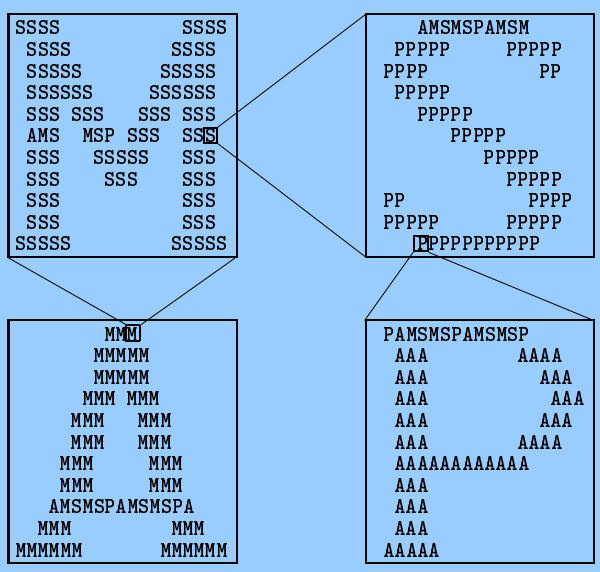
Analysis, Modeling and Simulation of Multiscale Problems

Dominant Paths Between Almost Invariant Sets of Dynamical Systems

R. Preis, M. Dellnitz, M. Hessel, Ch. Schütte, E. Meerbach

Preprint 154



Schwerpunktprogramm der Deutschen Forschungsgemeinschaft

Dominant Paths Between Almost Invariant Sets of Dynamical Systems

Robert Preis, Michael Dellnitz, Mirko Hessel
University of Paderborn
Paderborn, Germany

Christof Schütte, Eike Meerbach
Freie University of Berlin
Berlin, Germany

May 19, 2004

Abstract

In this paper we utilize variants of shortest paths algorithms in discrete, directed graphs to calculate the dominant paths between the centers of almost invariant sets in dynamical systems.

1 Introduction

The core macroscopic behavior of a dynamical system can be exhibited by the almost invariant sets of the system. An almost invariant set is a subset of the state space for which the system, once it enters the set, stays in the set, on average, for a long time. With other words, the probability to stay within the set for the following time period is very high and to move out of the set in the following time period is very low.

The use of set-oriented methods [DFJ01] features a discretization of the state space. The result is a Markov Chain which can also be viewed as a graph with directed edges. Thus, graph theoretic concepts and algorithms can be applied to calculate the almost invariant sets. The calculation of the almost invariant sets has recently been a focus in the context of the approximation of chemical conformations for molecules [DDJS98, DHFS00, FSDC01, DP03, FD03].

In this paper we go one step beyond the calculation of the almost invariant sets. The almost invariant sets provide some information about the number and position of the sets. Additionally, some characteristics of the sets and the quantity of transitions between the sets are monitored with the partition. However, there is still a lack of information about the almost invariants sets:

- Where and how large is the center region of each almost invariant set?
- By definition, a transition between two sets is of comparably low probability. However, when it occurs, which section of the boundary between them shows most transitions from one set to another?

- The dominant paths between the centers of two sets will cross along the part of the boundary with most transitions, but what are the complete dominant paths from one center of a set to the center of another set?

For all molecular dynamics models usually applied it has in fact been demonstrated that transitions between almost invariant sets are typically located close to some transition pathways [BCDG02b]. In the well known large deviation analysis of diffusive systems it even has been shown rigorously that for any two centers of almost invariant sets there is a connecting continuous path close to which almost all transitions occur [FW84]. Available algorithmic techniques for approximation of transition pathways for real-world molecular systems include, e. g., transition path sampling [BCDG02a], long-stepsize boundary value approximation [OE96], or the so-called string method [WWVE02]. All these approaches try to construct discretizations of the transition path starting from trajectories of the dynamical system directly. In addition these approaches require that initial and final state of the paths in state space are known in advance.

In contrast, in this paper we use the notion of shortest paths on directed, edge-weighted graphs to identify transition pathways. We will present an approach with which we can *simultaneously* compute almost invariant sets, their centers and the transition pathways between them.

Our experiments are based on the tools GADS, GAIO and PARTY. GAIO [DFJ01] is a tool for the analysis of dynamical systems using set-oriented numerical methods. PARTY [Pre00] is a tool for efficient graph partitioning, i. e. for partitioning the vertices of a graph into a number of parts such that the weight of the edges connecting the different parts is minimized. GADS is a toolbox of graph algorithms for the analysis of dynamical systems [Pre04]. It is based on the tools GAIO and PARTY and has interfaces to both tools. The set-oriented methods from GAIO are used to construct a discretized graph of the system. Some methods from PARTY are used to calculate partitions of a graph. Further on, GADS includes implementations of further graph based algorithms. As one example, the implementation of shortest path algorithms of this paper are added to GADS. Overall, the combination of GADS with GAIO and PARTY is a suitable setting to derive the key characteristics of dynamical systems.

