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

In this article 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 which describe the sequence of states that proteins follow as a function of time during their folding process (Figure 1).

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

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.

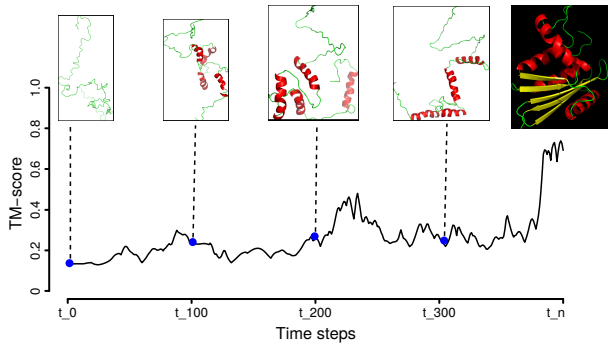


Figure 1 Protein folding trajectory. The evolution of the folding is measured at each step by comparing the current structure (white background images) to the native structure (black background image) using a structure comparison method, in this work we use the TM-Score [# Zhan2004]. Horizontal axis: simulation time from t_0 (initial) to t_n (final). Vertical axis: TM-Score value, from 0 (different) to 1 (similar).

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.

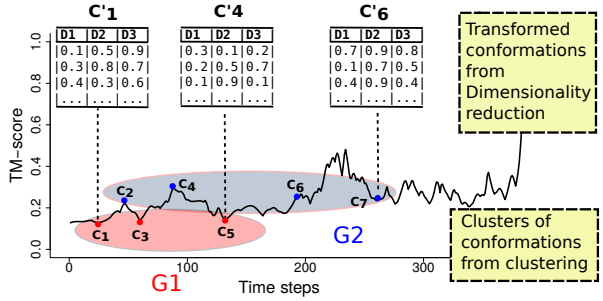


Figure 2 Loss of information in dimensionality reduction and clustering. The dimensionality reduction (above) transforms the conformations (c1, c2 and c3) into a new set of values (c'1, c'2 and, c'3), but their structural information is lost. While the clustering (bottom) forms two groups (G1 and G2),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

Our algorithm reduces a folding trajectory in three stages (Figure 3): (1) Trajectory partition, which divides the trajectory into segments with equal number of conformations; (2) Characteristics extraction, which quickly extracts the set of characteristic conformations for each segment by following the rapid clustering approach of Hobohm and Sander (1992); and (3) Representatives selection, which selects the most representative conformations from each of the previous sets by following a strategy of type k-medoids [13]. At the end, the results of each segment are joined to form the reduced trajectory containing the most representative conformations of the original trajectory, preserving their 3D representation and their temporal order.

The algorithm is implemented in the R language, except the function for pair-wise structure comparison, which is the function executed more times and that is implemented in the Fortran language. For structure pair-wise comparison, the algorithm uses the TM-Score metric [14].

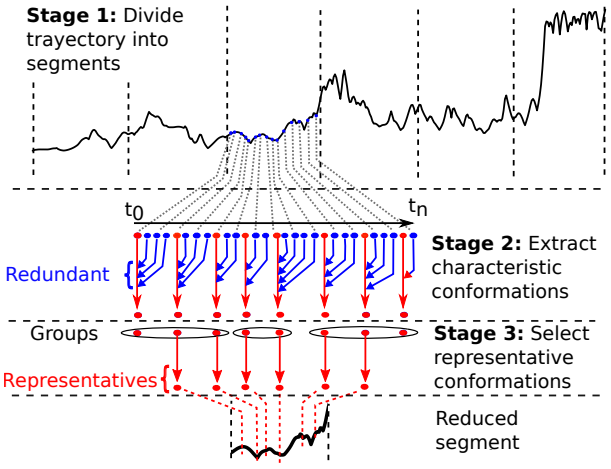


Figure 3 Proposed reduction algorithm. Stage 1: Trajectory is divided into segments (dashed lines) and each is reduced individually. Stage 2: For each segment, characteristic conformations are extracted (red dots) and the redundant ones are ignored (blue dots). Stage 3: The most representatives are selected from the previous conformations.

