JCTC Journal of Chemical Theory and Computation

Parallel Calculation of CCSD and CCSD(T) Analytic First and Second Derivatives

Michael E. Harding, Thorsten Metzroth, and Jürgen Gauss

Institut für Physikalische Chemie, Universität Mainz, Jakob-Welder-Weg 11, D-55099 Mainz, Germany

Alexander A. Auer*

Institut für Chemie, Technische Universität Chemnitz, Strasse der Nationen 62, D-09111 Chemnitz, Germany

Received June 25, 2007

Abstract: In this paper we present a parallel adaptation of a highly efficient coupled-cluster algorithm for calculating coupled-cluster singles and doubles (CCSD) and coupled-cluster singles and doubles augmented by a perturbative treatment of triple excitations (CCSD(T)) energies, gradients, and, for the first time, analytic second derivatives. A minimal-effort strategy is outlined that leads to an amplitude-replicated, communication-minimized implementation by parallelizing the time-determining steps for CCSD and CCSD(T). The resulting algorithm is aimed at affordable cluster architectures consisting of compute nodes with sufficient memory and local disk space and that are connected by standard communication networks like Gigabit Ethernet. While this scheme has disadvantages in the limit of very large numbers of compute nodes, it proves to be an efficient way of reducing the overall computational time for large-scale coupled-cluster calculations. In this way, CCSD(T) calculations of molecular properties such as vibrational frequencies or NMR-chemical shifts for systems with more than 1000 basis functions are feasible. A thorough analysis of the time-determining steps for CCSD and CCSD(T) energies, gradients, and second derivatives is carried out. Benchmark calculations are presented, proving that the parallelization of these steps is sufficient to obtain an efficient parallel scheme. This also includes the calculation of parallel CCSD energies and gradients using unrestricted (UHF) and restricted open-shell (ROHF) Hartree−Fock references, parallel UHF-CCSD(T) energies and gradients, parallel ROHF-CCSD(T) energies as well as parallel equation-of-motion CCSD energies and gradients for closed- and open-shell references. First applications to the calculation of the NMR chemical shifts of benzene using large basis sets and to the calculation of the equilibrium geometry of ferrocene as well as energy calculations with more than 1300 basis functions demonstrate the efficiency of the implementation.

1. Introduction

In electronic structure theory coupled-cluster (CC) methods have become a standard tool for high accuracy calculations.[1−5] With the exception of some difficult cases like multireference systems or cases of reference orbital instabilities, methods from the CC hierarchy represent robust black-box approaches providing increasing accuracy and a fast, systematic convergence to the full configuration-interaction (FCI) result. However, application of CC methods to larger chemical problems is limited by the rapidly increasing

* Corresponding author e-mail: alexander.auer@chemie.tu-chemnitz.de.

10.1021/ct700152c CCC: $40.75
Published on Web 11/16/2007

computational effort with growing number of electrons and basis functions.

A detailed analysis reveals that for CC methods like the coupled-cluster singles and doubles (CCSD)[6] and the coupled-cluster singles and double scheme augmented by a perturbative treatment of triple excitations (CCSD(T))[7] the limiting factor is CPU time and not storage requirements. If N is chosen as a measure of the system size, storage requirements for two-electron integrals, coupled-cluster amplitudes, and intermediates scale as N^4. In contrast to that, the operation count scales as N^6 for CCSD and N^7 for CCSD(T). Nowadays, efficient implementations allow calculations at the CCSD(T) level of theory with up to 800 basis functions.[8−11]

While for almost all methods numerous efficient parallel algorithms have been developed, the number of parallel implementations of CC methods has only increased in recent years.[12−19] Of note are the highly sophisticated algorithm for parallel calculation of CCSD(T) energies presented by Olson et al.[11] (within the program package GAMESS[18]) and an implementation of CCSD energies by Janowski et al.[20] (within the program package PQS[19]). In contrast to the algorithm presented in this work, the parallel implementation of CCSD(T) energies by Olson et al. is tailored to multiprocessor and/or multicore nodes connected by a dedicated communication network and based on the Distributed Data Interface (DDI/3).[11,21] And while Janowski et al. present CCSD energy calculations with more than 1500 basis functions on more than 30 compute nodes, their approach is based on the Array Files (AF)[20,22] scheme. We will focus on an ansatz which works without an additional layer of complexity provided by specialized libraries like DDI/3 or AF. The presented scheme is based on the message passing interface (MPI),[23] and all nonparallel steps run redundantly on every available processor at the same time.

To the best of our knowledge, however, no CC code capable of calculating general second-order molecular properties at the CC level using analytical derivatives has been adapted for parallel architectures. The main reason for this is that the mathematical structure of the CC equations makes an efficient fully parallel implementation or reimplementation demanding and time-consuming. In this paper we demonstrate an alternative approach, namely, the adaptation of an efficient serial algorithm to parallel environments.

The employed strategy is presented in a stepwise manner leading to an algorithm with parallelized routines for the time-determining steps in the CCSD and CCSD(T) energy, gradient, and analytical second-derivative calculations. We present benchmarks of large-scale CC applications using the Mainz−Austin−Budapest version of the ACES II program package[24] (ACES II MAB) modified in this way. A detailed investigation of the time-determining steps in CCSD and CCSD(T) calculations and the reduction of the overall execution time in the parallel algorithm is carried out.

2. Parallelization Strategy for CC Energies and Derivatives

A common approach for the parallelization of CC algorithms is to minimize storage requirements while aiming at constant (but in practice high) total communication by distributing parts of integrals, amplitudes, and intermediates and communicating the pieces as needed by other processors. While this approach guarantees a proper scaling of the algorithm in the limit of a large number of processors,[16,25] high-speed and expensive network connections are required. Furthermore, the structure of such an algorithm as well as the communication overhead, arising through dead times in which a node awaits data, may shift the crossover point with respect to efficient serial algorithms to a large numbers of nodes.

Following a different route, we apply a replicated storage scheme in order to minimize communication. Most of the quantities needed in the CC iterations are stored completely on every node in order to avoid communication of intermediate quantities. In contrast to the algorithm outlined by Olson et al.,[11] the work presented here is tailored to cluster architectures with moderate hardware specifications, assuming relatively slow interconnect structures like Gigabit Ethernet. Furthermore, it is assumed that memory is available to store the full set of T_1 and T_2 amplitudes locally on every node in fast memory and on hard disk. In this way, communication is minimized as only the CC amplitudes have to be communicated. The storage of the amplitudes rarely becomes a bottleneck: If one assumes a molecule with 20 occupied and 600 virtual spin orbitals, which would correspond to a basis set of better than quadruple-ζ quality, then the number of T_2 amplitudes, which scales as occ^2vrt^2, [26] would be of the order of a few hundred millions, which roughly corresponds to 1.5 GB of memory, if no symmetry is used.[27]

In the actual algorithm, parts of intermediates or integrals are contracted with the amplitudes on different nodes to give parts of the resulting quantity. In a final step, the amplitudes are updated and broadcast to all nodes. While allowing for distributed contractions during the CC iterations at minimal communication, this strategy has two drawbacks. Primarily, it does not allow for optimal scaling in the limit of a large number of processors as the distribution costs scale linearly with the size of the distributed entity and the number of processors. The exact scaling behavior for the communication depends on the employed communication hardware and the used algorithm. Furthermore, it does not reduce the storage requirements for the replicated quantities like the T_2 amplitudes or the molecular orbital (MO) integrals excluding the four-virtual index integrals. These are treated in partial atomic orbital (AO) algorithms which eliminate the need for a full transformation of the two-electron integrals and only require storage of the AO integrals. The needed MO integrals are calculated once in a semiparallel way and then are fully stored on each node (see subsections 2.2 and 2.3). It is straightforward to calculate and store AO integrals, which are usually the largest quantity in terms of disk space in modern CC algorithms, in a distributed manner. Together with the increased availability of large and cheap directly attached disk space the distributed storage of AO integrals makes it obsolete to recalculate or approximate these in every new step. At the same time, the efficiency of this algorithm is improving for increasing example size: The time-

