# LMProt: An Efficient Algorithm for Monte Carlo Sampling of Protein Conformational Space

Roosevelt Alves da Silva,* Léo Degrève,* and Antonio Caliri†
*Departamento de Química, Faculdade de Filosofia Ciências e Letras de Ribeirão Preto, and †Departamento de Física e Química, Faculdade de Ciências Farmacêuticas de Ribeirão Preto, Universidade de São Paulo, 14040–903 Ribeirão Preto, São Paulo, Brazil

ABSTRACT A new and efficient Monte Carlo algorithm for sampling protein configurations in the continuous space is presented; the efficiency of this algorithm, named Local Moves for Proteins (LMProt), was compared to other alternative algorithms. For this purpose, we used an intrachain interaction energy function that is proportional to the root mean square deviation (*rmsd*) with respect to $\alpha$-carbons from native structures of real proteins. For phantom chains, the LMProt method is $\sim 10^4$ and 20 times faster than the algorithms Thrashing (no local moves) and Sevenfold Way (local moves), respectively. Additionally, the LMProt was tested for real chains (excluded-volume all-atoms model); proteins 5NLL (138 residues) and 1BFF (129 residues) were used to determine the folding success $\xi$ as a function of the number $\eta$ of residues involved in the chain movements, and as a function of the maximum amplitude of atomic displacement $\delta r_{max}$. Our results indicate that multiple local moves associated with relative chain flexibility, controlled by appropriate adjustments for $\eta$ and $\delta r_{max}$, are essential for configurational search efficiency.

## INTRODUCTION

When the investigation of protein-solvent systems requires an exhaustive search on the configuration space, the MC method is considered the main available simulation tool. Particularly, there are different MC techniques to simulate thermodynamic and kinetic aspects of many distinct systems. In general, these techniques may be divided into two main classes, namely, dynamic and nondynamic MC methods. The first class is characterized by a set of standard moves that resemble the specific system dynamics at the microscopic scale (Binder and Baumgartner, 1992; Degrève and Caliri, 1995). It is important to observe that the term ''dynamic MC'' is currently used in reference to the emulation of real molecular moves, it may (and should not) be confused with another method also named ''dynamic Monte Carlo method'', which simulates the real-time evolution of physical systems (Aièllo and da Silva, 2003). On the other hand, the main concern of the nondynamic MC method is to adequately explore the phase space without worrying about the sequence of configurations in the time. Therefore, once the ergodic hypothesis is satisfied and the absence of bias is guaranteed, this method is exclusively used for dealing with thermodynamic amounts. However, for chain systems, the main general difficulty in using the nondynamic MC method is exactly at this point: to ensure a good statistically representative sampling (configurations without spurious bias) for ensemble averaging. Then, one may suggest that a practical test for a specific sampling technique is to use small systems for which it should be possible to exhaustively investigate the configurational space, and then compare the complete ensemble averages to the simulated averages. Indeed, this approach constitutes a basic criterion for the quality of an MC simulation (Zhou and Berne, 1997; Berne and Straub, 1997), but for problems like protein folding in which more than just equilibrium states are necessary, its use is still a challenge: the existence of a high number of metastable states (local minima) promotes energetic traps that may reduce the efficiency and the accuracy of nondynamic MC methods.

In this work, we present an MC algorithm that increases the efficiency of the configurational search through a simple method capable of producing multiple local moves in flexible peptide chains. The method is versatile, allowing the control of the magnitude of moves size (average displacement of atoms for movement) and of the number of involved residues in the local movement, improving the performance of the configurational search and is expansible for use during different stages of the same run.

## MONTE CARLO MOVES FOR PROTEINS

A traditional way to generate protein chain configurations is to promote independent small perturbations on the dihedral angles of the main chain (Lal, 1969; Madras and Sokal, 1988). This type of movement, called pivot or Thrashing algorithms, has been criticized for producing very low acceptance rates for the new configurations (Zhou and Berne, 1997; Favrin et al., 2001; Shimada et al., 2001). In real proteins, collective moves that result from simultaneous movements of sequential groups on the chain are among the main factors responsible for the escape from energy and topological barriers. So, chain movements that include such simultaneous small energetic changes along several sequential chain groups may become crucial to improve the method efficiency. In general, algorithms that do not present such properties include

*Submitted February 12, 2004, and accepted for publication June 2, 2004.*

Address reprint requests to Roosevelt Alves da Silva, E-mail: roos@obelix.ffclrp.usp.br.



0006-3495/04/09/1567/11 $2.00

doi: 10.1529/biophysj.104.041541

complementary artifacts to be successful (Berg and Neuhaus, 1991; Hansmann and Okamoto, 1993). For example, the multicanonic algorithm (Moret et al., 2002) and simulated tempering (Lyubartsev et al., 1992; Marinari and Parisi, 1992) can demand numbers of nonfeasible calculations depending on the system. Thus, the application of single pivot-type movements to generate protein configurations (or other macromolecules) has been considered inadequate for a complete and efficient MC simulation, because, as it has been pointed out (Cahill et al., 2002), simulation algorithms must prevent the breaking of kinematics principles, such as the overlapping and transposition of chain groups. The observance of this principle becomes even more critical when protein chains are absorbed in solvent, where drastic configuration changes are frequently forbidden by additional restrictions imposed by the solvent molecules (Shimada et al., 2001).

Therefore, to guarantee an appreciable acceptance rate, it is usual to change the configurations by considering only a few sequential chain elements at a time. This type of local chain deformation was first analyzed by Go and Sheraga (1970) and later, a new algorithm, named ''concerted rotation'', was introduced by Dodd et al. (1993) following the same reasoning. In this algorithm, the movements consist of combined rotations around seven adjacent skeletal bonds, leaving the rest of the chain unaffected. For the accomplishment of such moves, a temporary change of variables is necessary, and so it is required that the Jacobians of such transformations be calculated for both the original and the new attempted configurations, to ponder the transition probabilities adequately and to satisfy the detailed balance of the system. The combination of these local movements with other movements has enabled one to describe the equilibrium states of polymer systems involving chains of different sizes (Siepmann and Frenkel, 1992; de Pablo et al., 1992).

Similar to what occurs in the concerted rotation method used for polymers, a simple way to prevent large configurational changes in proteins is to allow that a set of $n$ adjacent dihedrals angles $\{i\}$ be modified only by small amounts $\{\delta\phi_i\}$, in such a way that the movement of the chain becomes practically local. Actually, this type of movement involving two, four, or six combined rotations has been successfully applied in a thermodynamic and kinetic study of crambin's folding (Shimada et al., 2001). The effective step size $S$, defined as

$$S = \left[\sum_{i=1}^{n} (\delta\phi_i)^2\right]^{1/2}, \quad (1)$$

was obtained from an angular Gaussian distribution of width $\sigma = 2°$, producing mostly local moves but still with a poor acceptance rate of only 10% for protein compact states. With the application of a conformation-dependent Gaussian distribution, Favrin et al. (2001) have increased $S$ by a factor of 3 when compared to the Gaussian distribution used by Shimada et al. (2001), without affecting the local move character of the algorithm.

Recently, another alternative move set has been presented and it can be considered as local moves, (Cahill et al., 2002, 2003) by allowing only very small rotations. This is done in such a way that the perturbed atoms are moved through a distance no larger than 0.05 Å. The efficiency of the proposed algorithm was tested by using a function that is proportional to the global root mean square deviation (*rmsd*), as interactional energy,

$$rmsd = \left[\frac{1}{N}\sum_{i=1}^{N} D_i^2\right]^{1/2}, \quad (2)$$

where $D_i$ is the distance between the $\alpha$-carbons of residue $i$ of the reference structure and of residue $i$ of the correspondent simulated structure; the sum of $i$ considers all chain's residues. The use of such nonphysical potential is important for the comparative determination of the efficiency of a proposed algorithm. This is done by isolating its capability in generating configurations that satisfy the basic topological constraints of the chain (for instance, the chain-excluded volume), which in turn avoids energetic traps generated by specific potentials for protein-like chains. For phantom chains (that is, chains with no physical volume), this approach can drive the chain to the native structure very quickly; however, its efficiency is strongly affected if the chain-excluded volume is introduced. Indeed, very small configurational jumps, determined by small angular change amounts $\{\delta\phi_i\}$, are not effective to promote the escape from local energy minima or topological traps, leading to a very inefficient search on phase space. Therefore, in this work, we introduce an alternative MC algorithm, named Local Move for Proteins (LMProt), capable of generating valid local configurational changes in flexible chains by means of larger configurational perturbations than the ones encountered in other methods (Eq. 1). This factor significantly increases the sampling efficiency of the conformational space. In this method, the chain degrees of freedom are represented by dihedral angles, bond lengths, and angles between bonds. Each configuration is produced by means of two simple well-known processes, which have still not been applied in this context. The first one consists of random modifications to positions of a set of atoms (nitrogen, N; carbon-$\alpha$, CA; and carbons, C) consisting of $\eta$ consecutive residues. In the second step, the modified atomic coordinates are corrected by means of Lagrange multipliers to preserve the chain basic geometry.