Methods

Protein Datasets

To evaluate the results and performance of the proposed algorithm, we used the trajectories of three proteins: Trp-cage, villin-headpiece, and Ribonuclease H. The Trp-cage trajectory was simulated with molecular dynamics using the Anton Supercomputer [2], with a simulation time of 208 μs, a 200 ps time step, and 1044001 conformations. The villin-headpiece trajectory was simulated with molecular dynamics using the folding@home distributed computing platform [15], with a simulation time of 8 μs, a 50 ps time step, and 15201 conformations. And the Ribonuclease H trajectory was simulated with the Probabilistic Roadmap Method (PRM) [16], with 429 folding steps or unfolding events (the PRM is an unfolding method and uses folding steps instead of time steps).

Comparaciones con nMDS, PCA, y clustering

Comparamos los resultados de nuestro algoritmo frente a los resultados de tres métodos comúnmente utilizados en reducción de trayectorias de plegamiento [11]: escalamiento multidimensional no-métrico (nMDS), análisis de componentes principales (PCA), y agrupamientos (Figura **??**). Para la reducción con nMDS, calculamos la matriz de *disimilaridades* entre las conformaciones mediante la métrica TM-score [14], con esta matriz calculamos los nuevos puntos para un espacio geométrico de 2D mediante la función *monoMDS* del paquete *vegan* del sistema R [17], y los desplegamos sobre un plano 2D. Para la reducción con PCA caracterizamos cada conformación con las coordenadas XYZ de sus átomos, calculamos los componentes principales mediante la función *pca.xyz* del paquete *Bio3D* del sistema R [18], y seleccionamos los dos primeros componentes que explican la mayor varianza. Y para el agrupamiento, caracterizamos cada conformación con sus dos primeros componentes principales y realizamos un agrupamiento jerárquico con el método *complete linkage* de la función *hclust* del paquete *stats* del sistema R [19]. El número de grupos k=7 lo seleccionamos mediante un enfoque de promedios Silhouette al variar k desde 1 hasta 10 utilizando la función *fviz_nbclust* del paquete *factoextra* del sistema R [20].

Implementación

El algoritmo reduce una trayectoria de plegamiento de proteínas en tres fases: particionamiento, selección, y extracción (Figura 3). Cada fase conlleva una estrategia para mejorar la eficiencia del algoritmo cuando las trayectorias de plegamiento son muy grandes.

Fase 1: Particionamiento y Paralelización

Dividimos la trayectoria en subtrayectorias con el objetivo de reducirlas de forma independiente y paralela (líneas verticales punteadas, Figura 3).

Esta estrategia de particionamiento tiene un doble objetivo: primero, reducir localmente cada subtrayectoria y así enfocarnos en sus características particulares, lo que al final resulta en la obtención de las características globales de toda la trayectoria. Y segundo, que sus reducciones se puedan realizar en paralelo y así mejorar notoriamente la eficiencia del algoritmo a la hora de ejecutarlo en una máquina con tecnología multi-core (Figura 4).

Fase 2: Extracción y Filtración

Esta fase del algoritmo extrae rápidamente de cada subtrayectoria las conformaciones características y filtra las redundantes. Para hacerlo de manera eficiente, modificamos la estrategia de agrupamiento rápido de Hobohm and Sander (1992) para trabajar con estructuras de proteínas y en vez agruparlas busque las más disimilares.

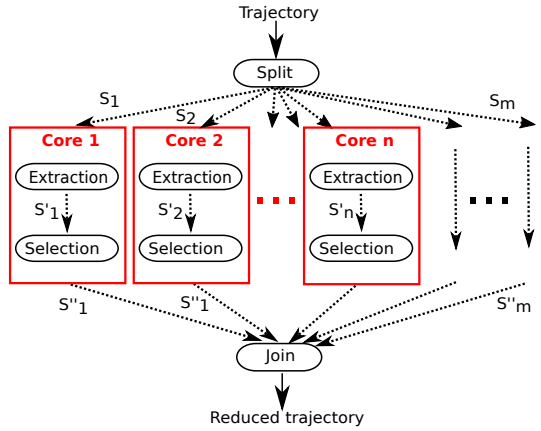


Figure 4 Parallel Reduction. La reducción de cada subtrayectoria S1...Sm se ejecuta en paralelo aprovechando la tecnología multi-core de las máquinas actuales. Los resultados tanto del proceso de selección de características (S'1...S'm) como los del de extracción de representativas (S''1...S''m) son independientes de los de las otras subtrayectorias. Al final los resultados de cada procesamiento se unen para obtener la trayectoria total reducida.