Table 1. Timings for the Perturbative Triples Step in CCSD(T) Energies Relative to the Total Walltime of the CC Part

molecule	basis set	no. of electrons	no. of basis functions	% of (T) in CCSD(T)
H_2O	cc-pCVTZ	10	115	13
H_2O	cc-pCVQZ	10	144	13
H_2O	cc-pCV5Z	10	218	13
Cl_2	cc-pCVTZ	34	118	52
Cl_2	cc-pCVQZ	34	218	52
benzene	cc-pCVDZ	42	138	51
benzene	cc-pCVTZ	42	354	57
hexachlorobenzene	cc-pVDZ	168	192	60

determining steps can be more efficiently parallelized (see 3). However, calculations that are not feasible due to memory or disk space limitations (for the MO integrals) will also not be feasible when multiple processors are used.

For methods like CCSD or CCSD(T), communication costs associated with the replication of T_2-like quantities are usually at least 2 orders of magnitude smaller than the CPU time required for their computation. For CCSD(T), the scaling of CPU time is N^7 ($occ^3\upsilon rt^4$), while storage and communication costs grow as $occ^2\upsilon rt^2$ per compute node. Thus, the distribution of the time-determining steps to a number of processors in the way described above leads to a major reduction of overall walltime, especially when large examples are considered. A detailed discussion and examples for the different aspects of this parallelization strategy will be given in the next sections.

As the basic outline of the formalism used here and the common algorithms that are the starting point for our current work have been described in several publications,[7,28−32] we will not reiterate them but rather give details only for the steps modified in our approach.

In subsection 2.1 we will describe the parallelization of the CCSD(T) perturbative triples part for energies, gradients, and second derivatives starting from an implementation[28−30] that proves to be an ideal structure for the adaptation to parallel architectures.

In subsection 2.2 we carry out an analysis of the time-determining steps in CCSD energy, gradient, and second derivative calculations and describe the modification of the AO-based calculation of the leading term (the so-called particle−particle ladder term that includes contraction over two virtual indices).

In subsection 2.3 further issues for the optimization of the parallel code are described concerning the evaluation of two-electron integrals, the Hartee−Fock self-consistent-field (HF-SCF) procedure, and the integral transformation. For all test calculations reported in these sections correlation-consistent and correlation-consistent core-valence Dunning basis sets[33−35] have been used throughout.

Finally (section 3), we present applications of the new algorithm with a detailed investigation of the scaling of overall time with the number of processors.

2.1. Parallel Algorithm for the Perturbative Triples Contributions to CCSD(T) Energies, Gradients, and Second Derivatives. The first step in the parallelization of the CCSD(T) scheme is to realize that almost all large CCSD(T) calculations are dominated by the calculation of the perturbative triples contribution. In Table 1 the timings for several standard serial CCSD(T) calculations are summarized. While the time-determining step for CCSD scales as $occ^2\upsilon rt^4$, the computational bottleneck of the perturbative triples correction scales as $occ^3\upsilon rt^4$. Thus, in comparison to CCSD the time spent for the (T)-correction more rapidly increases with the number of electrons, and this renders the computation of the perturbative triples correction the time-determining step in CCSD(T) calculations.

Our approach to parallelize the triples correction starts from the energy expressions[7]

$$E^{[4]} = \frac{1}{36}\sum_{ijk}\sum_{abc} t_{ijk}^{abc}\, D_{ijk}^{abc}\, t_{ijk}^{abc} \tag{1}$$

$$E^{[5]} = \frac{1}{4}\sum_{ijk}\sum_{abc} \langle jk||bc\rangle t_i^a\, t_{ijk}^{abc} \tag{2}$$

where $E^{[4]}$ and $E^{[5]}$ are energy contributions in fourth- and fifth-order perturbation theory, respectively. D_{ijk}^{abc} denotes the inverse orbital-energy denominator. As is the usual convention, i,j,k, ... denotes occupied and a,b,c, ... virtual spin orbitals. The perturbative-triple amplitudes t_{ijk}^{abc} are defined as

$$D_{ijk}^{abc}\, t_{ijk}^{abc} = P(k|ij)P(a|bc)\sum_e t_{ij}^{ae}\langle bc||ek\rangle - P(i|jk)P(c|ab)\sum_m t_{im}^{ab}\langle mc||jk\rangle \tag{3}$$

with $P(x|yz)$ being the cyclic permutation operator ($P(x|yz)$ $f(x,y,z) = f(x,y,z) + f(y,z,x) + f(z,x,y)$), t_i^a and t_{ij}^{ab} the CCSD amplitudes, and $\langle bc||ek\rangle$ the antisymmetrized two-electron integrals. The basic scheme utilized in the ACES II MAB algorithm for the formation of the T_3 amplitudes is an outer loop over an index triple i,j,k of the t_{ijk}^{abc} amplitudes. For energy calculations, for example, blocks of a,b,c index triples are calculated within the loop one at a time and immediately used to form the $E^{(4)}$ and $E^{(5)}$ energy contributions. In this way, storage of the full triples amplitudes is circumvented, as has also been reported on many other occasions in the literature.[12,37,38]

If the T_1 and T_2 amplitudes and the corresponding integrals are fully or at least partially locally available on all nodes, each node can independently form a,b,c energy contributions. Only a single number per node, namely the summed up energy contributions, has to be communicated. In a final step, the energy contributions from the i,j,k blocks are summed up to give the total energy correction. In this way, the parallelization of the (T) energy contributions can be achieved in a straightforward fashion.

For CCSD(T) gradients, Watts et al.[38] describe an algorithm which following an idea of Lee and Rendell[39] avoids recomputation of amplitudes due to the use of perturbed canonical orbitals. Here, the outer loop runs again over the index triples i,j,k, and after the formation of an a,b,c block of T_3 amplitudes not only the energy increment but also the

Table 2. Timings for the Parallel Perturbative Triples Step in CCSD(T) Energy Calculations, Geometry Optimizations (One Iteration), and the Calculation of NMR Chemical Shifts as Analytical Second Derivatives for the Benzene Molecule[a]

	number of nodes				
	1	2	4	8	16
	cc-pCVDZ (138)				
energy	75	40	19	9	5
geometry	251	126	63	31	16
NMR shieldings	2936	1452	727	363	192
	cc-pCVTZ (342)				
energy	2163	1081	540	269	146
geometry	6594	3241	1619	809	426
NMR shieldings	80779	40285	20171	10527	5171
	cc-pCVQZ (684)				
energy	29019	14514	7225	3592	1758
geometry	82171	40999	20425	10207	5489
	cc-pCV5Z (1200)				
energy	238882	119469	59764	29895	15184

[a] Walltime in s. The number of basis functions is given in parentheses.

contributions to the unperturbed effective one-particle density matrices, the two-particle density matrices, and the contributions to the inhomogeneous terms of the Λ equations have to be calculated from the available T_3 block. In CCSD(T) second derivative calculations,[29,31] the same loop structure is used to construct the perturbed triple amplitues $\partial t_{ijk}^{abc}/\partial x$ and $\partial \tilde{t}_{ijk}^{abc}/\partial x$[40] and the corresponding contributions to the perturbed density matrices as well as to the perturbed Λ equations for one perturbation at a time. Such a strategy is only possible when using an asymmetric expression for the second derivatives.[29,31] This issue renders the asymmetric strategy the preferred choice over an alternative symmetric approach which requires the simultaneous availability of all perturbed amplitudes. However, while for energy calculations only one double-precision quantity needs to be communicated, for gradients and second derivatives the corresponding contributions to the two-particle density matrices must be exchanged and summed as well.

