JCTC Journal of Chemical Theory and Computation

Clustering Molecular Dynamics Trajectories: 1. Characterizing the Performance of Different Clustering Algorithms

Jianyin Shao, Stephen W. Tanner,† Nephi Thompson,‡ and Thomas E. Cheatham, III*

Departments of Medicinal Chemistry, Pharmaceutics and Pharmaceutical Chemistry, and Bioengineering, College of Pharmacy, University of Utah, 2000 East 30 South, Skaggs Hall 201, Salt Lake City, Utah 84112

Received May 17, 2007

Abstract: Molecular dynamics simulation methods produce trajectories of atomic positions (and optionally velocities and energies) as a function of time and provide a representation of the sampling of a given molecule's energetically accessible conformational ensemble. As simulations on the 10−100 ns time scale become routine, with sampled configurations stored on the picosecond time scale, such trajectories contain large amounts of data. Data-mining techniques, like clustering, provide one means to group and make sense of the information in the trajectory. In this work, several clustering algorithms were implemented, compared, and utilized to understand MD trajectory data. The development of the algorithms into a freely available C code library, and their application to a simple test example of random (or systematically placed) points in a 2D plane (where the pairwise metric is the distance between points) provide a means to understand the relative performance. Eleven different clustering algorithms were developed, ranging from top-down splitting (hierarchical) and bottom-up aggregating (including single-linkage edge joining, centroid-linkage, average-linkage, complete-linkage, centripetal, and centripetal-complete) to various refinement (means, Bayesian, and self-organizing maps) and tree (COBWEB) algorithms. Systematic testing in the context of MD simulation of various DNA systems (including DNA single strands and the interaction of a minor groove binding drug DB226 with a DNA hairpin) allows a more direct assessment of the relative merits of the distinct clustering algorithms. Additionally, means to assess the relative performance and differences between the algorithms, to dynamically select the initial cluster count, and to achieve faster data mining by "sieved clustering" were evaluated. Overall, it was found that there is no one perfect "one size fits all" algorithm for clustering MD trajectories and that the results strongly depend on the choice of atoms for the pairwise comparison. Some algorithms tend to produce homogeneously sized clusters, whereas others have a tendency to produce singleton clusters. Issues related to the choice of a pairwise metric, clustering metrics, which atom selection is used for the comparison, and about the relative performance are discussed. Overall, the best performance was observed with the average-linkage, means, and SOM algorithms. If the cluster count is not known in advance, the hierarchical or average-linkage clustering algorithms are recommended. Although these algorithms perform well, it is important to be aware of the limitations or weaknesses of each algorithm, specifically the high sensitivity to outliers with hierarchical, the tendency to generate homogenously sized clusters with means, and the tendency to produce small or singleton clusters with average-linkage.

Introduction

Molecular dynamics (MD) and free energy simulation methods provide valuable insight into the structure, dynamics, and interactions of biological macromolecules.[1−4] Over the past three decades, MD simulation methods have proven to be an accurate tool for probing the detailed atomistic dynamics of models of biological systems on the picosecond to microsecond time scales.[5−14] MD simulations give direct insight into protein folding,[8,15−29] drug-receptor interaction,[3,30−35] and fast time scale motions of biological molecules.[36−47] As computer power continues to increase, and

* Corresponding author e-mail: tec3@utah.edu.
† Current address: Bioinformatics Program, University of California San Diego, La Jolla, CA 92093.
‡ Current address: Department of Physics, Wright State University, 248 Fawcett Hall, 3640 Colonel Glenn Hwy, Dayton, OH 45435.

10.1021/ct700119m CCC: $37.00
Published on Web 10/06/2007

simulations on the 10−100 ns time scale and beyond become routine, large amounts of data result. This sequence of data—the "MD trajectory"—fully specifies the history of the atomic motions in terms of a sequential time-dependent set of molecular configurations from the MD simulation and the larger set of derived properties calculated from the MD trajectory (such as energies, bond lengths, and angle distributions). These data not only provide insight into the structure, dynamics, and interactions of the biomolecules under study but also can be reused to score putative force field changes, as a set of "good" and "bad" representative structures sampled, and for the development of coarse-grained potentials. Although many of the properties derived from the MD trajectory are rather easy to extract, such as the time evolved root-mean-squared coordinate deviation (RMSd) to the initial structure or various distance and angle time series, some properties are more difficult to extract and may be significantly more time-consuming to evaluate (such as entropies and heat capacities). Further, even with elucidation of these properties, often the inherent relationships among the molecular configurations are hidden in the complexity of the data. One very useful way to expose some of these correlations is to group or cluster molecular configurations into subsets based on the similarity of their conformations (as measured by an appropriate metric).[48,49] Clustering is a general data-mining technique that can be applied to any collection of data elements (points) where a function measuring distance between pairs of points is available.[50,51] A clustering algorithm partitions the data points into a disjoint collection of sets called clusters. The points in one cluster are ideally closer, or more similar, to each other than to points from other clusters. In this work, we describe the implementation and application of a variety of well-known pairwise distance metric clustering algorithms into a general purpose (and freely available) C code library. To test and validate the implementations, a simple problem is the clustering of randomly (or systematically) placed points in the Euclidean plane where the pairwise metric is the distance between points. This provides an easy way, using the discrimination of our visual system, to *see* the results and to highlight bugs in the implementations. This contrived test system also nicely highlights the underlying limitations of each algorithm. After description of the algorithms and their relative performance, the clustering methods are then applied to a series of MD trajectories of various biomolecular systems.

The use of clustering algorithms to group together similar conformations visited during a MD simulation is not a novel concept.[48,49,52] A wide variety of algorithms has been applied in many studies to cluster molecular dynamics trajectories, group similar conformations, and otherwise search for similarities among structures. A subset of publications developing and applying clustering algorithms to analyze molecular dynamics trajectories spans the range from some of the earliest MD simulations to very recent studies.[48,49,52−75] In this work we build on the previous studies by comparing and contrasting the performance of various well-known clustering algorithms applied to the points in a plane example and multiple different sets of MD simulation data. The algorithms implemented include *top-down/divisive* (**hierarchical**), *bottom-up/agglomerative* (single-linkage/**edge**-joining, **centripetal**, **complete-linkage**, **centroid-linkage**, **average-linkage**, and **centripetal-complete**), *refinement* (**means**, **Bayesian**, and self-organizing maps or **SOM**), and *tree* clustering (**COBWEB**) algorithms. The choice of biomolecular systems to cluster includes MD simulation studies of a dynamic 10-mer polyadenine DNA single strand in aqueous solution, the interaction of the minor groove binding drug DB226 (the 3-pentyl derivative of 2,5-bis(4-guanylphenyl)furan) with a DNA hairpin loop in two different binding modes, and the conformational transition from an open to closed geometry of an drug-free cytochrome P450 2B4 structure (PDB: 1PO5).[76] In addition to the raw or production MD trajectory data, two artificial sets of data were constructed from independent trajectories of the polyA single strand to create trajectories containing 500 configurations at 1 ps intervals. The first represents five equally sized clusters created from 100 ps MD sampling around five distinct starting conformations, and the second is created from sampling around five distinct conformations to create clusters of different sizes, specifically containing 2, 15, 50, 100, or 333 configurations each.

When clustering the molecular configurations from a MD trajectory, ideally each clustering algorithm should group similar molecular configurations into distinct sets or groups. This gives a refined view of how a given molecule is sampling conformational space and allows direct characterization of the separate conformational substates visited by the MD.[77] As large-scale conformational change during the MD can lead to high variance for the calculation of time independent properties, such as MM-PBSA estimates of free energetics[3,78] or covariance estimates of the entropy,[79,80] it is expected that clustering of the trajectory into distinct substate populations can minimize this variance and provide more useful information about the ensemble of conformations sampled by MD. Clustering—no matter how valid in terms of its algorithmic success and ability to discern—is only useful if it can provide an unbiased means of exposing significant relationships and differences in the underlying properties. Ultimately, it is desired that an algorithm will naturally partition the data—with minimal user input—into representative clusters where each cluster may have different shapes, different variance, and different sizes. For example, structures sampled from a deep and narrow minimum energy well will typically have a smaller variance than those sampled from more flat and higher entropy wells. Clusters of configurations from MD simulation are also likely to have different sizes as sampling should ultimately progress according to a Boltzmann distribution, and, therefore, higher energy substates will be less populated than lower energy substates. In practice, except with artificially constructed and well-separated data, the performance of the underlying algorithm depends critically on the data, the pairwise comparison metric, the choice of atoms used in the comparison, and the choice of cluster count. With proper usage, we found that the clustering algorithms do seem to capture conformational substates of interest and that the clustering results highlight the similarity and differences among the

structures. However, some of the clustering algorithms have key limitations and hence are not recommended for clustering MD trajectory data. Moreover, there appears to be no "one size fits all" clustering algorithm that always does an appropriate job of grouping the molecular configurations; in other words, the clustering algorithm ideally suited for clustering a particular data set will depend on the data.

To better characterize the relative performance, we implemented a range of different clustering algorithms. Assessment was made via visual inspection of the resulting clusters and also through the use of various clustering metrics. The algorithms chosen vary widely in their approaches, their computational complexity, their sensitivity to outliers, and their overall effectiveness. Our examination of several rather different clustering algorithms allowed us to quantitatively assess the quality of their output as well as their overall similarities and limitations. Surprisingly different behavior was observed in application of the different algorithms to the same MD simulation data. Whereas the fast and top-down divisive (or **hierarchical**) clustering algorithm tends to produce uniformly sized clusters or clusters with similar diameters, across the set of molecular configurations, various implementations of the bottom-up "merging" (or **linkage**) clustering algorithms tended to group most of the molecular configurations into a single large cluster with small singleton cluster outliers that contained only one or a few molecular configurations. Although the merging algorithms may produce singleton clusters, these algorithms can form clusters of any "shape" (such as elongated or concave clusters) in contrast to the **hierarchical** clustering algorithm. Depending on the data set, cluster count, and metric, differences in the relative performance of the various algorithms are clearly evident. The observation that clustering depends on the choice of algorithm strongly justifies the exploration of multiple clustering algorithms when initially characterizing the MD trajectory data. In addition to multiple algorithms, users need guidance on choosing the appropriate cluster count and atoms to use for the pairwise comparison. In general, the appropriate choices that will best partition the data are not known in advance. Strategies to assess the proper cluster count include dynamically choosing the number of clusters based on quantitative measures of clustering quality. Metrics investigated in this work include the pseudo F-statistic (pSF), the Davies-Bouldin index (DBI), the SSR/SST ratio, and the "critical distance". These indices and the detailed progression of the partitioning or merging can be cached, thereby allowing characterization of clustering performance across a range of cluster counts in a single clustering run. Further information about the relative performance and optimal choice of cluster count can come from visual examination of the tree of clusters. Finally, a significant concern when clustering based on pairwise distance evaluations is that the computational costs rapidly become excessive as the number of conformations to cluster grows. To partially mitigate this N^2 growth in computational costs and memory requirements, we implemented a two-pass "sieved" approach as a way to efficiently cluster many thousands of points. The MD trajectory is scanned first at a coarse level to do the initial clustering, with a second pass through the data to add skipped configurations to existing clusters. We examine the usefulness and limitations of these approaches.

Methods

Eleven different cluster algorithms[51] were implemented: **hierarchical**, single linkage (**edge**), average linkage (**average**), centroid linkage (**linkage**), complete linkage (**complete**), K-means (**means**), **centripetal**, **centripetal-complete**, **COBWEB**,[81,82] **Bayesian**,[83] and self-organizing maps (**SOM**).[84] These were implemented in a library written in C, *libcluster*, which works abstractly on points and pairwise distances. It is not specific to MD simulations. By extending a few functions, such as the one that computes the centroid of a cluster, one can use this library to cluster arbitrary types of data. For instance, a separate program we developed, called *ClusterTest*, invokes *libcluster* to cluster collections of points in the plane. The *ClusterTest* utility can also measure distances between clustering outputs. In application to MD simulation, particular care needs be levied in calculation of the cluster centroid. Specifically, this relates to deciding the frame of reference for the averaging of conformations that form the centroid. In our initial development, the centroids produced were misleading as the molecules moved during the MD simulation and were not necessarily in the same reference frame as their centroid. To circumvent this problem, prior to construction of the centroid either the sampled configurations need to be placed into a common reference frame (such as by an RMSd fit to the first frame or a representative structure), or, better, separate reference frames should be created for each cluster where the frame of reference is the most representative configuration from that cluster. In the current implementation, all the structures in a given cluster are rms fit to the most representative structure before calculation of the centroid. The representative structure is the structure which has the minimal sum of the squared displacements between other structures in the cluster and itself. This is stored internally as the "bestrep" structure, and at present this is not necessarily equivalent to the representative structure output by **ptraj** (which currently writes out the structure closest to the centroid).

Code and Interface. All programs are written in portable C code and are available from the authors. For clustering MD trajectory data, this library was interfaced to the **ptraj** module of Amber.[85,86] We measured the distance between frames using mass-weighted, optimal-coordinate superposition root-mean-squared deviation (RMSd) or by using the distance measure D_{ab} (DME) defined by Torda and van Gunsteren.[52] Users can choose the subset of atoms to be used for pairwise comparison, specify the clustering algorithm and cluster count, and request to output new trajectories for each cluster, average structures for each cluster, and/or representative structures for each cluster. In this paper, each reference to distance indicates the RMSd between two simulation snapshots (i.e., two molecular configurations from different time points from the MD trajectory) unless otherwise specified.

Testing of the Implementation Using Points on a 2D Plane. To aid our analysis and algorithm development, we

Figure 1. Clustering on a simple data set of points on a 2D plane. This data set is simple for several reasons: each cluster (represented by a different color) is convex, the clusters are clearly separated, nearby clusters do not differ greatly in size, and we requested the "right" number (three) of clusters when clustering. Changing any of these conditions will cause problems for some of the clustering algorithms.

utilized a very simple test set for evaluating the performance of the various clustering algorithms. This test uses points in the Euclidian plane where the pairwise metric is simply the distance between the points. Using either systematically placed (as shown in Figure 1) or randomly distributed points, it is very easy to construct and visualize the test set. As the data are somewhat contrived and not fully representative of the 3D configurations sampled in a MD trajectory, an algorithm's good performance on the points in Euclidean space example does not guarantee it will classify simulation frames usefully. However, many properties—such as the relative memory requirements, the inability to generate concave clusters, or the sensitivity of **edge**-joining clustering to outliers—remain the same for any problem domain and are most easily discovered and visualized on a simple problem space. For simple test cases, where clusters are convex and clearly separated, all the algorithms perform equally well (see Figure 1). In other cases, the commonly applied algorithms (such as **hierarchical** clustering) break down.

In the following, we provide a general discussion of each of the clustering algorithms that were implemented. These common algorithms, or variants thereof, are classified as algorithms that are top-down (starting from a single cluster or divisive), bottom-up (starting from many clusters and merging or agglomerative), refinement (iteratively refining the membership of clusters starting from seed clusters), or tree based. A brief heuristic explanation of each is provided. A more technical description of each algorithm is provided in the Supporting Information.

Top-Down Clustering Algorithms. Top-down algorithms begin by assigning all points to one large cluster. They then proceed iteratively, splitting a large cluster into two subclusters at each stage. The cluster count increases by one at each step until the desired number of clusters is reached. **Hierarchical** clustering is the only top-down clustering algorithm we implemented. In our implementation, we defined the diameter of a cluster to be the maximal distance between any two points in the cluster. At each cycle, we find the cluster with greatest diameter. We split it around the two eccentric points that define the diameter, A and B: all points closest to A are assigned to one child cluster, and all points closest to B fall in the other.[52]

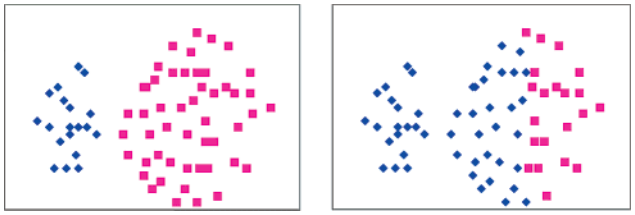

Figure 2. **Hierarchical** clustering (right) produces a nonintuitive clustering of the two distinct sets of points in the plane compared to the other algorithms (**centroid-linkage** clustering is shown on the left).

