Theor Chim Acta (1987) 71: 187-199

Theoretica Chimica Acta

A multireference direct CI program based on the symmetric group graphical approach

Włodzisław Duch* and Jacek Karwowski**

Max-Planck-Institut für Physik und Astrophysik, Institut für Astrophysik, Karl-Schwarzschild-Straße 1, D-8046 Garching bei München, Federal Republic of Germany

(Received January 9, revised and accepted September 29, 1986)

Specific features of an SGGA-based multireference direct CI program working in large internal spaces are discussed. In particular, advantages resulting from the explicit separation of the orbital and the spin spaces are explored. Concepts allowing for the efficient creation of a flexible and symmetry-adapted CI basis, for the high-speed generation of the coupling coefficients and for structuring a simple permutation driven algorithm to handle the orbital space are briefly discussed.

Key words: Configuration interaction method — Symmetric group graphical approach — Electron correlation problem — Multi-reference CI program

1. Introduction

The progress in designing multireference direct CI (MR DCI) algorithms is certainly one of the most spectacular events in the field of computational chemistry. Since the idea of direct CI has been formulated by Roos in 1972 [1] the length of the CI expansion, which still may be handled in a reasonable way, has grown up by two orders of magnitude crossing 10^6 several years ago [2]. The key to the success lies in the global approach to the problem in which symmetry properties and structural specificity of the N-electron space rather than explicit expressions for the basis functions or for the matrix elements determine the

* Alexander von Humboldt Fellow 1985/87

** *Permanent address*: Instytut Fizyki Uniwersytetu Mikołaja Kopernika, PL-87 100 Toruń, Poland

computational strategy. Significant progress in understanding the internal structure of the CI space, in particular due to extensive use of the group theory, resulted in two alternative approaches known as the graphical unitary group approach (GUGA) [3, 4] and the symmetric group graphical approach (SGGA) [5]. Both approaches take full advantage of all spin symmetries but, due to their sophistication, lead to algorithms which are rather complex and difficult to vectorize in their logical parts. On the other hand, development in the field of supercomputers stimulated work aimed at creating methods which are very simple and easy to vectorize at the cost of an increased amount of simple algebraic operations. To this class belongs the determinant-based approach of Handy [6] and the vectorizable full CI method of Siegbahn [7].

Though it is hard to predict whether the future belongs to the simplistic or to the sophisticated formulations, at present the most advanced and most general DCI programs are based on GUGA [8–11]. In the GUGA programs the external space, which usually provides a vast majority of the basis functions, constitutes a simple and vectorizable part of the algorithms (see, e.g., [9]). Efficiency in handling large internal spaces still needs improvement. In particular reducing (possibly eliminating) the formula tape and extending the flexibility of the CI basis would be most welcome modifications. It is natural to expect that SGGA, being both simpler and more general than its unitary-group-based counterpart, may offer some interesting solutions. This option has never been seriously explored.

It is the aim of this paper to present some preliminary data on a general DCI SGGA code. The program is still under construction, therefore we present here several examples of solutions to partial problems being specific for SGGA, i.e., resulting from the explicit separation of the orbital and the spin spaces. In particular we discuss the construction of the CI basis, the evaluation of the coupling constant matrices as well as their assigning to the pairs of interacting configurations and to the primitive integrals. We skip subjects being common to all DCI formulations, such as the four-index integral transformation and the construction of the eigenvectors. Also the question of dealing with the external space and, in particular, an important problem of the optimum strategy [9] are left aside, since our main objective is improving efficiency in handling large internal spaces.

A complete and self-contained description of the method is presented in [5]. In order to avoid unnecessary and lengthy explanations concerning notation and basic concepts we shall use the same notation as the one in [5] and we shall refer to this paper as I. Reports on several methods of evaluation the coupling constants, used in this program, have been published separately [12, 13]. A reader interested in a more detailed account is referred to these papers. The program is being developed as a part of the system MUNICH [14] in the Garching group of molecular astrophysics, on two computers, namely a Siemens 7880 and a CRAY XMP.

