Science and Engineering Research Council
Rutherford Appleton Laboratory
Chilton DIDCOT Oxon OX11 0QX

RAL-92-034

Electron correlation in molecules: *concurrent computation* Many-Body Perturbation Theory (*cc*MBPT) calculations using macrotasking on the NEC SX-3/44 computer

D Moncrieff V R Saunders and S Wilson

June 1992

Electron correlation in molecules: *concurrent computation* Many-Body Perturbation Theory (*cc*MBPT) calculations using macrotasking on the NEC SX-3/44 computer [§]

David Moncrieff,
Supercomputer Computations Research Institute, Florida State University, Tallahassee, Florida, U.S.A.

Victor R. Saunders,
Daresbury Laboratory, Daresbury, WA4 4AD, U.K.

Stephen Wilson
Rutherford Appleton Laboratory, Chilton, Oxfordshire OX11 0QX, U.K.

ABSTRACT

The *ab initio* determination of the electronic structure of molecules is a many-fermion problem involving the approximate description of the motion of the electrons in the field of fixed nuclei. It is an area of research which demands considerable computational resources but having enormous potential in fields as diverse as interstellar chemistry and drug design, catalysis and solid state chemistry, molecular biology and environmental chemistry. Electronic structure calculations almost invariably divide into two main stages: the approximate solution of an independent electron model, in which each electron moves in the average field created by the other electrons in the system, and then, the more computationally demanding determination of a series of corrections to this model, the electron correlation effects. The many-body perturbation theory expansion affords a systematic description of correlation effects, which leads directly to algorithms which are suitable for concurrent computation. We term this *concurrent computation* Many-Body Perturbation Theory (*cc*MBPT). The use of a dynamic load balancing technique on the NEC SX-3/44 computer in electron correlation calculations is investigated for the calculation of the most demanding energy component in the most accurate of contemporary *ab initio* studies. An application to the ground state of the nitrogen molecule is described. We also briefly discuss the extent to which the calculation of the dominant corrections to such studies can be rendered computationally tractable by exploiting both the vector processing and parallel processor capabilities of the NEC SX-3/44 computer.

[§] *An entry to the Canadian Supercomputing Challenge, 1992.*

1. Introduction.

The *ab initio* determination of the electronic structure of molecules is a many-fermion problem involving the approximate description of the motion of the electrons in the field of fixed nuclei [1]. It is an area of research which demands considerable computational resources but having enormous potential in fields as diverse as interstellar chemistry and drug design, catalysis and solid state chemistry, molecular biology and environmental chemistry. Molecular electronic structure calculations are finding an increasingly wide range of applications in chemistry and allied sciences in both academic and industrial research programmes. Their cost-effectiveness is widely recognized. Although the first molecular electronic structure calculations were carried out soon after the advent of modern quantum theory, it is only in the past decade or so that computing machines have become powerful enough to facilitate a broad range of useful applications.

Electronic structure calculations almost invariably divide into two main stages: the approximate solution (within the algebraic approximation arising from the use of a finite basis set expansion) of an independent electron model (usually the Hartree-Fock model), in which each electron moves in the average field created by the other electrons in the system, and then, the determination of a series of corrections to this model, the so-called electron correlation effects. The many-body perturbation theory expansion affords a systematic description of correlation effects, which, in addition to having the important theoretical property that calculated energies and other properties scale linearly with the number of electrons in the system, leads directly to algorithms which are suitable for concurrent computation. We term this *concurrent computation* Many-Body Perturbation Theory (*cc*MBPT) [2].

Accurate contemporary electron correlation calculations take account of correlation effects up to fourth order in the fluctuation potential (the correction to the independent electron model hamiltonian). The most demanding of the fourth order terms are those involving triply excited intermediate states and lead to algorithms which scale as the seventh power of the number of basis functions employed when invoking the algebraic approximation. The calculation of these terms was rendered tractable for the first time by constructing an algorithm capable of exploiting the vector processing capabilities of the CRAY 1 [3]. However, early calculations demonstrated that the fourth order triple excitation terms can be particularly sensitive to the quality of the basis set employed [4].

In a recent years, research has been directed towards the development of algorithms for many-body perturbation theory calculations which can effectively exploit multi-vector processor machines with shared memory. In previous work, we have explored the use of the CRAY X-MP [5], Y-MP [6] and C-90 [7] and the IBM 3090 VF [2,8]. The performance of the distributed memory Intel GAMMA and DELTA machines has also been investigated [9]. Of the implementations made to date the C-90 has yielded the maximum rate of computation of ~6 GFLOPS on 8 dedicated processors and an execution rate in excess of ~13 GFLOPS on a 16 processor machine.

In this work, we investigate the use of the NEC SX-3/44 computer which has a theoretical peak performance of 22 GFLOPS. This level of performance will

enable the use of extensive basis sets and thus a significant reduction of basis set truncation effects.

We also discuss the extent to which the calculation of the dominant corrections to the fourth order treatment which defines the most accurate contemporary treatments of electron correlation energies can be rendered computationally tractable on the most powerful vector multi-processor supercomputer available at present. Experience has repeatedly demonstrated the importance of evaluating all terms in the perturbation series through some order. The most computationally demanding of the fifth order terms are those which involve two triply excited intermediate states. The evaluation of these energy components scale as the eighth power of the number of basis functions used. We suggest that fifth order calculations with basis sets of adequate size can be carried out using macrotasking techniques which have proved useful in fourth order studies.

2. *cc*MBPT

In common with many areas of modern theoretical science, the determination of molecular electronic structure from first principles demands the use of state-of-the-art supercomputers [10] and the development of effective algorithms for these machines demands the exploitation of concurrent computing methods [11].

The first published many-body perturbation theory codes [12-14] for atomic and molecular electronic structure calculations using finite basis set expansions, which were employed in many of the early applications [15-20] of the method, were written for a scalar processing computer (the IBM 360/91). Furthermore, because of the limited memory available, the algorithms employed in these codes were constructed in such a way that only a small fraction of the two-electron integrals were required in the computer's central memory at any one time.

The linked diagram theorem is of central importance to the many-body perturbation theory. This theorem was developed by Brueckner [21], Goldstone [22], Hugenholtz [23] and others (see, for example, [24]) from the quantum field theory of Feynman [25], Dyson [26] and Wick [27]. It facilitates the subdivision of a many-electron system into a number of comparatively small subsystems which can be considered independent of each other.

For the idealized system of, say, a large number of non-interacting atoms the subsystems are *completely independent* and the level shift, ΔE, (that is, the correction to the zero-order energy) is just the sum of the subsystem energy shifts, ΔE_i,

$$\Delta E = \sum_i \Delta E_i$$

Other expectation values can also be calculated as the sum of contributions from each of the subsystems. The energy and other expectation values are thus strictly proportional to the number of electrons in the system, N; $\Delta E \propto N$.

For a more realist many-electron system, such as an array of interacting atoms, the subsystems are, of course, not completely independent. However, the correlation effects associated with a particular linked diagram behave as if they were an independent subsystem. As Brandow [28] has emphasized "The essential point is that the interactions between different clusters are *strictly negligible*, by definition." Any interaction between different diagrammatic terms is associated with higher order terms involving all particles in the original diagrams. These interactions can be either dynamical, that is involving the fluctuation potential, or statistical, that is through the common set of intermediate states. Thus for an actual many-electron system, the energy and other expectation values are thus also strictly proportional to the number of electrons in the system.

The theoretical importance of the linked diagram theorem and the consequential linear scaling of the energy and other expectation values with the number of electrons in the system is widely recognized in molecular physics and theoretical chemistry [29-36] and many contemporary treatments of the correlation problem in molecules are based on a perturbation expansion [37,38]. The sum-over-states expressions arising in each of the terms in the perturbation series can be written as matrix multiplications and, therefore, implemented with near optimal efficiency on vector processing computers [3,39]. The linked diagram theorem also has important computational consequences which are particularly relevant when parallel processing machines are employed [5,35,36,40-42]. The many-electron problem is decoupled into completely independent subproblems each of which can be handled on a separate processor.

