

MDMap : A System for Data-Driven Layout and Exploration of Molecular Dynamics Simulations

Robert Patro Cheuk Yiu Ip Sujal Bista *
Department of Computer Science
Institute for Advanced Computer Studies
University of Maryland, College Park

D. Thirumalai ‡
Biophysics Program
Institute for Physical Sciences and Technology
University of Maryland, College Park

Samuel S. Cho †
Departments of Physics and Computer Science
Wake Forest University

Amitabh Varshney §
Department of Computer Science
Institute for Advanced Computer Studies
University of Maryland, College Park

Abstract

Contemporary molecular dynamics simulations result in a glut of simulation data, making analysis and discovery a difficult and burdensome task. We present *MDMap*, a system designed to summarize long-running molecular dynamics (MD) simulations. We represent a molecular dynamics simulation as a state transition graph over a set of intermediate (stable and semi-stable) states. The transitions amongst the states together with their frequencies represent the *flow* of a biomolecule through the trajectory space. *MDMap* automatically determines potential intermediate conformations and the transitions amongst them by analyzing the conformational space explored by the MD simulation. *MDMap* is an automated system to visualize MD simulations as state-transition diagrams, and can replace the current tedious manual layouts of biomolecular folding landscapes with an automated tool. The layout of the representative states and the corresponding transitions among them is presented to the user as a visual synopsis of the long-running MD simulation. We compare and contrast multiple presentations of the state transition diagrams, such as conformational embedding, and spectral, hierarchical, and force-directed graph layouts. We believe this system could provide a road-map for the visualization of other stochastic time-varying simulations in a variety of different domains.

Keywords: Molecular dynamics, Protein folding, Time varying visualization, Graph layout, Bioinformatics, Clustering.

1 Introduction

The question of how biomolecules fold into their active structural state is one of the greatest unsolved problems in biology [2, 23]. Given that these processes are complex, computational simulations of biomolecules are critical for understanding the underlying principles determining their behavior. Such molecular dynamics (MD) simulations have become indispensable tools in biophysics and material science. However these simulations are often very long with tens of thousands to millions of time steps and the visual tools that help scientists comprehend MD simulations are still rudimentary. Although at first it may appear that visualizing time-varying simulations as they occur should be effective at conveying and understanding change over time, recent research presents evidence to the contrary. Tversky *et al.* [43] have shown that animations may in fact be less suited than static images for conveying physical phenomena that are too complex or too fast to be properly understood.

Biomolecular dynamics simulations are fairly complex with multiple moving components and have intra-molecular interactions and motions that occur over different time scales. Previous approaches for summarization of time-varying MD simulations have largely focused on one-dimensional temporal summarization by extraction of a small number of salient frames. In this work we present a different, 2D-layout-based approach for summarization of MD simulations that is inspired by the leading theories in the field of biomolecular folding.

It is now well established that biomolecules navigate a complex folding landscape in search of the folded structure. In this view, there exist a myriad of different unfolded structures from which biomolecules can start folding towards a native basin of attraction that is characterized by a well-defined conformation that corresponds to the folded structure [41, 47]. As such, for a single biomolecule, there exist many possible sets of parallel folding pathways for progressing from one of the many possible unfolded structures to the folded structure. These pathways can consist of multiple long-lived and highly populated intermediate states that a biomolecule may visit [7, 42].

In this paper we present a system, *MDMap* to visualize a molecular dynamics trajectory by representing it as a set of paths through a state transition graph. In this approach, the set of intermediate conformations of the biomolecule in the folding process become the states, the transitions amongst the states together with their frequencies represent the *flow* of a biomolecule through the trajectory space, and an automatic 2D layout represents a novel visualization of the MD simulation of the folding process. *MDMap* involves:

1. Determination of conformational states: We present an algorithm based on hierarchical agglomerative clustering, which identifies conformational states based on a matrix of root mean squared distances (RMSDs) between the molecular structures in a trajectory. This approach results in a cluster hierarchy, which also enables multi-level exploration of conformational states.

2. Discovery of the state transition graph: By observing how the identified states change over the course of a simulation, we infer the edges of the state transition graph. How often a particular transition occurs determines the thickness of the transition edge. Simple detail elision algorithms facilitate interactive discovery of the most frequent state transitions in the trajectory space of molecules while minimizing visual clutter.

3. Automatic Layout of the state transition graph: We present 2D layouts of the state transition graph discovered by our system, including conformational embedding, spectral, hierarchical, and force-directed layouts. We discuss the strengths and weaknesses of each layout for our target application.

We make use of several existing graph layouts. However, the novelty of our method resides in the representation and visualization of molecular dynamics trajectories as state-transition graphs; this has not been explored in the existing visualization literature.

*e-mail: {rob,ipcy,sujal}@cs.umd.edu

†e-mail: choss@wfu.edu

‡e-mail: thirum@umd.edu

§e-mail: varshney@cs.umd.edu

2 Background and Related Work

Description of self-organization principles in biology require a detailed understanding of how biomolecules arrange themselves into well-defined structures and interact with others. Among the most commonly studied biomolecules using MD simulations are proteins and RNA, due to their importance in virtually all cellular processes. Every organism depends on the ability of proteins and RNA molecules to undergo structural changes (i.e., fold) into specific, active structures that correspond to a host of critical biological functions in the cell including catalysis, regulation, and ligand binding.

Protein Folding Analysis Noe and Fisher [33] used hierarchical transition network to classify and visualize metastable states of peptides. Diego *et al.* [36] maps MD trajectories into a Conformational Markov Network based on the potential energy which allowed them to explore Free Energy Landscape. In our work, we use conformational distance to perform agglomerative clustering and provide an efficient visualization tool. We use multi-level hierarchical network which provides both higher level overview of the stable states and the detailed view within each stable states. In an independent development to our work, Voelz *et al.* [45] have very recently applied data mining methodology to simulation data to study the folding pathway of the protein NTL9(1-39). Though the computational mechanisms employed by Voelz *et al.* are different, this work is similar in spirit to the approach taken by Guo and Thirumalai [14] and Klimov and Thirumalai [20], in which the authors apply a neural net algorithm to cluster the structures in a trajectory such that a distance metric would classify similar structures into a predetermined number of states. While their work focuses primarily on the determination of intermediate states, our work also stresses the visualization and layout of the state transition graph arising from molecular dynamics trajectories.

