UC-4

LBL-9998

Lawrence Berkeley Laboratory
UNIVERSITY OF CALIFORNIA

Materials & Molecular Research Division

THE LOOP-DRIVEN GRAPHICAL UNITARY GROUP APPROACH TO THE ELECTRON CORRELATION PROBLEM, INCLUDING CONFIGURATION INTERACTION ENERGY GRADIENTS

Bernard Robinson Brooks
(Ph. D. thesis)

September 1979

For Reference

Not to be taken from this room

Prepared for the U.S. Department of Energy under Contract W-7405-ENG-48

The Loop-driven Graphical Unitary Group Approach to the Electron Correlation Problem, Including Configuration Interaction Energy Gradients

By

Bernard Robinson Brooks
Lawrence Berkeley Laboratory, University of California
Berkeley, CA 94720

ABSTRACT

The Graphical Unitary Group Approach (GUGA) has been cast into an extraordinarily powerful form by restructuring the Hamiltonian in terms of loop types. This allows the adoption of the loop-driven formulation which illuminates vast numbers of previously unappreciated relationships between otherwise distinct Hamiltonian matrix elements. The theoretical/methodological contributions made here include the development of the loop-driven formula generation algorithm, a solution of the upper walk problem used to develop a loop breakdown algorithm, the restriction of configuration space employed to the multireference interacting space, and the restructuring of the Hamiltonian in terms of loop types. Several other developments are presented and discussed. Among these developments are the use of new segment coefficients, improvements in the loop-driven algorithm, implicit generation of loops wholly within the external space adapted within the framework of the loop-driven methodology, and comparisons of the diagonalization tape method to the direct method. It is also shown how it is possible to implement the GUGA method without the time-consuming full (m^5) four-index transformation. A particularly promising new direction presented here involves the use of the GUGA methodology to obtain one-electron and two-electron density matrices. Once these

are known, analytical gradients (first derivatives) of the CI potential energy are easily obtained. Several test calculations are examined in detail to illustrate the unique features of the method. Also included is a calculation on the asymmetric $2^1A'$ state of SO_2 with 23,613 configurations to demonstrate methods for the diagonalization of very large matrices on a minicomputer.

To Terri

ACKNOWLEDGEMENTS

Special thanks are due to Fritz Schaefer for sharing with me a portion of his wealth of knowledge, for encouraging me to work independently, and for allowing me to maintain my nocturnal habits. I am extremely grateful to Professor Shi Shavitt for introducing me (and the world) to the Graphical Unitary Group formalism. I also thank Professor Shavitt for many invaluable discussions and communications throughout the development of this method. I am indebted to Professor Nick Handy for convincing me the CI gradients were feasible. I also wish to thank Professor Handy for presenting many of the equations expressed in the gradient section as well as proposing algorithms to solve them.

I am also indebted to Bill Laidig and Paul Saxe whose excellent work and great effort enabled the completion of the second generation of Unitary Group programs and CI gradient programs. I am also hopeful for the future, as the work here is not presented as a conclusion, but only a picture of a developing method that will certainly be carried much further by them and others. I also thank John Goddard, Per Siegbahn, Professor Paldus, Michel Dupuis, Richard Stratt, Ross Wetmore, Sigrid Peyerimhoff, and Yukio Yamaguchi for many invaluable discussions.

I thank Carol Hacker for much help throughout my four years at Berkeley (and even after leaving). I especially thank my wife Terri who put more work into this thesis than I did.

This research was supported by the U.S. National Science Foundation, Grant CHE-7622621, and by the Division of Chemical Sciences, Office of Basic Energy Sciences, U.S. Department of Energy under contract No. W-7405-Eng-48.

TABLE OF CONTENTS

I. INTRODUCTION

The importance of electron correlation in quantum mechanical predictions of chemical interest can hardly be overestimated. It has been abundantly documented that correlation effects can greatly alter theoretical predictions of bond energies,[1] activation energies,[2] and spectroscopic electronic energy differences,[3] to cite three of the most important examples. For this reason, much theoretical and computational effort has been justifiably expended in recent years in the development of new approaches to the correlation problem.

Fortunately, the past decade has witnessed great progress in the theorist's ability to describe electron correlation in a practical but rigorously variational manner. Most of the new methods are either variants of, or in some way equivalent to, the standard method of configuration interaction (CI).[4] Some of the most successful currently available techniques are the pair natural orbital (PNO) CI,[5,6] direct CI methods,[7-9] the method of self-consistent electron pairs (SCEP),[10] and a whole range of sophisticated general CI methods.[11-19] In practical terms, these advances mean that a nearly complete solution of the correlation problem is now feasible for essentially any molecule for which ab initio self-consistent-field (SCF) wave functions could be obtained ten years ago. From another perspective, one can now approach the correlation problem for the benzene molecule (C_6H_6) at a level of theory that was barely possible for methylene (CH_2) a decade ago.[20]

Rather than satisfying the desires of chemists to use increasingly sophisticated quantum mechanical techniques, the above developments have instead whetted appetites for further forays into even more

complicated chemical problems. Thus the need for ever more powerful theoretical methods is likely to be with us for some time to come. One of the most promising ideas advanced in the past several years is the Graphical Unitary Group Approach (GUGA). Actually the unitary group mathematical formalism has been available for more than a decade,[21] and some of it for much longer.[22] However, it was by no means obvious that this formalism had any particular advantage over existing variational many body techniques. The latter situation has been changed by the recent research of Paldus[23] and of Shavitt[24] on the adaptation of the unitary group approach to electronic structure problems. Indeed the present work was inspired by a series of lectures recently given by Shavitt[25] at Livermore.

The formal basis of the unitary group approach is the fact that a particular Gelfand-Zetlin canonical basis spans the space of spin-adapted many-electron functions for a given number of electrons and total spin S.[21] This in itself is not terribly novel since there are many available techniques for the generation of spin eigenfunctions, e.g. the genealogical method,[26] use of projection operators,[27] and diagonalization[28] of $\underset{\sim}{S}^2$. The power of the graphical unitary group approach, as developed by Paldus[23] and by Shavitt,[24,25] lies with the tremendous amount of insight it provides into the structure of the Hamiltonian matrix $\underset{\sim}{H}$. This structure is not only an edifice of mathematical beauty but also a practical tool for the calculus of matrix element evaluation.

Taking the work of Shavitt as our starting point, we present here the first implementation of the graphical[24] unitary group approach

for molecular electronic structure problems. The most critical contribution of our own work is the development of the loop-driven methodology. This method of generating the Hamiltonian contributions illuminates previously undiscovered relationships between otherwise distinct Hamiltonian matrix elements. Other theoretical contributions made here in addition to the development of the loop-driven algorithm are a solution of the upper walk problem, the restriction of configuration space employed to the multireference interacting space[29] suggested by second-order perturbation theory, and the restructuring of the Hamiltonian in terms of loop types. Within the Shavitt framework a number of additional methodological modifications and improvements have been made to harness the formalism into a sequence of functioning algorithms. To demonstrate the power of the unitary group approach, it has been applied to a series of problems, several of which were beyond the capabilities of our previous large scale configuration interaction (CI) techniques.[19,30] In fact for large, general, multireference configuration interaction (CI) problems, our unitary group methods require only a fraction of the computation time necessary for state-of-the-art conventional CI techniques.[19] When one realizes that these conventional methods have been perfected over a period of more than 25 years,[4,31] the magnitude of this achievement becomes apparent. Of course the basic formalism of the unitary group approach (UGA) has been available for more than a decade.[21]

The development of analytic gradient techniques for the investigation of potential energy surfaces has been one of the most important recent developments in electronic structure theory.[32] Much of the

early advancement in this area is largely due to the work of Pulay,[33] who presented an expression for the forces on all of the nuclei for closed-shell self-consistent-field theory. His techniques have been used mainly for prediction of force constants and equilibrium geometries. Other more recent ab initio gradient techniques and methods are due to the further work of Poppinger,[34] Schlesel,[35] Kormornicki,[36] Dupuis and King,[37] Pople,[38] Hehre,[39] and by Goddard, Handy and Schaefer.[40] With these methods, energy derivatives with respect to nuclear position can be obtained for a wide range of electronic wavefunctions, including open-shell restricted Hartree-Fock, unrestricted Hartree-Fock, two configuration MCSCF wavefunctions, and for second order Moller-Plesset perturbation theory. Notably missing from this list are analytic gradient methods for large scale configuration interaction wavefunctions.

We present here the new methods required to obtain analytic gradients for general large scale CI calculations, we well as a description of the first implementation of these methods. The most important contribution of this work is the use of the loop-driven algorithm to generate the two-particle density matrix. Another important contribution is in the area of determining the gradient contribution for perturbations in the CI expansion basis. Pople[41] has recently developed an efficient iterative computational method for solving the coupled-perturbed Hartree-Fock (CPHF) equations given by Gerratt and Mills[42] for restricted Hartree-Fock wavefunctions. We have improved this technique to simultaneously solve the CPHF equations for all 3N nuclear perturbations (where N is the number of nuclei)

with a further reduction of necessary work.

II. REVIEW OF THEORY

There are two rather distinct aspects of the GUGA method, the first being the generation of the distinct row table (DRT), which completely determines the configuration space that is included in the CI. The DRT can be as complex as necessary to describe any predetermined configuration set. The second aspect is the utilization of the DRT as a backbone or template to generate all of the loops which define the Hamiltonian matrix in a compact symbolic form. Each loop defines a set of equivalent matrix element contributions. The loop-driven algorithm is used to generate the loops in a rapid manner.

A. The Paldus Representation

Each Gelfand state[22] (or configuration) can be represented by a three column array of non-negative integers referred to as a Paldus array or tableau that spans the number of molecular orbitals (n) used in the CI,

$$[p] = \begin{bmatrix} a_n & b_n & c_n \\ a_{n-1} & b_{n-1} & c_{n-1} \\ \vdots & & \\ a_1 & b_1 & c_1 \\ 0 & 0 & 0 \end{bmatrix} . \qquad (1)$$

For this representation the top row determines the electronic state, and each other row, where

$$a_i + b_i + c_i = i \qquad (2)$$

represents a particular irreducible representation of the unitary group U(i). There are only four ways (or cases) in which one row (i-1) of any Paldus array can differ from the next row (i), which determines the occupancy of orbital i for that particular corresponding Gelfand state. The four cases are identified by case numbers s_i which can be summarized by:

s_i	Δa_i	Δb_i	Δc_i	n_i	ΔS_i
0	0	0	1	0	0
1	0	1	0	1	1/2
2	1	-1	1	1	-1/2
3	1	0	0	2	0

(3)

Here $\Delta a_i = a_i - a_{i-1}$ and n_i is the number of electrons in orbital i. Case 0 results when orbital i is unoccupied. Cases 1 and 2 result from the different spin couplings (referred to as alpha and beta coupled respectively) of the singly occupied orbital i, and Case 3 results from a doubly occupied orbital i. It is clear from this that

$$a_i = \frac{1}{2} N_i - S_i \tag{4}$$

and

$$b_i = 2S_i \quad , \tag{5}$$

where N_i is the number of electrons in the first i orbitals and S_i is the spin of this subgroup. For a given top row, all possible Paldus arrays that can be generated correspond to the configurations of a full CI treatment. Perhaps a simpler way to represent each Gelfand state is by a series of n case values $(s_1, s_2, \ldots, s_n)$ where n is the number of orbitals. There is a one-to-one correspondence between these case arrays and complete, orthonormal, spin-adapted Gelfand states.

B. The Shavitt Representation

In the graphical representation developed by Shavitt[24,25] primarily for illustrative purposes, each different case value corresponds to a different slope line segment (or arc) that spans one orbital. A vertical arc represents an unoccupied orbital (s_i=0). A sharply slanted arc represents a doubly occupied orbital (s_i=3) and the two singlet couplings are identified by differing intermediate slopes. When a series of these arcs are chained together as described by the case array s_i, one forms a path (or walk) that uniquely defines that Gelfand state. One of the most remarkable features of these paths is that when any two such paths of a given spin state are superimposed so that their tails (bottom) coincide, their heads (top) must also coincide. Their paths may also coincide at intermediate points. This suggests superimposing all of the Gelfand states desired for a given CI calculation. When this is done one obtains a Shavitt graph (or master graph as referred to previously). The numerical analog of a Shavitt graph is referred to as a distinct row table (DRT). A sample Shavitt graph is depicted in Figure 1. The corresponding DRT for this case is shown in Table I. The Shavitt graph[24,25] is identified by a series of vertices (distinct rows) and arcs. The rows of any level i correspond to the possible rows of the Paldus arrays of that Gelfand basis at that same level i. The index j refers to a given distinct row within i, such that

$$i = a_{ij} + b_{ij} + c_{ij} \quad . \tag{6}$$

Each level i is connected to the next lower level i-1 by the occupancy

of orbital i. Each row of level i can relate to up to four rows of level i-1 by the four different cases (s_i, s_i=0,1,2,3). Any such valid relation corresponds to an arc. Each arc can be represented by the three indices (i,j,s) which will uniquely determine its position within the DRT. Each vertex (at level i) identifies a particular component of Gelfand states containing this vertex and is determined by the number of electrons and total spin of these electrons in the first i orbitals. The Shavitt graph is also two-rooted in that single points determine its head or tail. The graph tail corresponds to the origin and the graph head uniquely determines the number of electrons and the spin state. Each possible walk represented by a series of n arcs from the graph head to the graph tail corresponds to a particular configuration. In general the number of different arcs and distinct rows is much smaller than the number of configurations. There is a one-to-one correspondence between these walks and the Gelfand states of the Gelfand-Zetlin basis. The selective elimination of certain arcs is used to generate a subset of the configurations of the full CI, such as all singly and doubly excited configurations from a given reference.

There is also a lexical ordering of states such that m' < m if and only if $s_i' < s_i$ at the highest level, where $s_i \neq s_i$ for those states. Here m refers to the state index and s_i refers to the case values (s_i=0,1,2,3). For each distinct row there are a number of x_r of lower walks (possible paths to reach the graph tail) and also a number of $\bar{x}_r$ of upper walks. The lexical ordering of states provides that for any upper walk from a given distinct row all states corresponding

to the different lower walks are sequential. Each given row of the Shavitt graph or DRT is also given an arc weight (y) for each valid arc. The index m for any state can be expressed as a sum of the arc weights over all the levels. Table I gives a sample DRT with all of the mentioned features.

The numerical analog of the Shavitt graph is the DRT (distinct row table) which in its simplest form is made up of different distinct rows in the Paldus representation.

C. Matrix Elements of the Unitary Group Generators

Matrix elements between two Gelfand states can be expressed as

$$\langle m'|H|m\rangle = \sum_{i,j} \langle i|\hat{h}|j\rangle \langle m'|E_{ij}|m\rangle + \tfrac{1}{2} \sum_{i,j,k,\ell} [ij;k\ell]\langle m'|E_{ij}E_{k\ell}-\delta_{jk}E_{i\ell}|m\rangle \quad . \tag{7}$$

The spin independent generator E_{ij} is given by

$$E_{ij} = \sum_{\sigma} X^{\dagger}_{i\sigma} X_{j\sigma} \quad , \tag{8}$$

where $X^{\dagger}_{i\sigma}$ and $X_{i\sigma}$ are creation and annihilation operators for an electron in orbital i and spin state σ. The generator E_{ij} when operating on a Gelfand state, removes an electron from orbital j and adds one to orbital i. Other properties of these generators are:

$$\langle m'|E_{ii}|m\rangle = \delta_{m'm}\, n_i \quad , \tag{9}$$

where E_{ii} is termed a weight generator, and

$$\langle m'|E_{ij}|m\rangle = \langle m|E_{ji}|m'\rangle = 0 \quad , \quad (i<j,\ m' \geqslant m) \tag{10}$$

due to the lexical ordering of states.

Direct formulas for the evaluation of matrix elements of any generator have been derived by Gouyet et al.[43] and by Drake and Schlesinger,[44] and these lead to a factorization of generators and generator products into segment contributions. Thus each different segment shape has certain coefficients associated with it that are determined by generator factorization. This factorization is one of

the essential features that distinguish the GUGA method from other unitary group CI methods.

The matrix elements of the generators are determined by a factorization (of the matrix element) over the levels between and including the indices of the generator or generator product. The one-electron generator matrix elements can be given by

$$\langle m'|E_{pq}|m\rangle = \prod_{i=p}^{q} W(T_i, s_i', s_i, \Delta b_i, b_i) \qquad (11)$$

where the different W segment values are simply predetermined coefficients for each possible segment shape. Whenever two configurations in the graphical representation m' and m have a nonzero matrix element such that $\langle m'|E_{ji}|m\rangle \neq 0$ (where j>i) are superimposed, they must coincide at all levels above level j, (referred to as an upper walk) and also coincide at all levels below level i (referred to as a lower walk). The distinct row at level j is referred to as the loop head. The distinct row at level i-1 is referred to as the loop tail. The separate walks between the loop head and loop tail are referred to as the m and m' branches of the loop. Within the loop each level connects with the next lower level by a given loop segment. For single generators there are only 18 valid segment shapes (or types). For generator products there are many more such segment shapes. The body of this loop is depicted by the behavior of the segment shapes between levels i and j inclusive. The coefficient of integral $\langle i|\hat{h}|j\rangle$ is determined only by the body of the loop itself, thus different configuration pairs that share the same loop body with different upper and lower walks will have the same one-electron

integral contribution. By defining a loop only by its body, the number of times this given loop will contribute to the Hamiltonian is given by the product of the number of upper walks from the loop head and the number of lower walks from the loop tail ($\bar{x}_h \cdot x_t$). A sample one-electron loop is depicted in Figure 2.

The two electron integral contribution of the Hamiltonian is given by

$$\tfrac{1}{2} \sum_{i,j,k,\ell} [ij;k\ell] \langle m' | E_{ij}E_{k\ell} - \delta_{jk}E_{i\ell} | m \rangle \quad . \tag{12}$$

In our preliminary implementation the matrix element of generator products was determined by a factorization requiring intermediate states such that

$$\langle m' | E_{pq}E_{rs} | m \rangle = \sum_{m''} \langle m' | E_{pq} | m'' \rangle \langle m'' | E_{rs} | m \rangle \quad . \tag{13}$$

For levels where pq and rs do not overlap, the segment coefficients are identical to those of the single generator. An efficient recursive scheme was developed by Shavitt[24] for the evaluation of these generator products within the overlap region. There are, however, several drawbacks to this approach (when compared with newer approach), the most serious being that at any level within the overlap region up to three segment coefficients are needed when the loop-driven algorithm is used. Another difficulty is that the $\delta_{qr}E_{ps}$ term is not included which results in calculation of extra contributions that cancel later.

Work by Paldus[45] and Shavitt[46] since the completion of our preliminary implementation have suggested that the matrix element generator products be represented by

$$\langle m'|\varepsilon_{pq,rs}|m\rangle = \{ \prod_{i(\text{no overlap})} W(T_i, s_i' s_i, \Delta b_i, b_i) \cdot \sum_{x=0,1} \prod_{i(\text{overlap})} W(T_i, s_i' s_i, \Delta b_i, b_i, x) \quad . \tag{14}$$

where $\varepsilon_{pq,rs}$ is given by

$$\varepsilon_{pq,rs} = E_{pq}E_{rs} - \delta_{qr}E_{ps} \quad . \tag{15}$$

This representation is superior to the previous method for several reasons. One obvious reason is that each segment in the overlap region now has at most two coefficients. The x=0 branch and the x=1 branch elucidate the relationship between direct and exchange terms for integrals that are related by index permutation. When the use of loop types is employed where two integrals are involved, the coefficients of both integrals can be determined by the same two segment coefficients in the overlap region, resulting in a further reduction of work. One final reason that coefficients defined in this manner are superior is that the $\delta_{qr}E_{ps}$ term is included, eliminating all of the cancellation of repeated terms present in the earlier version of the method. The implementation of these new coefficient values is one of the major reasons for the factor of two reduction of computational times from those of the already impressive preliminary implementation.

III. METHODS AND DEVELOPMENTS OF THE UNITARY GROUP APPROACH CI

A. The Interacting Space

For any general CI method to be useful it must be able to include those classes of configurations that are important without being required to also include a vast set of insignificant configurations. One such important distinction concerns the Hartree-Fock interacting space,[29] those configurations that potentially interact with the references, compared with the full spin configuration space. Experience has demonstrated[47] that these additional configurations within the full spin space are unimportant in terms of correlation energy and electronic properties, and their exclusion substantially reduces the computational effort required for the CI calculation.

In its earliest electronic formulations, the effectiveness of the GUGA was only apparent for full CI wavefunctions. This capability was thereafter considerably extended by Shavitt[24,25] to cases involving different levels of excitation relative to a single reference configuration. Our contribution in turn was the extension of the GUGA to the multireference interacting space. Since the term "interacting space" is not widely understood in this context, a specific example is in order.

The restricted Hartree-Fock wavefunction for the triplet ground state of methylene (CH_2) is of the form

$$1a_1^2\ 2a_1^2\ 1b_2^2\ 3a_1\alpha 1b_1\alpha \quad . \tag{16}$$

Now consider a double excitation of the form $2a_1\ 1b_2 \rightarrow ij$. This gives rise to the new orbital occupancy

$$1a_1^2\ 2a_1\ 1b_2\ i\ j\ 3a_1\ 1b_1 \qquad (17)$$

with six unpaired spins. These six unpaired spins give rise to no fewer than nine triplet (S=1) spin eigenfunctions. Most conventional CI methods require either none or all of these configurations to be included variationally.[4] However, this 9-dimensional spin space can be partitioned[29] in such a way that only two of the configurations have nonvanishing Hamiltonian matrix elements with the Hartree-Fock reference configuration (16). The two "interacting" configurations are of the form

$$1a_1^2\ [2a_1\ 1b_2\ i\ j \Rightarrow {}^1A_1]\ 3a_1\alpha 1b_1\alpha \qquad (18)$$

in which the four orbitals in brackets have been coupled in the two possible ways yielding a singlet (S=0) state.

The seven configurations neglected above are exactly those which do not contribute in second-order perturbation theory. Further, numerical tests on methylene and other molecules have shown that the interacting space does include all energetically important configurations.[47] That is, although the interacting space includes typically less than half of all symmetry-adapted configurations in a given double excitation space, it yields $\sim$ 99.9% of the correlation energy due to the complete space.

Consider a singles and doubles CI calculation from a single doublet reference configuration that can be described by a single Slater determinant

$$1a^2\ 2a^2\ 3a^2\ \ldots\ n a\alpha \qquad . \qquad (19)$$

Most of the doubly excited configurations of this calculation (assuming a large enough problem) will have five unpaired spins. For a given orbital occupancy the full S=1/2 spin space will consist of five configurations. These configurations can be expressed as linear combinations of the ten possible M_S=1/2 Slater determinants. On the other hand, the interacting space will consist of only two of these configurations and can be represented by a linear combination of six of the ten possible Slater determinants. (Four of the unpaired orbitals are singlet coupled and the fifth unpaired spin, corresponding to the unpaired reference orbital, is restricted to be of the same spin occupation as in the reference.)

Using the Clebsch-Gordan expansion, the coefficients of the five Gelfand states expressed over determinants (ignoring the doubly occupied and unoccupied orbitals) are given in Table II. By inspection the first and second configurations are the only pair of configurations that can select the interacting space from the full spin space. Notice that for this combination the fifth unpaired spin must always be alpha occupied. This must correspond to the unpaired orbital of the reference configuration. In order to have the fifth unpaired spin to be the last for all possible double excitations that orbital must be the last (or the top) orbital of the distinct row table.

For higher spin states, such as triplets, to generate the interacting space all of the spin restricted orbitals must be at the top of the distinct row table. For a single open-shell singlet reference configuration the spin restricted orbitals have two possible positions on the DRT, at either extreme. For the sake of continuity

we chose to put them at the top also. The remainder of the orbitals may lie anywhere within the rest of the DRT except that all of the doubly occupied orbitals should be grouped together and all of the virtual orbitals should be grouped together. This is done so that it is easy to keep track of the excitation levels, without adding additional dimensions to the DRT. For an optimized method the larger of these two blocks should be at the bottom of the DRT. For calculations of double zeta quality or better, the virtual space is larger than the doubly occupied space; thus within the DRT the orbitals are ordered (from top to bottom): the active space, the doubly occupied space, and the virtual space. Removed from the DRT are the frozen core orbitals and the deleted virtual orbitals.

