RESEARCH

A fast parallel algorithm to reduce protein folding trajectories

Luis Garreta[1†], Mauricio Martinez[2] and Pedro A Moreno[1*]

Abstract

Background: The simulations are one of the most important tools for studying and understanding the underlying mechanisms of the protein folding process. Protein folding simulations have experienced substantial progress in the last years, they are performed using diverse technologies and they are reaching the microseconds and greater timescales, which generates very long trajectories. As a result, the analysis of these trajectories entails to complications and is necessary to create tools 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 from each segment selects the most representative conformations using a rapid clustering strategy, which takes advantage of the temporal order of the conformations to compare them locally, avoiding an all-versus-all comparison. The algorithm reduces a trajectory in 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 shown by other efficient reduction techniques, and this performance 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 thus preserving both its structure and its temporal order. The reduced trajectories can be used as input for more complex analysis techniques and even for other reduction techniques that become impractical when faced with long folding trajectories. The algorithm is fast and is designed to run in parallel on conventional PCs with multi-core technology, which are present in most typical research laboratories.

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

Background

We present a parallel algorithm to reduce protein folding trajectories which quickly obtains representative conformations, conserving both their three-dimensional structure (3D) and their temporal order. Proteins play a fundamental role in all living beings, but to be functional, they must fold from their linear amino acid (AA) sequence to a unique 3D or native state, which is known as the protein folding process. 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 it has been the simulations of protein folding. These simulations generate folding trajectories (Figure 1), which describe the sequence of states that proteins follow as a function of time during their folding process.

Folding simulations mainly use the molecular dynamics (DM) 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 on time scales and technology to execute them. In 2011, using the Anton supercomputer, specially designed for protein folding [1], full simulations of 12 proteins were published, several on the order of milliseconds [2]. And 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

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

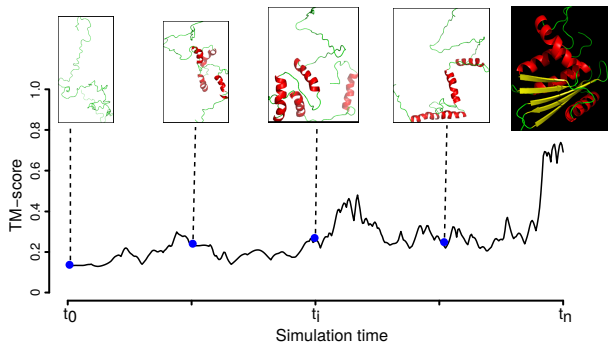

Figure 1 Protein folding trajectory. The evolution of the folding is measured at each step by comparing the structure at time t_i to the native structure (black image background) using a structure comparison metric, in this work we use the TM-Score (see Methods). X-axis: Simulation time from t_0 to t_n.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

simulate, on the order of microseconds, the folding of 17 proteins [4]. And 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 significant progress in protein folding simulations, both on time scales and technology to execute them, and as a result the generation of trajectories with millions of conformations. But due to their large number of conformations, their processing and analysis in conventional PCs is computationally expensive, and new algorithms are needed to efficiently simplify them, seeking to preserve as much information as possible.

Two approaches used to reduce these simulations have been the dimensionality reduction [6] and clustering [7]. In the dimensionality reduction approach, conformations are transformed into reduced sets of variables that represent them as well as possible. Here, 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, 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 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. Here, hierarchical and partitioned groupings have been used (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 also they require pairwise comparisons, which are computationally expensive when trajectories are very large.

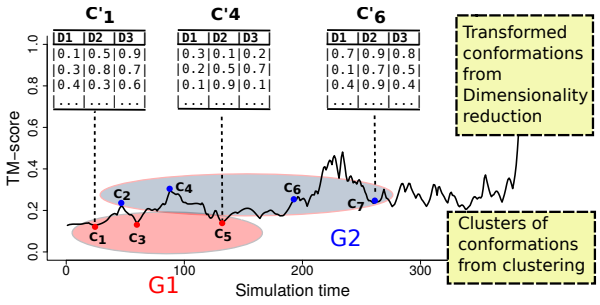

Figure 2 Loss of information in dimensionality reduction and clustering. The dimensionality reduction (above) transforms the conformations (c1, c2 and c3) into a new set of values (c'1, c'2 and, c'3), but their structural information is lost. While the clustering (bottom) forms two groups (G1 and G2), 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 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 their temporal order.

