Dissertation

Parallel Implementation of the CI-Vector Evaluation in Full CI/CASSCF and RASSCF

submitted to the University of London

in partial fulfilment of the requirements for the degree of

Doctor of Philosophy

by

Michael Klene

June 2002

Department of Chemistry

King's College

University of London

Abstract

This thesis is concerned with two methodical and one technical development to overcome computational problems relating to large scale systems in the context of the complete active space self-consistent field (CASSCF) method. The developed methods have been implemented and successfully applied to a number of electronic structure problems.

The central practical problems in large scale CASSCF computations are the construction of the matrix representation and the determination of eigenvalues of the Hamiltonian matrix. It is shown that formulae for matrix elements of the Hamiltonian matrix can be modified such that advantage can be taken of its sparsity. Non-vanishing matrix elements are identified and evaluated very efficiently. This is done by means of the new concept of reduced lists (compressed excitation lists). The Davidson or Lanczos methods are used to obtain eigenvalues of the Hamiltonian matrix in the direct CI formalism.

The second method developed is the extension of the reduced list concept to the restricted active space SCF (RASSCF) method. This very flexible multi-reference configuration interaction method is a very useful extension of the CASSCF method for an arbitrary number of weakly coupled orbitals, which are doubly occupied or unoccupied in the CASSCF reference space. The described contributions to the efficiency of the RASSCF method are based on two new concepts: the RAS model orbital space, and so called propagation rules, that allow to pinpoint whole blocks of non-vanishing elements of the Hamiltonian matrix in a very efficient manner.

In order to take advantage of modern supercomputer hardware and to reduce overall computation time both, the CASSCF and RASSCF methods have been implemented and parallelised using a distributed memory model for parallel computation on massively parallel architectures. Following an integral driven approach, a strategy of implicitly ordered asymmetric parallel task sizes and dynamic load balancing has been identified to be most efficient.

Acknowledgements

I would like to express my sincere thanks for all the support and encouragement I have received from many people.

In particular I would like to thank Prof. M.A. Robb for his supervision and guidance.

Thanks are also due to all past and present members of the research group, for a very memorable and pleasant time.

I gratefully acknowledge the assistance by Dr. Martial Boggio and Dr. Luis Blancafort, with first applications of the RASSCF program.

I would like to thank Dr. Sarah Wilsey and Dr. Graham Worth for proofreading of the manuscript.

For financial support, thanks are due to the EPSRC and to Gaussian Inc.

Finally, I wish to thank Aida and my family, for all their support and patience.

Contents

List of Tables

List of Figures

List of Algorithms

1. Introduction

The underlying physical laws necessary for the mathematical theory of a large part of physics and the whole of chemistry are thus completed and the difficulty is only that exact application of these laws leads to equations much too complicated to be soluble.
Paul Adrian Maurice Dirac (1902-1984) [1]

1.1. Perspective

Quantum chemistry is the application of the principles of quantum mechanics to chemical problems. The general aim is to explain and predict properties and reactivity of chemical compounds by calculating the expectation values of the time-independent Schrödinger equation. However, the inherent complexity of the underlying problem means that one is limited to finding approximations to the exact solution, giving rise to various methods ("levels of theory"). In practise, different levels of theory are associated with sometimes vastly different computational requirements for the treatment of a problem with a certain molecular size E (in terms of numbers of first-row non-hydrogen atoms). For example, the range of methods comprises the Hartree-Fock (HF) method, scaling with E^2–E^3 (current estimate of maximum feasible molecular size: 50–200 atoms), and the full configuration interaction (FCI) method, scaling factorially (currently feasible: 2 atoms) [2]. The choice of method will depend on the suitability of a method for the problem at hand, the required accuracy of the results and practical considerations like computational cost.

The deficiencies of the Hartree-Fock method are referred to with the term "correlation energy", and most higher levels of theory are designed to correct the Hartree-Fock errors. One of the most common treatments of the correlation energy to improve the HF wave function is based on perturbation theory. The idea is to treat electron correlation as a small perturbation to the Hamiltonian operator. A particularly successful application to molecules and the correlation problem goes back to Møller and Plesset in 1934 [3], and is referred to as the MPn method. The first important correction is the second order term,

which leads to MP2. Perturbation theory has been shown to be very successful in cases where the electron correlation can be approximated as a small perturbation.

Density functional theory (DFT) [4, 5] is structurally similar to the Hartree-Fock method, but includes electron correlation approximately with a functional. The choice of the functional is the main limitation of the DFT method. At the present time, there is no systematic way of choosing the functional. The DFT method consequently requires careful calibration to establish its accuracy on a case by case basis. However, the method scales as the HF method, thus providing a relatively inexpensive route to performing computational physics and chemistry. Although DFT is a ground state theory, time-dependent DFT (TD-DFT) has been developed and is in principle suitable for excited states. First applications have been published relatively recently [6], showing the need to further improve the functionals. For a review of the TD-DFT method see for example reference [7].

The most important post-Hartree-Fock method based on the variational principle is configuration interaction (CI). The CI method has been in practise since the early 1950s [8]. A linear combination of components, each of which represents an excited configuration wave function, is used to provide a better variational solution to the exact many-electron wave function. In principle, by increasing the number of configurations included, the CI wave function is capable of providing arbitrarily accurate solutions to the exact wave function. If all possible excited configurations are included, the method gives the exact solution within the space spanned by a given atomic orbital basis set and is termed full CI (FCI). As already pointed out, the scaling behaviour of the FCI method means that calculations are not computationally tractable for all but the smallest problems. Truncated CI expansions (for instance CISD, which mixes the HF wave function with all singly and doubly excited configurations) are more manageable, but have other disadvantages, like the lack of size-consistency. Other size consistent methods include the quadratic configuration interaction (QCI) method [9] and coupled cluster (CC) methods (for example [10]). For a comprehensive review of *ab initio* methods for electron correlation see references [11, 12].

With the exception of the FCI, all methods mentioned above are adequate, if the HF wave function is a good zero-order approximation. However, wherever bonds are being broken, and for the description of electronically excited states, this is not a valid assumption. The multi-configuration self-consistent-field (MCSCF) method may be applied here, where both the configuration expansion coefficients and the orbitals are variationally optimised in an iterative procedure. In particular, the complete-active-space self-consistent field

(CASSCF) [13] and the restricted-active-space self-consistent-field (RASSCF) [14] variants of the MCSCF method will be considered. In the following two sections we aim to provide a brief outline of the theory of the MCSCF method, and introduce the notation we will use in this thesis, followed by a brief historical context for the work described in the following chapters. For a more comprehensive discussion of the MCSCF method we refer to the literature [15–19].

1.2. An Overview of MCSCF Theory

One of the first steps in most theoretical approaches to the electronic structure of molecules is the use of orbital models. Orbital models such as the Hartree Fock self-consistent-field theory provide an excellent starting point that accounts for most (typically $\geq 99\%$) of the total energy of the molecule. Although at first sight this may seem to be a very good accuracy, unfortunately the error in such orbital theories is of the same order of magnitude as most energies of chemical interest, for example binding energies, ionisation energies and activation barriers. The term *electron correlation energy* is usually defined as the difference between the exact nonrelativistic energy of the system and the HF energy [15]. Electron correlation is critical for the accurate and quantitative evaluation of molecular energies [11, 12, 20]. In actual computations, the orbitals are usually expanded in terms of a finite basis set, i.e. a finite set of atom-centred functions. This introduces an additional error associated with basis set truncation effects. The correlation energy is typically defined within the finite basis set used.

Hartree-Fock self-consistent-field (SCF) theory is a mean field theory, in which each electron has its own wavefunction (orbital), which in turn obeys an effective 1-electron Schrödinger equation. The effective Hamiltonian (Fock operator) contains the *average* field of all other electrons in the system. Thus, instantaneous interactions of any two electrons 1 and 2 are neglected, and the probability density $P(\mathbf{r}_1,\mathbf{r}_2)$ for finding these electrons at positions $\mathbf{r}_1$ and $\mathbf{r}_2$, respectively, is

$$P(\mathbf{r}_1,\mathbf{r}_2) = P(\mathbf{r}_1)P(\mathbf{r}_2). \tag{1.1}$$

This behaviour is obviously unphysical as in reality the effect of Coulomb repulsion means that the existence of electron 1 at position $\mathbf{r}_1$ will define a region in space that electron 2 will avoid. This short-range correlation of the electronic motion is missing in HF theory.

The same mean field ansatz, however, also has detrimental effects on long-range correlation. This is the most important type of correlation effect which contributes to chemical bonding, and is often referred to as *left-right* correlation. For the classic example of the hydrogen molecule, this refers to the tendency that when one electron is near the first hydrogen atom, the other tends to be near the second hydrogen [12, 15]. Again, this cannot be reflected by the HF method where the spatial positions of the two electrons that occupy the lowest bonding molecular orbital are uncorrelated. Particularly for increasing interatomic distances the problem gets worse. However, an improvement to the energy of H_2 can be obtained by including a second electronic configuration, Ψ^*, where both electrons occupy the anti-bonding orbital. The wave function, Φ, may then be written as the linear combination

$$\Phi = C_1\Psi_{\mathrm{HF}} + C_2\Psi^* \tag{1.2}$$

and the coefficients C_1 and C_2 are determined variationally. This is an example of *configuration interaction* (CI). In principle, by increasing the number of configurations included in a CI expansion, the CI method is capable of providing arbitrarily accurate solutions to the exact wave function. If all possible excited configurations are included, the method gives the exact solution within the space spanned by a given basis set and is termed *full configuration interaction* (FCI). Unfortunately, the number of configurations in a FCI expansion grows exponentially with the size of the system, and FCI computations are not practical for many-electron systems with large basis sets. Therefore, some form of truncated CI needs to be employed instead.

Whenever electronically excited states, changing bond orders or delocalised π-systems are considered, the HF method is not sufficient to provide even a qualitatively correct wave function. In all these cases some form of configuration interaction is essential to recover what is termed *non-dynamical* or *static* correlation energy. MCSCF wave functions are used to treat non-dynamical correlation effects in larger molecules. Instead of a single configuration as in the HF method, a relatively small number of selected configurations are used. The expansion coefficients of the configurations are in the MCSCF method optimised together with the orbitals in a variational procedure. Traditionally the configurations included in a MCSCF calculation were selected using chemical insight. However, it is clear that such technique would introduce a bias into the calculation. This bias in the selection of individual configurations can be removed by using the *complete active space* (CAS) SCF approach. Here a set of active orbitals are identified, and all possible configurations of the active electrons in the space of the active orbitals are included in the

MCSCF expansion. All other orbitals are kept doubly occupied or unoccupied, as in the HF calculation. The selection of the active orbitals may still pose a problem, particularly for large molecules. Obviously, if all electrons and orbitals are included in the CAS, the method is identical to FCI. The length of the CASSCF expansion grows exponentially with the number of active orbitals. If no advantage can be taken of spatial symmetry the practical limit currently lies around 12 active orbitals, at least if the number of electrons is similar to the number of orbitals.

Variations of the CASSCF approach have been proposed, in order to overcome this size limit, such as the *restricted active space* (RAS) SCF [14] and, more recently, the *general active space* (GAS) SCF [21, 22] methods. The aim of these methods is to approximate a CASSCF wavefunction by subdividing the active space into groups of orbitals with certain occupancy restrictions, thereby reducing the size of the CI expansion. In the RASSCF method, the active orbitals are subdivided into up to three subsets; one subset contains nearly doubly occupied orbitals, and a minimum number of electrons in this subset is specified. Another subset contains those orbitals that are nearly unoccupied, and a maximum number of electrons in this subset is specified. The third subset comprises the remaining orbitals. The GASSCF method is a generalisation of the same principle, i.e. the number of orbital groups with individual occupancy restrictions is not limited to two. Ideally all configurations with nonzero expansion coefficients in the corresponding CASSCF wavefunction are included. If the restrictions for all orbital subspaces are lifted, both RASSCF and GASSCF are identical to CASSCF.

Dynamical correlation effects may be included in the calculation in a subsequent step, where a multi-reference CI (MRCI) calculation is performed using selected configurations from the CASSCF wave function as reference. The CASSCF calculation is then restricted to include only non-dynamic correlation effects, and is used to determine the orbitals. However, in cases where the shapes of the strongly occupied orbitals depend on dynamical correlation effects, those orbitals need to be included in the active space. Examples for such cases are anions, polar bonds and excited states [23]. Orbital optimisation under inclusion of dynamical correlation is thus the only consistent approach in these cases. This is another useful application of RASSCF or GASSCF methods.

1.2.1. SCF Theory

The object of most quantum mechanical methods is to approximate solutions to the non-relativistic time independent Schrödinger equation,

$$\hat{H}\Psi = E\Psi, \tag{1.3}$$

where Ψ is the wave function, E is the energy of the system, and $\hat{H}$ is the electronic Hamiltonian operator for N electrons,

$$\hat{H} = \sum_{i=1}^{N} \hat{h}(i) + \sum_{i \neq j}^{N} \hat{g}(i,j). \tag{1.4}$$

In atomic units $\hat{h}(i)$ is a one electron operator for electron i,

$$\hat{h}(i) = -\frac{1}{2}\nabla_i^2 - \sum_A \frac{Z_A}{r_{iA}} \tag{1.5}$$

and $\hat{g}(i,j)$ is the electrostatic interaction between electrons i and j, which simply has the form

$$\hat{g}(i,j) = \frac{1}{r_{ij}}, \tag{1.6}$$

where r_{ij} is the separation between the electrons. It is not possible to solve equation 1.3 analytically, and a suitable approximation must be applied.

At the heart of the HF SCF method [24, 25] lie two approximations, which are essential for making electronic structure computations tractable. The Born-Oppenheimer approximation [26] decouples the nuclear motion from those of the electrons in a molecule. This is usually valid due to the much higher mass of the nuclei . The Hartree approximation is more severe, as it replaces the remaining N-electron problem with N 1-electron problems. The two-body Coulomb repulsion is thus replaced by the interaction of an electron with the "average field" of the second electron, and *vice versa*. The resulting one-electron wave functions are also referred to as *orbitals*.

The success of the HF method [24, 25] in describing the electronic structure of most closed-shell molecules has made it natural to analyse the wave function in terms of molecular orbitals. It is convenient to expand the molecular orbitals (MOs) in a set of *basis wave*

functions (atomic orbitals, AOs), $\{\chi_i\}$,

$$\psi_i = \sum_{\mu=1}^{\Lambda} c_{\mu i}\chi_\mu, \tag{1.7}$$

where Λ is the number of basis functions, and the $c_{\mu i}$ are the orbital coefficients. The AO functions are usually combinations of Gaussian functions. The HF wavefunction, Ψ_{HF}, is then constructed as an anti-symmetrised product of the orbitals, usually in the form of a *Slater determinant* (SD). The general form of a SD for an N-electron system is

$$\begin{aligned}\Psi_{\mathrm{HF}}(\mathbf{x}_1,\mathbf{x}_2,\cdots,\mathbf{x}_N) &= (N!)^{-1/2}\begin{vmatrix} \psi_i(\mathbf{x}_1) & \psi_j(\mathbf{x}_1) & \cdots & \psi_k(\mathbf{x}_1) \\ \psi_i(\mathbf{x}_2) & \psi_j(\mathbf{x}_2) & \cdots & \psi_k(\mathbf{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \psi_i(\mathbf{x}_N) & \psi_j(\mathbf{x}_N) & \cdots & \psi_k(\mathbf{x}_N) \end{vmatrix} \\ &= |\psi_i(\mathbf{x}_1)\psi_j(\mathbf{x}_2)\cdots\psi_k(\mathbf{x}_N)\rangle, \end{aligned} \tag{1.8}$$

where we also introduce the short-hand notation. The $\psi(\mathbf{x}_1)$ are spin wave functions of electron 1 with space- and spin-coordinates $\mathbf{x}_1 = \{\mathbf{r}_1, \omega\}$, where ω signifies spin and takes the values α or β.

According to the variational theorem, the best wave function of this form is found by minimising the energy from equation 1.3 with respect to the orbitals. The best orbitals for this wave function have been shown to be eigenfunctions of the Fock operator

$$\hat{f}(1) = \hat{h}(1) + \sum_{i=1}^{M}\left(\hat{J}_i(1) - \hat{K}_i(1)\right), \tag{1.9}$$

where the summation is over all occupied spatial orbitals. The Coulomb operator, $\hat{J}_i(1)$, and exchange operator, $\hat{K}_i(1)$, are defined as

$$\langle\psi_k(\mathbf{x}_1)|\hat{J}_i(1)|\psi_l(\mathbf{x}_1)\rangle = \int \frac{\psi_k^*(\mathbf{x}_1)\psi_i^*(\mathbf{x}_2)\psi_l(\mathbf{x}_1)\psi_i(\mathbf{x}_2)}{r_{12}} d\mathbf{x}_1 d\mathbf{x}_2 \tag{1.10}$$

$$\langle\psi_k(\mathbf{x}_1)|\hat{K}_i(1)|\psi_l(\mathbf{x}_1)\rangle = \int \frac{\psi_k^*(\mathbf{x}_1)\psi_i^*(\mathbf{x}_2)\psi_i(\mathbf{x}_1)\psi_l(\mathbf{x}_2)}{r_{12}} d\mathbf{x}_1 d\mathbf{x}_2 \tag{1.11}$$

These integrals are often further abbreviated to

$$\langle\psi_k(\mathbf{x}_1)|\hat{J}_i(1)|\psi_l(\mathbf{x}_1)\rangle = (kl|ii) \tag{1.12}$$

$$\langle\psi_k(\mathbf{x}_1)|\hat{K}_i(1)|\psi_l(\mathbf{x}_1)\rangle = (ki|il) \tag{1.13}$$

Equation 1.9 represents the exact inter-electron repulsion, with each electron reacting to a field due to the average location of all the other electrons. The eigenfunctions of equation 1.9 must be found iteratively, since the Coulomb and exchange operators depend on the eigenfunctions. This is done by varying the expansion coefficients, $c_{\mu i}$ (cf. equation 1.7).

The two fundamental building blocks of HF theory are the molecular orbital and its occupation number. In closed shell systems, each occupied molecular orbital carries two electrons, each with opposite spin. The HF method can be extended to open-shell systems in two ways. In the restricted Hartree Fock (RHF) method [27], the open-shell orbitals are added to the closed-shell orbitals and the resulting wavefunction is then projected to have the correct spin and space symmetry. This may lead to a wavefunction comprising more than one Slater determinant. An alternative formulation is the unrestricted Hartree Fock (UHF) method [28], where the spin orbitals for a closed-shell electron pair are no longer assumed to be equal. In contrast to the closed-shell HF method, the UHF method gives a qualitatively correct description of bond dissociation [29]. However, it is known that the potential energy surfaces obtained with the UHF method lack accuracy [12]. All variants of the HF method lead to a wavefunction in which all the information about the electron structure is contained in the occupied molecular orbitals (or spin orbitals) and their occupation numbers, the latter being equal to 1 or 2.

The concept of the molecular orbital is, however, not restricted to the HF model. A set of orbitals can also be constructed for more complex wavefunctions, which include correlation effects. They can be used to obtain insight into the detailed features of the electronic structure. These are the natural orbitals, which are obtained by diagonalising the spinless first-order reduced density matrix [30]. The occupation numbers (η) of the natural orbitals are not restricted to 2, 1 or 0. Instead they fulfil the condition

$$0 < \eta < 2. \tag{1.14}$$

1.2.2. Configuration Interaction

The SCF method is best suited for describing the ground states of systems near the equilibrium geometry. It is unable, however, to describe excited states or potential energy surfaces where the overall wavefunction needs to be represented by more than a single determinant. It is also problematic with open shell systems. To further extend the theory it is usual to expand the wave function as a linear combination of the set of the other determinants that can be formed by formally "exciting" electrons from occupied (core)

MOs into unoccupied (virtual) MOs in the HF determinant, i.e.

$$\Phi_{\mathrm{CI}} = C_{\mathrm{HF}}|\Psi_{\mathrm{HF}}\rangle + \sum_a \sum_r C_a^r |\Psi_a^r\rangle + \sum_a \sum_b \sum_r \sum_s C_{ab}^{rs} |\Psi_{ab}^{rs}\rangle + \cdots . \tag{1.15}$$

The "excited" determinants are referred to as *configuration state functions* (CSFs) or simply *configurations*, and the method is termed configuration interaction (CI). The wave function is then variationally optimised with respect to the coefficients of the linear expansion of the configurations, C, by forming the matrix elements of the Hamiltonian,

$$\langle \Psi_K | \hat{H} | \Psi_L \rangle = \langle K | \hat{H} | L \rangle = H_{KL} \tag{1.16}$$

and diagonalising the matrix. The labels K, L have been used to describe arbitrary configurations. There are rules to determine the matrix elements of equation 1.16, the most well known of which are Slater's rules, which apply to Slater determinants (see Ref. [15]). The eigenvalues formed are upper bounds to the energies of the ground and excited states. It can be shown that if all possible configurations are included, the basis set limit is reached. This limit is the best wave function that can be formed with this basis. For an infinitely flexible and complete basis the limit becomes the exact solution to the Schrödinger equation, and the choice of orbitals in the reference configuration becomes irrelevant. Unfortunately even a small basis will produce far too many configurations to be handled, so a prescription for truncating the space of determinants must be used. However, this is not trivial: while there may be a few configurations that have major contributions to the wave function, the overall effect of the rest is often still quite large in magnitude.

It transpires that SDs are not the best choice for the CSF basis. The reason for this is that, although they are eigenfunctions of $\hat{S}_z$ with eigenvalues $\frac{1}{2}(N_\alpha + N_\beta)$, where N_α and N_β are the numbers of electrons with α and β spin, they are not generally eigenfunctions of $\hat{S}^2$. However, linear combinations can be found that are. So although one can limit the SD basis to those that have a certain eigenvalue of $\hat{S}_z$, defining a particular spin state is not possible, which makes convergence of the CI poor. In the non-relativistic theory, the Hamiltonian does not contain any spin coordinates, i.e. both $\hat{S}^2$ and $\hat{S}_z$ commute with it. Therefore, the exact eigenfunctions of the non-relativistic Hamiltonian should also be eigenfunctions of the spin operators. A basis that reflects this property is "spin adapted", and by choosing only CSFs that correspond to the desired spin state one will have the smallest possible basis [31].

1.2.3. The Second Quantisation Formalism

The CI expansion can be made computationally simpler by writing the matrix elements of the Hamiltonian in *second quantised* form [15, 16, 31]. This regards the one- and two-electron integrals as parameters that will be present or absent in a matrix element according to *Fock space* operators, which depend on the occupation numbers of orbitals in the basis functions. This approach is therefore sometimes called the Fock space approach or the occupation number representation. We have already introduced the ket vector notation for a Slater determinant (equation 1.8) and we will from now use the even more compact notation

$$\begin{aligned} |K\rangle &= |\psi_i\psi_j\cdots\psi_p\rangle \\ &= |ij\cdots p\rangle. \end{aligned} \tag{1.17}$$

A Fock space contains all the vectors

$$|\rangle, |i\rangle, |ij\rangle, \ldots, |ij\cdots p\rangle, \ldots \tag{1.18}$$

where $|\rangle$ is the "vacuum" containing no particles. This is a graded *Hilbert space*, where a Hilbert space is a series of vectors as in equation 1.18, corresponding to a certain number of electrons. So $|\rangle$ is a 0 electron Hilbert space, $|i\rangle$, $|j\rangle$ are 1 electron spaces, etc. Now we can introduce the operators that link the Hilbert spaces within a Fock space. The first of these is the creation operator $\hat{a}_r^+$ that adds a particle if the ket it operates upon does not contain r. If r is already present, the determinant vanishes (satisfying the Pauli principle: two electrons in the same spin orbital are not allowed). The effect of the creation operator on the vacuum is simply

$$\hat{a}_i^+|\rangle = |i\rangle, \tag{1.19}$$

while acting on any other n-electron Hilbert space ($n > 0$) results in

$$\hat{a}_r^+|ij\cdots p\rangle = \begin{cases} |ij\cdots pr\rangle & r \notin \{i,j,\ldots,p\} \\ 0 & r \in \{i,j,\ldots,p\} \end{cases} \tag{1.20}$$

It is clear that

$$\hat{a}_j^+|i\rangle = |ij\rangle, \quad \hat{a}_i^+|j\rangle = |ji\rangle = -|ij\rangle \tag{1.21}$$

so the wave function is still antisymmetric to the permutation of indices. From equations 1.19 and 1.21 we can see that

$$\hat{a}_i^+ \hat{a}_j^+ |\rangle = |ji\rangle = -|ij\rangle = -\hat{a}_j^+ \hat{a}_i^+ |\rangle \tag{1.22}$$

so the creation operator can be said to "anti-commute",

$$\hat{a}_i^+ \hat{a}_j^+ + \hat{a}_j^+ \hat{a}_i^+ = \{\hat{a}_i^+, \hat{a}_j^+\} = 0. \tag{1.23}$$

The annihilation operator $\hat{a}_r^-$ can be defined similarly:

$$\begin{array}{rcl} \hat{a}_r^- |ij\cdots pr\rangle & = & |ij\cdots p\rangle \\ \hat{a}_r^- |ij\cdots p\rangle & = & 0. \end{array} \tag{1.24}$$

If r is contained within the ket, but not at its end, we need to add a factor,

$$\hat{a}_r^- |ij\cdots p\rangle = \begin{cases} (-1)^{\nu_r} |ij\cdots \not{r} \cdots p\rangle & r \in \{i,j,\ldots,p\} \\ 0 & r \notin \{i,j,\ldots,p\} \end{cases} \tag{1.25}$$

where ν_r is the number of permutations required to bring r to the end (cf. equation 1.21). Following the same logic as for $\hat{a}_r^+$ we can write

$$\{\hat{a}_i^-, \hat{a}_j^-\} = 0. \tag{1.26}$$

The combined operator $\hat{a}_s^+ \hat{a}_r^-$ changes an occupied spin orbital from r into s by

$$\begin{array}{rcl} \hat{a}_s^+ \hat{a}_r^- |ij\cdots r\cdots p\rangle & = & \hat{a}_s^+ \hat{a}_r^- (-1)^{(\nu_r)} |ij\cdots pr\rangle \\ & = & \hat{a}_s^+ (-1)^{(\nu_r)} |ij\cdots p\rangle \\ & = & (-1)^{(\nu_r)} |ij\cdots ps\rangle \\ & = & (-1)^{(\nu_r)} (-1)^{(\nu_s)} |ij\cdots s\cdots p\rangle \\ & = & (-1)^{(\nu_r+\nu_s)} |ij\cdots s\cdots p\rangle. \end{array} \tag{1.27}$$

The phase factor $(-1)^{\nu_r}$ in equation 1.27 is multiplied by a second, identical phase factor that moves s from the end of the list to the position previously occupied by r, and clearly

$(-1)^{2\nu_r} = 1$. If $r = s$ we have

$$\hat{a}_r^+ \hat{a}_r^- |ij\cdots p\rangle = \begin{cases} |ij\cdots p\rangle & r \in \{i,j,\ldots,p\} \\ 0 & r \notin \{i,j,\ldots,p\} \end{cases} \tag{1.28}$$

$$\hat{a}_r^- \hat{a}i_r^+ |ij\cdots p\rangle = \begin{cases} |ij\cdots p\rangle & r \notin \{i,j,\ldots,p\} \\ 0 & r \in \{i,j,\ldots,p\} \end{cases} \tag{1.29}$$

So $(\hat{a}_r^+ \hat{a}_r^- + \hat{a}_r^- \hat{a}_r^+)$ is the unit operator for *any* ket. The anti-commutation relations for these operators can be generalised as

$$\{\hat{a}_r^+, \hat{a}_s^-\} = \delta_{rs}. \tag{1.30}$$

The requirement that the basis needs to be normalised means that we can express the creation and annihilation operator in another way:

$$\begin{aligned} \left\langle \hat{a}_r^+ |ij\cdots p\rangle \left| \hat{a}_r^+ |ij\cdots p\rangle \right.\right\rangle &= \langle ij\cdots p| (\hat{a}_r^+)^\dagger \hat{a}_r^+ |ij\cdots p\rangle \\ &= 1 \end{aligned} \tag{1.31}$$

(provided r is absent from the ket). This means that $(\hat{a}_r^+)^\dagger$ is reversing the effect of $\hat{a}_r^+$, which is the definition of $\hat{a}_r^-$. So in an orthonormal Fock space, the adjoint of the creation operator $\hat{a}_r^+$ equals the annihilation operator $\hat{a}_r^-$, and the operators can be written as:

$$\begin{aligned} \hat{a}_r^\dagger &= \hat{a}_r^+ \\ \hat{a}_r &= \hat{a}_r^- \end{aligned} \tag{1.32}$$

Now we are ready to express the Hamiltonian (cf. equation 1.4) in terms of these operators:

$$\hat{H} = \sum_{ij} \langle i|\hat{h}|j\rangle \hat{a}_i^\dagger \hat{a}_j + \frac{1}{2} \sum_{ijkl} (ij|kl) \hat{a}_i^\dagger \hat{a}_j^\dagger \hat{a}_l \hat{a}_k. \tag{1.33}$$

This is simply shown to be equivalent to equation 1.4 by writing out a matrix element of the second quantisation Hamiltonian in occupation number formalism. First the one electron operator

$$\begin{aligned} \langle \cdots i \cdots | \sum_{rs} \hat{h}_{rs} \hat{a}_r^\dagger \hat{a}_s | \cdots j \cdots \rangle &= \sum_{rs} \hat{h}_{rs} \langle \cdots i \cdots | \hat{a}_r^\dagger \hat{a}_s | \cdots j \cdots \rangle \\ &= \sum_{rs} \hat{h}_{rs} \delta_{ir} \delta_{js}, \end{aligned} \tag{1.34}$$

and then the two electron operator

$$\begin{aligned}\langle\cdots i\cdots k\cdots|\sum_{rstu}\hat{g}_{rstu}\hat{a}_r^\dagger\hat{a}_s^\dagger\hat{a}_u\hat{a}_t|\cdots j\cdots l\cdots\rangle &= \sum_{rstu}\hat{g}_{rstu}\langle\cdots i\cdots k\cdots|\hat{a}_r^\dagger\hat{a}_s^\dagger\hat{a}_u\hat{a}_t|\cdots j\cdots l\cdots\rangle \\ &= \sum_{rstu}\hat{g}_{rstu}\delta_{ir}\delta_{ks}\delta_{uj}\delta_{tl}. \qquad (1.35)\end{aligned}$$

We note that the only non-vanishing contributions arise where the CSFs on either side are identical, except for the orbitals that coincide with the operator indices. Those orbitals need to be occupied as prescribed by the operator. The explicit summation over the number of electrons in equation 1.4 has been removed in equation 1.33, so that the Hamiltonian is now a general many body operator, and no longer depends on the number of electrons in the system. The emphasis has been moved from the determinants to the terms consisting of creation and annihilation operators.

Thus far indices i referred to spin orbitals, and the creation and annihilation operators were correspondingly indexed. In electronic structure problems where the number of particles is conserved, creation and annihilation operators will always occur in equal numbers. Furthermore, if $\hat{S}_z$ is constant for the configurations of interest, then the number of electrons with α-spin and β-spin, respectively, will be constant, too. Replacing creation and annihilation operators $\hat{a}_i^\dagger$, $\hat{a}_j$ by $\hat{a}_{i\sigma}^\dagger$, $\hat{a}_{j\gamma}$, where the second subscripts indicate the spin state (α, β), we may now rewrite equation 1.33 as

$$\hat{H} = \sum_{ij}(i|j)\sum_{\sigma}\hat{a}_{i\sigma}^\dagger\hat{a}_{j\sigma} + \frac{1}{2}\sum_{ijkl}(ij|kl)\sum_{\sigma\gamma}\hat{a}_{i\sigma}^\dagger\hat{a}_{k\gamma}^\dagger\hat{a}_{l\gamma}\hat{a}_{j\sigma}. \qquad (1.36)$$

1.2.4. MCSCF Theory

Multi-configuration self-consistent-field theory is a powerful tool for exploring the potential energy surfaces of molecules. It is particularly suited to studying reactions on the ground and excited states. In an MCSCF calculation there are two problems that are solved simultaneously. The CI expansion coefficients, $\{C_K\}$, are determined by diagonalising a CI Hamiltonian constructed from symbolic matrix elements and MO integrals. The SCF part optimises the MO coefficients, $\{c_{\mu i}\}$, by diagonalising a Fock matrix. Since the solution to each part depends on the other, the process is an iterative one, repeated until the eigenvalues converge. In this thesis we are concerned only with the main computational bottleneck of a large scale MCSCF computation, namely the diagonalisation of the Hamiltonian matrix.

MCSCF theory can be thought of as a truncated CI calculation or as an SCF calculation that is not restricted to one determinant. Some form of CI calculation is performed optimising both the CI coefficients, C_K, and the MO coefficients contained in each configuration Ψ_K,

$$\Phi_{\mathrm{MCSCF}} = \sum_K C_K \Psi_K. \tag{1.37}$$

It is clear that if there is only one term in equation 1.37, then this method and HF theory are identical. Since the CI will be truncated in some way, MCSCF provides the best orbitals for the CI expansion, which improves the convergence of the CI. The two formulations of interest to this thesis are the complete active space SCF (CASSCF) and the restricted active space SCF (RASSCF) theories. CASSCF will be treated in chapter 2 and RASSCF in chapter 3. Both of these theories perform some form of CI within a subset of all the MOs. The remaining orbitals are either "core", i.e. doubly occupied in all configurations, or "virtual", i.e. unoccupied in all configurations.

Using the second quantised formalism introduced in the previous section, the matrix elements of the matrix representation of the MCSCF Hamiltonian operator can be written as

$$H_{KL} = \delta_{KL}\varepsilon_c + \sum_{ij}\langle i|\hat{h}_c|j\rangle A_{ij}^{KL} + \frac{1}{2}\sum_{ijkl}(ij|kl)B_{ijkl}^{KL}, \tag{1.38}$$

where ε_c is the SCF energy of the core orbitals and $\hat{h}_c$ is the one electron HF operator within the field of the core orbitals,

$$\langle i|\hat{h}_c|j\rangle = \sum_a^{\mathrm{Core}}(aa|ij) - \frac{1}{2}(ai|aj). \tag{1.39}$$

A_{ij}^{KL} and B_{ijkl}^{KL} are the symbolic matrix elements – the elements of the matrices that correspond to the creation and annihilation operators (cf. equation 1.33),

$$A_{ij}^{KL} = \langle K|\sum_\sigma \hat{a}_{i\sigma}^\dagger \hat{a}_{j\sigma}|L\rangle \tag{1.40}$$

$$B_{ijkl}^{KL} = \langle K|\sum_{\sigma\gamma} \hat{a}_{i\sigma}^\dagger \hat{a}_{k\gamma}^\dagger \hat{a}_{l\gamma}\hat{a}_{j\sigma}|L\rangle. \tag{1.41}$$

When the CSFs are just Slater determinants, these matrix elements are the same simple numerical coefficients (0, ± 1) of Slater's rules. It is clear that the symbolic matrix elements depend only on the configuration space, and so can be computed knowing only the

nature of the CI expansion.

Once the matrix representation of the Hamiltonian of the CSF basis is formed, it must be diagonalised to obtain the eigenvalues (energies) and eigenvectors. There are many methods of diagonalising matrices, the important details of two are presented here. The first is the full diagonalisation, or Jacobi method. This requires that the entire matrix must be explicitly assembled before the off-diagonal elements are reduced to zero by iterative operation of plane rotations. The advantages of this method are that it is simple, and that it gives the complete list of the eigenvalues and eigenvectors. The disadvantage is that there must be sufficient machine memory to store the entire $n_{\text{confs}} \times n_{\text{confs}}$ elements of the matrix. Even if the symmetry of the Hamiltonian is exploited and only one triangle of the matrix is stored there still needs to be $n_{\text{confs}} \times (n_{\text{confs}}+1)/2$ machine words of space. Since the number of configurations increases rapidly with the number of valence orbitals, this will be impractical for even modest active spaces.

The second procedure is the Lanczos tri-diagonalisation method [32]. The algorithm can be summarised as

$$\mathbf{H}\mathbf{C}^{(n)} = \gamma_n \mathbf{C}^{(n+1)} + \alpha_n \mathbf{C}^{(n)} + \beta_{n-1}\mathbf{C}^{(n-1)}. \tag{1.42}$$

The Hamiltonian is operated on a trial vector $\mathbf{C}$, producing a sum of three vectors, scaled by the constants α, β and γ. This sum is easily resolved into its components to give the constants, which form the elements of the tri-diagonal representation of the Hamiltonian, and a new trial vector. The procedure is then repeated on this new vector. After n iterations, estimates for the lowest n eigenvalues of the Hamiltonian can be obtained from the tridiagonal matrix. It is found that the lowest eigenvalues converge fastest, so the procedure is continued until the selected number of eigenvalues has converged. The eigenvectors corresponding to these eigenvalues can be back-transformed from the list of the vectors produced at each iteration.

The advantages of this method are that one can use the Hamiltonian matrix implicitly – when it operates on the vector, it is clear that the individual multiplications that comprise a matrix vector multiplication can be performed in any order, and the matrix never needs to be explicitely assembled, so all that is necessary is a list of the matrix elements. Convergence depends on how close the trial vector is to the eigenvector sought. Failure to converge is very rare, but may occur if the final eigenvector is deficient in the configuration(s) present in the trial vector. The procedure only requires enough memory to store a few vectors of length n_{confs}, and enough disk space to store the trial vectors and the elements of the tri-diagonal form. Therefore, this is the algorithm of choice for large

configuration space calculations and has proved capable of diagonalising matrices with hundreds of thousands of configurations.

In a traditional implementation of the Lanczos method the matrix elements had to be read from a file for every iteration of the Lanczos step. However, disk access is on modern computers several orders of magnitude slower than CPU speed, and it proves more efficient to recompute the elements of the Hamiltonian matrix when they are needed. This strategy is termed *direct CI* [33].

CASSCF Theory

Complete active space SCF (CASSCF) is an MCSCF calculation where the CI calculation is a full CI amongst a subset of the MO space. These orbitals are the *active* or *valence* orbitals. The choice of these orbitals is clearly important. Active orbitals are those that will change occupancy significantly during the process under investigation, or those that can only be described by a multi-referential method, such as resonant bonds. The CASSCF method has several desirable properties, due to the full CI character. The first is that the energy is invariant to any unitary transformation of the active orbitals. This means, for example, that the orbitals could be localised for some form of valence bond analysis after the MCSCF optimisation. Another is that one obtains a set of eigenvalues that will correspond well to the excited states of the system. Conversely the size of the CI expansion will increase astronomically as the number of active orbitals increases. The number of Slater determinants in a full CI is

$$n_{\text{confs}} = \binom{M}{N_\alpha} \times \binom{M}{N_\beta}, \tag{1.43}$$

where M is the number of active orbitals, and N_α and N_β are the number of active electrons with α-spin and β-spin, respectively. If spin adapted configurations are being used, the number of configurations is given by the Weyl dimension formula [16],

$$n_{\text{confs}} = \frac{2S+1}{M+1}\binom{M+1}{\frac{N}{2}+S+1}\binom{M+1}{\frac{N}{2}-S}, \tag{1.44}$$

where S is the total spin, and N is the total number of active electrons. The quantities in parentheses are binomial coefficients, i.e.

$$\binom{M}{N} = \frac{M!}{N!(M-N)!}. \tag{1.45}$$

RASSCF Theory

Restricted active space SCF (RASSCF) is similar to CASSCF in that one also defines a set of active orbitals, and a CI is carried out within this orbital space. However, the active orbitals are subdivided into three subsets, RAS1, RAS2 and RAS3. Occupancy restrictions are imposed on the RAS1 and RAS3 subspaces, so that a given maximum number of "holes" (empty spin orbitals) in RAS1, and a maximum number of electrons in RAS3 cannot be exceeded. This restriction generally results in a large reduction of the number of configurations. The number of Slater determinants in a RASSCF is

$$n_{\text{confs}} = \sum_{i_h=0}^{\text{MxHole}} \sum_{i_e=0}^{\text{MxElec}} \binom{M(\text{RAS1})}{i_h} \binom{M(\text{RAS2})}{N_\alpha - M(\text{RAS1}) + i_h - i_e} \binom{M(\text{RAS3})}{i_e} \sum_{i'_h=0}^{\text{MxHole}-i_h} \sum_{i'_e=0}^{\text{MxElec}-i_e} \binom{M(\text{RAS1})}{i'_h} \binom{M(\text{RAS2})}{N_\beta - M(\text{RAS1}) + i'_h - i'_e} \binom{M(\text{RAS3})}{i'_e}, \tag{1.46}$$

where $M(\text{RAS}X)$ is the number of orbitals within the RASX subspace ($X \in \{1,2,3\}$), MxHole and MxElec are the maximum number of holes in RAS1 and electrons in RAS3, respectively. It has been shown [14] that RASSCF has the potential to successfully approximate the CASSCF wave function, if the occupancy restrictions imposed on the RAS1 and RAS3 subspaces are chosen sensibly.

1.3. Motivation for this Thesis

We will now outline the motivation for the work carried out as part of this thesis. We start by providing a brief historical background of the problem addressed by us (for a comprehensive review of the history and evolution of CI, see references [8, 34, 35]).

The central technical problem in large scale CI (and MCSCF) problems is the evaluation of the matrix elements of the Hamiltonian matrix, H_{KL} (cf. equation 1.38). The symbolic matrix elements A_{ij}^{KL} and B_{ijkl}^{KL} depend on the nature of the CSFs, $\{|K\rangle\}$, where the symbol $|K\rangle$ stands for both a string of occupied orbitals in the CSF, and a sequence number or index. In the FCI (and CASSCF) method, the list of CSFs becomes very large, and the indexing becomes a major problem. Historically, there was no general solution to this problem, and in early implementations the list of CSFs was simply generated in some way, and written to a file. This file was then processed, comparing two configurations at

a time to generate the A_{ij}^{KL} and B_{ijkl}^{KL}, which were written to the so-called symbolic matrix element file. This file was then used to compute the Hamiltonian matrix elements,

$$H_{KL} = \sum_{ij}(i|j)A_{ij}^{KL} + \frac{1}{2}\sum_{ijkl}(ij|kl)B_{ijkl}^{KL}. \tag{1.47}$$

Even for fairly moderate problem sizes the symbolic matrix element file was unmanageably large. The solution to this problem was the idea of a *direct CI*, introduced by Roos in 1972 [33]. In direct CI the symbolic matrix elements get recomputed "on the fly", i.e. whenever they are needed, and storage is entirely avoided. However, the original direct CI formalism was developed for all single and double excitations of a single reference (CISD) only, and the computation of the symbolic matrix elements for FCI was so costly, that computing the matrix elements "on the fly" was not possible. No advantage was taken of the extreme sparsity of the Hamiltonian matrix [36].

A solution to this problem was first suggested by Paldus [37]. He realised that a more general formalism previously developed by Gelfand and Tsetlin [38] for the theory of the unitary group, takes a particularly simple form when applied to the N-electron problem. The resulting scheme comprises a prescription for constructing the CSFs in a well defined way, with inherent "lexical" ordering, and one-to-one correspondence between a CSF, $|K\rangle$, and its index, Addr$\{K\}$. In Paldus' scheme, all the information that is required about a CSF, and to establish an algorithm for the implicit representation of the basis, is a sequence of step vectors. A step vector is a string of M integers (steps), one for each molecular orbital of the orbital basis. The possible step values are 0, 1, 2 and 3, representing the addition of no electron, one electron (spin up), one electron (spin down) and two electrons, respectively, to the corresponding orbital.

The final break-through came when Shavitt proposed a graphical representation of Paldus' scheme (GUGA) [39, 40]. In the Shavitt graph, each step vector is associated with a continuous line ("walk") on a grid. Starting at the bottom of the graph, the walk advances one level upwards with every element of the step vector. The four possible step values (0, 1, 2, 3) are represented by four different slopes. All step vectors of a given basis start and finish in the same points, and when constructed using Paldus' scheme are ordered strictly left to right.

Shavitt noticed that non-vanishing symbolic matrix elements A_{ij}^{KL} and B_{ijkl}^{KL} correspond to closed loops on the graph. These loops are created by the graphical representation of the step vectors of the two CSFs $|K\rangle$ and $|L\rangle$ on the same graph. He further realised that the

values of matrix elements correspond to the shape and position of the associated loop. Finally, he found that these loop values could be obtained as a product of the values associated with the "segments" of the corresponding loop (each segment corresponding to one step). A further advantage of the loop structure comes from the fact that many symbolic matrix elements share the same loop (and thus also the same numerical value), and differ only in the "upper" and "lower" walks on the graph, i.e. the connection between top of the loop to top of the graph and from bottom of the loop to bottom of the graph. Therefore, the numerical value of the loop needs to be computed only once, and its contribution to all the matrix elements to which it contributes could be evaluated immediately by "following" the upper and lower walks. Following this development, "loop" or "shape" driven CI codes were developed [41], where all the matrix elements could be evaluated "on the fly" without the need of storing them in a file.

The final, and for the context of this thesis crucial development to be included in this brief historical background, came in 1984 with the contribution of Knowles and Handy [42]. They chose to express the CSFs as combinations of ordered "strings" of occupied orbitals of one spin type (α-strings and β-strings). Their stated main objective was to eliminate the remaining bottleneck due to the long list of coupling coefficients A_{ij}^{JK} in the term $A_{kl}^{KI}\sum_{ij}\sum_{J}A_{ij}^{JK}$, which had been proposed by Siegbahn [43] for efficient use of vector computers. This long list of coupling coefficients meant that demand on input/output facilities was very high, considerably slowing down the computation. Using blocks of strings on the other hand considerably reduced the memory reqirements, resulting in much improved efficiency. However, expressing the CSFs in terms of α-strings and β-strings has the additional advantage that the evaluation of the symbolic matrix elements is considerably simplified. Efficient string-based algorithms that take advantage of this fact have since been reported and are widely used (for instance [14, 44, 45]).

The CASSCF method is widely used and, although computationally expensive, is often the method of choice, for example to study electronically excited states. Because of the high cost of CASSCF computations, efficient programs that are able to effectively utilise current hardware, are needed. As the problem sizes in CASSCF computations grow, the Hamiltonian matrix becomes more and more sparse, and even efficient string-based approaches eventually become wasteful when typical index-testing algorithms are used to evaluate the symbolic matrix elements. In chapter 2 we present a new, efficient approach to solving the computationally most demanding part of of the CASSCF method. In part resembling the FCI method, the CASSCF method scales factorially, and the contribution of this thesis is the development and implementation of an algorithm that avoids the

redundancies present in existing CASSCF programs. This is achieved by taking full advantage of the extreme sparsity of the Hamiltonian matrix, which is done by introducing *reduced lists* (compressed excitation lists).

Modern supercomputer architectures are mostly massively parallel machines. In order to take full advantage of these computers, efficient parallel implementations of the CASSCF and RASSCF method need to be designed to maximise parallel efficiency and minimise overhead of the parallelisation. We describe an efficient parallelisation procedure of the CASSCF algorithm, designed to take full advantage of modern parallel computers, supporting both shared memory and distributed memory architectures.

Where CASSCF calculations are too costly, the RASSCF method may be used as an approximation. Other possible areas for applications of the RASSCF method are electronic structure problems where the shapes of the valence orbitals depend on dynamical correlation effects. However, the subdivision of the active orbital space means that the indexing problem becomes much more complicated, and an efficient algorithm is needed. In chapter 3 we develop a RASSCF algorithm, again by introducing a new approach to solving the computational bottleneck in this method. Our main contribution to the RASSCF method is the rationalisation of the structure of the Hamiltonian matrix into simple algebraic rules. These *propagation rules* correspond to certain blocks of non-vanishing contributions in the Hamiltonian matrix. Individual blocks are found efficiently using a *RAS model space*. The propagation rules are based on a graphical representation of spin strings. The method is implemented and parallelised, and first applications are presented and briefly discussed.

Some technical details of the implementation of both presented methods can be found in the appendix.

2. The Direct CASSCF Method

2.1. Introduction

In the CASSCF method [46], given a set of active orbitals, a full configuration interaction (FCI) calculation is carried out in the CI space constructed using all the active electrons. The CI eigenvector is usually computed at each iteration of the iterative MCSCF process. Since the Hamiltonian matrix, **H**, is too large to be stored in memory the appropriate matrix elements are recomputed each time they are needed in the eigensolver iteration (*direct* methods). However, since the size of the FCI grows rapidly with the number of active orbitals, the eigenvector computation time will eventually become the computational bottleneck. While many elegant and efficient algorithms have been devised and implemented for the FCI part of large active space CASSCF (for example the work by Siegbahn [43], Knowles and Handy [42], Olsen *et al.* [14], Zarrabian *et al.* [44] and others), only a few attempts have been made to design methods for scalable massively parallel computing architectures (for example the work by Rossi *et al.* [47] and Stephan and Wenzel [48]). Most attempts at parallelisation have focused on the linear algebra rather than the CI matrix element problem *per se*. In this chapter we present a parallel, *direct reduced list* method for integral-driven full CI algorithms used in CASSCF that is efficient and fully parallelised, supporting distributed memory as well as shared memory architectures and hybrids.

In section 2.2 we develop the formalism on which our algorithm is based. In section 2.5 we describe the parallelisation supporting distributed and/or shared memory architectures. Speedup and scalability are demonstrated in section 2.6 with the help of a few examples.