To implement the use of the interacting space, each row of the DRT is given a type (T) classification. This value, which corresponds to the number of excitations accounted for, determines the maximum number of electrons that can be excited into the virtual space for all walks containing this distinct row. For a singles and doubles CI, a value of T=0 implies that two electrons may be excited into the virtual space, because no excitations have occurred in the active space. A value of T=1 or T=2 implies that one or no electrons respectively may be excited into the virtual space. The T value is only meaningful for those rows of the DRT that are above the interface between the doubly occupied and virtual space. For the one reference doublet interacting space, if the orbital that is unpaired in the reference is doubly occupied (s=3) or beta coupled (s=2), the T value is set to one, because either of these occupations accounts for one of the two

possible excitations. The T values are determined by the active orbitals at the top levels of the DRT, and the same T values remain unchanged along any walk within the doubly occupied space. All potential excitations are compared with the T value to determine if that excitation is valid.

One problem with this approach to the interacting space is that it is possible for the two branches of a valid loop to end on two different rows that only differ by T values. The solution to this problem is to extend the loops for these cases until the rows coincide. This will occur at the level of the interface between doubly occupied and virtual orbitals. This loop extension will not alter the loop coefficient or the integrals that contribute to that loop, but is required so that the contribution will be added to the correct matrix elements. This extension is not a major computational consideration since only a very small percentage of the total loops will require this extension and all of the loops that require this extension are small.

This simple algorithm can be used to generate the interacting space from higher spin states as well as from open-shell singlet references.

B. The Multireference Space

Whereas all single and double excitations from a single reference configuration may be adequate for some molecular systems, many of the more interesting chemical problems require double excitations from more than one reference for an adequate description.[3,11,12,15,20] None of the other direct CI methods can handle these types of problems except for the most restrictive cases.[48] On the other hand the graphical unitary group approach is ideally suited for such problems because the configuration space is determined only by the DRT, which can be as complex as desired, to handle almost any problem.

For most cases the active orbitals, those orbitals that are either singly occupied or occupied differently by different references, must be adjacent within the distinct row table. For cases where the full spin space is desired, the active orbitals may be anywhere within the DRT, but with the most optimized method they should be between the doubly occupied and virtual space. Whenever the interacting space is also required the active orbitals must be placed at the top of the DRT. If the active orbitals are at the top of the DRT, for either full spin space or multireference interacting space calculations, then the same T values described in the interacting space section can be used to generate the multireference configuration space. For more complex multireference calculations, several different T type parameters must be used. Typically the number of T type parameters must be the same or less than the number of reference configurations, but these additional parameters are only needed within the active orbital space of the DRT. If the active orbitals are placed between the occupied

and virtual space, then another algorithm must be used. The system in its present form always restricts the active orbitals to be at the top of the DRT.

The method for implementing a given multireference calculation consists of two steps. First, a small model DRT without symmetry must be created describing the desired multireference configuration space; and secondly, an algorithm must be developed that can reproduce this model as well as the desired multireference configuration set in the general case. Once a given case has been worked out, the developed algorithm may be added to the rest of the system. The complexity of the model increases rapidly as the number of references increases, which can make this method quite difficult even for some three reference calculations. Table III gives an example of a model DRT for the interacting space from two open-shell singlet reference configurations. An important future advancement of the GUGA method will be the automation of the multireference configuration space generation. Once this has been done the capabilities of this method in terms of types of calculations will be almost limitless.

The algorithm that will make the GUGA method completely general is still in the design stage. In its current form it can roughly be broken up into the following steps:

1. Generate the distinct row table (DRT) with some or many unwanted configurations included.
2. Generate the indexing array.
3. Strike out unwanted configurations from the indexing array.
4. Restructure the indexing array in such a way as to maximize

similarities (this will result in the simplest final DRT).

5. From this new indexing array generate the compressed form of the indexing array.
6. Using these two arrays and the old DRT, generate a new DRT that has only the wanted configurations present.
7. Use this DRT for the rest of the CI.

This algorithm currently works for unrealistically simple cases, but its generalization in an efficient manner (that does not require a great deal of internal storage) will be rather difficult.

C. Spatial Symmetry

Orbital or spatial symmetry can be treated in several ways within the structure of the GUGA method. At the simplest level symmetry can obviously be ignored. This will result in an unnecessarily large formula tape as well as a much larger secular equation than necessary when symmetry is present. For example, if we are studying the $^3B_2$ ground state of methylene (C_{2v} point group), the number of unique configurations may be increased by a factor of 5 if spatial symmetry is ignored.

At the first level of sophistication, symmetry may be used only to eliminate those loops where the contributing integrals will be zero by symmetry. This approach will result in the smallest possible formula tape and fastest formula tape generation (if implemented properly); however the secular equation will still be large. The Hamiltonian matrix will be blocked with regards to the different irreducible representations, but the entire matrix will present itself to whatever diagonalization scheme is used. Since most diagonalization schemes require at least two real vectors in core at any one time, this approach is only feasible for large core machines. Even if some clever diagonalization scheme is devised that can reduce the problem of the large secular equation, it seems unlikely that the small savings in the formula tape (compared with other symmetry treatments) will warrant the increased difficulty in the diagonalization, especially if the formula tape is to be used more than once.

An innovative method proposed by Shavitt[49] left the DRT unchanged from that without spatial symmetry, except that for each row there are

stored separate configuration counting indices for each symmetry type of a nondegenerate point group. This results in the elimination of all configurations that are of a different irreducible representation than that of the reference. Each loop on the formula tape will have separate indexing indices for each symmetry structure of that loop.

We have carried this idea further by treating the different symmetry species of any row as if they were unrelated and only including those that have both a nonzero number of upper and lower walks. This provides for a simpler algorithm when constructing the loops except that a given loop may have to be created as many times as there are irreducible representations in the symmetry point group. Instead of making each row more complex, additional rows are added to the DRT by giving each row a symmetry type classification, where the symmetry type of any row is the product of the irreducible representations of all singly occupied orbitals (s=1 or s=2) of any lower walk. In principle this approach could multiply the complexity of the DRT by the number of symmetry types. This is an important consideration since both the computational effort required as well as the formula tape length is heavily dependent on the complexity of the DRT. The number of different symmetry type classifications at any level of the DRT can be expressed by

$$n_{\Gamma}(i) = \min [P_u(i), P_{\ell}(i)] \tag{20}$$

where P_u and P_{ℓ} are the number of different symmetry products of all possible combinations of the irreducible representations of all of the upper or lower orbitals, respectively. This number can vary greatly

depending on the order of the orbitals with regards to symmetry type. This suggests that the complexity of the DRT can be greatly reduced by a reordering of orbitals within each type of orbital space (active, doubly occupied, and virtual). The method adopted for our implementation results in a rigorous minimization of

$$\sum_{i=1}^{n} n_{\Gamma}(i) \quad , \tag{21}$$

where n is the number of levels. This is done by placing all of the symmetric orbitals at the extremes of the DRT so that n_{Γ} of as many levels as possible will be unity. Similar considerations are used in the ordering of the rest of the orbitals. This is apparently also equivalent to the minimization of the number of distinct rows in the DRT. For favorable cases the DRT is only slightly more complex than the DRT produced without symmetry.

D. The Importance of Loop Types

The two-electron integral contribution of the Hamiltonian is

$$\frac{1}{2} \sum_{i,j,k,\ell} [ij;k\ell] \langle m' | \varepsilon_{ij,k\ell} | m \rangle \quad . \tag{22}$$

The sum is over all ordering of the indices, and hence a given integral $[ij;k\ell]$ where i, j, k, and ℓ are all different will appear eight times with coefficients in terms of generator products of:

$$\begin{array}{ll} \tfrac{1}{2}\varepsilon_{ij,k\ell} & \tfrac{1}{2}\varepsilon_{k\ell,ij} \\ \tfrac{1}{2}\varepsilon_{ji,k\ell} & \tfrac{1}{2}\varepsilon_{k\ell,ji} \\ \tfrac{1}{2}\varepsilon_{ij,\ell k} & \tfrac{1}{2}\varepsilon_{\ell k,ij} \\ \tfrac{1}{2}\varepsilon_{ji,\ell k} & \tfrac{1}{2}\varepsilon_{\ell k,ji} \quad . \end{array} \tag{23}$$

As demonstrated by Paldus[50] and by Shavitt,[24] since only matrix elements where $m' \leq m$ are of interest, the last four of these contributions are zero. Also since

$$\varepsilon_{ij,k\ell} = \varepsilon_{k\ell,ij} \tag{24}$$

the first pair of these contributions is identical and the second pair is also identical. The contribution of this integral may be expressed as

$$[ij;k\ell] \langle m' | \varepsilon_{ij,k\ell} + \varepsilon_{ji,k\ell} | m \rangle \qquad (i<j<\ell,\ j \neq k<\ell) \tag{25}$$

A similar treatment of the remaining types of two-electron integrals[50,24] leads to the following results:

$$[ij;j\ell]\langle m'|\varepsilon_{j\ell,ij} + \varepsilon_{ji,j\ell}|m\rangle \qquad (i<j<\ell) \qquad (26)$$

$$[ij;i\ell]\langle m'|\varepsilon_{ij,i\ell} + \varepsilon_{i\ell,ji}|m\rangle \qquad (i<j<\ell) \qquad (27)$$

$$[i\ell;k\ell]\langle m'|\varepsilon_{i\ell,k\ell} + \varepsilon_{\ell i,k\ell} + \varepsilon_{\ell k,i\ell}|m\rangle \qquad (i<k<\ell) \qquad (28)$$

$$[ij;\ell\ell]\langle m'|\varepsilon_{ij,\ell\ell}|m\rangle \qquad (i<j,\ i\neq\ell,\ j\neq\ell) \qquad (29)$$

$$[i\ell;\ell\ell]\langle m'|\varepsilon_{\ell\ell,i\ell}|m\rangle \qquad (i<\ell) \qquad (30)$$

$$[ii;i\ell]\langle m'|\varepsilon_{ii,i\ell}|m\rangle \qquad (i<\ell) \qquad (31)$$

$$[i\ell;i\ell]\langle m'|\tfrac{1}{2}\varepsilon_{i\ell,i\ell} + \varepsilon_{\ell i,i\ell}|m\rangle \qquad (i<\ell) \qquad (32)$$

$$[ii;\ell\ell]\langle m'|\varepsilon_{ii,\ell\ell}|m\rangle \qquad (i<\ell) \qquad (33)$$

$$\tfrac{1}{2}[ii;\ell\ell]\langle m'|\varepsilon_{ii,ii}|m\rangle \ . \qquad (34)$$

It should be mentioned here that this representation of integral contributions in terms of generators is not unique. Since the choice of representation will affect the loop types that are used in the loop generator, other representations should be considered.

A rearrangement of these equations leads to the important development of using loop types. By combining integral generator products that can contribute to the same matrix elements (including the one-electron integrals), we find that there are fourteen different loop types.

1) $\langle m'|[ik;j\ell]\varepsilon_{ki,j\ell} + [ij;k\ell]\varepsilon_{ji,k\ell}|m\rangle \qquad (i<j<k<\ell) \qquad (35)$

2) $\langle m'|[i\ell;jk]\varepsilon_{kj,i\ell} + [ij;k\ell]\varepsilon_{ij,k\ell}|m\rangle \qquad (i<j<k<\ell) \qquad (36)$

3) $\langle m'|[ik;j\ell]\varepsilon_{ik,j\ell} + [i\ell;jk]\varepsilon_{jk,i\ell}|m\rangle \qquad (i<j<k<\ell) \qquad (37)$

4) $\langle m'|[ij;j\ell]\varepsilon_{ji,j\ell}|m\rangle \qquad (i<j<\ell) \qquad (38)$

5) $\langle m'|[ij;j\ell]\varepsilon_{j\ell,ij} + [jj;i\ell]\varepsilon_{jj,i\ell}|m\rangle \qquad (i<j<\ell) \qquad (39)$

6) $\langle m'|[ik;i\ell]\varepsilon_{ik,i\ell}|m\rangle \qquad (i<k<\ell) \qquad (40)$

7) $\langle m'|[ik;i\ell]\varepsilon_{i\ell,ki} + [ii;k\ell]\varepsilon_{ii,k\ell}|m\rangle \qquad (i<k<\ell) \qquad (41)$

8) $\langle m'|[i\ell;j\ell]\varepsilon_{i\ell,j\ell}|m\rangle$ $(i<j<\ell)$ (42)

9) $\langle m'|[i\ell;j\ell]\varepsilon_{\ell i,j\ell}|m\rangle$ $(i<j<\ell)$ (43)

10) $\langle m'|[i\ell;j\ell]\varepsilon_{\ell j,i\ell} + [ij;\ell\ell]\varepsilon_{\ell\ell,ij}|m\rangle$ $(i<j<\ell)$ (44)

11) $\langle m'|[i\ell;\ell\ell]\varepsilon_{\ell\ell,i\ell} + [ii;i\ell]\varepsilon_{i\ell,ii} + \langle i|\hat{h}|\ell\rangle E_{i\ell}|m\rangle$ $(i<\ell)$ (45)

12) $\langle m'|[i\ell;i\ell]\varepsilon_{i\ell,i\ell}/2|m\rangle$ $(i<\ell)$ (46)

13) $\langle m'|[i\ell;i\ell]\varepsilon_{\ell i,i\ell} + [ii;\ell\ell]\varepsilon_{ii,\ell\ell}|m\rangle$ $(i<\ell)$ (47)

14) $\langle m'|[\ell\ell;\ell\ell]\varepsilon_{\ell\ell,\ell\ell}/2 + \langle\ell|\hat{h}|\ell\rangle E_{\ell\ell}|m\rangle$. (48)

One obvious advantage of defining loop types in this manner is that loops now can represent a linear combination of integrals and this linear combination can be performed at an early stage of the calculation. A schematic of each loop type is shown in figure 3.[46]

Loop types 1, 2, and 3 are the most abundant, so most generated loop values will consist of a linear combination of two integrals. The importance of using loop types defined in this manner should not be underestimated. If loops were generated for each integral separately, not only would the effort required to generate the loops be doubled, but the same doubling would apply to the effort required each iteration of the diagonalization. Also, as stated previously, both integrals of any loop type that contains two integrals can be described by the same two coefficients (x=0, x=1) in the region where the generators of a generator product overlap.

When two Gelfand states differ by two orbitals there will be at most one contributing loop to the matrix element of the states. The vast majority of nonzero matrix elements fit into this category. Loops on the other hand can contribute to a great number of matrix

elements (but the average number of contributions per loop is about two for the reported test calculations). Thus a Hamiltonian stored in this manner is more compact than the conventional form of CI matrix storage.

Once the loop type structure has been chosen, this information must be transferred into a form that can be used by the program. The loop searching master table (Table IV) contains all of this information. Each loop is given four indices i, j, k, and ℓ, where $i \leqslant j \leqslant k \leqslant \ell$. For loop type 1 all four indices are different and for loop type 14 all four are the same. Each loop starts at its lop level ℓ and ends at its bottom level i-1. Any levels outside of this range have no effect on the generator coefficient values or integrals of the loop but would only affect the upper or lower walks of that loop and hence only affect which matrix elements a given loop will contribute to. The value of any loop depends only on the shape of that loop and the starting spin value (b_ℓ) of the loop head.

Each loop type is broken into at most four sections depending on the number of differing indices. Loop types 1-7 all behave identically within the ℓ to k section. When generating these loops the work required for this section can be performed once and used for any of the loops of these types. If each loop type was processed separately a sevenfold repetition of work would result in this section. Loop types that behave identically between the ℓ to j sections are:

1, 2, (and 7 if new loop types are used)

3 and 6

8 and 12

9, 10, and 13 .

Loop types 11 and 14 must be processed separately since they have no sections in common with any other loop types. Loop types that behave identically in the j to i section cannot be processed together but the same section of code can be used for each.

The use of loop types as a linear combination of integrals and overlaying them are essential features of the efficient implementation of the loop-driven algorithm.

E. The Loop-driven Methodology

The loop-driven algorithm is without question the most important element accounting for the success of our GUGA method. The term "loop-driven algorithm" is derived from the manner in which loops are generated. In a conventional CI (configuration driven), the next configuration on the configuration list determines what is done subsequently. With an integral driven approach it is the next integral on the list of integrals that determines what is processed next. In contrast to both of these approaches, with the loop-driven algorithm, the next loop that can be produced with the least effort is generated. Since most loops have large sections in common with many other loops, the savings are found in that the algorithm only generates portions of the next loop that differ from the previous loop. Generating loops in this manner corresponds to neither a configuration driven approach nor an integral driven approach. Not only does this method illuminate vast numbers of previously unappreciated relationships between otherwise distinct Hamiltonian matrix elements, it also points out the relationships between different integrals that share one or more common indices.

Figure 4 shows four sample two-electron loops. Once the first loop (solid lines) has been generated, the remaining three loops can be generated by simply finding and processing the segment shapes depicted by broken lines, usually with little additional effort. The four loops depicted also use different integrals that have little in common with one another except some common indices. Because of this, we claim that the integrals are processed simultaneously. While

generating the loops for one integral, major portions of loops for other integrals are also being produced. The method used to find the loops is based on an exhaustive tree search method,[51] allowing a systematic generation of all the loops for a given block of integrals. A block of integrals usually contains all two- and one-electron integrals that share a common largest index. At each intermediate level all valid lower segment shapes are investigated. If the accepted segment represents a valid closure, then that loop is processed and the search continues at the previous level. If at any level all segment shapes have been exhausted, then the search also reverts to the previous level. In order to generate all of the loops, this tree search must be processed for every possible loop head (each row of the DRT). The order in which distinct rows are processed in this manner is determined by the structure of the integral sorting arrays (see section III G.).

To process the above described tree search, pushdown stacks[51] are used. A pushdown stack is simply an array that spans the levels of the DRT. The use of these stacks is needed so that the program can "remember" where it left off at any level of the tree search when it returns to that level. Two of these stacks are coefficient stacks which are used to find the generator coefficients. Two are indexing stacks which are used to determine to which matrix elements the generated loop will contribute. Two of these stacks point to distinct rows of the DRT and are used to determine whether any given path to the next level is valid. An additional two stacks are segment stacks which determine the loop type section, starting segment shape,

and the segment shape that was last tested. The last is a tracking stack used to determine the final offsets (m values) of the integrals to be used in determining the loop value. Integral offsets are only updated whenever a k, j, or i value is reached; thus the offsets are not needed to be stored in pushdown stacks.

To minimize the searching, the different loop types are overlayed in all areas where they behave identically (as described in section III D.). Without this essential overlaying, the GUGA method would be at least a factor of two slower than it is in its present form. This is another manner in which the loop-driven algorithm provides insight into the Hamiltonian. Much additional unnecessary searching can be eliminated by recognizing at an early stage what segment shapes will not lead to valid loops despite the appearance of validity. In the first implementation such tests were primarily used with regards to the total symmetry structure of the loop (the m and m' branches of the loop must not differ by symmetry when the loop tail is reached). Since then, we have further reduced the amount of searching for valid segment shapes. Use of the new segment coefficients does not allow loops to end with both m and m' branches having an s value of zero at the bottom segment (the last orbital unoccupied in both Gelfand states). Thus, when there are no electrons remaining in all of the lower orbitals, there is no need to test any further segments along this line. Another reason that the amount of searching has been reduced is that by using the new segment coefficients, loop types 1, 2, and 7 now behave identically in the loop region between j and ℓ. In the preliminary implementation, they only coincided in the region

from k to ℓ. Since these types include the most common loops, the reduction of repetition in this area has led to improved timings, both because fewer segment shapes are searched for and because fewer segments are accepted and processed.

The loop-driven algorithm is a completely general method and it will efficiently generate all of the loops for any distinct row table (DRT), regardless of the complexity and the level of excitations included in the DRT.

F. The Loop Searching Master Table

The loop searching master table (Table IV) contains all of the information about loop types and how they are overlayed. Within the loop-driven formalism, it controls the path of the tree searching algorithm. In one sense this information completely defines the nature of the Hamiltonian and the DRT is only used as a template or backbone on which loops are generated. This information is in a similar form to that needed by an efficient implementation of this method. (In fact a listing of a working program using this method was consulted to produce this table.) This table is also an invaluable tool to anyone attempting to create or modify a program using the loop-driven algorithm.

The ISEG value refers to a particular searching section. For a given ISEG value all JSEG values within its range will be investigated at that level within the DRT before reverting to a previous higher level. For example, when ISEG=3, JSEG values from 35 to 52 are tested. Each JSEG value corresponds to a particular segment type. Not only is the shape of the segment determined but also its range and position within possible valid loops. The search is always initiated at the loop head where ISEG=1.

The columns labeled IS(Δb), by, and FS(Δb) uniquely determine the shape of the segment. These columns are not needed by the searching algorithm but are included here to make the table more readable. The heading IS refers to the "initial shape" (top of segment) and the heading FS refers to the "final shape" (bottom of segment); each identified by one or two letters. In this notation R represents a

raising generator and L represents a lowering generator between the m and m' branches of the loop at that intermediate level. A zero implies that the m and m' branches of the loop coincide at that intermediate level (a necessary condition for the loop head or tail). The segment shape is further specified by a Δb value where $\Delta b=b-b'$. Whenever an IS or FS term contains a "/" this implies that the segment shape may be used in a loop type where a linear combination of integrals is needed. When a compound integral track (two values) is present, the integral defined by the second track value often corresponds to the shape factor to the left of the "/". Whereas the IS and FS columns refer to the starting and ending shape, the "by" columns define the body of the loop segment. The letters R and L again refer to raising and lowering generators respectively. A bar over or under a letter indicates the initiation or termination of a generator (top or bottom segment of the generator range) respectively. The letter W indicates that a weight generator for this level is present. An asterisk on any generator simply means that a change of sign is introduced to the x=1 branch of the loop coefficient. Further details and use of this notation can be found elsewhere.[46]

The column labeled "next" specifies the next ISEG block that must be tested at the following lower level if the current segment at the present level is accepted. A "next" value of zero implies that this segment if accepted will terminate a loop and that this loop should then be processed before continuing further with the search. The s' and s values simply identify the case numbers for the m' and m branches of the loop. These values are compared with the arcs of

the Shavitt graph at the current position to determine if the current segment is acceptable. The column labeled "tracks" gives the final offset values used in determining the position of integrals within the current integral block. When a loop is terminated the current track value(s) is used when processing that loop. Whenever a segment is accepted where the track value is not specified, the track value(s) from the previous level is used as the current tracks value. A track value with only a single integer (m_1) implies that only one integral will contribute to the loop and that the X' coefficient will be used to determine the coefficient of that integral. For this case the loop value (v) is given by:

$$v = X' \; I(a' + m_1) \tag{49}$$

where I represents the array containing the current block of integrals and a' is the address offset within that block determined by (i, j, k, and ℓ) and is given by:

$$a' = K_{k\ell} + J_j(\Gamma_k \times \Gamma_\ell) + I_i(\Gamma_j \times \Gamma_k \times \Gamma_\ell) \tag{50}$$

where K, J, and I are the integral offset arrays (see section III G.). When the tracks value contains two integers (m_1,m_2) the loop value is given by:

$$v = X'[I(a' + m_1) + Z'I(a' + m_2)] \quad . \tag{51}$$

When the tracks value contains two or three integers (m_1,m_2,m_3) enclosed in parentheses only the X' value is used as the coefficient for all these integrals which implies that the loop value for three integrals can be expressed as:

$$v = X'[I(a' + m_1) + I(a' + m_2) + I(a' + m_3)] \quad . \qquad (52)$$

The column labeled "kjcond" contains information used to determine if a "k" or "j" value has been reached at the current level. When a "k" or "j" level has been reached its contribution to the integral address offset is determined. A k-condition of -1, 0, and 1 imply the k level has not yet been reached, (the current segment is above the k segment), the k segment has already been reached (the current segment is below the k segment), and that the current level is the k level segment, respectively. The distinction between k-condition values of -1 and 0 is needed to test whether or not a block of integrals sharing a largest common index has been divided into subgroups (due to internal storage limitations for large calculations). The j-condition can have only two values, 1 and 0. A value of 1 implies that the current level is the j level segment. It should also be pointed out that the j level segment cannot be above the k level segment and that a valid loop may not terminate unless both the j and k level segments have been reached.