El algoritmo aprovecha el orden temporal implícito en las subtrayectorias para organizar las conformaciones en orden creciente de tiempo de simulación (flecha horizontal negra, Figura 3). Se asigna la primera conformación como la primera representante característica (punto rojo en t0, Figura 3), y se toma la siguiente conformación y se comparan. Si son diferentes, entonces se convierte en una nueva representante (puntos rojos, Figura 3), de lo contrario es redundante y se filtra (puntos azules, Figura 3). Después, se toma la siguiente conformación y se continua el mismo proceso hasta terminar con todas.

Fase 3: Búsqueda y Selección

Esta última fase del algoritmo toma como entrada las conformaciones características de la fase anterior y realiza una búsqueda completa de las conformaciones que más las representen. Hablamos de completa porque la búsqueda implica comparar todas las conformaciones entre si, es decir calcular su matriz de disimilaridades. Por esta razón es que esta búsqueda es factible hacerla ahora y no antes ya que se hace sobre un conjunto mucho menor de conformaciones que el que se tiene al inicio por cada subtrayectoria, .

Para encontrar estas conformaciones representantes calculamos las k conformaciones cuya disimilaridad media a todas las demás integrantes del grupo es mínima, lo que se conoce como *medoides* y el algoritmo que usamos para realizar esto es el particionamiento alrededor de medoides PAM [13].

Esta fase necesita tres datos de entrada: el conjunto de conformaciones características de la fase anterior (C), el umbral mínimo de TM-score para aceptar dos conformaciones como similares (T), y el número deseado de representantes seleccionadas (K).

Comparación entre Estructuras de Proteínas

Para la comparación de las estructuras de las conformaciones utilizamos la métrica TM-score (Template Modeling score) [14]. El TM-score es más preciso que otras métricas usadas en comparación de estructuras, como el Root Mean Square-Deviation (RMSD), ya que es más robusta a variaciones locales.

Nuestro algoritmo requiere como uno de sus parámetro de entrada un valor de puntaje mínimo de TM-score para aceptar como similares a dos conformaciones. Este parámetro se usa después tanto en la fase de extracción de características, al comparar las conformaciones para encontrar disimilares y remover redundantes, como en la fase de selección de representativas, al calcular la matriz de distancias de todas las conformaciones.

Para tener una aproximación del rango de valores de este puntaje mínimo, los puntajes del TM-score varian de 0 a 1, donde 1 indica un emparejamiento perfecto. Además, las estadísticas hechas por sus autores [21] muestran que un puntaje < 0.17 indica dos estructuras aleatorias, sin relación de similaridad, y un puntaje > 0.5 indica que las estructuras tienen un grado de similaridad que no está dado por el azar.

Resultados y Discusión

Reducciones de tres trayectorias de plegamiento

Realizamos la reducción de las trayectorias de tres proteínas tomadas de distintos proyectos de simulación: Trp-cage (Supercomputador Anton [2], villin-headpiece (folding@home [15]), y la Ribonuclease H (Folding server [16]) (ver detalles de las simulaciones en la sección de Métodos). Los resultados se muestran en la figura 5 donde se presenta para cada proteína en el recuadro izquierdo sus detalles, y en el derecho sus dos trayectorias: la original (arriba) y la reducida (abajo).

Como se puede observar de la figura 5, los resultados de las reducciones son conformaciones de la misma trayectoria, las cuales siguen conservando tanto su estructura como su orden temporal. Este resultado es

importante ya que estas reducciones, al ser un resumen de la trayectoria original, se pueden usar enteramente como entrada para análisis más complejos que pueden volverse imprácticos cuando tratan con trayectorias muy grandes. Otros técnicas de reducción usadas en análisis de trayectorias o bien transforman las conformaciones en estructuras de menos dimensiones, solo interpretables cuando se observan en conjunto, como el caso de MDS, Isomap, y diffusion maps [22, 10]; o crean grupos de ellas que resaltan alguna