As an example, we consider the fourth order energy components corresponding to diagrams containing triply excited intermediate states. These terms are the most demanding of the fourth order terms leading to an algorithm which scales as the seventh power of the number of basis functions. In particular, we consider the diagram labelled F_t in previous work [43,44] and shown in Figure 1. A set of rules has been presented previously [44] for writing down the algebraic expression corresponding to such diagrams. The indices i,j,k,... are used to label single particle state functions lying below the Fermi level (*i.e.* occupied spin orbitals) and the indices a,b,c,... to label single particle state functions lying above the Fermi level (*i.e.* unoccupied spin orbitals). p,q,r,... are used to label arbitrary single particle functions. For closed-shell systems, $2N_{occ}$ defines the number of occupied spin orbitals and $2N_{virt}$ the number of virtual, or unoccupied, spin orbitals. Using these rules, the algebraic expressions for the energy components corresponding to diagram F_t may be written as follows

$$E_4(F_t) = \frac{1}{4} \sum_{\substack{ijk \\ abcde}} \frac{\langle ij|\hat{O}|ab\rangle \langle ak|\hat{O}|cd\rangle \langle cd|\hat{O}|ek\rangle \langle eb|\hat{O}|ij\rangle}{D_{ijab} D_{ijkbcd} D_{ijbe}}$$

where the numerators in the summand are products of the two-electron integrals

$$\langle pq|\hat{O}|rs\rangle = \iint dr_1\, dr_2\, \phi_p(r_1)\, \phi_q(r_2)\, r_{12}^{-1} \{\phi_r(r_1)\, \phi_s(r_2) - \phi_s(r_1)\, \phi_r(r_2)\}$$

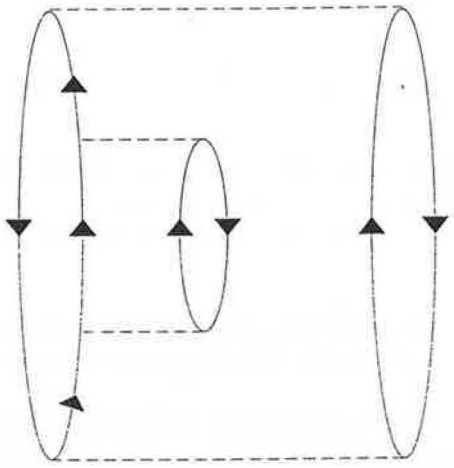

Figure 1. A typical fourth order energy diagram involving triply excited intermediate states

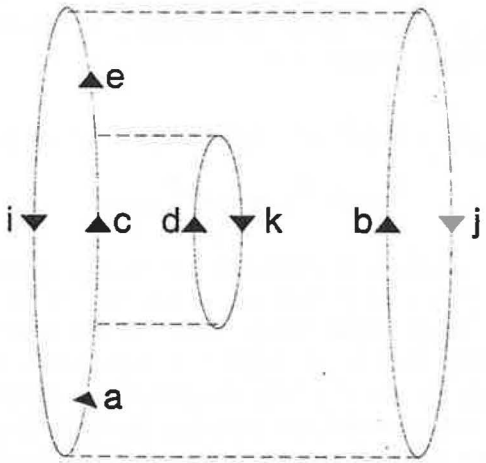

Figure 2. A fully labeled fourth order energy diagram involving triply excited intermediate states

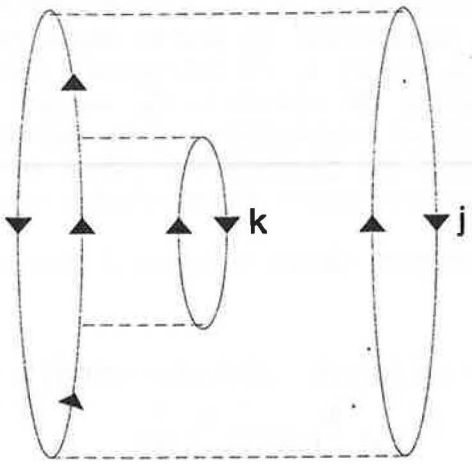

Figure 3. A partially labeled fourth order energy diagram involving triply excited intermediate states

which include electron exchange and the denominators are products of sums of orbital energies

$$D_{ij...ab...} = \varepsilon_i + \varepsilon_j + \ldots - \varepsilon_a - \varepsilon_b - \ldots$$

Diagram F_t gives rise to summations over 3 occupied spin orbital indices and 5 virtual spin orbital indices and the direct evaluation of the energy component corresponding to this diagram would lead to an algorithm scaling as $N_{occ}^3 N_{virt}^5 \sim n^8$.

In previous work [45], we have modified the rules presented in [44] in order to devise *cc*MBPT algorithms. We use the diagrammatic convention that

> *any particle or hole line which is labeled is not summed over and is not regarded as equivalent to any unlabeled line or any labeled line with a different label.*

Thus, for example, the diagram shown in Figure 2 in which all lines are labelled corresponds to the algebraic expression.

$$\frac{\langle ij|\hat{O}|ab\rangle \langle ak|\hat{O}|cd\rangle \langle cd|\hat{O}|ek\rangle \langle eb|\hat{O}|ij\rangle}{D_{ijab} D_{ijkbcd} D_{ijbe}}$$

The diagrammatic formulation of perturbation theory affords a simple pictorial representation of the physical processes associated with a given term (for a recent discussion see, for example, [46]). Thus, in the time-dependent picture of Goldstone, the diagram shown in Figure 2 represents four events at times $\tau_1<\tau_2<\tau_3<\tau_4$ which may be most easily described in a particle-hole formalism. At time τ_1, a double excitation occurs creating the hole states $|i\rangle$ and $|j\rangle$ below the Fermi level and the particle states $|a\rangle$ and $|b\rangle$ above the Fermi level. At some later time, τ_2, the electron in particle state $|a\rangle$ is transferred to particle state $|c\rangle$ and a third hole state $|k\rangle$ is created by excitation into particle state $|d\rangle$. The third interaction line, at time τ_3, describes the transfer of the electron in particle state $|c\rangle$ to particle state $|e\rangle$ and the destruction of the hole-particle pair ($|k\rangle$, $|d\rangle$). Finally, at time τ_4, the hole-particle pairs ($|i\rangle$, $|e\rangle$) and ($|j\rangle$, $|b\rangle$) are destroy returning the system to the vacuum state. In addition to providing a simple pictorial representation of the physical processes resulting from electron correlation, the diagrams also afford an elegant and compact representation of the various terms in the perturbation series.

The partially labeled diagram shown in Figure 3 corresponds to the algebraic expression

$$\sum_{iabcde} \frac{\langle ij|\hat{O}|ab\rangle \langle ak|\hat{O}|cd\rangle \langle cd|\hat{O}|ek\rangle \langle eb|\hat{O}|ij\rangle}{D_{ijab} D_{ijkbcd} D_{ijbe}}$$

It is then clear that diagram F_t can be written in the form shown in Figure 4.

An algorithm for the evaluation of all fourth order triple excitation diagrams, which scales as the seventh power of the number of basis functions employed in the calculation, may be obtained by defining the intermediates [3]

$$f^{i;a}_{jk;bc} = \sum_{d} \langle ij|\hat{O}|db\rangle \langle dk|\hat{O}|ac\rangle / D_{ijbd}$$

which involves a summation over a virtual spin orbital and, using the rule given above may be represented diagrammatically as shown in Figure 5a, and

$$g^{i;a}_{jk;bc} = \sum_{l} \langle jl|\hat{O}|ba\rangle \langle ik|\hat{O}|lc\rangle / D_{jlab}$$

which involves a summation over an occupied spin orbital and is represented in Figure 5b.

The energy component described by diagram F_t may then be written

$$E_4(F_t) = \frac{1}{4} \sum_{\substack{i\,j\,k \\ a\,b\,c}} f^{i;a}_{jk;bc} f^{i;a}_{jk;bc} / D_{ijkabc}$$

The evaluation of a single f intermediate involves N_{virt} operations and the determination of the energy component from these intermediate requires N_{occ}^3 N_{virt}^3 operations. Hence, as we have already noted, our algorithm scales as $\sim n^7$ (N_{occ}^3 N_{virt}^4).

All of the summations in the above expressions are over spin orbitals. By explicitly carrying out the integration over the electron spin coordinates "spin-free" expressions suitable for computational implementation can be obtained. Details have been given previously [3]. Here we emphasize that the kernel of the algorithm, the formation of the f and g intermediates, consists of matrix multiplications which can be executed very efficiently on vector processors. By performing the summations in the order implied in Figures 2, 3 and 4 leads to modest storage requirements for the f and g intermediates. We have, therefore, restricted our attention in the present work to cases in which the lists of two-electron integrals can be held in the shared memory, although we note that the order of the summations is closely related to that used in the early work of Wilson and Saunders [3] in which the processing of the longest list of integrals, those involving one hole and three particle indices, was minimized.