Hierarchical clustering tends to give clumsy results particularly near the boundary equidistant from the two eccentric points. Each "cut" made in hierarchical clustering may separate points near the boundary from their nearest neighbors. Hierarchical clustering can produce clusters of different population sizes (i.e., some with few points, some with many) but cannot produce clusters of greatly different diameters, such as might correspond to local energy minima of different depths (see Figure 2). This may or may not be mitigated by alteration of the refinement steps in the hierarchical algorithm to be cleverer about the "cut". As implemented, in each of the refinement steps the cluster centroids are calculated, and the points are reassigned between the two new clusters. As will be seen later, this behavior differs from the refinement algorithms where reassignment of points can occur over all the clusters. This implies that the algorithm cannot overcome mistakes in partitioning made in previous steps. Hierarchical clustering is also sensitive to outliers since repositioning an extreme point changes the location of a boundary, and hierarchical clustering cannot produce concave clusters. Its main strengths are that it is the fastest of the clustering algorithms we examined at low cluster counts and changes in the performance metrics as a function of cluster count, such as the variance explained by the data or distance between split clusters, are easy to interpret.

Bottom-Up Clustering Algorithms. Bottom-up algorithms begin by assigning each point to its own cluster and proceed by iteratively merging clusters, one merge at each stage, into larger clusters until the desired number of clusters remains. Algorithmic differences relate to the specific choice of which pair of clusters to merge and the definition of the intercluster distance. **Edge or single-linkage**, **centroid-linkage**, **average-linkage**, **complete-linkage**, **centripetal**, and **centripetal-complete** clustering are the bottom-up algorithms implemented in this work. Bottom-up algorithms, like top-down algorithms, can produce a tree of clusters, where each "leaf" is a cluster, and the "root" is the cluster containing all points. An advantage of these methods is that the cluster merging information can be saved at each step to provide in a single run the set of distinct clusters that result across a range of cluster counts. Examination of these data in terms of the performance metrics can guide users to the appropriate cluster count for the data.

Edge. Under the single-linkage or "edge-joining" or **edge** algorithm, the distance from one cluster to another is defined as the shortest intercluster point-to-point distance. At each iteration step, the two closest clusters are merged. This

merging continues until the desired number of clusters is obtained.[52] Centroid-linkage (or **linkage**) clustering is similar to single-linkage, except that the cluster-to-cluster distance is defined as the distance between the cluster centroids. **Average-linkage** (or **average**) clustering is also similar, except that the cluster-to-cluster distance is defined as the average of all distances between individual points of the two clusters. Complete-linkage (or **complete**) clustering defines the cluster-to-cluster distance as the maximal point-to-point intercluster distance between the two clusters. **Centripetal clustering** is derived from the CURE algorithm.[87] In centripetal clustering, we choose up to five "representatives" for each cluster. Representatives are taken as follows: choose up to five maximally distant points from the cluster and then move each point 1/4 of the way closer to the centroid to produce our representatives. This "centripetal" motion toward the centroid is intended to make the algorithm less sensitive to outliers. At each iteration step, the pair of clusters with the closest representatives is merged, and new representatives are chosen for the resulting larger cluster. The choice of five representatives and movement 1/4 of the way to the centroid is somewhat arbitrary. The centripetal clustering algorithms merging process is depicted graphically in Figure S0 of the Supporting Information.

Centripetal-complete is a variation on the centripetal algorithm that defines the distance between two clusters to be the largest distance ("complete") between the pairs of representative points from the two clusters (rather than the edge distance).

Refinement Clustering Algorithms. Refinement algorithms start with "seed" clusters. These seed clusters are refined, or "trained", over the course of one or more iterations through all the data points. After the clustering is determined to be good enough, or stable enough, the resulting clusters are saved. The number of clusters to form is set at the beginning and generally does not change during the refinement. These algorithms tend to depend on data presentation order and definition of the seed clusters. In our development, we evaluated the effect of the random (seed) and data order factors through multiple runs with different random seeds and comparison of chronological versus random ordering of the MD data. Means, Bayesian, and self-organizing maps are all refining algorithms. **Means** clustering starts by choosing a collection of seed points, each of which is assigned to its own cluster. We then iterate over all other data points. Each data point is assigned to the cluster whose centroid is closest; the centroid for this cluster is then recomputed.[88] To provide greater consistency between runs, we choose as our initial points a collection of maximally distant seed points, although random collections can also be used. **Bayesian clustering** starts with randomized seed clusters. A seed cluster has a random mean (and standard deviation) for each coordinate. Clusters are refined using an expectation-maximization (EM) algorithm. Points have probabilistic membership in each cluster. We first compute the odds that each point is in each cluster (the "expectation" step) and then alter the mean and standard deviation in the clusters to maximize the utility of each (the "maximization" step). This is based on the AUTOCLASS clustering algorithm.[88] In our experience, a large series of repetitive runs with different seeds need to be performed to get consistent results. **SOM**: Self-organizing maps are a form of artificial neural network. Each cluster is seeded with a random point, and the clusters are set up in a simple topology where each cluster has some "neighbor" clusters. The system is then run through several training cycles on the data. To process a data point, the most similar (closest) cluster is chosen. The coordinates of that cluster (and, to a lesser extent, its neighbors) are then shifted toward those of the training data point.[89]

Other Clustering Approaches. The **COBWEB**[81, 82] clustering system produces a tree describing the hierarchical relationships of members to their clusters. Each leaf node corresponds to a single point (or in the case of MD simulation, a single conformation or frame from the MD trajectory), and nonleaf nodes are clusters of all the descendant points. Points are placed in the tree by maximizing category utility (CU), a metric of cluster quality. Category utility is large for a cluster when the standard deviation of an attribute (over all points in the cluster) is smaller than the standard deviation of that same attribute in the cluster's parent.[88] Because of its unwieldy tree output, **COBWEB** results cannot be directly compared with those of other clustering algorithms. Although it is possible to "flatten" the tree into a standard partitioning of clusters, the straightforward flattening algorithm (choosing each merge in such a way as to maximize CU) may lead to terrible results, such as clusters consisting of disjoint batches of points. The thousand-node trees produced by **COBWEB** give a visualization of the relationships between MD configurations; however, they may be difficult to see and understand.

Clustering Metrics. To avoid bias, assessment via quantitative measures is desirable. Unfortunately, there is no universally accepted metric of "clustering quality". Despite this, metrics do provide a general indication of whether one clustering method is generally better than another.[49] In the current work we explored various distinct metrics, including the Davies-Bouldin index (DBI) and the pseudo F-statistic (pSF). DBI effectively measures the average over all clusters of the maximal values of the ratio that divides the pairwise sum of within-cluster scatter (where the scatter is the sum of the average distance of each point in the cluster from its centroid) by the intercluster separation.[90−93] It aims to identify clusters that are compact and well-separated. Low values of DBI indicate a better clustering. As the value of DBI is affected by cluster count, it makes sense to only compare DBI values for different clustering algorithms when the number of clusters is similar. Also, as the number of clusters decreases, the DBI value automatically tends toward smaller values. The pseudo-F statistic (pSF) is based on a comparison of intracluster variance to the residual variance over all points[94] and is determined from the classical regression model coefficients of SSR (sum of squares regression, or explained variation) and SSE (sum of squares error, or residual variation) through the ratio (for all points n and g clusters):

$$\mathrm{pSF} = \frac{\mathrm{SSR}/g - 1}{\mathrm{SSE}/(n - g)}$$

High values of pSF indicate better clustering. Since pSF sometimes rises with cluster count, one generally looks for a peak in pSF where the number of clusters is still manageably small. These metrics are imperfect. For instance, low DBI values result when the cluster algorithms produce several (likely uninformative) singleton clusters. On the other hand, pSF tends to give its highest scores when all clusters have approximately the same size, even if the clusters are badly formed. In our experience, using both metrics in conjunction with examination of the tree of clusters (see below) appears to be a promising way to assess clustering quality. Moreover, to assess cluster count, we can also use the "elbow criterion". This is a common evaluation tool that chooses the appropriate number of clusters by noting where adding in additional clusters does not add sufficient new information.[95] This can be seen by plotting the percentage of variance explained versus cluster count where a kink or angle in the graph (the elbow) illustrates the optimal cluster count. The percentage of variance explained by the data is the SSR/SST ratio where SSR is the sum of squares regression (or explained variation) from each cluster (summed over all clusters) and SST is the total sum of squares. The SSR/SST ratio is equivalent to the coefficient of determination or the R-squared value in classical regression. When this value is low, little variance is accounted for by the regression, and the clustering is likely poor. Another metric that provides insight into the proper cluster count is the critical distance. This is defined as the distance between the clusters that were just split or merged. The distance is different for each algorithm, as discussed previously; for example, the distance between clusters for the **centroid-linkage** algorithm is the distance between centroids, whereas for the **edge** algorithm it is the shortest point-to-point distance between clusters. Abrupt changes in the critical distance, as a function of cluster count, highlight optimal cluster counts. For example, if splitting a cluster leads to a significantly smaller critical distance than was seen previously, this suggests that the two new clusters are much closer together than clusters were in earlier splits and suggest that the split may have been unnecessary. The critical distance metric is not defined for the refinement algorithms.

One feature that emerged from the clustering of the real MD data is that the algorithms tended to group frames from a contiguous block of time together, even when sampling at 10−50 ps intervals. This is expected since with frequent sampling each simulation frame is necessarily close to its neighbors. However, the fact that clusters generally consist of frames from a single block of time shows that our sampling of conformational space may not be complete. Given a sufficiently long simulation, we would expect to see the system to revisit old clusters repeatedly (with sampling according to the Boltzmann distribution). As a rough quantification of this behavior, we define the "progress" of a cluster as $1 - S/E$, where S is the actual number of "switches" (i.e., the number of time points such that frames n and $n+1$ are in a different cluster), and E is the expected number of switches (based on the cluster's size and assuming random membership, $E = (n-1) * \Sigma(n_g/n * (n-n_g)/n)$ over clusters g). This number goes to zero as the actual number of switches approaches the expected number for a random distribution. The progress of large clusters for most of clustering is above 0.8, which means the continuous frames tend to be in the same cluster even at 10 ps sampling of the MD trajectory data. This observation can be used to guide choices of optimal cluster counts since the progress will likely decrease as the cluster count increases. Observation of progress values in the range of ∼0.5 may suggest that the data are overpartitioned or poorly clustered. The subjective choice of a cutoff for progress values will depend on the sampling frequency and rate of conformational exchange and therefore cannot be considered in isolation; it is better to examine how the progress changes as a function of cluster count.

Molecular Dynamics Trajectories

A variety of different production-phase MD simulations were performed to provide the raw MD trajectory data used to test and validate the clustering algorithms. All of the MD simulations were performed with the Amber software suite.[86] For the simulations of nucleic acids in solution, a particle mesh Ewald treatment of the electrostatics (with less than 1 Å FFT grid spacing, cubic interpolation for the reciprocal space charge grid, a 9 Å direct space cutoff with the Ewald coefficient adjusted so that the direct space energy is less than 0.00001 kcal/mol at the cutoff, SHAKE[96] on all bonds with hydrogen, and constant temperature (300 K) and pressure (1 atm) with weak Berendsen scaling[97]) was applied. The all-atom Cornell et al. force field[98] for the DNA was applied with necessary supplemental parameters (for the bound drug) as outlined below and is available in the Supporting Information. Two distinct sets of MD data of nucleic acids in solution were investigated, specifically a rather dynamic trajectory of an "unfolded" polyA DNA strand (10-mer) sampled at broad (20 ps) intervals from more than 15 different trajectories (each starting from a different "unfolded" conformation) and also from an artificially created small trajectory that sampled around five different structures at 100 ps intervals (which should only produce "good" clustering with a cluster count of 5) and a dynamic trajectory of a DNA hairpin loop with the drug DB226 bound in the minor groove that shifts from one binding mode to another over the course of 36 ns of simulation. Additional biomolecular systems clustered include a ∼75 ns simulation of a solvated mammalian cytochrome P450 with PDB entry 1PO5.[76] The relevant data for these systems are provided in the Supporting Information.

polyA Single Strand. Simulations were performed on a 10-mer polyadenine single strand of DNA. The initial model was built into an idealized B-DNA helix (of polyA-polyT deleting the polyT strand) using the Amber **nucgen** utility. The DNA was solvated with 2402 TIP3P[99] waters in a rectangular box (∼53 Å × 42 Å × 35 Å with a 60 × 45 × 40 charge grid), and the charge was neutralized through the addition of nine Amber-adapted Aqvist sodium ions.[100] Simulations of the polyA single strand remain fully stacked and helical on a 5 ns time scale.[101] To investigate single strand structures more representative of the true ensemble

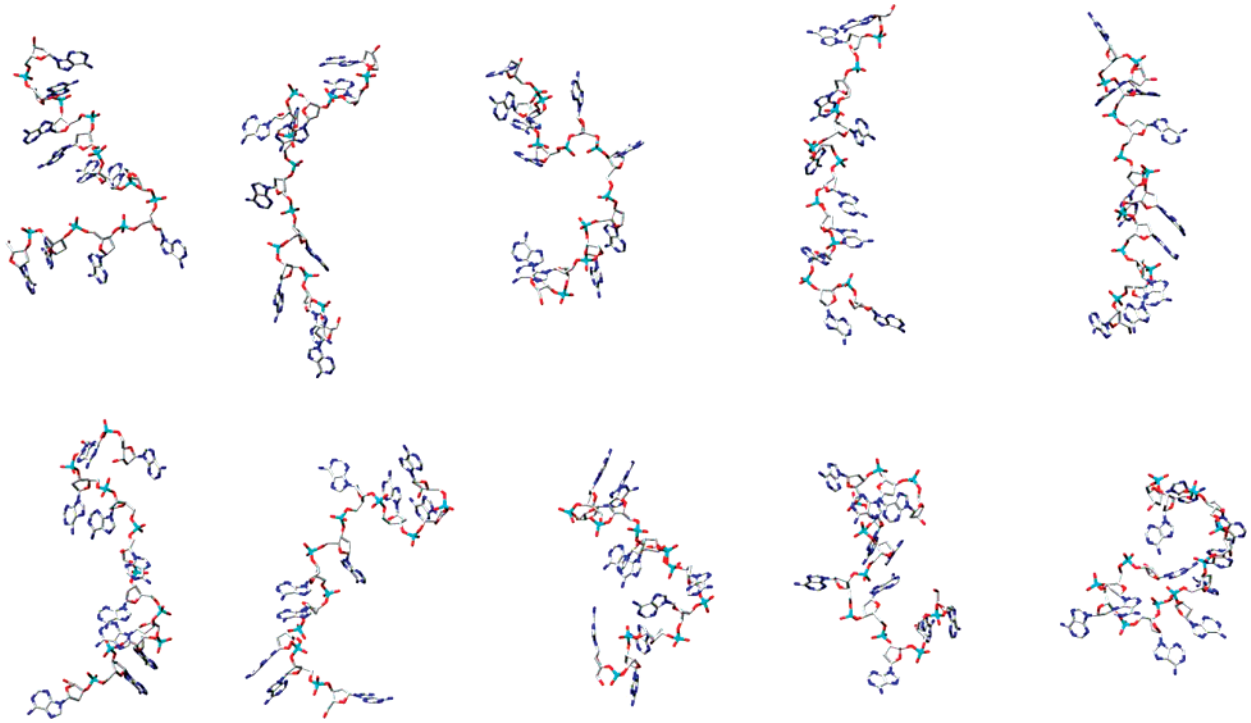

Figure 3. Snapshots from a 1050 ps self-guided MD simulation of a 10-mer polyA DNA single strand in explicit water (ions and water not shown) at 50 ps intervals (from left to right and top to bottom at 50 ps, 100 ps, 150 ps, 250 ps, 300 ps, 400 ps, 450 ps, 550 ps, 600 ps, and 650 ps) rendered with UCSF/Chimera.[106] Large guiding factors (0.5) and long (2 ps) local averaging times lead to considerable motion. When the SGMD is turned off, the single strands begin to "fold" into various stacked adenine structures.

and to generate a set of diverse conformations for clustering, the self-guided MD (SGMD) method was utilized with a guiding factor (0.5) and local averaging times (2 ps) significantly greater than are routinely applied (which are in the range of 0.1 and 0.2 ps, respectively).[102−105] When used in this manner, the SGMD rapidly moves the DNA and effectively samples a very wide range of "unfolded" conformations in short (1 ns) runs. Configurations from a 1050 ps SGMD simulation of this type, taken at 50 ps intervals (some of which are shown in Figure 3), were then run with standard MD protocols (Ewald and no SGMD) each on the 15−20 ns time scale. An aggregate trajectory for clustering was obtained by taking data from 15 of these trajectories at 20 ps intervals. As the starting configurations were widely different, this leads to a diverse set of single strand structures for clustering. In addition, an artificial trajectory of 500 frames was created from stable 100 ps regions of five of these independent trajectories. This creates a trajectory that should naturally split into five equally sized clusters. An additional 500 frame trajectory was created with unequally sized clusters of 2, 15, 50, 100, and 333 configurations, each sampling around distinct polyA geometries; this is a more difficult case to cluster as each resulting cluster has a different size. Moreover, the largest cluster samples multiple conformations and hence has relatively high variance compared to the smaller clusters. This is closer to what is expected for raw trajectory data; however, this trajectory is still easier to cluster than real MD trajectory data as there is no direct link or path between the clusters. During normal MD simulation and sampling on the picosecond time scale, the clusters are naturally linked due to the dynamics. As will be shown in the results, the contrived systems are easier to cluster. With real data, it is not obvious which algorithm is the best, and users likely have to explore multiple data clustering algorithms.