Biomolecular Visualization Visualization has played a vital role in helping understand the relationship between structure and function of biomolecules such as proteins, as well as in giving us important insights into the folding process itself. Generation and visualization of molecular surfaces is one such area [4, 8, 22, 38, 44]. Potzsch *et al.* [35] have presented a system for visualization of lattice-based protein-folding simulations by employing multiple views, focus+context, and table lenses in a very effective manner. Tools such as VMD [16] have greatly assisted by crafting new visualization solutions as well as dissemination of proven ones.

From 1D Summarization to 2D Layouts One of the natural avenues for enhancing the descriptive power of summarization is moving from a 1D to a 2D layout. For instance, in computational genomics, early efforts such as Artemis [37] and VISTA [27] focused on displaying the genomic data in a one-dimensional layout. In two-dimensional layouts, Nielsen *et al.* [31] present their system ABySS-Explorer to show the global assembly structure by laying out short DNA sub-sequences as a graph. Meyer *et al.* [29] visualize multiple genomes for comparison in a circular layout with MizBee. Barsky *et al.* [5] apply the interaction graph model to visualize molecular immunology data.

Temporal Summarization Girgensohn and Boreczky [13] summarize long video sequences by hierarchical clustering. They provide interactive visualization of these temporally-constrained key frames. Assa *et al.* [3] extracts important poses from skeletal animations by embedding the high dimensional animation parameters into a curve in low dimensional space. The extrema of the curve represents the extreme poses. Lu and Shen [25] present an efficient way to visualize large time-varying datasets using the metaphor of an interactive storyboard. Our work is different in its scope from previous work, as we attempt to rediscover and lay out the state transition graph of a MD simulation.

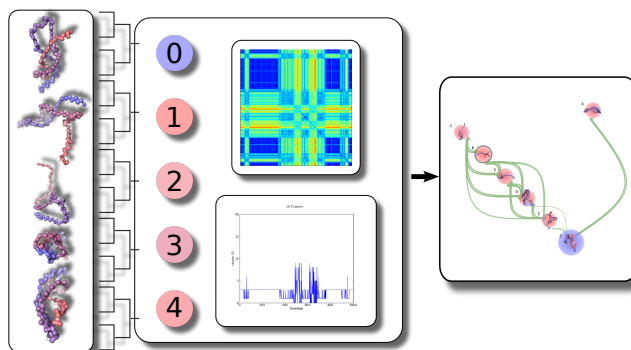


Figure 1: The *MDMap* system discovers the state transition graph for molecular dynamics (MD) simulations to produce visual layouts highlighting different aspects of the simulation.

Motion Depiction We aim to depict molecular motions to better our understanding in protein folding. Recent research in cognitive psychology [43] suggests that static visualizations are more effective at conveying information than animations. Rapid and complex interactions of different moving components of a molecule undergoing folding, hinder its accurate perception. This may be due to violation of what Tversky *et al.* [43] argue is the Apprehension principle of good depiction, according to which “graphics should be accurately perceived and appropriately conceived”. A static diagram of the important steps in a folding trajectory may be more effective because it allows comparison and re-inspection of the details of the motions.

In fact, designers and artists have long used static images to illustrate dynamics of scenes for motion. They have depicted dynamics to facilitate visual communication in comic books and storyboards [28]. The selection of key-frames and their relationship to the semantics of the perceived action is explored in depth by Whitaker and Halas [46]. Several researchers [19, 32, 34] have proposed illustration-based techniques to depict the dynamics of time-varying data in a compact way. Using principles inspired by visual art they generate one or a few images that are augmented by illustrative glyphs to visually communicate time-varying data. For instance, Joshi and Rheingans [19] have presented a method to convey change over time using illustration-inspired techniques such as the use of speedlines, flow ribbons, and strobe silhouettes. Nienhaus and Dollner [32] have used dynamic glyphs such as directed acyclic graphs and behavior graphs to provide further information about dynamics in the 3D scene.

3 Overview

The *MDMap* system consists of both analysis and presentation phases (Figure 1). *MDMap* first attempts to discover a set of intermediate conformations in the given simulation data. These intermediate conformations are semi-stable, and thus, can be characterized as persistent or repeated conformations which occur over the course of the simulation. We approach this problem using a hierarchical clustering approach to discover representative conformations; this procedure is detailed in section 4. After we have identified a set of intermediate states, we analyze the transitions among them. These transitions form the edges in our graphical representation of the simulation data. More formally, the state transition graph, $\mathcal{G} = (\mathcal{S}, \mathcal{E})$, is given by a set, \mathcal{S} , of conformational states discovered by our algorithm along with a set, \mathcal{E} , of transitions amongst these states, which are weighted by the frequency of their occurrence. The procedure for identifying and analyzing these transitions is given in section 5.

Once the analysis phase is complete, the data can be presented to the user in a number of different ways. In section 7, we illustrate

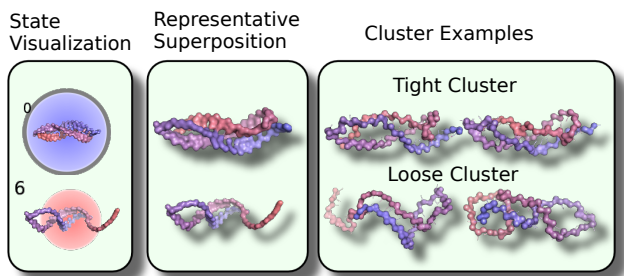


Figure 2: The ability of a representative to visually depict the conformations of the timesteps assigned to its cluster is dependent upon the distribution of intra-cluster distances. We refer to a cluster with a low mean intra-cluster distance as a *tight* cluster (top row) and one with a high intra-cluster distance as a *loose* cluster (bottom row). Both of these clusters were taken from the 1YMO MD simulation .

a number of different possible layouts of the state transition graph. Each of these layouts is augmented by a number of mechanisms designed to help the user comprehend the distributional nature of the selected states. We also present a level-of-detail mechanism which can be used to filter edges and reduce visual clutter in a layout. In section 8, we present the results of *MDMap* for a number of different data sets, showing, for each, some of the different layouts which can be produced for the state transition graph. Finally, in section 9, we present a discussion of the benefits and detriments of the different layouts of the state transition graph, and suggest how future work may combine the benefits provided by different layout strategies while avoiding their individual detriments. We close with a broader vision for the goals of our future work.

