Vectorizable Algorithm for Green Function and Many-Body Perturbation Methods

Vyacheslav G. Zakrzewski* **and Wolfgang von Niessen†**

Institut für Physikalische und Theoretische Chemie, Technische Universität Braunschweig, Hans-Sommer Str. 10, D-3300 Braunschweig, Germany

Received 22 January 1992; accepted 27 June 1992

The principles of an efficient, fast algorithm for the calculation of diagrams appearing in Green function and many-body perturbation methods are discussed and timing examples are given. Within the suggested algorithm, the third order-diagrams required in the Green function approach are evaluated by arranging computations in such a way that the most inner loops contain only simple scalar products and multiplication of vector by scalar operations. The molecular symmetry is taken into account for abelian groups. © 1993 by John Wiley & Sons, Inc.

INTRODUCTION

During the last two decades, the method of the one-particle Green function was developed and applied to the calculation of ionization energies of molecular systems.[1-4] In refs. 1–3, several approximations were suggested and in ref. 4 the numerical implementations were discussed. A perturbation approach of the so-called outer valence Green function approximation (OVGF) appears to be the simplest one in its structure among these approximations. The OVGF method remains valid only for those ionization processes where the quasiparticle picture of ionization is applicable.[1-4] In actual calculations, this method requires less computer resources as compared with the more sophisticated third-order *ADC*(3) (algebraic diagrammatic construction) method. Therefore, it may be applied to much larger molecular systems and provides reliable results if the one-particle picture of ionization is valid. There was a certain necessity to reconsider the OVGF numerical algorithm using the advantages of large core and disk storage that are available at the present time and keeping in mind the possibility of taking advantage of vector facilities. In this article, we report the results of our investigations concerning this problem.

Similar problems arise during the evaluation of diagram contributions in the calculation of the molecular total energy within the many-body perturbation methods.[5-17] The first program implementations of these methods[9-11] were written for IBM 360 series computers and were suitable for execution on scalar computers. The main problem was how to arrange the diagram evaluation when the majority of the molecular integrals is kept on disk. It was suggested[9-12] to form lists of integrals that represent partial lists of integrals where one or two indices are fixed and the remaining three or two are running. This allows us to read to the core only part of the two-electron integrals at a time. For the calculation of the fourth-order corrections to the total energy, it was suggested to split the calculation into several steps[12,13] when the appropriate intermediate lists are introduced. A slightly different approach was used in the implementation of the so-called post-SCF routines in the Gaussian series of programs.[5-8,18] For convenience in updating the lists of two-electron integrals, a number of routines was developed that allow us to perform different matrix-type operations over the lists of integrals, for example, different interchanges of indices in the four-index arrays, operations similar to matrix transposition, and matrix multiplication, while these matrices reside on disk and can be read to core in certain portions, etc.

In the latest implementations,[14,15] the possibility to use vector and parallel processing was considered. The main idea of how to vectorize the evaluation of diagrams is trivial enough. For this purpose, it is necessary to prepare first the appropriate lists of two-electron integrals or their combinations that allow us to split the computation of the specific diagram into steps, each of which represents a vectorizable operation. This was the main strategy of our consideration, but the problem still remains how to implement the vectorizable algorithm and utilize the molecular symmetry, which was not discussed in the articles cited above, although the previous version of OVGF program[4] already utilized the symmetry of abelian groups.

*Permanent address: Computing Center, Irkutsk State University, 664003, K. Marx str.1, Irkutsk, Russia.

†Author to whom all correspondence should be addressed.