2.2. Theory

Large-scale configuration interaction (CI) calculations require the computation of only a few eigenvalues and eigenvectors of large, real-symmetric matrices. If the methods of Lanczos [32] or Davidson [49] are used as the iterative eigenvector procedure, then the time-consuming step is the construction of the CI vector $\boldsymbol{\sigma}$, using

$$\boldsymbol{\sigma}_K = \sum_L H_{KL} C_L, \tag{2.1}$$

where $\mathbf{C}$ is an approximate eigenvector from the previous iteration. The Hamiltonian matrix elements, H_{KL} can be expressed as

$$H_{KL} = \sum_{ij} (i|j) A_{ij}^{KL} + \frac{1}{2} \sum_{ijkl} (ij|kl) B_{ijkl}^{KL}, \tag{2.2}$$

where $(i|j)$ and $(ij|kl)$ are the usual one and two electron repulsion integrals and A_{ij}^{KL} and B_{ijkl}^{KL} are symbolic matrix element coefficients. In second quantisation A_{ij}^{KL} and B_{ijkl}^{KL} are matrix elements of creation and annihilation operators, $\hat{a}_{i\sigma}^{\dagger}$ and $\hat{a}_{j\sigma}$, respectively,

$$A_{ij}^{KL} = \langle K| \sum_{\sigma} \hat{a}_{i\sigma}^{\dagger} \hat{a}_{j\sigma} |L\rangle \tag{2.3}$$

$$B_{ijkl}^{KL} = \langle K| \sum_{\sigma\gamma} \hat{a}_{i\sigma}^{\dagger} \hat{a}_{k\gamma}^{\dagger} \hat{a}_{l\gamma} \hat{a}_{j\sigma} |L\rangle, \tag{2.4}$$

where σ,γ denote spin, and take the values α or β. Equation 2.4 is often computed [37,42] using a resolution of the identity

$$B_{ijkl}^{KL} = \langle K| \sum_{J} \sum_{\sigma\gamma} \hat{a}_{i\sigma}^{\dagger} \hat{a}_{j\sigma} |J\rangle \langle J| \hat{a}_{k\gamma}^{\dagger} \hat{a}_{l\gamma} |L\rangle - \delta_{jk} \langle K| \sum_{\sigma} \hat{a}_{l\sigma}^{\dagger} \hat{a}_{i\sigma} |L\rangle. \tag{2.5}$$

However, Zarrabian *et al.* [44] recognised that the number of intermediate states could be reduced by using an alternative resolution of the identity

$$B_{ijkl}^{KL} = \langle K| \sum_{J'} \sum_{\sigma\gamma} \hat{a}_{i\sigma}^{\dagger} \hat{a}_{k\gamma}^{\dagger} |J'\rangle \langle J'| \hat{a}_{l\gamma} \hat{a}_{j\sigma} |L\rangle. \tag{2.6}$$

The advantage compared to previous methods comes from the reduced number of electrons in the intermediate states (up to two electrons less than the reference configurations).

We have taken this idea further by avoiding the resolution of the identity entirely and using *reduced lists*, which permit the B_{ijkl}^{KL} to be computed very easily.

There are essentially two different ways of interpreting equation 2.2:

Configuration driven approach (CDA): For every pair of CSFs $|K\rangle$,$|L\rangle$ defining an element of the CI Hamiltonian, all ij,kl with $A_{ij}^{KL} \neq 0$ and $B_{ijkl}^{KL} \neq 0$ need to be found [14, 42–44].

Integral driven approach (IDA): For every ij,kl defining integrals $(i|j)$ and $(ij|kl)$, all pairs of CSFs $|K\rangle$,$|L\rangle$ corresponding to non-zero coefficients $A_{ij}^{KL}$and B_{ijkl}^{KL} need to be found. This approach was first introduced in quantum chemistry by Roos [33]. For this approach we may rewrite equation 2.1.

$$\sigma_K = \underbrace{\sum_{ij}(i|j)\sum_L A_{ij}^{KL}C_L}_{^{1e}\sigma_K} + \underbrace{\frac{1}{2}\sum_{ijkl}(ij|kl)\sum_L B_{ijkl}^{KL}C_L}_{^{2e}\sigma_K} \tag{2.7}$$

The main problem of either approach is the efficient determination of the indices $|K\rangle$, $|L\rangle$ and $(ij|kl)$ that lead to non-zero matrix elements without explicit elaborate testing of the complete CI configuration space. In what follows we develop an efficient solution to this problem based on the IDA, without the necessity of index space testing.

A major step forward in IDA CI approaches based upon determinants was implemented by Knowles and Handy [42], who recognised the computational advantage in splitting the Slater determinants into α- and β-strings

$$|K\rangle = |K_\alpha K_\beta\rangle. \tag{2.8}$$

An α-string, K_α, is an ordered product of N_α creation operators for molecular spin orbitals with α spin, and a β-string, K_β, is an ordered product of N_β creation operators for molecular spin orbitals with β spin. The constant lengths of the α-strings and β-strings are defined from the total number of electrons N and the z-component of the total spin. Addressing of the CI vector is achieved in a fashion proposed by Knowles and Handy [42].

An addressing array Z is defined, separately for α- and β-strings, given by

$$\begin{aligned} Z(k,l) &= \sum_{m=M-l+k}^{M-k} \left[\binom{m}{N_\sigma - k} - \binom{m-1}{N_\sigma - k - 1} \right] \\ & \quad (M - N_\sigma + k \geq l \geq k; \quad k < N_\sigma) \\ Z(N_\sigma, l) &= l - N_\sigma \qquad (M \geq l \geq N_\sigma), \end{aligned} \tag{2.9}$$

where k refers to an electron, l to an orbital, M is the number of orbitals, and N_σ the number of electrons (the index σ denotes spin). Any string is identified by a list of occupied orbitals in strictly ascending order. The address of the string is then given by

$$\mathrm{Addr}\{K_\sigma\} = 1 + \sum_{k=1}^{N_\sigma} Z(k, l(k)) \tag{2.10}$$

and with the above definition of Z the addressing is lexical without gaps. Given the addresses of the α- and β-strings K_α, K_β forming a given Slater determinant, the determinant can in turn be addressed in lexical order as

$$\mathrm{Addr}\{K\} = (\mathrm{Addr}\{K_\alpha\} - 1) \times \binom{M}{N_\beta} + \mathrm{Addr}\{K_\beta\}. \tag{2.11}$$

Note that this addressing scheme is not limited to Slater determinants. If, for example, the CSFs are chosen to be spin-adapted Hartree Waller functions [50, 51], equation 2.11 may be rewritten (for $S = 0$)

$$\mathrm{Addr}\{K\} = \frac{\mathrm{Addr}\{K_\alpha\}(\mathrm{Addr}\{K_\alpha\} - 1)}{2} + \mathrm{Addr}\{K_\beta\}. \tag{2.12}$$

The previous formulation facilitates computer implementation, since strings can be represented as an ordered sequence of occupation numbers, 1 for occupied and 0 for unoccupied orbitals, and one can then perform binary operations on the strings. In a system containing N_α α-electrons and M orbitals, each α-string is represented by a binary word of length M, containing N_α 1's and $(M - N_\alpha)$ 0's. We will refer to this representation of a string K_σ as ϕ_{K_σ} and use the notation b_k to indicate the kth bit of ϕ_{K_σ}. Furthermore we will denote by $\mathcal{L}_\alpha = \mathcal{L}_{N_\alpha}^M$ the list of strings in lexical order. Table 2.1 shows an example of $M = 6, N_\alpha = 3$.

Table 2.1.: List $\mathcal{L}_\alpha$ of strings K_α and their associated binary representations ϕ_{K_α} for $M = 6$, $N_\alpha = 3$.

	$\mathcal{L}_\alpha$	
Addr$\{K_\alpha\}$	K_α	ϕ_{K_α}
1	123	000111
2	124	001011
3	125	010011
4	126	100011
5	134	001101
6	135	010101
7	136	100101
8	145	011001
9	146	101001
10	156	110001
11	234	001110
12	235	010110
13	236	100110
14	245	011010
15	246	101010
16	256	110010
17	345	011100
18	346	101100
19	356	110100
20	456	111000

2.3. Reduced lists

In practical implementation of the IDA CI algorithm, by far the most computationally demanding term in equation 2.7 is the two-electron part, ${}^{2e}\sigma_K$. In this part, operators $\hat{a}^\dagger_{i\alpha}\hat{a}_{j\alpha}$ only act on α-strings, leaving β-strings unaltered, and *vice versa*. Using equation 2.8 we may therefore rewrite the expression for the vector coupling coefficient B^{KL}_{ijkl} in ${}^{2e}\sigma_K$ (cf. equation 2.7) as

$$\begin{aligned} B^{KL}_{ijkl} &= \langle K_\alpha|\hat{a}^\dagger_{i\alpha}\hat{a}^\dagger_{k\alpha}\hat{a}_{l\alpha}\hat{a}_{j\alpha}|L_\alpha\rangle\langle K_\beta|L_\beta\rangle + \langle K_\alpha|L_\alpha\rangle\langle K_\beta|\hat{a}^\dagger_{i\beta}\hat{a}^\dagger_{k\beta}\hat{a}_{l\beta}\hat{a}_{j\beta}|L_\beta\rangle \\ &\quad + \langle K_\alpha|\hat{a}^\dagger_{i\alpha}\hat{a}_{j\alpha}|L_\alpha\rangle\langle K_\beta|\hat{a}^\dagger_{k\beta}\hat{a}_{l\beta}|L_\beta\rangle + \langle K_\alpha|\hat{a}^\dagger_{k\alpha}\hat{a}_{l\alpha}|L_\alpha\rangle\langle K_\beta|\hat{a}^\dagger_{i\beta}\hat{a}_{j\beta}|L_\beta\rangle. \end{aligned} \tag{2.13}$$

There are two different types of products appearing on the right hand side of equation 2.13: case a, where the first term corresponds to double excitation of the α-string, while the β-string remains unaltered, i.e. for a non-zero contribution we have $K_\beta = L_\beta$ (in the second term the situation is reversed) and case b, where the third and fourth terms are products of single excitations of both the α- and β-strings.

We now discuss the computation of the matrix elements in equation 2.13. We shall use two concepts: a *reduced list* and the *excitation list* [42] that can be generated from it. For the sake of simplicity we will start by dealing with case b. Consider the single excitation matrix element:

$$\langle K_\alpha|\hat{a}^\dagger_{4\alpha}\hat{a}_{2\alpha}|L_\alpha\rangle \neq 0. \tag{2.14}$$

For a non-zero matrix element the binary representations of all the $\{K_\alpha, L_\alpha\}$, i.e. all the $\{\phi_{K_\alpha}, \phi_{L_\alpha}\}$, must be identical except for bits b_4 and b_2 (i.e. in ϕ_{K_α} we must have $b_4 = 1$, $b_2 = 0$ and in ϕ_{L_α} we must have $b_4 = 0$, $b_2 = 1$). The set of all $\{K_\alpha, L_\alpha\}$ that give non-zero matrix elements for equation 2.14 is called the excitation list $\mathcal{X}^{4\alpha}_{2\alpha}$. However, all information about the required string lists $\{K_\alpha, L_\alpha\}$ is implicit in the excitation operator, $\hat{a}^\dagger_{4\alpha}\hat{a}_{2\alpha}$ and this leads to the reduced list concept which omits bits b_4 and b_2. The complete excitation list $\mathcal{X}^{4\alpha}_{2\alpha}$ for this single excitation may thus be computed from a much shorter list $\mathcal{L}^{M-2}_{N_\alpha-1}$ (with elements ϕ_α) for $N_\alpha - 1$ electrons and $M-2$ orbitals. The relevant strings K_α and L_α in $\mathcal{X}^{4\alpha}_{2\alpha}$ can easily be generated by inserting the known bits b_2 and b_4, as shown in table 2.2 (inserted bits in bold type). The crucial observation is that, the same list $\mathcal{L}^{M-2}_{N_\alpha-1}$, can be used for any index pair $i \neq j$ to generate *all* excitation lists $\mathcal{X}^{j\alpha}_{i\alpha}$. Accordingly we now have a general scheme to reconstruct all relevant string pairs $\{K_\alpha, L_\alpha\}$ for any excitation $\langle K_\alpha|\hat{a}^\dagger_{i\alpha}\hat{a}_{j\alpha}|L_\alpha\rangle$ with given $i \neq j$. For $i = j$ the scheme is similar with a reduced string list $\mathcal{L}^{M-1}_{N_\alpha-1}$. In this case we always have $K_\alpha = L_\alpha$ and only have to insert bit $b_i = 1$.

Table 2.2.: Illustration of the generation of the excitation list $\mathcal{X}_{2\alpha}^{4\alpha}$ from the reduced list $\mathcal{L}_{N_\alpha-1}^{M-2}$ with elements ϕ_α for excitation $\hat{a}_{4\alpha}^\dagger \hat{a}_{2\alpha}$ by adding bits a) $b_2 = 0, b_4 = 1$ and b) $b_4 = 0, b_2 = 1$ to generate the bra-ket pairs $K_\alpha\, L_\alpha$.

		a)		b)	
$\mathcal{L}_{N_\alpha-1}^{M-2}$	ϕ_α	ϕ_{K_α}	Addr$\{K_\alpha\}$	ϕ_{L_α}	Addr$\{L_\alpha\}$
1	0011	00**1**1**0**1	5	00**0**1**1**1	1
2	0101	01**1**0**0**1	8	01**0**0**1**1	3
3	1001	10**1**0**0**1	9	10**0**0**1**1	4
4	0110	01**1**1**0**0	17	01**0**1**1**0	12
5	1010	10**1**1**0**0	18	10**0**1**1**0	13
6	1100	11**1**0**0**0	20	11**0**0**1**0	16

We now briefly mention the computation of the numerical values of the matrix elements. The non-zero values of $\langle K_\alpha|\hat{a}_{i\alpha}^\dagger \hat{a}_{j\alpha}|L_\alpha\rangle$ are 1 and -1. The associated sign, $\mathrm{sgn}_{ij}^{K_\alpha}$, is defined through

$$|K_\alpha\rangle = \mathrm{sgn}_{ij}^{K_\alpha}\, \hat{a}_{i\alpha}^\dagger \hat{a}_{j\alpha}|L_\alpha\rangle \tag{2.15}$$

and is easily evaluated from ϕ_{K_α} [52]

$$\mathrm{sgn}_{ij}^{K_\alpha} = \left\{ \begin{array}{c} +1 \\ -1 \end{array} \right\} \text{ for } \sum_{a=i+1}^{j-1} b_a \left\{ \begin{array}{l} \text{even.} \\ \text{odd.} \end{array} \right. \tag{2.16}$$

For given i, j we re-define the excitation list $\mathcal{X}_{i\alpha}^{j\alpha}$ to contain Addr(K_α), Addr(L_α) and $\mathrm{sgn}_{ij}^{K_\alpha}$ for each relevant string pair $K_\alpha\, L_\alpha$. It should now be clear that numerical values of the matrix elements for any given i and j can be easily generated from the corresponding reduced list. We will return to discuss this concept in a general way subsequently.

First we consider the terms corresponding to double excitations (equation 2.13). Turning to case a, where one term corresponds to a double excitation, no new concepts arise. In the most general case we need a reduced list $\mathcal{L}_{N_\alpha-2}^{M-4}$, which contains only $N_\alpha - 2$ electrons in $M-4$ orbitals. Consider the double excitation

$$\langle K_\alpha|\hat{a}_{4\alpha}^\dagger \hat{a}_{3\alpha}^\dagger \hat{a}_{1\alpha} \hat{a}_{2\alpha}|L_\alpha\rangle \neq 0. \tag{2.17}$$

Here four bits, b_4, b_3, b_1 and b_2, are predefined instead of two but otherwise the considerations are the same as for the single excitation reduced lists. The reduced string list, $\mathcal{L}_{N_\alpha-2}^{M-4}$, ($N_\alpha - 2$ electrons in $M-4$ orbitals) provides the information to construct all rele-

Table 2.3.: Illustration of the generation of the excitation list $\mathcal{X}_{43}^{21}$ from reduced list $\mathcal{L}_{N_\alpha-2}^{M-4}$ for the excitation $\hat{a}_{4\alpha}^\dagger \hat{a}_{3\alpha}^\dagger \hat{a}_{1\alpha} \hat{a}_{2\alpha}$ through adding bits a) $b_1 = 0, b_2 = 0, b_3 = 1, b_4 = 1$ and b) $b_1 = 1, b_2 = 1, b_3 = 0, b_4 = 0$ to generate the bra-ket pairs K_α L_α.

		a)		b)	
$\mathcal{L}_{N_\alpha-2}^{M-4}$	ϕ_α	ϕ_{K_α}	Addr$\{K_\alpha\}$	ϕ_{L_α}	Addr$\{L_\alpha\}$
1	01	01 **1100**	17	01 **0011**	3
2	10	10 **1100**	18	10 **0011**	4

Table 2.4.: Comparison of lengths of reduced lists to complete string list $\mathcal{L}_\alpha$ analytically and for some examples.

		$\mathcal{L}_\alpha$	$\mathcal{L}_{N_\alpha-1}^{M-1}$	$\mathcal{L}_{N_\alpha-1}^{M-2}$	$\mathcal{L}_{N_\alpha-2}^{M-2}$	$\mathcal{L}_{N_\alpha-2}^{M-3}$	$\mathcal{L}_{N_\alpha-2}^{M-4}$
M	N_α	$\binom{M}{N_\alpha}$	$\binom{M-1}{N_\alpha-1}$	$\binom{M-2}{N_\alpha-1}$	$\binom{M-2}{N_\alpha-2}$	$\binom{M-3}{N_\alpha-2}$	$\binom{M-4}{N_\alpha-2}$
8	2	28	7	6	1	1	1
12	3	220	55	45	10	9	8
16	4	1820	455	364	91	78	66
20	5	15504	3876	3060	816	680	560
8	4	70	35	20	15	10	6
10	5	252	126	70	56	35	20
12	6	924	462	252	210	126	70
14	7	3432	1716	924	792	462	252
16	8	12870	6435	3432	3003	1716	924

vant string pairs $\{K_\alpha, L_\alpha\}$ from the list $\mathcal{L}_{N_\alpha-2}^{M-4}$, as is shown in table 2.3. Again, the same reduced list can be used to generate all excitation lists for the case of four distinct indices. In order to account for the special cases of two or three equal indices we need to introduce the reduced lists $\mathcal{L}_{N_\alpha-2}^{M-3}$ and $\mathcal{L}_{N_\alpha-2}^{M-2}$. Excitations of the form $\langle K_\alpha | \hat{a}_{i\alpha}^\dagger \hat{a}_{i\alpha}^\dagger \hat{a}_{i\alpha} \hat{a}_{i\alpha} | L_\alpha \rangle$ are not possible.

There are considerable advantages in using the reduced list concept. Obviously reduced lists are dramatically shorter than the full string list $\mathcal{L}_\alpha$. Table 2.4 shows analytically and for a number of examples the lengths of the actual string list $\mathcal{L}_\alpha$ and the reduced string lists. The combined length of all reduced lists is of the order of the length of $\mathcal{L}_\alpha$. Note that the number of CSFs in a CASSCF computation using Slater determinants equals $\mathcal{L}_\alpha \times \mathcal{L}_\beta$,

so the memory requirements for storing the reduced lists are negligible. Furthermore, since each entry of each reduced list reflects an actual non-zero contribution, no redundant testing is involved in finding these contributions and the construction of excitation lists $\mathcal{X}_{i\alpha}^{j\alpha}$ is accelerated accordingly. The greatest savings compared to conventional string testing are achieved using $\mathcal{L}_{N_\alpha-2}^{M-4}$ reduced lists.

Using the symmetry properties of two-electron integrals, it is possible to reduce the sums in equation 2.7 to unique integrals and summation over

$$\begin{gathered} i \geq j, \qquad k \geq l, \qquad (ij) \geq (kl) \\ \text{with} \quad (ij) = \tfrac{i(i-1)}{2} + j \quad ; \quad (kl) = \tfrac{k(k-1)}{2} + l. \end{gathered} \tag{2.18}$$

As described by Shavitt [40] this restriction of the index space gives rise to the subdivision into 11 *integral-classes* I_{ijkl} and to the addition of a factor $f(I_{ijkl})$ due to possible permutations of orbital indices (see table 2.5). The factor may be evaluated as

$$f(I_{ijkl}) = 4 \times \left(\frac{1}{2}\right)^{\delta_{ij}} \times \left(\frac{1}{2}\right)^{\delta_{kl}} \times \left(\frac{1}{2}\right)^{\delta_{(ij)(kl)}}. \tag{2.19}$$

Integral-classes 7–11 give rise to a second contribution, due to exchange of indices k and l.

As shown in table 2.5 several double excitations do not need to be considered, since their contribution contains double creation and/or destruction of an electron in the same orbital and therefore equals zero. The sign corresponding to non-zero contributions, $\mathrm{sgn}_{ijkl}^{K_\alpha}$, is defined (by analogy with equation 2.16) through

$$|K_\alpha\rangle = \mathrm{sgn}_{ijkl}^{K_\alpha} \hat{a}_{i\alpha}^\dagger \hat{a}_{k\alpha}^\dagger \hat{a}_{l\alpha} \hat{a}_{j\alpha} |L_\alpha\rangle. \tag{2.20}$$

It can be derived from $\mathrm{sgn}_{ij}^{K_\alpha}$ and $\mathrm{sgn}_{kl}^{K_\alpha}$ and suitable expressions for all integral-classes are given in table 2.5.

Let us briefly summarise. For given i, j an excitation list $\mathcal{X}_{i\alpha}^{j\alpha}$ is required containing $\mathrm{Addr}\{K_\alpha\}$, $\mathrm{Addr}\{L_\alpha\}$ and $\mathrm{sgn}_{ij}^{K_\alpha}$ for each relevant string pair K_α L_α. All the $\mathcal{X}_{i\alpha}^{j\alpha}$ are generated from the same reduced list $\mathcal{L}_{N_\alpha-1}^{M-2}$. The final matrix elements are generated directly from excitation lists of the form $\mathcal{X}_i^i$, $\mathcal{X}_i^j$, $\mathcal{X}_{ij}^{ij}$, $\mathcal{X}_{ik}^{jk}$, $\mathcal{X}_{ik}^{jl}$, which are generated from the precomputed reduced lists "on the fly". These precomputed reduced lists are quite short and can be held in memory in all feasible cases, thus avoiding any I/O. Further, the

Table 2.5.: Integral classes I_{ijkl} arising due to constraining the index loop to $i \geq j$, $k \geq l$ and $(ij) \geq (kl)$ (cf. equation 2.18). Column "αα-contribution" indicates a non-zero contribution, $f(I_{ijkl})$ is the factor corresponding to integral-class I_{ijkl}, and $\mathrm{sgn}_{ijkl}^{K_\alpha}$ gives the analytic expression for the associated sign (see text).

I_{ijkl}	$f(I_{ijkl})$	i	j	k	l	$\hat{a}_{i\alpha}^\dagger \hat{a}_{k\alpha}^\dagger = 0$	$\hat{a}_{l\alpha}\hat{a}_{j\alpha} = 0$	αα-contrib.	$\mathrm{sgn}_{ijkl}^{K_\alpha}$
1	$\frac{1}{2}$	i	i	i	i	×	×		
2	1	i	i	k	k			×	1
3	2	i	j	j	j		×		
4	2	i	j	k	k			×	$\mathrm{sgn}_{ij}^{K_\alpha}$
5	2	i	i	i	l	×			
6	2	i	i	k	l			×	$\mathrm{sgn}_{kl}^{K_\alpha}$
7	1	i	j	i	j	×	×		
		i	j	j	i			×	-1
8	2	i	l	k	l		×		
		i	l	l	k			×	$-\mathrm{sgn}_{ij}^{K_\alpha} \times \mathrm{sgn}_{kl}^{K_\alpha}$
9	2	i	k	k	l			×	$\mathrm{sgn}_{ij}^{K_\alpha} \times \mathrm{sgn}_{kl}^{K_\alpha}$
		i	k	l	k		×		
10	2	i	j	i	l	×			
		i	j	l	i			×	$-\mathrm{sgn}_{jl}^{K_\alpha}$
11	2	i	j	k	l			×	$\mathrm{sgn}_{ij}^{K_\alpha} \times \mathrm{sgn}_{kl}^{K_\alpha}$
		i	j	l	k			×	$\mathrm{sgn}_{ij}^{K_\alpha} \times \mathrm{sgn}_{kl}^{K_\alpha}$

Table 2.6.: Number of times each reduced string list is used inside the four-way index loop depending on the number of orbitals, M. In calculating these numbers $S = 0$ is assumed, i.e. α-string lists and β-string lists are identical (singlet) and full advantage is being taken of this symmetry.

	$\alpha\beta$		$\alpha\alpha$		
M	$\mathcal{L}^{M-1}_{N_\alpha-1}$	$\mathcal{L}^{M-2}_{N_\alpha-1}$	$\mathcal{L}^{M-2}_{N_\alpha-2}$	$\mathcal{L}^{M-3}_{N_\alpha-2}$	$\mathcal{L}^{M-4}_{N_\alpha-2}$
8	128	574	56	336	420
10	230	1365	90	720	1260
12	376	2783	132	1320	2970
14	574	5096	182	2184	6006
16	832	8620	240	3360	10920
20	1560	20805	380	6840	29070
32	6016	134168	992	29760	215760

potentially computationally intensive creation of strings in the excitation lists is done very efficiently by completing the appropriate reduced string lists using bit-operations.

The formalism just discussed becomes relatively more efficient as the number of active orbitals is increased. The important issue relates to the number of times a reduced list is used to create the needed excitation lists within the four-way index loop and thus to the efficiency of this approach relative to other approaches where the excitation lists are themselves created directly "on the fly". Remarkably, as the number of active orbitals increases, the lists used the most are the shortest ones in their respective category: $\mathcal{L}^{M-2}_{N_\alpha-1}$ for single excitations and $\mathcal{L}^{M-3}_{N_\alpha-2}$, $\mathcal{L}^{M-4}_{N_\alpha-2}$ for double excitations (see table 2.6). This effect is seen in context with table 2.4, which compares the actual size of string lists and reduced string lists in a number of cases. The interrelation between string list lengths, list usage, number of orbitals, M, and number of electrons, N_α, is not trivial, but the general tendencies are obvious: reduced string lists become small compared to $\mathcal{L}_\alpha$ if the number of orbitals becomes larger, and the relative usage of shorter string lists increases with larger active spaces. Thus, the efficiency of our algorithm increases with the size of the CI problem.

2.4. The direct CASSCF algorithm

We will now describe an algorithm for the implementation of the preceding ideas. This algorithm is inherently parallel but we postpone this discussion until section 2.5. For the

sake of simplicity we will assume the special case of "singlet" Hartree Waller CSFs, which we have implemented along with the "triplet" special case. In appendix A we outline the general algorithm using Slater determinants, including the one-electron contribution. With this third option any spin multiplicity can be used.

```
for 2N electrons and M orbitals,                          (All lists have to be built only once)
build string list L_N^M
build reduced string lists L_{N-2}^{M-2}, L_{N-2}^{M-3}, L_{N-2}^{M-4}
(parallel) loop over i,j                                  (cf. equation 2.18)
 loop over k,l                                            (cf. equation 2.18)
                                                          (If two or three indices in i,j,k,l
  loop over reduced string list L_{N-2}^{M-4}             are equal, then loop is over
                                                          L_{N-2}^{M-3} or L_{N-2}^{M-2}, respectively)
   (→ I_ijkl, f(I_ijkl))
   insert bits b_i,b_j,b_k,b_l → K_α, L_α, sgn(ijkl)
   loop over all K_β
    σ(K_α,K_β) := σ(K_α,K_β) + f(I_ijkl)sgn(ijkl)(ij|kl)C(L_α,K_β)   (Actual contribution)
    σ(L_α,K_β) := σ(L_α,K_β) + f(I_ijkl)sgn(ijkl)(ij|kl)C(K_α,K_β)
   end loop (K_β)
  end loop (L_{N-2}^{M-4})
 end loop (k,l)
end loop (i,j)
```

Algorithm 2.1: CAS: The 2e excitation; αα part.

The reduced list concept is the central feature of the method proposed in this work. The excitation lists $\mathcal{X}_i^i$, $\mathcal{X}_i^j$, $\mathcal{X}_{ij}^{ij}$, $\mathcal{X}_{ik}^{jk}$, $\mathcal{X}_{ik}^{jl}$, can be rapidly generated "on the fly" from these reduced lists. In the method described by Bendazzoli and Evangelisti [45] the excitation lists themselves were precomputed. Unfortunately the precomputed excitation lists, in most interesting cases, are too large to be held in working memory and must be stored on disk. Consequently repeated I/O destroys the overall efficiency. Thus Gagliardi *et al.* [52] suggested avoiding I/O by repeatedly creating the lists each time they are needed and called it *direct generation* of string lists. Their strategy to create the lists using this direct method requires the repeated testing of all entries of the complete string list $\mathcal{L}_\alpha$ inside the four-way loop over i,j,k,l and the building of temporary lists $\mathcal{X}_i^j$ etc. However, Gagliardi *et al.* also recognised that list creation has to be optimised, because it is called an enormous number of times inside the four-way loop over orbital indices i,j,k,l. To avoid redundancies, particularly in the case of $N_\alpha \ll M$, they developed a mixed approach in which $\mathcal{X}_i^i$ and $\mathcal{X}_{ij}^{ij}$ are precomputed and the remaining lists are created using the direct method, using the $\mathcal{X}_i^i$ and $\mathcal{X}_{ij}^{ij}$ as reference lists. This so called *two-step* method,

build reduced string lists $\mathcal{L}_{N-1}^{M-1}$, $\mathcal{L}_{N-1}^{M-2}$ (All lists haveto be built only once)
(parallel) loop over i,j (cf. equation 2.18)
 loop over reduced string list $\mathcal{L}_{N-1}^{M-2}$ (If the two indices in i,j are equal, then loop is over $\mathcal{L}_{N-1}^{M-1}$)
  insert bits $b_i,b_j \rightarrow K_\alpha,\ L_\alpha,\ \mathrm{sgn}(ij)$
  (excitation string $\rightarrow \mathcal{X}_i^j$)
  do 1-electron contribution (1-electron contribution not described here)
 end loop ($\mathcal{L}_{N-1}^{M-2}$)
 loop over k,l (cf. equation 2.18)
  ($\rightarrow I_{ijkl},\ f(I_{ijkl})$)
  loop over reduced string list $\mathcal{L}_{N-1}^{M-2}$ (If the two indices in k,l are equal, then loop is over $\mathcal{L}_{N-1}^{M-1}$)
   insert bits $b_k,b_l \rightarrow K_\beta,\ L_\beta,\ \mathrm{sgn}(kl)$
   (excitation string $\rightarrow \mathcal{X}_k^l$)
   loop over excitation list X_i^j
    loop over excitation list X_k^l
     $\sigma(K_\alpha,K_\beta) := \sigma(K_\alpha,K_\beta) + f(I_{ijkl})\mathrm{sgn}(ij)\mathrm{sgn}(kl)(ij|kl)C(L_\alpha,L_\beta)$ (Actual contribution)
     $\sigma(L_\alpha,L_\beta) := \sigma(L_\alpha,L_\beta) + f(I_{ijkl})\mathrm{sgn}(ij)\mathrm{sgn}(kl)(ij|kl)C(K_\alpha,K_\beta)$
    end loop (X_k^l)
   end loop (X_i^j)
  end loop ($\mathcal{L}_{N-1}^{M-2}$)
 end loop (k,l)
end loop (i,j)

Algorithm 2.2: CAS: The 2e excitation; αβ part.

contains only small redundancies if $N_\alpha \ll M$, but becomes increasingly inefficient if the number of electrons rises. These problems are avoided with the reduced list method presented in this chapter. Reduced lists are very economical and can be held in memory in all otherwise feasible cases, thus avoiding any I/O. For example, the number of Slater determinant CSFs in a CASSCF computation of a triplet state with 14 electrons in 14 active orbitals is $(\mathcal{L}_\alpha \times \mathcal{L}_\beta) = 9018009$, while the sum of all reduced strings in this case is 7218 and thus negligible. Further, the potentially computationally intensive creation of string lists is done very efficiently by completing the appropriate reduced string lists using bit-operations. We avoid any redundant index space testing inherent in the previously mentioned methods and thus we expect our algorithm to be equally efficient in the cases of few *and* many electrons.

We will discuss parallelisation in the next section. However it is important to remark at this stage that precomputation of reduced lists needs to be done only once and is very cheap. The subsequent recovery of excited string lists may be included in the parallel loop. Rossi *et al.* [47] recognised a scalability limitation in their parallelisation, since they chose to take advantage of data structures and thus parallelised the matrix multiplication itself. This strategy effectively excludes the list computation from parallelisation and thus severely limits the maximum useful number of processors.

2.5. Parallelisation

In this section we shall describe the parallel implementation of the above algorithm. We assume a scalable parallel distributed memory computer architecture consisting of nodes, each with local memory. The nodes themselves may be symmetric multi-processor (SMP) machines with shared memory. There are two key issues: splitting the total work to be done into sub-tasks and load balancing. The creation of sub-tasks essentially involves a decision about the parallel loop, where each cycle of the loop defines a sub-task that gets carried out in parallel on a node. However, the sub-tasks may be of different lengths, so load balancing is essential to ensure that each node is kept busy (i.e. one node may carry out several sub-tasks in the same time while another node does only a single task). Further, simple explicit allocation of equal numbers of tasks to all nodes would result in poor load balancing in cases where the processing elements, PEs (i.e. CPUs), are not equally loaded or have a different architecture.

In a parallel distributed memory computer architecture, the data in memory must be dis-

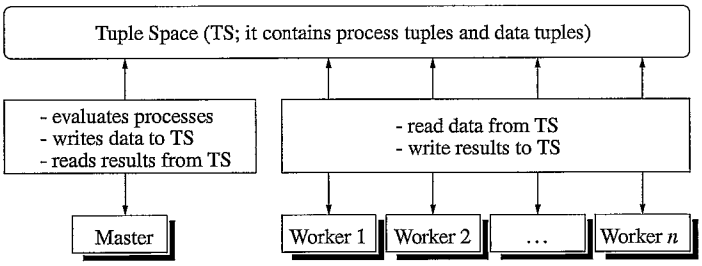


Figure 2.1.: Communication between processing units in the Linda parallel programming model.

tributed to the nodes on which it is used. We have implemented our algorithm within the Linda model [53]. In this model one assumes virtual shared memory. It is important to appreciate from the outset that we shall focus on the definition of the parallel loop and load balancing. This strategy is distinct from the strategy of other approaches where the focus is on data organisation and the parallelism comes from standard linear algebra subroutines.

Linda is defined as a language-independent set of coordination operations that have been integrated with C and Fortran to define the high-level programming languages C-Linda and Fortran-Linda, respectively. Thus we begin with a brief summary of the key concepts of the Linda model and then discuss our implementation.

One of the key concepts in the Linda coordination model is the shared, content-addressed, *virtual* memory called *tuple space*. All the interprocess communication is carried out *via* operations on tuple space. Thus coordination in Linda is *un-coupled*: the acts of sending and receiving data are independent. The technique of communication *via* the tuple space is visualised in figure 2.1. The data is moved from/to tuple space by using *tuples*. Tuples are sequences of data of different types. Linda interacts with the tuple space using four basic operations. Three operations can be used to add/read/remove tuples from the tuple space and a fourth operation is capable of creating new processes. These four operations are summarised in table 2.7.

Linda allows for distinguishing between the available PEs and the tasks to be computed. If a *worker-process* (worker) on a given PE finishes a given sub-task, it may look for the next sub-task in tuple space and continue working, without getting it pre-assigned explicitly. As we will discuss subsequently, this feature allows for a simple implementation of load balancing.

The IDA approach followed by us in this chapter involves the outer loops over the indices

Table 2.7.: The four basic Linda operations.

	Description
out()	Defines or adds a tuple in tuple space
in()	Reads and deletes a tuple from tuple space
rd()	Reads a tuple from tuple space
eval()	Creates a new process.

i,j,k,l. Inside these nested loops, the updating of the CI vector $\mathbf{C}$ takes place. Note that with the exception of the CI vector $\boldsymbol{\sigma}$ itself, all data inside the loops are non-varying. Thus we can minimise communication overhead by passing these static data to all processes only once at the beginning of each eigenvector iteration. This is of advantage especially for distributed memory architectures, and in particular for low-cost configurations with standard network interprocessor communication.

We must now define the parallel loop which in turn defines the sub-tasks to be run in parallel. Thus for example, if the indices i,j,k are used to define the parallel loop, then the inner loop over the remaining index l defines a sub-task that can execute independently on any node. Any combination of the outer loops over orbital indices i,j,k,l is, in principle, suitable. One wants to minimise the number of parallel sub-tasks, yet facilitate load balancing. To decide which is most appropriate we look at the relative operation count O in four cases (cf. equation 2.18):

1. The parallel loop is defined by all four indices, i,j,k,l: the operation count within the loop, $O(i,j,k,l)$, is dependent on the number of orbitals, M, the number of α- and β-electrons, N_α and N_β, respectively, and on the integral-class I_{ijkl}. However, for a qualitative comparison of the different possible parallel loops we assume

$$O(i,j,k,l) \approx \text{const.} \tag{2.21}$$

2. The parallel loop is defined by i,j,k: the operation count is proportional to all possible indices l, i.e.

$$O(i,j,k) = (j\delta_{ik} + k(1-\delta_{ik})) \times O(i,j,k,l). \tag{2.22}$$

3. The parallel loop is defined by i, j:

$$O(i,j) = \sum_{k=1}^{i} O(i,j,k). \tag{2.23}$$

4. The parallel loop is defined by i:

$$O(i) = \sum_{j=1}^{i} O(i,j). \tag{2.24}$$

In order to discuss the various options, it is convenient to define the task size quotient, Q, as the relative length of the longest to the shortest parallel task. For example, in case 4 the number of parallel tasks is very small (equal to the number of active orbitals, since $1 \leq i \leq M$) and the tasks are of very different length. Consider a FCI with 12 orbitals. In this case Q is given as

$$Q = \frac{O(12)}{O(1)} \approx \sum_{j=1}^{12} \sum_{k=1}^{12} (j\delta_{12k} + k(1-\delta_{12k})) = 870. \tag{2.25}$$

Thus the longest task will take 870 times longer than the shortest one. Consider instead the same FCI with 12 orbitals using a case 2 parallel loop. Using the three orbitals i, j, k to define the parallel loop significantly increases the number of independent tasks to $\sum_{i=1}^{12} i^2 = 650$ (because $1 \leq j \leq i$ and $1 \leq k \leq i$) and Q becomes 12.

When Q becomes very large and the number of sub-tasks is small, load balancing will be impossible. Furthermore, the number of PEs will be severely limited. Thus the optimum strategy for load balancing involves making a choice where Q is as small as possible, while keeping the number of tasks much larger than the number of workers. For example, the number of tasks for a CASSCF with $M = 12$ orbitals using i, j as the parallel loop is, according to equation 2.18, $\sum_i i = 78$. It increases, using indices i, j, k, to $\sum_i i^2 = 650$, and again, using all four indices, to $\sum_i (M+1-i)i^2 = 2366$. Number of tasks and operation count ratio of longest to shortest task, Q, are given in table 2.8 for 8, 10, 12, 14, 16, 20 and 32 orbitals.

Good load balancing can obviously easily be achieved by defining a large number of equally sized ($Q = 1$) very short tasks, but at the cost of communication overhead. Alternatively, if $Q \gg 1$, an implicit ordering of tasks (longest tasks first, shortest tasks last) may be required for good load balancing/scaling. If the amount of data defining a new

Table 2.8.: Number of independent parallel tasks and operation count ratio (Q) of most expensive to cheapest task as a function of the number of orbitals in the CI problem and the definition of the parallel loop.

	Parallel loop over:							
	i		i,j		i,j,k		i,j,k,l	
M	Q	Tasks	Q	Tasks	Q	Tasks	Q	Tasks
8	260	8	36	36	8	204	1	666
10	505	10	55	55	10	385	1	1540
12	870	12	78	78	12	650	1	3081
14	1379	14	105	105	14	1015	1	5565
16	2056	16	136	136	16	1496	1	9316
20	4010	20	210	210	20	2870	1	22115
32	16400	32	528	528	32	11440	1	139656

task is small and synchronising problems can be overcome, then communication overhead will be negligible. Since the tasks within the parallel loop are entirely defined by the chosen combination of indices, the amount of data defining each task is in fact very small. We may thus consider the parallel loops over i,j or i,j,k.

In our algorithm sub-tasks do not get pre-allocated to particular workers from the outset. Rather, load balancing is achieved by allocating a new task, dynamically, each time a worker finishes its current one. In order to keep overheads as low as possible, we have decided on using a relatively coarse-grained parallelism with indices i,j defining the parallel loop. This decision is a "trade-off" between the amount of communication, (reading tasks and data from, and writing results to, tuple space) and load balancing. In this case, as illustrated in table 2.8, the task size quotient, Q, increases linearly with the number of tasks. Therefore, in order to achieve good load balancing we use an implicit order of computation (most expensive tasks are computed first).

We now discuss the implementation of the general strategy just discussed for distributed memory architectures using Linda. Figure 2.2 shows a flowchart of our algorithm. We begin with a few definitions. We assume that each node may be an SMP. The number of processors on each SMP is indicated as $NProcS$ (on single processor nodes $NProcS = 1$). The main process is called the master process (master). The process spawned on the first processing node retrieves the index (ij) (initially $(ij) = \frac{M(M+1)}{2}$, corresponding to the most expensive task) and puts a new index $(ij)' = (ij) - NProcS$ into tuple space, which is then retrieved by the next process etc. If $NProcS > 1$ then $NProcS - 1$ shared memory

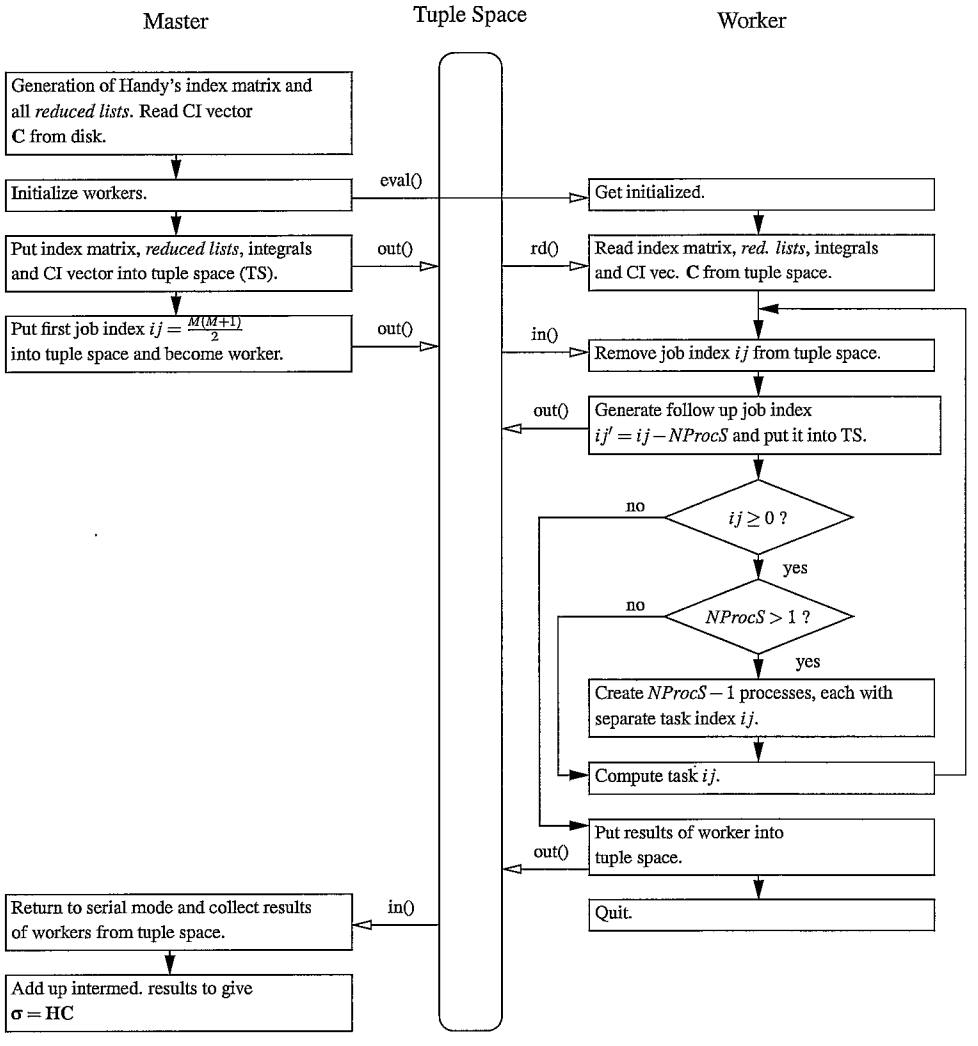


Figure 2.2.: General structure of the parallel code as described in the text.

processes are created, each of which will be assigned one unique (ij). Thus the original loop over i,j is parallelised. Each processor then carries out all the work for all inner loops over the indices k,l and performs the computation of $\boldsymbol{\sigma}$ using the original serial algorithm (cf. appendix A). On finishing, the next available index (ij) is retrieved from tuple space and this procedure is continued until all generator combinations are processed, i.e. the index $(ij) = 0$ is found. Then the intermediate results are passed through tuple space to be combined to give the final result of the computation.

SMP architectures are supported in three ways, i) using Linda, ii) using shared memory, or iii) a combination of both. In the first case, i), one Linda worker is created on each PE. Since communication overhead in our scheme is extremely small, this should not affect performance in a significant way, except where network performance is the bottleneck. Furthermore, using Linda even on SMPs takes full advantage of the load balancing mechanism described above. However, the static data (e.g. guess CI vector $\mathbf{C}$, integrals etc.) must be replicated for each Linda worker.

The alternative ii) is usage of shared memory ($NProcS > 1$). In this case all static data get replicated only once per SMP and are shared by all PEs. All result vectors (one per PE) are summed before the result of this worker is passed back to tuple space, where it is subsequently retrieved by the master process. This method has the advantage that all static data exist only once in working memory on a SMP node. For example, on a SMP with $NProcS = 2$ the memory requirements are for three CI vectors, i.e. one result vector for each PE plus one shared CI vector ($\mathbf{C}$). This is particularly useful if available working memory is limited and/or if the size of the CI vector is very large. A further advantage of this option is a reduction of communication by a factor of $NProcS$, since all static data and also the combined results have to be passed only once per $NProcS$ workers. The price to be paid in this case is that there is no load balancing between PEs on a SMP. The tasks defined by the $NProcS$ indices are of very similar length, but will be slightly different. In practise this means that efficiency will decrease if $NProcS$ becomes very large. However, if such computers were the target platform, the problem could be entirely avoided by having $Q = 1$, i.e. by using indices i,j,k,l as the parallel loop.

The third option is any combination of the previous two. In this case the number of PEs using shared memory, $NProcS$, may be chosen to comprise any available number of PEs on each node, i.e. on a SMP node with four processors, $NProcS$ can assume the values 1,2,3 or 4. Case iii) would then correspond to two Linda workers running on that node, each of them comprising $NProcS = 2$ shared memory processes. Taking advantage of this

option the mode of parallelism may be tailored to the available architecture.

The approach described above ensures great flexibility in choosing the number of processors and, due to the relatively high number of tasks, leads to an effective load balancing, provided the number of PEs is small compared to the number of tasks. Furthermore, since the tasks become considerably cheaper as (ij) gets smaller (cf. table 2.8), the implicit order of computation is an essential feature in order to ensure good efficiency, i.e. equal use of all PEs. The load balancing scheme implemented also allows for the fact that, in many environments, the CPU-time on different nodes of a parallel machine may be shared among a number of running programs and thus automatically uses the resources as they become available.

2.6. Examples

We have implemented both the serial and the parallel version of the algorithm performing the computation of the vector $\boldsymbol{\sigma}$ (cf. equation 2.1) in the CASSCF code of the Gaussian set of programs [54]. The purpose of this section is to give some indication of the performance of the method. Since the method is to be used within a CASSCF code, the most interesting cases are where the number of orbitals is either equal to the number of electrons or twice the number of electrons. Since the CI computation is performed iteratively with each CASSCF iteration, the computation is really only feasible if the eigenvector updating itself can be done in memory without additional I/O. In the algorithm just described, one needs 3 vectors that have the size of the CI space. Thus our target CI space is at most a few 10's of millions of configurations. To test performance and applicability of our implementation we carried out CASSCF calculations on two systems, the stilbene molecule with 14 electrons in 14 active orbitals and the pentalene molecule with 8 electrons in 16 active orbitals.

Since we are solely interested in measuring performance, we report timing data for one complete CASSCF iteration and measured the CPU-times used on all processors on an otherwise completely empty machine. An IBM SP2 with Power3 4way SMP nodes was used for the tests. In order to estimate the performance of our parallel code we examined three separate aspects of the timing data: the estimated CPU-time used for communication via tuple space; the CPU-times of both workers and master; and the elapsed time for the complete run.

The CPU-time needed for passing the task defining index (ij) was measured to be virtually

zero (on a ms scale). The amount of data to be passed to a Linda-worker is dominated by the vector **C**, which in the case of our stilbene test takes 47.1 MBytes of memory. We measured the CPU-time needed for passing a vector of this length to and from tuple space to be $t_{\mathrm{com}} \approx 0.5$s. Thus communication time is negligible on all workers. As explained in section 2.5 the results on the Linda-workers which have to be passed back to the master are again vectors of the same length. Thus the total communication time, in the master process, for retrieving these data from tuple space is given by the product $t_{\mathrm{com}} \times (NProcL-1)$. As discussed in section 2.5 this time can be reduced by using shared memory parallelism, if SMPs are being used.