To illustrate the efficiency of this scheme, energy calculations, geometry optimizations, and calculations of NMR chemical shifts have been carried out for the benzene molecule using Dunning's correlation consistent core-valence basis sets.[41] The timings for the perturbative triples part of the algorithm are displayed in Table 2.[42]

The timings for the perturbative energy correction in the CCSD(T) algorithm scale almost perfectly up to 16 processors, even for the smallest basis set. It should be noted that the communication time required for the distribution of intermediate quantities calculated in the *a*,*b*,*c* loop of the perturbative triples is typically of the order of at most a few minutes, using Gigabit Ethernet interconnection. This is even the case for the largest examples and the largest numbers of nodes tested so far. In contrast to this, the time required for the parallel computation of the triples quantities themselves is usually of the order of hours for these examples.

In this way, using a simple scheme for the adaptation of the serial code to cluster architectures, the overall time for the most CPU-time intensive steps in CCSD(T) calculations can be scaled down efficiently. However, the required effort for the underlying CCSD calculation that precedes the calculation of the perturbative triples has not been discussed so far but now becomes the dominant step in the overall execution time. The next section focuses on this issue.

2.2. Analysis and Parallelization of CCSD Energy, Gradient, and Second-Derivative Calculations. From the previous section it is obvious that the straightforward parallelization of the (T) step in large-scale CCSD(T) calculations allows a significant reduction of the overall execution time up to a certain point. Increasing the number of nodes further, however, does not lead to an additional gain in execution time, if the effort for the underlying serial CCSD calculation exceeds the time for the parallel calculation of the perturbative triples. Thus, the next meaningful step in the parallelization of the CCSD(T) method is to identify and to parallelize bottlenecks in the CCSD algorithm. The time-determining steps in a CCSD energy calculation are the so-called particle−particle ladder terms that scale as $occ^2\nu rt^4$ [32,43]

$$t_{ij}^{ab}\, D_{ij}^{ab} \leftarrow \frac{1}{2}\sum_{ef}\tau_{ij}^{ef}\langle ab||ef\rangle \tag{4}$$

where the intermediate

$$\tau_{ij}^{ab} = t_{ij}^{ab} + t_i^a\, t_j^b - t_i^b\, t_j^a \tag{5}$$

is used.

It should be noted that for CC energy and derivative calculations terms including $\langle ab||cd\rangle$ integrals or corresponding integral derivatives can, in general, be identified as the contributions with the highest scaling. For large basis sets the quartic dependence on the number of virtual indices will usually render this contraction expensive in terms of computational time.

One problem of the formulation in eq 4 is that the molecular-orbital integrals always represent a storage bottleneck, due to their lack of sparsity. As a consequence, the common practice in modern CC programs is an AO integral-driven algorithm in which the corresponding amplitudes are first partially transformed to the AO basis in an N^5 procedure

$$\tau_{ij}^{\mu\nu} = \sum_{ef} c_{\mu e} c_{\nu f} \tau_{ij}^{ef} \tag{6}$$

and then contracted with the AO integrals driven by the order in which integrals are retrieved from disk:

$$Z_{ij}^{\mu\nu} = \frac{1}{2}\sum_{\sigma\rho}\langle\mu\nu||\sigma\rho\rangle\tau_{ij}^{\sigma\rho} \tag{7}$$

Afterwards, the resulting intermediate will be back transformed and processed in the MO basis[44−50] as follows:

$$Z_{ij}^{ab} = \sum_{\mu\nu} c_{\mu a} c_{\nu b} Z_{ij}^{\mu\nu} \tag{8}$$

Olson et al.[11] give a detailed discussion on integral storage requirements and typical file size dimension.

Table 3. Timings per CCSD Iteration in Comparison to the Particle−Particle Ladder Term for the Benzene Molecule[a]

	number of nodes				
	1	2	4	8	16
	cc-pCVDZ (138)				
time per CCSD iteration	5.2	3.9	3.0	2.5	2.4
time for AO ladder term	3.4	2.0	1.1	0.6	0.5
	cc-pCVTZ (342)				
time per CCSD iteration	135	79	52	39	33
time for AO ladder term	113	57	31	16	9
	cc-pCVQZ (684)				
time per CCSD iteration	1902	1027	584	365	257
time for AO ladder term	1761	885	445	226	115

[a] Walltime in s. The number of basis functions is given in parentheses.

The use of partial AO algorithms has the advantage that only the more sparse AO integrals need to be stored at the expense of an additional transformation. While the operation count of the time-determining step scales as occ^2ao^4 in this scheme, in practice the reduced number of AO integrals also leads to a significantly reduced I/O and an overall saving of walltime. Thus, in almost all relevant cases the AO based algorithm outperforms the straightforward MO based scheme. Realizing that parallelizing this single contribution will lead to a major reduction of overall time in each CCSD iteration, we have chosen this AO based term as a starting point to improve our parallel CCSD(T) code.

The basic loop structure for which the power of multiple processors can be used effectively is an outer loop over batches of AO integrals that are read from disk and contracted with all matching T_2 amplitudes in the AO basis. After the contraction has been carried out, the resulting Z_{ij}^{ab} intermediate is communicated, and the CCSD iteration is continued.

It should be noted that the communication required after parts of the corresponding intermediate have been formed on all nodes scales at most as occ^2vrt^2 per compute node. It is expected that this step can be parallelized efficiently without running into communication bottlenecks.

Table 3 shows the timings of the CCSD iterations for the benzene molecule, where the AO ladder term has been parallelized in the described fashion.

As can be seen in the first column, the calculation of the AO-based particle−particle ladder term dominates the time for one iteration, even for the smallest examples by more than 60%. Furthermore, the parallelization of this term in batches of AO integrals results in an almost perfect reduction of the walltime for this contribution up to 16 processors and, thus, to a significant reduction of the overall time per iteration, especially for larger examples.

For the calculation of analytic gradients terms analogous to those in energy calculations appear in the CCSD Λ equations:[32]

$$\lambda_{ij}^{ab} D_{ij}^{ab} \leftarrow \frac{1}{2}\sum_{ef} \lambda_{ij}^{ef} \langle ef||ab\rangle \tag{9}$$

Table 4. Timings for the Solution of the Lambda Equations and the Solution of the Perturbed Amplitude and Lambda Equations for the Benzene Molecule[a]

	number of nodes				
	1	2	4	8	16
	Lambda Equations				
cc-pCVDZ	65	48	40	35	34
cc-pCVTZ	1819	1147	828	653	603
cc-pCVQZ	25166	14808	9510	6887	5854
	Perturbed Amplitude and Lambda Equations				
cc-pCVDZ	607	459	390	345	331
cc-pCVTZ	18526	12190	9255	7972	6022

[a] Walltime in s.