Throughout this paper, unless otherwise specified, we are using trajectories from a MD simulation of the folding of the human telomerase RNA pseudoknot (1YMO) as a running illustrative example. Telomerase is a ribonucleoprotein complex that adds DNA sequence repeats called telomeres to the ends of chromosomes in almost all cancerous cells. In normal cells, telomeres are successively cleaved at each generation of cell division until it reaches a critical length, and the cell is given the signal to degrade and die [26]. Telomerase activity effectively forces the cell to continue reproducing indefinitely, resulting in cancer. The RNA pseudoknot component of the telomerase enzyme has been shown to be critical for its activity [39], and properly characterizing its mechanism for folding is fundamental for understanding its biological activity [7, 39].

4 Intermediate State Identification

MDMap first identifies different conformational states in the MD simulation. Ideally, we seek to identify intermediate conformations of the protein. These are semi-stable conformations which persist for a period of time, and which may be visited repeatedly. Driven by this notion of an intermediate conformation, we choose to take a clustering approach to identify candidate states.

4.1 Conformational Distances

A prerequisite to seeking persistent or repeated molecular conformations is a definition of similarity between molecular conformations. The related literature suggests a number of ways that one might measure the similarity of two different proteins, or the same protein in two different conformational states. Here, we employ the minimum root mean squared distance (RMSD) measure because it is commonly used and can be quickly computed. However, distance computation is somewhat orthogonal to the rest of our framework, and our system can easily be extended to incorporate alternative metrics. We are concerned solely with different conformations of the same molecule and can thus assume a fixed residue sequence

over all timesteps. In this regard, determining the minimum RMSD between two different conformations reduces to finding the optimal rigid transformation (i.e. translation and rotation) between them such that the RMSD taken between corresponding atoms in each conformation is as small as possible. We compute the minimum RMSD by the QCProt software of Liu *et al.* [24]. We choose this approach for its robustness and speed. In particular, when only the minimum RMSD is required (i.e. we are not concerned with the actual rigid transformation, only the RMSD which it induces), this method is the fastest known approach by orders of magnitude. Our example MD simulation involves the folding of the 1YMO molecule of 10,000 timesteps. We subsample simulation timesteps for efficiency, it is not strictly necessary. For example, computing entire $10,000 \times 10,000$ distance matrix requires less than 4 minutes.

4.2 Agglomerative Clustering

The intermediate states persist or repeat during the simulation, therefore they occur frequently during the simulation process. We use agglomerative clustering to merge similar conformations into clusters. Agglomerative clustering starts by considering every element as a cluster. Then, during each iteration, the closest clusters with respect to the chosen inter-cluster distance metric are merged until the desired number of clusters remain. Specifically, given a set of clusters, \mathcal{C} , we merge the closest two clusters C_i and C_j to form a new cluster C_{new} . $D(C_i, C_j)$ is the distance function between the two clusters. In this work, we choose the average of the distances between the constituent conformations in the two clusters as the metric for merging.

$$C_{new} = \{C_i \cup C_j\}, \arg \min_{C_i, C_j \in \mathcal{C}} D(C_i, C_j)$$

$$D(C_i, C_j) = \frac{1}{|C_i||C_j|} \sum_{x \in C_i} \sum_{y \in C_j} d(x, y)$$

For each cluster, C_i , we produce a representative conformation R_i . It is possible to simply elect a cluster member at random to serve as the representative. However, in this work we compute a representative which illustrates the average conformation within its cluster. This is accomplished by computing representative R_i as the superposition, which is the average of the iteratively aligned conformations of all of the members of cluster C_i . We compute these superpositions using the Theseus software of Theobald *et al.* [40].

Figure 2 shows an example of our resulting clusters. The superposition of the cluster members is only representative when the cluster is “tight” with low intra-cluster distance. For the members of the “loose” clusters, the intra-cluster distance is high. We will discuss our visualization design in Section 6.1.

One general problem in clustering is to choose a desired number of clusters, k . We observe the mean, m , of the intra-cluster distances during the agglomerative process. In each iteration of the agglomerative clustering process, the mean intra-cluster distance increases. We stop merging clusters when the increase of this mean distance is above a user defined threshold, τ (τ is 10\AA in our experiment).

$$m = \frac{1}{|\mathcal{C}|} \sum_{C_i \in \mathcal{C}} \frac{1}{|C_i|^2} \sum_{x, y \in C_i, x \neq y} d(x, y)$$

Generating a Hierarchy of Clusters Agglomerative clustering naturally permits the creation of a cluster hierarchy. *MDMap* uses such a hierarchy to provide multi-level exploration of conformational states. This alleviates the problem of having to choose a single scale for state identification and it is particularly useful for exploring sub-clusters of the loose clusters. The initial coarse-grain clusters are refined by decreasing τ geometrically as $\tau_{i+1} = r\tau_i$ ($r = 1.5$ in our experiments) and extracting finer-grain clusters. These new clusters exhibit a parent-child relationship with those

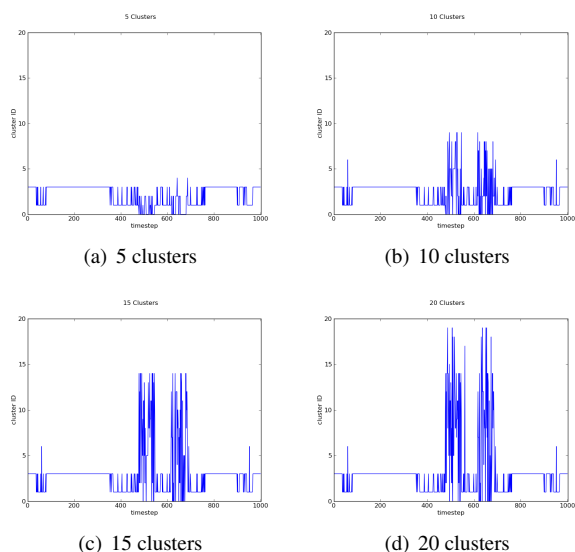


Figure 3: This figure illustrates how state (cluster) IDs change as the number of clusters increases. The x -axis of each plot represents the timesteps of the simulation while the y -axis represents the cluster ID assigned to each timestep. Thus, jumps in the graph denote transitions between states. As we increase the number of clusters from 5 (a) to 20 (d), note how the stable and semi-stable states (IDs 3 and 1 respectively) remain, while the less stable states (around timesteps 500 – 700) are split into sub-states of increasing sparsity and transience.

at the previous level which is defined by the agglomerative clustering tree. We stop subdividing a cluster when it only has a small number of members (0.5% in our experiments).

