A fast parallel algorithm to reduce protein folding trajectories

Luis Garreta[1], Mauricio Martinez[2], Néstor A Díaz[3] and Pedro A Moreno[4*]

Abstract

Background: Simulations are among the most important tools for studying the mechanisms underlying protein folding. Protein folding simulations have experienced substantial progress in recent years. Diverse technologies are employed, and simulations are reaching the microsecond timescale and greater, which generate very long trajectories. However, the analysis of these trajectories is complicated, and tools are necessary to simplify them so that both the main events and the temporal order in which they occur are preserved.

Results: We present an algorithm to reduce long protein-folding trajectories in a fast and parallel way. The algorithm divides a trajectory into segments to be processed in parallel and then selects from each segment the most representative conformations using a rapid clustering strategy. This strategy leverages the temporal order of the conformations to compare them locally, avoiding an all-versus-all comparison. The algorithm reduces a trajectory by a high percentage, preserving both the patterns and the structure obtained by other, more complex reduction techniques. In addition, its performance is close to that of other efficient reduction techniques and is improved when executed in parallel using more than one core.

Conclusions: The developed algorithm quickly reduces a protein folding trajectory by selecting its most representative conformations and preserves both the structure and temporal order of the trajectory. The reduced trajectories can be used as input for more complex analysis techniques and for other reduction techniques that are impractical for use with long folding trajectories. The algorithm is fast and is designed to run in parallel on conventional PCs with multicore technology, which are present in most typical research laboratories.

Keywords: Protein folding simulations; Protein structure comparison; Protein structure clustering

* Correspondence: pedro.moreno@correounivalle.edu.co
[4] Escuela de Ingeniería de Sistemas y Computación, Universidad Valle, Santiago de Cali, Colombia
Full list of author information is available at the end of the article.
† Equal contributor

Background

We present a parallel algorithm to reduce protein folding trajectories that quickly obtains representative conformations, conserving both the three-dimensional (3D) structure and temporal order of the trajectories. Proteins play fundamental roles in all living organisms, but to be functional, they must fold from their linear amino acid (AA) sequence into a unique 3D or native state; this process is known as protein folding. Understanding the mechanisms and rules of this process has been one of the most pursued objectives of computational biology, and an important theoretical tool to study these phenomena is simulation. Simulations are used to generate folding trajectories (Figure 1), which describe the sequence of states that proteins follow as a function of time during the folding process.

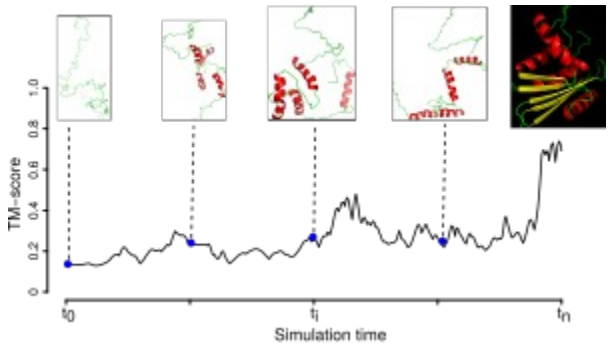


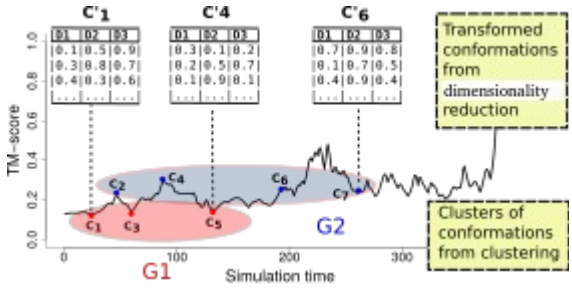
Figure 1. The evolution of folding is measured at each step by comparing the structure at time t_i to the native structure (black image background) using a structure comparison metric; in this work, we use the TM-score as the metric (see Methods). X-axis: Simulation time from t_0 to t_n. Y-axis: TM-score value from 0 (different) to 1 (similar).