Within the ACES II MAB program package this term is calculated using the same AO integral-driven scheme. Thus, the time-determining N^6 step in the gradient calculations can be parallelized in the same way as the corresponding term in the energy calculation.

For second derivative calculations, the contributions that have to be considered occur in the equations for the perturbed cluster and perturbed Λ amplitudes:

$$\frac{\partial t_{ij}^{ab}}{\partial \chi} D_{ij}^{ab} \leftarrow \frac{1}{2}\sum_{ef} \frac{\partial t_{ij}^{ef}}{\partial \chi} \langle ab||ef\rangle \tag{10}$$

$$\frac{\partial \lambda_{ij}^{ab}}{\partial \chi} D_{ij}^{ab} \leftarrow \frac{1}{2}\sum_{ef} \frac{\partial \lambda_{ij}^{ef}}{\partial \chi} \langle ab||ef\rangle \tag{11}$$

By parallelizing these contributions in the AO based scheme, the overall computational cost of the most time-consuming steps in the CCSD gradient and analytical second derivative calculations can be reduced as well. Due to the dominance of these steps compared to the overall time per CCSD iteration this simple strategy improves the overall CCSD time significantly if multiple processors are used.

Table 4 summarizes the timings for the corresponding modules that include the steps described above for calculations of NMR chemical shifts for the benzene molecule.

From the last columns of Tables 2 and 4 it becomes clear that for this example the overall time required for the CCSD- and CCSD(T)-derivative equations is of the same order as the time for the evaluation of the perturbative triples part, if 16 processors are used. Thus, the CCSD part of the calculation will still dominate the overall time for CCSD(T) derivative calculations if more than 16 processors are used. This is mainly due to contributions that have not been parallelized, which include terms of lower scaling, integral derivative transformations, etc.

So far only the particle−particle ladder terms for the CCSD, Λ, perturbed amplitudes and perturbed Λ equations are parallelized.

To further improve the algorithm, one could utilize parallel matrix multiplication routines for CCSD contributions that scale as occ^3vrt^3. In addition, for the special case of analytic second derivatives, one could also use a coarse-grained parallelization scheme[51] on top of the one suggested here.

Table 5. Timings for the Calculation of Two-Electron Integrals, the HF-SCF, and the Integral Transformation for the Benzene Molecule[a]

	number of nodes				
	1	2	4	8	16
	cc-pCVDZ (138)				
integral evaluation	8.5	4.6	2.2	1.1	0.7
HF-SCF	1.8	1.1	0.6	0.4	0.3
integral transformation	1.8	1.4	1.3	1.1	1.3
	cc-pCVTZ (342)				
integral evaluation	340	167	90	45	24
HF-SCF	57	30	16	9	5
integral transformation	61	37	29	23	21
	cc-pCVQZ (684)				
integral evaluation	19379	10012	4906	2518	1505
HF-SCF	1009	453	225	122	67
integral transformation	1096	601	399	267	233

[a] Walltime in s. The number of basis functions is given in parentheses.

Namely, one could distribute the different perturbations that are calculated independently, for example B_x, B_y, and B_z for NMR chemical shifts or geometric perturbations in the case of harmonic frequencies, to different processors. While this has not been carried out in the approach presented here, it is a straightforward addition to any code that could further help to improve the scaling of the algorithm with the number of processors.

However, the crossover point from which the preceding CCSD calculation will dominate over the CCSD(T) calculation time is pushed to a larger number of processors for larger examples. For larger molecules with medium-sized basis sets this crossover point should shift to 32 or even 64 processors, so that large-scale cluster architectures could be used to carry out calculations within days that would take months on a single processor.

2.3. Further Optimization Issues. The scheme for parallelizing the AO ladder terms described in subsection 2.2 requires only equally distributed integrals to be present on the different compute nodes. As a consequence, the evaluation of the integrals is carried out in parallel and in turn used in the parallel framework of the HF-SCF procedure and the integral transformation. Each node calculates and stores only a part of the integrals, and thus during the HF-SCF procedure only an incomplete Fock-matrix is built on each node, which is then exchanged between the compute nodes. The total Fock matrix is simply the sum of these incomplete matrices. The rest of the algorithm is unaltered. For the integral transformation all AO integrals are read in only once and communicated in the form of an intermediate array. After transformation of the MO integrals with two and three virtual indices each node stores all calculated MO integrals locally. Another non-negligible part of the derivative calculation is the evaluation of the integral derivatives which can be parallelized in an analogous manner. While this has not been done in the work presented here, it is the focus of future work, among other optimization issues.

The timings (in seconds) for parallel integral evaluation, HF-SCF, and integral transformation for the benzene molecule are shown in Table 5. This simple scheme results not only in a reduced storage requirement per node but also in a reduction of the overall time for the integral evaluation, transformation, and HF-SCF. It should be noted, that for some cases, even superlinear scaling of the evaluation of the two-electron integrals can be observed. This is due to automatic buffering schemes in the operating system that allow for a more efficient I/O if certain buffer sizes are reached and has also been reported by other groups.[52]

An important issue in parallel implementations is to avoid load balancing problems. In the work presented here, HF-SCF, the integral transformation, and any CCSD-like equations are automatically balanced by the local calculation and storage of equally sized amounts of two-electron integrals at the beginning of the calculation. The remaining steps in the calculation of the perturbative triples are balanced on average by the large number of these contributions. This applies for the calculation of energies, gradients, and any second-order properties. In practice, even multiprocessor systems do not show balancing problems since the load is kept equal on every processor. For the actual implementation we assume that dedicated nodes are available. However, load balancing problems will arise if heterogeneous resources are used or if compute nodes have different loads due to other calculations. This issue will have to be addressed in further developments of the current algorithm.

3. Results and Discussion

In this section we focus on the overall performance of the scheme presented here and the practicability for the usage on typical cluster architectures. The results of two applications are presented that outline typical problems in quantum chemistry for which high level ab initio methods are necessary but extremely time-consuming unless parallel implementations, like the one presented here, are applied.

Nowadays more than 300 GB of disk space and 8 GB of random access memory (RAM) are readily available even on single cluster nodes within medium sized computer clusters, so it is not foreseeable that memory or storage will present a bottleneck for larger calculations. As has been stressed before, only serial calculation times of the order of months or years will render large-scale CCSD(T) calculations infeasible. While a parallel implementation cannot combat the steep scaling of high-level CC methods, the power of parallel computer architectures can help to stretch the range of applicability far beyond what it has been in recent years.

In Table 6 the results of some representative benchmark calculations for energies and gradients are summarized. The timings for CCSD(T) energy calculations for benzene and cyclohexene and the timings of one step of the geometry optimization of the adamantyl cation ($C_{10}H_{15}^{+}$) are given.

For the high-symmetry case benzene in a hextuple-zeta basis set the serial energy calculation would take about 2 weeks and is reduced to less than a day using 16 processors. From Table 6 it also becomes obvious that the number of basis functions is not the only factor when considering the size of a system but also symmetry and the distribution of orbitals among the irreducible representations as well as the ratio of occupied to virtual orbitals.

Table 6. Overall Timings for the CCSD(T) Energy Calculations of Benzene and Cyclohexene and for One Step of the Geometry Optimization of the Adamantyl Cation

molecule	comput. symm	basis set	no. of basis functions	no. of nodes	execution time [h]
benzene[a]	D_{2h}	cc-pV5Z	876	16	4
benzene[a]	D_{2h}	cc-pV6Z	1386	16	21
cyclohexene[a,b]	C_2	aug-cc-pVQZ	940	16	40
adamantyl cation	C_s	cc-pVTZ	510	9	90

[a] Energies (frozen core) for benzene cc-pV5Z, cc-pV6Z, and cyclohexene are −231.8916163, −231.8987752, and −234.2797258 Hartree. [b] Carried out at the fc-MP2/cc-pVTZ geometry.