5 Transitions Amongst States

Let the set of n conformational states be given by $\mathcal{S} = \{s_i \mid 0 \leq i < n\}$. Each time step of the original simulation is identified with one of the n states, such that the state identified with time step t is given by s_t . Given the state for each time step, we can now infer the transitions which occur between these states. These transitions will become the edges in our state transition graph. Let the sequence of time steps in the simulation be given by $T = \{t_i\}_{i=0}^{N-1}$. N is the number of timesteps. We define the set of edges in the state transition graph for this simulation as follows:

$$\mathcal{E} = \{e_{ij} = (s_i, s_j) \mid t_j = t_i + 1\} \quad s_i \neq s_j \quad (1)$$

In this set, each edge denotes a transition from state s_i to state s_j . Such transitions will typically occur many times over the course of the simulation. It is useful to record not only what transitions occur, but the frequency with which they happen. We augment the edge set given in equation 1 with a frequency count which measures how many times each transition occurs. We refer to this augmented edge set as $\tilde{\mathcal{E}}$, and denote the frequency of edge e_{ij} as $\omega(e_{ij})$. This frequency information will be used later as a level-of-detail mechanism to highlight the most frequent transitions. Aesthetically, filtering by edge frequency can also be used as a mechanism to reduce visual clutter while still portraying the dominant transitions in the state transition graph.

There is a strong relationship between the number of clusters one discovers and the complexity of the transitions between them. In particular, when large clusters are split, the new clusters they produce will either correspond to the refinement of a particular conformational state into two or more well defined sub-states, or to the removal of a few of the most anomalous elements from the large

cluster. In the former of these cases, the newly introduced clusters will tend to have cardinalities of the same order, and a new semi-stable state is discovered; this tends to introduce a well-behaved set of transitions where new edges are introduced between only a small number of other states. In the latter of these cases, however, the sub-clusters produced tend to have markedly different cardinalities, and usually don't exhibit much stability. They tend to participate in a large number of transitions. Figure 3 illustrates the effect of the number of clusters on the set of transitions.

6 State Transition Graph Visualization

We visualize MD simulations as a state transition graph. The nodes represent the intermediate states, and are visualized as detailed in section 6.1. The edges show transitions between molecular states in the simulation, and are visualized as described in section 6.2. Intuitively, the directed state transition graph characterizes the flow of the molecule across the conformational landscape explored by the simulation.

6.1 Visualizing States

The nodes in the state transition graph correspond to the clusters of molecular conformations discovered in the simulation. For each node, we display the following information of the cluster; the representative conformation, the variability of the cluster, and the size of the cluster.

The representative conformation is the superposition of the cluster members. As shown in Figure 2, the superposition of conformations provides a good example for the members in tight clusters, while the conformation of members in loose clusters varies. When we display the state transition graph to the user, we denote each cluster by a rendering of its representative conformation. Each of these representative renderings is paired with a circle, which we use to convey additional information about the corresponding cluster. The color of the circles show the cluster variability from blue (low) to red (high), to ensure the users are aware of the representative power of the superposition. Furthermore, the size of the circle surrounding the representative conformation is proportional to the relative size of the cluster (adjusted according to a logarithmic scale). The shape of the protein is very stable in its folded native states, however it varies greatly when unfolded. The protein repeatedly opens and closes during the folding process as it attempts to take on the minimal energy (fully folded) conformation. This naturally results in many small and variable clusters of unfolded conformations and proportionally larger but fewer clusters arising from the stable folded timesteps. The logarithmic scale allows one to appropriately visualize this large range of cluster sizes.

6.2 Visualizing Transitions

We show the transitions of the simulation process as directed edges connecting different states. We acquire these edges from the raw simulation data; they are not expected to be clean. The state transition graph includes many edges, not all of which are highly significant. We address this issue in a few ways. First, we visually prioritize the transitions by drawing edges with widths proportional to their frequencies and allow interactive filtering. Furthermore, to disambiguate the meaning of transitions between highly occupied states, which may be a result of over-simplifying a cluster, we allow interactive hierarchical refinements of the states. Finally, we represent the bidirectional transitions on one edge with two lines in different directions to further reduce visual clutter.

Interactive Edge Filtering We provide a slider interface for users to filter the edges based on transition frequencies. Rare transitions may be removed to favor the frequent transitions shown with thick edges. When users find that certain edges or states occlude information, they can choose to de-activate them to reduce the visual

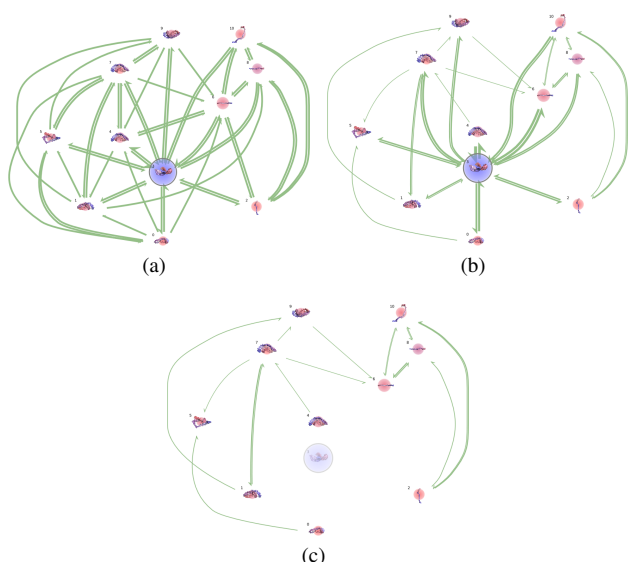


Figure 4: (a) A graph with uniform edge widths. (b) The (a) graph with frequency-weighted edges and filtering of less important transitions. When the user deactivates the center state of (b), they reduce clutter to focus on surrounding transitions in (c).

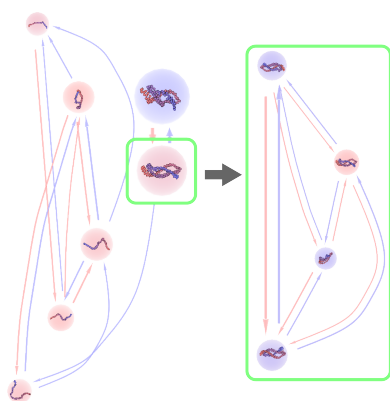


Figure 5: A coarse cluster is expanded to reveal refined states.

clutter. Edges that connect to the de-activated states are also disabled. Figure 4 shows an example.

