Improving molecular simulation : a meta optimisation of Monte Carlo parameters.

Benoit Leblanc $*$ ‡**, Evelyne Lutton** $*$
$*$ Institut National de Recherche en Informatique et Automatique, projet FRACTALES
78150 Le Chesnay - France
{Benoit.Leblanc, Evelyne.Lutton}@inria.fr

Bertrand Braunschweig ‡**, Hervé Toulhoat** ‡
‡ Institut Français du Pétrole
1 et 4, avenue de Bois-Préau
BP 311 - 92852 Rueil-Malmaison Cedex - France
{Bertrand.Braunschweig, Herve.Toulhoat}@ifp.fr

Abstract-

We present a new approach to perform molecular simulations using evolutionary algorithms. The main application of this work is the simulation of dense amorphous polymers and the goal is to improve the efficiency of sampling, in other words to obtain valid samples from the phase state more rapidly. Our approach is based on parallel Markovian Monte Carlo simulations of the same physico-chemical system, where we optimise some Monte Carlo parameters by means of a real coded genetic algorithm.

1 Introduction

Molecular simulations of dense amorphous polymers is facing the major challenge of sampling efficiency with the increase in the size and complexity of molecules : the potential energy surface of such systems are characterized by numerous local minima separated by very high barriers, hence difficult to sample either along the trajectories obtained from direct Molecular Dynamics (MD, see [1] [3]) or through conventional Markovian Monte Carlo simulations. Similar difficulties are encountered with protein folding simulations, that is translating the 1D genomic information into 3D bioactive edifice. Therefore, a major challenge for molecular simulation is to develop more efficient elementary moves ([7], [5]) in order to achieve more efficient exploration and sampling of configurational space for long chain molecules. Here, we are prototyping an approach using evolutionary algorithms (EA) in order to improve the statistical efficiency of Markovian Monte Carlo systems employing several types of such elementary moves. We show the basics of Markovian Monte Carlo simulations in section 2; section 3 presents the EA approach: encoding, fitness, algorithm, operators; section 4 details the Monte Carlo moves and the molecular model used; section 5 presents results on the case of polyethylene configurations, and is followed by a short conclusion.

2 The problem of sampling efficiency

The main purpose of molecular simulation is to simulate matter at the atomic level in order to compute some thermodynamic properties through integrals of the form :

$$< A >= \frac{\int_{x\in\Omega} A(x) exp(-\beta U(x)) dx}{\int_{y\in\Omega} exp(-\beta U(y)) dy} \qquad (1)$$

Where A is some observable property that one can compute from a given state, and Ω is the phase space, which is in the case of Markovian Monte Carlo simulation the space of geometrical coordinates of each component of the simulated system under prescribed conditions. The distribution associated to that space is the Boltzmann distribution π_B:

$$\forall x \in \Omega, \;\; \pi_B(x) = \frac{exp(-\beta U(x))}{\int_{y\in\Omega} exp(-\beta U(y)) dy} \qquad (2)$$

with $\beta = \frac{1}{k_B T}$, k_B the Boltzmann constant, T the temperature of the system, and $U(x)$ its energy. The Monte Carlo scheme is a way to estimate this integral by the following average :

$$\hat{A} = \sum_{i=1}^{N} A(x_i) \;, \;\; (x_i)_{i=1\ldots N} \in \Omega \qquad (3)$$

with the $(x_i)_{i=1\ldots N}$ some random trials according to π_B. The problem arises from the fact that one cannot take "direct" samples according to π_B, as its normalizing term would require to compute an integral over the whole phase space, which is infeasible for non trivial cases.

The key point is then to design a Markov Chain, whose state space is our phase state, in such a way that it is ergodic and admits the desired limit distribution π_B (see [4]). This is done by applying the Metropolis ([6]) algorithm that defines a random walk according to the following rules:

1 Choose any starting point x from Ω as the initial state.

2 Generate a new point y by operating a move step from x.

3 Accept or reject y as the new state with a probability $acc(x, y)$ and go back to 2 :

$$acc(x, y) = min\,(1, exp(-\beta \Delta U)) \qquad (4)$$

where ΔU is the difference of energy of the two states.