2. Construction of CI basis

The CI basis consists of antisymmetric and spin-adapted configuration state functions (CSFs) $|\lambda: SM, 1\rangle$ (Eq. (I.48)). All $f(S, s)$ CSFs corresponding to a single orbital configuration λ form a configuration state vector (CSV). The basis set of configurations is described by the orbital graph. Every path in the graph is supplied with its lexical index m_λ (I.97) and represents an orbital configuration (λ). The mapping between the lexical indices of the paths and the elements of the CI vector is facilitated by introducing the index vector $I(m_\lambda)$ (I.98). All these concepts are common to both GUGA and SGGA, except that in SGGA the index vector is shorter since it corresponds to the set of the orbital configurations only and the graph is simpler since it describes the pure orbital functions (i.e. the Hartree products of orbitals) only.

The procedure aimed at defining the CI basis consists in (1) determining the shape of the graph with certain vertices and/or arcs removed whenever this is needed, (2) Removing specific configurations, viz. those which do not fulfill the symmetry or the excitation limit requirements.

The input data include the reference configurations, their excitation limits and, optionally, identifications of the removed vertices and arcs. From these data the orbital graph is constructed and represented using weight vectors as it is described in Sect. II.1.1. of I. Both space and time spent on this introductory step is entirely negligible. Even in cases of very large internal bases (10^6–10^7) the time does not exceed 0.2 s.

The highest efficiency of the program is achieved when the configuration basis is defined entirely by the geometry of the graph. Therefore the development of insights regarding relations between the relevance of a configuration and its location in the graph is particularly important. For this reason the geometrical representation of the graph is printed out upon request. Several examples are shown in Figs. 1–4. The full CI graph for the water molecule in a double-zeta basis (cf. [7] contains 270270 CSVs (Fig. 1) corresponding (for $S = 0$) to 100 2001 CSFs. The basis is reduced by a modification of the graph to 24 031 CSVs (Fig. 2) if only configurations being singly-, doubly-, triply- and quadruply-excited relatively to a single reference are retained. The Abelian symmetry may be introduced by designating different "symmetry versions" to vertices (as it was done in GUGA [15]). However, when all orbitals (except those belonging to the totally symmetric representation) appear in the graph in an equivalent way, as is the case for a full CI, then symmetry-adapted graphs may be constructed by removing certain vertices. An example is shown in Fig. 3. Another example (for the case of full-CI $^1A_1$ state of water molecule in a basis of 11 orbitals) is given in Fig. 4. The reason for this construction is easily understood if we remember that the symmetry of a configuration containing m electrons in orbitals b_1, m' electrons in orbitals b_2 and $N-m-m'$ electrons in orbitals a_1 is A_1 if both m and m' are even, A_2 if both m and m' are odd, B_1 if m is odd and m' is even and B_2 if m is even and m' is odd.