Most folding simulations use the molecular dynamics (MD) method, which, due to its computational cost, is limited to small proteins (<100 AA) and very short times (picoseconds or microseconds). However, technological innovations have allowed significant advances in these simulations, both with respect to time scale and the technology employed to execute them. In 2011, using the Anton supercomputer, specifically designed for protein folding [1], full simulations of 12 proteins were published, several on the order of milliseconds [2]. More recently, in 2016, the Anton 2 supercomputer became operational [3], being up to ten times faster than its predecessor Anton. As an economic alternative, in 2014, graphic processing units (GPU) were used to simulate, on the order of microseconds, the folding of 17 proteins [4]. Years earlier, in 2007, the Folding@home distributed computing platform utilized as many as 250,000 PCs, voluntarily available around the world, to simulate on the order of microseconds the folding of the villin-headpiece protein [5].

These innovations ~~show~~ have allowed significant progress in protein folding simulations, both ~~on~~ with respect to time scale~~s~~ and the technology used to execute them, ~~and as a~~ resulting in the generation of trajectories with millions of

conformations. ~~But~~ However, due to the~~ir~~ large number of conformations, their processing and analysis ~~in~~ with conventional PCs is computationally expensive, and new algorithms are needed to ~~efficiently~~ simplify them~~, seeking to~~ while preserving~~e~~ as much information as possible.

Two approaches that have been used to reduce trajectories ~~these simulations have been the~~are dimensionality reduction~~–~~ [6] and clustering [7]. In the dimensionality reduction approach, conformations are transformed into reduced sets of variables that represent ~~them~~ the original conformations as well as possible. ~~Here, b~~Both linear and non-linear techniques have been used (e.g.~~,~~ principal component analysis (PCA) and multidimensional scaling [8], Isomap [9], diffusion maps [10]). However, many of these techniques analyze a trajectory~~,~~ instead of reducing ~~a trajectory, analyze~~ it, losing the structural information of the conformations (Figure 2, top) and making the results explainable only when observed together. In addition, many of these techniques require pairwise comparisons, which are computationally expensive when the trajectories are very large.



In the clustering approach, the conformations are assigned to groups that share similar characteristics (e.g., similarity with the native structure), and from each group~~,~~ an average representative or its general characteristics can be ~~taken~~obtained. ~~Here, h~~Hierarchical and partitioned groupings have been used for this approach (e.g., k-means [11], link [12]). However, the groups lose their temporal order since they can include conformations that occur in very distant times (Figure 2, bottom). ~~And~~ They also ~~they~~ require pairwise comparisons, which are computationally expensive when the trajectories are very large.

Figure 2. The dimensionality reduction method (above) transforms the conformations (c1, c2 and c3) into a new set of values (c'1, c'2 and, c'3), but their structural information is lost~~. While~~, whereas the clustering method (bottom) forms two groups (G1 and G2), but their temporal order is lost since they contain conformations that overlap in time (c2, c3, c4, c5).

To reduce a folding trajectory, the proposed algorithm divides the path into segments that are processed in parallel. For each segment, characteristic events are quickly extracted using the rapid clustering strategy of Hobohm and Sander (1992) adapted for protein folding trajectories~~; and f~~. From these results, the most representative events are selected by a strategy of k-medoids [13]. The results of each segment are joined to form the reduced trajectory with the most representative conformations of the original trajectory~~,~~ while retaining both its 3D representation ~~as~~ and ~~their~~ temporal order.

The algorithm is implemented in the R language~~,~~ except for the function used for pairwise structure comparison, the TM-score [14]~~;, which is the~~ this function is executed ~~more~~ many times and ~~that~~ is implemented in the Fortran language.

