# Performance Analysis, PVM and MPI Implementation of a DSCF Hartree Fock Program

Siegfried Höfinger[1] , Othmar Steinhauser[1] and Peter Zinterhof[2]

[1] Institute for Theoretical Chemistry, Molecular Dynamics Group, University of Vienna, Vienna, Austria
[2] Institute for Mathematics, University of Salzburg, Salzburg, Austria

A new Direct SCF-Hartree Fock program (DSCF) has been improved by the method of DIIS.[17], [18] General performance measurement tools, as provided on the SGI-Power Challenge/R10000 (194 MHz) running IRIX 6.2, which are the shell-commands **perfex -a a.out** and **ssrun -[pcsampl, ideal, usertime] a.out**, have been used for profiling the code. The main cpu-time-consuming subroutine was detected and parallel versions for PVM 3.3 as well as MPI have been deduced. An additional module for the purpose of achieving load-balance was introduced and obtained speed-up parameters are presented and compared.

## 1. Introduction

Electronic structure determination has always been one of the key topics in computational chemistry and, due to the enormous progress in the performance of today's hardware architecture, this subject is becoming more and more important even in the fields of traditional biosciences, such as molecular biology, immunology, pharmaceutics, drug design and many more. As a result of this, a well designed algorithm for the calculation of molecular orbitals of large systems is still an interesting task one can pay attention to.

### 1.1. Time Independent Schrödinger Equation

The state of the art method for gaining some information about the electrons within a certain molecule is iteratively solving the *time independent Schrödinger equation* [1]

$$\left[ \sum_{i}^{electrons} -\frac{1}{2}\Delta_i + V(\vec{r_1}, \vec{r_2} \ldots) \right] \psi_{el}(\vec{r_1}, \vec{r_2} \ldots) = E\, \psi_{el}(\vec{r_1}, \vec{r_2} \ldots) \quad (1)$$

where the square bracket term on the left hand side of equation 1 is usually referred to as *the Hamiltonian Operator* $\hat{H}$, whose first part comprises *the Kinetic Energy*[1] of the electrons and whose second part describes *the Potential* for the electrons

$$V(\vec{r_1}, \vec{r_2} \ldots) = \sum_{k}^{nuclei} \sum_{l}^{nuclei} \frac{Z_k\, Z_l}{|\vec{r_k} - \vec{r_l}|} - \sum_{i}^{electr} \sum_{k}^{nuclei} \frac{Z_k}{|\vec{r_i} - \vec{r_k}|} + \sum_{i}^{electr} \sum_{j}^{electr} \frac{1}{|\vec{r_i} - \vec{r_j}|} \quad (2)$$

consisting of *Nuclear Repulsion*, *Nuclear Attraction*[2] and *Electron Repulsion*.

Furthermore, the $\psi_{el}(\vec{r_1}, \vec{r_2} \ldots)$ of equation 1 is called *the Wave-function of the Electrons*[3] and

[1] As usual, the $\Delta_i$ is used as an abbreviation for *the Laplacian Operator*.
[2] the $Z_k$ stands for nuclear charge at center $\vec{r_k}$.
[3] Note, that for this very general formulation each electron has its own spatial coordinates $\vec{r_i}$.

the *Eigenvalue E* on the right hand side of equation 1 stands for *the Total Energy* of the electrons and from all of these it may become clear that the whole Schrödinger equation represents an *Eigenvalue Problem*.

For a graphical explanation of the individual variables, please have a look at the following figure 1.

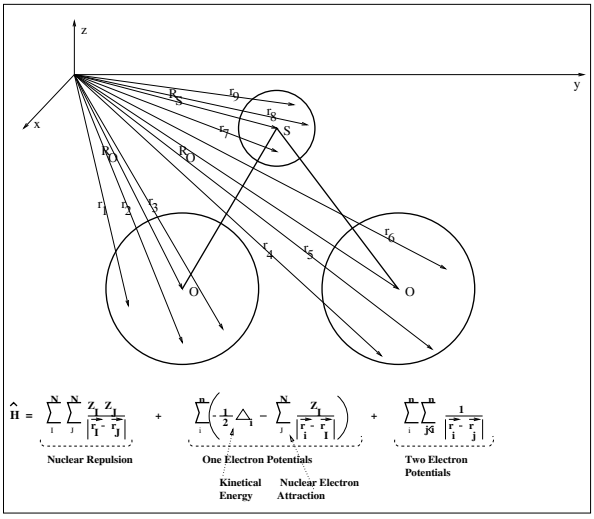


*Fig. 1.* Schematic view of the Hamiltonian Operator $\hat{H}$ describing the electronic structure of $SO_2$, which has been selected to function as a model-system for all further analysis.

## 1.2. Decoupling of Electron Repulsion

The coupling of electrons, as given from the last term of equation 2, leads to a correlated motion of electrons, that, explicitly treated, would result in tremendous computational efforts, which might be fairly released, if the following approximation is used,

$$\sum_{i}^{electrons} \sum_{j}^{electrons} \frac{1}{|\vec{r_i} - \vec{r_j}|} \approx \sum_{i}^{electrons} \left[ \int_{\vec{r}} \frac{\rho(\vec{r})d\vec{r}}{|\vec{r_i} - \vec{r}|} + v_{eff}(\vec{r_i}) \right] \quad (3)$$

with $\rho(\vec{r})$ standing for the *Electron Density* around point $\vec{r}$ and $v_{eff}(\vec{r_i})$ being an *Effective Potential*.

With this approximation, the general n-particle Schrödinger equation 1 reduces to its decoupled one-electron form of $n^4$ identical one-particle equations

$$\left[ -\frac{1}{2}\Delta - \sum_{k}^{nuclei} \frac{Z_k}{|\vec{r} - \vec{r_k}|} + \int_{\vec{r}} \frac{\rho(\vec{r})d\vec{r}}{|\vec{r} - \vec{r}|} + v_{eff}(\vec{r}) \right] \phi_k(\vec{r}) = \varepsilon_k \ \phi_k(\vec{r}) \quad (4)$$

where now the $\phi_k(\vec{r})$ are one–electron wave-functions or molecular orbitals ($MO$[5] ) and equation 4 once more reflects an eigenvalue problem.