DNA Duplex-Drug Interactions. Simulations were performed on a model of the minor groove binding drug 2,5-bis[4-(N-alkylamidino)phenyl]furans (DB226)[107−109] bound to the ATTG region of a DNA hairpin loop. The DNA hairpin used has sequence 5′-CCA**ATTG**G-(TCTC)-C**CAAT**-TGG where the start binding site is indicated in bold and the loop is in parentheses. During the simulation the drug DB226 shifted back to the canonical **AATT** binding region. The hairpin DNA model was created by building an idealized B-DNA helix (for the full symmetric sequence d(CCAAT-TGGTC)$_2$) using the Amber **nucgen** utility followed by manually linking the two strands at one end. The model structure was relaxed with 1000 steps of steepest descent minimization (no cutoff) allowing only the six residues centered on the hairpin to move and 100 ps of dynamics with a generalized Born implicit solvent model (igb=1,[110,111] no cutoff, SHAKE on hydrogens,[96] 300 K with 1.0 ps coupling time with Berendsen temperature control[97]) allowing only the four loop residues to move. As this force field does not contain parameters for DB226, parameters (see the Supporting Information) were obtained using Antechamber[112] and the GAFF force field[113] using RESP charges[114] from a 6-31G* optimization with Gaussian 98.[115]

To build the initial model system, in analogy with the crystal structure of 2,5-bis(4-guanylphenyl)furan (furamidine) bound to the d(CGCGAATTCGCG)$_2$ dodecamer (PDB accession number 227D),[116] the 3-pentyl diamidine derivative

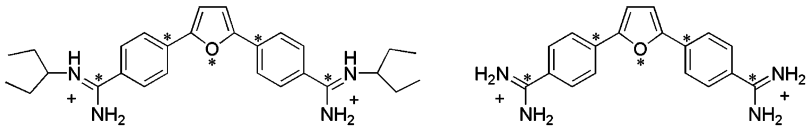

Figure 4. The molecular structure of DB226 (left) and furamidine (right). For full details on the parametrization, see the Supporting Information. The *'s denote atoms that were used for rms fits to the crystal structure during the initial docking, and the labels refer to atoms used for initial restraints to the DNA structure as discussed in the text.

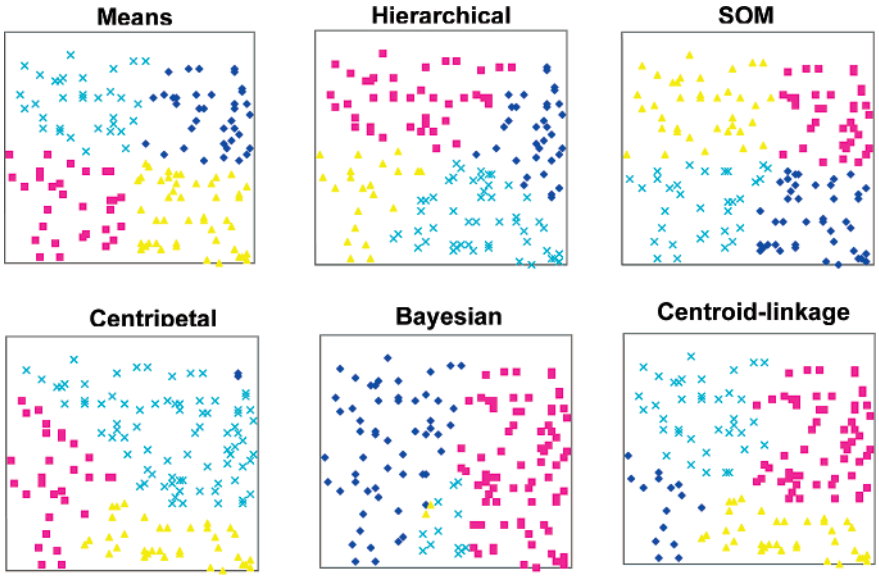

Figure 5. Clustering algorithms applied to the same set of random points in the 2D plane. The results highlight the features of six of the distinct clustering algorithms investigated, such as the uniform/linear cutting of the **hierarchical** clustering, the uniform sizes of the clusters created by the **means** clustering, and the ability of **centroid-linkage**, **centripetal**, and **Bayesian** algorithms to create clusters with distinct shapes and sizes.

of furamidine (DB226)[108,109] was hand-docked into the ATTG region. This was done by a rms fit of the Gaussian-optimized geometry of DB226 to the crystal structure of bound furamidine (using the five atoms denoted with an asterisk in Figure 4) and of four ATTG binding-site phosphates in the DNA hairpin to the crystal structure binding site. The system was then minimized for 500 steps using the steepest descent method in vacuo with no cutoff, followed by 100 ps of generalized Born implicit solvent simulations as above (except with a temperature coupling time of 10.0 ps). Distance restraints were applied (both to the heavy atoms and the hydrogens with a flat-well potential from 2.0 to 3.0 Å or 1.0 to 2.0 Å, respectively, with a 5.0 kcal/rad^2-mol lower bound force constant to 0.0 Å and a 15.0 kcal/rad^2-mol force constant beyond the upper bound) for DB226 atom N4 to O2 of base T17, N2 to N3 of base T7. Harmonic positional restraints with a force constant of 5.0 kcal/Å^2-mol were applied to the DNA duplex. Note that in early simulations of this system, where no restraints were applied during the in vacuo equilibration, the ligand either shifted to an alternate binding mode or escaped the groove entirely. This reinforces the need to be careful when initially setting such systems to avoid artifacts due to the equilibration and initial modeling procedure.

The system was then solvated with explicit TIP3P water[99] in a truncated octahedron periodic unit cell to a distance of 9 Å. Explicit net-neutralizing Na^+ and an additional 12 Na^+ and Cl^- ions were added to bring the system to a salt concentration of ~100−150 mM. The water and counterions were allowed to equilibrate via the same minimization and relaxation steps, with the DNA and ligand fixed. Finally, production MD was run for more than 36 ns.

Results

Clustering Points in the 2D Plane. To illustrate differences in the clustering algorithms, Figure 5 shows the performance of various clustering algorithms when applied to the same randomly selected collection of points in the 2D plane. Visualization of the data for each algorithm (run with a cluster count of four where each cluster is denoted by a different symbol and color) shows the significant variation in the cluster sizes and shapes. Each algorithm clusters the same data in very different ways. The properties of the various algorithms, such as the ability to handle cluster convexity and the preferences toward producing clusters of similar sizes, tend to carry over to other problem domains.

Unlike the data shown in Figure 1, the random distribution of points shown in Figure 5 does not have an obvious partitioning. How these data are clustered will depend on the details of the algorithm and how intra- and intercluster separation and variance are determined. As each algorithm is different, it is not surprising that rather different sets of clusters emerge with the different algorithms. The **means**, **hierarchical**, and **SOM** clustering algorithms tend to produce clusters of similar size with a linear partitioning of the data. The **centripetal** and **Bayesian** clustering algorithms are able

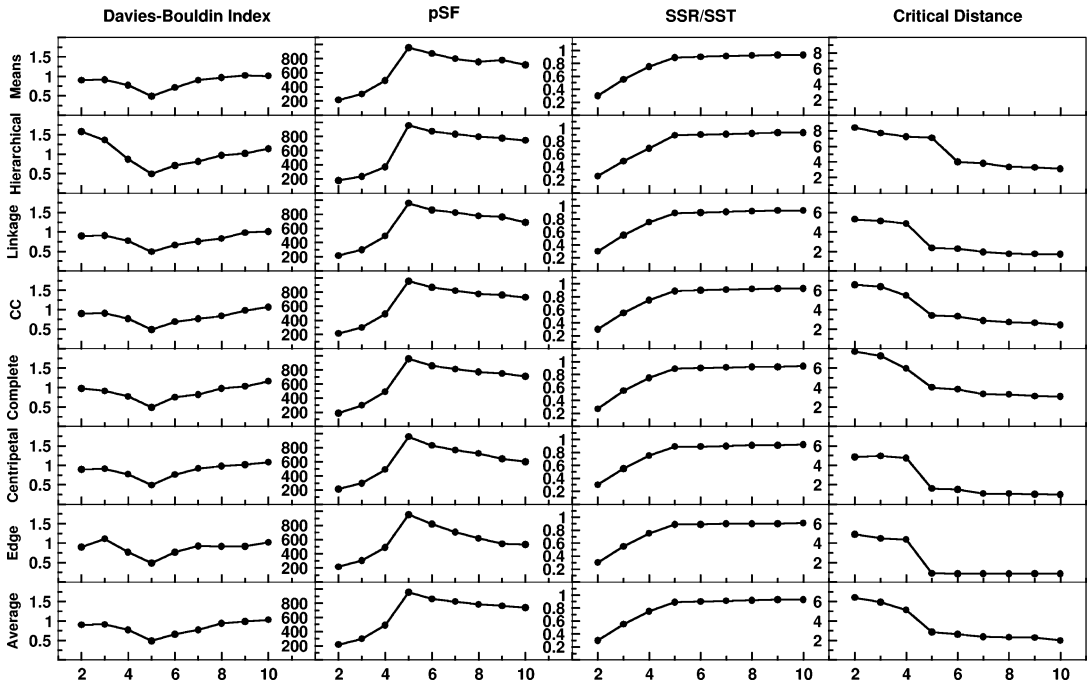

Figure 6. Cluster metrics for a subset of the algorithms investigated for the constructed polyA trajectory of five distinct equally sized clusters as a function of cluster count (*x*-axis). At the optimal cluster count of 5, DBI is at a minimum, pSF is at a maximum, the SSR/SST values plateau, and a transition occurs in the critical distance. "CC" is the centripetal complete clustering algorithm, and a critical distance is not shown for the **means** refinement clustering algorithm since it is ill-defined. Note that for the **means** refinement, five independent clustering runs (with random choices of configurations for the refinement steps) were performed, and the data shown are for the run with the highest pSF value.

to produce both small and large clusters. The **centroid-linkage** is able to produce clusters of very different shapes. The **centripetal** algorithm is able to associate distant points into a cluster despite having very few points near the "centroid" due to the use of representative points distant from the centroid. With the exception of the **Bayesian** algorithm, all shown tend to naturally partition the data.

Clustering Artificial MD Data: Five Equally Sized and Distinct Clusters. After development and testing of the algorithms on the points in the plane examples, clustering was performed on a series of MD trajectories using both the RMSd and the DME as a metric and a series of independent runs varying the cluster count and other variables. To demonstrate the results, two trajectories of 500 configurations from the polyA single strand MD simulations were created and then clustered. In each case, these trajectories were created from independent runs and sampling around five distinct conformations. The first test set has 100 configurations for each distinct conformation leading naturally to a partitioning into five equally sized clusters. Clustering metrics as a function of the cluster count are shown in Figure 6. The metrics show the expected (idealized) behavior including a minima in the DBI, maxima for pSF, a plateau in SSR/SST, and a sudden drop in the critical distance when a cluster count of 5 is reached. Beyond five clusters, little new information is gained from further partitioning of the data. The behavior of the critical distance at the transition point around the optimal cluster count is effectively opposite for the top-down compared to the bottom-up algorithms. In the case of the bottom-up algorithms and cluster merging, there is a sudden jump as the cluster count goes from 5 to 4; this indicates that the distance between the newly formed clusters is much larger than the distance (variance) between previous clusters. With the top-down or hierarchical algorithm, the change in the critical distance occurs as we split clusters from the optimal count of 5 to 6; this leads to a drop in the critical distance suggesting that the split leads to clusters that are significantly closer together than the previous clusters were. As an indicator of the proper cluster count, the drop in the critical distance occurs at the proper cluster count for the bottom-up algorithms and just after the proper cluster count for the top-down algorithms.

In general for this artificial data set most of the algorithms perform equally well. The exceptions are the **Bayesian** and **COBWEB** clustering algorithms which yield some of the expected 100-member clusters in some cases but incorrectly split other clusters; this contrasts with the **SOM** algorithm which correctly generates the five expected 100-configuration clusters. An additional limitation of the **SOM** and **Bayesian** algorithms that was uncovered is that both algorithms may fail to generate the expected cluster count (i.e., the algorithms can form clusters that contain no points). In some cases, when more than 5 clusters are requested, the **SOM** algorithm will yield only the 5 expected clusters. This property may be exploited to determine optimal cluster count. For more data on the **Bayesian**, **COBWEB**, and **SOM** clustering results refer to Table ST2 in the Supporting Information.

Clustering Artificial MD Data: Five Differentially Sized Clusters. The second set of artificial MD trajectory data was also constructed from sampling around five distinct conformations, but each cluster was constructed to be a different size, specifically with 2, 15, 50, 100, or 333

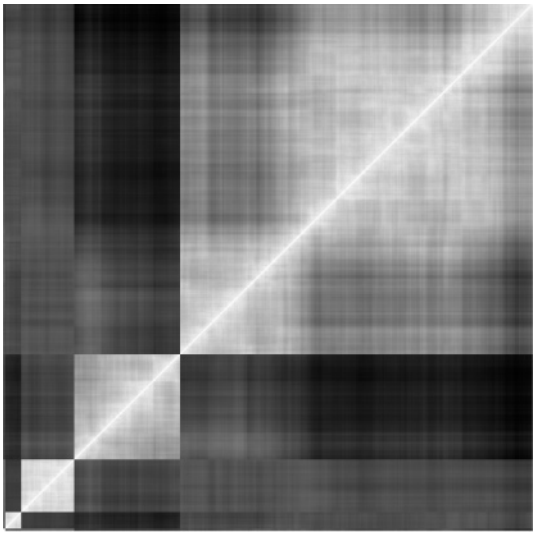

Figure 7. 2D RMSd plot (mass weighted) for all frames from the polyA single strand simulation with five differentially sized clusters. Note that only four clusters are readily visible since the first cluster is very small.

configurations in five separate clusters. This set is more difficult to cluster as it has both very small clusters with small variance and relatively large clusters with larger variance. The average distance of each conformation in the cluster to its centroid spans a large range. These average values for each cluster size : distance pair are as follows: 2 : 0.27 Å; 15 : 0.72 Å; 50 : 0.84 Å; 100 : 1.62 Å; and 333 : 2.34 Å. The maximal pairwise best fit RMSd between any two conformations is 9.8 Å. Although the intent was to create five clusters, the natural partitioning may be closer to six. This can be seen clearly from visualization of the 2D RMSd plot or effectively the visualization of the matrix of pairwise best fit RMSd values of every structure to every other structure for all the conformations (see Figure 7). The plot shows that each cluster is dissimilar from its neighbors and also that the largest cluster may best be represented by two similar clusters. The diagonal elements (white) represent self-comparison or zero RMSd and the black shows the largest pairwise RMSd of 9.8 Å. As the members of the smallest cluster are distinct from the others, partitioning ideally should create the small clusters before breaking the largest cluster. With a cluster count of 5, this does not happen with the **centripetal complete**, **COBWEB**, **complete**, **hierarchical**, **means**, and **SOM** algorithms. The cluster sizes for each of these algorithms, italicizing the incorrect cluster sizes, are **centripetal complete**: 15, *52*, 100, *106*, *227*, **COBWEB**: 50, *63*, *115*, *117*, *155*, **complete**: 15, *52*, 100, *137*, *196*, **hierarchical**: 15, *52*, 100, *112*, *221*, **means**: 15, *52*, 100, *113*, *220*, and **SOM**: *67*, *95*, 100, *114*, *124* algorithms. In most of these cases, the largest and smallest clusters are not found. **Edge**, **centripetal**, **average-linkage**, and **linkage** each partition the data as expected into five distinct clusters; when a cluster count of 6 is specified, the largest cluster is broken in two with each of these algorithms, although the sizes are different (**centripetal** 114, 219; **edge** 70, 263; **average-linkage** 102, 231; **linkage** 106, 227). The **centripetal complete**, **complete**, **means**, and **hierarchical** recover the natural partitioning when a cluster count of six is specified. This is not true of the **Bayesian**, **SOM**, or **COBWEB** algorithms, and the "small" cluster of 2 conformations is not found until a cluster count of 10 is reached with the **Bayesian** algorithm. The **SOM** and **COBWEB** algorithms appear to be unable to produce the smallest cluster. In the Supporting Information, trees outlining the partitioning of conformations by each of the algorithms are displayed. The **centripetal**, **edge**, **average-linkage**, and **linkage** algorithms show the best partitioning.