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

Methods

Datasets of protein folding trajectories

In this work, we used the trajectories from three protein folding simulations projects, two using Molecular Dynamics simulations (MD) and one using the Probabilistic Roadmap Method (PRM). First, the trajectory of the trp-cage protein, simulated with DM using the Anton supercomputer [2], with a simulation time of

208 μs, a 200 ps time step, and 1044001 conformations. Second, the trajectory of the villin-headpiece protein, simulated with DM using the folding@home distributed platform [15], with a simulation time of 8 μs, a 50 ps time step, and 15201 conformations. And third, the trajectory of the ribonuclease H protein, simulated with the PRM using the Parasol folding server [16], with 429 folding steps 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 a folding step, in the PRM and in our reduced trajectories, 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 pairwise comparison of protein structures [14]. This metric is used in both the proposed algorithm and in the techniques for reduction of protein folding used to compare its results. The TM-score is more sensitive to the global topology than local variations, and so 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 two random structures with no relation of similarity, and a Tm-score higher than 0.5 indicates that the structures have a degree of similarity that is not given by chance.

Other techniques for protein folding reduction

nMDS and clustering techniques were used to get the intrinsic information from both the original and two reduced trajectories of the villin-headpiece protein [5], and then compare them (See results).

nMDS reductions were carried out using the R-function monoMDS [18], taking as input the dissimilarity matrix obtained from the pairwise comparison of all the protein conformations of the folding trajectory. And, 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 with 7197 conformations (reduced by 52%), and the second with 2258 conformations (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 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.

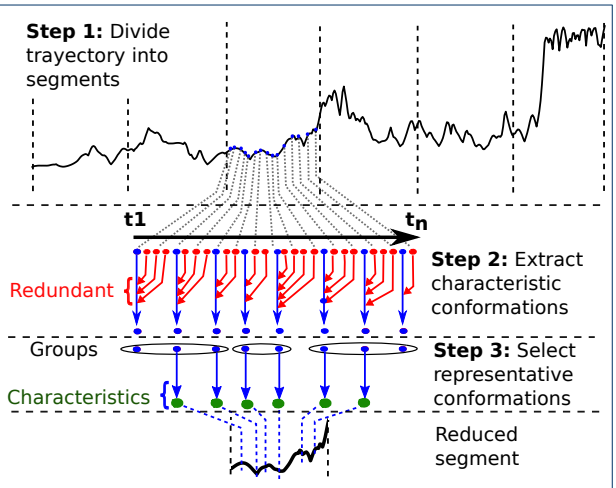

Figure 3 Overview of the proposed algorithm. Step 1: Divide the trajectory into segments (vertical dashed lines) and reduce each individually. Each segment has n conformations from time t_1 to t_n. Step 2: For each segment, extract characteristic conformations (blue dots) and remove redundant ones (red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

Step 1: Partitioning

Divide the trajectory into segments to reduce them locally and in parallel (dotted vertical lines, Figure 3). This 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 parallel reductions allow to improve the algorithm efficiency when it runs on machines with more than one processor (e.g. multi-core computers) (Figura 4).

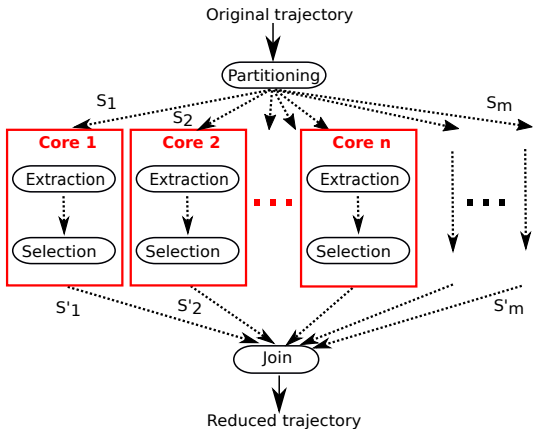

Figure 4 Parallel processing of trajectory segments. The trajectory is divided into m segments (S1, ..., Sm), and each segment is processed separately and in parallel on a machine with n processing cores. The results of the extraction and selection steps (S1 ', ..., Sm') are independent of each other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