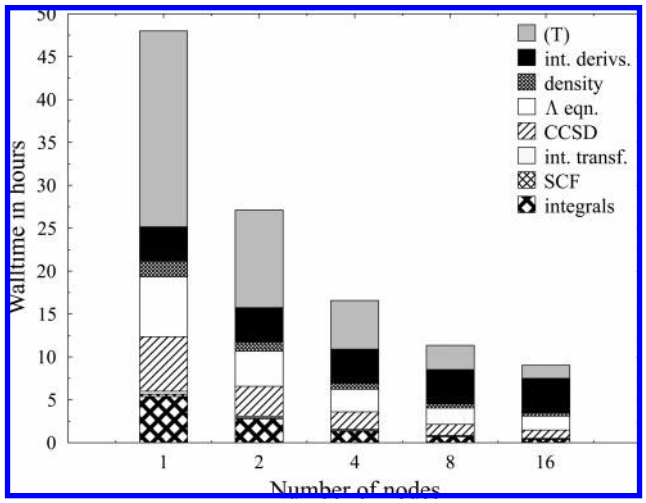

Figure 1. Composition of the overall walltime for one step in the geometry optimization of the benzene molecule at the CCSD(T)/cc-pCVQZ level of theory (684 basis functions).

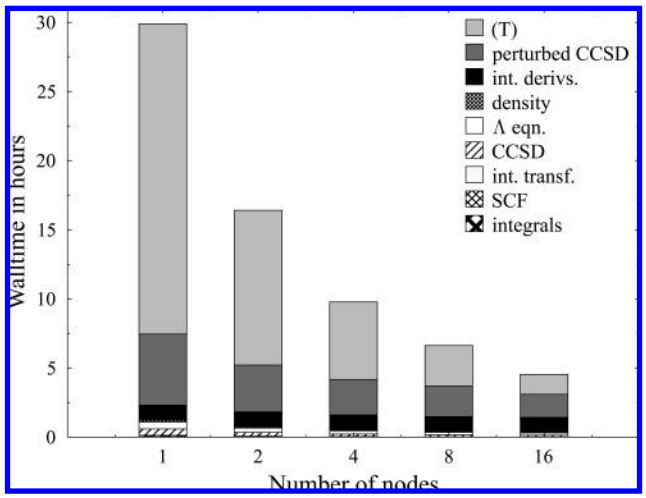

Figure 2. Composition of the overall walltime for the calculation of the NMR chemical shifts of the benzene molecule at the CCSD(T)/cc-pCVTZ level of theory (342 basis functions).

Figures 1 and 2 give a more detailed view on the different steps required for two smaller CCSD(T) calculations, namely the gradient for one step of a geometry optimization and for the calculation of NMR chemical shifts also for the benzene molecule.

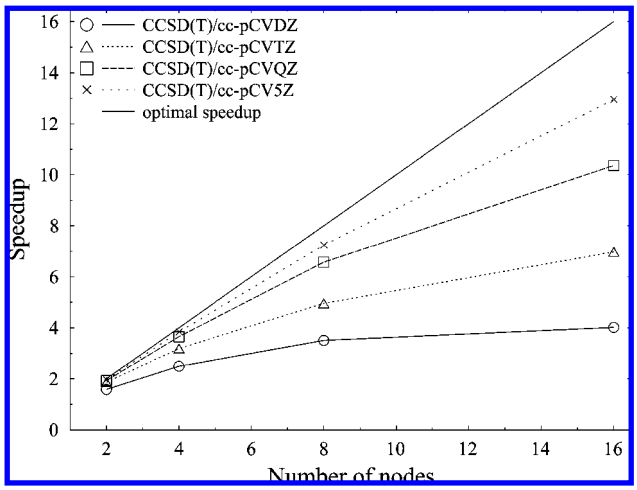

Figure 3. Parallel scaling of CCSD(T) energy calculations for the benzene molecule using different basis sets.

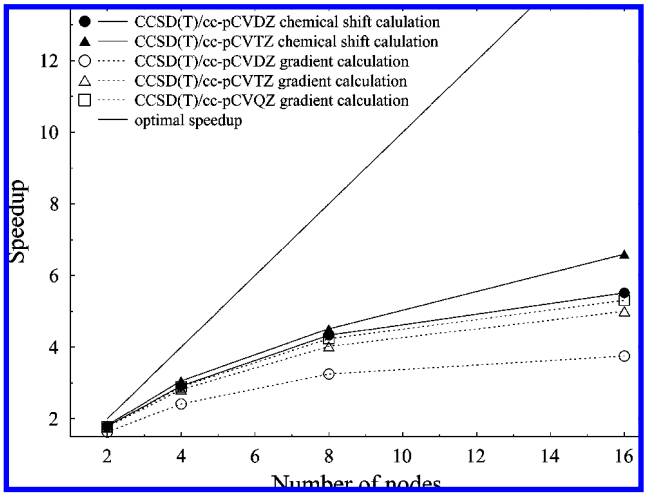

Figure 4. Parallel scaling of CCSD(T) first and second analytical energy derivative calculations for the benzene molecule using different basis sets.

For the gradient calculation, which takes about 47 h on a single processor, it can be seen that the calculation of the perturbative triples contribution takes only about half of the overall time of the optimization step. Using the current algorithm it is possible to scale down this contribution and also the calculation of the two-electron integrals, the CCSD, and Λ equations as well as the integral transformation and the contributions to the density matrices. Using 16 processors the overall time is reduced to less than 10 h. At this point steps dominate the overall time that have not been considered for parallelization in the algorithm, so that the usage of larger numbers of nodes would not yield significant further speedups.

The calculation of the NMR chemical shifts with a larger basis set shows a different profile. Here the perturbative triples contributions to the second derivatives clearly dominate compared to the other steps, such as the SCF or CCSD calculations. Using 16 processors the overall time of 30 h can be reduced to less than 5 h. After this point, the remaining steps in the perturbed CCSD equations that have not been parallelized dominate the overall calculation time.

Table 7. ^{13}C NMR Chemical Shifts for the Benzene Molecule Using Various Basis Sets[41] [a]

basis	no. of basis functions	absolute NMR shieldings CCSD(T)
cc-pCVDZ	138	82.07
cc-pCVTZ	342	66.61
tz2p[c]	198	68.42
qz2p[c]	228	64.95
pz3d2f[c]	474	63.22
vib corr[b]		−3.43
total		59.79
experiment		57.1[57]

[a] The experimental values have been taken from ref 57 using the absolute shifts of carbon monoxide ($\sigma_{T=300K}$=0.9 ± 0.9 ppm).[58] For a detailed description of the basis sets used and the scheme for the computation of the zero-point vibrational correction see ref 53. To avoid the gauge-origin problem in the calculation of NMR chemical shifts the gauge including atomic orbitals (GIAO)[59−61] approach has been used. [b] The vibrational correction is based on a perturbational approach.[53] The cubic force field was calculated at the MP2/cc-pVTZ and the NMR shieldings for the displacements at the MP2/qz2p level of theory. [c] The qz2p basis consists of a 11s7p2d/7s2p primitive set contracted to 6s4p2d/4s2p and the pz3d2f basis of a 13s8p3d2f/8s3p2d set contracted to 8s5p3d2f/5s3p2d.[62−65]

Figures 3 and 4 give detailed insight for the scaling of CCSD(T) energy and derivative calculations. As has been discussed in the previous sections, the scaling of total time with the number of nodes is improving for increasing system size as the importance of the parallelized, time-determining steps is even larger. Thus, this implementation will make applications feasible within weeks or even days that would take months or years to calculate, if 64 or 128 nodes were used.

