A NEAR-NEIGHBOUR ALGORITHM FOR
METROPOLIS MONTE CARLO SIMULATIONS.

MIHALY MEZEI

Department of Chemistry and Center for Study in Gene Structure and Function,
Hunter College of the CUNY, New York, NY 10021, USA.

A bitmap based implementation of a near-neighbour algorithm to speed up Metropolis Monte Carlo simulations is described. Tests of the algorithm on dilute aqueous solutions showed 8% - 22% overall reduction of computer time, depending on the system size.

KEYWORDS: Monte Carlo computer simulation; near-neighbour algorithm; bit map.

INTRODUCTION AND BACKGROUND

Monte Carlo computer simulations are routinely performed on molecular assemblies of O(100) molecules. These calculations are rather long: they consume 50-100 hour of mainframe time. A customary procedure to keep computer time down is the application of a spherical cutoff on the interaction potential — molecules that are farther apart than a certain treshold distance (cutoff) are considered to have zero interaction energy. This reduces the number of interactions to be computed. However, as the size of the simulation cell is increased in comparison to the cutoff value, an increasing proportion of the time will be spent on calculating distances (that, under the customary periodic boundary conditions, is somewhat time consuming).

Near-neighbour algorithms are designed to eliminate most of the unnecessary distance calculations by keeping track of the molecules that are in each other neighbourhood. The difficult part in designing such an algorithm is the efficient maintenance of this information as the molecules move. Generally, published algorithms deal with dynamical systems where all particles move simultaneously and are designed to reduce the number of interaction calculations from $O(N^2)$ to $O(N)$ [1]. The advent of supercomputers introduced an additional aspect, the problem of vectorization: algorithms that are optimal on a scalar machine may fare worse on a vector machine [2]. The comparison of algorithms is further complicated by the fact that their performance depend both on the physical problem at hand and on the computer used [3].

The Metropolis Monte Carlo technique [4] presents the additional difficulty that at each step only one molecule is moved, thus the relative cost of updating the neighbourhood informations generally becomes worse. This communication describes a (vectorizable) way to perform efficiently “local” updates using a bitmap to keep track of neighbours. Results of tests on aqueous solutions are also presented.
DESCRIPTON OF THE ALGORITHM

Let us denote by $r_{i,j}$ the distance between molecules $i$ and $j$, by $R_c$ the interaction cutoff, by $R_{nn}$ the threshold distance for neighbour definition ($R_{nn} > R_c$) and by $d_{max}$ the maximum distance a molecule can move in one step.

1. Establish a 0-1 matrix $B_{i,j}$ (bit map) such that

$$B_{i,j} = \begin{cases} 
0 & \text{if } r_{i,j} > R_{nn} \\
1 & \text{if } r_{i,j} \leq R_{nn} 
\end{cases}$$

(1)

2. Set the components of the molecular displacement vector $D_i$ for each molecule to zero.

3. When a molecule is displaced (attempted move), update the displacement vector of that molecule and check if its new magnitude is larger than $(R_{nn} - R_c)/2$. If it is not, continue with step 5.

4. For molecule $i$, calculate all distances $r_{i,j}$, update the corresponding values in the bit map $B_{i,j}$ (row $i$ and column $i$) and save the current position into $C_i$.

5. For the calculation of the energies, compute distance only for $(i, j)$ such that $B_{i,j} = 1$.

6. Repeat from step 3 for the next displacement.

RESULTS AND DISCUSSION

The algorithm has been tested in our Monte Carlo program using the Metropolis method [4] with force biased displacement [5] and preferential sampling around the solute [6] and has been tested on two different systems. The first system contained one small solute (glycine zwitterion as described by the potential library of Clementi and co-workers [7,8]) and 215 water molecules (MCY-CI potential [9] with $r_c=7.75$ Å) under face-centered cubic periodic boundary conditions while the second system consisted of one DNA tetramer duplex and 792 waters with $r_c=6$ Å in a hexagonal prism (again, using periodic boundary conditions). For both cases, $r_{nn} = r_c + 2$ Å was employed. This resulted in 50% and 90% of the waters being outside the neighbourhood sphere for the 215 and 792 water systems, respectively. The maximum water stepsize was 0.275 Å for both systems. An 8% reduction in the total run time was obtained for the first system and a 22% reduction for the second. The gain on the glycine system remained the same for $r_c+1.5$ Å $\leq r_{nn} \leq r_c+2$ Å. The frequency of bit-map updates turned out to be at every 100 configurations on the average. The possible relaxation of the update condition in step 3 has also been tested and it was found that even minor relaxation would lead to incorrect energy values within $10^6$ configurations.

The savings in computer time is already significant in the present implementation since a simulation on such aqueous systems requires 10-100 hours of present-day mainframe computer time. However, the above results can probably be significantly improved if the currently implemented bit manipulations using integer, logical and modulo operations are replaced by an assembler routine. Furthermore, since the data to be accessed is linearly spaced in
the array $B_{i,j}$, the code will easily vectorize. It may suit particularly well the Cyber-205 since that machine provides a vectorized access to an array when the items to be used are controlled by a bit map.

On the negative side, at each step one still has to look at all $N$ neighbours, even if briefly in most cases and the memory requirement of the bit map is $N^2/(\text{bits per computer word})$ computer word. Thus for very large systems or for very simple potentials the algorithm is not likely to be efficient. This disadvantageous limiting behaviour, however, did not degrade the performance for a system of $O(1000)$ molecules.

A priori comparisons with other methods can not be obtained since the efficiency of the present method depends strongly on the update frequency which is a function of the physical system at hand. Dense fluids, where the movement of the particles is largely oscillatory, appear to be able to take advantage of the present algorithm.

Acknowledgements

This research was supported by NIH grant GM 24914, NSF grant CHE-8293501 and RCMI grant #SRC5G12RR03037 from NIH to Hunter College, CUNY. Several useful discussions with Prof. K. Bencsáth are gratefully acknowledged. The referee is thanked for suggesting the use of molecular displacement vectors in Step 3.

References:


