Simulated annealing algorithm for the multiple sequence alignment problem: The approach of polymers in a random medium

M. Hernández-Guía
Henri-Poincaré Group of Complex Systems, Physics Faculty, University of Havana, La Habana, CP 10400, Cuba and National Bioinformatics Center, Industria y San José, Habana Vieja, Capitolio Nacional, CP 10200, Cuba

R. Mulet
Henri-Poincaré Group of Complex Systems and Department of Theoretical Physics, Physics Faculty, University of Havana, La Habana, CP 10400, Cuba

S. Rodríguez-Pérez
Henri-Poincaré Group of Complex Systems, Physics Faculty, University of Havana, La Habana, CP 10400, Cuba and Department of Informatics, University Center of Las Tunas Vladimir Illich Lenin, Las Tunas, CP 75200, Cuba

(Received 13 January 2005; revised manuscript received 21 July 2005; published 27 September 2005)

We propose a probabilistic algorithm to solve the multiple sequence alignment problem. The algorithm is a simulated annealing that exploits the representation of the multiple alignment between D sequences as a directed polymer in D dimensions. Within this representation we can easily track the evolution of the alignment through local moves of low computational cost. In contrast with other probabilistic algorithms proposed to solve this problem, our approach allows the creation and deletion of gaps without extra computational cost. The algorithm was tested by aligning proteins from the kinase family. When $D=3$ the results are consistent with those obtained using a complete algorithm. For $D>3$ where the complete algorithm fails, we show that our algorithm still converges to reasonable alignments. We also study the space of solutions obtained and show that depending on the number of sequences aligned the solutions are organized in different ways, suggesting a possible source of errors for progressive algorithms. Finally, we test our algorithm in artificially generated sequences and prove that it may perform better than progressive algorithms. Moreover, in those cases in which a progressive algorithm works better, its solution may be taken as an initial condition of our algorithm and, again, we obtain alignments with lower scores and more relevant from the biological point of view.

DOI: 10.1103/PhysRevE.72.031915 PACS number(s): 87.10.+e, 87.15.Cc, 87.14.−g

I. INTRODUCTION

The multiple sequence alignment (MSA) problem constitutes one of the fundamental research areas in bioinformatics. While at first sight it may seem a simple extension of the two-string alignment problem *two strings good, four strings better*, for biologists, the multiple alignment of proteins or DNA is crucial in deducing their common properties [1].

To determine a good multiple sequence alignment of D sequences is a relative task. The sequences, in general, consist in a linear array of symbols from an alphabet of k letters ($k=4$ for DNA and $k=20$ for proteins). Usually one defines a score function that depends on the distances between the letters of the alphabet, and assumes that the best alignment is the one that minimizes this score function.

It is common to define the MSA score in terms of the scores of the pairwise global alignments of the sequences (sum of pairs score) [2]. Given two sequences $\vec{a}=a_1,\ldots,a_m$ and $\vec{b}=a_1,\ldots,b_m$ let $\Delta(a,b)$ be the cost of the mutation of a into b and γ the cost of inserting or deleting a letter. Extending $\Delta(a,b)$ so that $\Delta(a,-)=\gamma$ and $\Delta(-,b)=\gamma$ and considering that a null $(-)$ symbol isolated from others $(-)$ pays an extra cost δ [2] we may define the score of a pairwise alignment $M_{i,j}$ for sequences a_i and b_j of size m as

$$s(M_{i,j}) = \sum_{h=1}^{m} \Delta(a_{i,h},b_{j,h}) + n\delta \tag{1}$$

where n is the number of isolated $(-)$. Then, the score for the multiple alignment M is given by

$$E(M_{i,j}) = \sum_{i,j} s(M_{i,j}). \tag{2}$$

The multiple sequence alignment has at least three important applications in biology: classification of protein families and superfamilies, the identification and representation of conserved sequences features of both DNA and proteins that correlate structure and/or function, and the deduction of the evolutionary history of the sequences studied [1,3].

Unfortunately the problem is known to be *NP complete* and no complete algorithm exist to solve real or random instances. Therefore, many heuristic algorithms have been proposed to solve this problem. The algorithm of Carrillo-Lipman [4] (which is complete), is a dynamic programming algorithm able to find the multiple alignment of three sequences. With some heuristics added, it finds the alignments, in a reasonable time, for up to six sequences [1]. However, its computational cost scales very fast with the number of sequences and is of little utility for more ambitious tasks.

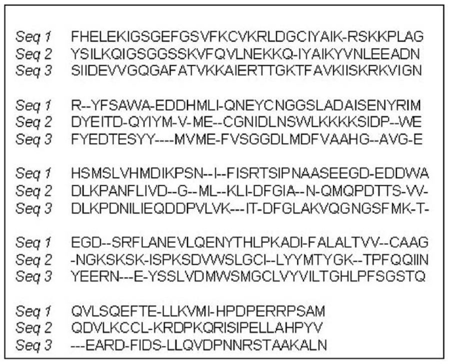
In the early 1990s the problem was approached using ideas coming from physics. Kim and collaborators [5] and Ishikawa and collaborators [6] used different versions of the simulated annealing [7] technique with some success, but their algorithms were unable to change the number of gaps in the alignment. This means that once they started with a given initial configuration (usually taken from some heuristics), any motion of segments in the sequences conserved the number of gaps. To extend these programs allowing the number of gaps to change would causes the appearance of global moves in the algorithm that are very expensive from the computational point of view.

Probably the most successful attempt to solve this problem has been the CLUSTAL project [8], a progressive algorithm that first organizes the sequences according to their distances and then aligns the sequences in a progressive way, starting with the most related ones. Moreover, it uses a lot of biological information, some motifs of residues rarely accept gaps, subsequences of residues associated with structural subunits are preferred to stay together during the alignment, etc. These features, and a platform easy to use and integrated with other standard bioinformatic tools, have made CLUSTAL the favorite multiple sequence alignment program for biologists and people doing bionformatics in general [9]. However, it also has important drawbacks. Once the first *k* sequences are aligned, the inclusion of a new sequence will not change the previous alignment, the gap penalties are the same independently from how many sequences have been already aligned or their properties, and being a progressive method the global minimum obtained is strongly biased by those sequences that are more similar [8].