Journal of Computational Chemistry, Vol. 14, No. 1, 13–18 (1993)

CCC 0192-8651/93/010013-06

DETAILS OF THE ALGORITHM

Let us consider an N-electron system in a nondegenerate ground state $|\Psi_0^N\rangle$ and the total energy E_0^N. The one-particle Green function is defined with respect to an appropriate basis of one-particle states $|p\rangle$ by

$$G_{pq}(t, t') = -i\langle\Psi_0|\hat{T}c_p(t)c_q^+(t')|\Psi_0\rangle \qquad (1)$$

where $c_p^+(t)[c_p(t)]$ denotes creation (annihilation operators) for a one-particle state $|p\rangle$ in the Heisenberg representation and $\hat{T}$ is Wick's time-ordering operator. The Fourier transform of the Green function $\mathbf{G}(\omega)$ can be expressed by the well-known spectral representation[19]

$$G_{pg}(\omega) = \sum_n \frac{\langle\Psi_0^N|c_p|\Psi_n^{N+1}\rangle\langle\Psi_n^{N+1}|c_q^+|\Psi_0^N\rangle}{(\omega + E_0^N - E_n^{N+1} + i\eta)} + \sum_m \frac{\langle\Psi_0^N|c_q^+|\Psi_m^{N-1}\rangle\langle\Psi_m^{N-1}|c_p|\Psi_0^N\rangle}{(\omega + E_m^{N-1} - E_0^N - i\eta)} \qquad (2)$$

Here, $E_n^{N\pm 1}$ and $|\Psi_n^{N\pm 1}\rangle$ denote the energies and states of the $(N \pm 1)$-particle systems and E_0^N is the ground-state energy. The ionization energies

$$I_n = E_n^{N-1} - E_0^N \qquad (3)$$

and the electron affinities

$$A_n = E_0^N - E_n^{N+1} \qquad (4)$$

are derived as the poles of $\mathbf{G}(\omega)$. The usual procedure to evaluate the Green function is to use a perturbation expansion. The Hamiltonian in the second-quantization form and assuming its usual decomposition can be written as

$$\hat{H} = \hat{H}_0 + \hat{H}_I \qquad (5)$$

$$\hat{H}_0 = \sum \epsilon_i c_i^+ c_i \qquad (6)$$

$$\hat{H}_I = \hat{W} + \hat{V} = \sum W_{ij} c_i^+ c_j + \frac{1}{2}\sum V_{ijkl} c_i^+ c_j^+ c_l c_k \qquad (7)$$

where $\hat{H}_0$ is the diagonal one-particle part and $\hat{H}_I$ is the perturbation that in turn consists of the two-particle (Coulomb) interaction and a nondiagonal one-particle part $\hat{W}$. As usual, the one-particle states are considered to be the Hartree–Fock (HF) ones and then ϵ_i are the HF orbital energies. The two-electron matrix elements V_{ijkl} are defined according to

$$V_{ijkl} = \langle\phi_i(1)\phi_j(2)|V(1, 2)|\phi_k(1)\phi_l(2)\rangle \qquad (8)$$

(which differs from the definition in the Mulliken charge density notation, V_{ijkl} = (ik|jl)). In the HF case, the nondiagonal one-particle part $\hat{W}$ has the form

$$W_{ij} = -\sum_k (V_{ikjk} - V_{ikkj})n_k \qquad (9)$$

Here, n_k denotes the HF occupation number, i.e., $n_k = 1$ if orbital k is occupied and $n_k = 0$ if orbital k is unoccupied. The one-particle Green function $\mathbf{G}$ is related to the so-called self-energy $\mathbf{\Sigma}$ by the Dyson equation, which in the ω representation reads

$$\mathbf{G}(\omega) = \mathbf{G}^0(\omega) + \mathbf{G}^0(\omega)\mathbf{\Sigma}(\omega)\mathbf{G}(\omega) \qquad (10)$$

The "free" Green function with the Hamiltonian above has the form

$$G_{pq}^0 = \delta_{pq}\left[\frac{n_p}{\omega - \epsilon_p - i\eta} + \frac{1 - n_p}{\omega - \epsilon_p + i\eta}\right] \qquad (11)$$