3. The NEC SX-3/44 computer

The NEC SX-3/44 computer [47,48] is a vector multi-processor with a theoretical peak performance of 22 GFLOPS. (The very recently announced [49] NEC SX-3R/44 has a theoretical peak performance of 25.69 GFLOPS.) Following Hockney [50,51], we can characterize the performance of vector processors in terms of two simple parameters. If t is the time required to calculate n results then we may put

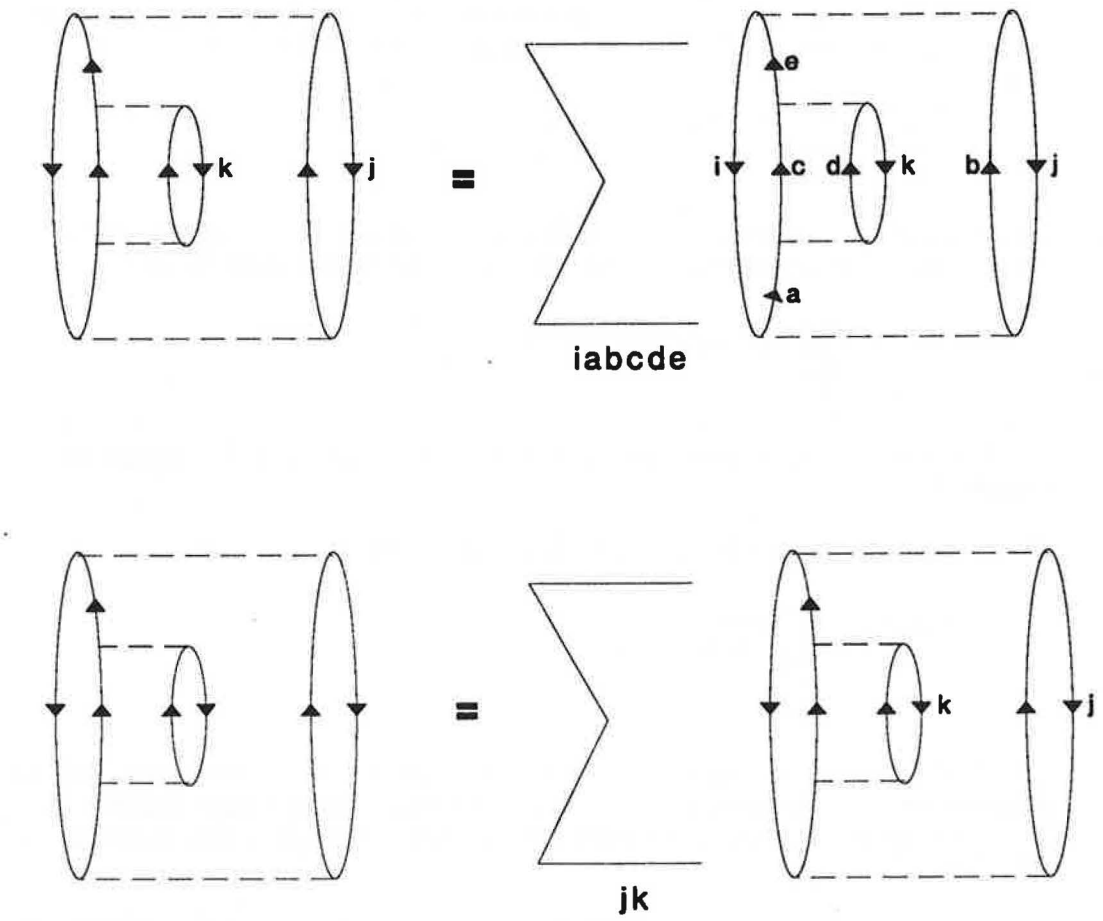

Figure 4. Diagrammatic relation for the fourth order energy diagram shown in Figure 1.

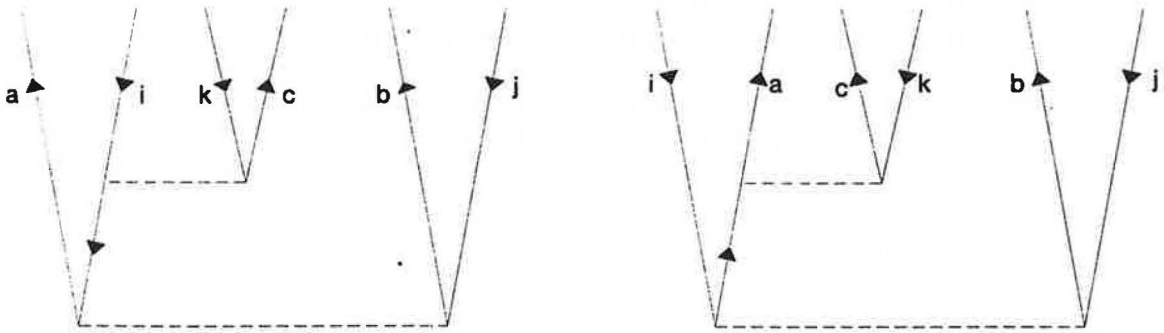

Figure 5. Diagrammatic representation of intermediates

$$t = r_{\infty} (n + n_{\frac{1}{2}})$$

where r_{∞} is the asymptotic performance and $n_{\frac{1}{2}}$ is the vector length required to achieve half the maximum performance. In a recent paper [52] on the performance of the LINPACK benchmark on the NEC SX-3/44, Dongarra reports a rate of execution of 13.420 GFLOPS in the solution of a system of equations of dimension 1000. He reports r_{∞} = 22 GFLOPS and $n_{\frac{1}{2}}$=832 in solving a system of equations of dimension n. For n=6144 he recorded a rate of execution of 20 GFLOPS.

Theoretical peak performance is given by (number of floating point operations/cycle) × (number of cycles/second) × (number of processors). On each processor of the NEC SX-3/44 computer there are four types of parallelism in the functional units. Firstly, there is an overlap of computation in each functional unit, that is the familiar vector operation, giving one result per cycle. Secondly, there are four sets of pipelines for each functional unit. Each pipeline processes $\frac{1}{4}$ of the vector. Each pipeline set handles every fourth vector operation giving what has been termed [48] "hardware microtasking". These multiple pipes allow four results to be obtained in each cycle. Although this increases the asymptotic rate of execution, r_{∞}, by a factor of 4, it also increases $n_{\frac{1}{2}}$ by the same factor and longer vectors are required to attain a given performance improvement. Thirdly, the NEC SX-3/44 computer employs chaining of addition and multiplication to obtain eight results per cycle. Fourthly, by overlapping two additions and two multiplications enabling the execution of two independent adds or multiplies we arrive at a maximum number of floating point operations per clock period of 16 on a single processor. The clock speed is 2.9 nsec and thus the peak performance of a single processor is 5517 MFLOPS. The maximum number of processors is 4 giving a total theoretical peak performance of 22.1 GFLOPS. The vector units each have 144Kb of vector register to store intermediate results of vector operations. Each processor has 64 Kbytes of cache memory and, in a fully configured system, there is a main memory of 2048 Mbytes. The 2.048 Gbytes (256 Mwords) of shared memory is organized into a maximum of 1024 independent banks, i.e. a 1024 way interleaved system constructed from 256 Kbyte, 20 nsec static RAM chips. There are four I/O processors with an aggregate transfer speed of 1 Gb s^{-1} and each I/O processor has up to 64 channels.

The FORTRAN compiler for the NEC SX-3 has both autovectorizing and parallel processing functions.

Dynamic load balancing on the NEC SX-3/44 computer. In the present work, we adopt the following notation: N denotes the number of parallel processes and T the number of tasks. I identifies a task and ranges from 1 to T. P identifies a process and, therefore, ranges from 1 to N.

Our algorithm for the evaluation of the contribution of the triply substituted configurations to the fourth order energy gives rise to a number of tasks of roughly equal size. Each task corresponds to the evaluation of the energy associated with the partially labeled diagram shown in Figure 3. Each task may

proceed independently of the others. Orbital symmetries and the spatial disposition of the atoms upon which the basis functions are centred will influence the size of the tasks.