The essential issue relating to the parallel efficiency of our algorithm relates to load balancing. We begin with a discussion of how this can be measured. The mean CPU time $\tilde{t}$ is taken of the CPU-time on all workers, t_j,

$$\tilde{t} = \frac{1}{\mathcal{N}} \sum_{j=1}^{\mathcal{N}} t_j, \tag{2.26}$$

where $\mathcal{N} = NProcL \times NProcS$ is the total number of processors used. This time is decomposed into contributions from individual Lanczos iterations and other parallel functions. In order to evaluate the load balancing of the parallel parts we define the *load balancing factor*

$$\mathcal{B} = \frac{\tilde{t}}{t_{\mathrm{max}}}, \tag{2.27}$$

with t_{max} being the maximum CPU-time used by a worker for a given part of the computation. The factor $\mathcal{B}$ constitutes an upper bound of the percentage of the total time in which all processors are doing useful work in a given part of the computation. Thus the product

$$\mathcal{N}_{\mathrm{eff},\|} = \mathcal{B} \times \mathcal{N} \tag{2.28}$$

denotes the effective number of processors of the respective parallel part of the program. However, to the end user, what is of interest is how much time, t_{elapsed}, an actual computation is going to take using $\mathcal{N}$ processors. To quantify this, we compare the total CPU-time used on all processors with the elapsed time t_{elapsed} (total wall clock time of the CASSCF run from start to finish),

$$\mathcal{N}_{\mathrm{eff}} = \frac{1}{t_{\mathrm{elapsed}}} \sum_{j=1}^{\mathcal{N}} t_j, \tag{2.29}$$

Note that $\mathcal{N}_{\mathrm{eff}}$ includes all elapsed time for a complete CASSCF iteration, including CI

vector and orbital update and other steps, and also including all bottlenecks due to I/O.

2.6.1. CAS(14,14)

Table 2.9 shows the CPU-times for one CASSCF iteration on 32 processors (8 nodes $\times$ 4 SMPs each). We chose to use the mixed, distributed and shared memory approach ($NProcL = 16$, $NProcS = 2$). The other main time-consuming routines, namely to compute the diagonal Hamiltonian and the density matrix use the same parallel loop over index (ij) as the CI vector updating routine. The elapsed time t_{elapsed} for this run was 2 h 45 min 47 s = 9947 s. The total CPU-time used up in parallel was 239853.5 s. We did not measure the CPU-time of the serial parts of the program. It is therefore not included in equation 2.29 and thus we underestimate $\mathcal{N}_{\text{eff}}$. The estimated speedup for this computation with 32 processors is 24.1. The difference between $\mathcal{N}_{\text{eff}}$ and $\mathcal{N}_{\text{eff},\|}$ is due mainly to the linear algebra and the associated I/O in the eigenvector computation.

2.6.2. CAS(8,16)

Table 2.10 shows the CPU-times for one CASSCF iteration on 24 processors (6 nodes $\times$ 4 SMPs each). Here we use the distributed memory parallelism only ($NProcL = 24$, $NProcS = 1$). Again, the other main time-consuming parallel routines, all of which use the same parallel loop over index ij, are listed. The elapsed time t_{elapsed} for this run was 1 h 7 min 33 s = 4053 s. The total CPU-time used up in parallel was 83072.0 s. Again, we did not measure the CPU-time of the serial parts of the program and therefore did not include it in equation 2.29. The estimated speedup for this computation with 24 processors is thus 20.5.

Due to the size of the computer available to us we are not able to provide any data about the scaling of our algorithm with larger configurations. However, we have demonstrated that the load balancing mechanism works very well, which is reflected in the obtained load balancing factors $\mathcal{B}$ for the sample Lanczos iterations (cf. tables 2.9 and 2.10). We expect this behaviour to continue as long as the total number of tasks (cf. table 2.8) is large with respect to the number of processors. The critical configuration, which should still give good values for $\mathcal{B}$ is reached if the number of processors equals half the number of tasks. If larger machine-configurations are available it is of advantage to adjust the parallel loop to include orbital indices i, j, k. Alternatively, the number of sub-tasks may

Table 2.9.: CPU-times (s) for 1 CASSCF iteration on stilbene with 14 active orbitals and 14 electrons ("CI vector updating" comprises 29 Lanczos iterations). The configuration space comprised 5891028 CSFs, corresponding to 11778624 Slater determinants. The calculation was done using 16 Linda-nodes with 2 SMPs each (physical machine configuration was 8 nodes with 4 SMPs each).

			CPU-time (s)					
Physical	Linda-		CI vector	Diagonal	Density	Sample Lanczos iterations (s)		
Node	Node	Processor	updating	Hamiltonian	Matrix	#1	#2	#3
1	0	1	7823.2	44.8	308.9	276.0	255.1	258.4
	(master)	2	7559.9	37.9	259.1	253.4	258.3	261.4
	1	1	6793.7	37.2	273.0	256.7	260.1	259.5
		2	6177.8	36.7	275.3	246.9	250.9	247.9
2	2	1	7170.9	37.6	290.4	255.5	248.7	246.7
		2	6682.9	37.0	285.1	241.1	258.4	267.5
	3	1	7296.6	40.8	281.8	252.8	257.0	248.0
		2	7370.0	31.2	286.8	247.6	242.6	253.7
3	4	1	7415.0	39.5	282.2	254.7	227.5	260.3
		2	7254.6	36.0	295.7	247.0	253.9	239.5
	5	1	7418.8	41.9	294.6	218.4	265.8	256.2
		2	7049.9	38.2	286.5	239.3	259.3	259.8
4	6	1	7306.9	38.0	298.3	227.9	244.8	263.6
		2	6376.8	36.9	272.4	250.8	248.3	247.8
	7	1	7440.8	27.1	276.0	253.5	223.8	264.6
		2	6844.2	38.5	276.3	245.9	248.8	252.5
5	8	1	7422.1	40.0	248.3	253.5	249.4	229.8
		2	7066.3	36.0	278.4	257.4	255.9	252.7
	9	1	7367.6	40.0	274.3	261.2	260.2	269.5
		2	6861.9	35.4	270.2	256.3	258.3	261.4
6	10	1	7292.5	36.9	249.7	253.2	270.2	258.2
		2	7256.1	36.4	277.0	257.0	254.8	238.2
	11	1	7385.7	38.5	277.7	259.7	265.0	261.0
		2	7159.9	38.4	273.6	267.2	253.3	252.8
7	12	1	7342.1	38.9	277.6	276.6	265.0	217.9
		2	7360.9	38.8	280.3	270.1	258.0	253.5
	13	1	7521.5	38.5	274.7	250.5	254.7	256.5
		2	6703.4	38.1	273.9	259.1	252.9	258.0
8	14	1	7461.5	36.1	272.9	258.0	258.9	262.7
		2	6973.9	38.2	269.2	259.9	259.8	245.6
	15	1	7391.8	34.6	196.1	262.9	259.7	253.2
		2	7364.8	38.3	200.8	238.4	229.4	245.0
	Σ		229914.0	1202.4	8737.1	8108.5	8108.8	8103.4
	$\tilde{t}$		7184.8	37.6	273.0	253.4	253.4	253.2
	$t_{\max}$		7823.2	44.8	308.9	276.6	270.2	269.5
	$\mathcal{B}$		0.92	0.84	0.88	0.92	0.94	0.94
	$\mathcal{N}_{\text{eff},\|\|}$		29.4	26.9	28.2	29.4	30.1	30.1

Table 2.10.: CPU-times (s) for 1 CASSCF iteration on pentalene with 16 active orbitals and 8 electrons (times in column "CI vector updating" comprises 48 Lanczos iterations). The configuration space comprised 1657110 CSFs, corresponding to 3312400 Slater determinants. The calculation was done using 24 Linda-nodes (no shared memory usage) (physical machine configuration was 6 nodes with 4 SMPs each).

		CPU-time (s)						
Physical	Linda-	CI vector	Full	Diagonal	Density	Sample Lanczos iterations (s)		
Node	Node	updating	Hamiltonian	Hamiltonian	Matrix	#1	#2	#3
1	0	3607.2	27.0	16.0	82.1	68.7	68.3	69.4
	1	3321.9	26.5	12.3	78.1	67.1	68.6	69.5
	2	3331.6	26.9	12.7	77.4	69.1	68.7	69.0
	3	3327.1	27.0	12.8	77.3	69.0	68.8	68.8
2	4	3332.2	26.6	12.5	77.8	68.9	68.8	69.3
	5	3328.9	26.9	12.7	77.5	68.3	68.3	68.8
	6	3327.7	26.8	12.9	77.1	68.3	68.6	68.5
	7	3327.0	26.8	12.9	76.3	69.0	68.3	69.2
3	8	3323.6	26.8	12.8	77.6	68.5	68.8	68.6
	9	3324.7	26.9	12.8	76.1	68.7	68.5	69.1
	10	3322.8	27.0	12.8	76.5	68.3	68.4	69.1
	11	3327.9	26.9	12.9	76.6	68.3	68.5	69.3
4	12	3328.7	26.5	12.8	76.6	69.2	68.9	69.2
	13	3333.7	26.6	12.8	77.4	68.5	68.3	69.4
	14	3338.0	27.0	12.7	76.3	68.8	69.5	68.8
	15	3334.8	27.0	13.0	76.6	68.6	68.5	68.8
5	16	3343.5	27.0	12.6	76.7	69.1	69.8	69.0
	17	3341.7	26.9	12.7	77.6	70.1	70.0	69.4
	18	3335.6	27.0	12.7	76.5	70.0	70.0	68.8
	19	3344.3	27.0	12.8	77.4	70.4	69.8	69.5
6	20	3340.2	27.0	12.6	77.1	70.1	70.3	68.9
	21	3342.3	26.9	12.8	76.8	70.3	70.4	68.9
	22	3340.0	27.0	13.0	77.2	70.2	70.4	69.6
	23	3338.9	26.9	12.5	77.1	70.3	69.8	68.9
Σ		80264.3	644.9	309.1	1853.7	1657.8	1658.3	1657.8
$\bar{t}$		3344.3	26.9	12.9	77.2	69.1	69.1	69.1
$t_{\max}$		3607.2	27.0	16.0	82.1	70.4	70.4	69.9
$\mathcal{B}$		0.93	1.00	0.80	0.94	0.98	0.98	0.99
$\mathcal{N}_{\text{eff},\parallel}$		22.3	23.9	19.2	22.6	23.5	23.5	23.8

be quadrupled (doubled for spin-adapted CSFs), if the different two electron contributions ($\alpha\alpha$, $\beta\beta$, $\alpha\beta$ and $\beta\alpha$) are each parallelised separately.

2.7. Summary

The direct reduced list algorithm presented in this chapter represents an efficient method for updating the CI vector within the algorithms by Davidson or Lanczos. Its main features are:

- memory efficient, compressed storage of precomputed excitation lists in main memory. The memory requirement for all reduced lists together roughly equals that of the complete string list $\mathcal{L}_\alpha$ and is thus negligible (cf. table 2.4). Uncompressing the reduced string lists in order to directly obtain the complete excitation lists is very efficient through usage of bit operations.
- usage of single excitation as well as double excitation lists. Decomposition of the identity is not required. This feature is an important prerequisite to avoid redundant index space testing.
- usage of outer nested loops over orbital indices i,j,k,l (integral driven). This allows for easy adjustment of the granularity of the parallelism, thus avoiding scaling problems: the parallel loop may be chosen to comprise any combination of nested loops, e.g. indices i,j or i,j,k or i,j,k,l. Consequently our algorithm can be easily adjusted to different parallel architectures.
- communication overhead (if run in parallel) is minimal through implicit task definition by only one integer. This technique also leads to a natural load balancing.

The algorithm is implemented in the current development version of the Gaussian package of programs [54]. The implementation comprises the option of parallel execution. Running in parallel may be done using distributed memory (Linda) or shared memory, or a combination of these. As well as the most general case of Slater determinants we also implemented the straightforward simplifications for singlets and triplets using Hartree Waller functions.

The parallelisation was done using the outer index loops over molecular orbital indices i,j as the parallel loop. We thus focused on keeping the communication overhead to a minimum. This was done having in mind our currently available machine (IBM SP2 with 8×4

processors) and usage on workstation clusters with standard network interconnection. If much larger hardware configurations are to be used it would be advantageous to include the inner loop over orbital index k into the parallel loop, as described in section 2.5. This would also have a positive effect on the scaling if purely shared memory mode without Linda is to be used, since the implemented load balancing scheme depends entirely on independently taking on parallel tasks using the Linda model.

3. The Direct RASSCF Method

3.1. Introduction

The CASSCF variant of the MCSCF method has been applied very successfully to many electronic structure problems [19], including the treatment of electronically excited states and potential energy surfaces (e.g. [55, 56]). It has many advantages over other truncated CI methods, e.g. the fact that it is size consistent. Interpretation of the resulting orbitals is also straightforward, since orbital rotations within the space of the active orbitals is invariant with respect to the energy [19], which allows one to "choose" the orbitals. Unfortunately, applicability of the CASSCF method is restricted to relatively small chemical systems even with advanced implementations and modern computer hardware, and truncated methods of some sort have to be employed in cases where CASSCF is not feasible. In particular, if no advantage can be taken of point group symmetry, then the practical limit with current hardware is around 12–14 active orbitals, at least in those cases where the number of electrons is about the same as the number of active orbitals. However, the resulting orbital occupancies of several orbitals in the active space are frequently either close to 2 or close to 0 over the entire range of electronic states and spatial configurations under investigation. It is assumed that in these cases the CASSCF configuration space can be successfully mimicked by applying certain occupancy restrictions on groups of orbitals. The idea is to ideally include all configurations that contribute to non-dynamical electron correlation, but to reduce the size of the CI vector compared to the CAS expansion.

Dynamic correlation effects are frequently included in an electronic structure computation in a subsequent step, for example by adding a calculation treating dynamic correlation perturbatively (e.g. [57, 58]). Alternatively a MRCI calculation may be performed, where selected CSFs from the CASSCF wave function are used as reference configura-

tions [59, 60]. However, this treatment assumes that the molecular orbitals are those obtained without taking dynamic correlation into account. Systems where the shapes of the strongly occupied orbitals depend on dynamical correlation effects include, for example, negative ions, electronic dipoles and excited states [23]. In these cases, an extended active space may be essential for an adequate description of the electronic structure problem at hand.

Both these cases, the approximation of an otherwise unfeasible CASSCF and the partial inclusion of dynamical correlation effects into a CASSCF computation, may be treated with the Restricted Active Space SCF (RASSCF) method [14]. The RASSCF method may be considered a logical extension to the CASSCF method. In the RASSCF formalism, the configuration space is specified by dividing the molecular orbitals into five subsets and imposing restrictions on the allowed configurations based upon occupations within those subsets. The first subset consists of the lowest lying (inactive) MOs, which interact only weakly with the other MOs, and are considered to be frozen. They are always doubly occupied for any allowed configuration. The second subset, denoted RAS1, typically includes all doubly occupied MOs in some accepted reference, e.g. a CASSCF wave function. Allowed configurations must contain a minimum of p electrons within RAS1. The third subset, denoted RAS2, includes MOs believed to be particularly important for the system under investigation. No occupancy restrictions are imposed on RAS2. The fourth subset, RAS3, consists of weakly occupied MOs which contribute relatively less to the description of the system of interest. Any allowed configuration can only have a maximum of q electrons in RAS3. The final subset includes all remaining MOs; they are unoccupied in all configurations of the RASSCF wavefunction. A useful feature of this separation of orbitals is the fact that almost any excitation based CI truncation scheme can be formulated using the RAS method. For example, a CISD (all single and double excitations from a single reference configuration) performed on a system with 10 electrons can be specified as having a minimum of 8 electrons in 5 RAS1 orbitals, RAS2 comprising the remaining orbitals, and RAS3 containing no orbitals. A Second Order CI (SOCI; all singly and double excitations from a multi-configuration reference) can be specified by requiring no orbitals in RAS1, placing the orbitals of the CASSCF active space in RAS2, and allowing a maximum of two electrons into the remaining orbitals in RAS3.

The main challenge in large scale CASSCF problems is the efficient computation of coupling coefficients, and this is not different in the RASSCF method. However, the restrictions introduced for the RAS1 and RAS3 subspaces mean that the indexing of the configuration state functions (CSFs) is more complicated. This in turn means that highly

efficient strategies such as the method of reduced excitation strings (cf. chapter 2) cannot be employed.

The first implementation of the RASSCF method was published by Olsen *et al.* [14]. Most subsequent implementations are based on the same Slater determinant (SD) based algorithm, including the work by Kozlowski and Pulay [61], although Malmquist *et al.* did report an implementation [23] using a different method based on spin-adapted CSFs. The aim of this chapter is to propose an efficient, integral driven algorithm of the RASSCF method, and to describe its implementation on a massively parallel computer. Our approach is based on a model space representation of the RAS active orbitals, and an efficient expansion of the model subspaces. This idea is inspired by the classic paper of Saunders and van Lenthe [60, 62]. The contributions of a block of integrals (with identical representation in the model space) are computed together. Reconstruction of binary strings is avoided and the full list of substring addresses is constructed using propagation rules. Finally proof of applicability of our RAS implementation is given in the form of two applications in section 3.8.

We start by reminding ourselves of the problem at hand. As in the CASSCF method, we seek the solution of the CI eigenvalue problem

$$\mathbf{HC} = \mathbf{SCE} \tag{3.1}$$

where

$$\mathbf{H} = \{\langle K|\hat{H}|L\rangle\} = \{H_{KL}\} \tag{3.2}$$

is the representation matrix in a basis of many-particle configuration state functions (CSF), which we denote as $\{|K\rangle\}$, and

$$\mathbf{S} = \{\langle K|L\rangle\} \tag{3.3}$$

is the metric. We will assume an orthonormal basis, i.e. $S_{KL} = \delta_{KL}$. In general only the lowest eigenvalues and their eigenvectors are of interest. In practise use is normally made of iterative eigenvector procedures, such as the methods of Lanczos or Davidson. The time consuming step in these methods is a linear transformation of the CI vector,

$$\boldsymbol{\sigma} = \mathbf{HC}, \tag{3.4}$$

where $\mathbf{C}$ is an approximate eigenvector from the previous iteration. The central practical

problem is the evaluation of the $\{H_{KL}\}$. The general result can be expressed as

$$H_{KL} = \sum_{ij}(i|j)A_{ij}^{KL} + \frac{1}{2}\sum_{ijkl}(ij|kl)B_{ijkl}^{KL}, \tag{3.5}$$

where the summation is over orbital indices, and $(i|j)$ and $(ij|kl)$ are the usual one and two electron repulsion integrals. The A_{ij}^{KL} and B_{ijkl}^{KL} are numerical *vector coupling* coefficients that depend on the nature of $|K\rangle$ and $|L\rangle$. In second quantisation, these numerical vector coupling coefficients emerge as matrix elements of creation and annihilation operators, $\hat{a}_{i\sigma}^{\dagger}$ and $\hat{a}_{j\sigma}$,

$$A_{ij}^{KL} = \langle K|\sum_{\sigma}\hat{a}_{i\sigma}^{\dagger}\hat{a}_{j\sigma}|L\rangle, \tag{3.6}$$

$$B_{ijkl}^{KL} = \langle K|\sum_{\sigma\gamma}\hat{a}_{i\sigma}^{\dagger}\hat{a}_{k\gamma}^{\dagger}\hat{a}_{l\gamma}\hat{a}_{j\sigma}|L\rangle, \tag{3.7}$$

where σ,γ denote spin.

Each Slater determinant of the CI expansion can be written as the tensor product of two strings, one of the occupied α-orbitals, K_α, and one of the occupied β-orbitals, K_β,

$$|K\rangle = |K_\alpha K_\beta\rangle. \tag{3.8}$$

A string K_σ is an ordered product of creation operators acting on the vacuum, and can be represented by a binary word of length M, where each bit b_i represents a spin orbital. The bit value of b_i indicates whether the orbital i is occupied or empty, i.e. the binary representation of K_σ, ϕ_{K_σ}, contains N_σ 1's and $(M-N_\sigma)$ 0's. We denote by $\mathcal{L}$ the list of all strings in lexical order. Each string K_σ can be identified by its address, Addr$\{K_\sigma\}$, which is identical to its position in $\mathcal{L}$. Knowles and Handy [42] have defined Addr$\{K_\sigma\}$ in full CI as

$$\mathrm{Addr}\{K_\sigma\} = 1 + \sum_{k=1}^{N_\sigma} Z(k,l(k)), \tag{3.9}$$

using the addressing array Z:

$$\begin{aligned} Z(k,l) &= \sum_{m=M-l+k}^{M-k}\left[\binom{m}{N_\sigma-k} - \binom{m-1}{N_\sigma-k-1}\right] \\ &\quad (M-N_\sigma+k \geq l \geq k; \quad k < N_\sigma) \\ Z(N_\sigma,l) &= l-N_\sigma \qquad (M \geq l \geq N_\sigma), \end{aligned} \tag{3.10}$$

where k refers to an electron, l to an orbital, M is the number of orbitals, N_σ the number of electrons and σ denotes spin. This addressing scheme for strings in full CI is simple. However, in anticipation of the added complexity that will arise when addressing RAS strings we now introduce a graphical representation of equation 3.10. Figure 3.1(a) illustrates the α-string space for $M = 8$ orbitals and $N_\alpha = 4$ electrons [63]. Each string corresponds to one *path* or *walk* on the grid of the graph. All paths begin at the foot of the graph, and finish in its head, advancing one level upwards for each orbital. If an orbital is occupied, the sloped segment step upwards is being used. The elements of the addressing array are added to the graph in figure 3.1(a). According to equation 3.9 the address of a string K_α is thus the sum of all circled values visited by the appropriate walk, plus 1.

We note in passing that both the graphical representation and the lexical indexing of strings used here are simply the special case of the general method introduced by Shavitt [40]. His method combined representation and indexing of spin adapted CSFs in a four dimensional graph. The simplification in the graphical representation used here comes from the fact that strings are built from spin-orbitals of one spin type only. The two possible directions that can be taken at each vertex of our string graph have historically been assigned so called *case numbers* [39, 40] 0 (for the addition of an empty orbital) and 1 (for the addition of an orbital containing an electron with α-spin). An alternative rule for constructing the binary representation of a string K_α is thus, following the corresponding walk from the head of the graph to its foot, to write down at each orbital level m the corresponding case number b_m, ordered from left to right.

To illustrate the above, consider the α-string $K_\alpha = 1478$. Its binary representation is $\phi_{K_\alpha} =$ 11001001 and the corresponding graphical representation is shown in figure 3.1(b). The address for the depicted walk is, according to equation 3.9, $\mathrm{Addr}\{K_\alpha\} = 1+0+16+10+4 = 31$. The β-string address is obtained in the same way. The address of the Slater determinant $|K\rangle$ may then be defined as

$$\mathrm{Addr}\{K\} = (\mathrm{Addr}\{K_\alpha\} - 1) \times \binom{M}{N_\beta} + \mathrm{Addr}\{K_\beta\}. \tag{3.11}$$

Equation 3.5 is a simple sum of products. While the one and two electron integrals are generally nonzero quantities, the coupling coefficients are mostly zero. Efficient computer implementations of any CI method take advantage of this fact. One can distinguish between two main types of strategies to deal with equation 3.5, namely the *configuration driven approach* (CDA) and the *integral driven approach* (IDA). In the CDA, given a configuration pair, $|K\rangle, |L\rangle$, all index pairs (i, j) and quadruples (i, j, k, l) that give nonzero

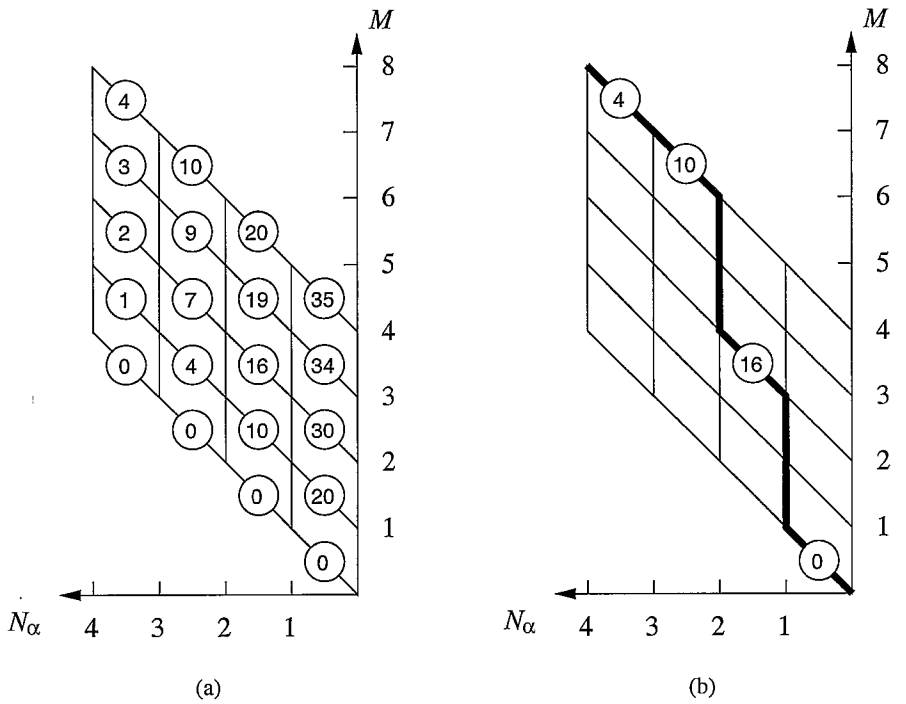


Figure 3.1.: (a): α-string graph for $N_\alpha = 4$ electrons in $M = 8$ orbitals. Every possible path from the bottom (foot) to the top (head) of the graph corresponds to an α-string. The orbitals are ordered and at each orbital level an α-electron may be added. At each vertex in the graph there are up to two paths upwards, corresponding to the next α-spin orbital being occupied (sloped path) or unoccupied. A lexical addressing of the strings is achieved by using Handy's addressing array [42](cf. equations 3.9 and 3.10). The numbers on the slopes are the *arc weights*, $Z(k,l)$, and the address of any string is obtained simply by taking the sum of the arc weights. This addressing scheme corresponds to a strict left-to-right ordering of the strings. (b): The thick line represents the walk corresponding to the binary string 11001001. The corresponding string address is 31.

coupling coefficients need to be found. In the IDA, by contrast, all configuration pairs $|K\rangle, |L\rangle$ for nonzero coupling coefficients need to be found, given a set of orbital indices, i,j,k,l. In modern implementations both strategies have found their application. To a degree, the optimum approach will depend on the properties of the computer hardware, and efficient implementations that take advantage of vector processors have been devised. More recently, the development of computer hardware has been towards parallel configurations, where distributed and shared memory architectures are often found together, and vector architectures are less relevant. This should be reflected in the design of any implementation. In particular, methods for a well functioning load balancing should be considered.

The true computational cost of evaluating the matrix multiplication of equation 3.4 consists of finding all nonzero coupling coefficients, $A_{ij}^{KL}, B_{ijkl}^{KL}$, and performing the SAXPY operations

$$\begin{aligned} \sigma(K) &:= \sigma(K) + (i|j)A_{ij}^{KL}C(L) \\ \sigma(K) &:= \sigma(K) + (ij|kl)B_{ijkl}^{KL}C(L) \end{aligned} \tag{3.12}$$

for all nonzero coupling coefficients. As the proportion of nonzero coupling coefficients is tiny (for example, $\ll 0.01\%$ for a CASSCF(12,12)) an efficient method of finding nonzero coupling coefficients *implicitly* and without trying or testing the complete space of orbital and/or configuration indices is thus highly desirable. To find a solution to this *indexing problem*, we start by rewriting the $\boldsymbol{\sigma}$ vector of equation 3.4 as a sum of 2 terms

$$\boldsymbol{\sigma} = {}^{1e}\boldsymbol{\sigma} + {}^{2e}\boldsymbol{\sigma} \tag{3.13}$$

The one-electron term,

$${}^{1e}\sigma(K) = \sum_L \sum_{ij} (i|j)A_{ij}^{KL}C(L), \tag{3.14}$$

may be rewritten with the outer sum over orbital indices i and j and the inner sums over α-strings and β-strings,

$$\begin{aligned} {}^{1e}\sigma(K_\alpha, K_\beta) = \sum_{ij}(i|j) \Bigg(& \sum_{L_\alpha} \langle K_\alpha | \hat{a}_{i\alpha}^\dagger \hat{a}_{j\alpha} | L_\alpha \rangle C(L_\alpha, K_\beta) \\ & + \sum_{L_\beta} \langle K_\beta | \hat{a}_{i\beta}^\dagger \hat{a}_{j\beta} | L_\beta \rangle C(K_\alpha, L_\beta) \Bigg) \quad \forall K_\alpha, K_\beta \end{aligned} \tag{3.15}$$

Likewise, the two-electron term

$$^{2e}\sigma(K) = \sum_{L}\sum_{ijkl}(ij|kl)B^{KL}_{ijkl}C(L), \tag{3.16}$$

may be rewritten along the same lines, yielding

$$\begin{aligned} ^{2e}\sigma(K_\alpha,K_\beta) \;=\; \sum_{ijkl}(ij|kl)\Bigg(&\sum_{L_\alpha}\langle K_\alpha|\hat{a}^\dagger_{i\alpha}\hat{a}^\dagger_{k\alpha}\hat{a}_{l\alpha}\hat{a}_{j\alpha}|L_\alpha\rangle C(L_\alpha,K_\beta)\\ &+\sum_{L_\beta}\langle K_\beta|\hat{a}^\dagger_{i\beta}\hat{a}^\dagger_{k\beta}\hat{a}_{l\beta}\hat{a}_{j\beta}|L_\beta\rangle C(K_\alpha,L_\beta)\\ &+\sum_{L_\alpha}\langle K_\alpha|\hat{a}^\dagger_{i\alpha}\hat{a}_{j\beta}|L_\alpha\rangle\sum_{L_\beta}\langle K_\beta|\hat{a}^\dagger_{k\beta}\hat{a}_{l\beta}|L_\beta\rangle C(L_\alpha,L_\beta)\\ &+\sum_{L_\alpha}\langle K_\alpha|\hat{a}^\dagger_{k\alpha}\hat{a}_{l\alpha}|L_\alpha\rangle\sum_{L_\beta}\langle K_\beta|\hat{a}^\dagger_{i\beta}\hat{a}_{j\beta}|L_\beta\rangle C(L_\alpha,L_\beta)\Bigg) \quad \forall K_\alpha,K_\beta \end{aligned} \tag{3.17}$$

Thus in equations 3.15 and 3.17 the outer summation is over orbital indices i,j and i,j,k,l, respectively. This implies the use of an IDA, where the orbital indices and the repulsion integrals are given, and all string pairs K_α,L_α and K_β,L_β that give a nonzero contribution to $^{1e}\sigma$ and $^{2e}\sigma$ need to be found. In chapter 2 we have developed the method of reduced excitation strings to solve equations 3.15 and 3.17 efficiently for a full CI / CASSCF computation, i.e. for unrestricted active orbital spaces. Figure 3.2 illustrates this method using the graphical representation of strings. Generally for non-zero contributions involving the factor $\langle K_\sigma|\hat{a}^\dagger_{i\sigma}\hat{a}_{j\sigma}|L_\sigma\rangle$, the walks representing strings K_σ,L_σ on the σ-graph coincide on all orbital levels, except in the range given by the operator $\hat{a}^\dagger_{i\sigma}\hat{a}_{j\sigma}$. Between orbital levels i and j the walks are parallel. As a result, every nonzero factor $\langle K_\sigma|\hat{a}^\dagger_{i\sigma}\hat{a}_{j\sigma}|L_\sigma\rangle$ is represented by a loop on the corresponding σ-graph. All such string pairs that give a nonzero contribution may be constructed by insertion of loop opening and loop closing elements at the required levels into the full set of strings on a reduced string graph, as shown in figure 3.2. The complete set of reduced excitation strings is denoted $\mathcal{L}^{M-2}_{N-1}$ and the list of string pairs $\{K_\sigma,L_\sigma\}$ that are constructed from the reduced list, using the operator $\hat{a}^\dagger_{i\sigma}\hat{a}_{j\sigma}$, is called the excitation list, $\mathcal{X}^{j\sigma}_{i\sigma}$. The value of each loop corresponding to a term $\langle K_\sigma|\hat{a}^\dagger_{i\sigma}\hat{a}_{j\sigma}|L_\sigma\rangle$ equals either $+1$ or -1, and this value may be stored in the excitation list together with the appropriate string pair. The same argument also applies to double excitation operators $\hat{a}^\dagger_{i\sigma}\hat{a}^\dagger_{k\sigma}\hat{a}_{l\sigma}\hat{a}_{j\sigma}$, with the only difference being that up to two loop opening

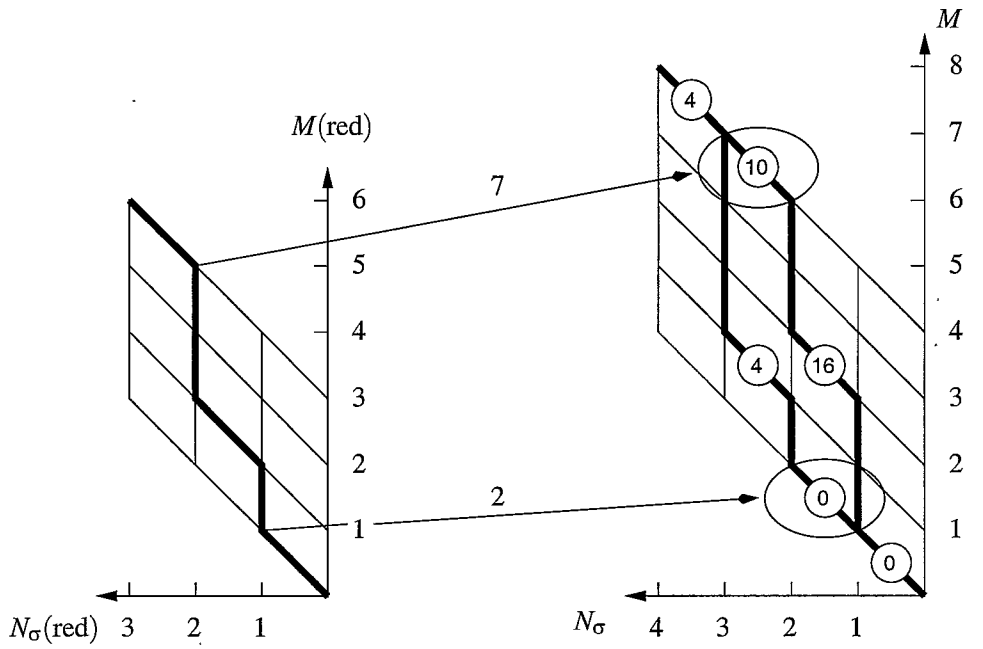


Figure 3.2.: Construction of a string pair K_σ, L_σ for excitation $\langle K_\sigma|\hat{a}^\dagger_{i\sigma}\hat{a}_{j\sigma}|L_\sigma\rangle$ from reduced string graph. The graph dimension of the reduced string graph is reduced by $M = 2$ orbitals and $N_\sigma = 1$ electron. The string pair, K_σ, L_σ is obtained by inserting a loop opening segment at orbital level $j = 2$ and a loop closing segment at level $i = 7$, starting with the lower orbital index. All remaining segments of the original string remain unchanged.

elements and two loop closing elements need to be inserted in the corresponding reduced string graph. Thus the method of reduced excitation lists provides an elegant and efficient means of finding all nonzero contributions to equations 3.15 and 3.17 for unrestricted CI methods.

In restricted CI methods in general, and in the RASSCF method in particular, the restrictions imposed on the active orbital space give rise to certain complications concerning the construction and indexing of CSFs and their strings, and we cannot apply the reduced excitation string method. However, an efficient solution to the indexing problem should nevertheless, as in the case of full CI / CASSCF implementation, aim to minimise or eliminate redundant index space testing. We propose in this chapter a solution to this problem, based on two steps: the introduction of a *RAS model orbital space*, and the efficient reconstruction of string pairs K_α, L_α from the model strings in that space, using *propagation rules*. We will see that the special properties of the RAS model space allow the usage of the reduced excitation string method.

The graphical representation of CSFs used in this work and outlined in section 3.2 greatly facilitates insight in and treatment of the RASSCF indexing problem. In section 3.3 we introduce the *model space* representation of the RAS space, again aided by a graphical representation. Based on the model space concept in section 3.4 we develop an efficient method for reconstructing the string pairs that give nonzero contributions to equations 3.15 and 3.17, using *propagation rules*. The method described in this work entirely avoids potentially costly index space testing.

3.2. A Graphical Representation of the RAS Configurations

The string space in the RASSCF method is a subset of the corresponding CAS string space, where the total number of active orbitals and electrons are identical. Due to the particle number restriction in the RAS1 and RAS3 orbital subspaces, the construction of CSFs is relatively more complicated. The complication arises on two levels, the indexing of strings and the combination of α-strings with β-strings. The remedy to both problems lies in the classification of certain groups of RAS strings. To this end we will make use of the concept of *string categories*. Within any string category indexing is then simple, as no restrictions apply. Allowed combination of α-strings with β-strings are found by identifying allowed combinations of string categories.

It is convenient to define several graphs. Each graph corresponds to a subset of the paths on the FCI graph. The space of strings is then described by the sum of all possible walks on all graphs. In addition, there will usually be restrictions on the allowed combination of graphs. If the state of interest is a singlet then the set of graphs for both spin spaces will be identical.

Figure 3.3 shows the complete set of α-string graphs for the case with $N_\alpha = 6$, $M = 12$, $M(\text{RAS1}) = 4$, $M(\text{RAS3}) = 4$ and a maximum of 2 electrons excited out of RAS1 and into RAS3, respectively. As in the CASSCF case, the graphical representation of the α-strings and β-strings may be used to order the paths. This is best done by assigning consecutive local addresses to strings within one graph. If the graphs themselves are ordered, then the global string address will be the sum of the local string address and an offset accounting for all paths in preceding graphs.

Kozlowski and Pulay [61] proposed a two level addressing scheme, the first level being a string category determined by the number of holes, i_h, in RAS1 and the number of electrons, i_e, in RAS3, while the second level gives the local string address within a given category. A category in this scheme corresponds exactly to a string graph as described above, and we will largely adopt the notation used in [61]. The category $\text{Cat}(i_h, i_e)$ is defined as

$$\text{Cat}(i_h, i_e) = (i_h + 1) + (\text{MxHole} + 1) \times i_e \tag{3.18}$$

and the number of categories (graphs) for each spin space is given by

$$\text{Cat}(\text{MxHole}, \text{MxElec}) = (\text{MxHole} + 1) \times (\text{MxElec} + 1). \tag{3.19}$$

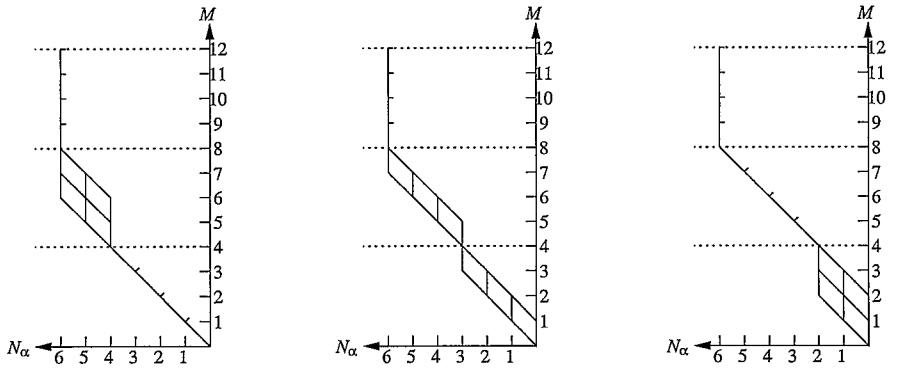
The length of that category (number of paths in the corresponding graph) is

$$L[\text{Cat}(i_h, i_e)] = \binom{M(\text{RAS1})}{i_h} \binom{M(\text{RAS2})}{N_\alpha - M(\text{RAS1}) + i_h - i_e} \binom{M(\text{RAS3})}{i_e}. \tag{3.20}$$

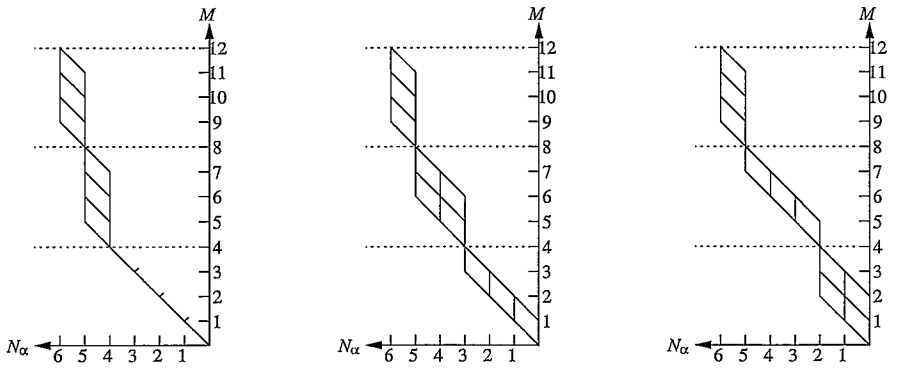
The address of an α-string $K_\alpha^{i_h, i_e}$ in category $\text{Cat}(i_h, i_e)$ can then be calculated as

$$\text{Addr}\{K_\alpha^{i_h, i_e}\} = \text{Addr}\{K_\alpha\} + \sum_{\text{Cat}=1}^{\text{Cat}(i_h, i_e)-1} L[\text{Cat}] \tag{3.21}$$

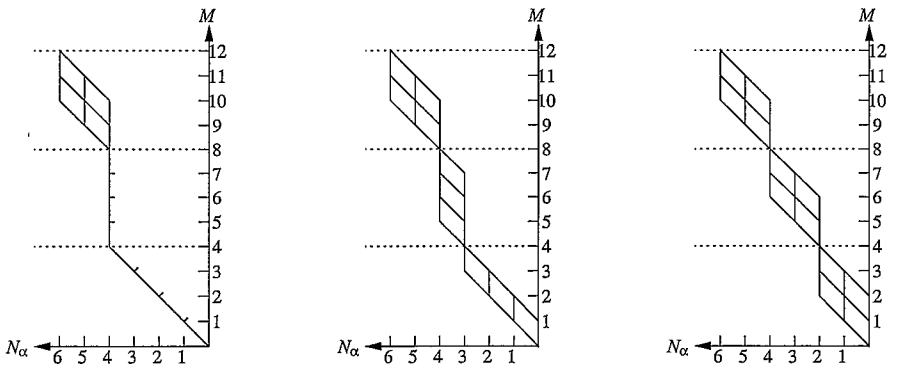
where $\text{Addr}\{K_\alpha\}$ denotes the local string address and the second term is the sum of lengths of all previous categories. The local address can be assigned in the following way: for a given category, $\text{Cat}(i_h, i_e)$, any string can be considered as a combination of appropriate



(a) $i_h = 0, i_e = 0$; Cat = 1 (b) $i_h = 1, i_e = 0$; Cat = 2 (c) $i_h = 2, i_e = 0$; Cat = 3



(d) $i_h = 0, i_e = 1$; Cat = 4 (e) $i_h = 1, i_e = 1$; Cat = 5 (f) $i_h = 2, i_e = 1$; Cat = 6



(g) $i_h = 0, i_e = 2$; Cat = 7 (h) $i_h = 1, i_e = 2$; Cat = 8 (i) $i_h = 2, i_e = 2$; Cat = 9

Figure 3.3.: RAS α-string graphs for $N_\alpha = 6$, $M = 12$ with RAS subspace definitions $M(\text{RAS1}) = 4$; MxHole = 2 and $M(\text{RAS3}) = 4$; MxElec = 2. The number of graphs (=categories) is here $(\text{MxHole} + 1) \times (\text{MxElec} + 1) = 9$.

subspace strings,

$$\begin{aligned}\mathrm{Addr}\{K_\alpha\} &= \left(\mathrm{Addr}\{K_\alpha^{\mathrm{RAS2}}\}-1\right)\binom{M(\mathrm{RAS1})}{i_h}\binom{M(\mathrm{RAS3})}{i_e} \\ &\quad +\left(\mathrm{Addr}\{K_\alpha^{\mathrm{RAS1}}\}-1\right)\binom{M(\mathrm{RAS3})}{i_e} \\ &\quad +\mathrm{Addr}\{K_\alpha^{\mathrm{RAS3}}\}, \end{aligned} \tag{3.22}$$

where each term K_α^{RAS1}, K_α^{RAS2}, K_α^{RAS3} denote the corresponding subspace strings. This is the crucial difference to the addressing scheme used by Olsen *et al.* [14], who chose to not logically separate the addressing of the individual RAS subspaces. The advantage of defining the local string address as in equation 3.22 will become clear shortly.

A choice must be made as to how the subspace strings themselves should be addressed. Since the string subspaces resemble CASSCF string spaces, it is straightforward to again use the indexing formula of Knowles and Handy [42] (equations 3.9 and 3.10), i.e.

$$\mathrm{Addr}\{K_\sigma^{\mathrm{RAS}X}\} = 1+\sum_{k=1}^{N_\sigma} Z(k,l(k)) \tag{3.23}$$

with

$$\begin{aligned} Z(k,l) &= \sum_{m=M(\mathrm{RAS}X)-l+k}^{M(\mathrm{RAS}X)-k}\left[\binom{m}{N_\sigma^{\mathrm{RAS}X}-k}-\binom{m-1}{N_\sigma^{\mathrm{RAS}X}-k-1}\right] \\ &\quad (M(\mathrm{RAS}X)-N_\sigma^{\mathrm{RAS}X}+k\geq l\geq k;\quad k<N_\sigma^{\mathrm{RAS}X}) \\ Z(N_\sigma^{\mathrm{RAS}X},l) &= l-N_\sigma^{\mathrm{RAS}X} \qquad (M(\mathrm{RAS}X)\geq l\geq N_\sigma^{\mathrm{RAS}X}, \end{aligned} \tag{3.24}$$

where $M(\mathrm{RAS}X)$ is the number of substring orbitals in the RASX subspace ($X\in\{1,2,3\}$), and $N_\sigma^{\mathrm{RAS}X}$ the number of electrons (σ denotes spin).