In the next sections, the LMProt algorithm is fully described and its efficiency and accuracy are compared against results obtained by other recently proposed algorithms involving local and nonlocal moves (Cahill et al., 2002, 2003). Two types of analysis were accomplished. First, the native structure of the protein 16PK was used as a reference

and initially the effect of excluded volume was not considered to enable direct comparison with results of other algorithms. Specifically, the same global interactional energy considered by Cahill et al. (2002) was also used here. In the second analysis, hard sphere-type potential was added to consider the chain-excluded volume effect (model with all atoms). Two proteins were employed as references in this case, namely the 5NLL and 1BFF; all atomic degrees of freedom were considered, including those of side-chain atoms. The folding success $\xi$ in finding the native structures in a given time window $t_w$ and its corresponding folding speed $\xi'$ were estimated as functions of the number of sequential residues $\eta$ involved in the local movements of the chain and of the maximum displacement $\delta r_{max}$ allowed for each atom.

## THE LMPROT ALGORITHM

### Local configurations method

The LMProt algorithm is specially designed to produce local chain configurations in a flexible chain. It is considered as being ''local configuration change'' when a new configuration is obtained through the modification of all atomic coordinates of $\eta$ consecutive residues without modifying the positions of the remaining residues (of course, preserving all restrictions imposed by the chain geometry) or breaking kinematics principles, such as the overlapping and transposition of chain groups.

If $\eta$ residues of the chain are randomly chosen to generate a new configuration, the LMProt method combines $3\eta + 2$ new dihedral angles, including the angles around peptide bonds, and all pertinent bond lengths and bond angles (see scheme in Fig. 1). Starting from a given configuration, a new one is obtained by the LMProt as follows:

1. A number $a \leq \eta \leq b$ of consecutive residues are randomly selected.
2. The coordinates $\{\mathbf{r}_i\}$ of all atoms (N, CA, and C) of the $\eta$ residues are changed to $\{\mathbf{r}'_i\}$ through specific random moves $\{\delta r_i\}$, each one satisfying $-\delta r_{max} < \delta r_i \leq \delta r_{max}$, where $\delta r_{max}$ is the maximum displacement allowed for each atom.
3. The coordinates $\mathbf{r}'_i$ are then recursively modified to $\mathbf{r}''_i$ until all pairwise geometric constraints $\{d_{ij}\}$, which depend only on the set of fixed interatomic distances, are recovered. Two particular atoms $i$ and $j$ are assumed as correctly constrained when the distance $d_{ij}$ between them (fixed by the chain geometrical constraints) is preserved, independent of the chain configuration, and remains inside a particular interval. In other words,

$$(l_{ij} - \delta l_{ij})^2 \leq d_{ij}^2 \leq (l_{ij} + \delta l_{ij})^2, \tag{3}$$

where $\delta l_{ij}$ is the maximum deviation allowed for the fixed geometrical distance $l_{ij}$ between atoms $i$ and $j$, which is the flexible chain condition. The $\{l_{ij}\}$ values used are those from the force field GROMOS96 (van Gunsteren et al., 1996), which are shown in the second column of Table 1 for atoms belonging to the main chain. For $\eta$ modified residues, the condition imposed by Eq. 3 must be obeyed for all constraints $\{d_{ij}\}$ involving all atoms (N, CA, and C) of the $\eta$ residues (see Fig. 1, for example). There is a total of $7\eta + 4$ of such $d$-constraints involved in preserving the geometry of the main chain: $6\eta + 3$ are related to bond lengths and bond angles, and the remainder $\eta + 1$ are responsible for preserving the amide plans of the main chain. For $\eta = 3$, the constraints involved in a local move are shown in Fig. 1.

To recover the geometrical constraints of the chain, the LMProt algorithm uses the method of the Lagrange multipliers applied iteratively after any set of movements $\{\mathbf{r}_i \rightarrow \mathbf{r}'_i\}$. Let $\mathbf{b}_{ij}$ be a vector defined by

$$\mathbf{b}'_{ij} = \mathbf{r}'_i - \mathbf{r}_j, \tag{4}$$

where $\mathbf{r}'_i$ and $\mathbf{r}_j$ are the positions of atoms $i$ and $j$, which are supposedly being geometrically constrained. In the iterative process, the corresponding Lagrange multiplier $\lambda_{ij}$ appropriately corrects the magnitude of $\mathbf{b}'_{ij}$ to satisfy Eq. 3. Therefore, it is defined by

$$\lambda_{ij} = \left(\frac{l_{ij}}{|\mathbf{b}'_{ij}|}\right), \tag{5}$$

and the new position of $i$ can be redefined for all atom coordinates through

$$\{\mathbf{r}''_i\} = \{\mathbf{r}_j + \lambda_{ij}\mathbf{b}'_{ij}\}, \tag{6}$$

to satisfy Eq. 3. For all the modified atom positions, Eq. 6 is applied until Eq. 3 is satisfied for all constraints involved for

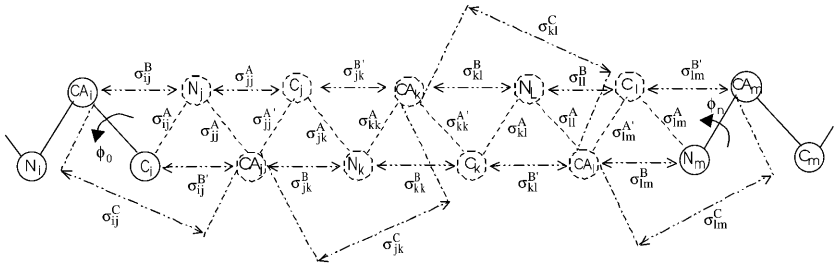



FIGURE 1 Scheme showing the constraints involved in a local movement for the LMProt algorithm. In this example, three residues have their *N*, *CA*, and *C* perturbed atoms (*hatched lines*). Bond lengths and bond angles are preserved by means of type A and B constraints, respectively. Type C constraints are defined to limit the changes in the amide planes. A local configuration produced by the LMProt is equivalent to the perturbation of dihedral angles, bond lengths, and bond angles, simultaneously.

**TABLE 1 Interatomic distances (Å)**

| Constraints | $l_{i,j}$* | SD† |
|---|---|---|
| $N_iC_{i-1}$ | 1.330 | 0.004 |
| $CA_iC_i$ | 1.530 | 0.004 |
| $N_iCA_i$ | 1.471 | 0.004 |
| $N_iCA_{i-1}$ | 2.41 | 0.02 |
| $N_iC_i$ | 2.45 | 0.02 |
| $CA_iC_{i-1}$ | 2.45 | 0.02 |
| $CA_iCA_{i-1}$ | 3.81 | 0.02 |
| $CB_i^{PRO}C_{i-1}$ | 3.72 | 0.04‡ |

*GROMOS96 force field (vanGunsteren et al., 1996).
†SD for tolerance level ($\delta l_{i,j}$) of 1%.
‡Proline residue: a tolerance level of 2% was permitted.

each moved atom; a similar technique is also used by the algorithm SHAKE (Ryckaert et al., 1977). A simplified scheme is shown in Fig. 2 to see how the LMProt generates a particular configuration from the solution of this system of linear equations. The positions of hydrogen (H) and $\beta$-carbon atoms in the main chain are directly determined from the positions of atoms that have already been determined. The side chains are also moved by a similar process, as shown in Fig. 2, but in this case the convergence is very fast.