Recent advances in these kind of algorithms (which includes T-COFFEE [10]) consist in a better use of information obtained *a priori*, mainly by a preprocessing of the sequences using, for example, a local pairwise alignment procedure and combining both global and local information of these pairwise alignments. This approach avoids some of the most important inconvenients of the original version of CLUSTAL, minimizing the errors during the early stage of the alignment. However, while it has been shown that it performs very well in different situations [10], these are algorithms based on heuristic solutions difficult to characterize and therefore difficult to improve by using further algorithmic or biological arguments.

Another recent and also successful approach uses the concepts of hidden Markov models [11]. While the sequences do not need to be organized *a priori*, one must start assuming a known model of protein (or DNA) organization, which is usually obtained after training the program in a subset of sequences. Then one must be aware that the results usually depend on the training set, especially if it is not too large. Moreover, if we are dealing with sequences of unknown family or that are difficult to characterize this approach does not guarantee good alignments.

In 1994 Godzik and Skolnick [12] proposed a simulated annealing (SA) algorithm that exploits the equivalence between the optimal multiple sequence alignment and a multidimensional lattice chain of minimum energy in a random medium [13]. The algorithm we propose in this work is based on the same idea, and therefore it shares the same advantages when compared with previous attempts [5,6]. Both algorithms allow for the insertion and deletion of gaps in a dynamic way using only local moves of low computational cost. The main differences between our algorithm and the one of Godzik and Skolnick consist in the type of local moves allowed and the score function used. Our moves are "smaller" and the score function we used is probably more relevant from the biological point of view. The use of smaller moves reduces the time spent in the calculation of the energy and increases the acceptance rates of the moves. However, it has an important drawback; if the initial conditions are too far from the global minimum, or the system reaches a local minimum, the probability to reach the conformation of minimal global energy is smaller than using large moves. Surely a good strategy should be the combination of both types of moves. Moreover, while Godzik and Skolnick tested their program aligning three sequences we present results for up to 18 sequences, probing by comparison with results from CLUSTAL the feasibility of our approach.



```
Seq 1   FHELEKIGSGEFGSVFKCVKRLDGCIYAIK-RSKKPLAG
Seq 2   YSILKQIGSGGSSKVFQVLNEKKQ-IYAIKYVNLEEADN
Seq 3   SIIDEVVGQGAFATVKKAIERTTGKTFAVKIISKRKVIGN

Seq 1   R--YFSAWA-EDDHMLI-QNEYCNGGSLADAISENYRIM
Seq 2   DYEITD-QYIYM-V-ME--CGNIDLNSWLKKKKSIDP--WE
Seq 3   FYEDTESYY----MVME-FVSGGDLMDFVAAHG--AVG-E

Seq 1   HSMSLVHMDIKPSN--I--FISRTSIPNAASEEGD-EDDWA
Seq 2   DLKPANFLIVD--G--ML--KLI-DFGIA--N-QMQPDTTS-VV-
Seq 3   DLKPDNILIEQDDPVLVK---IT-DFGLAKVQGNGSFMK-T-

Seq 1   EGD--SRFLANEVLQENYTHLPKADI-FALALTVV--CAAG
Seq 2   -NGKSKSK-ISPKSDVWSLGCI--LYYMTYGK--TPFQQIIN
Seq 3   YEERN---E-YSSLVDMWSMGCLVYVILTGHLPFSGSTQ

Seq 1   QVLSQEFTE-LLKVMI-HPDPERRPSAM
Seq 2   QDVLKCCL-KRDPKQRISIPELLAHPYV
Seq 3   ---EARD-FIDS-LLQVDPNNRSTAAKALN
```

FIG. 1. Usual representation of a multiple sequence alignment.

The rest of the paper is organized in the following way. In the next section we make a short review of the theoretical foundations of our algorithm. Then in Sec. III we explain the implementation details, and the results are discussed in Sec. IV. Finally the conclusions are presented including an outlook for future improvements of this program.

II. THEORETICAL BACKGROUND

Usually, multiple sequence alignments are studied and visualized by writing one sequence on top of the other, miming a table (see Fig. 1) and all the probabilistic algorithms devised so far use the simplicity of this representation to generate the moves.

Instead of that, we will use the well known fact [2,3] that the alignment of D sequences may be represented in a D-dimensional lattice (see Fig. 2 for $D=2$).

The cells of the D-dimensional lattice are labeled by D indices $(i_1, i_1, \ldots, i_D)$. The bonds encode the adjacency of letters: A diagonal bond in a D-dimensional space represents the D pairing $(a_{i_1}, b_{i_2}, \ldots, w_{i_D})$. The insertion of gaps is represented by bonds without components in the sequences where the gaps were inserted. For example, a D pairing $(a_{i_1}, b_{i_2}, -, d_{i_4}, \ldots, w_{i_D})$ is represented by a bond whose pro-

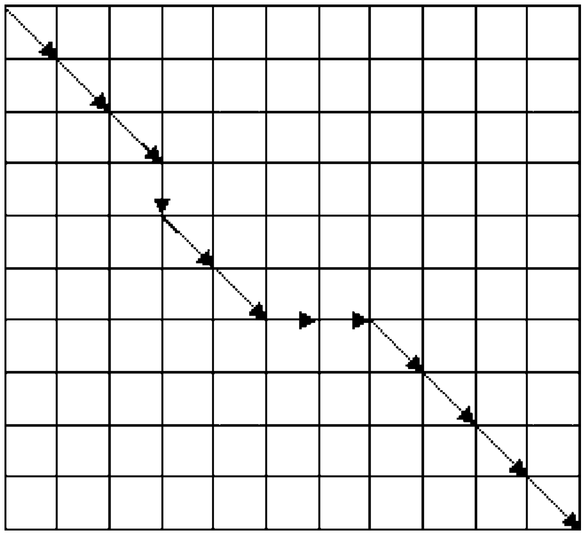


FIG. 2. A directed path (thick line) in a bidimensional grid.

jection on the third sequence is zero, and the D pairing $(a_{i_1},-,-,d_{i_4},\ldots,w_{i_D})$ is represented by a bond whose projections on the second and third sequences are zero.