An example may serve to clarify this. Consider a RAS problem with $M=12$ orbitals, $N_\alpha=6$ electrons, 4 orbitals in each RAS subspace, and maximum numbers of holes in RAS1 and particles in RAS3 both equal 2. One α-string that observes these restrictions is $K_\alpha=124578$. Its binary representation is $\phi_{K_\alpha}=0000\,1101\,1011$. The string category is $\mathrm{Cat}(1,0)=2$ (i.e. $i_h=1$, $i_e=0$). Figure 3.4 shows the graphical representation of K_α as a walk on the corresponding RAS graph (cf. figure 3.3(b)). The arc weights are obtained

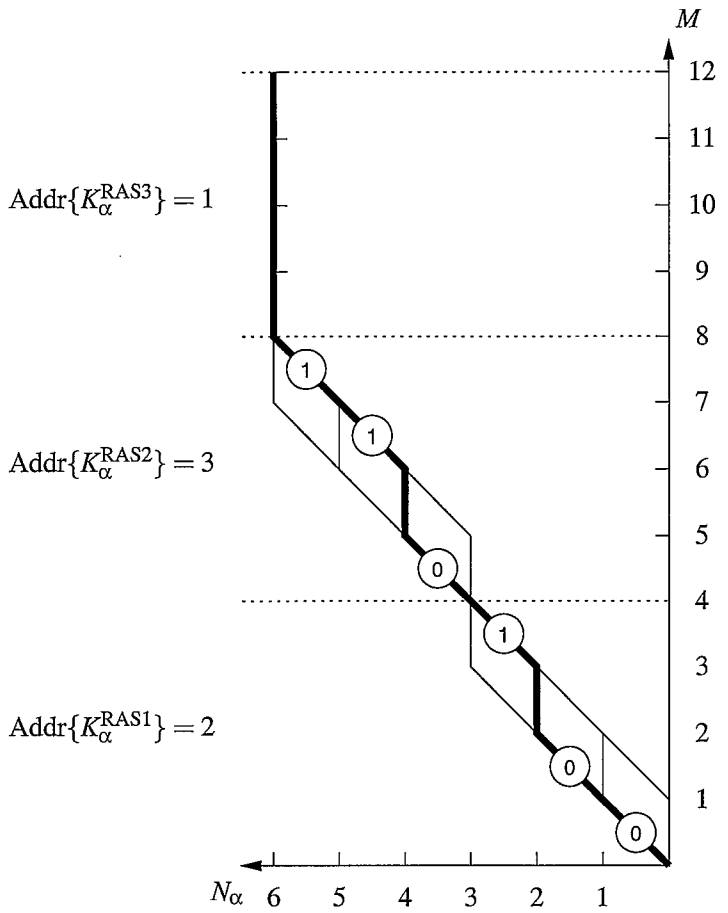


Figure 3.4.: Example of the addressing of a RAS α-string: $M(\text{RAS1}) = M(\text{RAS2}) = M(\text{RAS3}) = 4$, $K_\alpha = 124578$, $\phi_{K_\alpha} = 0000\,1101\,1011$, $i_h = 1$, $i_e = 0$. The numbers on the slopes are the arc weights, Z, for each subspace, as obtained from equation 3.24, and the resulting substring addresses are shown next to the corresponding graph segment. The local string address, Addr$\{K_\alpha\}$, is then obtained from equation 3.22.

individually for each subspace graph (equation 3.24) and the local string address is

$$\begin{aligned}\mathrm{Addr}\{K_\alpha\} &= (2-1)\binom{4}{1}\binom{4}{0}+(3-1)\binom{4}{0}+1\\ &= 4+2+1\\ &= 7.\end{aligned}$$

Only the graph corresponding to Cat = 1 precedes the current RAS graph, i.e. the lexical α-string address $\mathrm{Addr}\{K_\alpha^{i_h,i_e}\}$ is

$$\begin{aligned}\mathrm{Addr}\{K_\alpha^{i_h,i_e}\} &= 7+L[\mathrm{Cat}(0,0)]\\ &= 7+\binom{4}{0}\binom{4}{2}\binom{4}{0}\\ &= 7+6\\ &= 13.\end{aligned}$$

The α-strings cannot be combined freely with all β-strings, because of the restrictions for allowed number of holes ($i_h+i'_h \leq \mathrm{MxHole}$; the prime $'$ signifies β-spin) in RAS1 and allowed number of electrons ($i_e+i'_e \leq \mathrm{MxElec}$) in RAS3. The resulting expression for valid SDs in the RAS expansion can be obtained using once again a two level addressing scheme. The local address for a string pair in a given set of string graphs, $\mathrm{Addr}\{K_\alpha,K_\beta\}$, is given by

$$\mathrm{Addr}\{K_\alpha,K_\beta\} = (\mathrm{Addr}\{K_\alpha\}-1)\times L[\mathrm{Cat}(i'_h,i'_e)]+\mathrm{Addr}\{K_\beta\} \tag{3.25}$$

where $L[\mathrm{Cat}(i'_h,i'_e)]$ is the total number of β-string walks on the corresponding β-string graph. The address of CSF $|K\rangle = |K_\alpha,K_\beta\rangle$ is then

$$\mathrm{Addr}\{K\} \equiv \mathrm{Addr}\{K_\alpha^{i_h,i_e},K_\beta^{i'_h,i'_e}\} = \mathrm{Addr}\{K_\alpha,K_\beta\}+\mathcal{F}_{\mathrm{Cat}}^{\mathrm{Cat}'}, \tag{3.26}$$

where $\mathcal{F}_{\mathrm{Cat}}^{\mathrm{Cat}'}$ is the offset accounting for all CSFs prior to K. It is defined as

$$\mathcal{F}_{\mathrm{Cat}}^{\mathrm{Cat}'} = \sum_{\mathrm{Cat}=1}^{\mathrm{Cat}(i_h,i_e)-1}\ \sum_{\mathrm{Cat}'=1}^{\mathrm{Cat}(\mathrm{MxHole},\mathrm{MxElec})} L[\mathrm{Cat}]\times L[\mathrm{Cat}'] + \sum_{\mathrm{Cat}'=1}^{\mathrm{Cat}(i'_h,i'_e)-1} L[\mathrm{Cat}(i_h,i_e)]\times L[\mathrm{Cat}']. \tag{3.27}$$

Equation 3.25 assigns a unique local address to any combination of paths in a given α-string graph and a β-string graph (there are $L[\mathrm{Cat}(i_h,i_e)]\times L[\mathrm{Cat}(i'_h,i'_e)]$ such combina-

tions). The offset, $\mathcal{F}_{\mathrm{Cat}}^{\mathrm{Cat}'}$, may be precomputed for every allowed graph combination. It accounts for all previous allowed combinations, giving the global address for the SD.

Although we are conducting the discussion of the RAS algorithm on the basis of SDs as basis CSFs, note that it is straightforward to use spin-adapted CSFs. In particular, for Hartree Waller CSFs equation 3.25 turns into (singlet: $S=0$; triplet: $S=1$):

$$\mathrm{Addr}\{K_\alpha,K_\beta\}=\begin{cases}(\mathrm{Addr}\{K_\alpha\}-1)\times L[\mathrm{Cat}(i'_h,i'_e)]+\mathrm{Addr}\{K_\beta\} & \text{if } \mathcal{F}_{\mathrm{Cat}}^{\mathrm{Cat}'}<\mathcal{F}_{\mathrm{Cat}'}^{\mathrm{Cat}}\\ \frac{\mathrm{Addr}\{K_\alpha\}\times(\mathrm{Addr}\{K_\alpha\}-1)}{2}+\mathrm{Addr}\{K_\beta\} & \text{if } \mathcal{F}_{\mathrm{Cat}}^{\mathrm{Cat}'}=\mathcal{F}_{\mathrm{Cat}'}^{\mathrm{Cat}},S=0\\ \frac{(\mathrm{Addr}\{K_\alpha\}-1)\times(\mathrm{Addr}\{K_\alpha 2\}-2)}{2}+\mathrm{Addr}\{K_\beta\} & \text{if } \mathcal{F}_{\mathrm{Cat}}^{\mathrm{Cat}'}=\mathcal{F}_{\mathrm{Cat}'}^{\mathrm{Cat}},S=1\\ (\mathrm{Addr}\{K_\beta\}-1)\times L[\mathrm{Cat}(i_h,i_e)]+\mathrm{Addr}\{K_\alpha\} & \text{if } \mathcal{F}_{\mathrm{Cat}}^{\mathrm{Cat}'}>\mathcal{F}_{\mathrm{Cat}'}^{\mathrm{Cat}}\end{cases} \tag{3.28}$$

where the offset $\mathcal{F}_{\mathrm{Cat}}^{\mathrm{Cat}'}$ has been redefined as

$$\mathcal{F}_{\mathrm{Cat}}^{\mathrm{Cat}'}=\sum_{\mathrm{Cat}=1}^{\mathrm{Cat}(i_h,i_e)-1}\sum_{\mathrm{Cat}'=1}^{\mathrm{Cat}-1}L[\mathrm{Cat}]\times L[\mathrm{Cat}']+\begin{cases}\sum_{\mathrm{Cat}=1}^{\mathrm{Cat}(i_h,i_e)-1}L[\mathrm{Cat}]\times(L[\mathrm{Cat}]+1)/2 & \text{if } S=0\\ \sum_{\mathrm{Cat}=1}^{\mathrm{Cat}(i_h,i_e)-1}L[\mathrm{Cat}]\times(L[\mathrm{Cat}]-1)/2 & \text{if } S=1\end{cases} \tag{3.29}$$

It is apparent from equation 3.22, and perhaps even more so from the graphs in figure 3.3, that the subspace strings K_α^{RAS1}, K_α^{RAS2}, K_α^{RAS3} may be generated as all possible walks on their respective subspace graphs. The task of finding all allowed configurations in the RAS expansion is then simply to find all allowed combinations of subspace graphs.

3.3. The RAS Model Space

In this section we will introduce the RAS *model space*. The idea of a model space is not entirely new in the context of MRCI, and we start out with providing a brief historical background. We will then define a RAS model space, which shares many properties with the CAS orbital space, thus efficiently overcoming many problems associated with the restrictions imposed on the RAS1 and RAS3 orbital subspaces.

Historically, the use of graphical representations of CSFs in CI methods started with the introduction of the Shavitt graph [39, 40]. Shavitt's graphical representation was that of a spin adapted CSF in a single graph, using spatial orbitals with a possible occupancy of up to two electrons. If only electrons of one spin type are present, the graphical representation of strings introduced in section 3.1 and the Shavitt graph become identical. In the Shavitt graph the coupling coefficients A_{ij}^{KL} and B_{ijkl}^{KL} appear as loops. Shavitt was able to show that each loop segment at orbital level m of the graph could be attributed a value,

$W(T_m, b_m)$, determined exclusively by its shape, denoted T_m, and the value b_m, which is related to the accumulative spin, S_m at orbital level m ($b_m = 2S_m$). The one electron coupling coefficient could then be written as the product

$$A_{ij}^{KL} = \prod_{m=\min(i,j)}^{\max(i,j)} W(T_m, b_m) \tag{3.30}$$

The two electron coupling coefficients, B_{ijkl}^{KL} where obtained from the A_{ij}^{KL}, using the resolution of the identity,

$$B_{ijkl}^{KL} = \sum_J A_{ij}^{KJ} A_{kl}^{JL} - \delta_{jk} A_{il}^{KL}. \tag{3.31}$$

Siegbahn realized [59, 64] that Shavitt's segment level factorisation could be used for an efficient MRSD CI algorithm, by simply defining separate segment products for the reference (internal) orbital space and the external orbital space, containing the (in the reference configurations) unoccupied orbitals:

$$A_{ij}^{KL} = I_{ij}^{KL} \times E_{ij}^{KL} \tag{3.32}$$

$$B_{ijkl}^{KL} = I_{ijkl}^{KL} \times E_{ijkl}^{KL} \tag{3.33}$$

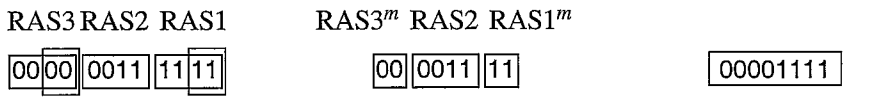
with

$$I_{ij}^{KL} = \prod_{m=1}^{M_{\text{int}}} W(T_m, b_m), \tag{3.34}$$

$$E_{ij}^{KL} = \prod_{m=M_{\text{int}}+1}^{M_{\text{tot}}} W(T_m, b_m), \tag{3.35}$$

where I_{ij}^{KL} comes from the internal (active orbitals) part and E_{ij}^{KL} comes from the external (virtual orbitals) part of the Shavitt graph [40]. The efficiency of this separation into two products comes from the fact that the external vector coupling coefficients are comparatively "easily evaluated and take on very simple values" [59], i.e. the more complicated internal coupling coefficients are being recycled.

In a subsequent paper Saunders and van Lenthe showed [60] that Siegbahn's result could be generalised by introducing a *model space* where the external orbitals are replaced by two model external orbitals. Model CSFs are in this scheme constructed by assigning 0,1 or 2 electrons to the model external space. The vector coupling coefficients are then computed in the model orbital space and remain constant for all CSF pairs K, L with the



(a) Complete RAS string consisting of RAS1, RAS2 and RAS3 substrings.

(b) RAS1 and RAS3 substrings represented by model substrings.

(c) RAS model string.

Figure 3.5.: Example of a RAS α-string consisting of four orbitals in each RAS subspace, and its representing model space string (MxHole = 2, MxElec = 2). The model string is constructed using the RAS2 substring and parts of the RAS1 and RAS3 substrings, of dimension MxHole and MxElec, respectively. The RAS model string has no occupancy restricted subspaces.

same occupation and spin pattern as the corresponding model CSFs. The Saunders/van Lenthe model space is limited to two virtual orbitals and may be used for MRSD CI only. Furthermore it was developed in the context of the underlying segmentation method for spin adapted CSFs. However, using the now familiar determinantal approach, we will show that the model space idea can be extended to treat any RAS problem.

Suppose we would like to construct a RAS wavefunction using the determinantal approach, with a maximum number of holes (MxHole) allowed in the RAS1 space and a maximum number of electrons (MxElec) allowed in the RAS3 space. A model RAS space may then be defined with M^m orbitals, given by

$$M^m = M(\text{RAS2}) + \text{MxHole} + \text{MxElec} \tag{3.36}$$

and $N^m = N_\alpha^m + N_\beta^m$ electrons where

$$N_\alpha^m = N_\alpha - M(\text{RAS1}) + \text{MxHole} \tag{3.37}$$

and

$$N_\beta^m = N_\beta - M(\text{RAS1}) + \text{MxHole}. \tag{3.38}$$

The remaining (spin) orbitals excluded from this model space will always be occupied in the RAS1 subspace and unoccupied in the RAS3 subspace. Figure 3.5 shows a RAS type α-string consisting of RAS1, RAS2 and RAS3 strings, and the model space string representing it (MxHole = 2, MxElec = 2). Note that the number of electrons in the model space is constant (other than in the individual RAS subspaces).

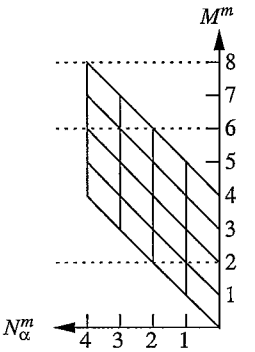


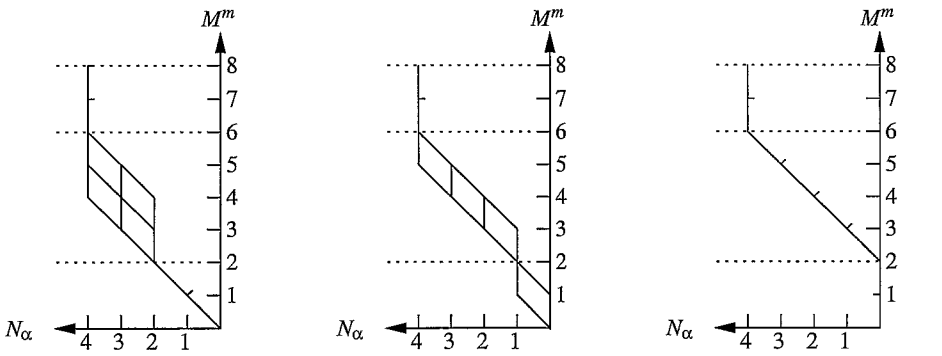
Figure 3.6.: Graph representing the complete α-model string space; $M^m = M(\text{RAS2}) + \text{MxHole} + \text{MxElec}$; ($M(\text{RAS2}) = 4$, $\text{MxHole} = \text{MxElec} = 2$) and $N_\alpha^m = 4$. The model string graph resembles a CAS string graph, due to the absence of occupancy restrictions in the RAS model space.

It is useful to think of the model space as

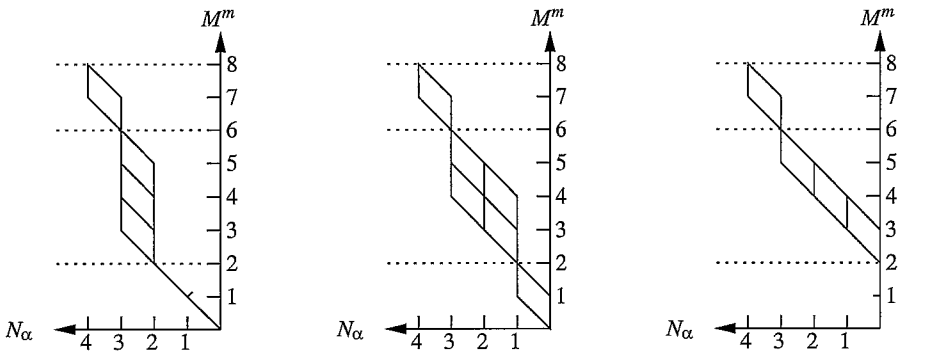
1. a compact way of representing the total RAS space, or
2. a combination of the RAS2 space with compacted RAS1 and RAS3 subspaces.

These two interpretations are visualised in figure 3.6 and figure 3.7, respectively. The most striking feature of representation 1 (figure 3.6) of the model string space is the fact that, although representing restricted string spaces, they are entirely unrestricted and are in fact identical to those used in a CASSCF problem with $M(\text{CAS}) = M^m(\text{RAS})$ orbitals and $N(\text{CAS}) = N^m(\text{RAS})$ electrons. This paves the way for an efficient construction of all model strings, since this is a well studied problem. In particular, it is clear that the method of reduced excitation lists presented in chapter 2 may be applied and we shall return to the details shortly.

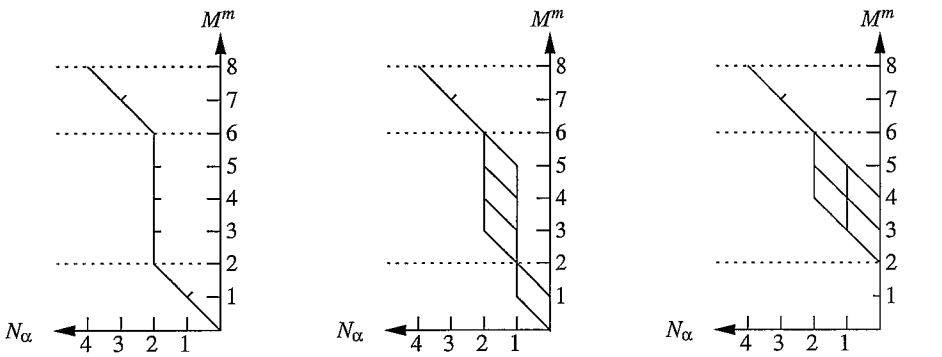
Crucial to the usefulness of the model space idea is representation 2: the subdivision of the model space into the RAS2 subspace and compacted RAS1 and RAS3 model subspaces. Because the dimension of these model subspaces is MxHole and MxElec, respectively, the number of model space graphs in this representation equals the number of categories as given by equation 3.19 for each set of RAS spin graphs. Furthermore, equation 3.18 is not dependent on any absolute number of particles in any RAS subspace, so that there is a one to one correspondence of each *complete* graph in figure 3.3 with one model graph in figure 3.7. This property will be of particular use once a model space string has been



(a) $i_h = 0, i_e = 0$; Cat = 1 (b) $i_h = 1, i_e = 0$; Cat = 2 (c) $i_h = 2, i_e = 0$; Cat = 3



(d) $i_h = 0, i_e = 1$; Cat = 4 (e) $i_h = 1, i_e = 1$; Cat = 5 (f) $i_h = 2, i_e = 1$; Cat = 6



(g) $i_h = 0, i_e = 2$; Cat = 7 (h) $i_h = 1, i_e = 2$; Cat = 8 (i) $i_h = 2, i_e = 2$; Cat = 9

Figure 3.7.: RAS α-model string spaces for $M(\text{RAS2}) = 4$, MxHole = 2 and MxElec = 2. Each graph corresponds to one string graph in figure 3.3, with equal string category. The superposition of all restricted model graphs gives the unrestricted RAS model string graph of figure 3.6.

found, and is central to the method described in this chapter. Clearly the superposition of all graphs in figure 3.7 will simply give the graph in figure 3.6. The idea is simple: once a model string is found, it may be associated with a certain model string space. It is then by implication also associated with a certain string graph in the complete RAS space.

In order to avoid confusion between orbital labels of different RAS subspaces, the model space and the full RAS space, we will now define the orbital indices used for labelling those respective orbitals.

- RAS1: a,b,c,d
- RAS2: i,j,k,l
- RAS3: p,q,r,s
- model space: w,x,y,z
- full RAS space: i,j,k,l

Note that we use the same set of indices i,j,k,l for both, the full RAS space, and the RAS2 subspace. The reason for this is that it is (at different times) useful to liken both sets of orbitals to the active space of a CASSCF. It will always be clear from the context which orbital set is meant.

We have already mentioned that the unrestricted character of the model string space may be exploited, since the problem of finding model string pairs $\{K^m_\sigma, L^m_\sigma\}$ closely resembles the corresponding problem for full CI. This problem is well studied and a number of efficient direct methods have been proposed (e.g. [14, 44, 45, 65]). We will here use the method of reduced excitation strings. We have introduced this method in chapter 2 to entirely avoid index space testing in direct full CI / CASSCF computations. However, the model space and reduced string method are not interdependent, and any direct method tackling the indexing problem in full CI could be used. The reduced list concept may be applied here without change, and we will now briefly remind ourselves of the method, while placing it in the current context.

The set of orbitals under consideration is restricted to those contained in the RAS model space. Thus, given the σ-model space orbital indices w,x (σ signifies spin), we define the model space excitation list $\mathcal{X}^{x\sigma}_{w\sigma}$ containing, in lexical order, all string pair addresses $\mathrm{Addr}\{K^m_\sigma\}$, $\mathrm{Addr}\{L^m_\sigma\}$ and $\mathrm{sgn}^{K^m_\sigma}_{wx}$ for which the relation

$$\langle K^m_\sigma | \hat{a}^\dagger_{w\sigma} \hat{a}_{x\sigma} | L^m_\sigma \rangle = \mathrm{sgn}^{K^m_\sigma}_{wx} \tag{3.39}$$

holds, with

$$\mathrm{sgn}_{wx}^{K_\sigma^m} = \begin{cases} +1 & \text{if } \sum_{n=x+1}^{w-1} b_n \text{ is even} \\ -1 & \text{if } \sum_{n=x+1}^{w-1} b_n \text{ is odd} \end{cases} \tag{3.40}$$

where b_n is the bit value of the n^{th} bit in the binary representation of K_σ^m, $\phi_{K_\sigma^m}$. The sums represent the number of occupied orbitals (i.e. $b_n = 1$) between orbitals w and x. By eliminating bits b_w and b_x from the binary representation of each string we obtain a list of reduced string pairs, where each pair consists of two identical reduced strings. The list of unique reduced strings is denoted $\mathcal{L}_{N_\sigma^m-1}^{M^m-2}$. Since the number of electrons in the σ-model space is constant, reinserting bits b_w and b_x for arbitrary w,x ($w \neq x$) yields the corresponding model space excitation list. Similarly, for 2e excitations a model space excitation list $\mathcal{X}_{w\sigma y\sigma}^{x\sigma z\sigma}$ is defined, containing all string pair addresses $\mathrm{Addr}\{K_\sigma^m\}$, $\mathrm{Addr}\{L_\sigma^m\}$ and $\mathrm{sgn}_{wxyz}^{K_\sigma^m}$ for which the relation

$$\langle K_\sigma^m | \hat{a}_{w\sigma}^\dagger \hat{a}_{y\sigma}^\dagger \hat{a}_{z\sigma} \hat{a}_{x\sigma} | L_\sigma^m \rangle = \mathrm{sgn}_{wxyz}^{K_\sigma^m} \tag{3.41}$$

holds, with

$$\mathrm{sgn}_{wxyz}^{K_\sigma^m} = \begin{cases} -\mathrm{sgn}_{wx}^{K_\sigma^m} \times \mathrm{sgn}_{yz}^{K_\sigma^m} & \text{if } w > y > x > z \\ +\mathrm{sgn}_{wx}^{K_\sigma^m} \times \mathrm{sgn}_{yz}^{K_\sigma^m} & \text{otherwise.} \end{cases} \tag{3.42}$$

Again, by eliminating b_w, b_x, b_y and b_z from the binary representation of each string we obtain a reduced list of strings that can be used to construct all excitation lists by inserting the corresponding bits $b_w, \ldots, b_z$. To accommodate for all possible 1e and 2e excitations we need only five reduced lists, which may be precomputed: $\mathcal{L}_{N_\sigma^m-1}^{M^m-1}$, $\mathcal{L}_{N_\sigma^m-1}^{M^m-2}$, $\mathcal{L}_{N_\sigma^m-2}^{M^m-2}$, $\mathcal{L}_{N_\sigma^m-2}^{M^m-3}$, $\mathcal{L}_{N_\sigma^m-2}^{M^m-4}$. Thus all model space excitation lists in the model space may be constructed in this fashion, and the efficiency of this method comes from the fact that no redundant index space testing is needed in order to find the needed string pairs.

Figure 3.8 shows an example for the reconstruction of one string pair K_α^m, L_α^m, for the model space excitation $\hat{a}_{6\alpha}^\dagger \hat{a}_{3\alpha}$ and using an arbitrary reduced model string. The resulting walks for K_α^m and L_α^m form a loop on the model string graph. For the assembly of the model excitation list $\mathcal{X}_{3\alpha}^{6\alpha}$ the complete set of walks of reduced model strings (in lexical order), i.e. all entries of $\mathcal{L}_{N_\sigma^m-1}^{M^m-2}$, will be used in the same manner. Once a model string pair K_σ^m, L_σ^m is found, each model string may be associated with one category, simply by counting the number of holes/electrons in the RAS1 and RAS3 model substrings. In graphical terms this means that for each walk corresponding to a σ-model string, we need to find the σ-model string space graph (cf. figure 3.7) that has the correct dimension to superimpose the walk onto it. If the loop stretches over more than one RAS subspace,

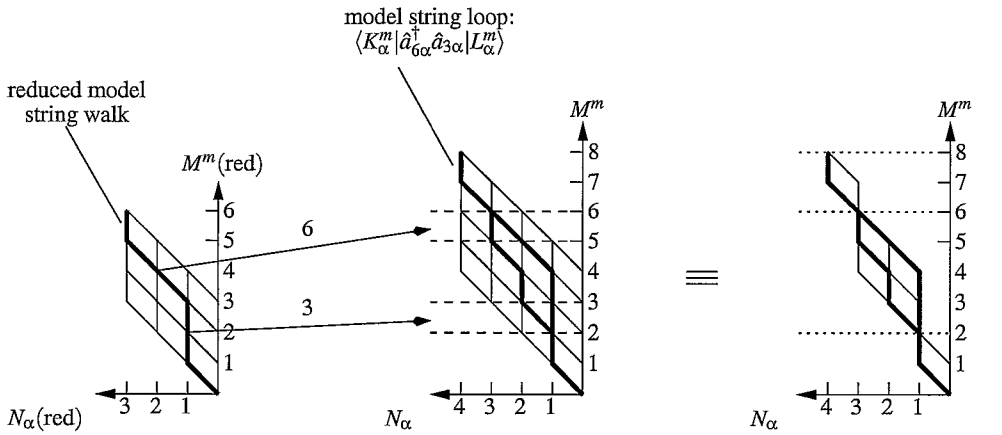


Figure 3.8.: Construction of RAS model string pair K_α^m, L_α^m from a reduced model string, for the model space excitation $\hat{a}_{6\alpha}^\dagger\hat{a}_{3\alpha}$. The string K_α^m is represented by the left walk of the loop.

the model strings K_σ^m, L_σ^m might then belong to different model space graphs (categories). In section 3.4 we will show how to expand the model substrings. However, the RAS2 substrings, K_σ^{RAS2} and L_σ^{RAS2} are already in final form.

At this point we should mention an additional benefit of using the model space approach. In traditional implementations of the RASSCF method (e.g. [14,61]), one has to deal with the possibility of *out of space* excitations, i.e. the excitation $\hat{a}_{i\sigma}^\dagger\hat{a}_{j\sigma}|L_\sigma\rangle$ could result in a string $|K_\sigma\rangle$ whose number of holes in RAS1 or number of electrons in RAS3 exceed the limits, namely MxHole and MxElec, respectively. Due to the resemblance of the RAS model space to the string space for a full CI it is, by construction, not possible to produce an *out of space* excitation, and the problem is entirely avoided. This holds true also for double excitations.

3.4. Expansion of the Model RAS Subspaces

In the previous section we have outlined, in a schematic fashion, how a model space may be constructed for an arbitrary RAS wavefunction. Each walk on a model string graph represents a set of strings on the full graph. Correspondingly, each pair of walks on the model string graph, related by a single ($\hat{a}_{w\sigma}^\dagger\hat{a}_{x\sigma}$) or double ($\hat{a}_{w\sigma}^\dagger\hat{a}_{y\sigma}^\dagger\hat{a}_{z\sigma}\hat{a}_{x\sigma}$) excitation and thus building a loop, represents a set of string pairs on the full graph. We now need to translate each model string pair K_σ^m, L_σ^m, together with its associated sign, $\mathrm{sgn}_{wx}^{K_\sigma^m}$, into a

set of RAS string pairs and their sign, $\{K_\sigma, L_\sigma, \mathrm{sgn}_{ij}^{K_\sigma}\}$. While the RAS2 substring is fully determined by the model string, the RAS1 and RAS3 substrings need to be evaluated separately by expanding the model substrings.

If we required MxHole = 0 in RAS1 and MxElec = 2 in RAS3, then the problem would be identical to that described by Saunders and van Lenthe [60] and we would always have $\mathrm{sgn}_{wx}^{K_\sigma^m} = \mathrm{sgn}_{ij}^{K_\sigma}$ (and $\mathrm{sgn}_{wxyz}^{K_\sigma^m} = \mathrm{sgn}_{ijkl}^{K_\sigma}$). By lifting all restrictions on MxHole and MxElec on the other hand, $\mathrm{sgn}_{ij}^{K_\sigma}$ ($\mathrm{sgn}_{ijkl}^{K_\sigma}$) may change,

$$\begin{aligned} \mathrm{sgn}_{ij}^{K_\sigma} &= \pm \mathrm{sgn}_{wx}^{K_\sigma^m}, \\ \mathrm{sgn}_{ijkl}^{K_\sigma} &= \pm \mathrm{sgn}_{wxyz}^{K_\sigma^m}. \end{aligned} \tag{3.43}$$

In other words, we combine the original idea of Siegbahn [59,64] (factorising internal and external contributions of vector coupling coefficients) with that of Saunders/van Lenthe (model space) and write

$$\begin{aligned} \langle K_\sigma | \hat{a}_{i\sigma}^\dagger \hat{a}_{j\sigma} | L_\sigma \rangle &= \mathrm{sgn}_{ij}^{K_\sigma} \\ &= \mathrm{sgn}_{wx}^{K_\sigma^m} \times \mathrm{sgn}_{ij}^{K_\sigma^{\mathrm{RAS1}}} \times \mathrm{sgn}_{ij}^{K_\sigma^{\mathrm{RAS3}}}, \end{aligned} \tag{3.44}$$

$$\begin{aligned} \langle K_\sigma | \hat{a}_{i\sigma}^\dagger \hat{a}_{k\sigma}^\dagger \hat{a}_{l\sigma} \hat{a}_{j\sigma} | L_\sigma \rangle &= \mathrm{sgn}_{ijkl}^{K_\sigma} \\ &= \mathrm{sgn}_{wxyz}^{K_\sigma^m} \times \mathrm{sgn}_{ijkl}^{K_\sigma^{\mathrm{RAS1}}} \times \mathrm{sgn}_{ijkl}^{K_\sigma^{\mathrm{RAS3}}}, \end{aligned} \tag{3.45}$$

where the factors $\mathrm{sgn}_{ij}^{K_\sigma^{\mathrm{RASX}}}$ and $\mathrm{sgn}_{ijkl}^{K_\sigma^{\mathrm{RASX}}}$ ($X \in \{1,3\}$) are potentially variables that can take the values ± 1. This strategy would include the computation of the external coefficients within the innermost loops, and hence their computation needs to be optimised.

In the next subsection we introduce the idea behind the second contribution of this chapter. For the sake of simplicity we restrict ourselves at this stage to the 1e excitation terms, although it will be seen later that the extension to 2e excitation terms is straightforward.

3.4.1. An efficient expansion technique: Propagation rules

The second contribution of this chapter is the concept of *propagation rules*. The aim of the propagation rules is to efficiently compute all string pairs, $\{K_\sigma, L_\sigma\}$, that give nonzero contributions for a given excitation term $\langle K_\sigma | \hat{a}_{i\sigma}^\dagger \hat{a}_{j\sigma} | L_\sigma \rangle$. In order to introduce the concept we now present four representative examples for single excitations within and between RAS subspaces. They are designed to provide insight into the structure of the

		$K_\alpha^{0,1}$				$L_\alpha^{0,1}$			
i	j	RAS3	RAS2	RAS1	Addr$\{K_\alpha^{\mathrm{RAS3}}\}$	RAS3	RAS2	RAS1	Addr$\{L_\alpha^{\mathrm{RAS3}}\}$
7	5	0001	0100	1111	1	0001	0001	1111	1
7	5	0010			2	0010			2
7	5	0100			3	0100			3
7	5	1000			4	1000			4

Figure 3.9.: Example 1: One electron excitation within the RAS2 orbital subspace (RAS2 $\rightarrow$ RAS2).

RAS string pairs that occur as a result of the corresponding excitations within the model string space. The examples given below are all based on a total RAS space of 12 orbitals ($M(\mathrm{RAS1}) = M(\mathrm{RAS2}) = M(\mathrm{RAS3}) = 4$), $N_\alpha = 6$ electrons and MxHole = MxElec = 2 (cf. figure 3.3 and 3.7). In order to arrive at the string pair $K_\alpha^{i_h,i_e}$, $L_\alpha^{i'_h,i'_e}$, we need to combine a (fixed) RAS2 substring with (variable) RAS1 and RAS3 substrings. We will use the notation $X \rightarrow Y$ to indicate an excitation from the RASX subspace into the RASY subspace, e.g. $1 \rightarrow 3$ for an excitation from RAS1 to RAS3. Subsequently we will show how to efficiently derive all subsequent substring addresses, $K_\alpha^{\mathrm{RAS1}}(n)$, once the first substring address, $K_\alpha^{\mathrm{RAS1}}(1)$, is known.

Example 1: $2 \rightarrow 2$ The first example is the model space excitation $\hat{a}^\dagger_{w\alpha}\hat{a}_{x\alpha}$, with $w = 5$, $x = 3$. We start with the (arbitrary) reduced model string 010011 from the reduced list $\mathcal{L}^{M^m-2}_{N^m_\alpha-1}$. By inserting the bits b_3, b_5 into the appropriate positions of the model strings (K^m_α: $b_3 = 0$, $b_5 = 1$; L^m_α: $b_3 = 1$, $b_5 = 0$), we arrive at a pair of model strings, $K^m_\alpha =$ 01 0100 11 and $L^m_\alpha =$ 01 0001 11. In both model strings, the number of holes in the RAS1 model subspace is $i_h = 0$, and the number of electrons in the RAS3 model subspace is $i_e = 1$, corresponding to an α-model spin graph as shown in figure 3.7(d). Now we know that all RAS α-string pairs will be of the type $K_\alpha^{0,1}$ and $L_\alpha^{0,1}$, i.e. there are no holes in RAS1, and one electron in RAS3. This in turn determines that there are $\binom{M(\mathrm{RAS1})}{i_h} \times \binom{M(\mathrm{RAS3})}{i_e} = 4$ α-string pairs $\{K_\alpha, L_\alpha\}$ to be found. It is clear that the RAS1 substring is constant (since $i_h = 0$) and that the RAS3 strings are identical for K_α^{RAS3} and L_α^{RAS3}, since the excitation does not involve orbitals in the RAS3 subspace. Note also that the corresponding integral and sign, $(i|j)$ and sgn^{KL}_{ij}, are constant for all CSF pairs of this excitation. Figure 3.9 shows how the string pairs may be constructed. The indices i and j label the orbitals of the total active space. The substring address Addr$\{K_\alpha^{\mathrm{RAS3}}\}$ is evaluated according to equation 3.23. However, since the labels i and j are both outside

		$K_\alpha^{1,0}$				$L_\alpha^{0,0}$			
i	j	RAS3	RAS2	RAS1	Addr$\{K_\alpha^{\mathrm{RAS1}}\}$	RAS3	RAS2	RAS1	Addr$\{L_\alpha^{\mathrm{RAS1}}\}$
7	1	0000	0111	1110	4	0000	0011	1111	1
7	2			1101	3			1111	1
7	3			1011	2			1111	1
7	4			0111	1			1111	1

Figure 3.10.: Example 2: One electron excitation between two RAS orbital subspaces (RAS1 $\rightarrow$ RAS2).

the RAS3 subspace, it is known from the outset that *all* strings with four orbitals and one electron need to be considered. If the first Addr$\{K_\alpha^{\mathrm{RAS3}}\}$ is known then, instead of *computing* subsequent address pairs we may simply use

$$\begin{aligned} \mathrm{Addr}\{K_\alpha^{\mathrm{RAS3}}(n)\} &= \mathrm{Addr}\{K_\alpha^{\mathrm{RAS3}}(n-1)\}+1 \\ \mathrm{Addr}\{L_\alpha^{\mathrm{RAS3}}(n)\} &= \mathrm{Addr}\{K_\alpha^{\mathrm{RAS3}}(n)\}, \end{aligned}$$

where n indicates the recursive nature of this equation.

Example 2: $1 \rightarrow 2$ For our second example let $w = 5$, $x = 1$. Inserting b_w and b_x into the reduced model string 000111 yields $K_\alpha^m =$ 00 0111 10 and $L_\alpha^m =$ 00 0011 11, which carries the information of the categories: $K_\alpha^{1,0}$ and $L_\alpha^{0,0}$. The type of excitation ($1 \rightarrow 2$) again implies that the RAS3 substrings are identical in both RAS configurations (in this case only one list element). However, the RAS1 strings K_α^{RAS1} are now different from L_α^{RAS1}. Furthermore, the $(i|j)$ and sgn_{ij}^{KL} will not be constant, because j is assuming each value within $M(\mathrm{RAS1})$. Figure 3.10 shows an example for excitations of this type. As in the previous example, the addresses Addr$\{K_\alpha^{\mathrm{RAS1}}\}$ and Addr$\{L_\alpha^{\mathrm{RAS1}}\}$ may be computed using Handy's index equation. But from the addresses for consecutive strings one may suspect that there is, again, a simpler way of producing them. For example, one may use

$$\begin{aligned} \mathrm{Addr}\{K_\alpha^{\mathrm{RAS1}}(a)\} &= \mathrm{Addr}\{K_\alpha^{\mathrm{RAS1}}(a-1)\}+1 \\ \mathrm{Addr}\{L_\alpha^{\mathrm{RAS1}}(a)\} &= \mathrm{Addr}\{L_\alpha^{\mathrm{RAS1}}(a-1)\}, \end{aligned}$$

where a equals the position of the varying bit, b_a. Obviously this relation only holds if the position of b_a follows a strict right to left rule for consecutive string pairs.

		$K_\alpha^{1,1}$				$L_\alpha^{1,1}$			
i	j	RAS3	RAS2	RAS1	Addr$\{K_\alpha^{\mathrm{RAS3}}\}$	RAS3	RAS2	RAS1	Addr$\{L_\alpha^{\mathrm{RAS3}}\}$
10	9	0010	0011	0111	2	0001	0011	0111	1
11	9	0100		0111	3	0001		0111	1
12	9	1000		0111	4	0001		0111	1
11	10	0100		0111	3	0010		0111	2
12	10	1000		0111	4	0010		0111	2
12	11	1000		0111	4	0100		0111	3
10	9	0010		1011	2	0001		1011	1
⋮	⋮	⋮		⋮	⋮	⋮		⋮	⋮
12	11	1000		1110	4	1000		1110	3

Figure 3.11.: Example 3: One electron excitation within the RAS3 orbital subspace (RAS3 $\rightarrow$ RAS3).

Example 3: $3 \rightarrow 3$ Consider a single excitation within a RAS subspace other than the RAS2 subspace. Let $w = 8$, $x = 7$ and $K_\alpha^m =$ |10|0011|01| and $L_\alpha^m =$ |01|0011|01|. The categories are now defined by $K_\alpha^{1,1}$ and $L_\alpha^{1,1}$. Clearly the sgn_{ij}^{KL} will be +1 for all string pairs, since there are no other electrons in RAS3. The integral $(i|j)$ on the other hand will change with i and j. There is a choice of RAS1 strings, and the total number of contributions for this pair of model space strings is $\binom{M(\mathrm{RAS1})}{i_h} \times M(\mathrm{RAS1}) = 24$. Figure 3.11 shows that here both the RAS1 and RAS3 strings vary. Since we have now two variables in RAS3, the resulting pattern for Addr$\{K_\alpha^{\mathrm{RAS3}}\}$ and Addr$\{L_\alpha^{\mathrm{RAS3}}\}$ at first sight look slightly more erratic. However, if the two varying indices in RAS3 are p and q ($p > q$) and the first address pair, Addr$\{K_\alpha^{\mathrm{RAS3}}(2,1)\}$ and Addr$\{L_\alpha^{\mathrm{RAS3}}(2,1)\}$, is known, then all consecutive address pairs may still be obtained iteratively using

$\Delta p = 1, \Delta q = 0$:

$$\begin{aligned} \mathrm{Addr}\{K_\alpha^{\mathrm{RAS3}}(p,q)\} &= \mathrm{Addr}\{K_\alpha^{\mathrm{RAS3}}(p-1,q)\}+1 \\ \mathrm{Addr}\{L_\alpha^{\mathrm{RAS3}}(p,q)\} &= \mathrm{Addr}\{L_\alpha^{\mathrm{RAS3}}(p-1,q)\} \end{aligned}$$

$\Delta p = 1, \Delta q = 1$:

$$\begin{aligned} \mathrm{Addr}\{K_\alpha^{\mathrm{RAS3}}(p,q)\} &= \mathrm{Addr}\{K_\alpha^{\mathrm{RAS3}}(p-1,q-1)\}+1 \\ \mathrm{Addr}\{L_\alpha^{\mathrm{RAS3}}(p,q)\} &= \mathrm{Addr}\{L_\alpha^{\mathrm{RAS3}}(p-1,q-1)\}+1, \end{aligned}$$

where the RAS3 string address is a simple recursive function of the varying indices. Again

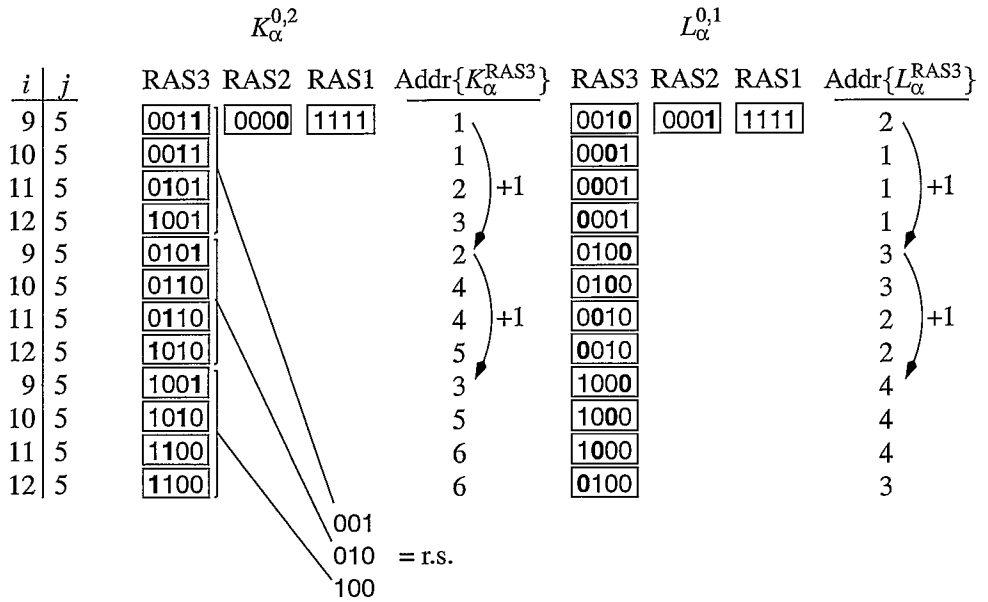


Figure 3.12.: Example 4: One electron excitation into already populated model subspace (RAS2 → RAS3).

a strict right to left migration of the moving bits b_p and b_q is essential.

Example 4: $2 \rightarrow 3$ A slight complication arises when an electron is excited into an already populated RAS3 subspace (or out of an already partly depopulated RAS1 subspace). Figure 3.12 shows such case for $w = 7$, $x = 3$ and $K_\alpha^m =$ |11|0000|11| and $L_\alpha^m =$ |10|0001|11| ($K_\alpha^{0,2}$ and $L_\alpha^{0,1}$). The sign, sgn_{ij}^{KL}, has the potential to change in this case. The integral $(i|j)$ on the other hand will change with i. The total number of contributions for this pair of model space strings is $\binom{M(\mathrm{RAS1})}{i_h} \times M(\mathrm{RAS3}) \times \binom{M(\mathrm{RAS3})-1}{i_e-1} = 12$, where the last term is the binomial coefficient for the *reduced string list* of the RAS3 subspace. The reduced string lists closely resembles those introduced in chapter 2, except for the fact that here they are constructed from a RAS subspace where the total number of particles is not conserved. In order to avoid a complicated labelling scheme that takes account of this, we will use a simplified notation, r.s. (short for *reduced string*) and it will always be clear from the context which reduced string is meant.

In figure 3.12 the RAS3 strings constructed from a reduced string have been grouped and ordered accordingly. The reduced strings are also shown for clarity. If Addr{$K_\alpha^{RAS3}(p;\mathrm{r.s.})$},

$\mathrm{Addr}\{L_\alpha^{\mathrm{RAS3}}(p;\mathrm{r.s.})\}$ are known for the first reduced string and $p=1$, it can be shown that

$$\begin{aligned}\mathrm{Addr}\{K_\alpha^{\mathrm{RAS3}}(p;\mathrm{r.s.})\} &= \mathrm{Addr}\{K_\alpha^{\mathrm{RAS3}}(p-1;\mathrm{r.s.})\}+\binom{M(\mathrm{RAS3})-p}{N_\alpha(M^{\mathrm{left}}(p))}\times(1-b_{p-1}(\mathrm{r.s.}))\\ \mathrm{Addr}\{L_\alpha^{\mathrm{RAS3}}(p;\mathrm{r.s.})\} &= \mathrm{Addr}\{L_\alpha^{\mathrm{RAS3}}(p-1;\mathrm{r.s.})\}-\binom{M(\mathrm{RAS3})-p}{N_\alpha(M^{\mathrm{left}}(p))}\times b_{p-1}(\mathrm{r.s.}),\end{aligned}$$

where $N_\alpha(M^{\mathrm{left}}(p))$ is the number of α-electrons to the left of orbital p of the RAS3 subspace, and $b_n(\mathrm{r.s.})$ indicates the n^{th} bit in the r.s. For subsequent reduced strings it is clear from equation 2.10 that

$$\begin{aligned}\mathrm{Addr}\{K_\alpha^{\mathrm{RAS3}}(1;\mathrm{r.s.}')\} &= \mathrm{Addr}\{K_\alpha^{\mathrm{RAS3}}(1;\mathrm{r.s.})\}+1\\ \mathrm{Addr}\{L_\alpha^{\mathrm{RAS3}}(1;\mathrm{r.s.}')\} &= \mathrm{Addr}\{L_\alpha^{\mathrm{RAS3}}(1;\mathrm{r.s.})\}+1.\end{aligned}$$

The examples shown above indicate an efficient algorithm for the computation of the remaining RAS1/RAS3 string addresses, if the first address of a given list is known. It is necessary to know only a number of reduced RAS1 and RAS3 string lists and some binomial coefficients, both of which may be precomputed at certain stages of the computation. The outlined scheme avoids redundant index space testing in a similar fashion the reduced list approach developed in chapter 2 does for unrestricted CI. The algorithm is based on the systematic propagation of substring variables a,b,c,d (or p,q,r,s) over the entries of a reduced string list of the RAS1 (or RAS3) orbital space.

The full set of propagation rules for 1e and 2e excitations is given in appendix B. However, it is straightforward to generalise the equations to only five cases, reflecting the possible 0,1,2,3, or 4 variable indices in an external RAS subspace:

0 indices:

$$\mathrm{Addr}\{K_\gamma^{\mathrm{RAS}X}(n)\}=\mathrm{Addr}\{K_\gamma^{\mathrm{RAS}X}(n-1)\}+1 \qquad \forall n\in\{2,\ldots,\binom{M(\mathrm{RAS}X)}{N_\gamma(K_\gamma^{\mathrm{RAS}X})}\} \quad (3.46)$$

1 index (a):

$$\mathrm{Addr}\{K_\gamma^{\mathrm{RAS}X}(n)\}=\mathrm{Addr}\{K_\gamma^{\mathrm{RAS}X}(n-1)\}+\delta_{1\Delta a}\times\binom{M(\mathrm{RAS}X)-a}{N_\gamma(M^{\mathrm{left}}(a))}\times(b_a-b_{a-1}(\mathrm{r.s.})) \quad (3.47)$$

2 indices (a, b):

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RAS}X}(n)\} = \; & \mathrm{Addr}\{K_\gamma^{\mathrm{RAS}X}(n-1)\} \\ & + \delta_{1\Delta a} \times \binom{M(\mathrm{RAS}X) - a}{N_\gamma(M^{\mathrm{left}}(a))} \times (b_a - b_{a-2}(\mathrm{r.s.})) \\ & + \delta_{1\Delta b} \times \binom{M(\mathrm{RAS}X) - b}{N_\gamma(M^{\mathrm{left}}(b))} \times (b_b - b_{a-2}(\mathrm{r.s.}))\end{aligned} \tag{3.48}$$

3 indices (a, b, c):

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RAS}X}(n)\} = \; & \mathrm{Addr}\{K_\gamma^{\mathrm{RAS}X}(n-1)\} \\ & + \delta_{1\Delta a} \times \binom{M(\mathrm{RAS}X) - a}{N_\gamma(M^{\mathrm{left}}(a))} \times (b_a - b_{a-3}(\mathrm{r.s.})) \\ & + \delta_{1\Delta b} \times \binom{M(\mathrm{RAS}X) - b}{N_\gamma(M^{\mathrm{left}}(b))} \times (b_b - b_{a-3}(\mathrm{r.s.})) \\ & + \delta_{1\Delta c} \times \binom{M(\mathrm{RAS}X) - c}{N_\gamma(M^{\mathrm{left}}(c))} \times (b_c - b_{a-3}(\mathrm{r.s.}))\end{aligned} \tag{3.49}$$

4 indices (a, b, c, d):

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RAS}X}(n)\} = \; & \mathrm{Addr}\{K_\gamma^{\mathrm{RAS}X}(n-1)\} \\ & + \delta_{1\Delta a} \times \binom{M(\mathrm{RAS}X) - a}{N_\gamma(M^{\mathrm{left}}(a))} \times (b_a - b_{a-4}(\mathrm{r.s.})) \\ & + \delta_{1\Delta b} \times \binom{M(\mathrm{RAS}X) - b}{N_\gamma(M^{\mathrm{left}}(b))} \times (b_b - b_{a-4}(\mathrm{r.s.})) \\ & + \delta_{1\Delta c} \times \binom{M(\mathrm{RAS}X) - c}{N_\gamma(M^{\mathrm{left}}(c))} \times (b_c - b_{a-4}(\mathrm{r.s.})) \\ & + \delta_{1\Delta d} \times \binom{M(\mathrm{RAS}X) - d}{N_\gamma(M^{\mathrm{left}}(d))} \times (b_d - b_{a-4}(\mathrm{r.s.})).\end{aligned} \tag{3.50}$$

These equations may be further generalised by introducing the *propagator function*

$$p(u) = \delta_{1\Delta u} \times \binom{M(\mathrm{RAS}X) - u}{N_\gamma(M^{\mathrm{left}}(u))} \times (b_u - b_{a-\xi}(\mathrm{r.s.})), \tag{3.51}$$

where $u \in \{a,b,c,d\}$ and the parameter ξ is the number of variable indices in the RASX

Table 3.1.: The addresses of RAS1/RAS3 substrings for consecutive reduced strings are themselves consecutive numbers. This follows directly from the strict left-to-right ordering of strings on the corresponding unrestricted substring graph.

