Implementation of Parallel and Divide and Conquer Algorithms in DeFT (LCGTO-DF)

by

Roger T. Gallant

Department of Chemistry
University of Ottawa
10 Marie Curie Pr.
Ottawa, Ontario
K1N 6N5
CANADA

A thesis presented in partial fulfillment
of the requirements for a Master's degree (M.Sc.)
in chemistry

May 1996

ISBN 0-612-16440-3

Canada

UNIVERSITÉ D'OTTAWA
UNIVERSITY OF OTTAWA

Dedicated to the memory of my grandfather,
Harold Thomas MacDonald

Table of Contents

List of Tables

Table of Figures

List of Abbreviations

CPU	**Central Processing Unit**
DAC	**Divide and Conquer**
DF	**Density Functional**
ERI	**Electron Repulsion Integral**
FTP	**File Transfer Protocol**
HF	**Hartree-Fock**
KS	**Kohn-Sham**
LCGTO	**Linear Combination of Gaussian-Type Orbitals**
LSDA	**Local Spin Density Approximation**
PVM	**Parallel Virtual Machine**
PVMD	**Parallel Virtual Machine Daemon**
SCF	**Self-Consistent Field**
TID	**Task Identifier**
VWN	**Vosko-Nusair-Wilk**
WWW	**World Wide Web**
XC	**Exchange and Correlation**

Acknowledgments

I would like to thank my research supervisor, Alain St-Amant, for accepting me into his research group. Over the past couple of years he has provided a tremendous amount of help on this project. He has always made himself available to answer questions. I am not a religious person, but thank God for Alain's patience!

Of course, I must make note of the loving support of my wonderful parents, Helen and Roger. And also, their chequebook.

I must thank Sor Koon Goh, my co-worker, who also understands the enjoyment of computer programming in a 10'x14' room with no windows.

Finally, I must acknowledge my partner in crime here in Ottawa, Fred Crouch. He and his family, Monique, Christine and Nicholas have made my stay in Ottawa much more enjoyable. Especially Nicholas, my personal alarm clock.

Abstract

The program DeFT is an implementation of Density Functional Theory (DFT). The formalism that is used in this program is the linear combination of gaussian-type-orbitals (LCGTO). In this approach, there exists a method for the variational fitting of the density. Exchange-correlation is fitted on a numerical grid. It is these fitting procedures that reduce the formal scaling of LCGTO-DF to N^3 from N^4.

The contribution to the total energy from the Coulomb repulsion integrals and the exchange-correlation integrals causes the current bottleneck of DFT calculations. The parallelization of the three-centered two electron integrals should lead to improved efficiency. The use of the PVM communication package gives a way to a relatively simple means for transforming serial to parallel code. An advantage of using PVM is that it is portable to a large variety of architectures. This allows PVM to act as a harness between clusters of workstations. The parallel implementation of DeFT is discussed.

A divide-and-conquer implementation of LCGTO-DF is incorporated in DeFT. The use of Yang's density and density matrix formulations are incorporated into this scheme. The former is used for the fitting of the XC potential while the latter is used for the fitting of the charge density. This type of algorithm should achieve linear scaling for very large systems. Test calculations with the glycine heptapeptide demonstrate the errors to be expected from the introduction of the DAC approach. The use of buffer spaces is also examined in order to reduce the errors introduced by the DAC algorithm.

Introduction

Traditionally, the majority of *ab initio* programs use the Hartree-Fock (HF) approximation to solve the Schrödinger equation,

$$H\Psi = E\Psi .$$

In the Hartree-Fock approximation, Ψ is a single determinantal 3N-dimensional function, where N is the number of electrons in the system. If the system of interest is a large biomolecule, then the complexity of the wavefunction approach will become computationally too burdensome.

A strong point of the Hartree-Fock method is that it incorporates exact electron exchange energy. However, a major downfall is that it completely neglects electronic correlation. Correlation energy is defined as the difference between the true electronic energy and the calculated HF electronic energy. The use of correlated post- HF methods have corrected for the neglect of the correlation energy, but they come at a high computational expense. Hartree-Fock calculations have a formal scaling of N^4 and correlated post-HF calculations scale at least as N^5, where N is the number of electrons in the system. In order to reduce this cost, a new *ab initio* approach must be adopted.

Density functional (DF) theory [1-4] is becoming an ever increasingly popular quantum mechanical computational tool for chemists. In DF theory, molecular properties can be calculated by knowing one simple quantity, the electronic density. This method was proposed by Hohenberg and Kohn in 1964 [5]. The attractiveness of this approach may be attributed to the fact the electronic density, $\rho(r)$, is a simple 3-dimensional function which is also a physical observable. Compared to its HF counterparts, DF methods are relatively inexpensive to carry out. Density functional theory **includes** the effects of **exchange and correlation** in such a way that the same scaling problems do not arise as in Hartree-Fock based methods. However, practical DF calculations cannot be performed with the electronic density alone because the functional of the density is not known.

In this research, the DF program used is DeFT. It, like many other DF programs, uses the Kohn-Sham (KS) equations [6] as a vehicle for determining the energy of a molecule. The KS equations have the form of

$$\left\{-\frac{1}{2}\nabla^2 + v_{coul} + v_{xc}\right\}\psi_i(r) = \varepsilon_i\psi_i(r)$$

where v_{coul} and v_{xc} are the Coulomb and exchange-correlation (XC) potentials, respectively. $\psi_i(r)$ is a KS molecular orbital with the corresponding energy, $\varepsilon_i(r)$. Many techniques have been developed to solve the KS equations: for example, linear combination of atomic orbitals density functional (LCAO-DF) theory provides efficient and accurate results. This branches into two areas, programs which use Slater-type orbitals (STO) and programs which use Gaussian-type orbitals (GTO). The former appears in such software as the Amsterdam DF code [7], and the latter in deMon [8], DeFT [9], DGauss [3] and Gaussian 92/DFT [10]. Numerical atomic basis sets have also been used to solve the KS equations in DMol [11]. More recently, the KS equations have been solved with the use of plane wave-basis sets [12]. Also, basis set-free approaches, such as NUMOL [13], been used.

The program DeFT also permits the calculation of many molecular properties such as equilibrium geometries, vibrational frequencies, Raman frequencies, and dipole moments. The program DeFT was developed to run on many different architectures, such as Silicon Graphics, IBM RISC and VAX. And more recently, it has become desirable to have DeFT run on massively parallel computers.

Parallel computing is desirable for the study of large molecules by *ab initio* methods because it uses several computer processors simultaneously to solve a common problem. This provides a huge computational advantage over a single-processor computer. To better understand exactly what is parallel computing, consider the following analogy. If there was one person with a bucket of water trying to put out a fire, he or she is going to have a tough time by themselves. That one person can only keep refilling that bucket of

water at a certain rate. The process of extinguishing the fire will become much easier if several people were involved. This is providing that the additional people are coordinated in such a way that they do not interfere with each other. This example shows how a task can be accomplished more efficiently, if divided among multiple workers. The dividing of the responsibility among the workers to put out the fire is known as task partitioning. The passing of the duties to the workers requires communication. This is a rather simple concept which can be applied to computing problems. Of course, to perform such calculations it is essential to have a computer that has several processors or some type of harness that can make use of a network of single processor computers.

The harness used, in this research, is the Parallel Virtual Machine (PVM) developed by Sunderam and Geist [14] for use on supercomputers, homogeneous and heterogeneous clusters of workstations. In PVM, communication between processors, is achieved with the use of a daemon process and library routines. The daemon is a process that is always running in the background of a comp ter, in the case of PVM, it directs messages being passed between processors. The library routines give a set of commands for message-passing. The source code of DeFT was modified with PVM, so that parallel computing could be performed.

Only a small portion of the subroutines that comprise DeFT are responsible for the majority of the CPU time used. It is these subroutines that were targeted for modification by PVM for parallel computing. The subroutine that uses the most CPU time is gridwork.f. This subroutine is used for the construction of the electronic density on a grid, as well as, numerical integration and fitting of the exchange and correlation terms. Other subroutines that were modified with PVM for parallel computing were the electron repulsion and exchange-correlation integrals that arise in the KS equations. The use of a parallel algorithm for these time consuming subroutines should make subsequent DF calculations more efficient in multiprocessor environments.

Currently, the use of fitting procedures for the charge density and XC terms within an auxiliary basis makes the solution of the KS equations scale as N^3. In order to improve this scaling, a better algorithm for the solution to the KS equations is needed. Recently, Yang proposed a divide-and-conquer approach (DAC) for the solution to the KS equations [15]. In this approach, a large system may be divided into small subsystems. The KS operator is then projected onto the subsystem's basis and the equations are solved within this basis. Subsystem densities are then constructed and pieced together. It has been shown that this procedure will scale linearly with system size [15].

However, it would be useless to have a DAC solution for the KS equations if a DAC approach for the fitting procedures didn't exist as well. In this research, a divide-and-conquer method for the fitting of the charge density and the exchange-correlation terms is proposed. This is carried out by modifying Yang's density formulation [15] for the fitting procedure of the exchange-correlation terms and his density matrix formulation [16] for the fitting of the electronic density. This formalism will be outlined and the errors arising from the DAC fitting procedures are examined with tests on the extended conformation of the glycine heptapeptide. The effect of buffer atoms [15] is also studied.

Hohenberg-Kohn Theorem

In 1964 Honenberg and Kohn gave a proof [5] that the ground state energy of a molecule is a functional of the electronic density of the molecule.

$$E = E[\rho(r)] \tag{1}$$

The theorem states that the energy of a system is uniquely determined by the electronic density, $\rho(r)$, and that any other density, $\rho'(r)$, will give rise to a higher energy.

$$E[\rho'(r)] > E[\rho(r)] \tag{2}$$

This would provide a simple alternative to the Hartree-Fock method for performing variational, *ab initio* electronic structure calculations. It would greatly simplify calculations because it would be no longer necessary to use a 3-N dimensional wavefunction that describes the behavior of each electron in the N electron system as in a HF calculation. Instead, a three dimensional function, the total electronic density would be used, greatly reducing the complexity of an electronic structure calculation. With the exact energy functional of the total electronic density in hand, the variational principal could be employed in which a trial density could be inputted into an energy functional and thus a resultant energy obtained. This procedure would then be repeated until the energy of the system could no longer be lowered. At this point, if indeed the energy functional is exact, the result is the exact ground state energy of the molecule. Unfortunately, this energy functional is not known, rendering density functional theory seemingly useless.

However, in 1965, Kohn and Sham [6] provided a method to take advantage of the electronic density as the principle variable in *ab initio* quantum mechanical calculations. In this method, the electronic energy functional is decomposed into three components

$$E[\rho\ (r)] = T[\rho\ (r)] + U[\rho\ (r)] + E_{xc}\ [\rho\ (r)] \tag{3}$$

where $T[\rho(\mathbf{r})]$ is the kinetic energy of the system of non-interacting particles of the same density, $\rho(\mathbf{r})$, $U[\rho(\mathbf{r})]$ is the classical electrostatic energy of all the electrons and nuclei, and $E_{XC}[\rho(\mathbf{r})]$ is the exchange-correlation energy of the system.

Kohn-Sham Approach

Kohn and Sham introduced molecular orbitals [6] into the problem so as to make the accurate calculation of the kinetic energy possible. The total electronic density can be expressed, for an N electron system, as the summation of the squares of the occupied Kohn-Sham (KS) molecular orbitals,

$$\rho(r) = \sum_{i}^{N} |\psi_i(r)|^2 . \tag{4}$$

This allows $T[\rho(\mathbf{r})]$ to be expressed as

$$T[\rho(r)] = \sum_{i}^{N} \int \psi_i(r) \frac{-\nabla^2}{2} \psi_i(r) dr . \tag{5}$$

This form of $T[\rho(\mathbf{r})]$ creates two deficiencies in the algorithm. This is the exact expression for the kinetic energy only if the electron motions are uncorrelated. Compensation for this approximation will be discussed later. The second deficiency is that the introduction of Kohn-Sham orbitals no longer ensures that this is a true density functional problem, because the energy will now depend on the nature of the occupied KS molecular orbitals. Kinetic energy functionals that make no use of KS orbitals are unfortunately not accurate enough for applications in chemistry.

The classical Coulomb energy, $U[\rho(r)]$, is

$$U[\rho(r)] = \sum_{A}^{nuclei} \int \frac{-Z_A \rho(r)}{|r - R_A|} dr + \frac{1}{2} \iint \frac{\rho(r)\rho(r')}{|r - r'|} dr dr' \tag{6}$$

where the nuclear charge, position of the electron and position of the nucleus are given by Z_A, r and R_A, respectively. At this point, we will not address the nature of $E_{xc}[\rho(r)]$. Suffice it to say, it contains all the remaining effects of exchange and correlation not handled by the first two terms.

Once again, recall that the energy functional must be minimized by the true density

$$\frac{\delta E[\rho(r)]}{\delta \rho(r)} = 0 \tag{7}$$

subject to the constraint that $\rho(r)$ be normalized to the total number of electrons, N,

$$\int \rho\,(r)dr = N\,. \tag{8}$$

From this, the one electron Kohn-Sham equations are obtained,

$$\left\{\frac{-\nabla^2}{2} - \sum_{A}^{atoms}\frac{Z_A}{|r-R_A|} + \int\frac{\rho\,(r')}{|r-r'|}dr' + \frac{\delta\,E_{xc}\,[\rho\,(r)]}{\delta\,\rho\,(r)}\right\}\psi_i(r) = \varepsilon_i\,\psi_i(r)\,. \tag{9}$$

From these equations, it is seen that an iterative procedure most be adopted. The KS orbitals depend on the density but the density is in turn determined by the KS orbitals. One takes an initial guess for $\rho(r)$, inserts it into the KS equations and obtains a set of KS orbitals, $\psi_i(r)$. The new KS orbitals are then used to construct new guesses at $\rho(r)$ and the procedure is then repeated until self-consistency is achieved, meaning the same KS orbitals and densities are being reproduced.

In the preceding discussion, there was no discussion of the exchange and correlation energy functional nor the XC potential, v_{XC}, which is simply the functional derivative of $E_{XC}[\rho(r)]$ with respect to $\rho(r)$,

$$v_{xc}(r) = \frac{\delta\,E_{xc}\,[\rho\,(r)]}{\delta\,\rho\,(r)}\,. \tag{10}$$

The true exchange-correlation energy, $E_{XC}[\rho(r)]$, which contains all the effects of exchange and correlation including that neglected by $T[\rho(r)]$ is not known, but approximations can be made so that it is possible to use the Kohn-Sham method. These approximations become the limiting factor for the quality of any DF calculation, but relatively good results can be made by simple approximations to the XC energy functional. The fact that the limiting factor of any DF calculation lies within the approximation of the XC energy functional justifies the focus in modern DF on better approximations to the XC energy functional [17,18]. One of the simplest approximations is the Local Spin Density Approximation (LSDA) [19]. Despite its simplicity, the LSDA can yield results that are comparable to those achieved with Hartree-Fock calculations [20].

Exchange and Correlation Functionals

The Local Spin Density Approximation approximates the XC energy in the following manner,

$$E_{xc}[\rho(r)] = \int \rho(r)\varepsilon_{xc}[\rho^{\alpha}(r),\rho^{\beta}(r)]dr \tag{11}$$

where $\varepsilon_{XC}[\rho^{\alpha}(r), \rho^{\beta}(r)]$ is the XC energy density at a point r with densities $\rho^{\alpha}(r)$ and $\rho^{\beta}(r)$, and α and β refers to spin up and spin down electrons. The exchange-correlation potential then becomes

$$v^{\sigma}_{xc}(r) = \frac{\delta \int \rho(r)\,\varepsilon_{xc}(r)dr}{\delta\ \rho^{\sigma}(r)} = \varepsilon_{xc}[\rho^{\sigma}(r)] + \rho^{\sigma}(r)\frac{\delta\ \varepsilon_{xc}[\rho^{\sigma}(r)]}{\delta\ \rho^{\sigma}(r)} \quad \text{where } \sigma = \alpha,\beta \tag{12}$$

In this case, it becomes evident that E_{XC} and v^{σ}_{XC} are dependent on the quality of the expression for ε_{XC}. For a LSDA calculation, DeFT uses the functional of Vosko, Wilk and Nusair (VWN) [19]. The VWN-LSDA for the XC potential does not use any experimental data. The XC potential is obtained by fitting the quantum Monte Carlo data of Ceperley and Alder [21] on homogeneous electron gases. The calculations of Ceperley and Alder were for a series of electron gases with different total and net spin densities.