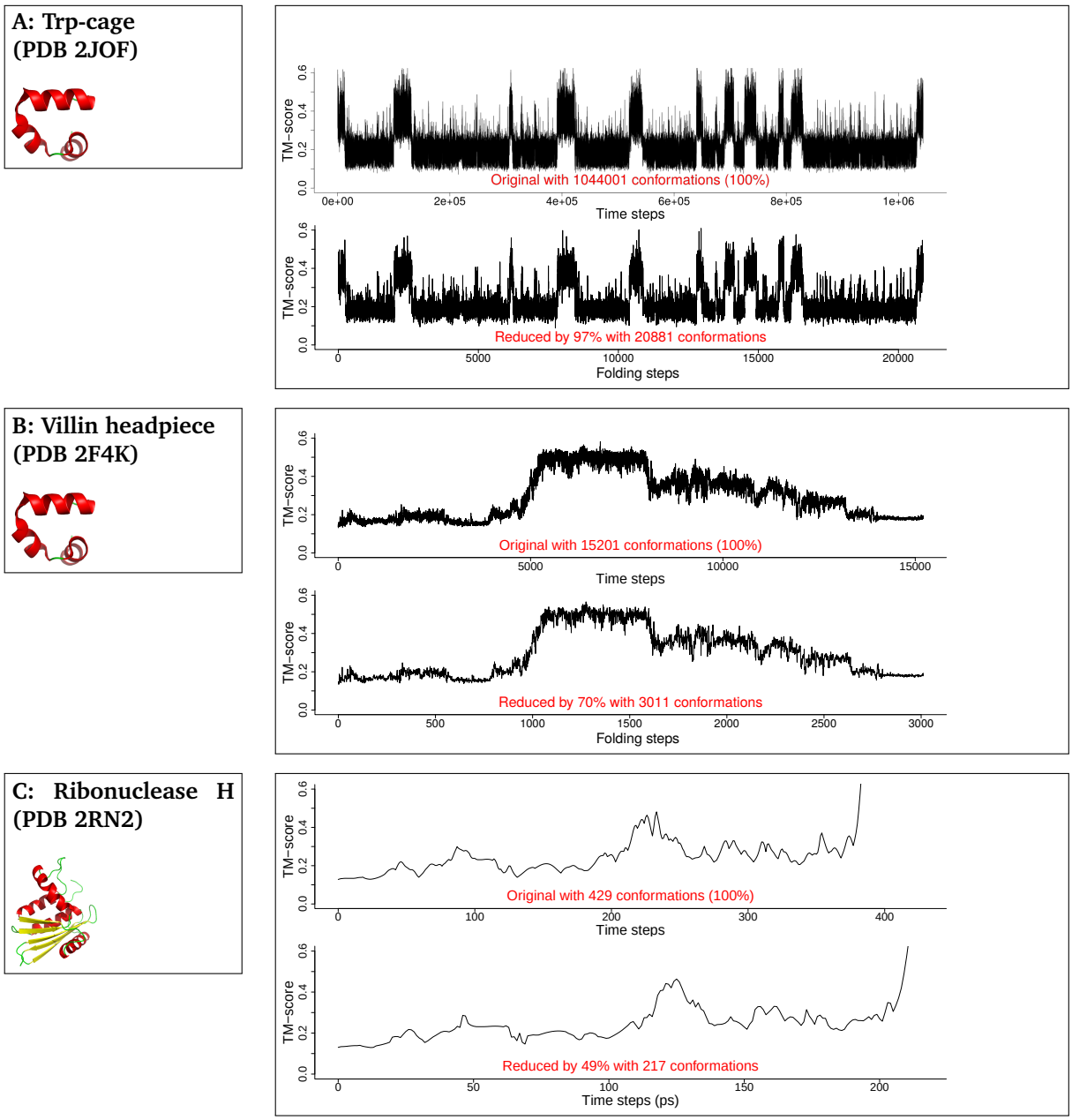


Figure 5 Reducciones de tres trayectorias de plegamiento. Recuadro izquierdo: nombre, código PDB y estructura nativa de la proteína. Recuadro derecho: trayectoria original en la parte de arriba y la trayectoria reducida en la parte inferior. La trayectoria original muestra en el eje horizontal los pasos de tiempo en que ocurren las conformaciones en la simulación, mientras que la trayectoria reducida muestra los pasos de plegamiento o secuencia de eventos que resumen la trayectoria. (A) Trayectoria de la proteína Trp-cage reducida en un 98%, de 1044001 a 20881 conformaciones. (B) Trayectoriade la proteína villin-headpiece reducida en un 80%, de 15201 a 3011conformaciones. (C) Trayectoria de la proteína Ribonuclease H reducida en un 49%, de 429 a 217 conformaciones.