$\phi_{K_\sigma^{\mathrm{RAS}X}}$	$\mathrm{Addr}\{K_\sigma^{\mathrm{RAS}X}\}$	$\Delta\mathrm{Addr}\{K_\sigma^{\mathrm{RAS}X}\}$
001**101**	5	-
010**101**	6	1
100**101**	7	1
011**001**	8	1
101**001**	9	1
110**001**	10	1

substring. The propagation rules, $\mathcal{P}$, for the RASX subspace may then be rewritten as

$$\mathcal{P}(\mathrm{RAS}X): \qquad \mathrm{Addr}\{K_\gamma^{\mathrm{RAS}X}(n,\mathrm{r.s.})\} = \mathrm{Addr}\{K_\gamma^{\mathrm{RAS}X}(n-1,\mathrm{r.s.})\} + \sum_{\nu=1}^{\xi} p(u(\nu)), \tag{3.52}$$

for a given reduced string (r.s.). For the purpose of referring to a particular propagation rule we will from now on adopt the notation $\mathcal{P}_{b_a^L}^{b_a^K}$ for the propagation rule with variable bit b_a^L in substring $L_\gamma^{\mathrm{RAS}X}$, and variable bit b_a^K in $K_\gamma^{\mathrm{RAS}X}$. Accordingly, $\mathcal{P}_{b_a^L b_b^L}^{b_a^K b_b^K}$ will signify variable bits b_a^L and b_b^L in substring $L_\gamma^{\mathrm{RAS}X}$, and bits b_a^K and b_b^K in $K_\gamma^{\mathrm{RAS}X}$ etc.

When proceeding to the next reduced string (in the case of 0 indices in RASX each entry may be interpreted as a reduced string), the relationship is trivial, as shown above,

$$\mathrm{Addr}\{K_\gamma^{\mathrm{RAS}X}(1,\mathrm{r.s.}')\} = \mathrm{Addr}\{K_\gamma^{\mathrm{RAS}X}(1,\mathrm{r.s.})\} + 1. \tag{3.53}$$

To demonstrate the usefulness of the propagation rules we need to show that advantage can be taken of them in the 2e excitation (four orbital index) part of equation 2.7. We will continue the discussion in subsection 3.4.3, but first we provide a proof for the propagation rules.

Proof of Propagation Rule I (equation 3.53)

Table 3.1 shows a series of strings, starting with $\boxed{001101}$. They are constructed from the complete list of strings generated from two 0s and two 1s, ordered according to Handy's index equation (equation 3.10) and supplemented by the short string **01** to the right. The

Table 3.2.: Propagation of the variable bits from left to right through all bit-positions of a fixed reduced string (01011) results in a change of substring address only if the two interchanged bits, b_a and b_{a-1}, differ. The number by which the address changes corresponds to a pre-computable binomial coefficient.

a	$\phi_{K_\sigma^{\mathrm{RASX}}}$	$\phi_{L_\sigma^{\mathrm{RASX}}}$	b_{a-1}	$\binom{M(\mathrm{RASX})-a}{N_\sigma^{\mathrm{left}}(a)}$	$\mathrm{Addr}\{K_\sigma^{\mathrm{RASX}}\}$	ΔK	$\mathrm{Addr}\{L_\sigma^{\mathrm{RASX}}\}$	ΔL
1	010111	010110	n/a	10	2	-	12	-
2	010111	010101	1	6	2	0	6	-6
3	010111	010011	1	3	2	0	3	-3
4	011011	010011	0	2	4	2	3	0
5	011011	001011	1	1	4	0	2	-1
6	101011	001011	0	1	5	1	2	0

resulting list are by construction consecutive elements of a complete list of strings generated from three 0s and three 1s, also ordered according to Handy's index equation (cf. table 2.1). This is the case due to the nature of equation 2.10.

Proof of Propagation Rule II (equation 3.51)

Table 3.2 shows a series of string pairs. They are constructed from a constant reduced string, $\boxed{01011}$ and a bit b_a whose position a is variable. The bit value of b_a is 1 for $\phi_{K_\sigma^{\mathrm{RASX}}}$ and 0 for $\phi_{L_\sigma^{\mathrm{RASX}}}$. As b_a propagates from the right to the left of the strings, the total string addresses change if the bit values in fixed bit positions in the strings change. In particular, if after the propagation of the variable bit to position a (from position $a-1$) we have $b_a = b_{a-1}$, the string and its address have not changed. If in contrast $b_a \neq b_{a-1}$, the string address has changed by a number equal to the number of binary permutations of all bits to the left of position a, represented by the binomial coefficient $\binom{M(\mathrm{RASX})-a}{N_\sigma^{\mathrm{left}}(a)}$, where $N_\sigma^{\mathrm{left}}(a)$ simply denotes the number of 1s to the left of position a. This is again due to the nature of Handy's index equation. If $b_a = 1$; $b_{a-1} = 0$ the string address rises, if $b_a = 0$; $b_{a-1} = 1$ the string address lowers, i.e.

$$\Delta a = 1: \qquad \Delta\mathrm{Addr}\{K_\sigma^{\mathrm{RASX}}\} = (b_a - b_{a-1})\binom{M(\mathrm{RASX})-a}{N_\sigma^{\mathrm{left}}(a)},$$

which is precisely the propagation rule given above. The extension to 2 or more variable bits is straightforward.

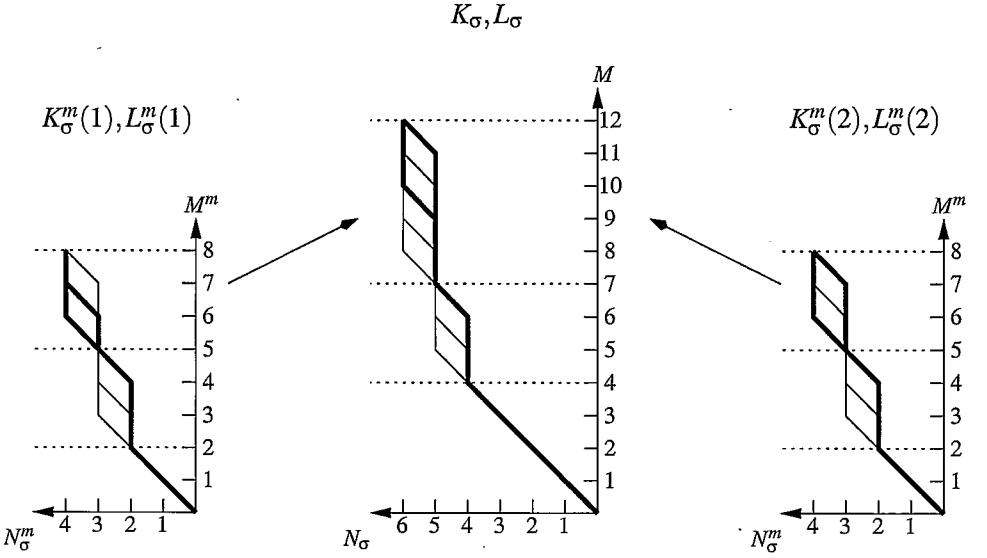


Figure 3.13.: Unwanted double contributions may arise when two model string pairs generated by different terms $\langle K_\sigma^m|\hat{a}_{w\sigma}^\dagger \hat{a}_{x\sigma}|L_\sigma^m\rangle$ result in the same contributing term $\langle K_\sigma|\hat{a}_{i\sigma}^\dagger \hat{a}_{j\sigma}|L_\sigma\rangle$, through application of the propagation rules. The example shown here illustrates the problem: the two model space excitations $\langle K_\sigma^m|\hat{a}_7^\dagger \hat{a}_6|L_\sigma^m\rangle$ and $\langle K_\sigma^m|\hat{a}_8^\dagger \hat{a}_6|L_\sigma^m\rangle$ both give (among others) the contribution $\langle K_\sigma|\hat{a}_{12}^\dagger \hat{a}_{10}|L_\sigma\rangle$.

3.4.2. Extension of the Model Subspaces

So far in our discussion of the RAS model space and its RASX^m subspaces we have taken the number of holes and electrons in the respective subspaces RAS1 and RAS3 to be not greater than 2. This was done merely for the sake of simplicity and we will now drop this restriction. However, care must be taken to ensure each term $\langle K_\sigma|\hat{a}_{i\sigma}^\dagger \hat{a}_{j\sigma}|L_\sigma\rangle$ is represented by only one model space term $\langle K_\sigma^m|\hat{a}_{w\sigma}^\dagger \hat{a}_{x\sigma}|L_\sigma^m\rangle$, in order to avoid multiple contributions. Figure 3.13 illustrates the problem. The same string pair, K_σ, L_σ, is arrived at by expanding the model space string pairs in the complete RAS space, using the appropriate propagation rules. The string pair, K_σ, L_σ, has been constructed using the same propagation rule, but starting from different model string pairs, $K_\sigma^m(1), L_\sigma^m(1)$ and $K_\sigma^m(2), L_\sigma^m(2)$. Each model string pair is related to a different excitation operator in the model space, namely $\hat{a}_{7\sigma}^\dagger \hat{a}_{6\sigma}$ and $\hat{a}_{8\sigma}^\dagger \hat{a}_{6\sigma}$, respectively.

This problem is, however, easily avoided by requiring the model space indices w, x, y, z to be flush-right in the RASX^m model subspaces. In graphical terms this means that loop-

opening and loop-closing segments will be concentrated on the lowest possible orbital level in the RASX^m model subspaces (cf. figure 3.13). It is then straightforward to see that the only restrictions on the values of MxHole and MxElec are

$$0 \leq \text{MxHole} \leq 2M(\text{RAS1}),$$

$$0 \leq \text{MxElec} \leq 2M(\text{RAS3}).$$

In the special case of both MxHole and MxElec assuming their maximum values the configuration spaces of CAS and RAS become identical.

3.4.3. The 2e Excitation Contribution

The 2e excitation term (equation 3.17) of the vector $\boldsymbol{\sigma}$ (equation 3.4) is by far the computationally more expensive one. It consists of two parts that differ conceptually. The $\alpha\alpha/\beta\beta$ parts (first two terms) each represent a double excitation involving electrons of one spin type only. All that is needed to accommodate for the double excitations in a single string space, using the model space approach, is an extension of the model subspace by up to two extra model space orbitals for each model subspace. The $\alpha\beta$-part and $\beta\alpha$-part (last two terms in equation 3.17) are simply combinations of single excitations in both the α-string space and the β-string space. Thus, for the purpose of finding all spin-string pairs, the $\alpha\beta$ part of the 2e excitation may be treated in an identical fashion to the 1e excitation and the discussion of the previous subsection applies.

As in the case of the 1e excitation, out-of-space excitations within a spin-string are entirely avoided by construction, through usage of the model space approach. The combination of two valid model string pairs for the $\alpha\beta$-part and $\beta\alpha$-part may, though, result in an out-of-space excitation. However, such combination may simply be rejected at the model space level, before any expansion of external subspaces takes place.

As we have seen in chapter 2, the symmetry properties of the two-electron integrals $(ij|kl)$ mean that it is possible to reduce the loops over i,j,k,l to unique integrals and summation in equation 2.18 over $i \geq j$, $k \geq l$ and $(ij) \geq (kl)$. By introducing the model space we replace summation over orbital indices i,j,k,l with model space orbital indices w,x,y,z. A single set of model space indices potentially represent a set of orbital indices. Nevertheless, the conditions given in equation 2.18 also hold for the RAS model space:

Table 3.3.: Possible combinations of excitations between RAS subspaces. (1→2 reads: excitation from RAS1 to RAS2)

$x \rightarrow w$	1→1	1→2		2→2			1→3			
$z \rightarrow y$	1→1	1→1	1→2	1→1	1→2	2→2	1→1	1→2	2→2	1→3

$x \rightarrow w$	2→3					3→3					
$z \rightarrow y$	1→1	1→2	2→2	1→3	2→3	1→1	1→2	2→2	1→3	2→3	3→3

Table 3.4.: Number of model space excitations $x \rightarrow w$ between RAS model spaces, provided MxHole ≥ 2 and MxElec ≥ 2; analytically and numerically for different RAS2 orbital subspace sizes. (1→2 reads: excitation from RAS1 to RAS2)

	1→1	1→2	2→2	1→3	2→3	3→3
M(RAS2)	2	M(RAS2)	$\frac{1}{2}(M(\text{RAS2})^2+M(\text{RAS2}))$	1	M(RAS2)	2
3	2	3	6	1	3	2
6	2	6	21	1	6	2
10	2	10	55	1	10	2

$$w \geq x, \quad y \geq z, \quad (wx) \geq (yz) \tag{3.54}$$

with

$$(wx) = \frac{w(w-1)}{2} + x, \qquad (yz) = \frac{y(y-1)}{2} + z.$$

Because of these restrictions we will only find a restricted number of excitation combinations between RAS subspaces. The possible combinations are shown in table 3.3. Assuming MxHole ≥ 2 and MxElec ≥ 2, how many excitations $x \rightarrow w$ are possible? There are only two possibilities for excitations of type $1 \rightarrow 1$, namely $w \rightarrow w$ and $x \rightarrow w$. Likewise there are only two realizations for excitations of type $3 \rightarrow 3$. Excitations $1 \rightarrow 3$ are all represented by one model space excitation, $x \rightarrow w$. We have M(RAS2) possible model space excitations $x \rightarrow w$ of type $1 \rightarrow 2$ (one for each RAS2 orbital), and also M(RAS2) excitations of type $2 \rightarrow 3$, again, one for each RAS2 orbital. Finally, there are $\binom{M(\text{RAS2})}{2}$ possibilities for excitations of type $2 \rightarrow 2$. Table 3.4 shows these numbers analytically and numerically for M(RAS2) =3, 6 and 10. Table 3.5 summarises the number of unique index quadruples, due to the restriction on possible combinations of (wx) with (yz) (cf. equation 3.54). These results show that

1. The contribution of $(w \rightarrow x,\ y \rightarrow z) = (2 \rightarrow 2,\ 2 \rightarrow 2)$ is the largest one and its relative importance grows with increasing size of the RAS2 subspace, M(RAS2).

Table 3.5.: Number of double excitations $x \rightarrow w$, $z \rightarrow y$ between RAS model orbital subspaces; analytically and numerically for different RAS2 orbital subspaces. (Constructed using tables 3.3 and 3.4 and equation 3.54)

		Number of combinations			
			$M(\mathrm{RAS2}) =$		
$x \rightarrow w$	$z \rightarrow y$	analytically	3	6	10
1→1	1→1	4	4	4	4
1→2	1→1	$2 \times M(\mathrm{RAS2})$	6	12	20
	1→2	$\frac{1}{2}\left(M(\mathrm{RAS2})^2 + M(\mathrm{RAS2})\right)$	6	21	55
	2→2	$\frac{1}{6}\left(M(\mathrm{RAS2})^3 - M(\mathrm{RAS2})\right)$	4	35	165
2→2	1→1	$M(\mathrm{RAS2})^2 + M(\mathrm{RAS2})$	12	42	110
	1→2	$\frac{1}{6}\left(2M(\mathrm{RAS2})^3 + 3M(\mathrm{RAS2})^2 + M(\mathrm{RAS2})\right)$	14	91	385
	2→2	$\frac{1}{8}\left(M^4 + 2M^3 + 3M^2 + 2M\right)$	21	231	1540
1→3	1→1	2	2	2	2
	1→2	$M(\mathrm{RAS2})$	3	6	10
	2→2	$\frac{1}{2}\left(M(\mathrm{RAS2})^2 + M(\mathrm{RAS2})\right)$	6	21	55
	1→3	1	1	1	1
2→3	1→1	$2 \times M(\mathrm{RAS2})$	6	12	20
	1→2	$M(\mathrm{RAS2})^2$	9	36	100
	2→2	$\frac{1}{2}\left(M(\mathrm{RAS2})^3 + M(\mathrm{RAS2})^2\right)$	18	126	550
	1→3	$M(\mathrm{RAS2})$	3	6	10
	2→3	$\frac{1}{2}\left(M(\mathrm{RAS2})^2 + M(\mathrm{RAS2})\right)$	6	21	55
3→3	1→1	2	4	4	4
	1→2	$2 \times M(\mathrm{RAS2})$	6	12	20
	2→2	$M(\mathrm{RAS2})^2 + M(\mathrm{RAS2})$	12	42	110
	1→3	2	2	2	2
	2→3	$2 \times M(\mathrm{RAS2})$	6	12	20
	3→3	4	4	4	4
	Σ		155	743	3242

Table 3.6.: General overview of the total numbers of distinctive w,x,y,z quadruples for MxHole = 2, MxElec = 2, analytically and numerically for a few RAS2 subspace sizes.

	Total dimension ...			
Number of indexes		M(RAS2) =		
w,x,y,z in *RAS2*	...analytically	3	6	10
4	$\frac{1}{8}\left(M^4+2M^3+3M^2+2M\right)$	21	231	1540
3	$M(\mathrm{RAS2})^3+M(\mathrm{RAS2})^2$	36	252	1100
2 (without $2\rightarrow 3$, $1\rightarrow 2$)	$\frac{7}{2}(M(\mathrm{RAS2})^2+M(\mathrm{RAS2}))$	42	147	385
2 ($2\rightarrow 3$, $1\rightarrow 2$ only)	$M(\mathrm{RAS2})^2$	9	36	100
1	$10M(\mathrm{RAS2})$	30	60	100
0	17	17	17	17
	Σ	155	743	3242

This means that in most cases the extension of RAS1 and RAS3 subspaces will simply result in a list of consecutive subspace addresses, and that both sign, $\mathrm{sgn}_{ijkl}^{K_\sigma}$, and integral, $(ij|kl)$, are constant for all contributions derived from a model space string pair.

2. All other relatively large contributions stem from excitations with at least one of the two excitations being a $2\rightarrow 2$. The advantage in this case is that the problem of finding relevant pairs $K_\alpha^{i_h,i_e}$, $L_\alpha^{i_h,i_e}$ is very similar to the 1e case.

In summary, the biggest part of the double excitations can be done as consecutive CSFs, with constant $(ij|kl)$ and $\mathrm{sgn}_{ijkl}^{K_\sigma}$. This part is getting relatively more important as the system size grows. The most relevant part after that consists logically of a 1e excitation type problem. This is particularly true for the $\alpha\alpha$ 2e part. In table 3.6 the dimension of the discussed cases is summarised analytically and for the same example cases already used above. Figure 3.14 illustrates this graphically. The two dominating contributions are becoming more important with growing size of the orbital space, and as a consequence, the efficiency of the propagation scheme becomes relatively greater for larger problems.

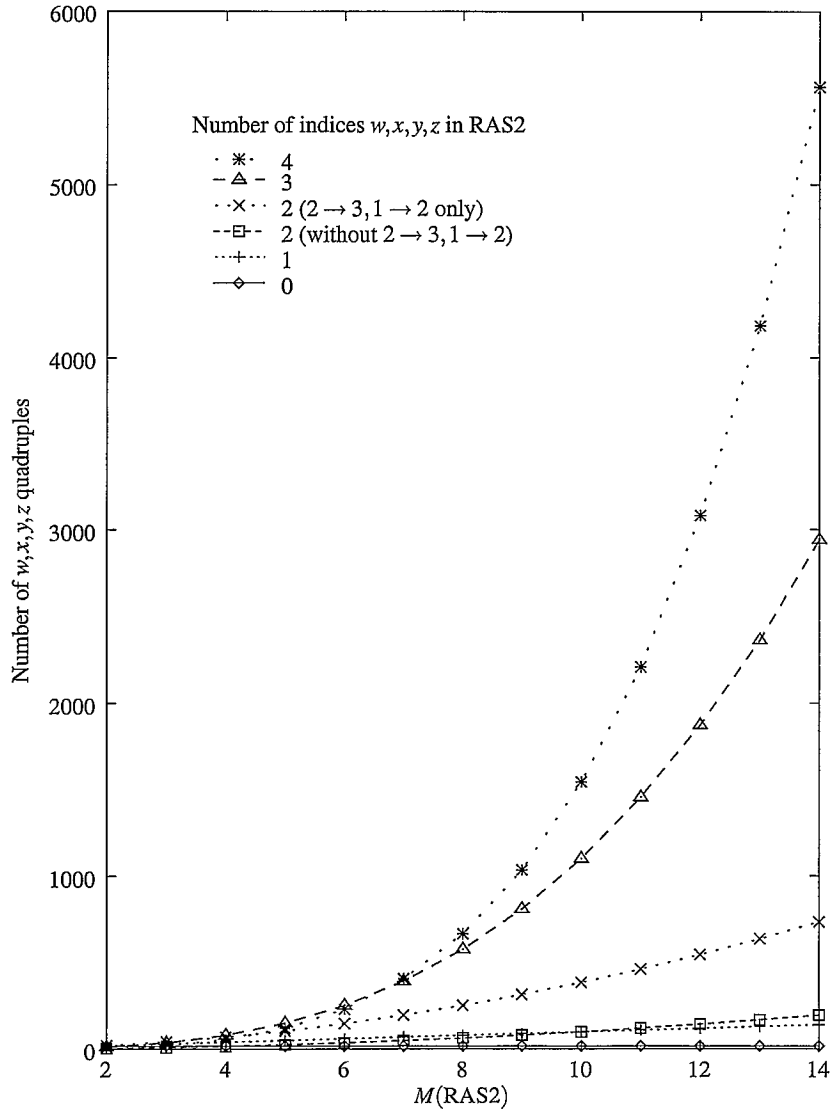


Figure 3.14.: Relative number of model space orbital indices w,x,y,z in the RAS2 orbital subspace, as a function of the RAS2 subspace size. MxHole and MxElec are both assumed to equal 2 (cf. table 3.6).

3.5. The Effect of Propagation Rules on the Order of Integrals and the Computation of Signs

In an actual implementation of the propagation rule algorithm, once a model string pair is found for both the α-spin and β-spin model string space, the RAS1 and RAS3 subspace strings need to be expanded. This will be done in a nested loop over all RAS1 and RAS3 expansions. If all model space indices w,x (w,x,y,z) lie within the RAS2 subspace, then all contributions will be for the same integral $(i|j)$ ($(ij|kl)$). If, on the other hand, some or all model space indices fall into one or both of the other subspaces, the contributions will be for a list of integrals. It is therefore useful to assemble the corresponding list of integrals, for every set of model space indices, in the order they will be accessed. This order is entirely given by the propagation rules, and only one list needs to be assembled for any set of model space indices. The integral lists for αα-, αβ-, βα- and ββ-contributions will be identical and need to be assembled only once.

It is by now clear that the value associated with a bra-ket term of the form $\langle K_\sigma|\hat{a}^\dagger_{i\sigma}\hat{a}_{j\sigma}|L_\sigma\rangle$ may differ from its model space analogue $\mathrm{sgn}_0^{m,\sigma} = \langle K^m_\sigma|\hat{a}^\dagger_{w\sigma}\hat{a}_{x\sigma}|L^m_\sigma\rangle$ at most by a factor of -1. If, for example, the index i lies within the RAS3 orbital subspace, then the propagation of the appropriate bit towards the left will result in a sign change every time the bit value of b_{i-1} equals 1. Because of this behaviour and the direct association of the sign change with the propagation of variable bits, it is convenient to pre-compute the sign-changing factors for each spin subspace together with the RAS1 and RAS3 subspace string address pairs. The term $\mathrm{sgn}_{ij}^{K_\sigma} = \langle K_\sigma|\hat{a}^\dagger_{i\sigma}\hat{a}_{j\sigma}|L_\sigma\rangle$ is then computed as

$$\mathrm{sgn}_{ij}^{K_\alpha} = \mathrm{sgn}_0^{m,\alpha} \times \mathrm{sgn}_{ij}^{K^{\mathrm{RAS1}}_\alpha} \times \mathrm{sgn}_{ij}^{K^{\mathrm{RAS3}}_\alpha}. \tag{3.55}$$

The same applies without change to the 2e equivalent, $\mathrm{sgn}_{ij}^{K_\sigma} = \langle K_\sigma|\hat{a}^\dagger_{i\sigma}\hat{a}^\dagger_{k\sigma}\hat{a}_{l\sigma}\hat{a}_{j\sigma}|L_\sigma\rangle$

$$\mathrm{sgn}_{ijkl}^{K_\alpha} = \mathrm{sgn}_0^{m,\alpha} \times \mathrm{sgn}_{ijkl}^{K^{\mathrm{RAS1}}_\alpha} \times \mathrm{sgn}_{ijkl}^{K^{\mathrm{RAS3}}_\alpha}. \tag{3.56}$$

3.6. The Direct RAS Algorithm

The algorithm that follows from the use of an integral driven CI is inherently parallel, and the parallel loop may include any combination of orbital indices i,j,k,l. We will now provide the algorithms for one- and two-electron excitation term contributions of equation

3.13. All reduced lists of model space and RAS subspaces and the associated Handy's index matrices are precomputed only once. The outer loop is over the model space orbital indices w,x,y,z. The computation of model space string pairs is analogous to string pair computation as described in chapter 2 for full CI string pairs. Once the model string pair K_σ, L_σ is known, its category and propagation rules for both the RAS1 and RAS3 subspaces are identified. This information is used to assemble the list of integrals into a vector, where they will be accessed in a sequential manner during updating of the $\boldsymbol{\sigma}$-vector. Then all excitation lists are assembled, by application of the propagation rules to the RAS subspace model strings, RAS1^m and RAS3^m. We denote these subspace excitation lists $\mathcal{E}_\sigma^{\mathrm{RAS}X}$. Finally, nested loops over subspace excitation list entries lead to the local address, and by combining α- and β-strings, to the CSF address. The algorithms 3.1, 3.2 and 3.3 show the appropriate algorithms for 1e, 2e $\alpha\alpha$ and 2e $\alpha\beta$ parts, respectively, of the $\boldsymbol{\sigma}$-vector. More detailed descriptions of our actual implementation can be found in appendix C.

The dimension of the integral list is a function of the number of orbital indices in RAS1 and RAS3, $n1$ and $n3$, respectively

$$D(n1,n3) = \binom{M(\mathrm{RAS1})}{n1} \times \binom{M(\mathrm{RAS3})}{n3} \quad \left| \begin{array}{l} 0 \le n1 \le 4 \\ 0 \le n3 \le 4-n1 \end{array} \right. . \tag{3.57}$$

If all orbital indices lie within RAS2, then the integral list contains only one element. The maximum length of the integral vector depends on the relative size of the RAS1 and RAS3 orbital subspaces. In the special case of $M(\mathrm{RAS1}) = M(\mathrm{RAS3}) > 2$ the maximum dimension is for two indices each in the RAS1 and RAS3 subspaces,

$$\begin{aligned} D(2,2) &= \binom{M(\mathrm{RAS1})}{2}^2 \qquad\qquad |M(\mathrm{RAS1}) = M(\mathrm{RAS3}) > 2 \\ &= \left(\frac{M(\mathrm{RAS1})^2 - M(\mathrm{RAS1})}{2} \right)^2 . \end{aligned}$$

For example, with $M(\mathrm{RAS1}) = M(\mathrm{RAS3}) = 10$ the maximum length of the integral vector is 2025. At any stage the length of the integral vector will be small when compared to the CI vector. The memory requirements for the reduced lists of RAS subspace strings and model space strings are equally small compared to the CI vector, as are the partial excitation strings, $\mathcal{E}_\sigma^{\mathrm{RAS}X}$.

Thus the memory requirements are dominated by the vectors $\boldsymbol{\sigma}$ and $\mathbf{C}$. However, if the

```
(parallel) loop over w,x
 loop over reduced string L^{M^m-2}_{N^m_α-1}
  insert bits b_w,b_x into red.model string
  (→ K^m_α,L^m_α)
  if K^m_α,L^m_α are invalid, jump to loop end
  (→ sgn^m_α, Cat(i_h,i_e), Addr{K^RAS2_α}, Cat(i'_h,i'_e),
  Addr{L^RAS2_α},P(RAS1), P(RAS3))
  assemble integral list {(i|j)}

  apply P(RAS1) to assemble E^RAS1_α

  apply P(RAS3) to assemble E^RAS3_α
  loop over all E^RAS1_α
   loop over all E^RAS3_α
    (→ Addr{K_α}, Addr{L_α}, (i|j),
    sgn_α = sgn^m_α × sgn^RAS1_α × sgn^RAS3_α)

    loop over all allowed Cat(i''_h,i''_e)

     loop over all K_β(Cat(i''_h,i''_e)
      σ(K_α,K_β) := σ(K_α,K_β) + sgn_α(i|j)C(L_α,K_β)
      σ(L_α,K_β) := σ(L_α,K_β) + sgn_α(i|j)C(K_α,K_β)
     end loop (K_β(Cat(i''_h,i''_e))
    end loop (Cat(i''_h,i''_e))
   end loop (E^RAS3_α)
  end loop (E^RAS1_α)
 end loop (L^{M^m-2}_{N^m_α-1})
end (parallel) loop (w,x)
```

(The reduced model string lists have been precomputed at program initialisation)

(This is to avoid double contributions; cf. section 3.4.2)

(This is done by applying the propagation rules, $\mathcal{P}$(RAS1) and $\mathcal{P}$(RAS3)) ($\mathcal{E}^{\text{RAS1}}_{\alpha}$ is the RAS1 substring excitation list, containing Addr$\{K^{\text{RAS1}}_{\alpha}\}$, Addr$\{L^{\text{RAS1}}_{\alpha}\}$ and $\text{sgn}^{\text{RAS1}}_{\alpha}$)

(Allowed Cat(i''_h,i''_e) are those with (Max(i_h,i'_h) + i''_h) $\leq$ MxHole and (Max$(i_e,i'_e)+i''_e) \leq$ MxElec)

(Actual contribution)

Algorithm 3.1: RAS: The 1e excitation (α part) contribution to the σ-vector.

(parallel) loop over w,x,y,z

loop over reduced string list $\mathcal{L}^{M^m-4}_{N^m_\alpha-2}$ (The reduced model string lists have been precomputed at program initialisation)

insert bits b_w,b_x,b_y,b_z into red. model string
$(\rightarrow K^m_\alpha, L^m_\alpha)$

if K^m_α, L^m_α are invalid, jump to loop end (This is to avoid double contributions; cf. section 3.4.2)

$(\rightarrow \mathrm{sgn}^m_\alpha, \mathrm{Cat}(i_h,i_e), \mathrm{Addr}\{K^{\mathrm{RAS2}}_\alpha\}, \mathrm{Cat}(i'_h,i'_e), \mathrm{Addr}\{L^{\mathrm{RAS2}}_\alpha\}, \mathcal{P}(\mathrm{RAS1}), \mathcal{P}(\mathrm{RAS3}))$

assemble integral list $\{(ij|kl)\}$ (This is done by applying the propagation rules, $\mathcal{P}(\mathrm{RAS1})$ and $\mathcal{P}(\mathrm{RAS3})$)

apply $\mathcal{P}(\mathrm{RAS1})$ to assemble $\mathcal{E}^{\mathrm{RAS1}}_\alpha$ ($\mathcal{E}^{\mathrm{RAS1}}_\alpha$ is the RAS1 substring excitation list, containing $\mathrm{Addr}\{K^{\mathrm{RAS1}}_\alpha\}$, $\mathrm{Addr}\{L^{\mathrm{RAS1}}_\alpha\}$ and $\mathrm{sgn}^{\mathrm{RAS1}}_\alpha$)

apply $\mathcal{P}(\mathrm{RAS3})$ to assemble $\mathcal{E}^{\mathrm{RAS3}}_\alpha$

loop over all $\mathcal{E}^{\mathrm{RAS1}}_\alpha$

loop over all $\mathcal{E}^{\mathrm{RAS3}}_\alpha$

$(\rightarrow \mathrm{Addr}\{K_\alpha\}, \mathrm{Addr}\{L_\alpha\}, (ij|kl), \mathrm{sgn}_\alpha = \mathrm{sgn}^m_\alpha \times \mathrm{sgn}^{\mathrm{RAS1}}_\alpha \times \mathrm{sgn}^{\mathrm{RAS3}}_\alpha)$

loop over all allowed $\mathrm{Cat}(i''_h,i''_e)$ (Allowed $\mathrm{Cat}(i''_h,i''_e)$ are those with $(\mathrm{Max}(i_h,i'_h) + i''_h) \leq \mathrm{MxHole}$ and $(\mathrm{Max}(i_e,i'_e) + i''_e) \leq \mathrm{MxElec}$)

loop over all $K_\beta(\mathrm{Cat}(i''_h,i''_e)$

$\sigma(K_\alpha,K_\beta) := \sigma(K_\alpha,K_\beta) + \frac{1}{2}\mathrm{sgn}_\alpha(ij|kl)C(L_\alpha,K_\beta)$ (Actual contribution)

$\sigma(L_\alpha,K_\beta) := \sigma(L_\alpha,K_\beta) + \frac{1}{2}\mathrm{sgn}_\alpha(ij|kl)C(K_\alpha,K_\beta)$

end loop $(K_\beta(\mathrm{Cat}(i''_h,i''_e))$

end loop $(\mathrm{Cat}(i''_h,i''_e))$

end loop $(\mathcal{E}^{\mathrm{RAS3}}_\alpha)$

end loop $(\mathcal{E}^{\mathrm{RAS1}}_\alpha)$

end loop $(\mathcal{L}^{M^m-4}_{N^m_\alpha-2})$

end (parallel) loop (w,x,y,z)

Algorithm 3.2: RAS: The 2e excitation ($\alpha\alpha$ part) contribution to the $\boldsymbol{\sigma}$-vector.

(parallel) loop over w,x,y,z — (Loop is the same as for $\alpha\alpha$ part, i.e. the integral list is already known)

 loop over reduced model string list $\mathcal{L}^{M^m-2}_{N^m_\alpha-1}$
  insert bits b_w,b_x into red.model string
  $(\rightarrow K^m_\alpha, L^m_\alpha)$
  if K^m_α, L^m_α are invalid, jump to loop end — (This is to avoid double contributions; cf. section 3.4.2)
  $(\rightarrow \mathrm{sgn}^m_\alpha, \mathrm{Cat}(i_h,i_e), \mathrm{Addr}\{K^{\mathrm{RAS2}}_\alpha\}, \mathrm{Cat}(i'_h,i'_e), \mathrm{Addr}\{L^{\mathrm{RAS2}}_\alpha\}, \mathcal{P}(\mathrm{RAS1}), \mathcal{P}(\mathrm{RAS3}))$
  apply $\mathcal{P}(\mathrm{RAS1})$ to assemble $\mathcal{E}^{\mathrm{RAS1}}_\alpha$
  apply $\mathcal{P}(\mathrm{RAS3})$ to assemble $\mathcal{E}^{\mathrm{RAS3}}_\alpha$
  loop over reduced string list $\mathcal{L}^{M^m-2}_{N^m_\beta-1}$
   insert bits b_y,b_z into red.model string
   $(\rightarrow K^m_\beta, L^m_\beta)$
   if K^m_β, L^m_β are invalid, jump to loop end — (This is to avoid double contributions; cf. section 3.4.2)
   $(\rightarrow \mathrm{sgn}^m_\beta, \mathrm{Cat}(i''_h,i''_e), \mathrm{Addr}\{K^{\mathrm{RAS2}}_\beta\}, \mathrm{Cat}(i'''_h,i'''_e), \mathrm{Addr}\{L^{\mathrm{RAS2}}_\beta\}, \mathcal{P}(\mathrm{RAS1}), \mathcal{P}(\mathrm{RAS3}))$
   if $[(i_h+i''_h) \le \mathrm{MxHole}$ and $(i'_h+i'''_h) \le \mathrm{MxHole}$ and $(i_e+i''_e) \le \mathrm{MxElec}$ and $(i'_e+i'''_e) \le \mathrm{MxElec}]$ then — (Check for allowed graph combinations)
    apply $\mathcal{P}(\mathrm{RAS1})$ to assemble $\mathcal{E}^{\mathrm{RAS1}}_\beta$
    apply $\mathcal{P}(\mathrm{RAS3})$ to assemble $\mathcal{E}^{\mathrm{RAS3}}_\beta$
    loop over all $\mathcal{E}^{\mathrm{RAS1}}_\alpha$
     loop over all $\mathcal{E}^{\mathrm{RAS3}}_\alpha$
      $(\rightarrow \mathrm{Addr}\{K_\alpha\}, \mathrm{Addr}\{L_\alpha\}, \mathrm{sgn}_\alpha = \mathrm{sgn}^m_\alpha \times \mathrm{sgn}^{\mathrm{RAS1}}_\alpha \times \mathrm{sgn}^{\mathrm{RAS3}}_\alpha)$
      loop over all $\mathcal{E}^{\mathrm{RAS1}}_\beta$
       loop over all $\mathcal{E}^{\mathrm{RAS3}}_\beta$
        $(\rightarrow \mathrm{Addr}\{K_\beta\}, \mathrm{Addr}\{L_\beta\}, (ij|kl), \mathrm{sgn}_\beta = \mathrm{sgn}^m_\beta \times \mathrm{sgn}^{\mathrm{RAS1}}_\beta \times \mathrm{sgn}^{\mathrm{RAS3}}_\beta)$ — (The integral list is known from $\alpha\alpha$-algorithm)
        $\sigma(K_\alpha,K_\beta) := \sigma(K_\alpha,K_\beta) + \frac{1}{2}\mathrm{sgn}_\alpha\mathrm{sgn}_\beta(ij|kl)C(L_\alpha,L_\beta)$ (Actual contribution)
        $\sigma(L_\alpha,L_\beta) := \sigma(L_\alpha,L_\beta) + \frac{1}{2}\mathrm{sgn}_\alpha\mathrm{sgn}_\beta(ij|kl)C(K_\alpha,K_\beta)$
       end loop $(\mathcal{E}^{\mathrm{RAS3}}_\beta)$
      end loop $(\mathcal{E}^{\mathrm{RAS1}}_\beta)$
     end loop $(\mathcal{E}^{\mathrm{RAS1}}_\alpha)$
    end loop $(\mathcal{E}^{\mathrm{RAS1}}_\alpha)$
   end if (valid graph combinations)
  end loop $(\mathcal{L}^{M^m-2}_{N^m_\beta-1})$
 end loop $(\mathcal{L}^{M^m-2}_{N^m_\alpha-1})$
end (parallel) loop (w,x,y,z)

Algorithm 3.3: RAS: The 2e excitation ($\alpha\beta$ part) contribution to the $\boldsymbol{\sigma}$-vector.

working memory is not large enough to accommodate the entire CI vectors, it is straightforward to adapt the algorithm to use only certain blocks of these vectors, by taking advantage of the blocking of string addresses of all strings that belong to a certain graph. This means that the vectors $\boldsymbol{\sigma}$ and $\mathbf{C}$ with elements $\sigma(K_\alpha, K_\beta)$ and $C(K_\alpha, K_\beta)$ are also blocked with respect to graph combinations (categories). Since the graph combination is fixed for all contributions of a found model string pair, it is possible to adapt the described scheme by either reading in the required block of the CI vectors from distributed memory, or by restricting contributions to a given CI vector block at a time.

Particularly in the first iterations of a typical MCSCF computation, there are only very few nonzero elements $C(K_\alpha, K_\beta)$. This means that in principle, only those contributions

$$\begin{aligned} \sigma(L_\alpha, L_\beta) &:= \sigma(L_\alpha, L_\beta) + (i|j)A_{ij}^{KL} C(K_\alpha, K_\beta) \\ \sigma(L_\alpha, L_\beta) &:= \sigma(L_\alpha, L_\beta) + (ij|kl)B_{ijkl}^{KL} C(K_\alpha, K_\beta) \end{aligned} \tag{3.58}$$

with $C(K_\alpha, K_\beta) \neq 0$ need to be considered to compute $\boldsymbol{\sigma}$. This is an inherent weakness of the integral driven approach, since the resulting algorithm does not provide for efficient exclusion of blocks of the CI vector. However, to increase efficiency in the model space approach proposed here, one could assemble a short list of flags that indicate whether *any* of the α-strings obtained by expanding a α-model string belongs to a block of $\mathbf{C}$ with at least one nonzero element An analogous list would be needed for β-string blocks. The most simple realization of this idea would be to define the blocks as having contributions of a unique α-graph/β-graph combination. More elaborate methods could include, for example, lists for specific RAS2 substrings, possibly combined with the appropriate propagation rules.

3.7. Parallelisation

We will now describe the parallel implementation of the direct RASSCF algorithm. We assume a scalable parallel distributed memory computer architecture consisting of nodes, each with local memory. The nodes themselves may be symmetric multi-processor (SMP) machines with shared memory. As mentioned in the previous section, the integral driven algorithm is already inherently parallel. The main issue to address is thus how to ensure good load balancing. A flexible load balancing mechanism helps to achieve optimum scaling and is essential in heterogenous environments, where a cluster may consist of

Table 3.7.: The number of independent parallel tasks, defined as unique model space index quadruples, w,x,y,z, for various RAS2 orbital subspaces (MxHole = MxElec = 2).

	Number of parallel tasks	
M(RAS2)	index pairs (w,x)	index quadruples (w,x,y,z)
4	23	410
6	38	947
8	57	1947
10	80	3638

nodes of different architecture. It is also important in the realistic situation of uneven pre-loading of allocated nodes, i.e. in situations where some workers must share resources with concurrently running programs on the same node.

Load balancing can be achieved when the total number of independent, parallel tasks is large compared to the number of processing elements (PEs, also: CPUs). The overhead of subdividing the total work into parallel sub-tasks should ideally be kept small compared to total execution time. Communication between workers should be avoided and should be handled by a separate process, in order to avoid synchronisation delays [53].

The sub-tasks that are executed on parallel nodes correspond to model space index pairs w,x (1e contribution), or model space index quadruples, w,x,y,z (2e contribution). The relatively small size of the model space (as opposed to the orbital space comprising all active orbitals) means the total number of independent tasks is much reduced. Table 3.7 shows for different sizes of the RAS2 subspace and MxHole = MxElec = 2 the number of parallel tasks for both the 1e and 2e contributions. The number of tasks is independent of the size of the RAS1 and RAS3 subspaces, and lies within an useful range, i.e. there are sufficiently many sub-tasks to enable load balancing for a large number of processing elements, even for the smallest RAS2 orbital subspace. For larger RAS2 subspaces, in order to reduce overhead due to communication, several subtasks may be grouped together in order to adapt granularity of the parallelism. This may be done dynamically, in order to adapt to the number of requested or allocated CPUs.

The IDA approach followed by us in this chapter involves the outer loops over the model space orbital indices w,x,y,z. Inside these nested loops, the assembly of certain blocks of the vector $\boldsymbol{\sigma}$ takes place. However, with the exception of the CI vector $\boldsymbol{\sigma}$ itself, all data inside the loops are static. Therefore, we can minimise communication overhead by pass-

ing these static data to all worker nodes only once at the beginning of each eigenvector iteration. This is of advantage especially for distributed memory architectures, and in particular for low-cost configurations with standard network interprocessor communication. On shared memory architectures (e.g. an SMP node) these static data need to exist only once.

We now discuss the implementation of the general strategy just discussed for distributed memory architectures using Linda [53]. Figure 3.15 shows a flowchart outlining our strategy for parallel execution of the program. We begin with a few definitions. We assume that each node may be an SMP. The number of processors on each SMP is indicated as $NProcS$ (on single processor nodes $NProcS = 1$). The value $NTask$ is computed, with $NTask = NProcS \times H$ and H is chosen to both optimise load-balancing and minimise communication overhead. The main process is called the master process (master). The process spawned on the first processing node retrieves the index $(wxyz)$ (initially $(wxyz) = \frac{(wx)_{\max}((wx)_{\max}+1)}{2}$ with $(wx)_{\max} = \frac{M^m(M^m+1)}{2}$). A list of $NTask$ indices is assembled on the worker, by testing the validity of $(wxyz)$, storing it if valid, testing $(wxyz)-1$, and so forth. Invalid indices are simply dropped when found. The worker then puts a new index $(wxyz)' = (wxyz) - NTask - N_{\text{invalid}}$ into tuple space, which is then retrieved by the next worker etc. If $NProcS > 1$, then $NProcS - 1$ shared memory processes are created, each of which will be assigned a subset of $\frac{NTask}{NProcS}$ tasks. Thus the original loop over model orbital space indices, w,x,y,z, is parallelised. Each processor then carries out the assembly of $\boldsymbol{\sigma}$ using the original serial algorithm (cf. algorithms 3.1–3.3 and appendix C). On finishing, the next available index $(wxyz)$ is retrieved from tuple space and this procedure is continued until all model space generator combinations are processed, i.e. the index $(wxyz) = 0$ is found. Finally the intermediate results are passed through tuple space to be combined to give the final result of the computation.

SMP architectures are supported in three ways, i) using Linda, ii) using shared memory, or iii) a combination of both. In the first case, i), one Linda worker is created on each PE. The communication overhead in this scheme is extremely small. Furthermore, using Linda even on SMPs takes full advantage of the load balancing mechanism described above. However, the static data (e.g. the CI vector $\mathbf{C}$, integrals etc.) must be replicated for each Linda worker.

The alternative ii) is usage of shared memory ($NProcS > 1$). In this case all static data get replicated only once per SMP and are shared by all PEs. All result vectors (one per PE) are summed before the result of this worker is passed back to tuple space, where it

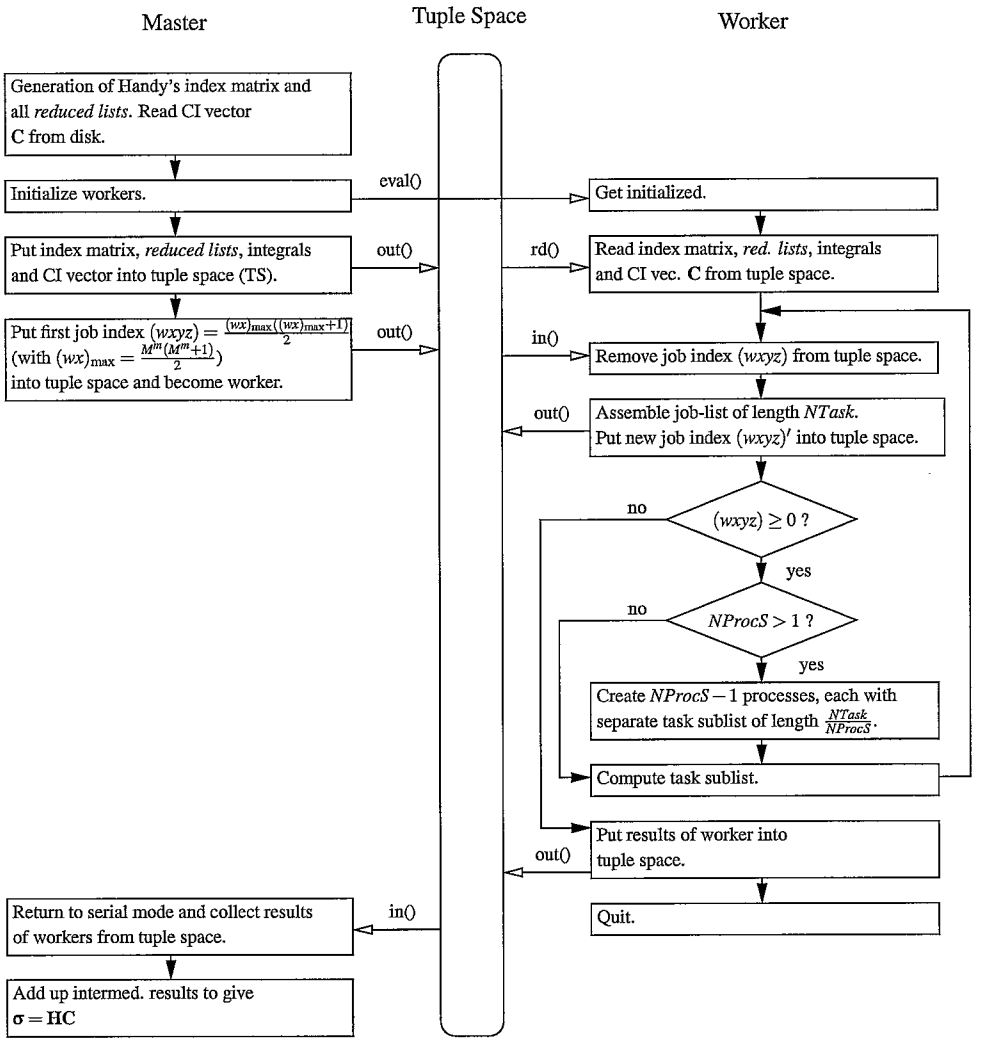


Figure 3.15.: General structure of the parallel RAS code as described in the text.

is subsequently retrieved by the master process. This method has the advantage that all static data exist only once in working memory on a SMP node. For example, on a SMP with $NProcS = 2$ the memory requirements are for three CI vectors, i.e. one result vector for each PE plus one shared CI vector (**C**). This is particularly useful if available working memory is limited and/or if the size of the CI vector is very large. A further advantage of this option is a reduction of communication by a factor of *NProcS*, since all static data and also the combined results have to be passed only once per *NProcS* workers. The price to be paid in this case is that there is no load balancing between PEs on a SMP. The tasks defined by the *NProcS* indices are of very similar length, but will be slightly different. In practise this means that efficiency will decrease if *NProcS* becomes very large.