Methods

Datasets of protein folding trajectories

We used the folding trajectories of three proteins ~~taken~~ obtained from different simulation projects. ~~First, the~~One of the trajectories was that~~y~~ of the trp-cage protein, simulated with molecular dynamics (MD) using the Anton supercomputer [2], with a simulation time of 208 ~~*u*~~*μs*, a time step of 200 *ps* ~~*time step*~~, and 1044001 conformations. ~~Second, the~~The second trajectory was that of the villin-headpiece protein, also simulated with ~~D~~MD using the F~~f~~olding@home distributed platform [15], with a simulation time of 8 *μ*~~*u*~~*s*, a time step of 50 *ps* ~~*time step*~~, and 15201 conformations. ~~And~~ The third~~, the~~ trajectory was that of the ribonuclease H protein, simulated with the Probabilistic Roadmap Method using the Parasol folding server [16], with 429 folding steps (instead of time steps, see below), each corresponding to 429 conformations.

Time steps and Folding steps

A time step in MD trajectories is the time length at which conformations are sampled or evaluated during the simulation~~. While~~, whereas a folding step, as employed in the PRM and ~~in~~ the reduced trajectories produced by our algorithm, represents the most likely conformation occurring during a time interval or from a set of likely candidate conformations.

Pairwise comparison of protein structures using the TM-score

In this work, we used the TM-score metric for the pairwise comparison of protein structure~~s~~ [14]. This metric is used in both the proposed algorithm and in the techniques for reduction of protein folding used ~~to~~ for compar~~e its results~~ison. The TM-score is more sensitive to the global topology than local variations~~, and so~~; thus, it estimates the pairwise similarity of protein structures much more accurately than the Root Mean Square-Deviation (RMSD), a common metric used for the same purpose. The TM-score ranges from 0 to 1, where 1 is a perfect match. Based on statistics [17], a TM-score lower than 0.17 indicates no similarity of two random structures ~~with no relation of similarity~~, ~~and~~ whereas a TM-score higher than 0.5 indicates ~~that the structures have~~ a degree of similarity between two structures beyond that expected ~~that is not given~~ by chance.

Other techniques for protein folding reduction

nMDS and clustering techniques were used to ~~get~~ obtain the intrinsic information from both the original and the two reduced trajectories of the villin-headpiece protein [5], and their results were~~n~~ compare~~themd~~ (s~~S~~ee R~~r~~esults).

nMDS reductions were carried out using the R-function monoMDS [18], taking as input the dissimilarity matrix obtained from the pairwise comparisons of all the protein conformations of the folding trajectory. ~~And, the~~The complete-linkage clustering reductions were carried out using the R-function hclust [19], taking as input a matrix with the first two principal components from a PCA analysis. This PCA analysis was carried out using the R-function pca.xyz [20], taking as input a matrix with the 3D coordinates of the *Cα* atoms of all the protein conformations of the folding trajectory.

The reduced trajectories were calculated with the proposed algorithm from the villin-headpiece trajectory with 15201 conformations. The first reduced trajectory had ~~with~~ 7197 conformations (representing a reduction of~~reduced by~~ 52%), and the second ~~with~~ had 2258 conformations (representing a reduction of ~~reduced by~~ 80%).

Implementation

The proposed algorithm reduces a trajectory of protein folding in three steps: partitioning, extraction, and selection. The first step runs only once, ~~while~~ whereas the other two runs several times independently, allowing them to run in parallel. Each step involves a strategy to improve the efficiency of the algorithm when working with large protein~~-~~-folding trajectories. Figure 3 shows the overview of the algorithm, and the steps are given below.