Stage 2: Extraction

Quickly extract the characteristic conformations of each segment and eliminate the redundant ones. This is carried out efficiently by means of a rapid clustering approach that performs relatively few pairwise comparisons and, instead of grouping similar conformations of a segment, 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 conformations until finishing in the final one at time t_n, thus producing the set of representative characteristics of the segment (green dots, Figure 3)

Step 3: Selection

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

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

Results and Discussion

To evaluate the capacity and performance of our algorithm, we carried out three tests: first, we reduced the trajectory of three proteins; second, we compare the results of the algorithm with the results of other reduction techniques; and third, we compare the performance of the algorithm against that of other techniques, as well as when it is executed in parallel.

Reduction of three protein folding trajectories

We reduced the folding trajectories of three proteins: trp-cage, villin-headpiece and ribonuclease H (see Methods for the details of the simulation). Figure 5 shows the TM-score plots resulting from the original and reduced trajectories. As it can be seen, the reductions contain the most representative conformations from their original trajectories, conserving two fundamental properties: the structure and the temporal ordering of the original conformations. Other folding reduction methods lose these properties in their reductions, as we will see in the next section.

As a result, the reduced trajectories generated by our algorithm become a summary of the original ones and can be used as inputs for more complex analyzes, or even for other reduction methods that require pairwise comparisons and become impractical for large trajectories.

Comparison with other methods

Here, we compared how the intrinsic information captured by other folding reductions techniques from a