The proline residue is a special case; for its description an additional constraint is necessary to keep the distance between the carbon atoms $C_{i-1}$ of the residue $i - 1$ and its $CB_i^{PRO}$ atom (see Table 1). This constraint is equivalent to keeping the dihedral angle between $N_i$ and $CA_i$ of the corresponding proline residue constant.

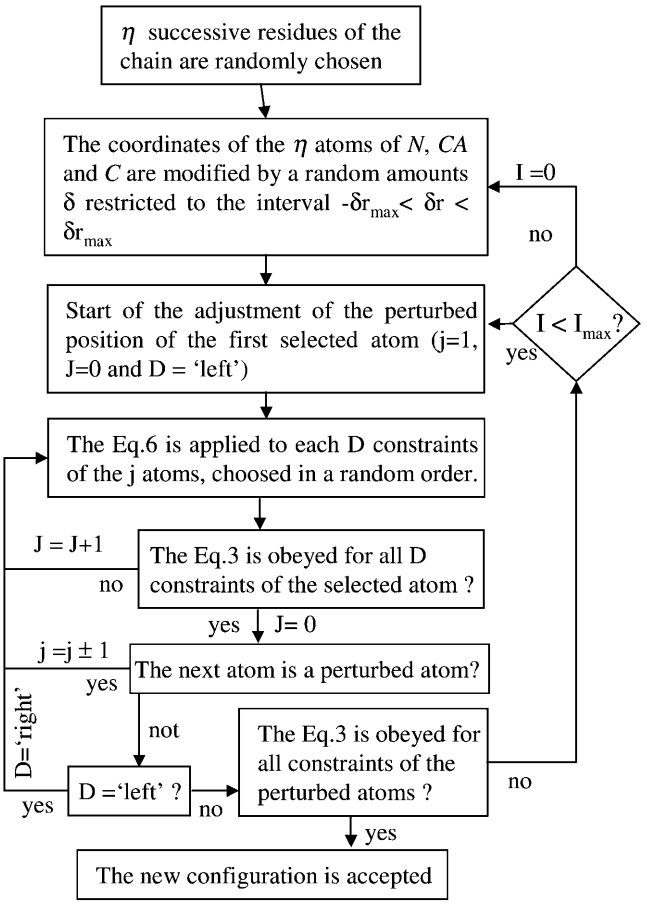



FIGURE 2 Scheme used for the LMProt algorithm to produce local configurations. The value of $I_{max}$ shown in the figure denotes the maximum number of iterations allowed when trying to satisfy Eq. 3 for all the chain constraints. All the simulations used $I_{max} = 400$. The direction of the iterative procedure is indicated by string $D$. $D =$ *'left'* indicates that the current adjustment of moved atoms must be executed in relation to left constraints. When all the moved atoms are adjusted with $D =$ *'left'*, then $D$ is inverted and the procedure is continued. Value $J$ indicates the number of iterations performed for an atom to have its position adjusted in relation to constraints of the left or right type ($\langle J \rangle = 4$).

## GENERAL ASPECTS OF THE ALGORITHM

Protein 1BFF (129 residues) was firstly used to perform preliminary check simulations (CS) of the LMProt algorithm. A total of 120 independent simulations was considered, each one being composed of $\sim 10^6$configurations generated according to the scheme shown in Fig. 2. This was done to analyze the general dependency of the code on its parameters. Half of them, that is 60 simulations, were performed for a tolerance level on the geometrical constraints of 0.5%. In other words, $\delta l_{i,j} = 0.5 l_{i,j}/100$, for six different values of $\eta$, namely, $3 \leq \eta \leq 8$. Similarly, the remainder of the cases were destined for simulations with a 1% tolerance level. In Table 1, the GROMOS prefixed main-chain geometrical constraints $\{l_{i,j}\}$ used in this work are listed, as well as the corresponding standard deviation (SD) estimated along the CS for a 1% tolerance level ($\delta l_{i,j}$). For a 0.5% tolerance (not shown here), SD is correspondingly smaller.

The flexibility of the main chain is obtained by means of small variations on the angles between the peptide planes $CA_i-C_i-N_i$ and $C_i-N_{i+1}-CA_{i+1}$, as a consequence of small changes allowed on the prefixed values of all geometrical distances $\{l_{i,j}\}$ of the main-chain atoms. Indeed, tolerances $\delta l_{i,j}$ of 0.5% and 1% on each constraints $l_{i,j}$ produce variations up to $\pm$ 11° and $\pm$ 15° between peptide planes, respectively.

The CS also showed that for the LMProt, the average timing cost $\tau_L^\eta$ for each new generated configuration depends on a combination of $\eta$ and $\delta l_{i,j}$, the code efficiency on appropriately adjusting the geometrical constraints $\{l_{i,j}\}$, as summarized in Fig. 3 and Table 2. Fig. 3 shows the distribution of the number $I$ of necessary adjusting iterations required to satisfy all the geometrical constraints $\{l_{i,j}\}$ for the corresponding tolerance $\{\delta l_{i,j}\}$. To construct each curve, data from $10^7$ configurations (10 complete runs) were employed. For both levels of tolerance, as increase in $\eta$ displaces the curve peak to the left (smaller $I$) and makes its amplitude increase sensibly. Therefore, increasing $\eta$ tends to reduce the timing cost $\tau_L^\eta$, which is numerically confirmed in Table 2. Note that the code efficiency (Table 2) is also improved as $\eta$

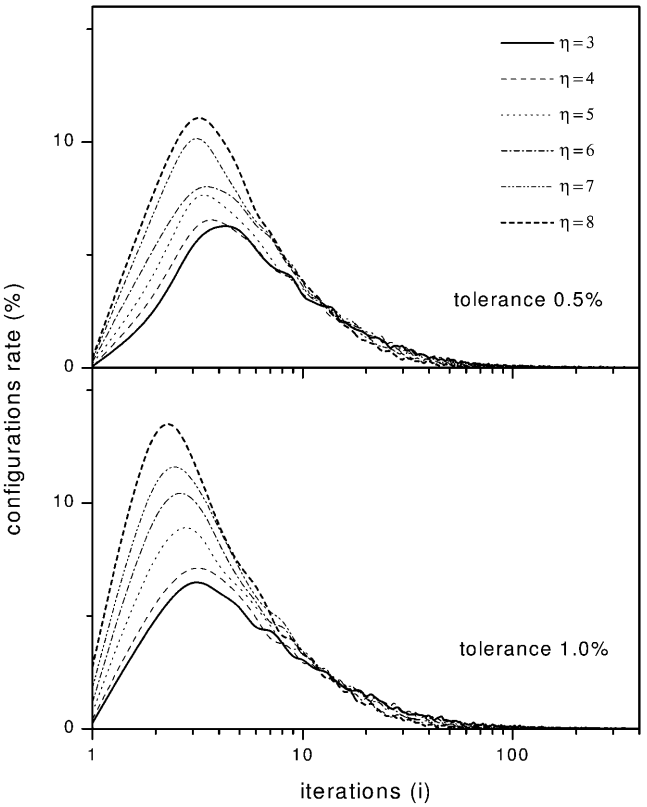



FIGURE 3 Distribution of the relative number $I$ of necessary adjusting iterations required to satisfy the geometrical constraints $\{l_{i,j}\}$, after attempt moves, for distinct $\delta l_{i,j}$ and $\eta$. The $I$ axis is shown in log scale to facilitate the viewing of the most significant part of the range $1 \leq I \leq 400$; the curves were also smoothed by adjacent averages. The area under each curve is the same: all curves collapse at about $I = 10.5$; for larger $I$, the original up-down configuration of the curves is reversed.

increases, because for $\delta l_{i,j}$ at 0.5% (1%), the fraction of failure $\Phi$ in adjusting the constraints $\{\delta l_{i,j}\}$ (number of attempts $I \leq I_{max} = 400$) changes from $\Phi = 22.2\%$ (5.1%) for $\eta = 3$, to $\Phi = 0.00\%$ (0.00%) for $\eta = 8$.