Then, any alignment maps onto a lattice path that is directed along the diagonal of the D-dimensional hypercube. This lattice path may be interpreted as a directed polymer and the random medium in the problem is provided by the structure of the sequences to be aligned and by the distance between the residues in the different sequences.

This mapping was already fruitfully used by Hwa and collaborators [13] to prove that the similarities between two sequences can be detected only if their amount exceeds a threshold value and to propose a dynamic way to determine the optimal parameters for a good alignment of two sequences.

Here, our main focus will be to optimize the directed polymer (lattice path) under the constraints imposed by the sequences and their interactions in dimensions larger than 2. To use a simulated annealing algorithm we extend the usual representation of the ground state of the problem to finite temperatures. Then, a finite-temperature alignment is defined as a probability distribution

$$P(C) = \frac{1}{Z}e^{-\beta E(C)} \tag{3}$$

over all possible conformations C of the polymer, where E is given by Eq. (2), and Z is the partition function of the alignment [14]. The temperature (β^{-1}) controls the relative weight of alignments with different scores (different conformations of the polymer) while $\Delta(a,b)$ controls the length of the polymer and the frequency of the gaps. In physical terms, $P(C)$ defines an ensemble at temperature β^{-1} with line tension γ and chemical potential $\Delta(a,b)$ [15].

This kind of algorithm, together with the already discussed advantages when compared with other SA algorithms, permits us to easily visualize and quantify the localization length [13]. This is a quantity that measures the distance between the correct alignment and the optimal alignment. By correct alignment we mean the alignment that characterizes the real evolutionary relation between the sequences. These alignments, when mapped as lattice chains, describe two polymers that will overlap more or less depending on the quality of the algorithm performance. To have this kind of representation may give some insight on which are the regions of alignment worse and/or better represented by the global minimum. This in turn may give clues about the type of moves to be used, the best scheduling procedure, etc.

III. THE ALGORITHM

Simulated annealing was introduced many years ago by Kirkpatrick *et al.* [7] to find a global minimum of a function in combinatorial optimization problems. SA is a probabilistic approach, which in general needs a state space, the different configurations of the directed path, and a cost or energy function to be minimized [Eq. (2)].

Simulated annealing generates new alignments from a current alignment by applying transition rules of acceptance. The criteria for acceptance are the following: (a) if $\Delta E < 0$, accept the new alignment; (b) if $\Delta E > 0$ accept it with probability $P(\Delta E) = e^{-\beta[E_{\text{new}}(C)-E_{\text{old}}(C)]}$.

The parameter β controls the probability to accept a new configuration. Initially, one starts at low values of β (high temperatures) and then increases it, applying an annealing schedule. If the temperature is lowered slowly enough, it can be proved that the system reaches a global minimum [16]. Unfortunately it will require infinite computational time and one usually selects the best schedule to match the computational facilities at hand. Then SA is run over many initial conditions, and one assumes that the output of minimum energy is (or is close to) the global minimum.

In multiple sequence alignment this is also the case, but differently to what happens in other combinatorial optimization problems, here the average solution, i.e., that obtained after averaging over all the local minima, may be interesting by itself. In fact, researchers are often interested not in the particular details of the alignment, but in its robust properties, and to compare all the outputs of the SA is a way to get this information.

From the technical point of view, once a cost function is defined, one needs to select the moves to be associated with the transition rates. Our description of the multiple sequence alignment problem as a directed polymer in a random medium allows the definition of three types of moves: insertion, deletion, and motion of gaps. All these moves are represented in Fig. 3 in a two-dimensional grid. The extension to D-dimensional systems is straightforward.

In this way we get an algorithm that allows the creation of gaps, which means a search space larger than that usually probed by similar methods. At the same time the algorithm is quadratic in the number of sequences. The computational cost of any move is limited by the square of the number of sequences to be aligned.

In this work, we do not follow any heuristic strategy of optimization. Our intention is to prove the potential of this strategy and we kept things as simple as possible. For example, if we start too far from the global minimum, the

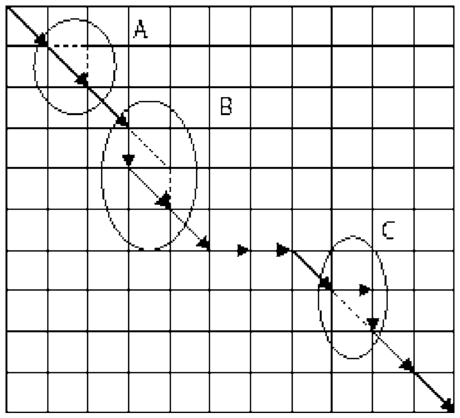


FIG. 3. Local moves of the algorithm in a two-dimensional grid (from arrows to dashed lines): (a) gap insertion, (b) gap motion, and (c) gap deletion.

selection of local moves alone will make the algorithm to converge very slowly to it. This drawback may be overcome using very different initial conditions or trying, every few time steps, global moves that change radically the conformation of the polymer. We did not take care of this. During the simulation the three moves were chosen with probability 1/3. The only biological information inserted was given by the cost matrix used to align the protein sequences. We avoid the use of important and well known biological information, fixed residues, phylogenetic tree of the sequences, etc., and the program was designed without the use of optimization tricks.

IV. RESULTS

A. Relaxation

All the results presented in this section reflect the alignments of proteins from the kinase family, but qualitatively similar results were obtained for the GPCR (G protein-coupled receptors) and CRP (cAMP receptor protein) families. The simulations started with $\beta=1.0$ and every $t_{\rm rel}$ Monte Carlo steps β was increased by a factor of 1.01 until $\beta=3.0$. We made sure that in all cases the system reached the equilibrium. The different initial conditions were chosen by inserting gaps randomly in all the sequences but the largest one, such that considering these gaps at $t=0$, all the sequences were of equal length. To define the distance between the letters of the alphabet we use the PAM-250 matrix [11].