Then, the simplest approximation to calculate ionization energies of molecules can be written as

$$I_p = (\epsilon_p + \Sigma_{pp}(I_p)) \qquad (12)$$

where p is the MO index for which the calculation is performed (in what follows, we will be dealing with the set $\{p\}$ of such MOs), ϵ_p is the ordinary orbital energy, and Σ_{pp} are the diagonal elements of the self-energy part, which assuming the third-order expansion can be written as follows:

$$\Sigma_{pp}(\omega) = \sum_{i=1}^{6} (A_i + C_i(\omega) + D_i(\omega)) + \Sigma_{pp}^{(2)} \qquad (13)$$

The explicit formulae of this expansion are listed in refs. 2–4, and therefore we do not list them here but will make use of the notation from these articles. The computational problem in the case of the OVGF method is to a certain extent more complicated than the problem of evaluation of the third-order contributions to the total energy. First, in the Green function method in the third order of perturbation theory more diagrams arise, and their expressions after turning to the spin-free formulation are to a certain extent more complicated as compared with the contributions of the diagrams for the total energy. Second, eq. (12) should be solved numerically by an iteration procedure putting as the zero approximation $I_p = \epsilon_p$, and these iterations are performed for each specific MO p from the set $\{p\}$. It was found[4] that usually only a few iterations are necessary if the OVGF method is valid but nevertheless certain care should be taken to remove the repeated calculations from the iteration loop.

All the terms in eq. (13) may be divided into the following parts: the constant third-order terms A, which are not ω-dependent and therefore their contribution is to be calculated only once for each orbital p and may be added during the iterations. Some principles of the algorithm for the evaluation of the A-diagrams were discussed in ref. 4 and our last program implementation may be considered as following the ideas presented in refs. 12 and 18. Then, there is the second-order $\Sigma^{(2)}$ term, which is ω-dependent but its calculation is trivial and may be vectorized easily. The main part of the problem is connected with the evaluation of the C, D terms, which contain the ω-dependent factors in the denominator, and therefore we will discuss their evaluation in some detail. First, there are some equalities between diagonal C and D terms:

$$C_2 = C_3, C_4 = C_5, D_2 = D_3, D_4 = D_5 \qquad (14)$$

and these terms contain only one ω-dependent factor in the denominator. But, in the C_1, C_6 and D_1, D_6 terms both factors in the denominator are ω-dependent. Therefore, let us consider first these more complicated terms. The expression for $C_{1pp}(\omega)$ is as follows:

$$C_{1pp}(\omega) = \sum_{abcd} \sum_{i} \times \frac{(2V_{piab} - V_{piba})V_{abcd}V_{picd}}{(\omega + \epsilon_i - \epsilon_a - \epsilon_b)(\omega + \epsilon_i - \epsilon_c - \epsilon_d)} \qquad (15)$$

where indices $i, j, \ldots$ run over doubly occupied orbitals, a, b, $\ldots$ over virtual ones, and p, q, $\ldots$ over both types of orbitals. The following arrays are prepared and stored:

$$XC1(a, b, c, d) = V_{abcd} \qquad (16)$$

and for each p from the set $\{p\}$ the following arrays:

$$YC1(a, b, i) = 2V_{piab} - V_{piba} \qquad (17)$$

$$ZC1(i, a, b) = V_{piab} \qquad (18)$$

$$EC1(a, b, i) = \epsilon_i - \epsilon_a - \epsilon_b \qquad (19)$$

$$FC1(i, a, b) = \epsilon_i - \epsilon_a - \epsilon_b \qquad (20)$$

Here, the indices run over the full range without taking into account index permutational symmetry. We consider the index in the right most position to be running first. Then, the evaluation of the $C1_{pp}(\omega)$ contribution for each specific ω can be performed by the following steps:

$$DE1(a, b, i) = \omega + EC1(a, b, i) \qquad (21)$$

$$YC1(a, b, i) = \frac{YC1(a, b, i)}{DE1(a, b, i)} \qquad (22)$$

$$GE1(i, a, b) = \omega + FC1(i, a, b) \qquad (23)$$

$$ZC1(i, a, b) = \frac{ZC1(i, a, b)}{GE1(i, a, b)} \qquad (24)$$

$$TC1(i, a, b) = \sum_{cd} XC1(c, d, a, b)YC1(c, d, i) \qquad (25)$$

$$C1 = \sum_{iab} TC1(i, a, b)ZC1(i, a, b) \qquad (26)$$

Let us consider next the $D1_{pp}$ diagram contribution:

$$D_{1pp} = \sum_{abc} \sum_{ij} \times \left\{ \frac{V_{piab}[V_{ajic}(V_{pjcb} - 2V_{pjbc}) + V_{ajci}(V_{pjbc} - 2V_{pjcb})]}{(\omega + \epsilon_i - \epsilon_a - \epsilon_b)(\omega + \epsilon_j - \epsilon_b - \epsilon_c)} + \frac{V_{piba}[V_{ajic}(4V_{pjbc} - 2V_{pjcb}) + V_{ajci}(V_{pjcb} - 2V_{pjbc})]}{(\omega + \epsilon_i - \epsilon_a - \epsilon_b)(\omega + \epsilon_j - \epsilon_b - \epsilon_c)} \right\} \qquad (27)$$

In this case, again the following arrays should be prepared and stored:

$$Z1D1(i, a, j, c) = 2V_{ajic} - V_{ajci} \qquad (28)$$

$$X2D1(i, a, j, c) = V_{ajic} \qquad (29)$$

$$X3D1(i, a, j, c) = V_{ajci} \qquad (30)$$

$$D1D1(i, a, b) = V_{piab} \qquad (31)$$

$$D2D1(b, j, c) = 2V_{pjbc} - V_{pjcb} \qquad (32)$$

$$D3D1(b, j, c) = 2V_{pjcb} - V_{pjbc} \qquad (33)$$

$$E2D1(b, j, c) = \epsilon_j - \epsilon_b - \epsilon_c \qquad (34)$$

$$E1D1(i, a, b) = \epsilon_i - \epsilon_a - \epsilon_b \qquad (35)$$

Then, the evaluation of the $D_{1pp}(\omega)$ contribution is performed in the following steps:

$$F1D1(i, a, b) = \omega + E1D1(i, a, b) \qquad (36)$$

$$F_2D_1(b, j, c) = \omega + E2D1(b, j, c) \qquad (37)$$

$$G2D1(b, j, c) = \frac{D2D1(b, j, c)}{F2D1(b, j, c)} \qquad (38)$$

$$G3D1(b, j, c) = \frac{D3D1(b, j, c)}{F2D1(b, j, c)} \qquad (39)$$

$$Y1D1(i, a, b) = \sum_{jc} [X1D1(i, a, j, c)G2D1(b, j, c) - X2D1(i, a, j, c)G2D1(b, j, c) - X3D1(i, a, j, c)G3D1(b, j, c)] \qquad (40)$$

$$G1D1(i, a, b) = \frac{D1D1(i, a, b)}{F1D1(i, a, b)} \qquad (41)$$

$$D1 = \sum_{iab} Y1D1(i, a, b)G1D1(i, a, b) \qquad (42)$$

From this representation it is easy to notice that at the last stage there is only the necessity to perform simple vector manipulations. [At first glance, the arrays given by eqs. (29) and (30) are only the necessary ones but in this case the combination represented by eq. (28) would have to be evaluated directly inside the iteration loop. Therefore, to save a certain amount of computations for the evaluation of contributions of diagrams of this type we preferred to rearrange the original expressions and then the array given by eq. (28) appears.] The evaluation of the C_2, C_4, D_2, and D_4 diagram contributions can be performed in a similar way and is much simpler because in this case combinations of the type

$$X(i, a, j, b) = \frac{V_{abij}}{(\epsilon_i + \epsilon_j - \epsilon_a + \epsilon_b)} \qquad (43)$$

may be precomputed and stored. Then, for the C_2 and C_4 diagrams only the steps analogous to those represented by eqs. (21), (22), and (26) and for the D_2 and D_4 by steps (36), (41), and (42) should be performed inside the iteration loop. All other calculations may be done only once for each orbital p. All the arrays involved in the algorithm are listed in

Table I. Arrays independent of the specific MO index from the set $\{p\}$, their notation, index sequence, and identification of the diagrams for which they are used.