Exploring the Hierarchy Sections 6.1 and 6.2 describe how a particular level of the the graph is visualized. Additionally, we allow the user to explore the hierarchical structure of the state transition graph interactively. Clicking on a state, s , at the current level, expands the cluster and displays all of its children. The screen position and zoom level are automatically adjusted to bring the newly expanded cluster into focus. This process may be repeated recursively to explore deeper levels of the hierarchy. Figure 5 shows this expansion.

Conversely, right-clicking collapses the current cluster into its parent. Again, the position and zoom level are automatically adjusted to bring the state transition graph at the previous resolution level into focus. These simple viewing mechanisms, layered upon the powerfully descriptive hierarchical state transition graph, allow the user to explore the putatively identified intermediate states, and to elucidate the structure represented by relatively loose clusters without introducing unnecessary and unwieldy clutter into the visualization.

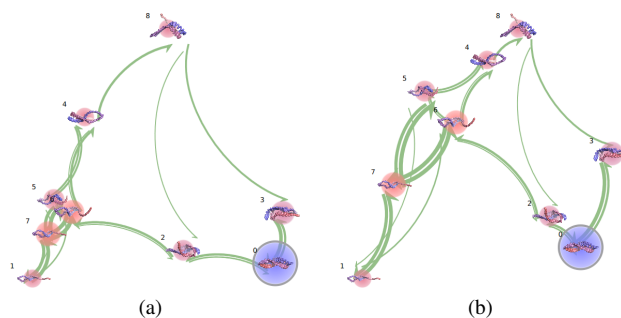


Figure 6: (a) Conformational and spectral (shown) layouts may place some nodes in close proximity thus introducing overlaps. (b) We use a grid-based adjustment to eliminate overlaps while maintaining the proper spatial relationships.

7 Laying out the MD Simulations

One of the main contributions of this paper is visualizing time varying MD simulations using state-transition diagrams. In this section, we examine the application of a variety of different graph layout approaches to the state transition graph extracted via the methods detailed in sections 4 and 5. In particular, we examine three approaches; conformational embedding and spectral graph layout, hierarchical layout, and force directed layout.

Each of them exhibits some advantages. The conformational embedding groups similar conformations nearby in the layout, while the spectral graph layout maintains spatial proximity between states sharing strong edges. The hierarchical layout orders the states according to the transition direction and tends to do the best job at avoiding edge crossings. Finally, the force directed layout tends to place tightly connected groups of states close to each other, and subsequently aids in the visualization of tightly connected components.

Conformational Embedding Given the state transition graph, \mathcal{G} , we seek to arrange the representative states so that their layout reflects the underlying conformational space of the biomolecule.

To achieve this, we compute a distance metric $d(\cdot, \cdot)$ that measures the distance between the conformations in a high dimensional vector space (\mathbb{R}^N where N is the number of atoms in the biomolecule). Then we seek a low-dimensional embedding of the conformations which attempts to preserve the high dimensional distances. To solve this dimensionality reduction problem, we employ the Laplacian Eigenmap algorithm of Belkin *et al.* [6] to determine the initial two-dimensional embedding of the biomolecular conformations. We use the minimum RMSD, as computed by the QCProt software of Liu *et al.* [24] as our distance between biomolecular conformations. We find this layout naturally clusters conformations.

Spectral Graph Layout In addition to the conformational embedding described above, we can also employ a traditional spectral graph layout [21]. Unlike the conformational embedding, the spectral graph layout considers the weighted adjacency matrix of the state transition graph. In this layout, we are concerned primarily with the proximal placement of nodes sharing strong edges. Figure 6 shows an example of such an embedding.

We find that whenever transitions between two states occurs frequently, the spectral graph layout will place these nodes close together. This approach is in some sense dual to the conformational embedding. The conformational embedding attempts to respect conformational similarity while the spectral graph layout attempts to respect similarity as defined by the frequency of transitions (i.e. the weights of edges).

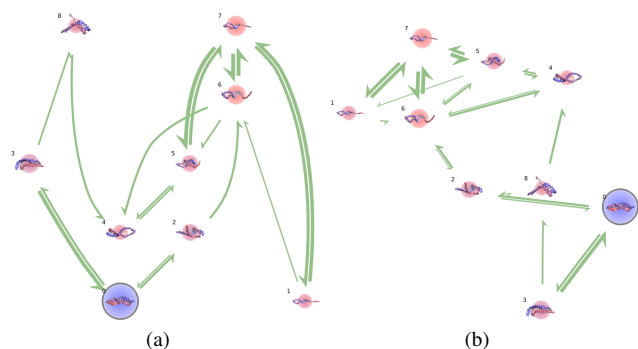


Figure 7: (a) The hierarchical layout attempts to minimize edge crossings by sorting the nodes and edges. (b) The force directed layout models the graph as a body-spring network and minimizes the spring energies to determine node positions.

Hierarchical Layout A hierarchical layout [10] of graphs lays out edges in one direction (top-down or left-right). This strategy aims to avoid edge crossings, keeps the edges short, and favors symmetry.

Figure 7(a) shows an example of the hierarchical layout applied to a state transition graph. The layout sorts the edges of the graph in the top-down direction. The hierarchical layout method places the starting states and ending states at the two ends if the graph does not contain cycles. For general graphs such as the one given in figure 7(a), it tries to break the cycle and places the end node near the top or bottom end. The directional sorting attempts to order the states from the start to the end. We find this layout helps to maintain the time sequence information.

Force Directed Layout The force directed layout algorithm models the input graph as a physical spring system. We apply the popular Fruchterman-Reingold algorithm [11] to layout the protein folding state diagrams in a force directed manner.

Figure 7(b) shows an example of the force directed layout applied to a MD state transition graph. This approach places highly connected nodes close to each other and tends to highlight the strongly-connected components of the graph. Similar to the edge-oriented spectral layout, it places the tightly connected nodes in close spatial proximity. This results in clusters of nodes exhibiting strongly-connected components. In this example, we find the strongly-connected components show different groups of conformations with frequent transitions. The edges between the connected components show the pathway for transitioning from one group to another.