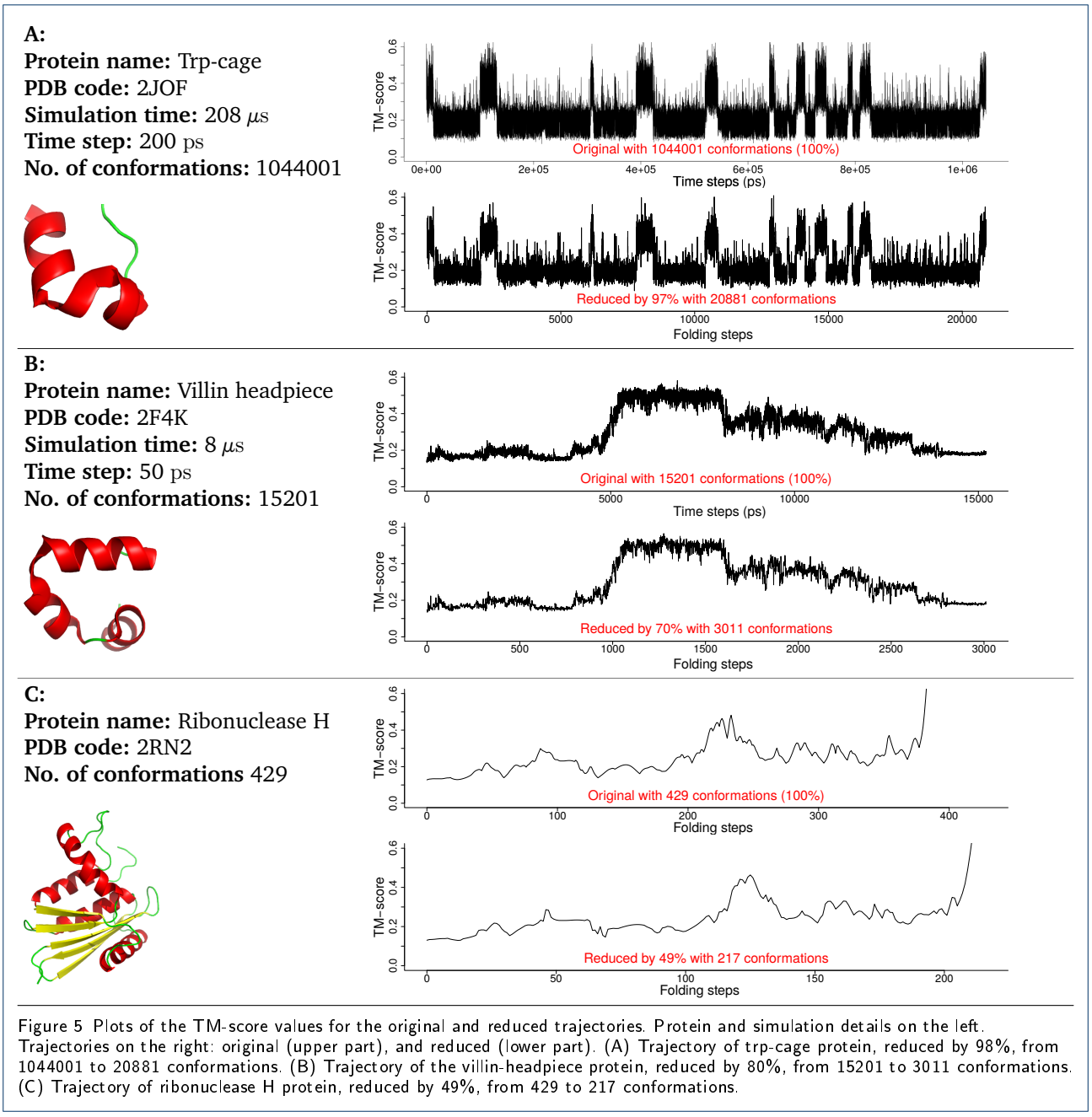

Figure 5 Plots of the TM-score values for the original and reduced trajectories. Protein and simulation details on the left. Trajectories on the right: original (upper part), and reduced (lower part). (A) Trajectory of trp-cage protein, reduced by 98%, from 1044001 to 20881 conformations. (B) Trajectory 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.

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 Figure 7, the pattern of circles of points from nMDS, and the structure of three groups from clustering, repeat in both the original and the reduced trajectories. This 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, it avoids the calculation of the dissimilarity matrix 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, contrary to reduced transformations as the produced with other techniques as nMDS, PCA, Isomap or dif-

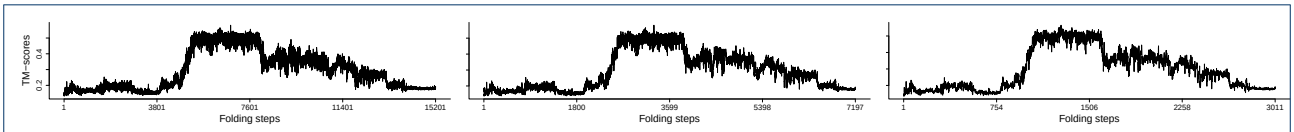

Figure 6 Plots for the reductions produced by the proposed algorithm for the villin-headpiece trajectories. The figure shows the TM-scores plots for the original, with 15201 conformations, and two reductions: one with with 7197 conformations (reduced by 52%), the other with 2258 conformations (reduced by 80%). It is noted that the reductions preserve the main folding events.