The outline of this paper is the following. Section 2 introduces the background. Section 3 describes the shortest path algorithms with its problem adapted generalizations. Section 4 describes the main concept of this paper. Finally, in Section 5 we present the results of some experiments showing the usefulness of our new approach.

2 Background

In this section we present the basic definitions.

Dynamical System: Let $T : X \to X$ be a continuous map on a compact manifold $X \subset \mathbb{R}^n$. Then the transformation T defines a discrete time dynamical system of the form

$$x_{k+1} = T(x_k), \quad k = 0, 1, 2, \ldots$$

on X. In typical applications T is an explicit numerical discretization scheme for an ordinary differential equation on a subset of $\mathbb{R}^n$.

Measure: In the following we assume that μ is an **invariant measure** for the transformation T describing the statistics of long term simulations of the dynamical system. That is, the probability measure μ satisfies

$$\mu(A) = \mu(T^{-1}(A)) \quad \text{for all measurable } A \subset X.$$

Moreover we assume that μ is a unique so-called **SRB-measure** in the sense that this is the only invariant measure which is robust under small random perturbations. In other words μ is the only physically relevant invariant measure for the dynamical system T. For a precise definition of SRB-measures and their theoretical relevance in our set oriented approach see e.g. [DJ99].

Transition Probability: For two sets $A_1, A_2 \subset X$ we define the **transition probability** ρ from A_1 to A_2 as

$$\rho(A_1, A_2) := \frac{\mu(A_1 \cap T^{-1}A_2)}{\mu(A_1)}$$

whenever $\mu(A_1) \neq 0$.

Almost Invariance: The transition probability $\rho(A, A)$ from a set $A \subset X$ into itself is called the **invariant ratio** of A. Heuristically speaking a set A is **almost invariant** if $\rho(A, A)$ is close to one, that is, if almost all preimages of the points in A are in A itself. Using this observation we state the optimization problem for the identification of almost invariant sets as follows.

Problem 1 (Almost Invariant) *Let $p \in \mathbb{N}$, $p > 1$. Find p pairwise disjoint sets $A_1, \ldots, A_p \subset X$ with $\bigcup_{1 \leq i \leq p} A_i = X$ such that*

$$\frac{1}{p} \sum_{j=1}^{p} \rho(A_j, A_j) = \max!$$

Box Collection: In general the infinite dimensional optimization problem 1 cannot be solved directly. Therefore we have to discretize the problem. Following [FD03] we suppose that we have a (fine) box covering of X consisting of d boxes $B_1, \ldots, B_d$ such that

$$X = \bigcup_{i=1}^{d} B_i.$$

In practice this box covering can be created by using a subdivision scheme as described in [DH97]. The optimization problem 1 is now reduced to all subsets which are finite unions of boxes, that is, on subsets of

$$\mathcal{C}_d = \left\{ A \subset X : A = \bigcup_{i \in I} B_i, I \subset \{1, \ldots, d\} \right\}.$$

Box Transition Probability: For two sets $A_1^d, A_2^d \in \mathcal{C}_d$ the **box transition probability** ρ_d from A_1^d to A_2^d is nothing else but

$$\rho_d(A_1^d, A_2^d) := \rho(A_1^d, A_2^d) = \frac{\mu(A_1^d \cap T^{-1}A_2^d)}{\mu(A_1^d)}$$

assuming that $\mu(A_1^d) \neq 0$.

Transition Matrix: The box collection $\mathcal{C}_d$ can be used to define a weighted transition matrix for our dynamical system. This is easily accomplished by defining the transition matrix $P \in \mathbb{R}^{d\times d}$ with $P_{ij} = \rho_d(B_j, B_i)$, $1 \leq i, j \leq d$. By this procedure we have constructed an approximation of our dynamical system via a finite state Markov chain: the boxes are the vertices and two vertices i and j are connected by an edge if the image of box i under the transformation T has a nontrivial intersection with box j. The transition probability is given by P_{ij}.

By construction the matrix P is stochastic, that is $\sum_{1\leq i\leq n} P_{ij} = 1$, and by our assumption on the SRB-measure μ it is reasonable to assume that P is also irreducible. Denote by $\mu^d \in \mathbb{R}^d$ the corresponding unique **stationary distribution** of P.