As stated previously (section II C.) the coefficient of any two-electron integral in a particular loop can be represented by:

$$\langle m' | \varepsilon_{pq,rs} | m \rangle = \{ \prod_{i(\text{no overlap})} W(T_i, s_i', s_i, \Delta b_i, b_i) \cdot$$

$$\sum_{x=0,1} \prod_{i(\text{overlap})} W(T_i, s_i', s_i, \Delta b_i, b_i, x) \quad , \qquad (53)$$

where there is a different set of W coefficients for any loop segment. These coefficients are stored in equations under the heading of

"coefficients". Within these equations there are two pushdown stacks[51] (X and Y) and a third simple variable Z'. The pushdown stacks used here can simply be thought of as arrays that span the levels of the DRT. During loop generation, when a segment is accepted the composite contribution of all upper segments (X and/or Y) are updated with the contribution of the current loop segment, and the new composite contribution is stored at the current level (X' and/or Y'). The X stack is always used at every level and the Y stack is only used for the x=1 term when both x=0 and x=1 terms are present. The Z' element is occasionally changed. The segment coefficients are expressed in terms of the initial b value of the m branch of the loop using the definitions:

$$t = 1/\sqrt{2} \tag{54}$$

$$A(p,q) = \sqrt{\frac{b+p}{b+q}} \tag{55}$$

$$B(p) = \sqrt{\frac{2}{(b+p)(b+p+1)}} \tag{56}$$

$$C(p) = \sqrt{\frac{(b+p-1)(b+p+1)}{b+p}} \tag{57}$$

$$D(p) = \sqrt{\frac{(b+p-1)(b+p+2)}{(b+p)(b+p+1)}} \tag{58}$$

$$E(p) = \sqrt{\frac{2}{(b+p)(b+p+2)}} \tag{59}$$

The coefficient codes listed in the last column simply assign an arbitrary number to each unique set of coefficient equations. This allows an easy and compact storage of the coefficient information through the use of a large "computed GO TO" within the loop generation algorithm.

It should be mentioned that the choice of segment coefficient and overall structure of this table is not unique, and a large number of decisions were made in its preparation. Most of the credit in determining the segment coefficients presented here goes to Paldus[45] and Shavitt.[46] When some of the newer developments are included such as repartitioning the Hamiltonian relative to a "reference state" instead of relative to the "vacuum" as described by Shavitt,[46] the loop searching master table must be expanded and adapted to allow for this additional complexity. One major advantage of presenting the Hamiltonian in this manner is that when a change of this nature is required, the loop generation algorithm need not be altered. It is sufficient to change only the information stored in the table.

G. Integral Sorting and Storage

There are three reasons for resorting[52] the one- and two-electron integrals for the GUGA method. The most important reasons is that many loops require two or three integrals to determine the loop value, and these different integrals must be in core at the same time. Even better, if the different integrals needed were stored in adjacent positions, then only one address pointer would be required for any loop. Since loops are generated by the tree search method, all integrals used for any given tree search will have a largest integral index of ℓ, the level of the loop head. It is therefore highly advantageous to hold all integrals of a given largest index in core at any one time, and they should all belong to the same integral block if possible. Another reason for sorting the integrals is that once they are sorted, they can be stored in the manner presented here. For some cases this implicit addressing storage method requires only half of the storage space required with a conventional integral tape that contains a label and value for every integral. The last reasons mentioned here is that since a reordering of the orbitals is almost a necessity for the GUGA method, the integrals must undergo an indices change, unless the orbitals are reordered before the integral transformation. The transformation algorithm developed for use with GUGA CI applications creates and stores integrals in this manner thus eliminating an explicit sorting step later with the additional benefit of reduced external storage requirements.

The integral sorting could be avoided with an integral driven method, but as stated previously this type of approach would result

in a factor of 2 increase in the size of the formula and diagonalization tapes. The integral driven approach would also result in a many-fold repetition of work in the loop generation steps. Another way of understanding this difference is that the integral driven approach treats each integral separately whereas the loop-driven algorithm processes a block of integrals simultaneously. For these reasons this approach would be at best an order of magnitude less effective than the loop-driven algorithm, (unless loops are generated implicitly).

For a given loop head all integrals that could be required are stored, if possible, in the same block. If there are more of these integrals than the block size, then these integrals are divided into different blocks in such a way that the repetition of work is minimized. The integral blocks are defined in such a way as to minimize the number of blocks required for a given available core with the above restriction.

Each integral is assigned a block and a position as follows:

(1) Each integral is given a type (t, t=1-7) and a subtype (m, m=1-3) and an i, j, k, and ℓ value where $i \leqslant j \leqslant k \leqslant \ell$.

The possible types and subtypes are:

t	m	Integral	Condition
1	1	$[ik;j\ell]$	$i<j<k<\ell$
	2	$[ij;k\ell]$	
	3	$[jk;i\ell]$	
2	1	$[ij;j\ell]$	$i<j=k<\ell$
	2	$[jj;i\ell]$	
	3	Not used	
3	1	$[ik;i\ell]$	$i=j<k<\ell$
	2	$[ii;k\ell]$	
4	1	$[i\ell;j\ell]$	$i<j<k=\ell$
	2	$[ij;\ell\ell]$	
	3	Not used	
5	1	$[i\ell;\ell\ell]$	$i<j=k=\ell$
	2	$[ii;i\ell]$	
	3	$\langle i\|\hat{h}\|\ell\rangle$ + (frozen core)	
6	1	$[i\ell;i\ell]$	$i=j<k=\ell$
	2	$[ii;\ell\ell]$	
7	1	$[\ell\ell;\ell\ell]$	$i=j=k=\ell$
	2	$\langle\ell\|\hat{h}\|\ell\rangle$ + (frozen core)	. (60)

The positions recorded as "not used" are kept as place holders (in cases $i\neq j$) so that the addressing can be defined simply by arrays. For large calculations these holders should only increase the integral space required by about 4%. Integrals that correspond to frozen core or deleted virtual orbitals are not stored. There is also no space held for integrals that are zero by symmetry.

(2) Given an i, j, k, ℓ, and m value, the address a within the proper block (determined by k and ℓ) is given by:

$$a = K_{k\ell} + J_j(\Gamma_k \times \Gamma_\ell) + I_i(\Gamma_j \times \Gamma_k \times \Gamma_\ell) + m \quad , \qquad (61)$$

where Γ_i is the irreducible representation of orbital i. K, J, and I are offset arrays generated based on the symmetry of each orbital and the number of orbitals. The integral addresses are generated during the loop generation. As k, j, and i values are reached, the integral offsets are updated. This minimizes the amount of computational effort required in generating integral addresses. The m values for any given loop are determined by the tracking pushdown stack.

Within this framework entire blocks of integrals are processed together; that is, all possible loops requiring these integrals are generated sequentially. This is also used very efficiently as a direct CI method[7] where the storage of the diagonalization tape is avoided.

H. The Loop Breakdown Algorithm (Solution of the Upper Walk Problem)

In the early stages of the development of the GUGA CI method one of the major stumbling blocks was that there was no simple way to easily find all of the Hamiltonian contributions of a given loop once it had been generated. Due to the nature of lexical ordering the lower walks of a loop were sequential, thus easily determined. This problem was then reduced to the task of finding all of the upper walks for a given loop ("the upper walk problem"). At that time there were several proposed methods to solve or circumvent this problem but of these all had serious drawbacks. One solution that would have been prohibitive in terms of computational efficiency, was to simply search for all upper walks whenever a loop was generated. Another solution with a similar deficiency would be to regenerate a given loop at different times for each upper walk. The only solution which appeared to be acceptable was to consult a list of upper walks from the loop head of each generated loop, but this approach was discarded in favor of the more efficient loop breakdown algorithm presented here.

Due to the nature of the lexical ordering[24] of configurations (walks), all configurations that pass through a given distinct row with a given upper walk are contiguous. All of these configurations can be simply referenced by a starting position and a length.

$$m = m_r + k \qquad k = 1, 2, \ldots x_r \quad , \tag{62}$$

where m_r is the weight of the given upper walk and x_r is the number of lower walks of that row. Whereas the lexical ordering of states greatly facilitates finding configurations of lower walks through any

row, this is not so for upper walks. The upper walk problem is to find an effective algorithm that can generate all the upper walks for any loop. Consider a different ordering of states where all configurations passing through a given distinct row with a given lower walk are contiguous or, in other words, all upper walks are sequential. This ordering of states would be roughly equivalent to the lexical ordering of states if the DRT was inverted. This ordering of states, termed the "reverse lexical order," is related to the lexical ordering of states by the indexing array Y, such that

$$\bar{m} = Y(m) \tag{63}$$

where $\bar{m}$ is the reverse lexical order index and m is the lexical order index of a given state. Each distinct row is also assigned a primary upper walk weight, z_r, and the number of upper walks, $\bar{x}_r$, as well as the number of lower walks, x_r. All states that pass through a given row are given by:

$$m = Y(z_r + k) + j \qquad \begin{matrix} j = 1, 2, 3, \ldots, \bar{x}_r \\ k = 1, 2, 3, \ldots, x_r \end{matrix} \quad . \tag{64}$$

Here the final states are in reverse lexical order. To extend the use of the indexing array to loops, all matrix element contributions $H_{\bar{m},\bar{m}'}$, can be found simply by

$$\begin{matrix} \bar{m} = Y(z_h + m_\ell + k) + j \\ \bar{m}' = Y(z_h + m'_\ell + k) + j \end{matrix} \qquad \begin{matrix} j = 1, 2, 3, \ldots, \bar{x}_h \\ k = 1, 2, 3, \ldots, x_t \end{matrix} \quad , \tag{65}$$

where m_ℓ and m'_ℓ are the weight of each branch of the loop and the subscripts h and t refer to the loop head and tail rows, respectively. Again the final indices are in reverse lexical order. A given loop will contribute to the Hamiltonian $x_t \cdot \bar{x}_h$ times. This breakdown algorithm is needed in the diagonalization step and in the generation of the one- and two-particle density matrices, since the information required for this algorithm can be stored in a compact form with only three quantities needed:

$$\begin{aligned} &1)\quad z_h + m_\ell \\ &2)\quad z_h + m'_\ell \\ &3)\quad x_t \qquad . \end{aligned} \tag{66}$$

Here $\bar{x}_h$ is not present because it is best stored as a flag since its value is constant for all loops from a given loop head. The final loop value must also be written as a real number.

Due to the nature of the indexing array Y all contributions from any given loop will lie on the same super-diagonal (i.e. $\bar{m}-\bar{m}' = Y(z_h+m_\ell)-Y(z_h+m'_\ell)$, which is a constant). This loop breakdown algorithm allows a rapid determination of all matrix element contributions and its only drawback is that an integer indexing array that exactly spans the configuration basis must be held in core during the diagonalization.

I. Implicit Treatment of the External Space

For a theoretical study in which only single and double excitations into the external space are allowed, it is possible to generate loops that are mostly or wholly within this space in a manner that is more efficient than the loop-driven algorithm. When the external orbitals are placed at the bottom of the DRT, the Shavitt graph is quite simple for reasons described earlier. At any external level there can be (ignoring symmetry distinctions) at most four rows, designated W (two electrons singlet coupled in lower orbitals), X (two electrons triplet coupled in lower orbitals), Y (one electron in lower orbitals), and Z (no electrons in lower orbitals). Because of this simple structure all loop shapes and coefficients in this area can be implemented implicitly. This idea was first suggested by Siegbahn[53] for an integral driven approach. Of course the generation of loops for a given integral on demand by using the factorization of coefficients over levels is a slow and repetitious process. Thus the only hope for an integral driven approach is to generate the vast majority of loops in an implicit manner. This idea was carried further by Shavitt,[54] who worked out the shapes and details for all of the different loop types with the virtual orbitals correctly placed at the bottom of the DRT. We have implemented the generation of the all-external implicit loops in a manner similar to the method suggested by Shavitt but with a few modifications. The method used is somewhat analogous to the loop-driven algorithm, in that we proceed by bringing in a block of integrals (abc) for a given largest integral index (d) stored in the manner described in section III G. Once this is done one enters

a set of three nested DO loops over the remaining indices (abc) where the number of operations in the inner loops has been minimized so that any symmetry pointer or offset that is used for different integrals is computed only once. In the next to innermost DO loop where the indices (bcd) have been established, the loops are essentially determined. The final offset is added and the GUGA loops are processed in the innermost DO loop. Also, in this formulation it is easy to treat those integrals with coefficients of ±1 in a more efficient manner by avoiding unnecessary multiplications.

We find that the implicit generation of loops is almost a factor of two faster than the loop-driven algorithm for the all-external space. For the partly external loops this ratio will not be as large as two-to-one. For Example I (BH_3, 34 orbitals in C_{2v} sugbroup) the loop-driven algorithm required 23.0 seconds to generate the all-external loops and the implicit generation required 13.4 seconds to generate the same loops within the 30 virtual orbitals. The time required for the remaining loops was 125.7 seconds, giving a total savings of 7% (neglecting time required for DRT generation and integral sorting) by the implementation of the implicit generation of all-external loops when the ratio of external to internal orbitals is 7.5 to 1. When the implicit generation is also put into practice for loops with two or three external indices, the overall savings will be perhaps 30% compared with the purely loop-driven algorithm. For Example II (CH_2, 61 orbitals C_{2v} point group) where the ratio of external to internal orbitals is 11 to 1 the percent savings in the loop generation phase is found to be 9.4% using the implicit generation of the all-external loops.

J. Diagonalization Tape vs. Direct Method

As shown earlier, the GUGA method is quite suitable as a formula tape method, albeit the formula tape generation step is so fast that it may not be worth saving from one calculation to the next. In this section we discuss the calculations in which the formula tape generation is omitted and the diagonalization tape (symbolic formulas already combined with molecular integrals) is generated directly. Since no sorting is required, it is also possible to implement the loop-driven algorithm as a direct method,[7] in which all of the loops are generated during each iteration of the diagonalization without losing any efficiency in terms of the simultaneous treatment of integrals.

For Example I (BH_3, 34 orbitals, C_{2v} subgroup) the computation time required to generate and process all of the loops (neglecting DRT generation, buffer packing and integral sorting) is 122.9 seconds for the loop-driven algorithm. The time required for the matrix multiplication $\sigma_i = \sum_j H_{ij}c_j$ at each iteration of the diagonalization is 32.2 seconds for an overall ratio of 3.8 to 1. It should be mentioned again that, because of the nature of the loop-driven algorithm, each nonzero H_{ij} has (on the average) just over one contributing loop, because loops are linear combinations of integrals, whereas for an integral driven method there will be almost two contributions per nonzero H_{ij}. Thus, the iteration time required for the multiplications in $\sigma_i = \sum_j H_{ij}c_j$ will be almost twice as large for an integral driven method.

The ratio of loop generation time to diagonalization iteration time is important because it will become cost effective to generate all of

the loops each iteration of the diagonalization when this ratio becomes small enough. For each machine the break-even point will depend on the particular charging algorithm used to calculate the cost. With the advantages of treating the all-external space implicitly the ratio reduces to 3.5 to 1. When the parts of other loops with three and two external indices are treated implicitly, the savings may be perhaps 30% over the purely loop-driven algorithm, and with that improvement the ratio will be reduced to 2.7 to 1. It is our contention that further improvements elsewhere can bring this ratio to 2 to 1; that is, it will take only twice as long to create the complete Hamiltonian matrix as it does to multiply it by a vector. Even if this approach is not cost effective, for very large calculations, the external storage requirement of the diagonalization tape may make the direct method preferable. Although many of our algorithms have been written with the direct method in mind, we cannot implement this on our Harris minicomputer[55] for any reasonably sized calculation because of severe internal (core) storage requirements.

K. Large Matrix Diagonalization Method

For the GUGA method, a straightforward diagonalization using the method of Davidson[56] requires that two real vectors and one integer vector spanning the configuration basis, and the buffer space reside in core simultaneously. On a minicomputer such as ours[19] (with only 20K of real words available to the user) this limits the size of the secular equation to at most 8000 configurations. By today's standards this is not considered to be large, and many problems of current interest require considerably more configurations than this.[57] To develop a method for minicomputers that will allow very large matrices to be diagonalized, it is clear that the matrix must be partitioned. One method of doing this is to read the entire Hamiltonian several times, using only the portions that correspond to the particular section that is currently being processed. However this rapidly becomes prohibitive when the vector is broken up into n sections if n is more than two, since the number of passes is given by $n(n+1)/2$. Clearly what is needed is to sort the matrix into $n(n+1)/2$ sections, each of which can be read when the proper portion of each vector is in core. If n is minimized in such a way that only two portions of vectors are needed at one time, most of the Hamiltonian will need to be read and processed twice, since only the lower half of the Hamiltonian is stored. Taking another approach, doubling the size of n implies that each portion of the CI vector is half as large so that four portions can be held in core at once. This allows the step $\underset{\sim}{\sigma} = \underset{\sim\sim}{H}\underset{\sim}{c}$ to be carried out by reading the matrix only once and processing a given H_{ij} twice when $i \neq j$ (the only drawback being an

increased number of Hamiltonian sections).

Since 20K real words are available to the user, after leaving some space for buffers each portion of the vector can at most be about 4000 configurations. A rather fortuitous coincidence is that the Harris Slash Four word size is 24 bits. Half of a word is thus 12 bits, which can be used to specify an address from 1 to 4096. Thus by choosing 4096 as our block size we also can specify a given element (I and J) with a single word.

Another fortuitous occurrence involves the use of the GUGA in reverse lexical ordering as described in the loop breakdown section. The average loop has more upper walks than lower walks, and each contribution for different upper walks can be specified by a starting matrix element and a length, which define other matrix element contributions along the given super-diagonal. These contributions will usually lie sequentially within the same Hamiltonian block. The combination of storing I and J addresses in a single word and using reverse lexical ordering allows us to store an entire Hamiltonian in about half the space required for the corresponding conventional matrix. Since this method is limited only by available disc space (up to about 30,000 configurations) this essentially doubles our capabilities.

The final "matrix" tape is also smaller than the unsorted diagonalization tape, so the Yoshimine sorting method[58] we use allows the final matrix tape to be stored in the same external space that the original unsorted diagonalization tape is read from.

One of the largest variational wavefunctions determined stemmed from a three reference interacting space treatment (two closed shell

singlets, one open shell singlet) of SO_2 that contains 23,613 configurations. This calculation required about 60 megabytes of external storage, or less than half what is available to any user on our system. Thus calculations on the order of 25,000 will become routine on our minicomputer. For this calculation, the diagonalization tape was generated in only 39 minutes at a real cost of about $5. The time required to sort this matrix is 57 minutes; however, this sorting would not be necessary if enough core storage were available. Each iteration of the diagonalization step required 19 minutes and a total of 9 iterations were required to obtain the second root only. The diagonalization time is between 10% and 20% longer than it would be on a similar computer that could store and process two complete real vectors in core. When including the sorting time as well as the reduced efficiency, we find that the present diagonalization procedure increases the (unlimited central memory) diagonalization time for any given calculation by about 50%. Since the diagonalization step is not usually the most time-consuming portion of a single calculation, we do not find this to be a major drawback.

L. Density Matrix Analysis

We have also developed a one-particle density matrix analysis which enables us to obtain one-electron properties at the CI level. Unlike conventional CI methods, where the optimum method for generating the density matrix is to read and process the one-electron portion of the formula tape, we do not have a separate one-electron portion of the formula tape, nor a configuration list. Instead we simply generate all of the one-electron loops and use the loop breakdown algorithm to rapidly find all of the CI vector coefficients. We find that the time required to generate the one-particle density matrix by this method is negligible compared with the other steps involved in the CI. Once the one-particle density matrix is created, it can be transformed to the AO basis and then passed to a one-electron properties routine.

M. Avoidance of the Full Integral Transformation

Previous work by Pople and coworkers[38] describes a method that partly avoids the explicit (m^5) four-index transformation of atomic integrals to a molecular basis. In particular, their work shows how the integrals [ab;cd] never need to be computed (where all of the indices (a,b,c,d) correspond to the external space). Their method allows the contributions of the all-external integrals to be included directly from the atomic integrals $[\lambda\mu;\nu\sigma]$, usually with a reduction of necessary operations. This approach has some similarities to the self-consistent electron pair (SCEP) method of Meyer[59] and Dykstra et al.[60]

One of the drawbacks to Pople's formalism is that it is only applicable to CI methods that include up to double replacements from a single reference Slater determinant. We will show how this formalism can be extended to a completely general CI procedure over spin-adapted Gelfand states of a multireference interacting basis, or any other set of Gelfand states where at most two electrons are allowed into some external space.

Using Davidson's method for diagonalization,[56] each iteration of a $\underset{\sim}{\sigma}$ vector must be found such that

$$\sigma_I = \sum_J H_{IJ} C_J \quad . \tag{67}$$

The all-external two-electron integral contribution to the $\underset{\sim}{\sigma}$ vector can be expressed as

$$\sigma_I^{ext} = \sum_J \sum_{\lambda\mu\nu\sigma} C_J \, c_{\lambda a} c_{\mu b} c_{\nu c} c_{\sigma d} [\lambda\mu;\nu\sigma] \, b_{IJ}^{abcd} \tag{68}$$

where C_J is the CI coefficient of configuration J and $[\lambda\mu;\nu\sigma]$ is an integral in the AO basis. The b_{IJ}^{abcd} term is simply a spin coupling coefficient where the indices (a,b,c,d) are all restricted to the external orbitals. For a determinant based CI the coupling coefficients b_{IJ}^{abcd} will either be unity or zero. On the other hand, when a Gelfand configuration basis is used the coupling coefficients become more complex. This additional complexity requires some modification of the formalism presented by Pople.[38]

If two electrons occupy the external space, they can only be coupled in two ways (singlet or triplet coupled) if the ordering of orbitals is chosen in the manner described earlier (where all of the external orbitals are placed at the bottom of the Shavitt graph or DRT). If the electrons are triplet coupled, then this Gelfand state can only have a nonzero matrix contribution of the all-external integrals with other Gelfand states that also have two triplet coupled external electrons. These states are referred to as X states and the spin coupling coefficients are given by

$$b_{IJ}^{abcd} = \delta_{I_{up},J_{up}} \, \mathrm{sgn}[a-c] \, \mathrm{sgn}[b-d] \tag{69}$$

where I_{up} refers to a particular upper walk of configuration I in the internal space. The function sgn[x] is defined as

$$\begin{aligned} &\mathrm{sgn}[x] = 1 \text{ if } x > 0 \\ &\mathrm{sgn}[0] = 0 \\ &\mathrm{sgn}[x] = -1 \text{ if } x < 0 \quad . \end{aligned} \tag{70}$$

For the X states, the coefficients are already factored into an ac term

and bd term.