The third option is any combination of the previous two. In this case the number of PEs using shared memory, *NProcS*, may be chosen to comprise any available number of PEs on each node, i.e. on a 4way SMP node *NProcS* can assume the values 1,2,3 or 4. Case iii) would then correspond to two Linda workers running on that node, each of them comprising $NProcS = 2$ shared memory processes. Taking advantage of this option the mode of parallelism may be tailored to the available architecture.

The approach described above ensures great flexibility in choosing the number of processors and leads, due to the relatively high number of tasks, to an effective load balancing, provided the number of PEs is small compared to the number of tasks. The intermediate assembly of a sublist of *NTask* tasks ensures the optimum granularity, through dynamical adaptation of *NTask*. The load balancing scheme implemented also allows for the fact that, in many environments, the CPU-time on different nodes of a parallel machine may be shared among a number of running programs and thus automatically uses the resources as they become available.

3.8. Example Applications

The RASSCF program developed and implemented as part of this work, has so far been applied in a small number of projects, including the calculation of the first two excited states of indole, carried out in collaboration with Luis Blancafort, and the excitation energies of the first two excited states of hexatriene, in collaboration with Martial Boggio. In the following two subsections we will give an overview of the motivation to use RAS in each of the two cases, and a summary of the results obtained.

The RASSCF method can be thought of as an extension to the CASSCF method. It

creates new possibilities in the context of the MCSCF method, along with new challenges and it may be applied in number of ways. Firstly, it may be used as an approximation to the CASSCF method, where the size of the active space means that CASSCF itself is not feasible. Obviously care must be taken in this case to ensure that the occupancy restrictions imposed on the RAS1 and RAS3 subspaces are "reasonable". Secondly, the RASSCF method may be used to describe electronic states, where dynamical correlation needs to be included in the iterative search for the MOs. This is essential, for example, if geometry optimisations are to be carried out. Thirdly, RASSCF may provide a reference space to which subsequent methods, such as perturbation theory, may be applied in order to recover dynamical correlation energy corrections non-iteratively. Part of the associated challenge is certainly how to make best use of this new tool, and relates partly to technical, and partly to theoretical issues.

In the next two subsections, we will denote by RAS($N, M_I + M_{II} + M_{III}$)[MxHole, MxElec] a RASSCF calculation with N electrons in an active space of M_I RAS1 orbitals, M_{II} RAS2 orbitals and M_{III} RAS3 orbitals. MxHole and MxElec signify the maximum number of particles excited out of the RAS1 space, and into the RAS3 orbital space, respectively.

3.8.1. Vertical Excitations and Potential Energy Surface of Indole

Indole is the chromophore of the aminoacid tryptophane. Its fluorescence is of particular interest, because of its use in protein structural studies as a *fluorescent probe*. Indole is a convenient fluorescent probe, because its fluorescence is highly dependent on the environment (i.e. the structure of the surrounding protein). However, the fluorescence of indole is not completely understood, which often makes interpretation of protein fluorescence spectra difficult.

Indole has two low singlet excited states, ^1L_b (covalent) and ^1L_a (ionic), which are separated by a small energy gap of approximately 0.4 eV. A CASSCF(10,9)/6-31G* calculation overestimates this gap (1.20 eV), mainly because the energy of the ionic ^1L_a state

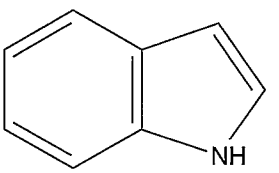


Figure 3.16.: Indole

Table 3.8.: CASSCF and CAS-PT2 [66] results for indole. For source of experimental results see references in [66].

		CASSCF(10,9)/6-31G*		CAS-PT2/ANO	Experiment
Geom.	State	E [hartree]	E_{rel} [eV]	E_{rel} [eV]	E_{rel} [eV]
FC	S2 (^1L_a)	−361.3371	6.20 (vert.)	4.73 (vert.)	4.77
FC	S1 (^1L_b)	−361.3810	5.00 (vert.)	4.43 (vert.)	4.37
FC	S0	−361.5648	0.00	0.00	0.00
^1L_b min.	S1	−361.3940	4.65 (0–0)	4.35 (0–0)	4.37
^1L_a min.	S1	-	-	4.66 (0–0)	4.54
	S2	−361.3506	5.83 (0–0)	-	-

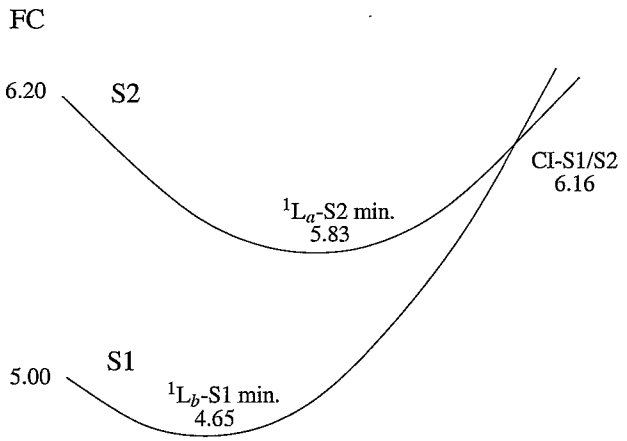


Figure 3.17.: Potential energy surfaces (S1,S2) of indole, calculated at the CASSCF level (CAS(10,9)/6-31G*).

is overestimated. CAS-PT2/ANO calculations by Serrano and Roos [66] reproduce the experimental value. The goal of this study is to give a balanced description of the excited states that includes a good approximation to the energy gap of 0.4 eV between the S1 and S2 excited states.

In table 3.8 and figure 3.17 we illustrate the results of the CASSCF computation. Comparison with experimental results in table 3.8 clearly shows that the results of the CASSCF computation are not only quantitatively, but also qualitatively incorrect. According to the experimental data the two states of indole (^1L_a and ^1L_b) have similar 0–0 transitions (emissions from the minimum for the state to the ground-state minimum). Therefore, the potential energy surface has two different minima on the surface of S1, one for each state. Presumably there is a conical intersection (surface crossing) between S1 and S2, because

Table 3.9.: Results of preliminary RAS computations of the first two singlet excited states of indole. The 3-21G basis set was used for these computations.

N	M_I	M_{II}	M_{III}	MxHole	MxElec	ΔE_{S1-S2}
10	3	5	10	1	1	0.90 eV
10	3	5	10	2	2	1.09 eV
30	13	5	20	1	1	0.78 eV

the ^1L_a state is S2 at the Franck-Condon geometry. At the CASSCF(10,9)/6-31G* level, the surface is wrong: the ^1L_a minimum lies on S2.

A common technique used in order to enhance qualitatively the description of electronic states is to "double" the active space of π-orbitals in a CASSCF computation. The corresponding CASSCF(10,18) in the case of indole is, however, outside the scope of practical computation. The RASSCF treatment of this enlarged active space on the other hand is absolutely practical. A second possibility for qualitatively improving the wavefunction is to include σ-orbitals into the set of active orbitals.

Preliminary computations using the 3-21G basis set were performed in order to identify the most suitable RAS active space definition. In a first test the active space was doubled by including the 9 2p orbitals of A'' symmetry (π-system used in the CASSCF computation) and the 9 3p orbitals of A'' symmetry. The additional 10 orbitals were all placed in the RAS3 subspace, and the 3 nearly doubly occupied orbitals from the CASSCF computation in the RAS1 subspace. This corresponds to a CAS(4,5) reference space in RAS2. Two computations were performed, one with MxHole = MxElec = 2 and one with MxHole = MxElec = 1. In a second test the active orbital space from the first test was augmented by 10 σ-orbitals and σ^*-orbitals corresponding to the 10 CC and CN σ-bonds. The RAS1 subspace now contained 13 orbitals, RAS2 4 orbitals and RAS3 20 orbitals, and excitations out of RAS1 and into RAS3 were limited to 1 particle each. Again the RAS2 subspace translates into a CAS(4,5) reference space. In table 3.9 we summarise the computations and their results. It is clear from these preliminary results that inclusion of σ-orbitals and single excitations relative to the reference space are particularly important.

The strategy from now on was a) to increase the basis to 6-31G*, b) to reduce the RAS space by eliminating the MOs with occupancies closest to 2.00 and 0.00, and vary the RAS2 subspace, and c) to improve the energies with the 6-311+G* basis set. Table 3.10 shows the orbital occupancies resulting from step a). A number of orbitals that are virtually doubly occupied or unoccupied are excluded from the active space and the orbital

Table 3.10.: Orbital occupancies for RAS(30,13+5+20)[1,1] and RAS(20,8+5+11)[1,1] computations at the Franck-Condon geometry (6-31G* basis set). Orbitals that have been excluded in the latter RAS computation are shown in italics.

	Sym.	RAS(30,13+5+20)[1,1]	Sym.	RAS(20,8+5+11)[1,1]
RAS1	$10A'$	*1.9986, 1.9984, 1.9983, 1.9979, 1.9978* 1.9960, 1.9957, 1.9952, 1.9949, 1.9932	$5A'$	1.9979, 1.9960, 1.9958, 1.9955, 1.9951
	$3A''$	1.9866, 1.9842, 1.9731	$3A''$	1.9847, 1.9808, 1.9716
RAS2	$5A''$	1.8567, 1.0755, 0.9368, 0.1057, 0.0509	$5A''$	1.8377, 1.0966, 0.9312, 0.1136, 0.0535
RAS3	$1A''$	0.0210	$1A''$	0.0233
	$10A'$	0.0062, 0.0039, 0.0038, 0.0037, 0.0036 *0.0036, 0.0033, 0.0024, 0.0019, 0.0016*	$5A'$	0.0059, 0.0038, 0.0038, 0.0031, 0.0031
	$9A''$	0.0025, 0.0013, 0.0011, 0.0011, 0.0010 *0.0010, 0.0009, 0.0009, 0.0001*	$5A''$	0.0028, 0.0013, 0.0013, 0.0013, 0.0011

Table 3.11.: Results of RASSCF computations of indole, at FC point. The experimental values are: 4.77 eV (S1), 4.27 eV (S2), 0.40 eV (ΔE_{S2-S1}).

Basis set	$(N, M_I + M_{II} + M_{III})$	RAS2	E_{rel} [eV]	ΔE_{S2-S1} [eV]
6-31G*	(30,13+5+20)	(4,5)	S2: 6.41	0.67
			S1: 5.74	
6-31G*	(20,8+5+11)	(4,5)	S2: 6.01	0.74
			S1: 5.27	
6-31G*	(20,7+5+12)	(6,5)	S2: 6.74	0.68
			S1: 6.06	
6-31G*	(20,7+6+11)	(6,6)	S2: 5.98	0.83
			S1: 5.15	
6-311+G*	(20,7+6+11)	(6,6)	S2: 5.73	0.65
			S1: 5.08	

occupancies from the resulting RAS(20,8+5+11)[1,1] are also shown in table 3.10. The excitation energies obtained from these calculations are shown in table 3.11. In general, all calculated energies for S1 are 0.7-1.4 eV higher than the experimental value. However, the energy gap between S1 and S2 is much more accurate than the CASSCF value of 1.20 eV (cf. table 3.8). The reduction of the active space has only a small effect, while the use of the larger (6-311+G*) basis set improves the results substantially.

Figure 3.18 shows the potential energy surface as computed at the RAS(20,7+6+11)[1,1]/6-31G* level. The results in the figure are qualitatively correct, although the 0–0 energy for the ^1L_a state is off by more than 1 eV. In contrast to the vertical excitations, the results with the 6-311+G* basis are worse than the ones with 6-31G*. The reason for this is

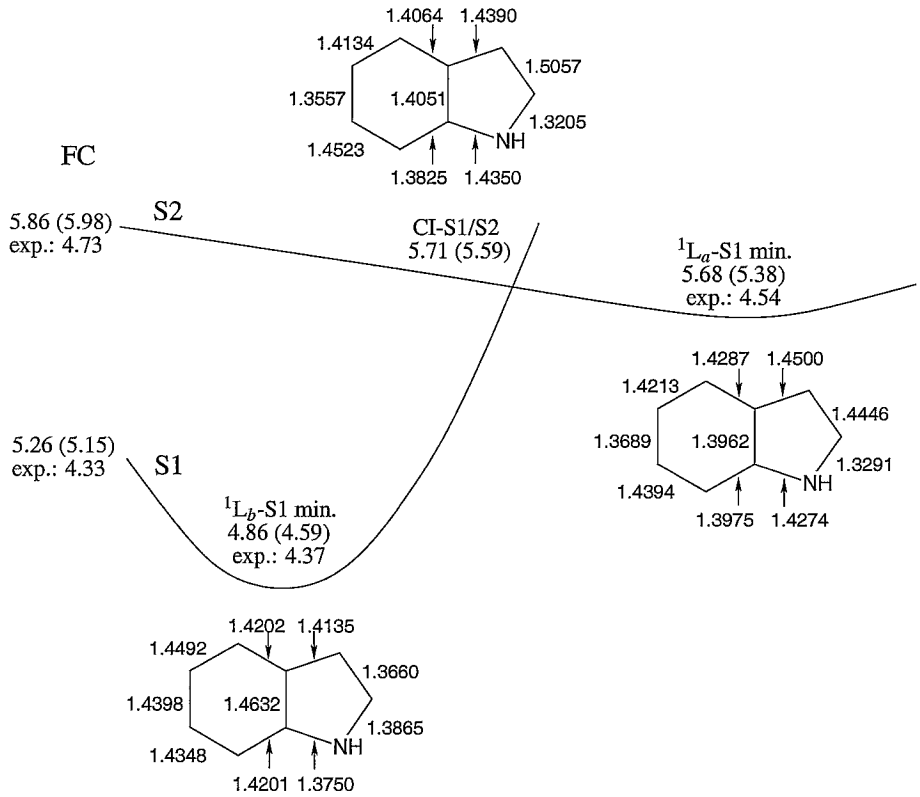


Figure 3.18.: Potential energy surfaces calculated at the RASSCF level (energies in eV, bond distances in Å). The geometry optimisations were carried out at the RAS(20,7+6+11)[1,1]/6-31G* level (number in brackets) end the energetics recalculated with a larger basis set, at the RAS(20,7+6+11)[1,1]/6-311+G* level.

probably that the points were optimised with the latter basis.

The result of this first application of our RASSCF program is encouraging. The improvements of the obtained excitation energies compared to the CASSCF(10,9)/6-31G* is considerable, but the crucial result is the qualitatively correct potential energy surface. The computational cost increases only moderately, with the number of CSFs increasing from 8001 for the CASSCF(10,9) computation, to 53735 for the RASSCF(20,7+6+11)[1,1].

3.8.2. Vertical excitation energy of all-trans-hexatriene

The first two singlet excited states of the all-trans polyene 1,3,5-hexatriene (*tEt*-hexatriene) are the 1^1B_u and the 2^1A_g states, respectively. As noted in reference [67], "the state ordering of 1^1B_u and 2^1A_g for polyenes is a rather challenging problem". In the case of *tEt*-hexatriene this is true for experimental verification [68, and references therein], and also for computational, *ab initio* methods. Experimental results show that very shortly (40–50 fs) after excitation to the 1^1B_u state, a conical intersection is encountered, leading to a lower lying intermediate, which must be the 2^1A_g state. We show some results obtained from the literature in table 3.12.

In order to study the photochemistry of *tEt*-hexatriene one needs a qualitatively and to a certain extent quantitatively accurate description of the potential energy surface in the vicinity of the Franck-Condon geometry and the S1-S2 surface crossing. *Ab initio* calculations generally find two very close lying states, and most methods seem to place the 1^1B_u state just below the 2^1A_g state at the Franck-Condon geometry.

The study undertaken here starts of with a CASSCF(6,6)/6-31G*, with the active space consisting the valence π and π^* orbitals. However, as expected the CASSCF(6,6) calculation gives poor results, as the polar 1^1B_u state is placed 1.9 eV above the 2^1A_g state (cf. table 3.13). Doubling the active space, and including explicitly 3p functions in the basis set for improved description of the polar 1^1B_u state, reduces the energy gap to 0.65 eV. RASSCF calculations allow to improve this result as the two states get closer in energy. Indeed, the 2^1A_g state is placed just 0.28 eV below the 1^1B_u state when using state averaged orbitals. This result is in reasonable agreement with previous theoretical

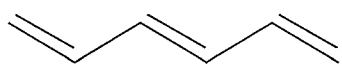


Figure 3.19.: *tEt*-hexatriene

Table 3.12.: Vertical excitation energies to the 1^1B_u and 2^1A_g states of *tEt*-hexatriene (Lit.).

	1^1B_u [eV]	2^1A_g [eV]
CASSCF(6,8)/ANO [69]	7.36	5.65
CASPT2 [69]	5.01	5.20
CASPT2 [68]	5.42	4.92
MRMP [70]	5.10	5.09
VSEH [71]	5.21	5.20
QR-CC3 [72]	5.73	5.73
TDDFT [67]	5.06	5.66
Expt. (references in [67])	4.95	5.21
Expt. (references in [68])	5.1–5.2	-

Table 3.13.: Vertical excitation energies to the 1^1B_u and 2^1A_g states of *tEt*-hexatriene (results from state-averaged calculations in italics).

	1^1B_u [eV]	2^1A_g [eV]
CASSCF(6,6)/6-31G*	*7.57*	5.69
CASSCF(6,12)/6-31G*	6.76	5.66
CASSCF(6,12)/6-31G*+3p	6.26	5.61
RASSCF(22,13+6+19)[1,1]/6-31G*+3p	*6.03*	5.84, *5.75*

calculations, although the excitation energies are probably too high.

We now describe the chosen method to select the orbitals included in the different RAS subspaces. The RAS2 subspace was taken to comprise the six valence π orbitals from the CASSCF computation. The remaining valence electrons were included in RAS1, resulting in 13 orbitals in that subspace. Finally, the RAS3 orbitals were chosen by comparison with the corresponding valence orbitals from a previous computation using the STO-3G basis set, and the extra 6 π-electrons were added, resulting in a total of 19 RAS3 orbitals. Excitation of up to 1 electron out of the RAS1 subspace and into the RAS3 subspace was allowed for.

As in the previous application, the improvement of the calculated energies compared to the CASSCF computation is encouraging, while the length of the RASSCF CI vector is still very manageable ($n_{\text{confs}} = 164185$). The good agreement with accurate *ab initio* calculations obtained at the RASSCF level allows considering the geometry optimisation of the conical intersection between the 1^1B_u and 2^1A_g state involved in the photochemistry of *tEt*-hexatriene as well as the calculations of minimum energy paths.

3.9. Summary

In this chapter we have developed an efficient method for updating the CI vector in a RASSCF computation within the iterative Davidson/Lanczos methods. The string based algorithm arises due to two new concepts:

- the RAS model space concept ensures very efficient identification of blocks of contributions to the $\boldsymbol{\sigma}$-vector, and we have made use of the reduced list algorithm previously introduced in chapter 2 for efficient CASSCF computations. The advantage of the RAS model space is that it represents a restricted space problem in an unrestricted manner, thus avoiding the complications inherent in the RASSCF method.
- propagation rules are used to efficiently construct the string addresses (as opposed to the strings) in a predefined order. Propagation rules are a simple, but very efficient algebraic tool avoiding table-lookups and construction of contributing strings themselves.

The resulting method is memory efficient, since it utilises compressed storage of precomputed model excitation lists in main memory. Construction of the full excitation strings is

fully avoided. The developed algorithm follows an integral driven approach. The resulting outer nested loops over model orbital indices w,x,y,z lends itself to efficient parallelisation. Every model space index quadruple w,x,y,z represents a list of integrals $(ij|kl)$, i.e. we have an *integral block driven* approach. Clusters of index quadruples with variable cluster size ensure adjustable granularity of the parallelism, thus avoiding scaling problems. Communication overhead (if run in parallel) is minimal through implicit task definition by only one integer. This technique also leads to a natural load balancing.

The algorithm is implemented in the current development version of the Gaussian package of programs [54]. The implementation comprises the option of parallel execution. Running in parallel may be done using distributed memory (Linda) or shared memory, or a combination of these. As well as the most general case of Slater determinants we also implemented the straightforward simplifications for singlets and triplets using Hartree Waller functions.

Two test example applications have been performed with the current version of the program. The results show that the RASSCF method is a useful tool allowing to approach problems that are currently not tractable with the CASSCF method.

4. Conclusion

The CASSCF method is a powerful tool for studying electronic structure problems of multi-configurational nature, but because a full CI is carried out amongst the active orbitals it quickly becomes impossible even for modest active spaces. The computational bottleneck lies within the CI part of the CASSCF formalism, where in modern implementations the full diagonalisation of the Hamiltonian matrix is replaced by iterative diagonalisation methods by Lanczos [32] or Davidson [49], and the Hamiltonian matrix is multiplied with a guess vector. For large scale CASSCF computations it is essential that the sparsity of the Hamiltonian matrix is fully exploited, and the algorithm computing the symbolic matrix elements is optimised with respect to efficiency.

This thesis addresses two ideas to alleviate the problem of the CASSCF bottleneck. The direct symbolic matrix element method presented here utilises lists of compressed excitation strings, so called reduced lists. Non-vanishing matrix elements are found efficiently and without index testing, using binary transformations of these strings. The improved efficiency comes from the fact that testing of index spaces is entirely avoided. The storage requirements of the reduced lists are very small compared to the length of the CI vector, and can be precomputed and held in working memory throughout. Separate reduced lists exist for double excitation contributions of the Hamiltonian, thereby avoiding the need for using the resolution of the identity. The implemented program is designed to take full advantage of the sparsity of the Hamiltonian matrix, which becomes more sparse as the problem size increases.

The method has been implemented and parallelised, allowing for distributed memory, shared memory and hybrid (distributed and shared memory) parallel architectures. Care has been taken to maximise efficiency and to achieve good scaling behaviour. The main features of the parallel implementation are a well functioning load-balancing scheme, aided by single integer task-definitions for parallel tasks, and linearly decreasing task

sizes coupled with an implicit task order. The program has been implemented as part of the Gaussian set of programmes [54]. Using the parallel version of the program, a calculation has been performed on stilbene, with 14 electrons in 14 active orbitals without any symmetry restrictions, achieving a speedup in real time of 24.1 on 32 processors (with speedup of parallel parts of the program being >29). Another calculation on pentalene, with 8 electrons in 16 orbitals, showed a speedup in real time of 20.5 on 24 processors (with speedup of parallel parts of the program ≈ 23). Comparison with the previously implemented string based direct CI algorithm shows a speedup of ≈ 2 for modest active spaces (8–10 active orbitals). For larger problems a higher speedup is expected.

The combined memory requirements of the reduced lists are negligible. The memory requirements of our CASSCF algorithm are dominated by the need to store two CI vectors in working memory (three in the parallel version). Parallel scaling behaviour improves with increasing problem size, and has been shown to be very good, due to an integrated load-balancing mechanism, and the minimisation of interprocess-communication. The program is thus only limited by the amount of available working memory.

The second methodical contribution of this thesis relates to the RASSCF method. This generalisation of the CASSCF method allows much larger active spaces, thus enabling the study of larger systems, or enhancing qualitative and quantitative accuracy by partial inclusion of dynamic electron correlation effects. A special direct symbolic matrix element method has been developed, based on the concept of a RAS model string, and so called propagation rules. The RAS model string technique is implemented using the same reduced string concept that was developed for the CASSCF method; it allows identification of blocks of non-vanishing matrix elements of the Hamiltonian matrix, by finding the representing model string pairs. The powerful propagation rules take full advantage of the structure of a given block of matrix elements, by computing the sequence of string addresses directly, without finding the corresponding strings first. The presented algorithm has been implemented as part of the Gaussian set of programmes [54]. The parallel version of the program, as implemented, allows for distributed, shared, and hybrid parallelism. Load balancing has been implemented as for the CASSCF program. First performance tests show that the algorithm is indeed very efficient.

As for the CASSCF, the precomputable reduced lists have negligible memory requirements. All that is needed is the assembly of a list of the integrals in the order they are processed, which in turn is given by the propagation rules. The assembly of this list needs to be done only once for every model space index quadruple.

4. *Conclusion*

Two calculations using the RASSCF program are presented as examples for possible application of the method, namely a study of the potential energy surface of indole, and the excited state ordering of *tEt*-hexatriene, including up to 32 electrons in 38 active orbitals (*tEt*-hexatriene). In both cases considerable improvements compared to the tractable corresponding CASSCF calculations have been achieved.

A. CASSCF: Implementation Details

We have implemented the reduced list algorithm as described in chapter 2 of this thesis, as part of the Gaussian package of programs [54]. Our implementation replaces the set of routines previously performing the same tasks within the direct CASSCF part of link l510 (the Gaussian program performing MCSCF computations). We have implemented both a serial and a parallel version of the program. Figures A.1 and A.2 show the subroutine-tree for serial and parallel execution. A brief description of the routines is added.

In order to improve overall scaling of the direct CASSCF procedure, we have parallelised the parts of the MCSCF program that also comprise outer loops over orbital indices i,j,k,l, using the same strategy as for the CI vector updating part. Figures A.7-A.12 show the subroutine trees for serial and parallel (Linda) implementations.

FlyUp/MCAXMI Top level routine. Perform memory allocation. Call subroutine UpdRas.

UpdatM Compute Handy's index matrix, **IIZ**, and all reduced lists ($\mathcal{L}^{M-1}_{N_\sigma-1}$, $\mathcal{L}^{M-2}_{N_\sigma-1}$, $\mathcal{L}^{M-2}_{N_\sigma-2}$, $\mathcal{L}^{M-3}_{N_\sigma-2}$ and $\mathcal{L}^{M-4}_{N_\sigma-2}$ $\forall \sigma \in \{\alpha,\beta\}$) . Read the integral list from file. Call subroutine UpdWor.

UpdWor (Parallel) loop over tasks, defined by orbital index pair, ij. – Call subroutine UpdJob.

- Serial version (no parallelism): simple loop over orbital index $(ij) = \frac{i(i-1)}{2} + j$.
- Shared memory parallelism only: create *NProcS* shared memory processes and let each PE compute the sub-task $(ij)^a_0$, $(ij)^a_0 - NProcS$, $(ij)^a_0 - 2NProcS$, ..., where $(ij)^a_0$ is the first task-index allocated to the PE labelled a; PEs a, b, ... will be allocated first sub-task indices $(ij)^a_0 = \frac{M(M+1)}{2}$, $(ij)^b_0 = (ij)^a_0 - 1$, etc.

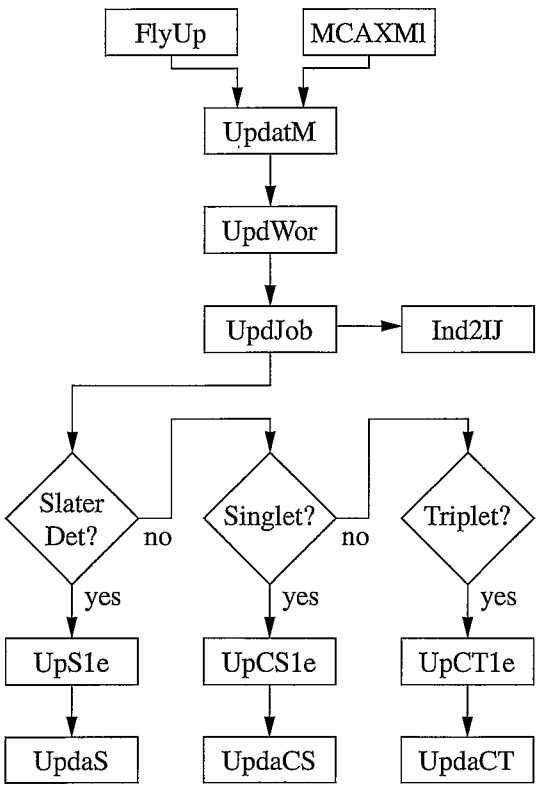


Figure A.1.: Subroutine calling sequence for CI vector updating algorithm.

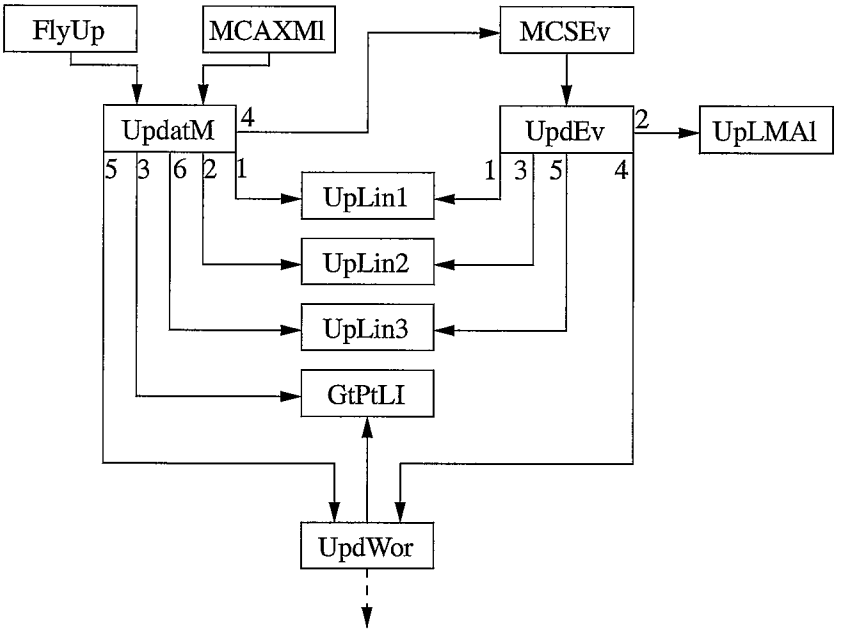


Figure A.2.: Linda communication for CI vector updating algorithm.

- Distributed memory parallelism (Linda) only: read (ij) from tuple space, and return $(ij)' = (ij) - 1$ to tuple space.
- Mixed (shared and distributed) parallel jobs are allocated to nodes as blocks of *NProcS* sub-tasks, i.e. $(ij)' = (ij) - NProcS$ is returned to tuple space, and one sub-task is allocated to each PE on the node.

UpdJob See figure A.3.

Ind2IJ Translate the task index (ij) into orbital indices i and j.

UpS1e See figure A.4.

UpCS1e Like UpS1e, but for spin adapted basis (Hartree Waller singlet).

UpCT1e Like UpS1e, but for spin adapted basis (Hartree Waller triplet).

UpdaS See figure A.5.

UpdaCS Like UpdaS, but for spin adapted basis (Hartree Waller singlet).

UpdsCT Like UpdaS, but for spin adapted basis (Hartree Waller triplet).

INumGt Compute sign factors corresponding to parity.

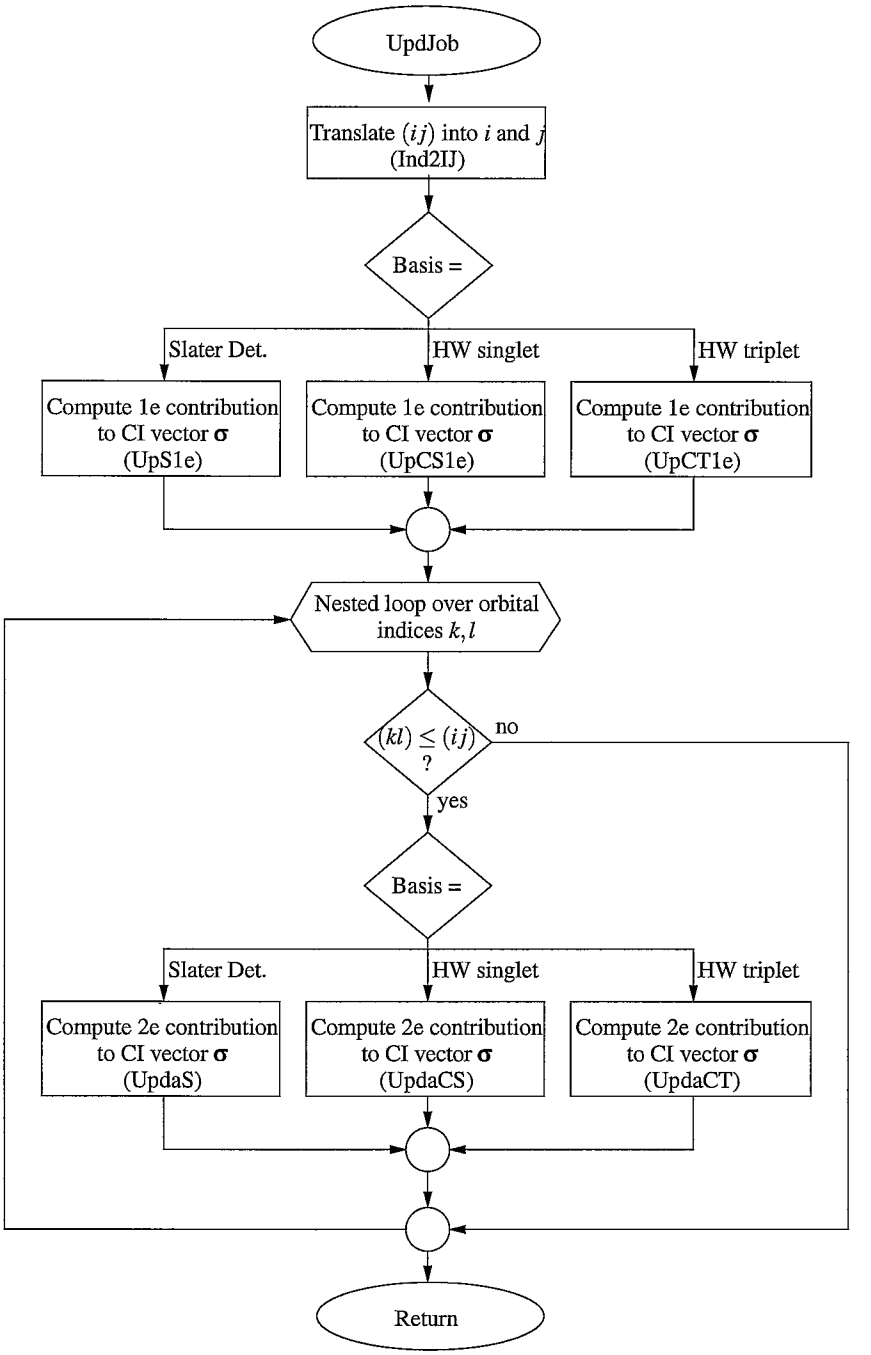


Figure A.3.: The subroutine UpdJob is the interface routine of the CI vector updating.

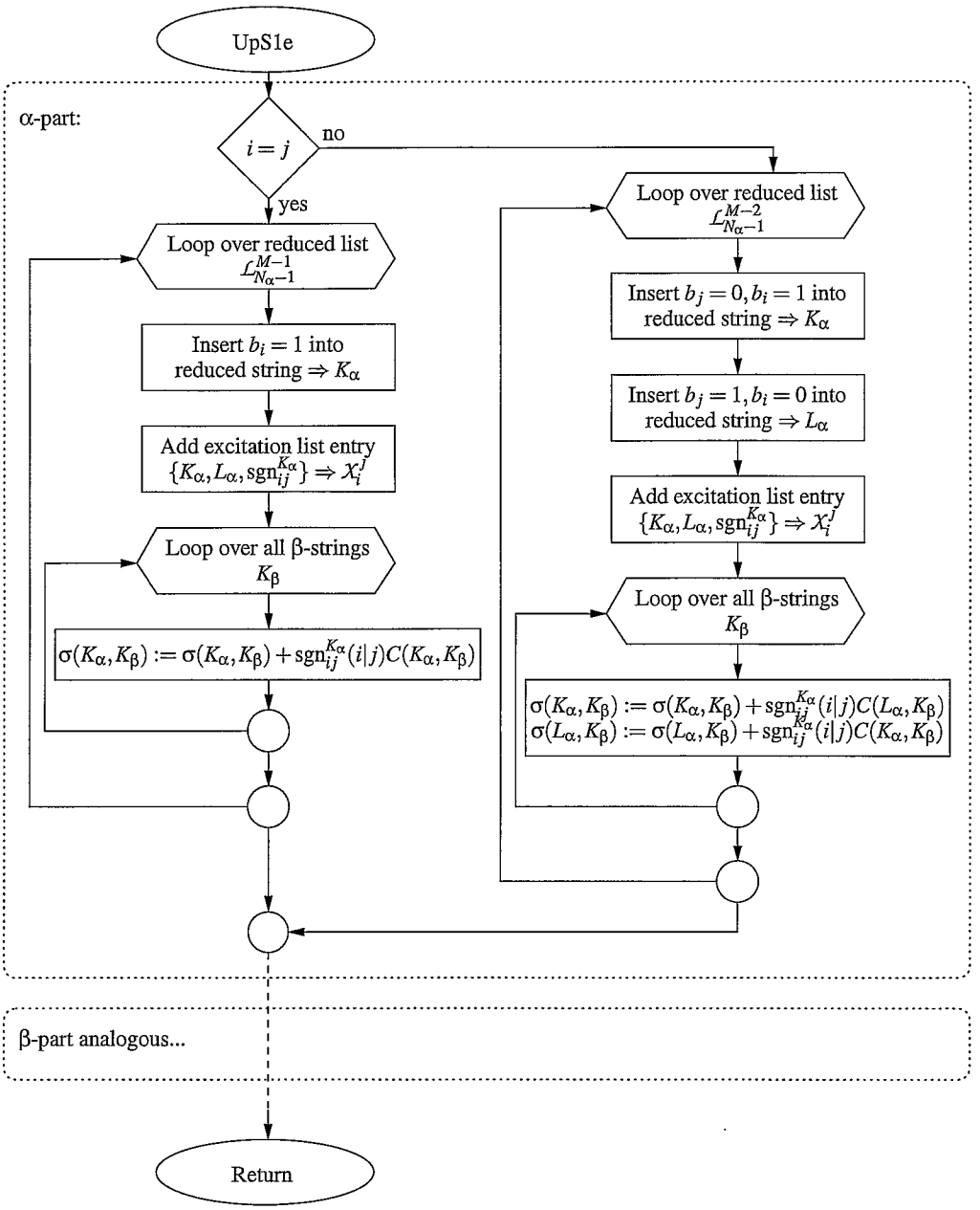


Figure A.4.: The subroutine UpS1e performs the updating for the 1e contribution.

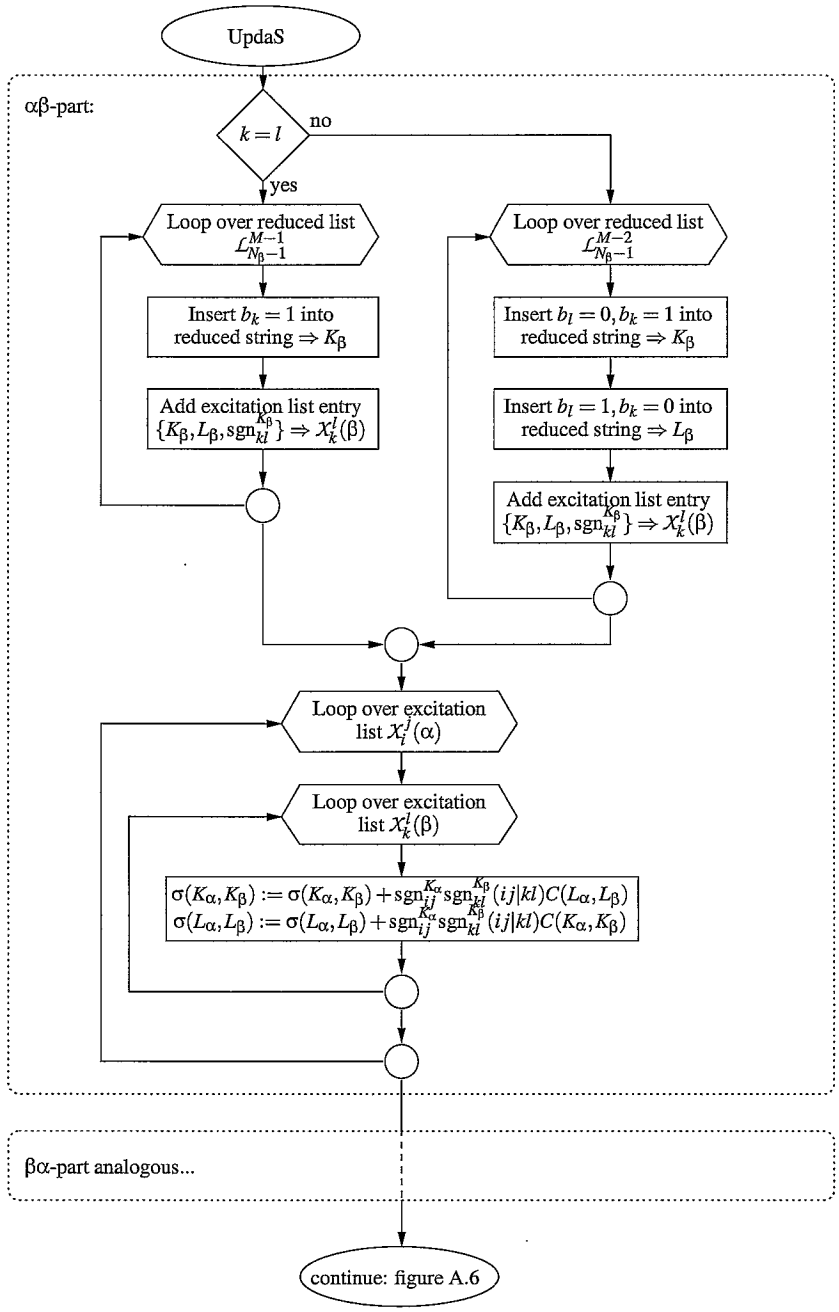


Figure A.5.: The subroutine UpdaS performs the updating for the 2e contribution.

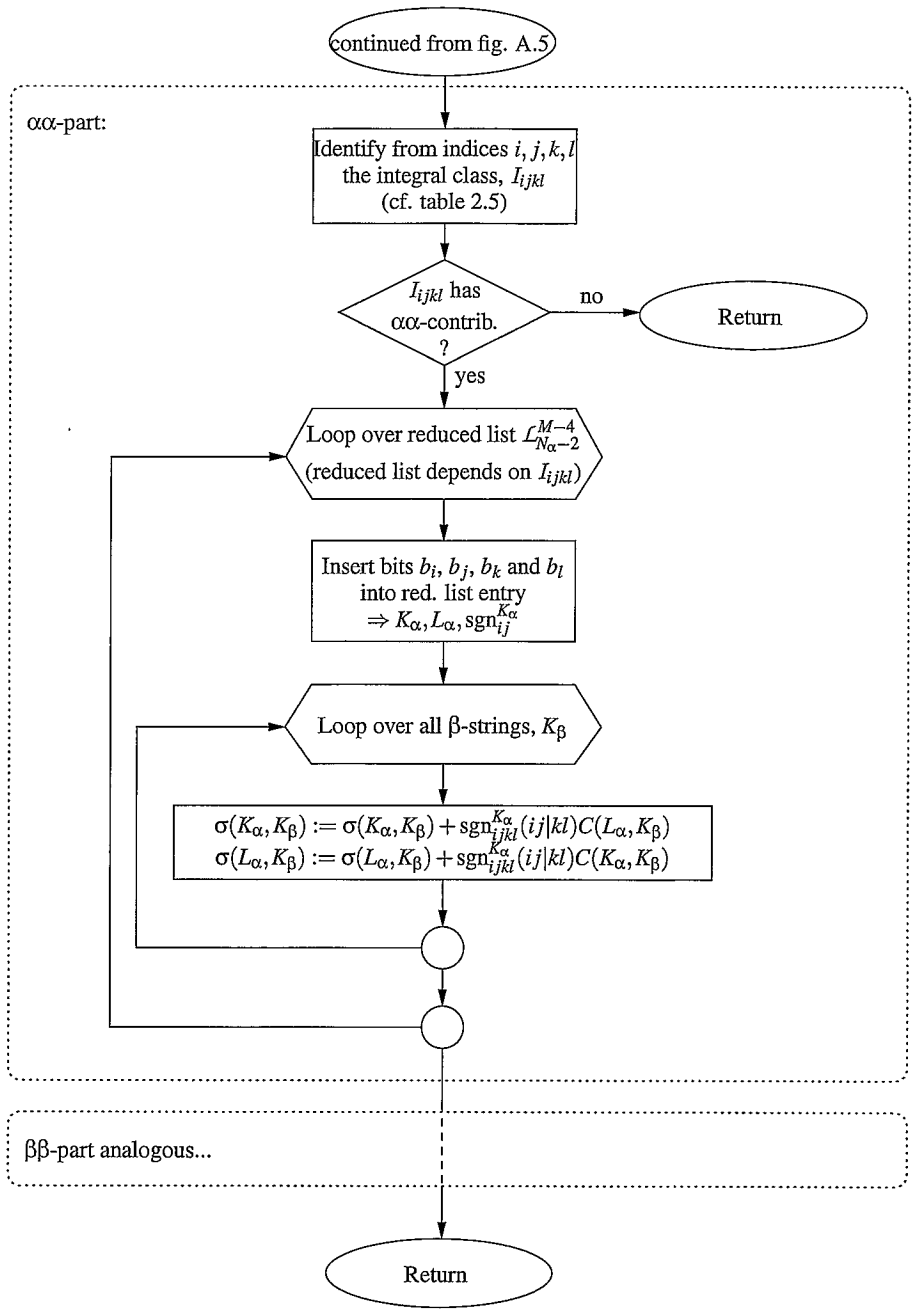


Figure A.6.: The subroutine UpdaS (continued from figure A.5).

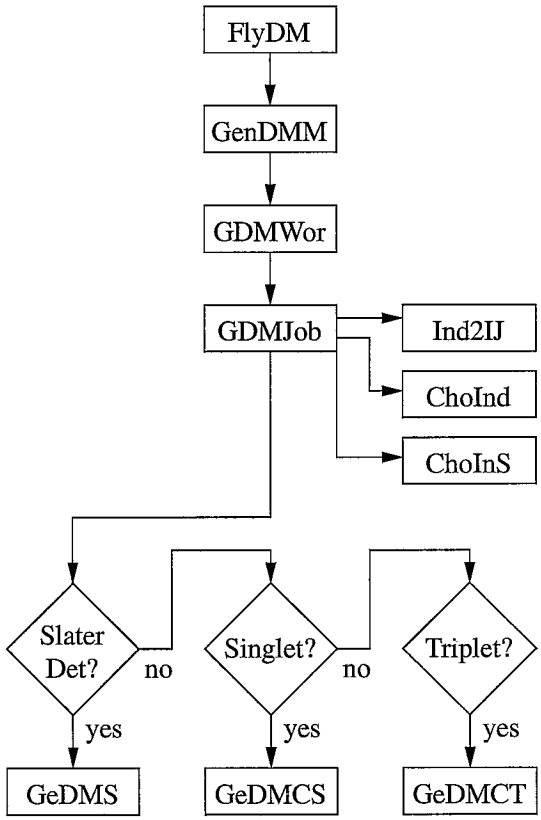


Figure A.7.: Subroutine calling sequence for density matrix algorithm.

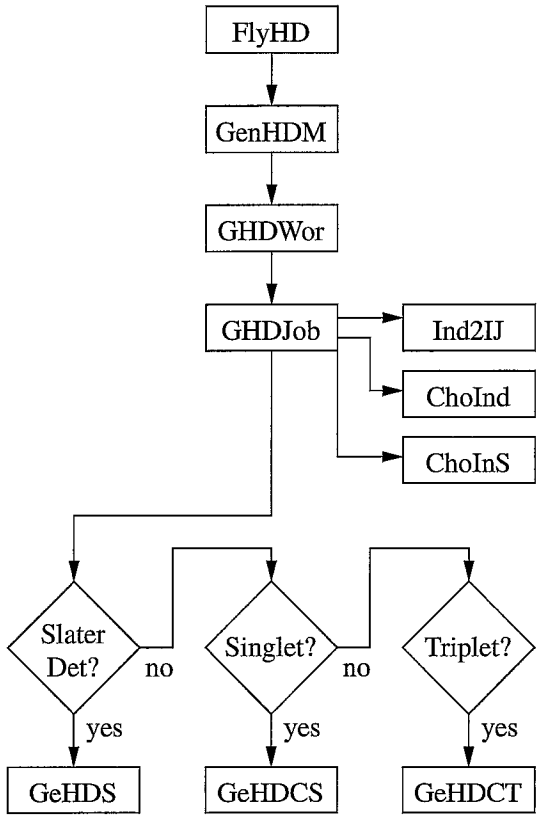


Figure A.8.: Subroutine calling sequence for diagonal Hamiltonian algorithm.