On shared memory architecture parallel computers, static load balancing of the work assigned to each process is usually far from optimal. On a dedicated machine the tasks may not be of equal size and some idle time will inevitably accumulate. In a multiuser environment, each processor is not normally dedicated to a single user; instead its computing power is shared out amongst a number of users. The effective computing power delivered to a given user (or a parallel process of that user) by a given processor is largely unpredictable, because it depends upon what other users are doing at any given time. Thus instead of requiring that all processors carry out an identical workload, it is better to require that each processor be given a workload commensurate with its effective power; the latter is far from constant with time, so that it is necessary to feed work to the processors at a rate which is similarly time dependent.

This may be accomplished by the technique of "dynamic load balancing" , using "global indices" under control of a "lock" [53]. Standard FORTRAN code for the evaluation of the energy component corresponding to diagram F_t might take the following form:

```
      DO 1 I = 1,T
      Execute task I
    1 CONTINUE
```

where task *I* involves the evaluation of the partially labelled diagram shown in Figure 3. Using NEC SX-3/44 multi-tasking software the dynamically balanced parallel version using *N* processes would be of the form shown in Figure 6. During the parallel phase of the job *N* copies of subroutine TRIPLE1 are in execution. On the NEC system TRIPLE1 is made re-entrant, circumventing the need to actually make the copies. TRIPLE1 then has the form shown in Figure 7. PLLOCK is called to acquire unique access to the global index *I* which is being protected by LOCKI; if another parallel copy of TRIPLE1 calls PLLOCK whilst such unique access is granted, the second copy will be halted until the first copy releases access with a call to PLUNLOCK. If I is greater than T then all tasks have been completed, and after releasing LOCKI, TRIPLE1 returns control to the calling program. A local copy of the global index is taken in ITEMP, and the global index updated, after which LOCKI is released. The task indicated by ITEMP (note that the global index itself is not used in this phase since it may be updated by another parallel copy of TRIPLE1) is then carried out, after which the routine branches to the first executable statement to see if work remains. Obviously, if task execution is controlled by a nest of loops (in the present work there is a four-fold nest of loops over the tasks), then all loop indices associated with the nest should be treated as a shared locked data structure [46] and accessed under control of a lock. The overheads associated with the above implementation are T O_r, where O_r is the time to acquire/release LOCKI. This is of the order of a few microseconds on the NEC SX-3/44 system. The individual tasks are relatively large and, therefore, contention for access to the lock is negligible. Overheads can be largely ignored. Each process generates only as much work as its corresponding processor is able to deal with (it is dynamically balanced). Given a reasonably large number of tasks, dynamic balance produces an approximately even loading over dedicated processors even if the tasks are not of equal size. The major disadvantage of dynamic balance is

```
      PARAMETER (NCPUS=4)
      COMMON /TASKID/ ITASK(2,NCPUMAX)
      COMMON /GLOBAL/ I, LOCKI, T
      EXTERNAL TRIPLE1
C
C     Initialize multitasking variables
C
      T = ? : number of tasks
      NCPUS = ? : number of processes
      CALL PLASGN(LOCKI)
      I = 1
      DO 1 K = 1,NCPUS
      ITASK(1,K) = 0
      ITASK(2,K) = K
    1 CONTINUE
C
C     Create N-1 out-of-line processes and one in-line processs - fork
C
      DO 2 K = 2,NCPUS
      CALL PTFORK(ITASK(1,K),ITASK(2,K),TRIPLE1,ITASK(2,K))
    2 CONTINUE
      CALL TRIPLE1
C
C     Wait until all processes have finished - barrier
C
      DO 3 K = 2,NCPUS
      CALL PTJOIN(ITASK(1,K))
    3 CONTINUE
```

Figure 6. Calling sequence for dynamically balanced parallel routine.

```
      SUBROUTINE TRIPLE1
      COMMON /GLOBAL/ I, LOCKI, T
    1 CALL PLLOCK(LOCKI)
      IF(I.GT.T)THEN
      CALL PLUNLOCK(LOCKI)
      RETURN
      ELSE
      ITEMP = I
      I = ITEMP + 1
      CALL PLUNLOCK(LOCKI)
      ENDIF
      ...
      Execute task ITEMP
      ...
      GOTO 1
      END
```

Figure 7. Dynamically balanced parallel routine.

that it requires the use of locks, hardware support for which is unavailable (and software emulations expensive - probably requiring that all processing of a lock be carried out by one processor which communicates its results to the other processors by message passing) on many current independent memory machines. Its use is at present restricted mainly to shared memory parallel architectures, which all provide hardware supported locks. We here observe that current shared memory systems, such as are available from CRAY, NEC and IBM, capable of running each processor in multi-user mode (and thus require dynamic balance and the associated locks for optimal performance).

The final evaluation of the fourth order energy corresponding to diagram X_t can be written as:

$$\Delta E_{total}[X_t] = \sum_{I}^{T} \Delta E[X_t](I)$$

where $\Delta E[X_t](I)$ is the contribution to ΔE_{total} $[X_t]$ from the *I th* task. ΔE_{total} $[X_t]$ is thus a shared locked data structure and must either be protected by a lock when it is updated, or else produced by the method of "multiple images" [53], and we have chosen the latter course. In this method partial contributions to $\Delta E_{total}[X_t]$ are accumulated in an array $\Delta E[X_t](P)$, P=1,N, by each process, the element of $\Delta E[X_t]$ to be used by a given process being indicated by the process identification index, P, which is passed via the argument list to the parallel routine in our implementation. $\Delta E[F_t](P)$ is a shared independent data structure; all other data structures manipulated by the central kernel of our algorithm are either shared independent (in fact read only) or local, and we therefore have a paradigm case of a sequence of completely independent tasks. When all parallel processing is complete, the fourth order energy is evaluated through:

$$\Delta E_{total}[X_t] = \sum_{P}^{N} \Delta E[X_t](P)$$

We have written a computer program for the CRAY Y-MP machine which evaluates the energy components corresponding to each of the fourth order, triple excitation energy terms. The source code, which is published elsewhere [6], was modified for implementation on the NEC SX-3/44 computer.

4. Benchmark computations, an application and comments.

The NEC SX-3/44 benchmark tests of the *cc*MBPT code were performed at the Dorval Computer Center (DCC) running under the Super-UX operating system (version 1.22F). The DCC installation has 1.024 Gbytes (128Mw) of main memory, half that of a fully configured system.

The f77sx Fortran compiler (version 1.22F) was used, and the NEC-supplied routine vdrMXMA was used to carry out the matrix multiply operations. The number of doubly occupied orbitals, N_{occ}, has consistently been set to 8. The number of unoccupied orbitals, N_{virt}, has been varied.

Matrix multiplication kernels. The determination of the intermediates $f^{i;a}_{jk;bc}$ is the most demanding stage in the computation. The intermediates $f^{i;a}_{jk;bc}$, i=1, ..., N_{occ}, a=1, ..., N_{virt}, can be constructed by means of a matrix multiplication involving matrices of dimension $N_{virt} \times N_{virt}$ and $N_{virt} \times N_{occ}$. We measured the rate of computation, ρ_{user}, that could be achieved on a single processor for the matrix multiplication kernel of our algorithm for a range of values of N_{occ} and N_{virt}. $\rho_{user} = N_{flops}/\tau_{user}$, where N_{flops} is the number of floating point operations and τ_{user} is the user central processing time. The results are presented in Table 1.

Benchmark computation on a single processor. The performance of the *cc*MBPT-4_t code on a single processor of a non-dedicated, but quiet, NEC SX-3/44 computer was measured as a function of the number of virtual orbitals, N_{virt}, using eight occupied orbitals. Ideally these measurements should have been made on a dedicated machine but operational restrictions at DCC made this impossible and the measurements were made on a "quiet" machine, i.e. we believe that our results were not influenced to any significant extent by the activities of other users. The results are presented in Table 2. τ_{real} is the real (elapsed or wall clock) time in seconds and τ_{user} is the user (central processing unit) time. τ_{system} is the systems time and can be seen to change little with increasing N_{virt}. τ_{cache} is the time attributable to "cache misses". τ_{cache} varies between 12.4% (N_{virt}=80) and 14.3% (N_{virt}=96) of τ_{user}. τ_{bank} is the time arises from memory bank conflicts and increases from 1.9% of τ_{user} at N_{virt}=64 to 4.1% for N_{virt}=192. The rate of computation, ρ_{user}, is given in units of 10^9 floating point operations per central processing unit second (GFLOPS). τ_{vector} is the time spent performing vector operations, ν_{len} is the average vector length and R is the ratio vector to scalar operations expressed as a percentage.