Finally, the distribution of the average effective move size ($\Delta$) along the simulations is shown. Such distribution is determined by the atoms' coordinates $\mathbf{r}_i$ and $\mathbf{r}''_i$ of all successful moves, that is,

$$\Delta = \frac{1}{\mathcal{N}} \sum \left\{ \frac{1}{m} \sum_{i=1}^{m} |\mathbf{r}_i - \mathbf{r}''_i| \right\}, \quad (7)$$

where the internal sum runs over all $m$ atoms $\{i\}$, and the external one runs over all $N$ complete simulations considered. Fig. 4 shows how $\Delta$-distribution depends on $\delta r_{max}$, as a function of $\eta$ in the interval $3 \leq \eta \leq 8$. The curves collapse at about $\Delta = \delta r_{max}$, where the reversal of their up/down order is observed (as it is necessary for a distribution function). The peak of the $\Delta$-distribution tends to occurs at $\Delta_{max} = \delta r_{max}$; however, mainly for $\eta = 3$ and $\eta = 4$, the number of possible geometric solutions for local moves is small, and so

**TABLE 2 Timing cost and efficiency**

| | $\delta l_{i,j}$ at 0.5% | | $\delta l_{i,j}$ at 1% | |
|---|---|---|---|---|
| $\eta$ | $\tau_L^\eta \pm$ SD* | $\Phi^\dagger$ | $\tau \pm$ SD* | $\Phi^\dagger$ |
| 3 | 12.3 ± 0.4 | 22.2 ± 0.7 | 5.1 ± 0.3 | 14.0 ± 0.4 |
| 4 | 6.4 ± 0.3 | 6.2 ± 0.3 | 2.80 ± 0.08 | 3.1 ± 0.2 |
| 5 | 3.6 ± 0.1 | 1.3 ± 0.1 | 1.74 ± 0.04 | 0.5 ± 0.1 |
| 6 | 2.61 ± 0.05 | 0.2 ± 0.1 | 1.50 ± 0.04 | 0.00 |
| 7 | 2.2 ± 0.1 | 0.00 | 1.37 ± 0.02 | 0.00 |
| 8 | 1.95 ± 0.07 | 0.00 | 1.28 ± 0.02 | 0.00 |

*Average CPU timing cost $\tau_L^\eta$ (ms) spent by the LMProt algorithm to generate $n$ configurations (one MC step) and its corresponding SD (Pentium IV 2.4 GHz; Linux-OS; Intel Fortran, Santa Clara, CA).
†Failure fraction $\Phi$ (%) in adjusting constraints after a total of $I_{max} = 400$ attempts.

$\Delta_{max}$ cannot correspond to $\delta r_{max}$. As shown in Fig. 4, for $\eta = 3$ and $\eta = 4$, $\Delta_{max}$ is almost always lower than $\delta r_{max}$; higher $\delta r_{max}$ increases this difference. For $\eta \geq 5$, $\Delta_{max}$; $\delta r_{max}$, except when $\delta r_{max} = 2.75$ Å, and possibly for greater values, because $\Delta_{max}$ has to be restricted by the main-chain constraints and $\eta$ values.

## RESULTS

### Comparative efficiency tests for phantom chains

Initially, the sampling efficiency of the LMProt algorithm is compared against two recently developed algorithms for generating phantom chain configurations; the native structure of protein 16PK (Bernstein et al., 1998) was used for this purpose (Cahill et al., 2002, 2003). In all cases, the same interactional potential energy based on the *rmsd* (as discussed above) was employed, and a new configuration was always accepted if its corresponding *rmsd* was smaller or equal to the *rmsd* of the previous one. This type of MC simulation where structural information of the study subject is employed in some way, without using physical energy potential, has been called ''reverse MC method'' (McGreevy and Pusztai, 1988).

For this test, 10 independent simulations were carried out with the LMProt and Thrashing algorithms, and their respective evolution compared. All simulations started from an independent random chain presenting large *rmsd*, always larger than 20 Å. Each configuration generated with protein 16PK by the LMProt results from perturbations ($\delta r_{max} = 1.0$ Å) and corrections on all N, CA, and C atoms of $\eta = 6$ consecutive residues. For both algorithms, each MC step generates $2n - n_p$ try configurations, where $n$ is the total number of residues of the chain and $n_p$ is the total number of proline residues (Cahill et al., 2003). Thus, for the Thrashing algorithm, all dihedral angles of the main chain are perturbed in each MC step, resulting in a total of $2n - n_p$ configurations. Each new configuration is produced after a rotation around one of its dihedral angles by a small value $\delta\phi$. For this work, we have used $\delta\phi = 0.0125$ Å, which is the

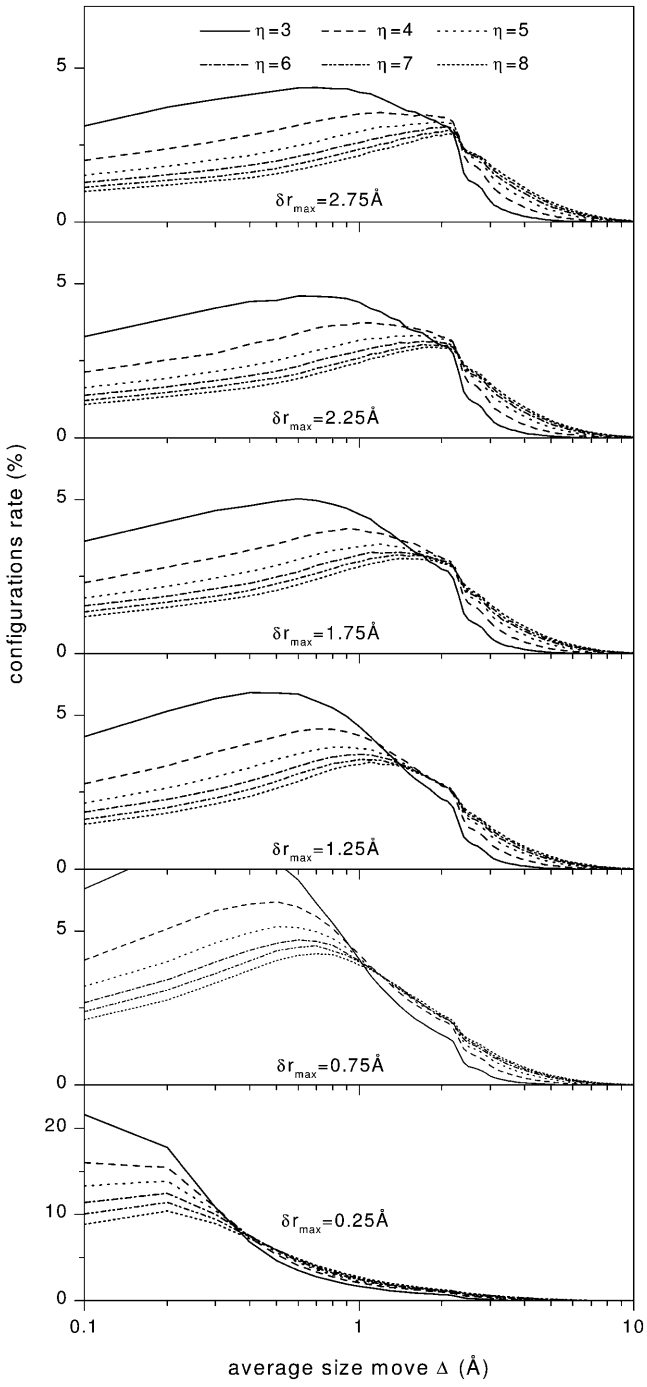
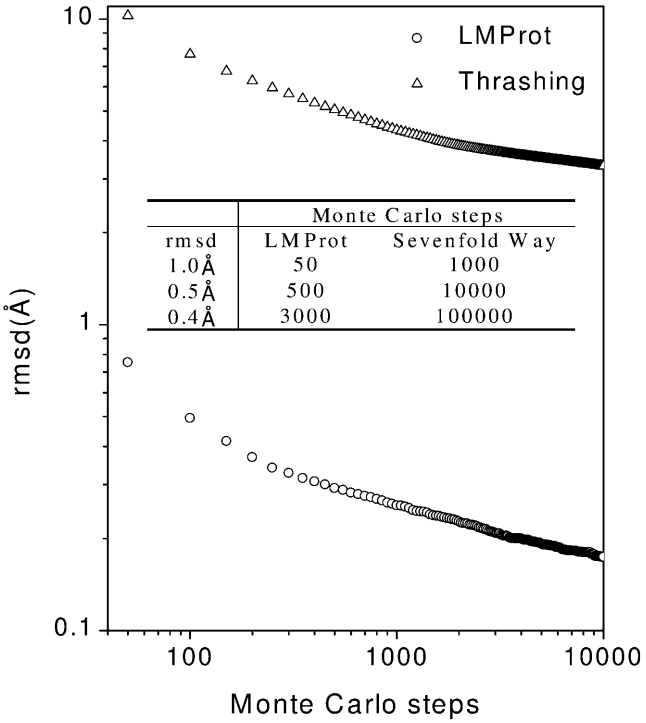