States where the two external electrons are singlet coupled are referred to as W states. The spin coupling coefficients for these states are given by

$$b_{IJ}^{abcd} = \delta_{I_{up},J_{up}} [(1+\sqrt{2}-1)\delta_{ac})(1+(\sqrt{2}-1)\delta_{bd} - \delta_{ac}\delta_{bd}] \quad . \tag{71}$$

For the W states, it is not as easy to factor the spin coupling coefficient into ac and bd parts. By rewriting configuration I as either $up_W \rightarrow ac$ or $up_X \rightarrow ac$, equation (68) can be rewritten as

$$\sigma_{up_W \rightarrow ac}^{ext} = \sum_{bd} \sum_{\lambda\mu\nu\sigma} C_{up_W \rightarrow bd}\ c_{\lambda a} c_{\mu b} c_{\nu c} c_{\sigma d}\ [\lambda\mu;\nu\sigma] \cdot$$

$$[(1+(\sqrt{2}-1)\delta_{ac})(1+(\sqrt{2}-1)\delta_{bd}) - \delta_{ac}\delta_{bd}] \tag{72}$$

and

$$\sigma_{up_X \rightarrow ac}^{ext} = \sum_{bd} \sum_{\lambda\mu\nu\sigma} C_{up_X \rightarrow bd}\ c_{\lambda a} c_{\mu b} c_{\nu c} c_{\sigma d}\ [\lambda\mu;\nu\sigma]$$

$$\text{sgn}[a-c]\ \text{sgn}[b-d] \quad . \tag{73}$$

Rather than first finding [ab;cd] as is usually done, these equations can be solved by first forming the intermediates $F_W^{\mu\sigma}$, $F_W^{\mu\sigma'}$, and $F_X^{\mu\sigma'}$ defined as

$$F_W^{\mu\sigma} = \sum_{b>d} (c_{\mu b} c_{\sigma d} + c_{\mu d} c_{\sigma b})\ C_{up_W \rightarrow bd} \tag{74}$$

$$F_W^{\mu\sigma'} = \sum_{b} \sqrt{2}\ c_{\mu b} c_{\sigma d}\ C_{up_W \rightarrow bb} \tag{75}$$

and

$$F_X^{\mu\sigma} = \sum_{b>d} (c_{\mu b}c_{\sigma d} - c_{\mu d}c_{\sigma b})\, C_{up_X \to bd} \quad . \tag{76}$$

The next step, which will be the most time-consuming, combines the atomic integrals with these F terms to form

$$\tilde{F}_W^{\lambda\nu} = \sum_{\mu\sigma} F_W^{\mu\sigma}\, [\lambda\mu;\nu\sigma] \tag{77}$$

$$\tilde{F}_W^{\lambda\nu'} = \sum_{\mu\sigma} F_W^{\mu\sigma'}\, [\lambda\mu;\nu\sigma] \tag{78}$$

and

$$\tilde{F}_X^{\lambda\nu} = \sum_{\mu\sigma} F_X^{\mu\sigma}\, [\lambda\mu;\nu\sigma] \quad . \tag{79}$$

Using the $\tilde{F}$ intermediates the σ^{ext} contribution can easily be found as

$$\sigma^{ext}_{up_X \to ac} = \sum_{\lambda\nu} [(\tilde{F}_X^{\lambda\nu} + \tilde{F}_X^{\lambda\nu'})(c_{\lambda a}c_{\nu c} + c_{\lambda c}c_{\nu a} + (\sqrt{2}-2)c_{\lambda a}c_{\nu a}\delta_{ac}) - \frac{\sqrt{2}}{2}\tilde{F}_X^{\lambda\nu'}(c_{\lambda a}c_{\nu a}\delta_{ac}) \tag{80}$$

and

$$\sigma^{ext}_{up_X \to ac} = \sum_{\lambda\nu} \tilde{F}_X^{\lambda\nu}\,(c_{\lambda a}c_{\nu c} - c_{\lambda c}c_{\nu a}) \quad . \tag{81}$$

As stated previously, the most time-consuming step is described by equations (77-79) and it is of the order (n^2m^4) where n and m are the number of internal orbitals and the number of basis functions respectively. The step that this avoids reduces the integral transformation from an (m^5) process to an (nm^4) process. Therefore, whether this type of approach is optimum for a given CI problem will depend on the relative comparison of n^2 and m as well as other factors. It should

also be remembered that the procedure presented here must be performed every iteration of the diagonalization. As it will be shown in a later section, the real advantage of this type of an approach will come about for analytic gradients of CI energies.

IV. ANALYTICAL GRADIENTS FOR CI ENERGIES

One of the most promising aspects of the GUGA formalism will be in the determination of analytical gradients (first derivatives) of configuration interaction energies with respect to changes in nuclear positions.

The current conventional approach to calculating energy derivatives at the CI level is through the use of the finite-difference method. This is done by performing several distinct calculations (at least two) at nearby geometries for each degree of freedom that is needed. The desired energy derivative for any perturbation can be approximated by

$$\frac{\partial E(x)}{\partial x} \approx \frac{E(x+\Delta x)-E(x)}{\Delta x} \quad . \tag{82}$$

Whereas the methods required to obtain CI derivatives in this manner are much simpler than those required to obtain analytic CI gradients, there are several drawbacks to this approach. The most obvious drawback is that it is an approximation. If the step size Δx is too large, the contamination effects of higher derivatives could become important. On the other hand, if the step size is too small, roundoff effects could dominate. The problem of roundoff errors at the CI level are much more pronounced than those at the SCF level (due to the nature of the four-index transformation). Perhaps a more serious drawback is that finding energy derivatives in this manner can be quite costly. Usually three calculations are required to obtain any derivatives accurately, thus for any (non-linear) molecule the approxi-

mate cost to find all of the derivatives can be expressed as

$$(3N-6) \times 3 \times I_1 \tag{83}$$

where I_1 is the cost of a single calculation. Using the methods presented here, the cost required to obtain the identical results would be

$$I_1 + I_2 \tag{84}$$

where I_2 is the cost to obtain all of the energy derivatives, once the CI calculation has been completed. Since (as it will be shown) that I_2 is approximately equal to I_1 for most cases, and for difficult cases I_2 could be twice as large as I_1, savings of an order of magnitude could easily be found for a molecule as large as C_4H_6 using analytic gradient methods.

In this section, the theory behind these new methods will be presented, as well as some details and insight into the first implementation (which at the time of this writing is near completion).

A. Theory of CI Energy Gradients

The GUGA method will be extremely useful in determining analytical gradients[61] (first derivatives) of CI energies with respect to changes in nuclear positions. It is clear that not only can this method be implemented in such a way as to be more efficient than performing several distinct CI calculations, but also in such as way as to be flexible enough to be useful for a general large scale multireference CI with any desired configuration set.

For any CI wavefunction

$$\phi = \sum C_I \phi_I \tag{85}$$

the energy can be written as

$$E = \sum C_I C_J (a_{IJ}^{ij} \langle i|\hat{h}|j\rangle + b_{IJ}^{ijk\ell} [ij;k\ell]) \quad . \tag{86}$$

The derivatives of the electronic energy can then be expressed as

$$E^a = \frac{\partial E}{\partial x_a} = \sum C_I C_J (a_{IJ}^{ij} \langle i|h|j\rangle^a + b_{IJ}^{ijk\ell} [ij;k\ell]^a) \quad , \tag{87}$$

since the CI coefficients are variationally determined, and thus the formula for the gradient will not involve their differentials.[61] In equation (87) a_{IJ}^{ij} and $b_{IJ}^{ijk\ell}$ are simply spin coupling constants.[63] The derivatives of the integrals are given by

$$\langle i|\hat{h}|j\rangle^a = \sum_{\lambda\mu} c_{\lambda i}c_{\mu j} \langle\lambda|\hat{h}|\mu\rangle^a + \sum_{\lambda\mu} (c^a_{\lambda i}c_{\mu j} + c_{\lambda i}c^a_{\mu j}) \langle\lambda|\hat{h}|\mu\rangle$$

$$= \sum_{\lambda\mu} c_{\lambda i}c_{\mu j} \langle\lambda|\hat{h}|\mu\rangle^a + \sum_{r} (U^a_{ri} \langle r|\hat{h}|j\rangle + U^a_{rj}\langle i|\hat{h}|r\rangle) \qquad (88)$$

and by

$$[ij;k\ell]^a = \sum_{\lambda\mu\nu\sigma} c_{\lambda i}c_{\mu j}c_{\nu k}c_{\sigma\ell} [\lambda\mu;\nu\sigma]^a = \sum_{r} (U^a_{ri} [rj;k\ell] +$$

$$U^a_{rj} [ir;k\ell] + U^a_{rk} [ij;k\ell] + U^a_{r\ell} [ij;k\ell]) \qquad (89)$$

where U^a_{ri} denotes the first order changes in the molecular orbitals.[62] To compute the energy gradients in this manner one needs, in addition to the information required in the ordinary CI, the derivatives of all of the atomic integrals $\langle\lambda|\hat{h}|\mu\rangle^a$ and $[\lambda\mu;\nu\sigma]^a$ as well as the U^a_{ri} coefficients.

It is true that the contribution of the atomic integral derivatives $[\lambda\mu;\nu\sigma]^a$ can be obtained simply by a four-index transformation, but this would require a separate (m^5) transformation process for each derivative under consideration. We will present two methods here to avoid this problem. The first method shows how an (m^5) transformation may be performed once and used for all of the different derivatives and the second method (given in section IV C.) demonstrates that much of even this one remaining (m^5) transformation can be avoided.

By combining equations (88) and (89) with equation (87) the energy gradient can be expressed in two terms as

$$E^{a\prime} = \sum_{\lambda\mu\nu\sigma} \sum_{ijk\ell} \sum_{IJ} C_I C_J b_{IJ}^{ijk\ell} \, c_{\lambda i} c_{\mu j} c_{\nu k} c_{\sigma \ell} \, [\lambda\mu;\nu\sigma]^a +$$

$$\sum_{\lambda\mu} \sum_{ij} \sum_{IJ} C_I C_J a_{IJ}^{ij} \, c_{\lambda i} c_{\mu j} \, \langle \lambda | \hat{h} | \mu \rangle^a \qquad (90)$$

and

$$E^{a\prime\prime} = \sum_{ijr} \sum_{IJ} C_I C_J a_{IJ}^{ij} \, (U_{ri}^a \, \langle r|\hat{h}|j\rangle + U_{ri}^a \, \langle i|\hat{h}|r\rangle + \sum_{ijk\ell r} \sum_{IJ} \qquad (91)$$

$$C_I C_J b_{IJ}^{ijk\ell} \, (U_{ri}^a \, [rj;k\ell] + U_{rj}^a \, [ir;k\ell] + U_{rk}^a \, [ij;r\ell] + U_{r\ell}^a \, [ij;kr])$$

where the total energy derivative is given by

$$E^a = E^{a\prime} + E^{a\prime\prime} \quad . \qquad (92)$$

The optimum method for evaluating these equations is by first computing the one- and two-particle density matrices[64] given respectively by

$$G_{ijk\ell} = \sum_{IJ} C_I C_J b_{IJ}^{ijk\ell} \qquad (93)$$

and

$$Q_{ij} = \sum_{IJ} C_I C_J a_{IJ}^{ij} \quad . \qquad (94)$$

The coupling coefficient $b_{IJ}^{ijk\ell}$ and a_{IJ}^{ij} can be found rapidly by using the loop-driven algorithm. The density matrices are used in two different ways to find both the gradient contributions of $E^{a\prime}$ and $E^{a\prime\prime}$.

To find the contribution $E^{a\prime}$ given by equation (90), the density matrices are transformed to the AO basis by

$$Q_{\lambda\mu} = \sum_{ij} c_{\lambda i} c_{\mu j} Q_{ij} \tag{95}$$

and by

$$G_{\lambda\mu\nu\sigma} = \sum_{ijk\ell} c_{\lambda i} c_{\mu j} c_{\nu k} c_{\sigma\ell} G_{ijk\ell} \quad . \tag{96}$$

This step will be of the order (m^5) and is similar to the four-index transformation, except that the transformation here is from the MO to the AO basis which requires less effort than the reverse process whenever orbitals are either deleted or treated as frozen core orbitals. By substituting equation (95) and (96) into (90), $E^{a,}$ reduces to

$$E^{a,} = \sum_{\lambda\mu\nu\sigma} G_{\lambda\mu\nu\sigma} [\lambda\mu;\nu\sigma]^a + \sum_{\lambda\mu} Q_{\lambda\mu} \langle\lambda|\hat{h}|\mu\rangle^a \quad . \tag{97}$$

The transformation of the density matrices is processed only once and the result is used for all of the different derivatives. Also, the integrals $[\lambda\mu;\nu\sigma]^a$ are never stored. As they are generated, its product with its corresponding density matrix element is immediately added to energy derivatives terms.

The derivative contribution $E^{a,,}$ given by equation (91) is found by first generating the Lagrangian X_{ir} given by

$$X_{ir} = \sum_{jk\ell} 4G_{ijk\ell}[rj;k\ell] + \sum_{j} 2Q_{ij} \langle r|\hat{h}|j\rangle \tag{98}$$

which is determined by an (m^5) process that is performed once. The effort required for this step is about one fourth of the effort required in a four-index transformation. This Lagrangian is an extremely

useful entity. When the density matrices are symmetric, that is

$$G_{ijk\ell} = G_{ij\ell k} = G_{jik\ell} = G_{ji\ell k} = G_{k\ell ij} = G_{k\ell ji} = G_{\ell kij} = G_{\ell kji} \tag{99}$$

and

$$Q_{ij} = Q_{ji} \tag{100}$$

then the Lagrangian given by equation (98) can greatly simplify equation (91) and becomes

$$E^{a\prime\prime} = \sum_{ir} X_{ir} U^{a}_{ri} \tag{101}$$

which is processed when the U^{a}_{ri} coefficients have been determined by the CPHF equations (see section IV F.). It should also be noted that this Lagrangian X_{ir} given by equation (98) has other uses, particularly when used in an iterative MCSCF scheme to find an optimum unitary transformation that will yield a "better" set of orbitals based on the CI wavefunction. Also, when the MCSCF wavefunction is converged, the Lagrangian will be symmetric (i.e. $X_{ir} = X_{ri}$). We are currently developing and implementing this MCSCF method, and we will present a more detailed description in section IV D.

The advantages of the above theoretical developments cannot be overestimated. Practical investigations[65,40] emphasize the enormous difficulty of locating transition states, let alone reaction paths, for organic molecules. The possibility of having accurate forces for CI wavefunctions will mean that advanced numerical techniques[66] can be applied to these forces and force constants to find reaction paths and transition states with a minimum of computational effort.

B. Generation of the One- and Two-Particle Density Matrix

The use of one- and two-particle density matrices allows the energy contribution to be found easily for all (3N) different derivatives, since the density matrices are independent of any particular derivative. In addition to determining CI energy derivatives with respect to nuclear position they can also be combined with any other set of integral derivatives to find other CI energy derivatives (e.g. electric field strength). The one- and two-particle density matrices are given by equations (93) and (94). The coupling coefficients $b_{IJ}^{ijk\ell}$ and a_{IJ}^{ij} are exactly those that are used in determining the CI energy, which were defined by loops. The simplest method for generating the density matrices is to generate these same loops. The only difference here is that once a loop has been generated it is processed differently. This allows the density matrix to be generated with the loop-driven algorithm. The effort required to compute the density matrices in this manner is only slightly longer than the effort required to generate the corresponding diagonalization tape.

In the diagonalization tape generation step, loop coefficients are combined with appropriate integrals to form an overall loop value. This loop value is then used a number of times, determined by the loop breakdown algorithm, for each diagonalization iteration. In generating the density matrices, this process is reversed. When a loop is generated, the loop breakdown algorithm is used first to determine the total loop contribution (d). This total loop contribution is simply a sum of the products of C_I and C_J for each separate loop contribution and is given by:

$$d = \sum_{j}^{\bar{x}_h} \sum_{k}^{x_t} C_{Y(z_h+m_\ell+k)+j} \cdot C_{Y(z_h+m'_\ell+k)+j} \qquad (102)$$

where m_ℓ and m'_ℓ are the weights of each branch of the loop, z_h is the primary upper walk weight, $\bar{x}_h$ and x_t are the number of upper and lower walks respectively, and Y is the indexing array. Once the total loop contribution has been determined, its product with the loop coefficients is added to correct density matrix element terms.

The density matrix elements here play the same role that the MO integrals play in the diagonalization tape generation step. These elements are stored in the same manner and a particular density matrix element can be found with the integral storage offset arrays by equation (59). Using the same storage method allows a block of density matrix elements to be computed simultaneously with the loop-driven algorithm.

C. Transformation and Partial Transformation of the Two-Particle Density Matrix

Since the density matrices are generated in the MO basis and the integral derivatives are created in the AO basis, either the integral derivatives, or the density matrices must be transformed to the other basis. If only one derivative (or degree of freedom) is investigated, then either choice is acceptable. On the other hand, when several sets of integral derivatives are to be computed, it is much more efficient to transform the density matrices to the AO basis as expressed by equations (95) and (96). The methods required to process these equations are almost identical to the methods for the four-index integral transformation step, with only minor modifications. One such modification is that the initial steps required for this transformation step, which include reading and sorting the MO one- and two-electron integrals, are identical to the initial steps of the methods required to generate the Lagrangian. Thus the computational methods for these rather distinct steps can be efficiently combined together.

Another important modification involves the storage structure of the final AO density matrix elements. Since the integral derivatives are computed using an adapted version of HONDO which generates integral derivatives with the method of Rys polynomials,[67] the density matrix elements must be stored by the same blocking structure that the integrals are generated by. This can be done by simply adding an additional sorting step after the transformation. A more efficient method involves producing the transformed AO density matrix elements in the sequence that they are needed. This avoids an extra major sorting step with only a modest increase in the complexity in the

transformation algorithm.

It is also possible to eliminate a large portion of this transformation given by equation (96). This advancement which describes an alternate method to compute the contribution of the all-external two-electron density matrix elements, is analogous to the avoidance of the full integral transformation in section III M. By computing the intermediates $F_W^{\lambda\nu}$, $F_W^{\lambda\nu'}$, $F_X^{\lambda\nu}$ as given by equations (74-76), the all-external contribution of the two-particle density matrix over AO's can be expressed as

$$G_{ext}^{\lambda\mu\nu\sigma} = \sum_{up_W} [(F_W^{\lambda\nu} + F_W^{\lambda\nu'})(F_W^{\mu\sigma} + F_W^{\mu\sigma'}) - \tfrac{1}{2} F_W^{\lambda\nu'} F_W^{\nu\sigma'}] + \sum_{up_X} F_X^{\lambda\nu} F_X^{\mu\sigma} \quad . \tag{103}$$

This step is of the order (n^2m^4) and could greatly reduce the effort required to obtain $E^{a'}$ given by equation (97).

Another potentially very useful approximation is the use of frozen core orbitals and deleted virtual orbitals. The approximation presented here treats the frozen core orbitals as they would act in a gradient procedure at the SCF level. In other words the contribution of $E^{a''}$ given by equation (101) is replaced for frozen core orbitals by

$$E_{core}^{a''} = \sum_{i=\text{frozen core}} \varepsilon_i S_{ii}^a \tag{104}$$

where ε_i is the orbital energy of orbital i and $\underset{\sim}{S}^a$ is the overlap derivative matrix. This is identical to the expression used for all doubly occupied orbitals in an SCF gradient procedure. The contribution of $E^{a''}$ is ignored for deleted virtual orbitals. The use of

this approximation implies that any MO density matrix elements that represent these orbitals need not be computed provided that these same orbitals were frozen core or deleted virtual orbitals within the CI calculation. This approximation can substantially reduce the (m^5) process required for both the density matrix transformation and the generation of the Lagrangian. It is important that the $E^{a'}$ term given by equation (97) is not altered by this approximation. Since the MO density matrix elements that represented frozen core orbitals were never computed, their contribution to the AO density matrix must be included directly. This can be accomplished by expressing the full AO density matrices as

$$G_{\lambda\mu\nu\sigma} = G'_{\lambda\mu\nu\sigma} + 2\,P_{\lambda\mu}P_{\nu\sigma} - \tfrac{1}{2}\,P_{\lambda\nu}P_{\mu\sigma} - \tfrac{1}{2}\,P_{\lambda\sigma}P_{\mu\nu} + P_{\lambda\mu}Q'_{\nu\sigma} + Q'_{\lambda\mu}P_{\nu\sigma}$$
$$- \tfrac{1}{4}\,P_{\lambda\nu}Q'_{\mu\sigma} - \tfrac{1}{4}\,Q'_{\lambda\nu}P_{\mu\sigma} - \tfrac{1}{4}\,P_{\lambda\sigma}Q'_{\mu\nu} - \tfrac{1}{4}\,Q'_{\lambda\sigma}P_{\mu\nu} \qquad (105)$$

and

$$Q_{\lambda\mu} = Q'_{\lambda\mu} + 2\,P_{\lambda\mu} \qquad (106)$$

where $\underset{\sim}{Q}'$ and $\underset{\sim}{G}'$ are the one- and two-particle density matrices in the AO basis constructued only from those orbitals that are included in the CI and $P_{\lambda\mu}$ is the frozen core density matrix defined as

$$P_{\lambda\mu} = \sum_{i=\text{frozen core}} c_{\lambda i}c_{\mu i} \quad . \qquad (107)$$

The usefulness of this approximation will depend on a number of factors. The main trade-off here is between computation efficiency and accuracy. It is hoped that the error introduced by this approximation will be less than one micro Hartree for most orbitals that can

be considered to be part of a frozen core. The capability to apply this approximation already exists within our CI gradient system.

D. Generation and Use of the Lagrangian Matrix

The initial step in generating the Lagrangian given by equation (98), is to process a sort on both the density matrix elements and the MO integrals. This sort is needed because the density matrix elements G_{ijab} for all ij for a fixed ab must be present with the MO integrals $[k\ell;ab]$ for all $k\ell$. The contribution to the Lagrangian for this fixed ab is then found by a simple matrix multiply

$$X_{ir}(ab) = \sum_{j} G_{ijab}[rj;ab] \quad . \tag{108}$$

The methods to process this matrix multiply are slightly modified because both G_{ijab} and $[k\ell;ab]$ are stored in lower triangular form. The procedure to generate the Lagrangian is combined with the procedure to transform the density matrices to eliminate an extra sorting step. The diagonal elements of the Lagrangian are roughly equivalent to the product of the orbital energy and the orbital occupancy of that orbital. The off-diagonal elements correspond to the interaction between two orbitals.

Once the Lagrangian is formed it is used to find the CI energy gradient contribution through equation (101). As mentioned earlier, it is also possible to use the Lagrangian to find the MCSCF wavefunction.[68] The CI energy can be expressed in a simple form as

$$E_{CI} = \sum_{ijk\ell} G_{ijk\ell}\,[ij;k\ell] + \sum_{ij} Q_{ij} \langle i|\hat{h}|j\rangle \quad . \tag{109}$$

If a unitary transformation was applied to the orbital basis to find a "better" set of orbitals, the CI energy could then be approximated by

$$E \approx \sum_{ijk\ell} \sum_{abcd} G_{ijk\ell} U_{ai} U_{bj} U_{ck} U_{d\ell} [ab;cd] + \sum_{ij} \sum_{ab} Q_{ij} U_{ai} U_{bj} \langle a|\hat{h}|b\rangle \quad . \quad (110)$$

This is only an approximation because the density matrices $\underset{\sim}{G}$ and $\underset{\sim}{Q}$ depend on the orbital basis in a complex manner. If the unitary transformation U was close to unity, then this approximation would be fairly good. This suggests an iterative procedure to find the MCSCF wavefunction. If for any CI calculation the transformation $\underset{\sim}{U}$ could be found that minimizes the energy through equation (110), then the CI calculation could be repeated in this new basis (i.e. compute the correct density matrix for this new basis). This procedure could then be repeated until the transformation matrix approaches unity, which should also yield the MCSCF wavefunction.

The problem then becomes to find a procedure that can determine the "best" unitary transformation each iteration within this procedure. By assuming that $\underset{\sim}{U}$ is fairly close to unity, $\underset{\sim}{U}$ can be represented as

$$\underset{\sim}{U} = \underset{\sim}{1} + \underset{\sim}{U}^1 \quad . \quad (111)$$

When this is substituted into equation (110) and all higher terms in $\underset{\sim}{U}^1$ are ignored, then the energy can be further approximated as

$$E \approx \sum_{ijk\ell} \sum_{abcd} G_{ijk\ell} [U^1_{ai} \delta_{bj} \delta_{ck} \delta_{d\ell} + \delta_{ai} U^1_{bj} \delta_{ck} \delta_{d\ell} + \delta_{ai} \delta_{bj} U^1_{ck} \delta_{d\ell}$$

$$+ \delta_{ai} \delta_{bj} \delta_{ck} U^1_{d\ell}] [ab;cd] + \sum_{ij} \sum_{ab} Q_{ij} [U^1_{ai} \delta_{bj} + \delta_{ij} U^1_{bj}] \langle a|\hat{h}|b\rangle \quad (112)$$

which can be simplified to

$$E \approx \sum_{ijk\ell r} G_{ijk\ell} (4\, U^1_{ir}) [rj;k\ell] + \sum_{ijr} Q_{ij} (2\, U^1_{ir}) \langle r|\hat{h}|j\rangle \quad . \quad (113)$$

Since the Lagrangian is defined as

$$X_{ir} = \sum_{ijk\ell} 4\, G_{ijk\ell}\, [rj;k\ell] + \sum_{j} 2\, Q_{ij}\, \langle r|\hat{h}|j\rangle \tag{114}$$

equation (112) can be further reduced to

$$E \approx \sum_{ir} X_{ir} U^{1}_{ir} = \sum_{ir} X_{ir} (U_{ir} - \delta_{ir}) \quad . \tag{115}$$

Minimizing the energy through equation (115) is equivalent to finding the unitary transformation U that minimizes

$$E' = \sum_{ir} X_{ir} U_{ir} \quad . \tag{116}$$

This can be done a number of ways. One such approach involves the use of a penalty function which blows up rapidly when U becomes non-unitary. This is done by solving a similar problem such as minimizing

$$E'_{P} = \sum_{ir} X_{ir} U_{ir} + P(\underset{\sim}{T}) \tag{117}$$

where P is the penalty function and $\underset{\sim}{T}$ is given by

$$\underset{\sim}{T} = \underset{\sim}{U}^{\dagger}\underset{\sim}{U} \quad . \tag{118}$$

One example of a penalty function is

$$P(\underset{\sim}{T}) = a\Big(\sum_{i} \exp[(1-T_{ii})^{2}] + \sum_{i\neq j} \exp[(T_{ij})^{2}] - N^{2}\Big) \tag{119}$$

where N is the dimension of T. This penalty function will be zero if U is unitary and be large whenever U deviates sufficiently from a unitary matrix. Equation (117) can now be solved as an unconstrained

problem. Another approximate method which is much simpler involves the direct diagonalization of a matrix constructed from the $\underset{\sim}{X}$ matrix that will give a good unitary transformation.