The resulting sequence of states (which may include multiple instances of the same state in case of multiple consecutive

rejections) defines a correct sample according to π_B as long as enough states are visited. Of course the minimum time of simulation depends on the considered distribution, and in the field of molecular simulation an autocorrelation criterion is often used to decide when to stop the random walk. The problem of efficiency is then to keep this *mixing time* as small as possible.

3 Evolutionary Approach to improving Markovian Monte Carlo simulations

In this section we show the chosen EA approach to improve Monte Carlo molecular simulations. The basic principle is to evolve sets of parameters describing distribution frequencies of allowed Monte Carlo movements.

3.1 Optimising parameter sets

We deal with molecular simulations of amorphous polymers for which generating valid configuration samples is long and difficult. We consider parallel simulations of the same system. The goal is to produce a correct sampling of the search space, with all parallel simulations contributing to the same sampling. In traditional Monte Carlo approaches, users empirically adjust the parameters of a simulation, e.g. in case of several allowed Monte Carlo movements, the relative frequencies of those movements along the simulation. These frequencies have no consequence on the limit distribution of valid configurations, since they only impact the way the search space is sampled during the simulation. However finding good sets of such frequencies for a specific problem can significantly improve the performance of the whole process.

The chosen reference algorithm (RA) that will be used here for the comparison of other approaches consists in n_s simulations of polymer systems with different initial states in the same physico-chemical conditions (i.e. different points from the same phase space), each simulation using an equiprobable distribution of allowed movements.

3.2 Real-valued encoding, multicriteria fitness function

The relative frequencies of movements are real numbers taking values in the interval $[0, 1]$ and are used as such in our algorithm. The sum of m frequencies for the m types of movements is equal to 1. Other real-valued parameters of movements are added to the chromosome.

The fitness function uses two complementary criteria: the first one, autocorrelation, is well known by practitioners of molecular simulation (see [5] for example). We compute the decreasing rate of autocorrelation of end-to-end chain vectors, previously normalised. The end-to-end chain vector is the vector linking both ends of a chain. For one simulation, autocorrelation of those vectors can be calculated periodically after a sufficient numbers of elementary moves, and is equal to the means of all single chain's autocorrelation over the set of chains in the simulation box:

$$a = \frac{1}{n_{samples}} \sum_{i=1}^{n_{samples}} < v_i, v_0 > \qquad (5)$$

Where $(v_i)_{i \in \{0 \ldots n_{samples}\}}$ are the normalized end-to-end vectors. The second criterion is the mass center displacement, that is, the distance between the current position of the mass center of a chain, and of its initial position. The greater the displacement, the more sampling has taken place in the simulation.

3.3 Cycles and generations

We can summarize the algorithm as follows:

- we simulate n_s systems with identical physico-chemical parameters;
- each system has a chromosome composed of the real-valued parameters of the simulation, in particular the frequencies distributions; the initial population is generated randomly;
- one simulation of one system consists in n_c cycles;
- those n_c cycles are further segmented in n_g generations (see figure 1) inside which individuals of the populations are evaluated along with the fitness criteria presented above (Note: since an individual is a set of parameters, its evaluation requires many Monte Carlo movements in cycles).
- Our EA uses a range of standard strategies and parameter settings: Stochastic universal sampling ([2]) with full population replacement, uniform crossover and gaussian mutation operators.

A cycle consists in several elementary Monte-Carlo moves (trials to generate new conformations), totalling a pre-defined CPU time. This way, fitnesses represent the true efficiency of a parameter set over a specified period of time. This makes our algorithm dependent on the hardware, OS and software used, but supportive of cheap and efficient mixing moves.