Shown in Figure 8 are the clustering metrics for some of the algorithms, omitting the poorly performing **Bayesian**, **SOM**, and **COBWEB** algorithms, on this artificial polyA MD trajectory with varying cluster sizes. It is clear from these data that when the configurations are not uniformly separated into similarly sized clusters, the metrics are less consistent across algorithms and also less informative. For many of the algorithms, a clear minimum in DBI or maximum in pSF is not readily evident. Similarly, rather than showing a clear elbow in the SSR/SST or critical distance plots as a function of cluster count, a smoother linear plot is often evident. These data can also be misleading. For example, the **hierarchical** clustering shows a clear minimum in DBI at a cluster count of 5 or 6, a maxima in pSF at a cluster count of 5, and a clear kink in the SSR/SST plot at a cluster count between 4 and 5, yet this algorithm does a poor job of clustering into five clusters. At a cluster count of 5, the **hierarchical** algorithm has already split the largest cluster but not split the 50 + 2 cluster into two separate clusters. Moreover, although the **centripetal** algorithm shows excellent clustering, the performance is not readily evident from the DBI, pSF, and SSR/SST metrics. The most definitive demonstration of metric success comes with the **edge** algorithm; this algorithm very naturally partitions the data.

Shown in the Supporting Information are schematics of the cluster trees or partitioning by various algorithms (Figures S2−S11) and the distinct performance metrics (Table ST3) for the **Bayesian**, **COBWEB**, and **SOM** algorithms.

As MD ideally samples according to a Boltzmann distribution, it is expected that the data will look more like the artificial trajectory with differentially sized clusters than data that partition into equally sized clusters. This is because the population of a given conformer is related to its free energy, with lower populations as the energy increases. This suggests that finding the ideal partitioning and clustering of the data will be messier with real data and that the various algorithms will each lead to distinct partitioning of the data.

Clustering Real MD Data: Simulation of Drug−DNA Interaction. A series of MD simulations were performed on a series of minor-groove binding agents binding into the minor grooves of various DNA hairpin sequences. The specific MD trajectory of DB226 binding to the ATTG sequence of a hairpin DNA was chosen for clustering and further analysis. This is an interesting case as the simulation revealed a major shift, by one base pair step, in the binding of the minor groove binding drug DB226 to the DNA hairpin. As this change in binding is easy to visualize, this is a good test of the clustering algorithms ability to discern and to naturally partition the data. To cluster the MD trajectory data,

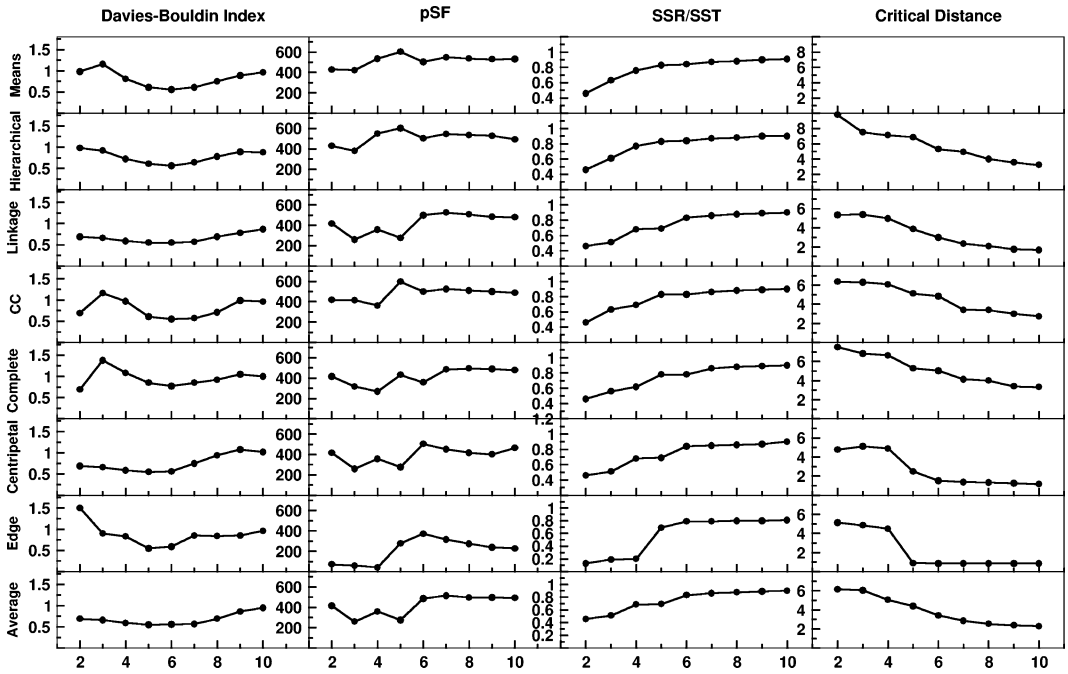

Figure 8. Cluster metrics for a subset of the algorithms investigated for the artificially constructed polyA trajectory representing five clusters of distinctly different sizes as a function of cluster count (*x*-axis). Note that for the **means** refinement, five independent clustering runs (with random choices of configurations for the refinement steps) were performed, and the data shown are for the run with the highest pSF value.

each algorithm was applied to the drug−DNA hairpin MD trajectory over 36 ns with configurations taken at 10 ps intervals, for a total of 3644 frames. For each of the algorithms investigated, a range of cluster counts from 2 to 20 was evaluated. To limit the structure comparisons to the binding region of the drug−DNA complex, the clustering metric used was the best-fit RMSd of the residues defining the drug and the binding region. Specifically, this included all the carbon, nitrogen, and oxygen atoms of the drug and from residues 3−8 and 14−19 of the DNA, i.e., the AATTGG binding region noting that the drug initially binds at the ATTG site. Figure 9 shows the shifting of DB226 to the AATT site that occurs during the MD simulation between 15 and 16 ns. The MD results suggest that multiple modes for DB226 binding to the DNA hairpin are thermally accessible. From the plot of the distances versus time shown, it appears that the drug attempts to shift down a base pair step at ∼6 ns, but moves back to the ATTG site, and then eventually successfully fully shifts by one base pair step by ∼16 ns. Additional data and discussion, including plots of the overall RMSd versus time (Figure S12) and molecular graphics of average structures before and after the change in drug binding (Figure S13) and the clustering data across the different cluster counts (Tables ST4−ST6), are shown in the Supporting Information.

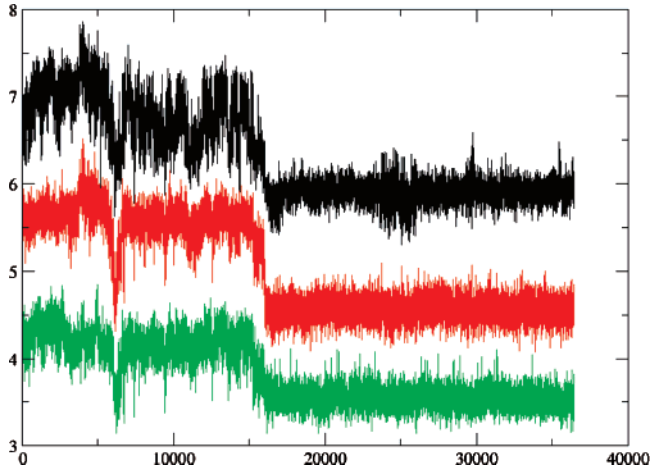

Figure 9. Graph of selected atom positions of the minor groove binding drug DB226 relative to the base pair step (*y*-axis, base pair step number) as a function of time (*x*-axis, in ps). In each frame, we calculated the least-squares fit plane for each base pair and averaged the normal vectors perpendicular to those planes. From this, we can interpolate the position of an arbitrary atom based on the midpoints of those least-square fit planes. The red (middle) represents the furan oxygen atom in the middle of DB226. The black and green represent the two guanyl nitrogen atoms at each side of the drug. From the plot, the shifting of DB226 from the ATTG region (base pair steps 4−7) to the AATT region (base pair steps 3−6) is evident.

Relative Performance of the Clustering Algorithms. The data in Table 1 provide a summary of the relative performance and properties of the various clustering runs as a function of cluster counts. Included are the runtimes, DBI, and pSF metrics, the SSR/SST ratio or R-squared value, the "progress" of the simulation, and the sizes of the resulting clusters. The full data, including cluster counts of 10 and 20, are in the Supporting Information (Table ST4−ST6). The runtime in the table is the time for running each clustering algorithm on an equivalent machine. The actual runtime will be increased by the time needed to precalculate the full pair wise distance matrix which amounted to 367 s for the

Table 1. Comparison of the Various Clustering Algorithms Applied to a 36 ns MD Trajectory of DB226 Bound to Hairpin DNA[a]

algorithm	cluster	runtime	DBI	pSF	SSR/SST	progress	cluster sizes
average	3	6197	1.10	1545.23	0.459	1.00	2061, 1581, 2
average	4	6279	1.59	1119.57	0.480	1.00	2061, 1454, 127, 2
average	5	6387	1.56	925.33	0.504	0.99	2061, 1340, 127, 114, 2
Bayesian	3	59	1.91	1820.10	0.500	0.89	2034, 929, 681
Bayesian	4	94	2.37	1395.03	0.535	0.73	1054, 996, 952, 642
Bayesian	5	120	2.17	1187.41	0.566	0.76	1366, 786, 689, 476, 327
centripetal	3	3272	1.14	108.29	0.056	0.98	3516, 127, 1
centripetal	4	3351	1.02	72.65	0.056	0.97	3516, 126, 1, 1
centripetal	5	3103	0.98	54.83	0.057	0.96	3515, 126, 1, 1, 1
CC	3	1516	1.54	1470.70	0.447	0.98	2148, 1492, 4
CC	4	1477	1.68	1066.73	0.468	0.98	2148, 1395, 97, 4
CC	5	1572	1.30	801.19	0.468	0.98	2148, 1395, 97, 3, 1
COBWEB	3	1854	1.87	1757.04	0.491	0.77	1594, 1109, 941
COBWEB	4	1236	2.12	1378.27	0.532	0.86	1804, 780, 764, 296
COBWEB	5	1221	2.88	1071.62	0.541	0.63	1025, 780, 764, 568, 507
complete	3	922	1.86	1585.13	0.465	0.85	1703, 1407, 534
complete	4	960	2.16	1163.30	0.490	0.83	1703, 1216, 534, 191
complete	5	1398	2.35	931.67	0.506	0.83	1703, 1060, 534, 191, 156
edge	3	923	0.54	3.43	0.0019	0.25	3642, 1, 1
edge	4	931	0.77	3.43	0.0028	0.37	3640, 2, 1, 1
edge	5	930	0.78	2.92	0.0032	0.30	3639, 2, 1, 1, 1
hierarchical	3	6	1.80	1898.54	0.510	0.91	2088, 960, 596
hierarchical	4	7	1.86	1362.83	0.529	0.90	2088, 960, 304, 292
hierarchical	5	9	2.13	1199.49	0.568	0.77	1350, 960, 738, 304, 292
linkage	3	2349	0.93	1544.21	0.459	0.99	2077, 1566, 1
linkage	4	2158	1.06	1035.76	0.460	0.99	2077, 1562, 4, 1
linkage	5	1782	1.07	805.17	0.469	0.99	2053, 1562, 24, 4, 1
means	3	1105	1.80	1899.67	0.511	0.91	2088, 965, 591
means	4	953	2.11	1490.24	0.551	0.79	1402, 967, 702, 573
means	5	909	2.02	1222.68	0.573	0.78	1322, 891, 756, 454, 221
SOM	3	663	1.70	1597.10	0.467	0.98	2066, 1546, 32
SOM	4	1391	1.96	1149.12	0.486	0.93	2035, 1396, 134, 79
SOM	5	1558	2.13	1059.96	0.538	0.74	1238, 1100, 956, 212, 138

[a] The RMSd of carbon, nitrogen, and oxygen atoms of the drug and residues 3−8 and 14−19 of the DNA hairpin were used as the pairwise distance between all configurations from the MD simulation at 10 ps intervals. The cluster count represents how many clusters were chosen. The DBI and psF values are metrics of clustering quality; low values of DBI and high values of pSF indicate better results. The R-squared (SSR/SST) value represents the percentage of variance explained by the data; plots of this as a function of cluster count can show where adding more clusters fails to add new information shown by the elbow criteria or a kink in the plot. The "progress" as discussed in the Methods section describes how often switching between the clusters is occurring. Larger values imply that most of the cluster members are sequential in time with values of 1.0 meaning all frames in a cluster are contiguous. **Edge** and **centripetal** clustering produced optimum values of DBI but generated pathological singleton clusters. **Centroid-linkage** clustering performs well under both metrics. **Hierarchical** clustering is the fastest of the algorithms applied. **CC** refers to **centripetal-complete** clustering. The **SOM**, **means**, **Bayesian**, and **COBWEB** algorithms were each run five times, and the results with the highest pSF are shown.

clustering data in Table 1. The time for calculating a DME pair wise distance matrix increases significantly as the number of atoms in the comparison increases. The DME matrix preparation time for the same atoms is 24 519 s. This also significantly increases the runtime for the algorithms which need to recalculate at each iteration the DME distance between the cluster centroid and every other clusters centroid, such as with the **centripetal** and **linkage** algorithms.

Clearly the fastest algorithm at low cluster counts is the **hierarchical** clustering and the most computationally demanding algorithms are **average linkage**, **centripetal**, **linkage**, the neural net refinement (**SOM**), and **COBWEB** algorithms. With the exception of the **Bayesian** and **SOM** refinement algorithms, the relative cost does not tend to increase dramatically as the number of clusters goes up.

In terms of the clustering quality metrics, algorithms that have high pSF and low DBI values at a given cluster count suggest better clustering. For all the algorithms applied to the MD trajectory of the drug bound to the DNA hairpin, where the similarity was ascertained by fitting to atoms in the binding region, this mix occurs at a cluster count of between 2 and 4. Based solely on pSF and DBI, a cluster count of 2 is suggested by the data; however, the SSR/SST ratio and critical distance plots suggests a count closer to 5

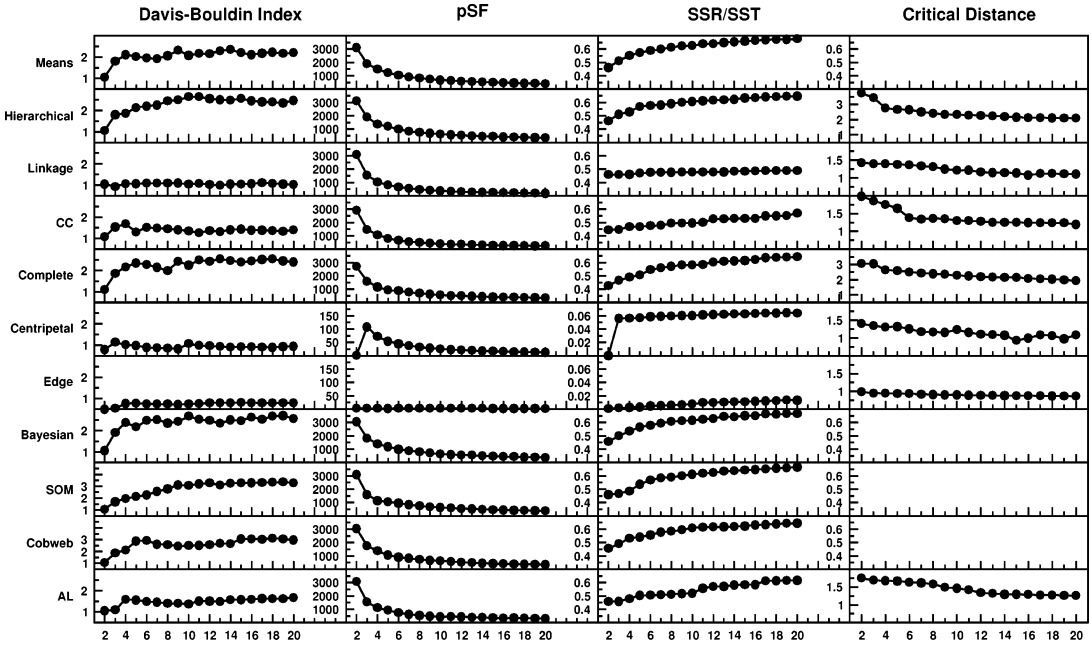

Figure 10. Cluster metrics for the MD trajectory of drug−DNA interaction as a function of cluster count (*x*-axis). Note that the scales of SSR/SST are different for the **centripetal** and **edge** clustering algorithms and that the scales for the critical distance are different for the **hierarchical** and **complete** clustering algorithms. The critical distance is not defined for the refinement algorithms and hence is not shown in these cases.