The computational effort required for each iteration of this MCSCF scheme will be roughly the same as with other MCSCF procedures (i.e. dominated by the integral transformation step). We have also implemented the capability to treat certain orbitals as part of a frozen core or as deleted virtuals by not allowing these orbitals to "mix" with the remaining orbital space in the MCSCF iterations. This approximation can substantially reduce the (m^5) integral transformation effort required each iteration.

The MCSCF procedure presented here allows the rigorous optimization of orbitals for any set of configurations that can be used in a CI calculation. These techniques can also be advanced for excited state calculations. We expect that MCSCF calculations involving 10000 configurations will become routine with our system.

E. Generation and Use of the Integral Derivatives

The integral derivatives are generated using an adapted version of the HONDO SCF gradient program.[67] The only portions of this program that have been modified are the section that interfaces the program to the rest of the program system and the sections of the program that process the computed integral derivatives. Each integral $\langle\lambda|h|\mu\rangle^a$ and $[\lambda\mu;\nu\sigma]^a$ are processed in two different ways to find the contribution to both $E^{a'}$ and $E^{a''}$ given by equations (97) and (101). The integral derivatives are first combined with the appropriate density matrix element and processed through

$$E^{a'} = \sum_{\lambda\mu} Q_{\lambda\mu}\langle\lambda|h|\mu\rangle^a + \sum_{\lambda\mu\nu\sigma} G_{\lambda\mu\nu\sigma}[\lambda\mu;\nu\sigma]^a \quad . \tag{120}$$

The integral derivatives are also required to process the coupled-perturbed Hartree-Fock equations. For a closed shell singlet calculation they are processed through

$$B^a_{\lambda\mu} = \langle\lambda|h|\mu\rangle^a + \sum_{\nu\sigma} P_{\nu\sigma}(2[\lambda\mu;\nu\sigma]^a - [\lambda\nu;\mu\sigma]^a) \tag{121}$$

which is simply a Fock matrix constructed from integral derivatives instead of ordinary integrals. For other types of SCF reference states more than one of these Fock matrices constructed from integral derivatives will be required. In any case, there must be at least one such matrix for each of the (3N) nuclear perturbation under consideration. Since all of these matrices must be maintained within internal storage (core), this applies a modest limit to the size of our calculations. The use of these Fock matrices constructed from integral derivatives will be described in the next section.

F. Coupled Perturbed Hartree-Fock

The CPHF equations are solved to find the energy derivative contribution of $E^{a,x}$ given by equation (101). The first order changes in the molecular orbitals U^a_{ir} are given by Gerratt and Mills[62] for closed shell RHF as

$$U^a_{rs} = (F^a_{rs} - \varepsilon_s S^a_{rs})/(\varepsilon_s - \varepsilon_r) \tag{122}$$

where ε_r is the orbital energy of orbital r, $\underset{\sim}{S}^a$ is the derivative of the overlap matrix, and $\underset{\sim}{F}^a$ is the derivative of the Fock matrix given by

$$F^a_{rs} = \langle r|h|s\rangle^a + \sum_k^{occ} (2[rs;kk]^a - [rk;sk]^a)$$

$$+ \sum_k^{occ} \sum_b^{virt} (4[rs;kb] - [rk;sb] - [rb;sk])\, U^a_{kb}$$

$$- \sum_k^{occ} \sum_\ell^{occ} S^a_{k\ell}\, (2[rs;k\ell] - [rk;s\ell]) \quad . \tag{123}$$

The difficulty with this representation is that $\underset{\sim}{U}$ appears on both sides of this equation. Straightforward methods to solve this equation have been applied by Pople and coworkers.[41] We find it more useful to rewrite equation (101) in a symmetric and antisymmetric contribution

$$E^{a,x} = \sum_{ir} X^+_{ir}(-S^a_{ir}/2) + \sum_{ir} X^-_{ir} U^{a-}_{ir} \tag{124}$$

where

$$X^+_{ir} = (X_{ir} + X_{ri})/2 \quad , \tag{125}$$

$$X_{ir} = (X_{ir} - X_{ri})/2 \quad , \tag{126}$$

$$U^{a-}_{ir} = (U^{a}_{ir} - U^{a}_{ri})/2 \quad , \tag{127}$$

and

$$U^{a+}_{ir} = (U^{a}_{ir} + U^{a}_{ri})/2 = -S^{a}_{ir}/2 \quad . \tag{128}$$

The second part of equation (128) is given since U^{a}_{ir} is constrained such that

$$U^{a}_{ir} + U^{a}_{ri} + S^{a}_{ir} = 0 \quad . \tag{129}$$

Rewriting equation (122) in this new basis we find that U^{a-}_{rs} is given by

$$U^{a-}_{rs} = [F^{a}_{rs} - (\varepsilon_s + \varepsilon_r)(S^{a}_{rs}/2)]/(\varepsilon_s - \varepsilon_r) \quad . \tag{130}$$

For this representation F^{a}_{rs} is given by equation (123), but in the new representation it is better expressed in a form dependent on $\underset{\sim}{U}^{a-}$ as

$$F^{a}_{rs} = B^{a}_{rs} + \sum_{k}^{occ} \sum_{b}^{virt} (4[rs;kb] - [rk;sb] - [rb;sk])\, U^{a-}_{kb}$$

$$- \sum_{k}^{occ} \sum_{t}^{all} (4[rs;kt] - [rk;st] - [rt;sk])(S^{a}_{tk}/2) \tag{131}$$

where B^{a}_{rs} is obtained by the transformation

$$B^{a}_{rs} = \sum_{\lambda\mu} c_{\lambda r} c_{\mu s} B^{a}_{\lambda\mu} \tag{132}$$

and $B^{a}_{\lambda\mu}$ is given by equation (121). The $\underset{\sim}{U}^{a+}$ contribution has been included in the last term.

As demonstrated by Pople,[41] $\underset{\sim}{U}^{a-}$ can be found by solving the coupled equations

$$(\underset{\approx}{1}-\underset{\approx}{A})\underset{\sim}{U}^{a-} = \underset{\sim}{U}^{a}_{0} \tag{133}$$

where

$$A_{(ai,bj)} = (4[ai;bj] - [aj;bi] - [ab;ij])/(\varepsilon_i - \varepsilon_a) \tag{134}$$

and

$$U^{a}_{0(ai)} = [\langle a|h|i\rangle^{a} - \sum_{k}^{occ}(2[ai;kk]^{a} - [ak;ik]^{a}) - \frac{(\varepsilon_i - \varepsilon_a)}{2} S^{a}_{ai} - \sum_{k}^{occ}\sum_{t}^{all}(\frac{S^{a}_{tk}}{2})(4[ai;tk] - [ak-ti] - [at;ik])]/(\varepsilon_i - \varepsilon_a) \quad . \tag{135}$$

The matrix $\underset{\approx}{A}$ is a non-symmetrical square matrix that has the dimensions of the product of the number of occupied orbitals (excluding frozen core orbitals used with the frozen core approximation) and the number of virtual orbitals (again excluding the deleted virtuals). $\underset{\sim}{U}^{a-}$ and $\underset{\sim}{U}^{a-}$ are column vectors of the same dimension. This equation must be solved for each different derivative under consideration.

Once equation (133) has been solved the complete $\underset{\sim}{U}^{a-}$ matrix can be found by equations (130) and (131), except that we are not really interested in computing the $\underset{\sim}{U}^{a-}$ matrix, but only its contribution to the energy gradient given by equation (124). The total contribution to the energy gradient is found in a straightforward manner for all terms except the contribution derived from the second term of equation (131). This is the only term which requires the solution of the CPHF equations and its contribution can be expressed as

$$E^{a\,\prime\prime\prime\prime} = 2 \sum_{r>s} [X^{-}_{rs}/(\varepsilon_s - \varepsilon_r][\sum_{k}^{occ} \sum_{b}^{virt} U^{a-}_{bk}(4[rs;bk]$$

$$- [rk;bs] - [rb;sk])] \qquad (136)$$

which is treated separately, thus avoiding the explicit calculation of $\underset{\sim}{U}^{a-}$ for the parts that do not connect the occupied and virtual spaces.

Pople[41] presented an innovative iterative method to solve the coupled equations given by equation (133). His method has many similarities to the Davidson diagonalization method,[56] except that the method solves a set of simultaneous equations. (In fact, to implement this method we simply made minor modifications to our diagonalization routine.) For each nuclear variable he defines an increasing set of orthogonal vectors given by

$$\underset{\sim}{B}^{a}_{0} = \underset{\sim}{U}^{a-}_{0} \qquad (137)$$

and

$$\underset{\sim}{B}^{a}_{n+1} = \underset{\approx}{A}\underset{\sim}{B}_{n} - \sum_{\ell=0}^{n} \frac{\langle \underset{\sim}{B}_{\ell} | \underset{\approx}{A} | \underset{\sim}{B}_{n} \rangle}{\langle \underset{\sim}{B}_{\ell} | \underset{\sim}{B}_{\ell} \rangle} \underset{\sim}{B}_{\ell} \quad , \qquad (138)$$

where n increases by one each iteration. The solution is approximated by solving a much smaller set of simultaneous equations given by

$$\langle \underset{\sim}{B}_{k} | \underset{\approx}{1}-\underset{\approx}{A} | \underset{\sim}{B}_{\ell} \rangle \, \alpha_{\ell} = \langle \underset{\sim}{B}_{k} | \underset{\sim}{B}_{0} \rangle \qquad (139)$$

for ℓ and k=0,n and then

$$\underset{\sim}{U}^{a-}_{n} \approx \sum_{\ell=0}^{n} \alpha_{\ell} \underset{\sim}{B}_{\ell} \qquad (140)$$

The time-consuming step for this procedure is the effort required to

multiply the matrix $\underset{\approx}{A}$ by the vector $\underset{\sim}{B}_n$ each iteration.

This concept can be improved even further by considering that the different solutions to equation (133) for each different derivative will not be orthogonal. Thus, by approximating each solution with the same set of orthogonal vectors it is possible to reduce the computation effort for this step. The savings can then be found in that the number of matrix multiplies per iteration can be substantially reduced (typically half of the number of nuclear variables) with only a modest increase in the number of iterations. This is done by defining an orthonormal set of n vectors $\underset{\sim}{b}_i$ and vector products $\underset{\sim}{\sigma}_i$ given by

$$\underset{\sim}{\sigma}_i = (\underset{\approx}{1}-\underset{\approx}{A})\underset{\sim}{b}_i \quad . \tag{141}$$

For each nuclear variable, solve a small set of coupled equations

$$\langle \underset{\sim}{b}_k | \underset{\sim}{\sigma}_\ell \rangle \, \alpha_\ell^a = \langle \underset{\sim}{b}_k | \underset{\sim}{U}^{a-} \rangle \tag{142}$$

for k and ℓ=1,n. The approximate solution to equation (133) is given by

$$\underset{\sim}{U}^{a-} \approx \sum_{\ell=1}^{n} \alpha_\ell^a \, \underset{\sim}{b}_\ell \tag{143}$$

and the correction vector to improve convergence is given by

$$\underset{\sim}{b}_n^{a'} = \underset{\sim}{b}_n^{a''} - \sum_{\ell=1}^{n} \langle \underset{\sim}{b}_\ell | \underset{\sim}{b}_n^{a''} \rangle \, \underset{\sim}{b}_\ell \tag{144}$$

where

$$\underset{\sim}{b}_n^{a''} = \sum_{\ell=0} \alpha_\ell^a \, \underset{\sim}{\sigma}_\ell \quad . \tag{145}$$

This will yield a set of (3N) new correction vectors which are neither normalized nor orthogonal to each other. From this set, approximately

(3N/2) orthonormal vectors are produced and added to the set of n vectors. These vectors, which are linear combinations of the $\underset{\sim}{b}_n^{a'}$ vectors, are chosen so as to most improve the nuclear degrees of freedom which had poorer convergence (i.e. where $\langle \underset{\sim}{b}_n^{a'} | \underset{\sim}{b}_n^{a'} \rangle$ is large), thus after several such iterations all (3N) solutions will be equally converged. The initial orthonormal set of $\underset{\sim}{b}_i$ vectors is obtained by orthogonalizing and normalizing the $\underset{\sim}{U}_0^{a-}$ vectors.

V. GRAPHICAL UNITARY GROUP APPROACH CI AND CI ENERGY GRADIENTS AS A SYSTEM OF PROGRAMS

The methods and algorithms described in the earlier sections have been combined into an effective system of programs. The system is referred to as the Berkeley Unitary Group Approach Configuration Interaction (BUGACI) system of programs. Figure 6 shows a simplified schematic of the information flow within this system.

The input to this system (other than direct user input) is a set of symmetry orbital (SO) integrals, and a file that contains the SCF transformation matrix and other orbital information. The program DRTGEN generates all of the distinct row table arrays as well as the integral sorting arrays. This is the only program in the system that requires any appreciable user supplied input (cards) beyond simple option parameters. The DRT file is used by many programs in the system.

The program TRANS performs the integral transformation. The algorithm used in this transformation program differs somewhat from conventional approaches in that extra sorting steps have been eliminated, integrals are always stored with implicit addressing, and the effective working space is doubled using a scratch file as virtual memory. The "frozen core" contribution is included through the use of a frozen core Fock matrix as a one-electron effective Hamiltonian. One current drawback to this program is that symmetry is not included explicitly.

The program NEWCORE is used to increase the number of frozen core orbitals by including the new frozen orbitals into the effective Hamiltonian. Since this process results in a loss of information, this program can only be used to increase the number of frozen core

orbitals.

The program BUGME generates the diagonalization tape with the loop-driven algorithm. This program has the capability of generating three sequential tapes; one for diagonal loops, one for off-diagonal loops, and the last one is used to store only "important" off-diagonal loops. The diagonalization program (DIAG) can iterate rapidly reading only the diagonal elements and the important off-diagonal loops and when convergence is near, the program can switch to the full off-diagonal loops tape. The program DIRECT avoids the storage of the diagonalization tape, and constructs all of the loops each diagonalization iteration.

After the CI has been completed, the program DENSITY can obtain the one-particle density matrix which can be used to obtain CI properties (CIPROPS), or used to obtain a set of natural orbitals. This set of natural orbitals can be used to transform the MO integral list to a new basis (the original integral tape is always used to eliminate excessive error accumulation) with the program MOTRANS. This program is almost identical to TRANS and is used for natural orbital iterations in this way.

The program TWOPDM generates the one- and two-particle density matrices from the CI vector. These can be used to find an optimum orbital transformation matrix with the program SUPERCI. By transforming the integrals to this optimum basis and iterating, the MCSCF wavefunction can be found.

The one- and two-particle density matrices can also be used to obtain CI energy gradients with respect to all nuclear perturbations. This is done by transforming the MO density matrices to the AO basis

with the program TRANSDM, and combining them with the integral derivatives generated by the program CIDER. The program CIDER also computes the integral derivative Fock matrices required to solve the CPHF equations which are processed in the program CICPHF. The program CICPHF combines the separate energy derivative contributions and produces the final result. The gradient system in its present form is restricted to closed shell RHF wavefunctions, or converged MCSCF wavefunctions.

This system of programs has not been completed at the time of this writing, and will undoubtedly undergo substantial modifications, improvements, and additions in the future. It is expected (and hoped) that this system of programs will be maintained and used for a number of years.

VI. APPLICATIONS

A. Examples and Timing Results

One of the more important aspects to this work is the reporting of a series of examples including timing breakdowns and comparisons between the current version and the preliminary version of this method, as well as with other CI methods. Also reported, in addition to timing results, are some statistical data on generated loops. All timing resuls are given in minutes on the Harris 6024/4 (about a factor of 27 slower than the CDC 7600).

Since our preliminary implementations, we have more than doubled the efficiency of the programs (excluding the diagonalization timings and integral transformation). There are a great number of factors that contribute to this increase in efficiency. The more important of these are: 1) the use of the new segment coefficients, 2) reduction in searching for acceptable segments due to the use of new coefficients and an additional test to determine if enough electrons remain in lower orbitals, 3) elimination of the buffer packing and unpacking required in the formula tape version, 4) elimination of coding loop coefficients and the associated packing operations, 5) use of the implicit treatment of the all-external loops, and 6) an assortment of other minor improvements in many other areas.

The first example involves the closed shell ground state of BH_3 in C_{2v} symmetry. It is a small calculation and reported here primarily for the comparison of CI methods as well as detailed timing breakdown of all aspects of the diagonalization tape generation depicted in Figure 5. Although this is a rather small calculation, it is not expected that the relative proportions of the different sections will

change substantially as one goes to larger calculations. The second example involves the $^1A_1$ state of CH_2 with a fairly extensive basis set. This example gives a direct comparison of the integral driven CI method[9] and the GUGA method for a moderately large calculation. The third example is a calculation of the $^1B_{1u}$ (V state) of ethylene. Example three consists of the Hartree-Fock interacting configuration space from three open-shell singlet reference configurations with a double zeta plus polarization diffuse functions basis set.

The fourth example represents a rather large calculation on the $2^1A'$ state of SO_2 at an unsymmetrical geometry (C_s symmetry). For this calculation the interacting space from three reference configurations was included after 7 core orbitals and the 7 highest virtuals were deleted. For the basis set used (double zeta plus polarization functions on the sulfur atom) 23,613 spin-adapted Gelfand states were required. Since only 20K real words are available to the user on the Harris minicomputer, the large diagonalization techniques described in section III L. were required.

These examples represent only a small subset of the types of calculations possible with the GUGA method as currently implemented.

The timing comparisons given in Examples 1-3 indicate definite trends. One obvious trend is that as the problem size gets larger the GUGA becomes increasingly superior to the other methods. This increase in efficiency can be traced to an increase in the size of the secular equation (as opposed to an increase in the number of molecular orbitals). This increase in efficiency can also be understood very easily if the loop statistics are compared. As the number

of configurations increase, the average number of total contributions for each loop increases. Each of these contributions must be dealt with separately for the conventional CI method and the integral driven CI method. For the GUGA method the time required depends heavily on the number of loops generated, not the number of contributions per loop. For this same reason, calculations with little or no spatial symmetry will enjoy an even greater efficiency. Another trend that should be mentioned is that the GUGA method becomes more efficient compared with the other CI methods, for a given number of configurations, as the number of electrons included in the CI increases.

The loop statistics given in Table V for each case separate the diagonal loops, those loops that contribute to diagonal matrix elements, from the off-diagonal loops. Since the diagonal loops are only processed once for a given calculation and the off-diagonal loops are processed each iteration, it is the off-diagonal loops that determine the computational effort required for the diagonalization. For the lower walks, two values are reported; the average number of lower walks and the percentage of loops with only one lower walk. This is done for upper walks, as well as for the total number of loop contributions. The large percentage of loops with total contributions of unity suggests that an alternate method of storing and processing these loops is worth considering.

B. Vertical Electronic Spectrum of Ketene

Three years ago Dykstra and Schaefer reported a careful SCF study of the low-lying electronic states of ketene[69] (C_2H_2O). However, no consideration was given to the effects of electron correlation. Using the present GUGA method, it is possible to reliably evaluate correlation effects in little more than the time required for the original SCF studies.

The standard Huzinaga-Dunning contracted Gaussian double zeta basis set was adopted,[70] as in the previous work of Dykstra and Schaefer and that of Harding and Goddard.[71] With the three 1s-like orbitals frozen (doubly occupied in all configurations), CI was carried out including all single and double excitations. Only the vertical (ground state equilibrium) geometry was considered, and our study was restricted to the lowest $^1A_1$, $^3A_2$, $^1A_2$, and $^3A_1$ electronic states.

The numbers of spin and space adapted configurations included were 5089, 6300, 6234, and 5981, respectively. For the largest calculation, the $^3A_2$ state, integral computation required 19 minutes, 20 SCF iterations required 29 minutes, the integral transformation 9 minutes, and the unitary CI a total of 54 minutes. Thus we see that the time for the CI step is comparable to that for the relatively routine SCF calculation. Since the formula tape required 23 mintues, further CI calculations on the $^3A_2$ potential energy surface would require only 40 minutes each, including the four-index transformation of two-electron integrals.

The SCF and CI total energies obtained here are -151.67244 and

-151.94659 ($^1A_1$), -151.56305 and -151.81489 ($^3A_2$), -151.55581 and -151.81331 ($^1A_2$), -151.49968 and -151.74846 Hartree ($^3A_1$). The best previous variational calculations on ketene were those of Harding and Goddard, who report a CI ground state energy of -151.8271 Hartree, or about 0.12 Hartree above the present unitary CI result.

Relative energies are presented in Table VI and compared with those of Dykstra and Schaefer, Harding and Goddard, and the experimental electron impact results of Frueholz, Flicker, and Kuppermann.[72] Our results provide very strong support for the theoretical predictions of Harding and Goddard,[71] the largest difference being 0.06 eV. In addition the agreement with the electron impact experiments is as good (±0.1 eV) as could reasonably be expected at this level of theory.

VII. CONCLUDING REMARKS

This work has demonstrated that the loop-driven graphical unitary group approach is superior to state-of-the-art conventional configuration interaction (CI) techniques for the description of electron correlation in molecules. Further developments described herein yield an additional factor of two improvement in the efficiency of this new method when compared with the preliminary implementation. Possibly even more important is the formulation of a method for a analytical evaluation of the gradient of the potential energy. The latter method allows the evaluation of all first derivatives of the potential energy with an amount of effort not more than 2-3 times that required to obtain a single CI energy on the potential surface.

Some of the material contained in this work has been presented previously.[73,74]

REFERENCES

1. G. Das and A.C. Wahl, Phys. Rev. Lett. 24, 440 (1970).
2. C.F. Bender, S.V. O'Neil, P.K. Pearson, and H.F. Schaefer, Science 176, 1412 (1972).
3. G.C. Lie, J. Hinze, and B. Liu, J. Chem. Phys. 59, 1872 (1973).
4. I. Shavitt, "The Method of Configuration Interaction," Modern Theoretical Chemistry, Vol. 3, edited by H.F. Schaefer (Plenum, New York, 1977), pp. 189-275.
5. W. Meyer, Int. J. Quantum Chem. 55, 341 (1971).
6. R. Ahlrichs, H. Lischka, V. Staemmler, and W. Kutzelnigg, J. Chem. Phys. 62, 1225 (1975).
7. B. Roos, Chem. Phys. Lett. 15, 153 (1972).
8. R.F. Hausman, S.D. Bloom, and C.F. Bender, Chem. Phys. Lett. 32, 483 (1975).
9. R.R. Lucchese and H.F. Schaefer, J. Chem. Phys. 68, 769 (1978).
10. C.E. Dykstra, H.F. Schaefer, and W. Meyer, J. Chem. Phys. 65, 2740 (1976).
11. P.S. Bagus, B. Liu, A.D. McLean, and M. Yoshimine, Wave Mechanics: The First Fifty Years, edited by W.C. Price, S.S. Chissick and T. Ravensdale (Butterworth, London, 1973), pp. 99-118.
12. R.J. Buenker and S.D. Peyerimhoff, Theor. Chim. Acta 35, 33 (1974).
13. M.F. Guest and W.R. Rodwell, Mol. Phys. 32, 1075 (1976); J. Kendrick and I.H. Hillier, Mol. Phys. 33, 635 (1977).
14. L.E. McMurchie and E.R. Davidson, J. Chem. Phys. 66, 2959 (1977).
15. L.B. Harding and W.A. Goddard, J. Chem. Phys. 67, 1777 (1977).
16. P.J. Hay and T.H. Dunning, J. Chem. Phys. 67, 2290 (1977).