Weighted, Directed Graph: The transition matrix P of a Markov chain can be viewed as a **weighted, directed graph** $G = (V, E)$. The vertex set is $V = \{v_1, \ldots, v_d\}$ with **vertex weight** $vw : V \to \mathbb{R}^+$ as $vw(v_i) = \mu_i^d$. The edge set is $E \subset V \times V$ with **edge weights** $ew : E \to \mathbb{R}^+$ as $ew((v_i, v_j)) = \mu_i^d P_{ji}$. It is $(v_i, v_j) \in E$ if and only if $\mu_i^d P_{ji} > 0$.

Internal Costs of a Partition: Let $p \in \mathbb{N}$, $p > 1$. Let $\pi : V \to \{1, \ldots p\}$ be a partition of a graph $G = (V, E)$ into the sets $V_1, \ldots V_p$ with $V = \bigcup_{1\leq i\leq p} V_i$. The **internal cost** of π is defined as

$$C_{int}(\pi) = \frac{1}{p}\sum_{i=1}^{p} \frac{\sum_{(v,w)\in E; v,w\in V_i} ew((v, w))}{\sum_{v\in V_i} vw(v)}$$

The optimization problem for the discretized problem for the identification of almost invariant sets is as follows.

Problem 2 (Internal Cost) *Let $p \in \mathbb{N}$, $p > 1$. Find a partition of V into p pairwise disjoint sets $V_1, \ldots, V_p \subset V$ with $\bigcup_{1\leq i\leq p} V_i = V$ such that*

$$C_{int}(\pi) = \max!$$

Paths in a Graph: A sequence $path = ((v_1, v_2), (v_2, v_3), \ldots (v_i, v_{i+1}))$ of connecting edges $(v_j, v_{j+1}) \in E$, $1 \leq j \leq i$, is called a **path** of size i and of length $l(path) = \sum_{j=1}^{i} ew((v_j, v_{j+1}))$ from vertex v_1 to vertex v_{i+1}. A **shortest path** from a vertex v_s to vertex v_d is a path of minimum length from v_s to v_d. The **distance** $dist(v_s, v_d)$ from v_s to v_d is the length of a shortest path from v_s to v_d.

3 Shortest Path Algorithms

3.1 The Dijkstra Algorithm for Shortest Paths

The Dijkstra algorithm is the standard algorithm for calculating shortest paths in graphs. It can be used to solve the so called *Single Source Shortest Path Problem* where the shortest paths from one source vertex $v_s \in V$ to all other vertices $v \in V$ have to be determined. The *Single Source, Single Destination Shortest Path Problem* is a special case in which only one path from v_s to a designated destination vertex v_d has to be determined. In both cases the runtime of the Dijkstra algorithm is $O(|V| \log(|V|) + |E|)$. For a deeper discussion we refer to e. g. [CLR90].

The outline of the Dijkstra algorithm is the following. It assigns an initial distance of infinity to all vertices, i.e. there are currently no known paths from the

source to each vertex. Then, the distance of the source vertex v_s is set to 0 and the distances of all neighbors of v_s are set to the weight of the edge connecting them to v_s. These vertices form the initial halo set, i.e. they are the vertices for which one path from v_s is known but it is not known whether this path is a shortest path. The main loop of the Dijkstra algorithm starts with removing a vertex from the halo set with the minimum known distance, say v_{min}. For v_{min} it holds that the known distance is also the distance of a shortest path. The loop continues with some operations on the neighbors $\{v \in V; (v_{min}, v) \in E\}$ of v_{min}. If a neighbor is also in the halo set, it is checked whether a path through v_{min} is shorter than the current known distance. Otherwise, if a neighbor still has a distance of infinity, it is added to the halo set with a distance being the sum of the distance of v_{min} and the weight of the edge connecting the neighbor to v_{min}. The algorithm terminates when the halo set becomes empty. If there are vertices not reachable from v_s, e.g. if the graph is not strongly connected, these vertices will have a distance of infinity after termination of the algorithm. If a destination vertex is supplied, the algorithm terminates as soon as this vertex is removed from the halo set. Besides the distances $dist(v_s, v)$ for all vertices $v \in V$, the algorithm also outputs for each vertex v a neighboring vertex which is the previous station along one shortest path from v_s to v. Thus, one can easily construct all shortest paths from v_s with this information.

The runtime of the Dijkstra algorithm depends on how the halo set is stored and manipulated. Fibonacci heaps are (theoretically) the best data structure for these manipulations. They guarantee a runtime of $O(|V| \log(|V|) + |E|)$.

3.2 Shortest Path between Sets of Vertices

The Dijkstra algorithm can easily be generalized to find a shortest path from any vertex of a source set $V_s \subset V$ to any vertex of a destination set $V_d \subset V$.