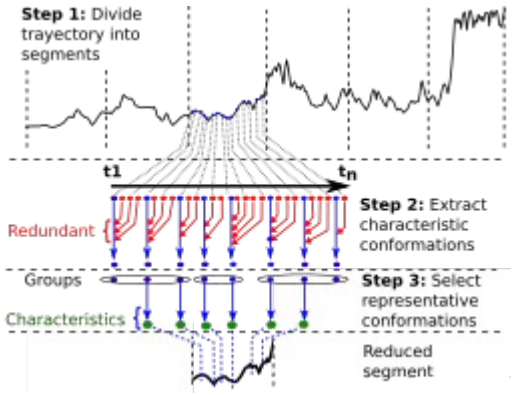
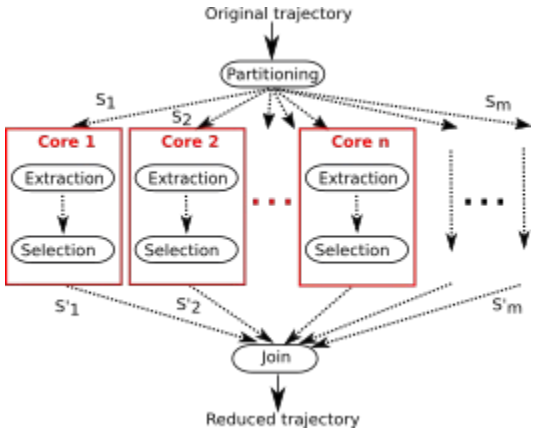


Figure 3. Step 1: Divide the trajectory into segments (vertical dashed lines) and reduce each individually. Each segment has n conformations from time t_1 to t_n. Step 2: For each segment, extract characteristic conformations (blue dots) and remove redundant ones (red dots). Stage 3: Select the most representative conformations~~s from~~ among the previous conformations.

Step 1: Partitioning

In step 1, ~~Divide~~ the trajectory is divided into segments to reduce them locally and in parallel (dotted vertical lines, Figure 3). This process is carried out by dividing the trajectory from the start to the end in segments with *N* conformations each, where *N* is an input parameter. Local reductions allow ~~to~~ focus on the particular characteristics of each segment that will determine the global characteristics of the trajectory. ~~And~~ The parallel reductions allow ~~to~~ improved ~~the~~ algorithm efficiency when it runs on machines with more than one processor (e.g., multi-core computers) (Figure~~a~~ 4).



Figure~~a~~ 4. The trajectory is divided into m segments (S1, ..., Sm), and each segment is processed separately and in parallel on a machine with n processing cores. The results of the extraction and selection steps (S1 ', ..., Sm') are independent of each other and are joined to form the reduced trajectory.

~~Stage~~ Step 2: Extraction

In step 2, ~~Quickly extract~~ the characteristic conformations of each segment are rapidly extracted, and ~~eliminate~~ the redundant ones are eliminated. This step is efficiently carried out ~~efficiently~~ by means of a rapid clustering approach that performs ~~relatively~~ few pairwise comparisons and extracts the most dissimilar conformations of a segment , instead of grouping similar ~~conformations of a segment~~ ones~~, extracts the most dissimilar ones~~.

Here, we improved the fast clustering algorithm of Hobohm and Sander (1992) to work with a trajectory segment and exploit the implicit order of its conformations given by its simulation time (black horizontal line, Figure 3). The algorithm selects the initial conformation at time t_1 as the first characteristic. Then, the algorithm compares the previous characteristic with the following conformation. If dissimilar, ~~then~~ the conformation becomes a new characteristic~~;~~, otherwise, the conformation is redundant and is removed (red dots, Figure 3). The process continues with the ~~rest of~~ remaining conformations until finishing ~~in~~ with the final one at time t_n, thus producing the set of representative characteristics of the segment (green dots, Figure 3).

Step 3: Selection

In step 3, ~~Take~~ the conformations of the previously extracted characteristics ~~and~~ are cluster~~-themed~~ to select the most representative characteristics. To ~~find~~ identify these representatives, the algorithm uses a k-medoids strategy (PAM algorithm [13]) that calculates the k conformations (medoids) ~~whose~~ with the minimal average difference between all the other members of the group ~~is minimal~~.

~~However, the~~The PAM algorithm needs as input ~~the~~ a dissimilarity matrix with the pairwise comparisons of all-versus-all conformations of the trajectory segment; generating this matrix~~, which~~ is an intensive computational task when the number of conformations is very large. ~~But~~However, this task is feasible ~~to perform~~here since the algorithm is working ~~here~~ with a reduced set of characteristic conformations (attained in the previous step) of a trajectory segment and not ~~of~~ the complete trajectory.