In Fig. 4 it is shown the approach to equilibrium of the multiple sequence alignment of three sequences averaged over 100 initial conditions for $t_{\rm rel}=10,100$, 1000, and 10 000. Comparing with the result of the Carrillo-Lipman algorithm it is evident that for $t_{\rm rel}=10\,000$ the algorithm is very close to the global minimum. However, one should notice that, unlike what is usually obtained in other algorithms for the multiple sequence alignment problem, the figure reflects average values over the multiple alignment.

One is often interested, rather than in the average, in the alignment of minimum energy. In Fig. 5 we present, for different values of $t_{\rm rel}$, a histogram of the energies for the alignments obtained using 1000 initial conditions. We used the same three proteins of the kinase family shown in Fig. 4. Note again that for $t_{\rm rel}=1000$ we obtain exactly the Carillo-Lipman result with nonzero probability. Moreover, looking at the structure of the histogram for $t_{\rm rel}=1000$, one can also conjecture that if the average multiple sequence alignment is calculated only with those alignments concentrated in the peak of lower energies, the result presented in Fig. 4 will be closer to that of the Carrillo-Lipman algorithm.

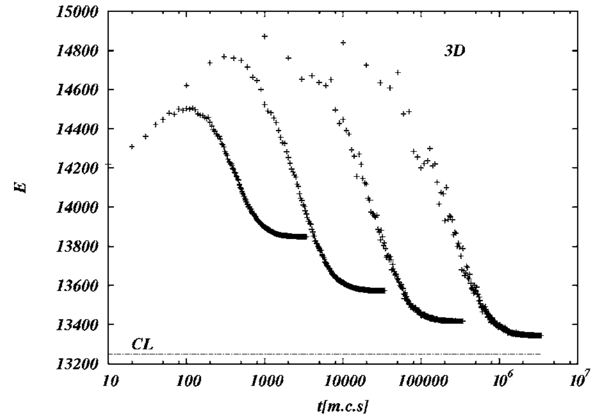


FIG. 4. Mean energy versus time for the alignment of three sequences from the kinase family. From left to right $t_{\rm rel}=10,100$, 1000, and 10 000. The averages were taken over 100 initial conditions. The horizontal line represents the result of the Carrillo-Lipman algorithm

Another symptom suggesting that the average over the realization must be taken with care comes from the analysis of Fig. 6. There we present again histograms for $t_{\rm rel}=1000$ but using three different samples of the kinase family. Note that while sample 1 and sample 3 are very well behaved and the results compare very well with the Carillo-Lipman method, the situation for sample 2 is different. To get good results in this case, it is clearly necessary to go beyond 1000 Monte Carlo steps.

In the same spirit of Fig. 4, Figs. 7 and 8 show results suggesting that also for higher dimensions, if $t_{\rm rel}$ is large enough the algorithm should produce good alignments. In

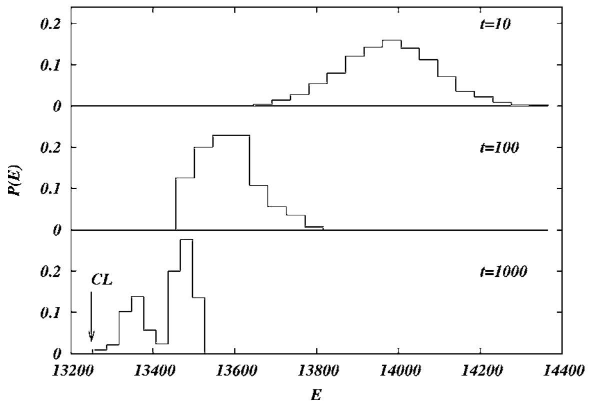


FIG. 5. Energy histograms for the alignment of three sequences from the kinase family at different $t_{\rm rel}$. The Carrillo-Lipman result is indicated by the arrow.

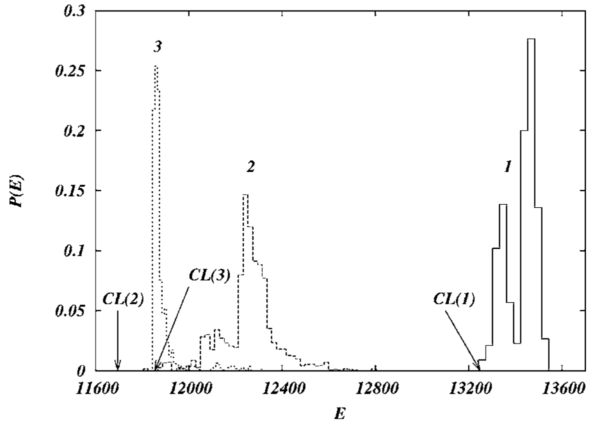


FIG. 6. Histograms of energy for the alignment of three different samples of three sequences from the kinase family at $t_{\rm rel}=1000$. The Carrillo-Lipman results are indicated by arrows.

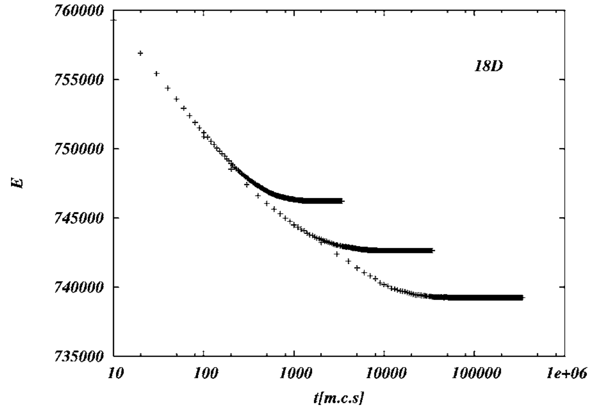


FIG. 8. Mean energy versus time for the alignment of 18 sequences from the kinase family. From left to right $t_{\rm rel}=10$, 100, and 1000. The averages were taken over 100 initial conditions.

fact, also in these cases the energy decreases linearly with $t_{\rm rel}$.