The following examples give the proof of principle, that this simple scheme for adapting a serial implementation leads to an efficient algorithm for cluster architectures that can be used to reduce the overall time of large-scale CCSD(T) calculations to acceptable dimensions.

3.1. Benchmark Calculation for the ^{13}C NMR Chemical Shifts of Benzene. In a previous study it has been demonstrated that methods like CCSD(T) can be used to achieve an accuracy of 2−4 ppm deviation from experiment in the calculation of ^{13}C NMR chemical shifts.[53] While this study included 16 small organic molecules, of which the largest cases were CF_4 and acetone (CH_3COCH_3), the limitations of the serial implementation and the limited computational resources did not allow for the calculation of larger molecules. One example for which accurate benchmark results are of immediate interest is benzene as computational studies, especially applying density functional theory, on all kinds of substituted benzene species are abundant in the literature.[54−56] Thus, the parallel algorithm for the calculation of second-order properties described above has been applied to perform CCSD(T) calculations of the NMR chemical shifts of benzene using various basis sets in order to estimate basis set convergence. Here, the new algorithm allows the use of very large basis sets even for a system with 12 atoms and 42 correlated electrons.

The results including NMR chemical shifts and also zero-point vibrational corrections are given in Table 7. An analysis of the basis-set convergence leads to the conclusion that the Dunning basis sets that have been optimized for energies from post-HF correlation methods and that are fairly diffuse are not very suitable for the calculation of the NMR chemical shifts that probe the electron density closer to the nucleus. Even the Dunning core-valence basis sets that are augmented with tight functions do not perform as well as the corresponding Karlsruhe basis sets[62−65] if one aims at quantitative accuracy in the prediction of NMR chemical shifts.

3.2. The Equilibrium Structure of Ferrocene. Within the last 25 years many attempts were made to determine the structure of ferrocene by applying various quantum-chemical methods.[66−70] A more recent study[71] presented first calculations employing analytical CCSD(T) gradients on this problem using a relatively small basis set and the frozen-core approximation. Up to now, quantum-chemical models have great difficulties to determine the equilibrium metal−ligand distance, a quantity that is not directly accessible to experiment but often used for benchmark studies in the framework of density-functional theory. The structural parameters of ferrocene in its eclipsed (equilibrium) and staggered (saddle point) conformations have been determined using analytic CCSD(T) gradients correlating all 96 electrons with a full triple-ζ quality basis set. Using the cc-pVTZ basis set[33,72] (508 basis functions) one geometry cycle takes about 2.3 days when performing the calculation on 15 nodes. With the cc-pwCVTZ basis set[72,73] (572 basis functions) a geometry cycle takes about 8.8 days using 14 nodes. The results in comparison with previous coupled-cluster studies are presented in Tables 8 and 9. The coupled-cluster results show a relatively pronounced basis set dependence. Quadruple-ζ quality CCSD(T) calculations again correlating all electrons are underway.

4. Conclusions

A detailed analysis of the time-determining steps in CC energy, gradient, and second derivative calculations shows that for almost all practical applications only a few terms completely dominate the overall computation time. This motivates a straightforward strategy for the parallelization of CCSD and CCSD(T) energies, gradients, and second derivatives that has been outlined in this paper. Starting from the highly efficient serial implementation of the ACES II MAB computer code an adaptation for affordable workstation

Table 8. Structure Parameters of the Eclipsed Conformation of Ferrocene[a]

method	no. of basis functions	Fe−C_5	Fe−C	C−C	C−H	<C_5−H	ref
fc-CCSD(T)/TZ2P+f[b]	373	1.655	2.056	1.433	1.077	1.03	71
CCSD(T)/cc-pVTZ	508	1.639	2.039	1.426	1.075	0.45	this work
CCSD(T)/cc-pwCVTZ	672	1.648	2.047	1.427	1.079	0.52	this work

[a] Bond lengths are given in Å; angles are given in deg. [b] fc (frozen core) denotes that only the 66 valence electrons were correlated.

Table 9. Structure Parameters of the Staggered Conformation of Ferrocene[a]

method	no. of basis functions	Fe−C_5	Fe−C	C−C	C−H	<C_5−H	ref
fc-CCSD(T)/TZ2P+f[b]	373	1.659	2.058	1.432	1.077	1.34	71
CCSD(T)/cc-pVTZ	508	1.642	2.041	1.425	1.075	0.67	this work
CCSD(T)/cc-pwCVTZ	672	1.652	2.050	1.426	1.078	0.61	this work

[a] Bond lengths are given in Å; angles are given in degrees. [b] fc (frozen core) denotes that only the 66 valence electrons were correlated.

clusters has been obtained by parallelizing the most time-consuming steps of the algorithm.

This also includes the calculation of parallel CCSD energies and gradients using unrestricted (UHF) and restricted open-shell (ROHF) Hartree−Fock references, parallel UHF-CCSD(T) energies and gradients, parallel ROHF-CCSD(T) energies as well as parallel equation-of-motion CCSD energies and gradients for closed- and open-shell references.

The central aspect of the implementation presented here is the replication of the cluster amplitudes and the distributed evaluation, storage, and access of the two-electron integrals to arrive at an algorithm for which sufficient local memory and disk space are necessary but which is not dependent on sophisticated high-speed network connections.

Benchmark calculations for systems with up to 1300 basis functions show that the resulting algorithm for energies, gradients, and second derivatives at the CCSD and CCSD(T) level of theory exhibits good scaling with the number of processors as long as the terms that are the time-determining steps in the serial calculation still dominate the overall time in the parallel computation. It is important to note that the communication steps within the algorithm are at no point bottlenecks in the current implementation, even if 16 or more processors are used. Nevertheless, at larger numbers of nodes the algorithm will break down, as steps in the CCSD algorithm that have not been parallelized prevent a better scaling of the overall execution time, especially for small systems and large number of nodes. The current limitation of the parallel implementation becomes obvious for more than 16 processors. However, an analysis of the algorithm leads us to the conclusion that the scaling behavior is much better for larger examples, where the time-determining steps that have been parallelized dominate the overall execution time more strongly.

If a very rough estimate is allowed at this point—implementations of this kind would open the field of application for the CC hierarchy of high accuracy ab initio methods to systems of about 30 atoms in a triple-ζ basis or about 15 atoms in a quadruple-ζ basis. Typical applications would be calculations of the type presented in the last sections of this paper like high accuracy calculations for structures and energies, vibrational frequencies, or properties related to the NMR spectroscopy of molecules with importance for homogeneous catalysis, model systems for biochemistry, or state-of-the-art spectroscopy.

Technical Details. All calculations were carried out on a 16 node single core 3.4 GHz Intel Xeon (EM64T) cluster with 2 MByte L2 Cache and 16 GB DDR-333 RAM on each node. For the network communication a channel bonded Gigabit Ethernet was used. Channel bonding was set up using the two already built-in network interfaces of the compute nodes by using the standard Linux kernel drivers. This resulted in about 50% more network throughput in comparison to one single Gigabit Ethernet connection per node. For the parallel implementation the message passing interface (MPI)[23] is used. The results presented here are obtained by using LAM/MPI.[74,75] All communication in our implementation is done by the MPI_ALLREDUCE subroutine.