Results and Discussion

Three tests were carried out to evaluate the capacity and performance of the proposed algorithm~~:~~. In the first test, reduction of three trajectories was performed using the proposed algorithm~~;~~. In the second, ~~comparison of~~ the intrinsic information preserved by the reductions ~~from both~~was compared between the proposed algorithm and two other folding reduction techniques~~; and~~. In the third, the ~~a~~ performance of the methods was ~~comparison~~compared.

Reduction of protein folding trajectories

Figure 5 shows the TM-score plots of the reduced trajectories produced by the proposed algorithm. ~~It can be seen from the~~The plots show that the algorithm ~~try~~ attempts to find the most representative conformations from the~~ir~~ original trajectories while~~,~~ conserving two fundamental properties: the structure and the temporal ordering of the original conformations. The o~~O~~ther folding reduction methods investigated lose these properties in their reductions, as ~~we will see~~described in the next section.

As a result, these reduced trajectories become a summary of the original ones and can be used as inputs for more complex analy~~sz~~es~~,~~ or ~~even for~~ other reduction methods that require pairwise comparisons ~~and~~ but are~~become~~ impractical for large trajectories.

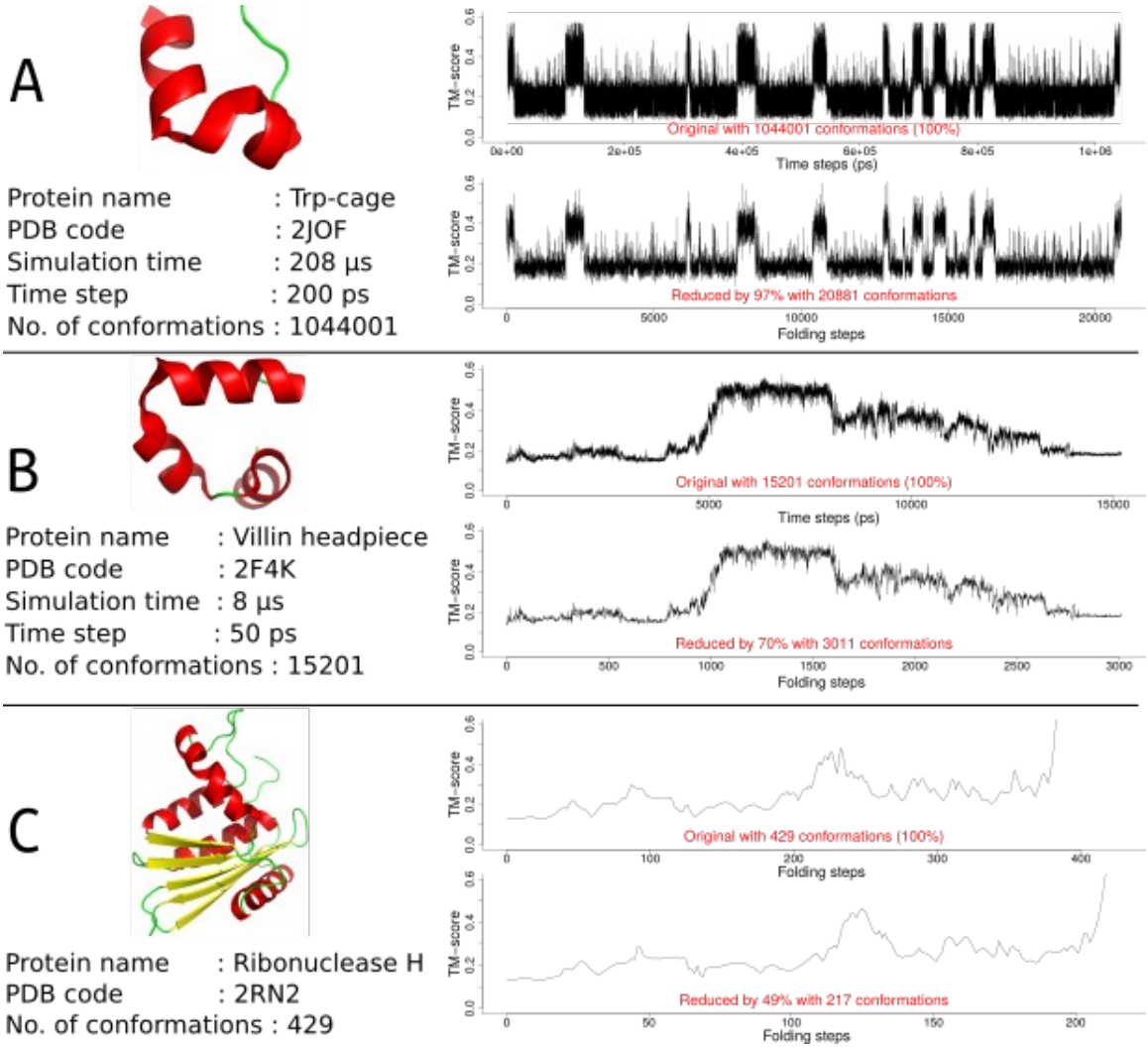


Figure 5. Plots of the TM-score values for the original and reduced trajectories. Protein and simulation details on the left. Trajectories on the right: original (upper part), and reduced (lower part). (A) Trajectory of trp-cage protein, reduced by 98%, from 1044001 to 20881 conformations. (B) Trajectory of the villin-headpiece protein, reduced by 80%, from 15201 to 3011 conformations. (C) Trajectory of ribonuclease H protein, reduced by 49%, from 429 to 217 conformations.