The fundamental property characterizing a molecule, the so called one-electron density $\rho(\vec{r})$, is related to the n-particle wave-function $\psi_{el}(\vec{r_1}, \vec{r_2} \ldots \vec{r_n})$ by

$$\rho(\vec{r_1}) = \int_{\vec{r_2}} \int_{\vec{r_3}} \ldots \int_{\vec{r_n}} \psi^*_{el}(\vec{r_1}, \vec{r_2} \ldots) \psi_{el}(\vec{r_1}, \vec{r_2} \ldots) d\vec{r_2} d\vec{r_3} \ldots d\vec{r_n} \quad (5)$$

and takes the special form

$$\rho(\vec{r}) = \sum_{i}^{electrons} \phi^*_i(\vec{r}) \phi_i(\vec{r}) \quad (6)$$

for a *Slater Determinant*.[6]

## 1.3. Effective Potential

The practical representation of $v_{eff}(\vec{r})$ in equation 4 gives rise to two different main stream directions in quantum mechanics, where the one is called *Density Functional Theory* [5] and the other is referred to as *Hartree Fock Theory* [6] [7]. The latter has been used for the present program we are talking about.

In Hartree Fock theory ( HF ) the $3^{rd}$ and $4^{th}$ term of equation 4 are replaced by the following closed form

[4] n stands for total number of electrons.
[5] Molecular orbitals shall be designated $\phi_k$ and are said to form an orthonormal set.
[6] $\psi_{el}(\vec{r_1}, \vec{r_2} \ldots) = \mathcal{P} \left[ \phi_1(\vec{r_1}) \phi_2(\vec{r_2}) \ldots \phi_n(\vec{r_n}) \right]$ where $\mathcal{P}$ means Permutation and all is done in order to respect *the Pauli Principle* [3], that results from the *antisymmetric* character of the wave-function.
[7] $J_j$ in equation 7 is called *Coulomb Operator* and $K_j$ in equation 7 is called *Exchange Operator*.
[8] Note, that the Index k refers to $\phi_k$ of equation 4 and thus shows up the exchange character of $K_j$.

$$\left[\int_{\vec{r}} \frac{\rho(\vec{r})d\vec{r}}{\left|\vec{r}-\vec{r}\right|} + v_{eff}(\vec{r})\right]_{HF} \hat{=} \sum_{j}^{electrons}\left(J_j - \frac{1}{2}K_j\right) \tag{7}$$

with the J- and K-terms[7] given by[8]

$$J_j{=}J_j(\vec{r})\phi_k(\vec{r}){=}\left[\int_{\vec{r}} \phi_j^*(\vec{r})\frac{1}{\left|\vec{r}-\vec{r}\right|}\phi_j(\vec{r})d\vec{r}\right]\phi_k(\vec{r}) \tag{8}$$

$$K_j{=}K_j(\vec{r})\phi_k(\vec{r}){=}\left[\int_{\vec{r}} \phi_j^*(\vec{r})\frac{1}{\left|\vec{r}-\vec{r}\right|}\phi_k(\vec{r})d\vec{r}\right]\phi_j(\vec{r})\,. \tag{9}$$

Thus the sum over J-terms together with equation 6 is precisely the *Coulomb Potential* ( $3^{rd}$ term of equation 4 ) and the sum over K-terms represents a *Nonlocal Effective Exchange Potential*.

So far we always talked about the electrons only and never spent a word on nuclear charge distribution, that theoretically should again be subject to another wave-function and treated in the same quantum way the electrons had been. The neglect of an explicit wave-function for the description of the nuclear charges is due to the fact, that the two wave-functions, for electrons and nuclei either, may be separated into two independent ones, which is commonly known as the *Born-Oppenheimer Approximation*[9] [2].

## 1.4. SCF-Procedure

As may be seen from equation 4, the decoupled one–electron Schrödinger equation depends on the electron density $\rho(\vec{r})$, which in its turn is a function of all MOs involved. Therefore all other electrons have a strong influence on the solution of the eigenvalue problem for a particular electron.

This leads to the paradox situation, that one should already know all MOs in order to be able to determine one particular MO explicitly.

To overcome this principal problem, equation 4 has to be solved via a so-called *Self Consistent Field* procedure ( SCF ), which in a sketchy way may be described as

- Set up a first trial density $\rho_1(\vec{r})$, that will naturally be far away from the actual physical relevant one.
- Solve the eigenvalue problem 4 and thereby get access to a set of MOs, which in turn results in a better, more realistic new density $\rho_2(\vec{r})$ via equation 6.
- Resolve the next eigenvalue problem using the improved density $\rho_2(\vec{r})$ and again get another, even more improved set of MOs and so on. . . until two subsequent sets of MOs are almost identical to each other and differ only by a predefined small threshold value.

## 1.5. DIIS-Method

Contrary to our initial report [16] within the present version, we make use of the *Convergence Acceleration of Iterative Sequences.*[17], which is also known as **DIIS**, meaning *Direct Inversion in Iterative Subspace*[18], and which in an oversimplifying way may be regarded as a method capable of drastically reducing the total number of necessary iterations. Thus for our special $SO_2$ case, the application of DIIS could reduce the total number of iterations necessary to achieve self consistency from 44 to 16 and, according to the fact that each individual iteration involves the most elaborate task of ERI-computation, the availability of DIIS becomes a limiting factor in terms of cpu-time as well.

## 1.6. LCAO, Main Problem and Method

According to *the LCAO – approach*[10], the MOs are again expanded into a linear combination of atomic orbitals( *AO*[11] ), where the latter might also simply be called *Basis Functions*[12]

$$\phi_k(\vec{r}) = \sum_i c_{k,i}\ \varphi_i(\vec{r}) \tag{10}$$

[9] All nuclear centers are considered fixed in space with fixed partial charges $Z(\vec{r_k})$ located at those very points in space.
[10] Linear Combination of Atomic Orbitals.
[11] Atomic orbitals shall be designated $\varphi_i$.
[12] Basis functions are those widely known 1s, 2s, 2px, 2py, 2pz. . . orbitals for the description of one-electron atomic systems such as H, $He^+$, $B^{2+}$. . .