Number	Array	Index sequence	Diagram	Note
1	$(2V_{abij} - V_{abji})/D(ijab)$	i, a, j, b	$A1, 3, 5, D2, 4$	
2	$V_{abij}/D(ijab)$	i, a, j, b	$A1, D2, D4$	
3	$V_{abji}/D(ijab)$	i, a, j, b	$D2, D4$	
4	$2V_{abij}/D(ijab)$	a, b, i, j	$A2$	*
5	V_{kaij}	k, a, i, j	$A5$	*
6	V_{ciab}	c, a, i, b	$A3$	*
7	$V_{abij}/D(ijab)$	a, b, i, j	$A2$	*
8	$V_{ijab}/D(ijab)$	i, j, a, b	$C2, C4$	
9	V_{abcd}	a, b, c, d	$C1$	
10	V_{ijkl}	i, j, k, l	$C6$	
11	$2V_{ajic} - V_{ajci}$	i, a, j, c	$D1, D6$	
12	V_{ajic}	i, a, j, c	$D1, D6$	
13	V_{ajci}	i, a, j, c	$D1, D6$	

Arrays marked by asterisk are used for the constant A diagrams evaluation only and this space may be used later for other arrays.

Tables I and II. The total amount of data to be stored is several times larger than the number of the transformed two-electron integrals because the latter are stored using the advantages of index permutational symmetry.

So far, we have not considered molecular point group symmetry, which leads to appreciable savings in quantum chemical calculations. In the problem under consideration, there also remains the possibility to use the abelian group symmetry. For these purposes, it was necessary to introduce the appropriate addressing scheme between the elements of all the arrays involved. Let us consider that all the MOs are ordered according to their irreducible representations and then within each representation in increasing order of their orbital energies. It follows from eqs. (25), (26), (40), and (42) that in all the summations to be performed one or two indices are fixed while three or two are running. Each element of the arrays we use remains nonzero by symmetry when the direct product of all four representations to which the appropriate MOs belong (in the case of "three-dimensional arrays," the first index is the p-index of the MO for which the calculation is performed) gives the totally symmetric one. Let us consider, for example, an array $X(a, b, c, d)$. Let the first index a be fixed and let the corresponding MO belong to the representation R_a. Then, the length of the list of elements $X(a, [b, c, d])$ when the indices b, c, and d are running over the whole of their range is determined from symmetry considerations. Moreover, this length remains the same for each index a from the set $\{a\}$ that belongs to the same representation R_a. Thus, it is necessary to introduce one integer array of length 8 that contains entry points to the lists of elements $X(a, [b, c, d])$ for the first a-index in each of the representations. The other integer array again of length 8 should contain the lengths of the lists $X(a, [b, c, d])$ for each representation. The same holds when the first pair of indices a and b is fixed while the remaining are running over the whole range. In the latter case, two two-dimensional 8 × 8 arrays should be introduced: one containing entry points to the list of $X(a, b, [c, d])$, where a and b are the first indices in representations R_a and R_b, and another containing corresponding lengths of the lists. Addressing arrays of such a type are used for all the arrays listed in Tables I and II. Some of the latter ones are of similar structure and therefore addressing arrays are identical.

Table II. Arrays that depend upon the specific MO p from the set $\{p\}$, their index sequence, and the diagrams for which they are used.