FIGURE 4 Distribution of the average effective move size Δ-distribution for several $\delta r_{\max}$, $\eta$ varying in the interval $3 \leq \eta \leq 8$ (Eq. 7). Here the Δ axis is also shown in log scale for clarity. The curves for $\eta = 8$ present the most regular distribution and a peak near their corresponding $\delta r_{\max}$.




| | Monte Carlo steps | |
|---|---|---|
| rmsd | LMProt | Sevenfold Way |
| 1.0Å | 50 | 1000 |
| 0.5Å | 500 | 10000 |
| 0.4Å | 3000 | 100000 |

FIGURE 5 Evolution of *rmsd* of the $\alpha$-carbons of protein 16PK for the Thrashing algorithm (nonlocal moves of the type pivot) and LMProt (local moves) in a log × log scale. The configurations generated from the Thrashing are produced by the rotation around dihedral angles and the LMProt moving only six consecutive residues ($\eta = 6$ and $\delta r_{\max} = 1.0$ Å). The points are the averages obtained from 10 independent simulations for each 50 MC steps. The *rmsd* × MC steps for Sevenfold Way shown in the inserted table were obtained from Fig. 1 of Cahill et al. (2003).

same value that Cahill et al. used in a recent work (Cahill et al., 2003).

As illustrated in Fig. 5 for phantom chains, the packing evolution differs significantly for each algorithm. Actually, after 50 MC steps, the LMProt method is able to reduce the *rmsd* from ~20 to 0.8 Å, whereas this value is still ~10 Å when Thrashing is employed. For a given level of accuracy, say $\langle rmsd \rangle \sim 3$ Å, this is about three orders of magnitude faster then the Thrashing algorithm. Indeed, for phantom chains, LMProt shows a configurational updating rate that is $\sim 10^3$ times faster than the Thrashing algorithm: it takes $10^4$ MC steps to drive the chain to a conformation exhibiting $rmsd = (3.2 \pm 0.3)$ Å (average over six runs) in the latter case, whereas the same accuracy is reached by the LMProt algorithm only in ~10 MC steps. On the other hand, after $10^4$ MC steps under the LMProt algorithm, the chain reaches a conformation presenting $\langle rmsd \rangle \simeq (0.17 \pm 0.01)$ Å, that is, the native structure is virtually found. The performance of the Thrashing algorithm, however, cannot be improved by simply changing $\delta\phi$. As it has been shown by Cahill et al. (2003), by increasing $\delta\phi$ from 0.0125 to 0.5 Å, $\langle rmsd \rangle$ is increased from 2.4 to 3.7 Å after $10^5$ MC steps.

We also compared the LMProt algorithm against another method, namely the ''Sevenfold Way algorithm'' (Cahill et al., 2003), in which the movements of the chain are also performed by local moves; the same protein 16PK and phantom chain were used here. The same simulation protocol used in the previous simulations of the LMProt and Thrashing algorithms was defined to correspond to the

simulations already executed by the Sevenfold Way of Cahill et al. Thus, the same number of configurations produced for MC step in the cases of the LMProt and Thrashing algorithms is also generated by the Sevenfold Way algorithm. As shown by the inset in Fig. 5 (*small table*), LMProt reaches $\langle rmsd \rangle$ of $\sim$0.5 Å after 500 MC steps, whereas the Sevenfold Way algorithm leads to the same level of accuracy only after $\sim 10^4$ steps MC; that is, LMProt was 20 times faster. If more accuracy is required, the relative performance of LMProt is still comparatively better: for $\langle rmsd \rangle = 0.4$ Å, $\sim 3 \times 10^3$ and $10^5$ MC steps were, respectively, necessary for the LMProt and Sevenfold Way algorithms. We also tested LMProt for another structure presenting mostly tertiary contacts, protein 1AF6 (Wang et al., 1997), with 421 residues, and the results for this case, which also involved phantom chains, were similar to those discussed above for protein 16PK.

## Effect of the excluded volume on the performance of the LMProt and Thrashing algorithms

In this section, a pairwise interaction potential, namely the hard sphere potential, is introduced to analyze the effect of the chain excluded volume on the performance of the LMProt and Thrashing algorithms. Two proteins were considered, namely protein 5NLL (Ludwig et al., 1997) and 1BFF (Kastrup et al., 1997), with 138 and 129 residues, respectively, to determine the efficiency dependency of the LMProt algorithm on the $\eta$ and $\delta r_{\max}$ parameters. The two very distinct tertiary structures were selected to verify how the search of configurations could be affected by the dominant structural topology; protein 5NLL is mainly formed by $\alpha$-helixes and 1BFF by $\beta$-sheets. Now, with the chain-excluded volume considered, a new configuration is accepted only if no superposition of atoms is verified, that is, for each pair of atoms, say atom $i$ and $j$, the distance $r_{ij}$ between them must satisfy the condition $r_{ij} \geq (R_i + R_j)$, being $R_i$ and $R_j$ the radii of the atomic spheres $i$ and $j$ representing the respective atoms. If this condition is satisfied, the corresponding pairwise potential energy $\varepsilon_{ij} = 0$; otherwise $\varepsilon_{ij} \rightarrow \infty$. Thus, the probability $P(x \rightarrow y)$ of transition between two consecutive configurations $x$ and $y$ will be governed by

$$P(x \rightarrow y) = \left\{ \begin{array}{ll} 1 & \text{if} \left[ \begin{array}{c} rmsd_y \leq rmsd_x \\ \text{and} \\ \epsilon_{ij} = 0, \quad \text{for all pairs}(i,j) \end{array} \right] \\ 0 & \text{if} \left[ \begin{array}{c} rmsd_y \leq rmsd_x \\ \text{or} \\ \epsilon_{ij} \rightarrow \infty, \quad \text{for any pairs}(i,j) \end{array} \right] \end{array} \right\}, \quad (8)$$

where $rmsd_x$ and $rmsd_y$ are the root mean standard deviation of configurations $x$ and $y$, respectively, with respect to the coordinates of $\alpha$-carbons of the reference structure. The atomic radius $R_i$ was fixed as $R_i = 1.0$ Å for all chain atoms (hydrogen atoms were not included).

A total of 10 independent simulations were performed for each pair $\eta$ and $\delta r_{\max}$ chosen, all beginning with chains presenting *rmsd* always higher than 20 Å. As described above, each new configuration is generated by perturbing the positions of all atoms of $\eta$ consecutive residues, chosen randomly. The values of $\eta$ and $\delta r_{\max}$ are allowed to vary from 5 to 8 and from 0.25 to 2.75 Å, respectively.

Ideal values for $\eta$ and $\Delta_{\max}$ are considered here as those that maximize the folding success $\xi$ in reaching the native structure, and the folding speed $\xi'$. A particular simulation is considered successful if the chain reaches conformations presenting $rmsd < 1.0$ Å in the time window $t_w = 2.5 \times 10^4$ MC steps. Thus, the parameter $\xi$ measures the ratio between the number of successful simulations and the total number of performed simulations, whereas, in turn, parameter $\xi'$ measures ''how fast'' the folding process is in guiding the chain to conformations near the reference structure (native), by counting those successful runs that reached the native structure in a time $t'_w$ equal to 30% of $t_w$ (that is, $t'_w = 0.75 \times 10^4$ MC steps). The results of this analysis for both proteins are summarized in Fig. 6. Note that to emphasize the combinations of the parameters $\eta$ and $\delta r_{\max}$ that result in larger values of $\xi$ and $\xi'$, the plot reference was fixed at $\xi = \xi' = 0.85$.