Since the work of Dirac [22], it has been known that the exchange energy for a homogeneous electron gas is

$$E_x = -\frac{3}{4}\left(\frac{6}{\pi}\right)^{\frac{1}{3}}\int \rho(r)^{\frac{4}{3}}dr. \tag{13}$$

This leaves the correlation energy, the other component of the XC energy, unknown. This component can be extracted from the Monte Carlo calculations. The work of Ceperley and Alder left a database of information that VWN subsequently fitted with the use of an analytical function. Perdew and Zunger [23] also fitted the same data but with a different analytical function. There is no clear advantage of using one function over the other because they both accurately fit the same data. Consequently, the algorithm for the evaluation of the XC potential is quite simple. For a point on a grid in which the XC term is to be fitted, the spin densities, $\rho^{\alpha}(r)$ and $\rho^{\beta}(r)$, are evaluated. These are then transferred

to a subroutine that calculates v_{xc} and ε_{xc} with the analytical function of VWN. Since the Monte Carlo calculations were essentially exact, the parameterization thus brought the LSDA to its limit.

The LSDA approach consistently yields results comparable and sometimes better than that of Hartree-Fock (HF) and correlated post-HF calculations [3,10,24]. This may be attributed to the built-in contributions of correlation effects to the total energy expression. However, it has been shown that LSDA usually underestimates the exchange energy by 10% [17] and overestimates the correlation energy by 100% [25]. These errors do not seem to seriously effect the shape of potential energy surfaces. As well, equilibrium geometries and force constants are well reproduced. However, they do seriously affect binding energies [10]. The systematic overestimation of binding energies causes great problems for hydrogen bonded systems [26] and van der Waals complexes [27] where calculated geometries often differ quite significantly from experiment. This approximation would, therefore, not be suitable for biological molecules in which hydrogen bonding plays a huge role.

To correct for this imperfection in the LSDA, gradient corrected XC potentials have been adopted. Such an XC energy functional still has roughly the same form as it previously did, but $\varepsilon_{xc}(r)$ is also a local functional depending on $\rho^{\sigma}(r)$ *and* its gradient, $\nabla\rho^{\sigma}(r)$. The new functional depends on both $\rho^{\sigma}(r)$ and $\nabla\rho^{\sigma}(r)$ as follows,

$$E_{xc}[\rho(r)] = \int \rho(r)\varepsilon_{xc}[\rho^{\sigma}(r), \nabla\rho^{\sigma}(r)]dr. \qquad (14)$$

Although this equation looks very similar to that of the LSDA approach, it becomes much more difficult conceptually. The simplicity of the LSDA model is lost. Several gradient corrected functionals have been proposed, but they often only deal with one component of the exchange and correlation energy. Popular gradient corrected exchange functionals are those of Perdew and Wang [28] and Becke [17]. As for gradient corrected correlation functions, the most widely used are those of Perdew [25] and Lee, Yang, and Parr [18].

Finally, Perdew *et al.* have developed a combined exchange and correlation functional [29]. Various gradient-corrected XC functionals may be constructed by combining the various functionals proposed for each component of the XC energy.

Gradient corrections add to the computational burden associated with a calculation because it is necessary to calculate not only $\rho^{\sigma}(r)$, but also $\nabla\rho^{\sigma}(r)$, as well as, the second derivative of $\rho^{\sigma}(r)$ [30]. Equation (12) gives an expression for $v^{\sigma}_{xc}(r)$ that also requires the second derivative of $\rho^{\sigma}(r)$ when gradient corrections are applied. Fortunately, it is not necessary to calculate the gradient corrections at each iteration of the SCF cycle. During the self-consistent field procedure, it is often sufficient to use only the LSDA functional and once convergence is achieved, the gradient corrections are added in a perturbative fashion. The perturbative gradient correction to the LSDA almost provides the same degree of improvement over the LSDA as does the fully self-consistent gradient correction [31].

A final note on gradient corrections: they do remedy the overestimation of binding energies, allowing a better treatment of weakly bound systems. However, gradient corrections rarely warrant the additional cost in the calculation of properties that are already handled well by the LSDA. For simple main group molecules, geometries, vibrational frequencies and dipole moments with gradient corrections are only marginally better than the LSDA results.

Linear Combination of Gaussian Type Orbital Method

In DeFT, the linear combination of Gaussian type orbitals method is employed, where the Kohn-Sham orbitals are expanded as

$$\psi_i(r) = \sum_{\nu}^{N} C_{\nu i}\chi_{\nu}(r), \tag{15}$$

where N is the number of functions in the basis, C_{vi} is the set of Kohn-Sham orbital expansion coefficients for the i^{th} molecular orbital and $\chi_v(r)$ is a contracted gaussian basis function. If equation (15) is inserted into the Kohn-Sham equation,

$$H\left(\sum_{v}^{N} C_{vi}\chi_v(r)\right) = \varepsilon_i\left(\sum_{v}^{N} C_{vi}\chi_v(r)\right). \tag{16}$$

Multiplying through by an arbitrary basis function, $\chi_\mu(r)$, and integrating over all of space yields

$$\sum_{v}^{N} C_{vi}\int \chi_\mu(r) H\chi_v(r)dr = \varepsilon_i \sum_{v}^{N} C_{vi}\int \chi_\mu(r)\chi_v(r)dr. \tag{17}$$

Since the above is true for any basis function, $\chi_\mu(r)$, it can be recast in the matrix form,

$$\mathbf{HC=SC\varepsilon} \tag{18}$$

where **H** is the Kohn-Sham matrix and its elements $H_{\mu\nu}$ are given as follows

$$H_{\mu\nu} = \int \chi_\mu(r) H\chi_v(r)dr \tag{19}$$

or

$$H_{\mu\nu} = \int \chi_\mu(r)\left\{\frac{-\nabla^2}{2} - \sum_{A}^{atoms}\frac{Z_A}{|r-R_A|} + \int\frac{\rho(r')}{|r-r'|}dr' + v_{xc}(r)\right\}\chi_v(r)dr \tag{20}$$

and **S** is the overlap matrix with its elements $S_{\mu\nu}$ given by,

$$S_{\mu\nu} = \int \chi_\mu(r)\chi_v(r)dr \tag{21}$$

Upon inspection of the KS matrix elements, it is possible to see that the Coulomb repulsion contribution will be the most computationally intensive. The electron-electron repulsion term is:

$$\iint \frac{\chi_\mu(r)\chi_v(r)\rho(r')}{|r-r'|}drdr' \tag{22}$$

keeping in mind that

$$\rho(r') = \sum_{i}^{N}|\psi_i(r')|^2 = \sum_{\sigma}^{k}\sum_{\lambda}^{k} P_{\sigma\lambda}\chi_\sigma(r')\chi_\lambda(r'). \tag{23}$$

The equation for electron-electron repulsion then becomes

$$\sum_{\sigma}^{k}\sum_{\lambda}^{k} P_{\sigma\lambda} \iint \frac{\chi_{\mu}(r)\chi_{\nu}(r)\chi_{\sigma}(r')\chi_{\lambda}(r')}{|r-r'|} dr dr' . \tag{24}$$

The number of these four-centered two-electron integrals which scale formally as N^4 would, seemingly, give the same bottle-neck as that of a Hartree-Fock calculation. In fact some DF packages, such as Gaussian 92/DFT, explicitly evaluate these integrals. However, DeFT uses auxiliary basis functions to fit the electronic density. The fitted density, $\tilde{\rho}(r)$, is given by

$$\rho(r') \approx \tilde{\rho}(r') = \sum_{k} c_k \chi'_k(r') . \tag{25}$$

In this case, $\chi'_k(r')$ is an uncontracted Gaussian function and the exponents will roughly be double of the exponents of those used in the orbital basis. It is conceptually easy to understand why the exponents are double: the auxiliary functions are used to fit the density, i.e., the sum of the square moduli of the occupied KS orbitals.

The coefficients of the auxiliary functions can be determined by one of two methods. The first method is a least squares fit of the density, where

$$\int [\rho(r) - \tilde{\rho}(r)]^2 dr \tag{26}$$

is minimized by the method set out by Baerends, Ellis, and Ros [7], who used this approach for Slater functions. For Gaussian functions, Sambe and Felton [32] used this approach to obtain the expression for the fit coefficients. The second method instead of minimizing the error on the density, minimizes the error in the electron-electron repulsion energy,

$$\iint \frac{[\rho(r) - \tilde{\rho}(r)][\rho(r) - \tilde{\rho}(r)]}{|r-r'|} dr dr' . \tag{27}$$

The precise equations for auxiliary fitting coefficients are outlined by Dunlap, Conolly, and Sabin [33]. It is now possible to rewrite the four-centered two-electron equation as a three-centered two-electron equation as follows

$$\sum_k c_k \iint \frac{\chi_\mu(r)\chi_\nu(r)\chi_k(r)}{|r-r'|} dr dr' . \tag{28}$$

The formal scaling of the Coulomb repulsion term now drops from N^4 to N^3, which becomes a tremendous saving in CPU time as N gets larger.

In DF theory, the exchange-correlation component of the KS matrix is given by,

$$H_{\mu\nu}^{xc} = \int \chi_\mu(r) v_{xc}(r) \chi_\nu(r) dr . \tag{29}$$

However, this expression can not be solved analytically because of the complexity of the XC functionals that generate $v_{XC}(r)$ from $\rho(r)$. In such a case, numerical grids (discussed in detail later) are employed where $v_{XC}(r)$ is calculated at each grid point. For gradient-corrected calculations, it becomes necessary to calculate $\rho(r)$ at each grid point, as well as, its first and second derivatives with respect to x, y, and z.

There are two approaches to calculating the exchange-correlation potential using a numerical grid that are used in today's DF codes. The first method is a straightforward numerical integration of the right hand side of equation (29),

$$H_{\mu\nu}^{xc} \approx \sum_l^{points} \chi_\mu(R_l) v_{xc}(R_l) \chi_\nu(R_l) W_l \tag{30}$$

This approach is found in the Amsterdam code [7], DMol [11], and in Gaussian 92/DFT [10]. An alternative approach is the fitting of the exchange-correlation potential. It has the same form as the charge density fit,

$$v_{xc}(r) \approx \tilde{v}_{xc}(r) = \sum_l^M b_l \chi_l(r) \tag{31}$$

where $\chi_l(r)$ is a second set of M auxiliary fitting functions, which like the charge density fitting basis are uncontracted s, p, and d gaussians [34]. In Slater's Xα theory, $v_{XC}(r)$ is

proportional to $\rho(r)^{1/3}$ [35]. Therefore, the exponents in the exchange correlation fitting basis should be one third of those found in the charge density fitting basis. The fit coefficients are found by a numerical least squares fitting procedure which minimizes

$$\sum_{I}^{points}[v_{xc}(R_I)-\tilde{v}_{xc}(R_I)]^2 W_I . \tag{32}$$

With the fitted XC potential now in hand, the matrix elements can be analytically integrated and are given by the following three-centered overlap integrals

$$H_{\mu\nu}^{xc} \approx \int \chi_\mu(r)\tilde{v}_{xc}(r)\chi_\nu(r)dr = \sum_{l}^{M} b_l \int \chi_\mu(r)\chi_\nu(r)\chi_l^{\cdot}(r)dr \tag{33}$$

Total Energy

When evaluating the total energy of the system, it is the sum of the orbital energies obtained from the KS equations plus correction factors. The total energy is expressed as

$$E = 2\sum_{i}^{N/2}\varepsilon_i - \frac{1}{2}\iint\frac{\rho(r)\rho(r')}{|r-r'|}drdr' + \int\rho(r)[\varepsilon_{xc}(r)-v_{xc}(r)] + \sum_{A}^{atoms}\sum_{B<A}\frac{Z_A Z_B}{|R_A-R_B|} \tag{34}$$

The second term is to offset the double counting of the electron-electron repulsions in the sum of the orbital energies. The third term corrects for the fact that the KS equations contain the exchange-correlation potential but the exchange-correlation density appears in the XC energy. The last term is simply the internuclear repulsion.

The second term that deals with the electron-electron repulsion should be expressed as,

$$-\frac{1}{2}\iint\frac{\rho(r)\rho(r')}{|r-r'|}drdr' \approx -\frac{1}{2}\iint\frac{\rho(r)\tilde{\rho}(r')}{|r-r'|}drdr' =$$

$$-\frac{1}{2}\sum_{\mu}^{N}\sum_{\nu}^{N}P_{\mu\nu}\sum_{k}^{M}c_k\iint\frac{\chi_\mu(r)\chi_\nu(r)\chi_k^{\cdot}(r')}{|r-r'|}drdr' . \tag{35}$$

However, this term is in fact expressed, within the LCGTO-DF formalism, as

$$-\frac{1}{2}\iint \frac{\rho(r)\tilde{\rho}(r')}{|r-r'|}drdr' \approx -\frac{1}{2}\iint \frac{\tilde{\rho}(r)\tilde{\rho}(r')}{|r-r'|}drdr' =$$

$$-\frac{1}{2}\sum_{k}^{M}\sum_{k'}^{M} c_{k'}c_{k}\iint \frac{\chi_{k'}(r)\chi_{k}(r')}{|r-r'|}drdr' \tag{36}$$

Expressing the total energy in this fashion makes the fitting procedure of the electronic density variational. However, a perfectly fitted density will yield an energy that serves as an *upper* bound.

The third term of equation (34) is used to remove the contribution of $v_{XC}(r)$ to the sum of the KS orbital eigenvalues. In DeFT, $\varepsilon_{XC}(r)$ is fitted with the same basis as $v_{XC}(r)$,

$$\varepsilon_{xc}(r) \approx \tilde{\varepsilon}_{xc}(r) = \sum_{l}^{M} e_l \chi_l(r), \tag{37}$$

where e_l is the set of fit coefficients for the exchange-correlation energy density. This results in the following approximation to the third term,

$$\int \rho(r)[\varepsilon_{xc}(r) - v_{xc}(r)]dr \approx \int \rho(r)[\tilde{\varepsilon}_{xc}(r) - \tilde{v}_{xc}(r)]dr$$

$$= \sum_{\mu}^{N}\sum_{\nu}^{N} P_{\mu\nu}\sum_{l}^{M}(e_l - b_l)\int \chi_\mu(r)\chi_\nu(r)\chi_l(r)dr. \tag{38}$$

This correction to the eigenvalues of the KS equations is performed at each iteration during the SCF procedure at minimal computational expense. The correction in this form is computationally inexpensive. However, upon SCF convergence, a more rigorous treatment of the expression in equation (38) is desired. The more exact treatment is performed with the use of an augmented grid for the numerical integration of

$$\int \rho(r)[\varepsilon_{xc}(r) - v_{xc}(r)]dr \approx \sum_{\mu}^{N}\sum_{\nu}^{N} P_{\mu\nu}\sum_{I}^{points}\chi_\mu(R_I)\chi_\nu(R_I)\varepsilon_{xc}(R_I)W_I - \int \rho(r)\tilde{v}_{xc}(r)dr. \tag{39}$$

This yields a total energy which is less prone to grid noise [20]. This calculation requires a greater computational effort, but it is only performed once at the end of the SCF procedure and does not impact too much on the overall cost of an SCF procedure.

Parallel Virtual Machine (PVM)

Presently, a popular method for parallel computing is the Communicating Sequential Process. In this model, each process has its own memory and works with other processes by passing messages. There are now several packages that use this structure, in particular one system is a parallel virtual machine or PVM developed by Sunderam and Geist [14]. A great advantage of PVM software is that it supports a heterogeneous network of parallel and serial computers that can be user defined to create one large distributed-memory computer or a so called virtual machine. This will allow the combination of several workstations with the possibility of a computing power equivalent to that of a conventional super-computer.

Serial codes can be modified with a minimum amount of recoding by replacing the serial parts of the code that could benefit most by parallelism. These parts are the most time consuming sections of the code which are replaced by parallel versions that run on many processors. The remainder of the serial code remains in the serial form running on one processor as the 'master' program while the parallel section is run as a collection of 'slave' programs that are spawned by the 'master'. When the parallel section is completed, the 'slaves' return the information to the master program and the slave processes terminate. Applications may be written in either FORTRAN 77 or C and are parallelized using message passing constructs that send and receive messages so that the multiple tasks work together to solve a problem. The PVM system is capable of doing this through its two main components.

The first is the PVM daemon (pvmd) which runs simultaneously on each of the computers in the virtual machine. This background, perpetual, pvmd process controls message-passing. In PVM, an integer task identifier or tid is assigned to each process that is created so that the user application can identify each process in the system. The use of tids was found to be the most efficient method, by the developers, of identifying processes in the virtual machine. Since the tids must be unique across the entire virtual machine, the

PVM daemon assigns the values to ensure that there can be no errors created by user defined assignment.