The fitness of an individual is computed over n_c/n_g cycles, and is defined as:

$$f(x, \omega) = [1 - a_{avg}(x, \omega)] \times d^2_{avg}(x, \omega) \qquad (6)$$

where ω denotes the random part of the simulation, a_{avg} the average autocorrelation, and d^2_{avg} the average square displacement. Strictly speaking, improving efficiency is a multiobjective problem (i.e. optimising a and d^2). We choose to formulate this problem via a single fitness because these two components are closely related, as we shall see in the numerical experiments. Moreover a desirable behaviour is a good "global" performance, i.e. all numerical simulations must be efficient. The goal of our evolutionary algorithm is thus to find rapidly many good solutions instead of locating a particular optimum.

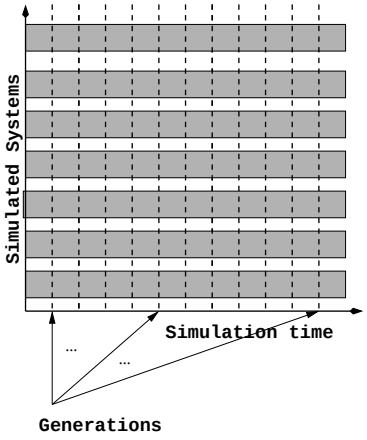

Figure 1: Schematic representation of our EA-based parallel MC simulation: bars stand for simulated systems. Each system is given MC move frequencies corresponding to an individual of the population, performance is measured on n_c/n_g MC cycles and returned as the fitness score. At each new generation, new individuals are created and each simulation continues with corresponding (hopefully better) frequencies. In comparison, for the reference algorithm (RA), a unique set of frequencies is used for all the systems during all the simulation time.

4 Molecular model, Monte Carlo moves

Molecular simulation specialists usually compare their results with experimental measurements. There is now a solid background of such comparisons that allows to conclude that molecular simulation correctly predicts physical properties of matter. Instead, we compare our results with those obtained by other authors on the same polymers systems. In this paper, we adopt the polyethylene model described in [5], with the same constants:

- unified atom model considering each CH2, CH3 group as a single active site;
- fixed monomer-to-monomer link length of 1.54 Å, corresponding to C-C bond length;
- Lennard-Jones pair interaction potential. The characteristic radius is $\sigma_{LJ} = 3.94$ Å, and the potential well's depth $\epsilon_{LJ} = 0.098\, kcal/mol$. For two sites i, j whose distance between is r_{ij}, then we have:

$$\vartheta_{LJ}(r_{ij}) = 4\epsilon_{LJ} \times \left[\left(\frac{\sigma_{LJ}}{r_{ij}} \right)^{12} - \left(\frac{\sigma_{LJ}}{r_{ij}} \right)^{6} \right] \quad (7)$$

  In practice, in order to compute the total interaction potential of a particular site, we only consider neighbouring sites that are located within a cutoff radius σ_{cut} (here $\sigma_{cut} = 8.67$ Å). A term taking into account long range interactions, depending on the density and on this radius, is finally added.

- Van der Ploeg and Berendsen tension potential, function of a bond angle θ (given by 3 following monomers of the same chain) allowed to fluctuate around the mean value $\theta_0 = 112^o$. Given $k_\theta = 57950K.rad^{-1}$:

$$\frac{\vartheta_{VPB}(\theta)}{k_B} = \frac{1}{2} \times k_\theta \times (\theta - \theta_0)^2 \quad (8)$$

- Ryckaert and Bellmans torsional potential, function of dihedral angle ϕ (given by 4 following monomers of the same chain). Given $c_0 = 1116\ K$, $c_1 = 1462\ K$, $c_2 = -1578\ K$, $c_3 = -368\ K$, $c_4 = 3156\ K$, $c_5 = -3788\ K$:

$$\frac{\vartheta_{tor}}{k_B} = \sum_{k=0}^{5} c_k cos^k(\phi) \quad (9)$$

- Cubic simulation model box with periodic boundary conditions;
- Splitting the simulation box in cubix cells so as to accelerate interaction potentials computation. The length of a cell has to be larger than σ_{cut}, hence the search for neighbours can be restricted to the cell of the site and the 26 neighbouring cells. This implies that this method is useful only if the simulation box is at least 4 times larger than σ_{cut}, corresponding to at least 64 cells.