In general, for larger $\delta r_{\max}$ and for specific values of $\eta$ (mostly for $\eta \geq 6$), the folding success reaches values $>$90%; especially for protein 5NLL. However, there are specific combinations of $\delta r_{\max}$ and $\eta$ that also result in absolute success: particularly for $\delta r_{\max} = 1.25$ Å, and $\eta$ varying from 6 to 8, the folding success is absolute for both proteins 5NLL and 1BFF.

The dependence of the folding speed $\xi'$ on the parameters $\eta$ and $\delta r_{\max}$ is stronger, but in general the folding speed is favored by using larger $\delta r_{\max}$ as suggested in the graphs from Fig. 6. For both proteins, there is a specific combination $\eta$ and $\delta r_{\max}$ that determines an absolute success (100% of the runs) in the time window $t'_w$, that is, absolute $\xi$ and $\xi'$. For protein 5NLL, this combination is $\delta r_{\max} = 2.25$ Å and $\eta$ varying from 6 to 7; for protein 1BFF, $\delta r_{\max} = 1.75$ Å, and $\eta$ also varies in this same interval. We consider the ideal adjustment for $\eta$ the one that appears more times for both proteins with $\xi = \xi' = 1.0$ and ideal $\delta r_{\max}$ are those where this ideal $\eta$ mostly occurs with $\xi$ and $\xi' \geq 0.9$, that is, $\eta =$ 6–7 and $\delta r_{\max} = 2.25$ Å. The average magnitude of the effective step size $S$ (Eq. 1) obtained in the simulations with these values of $\eta$ and $\delta r_{\max}$ is $\sim 2.86° \pm 0.13°$.

In the simulations executed with the Thrashing algorithm where the effect of the excluded volume was considered in the same way as in the LMProt algorithm, it was not possible to obtain *rmsd* values lower than 6.0 Å for both proteins after $10^5$ MC steps, independent of the values of $\delta\phi$ employed. The retention of the system in particular configurations was observed in all the cases.

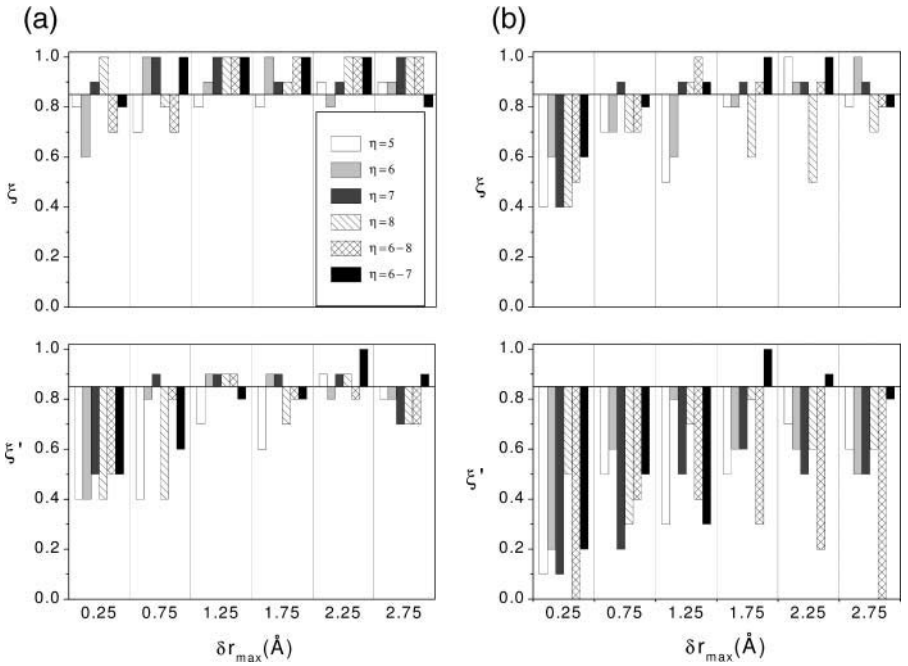



FIGURE 6 Degree of folding success for protein (*a*) 5NLL and (*b*)1BFF as a function of the maximum atomic displacements allowed $\delta r_{\max}$ and of the number of perturbed residues $\eta$ in a local movement. For each combination of $\delta r_{\max}$ and $\eta$, 10 simulations were initiated with the chain in random configurations, of *rmsd* always higher than 20. The values of $\xi$ (*right*) and $\xi'$ (*left*) measure, respectively, the fraction of these simulations that converge in one *rmsd* below 1.0 Å in $2.5 \times 10^4$ and $7.5 \times 10^3$ MC steps. In the figure, the value of $\eta = a - b$ indicates that this can assume random values from $a$ to $b$.

## Comparative CPU times between the LMProt and Thrashing algorithms

In this topic, we consider the calculation details of the LMProt and Thrashing algorithms to establish a general relation between the CPU time involved in each algorithm during one MC step, for a chain with $n$ residues. Let us first consider the Thrashing method. A possible new configuration is generated whenever a specific dihedral angle, located at a particular residue, is perturbed. In this way, because there are two distinct dihedral angles per residue ($\phi$ and $\psi$), with the exception of proline residues, a number of $(2n - n_p)$ configurations is obtained after each MC step, $n_p$ being the number of prolines in the chain. Then, assuming that a real CPU time $\tau_T$ is necessary for each new configuration generated, an average time equivalent to $2n\tau_T$ is spent in each MC step (considering $n_p = 0$). However, a valid new configuration must pass through the atomic overlap checking and for this, let us assume an average number $\overline{n}_r$ of atoms per residue. Therefore, the atomic overlap checking for two distinct residues involves $\overline{n}_r^2$ atom pairs. However, for one perturbed dihedral angle, say, at residue ''$i$'', exactly $i(n - i)$ residue pairs must be checked for overlapping. This implies that a total number of $i(n - i)\ \overline{n}_r^2$ atom pairs are involved. Summing two times over all residues to complete one MC step, that is $2\overline{n}_r^2 \sum_i^n i(n-i)$, one gets that about $N_T = 2\overline{n}_r^2[n(n+1)n/2 - n(n+1)(2n+1)/6] \simeq n^3\,\overline{n}_r^2/3$ atom pairs must be checked. Now, if a real CPU time $\tau_{cpu}$ is necessary for checking a single atom pair, in the case of the Thrashing algorithm a total average time $t_T \simeq 2n[\tau_T + (n^2\,\overline{n}_r^2/6)\tau_{cpu}]$ is spent for each MC step.

Clearly, this time can be significantly reduced by improved methods as the ''residue neighborhood list technique'', that is, a list containing $n_{list}$ neighbors is dynamically updated for each residue. Therefore, for this case, $N'_T \simeq 2\,\overline{n}_r^2\, n_{list}(n/2)(n/2+1)$, assuming that $n$ is even, and so $t'_T \simeq 2n[\tau_T + (nn_{list}\,\overline{n}_r^2/4)\tau_{cpu}]$.

Now, let us turn to the LMProt algorithm. In this case, a candidate for a new configuration is obtained after randomly moving $\eta$ sequential residues. If a CPU time $\tau_L^\eta$ is spent to move all $\overline{n}_r$ atoms of $\eta$ residues, then an average time equivalent to $2n\,\tau_L^\eta$ is spent on each MC step. Again, this try-configuration must pass through the atomic overlap checking, and in this case there are $N_L = \{\eta\overline{n}_r(\eta\overline{n}_r - 1) + 2\eta\overline{n}_r[(n-\eta)\overline{n}_r]\}$ atom pairs to check, each one spending a CPU time equal to $\tau_{cpu}$. Therefore, for the LMProt algorithm, a total average time $t_L \simeq 2n[\tau_L^\eta + n\eta\overline{n}_r^2\tau_{cpu}]$ is spent on each MC step. Otherwise, considering the method of analysis of neighbors, we have $N'_L = 2n\eta n_{list}\,\overline{n}_r^2$ and $t'_L = 2n[\tau_L^\eta + n_{list}\eta\overline{n}_r^2\tau_{cpu}]$. Then, for case B in particular, whose performance for both algorithms is better if compared to case A, the ratio $f = t'_T/t'_L$ between the CPU time of the two algorithms is given by

$$f = \frac{\tau_T + (nn_{list}\,\overline{n}_r^2/4)\tau_{cpu}}{\tau_L^\eta + n_{list}\eta\overline{n}_r^2\tau_{cpu}}, \qquad (9)$$

where $\tau_T$ and $\tau_L^\eta$ are specific for the Thrashing and LMProt algorithms, respectively, but $\tau_{cpu}$ is the same for both. Now, using specific values for $\eta = 3, 4, \ldots 8$ and the respective $\tau_L^\eta$ from Table 2 (tolerance of 1/2 and 1%), $\tau_T = 24\ \mu s$, $\tau_{cpu} = 3.4\ \mu s$ (Pentium IV 2.4 GHz; Linux-OS; Intel Fortran, Santa Clara, CA), and considering $n_{list}\bar{n}_r = 100$ (list of neighbor atoms), one can determine the ratio $f$ for different $n$ values (Eq. 9), as shown in Fig. 7. Note that for $n \gtrsim 34$, the ratio $f$ is