Comparison with other methods

~~Here, w~~We compared how the intrinsic information captured from a folding trajectory by other folding reduction~~s~~ techniques ~~from a folding trajectory~~ is also preserved in the reductions produced by the proposed algorithm. First, two reduced trajectories were computed from the original trajectory of the villin-headpiece protein using the proposed algorithm (Figure 6), and then the intrinsic information was computed on these trajectories using nMDS and clustering reductions (Figure 7) (see Methods for the details of the trajectory and techniques).

As ~~it can be seen from the~~shown in Figure 7, the pattern of circles of points from nMDS~~,~~ and the structure of the three groups from clustering~~, repeat in both~~ are consistent between the original and ~~the~~ reduced trajectories. This result shows that

the reductions produced by the proposed algorithm largely preserve the intrinsic information observed in the original trajectory. Furthermore, the proposed algorithm has several advantages. First, by employing nMDS and clustering, the algorithm ~~it~~ avoids the calculation of the dissimilarity matrix, which ~~as it is done by nMDS and clustering, that~~ is a computationally expensive task for medium to large trajectories. Second, its reductions are a set of protein conformations, in ~~contrary~~ contrast to the reduced transformations ~~as the~~ produced with other techniques, such as nMDS, PCA, Isomap or diffusion maps [8, 21, 10], that lose structural information and that can only be interpreted when viewed together. ~~And t~~Third, in the proposed algorithm, the temporal ordering of conformations is conserved, in contra~~str~~y to clustering methods [7] that merge configurations from different simulation times into clusters.