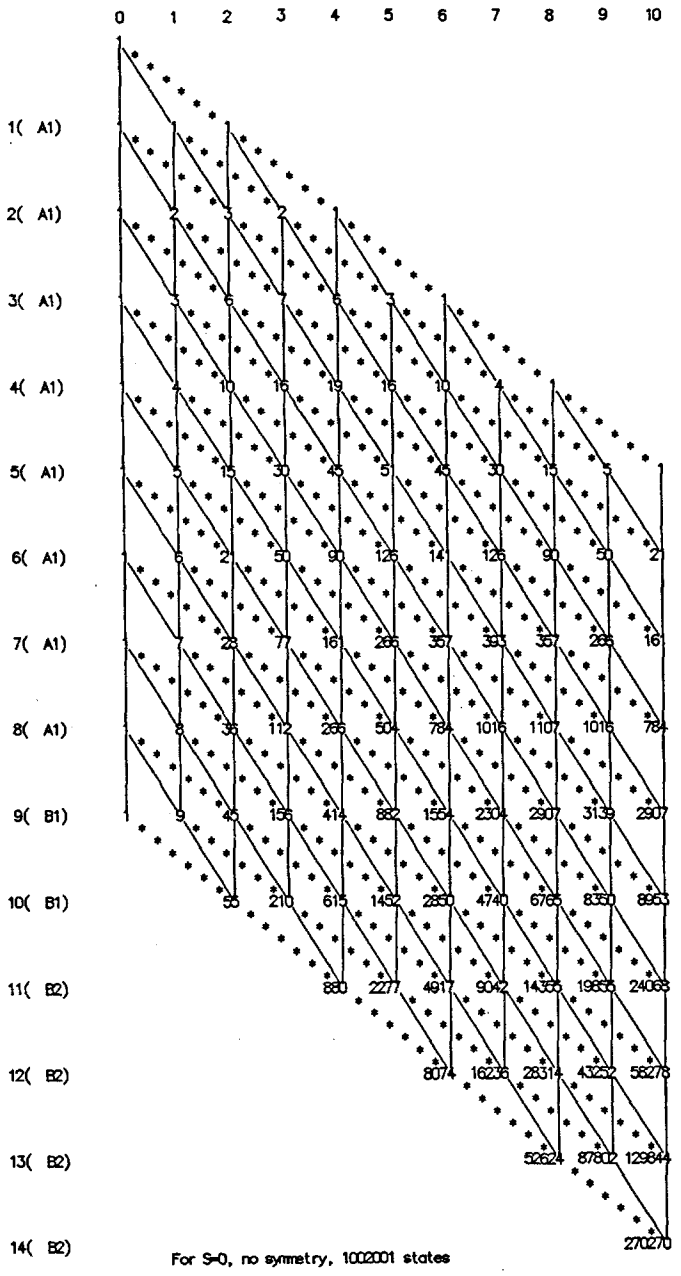

Fig. 1. The full CI graph of water molecule in a double-zeta basis (10 electrons, 14 orbitals). The orbital indices are supplied with the symmetry labels. The CI basis consists of 270 270 CSVs, what corresponds to 1 002 001 singlet CSFs

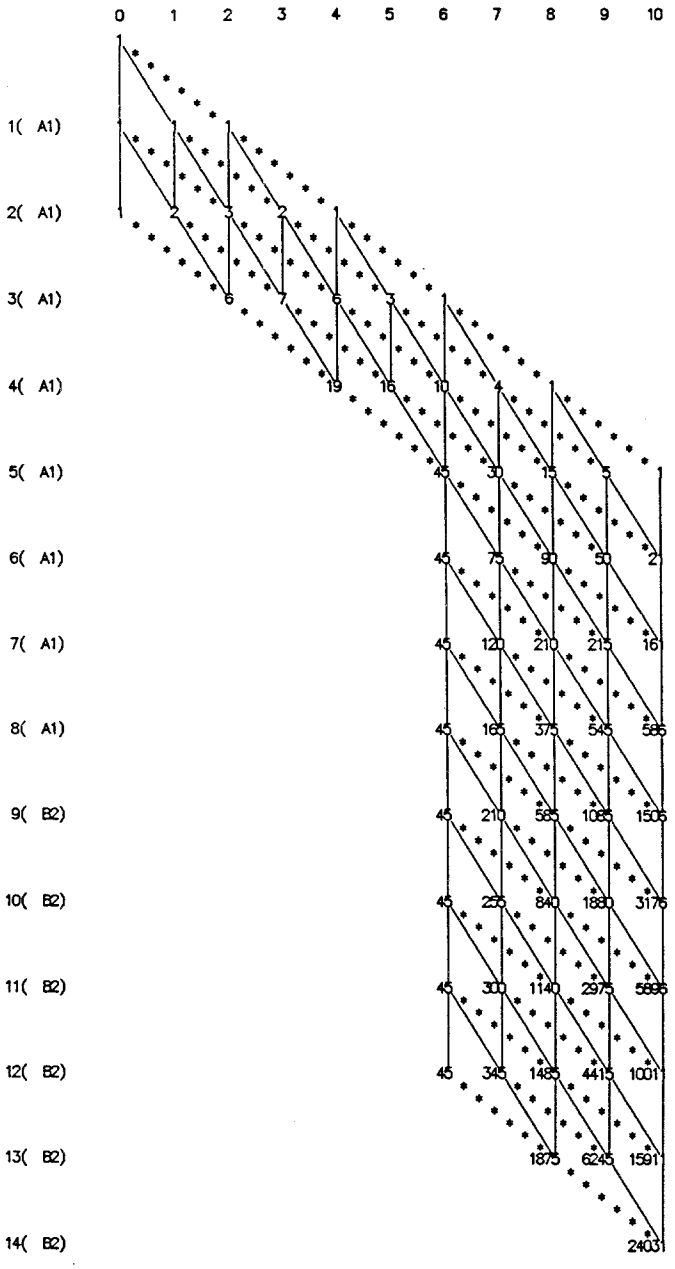

Fig. 2. The same as in Fig. 1 , but only configurations being at most quadruply excited relatively to the configuration for which the first 5 orbitals are doubly occupied are retained