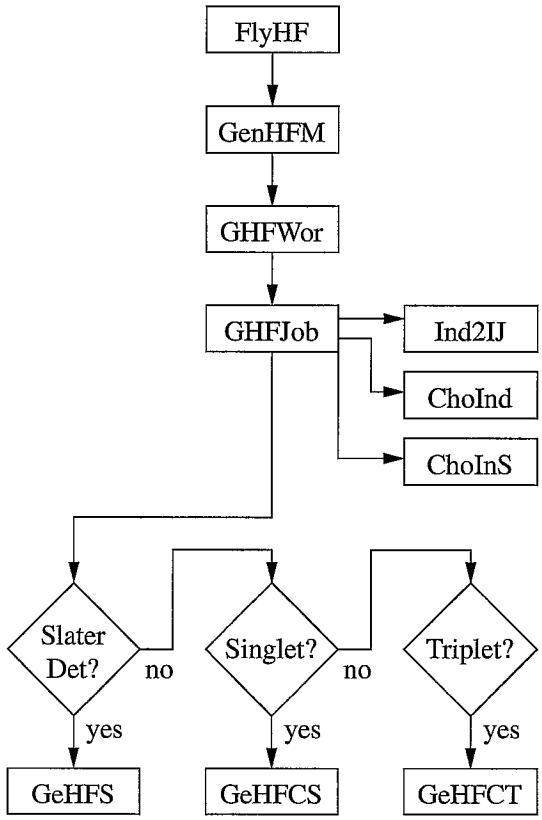


Figure A.9.: Subroutine calling sequence for reduced Hamiltonian algorithm.

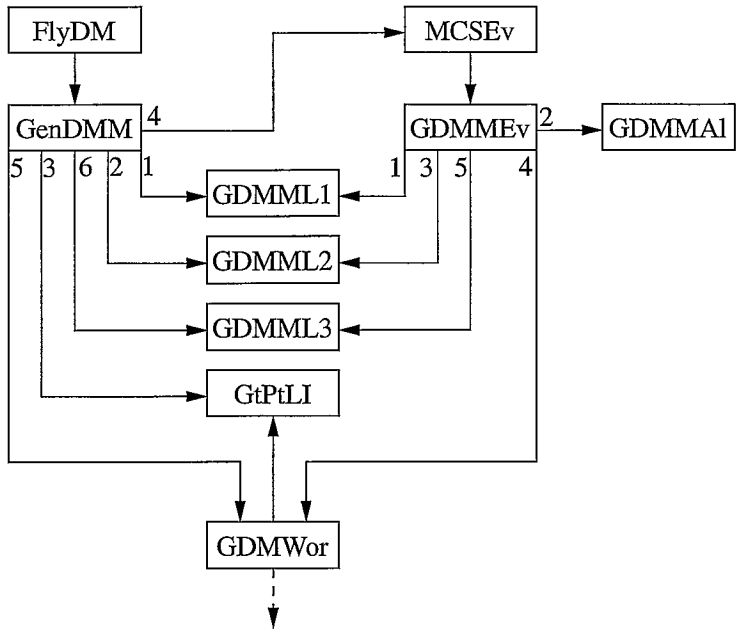


Figure A.10.: Linda communication for density matrix algorithm.

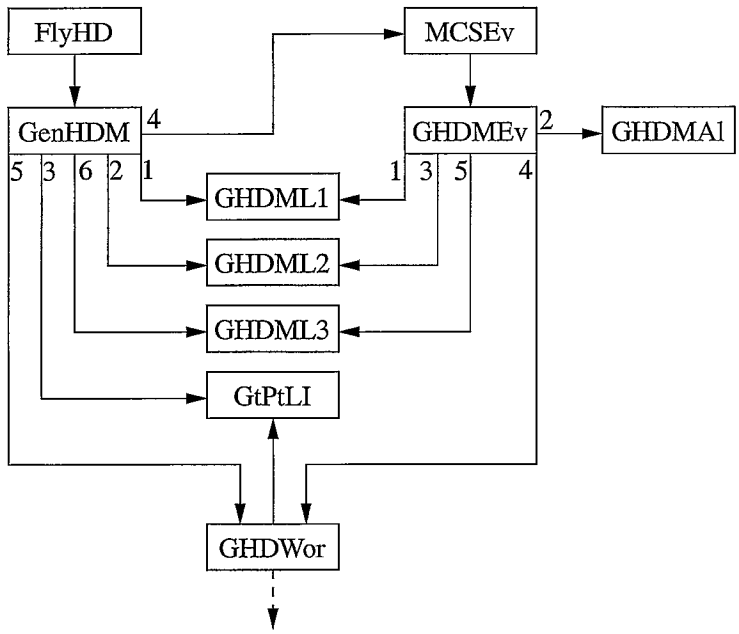


Figure A.11.: Linda communication for diagonal Hamiltonian algorithm.

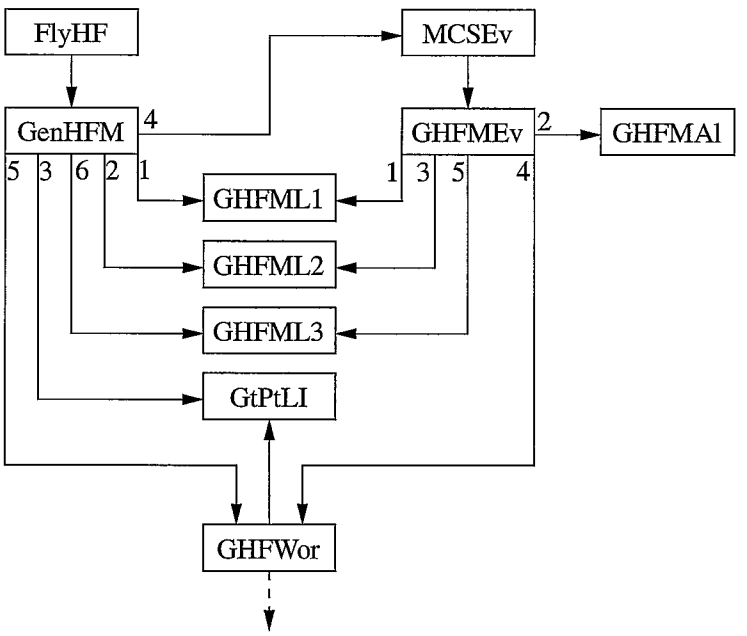


Figure A.12.: Linda communication for reduced Hamiltonian algorithm.

B. Propagation Rules

The full list of propagation rules is listed. The propagation rules are denoted $\mathcal{P}^{b_a(L_\gamma^{\mathrm{RASX}})}_{b_a(K_\gamma^{\mathrm{RASX}})}$, where the subscript contains the bit-values of the variable bits b_a, b_b, etc. in the RASX substring of string K_γ, and the superscripts contains the bit-values of the same bits b_a, b_b, etc. of string L_γ.

$\mathcal{P}^{-}_{-}\ (K_\gamma^{\mathrm{RASX}} = L_\gamma^{\mathrm{RASX}})$

This is the most common and simplest propagation rule. The subscript and superscript $(-)$ indicate that there are no variable bits in the corresponding RASX subspace, hence the substrings K_γ^{RASX} and L_γ^{RASX} coincide. As a result the propagation is trivial:

$$\begin{aligned} \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} &= \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\}+1 \\ \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} &= \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} \end{aligned}$$

This rule also applies for all other cases, whenever progression to the next reduced substring list takes place (for details see chapter 3, section 3.4)

$\underline{\mathcal{P}^{1}_{0}\ (K_\gamma^{\mathrm{RASX}}:1; L_\gamma^{\mathrm{RASX}}:0)}$

$\Delta a = 1$:

$$\begin{aligned} \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} &= \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\}+\binom{M(\mathrm{RAS}X)-a}{N(M^{\mathrm{left}}(a))}\times(1-b_{a-1}(\mathrm{r.s.})) \\ \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} &= \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\}-\binom{M(\mathrm{RAS}X)-a}{N(M^{\mathrm{left}}(a))}\times b_{a-1}(\mathrm{r.s.}) \end{aligned}$$

$\underline{\mathcal{P}^{0}_{1}\ (K_\gamma^{\mathrm{RASX}}:0; L_\gamma^{\mathrm{RASX}}:1)}$

$\Delta a = 1$:

$$\begin{aligned} \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} &= \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\}-\binom{M(\mathrm{RAS}X)-a}{N(M^{\mathrm{left}}(a))}\times b_{a-1}(\mathrm{r.s.}) \\ \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} &= \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\}+\binom{M(\mathrm{RAS}X)-a}{N(M^{\mathrm{left}}(a))}\times(1-b_{a-1}(\mathrm{r.s.})) \end{aligned}$$

$\underline{\mathcal{P}_1^1\ (K_\gamma^{\mathrm{RASX}}:1; L_\gamma^{\mathrm{RASX}}:1)}$

$\Delta a = 1$:

$$\begin{aligned}
\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} &= \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-1}(\mathrm{r.s.})) \\
\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} &= \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\}
\end{aligned}$$

$\underline{\mathcal{P}_{01}^{10}\ (K_\gamma^{\mathrm{RASX}}:10; L_\gamma^{\mathrm{RASX}}:01)}$

$\Delta a = 1, \Delta b = 0$:

$$\begin{aligned}
\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} &= \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-2}(\mathrm{r.s.})) \\
\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} &= \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} - \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-2}(\mathrm{r.s.})
\end{aligned}$$

$\Delta a = 1, \Delta b = 1$:

$$\begin{aligned}
\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} &+ \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-2}(\mathrm{r.s.})) \\
&- \binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times b_{b-1}(\mathrm{r.s.})
\end{aligned}$$

$$\begin{aligned}
\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} &- \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-2}(\mathrm{r.s.}) \\
&+ \binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times (1 - b_{b-1}(\mathrm{r.s.}))
\end{aligned}$$

note: $b_{a-2}(\mathrm{r.s.}) = b_{b-1}(\mathrm{r.s.})$! The binomial coefficients containing the variable b are not always identical to those containing the variable a and they are different for K_γ^{RASX} and L_γ^{RASX}. Instead their $N(M^{\mathrm{left}}(b))$ is increased by one for K_γ^{RASX}.

$\underline{\mathcal{P}_{10}^{01}\ (K_\gamma^{\mathrm{RASX}}:01; L_\gamma^{\mathrm{RASX}}:10)}$

$\Delta a = 1, \Delta b = 0$:

$$\begin{aligned}
\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} &= \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} - \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-2}(\mathrm{r.s.}) \\
\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} &= \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-2}(\mathrm{r.s.}))
\end{aligned}$$

$\Delta a = 1, \Delta b = 1$:

$$\begin{aligned}
\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} &- \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-2}(\mathrm{r.s.}) \\
&+ \binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times (1 - b_{b-1}(\mathrm{r.s.}))
\end{aligned}$$

$$\begin{aligned}\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \quad \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} \quad &+\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1-b_{a-2}(\mathrm{r.s.})) \\ &-\binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times b_{b-1}(\mathrm{r.s.})\end{aligned}$$

$\underline{\mathcal{P}_{11}^{00}\ (K_\gamma^{\mathrm{RASX}}: 00; L_\gamma^{\mathrm{RASX}}: 11)}$

$\Delta a = 1, \Delta b = 0$:

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} &= \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} - \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-2}(\mathrm{r.s.}) \\ \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} &= \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1-b_{a-2}(\mathrm{r.s.}))\end{aligned}$$

$\Delta a = 1, \Delta b = 1$:

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \quad \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} \quad &-\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-2}(\mathrm{r.s.}) \\ &-\binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times b_{b-1}(\mathrm{r.s.})\end{aligned}$$

$$\begin{aligned}\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \quad \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} \quad &+\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1-b_{a-2}(\mathrm{r.s.})) \\ &+\binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times (1-b_{b-1}(\mathrm{r.s.}))\end{aligned}$$

$\underline{\mathcal{P}_{00}^{11}\ (K_\gamma^{\mathrm{RASX}}: 11; L_\gamma^{\mathrm{RASX}}: 00)}$ $(K_\gamma^{\mathrm{RASX}}: 11; L_\gamma^{\mathrm{RASX}}: 00)$

$\Delta a = 1, \Delta b = 0$:

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} &= \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1-b_{a-2}(\mathrm{r.s.})) \\ \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} &= \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} - \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-2}(\mathrm{r.s.})\end{aligned}$$

$\Delta a = 1, \Delta b = 1$:

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \quad \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} \quad &+\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1-b_{a-2}(\mathrm{r.s.})) \\ &+\binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times (1-b_{b-1}(\mathrm{r.s.}))\end{aligned}$$

$$\begin{aligned}\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \quad \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} \quad &-\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-2}(\mathrm{r.s.}) \\ &-\binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times b_{b-1}(\mathrm{r.s.})\end{aligned}$$

B. Propagation Rules

$\underline{\mathcal{P}_{11}^{11}\,(K_\gamma^{\mathrm{RASX}}:11;L_\gamma^{\mathrm{RASX}}:11)}$

$\Delta a=1,\Delta b=0$:

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} &= \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\}+\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))}\times(1-b_{a-2}(\mathrm{r.s.}))\\ \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} &= \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\}\end{aligned}$$

$\Delta a=1,\Delta b=1$:

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\}= \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} &+\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))}\times(1-b_{a-2}(\mathrm{r.s.}))\\ &+\binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))}\times(1-b_{b-1}(\mathrm{r.s.}))\\ \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\}= \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\}&\end{aligned}$$

$\underline{\mathcal{P}_{10}^{11}\,(K_\gamma^{\mathrm{RASX}}:11;L_\gamma^{\mathrm{RASX}}:10)}$

$\Delta a=1,\Delta b=0$:

$$\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\}=\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\}+\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))}\times(1-b_{a-2}(\mathrm{r.s.}))$$

$$\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\}=\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\}+\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))}\times(1-b_{a-2}(\mathrm{r.s.}))$$

$\Delta a=1,\Delta b=1$:

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\}= \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} &+\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))}\times(1-b_{a-2}(\mathrm{r.s.}))\\ &+\binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))}\times(1-b_{b-1}(\mathrm{r.s.}))\end{aligned}$$

$$\begin{aligned}\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\}= \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} &+\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))}\times(1-b_{a-2}(\mathrm{r.s.}))\\ &-\binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))}\times b_{b-1}(\mathrm{r.s.})\end{aligned}$$

$\underline{\mathcal{P}_{11}^{10}\,(K_\gamma^{\mathrm{RASX}}:10;L_\gamma^{\mathrm{RASX}}:11)}$

$\Delta a=1,\Delta b=0$:

$$\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\}=\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\}+\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))}\times(1-b_{a-2}(\mathrm{r.s.}))$$

$$\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\}=\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\}+\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))}\times(1-b_{a-2}(\mathrm{r.s.}))$$

B. Propagation Rules

$\Delta a = 1, \Delta b = 1$:

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} \quad + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-2}(\mathrm{r.s.})) \\ & - \binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times b_{b-1}(\mathrm{r.s.})\end{aligned}$$

$$\begin{aligned}\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} \quad + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-2}(\mathrm{r.s.})) \\ & + \binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times (1 - b_{b-1}(\mathrm{r.s.}))\end{aligned}$$

$\underline{\mathcal{P}_{010}^{101}\ (K_\gamma^{\mathrm{RASX}} : 101; L_\gamma^{\mathrm{RASX}} : 010)}$

$\Delta a = 1, \Delta b = 0, \Delta c = 0$:

$$\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-3}(\mathrm{r.s.}))$$

$$\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} - \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-3}(\mathrm{r.s.})$$

$\Delta a = 1, \Delta b = 1, \Delta c = 0$:

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} \quad + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-3}(\mathrm{r.s.})) \\ & - \binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times b_{b-2}(\mathrm{r.s.})\end{aligned}$$

$$\begin{aligned}\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} \quad - \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-3}(\mathrm{r.s.}) \\ & + \binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times (1 - b_{b-2}(\mathrm{r.s.}))\end{aligned}$$

$\Delta a = 1, \Delta b = 1, \Delta c = 1$:

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} \quad + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-3}(\mathrm{r.s.})) \\ & - \binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times b_{b-2}(\mathrm{r.s.}) \\ & + \binom{M(\mathrm{RASX})-c}{N(M^{\mathrm{left}}(c))} \times (1 - b_{c-1}(\mathrm{r.s.}))\end{aligned}$$

B. *Propagation Rules*

$$\begin{aligned}\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} \quad -\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-3}(\mathrm{r.s.}) \\ & +\binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times (1-b_{b-2}(\mathrm{r.s.})) \\ & -\binom{M(\mathrm{RASX})-c}{N(M^{\mathrm{left}}(c))} \times b_{c-1}(\mathrm{r.s.})\end{aligned}$$

note: $b_{a-3}(\mathrm{r.s.}) = b_{b-2}(\mathrm{r.s.}) = b_{c-1}(\mathrm{r.s.})$!

$\underline{\mathcal{P}_{011}^{100}\,(K_\gamma^{\mathrm{RASX}}:100; L_\gamma^{\mathrm{RASX}}:011)}$

$\Delta a = 1, \Delta b = 0, \Delta c = 0$:

$$\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1-b_{a-3}(\mathrm{r.s.}))$$

$$\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} - \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-3}(\mathrm{r.s.})$$

$\Delta a = 1, \Delta b = 1, \Delta c = 0$:

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} \quad +\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1-b_{a-3}(\mathrm{r.s.})) \\ & -\binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times b_{b-2}(\mathrm{r.s.})\end{aligned}$$

$$\begin{aligned}\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} \quad -\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-3}(\mathrm{r.s.}) \\ & +\binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times (1-b_{b-2}(\mathrm{r.s.}))\end{aligned}$$

$\Delta a = 1, \Delta b = 1, \Delta c = 1$:

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} \quad +\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1-b_{a-3}(\mathrm{r.s.})) \\ & -\binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times b_{b-2}(\mathrm{r.s.}) \\ & -\binom{M(\mathrm{RASX})-c}{N(M^{\mathrm{left}}(c))} \times b_{c-1}(\mathrm{r.s.})\end{aligned}$$

$$\begin{aligned}\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} \quad -\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-3}(\mathrm{r.s.}) \\ & +\binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times (1-b_{b-2}(\mathrm{r.s.})) \\ & +\binom{M(\mathrm{RASX})-c}{N(M^{\mathrm{left}}(c))} \times (1-b_{c-1}(\mathrm{r.s.}))\end{aligned}$$

B. Propagation Rules

$\underline{\mathcal{P}_{001}^{110}\,(K_\gamma^{\mathrm{RASX}}:110;L_\gamma^{\mathrm{RASX}}:001)}$

$\Delta a=1,\Delta b=0,\Delta c=0$:

$$\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\}=\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\}+\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))}\times(1-b_{a-3}(\mathrm{r.s.}))$$

$$\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\}=\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\}-\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))}\times b_{a-3}(\mathrm{r.s.})$$

$\Delta a=1,\Delta b=1,\Delta c=0$:

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\}=\ \ \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\}\ &+\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))}\times(1-b_{a-3}(\mathrm{r.s.}))\\&+\binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))}\times(1-b_{b-2}(\mathrm{r.s.}))\end{aligned}$$

$$\begin{aligned}\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\}=\ \ \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\}\ &-\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))}\times b_{a-3}(\mathrm{r.s.})\\&-\binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))}\times b_{b-2}(\mathrm{r.s.})\end{aligned}$$

$\Delta a=1,\Delta b=1,\Delta c=1$:

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\}=\ \ \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\}\ &+\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))}\times(1-b_{a-3}(\mathrm{r.s.}))\\&+\binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))}\times(1-b_{b-2}(\mathrm{r.s.}))\\&-\binom{M(\mathrm{RASX})-c}{N(M^{\mathrm{left}}(c))}\times b_{c-1}(\mathrm{r.s.})\end{aligned}$$

$$\begin{aligned}\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\}=\ \ \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\}\ &-\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))}\times b_{a-3}(\mathrm{r.s.})\\&-\binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))}\times b_{b-2}(\mathrm{r.s.})\\&+\binom{M(\mathrm{RASX})-c}{N(M^{\mathrm{left}}(c))}\times(1-b_{c-1}(\mathrm{r.s.}))\end{aligned}$$

$\underline{\mathcal{P}_{11}^{01}\,(K_\gamma^{\mathrm{RASX}}:01;L_\gamma^{\mathrm{RASX}}:11)}$

$\Delta a=1,\Delta b=0$:

$$\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\}=\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\}-\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))}\times b_{a-2}(\mathrm{r.s.})$$

$$\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\}=\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\}+\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))}\times(1-b_{a-2}(\mathrm{r.s.}))$$

$\Delta a = 1, \Delta b = 1$:

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} \quad -\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-2}(\mathrm{r.s.}) \\ & +\binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times (1-b_{b-1}(\mathrm{r.s.}))\end{aligned}$$

$$\begin{aligned}\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} \quad +\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1-b_{a-2}(\mathrm{r.s.})) \\ & +\binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times (1-b_{b-1}(\mathrm{r.s.}))\end{aligned}$$

$\underline{\mathcal{P}_{101}^{010}\,(K_\gamma^{\mathrm{RASX}}:010; L_\gamma^{\mathrm{RASX}}:101)}$

$\Delta a = 1, \Delta b = 0, \Delta c = 0$:

$$\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} - \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-3}(\mathrm{r.s.})$$

$$\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1-b_{a-3}(\mathrm{r.s.}))$$

$\Delta a = 1, \Delta b = 1, \Delta c = 0$:

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} \quad -\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-3}(\mathrm{r.s.}) \\ & +\binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times (1-b_{b-2}(\mathrm{r.s.}))\end{aligned}$$

$$\begin{aligned}\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} \quad +\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1-b_{a-3}(\mathrm{r.s.})) \\ & -\binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times b_{b-2}(\mathrm{r.s.})\end{aligned}$$

$\Delta a = 1, \Delta b = 1, \Delta c = 1$:

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} \quad -\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-3}(\mathrm{r.s.}) \\ & +\binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times (1-b_{b-2}(\mathrm{r.s.})) \\ & -\binom{M(\mathrm{RASX})-c}{N(M^{\mathrm{left}}(c))} \times b_{c-1}(\mathrm{r.s.})\end{aligned}$$

$$\begin{aligned}\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} \\ & + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-3}(\mathrm{r.s.})) \\ & - \binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times b_{b-2}(\mathrm{r.s.}) \\ & + \binom{M(\mathrm{RASX})-c}{N(M^{\mathrm{left}}(c))} \times (1 - b_{c-1}(\mathrm{r.s.}))\end{aligned}$$

$\underline{\mathcal{P}_{110}^{001}\ (K_\gamma^{\mathrm{RASX}} : 001; L_\gamma^{\mathrm{RASX}} : 110)}$

$\Delta a = 1, \Delta b = 0, \Delta c = 0$:

$$\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} - \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-3}(\mathrm{r.s.})$$

$$\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-3}(\mathrm{r.s.}))$$

$\Delta a = 1, \Delta b = 1, \Delta c = 0$:

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} \\ & - \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-3}(\mathrm{r.s.}) \\ & - \binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times b_{b-2}(\mathrm{r.s.})\end{aligned}$$

$$\begin{aligned}\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} \\ & + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-3}(\mathrm{r.s.})) \\ & + \binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times (1 - b_{b-2}(\mathrm{r.s.}))\end{aligned}$$

$\Delta a = 1, \Delta b = 1, \Delta c = 1$:

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} \\ & - \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-3}(\mathrm{r.s.}) \\ & - \binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times b_{b-2}(\mathrm{r.s.}) \\ & + \binom{M(\mathrm{RASX})-c}{N(M^{\mathrm{left}}(c))} \times (1 - b_{c-1}(\mathrm{r.s.}))\end{aligned}$$

$$\begin{aligned}\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} \\ & + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-3}(\mathrm{r.s.})) \\ & + \binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times (1 - b_{b-2}(\mathrm{r.s.})) \\ & - \binom{M(\mathrm{RASX})-c}{N(M^{\mathrm{left}}(c))} \times b_{c-1}(\mathrm{r.s.})\end{aligned}$$

$\underline{\mathcal{P}_{101}^{110}\ (K_\gamma^{\mathrm{RASX}} : 110; L_\gamma^{\mathrm{RASX}} : 101)}$

B. *Propagation Rules*

$\Delta a = 1, \Delta b = 0, \Delta c = 0$:

$$\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-3}(\mathrm{r.s.}))$$

$$\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-3}(\mathrm{r.s.}))$$

$\Delta a = 1, \Delta b = 1, \Delta c = 0$:

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \quad \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} \quad &+ \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-3}(\mathrm{r.s.})) \\ &+ \binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times (1 - b_{b-2}(\mathrm{r.s.}))\end{aligned}$$

$$\begin{aligned}\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \quad \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} \quad &+ \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-3}(\mathrm{r.s.})) \\ &- \binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times b_{b-2}(\mathrm{r.s.})\end{aligned}$$

$\Delta a = 1, \Delta b = 1, \Delta c = 1$:

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \quad \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} \quad &+ \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-3}(\mathrm{r.s.})) \\ &+ \binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times (1 - b_{b-2}(\mathrm{r.s.})) \\ &- \binom{M(\mathrm{RASX})-c}{N(M^{\mathrm{left}}(c))} \times b_{c-1}(\mathrm{r.s.})\end{aligned}$$

$$\begin{aligned}\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \quad \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} \quad &+ \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-3}(\mathrm{r.s.})) \\ &- \binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times b_{b-2}(\mathrm{r.s.}) \\ &+ \binom{M(\mathrm{RASX})-c}{N(M^{\mathrm{left}}(c))} \times (1 - b_{c-1}(\mathrm{r.s.}))\end{aligned}$$

$\underline{\mathcal{P}_{110}^{101}\,(K_\gamma^{\mathrm{RASX}} : 101; L_\gamma^{\mathrm{RASX}} : 110)}$

$\Delta a = 1, \Delta b = 0, \Delta c = 0$:

$$\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-3}(\mathrm{r.s.}))$$

$$\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-3}(\mathrm{r.s.}))$$

$\Delta a = 1, \Delta b = 1, \Delta c = 0$:

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} \quad + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-3}(\mathrm{r.s.})) \\ & - \binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times b_{b-2}(\mathrm{r.s.})\end{aligned}$$

$$\begin{aligned}\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} \quad + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-3}(\mathrm{r.s.})) \\ & + \binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times (1 - b_{b-2}(\mathrm{r.s.}))\end{aligned}$$

$\Delta a = 1, \Delta b = 1, \Delta c = 1$:

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} \quad + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-3}(\mathrm{r.s.})) \\ & - \binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times b_{b-2}(\mathrm{r.s.}) \\ & + \binom{M(\mathrm{RASX})-c}{N(M^{\mathrm{left}}(c))} \times (1 - b_{c-1}(\mathrm{r.s.}))\end{aligned}$$

$$\begin{aligned}\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} \quad + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-3}(\mathrm{r.s.})) \\ & + \binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times (1 - b_{b-2}(\mathrm{r.s.})) \\ & - \binom{M(\mathrm{RASX})-c}{N(M^{\mathrm{left}}(c))} \times b_{c-1}(\mathrm{r.s.})\end{aligned}$$

$\underline{\mathcal{P}_{011}^{101}\,(K_\gamma^{\mathrm{RASX}} : 101; L_\gamma^{\mathrm{RASX}} : 011)}$

$\Delta a = 1, \Delta b = 0, \Delta c = 0$:

$$\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-3}(\mathrm{r.s.}))$$

$$\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} - \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-3}(\mathrm{r.s.})$$

$\Delta a = 1, \Delta b = 1, \Delta c = 0$:

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} \quad + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-3}(\mathrm{r.s.})) \\ & - \binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times b_{b-2}(\mathrm{r.s.})\end{aligned}$$

B. Propagation Rules

$$\begin{aligned}\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} \quad -\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-3}(\mathrm{r.s.}) \\ & +\binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times (1-b_{b-2}(\mathrm{r.s.}))\end{aligned}$$

$\Delta a = 1, \Delta b = 1, \Delta c = 1$:

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} \quad +\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1-b_{a-3}(\mathrm{r.s.})) \\ & -\binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times b_{b-2}(\mathrm{r.s.}) \\ & +\binom{M(\mathrm{RASX})-c}{N(M^{\mathrm{left}}(c))} \times (1-b_{c-1}(\mathrm{r.s.}))\end{aligned}$$

$$\begin{aligned}\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} \quad -\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-3}(\mathrm{r.s.}) \\ & +\binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times (1-b_{b-2}(\mathrm{r.s.})) \\ & +\binom{M(\mathrm{RASX})-c}{N(M^{\mathrm{left}}(c))} \times (1-b_{c-1}(\mathrm{r.s.}))\end{aligned}$$

$\underline{\mathcal{P}_{011}^{110}\,(K_\gamma^{\mathrm{RASX}}:110; L_\gamma^{\mathrm{RASX}}:011)}$

$\Delta a = 1, \Delta b = 0, \Delta c = 0$:

$$\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1-b_{a-3}(\mathrm{r.s.}))$$

$$\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} - \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-3}(\mathrm{r.s.})$$

$\Delta a = 1, \Delta b = 1, \Delta c = 0$:

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} \quad +\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1-b_{a-3}(\mathrm{r.s.})) \\ & +\binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times (1-b_{b-2}(\mathrm{r.s.}))\end{aligned}$$

$$\begin{aligned}\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} \quad -\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-3}(\mathrm{r.s.}) \\ & +\binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times (1-b_{b-2}(\mathrm{r.s.}))\end{aligned}$$

B. Propagation Rules

$\Delta a = 1, \Delta b = 1, \Delta c = 1$:

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} \quad + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-3}(\mathrm{r.s.})) \\ & + \binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times (1 - b_{b-2}(\mathrm{r.s.})) \\ & - \binom{M(\mathrm{RASX})-c}{N(M^{\mathrm{left}}(c))} \times b_{c-1}(\mathrm{r.s.})\end{aligned}$$

$$\begin{aligned}\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} \quad - \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-3}(\mathrm{r.s.}) \\ & + \binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times (1 - b_{b-2}(\mathrm{r.s.})) \\ & + \binom{M(\mathrm{RASX})-c}{N(M^{\mathrm{left}}(c))} \times (1 - b_{c-1}(\mathrm{r.s.}))\end{aligned}$$

$\underline{\mathcal{P}_{0101}^{1010}\,(K_\gamma^{\mathrm{RASX}} : 1010; L_\gamma^{\mathrm{RASX}} : 0101)}$

$\Delta a = 1, \Delta b = 0, \Delta c = 0, \Delta d = 0$:

$$\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-4}(\mathrm{r.s.}))$$

$$\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} - \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-4}(\mathrm{r.s.})$$

$\Delta a = 1, \Delta b = 1, \Delta c = 0, \Delta d = 0$:

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} \quad + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-4}(\mathrm{r.s.})) \\ & - \binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times b_{b-3}(\mathrm{r.s.})\end{aligned}$$

$$\begin{aligned}\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} \quad - \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-4}(\mathrm{r.s.}) \\ & + \binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times (1 - b_{b-3}(\mathrm{r.s.}))\end{aligned}$$

$\Delta a = 1, \Delta b = 1, \Delta c = 1, \Delta d = 0$:

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} \quad + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-4}(\mathrm{r.s.})) \\ & - \binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times b_{b-3}(\mathrm{r.s.}) \\ & + \binom{M(\mathrm{RASX})-c}{N(M^{\mathrm{left}}(c))} \times (1 - b_{c-2}(\mathrm{r.s.}))\end{aligned}$$

B. *Propagation Rules*

$$\begin{aligned}\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} \quad -\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-4}(\mathrm{r.s.}) \\ & +\binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times (1-b_{b-3}(\mathrm{r.s.})) \\ & -\binom{M(\mathrm{RASX})-c}{N(M^{\mathrm{left}}(c))} \times b_{c-2}(\mathrm{r.s.})\end{aligned}$$

$\Delta a = 1, \Delta b = 1, \Delta c = 1, \Delta d = 0$:

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} \quad +\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1-b_{a-4}(\mathrm{r.s.})) \\ & -\binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times b_{b-3}(\mathrm{r.s.}) \\ & +\binom{M(\mathrm{RASX})-c}{N(M^{\mathrm{left}}(c))} \times (1-b_{c-2}(\mathrm{r.s.})) \\ & -\binom{M(\mathrm{RASX})-c}{N(M^{\mathrm{left}}(c))} \times b_{d-1}(\mathrm{r.s.})\end{aligned}$$

$$\begin{aligned}\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} \quad -\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-4}(\mathrm{r.s.}) \\ & +\binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times (1-b_{b-3}(\mathrm{r.s.})) \\ & -\binom{M(\mathrm{RASX})-c}{N(M^{\mathrm{left}}(c))} \times b_{c-2}(\mathrm{r.s.}) \\ & +\binom{M(\mathrm{RASX})-c}{N(M^{\mathrm{left}}(c))} \times (1-b_{d-1}(\mathrm{r.s.}))\end{aligned}$$

$\underline{\mathcal{P}_{0110}^{1001}\ (K_\gamma^{\mathrm{RASX}} : 1001; L_\gamma^{\mathrm{RASX}} : 0110)}$

$\Delta a = 1, \Delta b = 0, \Delta c = 0, \Delta d = 0$:

$$\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1-b_{a-4}(\mathrm{r.s.}))$$

$$\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} - \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-4}(\mathrm{r.s.})$$

$\Delta a = 1, \Delta b = 1, \Delta c = 0, \Delta d = 0$:

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} \quad +\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1-b_{a-4}(\mathrm{r.s.})) \\ & -\binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times b_{b-3}(\mathrm{r.s.})\end{aligned}$$

$$\begin{aligned}\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} \quad -\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-4}(\mathrm{r.s.}) \\ & +\binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times (1-b_{b-3}(\mathrm{r.s.}))\end{aligned}$$

$\Delta a = 1, \Delta b = 1, \Delta c = 1, \Delta d = 0$:

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} \quad + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-4}(\mathrm{r.s.})) \\ & - \binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times b_{b-3}(\mathrm{r.s.}) \\ & - \binom{M(\mathrm{RASX})-c}{N(M^{\mathrm{left}}(c))} \times b_{c-2}(\mathrm{r.s.})\end{aligned}$$

$$\begin{aligned}\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} \quad - \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-4}(\mathrm{r.s.}) \\ & + \binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times (1 - b_{b-3}(\mathrm{r.s.})) \\ & + \binom{M(\mathrm{RASX})-c}{N(M^{\mathrm{left}}(c))} \times (1 - b_{c-2}(\mathrm{r.s.}))\end{aligned}$$

$\Delta a = 1, \Delta b = 1, \Delta c = 1, \Delta d = 0$:

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} \quad + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-4}(\mathrm{r.s.})) \\ & - \binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times b_{b-3}(\mathrm{r.s.}) \\ & - \binom{M(\mathrm{RASX})-c}{N(M^{\mathrm{left}}(c))} \times b_{c-2}(\mathrm{r.s.}) \\ & + \binom{M(\mathrm{RASX})-c}{N(M^{\mathrm{left}}(c))} \times (1 - b_{d-1}(\mathrm{r.s.}))\end{aligned}$$

$$\begin{aligned}\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} \quad - \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-4}(\mathrm{r.s.}) \\ & + \binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times (1 - b_{b-3}(\mathrm{r.s.})) \\ & + \binom{M(\mathrm{RASX})-c}{N(M^{\mathrm{left}}(c))} \times (1 - b_{c-2}(\mathrm{r.s.})) \\ & - \binom{M(\mathrm{RASX})-c}{N(M^{\mathrm{left}}(c))} \times b_{d-1}(\mathrm{r.s.})\end{aligned}$$

$\underline{\mathcal{P}_{0011}^{1100}\,(K_\gamma^{\mathrm{RASX}} : 1100; L_\gamma^{\mathrm{RASX}} : 0011)}$

$\Delta a = 1, \Delta b = 0, \Delta c = 0, \Delta d = 0$:

$$\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-4}(\mathrm{r.s.}))$$

$$\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} - \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-4}(\mathrm{r.s.})$$

B. *Propagation Rules*

$\Delta a = 1, \Delta b = 1, \Delta c = 0, \Delta d = 0$:

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} \\ & + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-4}(\mathrm{r.s.})) \\ & + \binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times (1 - b_{b-3}(\mathrm{r.s.}))\end{aligned}$$

$$\begin{aligned}\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} \\ & - \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-4}(\mathrm{r.s.}) \\ & - \binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times b_{b-3}(\mathrm{r.s.})\end{aligned}$$

$\Delta a = 1, \Delta b = 1, \Delta c = 1, \Delta d = 0$:

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} \\ & + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-4}(\mathrm{r.s.})) \\ & + \binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times (1 - b_{b-3}(\mathrm{r.s.})) \\ & - \binom{M(\mathrm{RASX})-c}{N(M^{\mathrm{left}}(c))} \times b_{c-2}(\mathrm{r.s.})\end{aligned}$$

$$\begin{aligned}\mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{L_\gamma^{\mathrm{RASX}}(n-1)\} \\ & - \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-4}(\mathrm{r.s.}) \\ & - \binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times b_{b-3}(\mathrm{r.s.}) \\ & + \binom{M(\mathrm{RASX})-c}{N(M^{\mathrm{left}}(c))} \times (1 - b_{c-2}(\mathrm{r.s.}))\end{aligned}$$

$\Delta a = 1, \Delta b = 1, \Delta c = 1, \Delta d = 0$:

$$\begin{aligned}\mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n)\} = \quad & \mathrm{Addr}\{K_\gamma^{\mathrm{RASX}}(n-1)\} \\ & + \binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times (1 - b_{a-4}(\mathrm{r.s.})) \\ & + \binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times (1 - b_{b-3}(\mathrm{r.s.})) \\ & - \binom{M(\mathrm{RASX})-c}{N(M^{\mathrm{left}}(c))} \times b_{c-2}(\mathrm{r.s.}) \\ & - \binom{M(\mathrm{RASX})-c}{N(M^{\mathrm{left}}(c))} \times b_{d-1}(\mathrm{r.s.})\end{aligned}$$

B. *Propagation Rules*

$$
\begin{aligned}
\mathrm{Addr}\{L_Y^{\mathrm{RASX}}(n)\} = \quad \mathrm{Addr}\{L_Y^{\mathrm{RASX}}(n-1)\} \quad & -\binom{M(\mathrm{RASX})-a}{N(M^{\mathrm{left}}(a))} \times b_{a-4}(\mathrm{r.s.}) \\
& -\binom{M(\mathrm{RASX})-b}{N(M^{\mathrm{left}}(b))} \times b_{b-3}(\mathrm{r.s.}) \\
& +\binom{M(\mathrm{RASX})-c}{N(M^{\mathrm{left}}(c))} \times (1-b_{c-2}(\mathrm{r.s.})) \\
& +\binom{M(\mathrm{RASX})-c}{N(M^{\mathrm{left}}(c))} \times (1-b_{d-1}(\mathrm{r.s.}))
\end{aligned}
$$

C. RASSCF: Implementation Details

We have implemented the propagation rule RASSCF algorithm as described in chapter 3, as part of the existing MCSCF program (link l510) of the Gaussian package of programs. We have implemented both a serial and a parallel version of the program. Figures C.1 and C.2 show the subroutine-tree for serial and parallel execution. A brief description of the routines is added.

We also implemented the RASSCF version of other affected parts of the MCSCF program, namely those computing the density matrix, the diagonal of the Hamiltonian matrix, and the reduced Hamiltonian used within the Lanczos algorithm. The internal structure of these parts resembles that of the CI vector updating algorithm, including an outer loop over model space indices w,x,y,z. Figures C.10-C.15 show the subroutine trees for serial and parallel (Linda) implementations.

FlyRas/MCAXMR Top level routine. Perform memory allocation. Call subroutine UpdRas.

UpdRas Compute all $L[\mathrm{Cat}(i_h,i_e]$ and $L[\mathrm{Cat}(i'_h,i'_e)]$ for all categories. Compute (1st address -1) of all permissible combinations of categories. Compute reduced string lists for model spaces. Build Handy's index array $\mathbf{IIZ}(IIZ\_P(M,N))$. Call subroutine RasWor.

RasWor (Parallel) loop over 1e tasks. Assemble task list. Call subroutine RJob1. (Parallel) loop over 2e tasks. Assemble task list. Call subroutine RJob2.

I2wx Check if task corresponding to model space indices w,x is valid.

RJob1 Loop over task list. Get corresponding index pairs w,x and i,j. Get corresponding integral list. Call subroutine RJobAA.

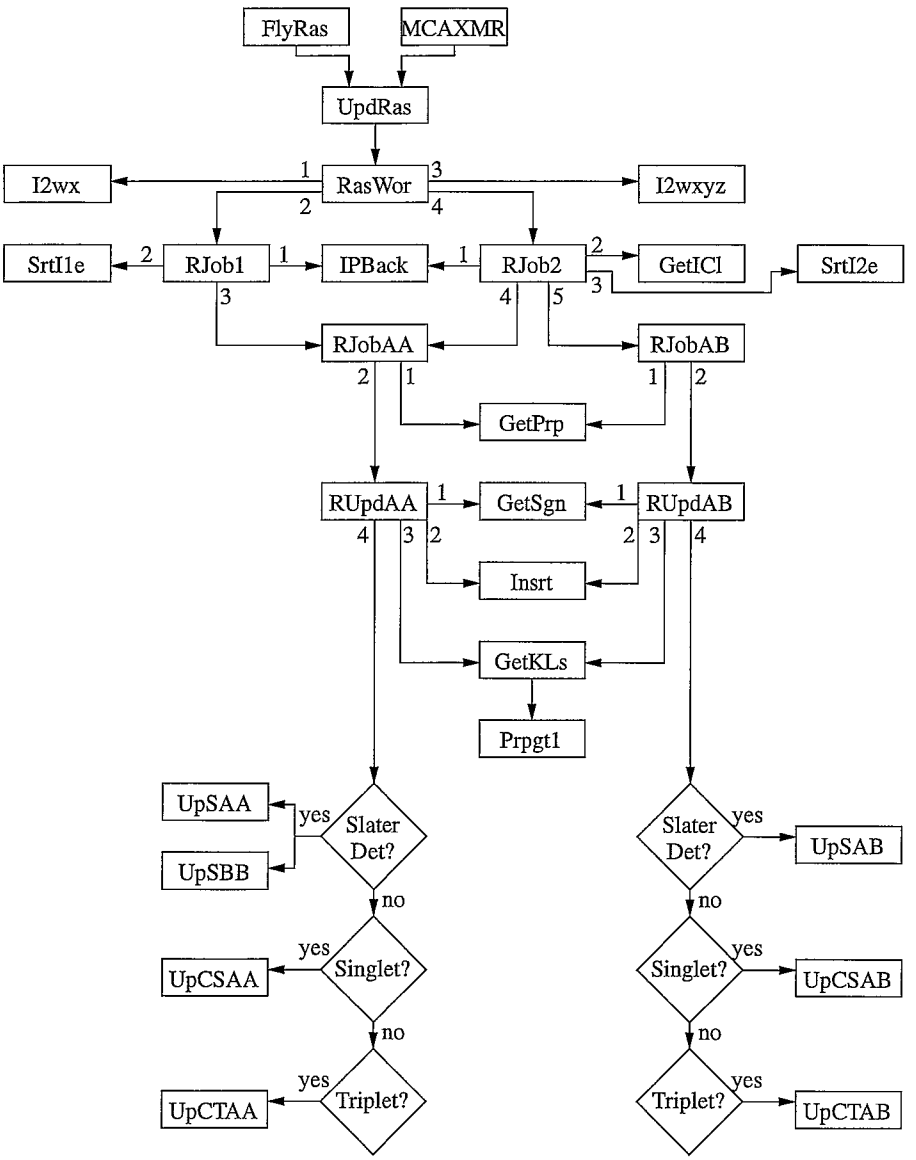


Figure C.1.: Subroutine calling sequence for CI vector updating algorithm.

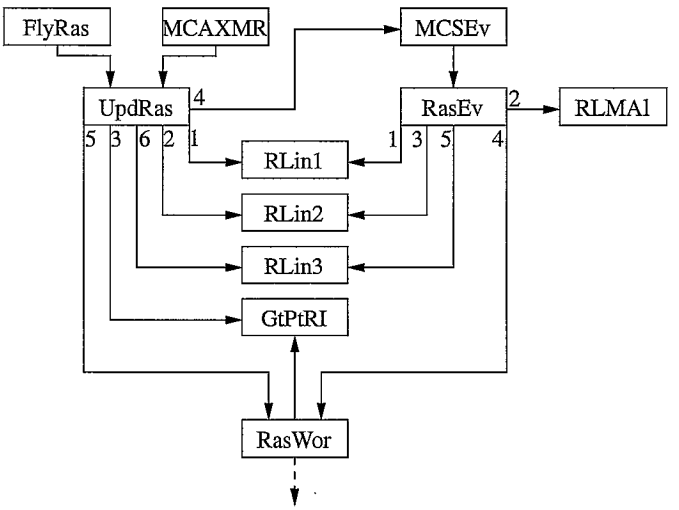


Figure C.2.: Linda communication for CI vector updating algorithm.

IPBack Map task index to model space index pair w,x.

SrtI1e Assemble 1e integral list corresponding (first element is $(i|j)$).

RJobAA Get propagation rule. Decide on 1e or 2e mode. Call subroutine RUpdAA.

GetPrp Compute propagation rules for RAS1 and RAS3.

RUpdAA See figure C.3.

GetSgn Compute sign factors corresponding to propagators in RAS1 and RAS3 subspaces.

Insrt Insert bits into bit-string.

INumGt Compute sgn_0^m.

GetKLs See figure C.5.

Prpgt1 See figure C.6.

UpSAA/UpSBB (or UpCSAA, or UpCTAA) See figure C.7.

I2wxyz Check if task corresponding to model space indices w,x,y,z is valid.

RJob2 Loop over task list. Get corresponding index quadruple w,x,y,z and i,j,k,l. Get integral class number. Get integral list. If $\alpha\alpha$-contribution then Call subroutine RJobAA. Call subroutine RJobAB.

GetICI Compute integral class.

SrtI2e Assemble 2e integral list (first element is $(ij|kl)$).

RJobAB Get propagation rule. Call subroutine RUpdAB.

RUpdAB See figure C.4.

UpSAB (or UpCSAB, or UpCTAB) See figures C.8 and C.9.

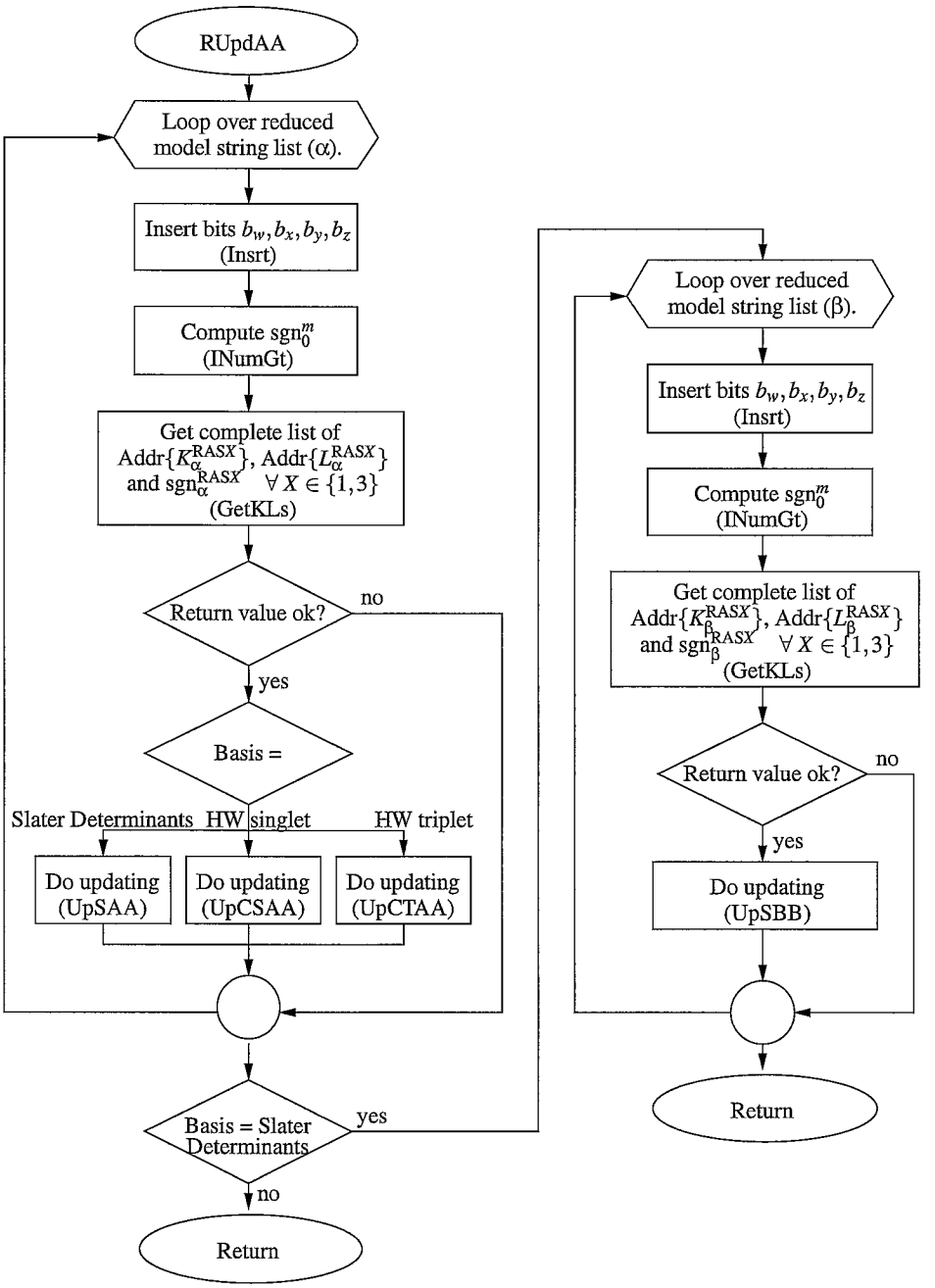


Figure C.3.: The subroutine RUpdAA completes the reduced α-model strings, requests the list of RAS1 and RAS3 subspace excitation lists and calls the correct αα updating routine corresponding to the used basis functions. If the basis are Slater determinants the procedure is repeated for β-model strings for ββ updating.