Number	Array	Index sequence	Diagram
1	$\epsilon_a - \epsilon_i - \epsilon_j$	i, j, a	all
2	$\epsilon_i - \epsilon_a - \epsilon_b$	i, a, b	all
3	$2V_{paij} - V_{paji}$	i, j, a	$B1, C4$
4	V_{paij}	i, j, a	$B1, C6, D4, 6$
5	$2V_{piab} - V_{piba}$	i, a, b	$B2, C2$
6	V_{piab}	i, a, b	$B2, C1, D1, 2$
7	$2V_{pijk} - V_{pikj}$	j, k, i	$C2$
8	$2V_{pabc} - V_{pacb}$	a, b, c	$C4$
9	$2V_{pjbc} - V_{pjcb}$	b, j, c	$D1$
10	$2V_{pjcb} - V_{pjbc}$	b, j, c	$D1$
11	$2V_{pcbj} - V_{pcjb}$	b, j, c	$D2$
12	$2V_{pcjb} - V_{pcbj}$	b, j, c	$D2$
13	$2V_{pjkb} - V_{pjbk}$	k, j, b	$D4$
14	$2V_{pjbk} - V_{pjkb}$	k, j, b	$D4$
15	$2V_{pbkj} - V_{pbjk}$	k, j, b	$D6$
16	$2V_{pbjk} - V_{pbkj}$	k, j, b	$D6$
17	$2V_{paij} - V_{paji}$	a, i, j	$C6$
18	$2V_{piab} - V_{piba}$	a, b, i	$C1$
19	$\epsilon_j - \epsilon_b - \epsilon_c$	b, j, c	$D1$
20	$\epsilon_b - \epsilon_j - \epsilon_k$	k, j, b	$D6$
21	$\epsilon_i - \epsilon_a - \epsilon_b$	a, b, i	$C1$
22	$\epsilon_a - \epsilon_i - \epsilon_j$	a, i, j	$C6$

The diagram notation is explained in the text; B_1 and B_2 denote the two second-order terms (see ref. 4).

The structure of our program is as follows. First, from all the arrays listed in Table I only those needed to calculate the constant diagrams are created, and the A-diagram contributions are evaluated and stored. Then, those arrays that are used for constant diagram evaluation only are deleted from the direct access file and all other four-index arrays from Table I are created. The iterational solution of Eq. (12) is performed in such a way that all the IPs from the same irreducible representation are calculated in one batch. This allows us to prepare, for example, arrays from Table II, which represent the orbital energy differences, only once for each irreducible representation. Then, for each orbital p for which the calculation should be made all other arrays from Table II are being created and the iteration of eq. (12) with the evaluation of all ω-dependent diagrams is performed. It should be pointed out that within the Green function formalism there exists another significant quantity, called the pole strength, and to determine these quantities it is necessary to calculate the derivatives of the ω-dependent diagrams. From the above discussion concerning evaluation of corresponding diagrams, it is clear enough that within the present approach these extra calculations cause no problems and do not need a lot of additional computer time. The main part of this algorithm was developed and coded by one of us (V.Z.) at the Irkutsk State University Computing Centre for the IBM-compatible computer EC-1061 (Russia) without any vector facilities. But, it turned out that this program code appeared to be suitable for vectorization also.