4.1 Monte-Carlo Moves

Our Monte Carlo moves are commonly used in molecular simulation literature:

- Translation: the whole molecule is translated along a random path. The translation distance is randomly chosen in the interval $[0, d_{max}]$. The parameter d_{max} is usually dynamically adjusted in order to reach a prescribed acceptance rate. In our case d_{max} is an element of the genome. This translation move can generate expensive calculations of Lennard-Jones potential if there are too many interaction sites and is therefore costly for long molecules, as we will see later;
- Rotation: an ending monomer is rotated within a sphere centered on its preceding site; energy variation must be calculated for the three potentials; this simple move only concerns one monomer and is therefore very fast to execute;
- Reptation (or "snaking"): an ending monomer is removed, added at the other end of the chain, and rotated. The calculation requirements are the same as with rotation;

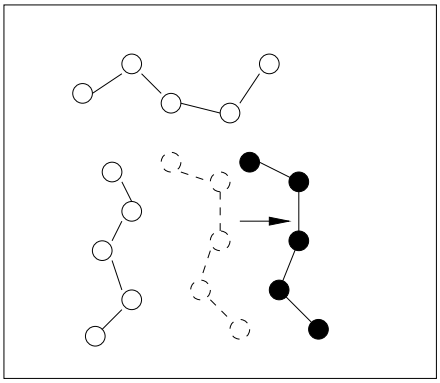

Figure 2: Translation

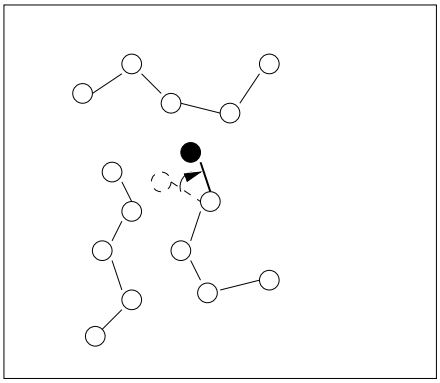

Figure 3: Rotation

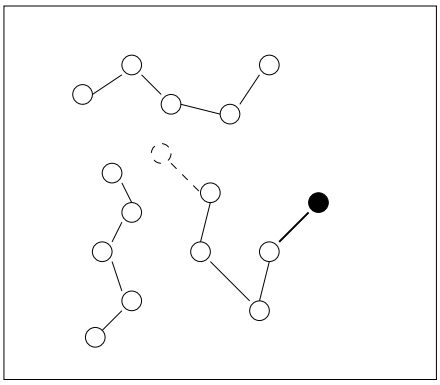

Figure 4: Reptation

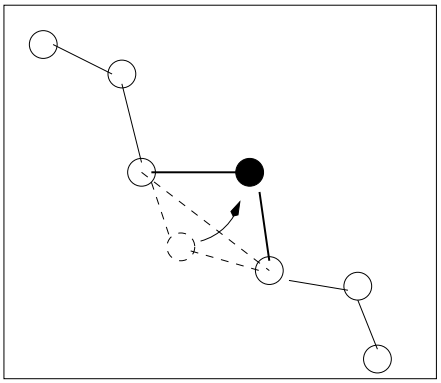

Figure 5: Flip

- Flip: a monomer inside a chain is rotated along the axis of its two neighbouring sites. The site is moved, two tension angles change, four torsion angles change. The rotation angle is randomly chosen in the interval $[0, \phi_{max}]$.The parameter ϕ_{max} is usually dynamically adjusted in order to reach a prescribed acceptance rate, and is an element of the genome in our case.

5 Numerical results.

We now present some numerical experiments in order to compare efficiency of our EA algorithm to reference algorithm (RA). We recall that the latter consists in the parallel and indepent MC simulations of n_s polymer systems, with the same MC parameters and in the same physico-chemical conditions.