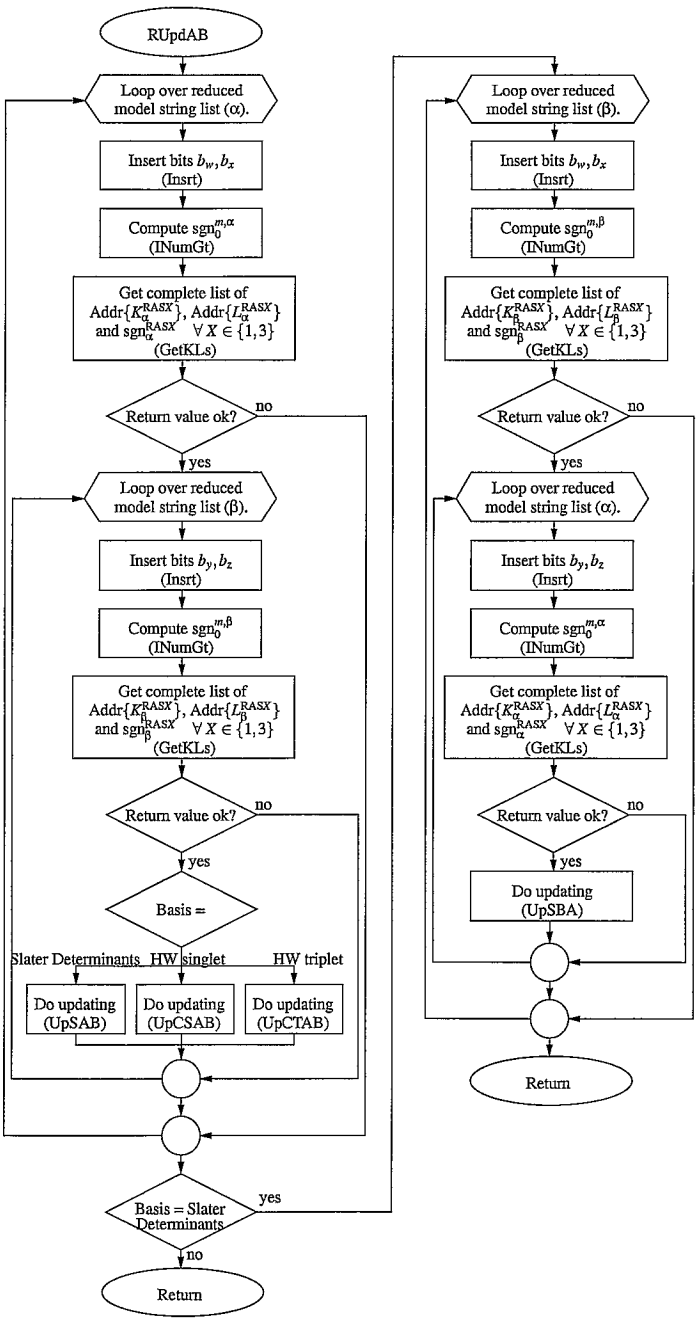


Figure C.4.: The subroutine RUpdAB completes the reduced α-model strings, requests the list of RAS1 and RAS3 subspace excitation lists and then does the same for β-model strings (nested loop). It then calls the correct αβ updating routine corresponding to the used basis functions. If the basis are Slater determinants the procedure is repeated for βα updating.

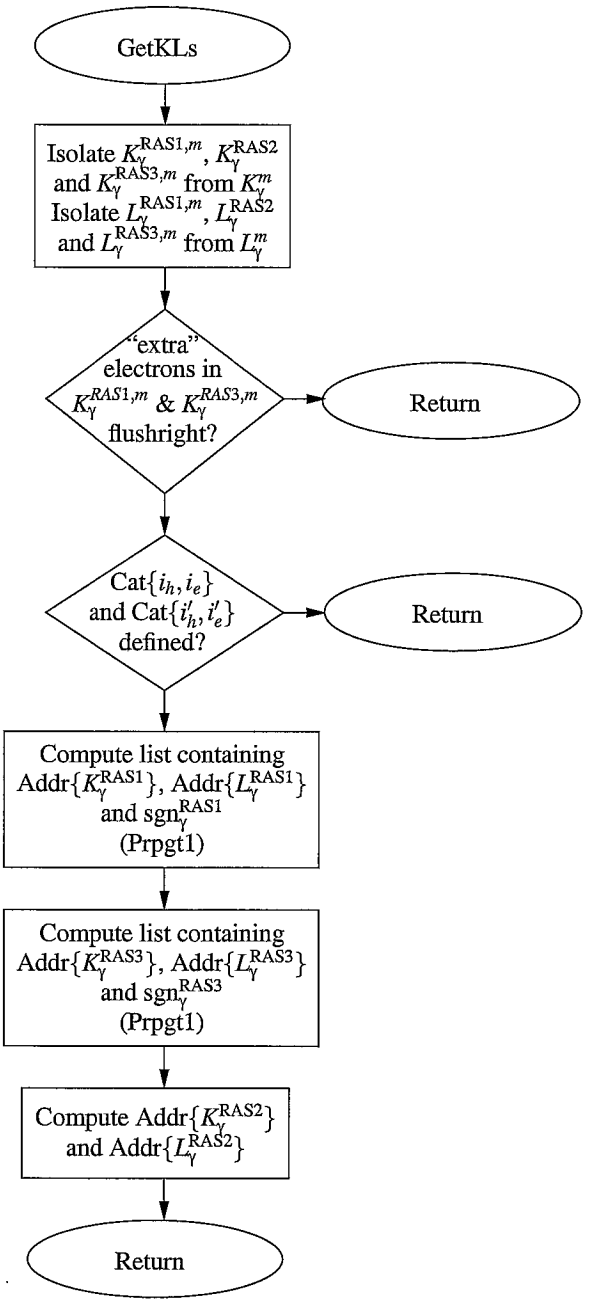


Figure C.5.: The subroutine GetKLs is called by RUpdAA and RUpdAB and provides the RAS1 and RAS3 subspace excitation lists. Before proceeding with excitation list assembly it performs validity tests for the model strings and their respective categories,Cat$\{i_h, i_e\}$. The routine is transparent with respect to spin.

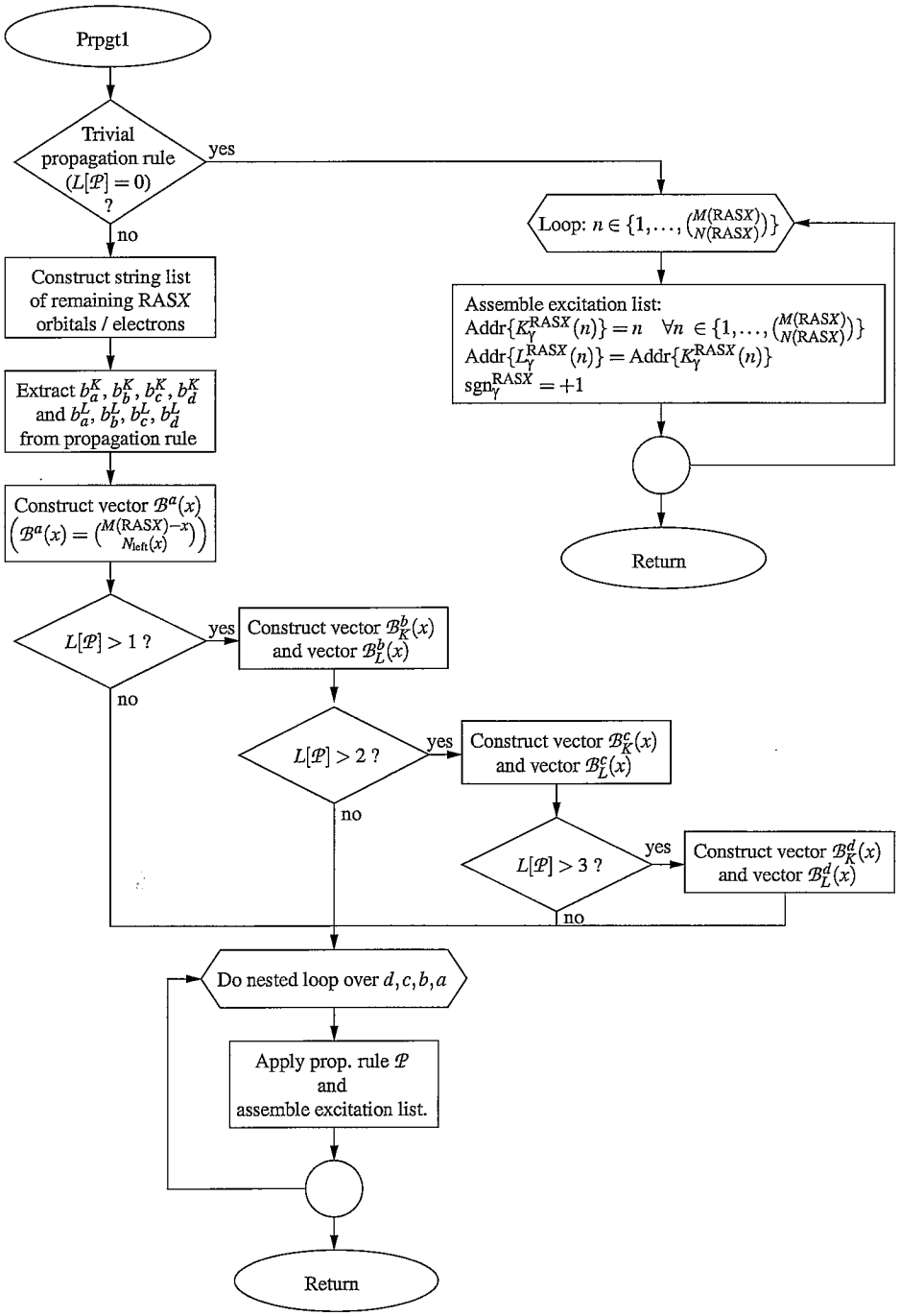


Figure C.6.: The subroutine Prpgt1 applies the propagation rule, $\mathcal{P}$, to the appropriate RAS substring. If the propagation rule is not trivial, it precomputes the binomial vectors $\mathcal{B}^a$ etc. and then assembles the excitation list within a nested loop over the variable orbital indices, a, b, c, d.

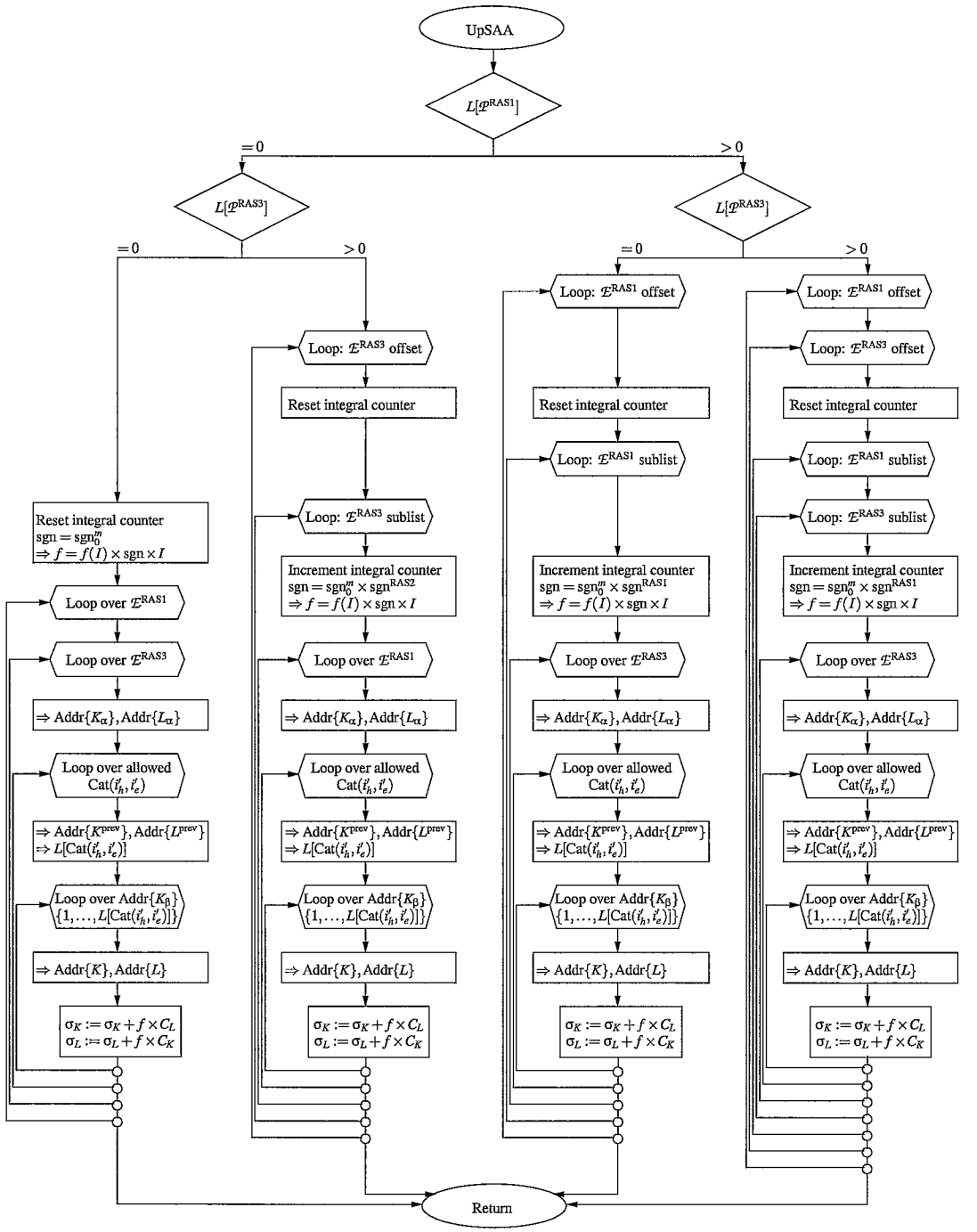


Figure C.7.: The subroutine UpSAA does the actual CI vector updating ($\alpha\alpha$ part).

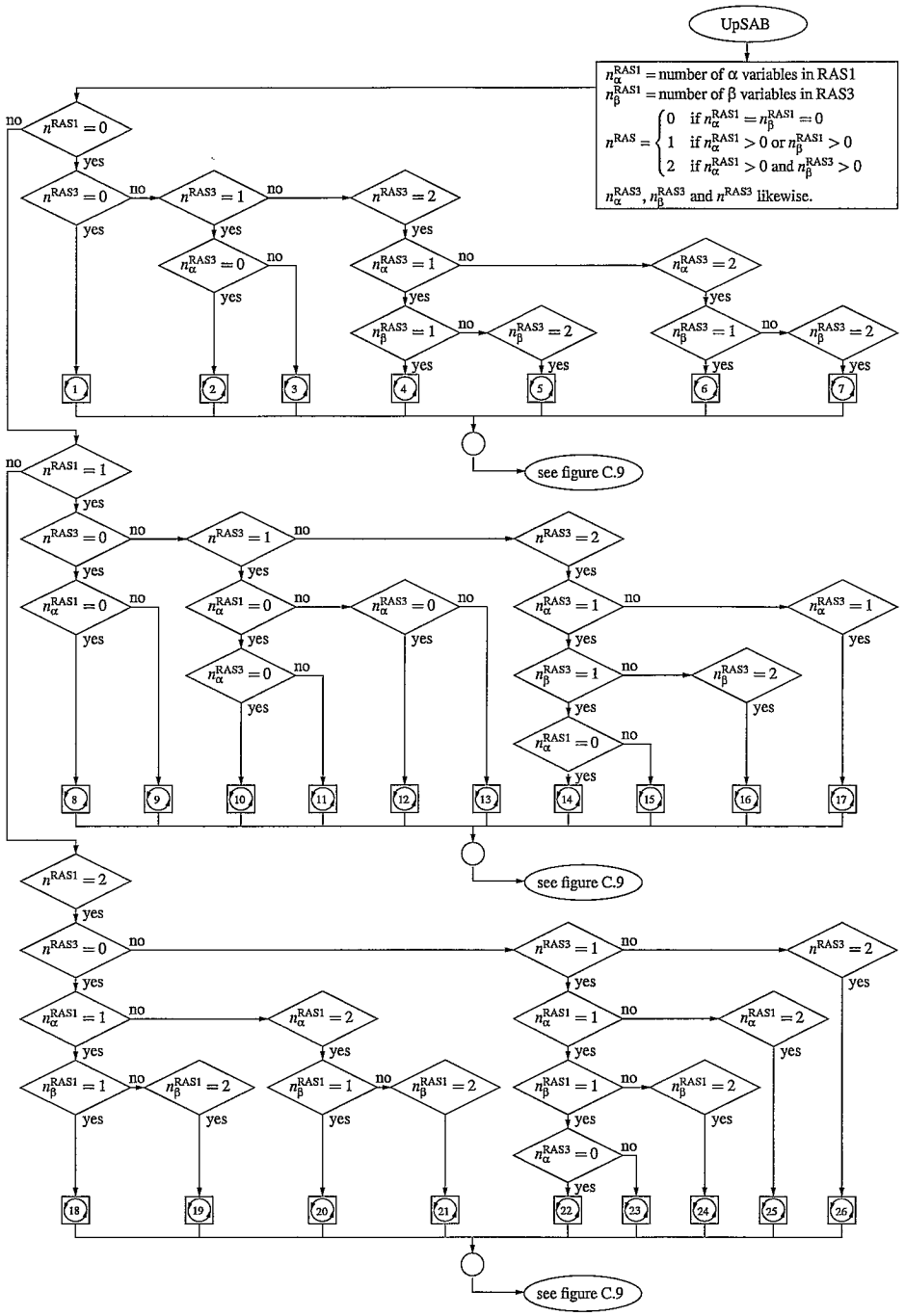


Figure C.8.: The subroutine UpSAB does the actual CI vector updating ($\alpha\beta$ part). In order to avoid slow logic within the nested loops over excitation strings, 26 cases are distinguished first on the basis of the number of variable indices, a, b, c, d, in each RAS substring. The flowchart is continued in figure C.9.

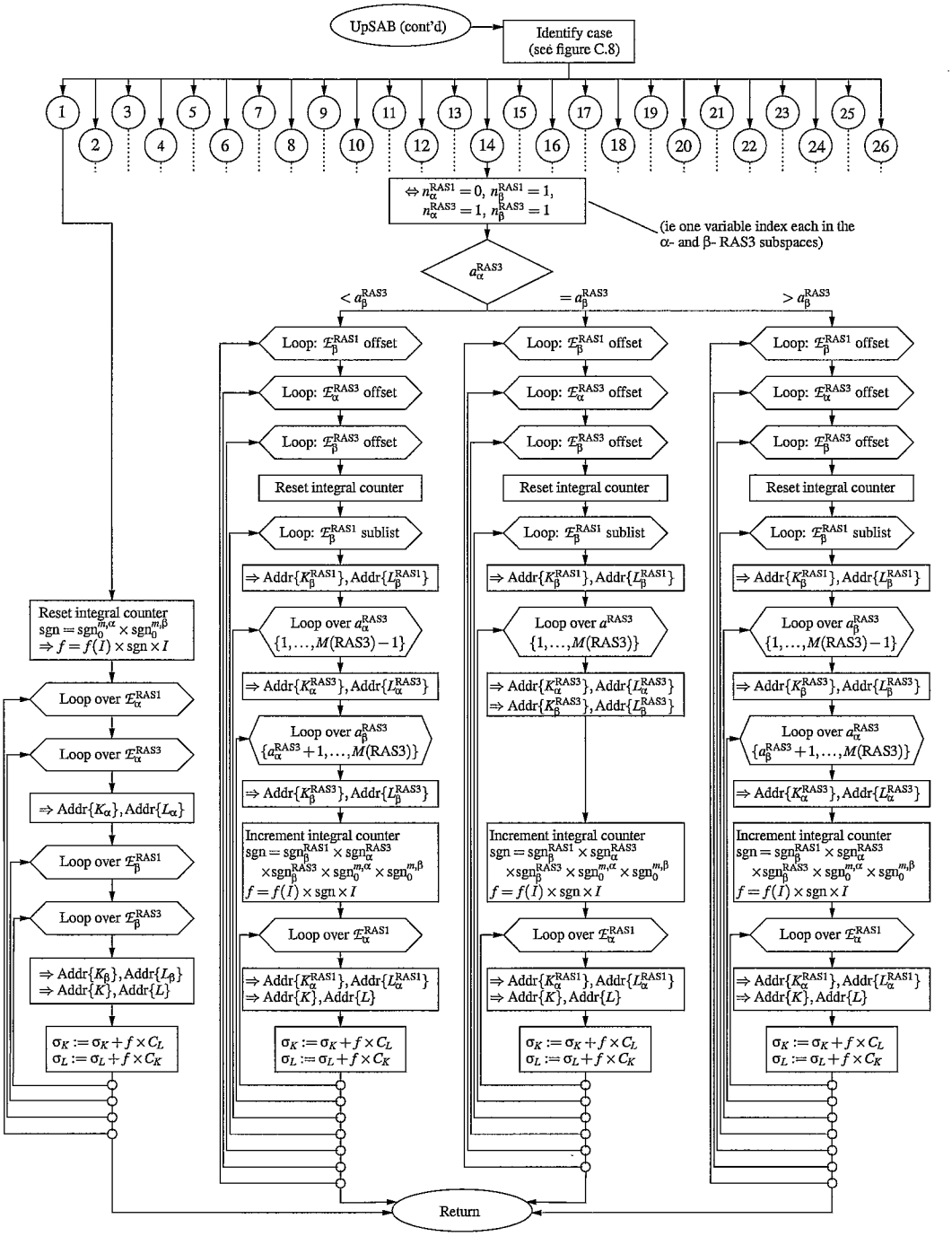


Figure C.9.: Continued from C.8. Two of the 26 cases are shown. Case 1 is the most common and the simplest case.

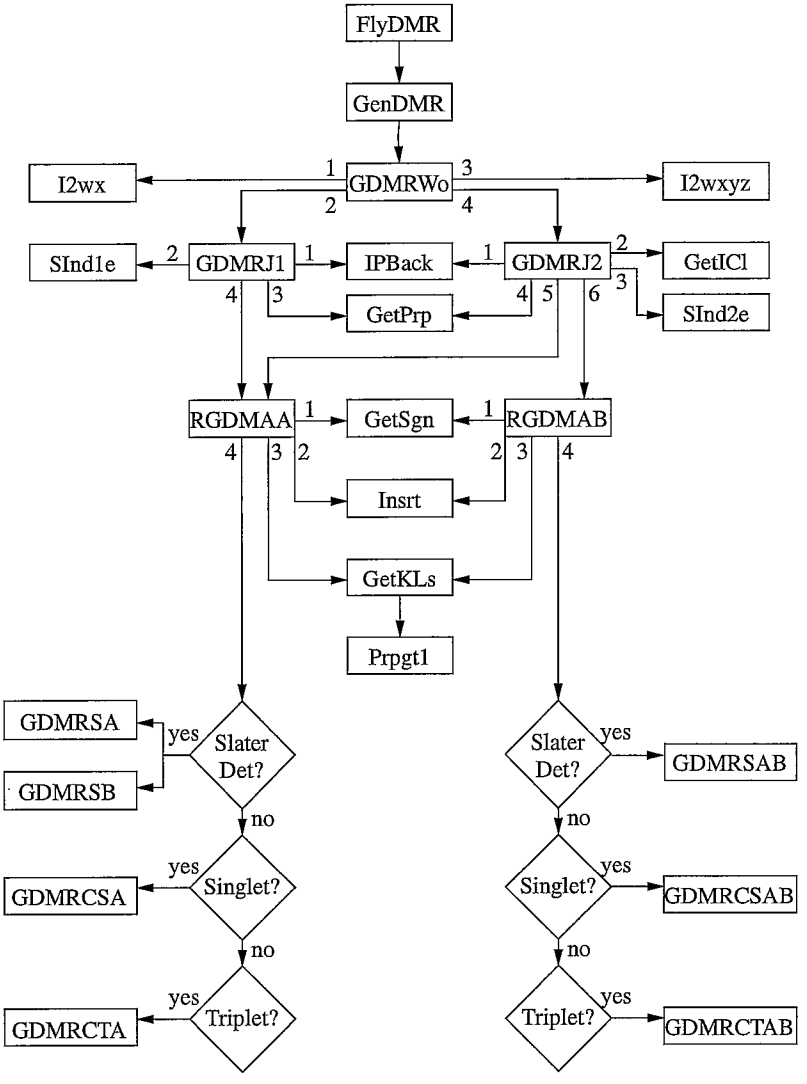


Figure C.10.: Subroutine calling sequence for density matrix algorithm.

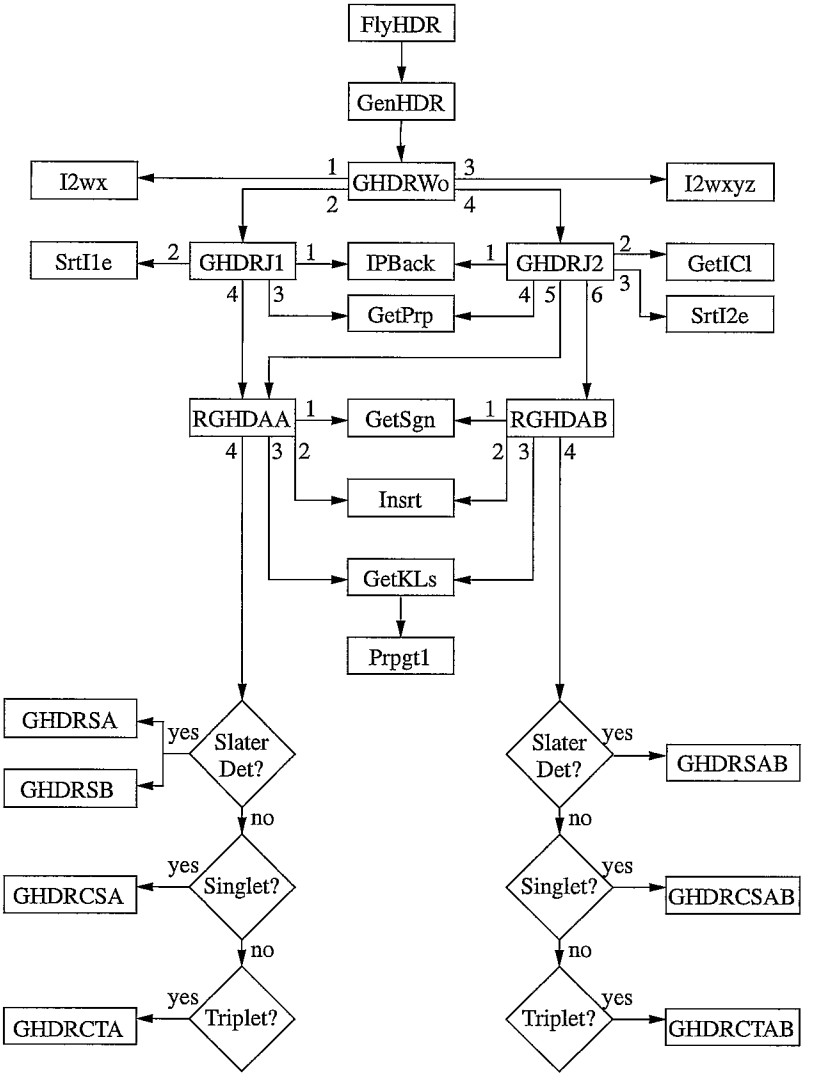


Figure C.11.: Subroutine calling sequence for diagonal Hamiltonian algorithm.

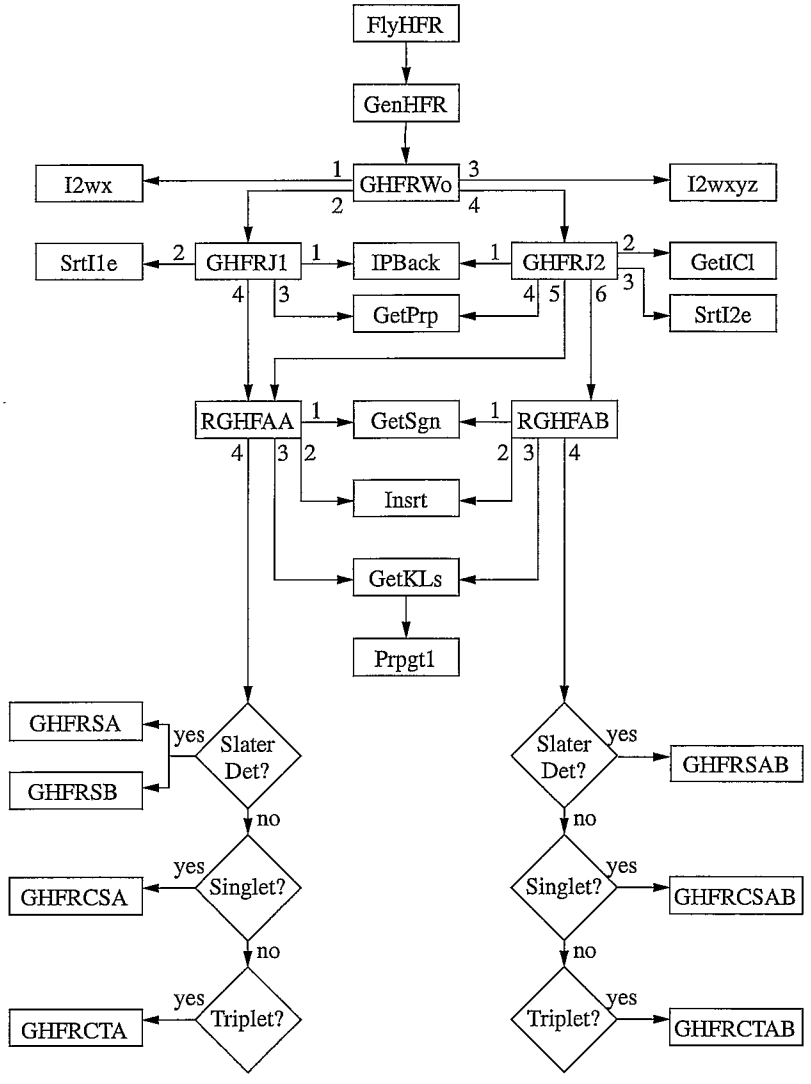


Figure C.12.: Subroutine calling sequence for reduced Hamiltonian algorithm.

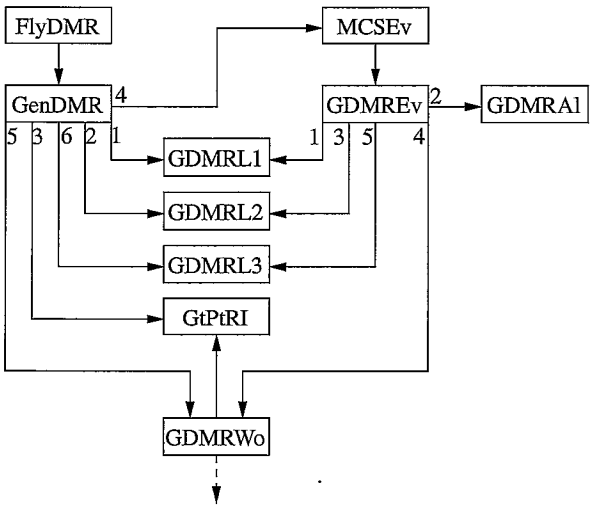


Figure C.13.: Linda communication for density matrix algorithm.

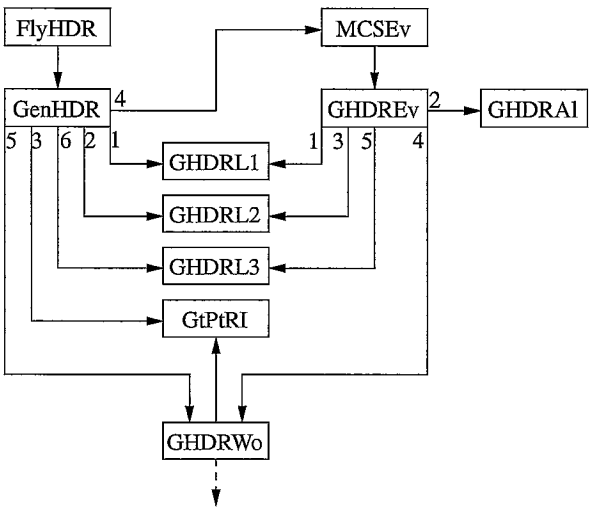


Figure C.14.: Linda communication for diagonal Hamiltonian algorithm.

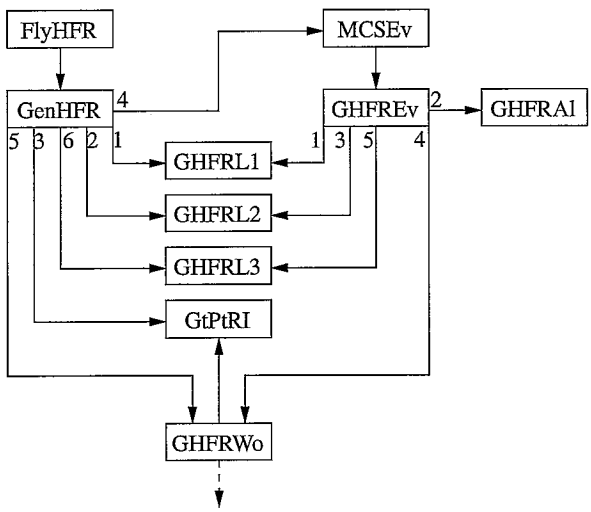


Figure C.15.: Linda communication for reduced Hamiltonian algorithm.

Bibliography

[1] A remark attributed to P. A. M. Dirac in 1929, as quoted for example in [17].

[2] M. Head-Gordon. Quantum chemistry and molecular processes. *J. Phys. Chem.*, 100(31):13213–13225, 1996.

[3] C. Møller and M. S. Plesset. Note on an approximation treatment for many-electron systems. *Phys. Rev.*, 46:618–622, 1934.

[4] P. Hohenberg and W. Kohn. Inhomogeneous electron gas. *Phys. Rev.*, 136(3B):B864–B871, 1964.

[5] W. Kohn and L. J. Sham. Self-consistent equations including exchange and correlation effects. *Phys. Rev.*, 140(4A):A1133–A1138, 1965.

[6] M. E. Casida, K. C. Casida, and D. R. Salahub. Excited-state potential energy curves from time-dependent density-functional theory: a cross section of formaldehyde's $^1A_1$ manifold. *Int. J. Quantum Chem.*, 70(4/5):933–941, 1998.

[7] K. Burke and E. K. U. Gross. *Density Functionals: Theory and Applications*, chapter A guided tour of time-dependent density functional theory, pages 116–146. Springer, Berlin, Germany, 1998.

[8] I. Shavitt. The history and evolution of configuration interaction. *Mol. Phys.*, 94:3–17, 1998.

[9] J. A. Pople, M. Head-Gordon, and K. Raghavchari. Quadratic configuration interaction. A general technique for determining electron correlation energies. *J. Chem. Phys.*, 87(10):5968–5975, 1987.

[10] H. Koch, O. Christiansen, R. Kobayashi, P. Jørgensen, and T. Helgaker. A direct atomic orbital driven implementation of the coupled cluster singles and doubles (CCSD) model. *Chem. Phys. Lett.*, 228(1–3):233–238, 1994.

[11] K. Raghavachari and J. B. Anderson. Electron correlation effects in molecules. *J. Phys. Chem.*, 100:12960–12973, 1996.

[12] P. J. Knowles, M. Schütz, and H. J. Werner. Ab initio methods for electron correlation in molecules. In J. Grotendorst, editor, *Modern Methods and Algorithms of Quantum Chemistry*, volume 3 of *NIC Series*, pages 97–179. John von Neumann Institute for Computing, Jülich, 2000.

[13] B. O. Roos, P. R. Taylor, and P. E. M. Siegbahn. A complete active space SCF method (CASSCF) using a density matrix formulated super-CI approach. *Chem. Phys.*, 48:157–173, 1980.

[14] J. Olsen, B. O. Roos, P. Jørgensen, and H. J. A. Jensen. Determinant based configuration-interaction algorithms for complete and restricted configuration-interaction spaces. *J. Chem. Phys.*, 89:2185–2192, 1988.

[15] A. Szabo and N. S. Ostlund. *Modern quantum chemistry*. McGraw-Hill, 1 edition, 1989.

[16] R. McWeeny. *Methods of molecular quantum mechanics*. Academic Press, London, UK, 2 edition, 1989.

[17] I. N. Levine. *Quantum chemistry*. Prentice-Hall, Englewood Cliffs, NJ, 4 edition, 1991.

[18] R. McWeeny and B. T. Sutcliffe. Fundamentals of self-consistent-field (SCF), Hartree-Fock (HF), multi-configuration (MC) SCF and configuration-interaction (CI) schemes. *Comput Phys Rep*, 2:217–278, 1985.

[19] B. O. Roos. The complete active space selfconsistent field method and its applications in electronic structure calculations. In K. P. Lawley, editor, *Ab initio methods in quantum chemistry, Part II*, Adv. Chem. Phys., pages 399–445. Wiley, Chichester, UK, 1987.

[20] W. T. Borden and E. R. Davidson. The importance of including dynamic electron correlation in ab initio calculations. *Acc. Chem. Res.*, 29(2):67–75, 1996.

[21] A. I. Panin and K. V. Simon. Configuration interaction spaces with arbitrary restrictions on orbital occupancies. *Int. J. Quantum Chem.*, 59:471–475, 1996.

[22] T. Fleig, J. Olsen, and C. M. Marian. The generalized active space concept for the relativistic treatment of electron correlation. I. Kramers-restricted two-component configuration interaction. *J. Chem. Phys.*, 114(11):4775–4790, 2001.

[23] P. Å. Malmqvist, A. Rendell, and B. O. Roos. The restricted active space self-consistent-field method, implemented with a split graph unitary group approach. *J. Phys. Chem.*, 94(14):5477–5482, 1990.

[24] D. R. Hartree. The wave mechanics of an atom with a non-Coulomb central field. Part I. Theory and methods. *Proc. Camb. Phil. Soc.*, 24:89–132, 1928.

[25] V. Fock. Näherungsmethode zur Lösung des quantenmechanischen Mehrkörperproblems. *Z. Phys.*, 61:126–148, 1930.

[26] M. Born and J. R. Oppenheimer. Zur Quantentheorie der Molekeln. *Ann. Phys.*, 84:457–484, 1927.

[27] C. C. J. Roothaan. Self-consistent field theory for open shells of electronic systems. *Rev. Mod. Phys.*, 32:179–194, 1960.

[28] J. A. Pople and R. K. Nesbet. Self-consistent orbitals for radicals. *J. Chem. Phys.*, 22:571–574, 1954.

[29] C. A. Coulson and I. Fischer. Notes on the molecular orbital treatment of the hydrogen molecule. *Phil. Mag.*, 40:386–393, 1949.

[30] P. O. Löwdin. Quantum theory of many-particle systems. I. Physical interpretations by means of density matrices, natural spin-orbitals, and convergence problems in the method of configurational interaction. *Phys. Rev.*, 97(6):1474–1489, 1955.

[31] R. Pauncz. *Spin Eigenfunctions*. Plenum Press, New York, NY, 1997.

[32] C. Lanczos. An iteration method for the solution of the eigenvalue problem of linear differential and integral operators. *J. Res. Nat. Bur. Stand.*, 45:255–281, 1950.

[33] B. O. Roos. A new method for large-scale CI calculations. *Chem. Phys. Lett.*, 1:153–159, 1972.

[34] W. Duch. Configuration-interaction method — the past and future perspectives. *Theochem-J. Mol. Struct.*, 80:27–49, 1991.

[35] M. A. Robb and U. Niazi. The unitary group approach to electronic structure computations. In *Rep. Mol. Theory*, pages 23–55. CRC Press, Boca Raton, FL, 1990.

[36] M. A. Robb. Perspective on "Group theoretical approach to the configuration interaction and perturbation theory calculations for atomic and molecular systems" - Paldus J (1974) J Chem Phys 61 : 5321. *Theor. Chem. Acc.*, 103(3–4):317–321, 2000.

[37] J. Paldus. Group theoretical approach to the configuration interaction and perturbation theory calculations for atomic and molecular systems. *J. Chem. Phys.*, 61(12):5321–5330, 1974.

[38] I. M. Gelfand and M. L. Tsetlin. Finite dimensional representations of the group of unimodular matrices. *Dokl. Akad. Nauk SSSR*, 71:825–828, 1950.

[39] I Shavitt. *Int. J. Quantum Chem. Symp.*, 11:131, 1977.

[40] I Shavitt. Matrix element evaluation in the unitary group approach to the electron correlation problem. *Int. J. Quantum Chem. Symp.*, 12:5–32, 1978.

[41] P. Saxe, D. J. Fox, H. F. Schaefer, and N. C. Handy. The shape-driven graphical unitary-group approach to the electron correlation-problem — application to the ethylene molecule. *J. Chem. Phys.*, 77:5584–5592, 1982.

[42] P. J. Knowles and N. C. Handy. A new determinant-based full configuration-interaction method. *Chem. Phys. Lett.*, 111:315–321, 1984.

[43] P. E. M. Siegbahn. A new direct CI method for large CI expansions in a small orbital space. *Chem. Phys. Lett.*, 109:417–423, 1984.

[44] S. Zarrabian, C. R. Sarma, and J. Paldus. Vectorizable approach to molecular CI problems using determinantal basis. *Chem. Phys. Lett.*, 155:183–188, 1989.

[45] G. L. Bendazzoli and S. Evangelisti. A vector and parallel full configuration-interaction algorithm. *J. Chem. Phys.*, 98(4):3141–3150, 1993.

[46] P. E. M. Siegbahn, J. Almlöf, A. Heiberg, and B. O. Roos. The complete active space SCF (CASSCF) method in a Newton-Raphson formulation with application to the HNO molecule. *J. Chem. Phys.*, 74(4):2384–2396, 1981.

[47] E. Rossi, G. L. Bendazzoli, and S. Evangelisti. Full configuration interaction algorithm on a massively parallel architecture: direct-list implementation. *J. Comp. Chem.*, 19(6):658–672, 1998.

[48] F. Stephan and W. Wenzel. An algorithm for the multi-reference configuration interaction method on distributed memory architectures. *J. Chem. Phys.*, 108:1015–1022, 1998.

[49] E. R. Davidson. The iterative calculation of a few of the lowest eigenvalues and corresponding eigenvectors of large real-symmetric matrices. *J. Comp. Phys.*, 17:87–94, 1975.

[50] I. Waller and D. R. Hartree. *Proc. Roy. Soc. A*, 124:119, 1929.

[51] C. R. Sarma and J. Paldus. Spinor group and its restrictions. *J. Math. Phys.*, 26:1140–1145, 1985.

[52] L. Gagliardi, G. L. Bendazzoli, and S. Evangelisti. Direct-list algorithm for configuration interaction calculations. *J. Comput. Chem.*, 18:1329–1343, 1997.

[53] Scientific Computing Associates. *Linda, User's Guide & Reference Manual, Version 3.0*, 1995.

[54] M. J. Frisch, G. W. Trucks, H. B. Schlegel, G. E. Scuseria, M. A. Robb, J. R. Cheeseman, V. G. Zakrzewski, J. A. Montgomery, R. E. Stratmann, J. C. Burant, S. Dapprich, J. M. Millam, A. D. Daniels, K. N. Kudin, M. C. Strain, O. Farkas, J. Tomasi, V. Barone, M. Cossi, R. Cammi, B. Mennucci, C. Pomelli, C. Adamo, S. Clifford, J. Ochterski, G. A. Petersson, P. Y. Ayala, Q. Cui, K. Morokuma, D. K. Malick, A. D. Rabuck, K. Raghavachari, J. B. Foresman, J. Cioslowski, J. V. Ortiz, B. B. Stefanov, G. Liu, A. Liashenko, P. Piskorz, I. Komaromi, R. Gomperts, R. L. Martin, D. J. Fox, T. Keith, M. A. Al-Laham, C. Y. Peng, A. Nanayakkara, C. Gonzalez, M. Challacombe, P. M. W. Gill, B. Johnson, W. Chen, M. W. Wong, J. L. Andres, M. Head-Gordon, E. S. Replogle and J. A. Pople. Gaussian 99, Development Version (Revision B.1), 1998. Gaussian, Inc., Pittsburgh PA.

[55] S. Wilsey, L. González, M. A. Robb, and K. N. Houk. Ground- and excited-state surfaces for the $[2+2]$-photocycloaddition of α,β-enones to alkenes. *J. Am. Chem. Soc.*, 122(4):5866–5876, 2000.

[56] A. Sanchez-Galvez, P. Hunt, M. A. Robb, M. Olivucci, T. Vreven, and H. B. Schlegel. Ultrafast radiationless deactivation of organic dyes: evidence for a two-state two-mode pathway in polymethine cyanines. *J. Am. Chem. Soc.*, 122(12):2911–2924, 2000.

[57] K. Andersson, P. Å. Malmquist, and B. O. Roos. Second-order perturbation theory with a complete active space self-consistent field reference function. *J. Chem. Phys.*, 96(2):1218–1226, 1992.

[58] B. O. Roos. Theoretical studies of electronically excited states of molecular systems using multiconfigurational perturbation theory. *Acc. Chem. Res.*, 32:137–144, 1999.

[59] P. E. M. Siegbahn. Generalizations of the direct CI method based on the graphical unitary group approach. II. Single and double replacements from any set of reference functions. *J. Chem. Phys.*, 72(3):1647–1656, 1980.

[60] V. R. Saunders and J. H. van Lenthe. The direct CI method. A detailed analysis. *Mol. Phys.*, 48(5):923–954, 1983.

[61] P. M. Kozlowski and P. Pulay. The unrestricted natural orbital-restricted active space method: methodology and implementation. *Theor. Chem. Acc.*, 100:12–20, 1998.

[62] V. R. Saunders and J. H. Van Lenthe. The direct CI method. A detailed analysis (Reprinted from Molecular Physics, vol 48, pg 923, 1983). *Mol. Phys.*, 100:167–187, 2002.

[63] W. Duch. Calculation of the one-electron coupling-coefficients in the configuration-interaction method. *Chem. Phys. Lett.*, 124:442–446, 1986.

[64] P. E. M. Siegbahn. Generalizations of the direct CI method based on the graphical unitary group approach. I. Single replacements from a complete CI root function of any spin, first order wave functions. *J. Chem. Phys.*, 70(12):5391–5397, 1979.

[65] N. Bajpai, C. R. Sarma, and J. Paldus. Dense indexing scheme for limited configuration-interaction calculations with determinantal basis. *Theochem-J. Mol. Struct.*, 73:67–77, 1991.

[66] L. Serrano Andrés and B. O. Roos. Theoretical study of the absorption and emission spectra of indole in the gas phase and in a solvent. *J. Am. Chem. Soc.*, 118(1):185–195, 1996.

[67] C.-P. Hsu, S. Hirata, and M. Head-Gordon. Excitation energies from time-dependent density functional theory for linear polyene oligomers: butadiene to decapentaene. *J. Phys. Chem.*, 105(2):451–458, 2001.

[68] C. Woywod, W. C. Livingood, and J. H. Frederick. S_1-S_2 vibronic coupling in *trans*-1,3,5-hexatriene. I. Electronic structure calculations. *J. Chem. Phys.*, 112(2):613–625, 2000.

[69] L. Serrano Andrés, M. Merchan, I. Nebotgil, R. Lindh, and B. O. Roos. Towards an accurate molecular-orbital theory for excited-states – ethene, butadiene, and hexatriene. *J. Chem. Phys.*, 98(4):3151–3162, 1993.

[70] K. Nakayama, H. Nakano, and K. Hirao. Theoretical study of the $\pi \rightarrow \pi^*$ excited states of linear polyenes: the energy gap between $1^1B_u^+$ and $2^1A_g^-$ states and their character. *Int. J. Quantum Chem.*, 66(2):157–174, 1998.

[71] R. L. Graham and K. F. Freed. Ab inition study of the trans-butadiene pi-valence states using the effective valence shell hamiltonian method. *J. Chem. Phys.*, 96(2):1304–1316, 1992.

[72] P. Cronstrand, O. Christiansen, P. Norman, and H. Agren. Ab initio modeling of excited state absorption of polyenes. *Phys. Chem. Chem. Phys.*, 3(13):2567–2575, 2001.