| **Schematic Representation of $SO_2$** | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| $Center_{Element}$ | $\boxed{1}_S$ | | | | $\boxed{2}_O$ | | | $\boxed{3}_O$ | | |
| Cntr. Shll.$_{Typ}$ | $(1)_S$ | $(2)_{SP}$ | $(3)_{SP}$ | $(4)_{SP}$ | $(5)_S$ | $(6)_{SP}$ | $(7)_{SP}$ | $(8)_S$ | $(9)_{SP}$ | $(10)_{SP}$ |
| Basisf.$_{Typ}$ | $1_S$ | $2_S$ | $6_S$ | $10_S$ | $14_S$ | $15_S$ | $19_S$ | $23_S$ | $24_S$ | $28_S$ |
| | | $3_{Px}$ | $7_{Px}$ | $11_{Px}$ | | $16_{Px}$ | $20_{Px}$ | | $25_{Px}$ | $29_{Px}$ |
| | | $4_{Py}$ | $8_{Py}$ | $12_{Py}$ | | $17_{Py}$ | $21_{Py}$ | | $26_{Py}$ | $30_{Py}$ |
| | | $5_{Pz}$ | $9_{Pz}$ | $13_{Pz}$ | | $18_{Pz}$ | $22_{Pz}$ | | $27_{Pz}$ | $31_{Pz}$ |

*Table 1.* Shell concept explained at the model-molecule $SO_2$, that has been used for further analysis.

and the basis functions $\varphi_i$ are again expanded in a series over *Primitive Gaussians* $\chi_j$

$$\varphi_i(\vec{r}) = \sum_j d_{i,j}\, \chi_j(\vec{r}) \quad , \tag{11}$$

which typically are *Cartesian Gaussian Functions* located at some place $(A_x, A_y, A_z)$ in space[13] [8] [9].

$$\chi_j(\vec{r}) = N_j(x-A_x)^l(y-A_y)^m(z-A_z)^n\; e^{-\alpha_j(\vec{r}-\vec{A})^2} \tag{12}$$

The main problem for SCF-calculations is the evaluation of the ERIs, *the Electron Repulsion Integrals*, that are 6-dimensional, 4-center integrals over the basis functions $\varphi$.

$$ERI = \int_{\vec{r_1}} \int_{\vec{r_2}} \varphi_i(\vec{r_1})\varphi_j(\vec{r_1}) \frac{1}{|\vec{r_2}-\vec{r_1}|} \varphi_k(\vec{r_2})\varphi_l(\vec{r_2}) d\vec{r_1} d\vec{r_2} \tag{13}$$

There are various ways to compute ERIs [10] [11] [12], but the method used in the present program is the recursive method[14]described by *Obara and Saika* [12].

### 1.7. Shell Concept and Model-Molecule $SO_2$

Without any intention of going into further details, we just want to outline, at least basically, the principal scheme behind the recursive construction of ERIs due to *Obara and Saika* [12]. For all further report we present data for a simple molecule, that has been used as a certain kind of reference – $SO_2$ in particular. A schematic representation of the molecule is given in table 1. The applied basis function specification has been the standard 6-31G basis-set [13] [14]. The main advantage of this 6-31G basis-set is, that there exist cross-contracted shells, e.g. SP-contracted shells, which will utilize the same exponential factor ( $\alpha$ in equation 12 ) for S-type basis functions as well as P-type basis functions[15], and hence will offer a chance to calculate the more complicated ERIs, that contain P-type basis functions, in an recursive way from easily generated pure S-type ERIs.

As stated above, ERIs are 4-center integrals, and therefore we will have to combine 4 centers together with the according contracted shells to determine how many ERIs may be built in one subsequent, recursive subprocess. For example, consider the center-quartet $\boxed{1}\,\boxed{1}\,\boxed{2}\,\boxed{3}$, then one possible combination of contracted shells would be $(2)(3)(5)(9)$[16], with the basic S-type ERI made up from basis functions 2 6 14 24, and after initial calculation of this basic S-type ERI, a total number of 63 related ERIs may be derived recursively, such as 2 6 14 25, 2 6 14 26, 2 6 14 27, 2 7 14 24, 2 7 14 25, 2 7 14 26 . . . 5 9 14 27.

## 2. Performance Analysis

### 2.1. General Status

To shed some light onto critical regions of our program, we have used performance measure-

[13] An S-type basis function will consist of primitive gaussians with $l = m = n = 0$, a P-type however of primitives with $l + m + n = 1$, which may be solved at 3 different ways, either $l = 1$ and $m = n = 0$, or $m = 1$ and $l = n = 0$, or $n = 1$ and $l = m = 0$. D-type specification will likewise be $l + m + n = 2$ and similarly F-type $l + m + n = 3$.

[14] All complicated ERI-types $(l + m + n > 0)$ may be deduced from the easier computed $(S, S_j|S_k, S_l)$ type.

[15] For example, basis functions 6, 7, 8 and 9 of contracted shell $(3)$ at center $\boxed{1}$ will all have the same exponent $\alpha$.

[16] There are 90 possible combinations of relevant contracted shells within this particular center-quartet and we have only picked one out of them.

ment tools, as provided on a SGI-Power Challenge/R10000 (194 MHz) running IRIX 6.2.

Some insight into the basic events of the R10000 processor for some specific program execution may be gained by using the shell-command **perfex -a a.out**. Then a table is given, which represents a listing of total counts of various events, such as $2^{nd}$ level data cache misses, or issued loads, or graduated loads, or ... that happened during program execution and thus[17] a relative weighting of critical events may be done to notice some basic bottlenecks. A listing of the top 6 events during our SCF-process together with some relative numbers of highly sensitive events is given in table 2.

#### 2.1.1. Interpretation

At least no principal performance bottlenecks were encountered, especially when looking at highly sensitive events, such as data cache misses and mispredicted branches. So we don't think that this application suffers from serious performance problems, which could be solved at the programmer's level.

### 2.2. Main Time Consuming Modules

The other thing we were highly interested in, was to select those subroutines and functions, that play the main part during program execution as far as cpu-time is concerned. Therefore we did some performance measurement on the SGI-Power Challenge/R10000 (194 MHz) running IRIX 6.2, namely **ssrun -[pcsampl, ideal, usertime] a.out**, which all produce ordered listings of the involved modules according to their fraction in the total cpu-time.

No matter which of the three optional parameters were selected in particular, it always turned out that 99.7 % of the total cpu-time was spent in *Subroutine FMAT*, which is our module for ERI–computation.

## 3. Parallelization

As has become clear from the previous section 2.2, our primary goal for parallelization purposes had to be subroutine *FMAT* — the ERI-calculation module.