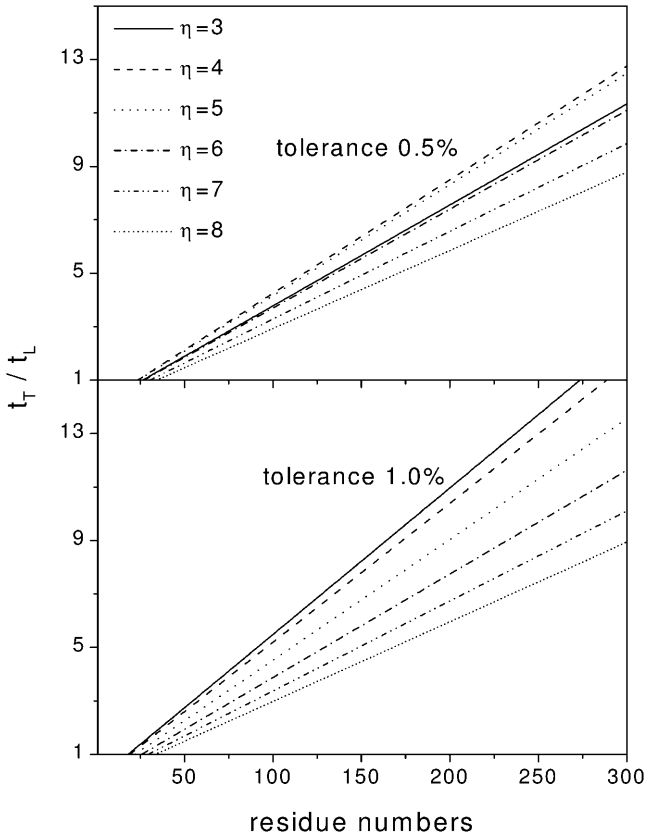



FIGURE 7 Rate of the CPU times in executing overlap check for the Thrashing ($t_T$) and LMProt ($t_L$) algorithms (Eq. 9) as a function of the residues number $n$ using $u = 3.4\ \mu s$, $\overline{a} = 10$, and $c = 100$. The times spent to produce each configuration for the Thrashing ($\tau_T = 24\ \mu s$) and LMProt ($\tau_L$ values of the Table 2) algorithms are also considered in $t_T$ and $t_L$ (Eq. 9).

$>1$ for both tolerances and $f$ is $>10$ for $n \gtrsim 270$; with $\eta \leq 6$. Note that for the same $n$ and $\eta$, $f$ increases when the tolerance $\eta$ is $\leq 6$. When only the overlapping check for protein 1BFF using the previously defined $\eta$ ideal value is considered, for example, the performance of LMProt is about three times higher than that of the Thrashing algorithm.

## DISCUSSION

The results obtained by the LMProt algorithm for multiple and single local moves is compared against other sampling Monte Carlo techniques. As it has been pointed out in previous works (Cahill et al., 2002), our results also emphasize the importance of the way the phase space is explored by simultaneous movements involving chain segments composed by multiple groups. Comparisons between LMProt and other algorithms show that the convergence time (which is proportional to the number of MC steps required to reach conformations the closest possible to the native structure) is significantly shorter for the LMProt method, as shown in Fig. 5. Indeed, it is up to three orders of magnitude faster than the Thrashing (nonlocal) algorithm and, depending on the accuracy required, it is at least 20 times faster than the Sevenfold Way (local) algorithm, as shown by the inset of Fig. 5.

The performance difference between the nonlocal and local algorithms, in the case of phantom chains, can be easily understood. For this, let us consider a specific chain configuration constituted of $N$ atoms; a new configuration is then tried by simultaneously moving $m$ of them. It is straightforward to see that as $m$ increases the chance to obtain a new optimized configuration (lower energy, or smaller *rmsd*) becomes exponentially smaller. It is obvious that, for real chains, extra constraints make the situation even worse. Therefore, the importance of the ''flexible chain idea'' follows, which permits a tolerance on the geometric constraints of the chain, also affecting the reduction of ergodic problems. Indeed, the set of MC moves must be achieved in such a way that any phase space point, $i$, has a chance $p > 0$ to be accessed from any other state, $j$, thus guaranteeing the ergodic hypothesis. In a matrix notation, this hypothesis is expressed as $[A^{k(i,j)}]_{ij} > 0$ for all $i$ and $j$, being $k$ a value that can depend on $i$ and $j$, and $A$ the matrix transition (Manousiouthakis and Deem, 1999). In a Markovian process, the number of required states to pass from $i$ to $j$ can, however, depend on the type of treatment applied. Obviously, if $k$ is a very large number, the search can be considered nonergodic for practical purposes, and so one has to find a way to reduce $k$. At this point is the virtue of the ''flexible-chain idea,'' which effectively permits one to reduce $k$.

A relevant factor that controls the efficiency of the LMProt algorithm is the adjustment of the movement parameters $\eta$ and $\delta r_{max}$. As shown in Fig. 6, the folding success $\xi$ and the folding speed $\xi'$ are significantly affected by the number $\eta$ of residues involved in the try configuration and by the amplitude of the atomic displacements $\delta r_{max}$. Structures like $\alpha$-helix are much less sensitive to these parameters than $\beta$-sheets, as shown by their corresponding convergence rates: protein 5NLL consists mostly of $\alpha$-helixes and 1BFF of $\beta$-sheets. Indeed, the LMProt method is generally faster for $\alpha$-helix than for the $\beta$-sheet type structure, which corroborates with experimental and theoretical results that characterize the folding times for such structural classes (Kauzmann, 1959). Therefore, for proteins presenting larger diversity of folds or tertiary contacts, $\eta$ and $\xi_{max}$ should be properly balanced to optimize the algorithm; the choice, for example, of a too-small amplitude $\delta r_{max}$ (to favor $\alpha$-helix structures; Cahill et. al., 2003), may compromise the configurational search for other structural patterns.

Another important aspect of the LMProt algorithm is the possibility of effectively controlling the magnitude of the atomic displacements at different stages of the folding progress. For instance, after chain collapse, smaller structural changes can be crucial to increase the acceptance rate, contrasting with an open chain, where small displacements can produce a very slow search. However, at any specific instant along the simulation, the chain may present a

nonuniform compactness (even a distinct class of secondary structure may present different compactness). So, instead of having a fixed displacement amplitude, one may think about another kind of amplitude distribution, as it was demonstrated to be useful in permitting the parameter $\eta$ to fluctuate between two limits. These aspects of the algorithm performance become even more significant if different temperatures have to be considered. Of course, the temperature determines the energy levels occupation of the different degrees of freedom of the system, and so one may think that the $\eta$ and $\xi$ parameters should also depend on the temperature. Therefore, it is also easy to understand that the LMProt algorithm is able to describe the physical nature of the systems by mimicking the system dynamics at the molecular level.

An additional advantage of the LMProt is the reduction of the CPU time of the simulations. In MC simulations the main consumption of CPU time is due to checks of overlaps between atoms. LMProt has a CPU time that is about three times shorter than that of the Thrashing algorithm (Fig. 7), for proteins with $\sim$100 residues. If the energy calculation is also considered, this difference can be increased even more.