The second part of the package is the set of library routines that are called from the application which contain routines for message passing, spawning processes, coordinating tasks, and modifying the virtual machine (See Appendix). PVM was designed so that any task can send messages to any other task with no limitation on the size or number of messages that are sent.

Building PVM

After obtaining a copy of the PVM software, it is necessary to configure the machine to ensure that it will run properly. Unpacking of the source code in the $HOME directory will create a directory called pvm3. The file Cshrc.stub located in the directory $HOME/pvm3/lib must be copied into the .cshrc file immediately after the path so as to determine what type of machine it is on and adds the location of PVM to your path.

It is necessary to build the software to support the architecture of the machine on which the software resides. Architectures supported are listed in Table 1. Fortunately this is done automatically by typing **make** in the pvm3 directory. The makefile automatically determines the architecture and builds the necessary library files and places them in the directory $HOME/pvm3/lib/ARCH.

Starting PVM

Starting PVM on a local machine can be done by executing **pvm** from the directory $HOME/pvm3/lib. The console (pvm), which automatically starts the PVM daemon (pvmd3), may be started and stopped many times but typically the user only starts one console. The console allows the user to add and delete machines from the virtual machine,

as well as, starting and stopping of PVM processes. By interactively adding and deleting machines, the user may tailor the virtual machine to a specific application. Perhaps some sections of a FORTRAN or C code are highly vectorizable, then the introduction of a vector computer to the virtual machine could accommodate this section of code. Likewise, scalar code may be best handled by the introduction of a single-processor workstation to the virtual machine. PVM allows the flexibility of changing the configuration of the virtual machine at any time.

Compiling and Running PVM Applications

Once PVM is running, an application using PVM can be started from a UNIX prompt on any machine in the virtual machine. PVM looks for executable files in the default directory $HOME/pvm3/bin/ARCH. An example: if a PVM application spawns a task called gradcdsspvm on an IBM RS6000 called theory, on the machine theory there should exist an executable file $HOME/pvm3/bin/RS6K/gradcdsspvm. If a SGI machine exists in the virtual machine, then the SGI binary file must exist and be found in $HOME/pvm3/bin/SGI/gradcdsspvm. The PVM daemon knows where to go for executables, since the type of machine is specified at the time it is added to the virtual machine. A FORTRAN or C program that uses PVM calls needs to be linked to the library files libpvm3.a, libfpvm3.a, and libgpvm3.a.

ARCH	Machine	ARCH	Machine
AFX8	Alliant FX/8	**IPSC2**	Intel iPSC/2 386
ALPHA	DEC Alpha	**KSR1**	Kendall Square KSR-1
BAL	Sequent Balance	**NEXT**	NeXT
BSD386	BBN Butterfly TC2000	**PGON**	Intel Paragon
CM2	Thinking Machines CM2	**PMAX**	DECstation 3100, 5100
CM5	Thinking Machines CM5	**RS6K**	IBM RS6000
CNVX	Convex C-series	**RT**	IBM RT
CNVXN	Convex C-series native mode	**SGI**	Silicon Graphics IRIS
CRAY	C-90, YMP, Cray-2	**SUN3**	Sun3
CRAYSMP	Cray S-MP	**SUN4**	Sun4 SPARCstation
DGAV	Data General Aviion	**SYMM**	Sequent Symmetry
HP300	HP-9000 model 300	**TITN**	Stardent Titan
HPPA	HP-9000 PA-RISC	**UVAX**	DEC microVAX
I860	Intel iPSC/860		

Table 1. Architecture types supported by PVM

Example

In the case of the applications that were developed in this research, the applications were organized in such a way that a master/slave scheme was employed. In a master/slave model the master program spawns and coordinates a user-defined number of slaves that perform the computations. To provide an example of the general structure of a master/slave scheme, segments from a section of code from DeFT, where chrgss (subroutine that calculates three-centred electron repulsion integrals where the first primitive pair is of the type *ss*) is the master program and chrgsspvm is the slave, are given in Figures 1 and 2.

In Figure 1, the master program, it is not important to know what the variables represent. It is more important to understand how PVM uses message-passing for parallel computing. The subroutine chosen as the example for the master program calculates a contribution to the KS matrix made by Coulomb repulsion. It is a recurrence relation for the three-centred two-electron Coulomb repulsion (three-centred because the electronic density was fitted). The first thing that must be done in any PVM application is to include the PVM header file so PVM parameters are defined in the FORTRAN or C code.

The first call of any PVM application must be pvmfmytid() so as to enroll an application in PVM. Upon its call, a unique tid is created for this application. Following this step, pvmfspawn() is used to start nprocedures. This is a user-defined number of nodes to be used in a calculation in which a copy of the slave program chrgsspvm runs on each node. Now that the slave process has been initiated, data must be sent to them for processing. The command pvmfinitsend() creates a message buffer that packs an array of data that must be sent to the slave from the master program. In the case of this master/slave scheme, the first message buffer that is sent, contains information needed by each and every slave running on the processors in the virtual machine, so the pvmfmcast() subroutine is used to multi-cast to all the nodes. The pvmfpack() subroutine is used to pack the data into the message buffer to be sent to the slave.

In the second section of the master program in Figure 1, in the loop numbered 1001, the three lines of code before the second pvmfinitsend() are used to divide ncds (the number of *s* auxiliary basis function used in the fit of the charge density) over the number of slaves running. The code also ensures that in the case of an unequal division that the remainder is sent to the slave as opposed to being simply truncated. This integer is divided because it acts as a loop counter in the slave program and division allows this loop to be broken into smaller loops with each smaller loop running on a different processor in the virtual machine. When each loop is finished calculating its contribution of the electron repulsion to the Kohn-Sham matrix, the data is sent from the slave back to the master

program to be summed. The pvmfrecv() subroutine is used to receive the incoming message buffers from the various slave processes, which are subsequently unpacked with the pvmfunpack() subroutine and the data is added to the Kohn-Sham matrix.

The slave program, chrgsspvm.f, is shown in Figure 2. It has a similar structure to that of the master program starting with inclusion of the PVM header file followed by pvmfmytid() to enroll the program as a PVM process just as the master program did. In this case, however, the slave program has the command pvmfparent() so as to identify the slave with a particular master program. It is conceivable that more than one master program may be running on the virtual machine and it is essential to make sure that a slave receives messages only from its 'master'.

The pvmfrecv() subroutine is used to receive the incoming message buffer from the master program as previously mentioned. The first block that is received and unpacked is the data that was multi-casted to all slaves running on the virtual machine. Recalling that this was done because all the slaves needed this common data to perform the desired calculations. The second block of data received is the data that was divided so that each segment could be sent to different processors.

After the data is received the slave can now perform the calculation it was designed to do. In this case, calculating the portions of the contribution of electron repulsion to the Kohn-Sham matrix. The variable istart was used to adjust the starting point of the loop, not the size, so that any array that depends on the value naux (an integer expressing the size of the loop associated with the number of auxiliary basis functions used in the charge density fitting) would be pointed to the correct position in an array. After the calculations, the pertinent data is then packed up and sent back to the master program. The final step in the slave process is to use the pvmfexit() command to let the PVM daemon know that this process is leaving PVM.

```
      subroutine chrgss
c     INCLUDE FORTRAN PVM HEADER FILE
      include 'fpvm3.h'
c     ENROLL THIS PROGRAM IN PVM
      call mytid(mytid)
c     INITIATE NPROCEDURES OF CHRGSSPVM PROGRAM
      call pvmfspawn('chrgsspvm',PVMARCH,'RS6K',
     &                nprocedures,itid,numt)
c     BROADCAST DATA TO ALL NODES
      call pvmfinitsend(PVMRAW,info)
      call pvmfpack(INTEGER4,m,1,1,info)
      call pvmfpack(INTEGER4,n,1,1,info)
      call pvmfpack(REAL8,px,n,1,info)
      call pvmfpack(REAL8,py,n,1,info)
      call pvmfpack(REAL8,pz,n,1,info)
      call pvmfmcast(nprocedures,itid,1,info)

      do 1001 i=1,nprocedures

      ncds1=ncds/nprocedures
      istart(i)=(i-1)*ncds1
      if (i.eq.nprocedures) then
          ncds1=ncds-(nprocedures-1)*ncds1

c     BROADCAST OF SEGMENTS TO NODES
      call pvmfinitsend(PVMRAW,info)
      call pvmfpack(INTEGER4,i,1,1,info)
      call pvmfpack(INTEGER4,istart(i),1,1,info)
      call pvmfpack(INTEGER4,ncds1,1,1,info)
      call pvmfsend(itid(i),1,info)
 1001 continue

c     RECEIVING RESULTS FROM NODES
      call pvmfrecv(itid(i),-1,info)
      call pvmfunpack(REAL8,fock,ndim,1,info)

      return
      end
```

Figure 1. FORTRAN example of the master program, chrgss.f.

```
      program chrgsspvm
c     INCLUDE FORTRAN PVM HEADER FILE
      include 'fpvm3.h'
c     ENROLL THIS PROGRAM IN PVM
      call pvmfmytid(mytid)
c     GET THE MASTER'S TID
      call pvmfparent(mptid)

c     RECEIVE DATA FROM MASTER
      call pvmfrecv(mptid,1,info)
      call pvmfunpack(INTEGER4,m,1,1,info)
      call pvmfunpack(INTEGER4,n,1,1,info)
      call pvmfunpack(REAL8,px,n,1,info)
      call pvmfunpack(REAL8,py,n,1,info)
      call pvmfunpack(REAL8,pz,n,1,info)

c     RECEIVE SEGMENT FROM MASTER
      call pvmfrecv(mptid,1,info)
      call pvmfunpack(INTEGER4,i,1,1,info)
      call pvmfunpack(INTEGER4,istart(i),1,1,info)
      call pvmfunpack(INTEGER4,ncds1,1,1,info)
c     DO CALCULATIONS
      do 1001 naux=istart+1,istart+ncds
      [Calculations of ERI contributions dependent on
       loop size]
 1001 continue
c     SEND RESULTS TO MASTER
      call pvmfinitsend(PVMDEFAULT,info)
      call pvmfpack(REAL8,fock,ndim,1,info)
      call pvmfsend(mptid,1,info)
c     PROGRAM FINISHED, LEAVE PVM
      call pvmfexit()

      stop
      end
```

Figure 2. FORTRAN example of the slave program, chrgsspvm.f.

Performance Considerations

Almost any programming structure may be implemented by a PVM user. However, not every message-passing model is necessarily efficient. The following are some of the main considerations a programmer must consider when developing a PVM application.

The number of messages sent to a processor must be considered. A number of bytes may be sent in many small messages or a few large messages. Sending large messages may reduce the time spent passing messages from master to slave programs, but does not mean the overall execution time will decrease as processors must stand idle during their transmission. A happy medium between time spent on communication and calculation must be found.

An application may be better suited for functional parallelism rather than data parallelism. In this case, functional parallelism occurs when different machines in the virtual machine perform different tasks. This is beneficial when, perhaps, part of a problem is more suited for a vector computer. Maybe in the same problem, a graphics workstation is needed for visualization of data in real time. This kind of problem is better handled by using a functional parallel algorithm. The data parallelism model is applied to problems where data is partitioned and sent to all the machines in the virtual machine, so that similar operations are performed until the problem is solved. This is the kind of parallelism that was employed in this research.

Finally, the type and physical location of a machine in the virtual machine plays a significant role in performance consideration. It is a fact that different workstations will have differing computational rates. Workstations from different manufacturers can have as much as two orders of magnitude in difference of power. For several workstations to be part of the virtual machine, they must be on a network. The distance between machines can cause delays when messages are being passed. The amount of traffic on the network will also be a factor. Some form of load balancing must be employed so as to take into account all of the above mentioned factors that affect performance. When several tasks are

spawned across the virtual machine, it is desirable to have all the tasks come to completion simultaneously. It becomes very inefficient to have certain processors sitting idle while others are still performing computations.

Numerical Grid

In many cases, integration cannot be carried out analytically. In such cases, numerical integrations must be performed with a three dimensional grid. The method employed was developed by Becke [36] where an atom-centred approach is used as opposed to a Cartesian coordinate approach. In this approach, an atom-centred grid is laid upon each atom in the molecule where the grid is divided into two parts, the radial grid and the angular grid. The radial grid determines the number and position of shells and the angular grid determines the number and position of the grid points on the shells.

It is known that the electronic distribution changes rapidly near the nucleus. It would be of great advantage to choose the radial points to be distributed in such a way as to take advantage of this fact. The radial points therefore should be distributed so that more shells are concentrated near the nucleus and gradually become spaced further apart the further one is from the nucleus. In consideration of this there is a simple formula to transform each radial variable r from the finite domain to the semi-infinite domain, $0 \rightarrow \infty$, by

$$r = r_m \frac{(1+x)}{(1-x)}, \tag{40}$$

where $-1 < x < 1$. The variable x, in this equation, is expressed by the Gauss-Chebyshev quadrature [37] points x_i such that

$$x_i = \cos\left(\frac{\pi\ i}{N+1}\right), \tag{41}$$

where N is the number of radial points. It is easy to see that these Gauss-Chebyshev points form a uniform mesh defined by the new variable, z, on the interval $0 \leq z \leq 1$ where

$$x = \cos(\pi z) \tag{42}$$

and

$$z_i = i/(N+1). \tag{43}$$

Now that the position of the spherical shells has been defined, an angular mesh must be established. The use of the unit sphere quadratures of Lebedev [38] will exactly integrate spherical harmonic functions up to the order of l=29, remembering that l is the angular momentum quantum number in the spherical harmonic, $Y_l^m(\theta,\phi)$. The quadratures of Lebedev integrate to a high order of angular momentum and also have a relatively simple point distribution of octahedral symmetry.

In principle, the grid encompasses all of space, but what happens in the case of real molecules where these grids are centred at the nucleus of each atom in the molecule? The obvious problem that will occur is the double counting of space leading to erroneous results. In order to avoid this phenomenon it is necessary to partition space in such a way that there is no double counting. The first method proposed was simply to bisect the bond between two atoms with a plane and not to count the space beyond that plane as part of a given atom. The perpendicular bisector can be found by evaluation of

$$\mu_{ij} = (r_i - r_j)/R_{ij} \tag{44}$$

where r_i, r_j, and R_{ij} denote the distance to nucleus i, distance to nucleus j, and the internuclear separation, respectively. Clearly, with this formula, it is easy to see that when μ_{ij}=0, the perpendicular bisector of the internuclear separation between atoms i and j is found. As such, a step function, $s(\mu_{ij})$, is defined, as follows, so as to avoid the double counting of space (also shown in Figure 3),

$$s(\mu_{ij}) = 1, \quad \text{for} \quad -1 \leq \mu_{ij} \leq 0 \tag{45}$$

and

$$s(\mu_{ij}) = 0, \quad \text{for} \quad 0 < \mu_{ij} \leq 1. \tag{46}$$

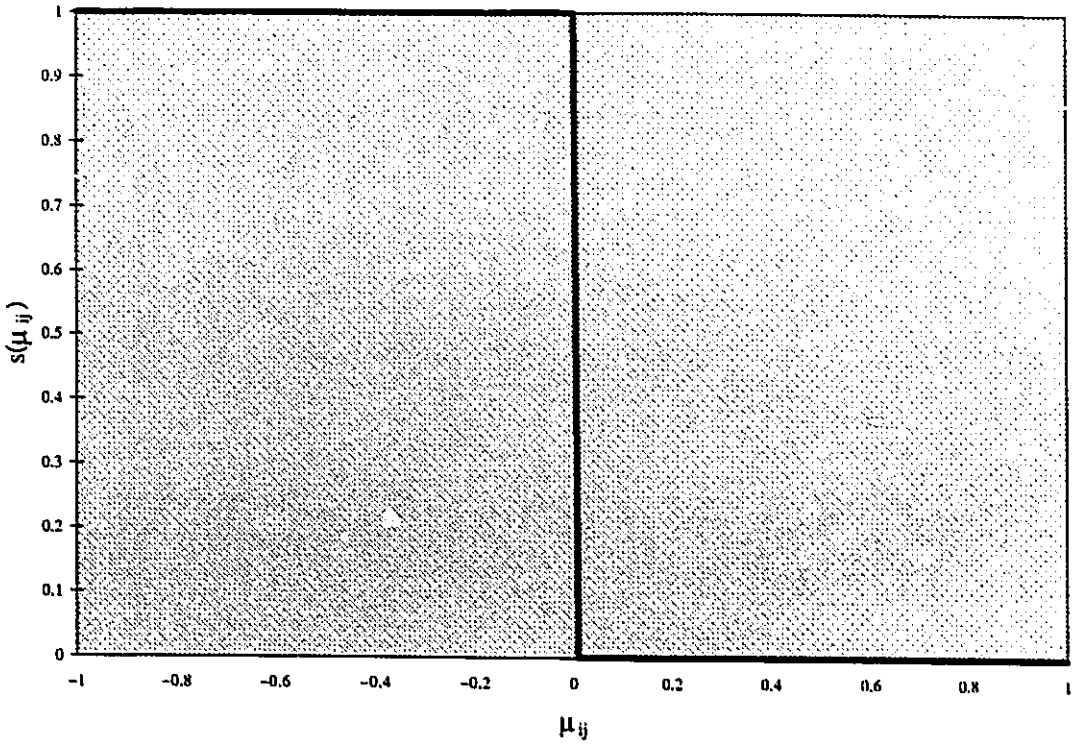

Figure 3. The Heavyside function for the partitioning of multi-centred integrals of polyatomic molecules.