8 Results and Discussion

We apply our pipeline in this paper to visualize MD simulations of three different RNAs. The simulations show the folding process of the RNA molecule with multiple trajectories. Each trajectory consists of 10,000 – 30,000 timesteps. We downsample them to 1000 – 3000 timesteps to speed up computation; this downsampling should be acceptable since, by definition, we expect intermediate conformations to be persistent. The entire analysis pipeline takes less than 1 minute for all of the trajectories we analyzed, and the graph layout and exploration are interactive.

8.1 Datasets

We performed coarse-grained MD simulations using the Three Interaction Site (TIS) Model as the energy function, and the details and parameters of the TIS model are described elsewhere [17]. The simulations were performed for HIV TAR (1ANR) [39], SRV-1 pseudoknot [1] (1E95), and the human telomerase pseudoknot (1YMO) [30]. The underdamped thermodynamic simulations

Molecule	Timesteps	RMSD(s)	Clustering(s)	Gen. Reps(s)
1YMO	1000	1.99	0.218	2.34
1ANR	3436	19.09	11.75	20.15
1E95	3000	21.93	7.68	12.4

Table 1: This table shows some example performance statistics from the analysis phase of *MDMap*. The leftmost column denotes the name of the molecule for which the MD trajectory was analyzed. The second column gives the number of timesteps analyzed for each trajectory ($1/10^{\text{th}}$ of the total timesteps of the full trajectory). The final three columns give the timing, in seconds, for the computation of the RMSD distance matrix, agglomerative clustering procedure, and generation of cluster representatives, respectively.

were performed at the respective melting temperature of the RNA molecules such that multiple transitions between the folded and unfolded states, as well as all of the intermediate states, were observed.

8.2 Experimental Setup and Performance

We performed all experiments on the Linux platform using the Intel Xeon 5140 processors. We implemented the hierarchical clustering of protein conformations using the Pycluster [9] package. Scipy [18] packages provide the routines necessary to perform the conformational embedding. Graphviz [12] provides the hierarchical and force directed graph layouts, while the NetworkX [15] package provides the spectral layout. We implemented the interactive viewer using Qt.

Table 1 provides a summary of some performance statistics for the analysis phase of *MDMap*. The time required for each step of the analysis process depends not only on the amount of data, but also on the data’s particular distribution (e.g. the distribution of distances among the timesteps in a simulation). For all of the datasets we analyzed, the identification of state transitions required a negligible amount of time ($< 0.01\text{s}$), so we have omitted the timing of this step from table 1.

8.3 Layout and Display of Example Data

In Figure 8, we show different layouts for two trajectories. The HIV TAR trajectory is visualized in Figure 8(a) and Figure 8(b). Figure 8(c) and Figure 8(d) show the SRV 1 trajectory.

Each of the graph layout strategies we employ exhibit certain strengths and weaknesses. The conformational embedding does a particularly good job of placing states with similar conformations in close proximity. This is useful to gain an intuitive notion of the distribution of discovered intermediate states over the space of molecular conformations. However, the conformational embedding procedure compares all state representatives with each other, essentially ignoring the information encoded in the edges of the state transition graph. Thus, it tends to produce layouts with more edge crossings than other methods when there are many transitions between conformationally dissimilar states. Figure 8(b) shows the result of applying the conformational embedding to a MD trajectory of the HIV TAR molecule. As expected, the states are nicely grouped, and we can see two major conformational clusters. However, when the edges are drawn, this placement of states results in a lot of edge crossings and overlaps; though these effects can be mitigated by employing our edge filtering tool, or by de-activating states which are not currently of interest.

Because the hierarchical layout algorithm attempts to minimize edge crossings, it tends to produce the least cluttered visualizations when all edges are drawn. This layout is most beneficial when one wishes to visualize the full state transition graph with its unfiltered collection of edges and states. Therefore, the hierarchical layout tends to produce some of the best *overview* diagrams of the state transition graph. Figure 8(a) shows the hierarchical layout of a state

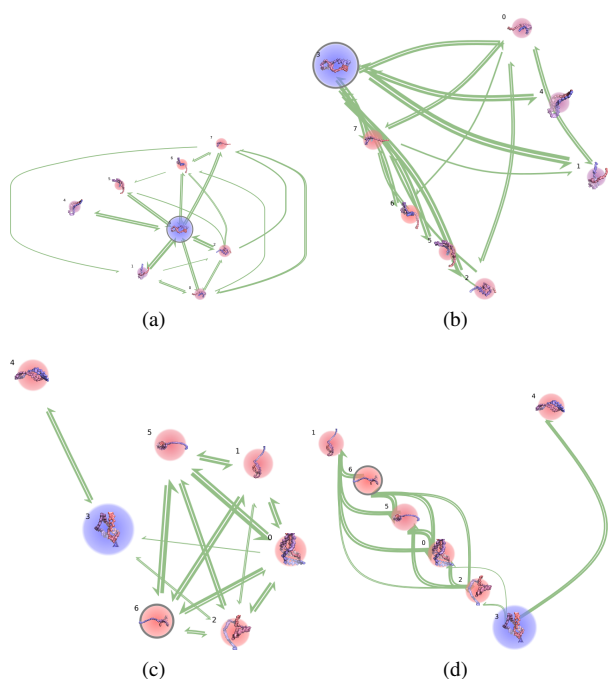


Figure 8: These images show each of the different layouts which *MDMap* can employ in visualizing the state transition graphs of MD simulations. (a) and (b) show a trajectory of the HIV TAR simulation using respectively the hierarchical layout and conformational embedding. (c) and (d) visualize a SRV 1 trajectory using force-directed layout and a spectral graph layout. As discussed in section 8.3, each of these layout strategies exhibit certain benefits.

transition graph from a trajectory of the HIV TAR molecule. The quantity and distribution of edges present in this graph tends to result in visual clutter when other layouts are employed. However, the hierarchical layout yields a visually appealing depiction of this state transition graph, even when all of the edges are shown. Despite the desirable properties of the hierarchical layout, the placement of the states themselves often fail to be meaningful. In particular, the layout has no mechanism to highlight states which are conformationally similar or between which frequent transitions occur. In certain cases, such clusterings of conformations and transitions can be important; yet these properties can be overlooked when just visualizing the state transition graph via the hierarchical layout.