Benchmark computation on two processors. The performance of the *cc*MBPT-4_t code on two processors of a non-dedicated, but quiet, NEC SX-3/44 computer was measured as a function of the number of virtual orbitals, N_{virt}, using eight occupied orbitals. Again τ_{real} is the real time, τ_{user} is the user time, τ_{system} is the systems time, τ_{cache} is the time attributable to "cache misses" and τ_{bank} is the time arises from memory bank conflicts. The rate of computation, ρ_{user}, is given in units of 10^9 floating point operations per central processing unit second (GFLOPS). N_{cpu} is the number of processors (2). $\rho_{real} = N_{flop}/\tau_{real}$ is also given in GFLOPS. N_{flop} is the number of floating point operations recorded in the corresponding calculation using a single processor and is given in Table 5.

Benchmark computation on four processors. In Table 4, the rate of computation ρ_{real} is given as a function of the number of processing units used, N_{cpu}, with fixed N_{virt} (=160) on a non-dedicated, but quiet, NEC SX-3/44. τ_{real} is also given.

The results of a series of benchmark calculations for different values of N_{virt} on a four processor NEC SX-3/44 are displayed in Table 5. For N_{virt}=64 the rate of execution ρ_{real} is 17.7% of the theoretical peak performance. ρ_{real} increases to 38.5% of the theoretical peak performance for N_{virt}=160 dropping to 38.4% for N_{virt}=192. The ratio R:N_{cpu}, where R=τ_{real}:τ_{user}, takes a value of 80.0% for the N_{virt}=64 case and rises to 90.8% for N_{virt}=160.

Table 1.

Rate of computation, ρ_{user} (in GFLOPS), achieved in the matrix multiplication kernel which arises in the formation of the intermediate f on a single processor of a non-dedicated NEC SX-3/44 computer.

	N_{occ}				
N_{virt}	8	16	32	48	64
128	3.20	3.33	3.28	2.84	2.77
144	3.57	3.71	3.61	3.05	3.00
160	3.94	4.12	3.98	3.43	3.37
176	4.24	4.45	4.36	3.71	3.64
192	4.50	4.69	4.69	4.21	4.13
208	4.71	4.91	4.92	4.44	4.35
224	4.66	4.83	4.87	4.69	4.85
240	4.83	5.00	5.07	4.92	4.85
256	4.77	4.92	5.00	4.99	4.99

Table 2.

Performance of a single processor on a non-dedicated, but quiet, NEC SX-3/44 computer as a function of the number of virtual orbitals, N_{virt}, using eight occupied orbitals. τ_{real} is the real (elapsed) time, τ_{user} is the user (cpu) time, τ_{system} is the systems time, τ_{cache} is the time attributable to "cache misses" and τ_{bank} is the time arises from memory bank conflicts. The rate of computation, ρ_{user}, is given in units of 10^9 floating point operations per central processing unit second (GFLOPS). τ_{vector} is the time spent performing vector operations, ν_{len} is the average vector length and R is the ratio vector to scalar operations expressed as a percentage. All times are given in seconds

N_{virt}	τ_{real}	τ_{user}	τ_{system}	τ_{cache}	τ_{bank}
64	83.5	82.4	1.0	10.6	1.6
80	158.5	157.7	0.8	19.5	3.2
96	289.9	280.9	2.1	40.1	6.1
128	611.6	610.7	0.8	76.7	18.4
160	1205.3	1204.4	0.9	157.9	42.9
192	2106.1	2105.1	0.9	265.6	85.4

N_{virt}	ρ_{user}	τ_{vector}	ν_{len}	R
64	1.419	70.1	79.6	99.1
80	1.687	137.1	91.9	99.3
96	1.869	241.9	107.5	99.4
128	2.543	550.7	135.0	99.5
160	3.020	1105.9	162.3	99.6
192	3.481	1981.7	189.5	99.7

Table 3.

Performance of two processors on a non-dedicated, but quiet, NEC SX-3/44 computer as a function of the number of virtual orbitals, N_{virt}, using eight occupied orbitals. τ_{real} is the real (elapsed) time, τ_{user} is the user (cpu) time, τ_{system} is the systems time, τ_{cache} is the time attributable to "cache misses" and τ_{bank} is the time arises from memory bank conflicts. The rate of computation, ρ_{user}, is given in units of 10^9 floating point operations per central processing unit second (GFLOPS). τ_{vector} is the time spent performing vector operations, v_{len} is the average vector length. N_{cpu} is the number of processors (2). $\rho_{real} = N_{flop}/\tau_{real}$ is also given in GFLOPS. N_{flop} is the number of floating point operations recorded in the corresponding calculation using a single processor and is given in Table 5. All times are given in seconds

N_{virt}	τ_{real}	τ_{user}	τ_{system}	τ_{cache}	τ_{bank}
64	46.5	90.7	0.8	14.0	1.7
80	84.3	166.6	0.8	23.2	3.2
96	160.3	284.7	2.0	42.1	6.2
128	335.1	662.3	0.8	96.4	18.6
160	651.1	1299.1	0.9	198.0	42.4
192	1169.3	2316.8	1.0	345.9	86.0

N_{virt}	ρ_{user}	τ_{vector}	v_{len}
64	1.290	74.1	79.6
80	1.597	141.8	91.9
96	1.844	243.3	107.5
128	2.345	576.8	135.0
160	2.800	1155.5	162.3
192	3.163	2089.0	189.5

N_{virt}	$N_{cpu}\ \rho_{user}$	ρ_{real}	$\tau_{user}:\tau_{real}$
64	2.580	2.52	1.95
80	3.194	3.16	1.98
96	3.688	3.28	1.78
128	4.690	4.64	1.98
160	5.600	5.59	2.00
192	6.326	6.27	1.98

Table 4.

Rate of computation in 10^9 floating point operations per real second (GFLOPS), ρ_{real}, real time, τ_{real}, and user time, τ_{user}, as a function of the number of processors, N_{cpu}, using eight occupied orbitals and 160 virtual orbitals on a non-dedicated, but quiet, NEC SX-3/44 computer.

N_{cpu}	ρ_{real}	τ_{real}	τ_{user}
1	3.02	1205.3	2105.1
2	5.59	651.1	2316.8
4	8.48	428.9	2562.4

Table 5.

Performance of four processors on a non-dedicated, but quiet, NEC SX-3/44 computer as a function of the number of virtual orbitals, N_{virt}, using eight occupied orbitals. τ_{real} is the real (elapsed) time, τ_{user} is the user (cpu) time, and τ_{system} is the systems time. N_{flop} is the number of floating point operations recorded in the corresponding calculation using a single processor. The rate of computation, $\rho_{real} = N_{flop}/\tau_{real}$, is given in units of 10^9 floating point operations per real (elapsed) unit second (GFLOPS). All times are given in seconds

N_{virt}	τ_{real}	τ_{user}	$\tau_{real}:\tau_{user}$	$N_{flop}/10^9$	ρ_{real}
64	30.1	96.2	3.2	116	3.89
80	56.2	192.6	3.4	266	4.74
96	92.2	324.5	3.5	525	5.69
128	226.6	772.1	3.4	1553	6.86
160	428.9	1156.6	3.6	3638	8.48
192	867.3	2562.4	3.0	7329	8.45