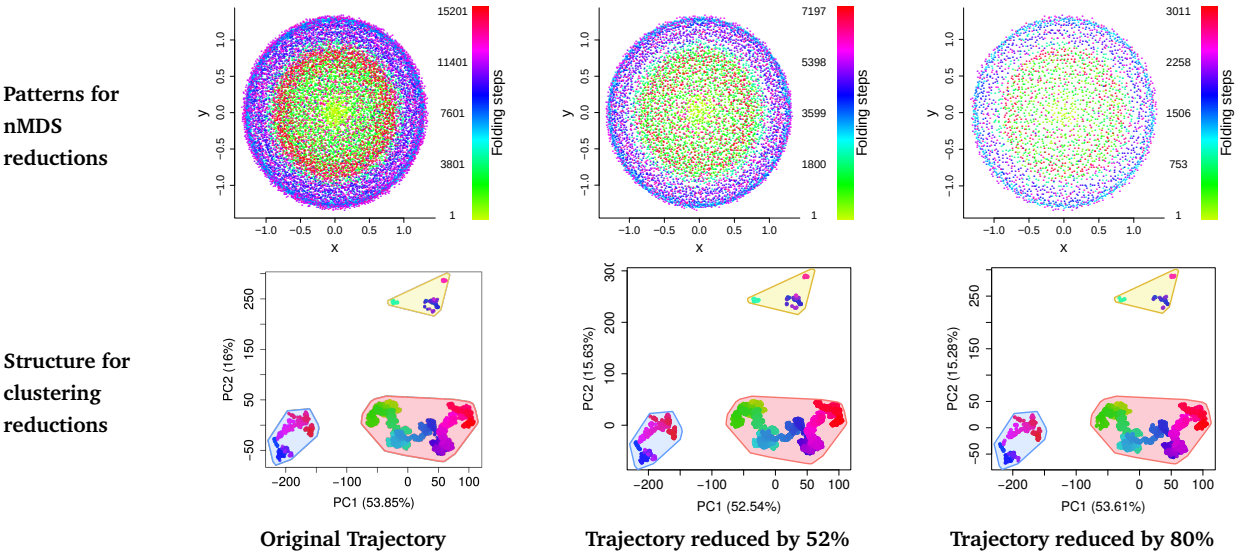

Figure 7 Intrinsic information captured by nMDS and clustering reductions. The figure shows that the intrinsic information captured from the original trajectory of the villin-headpiece protein (plots of the nMDS pattern and clustering structure on the left) largely preserves in the reduced trajectories produced by the proposed algorithm (plots on the center and right). The nMDS produces a set of circular patterns associated with conformations occurring in different times of the simulation, while clustering produces a structure of three clusters associated with the first two principal components of the Cα atoms of the conformations.

fusion maps [8, 21, 10] that lose structural information and that can only be interpreted when viewed together. And third, temporal ordering of conformations is conserved, contrary to clustering methods [7] that merge configurations from different simulation times into clusters.

Pwara las comparaciones utilizamos los datos de la simulación de plegamiento de la proteína villin-headpiece del proyecto folding@home [5]. Tomamos la trayectoria original y calculamos su reducción por los métodos de nMDS y PCA. Luego, calculamos dos reducciones con nuestro algoritmo sobre esta trayectoria y a los datos resultantes le calculamos nuevamente las reducciones por nMDS y PCA. Los resultados se muestran en la figura **??**, donde cada fila contiene tres despliegues en 2D: de la trayectoria, del patrón resultante de la reducción por nMDS, y del agrupamiento al proyectar los dos primeros componentes del PCA.

El desempeño de nuestro algoritmo lo evaluamos en dos situaciones: comparándolo frente a otros métodos de reducción (Figura 8) y ejecutándolo en paralelo usando múltiples núcleos de procesamiento (Figura 9). Para esto utilizamos las 100K primeras conformaciones de la trayectoria de la proteína Trp-cage (ver Métodos). Para la primera evaluación ejecutamos los métodos con diferentes tamaños de subtrayectorias, desde 100 hasta 100K conformaciones, y en la segunda evaluación ejecutamos nuestro algoritmo con diferente número de núcleos de procesamiento.

En la comparación con otros métodos de reducción, la figura 8 muestra que PCA es el más eficiente seguido de nuestro algoritmo FastReduction cuando se ejecuta con un solo núcleo de procesamiento. Sin embargo, si lo ejecutamos en paralelo con 2 núcleos, este se vuelve más eficiente que PCA. Por el contrario, nMDS y clustering se vuelven imprácticos con subtrayectorias medianamente largas. Ahora, si ejecutamos nuestro algo-

ritmo en paralelo con 2 cores (FR2, línea azul), este se vuelve más eficiente que PCA.