Both the force directed layout and the spectral layout clearly depict the one strongly connected component and one other state in the transition graph of the SRV1 trajectory. The force directed layout arranges the states as a planar network, whereas the spectral layout arranges them in a line like fashion. This is because the first Laplacian eigenvector represents the optimal linear embedding. It will be very interesting to investigate if this ordering corresponds to the true dynamics of the SRV1 molecule. The spectral layout lined up the states in this MD simulation in a data-driven arrangement.

In addition, we show visualizations of multiple trajectories of the human telomerase pseudoknot simulation. Figure 9 shows hierarchical layouts of four different trajectories. We found trajectory three contains considerably less intermediate states than the other three trajectories. As the hierarchical layout attempts to sort the state transition graphs according to the transition order from top to bottom. We see three out of the four trajectories have the stable folded state at either end of the diagram.

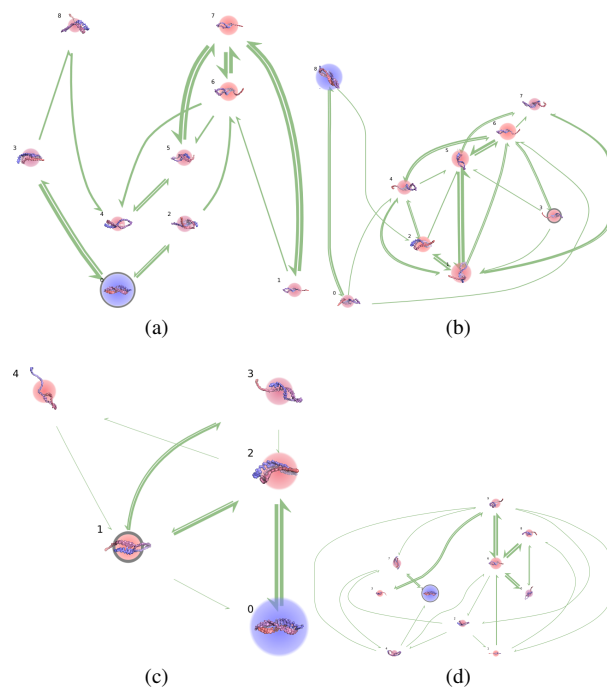


Figure 9: We computed four different trajectories of the human telomerase pseudoknot 1YMO simulations and applied hierarchical layouts to visualize them.

9 Conclusion and Future Work

We have presented *MDMap*, which encompasses a novel way to visualize molecular dynamics simulations as a state transition graph. *MDMap* detects intermediate conformational states in a MD simulation, and analyzes the transitions amongst these states to produce a state transition graph given raw MD simulation data. The state transition graphs are then laid out, and can be browsed by the user in 2D. In two-dimensional layouts, we have compared the relative advantages of several layout schemes, such as conformational embeddings, spectral graph layout, hierarchical layout and force-directed layout. We augment the display of the state transition graph with a number of user interaction mechanisms, such as edge filtering and node activation/de-activation, which are designed to reduce visual clutter and allow the user to search for particular patterns in the graph. Furthermore, we employ circles of varying colors and sizes to present the user with the statistical characteristics of the clusters from which intermediate state representations are formed.

We believe that our system conforms well to the apprehension principle of good graphics depiction advocated by previous research[43]. Our research leverages the well established view in the scientific community that biomolecules navigate a complex folding landscape in search of the folded structure. The edge strengths give a visual indication of the strength of the flow present in different pathways through the trajectory space.

The idea of visual summarization of time-varying data, either acquired or simulated, using state transition graphs is a powerful one. Stochastic simulation is a widely used tool in data driven sciences. In addition to driving MD simulations it finds widespread use in a number of other application domains, such as climate studies, nano-scale assembly, network traffic analysis and material science. We believe that visualization techniques similar to those presented here are highly likely to find ready applications in these diverse fields.

Acknowledgements

This work has been supported in part by the NSF grants: CNS 04-03313, CCF 05-41120, CMMI 08-35572, NSF 09-59979, NSF 09-14033, NIH grant GM 076688-08 and the NVIDIA CUDA Center of Excellence. Samuel S. Cho is supported by the Wake Forest University Science Research Fund. Any opinions, findings, conclusions, or recommendations expressed in this article are those of the authors and do not necessarily reflect the views of the research sponsors.