The simulations are performed in the $NnVT$ ensemble :

- N : constant number of chains. $N = 20$.
- n : constant number of monomers. $n = 1200$ (which means polymers of size 60).
- V : constant volume. This condition is imposed by a constant simulation box length $l_B = 35.64\mathring{A}$.
- T : constant temperature. $T = 394.4K$

The simulations were performed on a bi-processor PC (Intel Pentium II 350 MHz), and the simulation cycle was defined as 1 second of process time. On average, about 500 elementary moves are performed within a cycle. The test consisted in 16 instances of this system, which means that the quantities plotted in the following figures are averages on these 16 parrall runs. The total simulation time consisted in $n_c = 5000$ cycles (divided in $n_g = 100$ generations of 50 cycles in the case of the EA). To summarize EA parameters :

- Population size : 16
- Number of generations : 100
- Full population replacement with Stochastic Universal Sampling

- Crossover probability : $p_c = 0.5$
- Mutation probability : $p_m = 0.1$

5.1 Set A

This first set of tests use the four MC moves presented before, with the following frequency distribution for the RA :

Rotation	Reptation	Flip	Translation
60/181	60/181	60/181	1/181

As the translation simultaneously moves 60 monomers at a time, its frequency is divided by this amount.

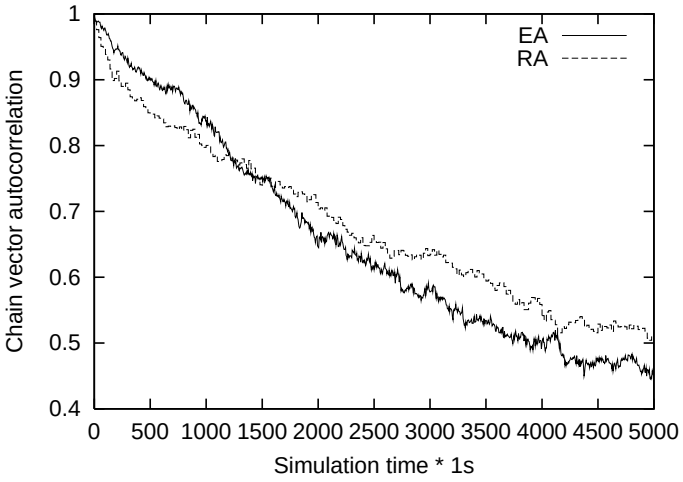

Figure 6: Set A : averaged end-to-end vector autocorrelation plotted against cycles of simulation

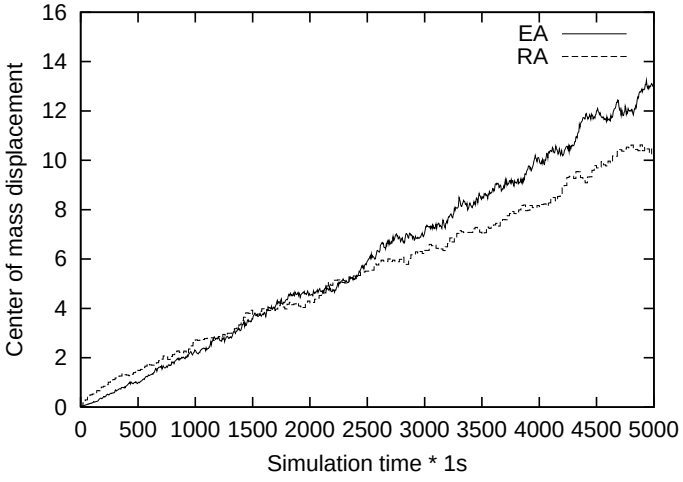

Figure 7: Set A : averaged mass center displacement (in reduced units, $1unit = \sigma_{LJ}^2 = 15.52\ \mathring{A}^2$) plotted against cycles of simulation.