Acknowledgment. We thank Dr. Jonas Jusélius (University of Tromsø, Norway) for help with various computational aspects of this work and Professor Kenneth Ruud (University of Tromsø, Norway) for hospitality and helpful discussions. Work in Chemnitz and in Mainz has been supported by the Deutsche Forschungsgemeinschaft (AU 206/1-1 and GA 370/5-1) as well as by the Fonds der Chemischen Industrie.

References

(1) Tajti, A.; Szalay, P. G.; Császár, A. G.; Kállay, M.; Gauss, J.; Valeev, E. F.; Flowers, B. A.; Vázquez, J.; Stanton, J. F. *J. Chem. Phys.* **2004**, *121*, 11599.

(2) Bomble, Y. J.; Vázquez, J.; Kállay, M.; Michauk, C.; Szalay, P. G.; Császár, A. G.; Gauss, J.; Stanton, J. F. *J. Chem. Phys.* **2006**, *125*, 064108.

(3) Boese, A. D.; Oren, M.; Atasoylu, O.; Martin, J. M. L.; Kállay, M.; Gauss, J. *J. Chem. Phys.* **2004**, *120*, 4129.

(4) Karton, A.; Rabinovich, E.; Martin, J. M. L.; Ruscic, B. *J. Chem. Phys.* **2006**, *125*, 144108.

(5) Heckert, M.; Kállay, M.; Tew, D. P.; Klopper, W.; Gauss, J. *J. Chem. Phys.* **2006**, *125*, 044108.

(6) Purvis, G. D.; Bartlett, R. J. *J. Chem. Phys.* **1982**, *76*, 1910.

(7) Raghavachari, K.; Trucks, G. W.; Pople, J. A.; Head-Gordon, M. *Chem. Phys. Lett.* **1989**, *157*, 479.

(8) Kállay, M.; Gauss, J. *J. Chem. Phys.* **2005**, *123*, 214105.

(9) Botschwina, P. *Theor. Chem. Acc.* **2005**, *114*, 350.

(10) Hill, J.; Platts, J. A.; Werner, H.-J. *Phys. Chem. Chem. Phys.* **2006**, *8*, 4072.

(11) Olson, R. M.; Bentz, J. L.; Kendall, R. A.; Schmidt, M. W.; Gordon, M. S. *J. Chem. Theory Comput.* **2007**, *3*, 1312.

(12) Watts, J. D. *Parallel Computing* **2000**, *26*, 857.

(13) Werner, H.-J.; Knowles, P. J.; Lindh, R.; Manby, F. R.; Schütz, M.; Celani, P.; Korona, T.; Rauhut, G.; Amos, R. D.; Bernhardsson, A.; Berning, A.; Cooper, D. L.; Deegan, M. J. O.; Dobbyn, A. J.; Eckert, F.; Hampel, C.; Hetzer, G.; Lloyd, A. W.; McNicholas, S. J.; Meyer, W.; Mura, M. E.; Nicklass, A.; Palmieri, P.; Pitzer, R.; Schumann, U.; Stoll, H.; Stone, A. J.; Tarroni, R.; Thorsteinsson, T. *MOLPRO, version 2006.1 and earlier versions, a package of ab initio programs*; 2006. See http://www.molpro.net (accessed August 2007).

(14) Kállay, M.; Harding, M. E. *Parallel version of the string-based general coupled-cluster program;* 2006. See http://www.mrcc.hu (accessed August 2007).

(15) Köhn, A.; Hättig, C. *J. Chem. Phys.* **2003**, *118*, 7751.

(16) Hirata, S. *J. Phys. Chem. A* **2003**, *107*, 9887.

(17) Aprá, E.; Windus, T. L.; Straatsma, T. P.; Bylaska, E. J.; de Jong, W.; Hirata, S.; Valiev, M.; Hackler, M.; Pollack, L.; Kowalski, K.; Harrison, R.; Dupuis, M.; Smith, D. M. A.; Nieplocha, J.; Tipparaju, V.; Krishnan, M.; Auer, A. A.; Brown, E.; Cisneros, G.; Fann, G.; Fruchtl, H.; Garza, J.; Hirao, K.; Kendall, R.; Nichols, J.; Tsemekhman, K.; Wolinski, K.; Anchell, J.; Bernholdt, D.; Borowski, P.; Clark, T.; Clerc, D.; Dachsel, H.; Deegan, M.; Dyall, K.; Elwood, D.; Glendening, E.; Gutowski, M.; Hess, A.; Jaffe, J.; Johnson, B.; Ju, J.; Kobayashi, R.; Kutteh, R.; Lin, Z.; Littlefield, R.; Long, X.; Meng, B.; Nakajima, T.; Niu, S.; Rosing, M.; Sandrone, G.; Stave, M.; Taylor, H.; Thomas, G.; van Lenthe, J.; Wong, A.; Zhang, Z. *NWChem, A Computational Chemistry Package for Parallel Computers, Version 4.7;* Pacific Northwest National Laboratory: Richland, WA 99352−0999, U.S.A., 2005.

(18) Schmidt, M.; Baldridge, K.; Boatz, J.; Elbert, S.; Gordon, M.; Jensen, J.; Koseki, S.; Matsunaga, N.; Su, K. N. S.; Windus, T.; Dupuis, M.; Montgomery, J. *J. Comput. Chem.* **1993**, *14*, 1347.

(19) *PQS version 3.2;* Parallel Quantum Solutions: 2005. See http://www.pqs-chem.com.

(20) Janowski, T.; Ford, A. R.; Pulay, P. *J. Chem. Theory Comput.* **2007**, *3*, 1368.

(21) Olson, R. M.; Schmidt, M. W.; Gordon, M. S.; Rendell, A. P. Enabling the Efficient Use of SMP Clusters: The GAMESS/DDI Approach. In Supercomputing, 2003 ACM/IEEE Conference, Phoenix, AZ, 2003; p 41.

(22) Ford, A. R.; Janowski, T.; Pulay, P. *J. Comput. Chem.* **2007**, *28*, 1215.

(23) The MPIForum, MPI: a message passing interface. In Proceedings of the 1993 ACM/IEEE conference on Supercomputing, ACM Press: Portland, OR, U.S.A., 1993.

(24) Stanton, J. F.; Gauss, J.; Watts, J. D.; Szalay, P. G.; Bartlett, R. J.; with contributions from: Auer, A. A.; Bernholdt, D.; Christiansen, O.; Harding, M. E.; Heckert, M.; Heun, O.; Huber, C.; Jonsson, D.; Jusélius, J.; Lauderdale, W. J.; Metzroth, T.; Michauk, C.; Price, D. R.; Ruud, K.; Schiffmann, F.; Tajti, A.; Varner, M. E.; Vázquez, J. including the following integral packages: MOLECULE (J. Almlöf and P. R. Taylor), PROPS (P. R. Taylor), and ABACUS (T. Helgaker, H. J. Aa. Jensen, P. Jørgensen, and J. Olsen). *Aces II Mainz-Austin-Budapest version*; 2007. See http://www.aces2.de (accessed August 2007).

(25) Baumgartner, G.; Auer, A. A.; Bernholdt, D. E.; Bibireata, A.; Choppella, V.; Cociorva, D.; Gao, X.; Harrison, R.; Hirata, S.; Krishanmoorthy, S.; Krishnan, S.; Lam, C.-C.; Nooijen, M.; Pitzer, R.; Ramanujam, J.; Sadayappan, P.; Sibiryakov, A. *Proc. IEEE* **2005**, *93*, 276.