17. C.W. Bauschlicher and I. Shavitt, J. Am. Chem. Soc. 100, 739 (1978).

18. R. Seeger, R. Krishnan, and J.A. Pople, J. Chem. Phys. 68, 2519 (1978).

19. B.R. Brooks and H.F. Schaefer, Int. J. Quantum Chem. 14, 603 (1978).

20. C.F. Bender and H.F. Schaefer, J. Am. Chem. Soc. 92, 4984 (1970).

21. M. Moshinsky, in Group Theory and the Many-Body Problem (Gordon and Breach, New York, 1968).

22. I.M. Gelfand and M.L. Zetlin, Dokl. Akad. Nauk SSSR 71, 825, 1017 (1950).

23. J. Paldus, J. Chem. Phys. 61, 5321 (1974); Int. J. Quantum Chem. Symp. 9, 165 (1975); Phys. Rev. A 14, 1620 (1976).

24. (a) I. Shavitt, Int. J. Quantum Chem. Symp. 11, 131 (1977); (b) 12, 5 (1978).

25. I. Shavitt, "A Short Course on Matrix Element Evaluation Using the Unitary Group," Theoretical Atomic and Molecular Physics Group, Lawrence Livermore Laboratory, May 30 - June 2, 1978.

26. M. Kotani, A. Ameniya, E. Ishiguro, and T. Kimura, Tables of Molecular Integrals (Maruzen, Tokyo, 1955).

27. P.O. Löwdin, Rev. Mod. Phys. 36, 966 (1964).

28. H.F. Schaefer and F.E. Harris, J. Comput. Phys. 3, 217 (1968).

29. A. Bunge, J. Chem. Phys. 53, 20 (1970).

30. R.R. Lucchese, B.R. Brooks, J.H. Meadows, W.C. Swope, and H.F. Schaefer, J. Comput. Phys. 26, 243 (1978).

31. See, for example, S.F. Boys, Proc. Roy. Soc. (London) A217, 325 (1953).

32. P. Pulay, pages 152-185 of Volume 4, Modern Theoretical Chemistry, edited by H.F. Schaefer (Plenum, New York, 1977).

33. P. Pulay, Mol. Phys. 17, 197 (1969); 18, 473 (1970).

34. D. Poppinger, Chem. Phys. Lett. 34, 332 (1975); 35, 550 (1975).

35. H.B. Schlegel, S. Wolfe, and F. Bernardi, J. Chem. Phys. 63, 3632 (1975).

36. A. Kormornicki, K. Ishida, K. Morokuma, R. Ditchfield, and M. Conrad, Chem. Phys. Lett. 45, 595 (1977).

37. M. Dupuis and H.F. King, J. Chem. Phys. 68, 3998 (1978).

38. Pople, J.A. Seeger, R., and Krishnan, R., Intern. J. Quantum Chem. 11, 149 (1977).

39. J.S. Binkley, J.A. Pople, and W.J. Hehre, to be published.

40. J.D. Goddard, N.C. Handy, and H.F. Schaefer, to be published.

41. J.A. Pople, invited lecture at March 1979 Sanibel Symposium, Palm Coast, Florida.

42. J. Gerratt and I. Mills, J. Chem. Phys. 49, 1719 (1968).

43. J.-F. Gouyet, R. Schranner, and T.H. Seligman, J. Phys. A 8, 285 (1975).

44. G.W.F. Drake and M. Schlesinger, Phys. Rev. A 15, 1990 (1977).

45. J. Paldus, Physica Scripta 00, 0000 (1979).

46. I. Shavitt, Annual Report on "New Methods in Computational Quantum Chemistry and their Application on Modern Super-Computers" to NASA Ames Research Center, June 29, 1979.

47. C.F. Bender and H.F. Schaefer, J. Chem. Phys. 55, 4798 (1971); R.R. Lucchese and H.F. Schaefer, J. Amer. Chem. Soc. 99, 6766 (1977).

48. B.O. Roos and P.E.M. Siegbahn, Int. J. Quantum Chem. (1978).

49. I. Shavitt, Chem. Phys. Lett. 00, 0000 (1979).

50. J. Paldus, Theoretical Chemistry: Advances and Perspectives 2, 131 (1976).

51. D.E. Knuth, "Sorting and Searching," The Art of Computer Programming, Vol. 3 (Addison-Wesley, Reading, MA, 1973).

52. M. Yoshimine, J. Comput. Phys. 11, 449 (1973).

53. P. Siegbahn, J. Chem. Phys. 70, 0000 (1979).

54. I. Shavitt, private communication.

55. P.K. Pearson, R.R. Lucchese, W.H. Miller, and H.F. Schaefer, pages 171-190 of Minicomputers and Large Scale Computations, Symposium Series 57, editor P. Lykos (American Chemical Society, Washington, D.C., 1977).

56. E.R. Davidson, J. Computational Phys. 17, 87 (1975).

57. See for example, G.H.F. Diercksen, W.P. Kraemer, and B.O. Roos, Theoret. Chim. Acta 36, 249 (1975).

58. J. Paldus, in Theoretical Chemistry: Advances and Perspectives, Vol. 2, edited by H. Eyring and D.J. Henderson (Academic, New York, 1976), p. 131.

59. W. Meyer, J. Chem. Phys. 64, 2901, (1976).

60. C.E. Dykstra, H.F. Schaefer, and W. Meyer, J. Chem. Phys. 65, 2740 (1976).

61. At the SCF level of theory, such gradient procedures are well established. See P. Pulay, pages 152-185 of Volume 4, Modern Theoretical Chemistry, editor H.F. Schaefer (Plenum, New York, 1977).

62. J. Gerratt and I. Mills, J. Chem. Phys. 49, 1719 (1968).

63. B. Roos, Chem. Phys. Lett. 15, 153 (1972).

64. B.T. Sutcliffe, Methods of Molecular Quantum Mechanics (Academic Press, London, 1969).

65. J.A. Pople, R. Krishnan, H.B. Schlegel, and J.S. Binkley, Intern. J. Quantum Chem. 14, 545 (1978).

66. M.C. Flanigan, A. Komornicki, and J.W. McIver, pages 1-47 of Volume 8, Modern Theoretical Chemistry, editor G.A. Segal (Plenum, New York, 1977).

67. M. Dupuis and H.F. King, J. Chem. Phys. 68, 3998 (1978).

68. J. Hinze, J. Chem. Phys. 59, 6424 (1973).

69. C.E. Dykstra and H.F. Schaefer, J. Am. Chem. Soc. 98, 2689 (1976).

70. S. Huzinaga, J. Chem. Phys. 42, 1293 (1965); T.H. Dunning, J. Chem. Phys. 53, 2823 (1979).

71. L.B. Harding and W.A. Goddard, J. Am. Chem. Soc. 98, 6093 (1976).

72. R.P. Frueholz, W.M. Flicker, and A. Kuppermann, Chem. Phys. Lett. 38, 57 (1976).

73. B.R. Brooks and H.F. Schaefer, J. Chem. Phys. 70, 5092 (1979).

74. B.R. Brooks, W.D. Laidig, P. Saxe, N.C. Handy, and H.F. Schaefer, To be published in the Proceedings of Nobel Symposium 46 (1979).

FIGURE CAPTIONS

Figure 1. A sample Shavitt graph that describes the full CI configuration space for a system with 3 electrons and 4 orbitals.

Figure 2. A sample one electron loop that depicts the contribution of the integral $(i|\hat{h}|j)$ to the matrix element $\langle m'|H|m\rangle$.

Figure 3. A simplified schematic of all 14 loop types (and subtypes). Symbols R, L, and W refer to raising, lowering, and weight generators respectively. The solid lines refer to the m branch of the loop and the dashed lines refer to the m' branch. Intermediate lines are partially indicated to show their connections by dots where significant, even though they are not used in the new coefficient representation. Loop types 9, 10a, and 13a are restricted to avoid contributions to $\langle m'|H|m\rangle$ where the lexical order of m'>m.

Figure 4. Four sample loops that share a common loop head and depict the effectiveness of the loop-driven algorithm. Once any loop has been generated (solid arcs), only the differing portions of additional loops (dashed arcs) need to be processed.

Figure 5. A detailed timing breakdown in minutes of the major sections of the current verion of the GUGA method for Example I. Also included is the time required to perform the step $\underset{\sim}{\sigma} = \underset{\sim\sim}{H}\underset{\sim}{c}$ each iteration of the diagonalization.

Figure 1.

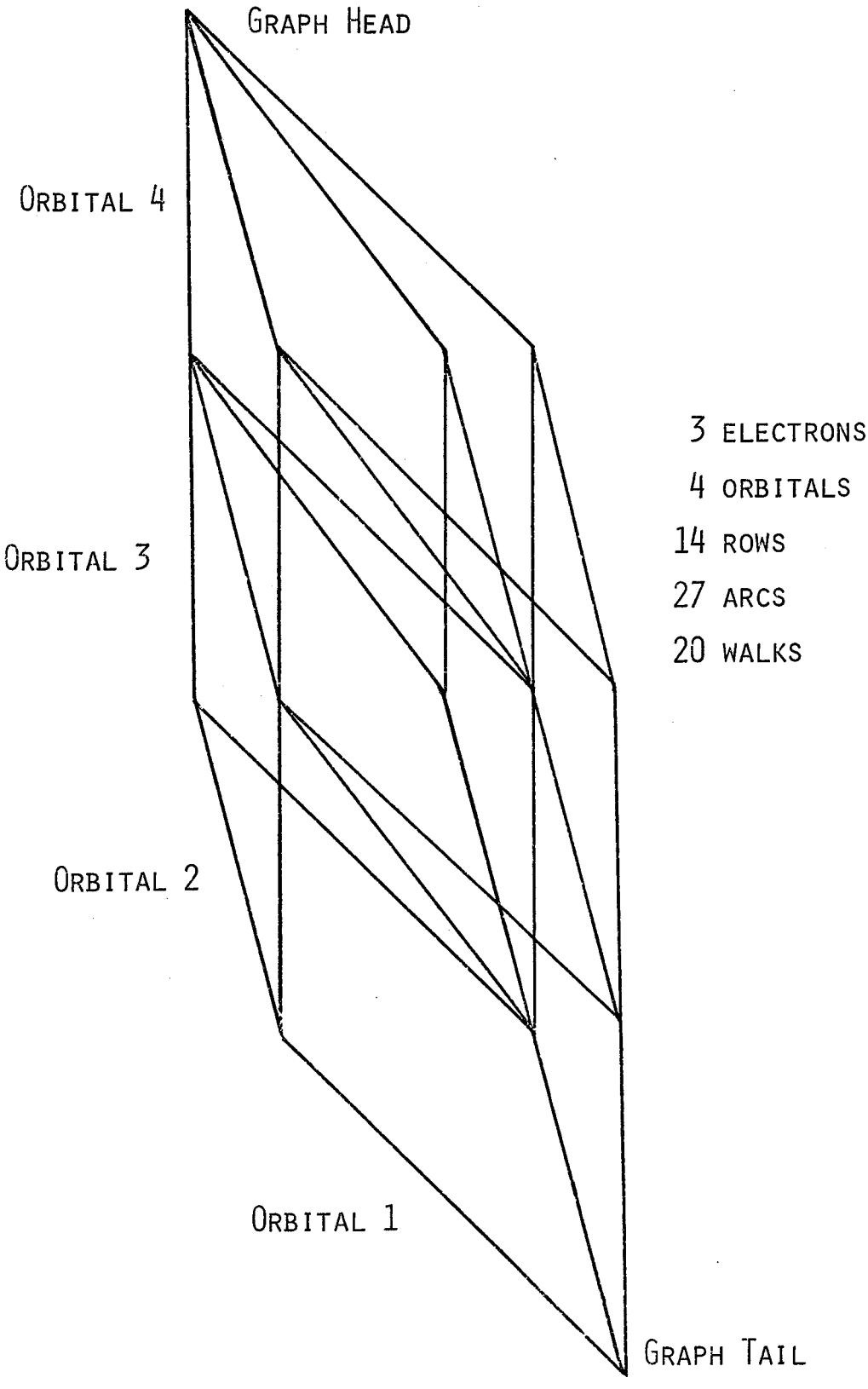

GRAPH HEAD
ORBITAL 4
ORBITAL 3
ORBITAL 2
ORBITAL 1
GRAPH TAIL
3 ELECTRONS
4 ORBITALS
14 ROWS
27 ARCS
20 WALKS

Figure 2.

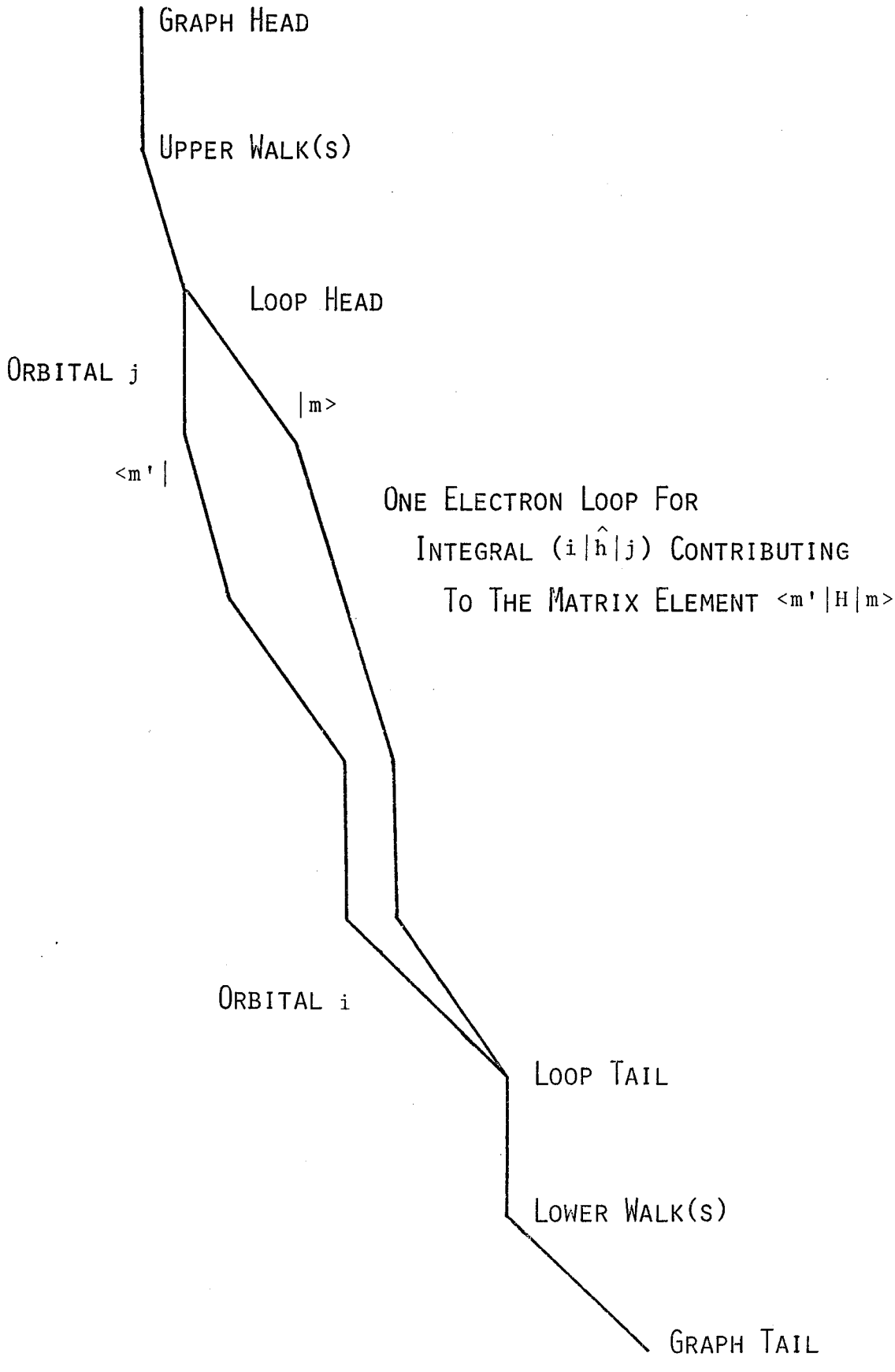

GRAPH HEAD
UPPER WALK(S)
LOOP HEAD
ORBITAL j
|m>
<m'|
ONE ELECTRON LOOP FOR
INTEGRAL (i|ĥ|j) CONTRIBUTING
TO THE MATRIX ELEMENT <m'|H|m>
ORBITAL i
LOOP TAIL
LOWER WALK(S)
GRAPH TAIL

Figure 3.

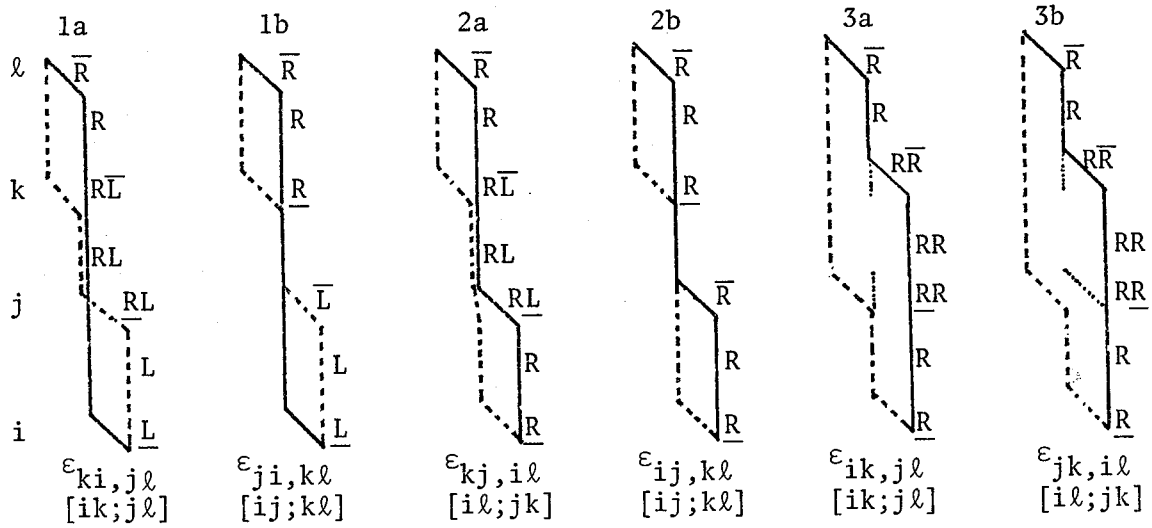

1a
1b
2a
2b
3a
3b
ℓ
k
j
i
R̄
R
RL̄
RL
L
L̄
RW
RR̄
RR
W
$\varepsilon_{ki,j\ell}$ [ik;jℓ]
$\varepsilon_{ji,k\ell}$ [ij;kℓ]
$\varepsilon_{kj,i\ell}$ [iℓ;jk]
$\varepsilon_{ij,k\ell}$ [ij;kℓ]
$\varepsilon_{ik,j\ell}$ [ik;jℓ]
$\varepsilon_{jk,i\ell}$ [iℓ;jk]
4
5a
5b
6
7a
7b
$\varepsilon_{ji,j\ell}$ [ij;jℓ]
$\varepsilon_{i\ell,ij}$ [ij;jℓ]
$\varepsilon_{jj,i\ell}$ [jj;iℓ]
$\varepsilon_{ik,i\ell}$ [ik;iℓ]
$\varepsilon_{i\ell,ki}$ [ik;iℓ]
$\varepsilon_{ii,k\ell}$ [ii;kℓ]
8
9
10a
10b
13a
13b
$\varepsilon_{i\ell,j\ell}$ [iℓ;jℓ]
$\varepsilon_{\ell i,j\ell}$ [iℓ;jℓ]
$\varepsilon_{\ell j,i\ell}$ [iℓ;jℓ]
$\varepsilon_{\ell\ell,ij}$ [ij;ℓℓ]
$\varepsilon_{\ell i,i\ell}$ [iℓ;iℓ]
$\varepsilon_{ii,\ell\ell}$ [ii;ℓℓ]
11a
11b
11c
12
14a
14b
WW
$\varepsilon_{i\ell,ii}$ [ii;iℓ]
$\varepsilon_{\ell\ell,i\ell}$ [iℓ;ℓℓ]
$E_{i\ell}$ <i|ĥ|ℓ>
$\frac{1}{2}\varepsilon_{i\ell,i\ell}$ [iℓ;iℓ]
$\frac{1}{2}\varepsilon_{\ell\ell,\ell\ell}$ [ℓℓ;ℓℓ]
$E_{\ell\ell}$ <ℓ|ĥ|ℓ>

Figure 4.

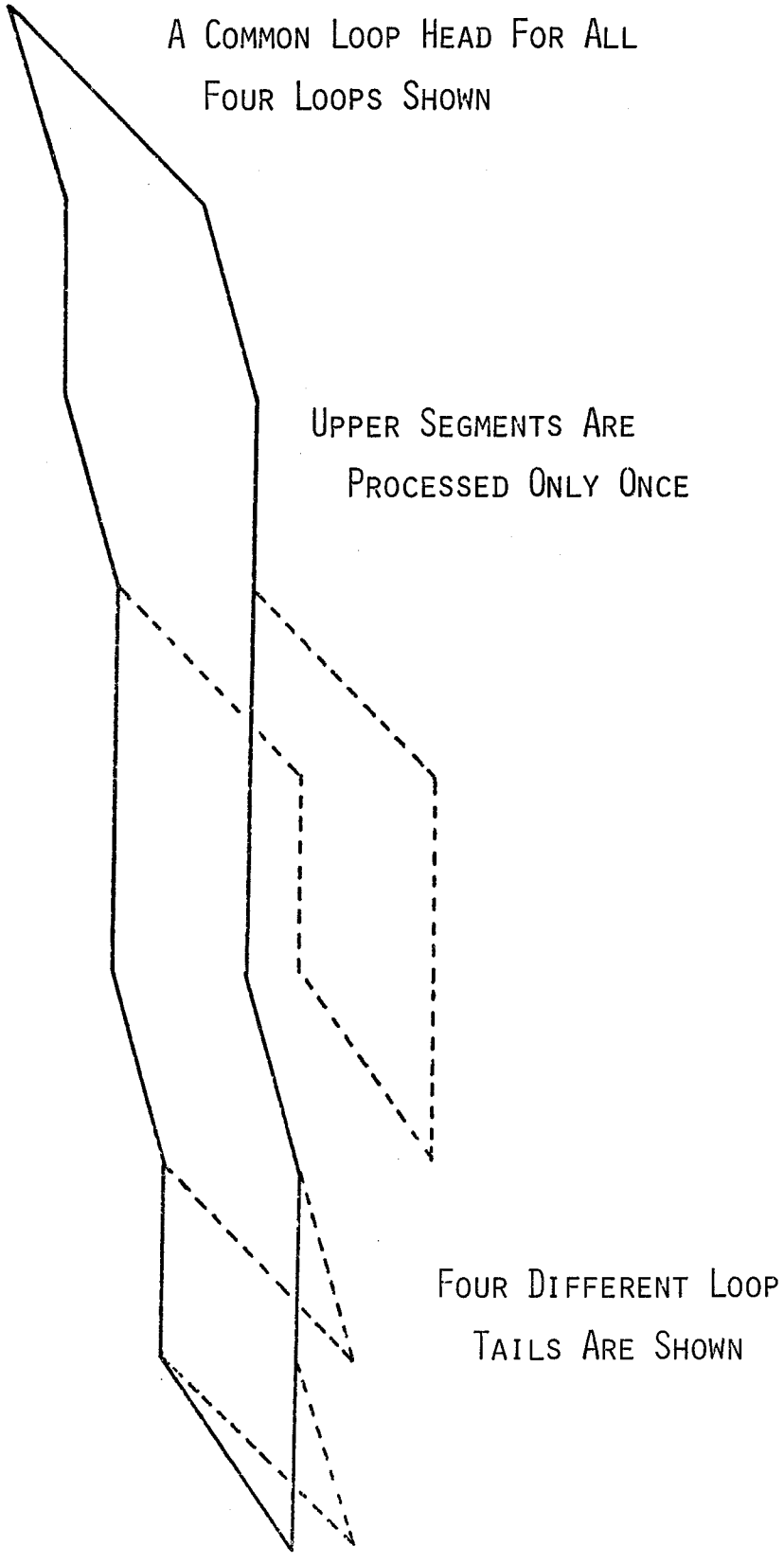

A Common Loop Head For All
Four Loops Shown
Upper Segments Are
Processed Only Once
Four Different Loop
Tails Are Shown

Figure 5.