If the number of sequences is higher, the correlations between the sequences increase, and the algorithm should find better results. This fact may be clearly seen in Fig. 9 where the equilibration time of the algorithm [in Monte Carlo Steps (MCS)] is shown as a function of the number of sequences. The equilibration time, measured as the time necessary to reduce the energy by a factor e, decreases linearly when the number of sequences to be aligned increases. Of course the results may change if very different sequences are aligned, but in this case all other known algorithms fail to predict good alignments. Then, we may say that for the most common cases of correlated sequences, we present an algorithm whose convergence time decreases with increasing D, and whose moves only increase quadratically with D.

B. Space of solutions

With these results at hand, we go on to study the structure of the space of the solutions as a function of the number of aligned sequences. We define a distance (d) between two alignments $\mathcal{A}$ and $\mathcal{B}$ in the following way. Given two solutions $A_{i,j}$ and $B_{i,j}$ (where the index i stands for the sequence and j for the position of the symbol in the sequence) we aligned one by one the D sequences of each solution using a dynamic programming algorithm reminiscent of the Needleman-Wunsh algorithm with the following score function:

$$c_a(A,B)=\begin{cases}0 & \text{if } A_{i,j}=B_{i,j},\\ 1 & \text{if } A_{i,j}\neq B_{i,j},\\ r>1 & \text{if a gap is inserted,}\end{cases}$$

and express $d_{\mathcal{A},\mathcal{B}}$ as

$$d_{\mathcal{A},\mathcal{B}}=\frac{1}{D}\sum_{i,j}c_a(A_{i,j},B_{i,j}). \qquad (4)$$

In this way identical alignments will be at distance 0 from each other, and the insertion of gaps to obtain good alignments is penalized, such that the original alignments are altered minimally during the calculation of $d_{\mathcal{A},\mathcal{B}}$. We calculated d with $r=2$ and 8 and the results were the same (apart from a constant shift in d). Below we present the results for $r=2$.

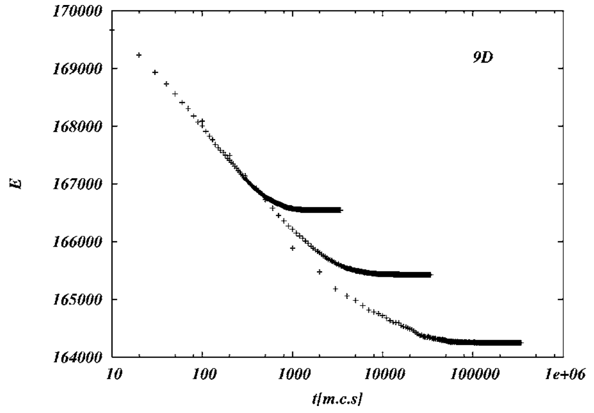


FIG. 7. Mean energy versus time for the alignment of nine sequences from the kinase family. From left to right $t_{\rm rel}=10,100$, and 1000. The averages were taken over 100 initial conditions.

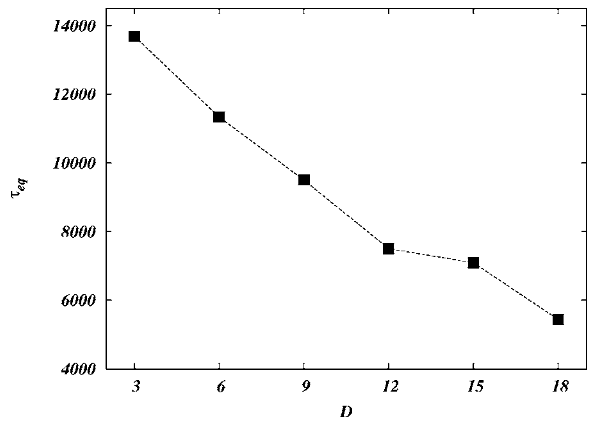


FIG. 9. Equilibration time versus D. The average was done over 500 initial conditions for $t_{\rm rel}=1000$.

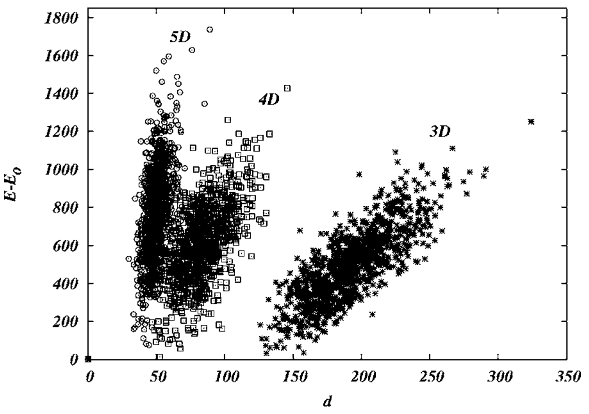


FIG. 10. $E-E_0$ versus d for the alignment of three, four, and five sequences of the kinase family.

We study how different from the solution of minimum score are the other solutions obtained aligning D sequences. For every value of D we use 1000 initial conditions and $t_{\mathrm{rel}}=1000$. Note that the sequences used were always the same. This means, we first aligned three sequences from the kinase family. Then, to align four sequences we just added a new one to the previous three and started the alignment from scratch. The procedure was repeated for every new value of D.

The results appear in Figs. 10 and 11 where $E-E_{\mathrm{min}}$ is plotted as a function of d. From the figures it becomes evident that the space of solutions strongly depends on the number of sequences aligned. For example, while for three sequences the distance between the alignments is correlated with the difference in score, for more than four sequences it is not the case. Moreover, while for $D<6$ the distance between the solutions decreases, it increases for $D>6$ and remains constant for $D>12$. Surely, these results reflect the correlation between the proteins aligned. For example, we may speculate that for $3<D<6$, any new protein added contributed to finding more similar alignments, i.e., added relevant information to the system. However, for $D>6$ the sequences contributed with new and uncorrelated information that produces more distant alignments, and for $D>12$ adding new proteins does not change the relevant characteristics of the alignment.

