Parallel Computation of the MP2 Energy on Distributed Memory Parallel Computers

Antonio M. Márquez

Dpt. de Química-Física, Fac. de Química, Univ. of Sevilla

C/ Prof. García Gonzalez s/n, E-41012, Sevilla, SPAIN

email: marquez@quantix.us.es

1. Introduction.

The starting point for essentially all methods of *ab initio* molecular electronic structure theory is the Hartree-Fock or self-consistent-field method. Although it has long been recognized that, in many instances, it is necessary to go beyond the Hartree-Fock approximation and include the effects of the electronic correlation [1], the SCF method continues to be of immense value in its own right. The Möller-Plesset perturbation theory methodology carried-out to the 2^{nd} order has proven to be a very efficacious means to account for electron correlation, and yields significant improved computational results [2]. Current trend is towards applying both methods to increasingly larger, highly complex molecular systems.

Quantum Chemistry methods exhibit a large non-linear scaling of computational requirements with the size of the systems simulated and its adecuate application require a balanced mix of raw computer power, theoretical, and computational innovation. Parallel computers are opening the door to the study of increasingly large molecular systems, but the adecuate exploitation of this new computer architecture require fundamental algorithmic innovation to make a real impact in the study of such large molecular systems. In this paper we describe the parallelization of the SCF and MP2 energy modules of the HONDO program using the PVM system and its performance with representative samples.

2. Implementation.

An MP2 energy calculation has three major computationally intensive steps: an SCF calculation, an integrals transformation, and the evaluation of the MP2 correction to the SCF energy

itself.

In HONDO, as in others quantum chemical programs, the two-electron repulsion integrals program is driven by loops over shells, irrespective of the details of how the shells blocks are computed. The evaluation of the integrals in one block is independent of that of other blocks and the program is easily parallelized by distributing the loop over shells blocks over the different processors. This technique produces an statistical distribution of the work over the different processors and it has been found to give an excellent load balancing and consequently the parallel efficiency is very high [3]. The scalability of the integral evaluation step with up to a large number of processors has been demonstrated by Schmidt et al. [4].

The computation of the Fock matrix is also easily parallelized by having each processor compute a partial Fock matrix from its own integral sublist. The Fock matrix is then obtained by adding the contributions from all processors in a global add operation. On the standard implementation of the SCF method, each processor reads from disk its own integrals sublist on each iteration and builds its partial Fock matrix; in the direct implementation, the computation of the partial Fock matrix on each node is done on the fly, as the integrals are recomputed on each iteration. The computation of the Fock matrix is asymtotically an $O(n^4)$ step, while diagonalization and other steps on the SCF iterations are, at the most, $O(n^3)$ steps. Experience shows that for reasonably large basis sets, that are good candidates for parallelization, the performance of the program is not affected by the serial execution of those $O(n^3)$ steps, and no effort has been made for further parallelization. We note that this strategy uses the fully replicated data model which requires that the nodes have enough memory to hold the full density and Fock matrices at the same time. With current hardware trends of large memories in workstations this is not a serious limitation.

The integral transformation is the task of transforming the two-electron repulsion integrals from the AO basis set to the MO basis set, according to the fourth rank tensor transformation

$$(ij|kl) = \sum_{\mu\nu\lambda\sigma}^{N} C_{\mu i} C_{\nu j} C_{\lambda k} C_{\sigma l} (\mu\nu|\lambda\sigma)$$

In HONDO the integrals transformation step is done without pre-sorting step; i.e. the unordered list of AO integrals is taken and the first index transformation is done as

$$(\mu\nu|\lambda j) = \sum_{\sigma}^{N} (\mu\nu|\lambda\sigma) C_{\sigma j}$$

Memory requirements are important on this step. To hold all one-index transformed integrals in memory we need $N^2(N+1)n/2$ words of memory, as the μν indices are treated as a combined index. If less memory is available, the program proceeds by batches, transforming as many occupied orbitals (j) as possible for the given memory. After all (or some) of this first transformation has been accomplished, the rest of the indices are transformed step-wise to produce the full set (or a subset) of fully transformed MO integrals.

A simple way to parallelize this step is to distribute the loop over occupied MO orbitals (j) in the first index transformation over the available processors. However, because of the fourth-rank tensor nature of the integrals, the computation of a given one-index transformed integral requieres AO integrals with all possible values of σ that can be distributed over the different processors. Thus, each processor need to process its own AO integrals sublist, as well as those from other processors. This is done using a broadcast function. In sequence each processor takes a buffer of non-zero integrals, broadcast them to the other processors and then read nproc-1 buffers from them, one buffer from each processor. An important point here is that the global memory available for the computation increases with the number of processors as each processor is working in a different set of occupied MO. This is particularly important for calculations with large basis sets: in most cases the sequential algorithm will have to process the AO integrals in many batches, as not enough memory will be available to fit all one-index transformed integrals. Increasing the number of processors will reduce the number of batches needed, making possible to have super-linear speed-ups on this step.

When the integrals transformation is completed, they are used directly to form partial contributions to the MP2 correction to the SCF energy

$$E^{(2)} = -\sum_j E_j^{(2)} = -\sum_j \sum_{abi} (ai|bj)\frac{2(ai|bj)-(bi|aj)}{(\epsilon_a+\epsilon_b-\epsilon_i-\epsilon_j)}$$

This shows that the total MP2 correction to the SCF energy can be computed by adding contributions from transformed MO integrals that have the same last index (j) independently of that from others MOs. The total MP2 energy is then computed by adding all contributions together in a global add operation.

3. Concluding remarks.

Preliminary results with sample molecular systems of small an medium size show that this

parallel implementation of the computation of the MP2 energy in the HONDO program system performs extremely well [5]. Two points are still object of work: the reduction of the cubic memory requirement on the first-index integrals transformation and the reduction of communication on this step. As presented, our code can be used on a network of a few workstations or in multiprocessor computers with a small number of CPUs. Progress is being made in writting a new code that will eliminate the cubic memory requierement and will be suitable to perform large scale calculations on the current and future MPP computers.

4. Acknowledgements.

Part of this work has been finnancially supported by the DGCyT Project No. PB92-0662 and by IBM Corp., University Agreement MHVU4802. The author thanks the staff of the Centro Informático Científico de Andalucía and Convex Supercomputer SAE for computational facilities.

4. References.

[1] H.F.Schaefer, *The Electronic Structure of Atoms and Molecules: A Survey of Rigurous Quantum Mechanical Results*, Addison-Wesley, 1972.

[2] J.A.Pople, J.S.Blinkey, and R.Seeger, *Int.J.Quantum.Chem.* **5**, 280 (1984).

[3] M.Dupuis and J.D.Watts, *Theor.Chim.Acta* **71**, 9 (1987)

[4] M.W.Smidth, K.K.Balbridge, J.A.Boatz, S.T.Elbert, M.S.Gordon, J.H.Jensen, S.Koseki, N.Matsunaga, K.A.Nguyen, S.Su, T.L.Windus, M.Dupuis, and J.A.Montgomery Jr. *J.Comp.Chem.* **14**, 1347 (1993)

[5] A.M.Márquez and M.Dupuis, *J.Comp.Chem.* in press.