References

- [1] F. Aboul-ela, J. Karn, and G. Varani. Structure of hiv-1 tar rna in the absence of ligands reveals a novel conformation of the trinucleotide bulge. *Nucleic Acids Res*, 24(20):3974–3981, 1996.
- [2] C. B. Anfinsen. Principles that govern the folding of protein chains. *Science (New York, N.Y.)*, 181(96):223–230, 1973.
- [3] J. Assa, Y. Caspi, and D. Cohen-Or. Action synopsis: pose selection and illustration. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH)*, 24(3):667–676, 2005.
- [4] C. Bajaj, P. Djeu, V. Siddavanahalli, and A. Thane. Texmol: Interactive visual exploration of large flexible multi-component molecular complexes. In *VIS '04: Visualization*, pages 243–250, 2004.
- [5] A. Barsky, T. Munzner, J. Gardy, and R. Kincaid. Cerebral: Visualizing multiple experimental conditions on a graph with biological context. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1253–1260, 2008.
- [6] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Comp.*, 15:1373–1396, 2002.
- [7] S. S. Cho, D. L. Pincus, and D. Thirumalai. Assembly mechanisms of RNA pseudoknots are determined by the stabilities of constituent secondary structures. *PNAS*, 106(41):17349–17354, 2009.
- [8] M. L. Connolly. Analytical molecular surface calculation. *Journal of Applied Crystallography*, 16(5):548–558, 1983.
- [9] M. J. de Hoon, S. Imoto, J. Nolan, and S. Miyano. Open source clustering software. *Bioinformatics*, 20(9):1453–1454, 2004.
- [10] P. Eades and K. Sugiyama. How to draw a directed graph. *J. Inf. Process.*, 13(4):424–437, 1990.
- [11] T. M. J. Fruchterman and E. M. Reingold. Graph drawing by force-directed placement. *Softw. Pract. Exper.*, 21(11):1129–1164, 1991.
- [12] E. R. Gansner and S. C. North. An open graph visualization system and its applications to software engineering. *Software—Practice and Experience*, 30(11):1203–1233, 2000.
- [13] A. Girgensohn and J. Boreczky. Time-constrained keyframe selection technique. *Multimedia Tools Appl.*, 11(3):347–358, 2000.
- [14] Z. Guo and D. Thirumalai. The nucleation-collapse mechanism in protein folding: evidence for the non-uniqueness of the folding nucleus. *Fold Des*, 2(6):377–391, 1997.
- [15] A. Hagberg, D. Schult, and P. Swart. Networkx. High productivity software for complex networks. <https://networkx.lanl.gov/>.
- [16] W. Humphrey, A. Dalke, and K. Schulten. VMD – visual molecular dynamics. *Journal of Molecular Graphics*, 14:33–38, 1996.
- [17] C. Hyeon, R. I. Dima, and D. Thirumalai. Pathways and kinetic barriers in mechanical unfolding and refolding of RNA and proteins. *Structure*, 14(11):1633–1645, 2006.
- [18] E. Jones, T. Oliphant, and P. Peterson. Scipy: Open source scientific tools for python, 2001.
- [19] A. Joshi and P. Rheingans. Illustration-inspired techniques for visualizing time-varying data. In *IEEE Visualization*, pages 679–686, 2005.
- [20] D. Klimov and D. Thirumalai. Lattice models for proteins reveal multiple folding nuclei for nucleation-collapse mechanism. *Journal of Molecular Biology*, 282(2):471–492, 1998.
- [21] Y. Koren. On spectral graph drawing. In *Procs. of the 9th International Computing and Combinatorics Conf*, pages 496–508, 2003.
- [22] M. Krone, K. Bidmon, and T. Ertl. Interactive visualization of molecular surface dynamics. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1391–1398, 2009.
- [23] C. Levinthal. How to fold gracefully. *Mössbauer Spectroscopy in Biological Systems*, pages 22–24, 1969.
- [24] P. Liu, D. K. Agrafiotis, and D. L. Theobald. Fast determination of the optimal rotational matrix for macromolecular superpositions. *Journal of Computational Chemistry*, 31(7):1561–1563, 2010.
- [25] A. Lu and H.-W. Shen. Interactive storyboard for overall time-varying data visualization. In *IEEE VGTC Pacific Visualization Symposium 2008, PacificVis 2008*, pages 143–150, 2008.
- [26] R. A. Marciniak, B. F. Johnson, and L. Guarente. Dyskeratosis congenita, telomeres and human ageing. *Trends in Genetics*, 16(5):193–195, 2000.
- [27] C. Mayor, M. Brudno, J. Schwartz, A. Poliakov, E. Rubin, K. Frazer, L. Pachter, and I. Dubchak. VISTA : visualizing global DNA sequence alignments of arbitrary length. *Bioinformatics*, 16(11):1046–1047, 2000.
- [28] S. McCloud. *Understanding Comics – The Invisible Art*. Harper Perennial, 1994.
- [29] M. Meyer, T. Munzner, and H. Pfister. MizBee: A multiscale synteny browser. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):897–904, 2009.
- [30] P. J. Michiels, A. A. Versleijen, P. W. Verlaan, C. W. Pleij, C. W. Hilbers, and H. A. Heus. Solution structure of the pseudoknot of srV-1 RNA, involved in ribosomal frameshifting. *Journal of Molecular Biology*, 310(5):1109–1123, 2001.
- [31] C. B. Nielsen, S. D. Jackman, I. Birol, and S. J. M. Jones. ABySS-explorer: Visualizing genome sequence assemblies. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):881–888, 2009.
- [32] M. Nienhaus and J. Dollner. Depicting dynamics using principles of visual art and narrations. *IEEE Computer Graphics and Applications*, 25(3):40–51, 2005.
- [33] F. Noe and S. Fischer. Transition networks for modeling the kinetics of conformational change in macromolecules. *Current Opinion in Structural Biology*, 18(2):154–162, Apr. 2008.
- [34] G. Pingali, A. Opalach, Y. Jean, and I. Carlbom. Visualization of sports using motion trajectories: Providing insights into performance, style, and strategy. In *Procs. Visualization 2001*, pages 75–82, 2001.
- [35] S. Potzsch, G. Scheuermann, P. F. Stadler, M. T. Wolfinger, and C. Flamm. Visualization of lattice-based protein folding simulations. In *INFOVIS '06: 2006 IEEE Symposium on Information Visualization*, pages 89–94, 2006.
- [36] D. Prada-Gracia, J. Gómez-Gardeñes, P. Echenique, and F. Falo. Exploring the free energy landscape: from dynamics to networks and back. *PLoS computational biology*, 5(6):e1000415, June 2009.
- [37] K. Rutherford, J. Parkhill, J. Crook, T. Horsnell, P. Rice, M. Rajandream, and B. Barrell. Artemis: sequence visualization and annotation. *Bioinformatics*, 16(10):944–945, 2000.
- [38] M. F. Sanner and A. J. Olson. Real time surface reconstruction for moving molecular fragments. In *Pacific Symposium on Biocomputing 97*, pages 385–396, 1997.
- [39] C. A. Theimer, C. A. Blois, and J. Feigon. Structure of the human telomerase rna pseudoknot reveals conserved tertiary interactions essential for function. *Molecular Cell*, 17(5):671 – 682, 2005.
- [40] D. L. Theobald and D. S. Wuttke. Theseus: maximum likelihood superpositioning and analysis of macromolecular structures. *Bioinformatics*, 22(17):2171–2172, 2006.
- [41] D. Thirumalai and C. Hyeon. RNA and protein folding: Common themes and variations. *Biochemistry*, 44:4957–4970, 2005.
- [42] D. Thirumalai and S. Woodson. Kinetics of folding of proteins and RNA. *Acc Chem Res*, 29(9):433–439, 1996.
- [43] B. Tversky, J. B. Morrison, and M. Betrancourt. Animation: can it facilitate? *Int. J. Hum.-Comput. Stud.*, 57(4):247–262, 2002.
- [44] A. Varshney, F. P. Brooks, Jr., and W. V. Wright. Linearly scalable computation of smooth molecular surface. *IEEE Computer Graphics and Applications*, 14(5):19–25, 1994.
- [45] V. A. Voelz, G. R. Bowman, K. Beauchamp, and V. S. Pande. Molecular simulation of ab initio protein folding for a millisecond folder nt19(139). *Jnl. of the Am. Chem. Soc.*, 132(5):1526–1528, 2010.
- [46] H. Whitaker and J. Halas. *Timing for Animation*. Focal Press, 2002.
- [47] P. G. Wolynes, J. N. Onuchic, and D. Thirumalai. Navigating the folding routes. *Science*, 267(5204):1619–1620, 1995.