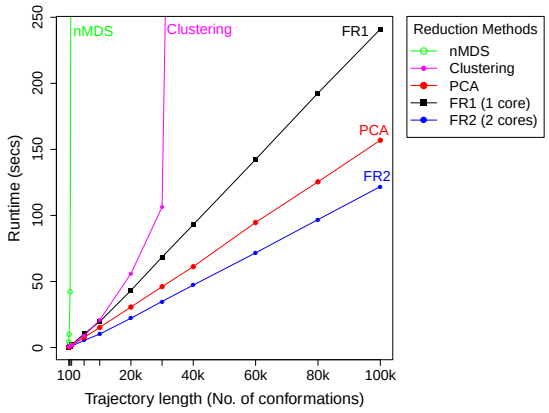

Figure 8 Desempeño del algoritmo frente a otros métodos. Comparación del nuestro algoritmo FR1 con nMDS, PCA, y agrupamiento. PCA y FR1 son los más eficientes, pero si nuestro algoritmo utiliza dos núcleos (FR2), el tiempo se disminuye a la mitad y se vuelve más eficiente que PCA. Por el contrario, nMDS y clustering toman demasiado tiempo, aún con trayectorias pequeñas.

Este comportamiento lo podemos ver más claramente en la figura 9, donde se muestran los tiempos y la aceleración que alcanza el algoritmo a medida que se ejecuta con más núcleos. Cada que duplicamos el número de núcleos, el tiempo de ejecución se disminuye casi a la mitad, hasta los 8 núcleos esta relación se conserva y luego la disminución es menor hasta volverse mínima pasados los 30 núcleos.

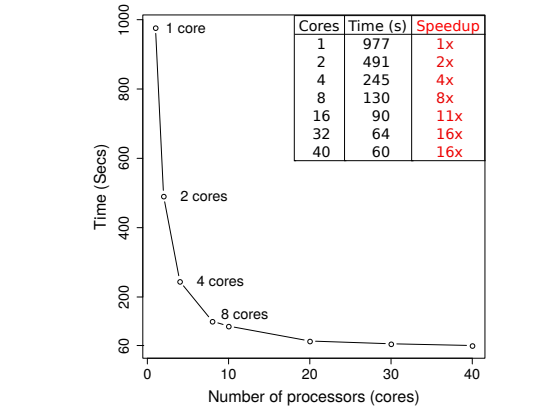

Figure 9 Desempeño del algoritmo cuando se ejecuta en paralelo con múltiples núcleos. Al duplicar el número de núcleos de procesamiento el tiempo se disminuye casi a la mitad y por lo tanto la aceleración crece casi de forma lineal, por lo menos hasta los 8 núcleos (8x). Después de esto, la aceleración sigue siendo apreciable hasta casi después de los 30 núcleos.

Todo lo anterior nos muestra que el algoritmo presenta un buen desempeño comparado con los otros métodos, y que este mejora más cuando aprovecha su paralelismo y se ejecuta con más de un núcleo. Como consecuencia, la aceleración de nuestro algoritmo escala de forma lineal con el número de núcleos que utiliza, por lo menos hasta 8x, es decir, la velocidad de ejecución cuando utiliza 8 núcleos es 8 veces más que cuando utiliza solo uno. Además, con 32 núcleos todavía se logra una aceleración de 16x, después de lo cual esta se mantiene sin mayor aumento (ver recuadro figura **??**.B). Ahora, considerando que la tecnología multi-core ya está presente en muchas de los computadores de hoy día, el algoritmo tiene la capacidad de aprovechar esta tecnología para reducir trayectorias largas en tiempos cortos, cercanos e incluso mejores que los que toman algunos de los métodos cómunes usados en reducción de trayectorias de plegamiento.

Conclusiones

Las simulaciones de plegamiento de proteínas están avanzando significativamente y cada vez se realizan más para nuevas proteínas, con tiempos de duración más largos, y llevadas a cabo sobre diversas tecnologías. Como consecuencia, las trayectorias generadas por estas simulaciones cada vez son más extensas, del orden de millones de conformaciones, lo cual hace difícil su procesamiento y análisis. Para simplificarlas se han planteado diferentes técnicas que más bien son técnicas de análisis que transforman las conformaciones o crean grupos de ellas y sus resultados tienen sentido solo cuando se observan en conjunto.