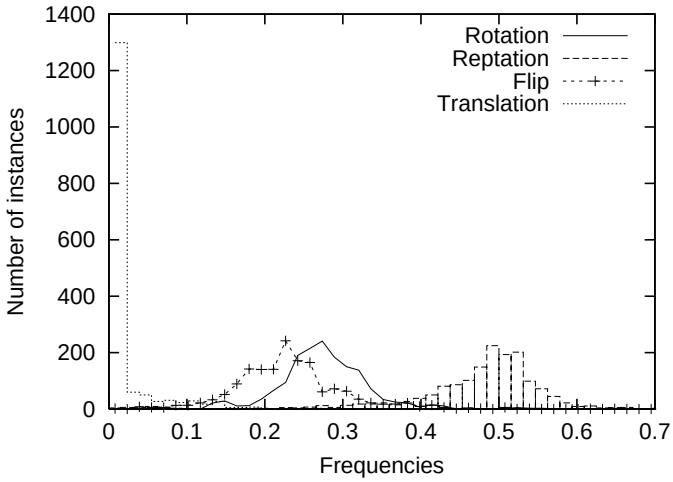

Figure 8: Set A : histogram of MC moves frequencies. For each move, values of all individuals of all generations are counted.

We see on figures 6 and 7 that the RA starts to perform better than the EA, but both finish at approximately the same performance, with a little advantage for the EA. This can be explained by the fact that in the initial population each move have on average the same frequency $1/4$ (in comparison to $1/181$ for the RA), including the translation, which reveals to be bad moves for this system. And as the evolution discards it, performances improve. We can check this if we look at the histogram (figure 8) of the moves where translation frequencies are located near 0 for most, and only a few values a little greater (corresponding to first generations).

5.2 Set B : without translation move

We now drop the translation move. The three remaining moves have now an equal frequency of $1/3$ in the RA.

Figures 9 and 10 show that our EA is clearly able to find good parameters and at the same time leads to good performances in comparison to the RA. The histogram of moves (figures 11) shows that the reptation has a major role for this system, which was also the case for the previous set, but this cannot be a general rule. In fact each move may perform differently depending of the type of simulated molecules and also depending on the conditions (temperature, density, pressure, etc...), and this motivates the adoption of a strategy that finds “on the fly” some good parameter settings.

6 Conclusion and future exciting work

We have shown how to evolve sets of parameters of Markovian Monte Carlo molecular simulations using a real-valued genetic algorithm. Using this approach, a better sampling of the configurational space is obtained at almost no cost, compared to similar reference simulation based on fixed sets of parameters. The practicality of our approach has been shown

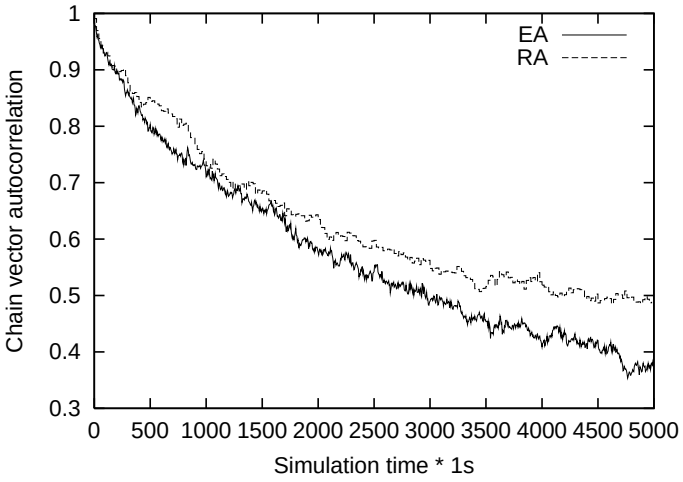

Figure 9: Set B (without translation move) : averaged end-to-end vector autocorrelation plotted against cycles of simulation.