RESULTS AND DISCUSSION

To determine the real performance of our new OVGF program, we calculated the ionization energies (IPs) of the benzene and nitrogen molecules. In both cases, the experimental geometry was used. The calculation of the benzene molecule was done in D_{2h} and C_{2v} symmetry. The appropriate SCF calculations and the two-electron integral transformation were performed using the MOLCAS system of programs.[20] In the case of the benzene molecule, the basis set $[9s5p1d]/(4s2p1d)_C$ $[4s1p]/(2s1p)_H$ taken from the work of Huzinaga[21] was used and in the case of the nitrogen molecule the ANO basis set[22] $(14s9p4d2f)/[7s6p4d2f]$ was used. For the benzene calculation, only the MOs corresponding to the 1s AOs and their virtual counterparts were omitted. The 15 occupied and 99 unoccupied orbitals were taken into account in these calculations and 7 IPs were calculated. For the nitrogen molecule, except for the core 1s orbitals six upper virtual orbitals with the orbital energies exceeding 20 au were omitted and 5 occupied and 125 virtual orbitals were taken into account in the Green function calculation. The calculations using both OVGF programs were performed on an IBM 3090/600J computer under the VM CMS operating system. In Table III, the corresponding times are given with the appropriate times for the same calculations using the previous version of the OVGF program.[4] It can be seen that the new version of the OVGF program is many times faster and comparatively large time (up to 25%) is spent in the vector mode. For calculation of the benzene molecule, 117 and 222.3 Mb of scratch space area is necessary to store all the intermediate arrays that arise when using D_{2h} and C_{2v} symmetry, respectively. From Table III, it can be seen that when lower symmetry is used and when more IPs belonging to the same irreducible representation are calculated the relative time spent in vector mode increases and in turn the new algorithm appears to be more advantageous as compared with the old scalar one. This feature seems to be of great importance for calculations of organic molecules where usually the symmetry is not very high. We also performed the calculation of the benzene molecule in C_{2v} symmetry with the same program compiled without the vector option and obtained a virtual CPU time of 1015 s (instead of 536 s in Table III), thus getting an acceleration factor of 1.9. This factor for the case of the simple scalar product for one processor of the IBM 3090/600J is 2.4–2.5 (with the peak performance for scalar products of 24 Mfp) and therefore our result can be considered satisfactory. The performance may be increased by using instead of scalar products matrix multiplications but this in turn can only be done easily when molecular symmetry is not taken into account.

The previous version of the OVGF program was designed to be executed on the computers of the IBM 360 series and the possibility to use vector features was not taken into account. When this program of the OVGF method was designed, it was not possible to store large intermediate arrays because usually scratch disk space of the appropriate size was not available.

To compare the corresponding expenses for the calculation of ionization energies by the Green function approach with the more traditional one when the ionization energies are calculated as the total energy differences within the CI method, we performed the CI calculations of the benzene molecule

Table III. Timings (in seconds) of the OVGF programs for the benzene and nitrogen molecules on an IBM3090/600J computer.

		New OVGF			Old OVGF
Molecule	Group	Virtual	Total	Vector	total
C_6H_6	D_{2h}	244	288	61	6320
C_6H_6	C_{2v}	536	627	194	19059
N_2	D_{2h}	254	314	37	1617

Details of the calculations are given in the text. For the new OVGF program, the "virtual CPU," "total CPU" (the latter includes processor service of the I/O activities), and "vector CPU" times are given.

in the C_{2v} symmetry. With the MOLCAS1[20] program for the SDCI (single and double excitation CI with a single reference configuration), calculation of the ground state takes approximately 11 min and of the ionic state $^2E_{1g}$ (in the C_{2v} symmetry, this state splits into B_2 and A_2; therefore, actually in the C_{2v} symmetry the $^2B_2$ state was calculated) approximately 26 min. A separate CI calculation should be performed for each ionic state and therefore the total amount of computer time would be larger than with the present algorithm within the Green function approach. Even when comparing the time needed to perform the CI calculation with the time for the Green function calculation using the previous version of the OVGF method, we should take into account that the CI program from the system MOLCAS is already a vectorized one. The vectorization for the computer used in the present calculations provides an acceleration factor of 3–4 and therefore on the scalar computers the CI calculation of the same number of IPs would take more time than the calculation with the previous version of the OVGF method.

