Technical Report

IRI-TR-11-01

Monte Carlo simulations with few dof's in large molecules

Vicente Ruiz de Angulo

Februray, 2011

Institut de Robòtica i Informàtica Industrial

Abstract

This Technical Report introduces a data structure and an algorithm to efficiently to efficiently determine rigid clusters in a given molecule. This is a relevant problem in Monte Carlo simulations where the actuated DOFs change at each iteration. If the number of actuated DOFs is small, large temporary rigid groups appear during the time spanned by a MCS step. This document also provides a way of calculating the total energy in the molecules taking into account only inter-rigid energy terms. Thus, a large fraction of energy terms are not required to be computed, namely intra-rigid terms.

Institut de Robòtica i Informàtica Industrial (IRI)
Consejo Superior de Investigaciones Científicas (CSIC)
Universitat Politècnica de Catalunya (UPC)
Llorens i Artigas 4-6, 08028, Barcelona, Spain
Tel (fax): +34 93 401 5750 (5751)
http://www.iri.upc.edu

Corresponding author:
V. Ruiz de Angulo
tel: +34 93 401 7337
vruiz@iri.upc.edu
http://www.iri.upc.edu/staff/vruiz

1 Montecarlo with few dof's in large molecules

When a Montecarlo step changes simultaneously a limited number of random dof's, some distances between atoms remain constant will remain the same than in the previous step. When the molecule is large and the number of simultaneous dof is small, the fraction of constant distances is very large. It is possible to take advantage of this fact applying the trick exposed in the paper. However, since the set of dof's is different in each step, what can be considered a rigid changes from one step to the other. Here, we show how to obtain efficiently: 1) The set of atom pairs whose distance changed in the last iteration. This is enough for a steric-clash filter. 2) Total energy, avoiding the calculation of the terms that did not change in the last movement.

Our first scope is therefore to obtain

$$R_t \equiv \{(i,j)\ s.t.\ i \text{ and } j \text{ the distance between them changed in time } t\}, \tag{1}$$

or, considering the set every set of atoms whose internal distance do not change in the step as temporary rigid,

$$R_t \equiv \{(i,j)\ s.t.\ i \text{ and } j \text{ do not belong to the same rigid in time } t\}. \tag{2}$$

A similar algorithm can be used for an steric-clash filter of to calculate $E_{inter(t)}(t)$, the sum of the terms $E_{i,j}$ whose value changed in time t:

$$E_{inter(t_1)}(t) \equiv \sum_{i,j \in R_{t_1}} E_{i,j}(t) \tag{3}$$

We illustrate the algorithm with the calculus of $E_{inter(t)}(t)$, explained first for molecules with tree topology.

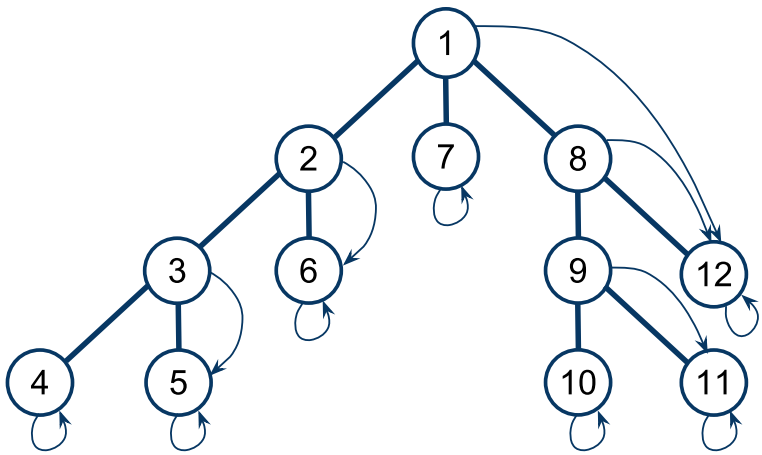

Figure 1: Depth-first numeration of atoms in a loop-free molecule. The thick segments represent bonds between atoms and the arrows represent the *end* tag, indicating the end of the subtree.

Before beginning the Montecarlo simulation (i.e., only once), the atoms should be conveniently tagged:

- Numeration tags. Each atom must numbered following a depth-first ordering (see Figure 1). Any of the possible numerations is valid, for example the first atom is arbitrary.

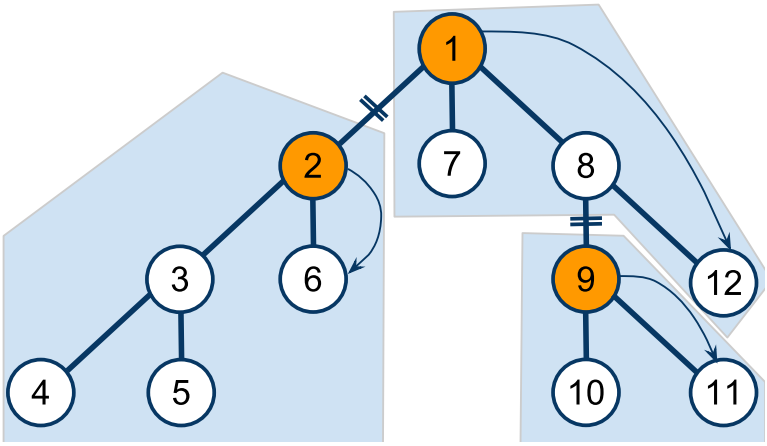

Figure 2: Example of temporary rigid configuration for the molecule in Figure 1. In the current movement the dof's between 1 and 2 and between 8 and 9 have changed. This creates three temporary rigids. The atoms with the *br* tag set to *true* (in color) indicate the beginning of a rigid. Only the values of the *end* tags (arrows) used in the current MCS step are shown.