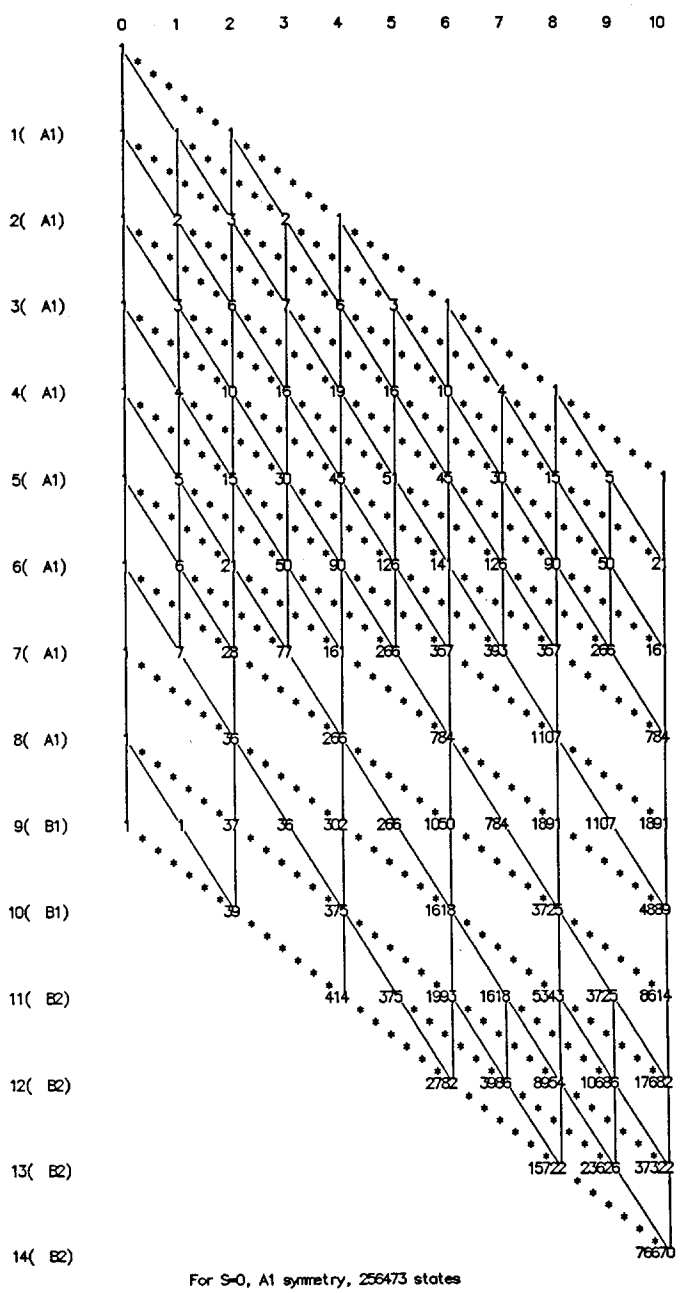

Fig. 3. The full CI graph for water molecule in a double-zeta basis containing only A_1 configurations; 76 670 CSVs corresponds to 256 473 $^1A_1$ CSFs