This results in the following mathematical definition of a single Voronoi polyhedron centred about nucleus *i*

$$P_i(r) = \prod_{j \neq i} s(\mu_{ij}) \tag{47}$$

where the cell function, $P_i(r)$, is given by the product of the step function, $s(\mu_{ij})$, over the interval $-1 \leq \mu_{ij} \leq 1$. Subsequently, if r lies within the cell, the cell function has a value of unity and if it is outside the cell, the function becomes zero.

However, this partitioning proved to be too drastic because as molecules were rotated in space, grid points were suddenly being eliminated entirely. Becke [36] proposed a unique method to smooth out the boundary of each cell, so as to create so called 'fuzzy' Voronoi polyhedra. In this method, Becke experimented with various functional forms to create a continuous, overlapping cutoff function

$$s(\mu_{ij}) = (1/2)[1 - f(\mu_{ij})] \tag{48}$$

where

$$f(\mu_{ij}) = p\{p[p(\mu_{ij})]\} \tag{49}$$

and

$$p(\mu_{ij}) = (3/2)\mu_{ij} - (1/2)\mu_{ij}^3. \tag{50}$$

In equation (48), $s(\mu_{ij})$ goes from a value of 1 near its nucleus and goes smoothly to zero as j approaches. The change from 1 to 0 occurs mostly over a relatively short range spanning the midpoint between i and j.

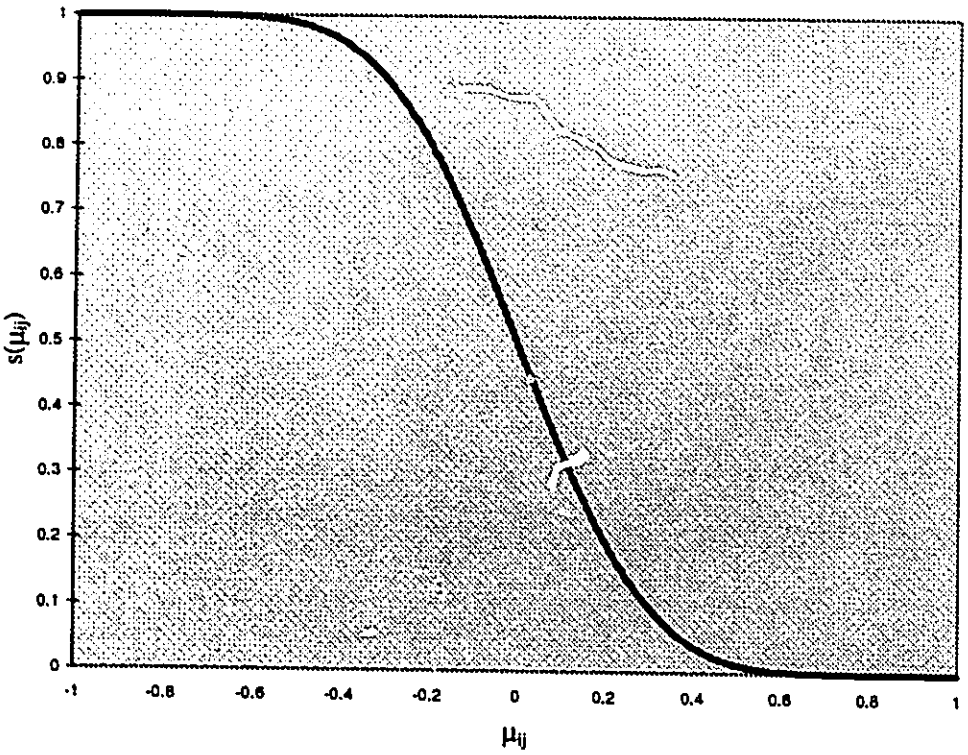

Figure 4. Becke's step function, $s(\mu_{ij})$, for partitioning of multi-centred integrals of polyatomic molecules.

In the superposition of the atom centred grids in the molecule, no grid points are eliminated, but the quadrature weights are readjusted so as not to double count space. This is done using a new weight function, $w(r)$, where

$$\sum_{n}^{atoms} w_n(r) = 1 \tag{51}$$

such that $w_n(r)$ has the value of unity at the nucleus n, but vanishes smoothly and continuously near any other nucleus. The function $w_n(r)$ is defined by Becke as follows.

$$w_n(r) = \frac{P_n(r)}{\sum_n P_n(r)} \tag{52}$$

It is now possible to divide a multi-centred integral into a summation of mono-centred contributions to the multi-centred integral. The multi-centred integral

$$I = \int dr\, F(r) \tag{53}$$

over all of space becomes

$$I = \sum_n^N I_n \tag{54}$$

the sum of the mono-centred contributions. Numerical integration about a lone atom is given by

$$I_n = \sum_{i_n}^{M_n} W(i_n) F(i_n) \tag{55}$$

where M_n is the number of points on the grid around the centre n, $W(i_n)$ is the volume of space associated or weight of the gridpoint i_n and $F(i_n)$ is the value of F at the gridpoint i_n. For an atom in the presence of other centres, such as in a molecule, I_n of equation (55) becomes

$$I_n = \sum_{i_n}^{M_n} w_n(i_n) W(i_n) F(i_n) \tag{56}$$

where $w_n(i_n)$ is the relative function given in equation (52).

One problem that arises with the use of grids is grid noise. When the molecule is rotated in space, the angular grid maintains the same orientation in Cartesian space causing a fluctuation in the total energy due to the relative orientation between angular grids centred on different atoms being altered [39]. Built into the functions $w_n(r)$ is the ability to minimize this effect for a particular number of grid points. An acceptable level of

grid noise in the total energy is on the order of 0.1 to 0.2 kcal mol^{-1} which is standard for most default grids used in DF programs.

Parallel Implementation

One of the most time consuming parts of DeFT is generating the molecular orbitals on the grid. It is because of this fact that the subroutine gridwork.f needed improvement. It was a good candidate for parallelization due to the fact that the calculations at the gridpoints could be divided over a number of processors. Figure 5 shows the loop structure that was used in the parallel version of gridwork.f.

```
DO I=1,iterations
    DO J=1,ncentres
        DO K=1,npoints,128
            DO L=1,nprocedures
                call slave(atomic orbitals)
                [Pass block of 128 gridpoints over the processors]
                [Calculations of atomic orbitals at each gridpoint]
            CONTINUE
            DO L=1,nprocedures
                call slave(matrix multiply)
                [Pass sections of matrix of atomic orbitals to processors]
                [Multiply atomic orbitals at each gridpoint by coefficients
                giving molecular orbitals at each gridpoint]
            CONTINUE
        CONTINUE
    CONTINUE
CONTINUE
```

Figure 5. Loop structure of parallel version of gridwork.f.

The outermost loop is a loop over each iteration in the SCF procedure until convergence is achieved or the loop reaches a maximum value and it is determined the calculation will not converge. The next loop runs over all the atoms in the molecule because Becke's numerical grid is atom centred. The third loop is used to cut the number of gridpoints involved in the calculation into blocks of 128 points so as to ensure that no problems occur with limitations on memory size. It was decided that these blocks of 128 gridpoints would be the focus of the parallel implementation. The 128 gridpoints could be divided over the number of processors in a cluster of workstations or super-computer to perform independent calculations at each gridpoint. For example, the massively parallel super-computer, Cray T-3D, has 128 processors. It would be possible to spawn 128 slaves and send one gridpoint to each of the slaves from the block of 128 gridpoints. Each slave would then return the value of every atomic orbital for that gridpoint. However, in practice this would not be done. In this manner, the parallel computation would become inefficient, since it is necessary to maximize the amount of work done with each message passed. Division of the grid into blocks of 128 points is fine for the purpose of this research because only four processors were available. These atomic orbitals are stored in a matrix to be multiplied by the appropriate coefficients so as to form the molecular orbitals at each gridpoint. This is the purpose of the second of the inner most loops. This loop cuts up the matrix holding the values of atomic orbitals at the gridpoints over the number of processes for a parallel matrix multiply.

Finally, the elements of the matrix containing the molecular orbitals at the gridpoints are squared and summed to yield the electronic density. The time consuming steps of generating the electronic density benefit greatly from this parallel algorithm.

Integrals

As discussed earlier, it is the two-electron integrals that demand the majority of the CPU time in any electronic structure calculation. Although in the LCGTO-DF scheme, the fitting procedure yields a N^2M scaling, where M is the number of auxiliary basis functions. This reduction in CPU time is a tremendous advantage, however the evaluation of the integrals is still an important component of any calculation. It is therefore necessary to implement the most efficient algorithm possible for their evaluation. DeFT uses the formulas of Obara and Saika [40], a recursive approach for evaluating molecular integrals with cartesian gaussian functions that is highly vectorizable and parallelizable.

The integrals of Obara and Saika that were evaluated with the parallel algorithm were the following:

Coulomb Repulsion Integrals

Exchange and Correlation Integrals

The unnormalized Cartesian gaussian function is denoted by

$$\phi(\boldsymbol{r};\zeta,\boldsymbol{n},\boldsymbol{R}) = (x-R_x)^{n_x}(y-R_y)^{n_y}(z-R_z)^{n_z}\exp\{-\zeta(\boldsymbol{r}-\boldsymbol{R})^2\} \qquad (57)$$

where $\boldsymbol{R} = (R_x, R_y, R_z)$ is the centre of the function, $\boldsymbol{r} = (x, y, z)$ is the position of the electron, ζ is the orbital exponent and $\boldsymbol{n} = (n_x, n_y, n_z)$ is defined as the angular momentum index. The basic equation used by Obara and Saika [40] for the derivations of the molecular integrals is the derivative of the Cartesian Gaussian function with respect to its centre,

$$\frac{\partial}{\partial R_i}\phi(\boldsymbol{r};\zeta,\boldsymbol{n},\boldsymbol{R}) = 2\zeta\phi(\boldsymbol{r};\zeta,\boldsymbol{n}+\boldsymbol{1}_i,\boldsymbol{R}) - N_i(\boldsymbol{n})\phi(\boldsymbol{r};\zeta,\boldsymbol{n}-\boldsymbol{1}_i,\boldsymbol{R}) \ (\mathrm{i} = \mathrm{x},\mathrm{y},\mathrm{z}). \qquad (58)$$

Note that a Gaussian of higher angular momentum is created and another Gaussian of lower angular momentum is potentially created if a unit of angular momentum along i exists already. The function, $N_i(n)$, behaves somewhat like Kronecker delta. It returns the

component, n_i, of the angular momentum. For example, the f_{xxz} function would be characterized by the set of integers $\mathbf{n} = (2,0,1)$ and the resulting values of $N_i(n)$ are

$$N_x(n)=2, \quad N_y(n)=0, \quad N_z(n)=1.$$

The function, $N_i(n)$, thus returns the number of units of angular momentum in each direction. This property of Cartesian Gaussian functions, differentiation of a Gaussian results in a Gaussian, is the basis of the recursive algorithm. This leads to integrals of higher order of angular momentum being recursively expressed as linear combinations of integrals of lower angular momentum.

Coulomb Repulsion Integrals

The Coulomb repulsion integrals given in Obara and Saika are four-centred integrals of the following form:

$$[\mathbf{ab}||\mathbf{cd}] = \int dr_1 \int dr_2 \; \phi\,(r_1;\zeta_a,a,A)\phi\,(r_1;\zeta_b,b,B)\frac{1}{|r_1-r_2|}\phi\,(r_2;\zeta_c,c,C)\phi\,(r_2;\zeta_d,d,D) \quad (59)$$

However, in DeFT these four centred integrals are replaced by three centred integrals by simply setting the exponent of the fourth centre equal to zero, meaning that the fourth gaussian equals one over all of space.

$$[\mathbf{ab}||\mathbf{c}] = \int dr_1 \int dr_2 \; \phi\,(r_1;\zeta_a,a,A)\phi\,(r_1;\zeta_b,b,B)\frac{1}{|r_1-r_2|}k\,(r_2;\zeta_c,c,C) \quad (60)$$

Two centred integrals required by the fitting procedure are similarly obtained by setting the exponent on the second centre to zero,

$$[\mathbf{c}||\mathbf{d}] = \int dr_1 \int dr_2 \; k(r_1;\zeta_c,c,C)\frac{1}{|r_1-r_2|}k\,(r_2;\zeta_d,d,D)\,. \quad (61)$$

In the above two equations, k is a cartesian gaussian function of the auxiliary basis set that fits the charge density. The recurrence formulas for the three centred electron repulsion integrals become

$$
\begin{aligned}
[(\mathbf{a}+\mathbf{1}_i)\mathbf{b}\|\mathbf{c}]^{(m)} &= (P_i - A_i)[\mathbf{ab}\|\mathbf{c}]^{(m)} + (W_i - P_i)[\mathbf{ab}\|\mathbf{c}]^{(m+1)} \\
&+ \frac{1}{2\zeta} N_i(\mathbf{a})\{[(\mathbf{a}-\mathbf{1}_i)\mathbf{b}\|\mathbf{c}]^{(m)} - \frac{\rho}{\zeta}[(\mathbf{a}-\mathbf{1}_i)\mathbf{b}\|\mathbf{c}]^{(m+1)}\} \\
&+ \frac{1}{2\zeta} N_i(\mathbf{b})\{[\mathbf{a}(\mathbf{b}-\mathbf{1}_i)\|\mathbf{c}]^{(m)} - \frac{\rho}{\zeta}[\mathbf{a}(\mathbf{b}-\mathbf{1}_i)\|\mathbf{c}]^{(m+1)}\} \\
&+ \frac{1}{2(\zeta+\zeta_c)} N_i(\mathbf{c})\{[(\mathbf{ab}\|(\mathbf{c}-\mathbf{1}_i)]^{(m+1)}
\end{aligned}
\tag{62}
$$

where

$$
\boldsymbol{P} = \frac{\zeta_a \boldsymbol{A} + \zeta_b \boldsymbol{B}}{\zeta_a + \zeta_b}, \tag{63}
$$

$$
\zeta = \zeta_a + \zeta_b \tag{64}
$$

$$
\rho = \frac{\zeta\zeta_c}{\zeta+\zeta_c}. \tag{65}
$$

and +/- $\mathbf{1}_i$ means that one unit of angular momentum has been added/subtracted in the direction *i*. Also,

$$
\begin{aligned}
[\mathbf{ab}\|(\mathbf{c}+\mathbf{1}_i)]^{(m)} &= (W_i - C_i)[\mathbf{ab}\|\mathbf{c}]^{(m+1)} + \frac{1}{2(\zeta+\zeta_c)} N_i(\mathbf{a})\{[(\mathbf{a}-\mathbf{1}_i)\mathbf{b}\|\mathbf{c}]^{(m+1)} \\
&+ \frac{1}{2(\zeta+\zeta_c)} N_i(\mathbf{b})\{[\mathbf{a}(\mathbf{b}-\mathbf{1}_i)\|\mathbf{c}]^{(m+1)} \\
&+ \frac{1}{2\zeta_c} N_i(\mathbf{c})\{[(\mathbf{ab}\|(\mathbf{c}-\mathbf{1}_i)]^{(m)} - \frac{\rho}{\zeta_c}[(\mathbf{ab}\|(\mathbf{c}-\mathbf{1}_i)]^{(m+1)}\}
\end{aligned}
\tag{66}
$$

where

$$
\boldsymbol{W} = \frac{\zeta \boldsymbol{P} + \zeta_c \boldsymbol{C}}{\zeta+\zeta_c}. \tag{67}
$$