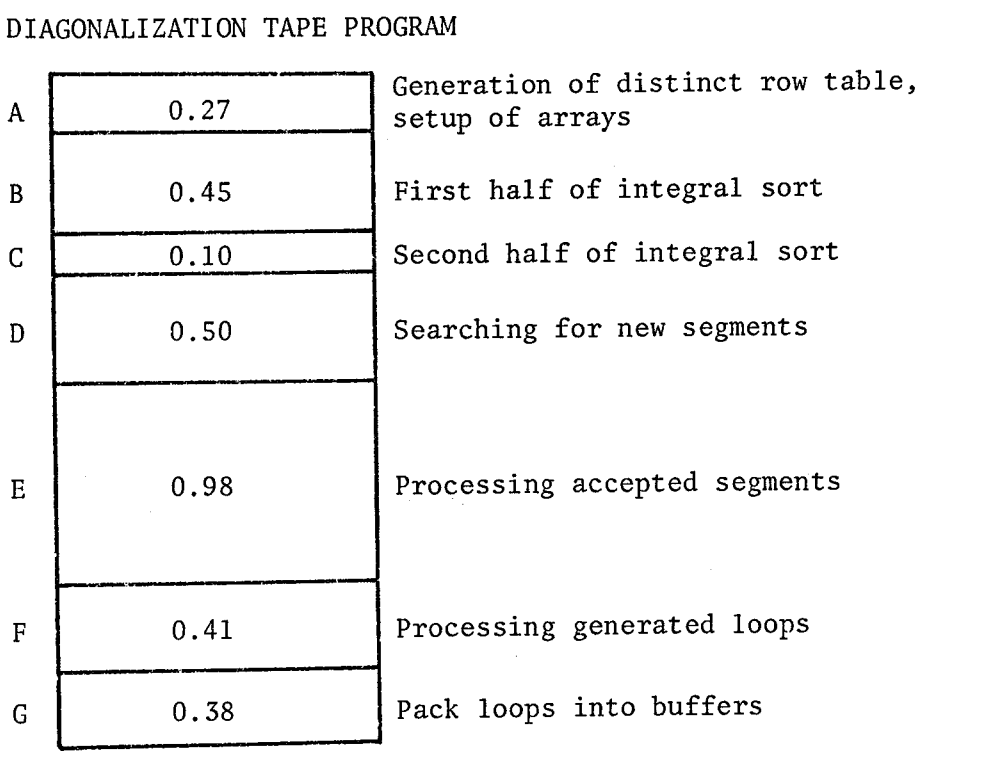

DIAGONALIZATION TAPE PROGRAM
A
0.27
Generation of distinct row table, setup of arrays
B
0.45
First half of integral sort
C
0.10
Second half of integral sort
D
0.50
Searching for new segments
E
0.98
Processing accepted segments
F
0.41
Processing generated loops
G
0.38
Pack loops into buffers
DIAGONALIZATION PROGRAM
H
0.54
Each iteration of the diagonalization

Figure 6. The BUGACI System of Programs

DRTGEN - Distinct row table generator

TRANS - Integral transformation program

NEWCORE - Adjusts the MO integral file to include more frozen core orbitals

BUGME - Diagonalization tape generation program (matrix elements)

DIRECT - Unitary group direct CI program

DIAG - Diagonalization program

DENSITY - One particle density matrix program and important configuration listing and natural orbital generator

CIPROPS - One electron properties

TWOPDM - Two particle density matrix generator

SUPERCI - Generate the Lagrangian and compute the MCSCF transformation matrix

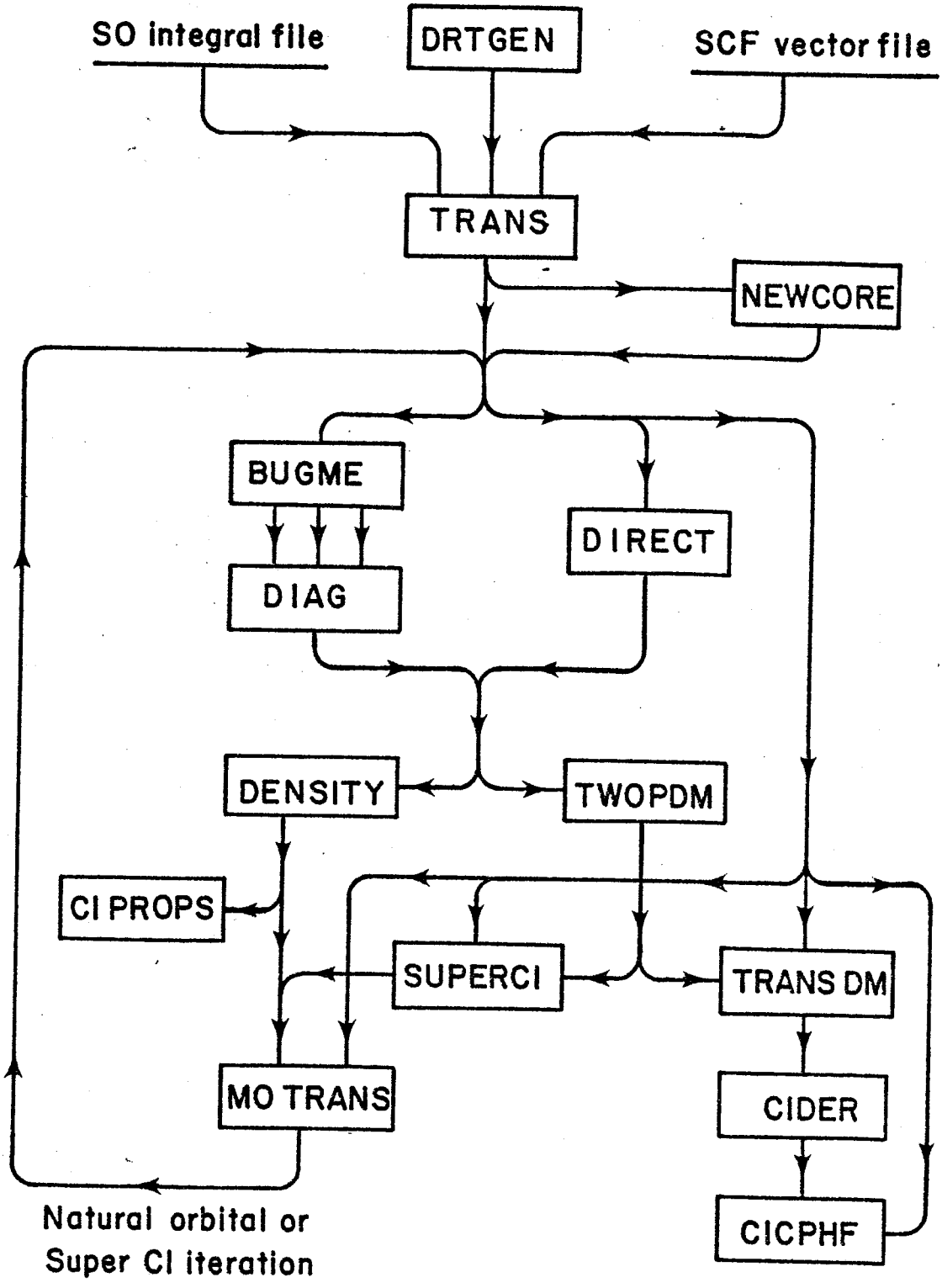
MOTRANS - Integral transformation program transforming the original MO integrals to a new MO basis

TRANSDM - Generate the Lagrangian and transform the two particle density matrix

CIDER - Compute CI gradient contribution of integral derivatives

CICPHF - Solves coupled perturbed Hartree-Fock equation and computes the CI gradient contribution

The BUGACI System of Programs



Example I. BH_3 Ground State C_{2v} Subgroup

Extended basis set, 34 basis functions, 2355 configurations

Integrals over basis functions	9.01 min
SCF (11 iterations)	6.23 min
Integral Transformation	9.08 min

Timing Comparisons:

	Formula Tape Generation	Diagonalization Tape Construction	Diagonalization	Totals for One Calculation
Conventional CI	33.80 min	4.97 min	4.14 min	42.91 min
Integral Driven Direct CI	No Formula Tape	11.78 min	5.77 min	17.55 min
Preliminary GUGA	4.93 min	1.98 min	4.37 min	11.28 min
Current Version of GUGA	No Formula Tape	3.07 min	4.35 min	7.42 min

Timing Breakdown of Current Version:

Distinct row table generation	16.1 seconds
Integral sorting	28.6 seconds
Loop generation (with implicit external)	139.3 seconds
Loop generation (purely loop-driven)	149.8 seconds
Matrix multiply each diagonalization iteration	32.7 seconds

Example II. CH_2 ($^1A_1$) C_{2v} Symmetry

$$1a_1^2\ 2a_1^2\ 3a_1^2\ 1b_2^2$$

Extended basis set, 61 basis functions
(58 molecular orbitals), 7310 configurations

Integral Transformation 174.32 min

Timing Comparisons:

	Formula Tape Generation	Diagonalization Tape Construction	Diagonalization	Totals for One Calculation
Integral Driven Direct CI	No Formula Tape	89.72 min	51.02 min	140.74 min
Preliminary GUGA	28.18 min	14.00 min	25.95 min	69.13 min
Current Version of GUGA	No Formula Tape	23.78 min	25.89 min	49.67 min

Timing Breakdown of Current Version:

Distinct row table generation	81.0 seconds
Integral sorting	379.2 seconds
Loop generation (with implicit external)	966.8 seconds
Loop generation (purely loop-driven)	1066.8 seconds
Matrix multiply each diagonalization iteration	219.7 seconds

Example III. C_2H_4 $^1B_{1u}$ (V state) D_{2h} Symmetry

3 open shell singlet reference configurations

$$2a_g^2\ 2b_{1u}^2\ 1b_{1u}^2\ 3a_g^2\ 1b_{3g}^2\ 1b_{3u}\ 1b_{2g}$$
$$1b_{3u}\ 2b_{2g}$$
$$1b_{3u}\ 3b_{2g}$$

Double zeta plus polarization and diffuse functions basis set,
44 basis functions (36 molecular orbitals included in the CI)
7022 configurations (interacting space only)

Integral Transformation 28.31 min

Timing Comparisons:

	Formula Tape Generation	Diagonalization Tape Construction	Diagonalization	Totals for One Calculation
Conventional CI	211.9 min	17.4 min	16.9 min	246.2 min
Preliminary GUGA	18.9 min	4.2 min	14.2 min	37.3 min
Current Version of GUGA	No Formula Tape	10.2 min	14.1 min	24.3 min

Timing Breakdown of Current Version:

Distinct row table generation	73.1 seconds
Integral sorting	46.1 seconds
Loop generation (with implicit external)	493.6 seconds
Loop generation (purely loop-driven)	501.0 seconds
Matrix multiply each diagonalization iteration	74.8 seconds

Example IV. SO_2 $2^1A'$ C_s Symmetry

3 single reference configurations

$7a'^2$ $8a'^2$ $9a'^2$ $10a'^2$ $11a'^2$ $12a'^2$ $13a'^2$ $2a''^2$ $3a''^2$

$3a''$ $4a''$

$4a''^2$

Double zeta plus sulfur d function basis set;
44 basis functions (29 orbitals included in the CI)
23,613 configurations (interacting space only)

Integral Transformation	77.96 min
Diagonalization Tape Construction	38.59 min
Sorting Diagonalization to Form a Partitioned Hamiltonian	57.51 min
Diagonalization (9 iterations)	164.39 min

Table I. A small sample distinct row table (DRT).[a] n=4, N=3, S=½

i	j	a	b	c	k_0	k_1	k_2	k_3	y_0	y_1	y_2	y_3	x	$\bar{x}$
4	1	1	1	2	1	2	3	4	0	8	14	17	20	1
3	1	1	1	1	1	2	3	4	0	2	5	6	8	1
	2	1	0	2	2	-	4	5	0	-	3	5	6	1
	3	0	2	1	3	4	-	-	0	1	-	-	3	1
	4	0	1	2	4	5	-	-	0	2	-	-	3	1
2	1	1	1	0	-	1	-	2	-	0	-	1	2	1
	2	1	0	1	1	-	2	3	0	-	1	2	3	2
	3	0	2	0	-	2	-	-	-	0	-	-	1	2
	4	0	1	1	2	3	-	-	0	1	-	-	2	4
	5	0	0	2	3	-	-	-	0	-	-	-	1	2
1	1	1	0	0	-	-	-	1	-	-	-	0	1	3
	2	0	1	0	-	1	-	-	-	0	-	-	1	9
	3	0	0	1	1	-	-	-	0	-	-	-	1	8
0	1	0	0	0	-	-	-	-	-	-	-	-	1	20

[a]Reference 24.

Table II. Coefficients of the five S=1/2 Gelfand states (configurations) represented by Paldus tableaux expressed over Slater determinants for five unpaired electrons.

	1*	2*	3	4	5
[p] =	$\begin{bmatrix} 2&1&2\\ 2&0&2\\ 1&1&1\\ 1&0&1\\ 0&1&0\\ 0&0&0 \end{bmatrix}$	$\begin{bmatrix} 2&1&2\\ 2&0&2\\ 1&1&1\\ 0&2&0\\ 0&1&0\\ 0&0&0 \end{bmatrix}$	$\begin{bmatrix} 2&1&2\\ 1&2&1\\ 1&1&1\\ 1&0&1\\ 0&1&0\\ 0&0&0 \end{bmatrix}$	$\begin{bmatrix} 2&1&2\\ 1&2&1\\ 1&1&1\\ 0&2&0\\ 0&1&0\\ 0&0&0 \end{bmatrix}$	$\begin{bmatrix} 2&1&2\\ 1&2&1\\ 0&3&0\\ 0&2&0\\ 0&1&0\\ 0&0&0 \end{bmatrix}$
$1\beta\ 2\beta\ 3\alpha\ 4\alpha\ 5\alpha$	0	$-2/\sqrt{12}$	0	1/3	$-1/\sqrt{18}$
$1\beta\ 2\alpha\ 3\beta\ 4\alpha\ 5\alpha$	1/2	$1/\sqrt{12}$	$-1/\sqrt{12}$	-1/6	$-1/\sqrt{18}$
$1\alpha\ 2\beta\ 3\beta\ 4\alpha\ 5\alpha$	-1/2	$1/\sqrt{12}$	$1/\sqrt{12}$	-1/6	$-1/\sqrt{18}$
$1\beta\ 2\alpha\ 3\alpha\ 4\beta\ 5\alpha$	-1/2	$1/\sqrt{12}$	$-1/\sqrt{12}$	1/6	$1/\sqrt{18}$
$1\alpha\ 2\beta\ 3\alpha\ 4\beta\ 5\alpha$	1/2	$1/\sqrt{12}$	$1/\sqrt{12}$	1/6	$1/\sqrt{18}$
$1\alpha\ 2\alpha\ 3\beta\ 4\beta\ 5\alpha$	0	$-2/\sqrt{12}$	0	-1/3	$1/\sqrt{18}$
$1\beta\ 2\alpha\ 3\alpha\ 4\alpha\ 5\beta$	0	0	$2/\sqrt{12}$	-1/3	$1/\sqrt{18}$
$1\alpha\ 2\beta\ 3\alpha\ 4\alpha\ 5\beta$	0	0	$-2/\sqrt{12}$	-1/3	$1/\sqrt{18}$
$1\alpha\ 2\alpha\ 3\beta\ 4\alpha\ 5\beta$	0	0	0	2/3	$1/\sqrt{18}$
$1\alpha\ 2\alpha\ 3\alpha\ 4\beta\ 5\beta$	0	0	0	0	$-1/\sqrt{2}$

*Hartree-Fock interacting configurations

Table III. A sample distinct row table (DRT) that describes the configurations of the Hartree-Fock interacting space from the two open-shell reference configurations:

$$1a^2\ 2a^2\ 3a\ 4a$$

$$1a^2\ 2a^2\ 3a\ 5a$$

for a seven orbital calculation without spatial symmetry.

i	j	a	b	c	T	T'	k_0	k_1	k_2	k_3	x	$\bar{x}$
7	1	3	0	4	0	0	1	-	2	3	177	1
Orbital 3a active in references												
6	1	3	0	3	0	0	1	-	3	6	46	1
	2	2	1	3	0	0	2	4	7	8	96	1
	3	2	0	4	1	1	5	-	8	9	35	1
Orbital 4a active in references												
5	1	3	0	2	0	0	1	-	2	4	18	1
	2	2	1	2	0	0	2	3	5	6	39	1
	3	2	1	2	0	1	2	4	5	7	21	1
	4	2	0	3	0	1	3	-	6	8	28	1
	5	2	0	3	1	1	4	-	6	8	18	1
	6	2	0	3	1	2	4	-	7	-	7	1
	7	1	2	2	1	1	5	6	-	9	15	1
	8	1	1	3	1	2	6	8	9	-	14	2
	9	1	0	4	2	3	8	-	-	-	3	1
Orbital 5a active in references												
4	1	3	0	1	0		-	-	-	1	3	1
	2	2	1	1	0		-	1	2	3	10	3
	3	2	0	2	0		1	-	3	5	15	2
	4	2	0	2	1		-	-	4	6	5	4
	5	1	2	1	1		-	4	-	8	4	3
	6	1	1	2	1		4	6	8	9	10	6
	7	1	1	2	2		-	7	-	10	2	2
	8	1	0	3	2		7	-	10	11	3	5
	9	0	2	2	2		-	10	-	-	1	3

Table III. (continued)

i	j	a	b	c	T	T'	k_0	k_1	k_2	k_3	x	$\bar{x}$
Orbital 1a doubly occupied in references												
3	1	2	0	1	0		-	-	-	1	3	6
	2	1	2	0	0		-	-	-	2	1	3
	3	1	1	1	0		-	1	2	3	6	5
	4	1	1	1	1		-	-	-	3	2	13
	5	1	0	2	0		1	-	3	4	6	2
	6	1	0	2	1		-	-	3	4	3	10
	7	1	0	2	2		-	-	-	4	1	7
	8	0	2	1	1		-	3	-	-	2	9
	9	0	1	2	1		3	4	-	-	3	6
	10	0	1	2	2		-	4	-	-	1	10
	11	0	0	3	2		4	-	-	-	1	5
Orbital 2a doubly occupied in references												
2	1	1	0	1			1	-	2	3	3	13
	2	0	2	0			-	2	-	-	1	8
	3	0	1	1			2	3	-	-	2	45
	4	0	0	2			3	-	-	-	1	40
Orbital 6a unoccupied in references												
1	1	1	0	0			-	-	-	1	1	13
	2	0	1	0			-	1	-	-	1	66
	3	0	0	1			1	-	-	-	1	98
Orbital 7a unoccupied in references												
0	1	0	0	0			-	-	-	-	1	177

Table IV. The Loop Searching Master Table

ISEG	JSEG	IS(Δb)	by	FS(Δb)	next	s's	tracks	kjcond	coefficients	code
1	1	0(0)	WW/W	0(0)	0	33	2,1	11	X'=2X,Z'=½	50
	2	0(0)	W	0(0)	0	22	2	11	X'=X	1
	3	0(0)	W	0(0)	0	11	2	11	X'=X	1
	4	0(0)	$\overline{R}/\overline{R}$	R(+1)	17	02	3	11	X'=X	1
	5	0(0)	$\overline{R}/W\overline{R}$	R(+1)	15	13	(3,1)	11	X'=XA(0,1)	3
	6	0(0)	$\overline{R}/\overline{R}$	R(-1)	18	01	3	11	X'=X	1
	7	0(0)	$\overline{R}/W\overline{R}$	R(-1)	16	23	(3,1)	11	X'=XA(2,1)	4
	8	0(0)	$\overline{RR}$	RR(0)	10	03	1	11	X'=X	1
	9	0(0)	$\overline{RL}/W$	RL/0(0)	5	11	1,2	10	X'=Xt,Y'=-XtA(-1,1)	44
	10	0(0)	$\overline{RL}/W$	RL/0(0)	5	22	1,2	10	X'=Xt,Y'=XtA(3,1)	45
	11	0(0)	$\overline{RL}/W$	RL/0(0)	4	33	1,2	10	X'=$\sqrt{2}$X	51
	12	0(0)	$\overline{RL}$	RL(+2)	7	12	1	10	X'=X	1
	13	0(0)	$\overline{R}$	R(+1)	2	02	1	-10	X'=X	1
	14	0(0)	$\overline{R}$	R(+1)	2	13	1	-10	X'=XA(0,1)	3
	15	0(0)	$\overline{R}$	R(-1)	3	01	1	-10	X'=X	1
	16	0(0)	$\overline{R}$	R(-1)	3	23	1	-10	X'=XA(2,1)	4

ISEG	JSEG	IS(Δb)	by	FS(Δb)	next	s's	tracks	kjcond	coefficients	code
2	17	R(+1)	$\underline{R}\overline{L}$	L(+1)	20	30	-	11	X'=XA(1,0)	5
	18	R(+1)	WR/$\overline{R}\underline{R}$	R(+1)	21	22	2,1	11	X'=-X,Z'=-1	52
	19	R(+1)	WR/$\overline{R}\underline{R}$	R(+1)	21	33	2,1	11	X'=-2X,Z'=-½	53
	20	R(+1)	WR	R(+1)	21	11	2	11	X'=XC(0)	6
	21	R(+1)	WR/$\overline{R}\underline{R}$	R(-1)	22	21	2,1	11	X'=X/b,Z'=b	54
	22	R(+1)	R$\overline{R}$	RR(0)	11	01	1,3	10	X'=XtA(1,0),Y'=-XtA(-1,0)	40
	23	R(+1)	R$\overline{R}$	RR(0)	11	23	1,3	10	X'=-Xt,Y'=XtA(2,0)	41
	24	R(+1)	R$\overline{R}$	RR(+2)	12	02	1,3	10	X'=X	1
	25	R(+1)	R$\overline{R}$	RR(+2)	12	13	1,3	10	X'=XA(-1,0)	7
	26	R(+1)	$\overline{R}\underline{L}$/$\underline{R}$	RL/0(0)	6	20	1,2	10	X'=Xt,Y'=XtA(2,0)	46
	27	R(+1)	$\overline{R}\underline{L}$/$\underline{R}$	RL/0(0)	6	31	1,2	10	X'=XtA(1,0),Y'=XtA(-1,0)	47
	28	R(+1)	R$\overline{L}$	RL(+2)	7	10	3	10	X'=X	1
	29	R(+1)	R$\overline{L}$	RL(+2)	7	32	3	10	X'=-XA(1,0)	8
	30	R(+1)	R	R(+1)	2	00	-	-10	X'=X	1
	31	R(+1)	R	R(+1)	2	11	-	-10	X'=XC(0)	6
	32	R(+1)	R	R(+1)	2	22	-	-10	X'=-X	2
	33	R(+1)	R	R(+1)	2	33	-	-10	X'=-X	2
	34	R(+1)	R	R(-1)	3	21	-	-10	X'=X/b	9