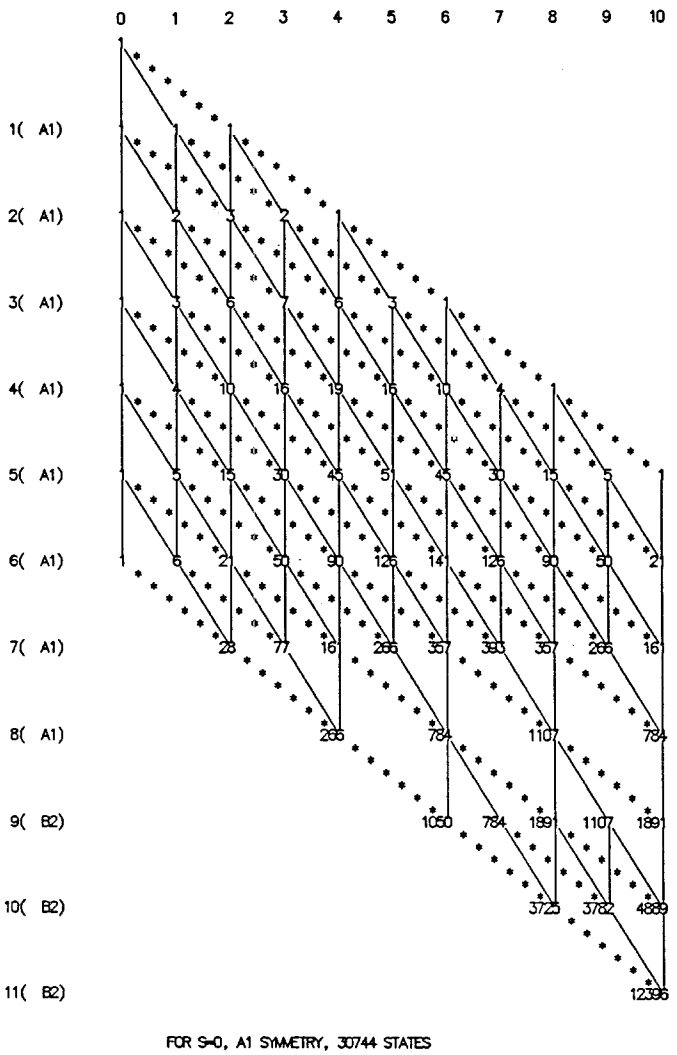

Fig. 4. The full CI graph for A_1 configurations of water molecule in a basis of 11 orbitals

A further reduction of the CI basis by removing specific configuration is facilitated by means of the index vector. At this step we run over the entire graph checking fulfillment of the additional restrictions imposed upon the configurations. The following options have been included: the maximum number of singles, excitation multiplicity (which may be different for different reference configurations), Abelian point-group symmetry and the first-order interaction space restrictions [16] for CSFs (cf. I.5.3. in I). Besides, a set of configurations to be removed may be specified explicitly. The dimension of the index vector is equal to the number of paths in the configuration graph, i.e. it is much smaller than the length of the CI basis (in the examples in Figs. 1–4 by factors of 3.7, 2.5, 4.1 and 2.5, respectively).

Table 1. CUP times (in seconds) on Siemens 7880 for creation of the index vector. The CI bases are the ones defined in Fig. 4 ($n = 11$), in Fig. 3 ($n = 14$) and in Fig. 1. The last basis has been reduced to $^1A_1$ by removing a part of configurations during construction of the index vector ($n = 14^*$).

CI basis	$n = 11$	$n = 14$	$n = 14^*$
No. of CSVs accepted	12 396	76 670	76 670
No of CSVs removed	0	0	193 600
Generation of the graph[a]	0.12	0.12	0.14
Index vector: modified depth-search algorithm	0.20	1.09	3.59
Index vector: broken depth-search algorithm	0.04	0.24	0.83

[a] Includes reading and analysing input data as well as constructing and printing the geometrical representation of the graph

The running-over-the-graph algorithm belongs to the most frequently used ones - not only at the step determining the index vector but, more importantly, in packing up pairs of interacting configurations. Therefore, we have tested several kinds of the algorithms:

1. *Breadth search.* The algorithm is described in detail in Sect. II.1.2 of I. It is a fast but space-consuming method.

2. *Depth search.* The algorithm is based on the classical idea of the tree search. It was successfully applied in GUGA-based methods (see, e.g., [8]). The method needs almost no space but it is slightly slower than the breadth search one.

3. The depth search algorithm modified by taking an advantage of the simple structure of the right border area of the graph. It has all advantages of the conventional depth search algorithm and is always faster than the breadth search one.

4. *Broken depth search.* The depth search algorithm is here executed to find all contributions from the head of the graph to a chosen level and from the tail to the same level. The results are then combined together. This is the otpimum algorithm. For large graphs the execution is here approximately equal to the square root of that for the conventional depth search algorithm and the space requirements are only slightly enlarged compared with the classical tree-search method.

Typical timings for construction of the index vector are collected in Table 1.