or 6. As the cluster count goes up, DBI is relatively constant, pSF gets smaller, and more information is added according to the SSR/SST. This is likely characteristic of MD simulation data as finer grained partitioning among the dynamic continuum of states is possible until all of the substates have been defined; as the potential energy surface is rough and there are many degrees of freedom, there are likely many different substates defining the path of the molecule in time. Although each algorithm suggests an optimal cluster count somewhere in the range of 2−6, the resulting cluster sizes vary considerably, and this impacts the relative performance of each. Most *inconsistent* with the natural partitioning of the data are the results from the **centripetal** and **edge** algorithms. These display a single large cluster with either small or singleton clusters outliers. In these cases, high pSF values are not obtainable, and the "progress" of 1.0 implies that each cluster has contiguous frames in time. Considering the data shown in Figure 9 and in the Supporting Information, and given our knowledge that two distinct binding modes for the drug were explored in the MD, we would expect the natural partitioning of this data to include the two distinct binding modes with drug binding to the ATTG and AATT binding sites. Shown in Figure 11 is a plot of the 2D RMSd values over the binding region during the ∼36 ns of simulation at 100 ps intervals. Two clear clusters are evident. The partial transition to the alternate binding mode at ∼6 ns is evident as the horizontal and vertical light lines which show agreement of frames from early in the trajectory (∼6 ns) with those from later in the trajectory. From the 2D rms plot, the next partitioning could split up either the smaller or larger cluster. Starting from the lower left (or the early part of the trajectory), the first cluster should have ∼1600 frames and the second ∼2040. This partitioning with a cluster count of three is seen with the **hierarchical** algorithm and

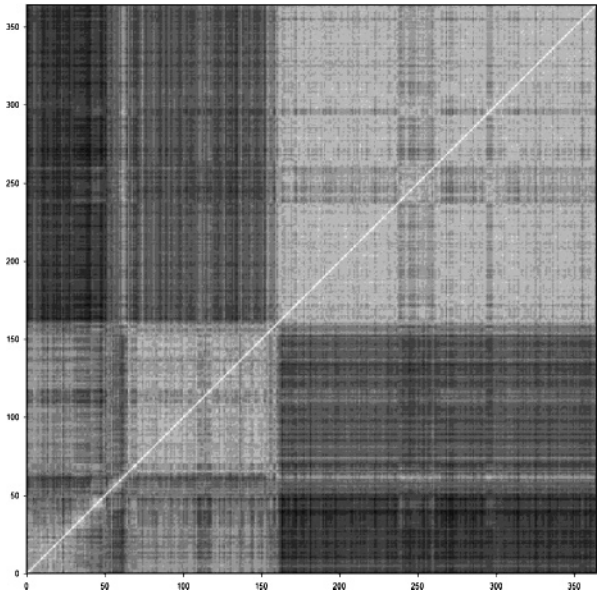

Figure 11. 2D RMSd plot (mass weighted) for frames at 100 ps intervals from the 36 ns simulation of DB226 binding to the DNA hairpin.

cluster counts of 596, 960, and 2088. The highest pSF values are observed with **hierarchical** and **means** clustering algorithms. **Average-linkage** and **linkage** both obtain a low DBI and high pSF; however, the SSR/SST plot is essentially flat. The reason the plot is flat beyond a cluster count of 2 is that only clusters with very few configurations are newly formed.

It is important to note that the performance of a given clustering algorithm is affected by the character of the data under consideration. In the case of the molecular dynamics trajectories of DNA interacting with DB226, the data did

not include extreme outlier points. As a quick screen for outliers, we examined the rmsd from each simulation frame to its nearest neighbor. For the major heavy atoms in the binding region in the trajectory, these nearest-neighbor distances were distributed between 0.4 and 1.2 Å, mostly around 0.8 Å. If we call the rms deviation from one frame to the next the "velocity" of a simulation, it is reasonable to suspect that early equilibration stages will have a relatively high velocity and therefore account for the bulk of the variation between clusters. This is one reason why it is best, when clustering MD trajectories, to exclude the initial equilibration portion of the simulation. For our DB226 trajectory, the equilibration protocol was successful in starting the system in a reasonable state—the velocity is consistent over time.

Distances between the Various Clustering Algorithms. In addition to evaluating the performance of a single clustering algorithm, we can measure the distance between two sets of different clusters of the same data produced by different clustering algorithms. This provides a measure of the disagreement between different clustering algorithms. One reasonable approach to measure the distances between clusters is to compute the rms distance between cluster representatives from the different sets of different clusters. However, in practice this is tricky as it requires guesswork to set up the correspondence between the clusters in each distinct set. To avoid this problem, we devised the following metric, $\partial(A,B)$. To measure the distance between clustering A and B, we consider the set of all pairs of points being clustered. We say that A and B agree on a pair of points if both A and B assign the points to the same cluster. If one of A or B assigns the pair of points to the same cluster but the other does not, this is counted as a disagreement. We compare the actual number of disagreements to the number of disagreements we would expect to see if A and B were unrelated. To do this, we first note the odds that two randomly chosen points will fall in the same cluster in A:

$$P_A = \sum_k \frac{S_k(S_k - 1)}{n(n-1)}$$

Here n is the number of points, and S_k is the size of cluster k. Now the expected number of disagreements (ED) can be computed:

$$\mathrm{ED}(A,B) = C^*(P_A(1 - P_B) + P_B(1 - P_A))$$

Here C is the number of pairs of points, and P_A and P_B are the probability that a pair of points falls in the same cluster in A and B, respectively.

Similarly, we can compute the expected number of the true agreements (EA), where a pair of points is both in the same cluster in clustering A and clustering B:

$$\mathrm{EA}(A,B) = C^*P_A^*P_B$$

As a finer-grained metric, we define a function ∂ measuring the ratio of the actual and expected number of disagreements over the ratio of the actual and expected number of true agreements:

$$\partial(A,B) = \frac{\dfrac{\mathrm{AD}(A,B)}{\mathrm{ED}(A,B)}}{\dfrac{\mathrm{AA}(A,B)}{\mathrm{EA}(A,B)}}$$

This distance function has several intuitive properties: $\partial(A, B) = \partial(B, A)$, and $\partial(A, A) = 0$, as we would expect. For random clustering C and D, $\partial(C, D)$ is very near 1. In general, a small distance reading ($<\sim 0.2$) indicates very similar clustering, such as is obtained when the same algorithm is used to generate a similar number of clusters. Readings less than ~ 0.6 indicate some similarity, and distances greater than ~ 0.6 indicate low levels of agreement.

The distance metrics, $\partial(A,B)$, for the various algorithms applied to clustering the DNA hairpin-DB226 trajectory are displayed in Table 2 and can be used to compare the output of the different algorithms. The data suggest that the **average-linkage**, **linkage**, and **centripetal-complete** algorithms generate very similar clusters. Similarly, the **Bayesian**, **SOM**, **hierarchical**, and **means** clustering all give related results, whereas the **complete** and **COBWEB** algorithms only generate somewhat similar sets of clusters when compared to the other algorithms. On the contrary, due to the production of singleton clusters, the resulting sets of clusters from the **edge** and **centripetal** algorithms are very dissimilar to the sets of clusters that result from the other algorithms under this distance metric. Direct comparisons of a given algorithm's clustering of the data using the two distance metrics, RMSd and DME, indicate that the resulting difference is fairly small—usually less than 0.2. However, again, for **edge** and **centripetal** algorithms, the differences go beyond 0.5. This indicates that both metrics (RMSd and DME) are capturing the conformational changes of the molecular system, though not in precisely the same way.

A useful application of this metric is to quantify the consistency between different runs of the stochastic clustering algorithms. The **SOM** clustering algorithm was run five times on the same system, and the distance between each resulting set of clusters was compared. The average distance was 0.03, indicating that **SOM** is very consistent. Similar runs for **means**, **Bayesian**, and **COBWEB** clustering yielded an average distance of 0.07, 0.15, and 0.25, respectively. Thus, it appears that the **SOM** and **means** algorithm provides more consistent (if not always better) results than those produced by the **Bayesian, COBWEB** clustering algorithm.

The Choice of Atoms for the Pairwise Comparisons. As might be expected, the clustering outcome is strongly influenced by which atoms are used to determine the similarity of the different molecular configurations. If too small a region is chosen, the fine partitioning that results may not be meaningful, similarly, choosing too many atoms may hide conformational substates visited by substructures during the MD. With the DB226-DNA hairpin trajectory, the distances between clusters are strongly influenced by the choice of atoms. For example, distances between clustering only the DB226 atoms compared to all solute atoms are generally above 0.9 for the bottom-up algorithms (i.e., **linkage**, **edge**, **complete**, **centripetal**) and $\sim 0.3-0.4$ for **SOM**, **hierarchical**, **means**, and **Bayesian** algorithms. If

Table 2. Distances between the Various Clustering Algorithms, in the Dimensionless Metric Defined in the Text That Compares the Ratio of the Actual to Expected Agreements and Disagreements[a]

RMS	average	linkage	CC	SOM	hierarchical	means	Bayesian	COBWEB	complete	centripetal	edge
average	**0.000**	**0.039**	**0.071**	0.169	0.182	0.205	0.240	0.245	0.209	0.887	0.994
linkage		**0.000**	**0.085**	0.169	0.196	0.224	0.255	0.259	0.234	0.951	0.996
CC			**0.000**	0.203	0.212	0.243	0.278	0.283	0.255	0.872	0.996
SOM				**0.000**	**0.137**	**0.154**	**0.158**	0.163	0.227	0.968	0.999
hierarchical					**0.000**	**0.064**	**0.130**	0.202	0.211	0.892	0.997
means						**0.000**	**0.100**	0.197	0.200	0.909	0.997
Bayesian							**0.000**	0.204	0.239	0.928	0.998
COBWEB								0.000	0.242	0.966	0.998
complete									0.000	0.914	0.997
centripetal										0.000	0.978
edge											0.000

[a] The trajectory being clustered was DB226 bound to hairpin DNA. Distances for cluster counts of 3, 4, 5, and 10 were computed and averaged. Clustering was performed using RMSd distances as the pairwise metric, the refinement algorithms were each run five times, and the data with the highest pSF values were used.

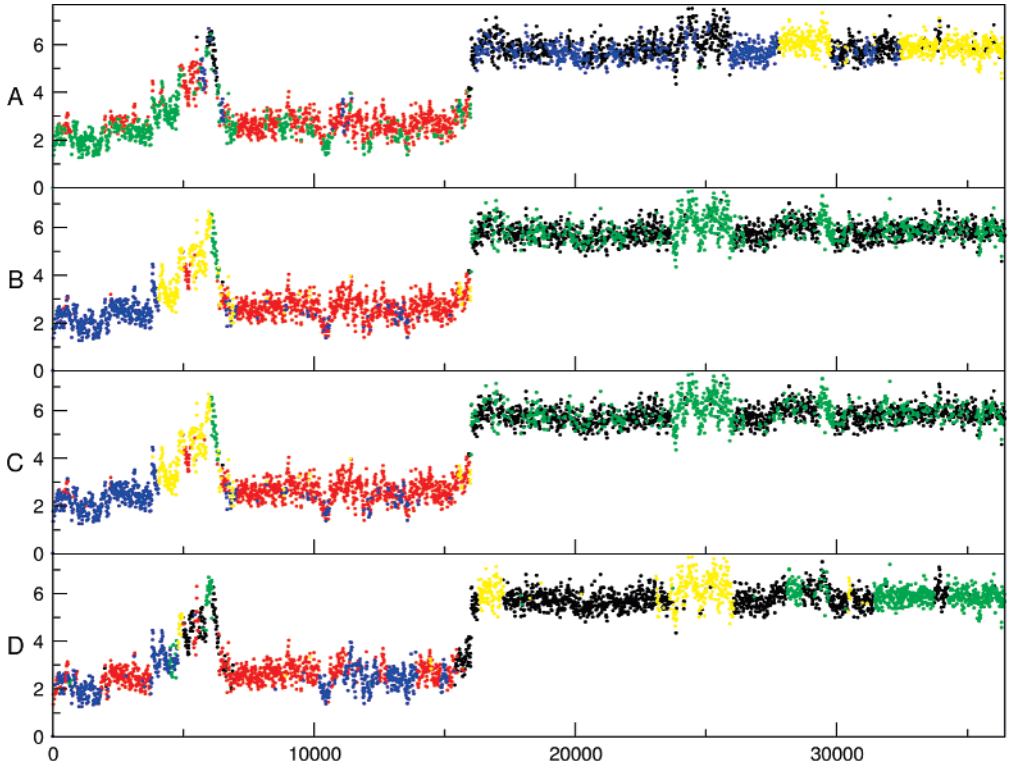

Figure 12. The effect of choosing different atoms for the pairwise comparisons when clustering. Based on the DNA hairpin-DB226 MD trajectory, shown is the RMSd (Å) as a function of time over the trajectory where individual points are colored based on their cluster identity. Four different sets of atoms for the pairwise comparison were chosen. (A) :1−21 represents all solute atoms, (B) :3−8:14−19:21 represents the atoms in binding region, (C) :3−8,14−19,21@C*,O*,N* represents the carbon, oxygen, and nitrogen atoms in binding region, and (D) :21 represents the atoms in the drug DB226 only. The trajectory data were taken every 10 ps and clustered using the **means** algorithm with a cluster count of 5.

similar atoms are chosen, the distances between the resulting sets of clusters are quite small, usually less than 0.01 for **linkage**, **hierarchical**, **means**, and **SOM** (i.e., comparing the major heavy atoms in the binding region to all atoms in the binding region). This suggests that it is best to narrow one's focus to the residues of interest before clustering a trajectory. Figure 12 shows the effects of clustering based on different choices of atoms for the pairwise comparison. Shown are the RMSd as a function of time for the atoms in the binding region with colors representing the distinct clusters that result based on different choices of the atoms for the pairwise comparison. The resulting sets of clusters based on pairwise comparisons of two sets of atoms describing the binding region, shown as the middle two plots in the figure, are almost identical as expected. The clustering based on the drug DB226 alone (residue 21, bottom), in comparison to the partitioning based on including all the DNA and drug atoms in the binding region, shows that although the drug conformation largely determines what cluster the configuration will adopt, clearly conformational substates of the DNA are also relevant. For example, compare the "green" cluster with and without the DNA binding region included. When the clustering is done on all of the DNA and drug atoms from the MD trajectory, some

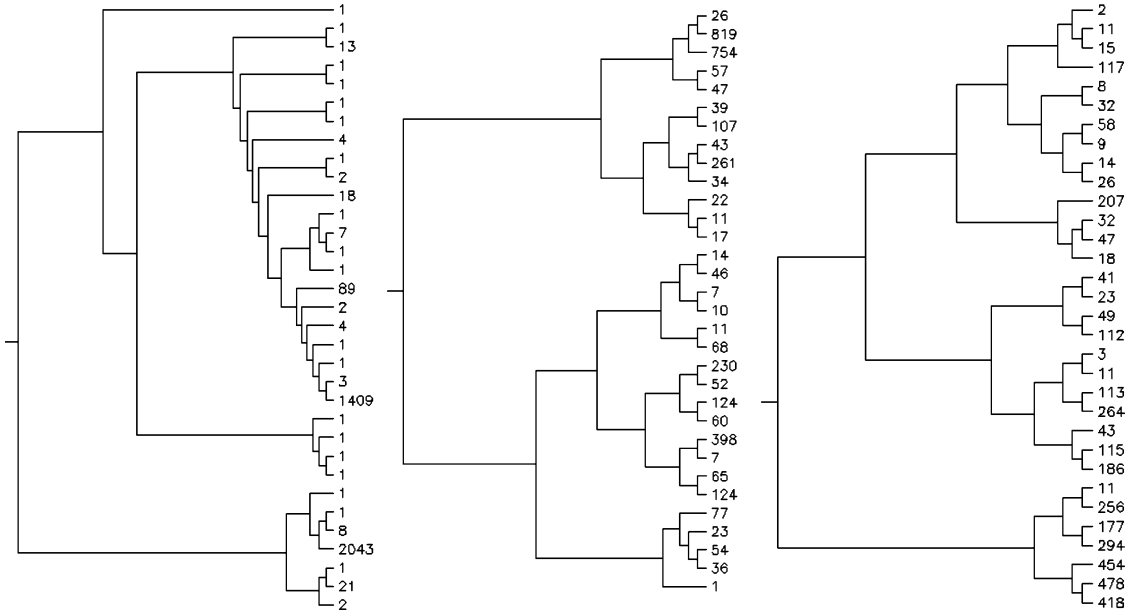

Figure 13. Cluster trees for **linkage** (left), **complete** (middle), and **hierarchical** (right) clustering algorithms applied to the DNA hairpin-DB226 trajectory. The last 30 steps of the algorithms are shown with the initial cluster size noted at each branch tip. Note that at this stage, the **linkage** algorithm has essentially already added most of the configurations to one of two large clusters. In contrast, the **complete** algorithm produces a well-balanced tree. The tree plots shown were generated using software available on the WWW from the Laboratory of Bioinformatics, Wageningen University and Research Center, The Netherlands; see http://www.bioinformatics.nl/tools/plottree.html.

of the important conformational differences or substates are concealed in the collective motion of the entire DNA structure. These data highlight the importance of narrowing the focus to relevant residues for the pairwise comparisons before clustering begins.