For the first approach we used PVM 3.3[18] for a host–node model due to the MPMD scheme[19], where the outermost loop[20] within subroutine *FMAT* had been split and partitioned over a certain range of nodes.

| Top 6 Events and Relative Timings | | | | |
|---|---|---|---|---|
| Event | Counts | Typical Event–Cost | Absolute Time [min] | Relative Time [%] |
| Cycles | $6.663\ 10^{10}$ | 1.00 | 5.724 | 100.00 |
| Issued Loads | $3.674\ 10^{10}$ | 1.00 | 3.157 | 55.15 |
| Graduated Loads | $2.954\ 10^{10}$ | 1.00 | 2.538 | 44.34 |
| Issued Stores | $1.555\ 10^{10}$ | 1.00 | 1.336 | 23.34 |
| Graduated Stores | $1.548\ 10^{10}$ | 1.00 | 1.329 | 23.23 |
| Decoded Branches | $5.888\ 10^{9}$ | 1.00 | 0.506 | 8.84 |
| Graduated FP Instruc. | $3.881\ 10^{9}$ | 1.00 | 0.334 | 5.83 |
| Prim. Data Cache Miss | $2.033\ 10^{8}$ | 9.03 | 0.158 | 2.76 |
| Mispredicted Branch | $6.499\ 10^{8}$ | 1.50 | 0.084 | 1.46 |
| Sec. Data Cache Miss | $1.381\ 10^{5}$ | 192.12 | 0.002 | 0.04 |

*Table 2.* Relative timings of specific hardware counters on R10000, SGI for one SCF-process completion.

[17] After multiplying those absolute counts with specific event–cost–figures.
[18] Rel. 11 on the alpha-cluster made up of equal dec 3000 nodes, Rel. 10 on the SGI Power Challenge R10000.
[19] Multiple ( different and executable ) Programs Multiple Data.
[20] Loop over quartets of center.

| Wall Clock Times and Speed–Up | | | |
|---|---|---|---|
| | Number of Nodes | Wall Clock Time [min] | Speed Up |
| Dec 3000 Cluster | 1 – sequential | 46:12 | |
| Dec 3000 Cluster | 2 | 32:03 | 1.44 |
| Dec 3000 Cluster | 4 | 21:00 | 2.20 |
| Dec 3000 Cluster | 8 | 13:34 | 3.46 |

*Table 3.* Wall-clock times for parallel SCF on Dec 3000 cluster for different number of nodes and derived speed-up parameter. Arithmetic average partitioning scheme. ( Without DIIS ! )

To begin with, we present data ( table 3 ) resulting from a simple arithmetic average partitioning scheme, where each of the involved nodes got to work on a certain subset of center quartets and the number of individual items within this subsets was derived by simply dividing the total number of all possible center quartets through the number of nodes involved. ( partitioning scheme also shown on left side of table 4 )

### 3.1. Load Balancing

After realizing the poor speed up factors obtained from the initial, simple arithmetic average partitioning scheme ( left side of table 4 ), we went one step further and introduced a pre-scanning subroutine, with which we could estimate the net work to be done. This means, that instead of actual calculating ERIs, we just increment a counter variable in the according section of the program, where usually the recursive relations for the computation of ERIs come into play, and thus we end up with a representative number for the total computational work to be done, which may be divided through the number of nodes involved and in a subsequent repetition of the dummy loops it may be useful to assign the upcoming center quartets to the same node-specific pair-list as long as the mentioned fraction $\frac{total\ work\ counter}{number\ of\ nodes}$ is not reached. In this way we were able to build so called load-balanced pair-lists, that represented the outermost loop over center quartets. The latter scheme gave rise to the following pair-lists of center[21] ( right side of table 4 ) and a summary of corresponding speed-up data for the likewise improved parallel versions is given in table 5 for the PVM version as well as for the MPI version both executed on the SGI-Power Challenge/R10000 (194 MHz). A graphical representation of table 5 is given in figure 2.

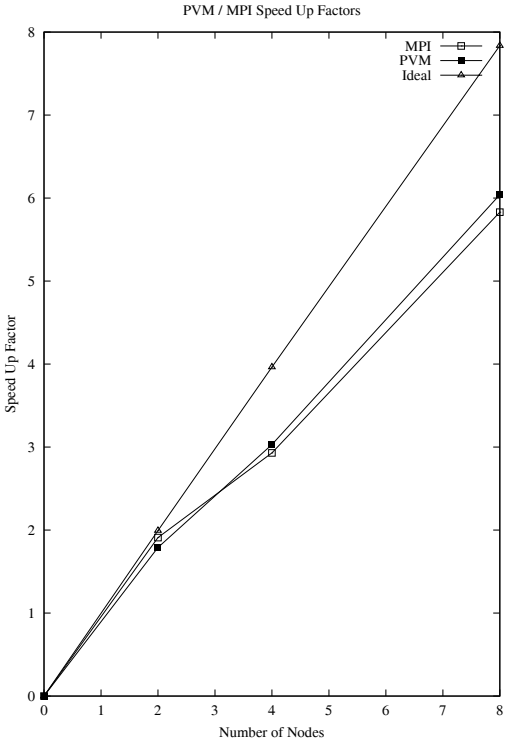


*Fig. 2.* Comparison of Speed-up Factors for PVM and MPI approach.

## 4. Discussion

### 4.1. Dominance of Heavy Atoms like S

As might have become clear from section 1.7 all center quartets containing atom S ( or related pairs of centers – 1, 2, 3 ) will result in larger numbers of according ERIs, because S is built up from 3 cross contracted SP-shells — (2), (3)

[21] According to the specification done in table 1 pair 1 will be $\boxed{1}\boxed{1}$, 2 is $\boxed{1}\boxed{2}$, 3 is $\boxed{1}\boxed{3}$, 4 is $\boxed{2}\boxed{2}$, 5 is $\boxed{2}\boxed{3}$ and 6 is $\boxed{3}\boxed{3}$.

and (4) — whereas the remaining two O-atoms are only built up from 2 SP-shells, which is also noticed in table 4 ( right side ), because node IV for example is mainly working on ERIs not containing atom S, thus effecting 10 different center quartets, whereas node I may only work on 3 center quartets but these will always include atom S and the work of both nodes should still be almost equal to each other.

## 4.2. Amdahl's Law and a Detailed Picture of the Time Distribution within the Parallel Process

According to *Amdahl's Law*

$$SpeedUp \leq \frac{1}{s + \frac{1-s}{N_{cpu}}} \qquad (14)$$

with $s$ standing for the serial fraction and $N_{cpu}$ for the number of nodes, we should obtain speed up values almost equal to the number of nodes involved, if we assume that $s$ is sufficiently small and that communication may be neglected. From section 2.2 we may conclude that $s = 0.003$, which leads to theoretical speed up values of 1.994 for 2 nodes, 3.964 for 4 nodes and 7.836 for 8 nodes.

