) Hanrath, M. *Theor. Chem. Acc.* **2008**, *121*, 187−195.
(76) Kong, L. *Int. J. Quantum Chem.* **2009**, *109*, 441−447.
(77) Kowalski, K.; Piecuch, P. *Mol. Phys.* **2004**, *102*, 2425−2449.
(78) Bhaskaran-Nair, K.; Brabec, J.; Aprà, E.; van Dam, H. J. J.; Pittner, J.; Kowalski, K. *J. Chem. Phys.* **2012**, *137*, 094112.
(79) Hirata, S. *J. Phys. Chem. A* **2003**, *107*, 9887−9897.
(80) Blackford, S.; et al. *ACM Trans. Math. Softw.* **2002**, *28*, 135−151.
(81) Kowalski, K.; Krishnamoorthy, S.; Olson, R. M.; Tipparaju, V.; Aprà, E. Scalable Implementations of Accurate Excited-state Coupled Cluster Theories: Application of High-level Methods to Porphyrin-based Systems. *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*, New York, NY, USA, 2011; pp 72:1−72:10.
(82) Intel-Corporation. *Intel Xeon Phi Coprocessor Instruction Set Architecture Reference Manual*, document number 327364-001; 2012.
(83) Dunning, T. H. *J. Chem. Phys.* **1989**, *90*, 1007−1023.
(84) Hehre, W.; Ditchfield, R.; Pople, J. *J. Chem. Phys.* **1972**, *56*, 2257.
(85) Sekino, H.; Kobayashi, H. *J. Chem. Phys.* **1981**, *75*, 3477−3484.
(86) Martin, J.; El-Yazal, J.; Francois, J. *Chem. Phys. Lett.* **1996**, *248*, 345−352.
(87) Brabec, C.; Anderson, E.; Davidson, B.; Kajihara, S.; Zhang, Q.; Bernholc, J.; Tomanek, D. *Phys. Rev. B: Condens. Matter Mater. Phys.* **1992**, *46*, 7326−7328.
(88) Zhang, B.; Wang, C.; Ho, K.; Xu, C.; Chan, C. *J. Chem. Phys.* **1992**, *97*, 5007−5011.
(89) Raghavachari, K.; Strout, D.; Odom, G.; Scuseria, G.; Pople, J.; Johnson, B.; Gill, P. *Chem. Phys. Lett.* **1993**, *214*, 357−361.
(90) Vonhelden, G.; Hsu, M.; Gotts, N.; Bowers, M. *J. Phys. Chem.* **1993**, *97*, 8182−8192.
(91) Grossman, J.; Mitas, L.; Raghavachari, K. *Phys. Rev. Lett.* **1995**, *75*, 3870−3873.
(92) Taylor, P.; Bylaska, E.; Weare, J.; Kawai, R. *Chem. Phys. Lett.* **1995**, *235*, 558−563.
(93) Bylaska, E.; Taylor, P.; Kawai, R.; Weare, J. *J. Phys. Chem.* **1996**, *100*, 6966−6972.
(94) Galli, G. *Comput. Mater. Sci.* **1998**, *12*, 242−258.

DOI: 10.1021/acs.jctc.5b00957
J. Chem. Theory Comput. 2016, 12, 1129−1138

(95) Sokolova, S.; Luchow, A.; Anderson, J. *Chem. Phys. Lett.* **2000**, *323*, 229−233.
(96) Grimme, S.; Muck-Lichtenfeld, C. *ChemPhysChem* **2002**, *3*, 207.
(97) An, W.; Gao, Y.; Bulusu, S.; Zeng, X. *J. Chem. Phys.* **2005**, *122*, 204109.
(98) An, Y.-P.; Yang, C.-L.; Wang, M.-S.; Ma, X.-G.; Wang, D.-H. *J. Chem. Phys.* **2009**, *131*, 024311.
(99) Heaton-Burgess, T.; Yang, W. *J. Chem. Phys.* **2010**, *132*, 234113.
(100) Jin, Y.; Perera, A.; Lotrich, V. F.; Bartlett, R. J. *Chem. Phys. Lett.* **2015**, *629*, 76−80.
(101) Brabec, J.; van Dam, H. J.; Pittner, J.; Kowalski, K. *J. Chem. Phys.* **2012**, *136*, 124102.
(102) Bhaskaran-Nair, K.; Kowalski, K. *J. Chem. Phys.* **2013**, *138*, 204114.
(103) Banik, S.; Ravichandran, L.; Brabec, J.; Hubač, I.; Kowalski, K.; Pittner, J. *J. Chem. Phys.* **2015**, *142*, 114106.

DOI: 10.1021/acs.jctc.5b00957
J. Chem. Theory Comput. 2016, 12, 1129−1138