Critical Distance. Most clustering algorithms require the user to specify in advance the number of clusters to create. Doing this is difficult as the proper choice of the cluster count will depend on the underlying data. As an example of the difficulty, consider how poor the clustering would be for the trivial example of points in the plane shown in Figure 1 if a cluster count other than three was chosen. To provide users with more guidance on the proper choice of cluster count, we experimented with ways to dynamically choose a correct cluster count. For example, the bottom-up clustering algorithms can be instructed to stop when the intercluster distance or critical distance for the next merge is greater than some threshold ϵ rather than when a preselected number of clusters is reached. This approach has some promise, but it still requires the user to choose a value of ϵ. The appropriate value will be different from one algorithm to another—for instance, an ϵ value of 1.2 Å RMSd applied to **linkage** algorithm produces 12 clusters, while this same ϵ produces 74 clusters when the **centripetal** algorithm is applied, 181 clusters with the **complete** algorithm, and only 1 cluster with the **edge** algorithm. The appropriate value of ϵ also depends on the distance metrics used in the algorithm, the molecular system, and the choice of atoms used for the pairwise comparison.

Cluster Trees. For the bottom-up algorithms, a better approach may be to examine the tree of clusters that results (Figure 13). The cluster tree provides more information than does the individual clustering output at any particular stage. For instance, the cluster tree for a **linkage** clustering of the entire trajectory is not balanced; it consists mainly of two large clusters. For the **edge** and **centripetal** algorithms, the unbalanced tree that results will essentially be one big cluster. With the **complete** and **hierarchical** algorithms, the cluster trees are more balanced. In general, large clusters which remain unchanged for several iterations of the clustering algorithm seem more meaningful than clusters that shift at each stage, producing an unbalanced "bushy" tree.

Efficient Sieved Clustering. Speed and memory considerations make it difficult to cluster large trajectories using algorithms that compute the full pairwise comparison across all configurations. All of the algorithms investigated in this work precomputed an N × N symmetric matrix of the complete set of frame-to-frame distances, where *N* is the number of the frames in the trajectory. The matrix was precomputed since computing these distances on demand greatly increased the runtime. As discussed in the section comparing the relative performance of the different algorithms, computing this matrix is expensive and memory intensive. Even with precomputing the similarity matrix, the calculations quickly become intractable as more and more configurations are to be clustered. For the **SOM**, **COBWEB**, and **Bayesian** algorithms, the similarity matrix is only actually needed to calculate the DBI and pSF metrics; the pairwise comparisons could be calculated on the fly or loaded after the clustering is finished to improve performance. In spite of this, the refinement algorithms quickly become intractable as the trajectories to cluster become very long. To cluster very large trajectories, a better way to enable the efficient clustering is to cluster in a hierarchical fashion, specifically to initially cluster a subset of the data, such as that from a coarser-grained time sampling, with subsequent

partitioning or clustering to put the skipped data into the existing clusters. To test the utility of this approach, we implemented a two-pass, or "sieved", clustering method that initially clusters only part of the data and then on the second pass puts the missing data into existing clusters. With a sieve of *n*, initially clustering every "*n*" configurations, the pairwise-distance matrix is reduced in size by a considerable factor; specifically by n^2. The savings in computer time and memory more than compensate for the expense of making a second pass through the data. The one drawback of sieved clustering is that we may not sample all the conformations of the system during our first pass, especially if the sieve size is too large or if there are periodic components of the data with a period close to the sampling rate. In this case, the clustering output will not accurately partition the data. As a means to mitigate the problem with potential periodicity of the data, we can randomly select points for the first pass.

To assess the advantages and drawbacks of this scheme, we clustered the DB226-DNA hairpin trajectory data using sieves of various sizes. We compared the results and the runtime to the earlier results. Table 3 indicates the differences for the various clustering algorithms between the ordinary and the sieved results. In general, the sieving provides a dramatic decrease in runtime, particularly for the slower algorithms (bottom-up algorithms) where the time required decreases to about $1/n^2$. For the refinement and tree algorithms, including the **means**, **SOM**, **Bayesian**, and **COBWEB** algorithms, the time decreased to about $1/n$. Interestingly, with the top-down algorithm (**hierarchical**), the time decrease only occurs in the initial distance matrix calculation as the actual clustering algorithm is not very computationally demanding when small cluster counts are used. As the output, second pass through the data, and calculation of the statistics takes more time than the clustering, the time savings with this algorithm are modest.

Small sieve sizes (effectively less than 50 ps with this trajectory) produce negligible changes in DBI and pSF values with the **means**, **average-linkage**, **linkage**, and **SOM** algorithms. Additionally, for these algorithms the distances between the sieved clustering and unsieved clustering are small. This is likely a result of the second pass grouping procedure which assigns configurations to the closest centroid with each of these algorithms. Interestingly, the sieved clustering results are sometimes slightly better, with smaller DBI and larger pSF values, than the results obtained when clustering without sieving. This again suggests that the clustering depends on the data set. The data also suggest that the algorithms like **complete** and **COBWEB** seem more dependent on the choice of configurations for the first pass clustering. The small distance between the various distinct sets of clusters, as a function of sieve size, suggest that a sieve size of 5, with sampling every 50 ps, seems to be sufficient with this MD trajectory. The larger the desired number of clusters, the tighter the sieve should be, as rare conformational states (corresponding to smaller clusters) must still be adequately sampled in the first pass through the data. Interestingly, for larger sieve sizes (up to 500 ps), the **average-linkage**, **linkage**, and **SOM** algorithms still perform well, in contrast to the **means** algorithm which

Table 3. Performance of Sieved Clustering on the DNA Hairpin-DB226 MD Trajectory under Various Conditions[a]

algorithm	sieve size	sieve start	total time	DBI	pSF	clustering distance
means	no sieve		1376	2.02	1223	0.000
means	2	1	470	2.03	1223	0.014
means	2	2	386	1.99	1221	0.023
means	2	random1	421	2.00	1221	0.022
means	2	random2	671	2.05	1221	0.013
means	5	1	118	2.03	1220	0.013
means	5	2	140	2.04	1221	0.022
means	5	random1	117	2.02	1221	0.028
means	5	random2	127	2.05	1221	0.030
means	50	1	36	1.62	1007	0.181
means	50	2	33	2.10	1111	0.060
means	50	random1	34	1.99	1168	0.064
means	50	random2	37	2.03	1106	0.060
average	no sieve		6816	1.56	925	0.000
average	2	1	910	1.57	852	0.029
average	2	2	820	1.65	897	0.014
average	2	random1	897	1.66	842	0.069
average	2	random2	790	1.57	941	0.013
average	5	1	112	1.50	855	0.031
average	5	2	115	1.56	955	0.024
average	5	random1	203	1.76	882	0.030
average	5	random2	207	1.56	942	0.016
average	50	1	32	1.53	948	0.027
average	50	2	27	1.58	938	0.031
average	50	random1	33	1.58	973	0.041
average	50	random2	32	1.69	1031	0.081
linkage	no sieve		2149	1.04	805	0.000
linkage	2	1	259	1.49	795	0.019
linkage	2	2	305	1.76	830	0.015
linkage	2	random1	274	1.65	816	0.025
linkage	2	random2	336	1.64	856	0.026
linkage	5	1	70	1.70	798	0.020
linkage	5	2	64	1.88	842	0.041
linkage	5	random1	80	1.76	823	0.026
linkage	5	random2	75	1.65	865	0.030
linkage	50	1	17	1.61	870	0.035
linkage	50	2	18	1.38	926	0.042
linkage	50	random1	17	1.58	945	0.072
linkage	50	random2	18	1.48	856	0.026
SOM	no sieve		1925	2.13	1060	0.000
SOM	2	1	857	2.18	1092	0.019
SOM	2	2	587	2.18	1094	0.015
SOM	2	random1	730	2.17	1090	0.025
SOM	2	random2	546	2.17	1089	0.026
SOM	5	1	224	2.18	1121	0.020
SOM	5	2	281	2.16	1119	0.041
SOM	5	random1	293	2.16	1116	0.026
SOM	5	random2	212	2.16	1121	0.030
SOM	50	1	32	2.15	1132	0.076
SOM	50	2	32	2.09	1104	0.071
SOM	50	random1	31	2.05	1130	0.092
SOM	50	random2	31	2.40	1314	0.285

[a] In each case a cluster count of five was chosen. Various sieve sizes were chosen ranging from no sieve (10 ps sampling), to 2, 5, or 50 (representing 20, 50, or 500 ps sampling, respectively). For each algorithm, different choices of the starting configuration were investigated either with uniform sampling (sieve starting configuration of 1 or 2) or random sampling of the configurations to be clustered. Comparisons of the compute time, DBI, pSF, and clustering distance, relative to the unsieved clustering, show that the sieving process does not drastically alter the outcome.

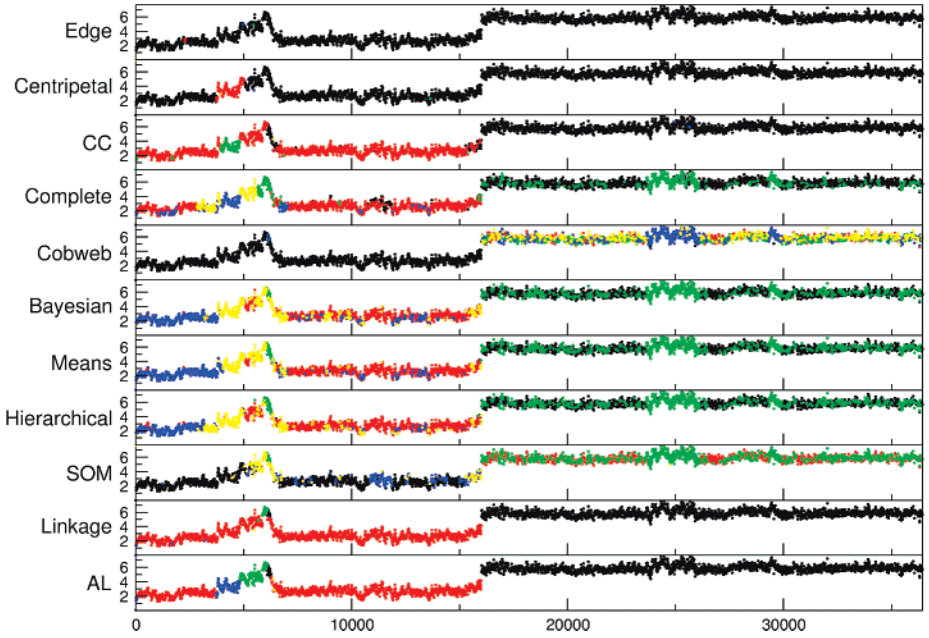

Figure 14. RMSd (Å) versus time (ps) for the DB226-DNA hairpin trajectory for different clustering algorithms. Trajectory data were taken every 10 ps, and a cluster count of 5 was used. The RMSd is the unfit distance (displacement) of the drug after fit the DNA structure to the first frame. The color scheme is based on the size of the cluster with black > red > green > blue > yellow.

sometimes breaks down. However, the performace of means may be improved by running multiple trials with different random selections of configurations for the refinement steps in each run. An additional problem of the **SOM** algorithm is that it may produce fewer clusters than are expected for a particular cluster count. The larger distance of **SOM** with a sieve size of 50 and a sieve start random2 (the last row in the table) is due to the fact that only 4 clusters have been formed during the **SOM** clustering. In addition to the applications to DNA shown, clustering has been applied to a relatively long trajectory of a dynamic protein system. Specifically, this involves a cytochrome P450 2B4 structure that converts from the "open" geometry seen in the crystal to a closed geometry over the course of a ~75 ns simulation. The cluster metrics for clustering over 7000 configurations at 10 ps intervals, a description of the simulation protocols, and RMSd plots and molecular graphics are shown in the Supporting Information.

Summary of the Relative Performance of the Various Clustering Algorithms. Using color to identify each distinct cluster, Figures 14 and 15 show the RMSd or MM-PBSA free energy of binding versus time for the DB226-DNA hairpin trajectories. This provides another means to visualize the relative performance of the various algorithms. From the figure, it is clear that similar RMSd or $\Delta G_{\text{binding}}$ values do not necessarily imply equivalent cluster membership. Also evident is that **edge**, **linkage**, and **centripetal** do not produce very meaningful sets of clusters as only one or two large clusters result. The **means**, **hierarchical**, **complete**, **Bayesian**, and **average-linkage** algorithms, on the other hand, all tend to produce meaningful sets of clusters.

Bottom-up algorithms iteratively merge small clusters into larger ones. With MD simulation data, the algorithms have a tendency to produce outlier or singleton clusters. If the algorithm generates 10 clusters, 9 of which are single points, little is learned about the underlying structure of the data (other than identifying the most extreme conformations). Careful choice of cluster count (see below) is one way to mitigate this sensitivity.

- The **single-linkage** algorithm is rather fragile in that the presence or absence of a single point can control the grouping of the rest. It is very sensitive to lines of closely spaced "breadcrumb" points, which can add arbitrarily long trails of data to one cluster.[52] Although it can handle clusters of differing sizes, the algorithm often does a poor job delineating clusters whose points are very close. Its results were passable on points in the plane but very poor on real MD trajectories.
- **Complete-linkage** and **centripetal-complete** clustering are the two bottom-up clustering algorithms that do not have the tendency to produce singleton clusters. In spite of this, the resulting clusters tend to be small.
- **Centripetal** clustering gives results that are similar, but inferior, to those of **centroid-linkage**. It tends to produce a larger minimum cluster size, since the representatives from a small cluster are not drawn away as far from the 'frontier' as those in a large cluster. The parameters of **centripetal** clustering (the number of representatives per cluster, and the distance they are drawn toward the centroid) may be amenable to further tuning to improve cluster quality.
- **Centroid-linkage** and **average-linkage** clustering gave consistently good results as quantified by the Davies-Bourdin Index (DBI) and pseudo-F statistic (pSF). They can produce clusters of varying sizes and possibly concave shapes. They are two of the most useful of the clustering algorithms we have examined for use with MD trajectories.

Refinement Clustering Algorithms. Because the refinement algorithms include a random factor, we ran the algorithms several times and kept the best (as measured by DBI and pSF) clustering results.

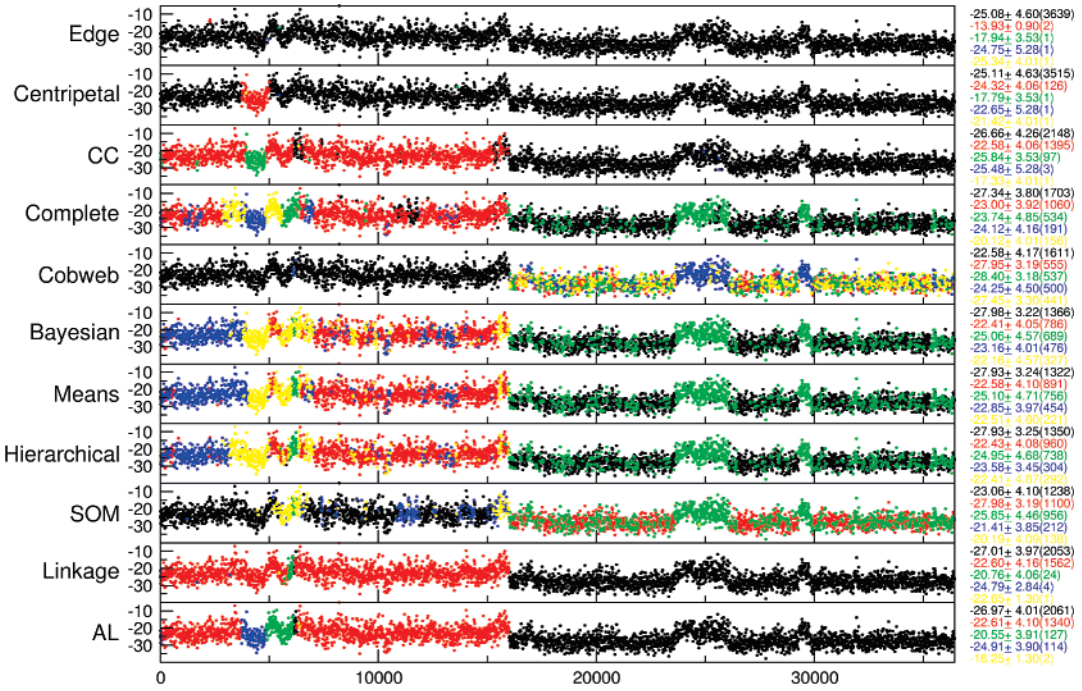

Figure 15. MM-PBSA binding energy versus time (ps) for the DB226-DNA hairpin trajectory for different clustering algorithms. Trajectory data were taken every 10 ps, and a cluster count of 5 was used. The color scheme is based on the size of the cluster with black > red > green > blue > yellow. The data on the right side of the figure show the approximate free energy and standard deviation (kcal/mol) of a cluster. The number in the parentheses is the number of snapshots in that cluster. Very different average free energies (all neglecting solute entropic components) are seen between the different clustering algorithms.