Nevertheless our measured speed up values as represented in table 5 differ quite a lot from those theoretical values, which is due to the fact, that despite serious efforts to enable a well balanced work distribution one cannot overcome the principal block structure for ERI-computation ( section 1.7 ). This means, that even after building load balanced pair-lists all nodes will still have to work on slightly varying fractions of partial work, that are comparable to each other, but not exactly equal.[22]

So after 3 nodes, for example, have already finished their partial work, node IV may still be busy completing its last center quartet, which

| **Pair-list of Center for $SO_2$ Arithmetic Average** | | |
|---|---|---|
| Node | Left Pair of Center | Right Pair of Center |
| I | 1 | 1 |
| | 1 | 2 |
| | 1 | 3 |
| | 1 | 4 |
| | 1 | 5 |
| II | 1 | 6 |
| | 2 | 2 |
| | 2 | 3 |
| | 2 | 4 |
| | 2 | 5 |
| III | 2 | 6 |
| | 3 | 3 |
| | 3 | 4 |
| | 3 | 5 |
| | 3 | 6 |
| IV | 4 | 4 |
| | 4 | 5 |
| | 4 | 6 |
| | 5 | 5 |
| | 5 | 6 |
| | 6 | 6 |

⇒

| **Pair-list of Center for $SO_2$ Load Balanced** | | |
|---|---|---|
| Node | Left Pair of Center | Right Pair of Center |
| I | 1 | 1 |
| | 1 | 2 |
| | 1 | 3 |
| II | 1 | 4 |
| | 1 | 5 |
| | 1 | 6 |
| | 2 | 2 |
| III | 2 | 3 |
| | 2 | 4 |
| | 2 | 5 |
| | 2 | 6 |
| IV | 3 | 3 |
| | 3 | 4 |
| | 3 | 5 |
| | 3 | 6 |
| | 4 | 4 |
| | 4 | 5 |
| | 4 | 6 |
| | 5 | 5 |
| | 5 | 6 |
| | 6 | 6 |

*Table 4.* Arithmetic average and load-balanced partitioning scheme of the outermost loop over center quartets into node-specific pair-lists of center for $SO_2$ — 4 nodes considered.

[22] Like the area of a puzzle can be divided into a number of almost equal partial areas, but not into exactly equal partial areas, without destroying some particular puzzle slice.

| Wall Clock Times and Speed–Up | | | |
|---|---|---|---|
| | Number of Nodes | Wall Clock Time [sec] | Speed Up |
| SGI Pow.Chll. PVM | 1 – sequential | 268.19 | |
| SGI Pow.Chll. PVM | 2 | 149.78 | 1.79 |
| SGI Pow.Chll. PVM | 4 | 88.66 | 3.03 |
| SGI Pow.Chll. PVM | 8 | 44.40 | 6.04 |
| SGI Pow.Chll. MPI | 1 – sequential | 252.51 | |
| SGI Pow.Chll. MPI | 2 | 132.50 | 1.91 |
| SGI Pow.Chll. MPI | 4 | 86.10 | 2.93 |
| SGI Pow.Chll. MPI | 8 | 43.28 | 5.83 |

*Table 5.* Wall clock times for parallel DSCF for either PVM 3.3 or MPI approach on SGI-Power Challenge/R10000 (194 MHz) for different number of nodes and derived speed-up parameter. Load balanced partitioning scheme.

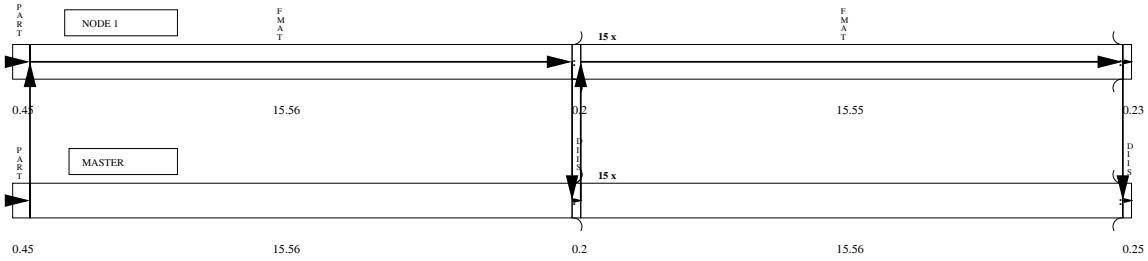


*Fig. 3.* Cpu-time distribution to the different threads for the MPI-approach of the DSCF calculation of $SO_2$ considering 1 master and 1 node process. The execution cycle is indicated by the arrows.

forces all other 3 nodes to stay idle in the meantime and thus induces an artificial enlargement of the serial fraction of program execution. According to this, in order to get a closer picture of how the cpu-time is spent in the different parallel tasks, we inserted time stamps in the source code of the program, so that each of the working nodes could report its own individual time it spent on the execution of its partial work. The authors found it most useful to utilize the **dtime()** function, that returns the actual cpu-time since the last call to dtime(). In this manner we actually could visualize the aforementioned latency-effect due to the undestroyable block-structure of the different nodes concerning their individual partial work to perform and a summary of the obtained cpu-time-flowcharts is given within figures 3, 4 and 5.

## 4.3. PVM versus MPI

After treating the PVM program the same way and inserting the same kind of time stamps there, we noticed slight differences between the MPI and the PVM approach concerning cpu-time distribution, which surprisingly was not due to communication impairment in the PVM task.

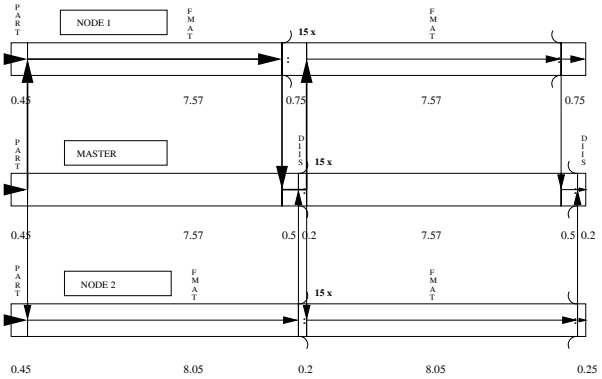


*Fig. 4.* Cpu-time distribution to the different threads for the MPI-approach of the DSCF calculation of $SO_2$ considering 1 master and 2 node processes. The execution cycle is indicated by the arrows.