(26) In statements of scaling behavior "*occ*" stands for number of occupied orbitals, and "*vrt*" stands for the number of virtual orbitals. For methods from the coupled cluster hierarchy only the scaling for the time-determining step is given as an estimate for the overall scaling, assuming that the number of virtual orbitals is larger than the number of occupied orbitals.

(27) It should be noted that the size required for storage of the two-electron integrals is of the order of more than 500 GB for a comparable example.

(28) Watts, J. D.; Gauss, J.; Bartlett, R. J. *Chem. Phys. Lett.* **1992**, *200*, 1.

(29) Gauss, J.; Stanton, J. F. *Chem. Phys. Lett.* **1997**, *276*, 70.

(30) Szalay, P. G.; Gauss, J.; Stanton, J. F. *Theor. Chem. Acc.* **1998**, *100*, 5.

(31) Gauss, J.; Stanton, J. F. *J. Chem. Phys.* **1996**, *104*, 2574.

(32) Gauss, J.; Stanton, J. F.; Bartlett, R. J. *J. Chem. Phys.* **1991**, *95*, 2623.

(33) Dunning, T. H. *J. Chem. Phys.* **1989**, *90*, 1007.

(34) Woon, D. E.; Dunning, T. H. *J. Chem. Phys.* **1994**, *100*, 2975.

(35) Woon, D. E.; Dunning, T. H. *J. Chem. Phys.* **1995**, *103*, 4572.

(36) Rendell, A. P.; Guest, M. F.; Kendall, R. A. *J. Comput. Chem.* **1993**, *14*, 1429.

(37) Auer, A. A.; Baumgartner, G.; Bernholdt, D. E.; Bibireata, A.; Choppella, V.; Cociorva, D.; Gao, X.; Harrison, R.; Krishanmoorthy, S.; Krishnan, S.; Lu, Q.; Lam, C.-C.; Nooijen, M.; Pitzer, R.; Ramanujam, J.; Sadayappan, P.; Sibiryakov, A. *Mol. Phys.* **2006**, *104*, 211.

(38) Watts, J. D.; Gauss, J.; Bartlett, R. J. *J. Chem. Phys.* **1993**, *98*, 8718.

(39) Rendell, A. P.; Lee, T. J. *J. Chem. Phys.* **1991**, *94*, 6219.

(40) The quantity $\tilde{t}_{ijk}^{abc}$ arises as a disconnected term in the fifth-order energy correction of the perturbative triples correction.

(41) All calculations for the benzene molecule were carried out at the all electron CCSD(T)/cc-pVQZ geometry (r_{CH}=1.0800 Å, r_{CC}=1.3911 Å), which was taken from ref 76.

(42) Due to machine load and other influences the timings can be assumed to be accurate to a few seconds walltime.

(43) In practice the factorization of the CC equations leads to additional terms that are included in this contraction.

(44) Meyer, W. *J. Chem. Phys.* **1975**, *64*, 1975.

(45) Ahlrichs, R.; Zirz, C. Proceedings of the workshop "Molecular Physics and Quantum Chemistry", Wollongong, 1980.

(46) Ahlrichs, R.; Zirz, C. *Theor. Chim. Acta* **1976**, *36*, 275.

(47) Pople, J. A.; Binkley, J. S.; Seeger, R. *Int. J. Quantum Chem. Symp.* **1976**, *10*, 1.

(48) Hampel, C.; Peterson, K.; Werner, H.-J. *Chem. Phys. Lett.* **1992**, *192*, 1.

(49) Koch, H.; Christiansen, O.; Kobayashi, R.; Jørgensen, P.; Helgaker, T. *Chem. Phys. Lett.* **1994**, *228*, 233.

(50) Gauss, J.; Stanton, J. F. *J. Chem. Phys.* **1995**, *103*, 3561.

(51) Price, D.; Szalay, P. G.; Harding, M. E.; Vázquez, J.; Stanton, J. F. to be published, 2006.

(52) Fossgård, E.; Ruud, K. *J. Comput. Chem.* **2006**, *27*, 326.

(53) Auer, A. A.; Gauss, J.; Stanton, J. F. *J. Chem. Phys.* **2003**, *118*, 10407.

(54) Dumont, E.; Chaquin, P. *Chem. Phys. Lett.* **2007**, *435*, 354.

(55) Cheng, J.; Zhu, W.; Tang, Y.; Xu, Y.; Li, Z.; Chen, K.; Jiang, H. *Chem. Phys. Lett.* **2006**, *422*, 455.

(56) Heine, T.; Corminboeuf, C.; Grossmann, G.; Haeberlen, U. *Angew. Chem., Int. Ed.* **2006**, *45*, 7292.

(57) Jameson, A. K.; Jameson, C. J. *Chem. Phys. Lett.* **1987**, *134*, 461.

(58) Sundholm, D.; Gauss, J.; Schäfer, A. *J. Chem. Phys.* **1996**, *105*, 11051.

(59) London, F. *J. Phys. Radium* **1937**, *8*, 397.

(60) Wolinski, K.; Hinton, J. F.; Pulay, P. *J. Am. Chem. Soc.* **1990**, *112*, 8251.

(61) Ditchfield, R. *Mol. Phys.* **1974**, *27*, 789.

(62) Schäfer, A.; Horn, H.; Ahlrichs, R. *J. Chem. Phys.* **1992**, *97*, 2571.

(63) Schäfer, A.; Huber, C.; Ahlrichs, R. *J. Chem. Phys.* **1994**, *100*, 5829.

(64) Gauss, J. *J. Chem. Phys.* **1993**, *99*, 3629.

(65) Auer, A. A. Ph.D. Thesis, Universität Mainz, Mainz, Germany, 2002.

(66) Lüthi, H. P.; Ammenter, J. H.; Almlöf, J.; Fægri, K. *J. Chem. Phys.* **1982**, *77*, 2002.

(67) Klopper, W.; Lüthi, H. P. *Chem. Phys. Lett.* **1996**, *262*, 546.

(68) Koch, H.; Jørgensen, P.; Helgaker, T. *J. Chem. Phys.* **1996**, *104*, 9528.

(69) Lüthi, H. P. *J. Mol. Struct.* (*THEOCHEM*) **1996**, *388*, 299.

(70) Xu, Z.-F.; Xie, Y.; Feng, W.-L.; Schaefer, H. F., III *J. Phys. Chem. A* **2003**, 107, 2716.

(71) Coriani, S.; Haaland, A.; Helgaker, T.; Jørgensen, P. *Chem. Phys. Chem.* **2006**, *7*, 245.

(72) Peterson, K. A.; Dunning, T. H. *J. Chem. Phys.* **2002**, *117*, 10548.

(73) Balabanov, N. B.; Peterson, K. A. *J. Chem. Phys.* **2005**, *123*, 064107.

(74) Burns, G.; Daoud, R.; Vaigl, J. LAM: An Open Cluster Environment for MPI. Proceedings of Supercomputing Symposium, 1994; pp 379−386.

(75) Squyres, J. M.; Lumsdaine, A. A Component Architecture for LAM/MPI. Proceedings, 10th European PVM/MPI Users' Group Meeting, Venice, Italy, 2003; pp 379−387.

(76) Gauss, J.; Stanton, J. F. *J. Phys. Chem. A* **2000**, *104*, 2865.

CT700152C