• **Means** clustering tends to produce "blocky" clusters of similar sizes. The seed cluster centroid positions start at the edges of the data set but move toward the eventual centroid over the course of the clustering run. This algorithm cannot produce concave clusters and does not generate clusters of different sizes, but in general it performs very well.

• **Bayesian** clustering produces decent results, but these results become poor for high cluster counts. It can produce clusters of different sizes. **Bayesian** clustering often has difficulty "recognizing" obvious clusters in simple test cases in the plane, even when the algorithm is reseeded and rerun many times. To give good results, the algorithm must be repeated many times with new random seeds. This is computationally expensive, particularly on MD trajectories where there are often hundreds of coordinates and thousands of configurations to consider.

• Self-organizing maps (**SOM**) produced the best results of the refinement algorithms. The performance was more consistent between runs than **Bayesian** clustering. However, self-organizing maps share some of the problems characteristic of the hierarchical clustering algorithm; specifically, the **SOM** algorithm cannot produce concave clusters, and it has difficulty producing clusters of varying sizes.

We also find that the **COBWEB** clustering algorithm is also promising. Visualization of the resulting tree structure, before flattening, can provide hints as to the reasonable number of clusters to specify the data. However, a severe limitation of the **COBWEB** algorithm is that it is highly dependent on the order of the points incorporated into the COBWEB tree. Thus, the variations between multiple **COBWEB** runs are relatively large.

Discussion

We described the development of a series of different clustering algorithms into a C program library, their application to the easy to visualize test case of clustering 2D points on the plane, integration of the clustering algorithms into the **ptraj** trajectory analysis program, and the subsequent application of the various algorithms to a series of contrived and real MD trajectories. Overall, we were rather surprised by the results which clearly show widely different behavior among the various algorithms. Moreover, the performance of a given algorithm is strongly dependent on the choice of cluster count and, less surprisingly, the choice of atoms for the pairwise comparison. On the other hand, the results appeared to only be weakly sensitive to the choice of the pairwise metric when comparing RMSd to DME measures of similarity. Evaluation of the relative performance was made possible through visualization of the results and also through the exploration of various metrics defining the performance. Specifically, low DBI values and high pSF values signal better clustering. Information on the appropriate cluster count comes from analysis of SSR/SST ratios and critical distance measures as a function of cluster count. In order to more efficiently handle very large data sets, a sieving approach was introduced where only a portion of the data is initially clustered, and then the remaining data are added to existing clusters. For the MD simulations investigated in this work, sieves up to 50 ps only moderately alter the outcome. Overall, the best performance was observed with the average-linkage, means, and SOM algorithms. If the cluster count is not known in advance, one of the other algorithms, such as

hierarchical or average-linkage, are recommended. These two also can be used effectively with a distance threshold for separating clusters. In addition to performing reasonably well, it is important to be aware of the limitations or weaknesses of each algorithm, specifically the high sensitivity to outliers with hierarchical, the tendency to generate homogenously sized clusters with means, and the tendency to produce small or singleton clusters with average-linkage and linkage.

Acknowledgment. We would like to thank Carlos Simmerling, Adrian Roitberg, and Holger Gohlke and their group members for testing early versions of the code. We gratefully acknowledge the award of computational time and support from the Pittsburgh Supercomputer Center, the National Center for Supercomputing Applications, the San Diego Supercomputer Center, the TeraGrid, and the University of Kentucky under NRAC and LRAC awards MCA01S027S. We also acknowledge computational time from the Center for High Performance Computing at the University of Utah including resources provided by the NIH (1S10RR17214-01) on the Arches Metacluster. Stephen Tanner would like to acknowledge support from the Department of Pharmaceutics and Pharmaceutical Chemistry's Summer Undergraduate Research Fellowship Program (2002).

Supporting Information Available: More technical descriptions of the cluster algorithms implemented, the parameters for the minor groove binding drug DB226, more detailed comparison of the relative performance and properties of various of the clustering algorithms, schematic cluster trees highlighting relative performance, RMSd plots, and molecular graphics. This material is available free of charge via the Internet at http://pubs.acs.org.

References

(1) van Gunsteren, W. F.; Berendsen, H. J. Molecular dynamics: perspective for complex systems. *Biochem. Soc. Trans.* **1982**, *10*, 301−305.

(2) van Gunsteren, W. F.; Karplus, M. Protein dynamics in solution and in a crystalline environment: a molecular dynamics study. *Biochemistry* **1982**, *21*, 2259−2274.

(3) Kollman, P. A.; Massova, I.; Reyes, C.; Kuhn, B.; Huo, S.; Chong, L.; Lee, M.; Lee, T.; Duan, Y.; Wang, W.; Donini, O.; Cieplak, P.; Srinivasan, J.; Case, D. A.; Cheatham, T. E., III Calculating structures and free energies of complex molecules: Combining molecular mechanics and continuum models. *Acc. Chem. Res.* **2000**, *33*, 889−897.

(4) van Gunsteren, W. F.; Bakowies, D.; Baron, R.; Chandrasekhar, I.; Christen, M.; Daura, X.; Gee, P.; Geerke, D. P.; Glattli, A.; Hunenberger, P. H.; Kastenholz, M. A.; Oostenbrink, C.; Schenk, M.; Trzesniak, D.; van der Vegt, N. F.; Yu, H. B. Biomolecular modeling: Goals, problems, perspectives. *Angew. Chem., Int. Ed.* **2006**, *45*, 4064−4092.

(5) Levitt, M. Molecular dynamics of native protein: I. Computer simulation of trajectories. *J. Mol. Biol.* **1983**, *168*, 595−617.

(6) Karplus, M.; McCammon, J. A. Molecular dynamics simulations of biomolecules. *Nat. Struct. Biol.* **2002**, *9*, 646−652.

(7) Cheatham, T. E., III; Kollman, P. A. Molecular dynamics simulation of nucleic acids. *Ann. Rev. Phys. Chem.* **2000**, *51*, 435−471.

(8) Duan, Y.; Kollman, P. A. Pathways to a protein folding intermediate observed in a 1-microsecond simulation in aqueous solution. *Science* **1998**, *282*, 740−744.

(9) Hansson, T.; Oostenbrink, C.; van Gunsteren, W. F. Molecular dynamics simulations. *Curr. Opin. Struct. Biol.* **2002**, *12*, 190−196.

(10) Tajkhorshid, E.; Aksimentiev, A.; Balabin, I.; Gao, M.; Israelwitz, B.; Phillips, J. C.; Zhu, F.; Schulten, K. Large scale simulation of protein mechanics and function. *Adv. Protein Chem.* **2003**, *66*, 195−247.

(11) Cheatham, T. E., III Simulation and modeling of nucleic acid structure, dynamics and interactions. *Curr. Opin. Struct. Biol.* **2004**, *14*, 360−367.

(12) Feig, M.; Brooks, C. L., III Recent advances in the development and application of implicit solvent models in biomolecule simulations. *Curr. Opin. Struct. Biol.* **2004**, *14*, 217−224.

(13) Wong, C. F.; McCammon, J. A. Protein simulation and drug design. *Adv. Protein Chem.* **2003**, *66*, 87−121.

(14) Rueda, D.; Ferrer-Costa, C.; Meyer, T.; Perez, A.; Camps, J.; Hospital, A.; Gelpi, J. L.; Orozco, M. A consensus view of protein dynamics. *Proc. Natl. Acad. Sci.* **2007**, *104*, 796−801.

(15) Brooks, C. I. Protein and peptide folding explored with molecular simulations. *Acc. Chem. Res.* **2002**, *35*, 447−454.

(16) Daggett, V. Molecular dynamics simulations of the protein unfolding/folding reaction. *Acc. Chem. Res.* **2002**, *35*, 422−449.

(17) Simmerling, C.; Strockbine, B.; Roitberg, A. All-atom structure prediction and folding simulations of a stable protein. *J. Am. Chem. Soc.* **2002**, *124*, 11258−11259.

(18) Pande, V. S.; Baker, I.; Chapman, J.; Elmer, S. P.; Khaliq, S.; Larson, S. M.; Rhee, Y. M.; Shirts, M. R.; Snow, C. D.; Sorin, E. J.; Zagrovic, B. Atomistic protein folding simulations on the submillisecond time scale using worldwide distributed computing. *Biopolymers* **2003**, *68*, 91−109.

(19) Wickstrom, L.; Okur, A.; Song, K.; Hornak, V.; Raleigh, D. P.; Simmerling, C. L. The unfolded state of the villin headpiece helical subdomain: computational studies of the role of locally stabilized structure. *J. Mol. Biol.* **2006**, *360*, 1094−1107.

(20) Day, R.; Daggett, V. Direct observation of microscopic reversibility in single-molecule protein folding. *J. Mol. Biol.* **2006**, *366*, 677−686.

(21) Juraszek, J.; Bolhuis, P. G. Sampling the multiple folding mechanisms of Trp-cage in explicit solvent. *Proc. Natl. Acad. Sci.* **2006**, *103*, 15859−15864.

(22) Eleftheriou, M.; Germain, R. S.; Royyuru, A. K.; Zhou, R. Thermal denaturing of mutant lysozyme with both the OPLSAA and the CHARMM force fields. *J. Am. Chem. Soc.* **2006**, *128*, 13388−13395.

(23) Yoda, T.; Sugita, Y.; Okamoto, Y. Cooperative folding mechanism of a beta-hairpin peptide studied by a multicanonical replica-exchange molecular dynamics simulation. *Proteins* **2007**, *66*, 846−859.

(24) Baumketner, A.; Shea, J. E. The structure of the Alzheimer, amyloid beta 10−35 peptide probed through replica-exchange molecular dynamics simulations in explicit solvent. *J. Mol. Biol.* **2007**, *366*, 275−285.

(25) Chen, H. F.; Luo, R. Binding induced folding in p53-MDM2 complex. *J. Am. Chem. Soc.* **2007**, *129*, 2930−2937.

(26) Paschek, D.; Nymeyer, H.; Garcia, A. E. Replica exchange simulation of reversible folding/unfolding of the Trp-cage miniprotein in explicit solvent: on the structure and possible role of internal water. *J. Struct. Biol.* **2007**, *157*, 524−533.

(27) Li, W.; Zhang, J.; Wang, W. Understanding the folding and stability of a zinc finger-based full sequence design protein with replica exchange molecular dynamics simulations. *Proteins* **2007**, *67*, 338−349.

(28) Periole, X.; Mark, A. E. Convergence and sampling efficiency in replica exchange simulations of peptide folding in explicit solvent. *J. Chem. Phys.* **2007**, *126*, 014903.

(29) Scheraga, H. A.; Khalili, M.; Liwo, A. Protein-folding dynamics: overview of molecular simulation techniques. *Ann. Rev. Phys. Chem.* **2007**, *58*, 57−83.

(30) Spackova, N.; Cheatham, T. E., III; Ryjacek, F.; Lankas, F.; van Meervelt, L.; Hobza, P.; Sponer, J. Molecular dynamics simulations and thermodynamic analysis of DNA-drug complexes. Minor groove binding between 4′,6-diamidino-2-phenylindole (DAPI) and DNA duplexes in solution. *J. Am. Chem. Soc.* **2003**, *125*, 1759−1769.

(31) Bui, J. M.; McCammon, J. A. Protein complex formation by acetylcholinesterase and the neurotoxin fasciculin-2 appears to involve an induced-fit mechanism. *Proc. Natl. Acad. Sci.* **2006**, *103*, 15451−15456.

(32) Lu, Y.; Yang, C. Y.; Wang, S. Binding free energy contributions of interfacial waters in HIV-1 protease/inhibitor complexes. *J. Am. Chem. Soc.* **2006**, *128*, 11830−11839.

(33) Xu, Y.; Wang, R. A computational analysis of the binding affinities of FKBP12 inhibitors using the MM-PB/SA method. *Proteins* **2006**, *64*, 1058−1068.

(34) de Jonge, M. R.; Koymans, L. H.; Guillemont, J. E.; Koul, A.; Andries, K. A computational model of the inhibition of Mycobacterium, tuberculosis ATPase by a new drug candidate R207910. *Proteins* **2007**, *67*, 971−980.

(35) Ode, H.; Matsuyama, S.; Hata, M.; Hoshino, T.; Kakizawa, J.; Sugiura, W. Mechanism of drug resistance due to N88S in CRF01_AE HIV-1 protease, analyzed by molecular dynamics simulations. *J. Med. Chem.* **2007**, *50*, 1768−1777.

(36) Hornak, V.; Okur, A.; Rizzo, R. C.; Simmerling, C. HIV-1 protease flaps spontaneously open and reclose in molecular dynamics simulations. *Proc. Natl. Acad. Sci.* **2006**, *103*, 915−920.

(37) Hornak, V.; Okur, A.; Rizzo, R. C.; Simmerling, C. HIV-1 protease flaps spontaneously close to the correct structure in simulations following manual placement of an inhibitor into the open state. *J. Am. Chem. Soc.* **2006**, *128*, 2812−2813.

(38) Lankas, F.; Lavery, R.; Maddocks, J. H. Kinking occurs during molecular dynamics simulations of small DNA minicircles. *Structure* **2006**, *14*, 1527−1534.

(39) Noy, A.; Perez, A.; Laughton, C. A.; Orozco, M. Theoretical study of large conformational transitions in DNA: the B<−> A conformational change in water and ethanol/water. *Nucl. Acids Res.* **2007**, *35*, 3330−3338.

(40) van der Vaart, A.; Karplus, M. Minimum free energy pathways and free energy profiles for conformational transitions based on atomistic molecular dynamics simulations. *J. Chem. Phys.* **2007**, *126*, 164106.

(41) Noe, F.; Horenko, I.; Schutte, C.; Smith, J. C. Hierarchical analysis of conformational dynamics in biomolecules: transition networks of metastable states. *J. Chem. Phys.* **2007**, *126*, 155102.

(42) Li, D. W.; Han, L.; Huo, S. Structural and pathway complexity of beta-strand reorganization within aggregates of human transthyretin(105−115) peptide. *J. Phys. Chem. B* **2007**, *111*, 5425−5433.

(43) Patel, S.; Balaji, P. V.; Sasidhar, Y. U. The sequence TGAAKAVALVL from glyceraldehyde-3-phosphate dehydrogenase displays structural ambivalence and interconverts between alpha-helical and beta-hairpin conformations mediated by collapsed conformational states. *J. Pept. Sci.* **2007**, *13*, 314−326.

(44) Roccatano, D.; Barthel, A.; Zacharias, M. Structural flexibility of the nucleosome core particle at atomic resolution studied by molecular dynamics simulation. *Biopolymers* **2007**, *85*, 407−421.

(45) Sefcikova, J.; Krasovska, M. V.; Sponer, J.; Walter, N. G. The genomic HDV ribozyme utilizes a previously unnoticed U-turn motif to accomplish fast site-specific catalysis. *Nucl. Acids Res.* **2007**, *35*, 1933−1946.

(46) Razga, F.; Zacharias, M.; Reblova, K.; Koca, J.; Sponer, J. RNA kink-turns as molecular elbows: hydration, cation binding, and large-scale dynamics. *Structure* **2006**, *14*, 825−835.

(47) Kormos, B. L.; Baranger, A. M.; Beveridge, D. L. A study of collective atomic fluctuations and cooperativity in the U1A-RNA complex based on molecular dynamics simulations. *J. Struct. Biol.* **2007**, *157*, 500−513.

(48) Karpen, M. E.; Tobias, D. J.; Brooks, C. L., III Statistical clustering techniques for the analysis of long molecular dynamics trajectories: analysis of a 2.2-ns trajectories of YPGDV. *Biochemistry* **1993**, *32*, 412−420.