- End of subtree tags. Each atom a (or at least, every atom that can be the beginning of a rigid) must have a field *end* with the number of the last atom of the subtree to which a belongs.

Assuming a number of dofs k change in each MCS step, a molecule without loops is composed of $k+1$ rigids. Each atom has a boolean field *br* indicating whether it is currently the first atom of a rigid or not. Besides, the same information is contained in an array $B[]$ of length $k+1$ containing the numbers of the atoms with which each rigid begins. Before each MCS step, when the i-th, $1 \leq i \leq k$, dof of freedom is selected –assume for instance a torsional dof between atom a_1 and a_2 –, $B[i]$ is set to a_2 and $a_2.br$ is set to $true$. Anyway, the algorithm below works also with non-torsional dofs (creo). Since the first atom of the molecule is not associated with any dof and is however the first atom of a rigid, $B[k+1]$ is always set to '1'. An example of rigid configuration in a MCS step is depicted in Figure 2. At the end of the MCS step the first k atoms contained in $B[]$ have to reset their *br* field to false.

The algorithm to calculate $E_{inter(t)}(t)$ is basically rigid-CLL. This processing of each rigid in this algorithm consists in two loops going through the atoms of the rigid: the first checking the interactions of each atom in the rigid an the second inserting the atoms in the cell grid. The modifications to this algorithm involves only the way in which the atoms of a rigid is obtained (a simple array or list in rigid-CLL). We will illustrate the method with the insert loop of rigid-CLL, that becomes the one showed in Algorithm 1.

Given the topology (without loops) of a molecule, and the *br* labels of the atoms indicating where the dofs that changed are located, Algorithm 1 identifies exactly the set of atoms corresponding to current rigid beginning with $B[i]$. When the molecule is large and there are few dofs, a very small percentage of atoms have their *br* fields set to true. In consequence, the most internal loop is almost never entered (i.e., it behaves like an 'if') and thus, the cost of Algorithm 1 is very similar to going over the elements of a list or array of atoms to call the $Insert$ function with them.

With respect to possible loops in topology, we distinguish between small ones, and large ones, those that cannot be closed with a chain of six or less atoms. In the first group we find all the rings in the side-chains of the amino acids and the loop in proline. Thus, a protein without

Algorithm 1: Loop through the atoms of the temporal rigid i, i.e, the rigid beginning by B[i in the current MCS step, calling $Insert$ with them.

```
1 j ← B[i];
2 end ← j.end;
3 while j ≤ end do
4 |   Insert(a) ;
5 |   j ← j + 1;
6 |   while j.br do j ← j.end + 1;
```

long range bonds only contains this kind of loops. The algorithm works with the small loops if the dofs inside the loop are not changed, which in most simulations should be avoided, because they behave like a rigid (there is no possibility of changing an internal torsion dof, without changing also an angle bond). In this case, by numbering the atoms of the loop like if they were in a linear chain, the algorithm will still determine correctly the set of atoms corresponding to each rigid. The standard PDB numeration of the atoms in a protein, is a depth-first ordering in which the atoms of the small loops satisfy this condition [Juan o alguien deberia comprobar esto].

When large loops exist, the algorithm is still the same, but in the depht-first numbering procedure a long-range bond is obviated (the two involved atoms are considered branch ends). This implies that the algorithm will not find always the maximal rigids and, therefore, not all possible distance computations will be saved. For example, in a protein loop caused by a disulphide bond, if all dofs in the loop remain fixed, the whole loop may be considered as a rigid, but the algorithm will divide it into two. Although not optimal, this is still better than classical cell-lists and the remaining rigids of the protein will be correctly identified.

The second scope is to calculate the energy $E(t) = \sum_{i,j} E_{i,j}(t)$. It can be decomposed as follows:

$$E(t) = E(t-1) + E_{inter(t)}(t) - E_{inter(t)}(t-1) \tag{4}$$

Calculating $E(t)$ in this way the computation of all intra-rigid components is avoided. $E_{inter(t)}(t)$ is computed as explained above and $E_{inter(t)}(t-1)$ is subtracted at the same time with the help of a $n \times n$ matrix containing the values of $E_{i,j}$ in the previous step. Note that the only elements of the matrix that change in MCS step t are those corresponding to R_t. Thus, we have only to initialize the matrix and the global energy variable E with $E(0)$ at $t = 0$. Then, initialize the value of E at the beginning of a MCS step with the value obtained in the previous step. And, inside the $E_{inter(t)}(t)$ calculation, at the moment of summing $E_{i,j}(t)$ to E, subtracting also $E_{i,j}(t-1)$ stored in the matrix, and replace the matrix value by $E_{i,j}(t)$ afterwards.

Acknowledgements

Thanks to the J.L. Rivero for designing the cover page.

IRI reports

This report is in the series of IRI technical reports.
All IRI technical reports are available for download at the IRI website
`http://www.iri.upc.edu`.