The generation of configurations by local moves requires changes of variables to recover the geometric constraints of the system after each set of proper moves. Therefore, the determinants of the set of variable transformations (Jacobians) must be calculated for both states involved: the original and the new one, to properly weigh up each new configuration. Such Jacobian depends only on the equations of constraints, and so if $m$ atoms are moved, $3m$ constraints are required (Pant and Theodorou, 1995). However, in generating new configurations, LMProt uses a smaller number of constraints, because the number of solutions is always $>1$ for a local move. Nevertheless, fictitious constraints can be created in a way that the set of equations of constraints results in a unique solution, the new configuration itself already determined by LMProt. In the example of Fig. 1, once a particular solution involving nine atoms is determined, 25 constraints are required. The calculation of the Jacobians, on the other hand, requires 27 constraints for each generated configuration. The choice of two fictitious constraints, in this example, can directly be made between atoms $N_j-N_k$ and $C_j-C_k$ of Fig. 1. In a general way, it was verified that these Jacobians are directly obtained by the LMProt method without significant extra computational cost. In the next work, applications of the LMProt algorithm for MC simulations in which the detailed balance is focused will be presented.

## CONCLUSION

In short, this work presents an efficient MC simulation algorithm to generate protein-like chain configurations. It uses the ''flexible chain idea'' and new configurations are generated by multiple residue moves to mimic a local molecular movement. The parameters that control the moves can be adjusted to optimize the efficiency of the method for MC distinct applications. It was shown that the number of moved residues and the atomic displacement amplitude of the moves, when adequately combined, can speed up the process for finding the native structure. The algorithm presented here is up to several orders of magnitude faster than other recently developed and published algorithms. Its application can be extended to several other problems involving configurational changes ranging from linear polymers to proteins. Some advantages of this algorithm are clearly linked to its ability in reproducing peculiar physical aspects of the macromolecular systems. The availability of a highly efficient algorithm able to generate configurations of proteins is a key point in the protein-folding problem approach by means of general MC technique-based energy criterion.

## SUPPLEMENTARY MATERIAL

An online supplement to this article can be found by visiting BJ Online at http://www.biophysj.org.

We thank Marco Antônio Alves da Silva, from the Department of Physics and Chemistry of Faculdade de Ciências Farmacêuticas de Ribeirão Preto, Universidade de São Paulo, Brazil for the discussions during the development of this work.

We also thank Fundaç ão de Amparo à Pesquisa do Estado de São Paulo (FAPESP, Proc. 01/13970-0) and Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) for financial support.

## REFERENCES

Aièllo, O. E., and M. A. A. da Silva. 2003. New approach to dynamical Monte Carlo methods: application to a epidemic model. *Physica A*. 327:525–534.

Berg, B. A., and T. Neuhaus. 1991. Multicanonical algorithms for 1st order phase-transitions. *Phys. Lett. B*. 267:249–253.

Berne, B. J., and J. E. Straub. 1997. Novel methods of sampling phase space in the simulation of biological systems. *Curr. Opin. Struct. Biol.* 7:181–189.

Bernstein, B. E., D. M. Williams, J. C. Bressi, P. Kuhn, M. H. Gelb, G. M. Blackburn, and W. G. Hol. 1998. A bisubstrate analog induces unexpected conformational changes in phosphoglycerate kinase from trypanosoma brucei. *J. Mol. Biol.* 279:1137–1148.

Binder, K., and A. Baumgartner. 1992. Monte Carlo method in condensed matter physics. *In* Topics in Applied Physics, Vol. 71, 2nd Ed. K. Binder, editor. Springer-Verlag, New York, NY.

Cahill, M., S. Cahill, and K. Cahill. 2002. Proteins wriggle. *Biophys. J.* 82:2665–2670.

Cahill, S., M. Cahill, and K. Cahill. 2003. On the kinematics of protein folding. *J. Comput. Chem.* 24:1364–1370.

Degrève, L., and A. Caliri. 1995. Geometric constraints in polymer chains: analysis on the pearl-necklace model by Monte Carlo simulation. *J. Mol. Struct.* 335:123–127.

de Pablo, J. J., M. Laso, and W. U. Suter. 1992. Simulation of polyethylene above and below the melting-point. *J. Chem. Phys.* 96:2395–2403.

Dodd, L. R., T. D. Boone, and D. N. Theodorou. 1993. The concerted rotation algorithm for atomistic Monte Carlo simulation of polymer melts and glasses. *Mol. Phys.* 78:961–996.

Favrin, G., A. Irbäck, and F. Sjunnesson. 2001. Monte Carlo update for chain molecules: biased Gaussian steps in torsional space. *J. Chem. Phys.* 114:8154–8158.

Go, N., and H. A. Scheraga. 1970. Calculation of conformation of pentapeptide cyclo (glycylglycylglycylprolylprolyl).2. statistical weights. *Macromolecules.* 3:178–187.

Hansmann, U. H. E., and Y. Okamoto. 1993. Prediction of peptide conformation by multicanonical algorithm: new approach to the multiple-minima problem. *J. Comput. Chem.* 14:1333–1338.

Kastrup, J. S., E. S. Eriksson, H. Dalboge, and H. Flodgaard. 1997. X-ray structure of the 154-amino-acid form of recombinant human basic fibroblast growth factor. comparison with the truncated 146-amino-acid form. *Acta Crystallogra. D.* 53:160–168.

Kauzmann, W. 1959. Some factors in the interpretation of protein denaturation. *Adv. Protein Chem.* 14:1–63.

Lal, M. 1969. Monte Carlo computer simulation of chain molecules I. *Mol. Phys.* 17:57–64.

Ludwig, M. L., K. A. Pattridge, A. L. Metzger, M. Dixon, M. Eren, Y. Feng, and R. Swenson. 1997. Control of oxidation-reduction potentials in flavodoxin from Clostridium beijerinckii: the role of conformation changes. *Biochemistry.* 36:1259–1280.

Lyubartsev, A. P., A. A. Martsinovski, S. V. Shevkunov, and P. N. Vorontsov-Velyaminov. 1992. New approach to Monte Carlo calculation of the free-energy-method of expanded ensembles. *J. Chem. Phys.* 96:1776–1783.

Madras, N., and A. D. Sokal. 1988. The pivot algorithm: a highly efficient Monte Carlo method for the self-avoiding walk. *J. Stat. Phys.* 50: 109–186.

Manousiouthakis, V. I., and M. W. Deem. 1999. Strict detailed balance is unnecessary in Monte Carlo simulation. *J. Chem. Phys.* 110:2753–2756.

Marinari, E., and G. Parisi. 1992. Simulated tempering: a new Monte Carlo scheme. *Europhys. Lett.* 19:451–458.

McGreevy, R. L., and L. Pusztai. 1988. Reverse Monte Carlo simulation: a new technique for the determination of disordered structures. *Mol. Simul.* 1:359–367.

Moret, M. A., P. M. Bisch, and K. C. Mundim. 2002. New stochastic strategy to analyze helix folding. *Biophys. J.* 82:1123–1132.

Pant, P. V. K., and D. N. Theodorou. 1995. Variable connectivity method for the atomistic Monte Carlo simulation of polydisperse polymer melts. *Macromolecules.* 28:7224–7234.

Ryckaert, J. P., G. Ciccotti, and H. J. C. Berendsen. 1977. Numerical integration of the Cartesian equations of motion of a system with constraints: molecular dynamics of n-alkanes. *J. Comput. Phys.* 23: 327–341.

Shimada, J., E. L. Kussel, and E. I. Shakhnovich. 2001. The folding thermodynamics and kinetics of crambin using an all-atom Monte Carlo simulation. *J. Mol. Biol.* 308:79–95.

Siepmann, J. I., and D. Frenkel. 1992. Configurational bias Monte Carlo: a new sampling scheme for flexible chains. *Mol. Phys.* 75:59–70.

van Gunsteren, W. F., S. R. Billeter, A. A. Eising, P. H. Hünenberger, P. Krüger, A. E. Mark, W. R. P. Scott, and I. G. Tironi. 1996. Biomolecular Simulation: The GROMOS96 Manual and User Guide. Biomos b. v. Zürich, Groningen, The Netherlands.

Wang, Y. F., R. Dutzler, P. J. Rizkallah, J. P. Rosenbusch, and T. Schirmer. 1997. Channel specificity: structural basis for sugar discrimination and differential flux rates in maltoporin. *J. Mol. Biol.* 272:56–63.

Zhou, R., and B. J. Berne. 1997. Smart walking: a new method for Boltzmann sampling of protein conformations. *J. Chem. Phys.* 107: 9185–9196.