ISEG	JSEG	IS(Δb)	by	FS(Δb)	next	s's	tracks	kjcond	coefficients	code
3	35	R(-1)	$\underline{R}\overline{L}$	L(-1)	19	30	-	11	X'=XA(1,2)	10
	36	R(-1)	WR/$\overline{R}\underline{R}$	R(-1)	22	11	2,1	11	X'=-X,Z'=-1	52
	37	R(-1)	WR/$\overline{R}\underline{R}$	R(-1)	22	33	2,1	11	X'=-2X,Z'=-½	53
	38	R(-1)	WR	R(-1)	22	22	2	11	X'=XC(2)	11
	39	R(-1)	WR/$\overline{R}\underline{R}$	R(+1)	21	12	2,1	11	X'=-X/(b+2),Z'=-(b+2)	55
	40	R(-1)	R$\overline{R}$	RR(0)	11	02	1,3	10	X'=XtA(1,2),Y'=XtA(3,2)	42
	41	R(-1)	R$\overline{R}$	RR(0)	11	13	1,3	10	X'=-Xt,Y'=-XtA(0,2)	43
	42	R(-1)	R$\overline{R}$	RR(-2)	13	01	1,3	10	X'=X	1
	43	R(-1)	R$\overline{R}$	RR(-2)	13	23	1,3	10	X'=XA(3,2)	12
	44	R(-1)	$\overline{R}\underline{L}$/R	RL/0(0)	6	10	1,2	10	X'=Xt,Y'=-XtA(0,2)	48
	45	R(-1)	$\overline{R}\underline{L}$/R	RL/0(0)	6	32	1,2	10	X'=XtA(1,2),Y'+-XtA(3,2)	49
	46	R(-1)	R$\overline{L}$	RL(-2)	9	20	3	10	X'=X	1
	47	R(-1)	R$\overline{L}$	RL(-2)	9	31	3	10	X'=-XA(1,2)	13
	48	R(-1)	R	R(-1)	3	00	-	-10	X'=X	1
	49	R(-1)	R	R(-1)	3	11	-	-10	X'=-X	2
	50	R(-1)	R	R(-1)	3	22	-	-10	X'=XC(2)	11
	51	R(-1)	R	R(-1)	3	33	-	-10	X'=-X	2
	52	R(-1)	R	R(+1)	2	12	-	-10	X'=-X/(b+2)	36

ISEG	JSEG	IS(Δb)	by	FS(Δb)	next	s's	tracks	kjcond	coefficients	code
4	53	RL/0(0)	RL/W	0(0)	0	11	-	01	X'=-Xt,Z'=-2	77
	54	RL/0(0)	RL/W	0(0)	0	22	-	01	X'=-Xt,Z'=-2	77
	55	RL/0(0)	RL/W	0(0)	0	33	-	01	$X'=-\sqrt{2}X,Z'=-2$	78
	56	RL/0(0)	RL/R	R(+1)	21	02	-	01	X'=-Xt,Z'=-2	77
	57	RL/0(0)	RL/R	R(+1)	21	13	-	01	X'=-XtA(0,1),Z'=-2	79
	58	RL/0(0)	RL/R	R(-1)	22	01	-	01	X'=-Xt,Z'=-2	77
	59	RL/0(0)	RL/R	R(-1)	22	23	-	01	X'=-XtA(2,1),Z'=-2	80
	60	RL/0(0)	RL/0	RL/0(0)	4	00	-	00	X'=X	1
	61	RL/0(0)	RL/0	RL/0(0)	4	11	-	00	X'=X	1
	62	RL/0(0)	RL/0	RL/0(0)	4	22	-	00	X'=X	1
	63	RL/0(0)	RL/0	RL/0(0)	4	33	-	00	X'=X	1

ISEG	JSEG	IS(Δb)	by	FS(Δb)	next	s's	tracks	kjcond	coefficients	code
5	64	RL/0(0)	RL/W	0(0)	0	11	-	01	X'=-Xt+YtA(2,0)† Z'=√2X/X'	87
	65	RL/0(0)	RL/W	0(0)	0	22	-	01	X'=-Xt-YtA(0,2)† Z'=√2X/X'	68
	66	RL/0(0)	RL/W	0(0)	0	33	-	01	X'=-√2X,Z'=-2	82
	67	RL/0(0)	RL/$\overline{R}$	R(+1)	21	02	-	01	X'=-Xt-YtA(0,2)† Z'=√2X/X'	68
	68	RL/0(0)	RL/$\overline{R}$	R(+1)	21	13	-	01	X'=-XtA(0,1)-YtA(2,1)† Z'=√2XA(0,1)/X'	69
	69	RL/0(0)	RL/$\overline{R}$	R(-1)	22	01	-	01	X'=-Xt+YtA(2,0)† Z'=√2X/X'	67
	70	RL/0(0)	RL/$\overline{R}$	R(-1)	22	23	-	01	X'=-XtA(2,1)+YtA(0,1)† Z'=√2XA(2,1)/X'	70
	71	RL/0(0)	RL/0	RL/0(0)	5	00	-	00	X'=X,Y'=Y	71
	72	RL/0(0)	RL/0	RL/0(0)	5	11	-	00	X'=X,Y'=YD(0)	75
	73	RL/0(0)	RL/0	RL/0(0)	5	22	-	00	X'=X,Y'=YD(1)	76
	74	RL/0(0)	RL/0	RL/0(0)	5	33	-	00	X'=X,Y'=Y	71
	75	RL(0)	RL	RL(+2)	7	12	1	00	X'=-YE(0)	83

†If X'=0 transfer (X'Z') to X' and tracks to 2

ISEG	JSEG	IS(Δb)	by	FS(Δb)	next	s's	tracks	kjcond	coefficients	code
6	76	RL/0(0)	$\underline{RL}/W$	0(0)	0	11	-	01	$X'=-\underline{X}t+YtA(2,0)$ † $Z'=\sqrt{2}X/X'$	87
	77	RL/0(0)	$\underline{RL}/W$	0(0)	0	22	-	01	$X'=-\underline{X}t-YtA(0,2)$ † $Z'=\sqrt{2}X/X'$	68
	78	RL/0(0)	$\underline{RL}/W$	0(0)	0	33	-	01	$X'=-\sqrt{2}X/X, Z'=-2$	82
	79	RL/0(0)	$\underline{RL}/\overline{R}$	R(+1)	21	02	3,2	01	$X'=-\underline{X}t-YtA(0,2)$ † $Z'=\sqrt{2}X/X'$	68
	80	RL/0(0)	$\underline{RL}/\overline{R}$	R(+1)	21	13	3,2	01	$X'=-\underline{X}tA(0,1)-YtA(2,1)$ † $Z'=\sqrt{2}XA(0,1)/X'$	69
	81	RL/0(0)	$\underline{RL}/\overline{R}$	R(-1)	22	01	3,2	01	$X'=-\underline{X}t+YtA(2,0)$ † $Z'=\sqrt{2}X/X'$	67
	82	RL/0(0)	$\underline{RL}/\overline{R}$	R(-1)	22	23	3,2	01	$X'=-\underline{X}tA(2,1)+YtA(0,1)$ † $Z'=\sqrt{2}XA(2,1)/X'$	70
	83	RL/0(0)	$\underline{RL}/\overline{L}$	L(-1)	19	20	-	01	$X'=-\underline{X}t-YtA(0,2)$ † $Z'=\sqrt{2}X/X'$	68
	84	RL/0(0)	$\underline{RL}/\overline{L}$	L(-1)	19	31	-	01	$X'=-\underline{X}tA(0,1)-YtA(2,1)$ † $Z'=\sqrt{2}XA(0,1)/X'$	69
	85	RL/0(0)	$\underline{RL}/\overline{L}$	L(+1)	20	10	-	01	$X'=-\underline{X}t+YtA(2,0)$ † $Z'=\sqrt{2}X/X'$	67
	86	RL/0(0)	$\underline{RL}/\overline{L}$	L(+1)	20	32	-	01	$X'=-\underline{X}tA(2,1)+YtA(0,1)$ † $Z'=\sqrt{2}XA(2,1)/X'$	70
	87	RL/0(0)	RL/0	RL/0(0)	6	00	-	00	X'=X,Y'=Y	71
	88	RL/0(0)	RL/0	RL/0(0)	6	11	-	00	X'=X,Y'=YD(0)	75
	89	RL/0(0)	RL/0	RL/0(0)	6	22	-	00	X'=X,Y'=YD(1)	76
	90	RL/0(0)	RL/0	RL/0(0)	6	33	-	00	X'=X,Y'=Y	71
	91	RL(0)	RL	RL(+2)	7	12	3	00	X'=-YE(0)	83
	92	RL(0)	RL	RL(-2)	9	21	3	00	X'=-YE(0)	83

† If X'=0 transfer (X'Z') to X' and tracks to 2

ISEG	JSEG	IS(Δb)	by	FS(Δb)	next	s's	tracks	kjcond	coefficients	code
7	93	RL(+2)	RL	0(0)	0	21	1	01	X'=XC(0)	6
	94	RL(+2)	RL	R(+1)	21	01	-	01	X'=XC(0)	6
	95	RL(+2)	RL	R(+1)	21	23	-	01	X'=-XA(-1,0)	74
	96	RL(+2)	RL	L(+1)	20	20	1	01	X'=XC(0)	6
	97	RL(+2)	RL	L(+1)	20	31	1	01	X'=-XA(1,0)	8
	98	RL(+2)	RL	RL(+2)	7	00	-	00	X'=X	1
	99	RL(+2)	RL	RL(+2)	7	11	-	00	X'=-XC(0)	16
	100	RL(+2)	RL	RL(+2)	7	22	-	00	X'=-XC(0)	16
	101	RL(+2)	RL	RL(+2)	7	33	-	00	X'=X	1
	102	RL(+2)	RL	RL(0)	8	21	-	00	$X'=-X\sqrt{2}/b$	17

ISEG	JSEG	IS(Δb)	by	FS(Δb)	next	s's	tracks	kjcond	coefficients	code
8	103	RL(0)	RL	0(0)	0	11	1	01	X'=XtA(2,0)	18
	104	RL(0)	RL	0(0)	0	22	1	01	X'=-XtA(0,2)	19
	105	RL(0)	RL	R(+1)	21	02	-	01	X'=-XtA(0,2)	19
	106	RL(0)	RL	R(+1)	21	13	-	01	X'=-XtA(2,1)	20
	107	RL(0)	RL	R(-1)	22	01	-	01	X'=XtA(2,0)	18
	108	RL(0)	RL	R(-1)	22	23	-	01	X'=XtA(0,1)	21
	109	RL(0)	RL	L(-1)	19	20	1	01	X'=-XtA(0,2)	19
	110	RL(0)	RL	L(-1)	19	31	1	01	X'=-XtA(2,1)	20
	111	RL(0)	RL	L(+1)	20	10	1	01	X'=XtA(2,0)	18
	112	RL(0)	RL	L(+1)	20	32	1	01	X'=XtA(0,1)	21
	113	RL(0)	RL	RL(0)	8	00	-	00	X'=X	1
	114	RL(0)	RL	RL(0)	8	11	-	00	X'=XD(0)	22
	115	RL(0)	RL	RL(0)	8	22	-	00	X'=XD(1)	23
	116	RL(0)	RL	RL(0)	8	33	-	00	X'=X	1
	117	RL(0)	RL	RL(+2)	7	12	-	00	X'=-XE(0)	24
	118	RL(0)	RL	RL(-2)	9	21	-	00	X'=-XE(0)	24

ISEG	JSEG	IS(Δb)	by	FS(Δb)	next	s's	tracks	kjcond	coefficients	code
9	119	RL(-2)	RL	0(0)	0	12	1	01	X'=XC(2)	11
	120	RL(-2)	RL	R(-1)	22	02	-	01	X'=XC(2)	11
	121	RL(-2)	RL	R(-1)	22	13	-	01	X'=-XA(3,2)	81
	122	RL(-2)	RL	L(-1)	19	10	1	01	X'=XC(2)	11
	123	RL(-2)	RL	L(-1)	19	32	1	01	X'=-XA(1,2)	13
	124	RL(-2)	RL	RL(-2)	9	00	-	00	X'=X	1
	125	RL(-2)	RL	RL(-2)	9	11	-	00	X'=-XC(2)	27
	126	RL(-2)	RL	RL(-2)	9	22	-	00	X'=-XC(2)	27
	127	RL(-2)	RL	RL(-2)	9	33	-	00	X'=X	1
	128	RL(-2)	RL	RL(0)	8	12	-	00	$X'=-\sqrt{2}X/(b+2)$	28

ISEG	JSEG	IS(Δb)	by	FS(Δb)	next	s's	tracks	kjcond	coefficients	code
10	129	RR(0)	RR	0(0)	0	30	-	01	X'=X	1
	130	RR(0)	RR	R(+1)	21	10	-	01	X'=XA(0,1)	3
	131	RR(0)	RR	R(+1)	21	32	-	01	X'=-X	2
	132	RR(0)	RR	R(-1)	22	20	-	01	X'=XA(2,1)	4
	133	RR(0)	RR	R(-1)	22	31	-	01	X'=-X	2
	134	RR(0)	RR	RR(0)	10	00	-	00	X'=X	1
	135	RR(0)	RR	RR(0)	10	11	-	00	X'=-X	2
	136	RR(0)	RR	RR(0)	10	22	-	00	X'=-X	2
	137	RR(0)	RR	RR(0)	10	33	-	00	X'=X	1

ISEG	JSEG	IS(Δb)	by	FS(Δb)	next	s's	tracks	kjcond	coefficients	code
11	138	RR(0)	RR	0(0)	0	30	1	01	$X'=\sqrt{2}X$	29
	139	RR(0)	RR/RR*	R(+1)	21	10	-	01	X'=XtA(0,1)+YtA(2,1)†† Z'=(XtA(0,1)-YtA(2,1))/X'	63
	140	RR(0)	RR/RR*	R(+1)	21	32	-	01	X'=-Xt-YtA(0,2)†† Z'=(-Xt+YtA(0,2))/X'	64
	141	RR(0)	RR/RR*	R(-1)	22	20	-	01	X'=XtA(2,1)-YtA(0,1)†† Z'=(XtA(2,1)+YtA(0,1))/X'	65
	142	RR(0)	RR/RR*	R(-1)	22	31	-	01	X'=-Xt+YtA(2,0)†† Z'=(-Xt-YtA(2,0))/X'	66
	143	RR(0)	RR	RR(0)	11	00	-	00	X'=X,Y'=Y	71
	144	RR(0)	RR	RR(0)	11	11	-	00	X'=-X,Y'=-YD(0)	72
	145	RR(0)	RR	RR(0)	11	22	-	00	X'=-X,Y'=-YD(1)	73
	146	RR(0)	RR	RR(0)	11	33	-	00	X'=X,Y'=Y	71
	147	RR(0)	RR	RR(+2)	12	12	-	00	X'=YB(1)	84
	148	RR(0)	RR	RR(-2)	13	21	-	00	X'=YB(0)	85

†† If X'=0 transfer (X'Z') to X' and tracks to 3

ISEG	JSEG	IS(Δb)	by	FS(Δb)	next	s's	tracks	kjcond	coefficients	code
12	149	RR(+2)	RR/RR*	R(+1)	21	20	-	01	X'=X,Z'=-1	56
	150	RR(+2)	RR/RR*	R(+1)	21	31	-	01	X'=XA(1,0),Z'=-1	57
	151	RR(+2)	RR	RR(+2)	12	00	-	00	X'=X	1
	152	RR(+2)	RR	RR(+2)	12	11	-	00	X'=XD(-1)	30
	153	RR(+2)	RR	RR(+2)	12	22	-	00	X'=X	1
	154	RR(+2)	RR	RR(+2)	12	33	-	00	X'=X	1
	155	RR(+2)	RR	RR(0)	14	21	-	00	X'=XB(-1)	86

ISEG	JSEG	IS(Δb)	by	FS(Δb)	next	s's	tracks	kjcond	coefficients	code
13	156	RR(-2)	RR/RR*	R(-1)	22	10	-	01	X'=X,Z'=-1	56
	157	RR(-2)	RR/RR*	R(-1)	22	32	-	01	X'=XA(1,2),Z'=-1	58
	158	RR(-2)	RR	RR(-2)	13	00	-	00	X'=X	1
	159	RR(-2)	RR	RR(-2)	13	11	-	00	X'=X	1
	160	RR(-2)	RR	RR(-2)	13	22	-	00	X'=XD(2)	31
	161	RR(-2)	RR	RR(-2)	13	33	-	00	X'=X	1
	162	RR(-2)	RR	RR(0)	14	12	-	00	X'=XB(2)	32

ISEG	JSEG	IS(Δb)	by	FS(Δb)	next	s's	tracks	kjcond	coefficients	code
14	163	RR(0)	RR/RR*	R(+1)	21	10	-	01	X'=XtA(2,1),Z'=-1	59
	164	RR(0)	RR/RR*	R(+1)	21	32	-	01	X'=-XtA(0,2),Z'=-1	60
	165	RR(0)	RR/RR*	R(-1)	22	20	-	01	X'=-XtA(0,1),Z'=-1	61
	166	RR(0)	RR/RR*	R(-1)	22	31	-	01	X'=XtA(2,0),Z'=-1	62
	167	RR(0)	RR	RR(0)	14	00	-	00	X'=X	1
	168	RR(0)	RR	RR(0)	14	11	-	00	X'=-XD(0)	33
	169	RR(0)	RR	RR(0)	14	22	-	00	X'=XD(-1)	30
	170	RR(0)	RR	RR(0)	14	33	-	00	X'=X	1
	171	RR(0)	RR	RR(+2)	12	12	-	00	X'=XB(1)	34
	172	RR(0)	RR	RR(-2)	13	21	-	00	X'=XB(0)	35

ISEG	JSEG	IS(Δb)	by	FS(Δb)	next	s's	tracks	kjcond	coefficients	code
15	173	R(+1)	R	0(0)	0	20	-	00	X'=X	1
	174	R(+1)	R/WR	0(0)	0	31	(3,2,1)	00	X'=XA(1,0)	5
	175	R(+1)	R	R(+1)	15	00	-	00	X'=X	1
	176	R(+1)	R	R(+1)	15	11	-	00	X'=XC(0)	6
	177	R(+1)	R	R(+1)	15	22	-	00	X'=-X	2
	178	R(+1)	R	R(+1)	15	33	-	00	X'=-X	2
	179	R(+1)	R	R(-1)	16	21	-	00	X'=X/b	9
16	180	R(-1)	R	0(0)	0	10	-	00	X'=X	1
	181	R(-1)	R/WR	0(0)	0	32	(3,2,1)	00	X'=XA(1,2)	10
	182	R(-1)	R	R(-1)	16	00	-	00	X'=X	1
	183	R(-1)	R	R(-1)	16	11	-	00	X'=-X	2
	184	R(-1)	R	R(-1)	16	22	-	00	X'=XC(2)	11
	185	R(-1)	R	R(-1)	16	33	-	00	X'=-X	2
	186	R(-1)	R	R(+1)	15	12	-	00	X'=-X/(b+2)	36

ISEG	JSEG	IS(Δb)	by	FS(Δb)	next	s's	tracks	kjcond	coefficients	code
17	187	R(+1)	R	0(0)	0	20	-	00	X'=X	1
	188	R(+1)	R/WR	0(0)	0	31	(3,2)	00	X'=XA(1,0)	5
	189	R(+1)	R	R(+1)	17	00	-	00	X'=X	1
	190	R(+1)	R	R(+1)	17	11	-	00	X'=XC(0)	6
	191	R(+1)	R	R(+1)	17	22	-	00	X'=-X	2
	192	R(+1)	R	R(+1)	17	33	-	00	X'=-X	2
	193	R(+1)	R	R(-1)	18	21	-	00	X'=X/b	9
18	194	R(-1)	R	0(0)	0	10	-	00	X'=X	1
	195	R(-1)	R/WR	0(0)	0	32	(3,2)	00	X'=XA(1,2)	10
	196	R(-1)	R	R(-1)	18	00	-	00	X'=X	1
	197	R(-1)	R	R(-1)	18	11	-	00	X'=-X	2
	198	R(-1)	R	R(-1)	18	22	-	00	X'=XC(2)	11
	199	R(-1)	R	R(-1)	18	33	-	00	X'=-X	2
	200	R(-1)	R	R(+1)	17	12	-	00	X'=-X/(b+2)	36

ISEG	JSEG	IS(Δb)	by	FS(Δb)	next	s's	tracks	kjcond	coefficients	code
19	201	L(-1)	L	0(0)	0	02	-	00	X'=X	1
	202	L(-1)	L	0(0)	0	13	-	00	X'=XA(2,1)	4
	203	L(-1)	L	L(-1)	19	00	-	00	X'=X	1
	204	L(-1)	L	L(-1)	19	11	-	00	X'=XC(1)	37
	205	L(-1)	L	L(-1)	19	22	-	00	X'=-X	2
	206	L(-1)	L	L(-1)	19	33	-	00	X'=-X	2
	207	L(-1)	L	L(+1)	20	12	-	00	X'=X/(b+1)	38
20	208	L(+1)	L	0(0)	0	01	-	00	X'=X	1
	209	L(+1)	L	0(0)	0	23	-	00	X'=XA(0,1)	3
	210	L(+1)	L	L(+1)	20	00	-	00	X'=X	1
	211	L(+1)	L	L(+1)	20	11	-	00	X'=-X	2
	212	L(+1)	L	L(+1)	20	22	-	00	X'=XC(1)	37
	213	L(+1)	L	L(+1)	20	33	-	00	X'=-X	2
	214	L(+1)	L	L(-1)	19	21	-	00	X'=-X/(b+1)	39

ISEG	JSEG	IS(Δb)	by	FS(Δb)	next	s's	tracks	kjcond	coefficients	code
21	215	R(+1)	R	0(0)	0	20	-	00	X'=X	1
	216	R(+1)	R	0(0)	0	31	-	00	X'=XA(1,0)	5
	217	R(+1)	R	R(+1)	21	00	-	00	X'=X	1
	218	R(+1)	R	R(+1)	21	11	-	00	X'=XC(0)	6
	219	R(+1)	R	R(+1)	21	22	-	00	X'=-X	2
	220	R(+1)	R	R(+1)	21	33	-	00	X'=-X	2
	221	R(+1)	R	R(-1)	22	21	-	00	X'=X/b	9
22	222	R(-1)	R	0(0)	0	10	-	00	X'=X	1
	223	R(-1)	R	0(0)	0	32	-	00	X'=XA(1,2)	10
	224	R(-1)	R	R(-1)	22	00	-	00	X'=X	1
	225	R(-1)	R	R(-1)	22	11	-	00	X'=-X	2
	226	R(-1)	R	R(-1)	22	22	-	00	X'=XC(2)	11
	227	R(-1)	R	R(-1)	22	33	-	00	X'=-X	2
	228	R(-1)	R	R(+1)	21	12	-	00	X'=-X/(b+2)	36

Table V. Loop Statistics for Given Examples

	Example I BH_3 ground state 34 orbitals 2355 walks		Example II $CH(^1A_1)$ 58 orbitals 7310 walks		Example III $C_2H_4(^1B_{1u})$ 36 orbitals 7022 walks	
	Off-Diagonal	Diagonal	Off-Diagonal	Diagonal	Off-Diagonal	Diagonal
Total Number of Loops	285780	77525	1744874	403995	615597	297114
Average number of lower walks (x_t)	1.312	2.799	1.301	2.897	1.340	2.464
Percentage of loops with one lower walk	95.2%	89.3%	97.1%	93.3%	91.8%	81.8%
Average number of upper walks ($\bar{x}_h$)	1.638	1.887	1.898	1.928	1.771	2.551
Percentage of loops with one upper walk	76.7%	73.1%	68.7%	72.9%	74.9%	62.4%
Average number of contributions ($x_t \cdot \bar{x}_h$)	2.006	4.470	2.258	4.683	2.275	5.498
Percentage of loops with one contribution	72.4%	65.2%	66.1%	68.0%	68.5%	50.5%

Table VI. Theoretical results (in eV) for the vertical electronic spectrum of ketene. The entries for DZ + Rydberg SCF and DZ + Polarization SCF are changes relative to the DZ SCF excitation energies.

State	Double Zeta (DZ)[a] Self-Consistent-Field	Double Zeta Unitary CI	DZ + Rydberg SCF	DZ + Polarization SCF	Experiment[b]	Harding-Goddard[c]
$^1A_1$	0	0	0	0	0	0
$^3A_2$	2.98	3.58	-0.06	+0.18	-	3.62
$^1A_2$	3.17	3.63	-0.05	+0.23	3.7	3.69
$^3A_1$	4.70	5.39	-0.08	+0.34	5.3	5.39

[a]Reference 69.

[b]Reference 72.

[c]Reference 71.