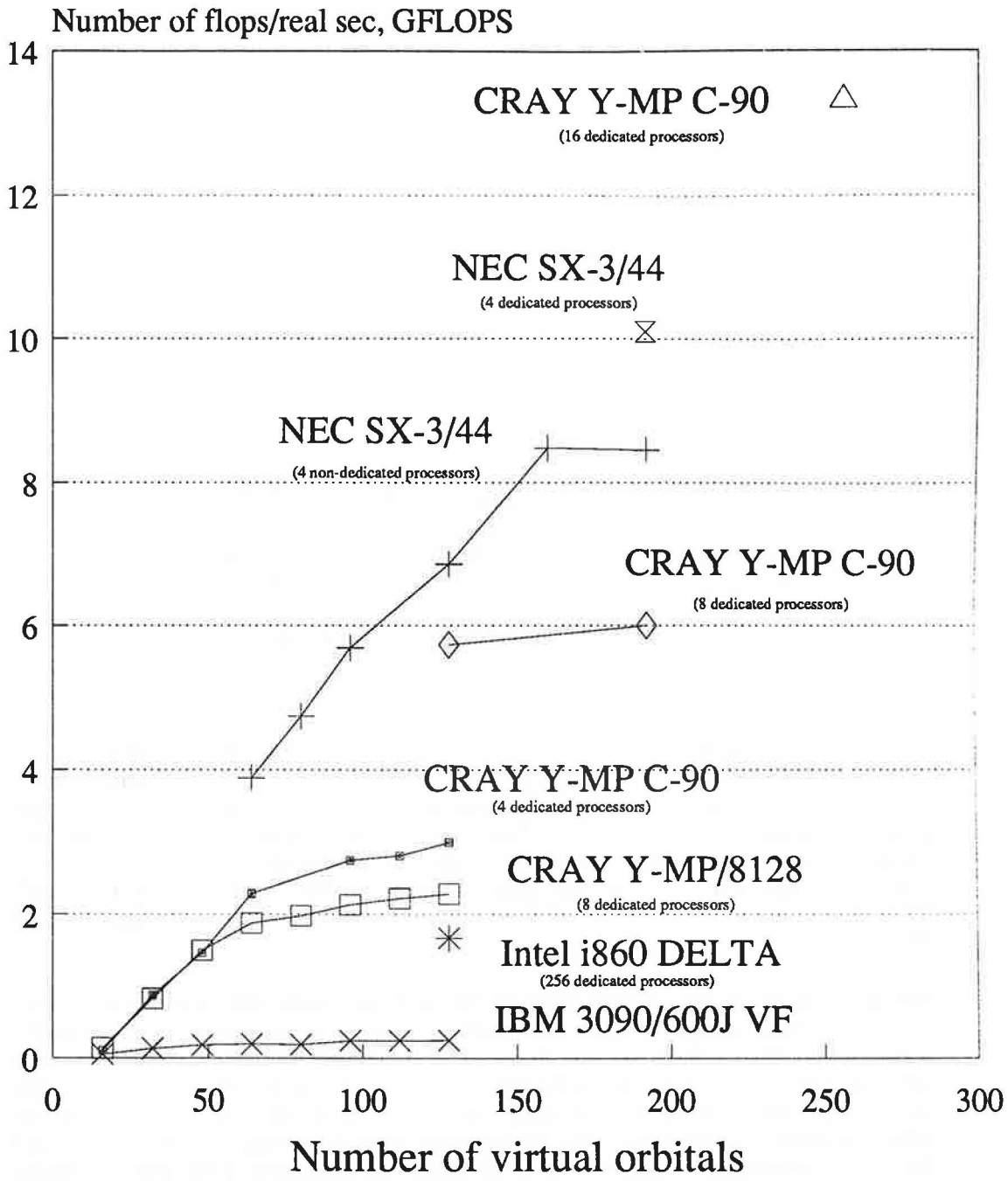

Figure 8. Rate of computation achieved in *cc*MBPT calculations as a function of the number of virtual orbitals on various high performance computers. For all computations the number of occupied orbitals was fixed at four except for those carried out on the NEC SX-3/44 computer where eight occupied orbitals were used. The IBM 3090/600J VF results are taken from [2,8], those for the Intel i860 DELTA machine are from [9], for the CRAY Y-MP/8128 the results are from [2,45], for the CRAY Y-MP C-90 (4 dedicated processors) detailed measurements are given in [7] whilst those for 8 and 16 dedicated processors have not been published previously.

In Figure 8, the rate of execution ρ_{real} observed on the non-dedicated but quiet four processor NEC SX-3 machine is compared with that obtained on four and eight dedicated processors of a CRAY Y-MP C-90 computer.

Dedicated benchmark computation on four processors. One benchmark calculation on a dedicated NEC SX-3/44 was carried out. For this calculation we put N_{occ}=8 and N_{virt}=192. This choice was determined by the memory available on the DCC installation (128 Mwords). A rate of execution of ~10.1 GFLOPS was obtained, which represents 46% of the theoretical peak performance. This result is shown in Figure 8, where it is compared with the results of a benchmark computation carried out on a dedicated sixteen processor CRAY Y-MP C-90.

Comments on a multi-user environment. The benefits of multi-processing in a multiuser environment have until quite recently largely been seen in terms of the "$1/N_{cpu}$ rule" [54,55], where N_{cpu} denotes the number of processors in the system. In this model, the use of more than $1/N_{cpu}$ *th* of a scarce resource (for example, memory) requires that one uses more one processor, otherwise it is considered that other jobs will be unlikely to saturate the remaining processors because of shortage of the resource. We have presented previously an argument in favour of multi-processing even small memory jobs. We estimate that it is desirable to have about two executing processes per processor to ensure that no idle time accumulates, largely because of wait times for input/output. This is very difficult to ensure given the type of workload (many large memory heavily input/output bound jobs) often encountered. The presence of a single processor-bound small memory four-way multiprocessed job (in dynamic balance) running at low scheduling priority is capable of consuming all potentially idle cycles on all the processors of an NEC SX-3/44 system, with little disturbance to the flow of other work, and a greatly improved overall throughput.

An application to the ground state of the nitrogen molecule. The fourth order triple excitation component of the correlation energy is known to be particularly sensitive to the quality of the basis set employed [4]. Contemporary high performance computers are permitting the use of larger basis sets than has previously been possible resulting in more accurate electronic structure determinations. Large basis sets of gaussian-type functions can be constructed by using systematic sequences of even-tempered functions [56] and exploring the convergence of the calculation. Recent studies of small molecules [57] have demonstrated that sub μ-Hartree accuracy can be achieved by this procedure.

It is also known that the fourth order triple excitation component of the correlation energy is particularly large in multiply bonded systems such as N_2. Using an even-tempered basis set consisting of 102 Gaussian-type functions (16s/8p/5d/4f/1g) on each atomic centre, giving a total of 204 basis functions (Nocc = 7, Nvirt = 197), we have used the NEC SX-3/44 implementation of the *cc*MBPT-4_t package to calculate E_{4t} and all its diagrammatic components for a range of internuclear separations. This forms part of an on-going project to calculate high precision correlation components for the ground state of the nitrogen molecule and other small diatomic species.

In Figure 9 an analysis of the linked diagram triple-excitation component of the electron correlation energy for the ground state of the nitrogen molecule is

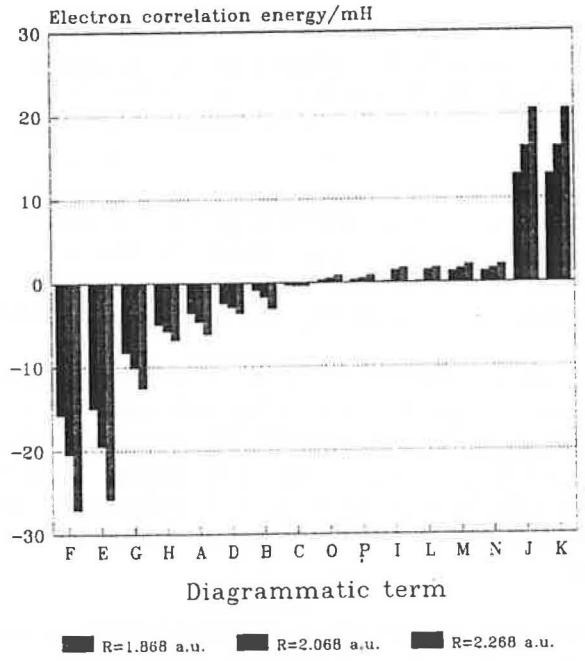

Figure 9. Analysis of the linked diagram triple-excitation component of the electron correlation energy for the ground state of the nitrogen molecule for three internuclear separations.