3. Matrix elements

Due to the separation of the orbital space from the spin space, the evaluation of matrix elements is performed in two steps. In the first step the orbital graph is analyzed. In the second step the spin branching diagram is used to generate the coupling coefficients. Subsequently, the coefficients are multiplied by the

appropriate integrals to form the matrix elements. Though in some GUGA-based programs, as e.g. in the one developed by Saunders and van Lenthe [9], the orbital and the spin parts are also treated independently, this separation is intrinsic to the SGGA approach in the most natural way.

The algorithm for evaluation the coupling coefficients has been described elsewhere [5, 12, 13]. The one-electron coupling coefficients are generated in core with a speed comparable with reading the formula list from a tape, requiring less than one multiplication per an element of $\mathbf{U}(P)$. For example [13], in the case of 10 open shells coupled to singlets, the dimension of $\mathbf{U}(P)$ is 42 and there are 45 different cycles $(i \cdots j)$ with $j-i=1,2,\ldots,9$. The total number of $\mathbf{U}(P)$ matrix elements corresponding to all these cycles is equal to $42\times 42\times 45 = 79\,380$. Their calculation takes 0.2 s on the Siemens 7880 and less than 0.03 s on the CRAY 1. This gives an average speed of more than 2.6 million elements per second on the CRAY 1. Evaluation of the one-electron coupling coefficients, which includes also matrix elements $2^{1/2}\mathbf{U}((i \cdots j))|_k$ is performed with an average speed of about 400 thousand per second which compares with the speed of about 650 per second when reading a tape. This timing may be further improved by an optimization of the program. We therefore avoid storing the coupling coefficients on an external file, even in cases of very large internal spaces. It is worthwhile to note that the efficiency of the algorithm grows with the number of open shells [13]. A detailed discussion of the method, including a discussion of the algorithms for evaluation of the two-electron coupling coefficients, is given in [12] and [13].

The analysis of the orbital graph is aimed at finding all pairs of interacting configurations for a given set of orbitals being involved in the integrals under consideration and at identifying the permutations which determine the coupling constants. The three types of matrix elements - namely (i) those between the same configurations, forming the diagonal part of the CI matrix; (ii) those between configurations differing by one orbital, corresponding to the two-segment loops in the orbital graph; and (iii) those between configurations differing by two orbitals, corresponding to the three- and four-segment loops in the orbital graph - are expressed by different kinds of formulas and therefore are treated using three different algorithms.

The diagonal part is calculated using the running-over-the-graph algorithm, as it is described in the previous section. Since the method contains nothing particularly new compared with other DCI codes, we do not discuss here any details.

The two-segment loop contributions comprise the pairs of configurations differing by one orbital. In a single matrix element expression there appear integrals which depend upon all orbitals. The coupling coefficients depend upon distribution of singles between the corresponding levels in the graph (cf. Eq. (I.182)). Therefore all the matrix elements are different and in the process of their evaluation occupation numbers of all orbitals in both configurations involved must be known. Further, information about the entire paths representing the pair of configurations must be available in each case. The algorithm for the matrix element evaluation may be divided into two parts. The first one, referred to as the "logical part",

finds pairs of the interacting CSVs, the permutations P_x (Eq. (I.182)), and calculates the values of $(i|G|j)$ (Eq. (I.183)). The logical part is essentially based on a properly modified routine for running over the graph and takes less than 10% of the entire time spent for the two-segment loop calculations. In the second part the contributions $\mathbf{A}_{\lambda\mu}(P)(ip|jp)$ are calculated and summed together.

The two-segment loop procedure is illustrated in Table 2, where some statistical data and CPU times are given for the CI bases presented in Figs. 4 ($n = 11$) and 3 ($n = 14$). The quantities displayed include: the number of CSVs (K_v), the length of the CI expansion (K_s), the number of blocks of matrix elements, i.e. the number of interacting CSVs (M_B), the number of matrix elements (M_E) - each of the M_B blocks contributes $f(S, s_\lambda)f(S, s_\mu)$ elements - and CPU times on the Siemens 7880 for execution of the first and the second part of the algorithm (t_1 and t_2 respectively). In general, the bigger the ratio K_s/K_v, i.e. the bigger the proportion of the configurations with many open shells, the higher is the efficiency. The ratio M_E/M_B tells us how many matrix elements, on the average, are determined by a single pair of CSVs. As we can see from the table, the logical part may handle 120–140 thousand of blocks in a second on the Siemens 7880. The speed of the matrix element evaluation in the second part of the algorithm is slightly greater, viz. 150–210 thousand elements per second. Of course, both parts of the algorithm are executed during the same graph searching process. It is interesting to note that the execution time grows almost (a little less than) linearily with the number of CSFs (cf the last entry of Table 2).