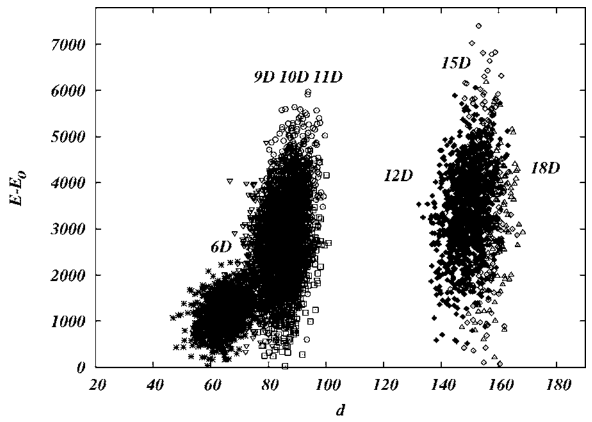


FIG. 11. $E-E_0$ versus d for the alignment of 6, 9, 10, 11, 12, 15, and 18 sequences of the kinase family.

Considering the limitations of many progressive algorithms to change the previous alignment when new sequences are added these results appear particularly relevant. Figures 10 and 11 clearly suggest that the inclusion of one single sequence may dramatically change the character of the solutions.

C. Evolutionary dynamics

It is quite difficult to unambiguously define what is a better sequence alignment; it often depends on *a posteriori* processing, the comparison with structural information, the motif conservation, or even the criterion of an expert. This is particularly relevant when comparing unknown or very divergent sequences. Therefore, to test our algorithm, we decided to develop two kinds of evolutionary processes. Starting from a real sequence, we generate new (now artificial) sequences but keeping a trace of the polymer created by the evolutionary processes. In this way we are able to compare the distance between the alignments generated by our simulated annealing and CLUSTALW with the "real" evolutionary relation between the sequences.

Evolutionary process I. Starting from a real sequence taken from the kinase family, we generated k new sequences using the following procedure. We move sequentially through a copy of the original sequence and at each position i of the sequence we perform, with probability, 0.25 one of the following actions: (1) Substitute the aminoacid at the position i by a new aminoacid with probability p; (2) delete the aminoacid at position i with probability p; (3) Insert a new aminoacid at position i with probability p; (4) do nothing.

The deletion of aminoacids in the new sequence is equivalent (keeping the alignment) to the insertion of a gap. For the same reason the insertion of a new aminoacid is equivalent to introduce a gap, at the same position in the original sequence.

Once the last aminoacid is reached, we already have two sequences and the correct alignment that reproduces the evolutionary relation between them. To obtain more sequences we follow a similar procedure, starting from the (now modified) original sequence we repeat the steps described above but every time an aminoacid was inserted in the new copy, we insert a gap at the same position in all the already generated sequences.

The aminoacid to be inserted in step 3 was selected randomly from the 20 known aminoacids. The substitution was done trying to reflect the structure of the matrix PAM-250 (which is used latter by our SA and by CLUSTALW to find the best alignment). Roughly speaking the PAM matrix represents the probabilities that one aminoacid is substituted by the other 19. For each aminoacid we choose the set of aminoacids with higher probabilities of substitution and each time step 1 above was executed, one of them was randomly selected. In this way, we may obtain D evolutionary related sequences, all descending from the same real sequence, and the alignment or (equivalently) the chain path that correctly describes this process.

The next step was to test the capability of our algorithm to reproduce this correct alignment starting from random initial

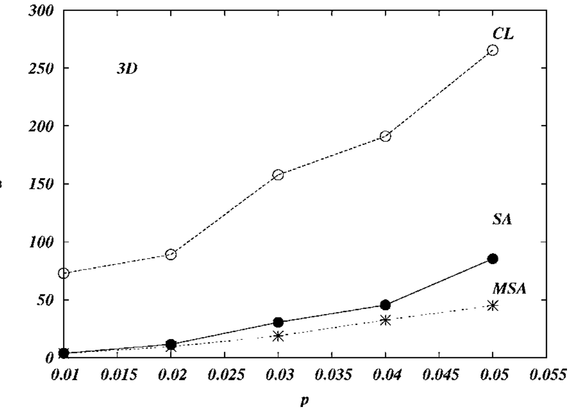


FIG. 12. Distances between the best solution of our simulated annealing, the Carrillo-Lipman algorithm and CLUSTALW with the correct alignment for three sequences generated through the evolutionary process I.

conditions. To do this we run our program and calculate the distance [see Eq. (4)] between the best alignment generated by the simulated annealing and the correct alignment. For comparison we show in Figs. 12 and 13 the plot of these distances for different values of p together with the distances obtained using the output of CLUSTALW and MSA. From now on, all the figures obtained are the results of runs starting with $\beta=0.1$ at multiplicative steps of 1.01 and relaxing at each β for 100 MCS. These are very fast runs; in fact most of the computer time was spent doing the statistics (on 20 generated set of sequences for each D) and calculating the distances.

As the figures clearly show, for fewer than nine sequences the larger the value of p (the more divergent are the sequences), the better is the result of the simulated annealing compared with CLUSTALW. For nine or more sequences CLUSTALW performs better.

A more realistic evolutionary process should include a parameter τ that quantifies the temporal evolution of the sequences generated. Therefore we made a similar study but generating sequences in a different way.

Evolutionary process II. Starting from a real sequence we

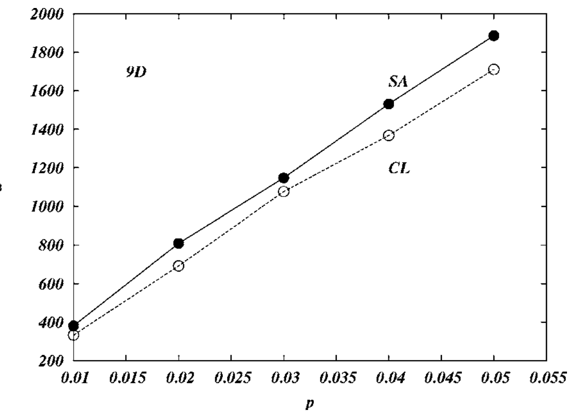


FIG. 13. Distances between the best solution of our simulated annealing and CLUSTALW with the correct alignment for nine sequences generated through the evolutionary process I.