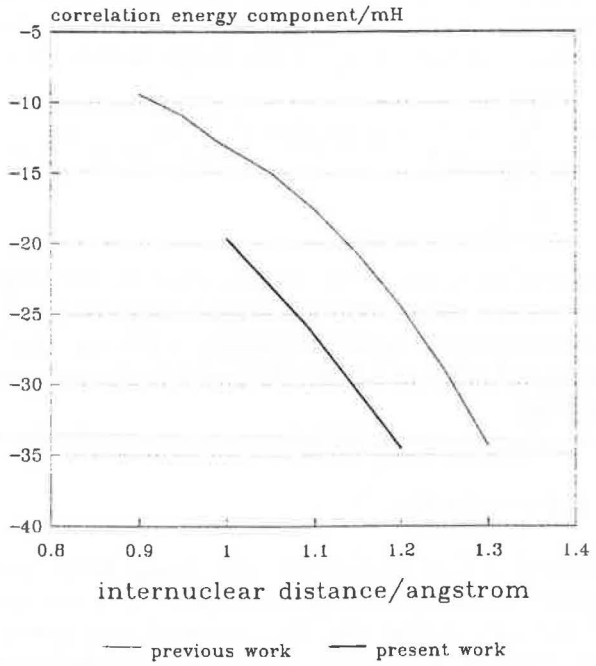

Figure 10. A comparision of the present calculations of the linked diagram triple-excitation component of the electron correlation energy for the ground state of the nitrogen molecule with previous work (see text for details).

given for three internuclear separations. The energy diagrams are labelled according to the conventions employed in reference [1]. The energies corresponding to each of the sixteen fourth order diagrams containing triply excited intermediate states are displayed in non-decreasing order. It is apparent that there is a considerable degree of cancellation between the different diagrammatic components. The magnitude of these components increases with increasing nuclear separation; a refection of the fact that the single determinant reference function that has been employed in the present study does not provide a useful description of the dissociation of the nitrogen molecule. No practial approximation scheme is suggested by the analysis presented in Figure 8.

A comparison of the present calculations of the triple-excitation component of the electron correlation energy for the ground state of the nitrogen molecule with previous work is made in Figure 10. The previous calculations were carried out in 1980 on a CRAY 1 computer. The higher precision achieved in the present work is a reflection of the increased performance of contemporary high performance computers allowing the use of larger basis sets to improve the theoretical model.

Comments on the corrections to fourth order correlation energies. Although the dominant corrections to fourth order correlation treatments, that is the fifth order terms, were written down for the first time in 1979 [58], they are, even today, not routinely calculated.

Computationally, the most demanding of the fifth order energy terms are those which involve two intermediate states which are triply excited with respect to the reference function. Such diagrammatic terms lead to algorithms which scale as the eighth power of the number of basis functions employed. A typical diagram of this type is shown in Figure 11. It can easily shown that the algebraic expression corresponding to this diagram may be cast in the form

$$E_5(X_{tt}) = \frac{1}{8} \sum_{\substack{i\,j\,k \\ a\,b\,c\,d\,e}} f^{i;a}_{jk;bc} \langle bc|\hat{O}|de\rangle f^{i;a}_{jk;de} / D_{ijkabc} D_{ijkade}$$

where the f intermediates are those which arise in the evaluation of the fourth order, triple excitation terms. We comment here that the performance levels achieved on contemporary supercomputers, such as the NEC SX-3/44, will enable the routine calculation of the fifth order energy components using basis sets of adequate size.

5. Discussions and conclusions

We have presented a formulation of the many-body perturbation theory of electron correlation suitable for implementation in a computing environment in which parts of the calculation can be executed concurrently. We have shown that the linked diagram expansion leads to a formulation of the correlation problem which maps very effectively on to multiprocessor machines. We have considered in detail the evaluation of the most computationally demanding of the fourth order component of the correlation energy, the contribution of associated with the full set of sixteen diagrammatic components associated with triply substituted intermediate states.

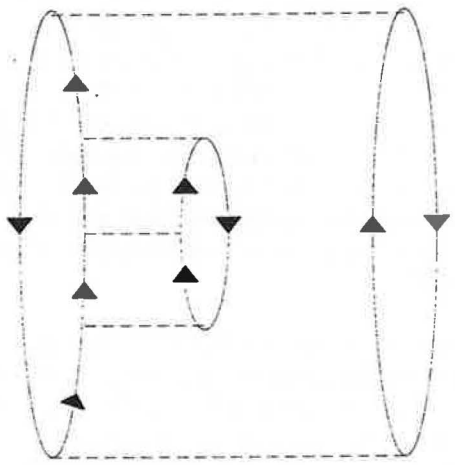

Figure 11. A typical fifth order energy diagram involving two triply excited intermediate states. The determination of the energy components corresponding to diagrams of this type represents the most demanding part of a fifth order calculation giving rise to an algorithm which scales as the eighth power of the number of basis functions employed.

We have shown that in each case the diagrammatic components can be decomposed into a set of partial contributions which can be computed by an algorithm structured around the matrix multiply operation. Each of these matrix multiplications is performed on a separate processor and so not only can conventional vector processors can be driven nearly optimally but also the performance of the algorithm does not degrade severely as the number of processors is increased. Thus on a NEC SX-3 system running in one processor mode a rate of computation of ~3.5 Gflops is typical. Since the evaluation of the partial contributions may proceed entirely independently the theory is well suited for implementation on a parallel processing system with either a distributed or shared memory. For typical shared memory systems, which normally operate in multi-user mode, dynamic balance allows one to partition the work amongst the processors according to their ability to deal with it. Indeed, recent work [59] has demonstrated that the dynamic load balancing strategy leads to an effective "scavenger" of potential idle cycles on a heavily loaded CRAY X-MP system.

In contrast to our previous work on a CRAY Y-MP computer where a linear speed-up with the number of processors was observed, on the NEC SX-3/44 machine a sub-linear is found. On a CRAY Y-MP, in a dedicated environment, close to linear speed-up with number of processors has been previously observed; the slight degradation observed was attributed to memory bank conflicts or other overheads of the parallel implementation. In the present work, it is clear from Tables 2 and 3 that bank conflicts are not the source of the sub-linear speed-up of the NEC SX-3/44; the time attributed to bank conflicts, τ_{bank}, in the single and dual processor runs are very close. However, the time associated with cache misses, τ_{cache}, and the vector time, τ_{vector}, increase noticeably when multiprocessing. We attribute this observation to the fact that each pair of processors share two vector load paths, one vector store path, one scalar load path and one scalar store path leading to the possibility of contention.

Because the DCC installation has 1.024 Gbytes of main memory, maximum value of N_{virt} that could handled in the present experiments was 192. On a fully configured systems with 2.048 Gbytes of main memory the N_{virt}=256 case could be handled with the present code. Using the measured rates of execution for the matrix multiplication kernel given in Table 1, we can obtain a estimate of the rate of execution, ρ_{real}, for N_{virt}=256 of 10.1 × (5.00/4.50) = 11.2 GFLOPS, which is 51% of the theoretical peak performance. This should be compared with the 13.3 GFLOPS achieved on the CRAY Y-MP C90, which represents 83% of the theoretical peak performance.

For typical shared memory systems, which normally operate in multi-user mode, dynamic balance allows one to partition the work amongst the processors according to their ability to deal with it. Indeed, recent work [59] has demonstrated that the dynamic load balancing strategy leads to an effective "scavenger" of potential idle cycles on a heavily loaded CRAY X-MP system. The program described in the present work has also been used [60] to demonstrate that there is no degradation in the performance of a multiuser CRAY Y-MP system as the proportion of multitasked jobs is increased.

The dynamic load balancing technique described in this work for the NEC SX-3/44 will prove essential to the efficient implementation of the recently

developed relativistic many-body perturbation theory [61-63]. It should also prove useful in the recent developed [64-67] generalizations of the many-body perturbation theory for calculating potential energy curves and surfaces, and studying dissociative processes.

Acknowledgement. We are grateful to Canada's Atmospheric Environment Service for allowing access to the NEC SX-3/44 computer at the Dorval Computer Centre as part of the Canadian Supercomputer Challenge, 1992, and to the staff of Dorval Computer Centre for their help and advice. One of the authors (DM) acknowledges the support of the US Department of Energy through contract no. DE-FC05-85ER250000. SW is grateful to the Supercomputer Computations Research Institute of Florida State University for funding a visit during which the initial phases of this project were carried out.