similaridad ya sea estructural o energética, sin importar su orden temporal, como en el caso de los agrupamientos [7]. Además, debido a que varias de estas técnicas se basan en el cálculo de las distancias entre pares de conformaciones, el alto costo computacional de realizar esos cálculos para millones o incluso miles de conformaciones, las puede volver imprácticas sino se utilizan trayectorias reducidas como las que produce nuestro algoritmo.

Sin embargo, aunque las conformaciones de las trayectorias reducidas conservan el orden temporal que tienen en la trayectoria original, el tiempo de simulación en que suceden no se conserva explícitamente. Es decir, las reducciones no describen pasos de tiempo sino pasos de plegamiento, que se refieren a la secuencia de eventos destacados que resumen el plegamiento de la proteína y no al tiempo exacto en que estos ocurren. No obstante, para obtener estos tiempos, se puede tomar el nombre o identificador de la conformación de interés en la trayectoria reducida y localizar su tiempo en la trayectoria original.

Comparación frente a otros métodos de reducción

Para 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 6, 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.

Observamos que las reducciones de la trayectoria original producen un despliegue en 2D característico en ambos métodos de reducción: un patrón de círculos de puntos, para el nMDS; y una estructura de 7 grupos, para el agrupamiento por PCA (fila superior, figura 6). Así mismo, este mismo despliegue se repite en gran medida en las dos reducciones calculadas por nuestro algoritmo, la de compresión media del 52% y la de compresión alta del 80% (filas central e inferior de la figura 6, respectivamente).

Lo anterior nos indica que nuestras reducciones preservan en gran medida los eventos principales de la trayectoria al observar que tanto las reducciones con nMDS y PCA siguen conservando el mismo patrón y la misma estructura de grupos. Además, nuestro algoritmo presenta ventajas adicionales sobre los