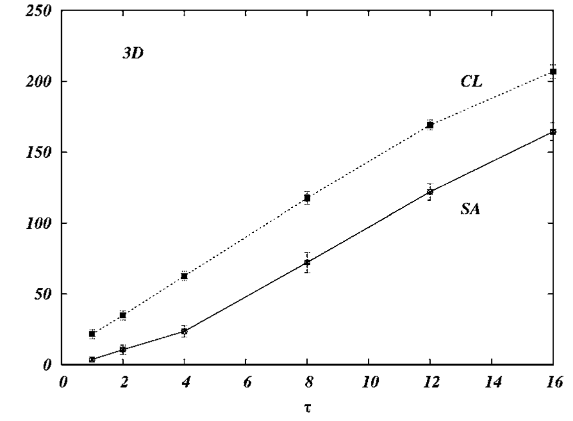


FIG. 14. Distances between the best solution of our simulated annealing, the Carrillo-Lipman algorithm and CLUSTAL with the correct alignment for three sequences generated through the evolutionary process II.

generate D copies of this sequence. Then at each iteration step we move sequentially through all the sequences at the same time and we perform, with probability 0.25, in each sequence, at each position, one of the following actions: (1) Substitute the aminoacid at position i by a new aminoacid with probability $p=0.001$; (2) delete the aminoacid at position i with probability $p=0.001$; (3) insert a new aminoacid at position i with probability $p=0.001$; (4) do nothing.

If an insertion action is taken in $D-k$ sequences, we introduce a gap, at the same position, in the k remaining sequences, independently from the action previously performed on them. A time step is defined after we reach the end of the sequences. Then, starting from these (now modified) sequences, we repeat the procedure τ times, obtaining in this way the correct alignment between D sequences that diverged in time, at the same rate, from an original one.

The main difference with the previous approach is the relevance of the time. While in the first evolutionary process it was hidden in the mutation rate p, now the mutation rate is fixed, and the time is explicitly taken into account. As before the aminoacid to be inserted in step 3 was selected randomly between the 20 known aminoacids and the substitution was done trying to reflect the structure of the PAM matrix.

Calculating again the distance between the best solution of the SA and the correct alignment derived using the evolutionary process II we get, as Figs. 14 and 15 show, results qualitatively similar to those obtained using the evolutionary process I. Again, our simulated annealing performs better than CLUSTAL for a small number of sequences, specially when the sequences are very different.

It may well be that a critical dimension exists above which the properties of the polymer are difficult to reproduce by simple minimization procedures like SA. Another explanation may be that since the sequences compared are statistically correlated, increasing its number the progressive alignment has more information to improve its performance and therefore works better at higher dimensions. In any case, the results presented above clearly proves that the simulated annealing may perform better than CLUSTAL.

Moreover, even in those cases in which CLUSTAL behaves

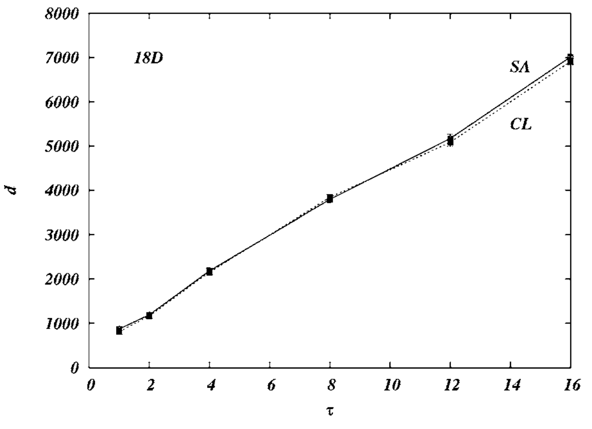


FIG. 15. Distances between the best solution of our simulated annealing and CLUSTAL with the correct alignment for 18 sequences generated through the evolutionary process II.

better than SA, its solution may be used by our algorithm for further optimization. We tested the performance of our algorithm using the output of the CLUSTAL as an input of our simulated annealing. Then we calculated the distance between the solution of the CLUSTALW and the correct alignment (d_c), and the distance between the best solution of the SA (starting from the output of CLUSTALW) and the correct alignment d^c_{SA}. In this case, as shown in Fig. 16 the simulated annealing again outperforms CLUSTAL, especially if the sequences are very distant from the evolutionary point of view. Therefore, we may conclude that even in those cases in which CLUSTALW works better our SA may be used as a further method of optimization.

V. CONCLUSIONS AND OUTLOOK

In this work we presented a probabilistic algorithm to perform the multiple alignment of proteins. The algorithm is based on the mapping between the DPRM and the multiple sequence alignment problem. In contrast with other probabilistic algorithms our algorithm permits the variation of the number of gaps in the alignment without the necessity of expensive global moves. It is proved that for small number of sequences it reproduces the results of a complete algorithm. Moreover, we show that for practical purposes the equilibration time is almost independent of the number of sequences aligned, D, and in the worst case, it scales linearly with D. We studied the space of solutions for different numbers of aligned sequences, and find a very rich structure that indicates the importance of just one sequence in the multiple alignment. Furthermore, we showed that our algorithm may perform better than CLUSTAL especially if the sequences are very divergent and if they are not too many. Finally, in those cases in which CLUSTAL performs better than our algorithm, to use the SA with the CLUSTAL solution as initial condition gives space for further optimization.

We are already working in implementing a similar work but using parallel tempering instead of simulated annealing. It is known that parallel tempering is more useful than SA when dealing with very hard problems, like spin glasses. Moreover, it is very suitable to parallelization. Also a direction of current work is the introduction of biological information relevant to the alignment. This may impose important constraints in the possible alignments, that may in turn strongly reduce the space of possible solutions. And last, but not least, important programming optimizations are in progress to make competitive this program from the computing time point of view.