Aquí, nosotros hemos planteado un algoritmo para simplificar trayectorias de plegamiento que divide la trayectoria en segmentos y extrae de ellos sus eventos principales o conformaciones destacadas en dos fases: primero extrae rápidamente las conformaciones disimilares y luego una selecciona de estas a las más representativas. El algoritmo se caracteriza por ser rápido y fácilmente paralelizable, y por lo tanto ejecutable en máquinas ordinarias con múltiples cores, disponibles ya en la mayoría de laboratorios de investigación.

De acuerdo a los resultados, el algoritmo produce simplificaciones de las trayectorias originales con una compresión alta y con los eventos principales visualmente conservados. Así mismo, estos resultados conservan en gran medida los patrones y la estructura que producen las reducciones hechas por otras técnicas de reducción y análisis de trayectorias. En cuanto al desempeño del algoritmo, este se aproxima al mostrado por algunas de las técnicas más eficientes y mejora mucho cuando se ejecuta en paralelo.

Sin embargo, las simplificaciones producidas por el algoritmo están limitadas a crear resúmenes de las trayectorias sin realizarles ningún tipo de análisis, como lo hacen otras técnicas. Por esta misma razón, estas trayectorias resumidas pueden servir de entrada tanto a técnicas de análisis complejas como a otras técnicas de reducción que empiezan a tener problemas a medida que las trayectorias se vuelven más grandes.

Author details
[1]Escuela de Ingeniería de Sistemas y Computación, Universidad del Valle, Santiago de Cali, Colombia. [2]The European Bioinformatics Institute (EMBL-EBI), Hinxton, Cambridgeshire, UK.