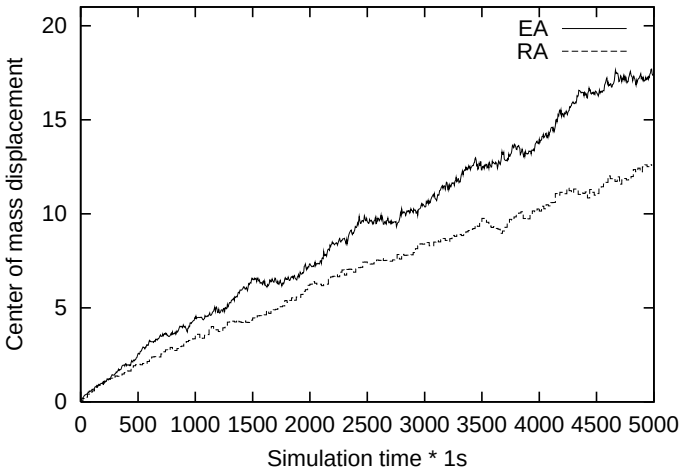

Figure 10: Set B (without translation move) : averaged mass center displacement (in reduced units, $1\ unit = \sigma_{LJ}^2 = 15.52\ \mathring{A}^2$) plotted against cycles of simulation.

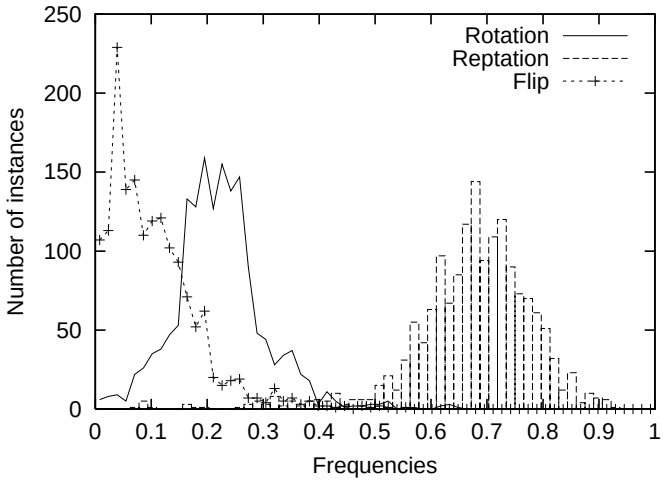

Figure 11: Set B (without translation move) : histogram of MC moves frequencies. For each move, values of all individuals of all generations are counted.

on a well-known reference problem in simulation of amorphous polymers. It can and will be extended to more complex material for which obtaining a correct sampling is even longer and more difficult. For this purpose, we plan to implement our algorithm on a "PC farm" of several hundreds of PCs.

Our approach is not a direct optimisation task. Instead, our only goal is to generate a better sampling of a very large search space using evolution as a means to increasing diversity while keeping a high acceptance rate. It is particularly important in molecular simulation where sampling is the key factor. On the contrary to pure optimisation EA implementations, the important output of our algorithm is not the best individual of the final population but rather the quality of all individuals generated along the evolution of the EA. As such it can be considered as an evolutionary adaptive system acting in a noisy environment.

Bibliography

[1] (1987) M.P. Allen, D.J. Tildesley, *Computer Simulation of Liquids*, Oxford Science Publications.

[2] (1987) J.E. Baker. *Reducing bias and inefficiency in the selection algorithm*. Genetic Algorithms and their applications : Proceedings of the Second International Conference on Genetic Algorithms, 14-21.

[3] (1996) Daan Frenkel and Berend Smit *Understanding Molecular Simulation, From Algorithms to Applications*, Academic Press.

[4] (1998) Mark Jerrum. *Mathematical Foundations of the Markov Chain Monte Carlo Method*, in "Probabilistic Methods for Algorithmic Discrete Mathematics", M.

Habib, C. Mc Diarmid. J. Ramirez-Alfonsin, B. Reeds (Eds), Springer Verlag.

[5] (1999) V.G. Mavrantzas, T.D. Boone, E. Zervopoulou and D.N. Theodorou. *End-Bridging Monte Carlo: A Fast Algorithm for Atomistic Simulation of Condensed Phases of Long Polymer Chains.*, Macromolecules, **32**, 5072-5096.

[6] (1953) N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.N. Teller, and E. Teller, *Equation of state calculations by fast computing machines*. J. Chem. Phys., **21**:1087-1092.

[7] (1992) J.I. Siepmann, D. Frenkel, *Configurationnal-bias Monte Carlo: A new sampling scheme for flexible chains*, Mol. Phys., **75**, 59-70.