Table 2. Evaluation of matrix elements resulting from the two-segment loops in the orbital graph. The CI bases are the ones presented in Fig. 4 ($n = 11$) and in Fig. 3 ($n = 14$). The quantities displayed: the number of configuration state vectors (K_V), the length of the CI expansion (K_S), the number of blocks of matrix elements (M_B), the number of matrix elements (M_E), CPU times (in seconds) on Siemens 7880 for creation the matrix elements (t_1 = the logical part and t_2 = the computational part)

	$n = 11$	$n = 14$
K_V	12 396	76 670
K_S	30 744	256 473
K_S/K_V	2.5	3.3
M_B	179 251	1 149 436
M_E	2 194 486	31 564 958
M_E/M_B	12.2	27.5
t_1^a	1.4	8.2
t_2	14.4	149.4
M_B/t_1	128 036	140 175
M_E/t_2	151 344	211 278
$K_S/(t_1+t_2)$	1 946	1 623

[a] Includes evaluation of $(i|G|j)$ integrals (cf Eqs. (I.182) and (I.183))

In the case of three- and four-segment loops (configurations differing by two orbitals) a matrix element depends upon three/four orbital indices only. The information being derived from the graph and associated with the path segments other than that corresponding to the three/four orbitals mentioned above, includes the number of singles between the head of the graph and the levels defined by the three/four orbitals and the lexical contributions from the parallel segments of the paths only. For a given set of orbital indices, matrix elements associated with a given distribution of singles are the same. It is desirable to construct the algorithm in such a way that the coupling coefficients are calculated for the different blocks only. We denote by M_B^* the number of different blocks and by M_E^* the number of the matrix elements derived from different blocks only. The ratio M_E^*/M_B^* gives us the average size of the block being calculated by this kind of algorithm. It may be used as an estimate of what can be gained by an explicit separation of the spin space.

Here too, as was the case for the two-segment loops, the algorithm may be divided into a "logical" and a "computational" part. The logical part of the DCI procedure has here the following structure:

(1) DO-loop over the orbital indices;
(2) DO-loops over all distributions of singles in the path segments contained between the levels in the graph determined by the indices defined in (1);
(3) Consideration of all chains of loops in order to find the permutations defining the coupling coefficients (Eq. 4.4.b in I);
(4) Determination of the lexical indices of the pairs of configurations which are coupled (in terms of the results of (3)) and belong to the CI basis (consulting the index vector);
(5) For each *set* of configurations being coupled by the same coupling constants determine the constants.

Consequently the coupling coefficients are determined after the removed configurations are eliminated by the index vector. In addition, the algorithm is in fact permutation driven. For a given integral (1) we first determine the distribution of singles (2) and, considering all chains of loops (3), the set of permutations associated with the distribution of singles. The permutations are determined *once* for all those pairs of configurations which are compatible with the data defined in steps (1) and (2). In the case of the four-segment loops there are 8 different chains (consisting of 6 loops each). Only 3 permutations, i.e. only 3 matrices of the coupling coefficients, are associated with a chain (cf. I, Tables 8 and 9). Some statistical data reflecting the efficiency of this scheme are collected in Table 3. In the cases considered, the ratio M_B/M_B^* for three- and four-segment loops ranges from 221 to 8. Then, we calculate the coupling coefficients for only 0.5–13% of the interacting pairs of CSVs. The CPU time for executing the logical part of the algorithm in the case of the four-segment loops, may be expressed as $t = aK_V^b$. In the case of full CI and $n = N$, for the Siemens 7880, we have $b = 1.2$ and, if $a = 2.5$ and K_V is in thousands, then t is in seconds. Similarly, for $N = 6$ and $n \geq 6$, we have $b = 1.5$ and $a = 1.8$. This timing, though quite satisfactory, is expected to be considerably reduced after further optimization of the program.