References

[1] S. Wilson, Electron correlation in molecules, Clarendon Press, Oxford, 1984

[2] S. Wilson and D. Moncrieff, Supercomputer **45**, 28, 1991

[3] S. Wilson and V.R. Saunders, Comput. Phys. Commun. **19**, 293, 1980

[4] S. Wilson and M.F. Guest, Chem. Phys. Lett. **73**, 607 1980

[5] D.J. Baker, D. Moncrieff, V.R. Saunders and S. Wilson, Comput. Phys. Commun. **62** 25, 1991

[6] D. Moncrieff, V.R. Saunders and S. Wilson, Computer Physics Communications *(in press)*

[7] D. Moncrieff, V.R. Saunders and S. Wilson, Supercomputer *(submitted for publication)*

[8] D. Moncrieff, V.R. Saunders and S. Wilson, Rutherford Appleton Laboratory Report, RAL 91064

[9] A.P. Rendell, T.J. Lee, A. Komornicki and S. Wilson, Theoret. chim. Acta *(in press)*

[10] R.G. Evans and S. Wilson, editors, *Supercomputational Science* (Plenum Press, New York, 1990)

[11] S. Wilson, editor, *Concurrent computation in chemical calculations*, Methods in Computational Chemistry **3** (Plenum Press, New York, 1989)

[12] D.M. Silver, Comput. Phys. Commun. **14** (1978) 71

[13] D.M. Silver, Comput. Phys. Commun. **14** (1978) 81

[14] S. Wilson, Comput. Phys. Commun. **14** (1978) 91

[15] R.J. Bartlett and D.M. Silver, erratum, J. Chem. Phys. **64** (1976) 4578

[16] D.M. Silver, S. Wilson, and R.J. Bartlett, Phys. Rev. **A16** (1977) 477

[17] S. Wilson, Intern. J. Quantum Chem. **12** (1977) 604; Molec. Phys. **35** (1978) 1

[18] S. Wilson, and D.M. Silver, Phys. Rev. **A14** (1976) 1949

[19] S. Wilson, and D.M. Silver, J. Chem. Phys. **66** (1977) 5400

[20] S. Wilson, D.M. Silver and R.J. Bartlett, Molec. Phys. **33** (1977) 1177

[21] K.A. Brueckner, 1955, Phys. Rev. **100** (1991) 36

[22] J. Goldstone, Proc. Roy. Soc.(London) A **239** (1957) 267

[23] N.M. Hugenholtz, Physica **23** (1957) 481

[24] N.H. March, W.H. Young, and S. Sampanthar, *The many body problem in quantum mechanics*, (Cambridge University Press, 1967).

[25] R. Feynman, Phys. Rev. **76** (1949) 769

[26] F.J. Dyson, Phys. Rev. **75** (1949) 1736

[27] G.C. Wick, Phys. Rev. 80 (1950) 268

[28] B.H. Brandow, Rev. Mod. Phys. **39** (1967) 771

[29] R.J. Bartlett, Ann. Rev. Phys. Chem. **32** (1981) 359

[30] P. Carsky, and M. Urban, *Ab initio Calculations. Methods and Applications*, Lecture Notes in Chemistry **16** (Springer, Berlin, 1980)

[31] K. Jankowski, Meth. Comput. Chem. 1 (1987) 1

[32] I. Lindgren, and J. Morrison, *Atomic Many-Body Theory* (Springer, Berlin, 1982)

[33] M. Urban, I. Cernusak, V. Kellö, and J. Noga, Meth. Comput. Chem **1** (1987) 117

[34] S. Wilson, Specialist Periodical Reports : Theoretical Chemistry **4** (1981) 1

[35] S. Wilson, in: *Methods in Computational Molecular Physics*, eds. G.H.F. Diercksen and S. Wilson,(Reidel, Dordrecht, 1983)

[36] S. Wilson, Comput. Phys. Reports 2 (1985) 389

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

[38] P.J. Knowles, K. Somasundram, N.C. Handy and K. Hirao, Chem. Phys. Lett. **113** (1985) 8

[39] M.F. Guest and S. Wilson, in: *Supercomputers in Chemistry*, ed. P. Lykos and I. Shavitt (American Chemical Society, Washington D.C., 1981) p.1.

[40] D.J. Baker, D. Moncrieff and S. Wilson, in *Supercomputational Science*, edited by R.G. Evans and S. Wilson,(Plenum, New York, 1990)
[41] D. Moncrieff, D.J. Baker and S. Wilson, Comput. Phys. Commun. **55** (1989) 31
[42] S. Wilson, in *Proceedings of Fifth Seminar on Computational Problems in Quantum Chemistry*, Groningen, eds. Van Duijknen, P.Th., and Nieuwpoort, W.C., (Max-Planck-Institut für Physik und Astrophysik, Garching, 1981)
[43] S. Wilson, in *Correlated Wavefunctions*, Proceeding of a Study Weekend, December 1977, edited by V.R. Saunders, (Daresbury Laboratory, 1978).
[44] S. Wilson, and D.M. Silver, Intern. J. Quantum Chem. 15 (1979) 683
[45] D. Moncrieff, V.R. Saunders and S. Wilson, Comput. Phys. Commun. *(in press)*
[46] S. Wilson, J. Molec. Struct. *(in press)*
[47] NEC, Macrotasking and Microtasking, SUPER-UX R1.2, April 16 (1991)
[48] T. Watanabe and A. Iwaya, in *High Performance Computing: Research and Practice in Japan*, edited by R. Mendez, Wiley, London, 1992
[49] Supercomputing Review **5** (1992) 11.
[50] R.W. Hockney and C.R. Jesshope, *Parallel Computers 2 Architecture, Programming and Algorithms*, revised edition, Adam Hilger, Bristol, 1988
[51] R.W. Hockney, in Infotech State of the Art Conference: Future Systems, (Infotech Intl Ltd, Maidenhead, 1977)
[52] J.J. Dongarra, Supercomputing Review, **5** (1992) 54
[53] V.R. Saunders, in *Supercomputational Science*, edited by Evans, R.G. and S. Wilson, (Plenum, New York, 1990)
[54] P.R. Taylor, and C.W. Bauschlicher, Theoret. chim. Acta. **71** (1987) 105
[55] P.R. Taylor, C.W. Bauschlicher and D.W. Schwenke, Meth. Comput. Chem. 4 (1989) 63
[56] S. Wilson, Adv. Chem. Phys. **67** (1987) 439
[57] B.H. Wells and S. Wilson, J. Phys. B: At. Mol. Opt. Phys. 22 (1989) 1285
[58] S. Wilson, in *Electron correlation, Proc. Daresbury Study Weekend Nov. 1979*, edited by M.F. Guest and S. Wilson, SERC Daresbury Laboratory (1980); Specialist Periodical Reports: Theoretical Chemistry **4** (1981) 1
[59] V.R. Saunders, and S. Wilson, Parallel Computing **17** (1991) 1025
[60] D. Moncrieff, V.R. Saunders and S. Wilson, Parallel Computing **17** (1991) 733
[61] H.M. Quiney, I.P. Grant *and* S. Wilson, *in Many-Body Methods in Quantum Chemistry*, Lecture Notes in Chemistry 52 307, ed. U. Kaldor, (Springer, Berlin, 1989)
[62] H.M. Quiney, I.P. Grant and S. Wilson, J. Phys. B : At. Mol. Opt. Phys. **23** (1990) L272
[63] S. Wilson, in *The Effects of Relativity in Atoms, Molecules and the Solid State*, edited by S. Wilson, I.P. Grant and B.L. Gyorffy, (Plenum, New York, 1991)
[64] T.E. Sorensen, W.B. England and D.M. Silver, J. Phys. B. At. Mol. Opt. Phys. **22** (1989) L539
[65] T.E. Sorensen, W.B. England, D.M. Silver and E.O. Steinborn, *Quantum Field Theoretical Methods in Chemically Bonded Systems. II. Diagrammatic Perturbation Theory*, preprint
[66] T.E. Sorensen, W.B. England and D.M. Silver, *Quantum Field Theoretical Methods in Chemically Bonded Systems. III. BCSLN-HL(N) Potential Energy Curves for the Ground States of H_2, LiH, FH and F_2*, preprint
[67] T.E. Sorensen, W.B. England and D.M. Silver, *Quantum Field Theoretical Methods in Chemically Bonded Systems. IV. Analysis of Perturbative Energy Terms for H_2, LiH, FH and F_2*, preprint