The effect decreases with increasing number of nodes and thus becomes less harmful for the actual interesting jobs. An overview of the different influence of the PVM-delay is given within figures 6, 7, 8 and 9.

## 4.4. Neglectable Influence of the Additional Partitioning Module

With the help of the mentioned time labels we furthermore could easily estimate the cost of

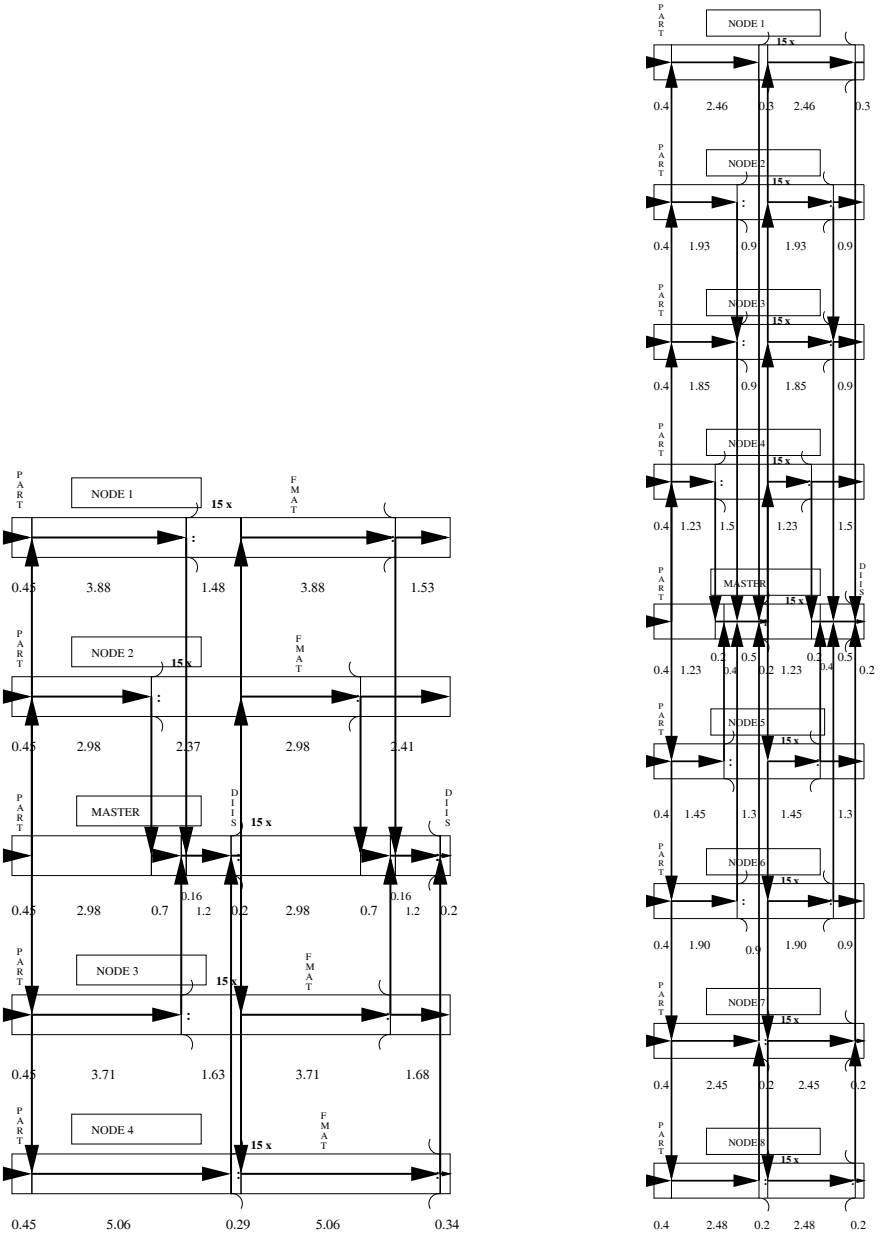


*Fig. 5.* Cpu-time distribution to the different threads for the MPI-approach of the DSCF calculation of $SO_2$ considering 1 master and 4 node processes (left) 1 master and 8 node processes (right). The execution cycle is indicated by the arrows.

this additional load balancing subroutine, that from the point of view of the SCF process is regarded artificial and not necessary and therefore should be kept at a minimum expensive level. It turned out, that the actual cpu-time spent for this additional task was 0.33 sec, which, even for the fastest run, is only a percentage of 0.76 %.

## 4.5. Conclusion

Recursive ERI-computation according to *Obara and Saika* [12] cannot be done in 100 % parallel mode, but after introduction of some load balanced pair-lists, a considerable amount of the global work may be parallel computed on distributed nodes.

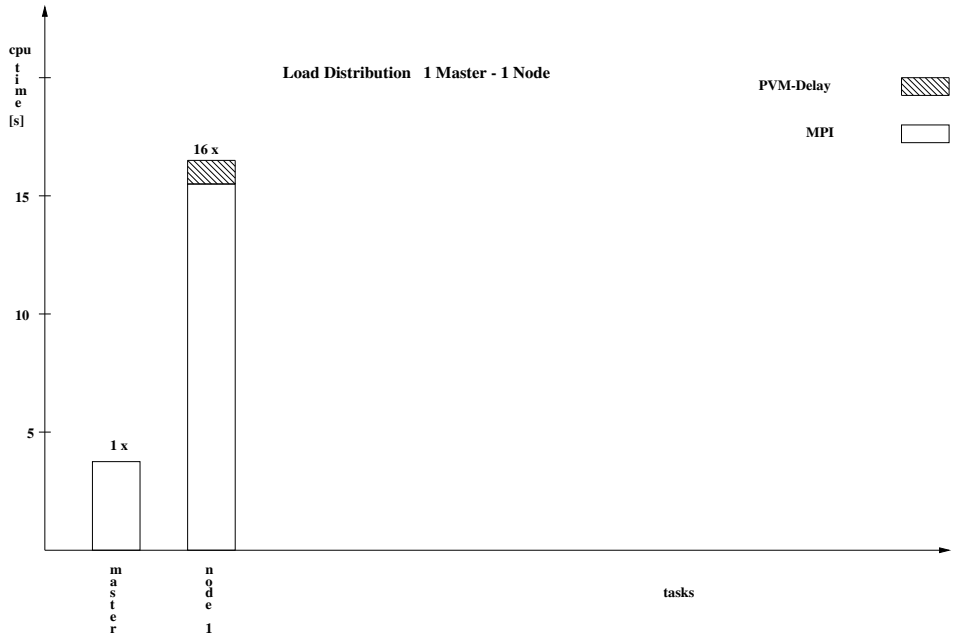


*Fig. 6.* Comparison of the cpu-time the different nodes have to spend on performing their partial work and indication of the effect of PVM-delay. ( 1 master 1 node )