CONCLUSIONS

We have presented an algorithm for the efficient evaluation of diagrams in Green function methods. The algorithm involves the decomposition of the diagram evaluation procedure into the step of preparing intermediate arrays and the step of the evaluation of the corresponding contributions of the third-order diagrams using intermediate arrays of integrals and their combinations. When solving the Dyson equation, this allows us to perform within the iteration loop only vector operations of a scalar product and a multiplication of a vector by a scalar. The molecular symmetry of abelian groups is included directly and this provides considerable savings, especially of the scratch disk space needed to store intermediate arrays. The most time-consuming operations of division by energy denominators are mainly done outside the iteration loop where the Dyson equation is being solved. Similar approaches may be used for the matrix element evaluation that are necessary in more sophisticated Green function approximations as ADC(3) or ADC(4) or approximations based upon any kind of many-body perturbation theory.

The Alexander von Humboldt Foundation fellowship for V.Z. is gratefully acknowledged. Part of this work was supported by the Fonds der Chemischen Industrie.

References

1. L.S. Cederbaum, *J. Phys. B*, **8,** 290 (1975).
2. L.S. Cederbaum and W. Domcke, in *Advances in Chemical Physics*, vol. 36, I. Prigogine and S.A. Rice, Eds., Wiley, New York, 1977, p. 205.
3. J. Schirmer, L.S. Cederbaum, and O. Walter, *Phys. Rev. A*, **28,** 1237 (1983).
4. W. von Niessen, J. Schirmer, and L.S. Cederbaum, *Comp. Phys. Rep.*, **1,** 57 (1984).
5. J.A. Pople, J.S. Binkley, and R. Seeger, *Int. J. Quantum Chem.*, **S10,** 1 (1976).
6. J.A. Pople, R. Seeger, and R. Krishnan, *Int. J. Quantum Chem.*, **S11,** 149 (1977).
7. R. Krishnan and J.A. Pople, *Int. J. Quantum Chem.*, **14,** 91 (1978).
8. J.A. Pople, R. Krishnan, H.B. Schlegel, and J.S. Binkley, *Int. J. Quantum Chem.*, **S13,** 225 (1979).
9. D.M. Silver, *Comp. Phys. Comm.*, **14,** 71 (1978).
10. D.M. Silver, *Comp. Phys. Comm.*, **14,** 81 (1978).
11. S. Wilson, *Comp. Phys. Comm.*, **14,** 91 (1978).
12. S. Wilson, *Comp. Phys. Rep.*, **2,** 389 (1985).
13. J. Noga, *Comp. Phys. Comm.*, **29,** 117 (1983).
14. D. Moncrief, D.J. Baker, and S. Wilson, *Comp. Phys. Comm.*, **55,** 31 (1989).
15. D.J. Baker, D. Moncrief, V.R. Saunders, and S. Wilson, *Comp. Phys. Comm.*, **62,** 25 (1991).
16. S. Wilson, *Electron Correlation in Molecules*, Clarendon, Oxford, UK, 1984; Russian translation, Mir, Moscow, 1987.
17. J.S. Binkley, in *Advanced Theories and Computational Approaches to the Electronic Structure of Molecules. NATO ASI Series, Series C: Mathematical and Physical Sciences*, vol. 133, C. Dykstra, Ed., D. Reidel, Dortrecht, p. 209.
18. J.S. Binkley, M. Frish, K. Raghavachari, D. De Frees, H.B. Schlegel, R. Whiteside, E. Fluder, R. Seeger, and J.A. Pople, Gaussian 82, Release A, Carnegie-Mellon University, Pittsburgh, PA.
19. A.B. Migdal, *Theory of Finite Fermi Systems*, Wiley-Interscience, New York, 1967.
20. G. Karlstöm, P.-Å. Malmqvist, B.O. Roos, A.J. Sadlej, and P.-O. Widmark, MOLCAS-1, University of Lund, Lund, Sweden, 1990.
21. S. Huzinaga, *J. Chem. Phys.*, **42,** 1293 (1965).
22. P.-O. Widmark, P.-Å Malmqvist, and B.O. Roos, *Theor. Chim. Acta*, **77,** 291 (1990).