The two centred electron repulsion integrals are

$$
\begin{aligned}
[\mathbf{c}+\mathbf{1}_i\|\mathbf{d}]^{(m)} &= (Q_i - C_i)[\mathbf{c}\|\mathbf{d}]^{(m+1)} + \frac{1}{2(\zeta_c+\zeta_d)} N_i(\mathbf{d})[[\mathbf{c}\|\mathbf{d}-\mathbf{1}_i]^{(m+1)} \\
&+ \frac{1}{2\zeta_c} N_i(\mathbf{c})\{[\mathbf{c}-\mathbf{1}_i\|\mathbf{d}]^{(m)} - \frac{\sigma}{\zeta_c+\zeta_d}[\mathbf{c}-\mathbf{1}_i\|\mathbf{d}]^{(m+1)}
\end{aligned}
\tag{68}
$$

where

$$Q = \frac{\zeta_c C + \zeta_d D}{\zeta_c + \zeta_d}, \tag{69}$$

and

$$\sigma = \frac{\zeta_c \zeta_d}{\zeta_c + \zeta_d}. \tag{70}$$

In a recursive algorithm, there always has to be a starting point from which to build the recursive formulas. Since the molecular integrals are formed by building up angular momentum on a centre, the starting point should be the integral with the lowest angular momentum on all of the centres, i.e., all the functions are *s* functions. The base integral for three-centred electron repulsion is given by

$$[s_A\, s_B \,||\, s_C\,]^{(m)} = \frac{2\pi^{5/2}}{\zeta\zeta_c(\zeta+\zeta_c)^{-1/2}} \exp\{-\xi\,(A-B)^2\}\, F_m(T) \tag{71}$$

where

$$F_m(T) = \int_0^1 dt\, t^{2m} \exp[-Tt^2]$$

and

$$T = \rho\,(P-C)^2 \tag{72}$$

The two-centred electron repulsion base integral is

$$[s_C \,||\, s_D\,]^{(m)} = \frac{2\pi^{5/2}}{\zeta_c\,\zeta_d\,(\zeta+\zeta_c)^{-1/2}} F_m(T) \tag{73}$$

where

$$T = \sigma\,(C-D)^2. \tag{74}$$

Exchange and Correlation Integrals

The exchange and correlation integrals in the LCGTO-DF formalism have the form,

$$<\mathbf{ab|c}> = \int dr\, \phi(r;\zeta_a,a,A)\phi\,(r;\zeta_b,b,B) l(r;\zeta_c,c,C) \tag{75}$$

where l is an auxiliary basis cartesian gaussian function for the fitting of the XC terms. This integral is a three-centred overlap integral and is treated explicitly by Obara and Saika. The recurrence formula is

$$\begin{aligned} &<(\mathbf{a}+\mathbf{1}_i)b|\mathbf{c}>=(G_i-A_i)<\mathbf{ab}|\mathbf{c}>+ \\ &\frac{1}{2(\zeta+\zeta_c)}\{N_i(\mathbf{a})<(\mathbf{a}+\mathbf{1}_i)b|\mathbf{c}>+N_i(\mathbf{b})<a(\mathbf{b}+\mathbf{1}_i)|\mathbf{c}>+N_i(\mathbf{c})<ab|\mathbf{c}+\mathbf{1}_i>\} \end{aligned} \tag{76}$$

where

$$G_i=\frac{\zeta P_i+\zeta_c C_i}{\zeta+\zeta_c} \tag{77}$$

and the base integral is

$$<s_A\, s_B\,|s_C>=\frac{\zeta^{3/2}}{(\zeta+\zeta_c)^{3/2}}<s_A\,|s_B>\exp\{-\frac{\zeta\zeta_c}{\zeta+\zeta_c}(\boldsymbol{P}-\boldsymbol{C})^2\}. \tag{78}$$

Transfer of Angular Momentum

In the previous recursive formulas, integrals of higher order of angular momentum are expressed as a linear combination of integrals of lower angular momentum. In the work of Head-Gordon and Pople [41] they refer to this as a vertical recurrence relation because angular momentum is being built up on the centres.

Head-Gordon and Pople came up with a new recurrence relation. This will be illustrated using the three centred overlap integrals for the exchange and correlation. The new recurrence relation is obtained by subtracting the vertical recurrence relation for building up angular momentum in **b**,

$$\begin{aligned} &<\mathbf{a}(\mathbf{b}+\mathbf{1}_i)|\mathbf{c}>=(G_i-B_i)<\mathbf{ab}|\mathbf{c}>+ \\ &\frac{1}{2(\zeta+\zeta_c)}\{N_i(\mathbf{a})<(\mathbf{a}+\mathbf{1}_i)b|\mathbf{c}>+N_i(\mathbf{b})<a(\mathbf{b}+\mathbf{1}_i)|\mathbf{c}>+N_i(\mathbf{c})<ab|\mathbf{c}+\mathbf{1}_i>\} \end{aligned} \tag{79}$$

from the vertical recurrence relation on A

$$< (\mathbf{a}+\mathbf{1}_i)\mathbf{b}|\mathbf{c} >= (G_i - A_i) < \mathbf{ab}|\mathbf{c} > +$$
$$\frac{1}{2(\zeta+\zeta_c)}\{N_i(\mathbf{a}) < (\mathbf{a}+\mathbf{1}_i)b|\mathbf{c} > + N_i(\mathbf{b}) < a(\mathbf{b}+\mathbf{1}_i)|\mathbf{c} > + N_i(\mathbf{c}) < ab|\mathbf{c}+\mathbf{1}_i >\} \quad (80)$$

This gives

$$< \mathbf{a}(\mathbf{b}+\mathbf{1}_i)|\mathbf{c} >= (A_i - B_i) < \mathbf{ab}|\mathbf{c} > + < (\mathbf{a}+\mathbf{1}_i)\mathbf{b}|\mathbf{c} > \quad (81)$$

a new recurrence relation for a given integral in terms of another integral with the same total angular momentum, but a unit of angular momentum shifted from centre one to centre two, plus a second integral of lower angular momentum. This shifting of angular momentum from centre to centre was referred to as a horizontal recurrence relation. The two integrals on the right hand side of equation (81) are obtained from the vertical recurrence relations by the formulas of Obara and Saika [40].

It was shown that the number of operations needed to be performed was significantly reduced for evaluation of integrals involving *d* functions or higher [31]. The Head-Gordon and Pople method may also be applied to any integral where two basis functions are associated with the same electron. Therefore, only two-centred two-electron integrals, equation (61), cannot make use of the horizontal recurrence relation.

Programming Strategy

In DeFT, each combination of integrals is explicitly programmed. This is done because the majority of mathematical operations include the factor, $N_i(\boldsymbol{n})$. Many of these factors are equal to zero or one. By explicitly programming the integrals, $N_i(\boldsymbol{n})$ can be replaced by its numerical value, zero, or one, or two, etc. If it zero, then the term is omitted. In doing so, the number of operations to calculate the molecular integrals is significantly reduced. For example, the Coulomb repulsion integrals of the form [dd|ld] will have 6x6x6=216 possible combinations. For each combination there exists several operations to be performed, in which many depend on the factor, $N_i(\boldsymbol{n})$. The omission of the terms where

$N_i(\boldsymbol{n})$ equals zero results in many operations being neglected. This should make the process more efficient.

A group of integrals, such as the electron repulsion integrals, are programmed in the following manner. The base integral consisting of entirely *s*-type functions is evaluated. Then, the angular momentum is built up on the centre with the highest final angular momentum. Once the appropriate value of angular momentum is attained on this centre, the angular momentum is transferred to the other centre, if required. Therefore, whenever possible, during the course of the programming of these integrals, the formulas for the transfer of angular momentum by Head-Gordon and Pople are used.

With the combination of both the Obara-Saika and Head-Gordon and Pople methods, the integrals were vectorized. Since everything that is vectorizable is also parallelizable, these integrals were a good candidate for modification by PVM to take advantage of the power of super-computers or clusters of workstations. The programming strategy is separated into two steps. The first is to calculate the parameters, $\boldsymbol{P}$, ζ, and $\exp\{-\xi(\boldsymbol{A}-\boldsymbol{B})^2\}$ for each pair of primitives between the orbital basis functions of types s and s, p and s, p and p, d and s, d and p, and d and d. During the second step the integrals are calculated. It is here where the integrals are vectorized, the length of each vector is the number of combinations of primitives evaluated during the first step. The loop structure for the evaluation of the electron repulsion integrals is shown in Figure 6.

```
DO I=1,nauxiliary
    DO J=1,n
        [Evaluation of three-centred ERI or overlap integrals]
    CONTINUE
CONTINUE
```

Figure 6. Loop structure of ERI evaluation.

The outer loop is over the number of auxiliary functions used in the fitting procedure. The inner loop is a loop for the first pair of primitive shells. It was decided that in the parallel implementation of the evaluation of the molecular integrals, the outer loop over the auxiliary primitives would be divided by the number of slaves processes spawned. As a result, the evaluation of the three-centred integrals are divided over several processors. The above shown loop structure, then becomes the slave program with the outer loop reduced in size. Each process returns its contribution to the KS matrix to the master program, where they are summed. The master program simply requires a loop over the number of processes that wish to be spawned.

Final Note

A final note: copies of the PVM software may be obtained from *netlib. Netlib* is a software archive based at the Oak-Ridge National Laboratories in Tennessee, U.S.A. Four available methods of retrieving the software are through anonymous FTP, the World Wide Web, gopher, or through e-mail.

The anonymous FTP is done by FTPing to netlib.ornl.gov with username ftp and your e-mail address as password. The URL for the WWW site of Oak-Ridge National Laboratories is http://netlib.ornl.gov/index.html and gopher is done using the address netlib.ornl.gov. Finally, through e-mail, send a message to netlib@ornl.gov stating 'send index' and an automatic mail handler will return the necessary information.

Divide and Conquer Approach

Density Functional Theory offers perhaps the best ratio of quality of results to the cost of calculation, among *ab initio* methods. This is due to the fact that the theory incorporates both exchange and correlation effects and formally scales as N^3, where N refers to system size. However, to study truly large molecular systems, traditional DF calculations are still too computationally demanding.

Yang has recently developed two divide and conquer approaches (DAC): the density [15] and density matrix [16] formulations. In these DAC formulations, large molecules are divided into chemically intuitive subsystems with the aid of partition functions. The KS equations are solved within a subsystem with only the basis functions in or near the subsystem. The density is then pieced together. For very large systems, the amount of CPU time required for calculations on any one given subsystem will level off making the scaling dependent only on the number of subsystems. The overall procedure will scale linearly with system size since the number of subsystems is directly proportional to the size of the molecule.

In this work, a DAC approach for the fitting of the charge density and exchange-correlation terms using auxiliary bases of uncontracted Gaussian type orbitals is examined. This is necessary in a DAC procedure for LCGTO-DF because it would be useless to have a linear scaling method for the solution to the KS equations if a linear scaling method does not exist for the fits as well. In this methodology, Yang's density matrix formulation is used for the fitting of the electronic density and his density formulation is used for the fit to the exchange-correlation terms. The former was recently implemented in DeFT [42], the latter has also been developed and implemented, while work on a fully developed DAC approach for LCGTO-DF is still underway [43].

For a DAC approach to be deemed efficient, it is desirable that errors introduced by employing this approach, relative to conventional DF results, are no larger than roughly 0.1-0.2 kcal mol^{-1} for relative conformational energies of small peptides. This value is

chosen since the average error in the DF XC functionals is roughly double the above mentioned error [20]. The greatest source of error should not be that of the DAC procedure. A test case using the beta sheet conformation of the glycine heptapeptide is examined to get a feel for the errors introduced by the DAC procedure.

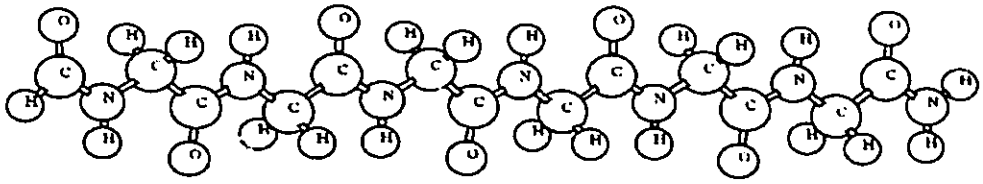

Figure 7. Glycine heptapeptide

Density Formulation

In both the density and density matrix formulations, the total density is expressed as a sum of subsystem densities,

$$\rho(\boldsymbol{r}) = \sum_{\alpha}^{subsystems} \rho^{\alpha}(\boldsymbol{r}). \tag{82}$$

However, in the density formulation, a subsystem density is given by

$$\rho^{\alpha}(\boldsymbol{r}) = p^{\alpha}(\boldsymbol{r})\rho(\boldsymbol{r}) \tag{83}$$

where $p^{\alpha}(\boldsymbol{r})$ is a partition function that must obey the constraint

$$\sum_{\alpha}^{subsystems} p^{\alpha}(\boldsymbol{r}) = 1 \tag{84}$$

at each point in space. Provided this constraint is obeyed, the correct total density, $\rho(\boldsymbol{r})$, from equation (82) will be obtained regardless of the form of the partition function, $p^{\alpha}(\boldsymbol{r})$.

The expression for $\rho(\boldsymbol{r})$ in equation (4) is modified to the equivalent expression,

$$\rho(\boldsymbol{r}) = 2\sum_{i}^{M} \eta(\varepsilon_F - \varepsilon_i)|\psi_i(\boldsymbol{r})|^2 \tag{85}$$

where the summation is over the doubly occupied KS orbitals and that of the virtual orbitals that may exist. These virtual orbitals do not contribute to the sum. In the above

equation, ε_F is the Fermi energy and η is a Heavyside function that returns a value of 1 if $\varepsilon_i \leq \varepsilon_F$ and 0 if $\varepsilon_i > \varepsilon_F$.

Equations (83) and (85) are combined to give a new expression for the subsystem density,

$$\rho^{\alpha}(r) = 2p^{\alpha}(r)\sum_{i}^{M}\eta\,(\varepsilon_F - \varepsilon_i)|\psi_i(r)|^2 \tag{86}$$

and it follows that the total density, $\rho(r)$, becomes

$$\rho\,(r) = 2\sum_{\alpha}^{subsystems} p^{\alpha}(r)\sum_{i}^{M}\eta\,(\varepsilon_F - \varepsilon_i)|\psi_i(r)|^2\,. \tag{87}$$

The partition function is constructed in such a way so that it vanishes quite rapidly when moving away from the atoms of subsystem α. It is defined as,

$$p^{\alpha}(r) = \frac{\sum_{A\in\alpha}^{atoms}[\rho^{A}_{atom}(|r - R_A|)]^2}{\sum_{B}^{atoms}[\rho^{B}_{atom}(|r - R_B|)]^2} \tag{88}$$

where the summation in the numerator is over only the atoms in subsystem α, the summation in the denominator runs over all the atoms in the molecule and $\rho^{A}_{atom}(|r - R_A|)$ is the spherical atomic density of atom A.

Until now, no approximations have been made. However, since $p^{\alpha}(r)$ localizes the contributions to $\rho^{\alpha}(r)$ about the subsystem atoms, only the KS orbitals about the atoms of subsystem α are needed when constructing $\rho^{\alpha}(r)$. Equation (18) may now be solved with only the orbital bases of the subsystem atoms,

$$\boldsymbol{H}^{\alpha}\boldsymbol{C}^{\alpha} = \boldsymbol{S}^{\alpha}\boldsymbol{C}^{\alpha}\boldsymbol{\varepsilon}^{\alpha} \tag{89}$$

where the matrices and vectors have reduced dimensions due to the fact that they only span the basis functions of the subsystem α. The subsystem KS orbitals, $\psi^{\alpha}_{i}(r)$, can then be constructed with the elements of $\boldsymbol{C}^{\alpha}$,

$$\psi^{\alpha}_{i}(r) = \sum_{\mu\in\alpha}^{K} C^{\alpha}_{\mu i}\chi_{\mu}(r) \tag{90}$$

where the summation is over only the basis functions that are centered on the atoms in the subsystem α. The total density is then reconstructed from the DAC KS orbitals via equation (85) to give,

$$\rho(r) \approx 2 \sum_{\alpha}^{subsystems} p^{\alpha}(r) \sum_{i}^{K^{\alpha}} \eta(\varepsilon_F - \varepsilon_i^{\alpha}) |\psi_i^{\alpha}(r)|^2 . \tag{91}$$

The total density is now approximate because the true KS orbitals were not used. The sum is now over K^{α}, the number of orbital basis functions in the subsystem. The Fermi energy, ε_F, is common to all subsystems. It is adjusted so that the approximate total density obtained in the DAC scheme is appropriately normalized to the number of electrons in the system.