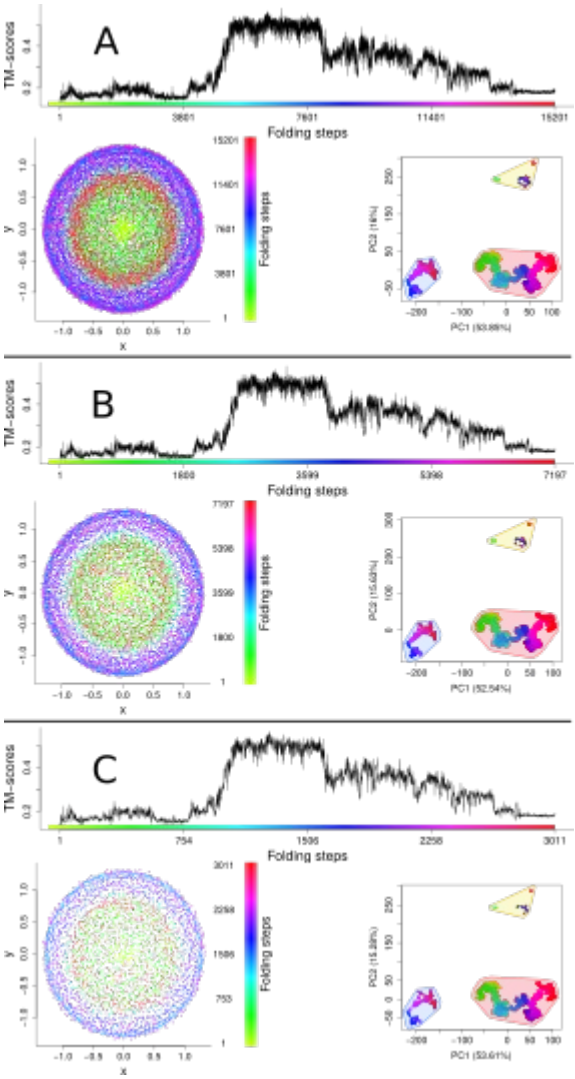


Figure 6. *Intrinsic information captured from the original trajectory of the villin-headpiece protein is largely preserved in the reduced trajectories produced by our algorithm. TM-score plot, nMDS pattern and clustering groups are presented at the top, bottom left, and bottom right of each ~~figure~~panel. (A) Original trajectory. (B and C) ~~Reduced t~~Trajectories reduced by 52%~~.~~ and 80%, respectively. The nMDS produces a set of circular patterns associated with conformations occurring in different times of the*

simulation, ~~while~~ whereas clustering produces a structure of three clusters associated with the first two principal components of the Cα atoms of the conformations.

Algorithm~~-~~ performance

The performance of the proposed algorithm was tested ~~in~~ via two ~~cases~~tests: ~~first,~~ a runtime comparison of the proposed algorithm with the three other ~~three~~ typical techniques for folding reduction, i.e.,~~:~~ nMDS, clustering, and PCA; and ~~second,~~ a parallel comparison in which~~,~~ ~~running~~ the algorithm was run with several cores (Figure 7). For both tests, the trajectory correspond~~ed~~eds to the first 100k conformations from the full trajectory of the trp-cage protein, described above in datasets section.

For the comparison ~~between~~ among techniques (Figure 7~~.~~A), several subtrajectories of different lengths were reduced by all the techniques. When the techniques were run with one core, PCA showed the most efficient runtimes, followed by our proposed algorithm FR ~~(1 core)~~, ~~contrary to~~whereas nMDS and clustering ~~that becomes~~became impractical when ~~faced~~ used with short to medium trajectories. ~~However~~Furthermore, in contrast to the other techniques, the proposed algorithm has the advantage of being able to run easily ~~run~~ in parallel~~, contrary to the other techniques, and w~~. When it was run~~s~~ using two cores, ~~its~~ the runtime ~~is~~ was reduced by half and ~~becomes~~ was ~~faster~~ shorter than that achieved with PCA~~.~~ (FR (2 cores), black dashed line).