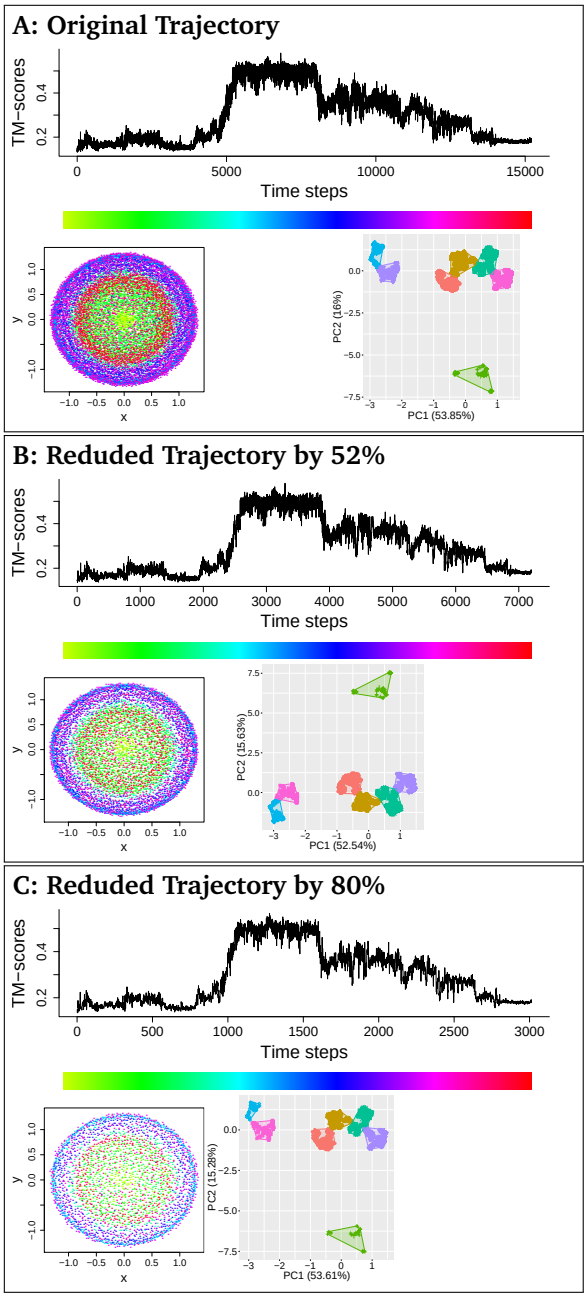


Figure 6 Comparación frente a otros métodos.

otros métodos de reducción. Primero, el cálculo de las reducciones es más eficiente que el de nMDS ya que no necesita la matriz de disimilaridades, que es sumamente costosa de calcular cuando el número de conformaciones es grande. Segundo, la interpretación de los resultados es directa ya que los resultados son conformaciones de la proteína y no transformaciones de los datos, como en el caso del nMDS y PCA, o grupos de conformaciones, como en el caso de los agrupamientos. Y tercero, el orden temporal se conserva ya

que el resultado es una nueva trayectoria, a diferencia del agrupamiento en donde los grupos resultantes pueden contener conformaciones que ocurren en tiempos muy distintos.

Desempeño del algoritmo

El desempeño de nuestro algoritmo lo evaluamos en dos situaciones: comparándolo frente a otros métodos de reducción (Figura 7) y ejecutándolo en paralelo usando múltiples núcleos de procesamiento (Figura 8). 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 7 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 algoritmo en paralelo con 2 cores (FR2, línea azul), este se vuelve más eficiente que PCA.

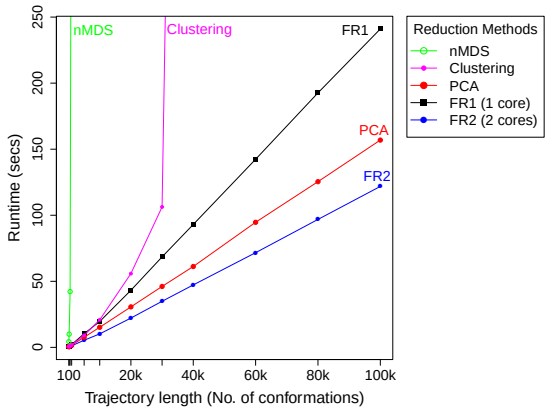
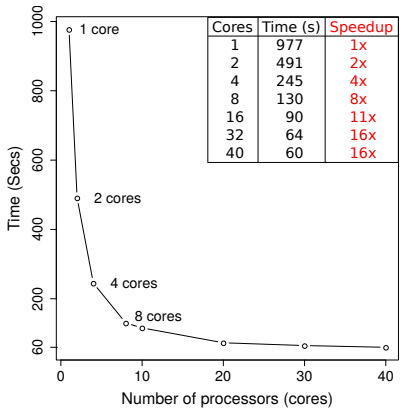


Figure 7 Desempeño del algoritmo frente a otros métodos. Comparación del nuestro algoritmo FR1 con nMDS, PCA, y agrupamiento. PCA y FR1 son los más eficientes, pero si nuestro algoritmo utiliza dos núcleos (FR2), el tiempo se disminuye a la mitad y se vuelve más eficiente que PCA. Por el contrario, nMDS y clustering toman demasiado tiempo, aún con trayectorias pequeñas.

Este comportamiento lo podemos ver más claramente en la figura 8, 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.



Cores	Time (s)	Speedup
1	977	1x
2	491	2x
4	245	4x
8	130	8x
16	90	11x
32	64	16x
40	60	16x

Figure 8 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 exten-

sas, 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. 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
18. 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
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. Kassambara, A., Mundt, F.: factoextra: Extract and Visualize the Results of Multivariate Data Analyses. (2017). https://cran.r-project.org/package=factoextra
21. 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
22. Duan, M., Han, L., Rudolph, L., Huo, S., Carlson, G.H.: Geometric Issues in Dimensionality Reduction and Protein Conformation Space. (2014)