A Heavyside function will not allow an SCF procedure to converge. A further approximation is made to ensure convergence by replacing the Heavyside function by a Fermi function,

$$\rho(r) \approx 2 \sum_{\alpha}^{subsystems} p^{\alpha}(r) \sum_{i}^{K^{\alpha}} \frac{1}{1+e^{-\beta(\varepsilon_F - \varepsilon_i^{\alpha})}} |\psi_i^{\alpha}(r)|^2 \tag{92}$$

where β is an adjustable parameter. The value of ε_F is set so that the DAC density is normalized,

$$2 \sum_{\alpha}^{subsystems} \sum_{i}^{K^{\alpha}} \frac{1}{1+e^{-\beta(\varepsilon_F - \varepsilon_i^{\alpha})}} \int p^{\alpha}(r) |\psi_i^{\alpha}(r)|^2 \, dr = N . \tag{93}$$

The Fermi function allows for convergence since it is continuous. The adjustable parameter, β, has little effect on the final results provided the Fermi function is sufficiently steep.

Density Matrix Formulation

The density matrix formulation is a DAC approach based on the density matrix instead of the density itself. The total density is expressed as a sum of subsystem densities as in equation (82). However, the subsystem density is now expressed as

$$\rho^{\alpha}(r) = \sum_{\mu}^{M}\sum_{\nu}^{M} p^{\alpha}_{\mu\nu} P_{\mu\nu} \chi_{\mu}(r)\chi_{\nu}(r) \tag{94}$$

where the partition function, $p^{\alpha}_{\mu\nu}$, in this case, is expressed as,

$$f^{\alpha}_{\mu\nu} = \begin{cases} 1 & \text{if } \mu \in \alpha \text{ and } \nu \in \alpha \\ 1/2 & \text{if } \mu \in \alpha \text{ and } \nu \notin \alpha \text{ or } \mu \notin \alpha \text{ and } \nu \in \alpha \\ 0 & \text{if } \mu \notin \alpha \text{ and } \nu \notin \alpha \end{cases} \tag{95}$$

The partition function, $p^{\alpha}_{\mu\nu}$, clearly obeys the constraint,

$$\sum_{\alpha}^{subsystems} p^{\alpha}_{\mu\nu} = 1 \tag{96}$$

for each pair of basis functions.

Summation over all the occupied and virtual orbitals yields the following expression for $\rho^{\alpha}(r)$,

$$\rho^{\alpha}(r) = 2\sum_{\mu}^{K}\sum_{\nu}^{K} p^{\alpha}_{\mu\nu} \sum_{i}^{M} \eta(\varepsilon_F - \varepsilon_i)\, C_{\mu i}\, C_{\nu i} \chi_{\mu}(r)\chi_{\nu}(r) \tag{97}$$

In a manner similar to the density formulation, the true KS orbital coefficients, $C_{\mu i}$, are replaced by the subsystem cofficients, $C^{\alpha}_{\mu i}$, that arise from equation (89). Also, the Heavyside function is again replaced by the Fermi function,

$$\rho^{\alpha}(r) = 2\sum_{\mu}^{K}\sum_{\nu}^{K} p^{\alpha}_{\mu\nu} \sum_{i}^{K^{\alpha}} \frac{1}{1+e^{-\beta(\varepsilon_F-\varepsilon_i)}}\, C^{\alpha}_{\mu i}\, C^{\alpha}_{\nu i} \chi_{\mu}(r)\chi_{\nu}(r)\,. \tag{98}$$

As before, the sum of the subsystem densities is properly normalized with the adjustment of ε_F,

$$\sum_{\alpha}^{subsystems} 2\sum_{\mu}^{K}\sum_{\nu}^{K} p^{\alpha}_{\mu\nu} \sum_{i}^{K^{\alpha}} \frac{1}{1+e^{-\beta(\varepsilon_F-\varepsilon_i)}} C^{\alpha}_{\mu i}\, C^{\alpha}_{\nu i} \int \chi_{\mu}(r)\chi_{\nu}(r)dr = N \tag{99}$$

Buffer Atoms

Unfortunately, the DAC formulation's approximations are too severe and result in large errors in the total energy of the molecule. The subsystem KS orbitals that span only the basis functions of the subsystem atoms are not sufficient to represent the true KS orbitals in the vicinity of the subsystem. If the basis functions of neighboring atoms of the subsystem are included, there is an improvement in the resulting total energies of molecules. This is the introduction of buffer atoms [15]. Buffer atoms of the subsystem α are neighboring the subsystem and are not part of the subsystem. However, their basis functions will be included in the basis set of subsystem α to help the subsystem KS orbitals to more closely resemble the true KS orbitals in the vicinity of the subsystem. Equation (89) is then solved within the orbital bases of the subsystem and buffer atoms.

The addition of the buffer atoms in no way affects the partition functions of either DAC formulation. If an atom is in the buffer space of a subsystem, then the atoms in this subsystem will in turn be found in the buffer space of this atom's subsystem. Buffer atoms could be assigned in two ways. One way is by defining a certain cutoff distance about a subsystem as a buffer zone and any atom in this zone would become a buffer atom. The second method is to assign buffer atoms according to the number of bonds separating them from the subsystem atoms.

Mulliken Population Analysis

It is often desirable to be able to somehow allocate portions of the electronic distribution to each atom within a molecule. One such method, presented here, is the Mulliken population analysis [44]. If a DAC procedure for the fitting of the electronic density is to be developed, the electronic density of a subsystem must be normalized to the number of electrons in the subsystem. Since the DAC density matrix formulation is in the spirit of the Mulliken approach to partitioning of the density, the Mulliken population

analysis is used to assign the number of electrons to a subsystem. Other population analyses exist, but are not "compatible" with the DAC scheme. This section focuses on the calculation of the Mulliken population that is used in the DAC method for the fitting of charge density.

The electronic density, $\rho(r)$, is defined such that $\rho(r)dr$ is the probability of finding an electron in the small volume element, dr, at the point in space, r. Of course integration over all of space must yield

$$\int \rho\,(r)\,dr = N \tag{100}$$

where N is the total number of electrons. In the LCGTO method, the electronic density is expressed in terms of Gaussian type basis functions,

$$\rho\,(r) = \sum_{\mu}^{L}\sum_{\nu}^{L} P_{\mu\nu}\chi_{\mu}(r)\chi_{\nu}(r) \tag{101}$$

where $P_{\mu\nu}$ are the elements of density matrix and L is the number of basis functions. The above equation can be re-expressed using the overlap matrix $S_{\mu\nu}$

$$\int \rho\,(r)\,dr = \sum_{\mu}^{L}\sum_{\nu}^{L} P_{\mu\nu}S_{\mu\nu} = N\,. \tag{102}$$

The total electron count of the system can be regarded as the summation of the individual terms $P_{\mu\nu}S_{\mu\nu}$. Given that the basis functions χ_{μ} are normalized, $S_{\mu\mu}$=1. This means the diagonal terms $P_{\mu\mu}$ are directly associated with the number of electrons on the basis function χ_{μ}. The off-diagonal elements, $P_{\mu\nu}S_{\mu\nu}$ and $P_{\nu\mu}S_{\nu\mu}$, are equal in magnitude and expressed as

$$Q_{\mu\nu} = 2P_{\mu\nu}S_{\mu\nu} \qquad (\mu \neq \nu)\,, \tag{103}$$

also called the overlap population. The total electronic charge is then expressed as sum of two parts; the diagonal and the off-diagonal contributions,

$$\sum_{\mu}^{L} P_{\mu\mu} + \sum_{\mu}^{L}\sum_{\nu<}^{L} Q_{\mu\nu} = N\,. \tag{104}$$

This particular form of the total electronic charge is not desirable when the two different basis functions, χ_μ and χ_ν, are not necessarily centered on the same atom. Mulliken decided to partition the total charge among only the individual basis functions. The method used was to divide the overlap populations, $Q_{\mu\nu}$, equally between the basis functions χ_μ and χ_ν adding half to each of the net populations $P_{\mu\mu}$ and $P_{\nu\nu}$. The gross population of χ_μ is defined as

$$Q_\mu = P_{\mu\mu} + \sum_{\mu \neq \nu} P_{\mu\nu} S_{\mu\nu} . \tag{105}$$

The summation of the gross populations over all the basis functions is equal to the total number of electrons,

$$\sum_{\mu}^{L} Q_\mu = N . \tag{106}$$

The gross atomic population for any centre is given by

$$Q_A = \sum_{\mu}^{A} Q_\mu \tag{107}$$

where the summation is over all basis functions on the centre A. The total atomic charge on the atom A may be defined now as $(Z_A - Q_A)$, where Z_A is the atomic number of A.

Caution must be used, so as not to read too much information into absolute values of the atomic charges. Remember that the overlap population was equally partitioned into individual orbital contributions potentially causing errors in cases when electron density of an atomic orbital may be greater at a region closer to a neighboring atom rather than the atom on which the orbital is centred. This separation of electronic density is arbitrary. Mulliken charges are also very dependent on basis sets. Caution must also be taken when using Mulliken charges for studies that require atomic charges, such as molecular mechanics. Mulliken charges are often quite poor at reproducing electrostatic potentials. Other charge schemes are far more appropriate as they adjust atomic charges to best reproduce the electrostatic potential [45].

DAC Fitting of the Electronic Density

As discussed earlier, to ease the computational burden, the LCGTO-DF methodology, as proposed by Dunlap, Connolly and Sabin [33], fits $\rho(r)$ by a linear combination of M uncontracted GTO auxiliary basis functions,

$$\tilde{\rho}(r) = \sum_{k}^{M} c_k \chi_k(r) \tag{108}$$

where the optimal set of expansion coefficients, c_k, minimizes the error in the Coulomb repulsion energy,

$$\int \frac{[\rho(r) - \tilde{\rho}(r)][\rho(r) - \tilde{\rho}(r)]}{|r - r'|} dr dr' \tag{109}$$

subject to the constraint that $\tilde{\rho}(r)$ be properly normalized to the total number of electrons, N,

$$\int \tilde{\rho}(r) dr = N. \tag{110}$$

As stated earlier, the Coulomb repulsion energy can be made variational (the ideal fit giving us an *upper* bound to the Coulomb repulsion energy) if it is expressed as

$$\iint \frac{\rho(r)\tilde{\rho}(r)}{|r - r'|} dr dr' - \frac{1}{2} \iint \frac{\tilde{\rho}(r)\tilde{\rho}(r)}{|r - r'|} dr dr'. \tag{111}$$

The following solution for the expansion coefficients can be obtained from the above expressions,

$$c = S^{-1}(t + \lambda n) \tag{112}$$

where

$$S_{ij} = \iint \frac{\chi_i(r)\chi_j(r)}{|r - r'|} dr dr', \tag{113}$$

$$t_i = \iint \frac{\chi_i(r)\rho(r')}{|r - r'|} dr dr' = \sum_{\mu}^{N} \sum_{\nu}^{N} P_{\mu\nu} \iint \frac{\chi_i(r)\chi_\mu(r)\chi_\nu(r)}{|r - r'|} dr dr', \tag{114}$$

$$n_i = \int \chi_i(r) dr, \tag{115}$$

and

$$\lambda = N - \frac{nS^{-1}t}{nS^{-1}n}. \tag{116}$$

In the fitting procedure outlined above, the time consuming step is obviously the construction of t, which requires the evaluation of three-centred two-electron integrals. It scales as N^2M. Since M is of the same order as N, this step formally scales as N^3. However, once $\tilde{\rho}(r)$ is in hand, it is used to ease the computational burden of evaluating the Coulomb repulsion integrals and results in a scaling of N^3. Now one is resigned to performing two N^3 steps during each self-consistent field iteration to construct the Coulomb repulsion integrals with the fitted density. For small molecules, it is preferable to have two N^3 steps rather than a single one that scales as N^4 (evaluation of the Coulomb repulsion energy without fitting). However, for very large systems, efficient cutoff schemes allow the reduction of the evaluation of these integrals to processes that scale as N^2 [46], whether or not a fitted density is used. At this point, fitting is not advantageous because two N^2 steps and inversion of S is still needed. Inversion always scales as M^3.

In this research, a DAC procedure for the fitting of charge density is implemented. Instead of carrying out a fitting procedure of the charge density for the whole molecule, the molecule is divided into subsystems and a series of fits are performed, one per subsystem. This means that only the auxiliary bases centered on the subsystem and buffer atoms are used in the fitting procedure of the charge density.

As mentioned before, the density matrix formulation is used in the DAC procedure for the fitted density. This means that the definitions of the subsystem density and partition function are those of equations (94) and (95), respectively. With this in mind, the prescription for the fitting of a subsystem density follows exactly that of the conventional fitting of the total charge density. The optimal set of fitting coefficients are now found through the re-expression of equation (109) as

$$\iint \frac{[\rho^{\alpha}(r)-\tilde{\rho}^{\alpha}(r)][\rho^{\alpha}(r')-\tilde{\rho}^{\alpha}(r')]}{|r-r'|}drdr'. \tag{117}$$

Remember that the superscript α indicates the charge density of the subsystem and not that of the whole molecule. The total density still needs to be appropriately normalized. Therefore, each fit must be normalized to the subsystem population. Therefore,

$$\int \tilde{\rho}^{\alpha}(r)\,dr = \sum_{A\in\alpha} Q_A \tag{118}$$

where Q_A is the Mulliken population of subsystem atom A. The Mulliken population is well suited for the normalization of the density because it uses the same partitioning philosophy as the density matrix formulation. Equation (112) is now expressed as

$$c^{\alpha} = (S^{\alpha})^{-1}(t^{\alpha} + \lambda^{\alpha} n^{\alpha}) \tag{119}$$

where all the variables carry the same definition as previously, but now denoted with the superscript α to indicate that the vectors and matrices span the basis functions of the buffer and subsystem atoms. The Lagrangian multiplier is given as

$$\lambda^{\alpha} = \sum_{A\in\alpha} Q_A - \frac{n^{\alpha}(S^{\alpha})^{-1}t^{\alpha}}{n^{\alpha}(S^{\alpha})^{-1}n^{\alpha}} \tag{120}$$

and the vector t^{α} by

$$t_i^{\alpha} = \sum_{\mu}^{N}\sum_{\nu}^{N} p_{\mu\nu}^{\alpha} P_{\mu\nu} \iint \frac{\chi_i(r)\chi_{\mu}(r)\chi_{\nu}(r)}{|r-r'|}drdr' \tag{121}$$

Since $p^{\alpha}{}_{\mu\nu}$ vanishes unless at least one basis function is centered on an atom of subsystem α, equation (121) reduces to

$$t_i^{\alpha} = \sum_{\mu\in\alpha}\sum_{\nu}^{N} p_{\mu\nu}^{\alpha} P_{\mu\nu} \iint \frac{\chi_i(r)\chi_{\mu}(r)\chi_{\nu}(r)}{|r-r'|}drdr' \tag{122}$$

where $p^{\alpha}{}_{\mu\nu}$ will equal 1 if ν is in the subsystem α and otherwise, 1/2. The cost of evaluating the new vector t^{α} will level off once the system size becomes large enough. The first summation of equation (122) is a summation over only the basis centred on the atoms of subsystem α and its buffer. Regardless of the total system size, this value will remain

fixed for each subsystem. The second summation of equation (122) is over all the basis functions, $\chi_\nu(r)$, seemingly indicating that the time spent evaluating t^α scales linearly with system size. However, if there is no appreciable overlap between $\chi_\mu(r)$ and $\chi_\nu(r)$, there will be no appreciable contribution to t^α. As the system size increases, the number of basis functions, $\chi_\nu(r)$, overlapping the basis functions, $\chi_\mu(r)$, of subsystem α will level off at a fixed value. The evaluation of the vector t^α will be independent of system size for very large molecules. Since a fit must still be carried out for each subsystem and the number of subsystems scales linearly with system size, the overall fitting procedure will scale linearly with system size.

DAC Fitting of the Exchange-Correlation Terms

A DAC procedure for the fitting of the exchange correlation energy was also developed. The exchange-correlation energy is expressed as

$$E_{xc}[\rho(r)] = \int \varepsilon_{xc}(r)\rho(r)dr \tag{123}$$

where $\varepsilon_{xc}(r)$ is the exchange correlation energy density. The contribution of exchange-correlation to the Kohn-Sham equations is given by,

$$\int \chi_\mu(r) v_{xc}(r) \chi_\nu(r) dr \tag{124}$$

where $v_{xc}(r)$ is the XC potential. Because of the complex nature of these functionals they cannot be integrated analytically, however, they could be integrated numerically with the use of a grid. In DeFT, the grid is used to fit both $\varepsilon_{xc}(r)$ and $v_{xc}(r)$ with an auxiliary basis of K uncontracted Gaussians,

$$\tilde{\varepsilon}_{xc}(r) = \sum_{l}^{K} e_l \chi_l(r) \tag{125}$$

and

$$\tilde{v}_{xc}(r) = \sum_{l}^{K} b_l \chi_l^*(r) . \tag{126}$$

The fits minimize the error in XC terms over the set of grid points:

$$\sum_{l}^{grid} [\varepsilon_{xc}(R_l) - \tilde{\varepsilon}_{xc}(R_l)]^2 W_l \tag{127}$$

and

$$\sum_{l}^{grid} [v_{xc}(R_l) - \tilde{v}_{xc}(R_l)]^2 W_l . \tag{128}$$

It should be noted that the uncontracted Gaussian auxiliary basis used here is a different basis from than that used to fit the charge density. However, an equal number of basis functions are invariably used in each fitting procedure. The fit coefficients that minimize the errors in the XC terms are given by

$$e = S^{-1} t^e \tag{129}$$

and

$$b = S^{-1} t^b \tag{130}$$

where the matrix elements of S are given by

$$S_{kl}^{-1} = \sum_{l}^{grid} \chi_k^*(R_l) \chi_l^*(R_l) W_l \tag{131}$$

and the elements of t^e and t^b are given by

$$t_l^e = \sum_{l}^{grid} \chi_l^*(R_l) \varepsilon_{xc}(R_l) W_l \tag{132}$$

and

$$t_l^b = \sum_{l}^{grid} \chi_l^*(R_l) v_{xc}(R_l) W_l . \tag{133}$$

Once the fitted terms are synthesized, the following approximations are made to the exchange-correlation terms and the analytical integration can be performed,

$$\int \varepsilon_{xc}(\boldsymbol{r})\rho(\boldsymbol{r})d\boldsymbol{r} \approx \int \tilde{\varepsilon}_{xc}(\boldsymbol{r})\rho(\boldsymbol{r})d\boldsymbol{r} = \sum_{\mu}^{N}\sum_{\nu}^{N} P_{\mu\nu}\sum_{l}^{K} e_l \int \chi_l^{*}(\boldsymbol{r})\,\chi_\mu(\boldsymbol{r})\,\chi_\nu(\boldsymbol{r})d\boldsymbol{r} \quad (134)$$

and

$$\int \chi_\mu(\boldsymbol{r}) v_{xc}(\boldsymbol{r})\chi_\nu(\boldsymbol{r})d\boldsymbol{r} \approx \int \chi_\mu(\boldsymbol{r})\tilde{v}_{xc}(\boldsymbol{r})\chi_\nu(\boldsymbol{r})d\boldsymbol{r} = \sum_{l}^{K} b_l \int \chi_l^{*}(\boldsymbol{r})\,\chi_\mu(\boldsymbol{r})\,\chi_\nu(\boldsymbol{r})d\boldsymbol{r}\,. \quad (135)$$

This results in the evaluation of three-centred overlap integrals and scales formally as N^2K. Since K is of the same order as N, it can be said that the scaling is N^3.

The goal of the DAC procedure is to achieve linear scaling, so a partitioning scheme of the XC terms is also necessary. Following Slater's work [35], it is also clear that $v_{xc}(\boldsymbol{r})$ is roughly proportional to the cube root of the electronic density,

$$v_{xc}(\boldsymbol{r}) \approx \rho(\boldsymbol{r})^{1/3}$$

Therefore, construction of subsystem $v_{xc}^{\alpha}(\boldsymbol{r})$ contributions to the total $v_{xc}(\boldsymbol{r})$ from the corresponding subsystem density is not possible. Mathematically this can be shown as,

$$[\rho^{\alpha}(\boldsymbol{r})+\rho^{\beta}(\boldsymbol{r})+\rho^{\gamma}(\boldsymbol{r})+\ldots+\rho^{\omega}(\boldsymbol{r})]^{1/3} \neq [\rho^{\alpha}(\boldsymbol{r})^{1/3}+\rho^{\beta}(\boldsymbol{r})^{1/3}+\rho^{\delta}(\boldsymbol{r})^{1/3}+\ldots+\rho^{\omega}(\boldsymbol{r})^{1/3}]$$

As a result, the total density must be constructed, then followed by evaluation of the XC potential. At this point, the XC potential can be decomposed into subsystem contributions

$$v_{xc}(\boldsymbol{R}_I) = \sum_{\alpha}^{subsystems} p^{\alpha}(\boldsymbol{r})v_{xc}(\boldsymbol{R}_I) = \sum_{\alpha}^{subsystems} v_{xc}^{\alpha}(\boldsymbol{R}_I)\,. \quad (136)$$

This is achieved using Yang's density formulation partition function, $p^{\alpha}(\boldsymbol{r})$, equation (88). The partition function localizes the corresponding subsystem contributions, to the total XC potential, about the atoms in the subsystem. It is now possible to use a fitting procedure for the subsystem XC potential using only auxiliary basis functions that span a subsystem and nearby buffer atoms.

Equations (127) and (128) now become

$$\sum_{l}^{grid}[\varepsilon_{xc}^{\alpha}(\boldsymbol{R}_I) - \tilde{\varepsilon}_{xc}^{\alpha}(\boldsymbol{R}_I)]^2 W_l \quad (137)$$

and

$$\sum_{I}^{grid}[v_{xc}^{\alpha}(\boldsymbol{R}_I)-\tilde{v}_{xc}^{\alpha}(\boldsymbol{R}_I)]^2 W_I \tag{138}$$

The new optimized subsystem fit coefficient vectors are defined as

$$e^{\alpha}=(S^{\alpha})^{-1}(t^{e})^{\alpha} \tag{139}$$

and

$$b^{\alpha}=(S^{\alpha})^{-1}(t^{b})^{\alpha} \tag{140}$$

Once the coefficient vectors are calculated, each subsystem coefficient vector is summed to create a coefficient vector for the entire molecule. This vector is then used to calculate the XC contributions to the total energy, as well as to the KS matrix elements. Following the same reasoning as before for the fit to the charge density, this procedure will scale linearly with the total size of the system.

Results and Discussion

The tests of the DAC approaches to fitting the charge density and the exchange-correlation potential were performed with a modified version of DeFT. The test calculations were performed with the extended conformation of the glycine heptapeptide, shown in Figure (7), that had been geometry optimized with PM3 [47]. The peptide was divided into eight subsystems: a terminal HCO subsystem, six central $NHCH_2CO$ subsystems, and a terminal NH_2 subsystem. All calculations were performed within the Local Spin Density Approximation using a double-ζ plus polarization orbital basis for the hydrogen atoms and triple-ζ plus polarization orbital basis for the C, N and O atoms. The auxiliary basis sets used for fitting the density were three s functions and a set of s, p and d functions constrained to have the same exponent [34] for the hydrogen atoms. Likewise, four s functions and three sets of s, p and d functions were used for the C, N and O atoms. In all, this leads to the use of 618 orbital basis functions and 1191 auxiliary basis functions for each of the fits.

In the calculations that were performed, a series of buffer spaces were used to study the effect of buffer space size on errors in the total energy. The buffer spaces were set by simply keeping all atoms located in a predetermined number of nearest neighboring subsystems. The buffer of a subsystem ranged from using no buffer space at all, to using all the atoms no more than four subsystems away.

Since this was just a test to see if the DAC procedure could be applied to the fits of the LCGTO-DF method, the implementation was very crude. In the case of the fitting of $\rho(r)$, the prescription for partitioning of the density matrix, discussed early, was introduced in DeFT. Other required data were the molecular orbital coefficients obtained from a conventional calculation and Mulliken population analysis, so as to have the electronic population of each subsystem. Eight job cards were set up, one for each subsystem of the molecule. The molecular orbital coefficients were read into the calculation, as well as the electronic population of a subsystem. The calculation results in obtaining the DAC charge

density fit coefficients for that particular subsystem. The DAC charge density fit coefficients were collected from each of the eight jobs and were re-ordered into their proper position in the fit coefficient vector, so that the eight sets of coefficients could be added together. The reason why the positions of the coefficients in the array must be re-adjusted is because the order of the coefficients becomes disrupted when the auxiliary basis centered on atoms not in a subsystem or its buffer are discarded. This results in no fitting coefficients corresponding to the auxiliary basis that had been removed by the DAC procedure. This is not a true test of the DAC procedure since it is not being used in an SCF procedure. It is simply run for one iteration to see the effect on the fit to $\rho(r)$.

Since the buffers that are tested are simply atoms of neighboring subsystems, it would be useful to know the approximate size of each subsystem. Knowing the size of the subsystems used as buffers will allow one to get a feel for the distances to which buffer spaces must be extended to ensure that errors for a DAC procedure are kept under control when dealing with globular peptides rather than extended ones.

Subsystem	Size (Å)
1	1.3898
2	2.7570
3	2.9337
4	2.7240
5	2.9151
6	2.7084
7	2.9074
8	1.2415
Total	19.5769

Table 2. The size of each subsystem of the glycine heptapeptide.

Table 3 shows the errors in the fitting coefficients of the charge density. Using various buffer spaces, the errors in the DAC procedure relative to the conventional DF calculation are listed. The first column of the table indicates how many of the neighboring subsystems were included in the buffer space in each direction. At four subsystems, the buffer of the central group extends over the entire molecule.

	Average Absolute Errors on Charge Density Coefficients	Average Percent Error on Charge Density Coefficients
No Buffer	0.0738744	480.3474
1	0.0044144	11.4965
2	0.0018761	4.1106
3	0.0002345	0.4395
4	0.0001001	0.1903

Table 3. Errors introduced in the charge density fitting coefficients by the DAC algorithm.

Since it is difficult to get a feel for the errors given in Table 3 and potential energy surfaces are of more interest, more tests were conducted. With the DAC charge density fitting coefficients, new jobs were set up where conventional DF was used but the DAC charge density fitting coefficients were read into the calculation during each iteration of the SCF procedure until convergence was re-achieved. This was done to obtain a feel for what effect the errors from the DAC charge density fitting coefficients would have on the total energy of the glycine heptapeptide. The results are summarized in Table 4, where the first column indicates the number of neighboring subsystems used as a buffer. The second column gives the total number of subsystem and buffer atoms associated with each subsystem of the molecule, progressing from left to right, in Figure 7. All entries in the total energy have been converged to 1.0×10^{-7} hartrees. Remember that the DAC fit was

not allowed to adjust, so this is not an entirely fair test of the DAC fitting procedure. However, the results are clearly encouraging.

	# atoms (subsystem+buffer)	Total Energy (hartrees)	Error (kcal mol^{-1})
Reference		-1407.1545837	
0	3,7,7,7,7,7,7,3	-1430.2651260	14503
1	10,17,21,21,21,21,17,10	-1407.1545741	9.6411
2	17,24,31,35,35,31,24,17	-1407.1545850	0.0060
3	24,31,38,45,45,38,31,24	-1407.1545837	0.0000
4	31,38,45,48,48,45,38,31	-1407.1545837	0.0000

Table 4. The error in the total energy introduced from DAC fitting procedure of the charge density.

It is quite clear that the use of a buffer is mandatory from the huge error seen in the calculation without a buffer. The introduction of a buffer including the atoms of the neighboring subsystems, dramatically reduces the error created by the DAC procedure. Inclusion in the buffer of the atoms of the two neighboring subsystems on either side reduces the error in the total energy enough to be acceptable for most applications. This would correspond to cutoff for the buffer zone to something between 3.0 - 6.0 Å on either side of the subsystem of interest. Extending the buffer to 3 or 4 subsystems on either side gives the same energy as the conventional DF calculation (within the convergence criterion of 10^{-7} hartrees). In light of this, it was decided that a DAC procedure for the fitting of the electronic density is quite viable. A fully automated procedure was developed and is discussed later.

Exactly the same procedure was carried out for the DAC exchange-correlation fitting procedure. In Table 5, the errors of the exchange-correlation fitting coefficients are tabulated.

	Average Absolute Errors on Exchange-Correlation Coefficients	Average Percent Error on Exchange-Correlation Coefficients
No Buffer	0.51077	670.3858
1	0.40999	185.0470
2	0.17447	50.8231
3	0.13032	33.1307
4	0.06936	16.4991

Table 5. Errors introduced in the exchange-correlation fitting coefficients by the DAC algorithm.

At first glance it would appear that the large errors of the XC fitting coefficients would be clearly unacceptable resulting in huge errors in the total energy. However, subsequent calculations, summarized in Table 6, where the DAC XC fitting coefficients are read into a conventional DF calculation, yield good results.

	# atoms (subsystem+buffer)	Total Energy	Error (kcal mol^{-1})
Reference		-1407.1545837	
0	3,7,7,7,7,7,7,3	-1407.1392210	9.6411
1	10,17,21,21,21,21,17,10	-1407.1544409	0.0896
2	17,24,31,35,35,31,24,17	-1407.1545804	0.0020
3	24,31,38,45,45,38,31,24	-1407.1545849	0.0007
4	31,38,45,48,48,45,38,31	-1407.1545825	0.0008

Table 6. The error in the total energy introduced from DAC fitting procedure of the exchange-correlation potential.

The errors appear to be relatively small in the total energy for the fitting of the exchange-correlation considering that the errors in the fitting coefficients are much larger than those of the charge density fitting. The smaller error is expected since the exchange-correlation

energy contributes little to the total energy of a molecule in comparison to the Coulomb repulsion. Again, extending the buffer to two subsystems on either side would seem to be sufficient. Note that the error is not brought down to zero even if the buffer is taken out as far as four subsystems.

Automated DAC Procedure

Given that the DAC procedure for the fitting of the charge density and exchange-correlation appears to be accurate, a fully automated self-consistent procedure was implemented in DeFT. In such a method, only one job card is submitted, where each subsystem is defined in the input deck. During each iteration of the SCF procedure, a loop over the number of the subsystems is performed to calculate the subsystem contributions to the charge density and the exchange-correlation potential. Subsequently, the contributions from each subsystem are pieced together resulting in a global expression for the charge density and the exchange-correlation potential of the molecule. This procedure is repeated at each iteration in an SCF procedure.

In these calculations a series of buffer spaces were tested with buffer space cutoffs of 2.0Å to 14.0Å, as well as, no buffer at all. The 2.0 Å buffer cutoff is equivalent to having only atoms directly bound to a subsystem added to the buffer space. In the 14.0Å buffer cutoff, the central subsystem's buffer encompasses the entire molecule. The results for the automated procedure of the DAC fitting of charge density are tabulated in Table 7. As before, the PM3 geometry was used. The DF calculations were performed using the same orbital and auxiliary basis with the total energies converged to 10^{-7} hartrees.

Buffer Space Cutoff (Å)	# atoms (subsystem + buffer)	Total Energy (hartrees)	Error (kcal/mol^{-1})
true		-1407.1545851	
no buffer space	3,7,7,7,7,7,7,3	-1407.1709566	10.2742
2.0	4,9,9,9,9,9,9,4	-1407.1553139	0.4574
3.0	8,15,17,17,17,17,15,8	-1407.1546644	0.0498
4.0	9,16,20,20,20,20,16,9	-1407.1546199	0.0218
5.0	11,18,23,23,23,23,18,11	-1407.1545932	0.0051
6.0	12,19,26,25,25,26,19,12	-1407.1545920	0.0043
7.0	15,22,29,31,32,29,22,15	-1407.1545892	0.0026
8.0	17,24,31,35,35,31,24,17	-1407.1545877	0.0016
9.0	19,26,33,39,39,33,26,19	-1407.1545856	0.0003
10.0	22,29,36,43,43,36,29,22	-1407.1545856	0.0003
11.0	22,29,36,43,43,36,29,22	-1407.1545856	0.0003
12.0	24,31,38,45,45,38,31,24	-1407.1545854	0.0002
13.0	26,33,40,47,47,40,33,26	-1407.1545854	0.0002
14.0	29,36,43,48,48,43,36,29	-1407.1545853	0.0001

Table 7. The effect of buffer space on the error in total energy introduced by the DAC charge density fitting procedure.