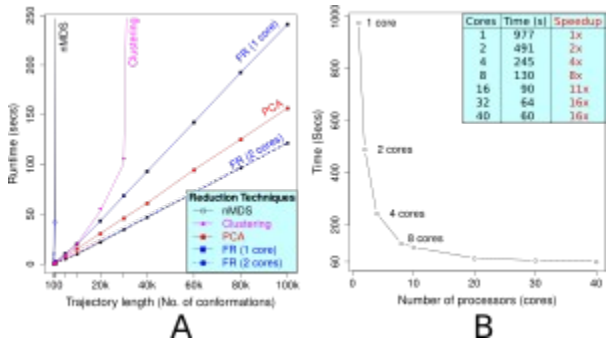


Figure 7. *(A) Runtime comparisons for four folding reduction techniques: nMDS, clustering, PCA, and our algorithm FR (running with one and two cores). With one core, PCA ~~is~~ was the most efficient algorithm, followed by our algorithm FR ~~with one core, bu~~. However,~~t~~ when ~~it~~ the algorithms were run~~s~~ with two cores, ~~its~~ the runtime of the proposed algorithm was~~is~~ reduced by half, ~~being~~ with the proposed algorithm being more efficient than PCA. (B) Evaluation of ~~the~~ algorithm ~~paralellization~~parallelization. When the number of processing cores is multiplied by two, the processing time reduces by half with a quasi-linear speedup that is maintain~~eds~~ for up to ~8 cores (8x).*

To test how ~~the~~ parallelization improves the algorithm performance, the full dataset of 100k conformations was reduced by the algorithm using different number~~s~~ of cores. The runtimes are shown in ~~the~~ Figure 7~~.~~B, ~~where it is notable a~~which reveals good speedup~~, that reduces~~ with the processing time reduced by half with every ~~time~~ doubling of ~~the~~ core number ~~of cores is duplicated~~. This speedup is maintain~~eds~~ for up to ~8 cores~~,~~ and ~~then it reduces~~decreases to the minimum after ~30 cores.

These results show that the algorithm has ~~a~~ good performance ~~when~~ compared with that of the other techniques~~,~~ and that this performance is improved~~s more~~ when ~~it~~ the algorithm is run in parallel using more than one core. As a consequence, the speedup of the algorithm scales quasi-linearly with the number of processing cores~~,~~ to almost ~~until~~ 8x, and with 32 cores, the algorithm ~~still~~ achieves a speedup of 16x. ~~Now, e~~ Considering that multi-core technology is ~~quite~~ commonplace ~~for~~ even for desktop computers, the proposed algorithm has the capacity to take advantage of this technology to reduce large protein folding trajectories in a fast parallel manner, with runtimes closer or better than those of other techniques commonly used for this task.

Conclusions

Although the progress in long timescale simulations of protein folding has enabled the generation of large folding trajectories, ~~the~~ a new challenge ~~is~~ lies in their analysis~~., but due~~ Due to the millions of conformations they can contain, their processing and analysis ~~becomes~~ is difficult or impractical.

Here, we have proposed a fast and parallel algorithm to simplify large protein-folding trajectories. The algorithm reduces a trajectory by splitting it into segments and then reducing each in parallel using a fast clustering strategy ~~which~~ that avoids the pairwise comparison of all structures.

According to the results, the algorithm can achieve resumed trajectories with high compression of data and preserv~~e~~ing their main conformations, ~~what was~~as confirmed ~~when~~ by comparison with the patterns and clusters produced by other folding reduction techniques~~ were also observed in the algorithm reductions~~. Furthermore, when run on a single core, the algorithm outperformed all of the ~~performance of the~~ other techniques~~, apart from~~ except the PCA technique. However~~, if the algorithm uses~~when additional processing cores were used, ~~it~~ the proposed algorithm outperform~~ed~~s all the other techniques at larger values.

Nevertheless, ~~the reductions produced by~~ the proposed algorithm ~~are~~ is limited to creat~~e~~ing a summary of the main events of a protein folding trajectory without performing any kind of analysis, ~~as~~ in contrast to other techniques~~ do~~. ~~But~~However, the~~se~~ summarized trajectories can be used as input to these and other techniques that serve the same purpose ~~and~~ but ~~which~~ were not designed to handle large protein-folding trajectories.

Author details

[1]Corporación Colombiana de Investigación Agropecuaria – AGROSAVIA, CI Tibaitatá, Kilómetro 14, Vía a Mosquera, Colombia. [2]The European Bioinformatics Institute 345 (EMBL-EBI), Hinxton, Cambridgeshire, UK. [3]Facultad de Ingeniería Electrónica y Telecomunicaciones, Universidad del Cauca, Popayán, Colombia. [4]Escuela de Ingeniería de Sistemas y Computación, Universidad del Valle, Santiago de Cali, Colombia