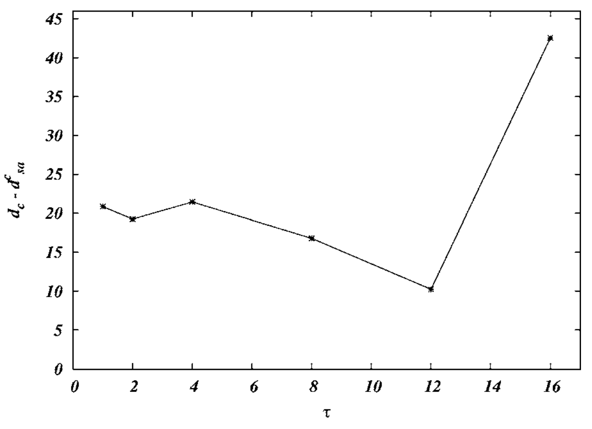


FIG. 16. Difference in the distances between the solution of CLUSTAL and the best solution of our simulated annealing using CLUSTAL solution as an input with the correct alignment for 18 sequences generated through the evolutionary process II.

ACKNOWLEDGMENTS

R.M. wishes to thank K. Leon and E. Moreno for useful discussion at the early stage of this work. We acknowledge the support of NET-61 from ICTP. We acknowledge E. Altshuler and C. Noda for comments on the manuscript.

APPENDIX

In this appendix we present a portion of the alignment obtained by Clustal and by our algorithm of three sequences artificially generated using the evolutionary process II. For comparison, also the correct alignment is shown.

1. Clustal output

```
E=13116.0
Seq 1+ -----FHELWKIGSGEFGW-FKCVKRLW
Seq 2+ FHEEKGSWEFGGSVFCCVKLRLDGCIMY
Seq 3+ -----FHELEKIGSGEFGSVFCCVKCLD
Seq 1+ GCIYAIKKKPLAGSVDEQNALREVYAHV
Seq 2+ AIKRSGKKPLAGSVWDEQNWLREVYAHA
Seq 3+ GCIYAIKRSKKPLAGSVDEALREVYAHA
Seq 1+ LGGQHFHVVRYDSAWAEDDHMLIPHQNE
Seq 2+ VLGQHSHVVRYFNSAWADWHMLI--QNE
Seq 3+ VLGQHSHVVRYFSAWAEDDMLI---QNE
```

2. Simulated annealing output

```
E=11664.0
Seq 1+ FHELWKIGSGEFG--WFKCVK-RLWGCI
Seq 2+ FHE--EKGSWEFGGSVFCCVKLRLDGCI
Seq 3+ FHELEKIGSGEFG-SVFCCVK-CLDGCI
Seq 1+ -YAIK---KKPLAGSV-DEQNALREVYA
Seq 2+ MYAIKRSGKKPLAGSVWDEQNWLREVYA
Seq 3+ -YAIKRS-KKPLAGSV-DEA--LREVYA
Seq 1+ HVLGGQHFHVVRYD-SAWAEDDHMLIPH
Seq 2+ HAVLGQHSHVVRYFNSAWA-DWHMLI--
Seq 3+ HAVLGQHSHVVRYF-SAWA-EDDMLI--
```

3. Correct alignment

```
Seq 1+ FHELWKIGSGEF-G-WFKCVK-RLWGCI
Seq 2+ FHE-EK-GSWEFGGSVFCCVKLRLDGCI
Seq 3+ FHELEKIGSGEF-GSVFCCVK-CLDGCI
Seq 1+ -YAIK---KKPLAGSV-DEQNALREVYA
Seq 2+ MYAIKRSGKKPLAGSVWDEQNWLREVYA
Seq 3+ -YAIKRS-KKPLAGSV-DE--ALREVYA
Seq 1+ H-VLGGQHFHVVRYD-SAWAEDDHMLIP
Seq 2+ HAVLG-QHSHVVRYFNSAWA-DWHMLI-
Seq 3+ HAVLG-QHSHVVRYF-SAWAEDD-MLI-
```

[1] D. Gunsfield, *Algorithms on Strings, Trees and Sequences* (Cambridge University Press, Cambridge, U.K., 1999).
[2] M. S. Waterman, *Introduction to Computational Biology* (Chapman & Hall, London, 2000).
[3] P. Pevzner, *Computational Molecular Biology* (MIT Press, Cambridge, MA, 2000).
[4] H. Carrillo and D. Lipman, SIAM J. Appl. Math. **48**, 1073, (1988).
[5] J. Kim, S. Pramanik and M. J. Chung, CABIOS, Comput. Appl. Biosci. **10**, 419 (1994).
[6] M. Ishikawa, T. Toya, M. Hoshida, K. Nitta, A. Ogiwara and M. Kanehisa, CABIOS, Comput. Appl. Biosci. **9**, 267 (1993).
[7] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, Science **220**, 671 (1983).
[8] J. D. Thompson, D. G. Higgings and T. J. Gibson, Nucleic Acids Res. **22**, 4673 (1994).
[9] We are not aware of any statistical study proving this, but we believe that this statment is well accepted among the colleagues we contacted.
[10] C. Notredame, D. G. Higgins, and J. Heringa, J. Mol. Biol. **302**, 205 (2000).
[11] R. Durbin, S. R. Eddy, A. Krogh, and G. Mitchison, *Biological Sequence Analysis* (Cambridge University Press, Cambridge, U.K., 1998).
[12] A. Godzik and J. Skolnick, CABIOS, Comput. Appl. Biosci. **10**, 587 (1994).
[13] T. Hwa and Michael Lassig, Phys. Rev. Lett. **76**, 2591 (1996).
[14] M. Q. Zhang and T. G. Marr, J. Theor. Biol. **174**, 119 (1995).
[15] M. Kschischo and M. Lässig (unpublished).
[16] S. Geman, IEEE Trans. Pattern Anal. Mach. Intell. **6**, 721 (1984).