Table 3. Structure of the orbital part of the internal space in the case of three- and four-segment loops. The CI bases are the ones presented in Fig. 4 ($n=11$) and in Fig. 2 ($n=14$, SDTQ). The quantities displayed: The number of configuration state vectors (K_V), the length of the CI expansion (K_S), the number of blocks of matrix elements (M_B), the number of different blocks (M_B^*), the number of matrix elements (M_E), the number of matrix elements derived from the different blocks only (M_E^*)

	$n=11$	$n=14$, SDTQ[a]
K_V	12 396	7 071
K_S	30 744	17 678
K_S/K_V	2.5	2.5
Three-segment loops:		
M_B	354 204	125 992
M_B/M_B^*	220.7	31.7
M_E	2 821 784	679 466
M_E/M_E^*	35.7	14.8
M_E^*/M_E^*	49.3	11.5
Four-segment loops:		
M_B	1 241 184	464 256
M_B/M_B^*	34.2	7.8
M_E	16 637 376	5 440 434
M_E/M_E^*	9.5	4.8
M_E^*/M_B^*	48.2	19.0

[a] Only $^1A_1$ configurations

4. Summary

It is interesting to see how far the configuration interaction method - the classical approach to stationary problems of quantum mechanics - can be developed. The introduction of graphical representation of model (CI) spaces in molecular [3, 5] and recently also in atomic and nuclear physics [17], giving deep insight into the structure of CSFs space, has opened the way to a dramatic increase in the efficiency of computations. So far in GUGA-based programs only the simple structure of singly and doubly excited CSFs (external space) is used and the programs are inefficient for more than double excitations. Results of full CI or nearly full CI calculations are interesting not only as benchmarks: for example, in nuclear physics they are of fundamental importance. It is our goal to achieve similar efficiency for general CI expansions as is achieved by GUGA programs with separation of external spaces. Preliminary data on a general direct CI program based on SGGA are given in this paper. The most promising approach, but logically very complicated if all internal space relations are used, is based on the broken depth search algorithm. Although this approach was used only to a very limited extent timings obtained in the test cases suggest the possibility of a formula tape elimination.

Acknowledgements. The authors are very grateful to G. H. F. Diercksen for his kind hospitality at the Max-Planck-Institut für Astrophysik in Garching and for many useful discussions, comments and advice. The financial support from the Max-Planck Foundation and from the Alexander-von-Humboldt Stiftung are gratefully acknowledged. A part of this work was also sponsored by the Institute for Low Temperature and Structure Research of the Polish Academy of Sciences.

References

1. Roos BO (1972) Chem Phys Lett 15:153
2. Saxe P, Fox DJ, Schaefer HF, Handy NC (1982) J Chem Phys 77:5584
3. Shavitt I (1977) Int J Quantum Chem S11:131; (1978) S12:5
4. Robb MA, Niazi U (1984) Comp Phys Rep 1:127
5. Duch W, Karwowski J (1985) Comp Phys Rep 2:93
6. Handy NC (1980) Chem Phys Lett 74:280; Knowles PJ, Handy NC (1984) Chem Phys Lett 111:315
7. Siegbahn PEM (1984) Chem Phys Lett 109:417
8. Brooks BR, Schaefer HF (1979) J Chem Phys 70:5092
9. Saunders VR, van Lenthe JH (1983) Mol Phys 48:923
10. Lischka H, Shepard R, Brown FB, Shavitt I (1981) Int J Quantum Chem S15:91
11. Siegbahn PEM (1980) J Chem Phys 72:1647
12. Duch W (1985) Int J Quantum Chem 27:59
13. Duch W (1986) Chem Phys Lett 124:442
14. Diercksen GHF, Kraemer WP, MUNICH Molecular Program System. Reference Manual. Special Technical Report. Max-Planck-Institut für Physik und Astrophysik, Institut für Astrophysik, Garching b. München
15. Shavitt I (1979) Chem Phys Lett 63:421
16. McLean AD, Liu B (1973) J Chem Phys 58:1066
17. Duch W (1986) Graphical representation of model spaces, vol. I. Lect Notes Chem **42**. Springer, Berlin Heidelberg New York