(49) Shenkin, P. S.; McDonald, D. Q. Cluster analysis of molecular conformations. *J. Comput. Chem.* **1994**, *15*, 899−916.

(50) Cormack, R. M. A review of classification. *J. R. Stat. Soc. A* **1971**, *134*, 321−367.

(51) Jain, A. K.; Murty, M. N.; Flynn, P. J. Data clustering: A review. *ACM Comp. Surv.* **1999**, *31*, 264−323.

(52) Torda, A. E.; van Gunsteren, W. F. Algorithms for clustering molecular dynamics configurations. *J. Comput. Chem.* **1994**, *15*, 1331−1340.

(53) Marchionini, C.; Maigret, B.; Premilat, S. Models for the conformational behaviour of angiotensin-II in acidic aqueous solutions. *Biochem. Biophys. Res. Comm.* **1983**, *112*, 339−346.

(54) Willett, P. *Similarity and clustering in chemical information systems*; John Wiley & Sons, Inc.: New York, 1987; Vol 1, p 266.

(55) Kreissler, M.; Pesquer, M.; Maigret, B.; Fournie-Zaluski, M. C.; Roques, B. P. Computer simulation of the conformational behavior of cholecystokinin fragments: Conformational families of sulfated CCK8. *J. Comput.-Aided Mol. Des.* **1989**, *3*, 85−94.

(56) Unger, R.; Harel, D.; Wherland, S.; Sussman, J. L. A 3D building blocks approach to analyzing and predicting structure of proteins. *Proteins* **1989**, *5*, 355−373.

(57) Gordon, H. L.; Somorjai, R. L. Fuzzy cluster analysis of molecular dynamics trajectories. *Proteins* **1992**, *14*, 249−264.

(58) Michel, A.; Jeandenans, C. Multiconformational, investigations of polypeptidic structures, using clustering methods and principal component analysis. *Comput. Chem.* **1993**, *17*, 49−59.

(59) Troyer, J. M.; Cohen, F. E. Protein conformational landscapes: energy minimization and clustering of a long molecular dynamics trajectory. *Proteins* **1995**, *23*, 97−110.

(60) Daura, X.; van Gunsteren, W. F.; Mark, A. E. Folding-unfolding thermodynamics of a b-heptapeptide from equilibrium simulations. *Proteins* **1999**, *34*, 269−280.

(61) Gabarro-Arpa, J.; Revilla, R. Clustering of a molecular dynamics trajectory with a Hamming distance. *Comput. Chem.* **2000**, *24*, 696−698.

(62) Watts, C. R.; Mezei, M.; Murphy, R. F.; Lovas, S. Conformational space comparison of GnRH and lGnRH-III using molecular dynamics, cluster analysis and Monte Carlo thermodynamic integration. *J. Biomol. Struct. Dyn.* **2001**, *18*, 733−748.

(63) Laboulais, C.; Ouali, M.; Le Bret, M.; Gabarro-Arpa, J. Hamming distance geometry of a protein conformational space: Application, to the clustering of a 4-ns molecular dynamics trajectory of the HIV-1 integrase catalytic core. *Proteins* **2002**, *47*, 169−179.

(64) Feher, M.; Schmidt, J. M. Fuzzy clustering as a means of selecting representative conformers and molecular alignments. *J. Chem. Inf. Comput. Sci.* **2003**, *43*, 810−818.

(65) Bystroff, C.; Garde, S. Helix propensities of short peptides: Molecular, dynamics versus Bioinformatics. *Proteins* **2003**, *50*, 552−562.

(66) Moraitakis, G.; Goodfellow, J. M. Simulations of human lysozyme: Probing, the conformations triggering amyloidosis. *Biophys. J.* **2003**, *84*, 2149−2158.

(67) Lee, M. C.; Deng, J.; Briggs, J. M.; Duan, Y. Large-scale conformational dynamics of the HIV-1 integrase core domain and its catalytic loop mutants. *Biophys. J.* **2005**, *88*, 3133−3146.

(68) Rao, F.; Settanni, G.; Guarnera, E.; Caflisch, A. Estimation of protein folding probability from equilibrium simulations. *J. Chem. Phys.* **2005**, *122*, 184901.

(69) Lyman, E.; Zuckerman, D. M. Ensemble-based convergence analysis of biomolecular trajectories. *Biophys. J.* **2006**, *91*, 164−172.

(70) Sullivan, D. C.; Lim, C. Quantifying polypeptide conformational space: sensitivity to conformation and ensemble definition. *J. Phys. Chem. B* **2006**, *110*, 16707−16717.

(71) Li, Y. Bayesian model based clustering analysis: application to a molecular dynamics trajectory of the HIV-1 integrase catalytic core. *J. Chem. Inf. Model.* **2006**, *46*, 1742−1750.

(72) Elmer, S. P.; Pande, V. S. Foldamer simulations: Novel, computational methods and applications to poly-phenylacetylene oligomers. *J. Chem. Phys.* **2004**, *121*, 12760−12771.

(73) Sorin, E. J.; Pande, V. S. Exploring the helix-coil transition via all-atom equilibrium ensemble simulations. *Biophys. J.* **2005**, *88*, 2472−2493.

(74) Sims, G. E.; Choi, I.-G.; Kim, S.-H. Protein conformational space in higher order phi-psi maps. *Proc. Natl. Acad. Sci.* **2005**, *102*, 618−621.

(75) Satoh, D.; Shimizu, K.; Nakamura, S.; Terada, T. Folding free-energy landscape of a 10-residue mini-protein, chignolin. *FEBS Lett.* **2006**, *580*, 3422−3426.

(76) Scott, E. E.; He, Y. A.; Wester, M. R.; White, M. A.; Chin, C. C.; Halpert, J. R.; Johnson, E. F.; Stout, C. D. An open conformation of mammalian cytochrome P450 2B4 at 1.6-A resolution. *Proc. Natl. Acad. Sci.* **2003**, *100*, 13196−13201.

(77) Poncin, M.; Hartmann, B.; Lavery, R. Conformational substates in B-DNA. *J. Mol. Biol.* **1992**, *226*, 775−794.

(78) Srinivasan, J.; Cheatham, T. E., III; Cieplak, P.; Kollman, P. A.; Case, D. A. Continuum solvent studies of the stability of DNA, RNA and phosphoramidate helices. *J. Am. Chem. Soc.* **1998**, *120*, 9401−9409.

(79) Schlitter, J. Estimation of absolute and relative entropies of macromolecules using the covariance matrix. *Chem. Phys. Lett.* **1993**, *215*, 617−621.

(80) Harris, S. A.; Gavathiotis, E.; Searle, M. S.; Orozco, M.; Laughton, C. A. Cooperativity in drug-DNA recognition: a molecular dynamics study. *J. Am. Chem. Soc.* **2001**, *123*, 12658−12663.

(81) Fisher, D. H. In *Improving inference through conceptual clustering*; AAAI: Seattle, WA, 1987; pp 461−465.

(82) Fisher, D. Knowledge acquisition via incremental conceptual clustering. *Machine Learning* **1987**, *2*, 139−172.

(83) Cheeseman, P.; Stutz, J. Bayesian classification (AutoClass): theory and results. In *Advances in knowledge discovery and data mining*; Fayyad, U. M., Piatetsky-Shapiro, G., Smyth, P., Uthurusamy, R., Eds.; American Association of Artificial Intelligence Press: Menlo Park, CA, 1996; pp 61−83.

(84) Kohonen, T. *Self-organizing maps*, 3rd ed.; Springer: Berlin-Heidelberg, 2001; Vol. 30, p 501.

(85) Pearlman, D. A.; Case, D. A.; Caldwell, J. W.; Ross, W. S.; Cheatham, T. E.; Debolt, S.; Ferguson, D.; Seibel, G.; Kollman, P. AMBER, a package of computer programs for applying molecular mechanics, normal mode analysis, molecular dynamics and free energy calculations to simulate the structure and energetic properties of molecules. *Comp. Phys. Comm.* **1995**, *91*, 1−41.

(86) Case, D. A.; Cheatham, T. E., III; Darden, T. A.; Gohlker, H.; Luo, R.; Merz, K. M., Jr.; Onufriev, A. V.; Simmerling, C.; Wang, B.; Woods, R. The AMBER biomolecular simulation programs. *J. Comput. Chem.* **2005**, *26*, 1668−1688.

(87) Guha, S.; Rastogi, R.; Shim, K. In *CURE: An efficient clustering algorithm for large databases*; Proceedings of the ACM SIGMOD International Conference on Management of Data: New York, 1998; pp 73−84.

(88) Witten, I. H.; Frank, E. *Data mining: Practical machine learning tools and techniques with Java implementations*; Morgan Kaufmann: 1999; p 525.

(89) Kohonen, T. *Self-organization and Associative Memory*; Springer-Verlag: Berlin, 2001; Vol. 30, p 501.

(90) Davies, D. L.; Bouldin, D. W. A cluster separation measure. *IEEE Trans. Pattern Anal. Mach. Intelligence* **1979**, *1*, 224−227.

(91) Vesanto, J.; Alhoniemi, E. Clustering of the self-organizing map. *IEEE Trans. Neural Networks* **2000**, *11*, 586−600.

(92) Bolshakova, N.; Azuaje, F. *Cluster validation techniques for genome expression data*; University of Dublin, Trinity College: Dublin, 2002; p 13.

(93) Speer, N.; Spiet, C.; Zell, A. Biological cluster validity indices based on the gene ontology. In *Advances in intelligent data analysis VI*; Famili, A. F., Kok, J. N., Pena, J. M., Siebes, A., Feelders, A., Eds.; Springer: Berlin, Heidelberg, 2005; Vol. 3646, pp 429−439.

(94) Calinski, T.; Harabasz, J. A dendrite method for cluster analysis. *Comm. Stat.* **1974**, *3*, 1−27.

(95) Mitchell, T. *Machine Learning*; McGraw-Hill: 1997; p 432.

(96) Ryckaert, J. P.; Ciccotti, G.; Berendsen, H. J. C. Numerical integration of the cartesian equations of motion of a system with constraints: Molecular dynamics of n-alkanes. *J. Comp. Phys.* **1977**, *23*, 327−341.

(97) Berendsen, H. J. C.; Postma, J. P. M.; van Gunsteren, W. F.; DiNola, A.; Haak, J. R. Molecular dynamics with coupling to an external bath. *J. Comp. Phys.* **1984**, *81*, 3684−3690.

(98) Cornell, W. D.; Cieplak, P.; Bayly, C. I.; Gould, I. R.; Merz, K. M.; Ferguson, D. M.; Spellmeyer, D. C.; Fox, T.; Caldwell, J. W.; Kollman, P. A. A second generation force field for the simulation of proteins, nucleic acids, and organic molecules. *J. Am. Chem. Soc.* **1995**, *117*, 5179−5197.

(99) Jorgensen, W. L.; Chandrasekhar, J.; Madura, J. D.; Impey, R. W.; Klein, M. L. Comparisons of simple potential functions for simulating liquid water. *J. Chem. Phys.* **1983**, *79*, 926−935.

(100) Aqvist, J. Ion-water interaction potentials derived from free energy perturbation simulations. *J. Phys. Chem.* **1990**, *94*, 8021−8024.

(101) Cheatham, T. E., III; Srinivasan, J.; Case, D. A.; Kollman, P. A. Molecular dynamics and continuum solvent studies of the stability of polyG-polyC and polyA-polyT DNA duplexes in solution. *J. Biomol. Struct. Dyn.* **1998**, *16*, 265−280.

(102) Wu, X. W.; Wang, S. M. Self-guided molecular dynamics simulation for efficient conformational search. *J. Phys. Chem.* **1998**, *102*, 7238−7250.

(103) Wu, X.; Wang, S. Helix Folding of an Alanine-Based Peptide in Explicit Water. *J. Phys. Chem. B* **2001**, *105*, 2227−2235.

(104) Wu, X.; Brooks, B. R. Beta-hairpin folding mechanism of a nine-residue peptide revealed from molecular dynamics simulations in explicit water. *Biophys. J.* **2004**, *86*, 1946−1958.

(105) Wu, X.; Wang, S.; Brooks, B. R. Direct observation of the folding and unfolding of a beta-hairpin in explicit water through computer simulation. *J. Am. Chem. Soc.* **2002**, *124*, 5282−5283.

(106) Pettersen, E. F.; Goddard, T. D.; Huang, C. C.; Couch, G. S.; Greenblatt, D. M.; Meng, E. C.; Ferrin, T. E. UCSF Chimera−a visualization system for exploratory research and analysis. *J. Comput. Chem.* **2004**, *25*, 1605−1612.

(107) Boykin, D. W.; Kumar, A.; Xiao, G.; Wilson, W. D.; Bender, B. C.; McCurdy, D. R.; Hall, J. E.; Tidwell, R. R. 2,5-bis-[4-(N-alkylamidino)phenyl]furans as anti-Pneumocystis carinii agents. *J. Med. Chem.* **1998**, *41*, 124−129.

(108) Wilson, W. D.; Tanious, F. A.; Ding, D.; Kumar, A.; Boykin, D. W.; Colson, P.; Houssier, C.; Bailly, C Nucleic acid interactions of unfused aromatic cations: Evaluation of proposed minor-groove, major-groove, and intercalation binding modes. *J. Am. Chem. Soc.* **1998**, *120*, 10310−10321.

(109) Mazur, S.; Tanious, F. A.; Ding, D.; Kumar, A.; Boykin, D. W.; Simpson, I. J.; Neidle, S.; Wilson, W. D. A thermodynamic and structural analysis of DNA minor-groove complex formation. *J. Mol. Biol.* **2000**, *300*, 321−337.

(110) Hawkins, G. D.; Cramer, C. J.; Truhlar, D. G. Pairwise solute descreening of solute charges from a dielectric medium. *Chem. Phys. Lett.* **1995**, *246*, 122−129.

(111) Tsui, V.; Case, D. A. Molecular dynamics simulations of nucleic acids with a generalized Born solvation model. *J. Am. Chem. Soc.* **2000**, *122*, 2489−2498.

(112) Wang, J.; Wang, W.; Kollman, P. A.; Case, D. A. Automatic atom type and bond type perception in molecular mechanical calculations. *J. Mol. Graphics Modell.* **2006**, *25*, 247−260.

(113) Wang, J.; Kollman, P. A. Automatic parameterization of force field by systematic search and genetic algorithms. *J. Comput. Chem.* **2001**, *22*, 1219−1228.

(114) Bayly, C. I.; Cieplak, P.; Cornell, W. D.; Kollman, P. A. A well-behaved electrostatic potential based method using charge restraints for deriving atomic charges- the RESP model. *J. Phys. Chem.* **1993**, *97*, 10269−10280.

(115) Frisch, M. J.; Trucks, G. W.; Schlegel, H. B.; Scuseria, G. E.; Robb, M. A.; J. R. Cheeseman, V. G. Z.; Montgomery, J. A., Jr.; Stratmann, R. E.; Burant, J. C.; Dapprich, S.; Millam, J. M.; Daniels, A. D.; Kudin, K. N.; Strain, M. C.; Farkas, O.; Tomasi, J.; Barone, V.; Cossi, M.; Cammi, R.; Mennucci, B.; Pomelli, C.; Adamo, C.; Clifford, S.; Ochterski, J.; Petersson, G. A.; Ayala, P. Y.; Cui, Q.; Morokuma, K.; Salvador, P.; Dannenberg, J. J.; Malick, D. K.; Rabuck, A. D.; Raghavachari, K.; Foresman, J. B.; Cioslowski, J.; Ortiz, J. V.; Baboul, A. G.; Stefanov, B. B.; Liu, G.; Liashenko, A.; Piskorz, P.; Komaromi, I.; Gomperts, R.; Martin, R. L.; Fox, D. J.; Keith, T.; Al-Laham, M. A.; Peng, C. Y.; Nanayakkara, A.; Challacombe, M.; Gill, P. M. W.; Johnson, B.; Chen, W.; Wong, M. W.; Andres, J. L.; Gonzalez, C.; Head-Gordon, M.; Replogle, E. S.; Pople, J. A. *Gaussian 98 (Revision A.10)*; Gaussian, Inc.: Pittsburgh, PA, 2001.

(116) Laughton, C. A.; Tanious, F. A.; Nunn, C. M.; Boykin, D. W.; Wilson, W. D.; Neidle, S. A crystallographic and spectroscopic study of the complex between d(CGCGAATTCGCG)2 and 2,5-bis(4-guanylphenyl)furan, an analogue of Berenil: structural origins of enhanced DNA-binding affinity. *Biochemistry* **1996**, *35*, 5655−5661.

CT700119M