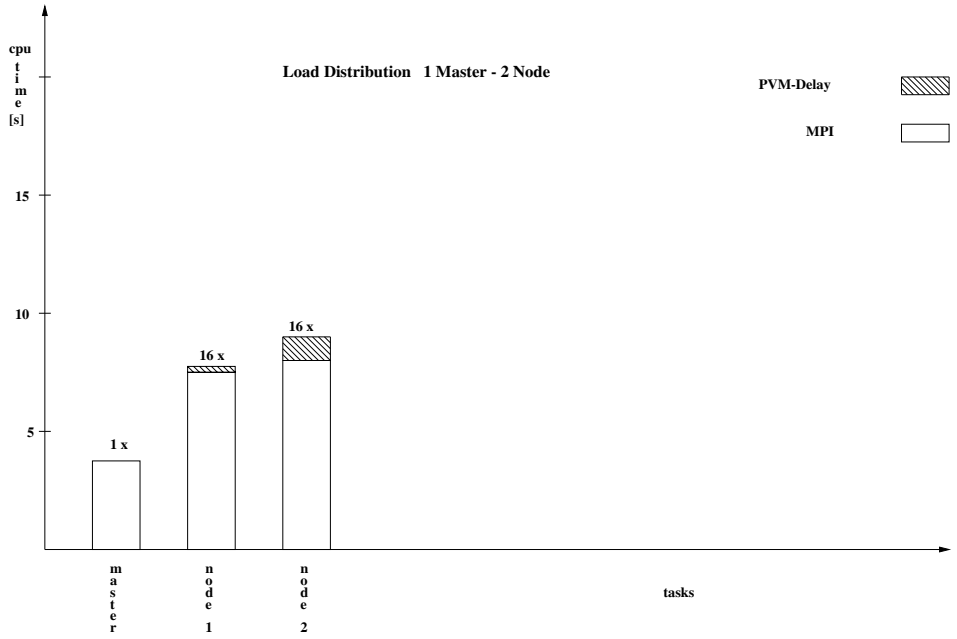


*Fig. 7.* Comparison of the cpu-time the different nodes have to spend on performing their partial work and indication of the effect of PVM-delay. ( 1 master 2 nodes )

## References

[1] SCHRÖDINGER, E. Quantisierung als Eigenwertproblem. Annalen der Physik. **79**, **80**, **81** (1926).

[2] BORN, M., OPPENHEIMER, R., Zur Quantentheorie der Moleküle. Annalen der Physik. **84** (1927) 457.

[3] PAULI, W., Exclusion Principle and Quantum Mechanics. Neuchatel, Griffon, 1st. Ed., Nobel Prize Lecture. (1947).

[4] SLATER, J.C., The Self Consistent Field for Molecules and Solids: Quantum Theory of Molecules and Solids Mc Graw-Hill, New York, **4** (1974).

[5] PARR, R.G., YANG, W., Density Functional Theory of Atoms and Molecules. Oxford University Press, New York, (1989).

[6] HARTREE, D.R., Proc. Camb. Phil. Soc., **24** (1928) 89.

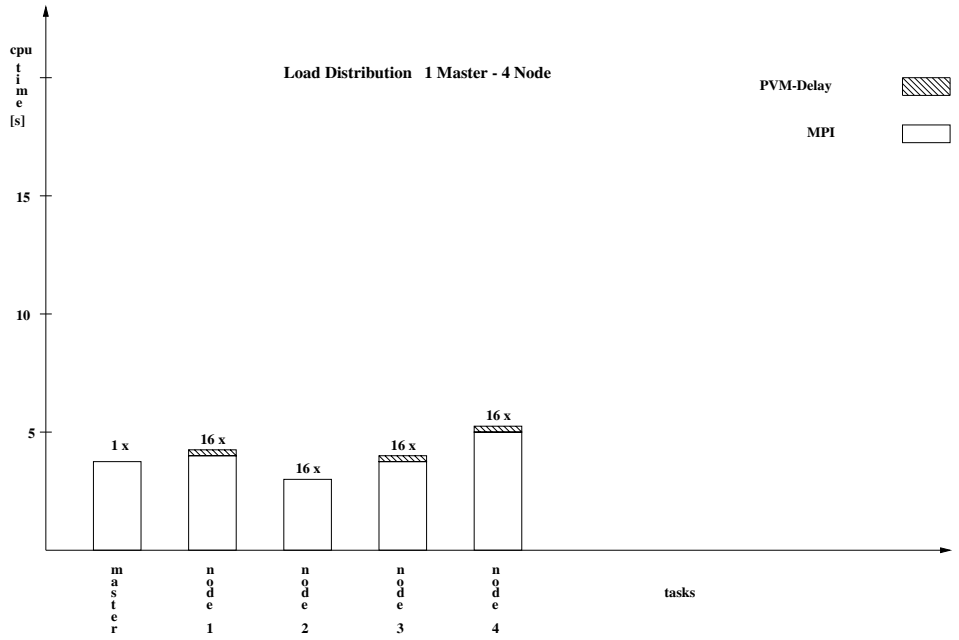


*Fig. 8.* Comparison of the cpu-time the different nodes have to spend on performing their partial work and indication of the effect of PVM-delay. ( 1 master 4 nodes )