As can be seen from Table 7, the error (the difference between the DAC and conventional energies) arising from the DAC procedure when no buffer space is used, is clearly too large. Introduction of a small buffer space of 2.0 Å sees the error drop by an order of magnitude. The 2.0 Å buffer is the equivalent of adding atoms that are directly bound to the subsystem. This error of 0.46 kcal mol^{-1} is still too large and larger buffer spaces should be employed. When a buffer space of 3.0 Å was used, the error dropped to just below 0.05 kcal mol^{-1}. This is acceptable for a wide range of applications. To reduce the

error to below 0.01 kcal mol^{-1}, a buffer space of 5.0 Å is required. At this point, no more than half of the atoms in the molecule are associated with any one subsystem. Finally, to reduce the error another order of magnitude the buffer cutoff must be extended to 9.0 Å. In the test cases with cutoffs greater than 9.0 Å, the heptapeptide's central subsystems have fits to the charge density involving 39-48 atoms. These subsystems span virtually the entire molecule. At 14.0 Å, the central subsystems indeed span the entire molecule with an error of 10^{-4} kcal mol^{-1} in the total energy. The conventional LCGTO-DF result is not fully recovered due to the fact that the subsystems associated with the terminal ends of the peptide still only span a fraction of the molecule. A buffer cutoff any larger than 9.0 Å does not significantly reduce the error in the energy and the DAC approach becomes clearly inefficient as the buffer essentially covers the entire molecule.

In a similar manner to the DAC fitting of charge density, a fully automated DAC procedure was implemented for the fitting of the exchange-correlation terms. The results obtained for the DAC fits to both charge density and exchange-correlation are summarized in Table 8. In these results, a buffer space had to be extended to 6.0 Å to have an error in the energy lower than 0.05 kcal mol^{-1}. This error corresponded to a buffer space of 3.0 Å in the DAC fitting of the charge density. With a buffer space of 7.0 Å and larger, the error is quite acceptable with values under 0.005 kcal mol^{-1}. We also note that the total energy is sometimes underestimated and sometimes overestimated. With only the DAC fit to $\rho(\boldsymbol{r})$, it was always underestimated. This is consistent with the fact that the charge density fitting procedure is variational while the XC fits are not. Also, a large buffer space is needed for the XC DAC fits. This is expected due to the XC fitting functions being more diffuse than the charge density fitting functions.

Buffer Space Cutoff (Å)	# atoms (subsystem + buffer)	Total Energy (hartrees)	Error (kcal/mol^{-1})
true		-1407.1545851	
no buffer space	3,7,7,7,7,7,7,3	-1407.1472272	4.6175
2.0	4,9,9,9,9,9,9,4	-1407.1513734	2.0155
3.0	8,15,17,17,17,17,15,8	-1407.1542530	0.2084
4.0	9,16,20,20,20,20,16,9	-1407.1544430	0.0892
5.0	11,18,23,23,23,23,18,11	-1407.1544780	0.0672
6.0	12,19,26,25,25,26,19,12	-1407.1545177	0.0423
7.0	15,22,29,31,32,29,22,15	-1407.1545830	0.0013
8.0	17,24,31,35,35,31,24,17	-1407.1545796	0.0034
9.0	19,26,33,39,39,33,26,19	-1407.1545822	0.0018
10.0	22,29,36,43,43,36,29,22	-1407.1545855	0.0003
11.0	22,29,36,43,43,36,29,22	-1407.1545855	0.0003
12.0	24,31,38,45,45,38,31,24	-1407.1545854	0.0002
13.0	26,33,40,47,47,40,33,26	-1407.1545861	0.0006
14.0	29,36,43,48,48,43,36,29	-1407.1545858	0.0004

Table 8. The effect of buffer space on the error in total energy introduced by the DAC exchange-correlation fitting procedure.

As a final note, the total energy of the glycine heptapeptide in Tables (4) and (6) is different from the total energy listed in Tables (7) and (8). This is because the latter test cases did not make use of canonical orthogonalization of the charge density and XC auxiliary basis sets. For large systems and the conventional approach, canonical orthogonalization is performed in DeFT to remove linear dependencies that hinder convergence. In the DAC procedures, linear dependencies are not a problem since a large fitting basis is never used for any one calculation. For testing the automated DAC fits,

canonical orthogonalization had to be turned off, thus yielding a different final total energy. Turning off canonical orthogonalization did severely hamper the convergence of the conventional LCGTO-DF calculations in these large systems.

Conclusions

Density functional methods may be the most realistic approach for *ab initio* studies of large molecules. The use of gradient-corrected XC functionals yields results that consistently surpass those of HF. In fact, with today's XC functionals, one can obtain results with roughly the same accuracy as MP2 theory for organic systems [20]. At the same time, calculations are performed more rapidly than those of their HF counterparts. Consequently, the use of correlated post-HF calculations is restricted to smaller systems than those modeled by gradient-corrected DF theory. This should clearly lead a computational chemist to DF methods for large systems.

The greatest drawback of DF theory lies in the XC functionals. As stated earlier, the XC functionals of Vosko-Wilk-Nusair and Perdew-Zunger are at the limit of the LSDA. Subsequent gradient-corrected functionals fail to systematically improve DF theory results. In conventional *ab initio* theory, there exists a well established hierarchy of formalisms that proceed towards more accurate results. However, the use of these formalisms is restricted by the computational power of today's computers. With the advent of faster computers, these approaches will clearly lead to more reliable results. The problem with DF theory is that no such hierarchy currently exists. The only solution to this problem is more effort in finding better XC functionals. Presently, much work is focused in this area. However, it is not clear what new computational burden will be associated with these new functionals.

A crude parallel version of DeFT has been developed with PVM as the message-passing harness and needs to be extensively tested on various architectures to test parallel efficiency, as well as, to ensure portability. The primary bottlenecks of any DF calculation are addressed in this parallel code. This includes parallelization of the computationally demanding Coulomb repulsion and exchange-correlation integrals which formally scale as N^3. Also, generation of the charge density, $\rho(r)$, on a numerical grid proposed by Becke was targeted for parallelization. Due to the nature of the calculations, the master/slave

scheme was adopted for this parallel implementation. It is believed that the parallel version of DeFT is not suited for smaller systems. This can be explained by the amount of time required for the packing of data into a buffer, the sending of the buffer to the slave and the unpacking of the buffer by the slave. The same procedure must be performed to send the data back to the master. This communication requires a substantial amount of CPU time relative to the actual calculation. It is for this reason that as the system size increases, so will the parallel efficiency.

The DAC approach employed in DeFT was focused on the LCGTO-DF fitting procedures. This was carried out using modified versions of Yang's density and density matrix formulations for the fitting of $v_{xc}(r)$ and $\rho(r)$, respectively. This work is only part of a yet to be fully developed DAC algorithm that will be employed in DeFT. Future work entails using a DAC approach for the solution of the KS equations and the evaluation of the LCGTO-DF energy gradients. Other work related to the DAC approach concerns the optimization of the code for parallel computing.

Through the study of the glycine heptapeptide, a good appreciation for the errors introduced by the DAC approach was gained. It was felt that any discrepancies between the conventional and DAC LCGTO-DF should be regarded as a problem with the buffer space that is used. To alleviate this problem, buffer space size is adjusted until the error introduced by the DAC approach is brought to an acceptable level.

The combination of systematic improvements in XC functionals and more efficient codes should give DF theory an advantage over its HF and post-HF counterparts; at the same time allowing insight into larger and larger systems, that have been previously out of reach for *ab initio* methods.

Appendix

Principle Subroutines of PVM

Perhaps one of the best features of PVM is the fact that there are really only a few commands needed to create a PVM application. The main functions needed are the initiation of the sending of data, the receiving of data, and the packing and unpacking of data from the buffer.

pvmfmytid()

FORTRAN Format call pvmfmytid(tid)
Parameters:

tid This is the task identifier which is an integer associated with the calling PVM process. The **tid** is a 32 bit positive integer created by the pvm daemon that encodes information about the location of a process on the virtual machine, the CPU number in the case of a multiprocessor machine. This information is used by PVM for organization purposes and is not used by the applications or user.

pvmfspawn()

FORTRAN Format call pvmfspawn (task, flag, where, ntask, tids, numt)
Parameters:

task This is a character string which is the executable file name of the process to started. The executable must reside on the machine where the process is to be started in the directory $HOME/pvm3/bin/ARCH/**task**.

flag This is an integer specifying the following spawn options:

PVMDEFAULT	0	PVM can choose any machine to start a task
PVMHOST	1	where specifes a particular host
PVMARCH	?	where specifies a type of architecture

where A character string specifying where to start a process. Depending on the value of the flag, either a specific host name is given or an architecture type. In the case of using the default flag, PVM will select the most appropriate machine.

ntask An integer number indicating the number of executable copies to be started on the virtual machine.

tids This is an integer array of length of ntasks containing the tids of the PVM processes that were started. If there was an error starting a task, then an error code will appear in the array corresponding to the task.

numt An integer returning the actual number of tasks started. A negative value indicates a system error, a positive value less than **ntask** indicates partial failure.

pvmfinitsend()

FORTRAN Format call pvmfinitsend(encoding,bufid)

Parameters:

encoding An integer indicating the messages encoding scheme.

PVMDEFAULT	0	XDR if heterogeneous
PVMRAW	1	no encoding
PVMINPLACE	2	data left in place

bufid An integer returned that contains the message buffers identifier.

The routine pvmfinitsend's main task is to clear the send buffer and prepare it for packing a new message.

pvmfsend()

FORTRAN Format pvmfsend(tid, msgtag, bufid)

Parameters:

tid An integer task identifier of the sending process.

msgtag A message tag supplied by the user to all the application to distinguish between different messages.

bufid An integer assigned to identify the new active receive buffer.

pvmfrecv()

FORTRAN Format pvmfrecv(tid, msgtag, bufid)

The parameters in the pvmfrecv routine have the same definition as those in the pvmfsend routine. The values parameters in the two routines should be the same.

pvmfpack()

FORTRAN Format pvmfpack(what,xp,nitem,stride,info)

Parameters:

what An integer that specifies the type of data to be packed.

STRING	0	REAL4	4
BYTE1	1	COMPLEX8	5
INTEGER2	2	REAL8	6
INTEGER4	3	COMPLEX16	7

xp A pointer to the beginning of a block of bytes

nitem The total number of items to be packed.

stride The stride to be used when packing the items. For example, typically a value of one is used so as to pack every item in an array.

info A status code returned by the routine where values less than zero indicate an error.

pvmfunpack()

FORTRAN Format pvmfunpack(what, xp, nitem, stride, info)

The parameters in the pvmfunpack routine have the same definition as those in the pvmfpack routine. In fact they should have the same values as their packing counterpart in a PVM application.

References

[1] R.G. Parr and W. Yang, *Density-Functional Theory of Atoms and Molecules*, (Oxford University Press, New York, N.Y., 1989).

[2] T. Ziegler, *Chem. Rev.*, **91**, 651 (1991).

[3] J. Andzelm and E. Wimmer, *J. Chem. Phys.*, **96**, 1280 (1992).

[4] N H. March, *Electronic Density Theory of Atoms and Molecules*, (Academic Press, New York, N.Y., 1992).

[5] P. Hohenberg and W. Kohn, *Phys. Rev. B*, **136**, 864 (1964).

[6] W. Kohn and L.J. Sham, *Phys. Rev. A*, **140**, 1133 (1965).

[7] E.J. Baerends, D.E. Ellis and P. Ros, *Chem. Phys.*, **2**, 41 (1973).

[8] A. St-Amant, Ph.D. thesis, Université de Montréal (1992). deMon, un programme LCGTO-DF, et une étude théorique de l'adsorption de l'hydrogène sur les monomères et dimères de Ni, Rh et Pd.

[9] A. St-Amant, the complete source code and documentation for the DeFT program package may be downloaded from http://www.chem.uottawa.ca/DeFT.html.

[10] B.G. Johnson, P.M.W. Gill and J.A. Pople, *J. Chem. Phys.*, **98**, 5612 (1993).

[11] B. Delley, *J. Chem. Phys.*, **92**, 508 (1990).

[12] M.P. Teter, M.C. Payne and D.C. Allan, *Phys. Rev.*, **B40**, 12255 (1989).

[13] A.D. Becke, *Int. J. Quantum Chem., Quantum Chem. Symp.*, **23**, 599 (1989).

[14] G.A. Geist, A. Beguelin, J.J. Dongarra, Weicheng Jiang, R. Manchek and V.S. Sunderam. "PVM3 User's Guide and Reference Manual", Technical Report ORNL/TM-12187, Oak Ridge National Laboratory, Oak Ridge, Tennessee, May 1993.

[15] W. Yang, *J. Mol. Struct. (THEOCHEM)*, **225**, 461 (1992).

[16] W. Yang and T.S. Lee, *J. Chem. Phys.*, **103**, 5674 (1995).

[17] A.D. Becke, *Phys. Rev. A*, **38**, 3098 (1988).

[18] C. Lee, W. Yang, and R.G. Parr, *Phys. Rev. B*, **37**, 785 (1988).

[19] S.H. Vosko, L. Wilk, and M. Nusair, *Can. J. Phys.*, **58**, 1200 (1980).

[20] A. St-Amant, W.D. Cornell, T.A. Halgren and P.A. Kollman, *J. Comp. Chem.*, **16**, 1483 (1995).

[21] D. M. Ceperley and B. J. Alder, *Phys. Rev. Lett.*, **45**, 566 (1980).

[22] P.A.M. Dirac, *Proc. Cambridge Phil. Soc.*, **26**, 376 (1930)

[23] J. P. Perdew and A. Zunger, *Phys. Rev. B*, **23**, 5048 (1981).

[24] I. Papai, A. St-Amant, J. Ushio and D. Salahub, *Int. J. Quantum Chem.*, **S24**, 29 (1990).

[25] J.P. Perdew, *Phys. Rev B*, **33**, 8822 (1986).

[26] F. Sim, A. St-Amant, I. Papai, and D.R. Salahub, *J. Am. Chem. Soc.*, **114**, 4391 (1992).

[27] P.Mlynarski and D.R. Salahub, *Phys. Rev. B*, **43**, 1399 (1991).

[28] J.P. Perdew and Y. Wang, *Phys. Rev. B*, **33**, 8800 (1986).

[29] J. Perdew, J.A. Chevary, S.H. Vosko, K.A. Jackson, M.R. Pederson, D.J. Singh and C. Fiolhais, *Phys. Rev. A*, **46**, 6671 (1992).

[30] A. St-Amant, "Practical Density Functional Approaches in Chemistry and Biochemistry", *Quantum Mechanical Simulation Methods For Studying Biological Systems*, D. Bicout and M. Field, Eds. (Les Editions de Physique, France,1996)

[31] A. St-Amant, "Density Functional Methods in Biomolecular Modelling", *Reviews in Computational Chemistry*, **Volume 7**, K.B. Lipkowitz and D.B. Boyd, Eds. (VCH Publishers, New York, 1995).

[32] H. Sambe and R.H. Felton, *J. Chem. Phys.*, **62**, 1122 (1975).

[33] B. I. Dunlap, J.W.D. Connolly, and J.R. Sabin, *J. Chem. Phys.*, **71**, 3396 (1979).

[34] N. Goudbout, D.R. Salahub, J. Andzelm, and E. Wimmer, *Can. J. Chem.*, **70**, 560 (1992).

[35] J.C. Slater, *Phys. Rev.*, **81**, 385 (1951).

[36] A. Becke, *J. Chem. Phys.*, **88**, 2547 (1988).

[37] *Handbook of Mathematical Functions*, M. Abramowitz and I.A. Stegun, Eds. (Dover, New York, 1970).

[38] A.H. Stoud, *Approximate Calculation of Multiple Integrals* (Prentice Hall, Englewood Cliffs, 1971).

[39] B.G. Johnson, P.M.W. Gill and J.A. Pople, *Chem. Phys. Lett.*, **220**, 377 (1994).

[40] S. Obara and A. Saika, *J. Chem. Phys.*, **84**, 3963 (1986).

[41] M. Head-Gordon and J.A. Pople, *J. Chem. Phys.*, **89**, 5777 (1988).

[42] R.T. Gallant and A. St-Amant, *Chem. Phys. Lett.*, **256**, 569 (1996).

[43] A. St-Amant, S.K. Goh and R.T. Gallant, "A Divide-and-Conquer Implementation of the Linear Combination of Gaussian-Type Orbitals Density Functional (LCGTO-DF) Method", (Elsevier Science Publishers B.V., Amsterdam, *to be published in 1996*).

[44] R.S. Mulliken, *J. Chem. Phys.*, **23**, 1833 (1955).

[45] B. Besler, K. Merz and P. Kollman, *J. Comp. Chem.*, **4**, 431 (1990).

[46] M. Haser and R. Ahlrichs, *J. Comp. Chem.*, **10**, 104 (1989).

[47] J.P. Stewart, *J. Comp. Chem.*, **10**, 209 (1989).