References

1. Shaw, D.E., Chao, J.C., Eastwood, M.P., Gagliardo, J., Grossman, J.P., Ho, C.R., Lerardi, D.J., Kolossváry, I., Klepeis, J.L., Layman, T., McLeavey, C., Deneroff, M.M., Moraes, M.A., Mueller, R., Priest, E.C., Shan, Y., Spengler, J., Theobald, M., Towles, B., Wang, S.C., Dror, R.O., Kuskin, J.S., Larson, R.H., Salmon, J.K., Young, C., Batson, B., Bowers, K.J.: Anton, a special-purpose machine for molecular dynamics simulation. Communications of the ACM 51(7), 91 (2008). doi:10.1145/1364782.1364802
2. Lindorff-Larsen, K., Piana, S., Dror, R.O., Shaw, D.E.: How fast-folding proteins fold. Science 334(6055), 517–520 (2011). doi:10.1126/science.1208351. arXiv:1011.1669v3
3. Shaw, D.E., Grossman, J.P., Bank, J.A., Batson, B., Butts, J.A., Chao, J.C., Deneroff, M.M., Dror, R.O., Even, A., Fenton, C.H., Forte, A., Gagliardo, J., Gill, G., Greskamp, B., Ho, C.R., Ierardi, D.J., Iserovich, L., Kuskin, J.S., Larson, R.H., Layman, T., Lee, L.-S., Lerer, A.K., Li, C., Killebrew, D., Mackenzie, K.M., Mok, S.Y.-H., Moraes, M.A., Mueller, R., Nociolo, L.J., Peticolas, J.L., Quan, T., Ramot, D., Salmon, J.K., Scarpazza, D.P., Schafer, U.B., Siddique, N., Snyder, C.W., Spengler, J., Tang, P.T.P., Theobald, M., Toma, H., Towles, B., Vitale, B., Wang, S.C., Young, C.: Anton 2: Raising the Bar for Performance and Programmability in a Special-Purpose Molecular Dynamics Supercomputer. In: Shaw2014 (ed.) SC14: International Conference for High Performance Computing, Networking, Storage and Analysis, pp. 41–53. IEEE, Los Alamitos, CA, USA (2014). doi:10.1109/SC.2014.9. http://ieeexplore.ieee.org/document/7012191/
4. Nguyen, H., Maier, J., Huang, H., Perrone, V., Simmerling, C.: Folding simulations for proteins with diverse topologies are accessible in days with a physics-based force field and implicit solvent. Journal of the American Chemical Society 136(40), 13959–13962 (2014). doi:10.1021/ja5032776
5. Larson, S.M., Snow, C.D., Shirts, M., Pande, V.S.: Folding@Home and Genome@Home: Using distributed computing to tackle previously intractable problems in computational biology (2009). 0901.0866
6. Duan, M., Fan, J., Li, M., Han, L., Huo, S.: Evaluation of Dimensionality-reduction Methods from Peptide Folding-unfolding Simulations. Journal of chemical theory and computation 9(5), 2490–2497 (2013). doi:10.1021/ct400052y
7. Peng, J.-h., Wang, W., Yu, Y.-q., Gu, H.-l., Huang, X.: Clustering algorithms to analyze molecular dynamics simulation trajectories for complex chemical and biological systems. Chinese Journal of Chemical Physics 31(4), 404–420 (2018). doi:10.1063/1674-0068/31/cjcp1806147
8. Rajan, A., Freddolino, P.L., Schulten, K.: Going beyond clustering in MD trajectory analysis: An application to villin headpiece folding. PLoS ONE 5(4), 9890 (2010). doi:10.1371/journal.pone.0009890
9. Das, P., Moll, M., Stamati, H.: Low-dimensional, free-energy landscapes of protein-folding reactions by nonlinear dimensionality reduction. Proceedings of the ... 103(26) (2006)
10. Kim, S.B., Dsilva, C.J., Kevrekidis, I.G., Debenedetti, P.G.: Systematic characterization of protein folding pathways using diffusion maps: Application to Trp-cage miniprotein. The Journal of Chemical Physics 142(8), 85101 (2015). doi:10.1063/1.4913322
11. Doerr, S., Ariz-Extreme, I., Harvey, M.J., De Fabritiis, G.: Dimensionality reduction methods for molecular simulations (2017). 1710.10629
12. Shao, J., Tanner, S.W., Thompson, N., Cheatham, T.E.: Clustering Molecular Dynamics Trajectories: 1. Characterizing the Performance of Different Clustering Algorithms. Journal of chemical theory and computation 3(6), 2312–34 (2007). doi:10.1021/ct700119m
13. Kaufman, L., Rousseeuw, P.: Finding Groups in Data. Wiley-Interscience; New York, ??? (1990)
14. Zhang, Y., Skolnick, J.: Scoring function for automated assessment of protein structure template quality. Proteins: Structure, Function, and Bioinformatics 68(4), 1020 (2007). doi:10.1002/prot.21643
15. Ensign, D.L., Kasson, P.M., Pande, V.S.: Heterogeneity even at the speed limit of folding : Large-scale molecular dynamics study of a fast-folding variant of the villin headpiece. Journal of molecular biology 374(3), 806–816 (2007)
16. Amato, N.M., Tapia, L., Thomas, S.: A Motion Planning Approach to Studying Molecular Motions. Communications in Information and Systems 10(1), 53–68 (2010). doi:10.4310/CIS.2010.v10.n1.a4
17. Xu, J., Zhang, Y.: How significant is a protein structure similarity with TM-score = 0.5? Bioinformatics 26(7), 889–895 (2010). doi:10.1093/bioinformatics/btq066
18. Oksanen, J., Blanchet, F.G., Friendly, M., Kindt, R., Legendre, P., McGlinn, D., Minchin, P.R., O'Hara, R.B., Simpson, G.L., Solymos, P., Stevens, M.H.H., Szoecs, E., Wagner, H.: vegan: Community Ecology Package. (2019). https://cran.r-project.org/package=vegan
19. R Core Team: R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria (2018). R Foundation for Statistical Computing. https://www.r-project.org/
20. Grant, B.J., Rodrigues, A.P.C., ElSawy, K.M., McCammon, J.A., Caves, L.S.D.: Bio3D: An R package for the comparative analysis of protein structures. Bioinformatics 22(21), 2695–2696 (2006). doi:10.1093/bioinformatics/btl461
21. Duan, M., Han, L., Rudolph, L., Huo, S., Carlson, G.H.: Geometric Issues in Dimensionality Reduction and Protein Conformation Space. (2014)