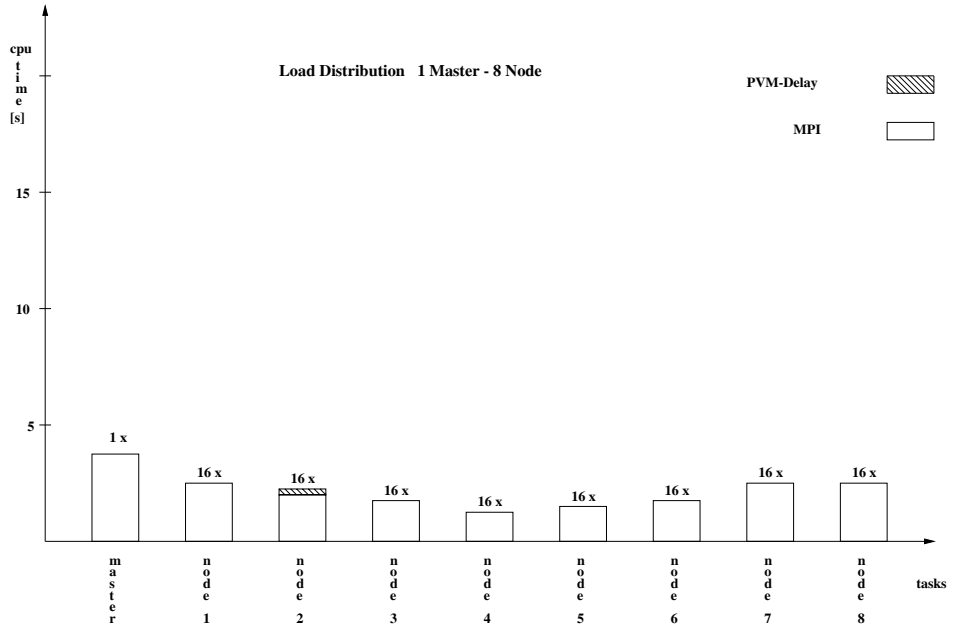


*Fig. 9.* Comparison of the cpu-time the different nodes have to spend on performing their partial work and indication of the effect of PVM-delay. ( 1 master 8 nodes )

[7] FOCK, V., Näherungsmethoden zur Lösung des Quantenmechanischen Mehrkörperproblems. Z. Physik, **61** (1930) 126 **62** (1930) 795.

[8] DAVIDSON, E.R., FELLER, D., Basis Set Selection for Molecular Calculations. Chem. Rev., **86** (1986) 681–696.

[9] SHAVITT, I., The Gaussian Function in Calculations of Statistical Mechanics and Quantum Mechanics. Methods in Computational Physics, academic, New York, **2** (1963) 1–44.

[10] SAUNDERS, V.R., An Introduction to Molecular Integral Evaluation. Computational Techniques in Quantum Chemistry and Molecular Physics, Reidel–Dordrecht (1975) 347–424.

[11] MCMURCHIE, L.E., DAVIDSON, E.R., One–and Two–Electron Integrals over Cartesian Gaussian Functions. J. Comp. Phys. **26** (1978) 218–231.

[12] OBARA, S., SAIKA, A., Efficient recursive computation of molecular integrals over Cartesian Gaussian functions. J. Chem. Phys. **84** (7) (1986) 3963–3974.

[13] HEHRE, W.J., DITCHFIELD, R., POPLE, J.A., J. Chem. Phys. **56** (1972) 2257.

[14] FRANCL, M.M., PETRO, W.J., HEHRE, W.J., BINKLEY, J.S., GORDON, M.S., DEFREES, D.J., POPLE, J.A., J. Chem. Phys. **77** (1982) 3654.

[15] GEIST, A., BEGUELIN, A., DONGARRA, J., JIANG, W., MANCHEK, R., SUNDERAM, V., PVM: Parallel Virtual Machine. A Users' Guide and Tutorial for Networked Parallel Computing MIT Press (1994).

[16] HÖFINGER, S., STEINHAUSER, O., ZINTERHOF, P., Performance Analysis and Derived Parallelization Strategy for a SCF Program at the Hartree Fock Level. Lect. Nt. Comp. Sc. **1557** (1999) 163.

[17] PULAY, P., Convergence Acceleration of Iterative Sequences. The Case of SCF Iteration. Chem. Phys. Letters **73** (1980) 393–398.

[18] PULAY, P., Improved SCF Convergence Acceleration. J. Comp. Chem. **3**, No.4 (1982) 556–560.

*Received:* May 15, 1999
*Accepted in revised form:* January 21, 2000

*Contact address:*
Siegfried Höfinger and Othmar Steinhauser
Institute for Theoretical Chemistry
Molecular Dynamics Group
University of Vienna
Währingerstr. 17, Ground Floor,
A-1090 Vienna
Austria
e-mail: {sh,os}@mdy.univie.ac.at
http://www.mdy.univie.ac.at

Peter Zinterhof
Institute for Mathematics
University of Salzburg
Hellbrunnerstr. 34
A-5020 Salzburg
Austria
e-mail: zinterhof@edvz.sbg.ac.at
http://www.mat.sbg.ac.at

SIEGFRIED HOEFINGER received the M.S. degree in biochemistry from the University of Vienna in 1996 and the Ph.D. degree in theoretical chemistry from the University of Vienna in 1998. In 1999 he was a postdoctroal research fellow at the IGBMC, the Institut de Genetique et de Biologie Moleculaire et Cellulaire in Strasbourg, where he worked in the lab of Dr. Thomas Simonson on the development of new algorithms for the large scale dielectric response in proteins. Currently he is a research fellow at the Institute for Theoretical Chemistry and Structural Biology of the University of Vienna. His scientific interests include parallel algorithms in quantum chemistry, high performance computing on multiprocessor architectures and algorithm design and analysis of programs used for ab-initio simulations in computational chemistry.

OTHMAR STEINHAUSER graduated from the University of Vienna in 1975 with a Ph.D. in physics. After promotion sub auspiciis praesidentis he worked as a research assistant at the Institute for Theoretical Chemistry, University of Vienna and became a DFG-assistent at the Institute for Physical Chemistry, University of Karlsruhe in 1979. After habilitation in theoretical chemistry in 1984 he became Assistant Professor at the Johannes-Gutenberg University of Mainz in 1987 and Full Professor of chemical molecular dynamics at the Institute for Theoretical Chemistry, University of Vienna in 1991. He is head of the Computing Center of the University of Vienna since 1992. His current research interests include computer simulation on the structure and dynamics of solvated biomolecules, algorithm design for the treatment of electrostatic interactions in biomolecules, multiprocessor computer architectures and parallel machines.

PETER ZINTERHOF is director of the Research Institute for Software Technology and chair of the Department of Scientific Computing at Salzburg University, Austria. He holds a full professorship of mathematics and theoretical computer science. His research interests include number-theoretical numerics, parallel and distributed processing, image processing, and stochastics. He is project leader of several national (FWF-funded) and international (INTAS-funded) research projects and is pursuing industrial cooperations.