There are just two modifications. Firstly, in the initialization step all vertices of V_s get the distance value 0 and all neighbors of the vertices from V_s form the initial halo set. Secondly, every time a vertex v is removed from the halo set, it is checked whether $v \in V_d$.

This generalization does not increase the runtime of the Dijkstra algorithm asymptotically. Thus, its runtime still is $O(|V| \log(|V|) + |E|)$.

3.3 Several Short Paths

A shortest path between two vertices v_s and v_d is not necessarily unique. Furthermore, we are not only interested in one or all shortest paths, but we are also interested in all paths which are only slightly longer than the shortest path.

In more detail, we want to calculate all paths from a vertex v_s to a vertex v_d which have a length of at most $(1 + eps)$ times the distance between v_s and v_d. In order to do so, we need to apply the Dijkstra algorithm two times. Firstly, we calculate all distances from v_s to all other vertices. Denote these distances by $dist_1(v)$ for all vertices $v \in V$. Among all distances this also includes the distance between v_s and v_d. Secondly, we generate a new graph $G_r = (V, F)$ by F being the same set of edges as E, but in reversed direction. Then, we calculate all distances from v_d to all other vertices in G_r. Denote these distances by $dist_2(v)$ for all vertices $v \in V$. Note that $dist_2(v)$ is also the distance from v to v_d in G for any vertex $v \in V$.

We can now decide whether or not an edge (v_i, v_j) lies on a path between v_s and v_d of length at most $dist(v_s, v_d)(1 + eps)$. Such a path has to consist of three

parts: a path from v_s to v_i, the edge (v_i, v_j) itself and a path from v_j to v_d. The shortest length for the first part is $dist_1(v_i)$ and the shortest length of the last part is $dist_2(v_j)$. Thus, an edge (v_i, v_j) lies on a path between v_s and v_d of length at most $dist(v_s, v_d)(1 + eps)$ if and only if

$$dist_1(v_i) + ew((v_i, v_j)) + dist_2(v_j) \leq dist(v_s, v_d)(1 + eps) \ .$$

The result is a subset $E_{sp} \subset E$ of edges belonging to the short paths.

The asymptotic runtime of this algorithm is the same as the runtime of the Dijkstra algorithm, i. e. it runs in time $O(|V| \log(|V|) + |E|)$.

4 Dominant Paths between the Centers of Almost Invariant Sets

As already outlined in the introduction, in the following we will present a graph approach to transition pathway computation which in contrast to all other available techniques is based on a set-oriented description of the underlying dynamics and not on trajectory-oriented concepts.

Our aim is to calculate the dominant paths between almost invariant sets as the shortest paths of the associated graph with respect to a dynamically relevant distance measure. Therefore, we use the algorithms described in Section 3 and follow the following strategy:

1. Calculate the almost invariant sets.
2. Generate a weight of the edges reflecting dynamically relevant distances between vertices/boxes.
3. Calculate the center of each set.
4. Calculate the paths between all pairs of centers.

We describe each step in the following.

4.1 Partitioning into Almost Invariant Sets

We use graph based methods to calculate the almost invariant sets of the system. Although there are several graph partitioning tools available like e.g. the tool PARTY [Pre00], they are generally not designed to partition with respect to the internal cost of problem 2. Thus, the work in [FD03, DP03] exhibits algorithms which extend the graph partitioning algorithms and take the internal cost into account. The algorithms from [DP03] are integrated in the toolbox GADS [Pre04] which is also used for the experiments in this paper. Furthermore, the work in [DP03] describes how to derive a natural number p of almost invariant sets during the calculation. Finally, the work in [PPD04] describes the interlock between the standard multilevel graph partitioning paradigm and the hierarchical set-oriented approach for the analysis of dynamical systems.

4.2 Weight of the Edges

In Section 2 we have set the edge weight of an edge $(v_i, v_j) \in E$ as $ew((v_i, v_j)) = \mu_i^d P_{ji}$. This weighting is appropriate for graph partitioning algorithms due to the minimization of the internal cost of a partition. However, it is not appropriate for shortest path algorithms.

Instead, we want to install an edge weight such that the length of a path $path = ((v_1, v_2), (v_2, v_3), \ldots, (v_i, v_{i+1}))$ from a vertex v_1 to a vertex v_{i+1} is reflected by the product of the probabilities to choose the next edge along the path, i.e. $\prod_{j=1}^{i} P_{j+1,j}$. Thus, a high probability to go along this path should be reflected by a short path and vice versa.

We can do this by using shortest paths algorithms on the graph with edge weights

$$ew(v_i, v_j) := \frac{1}{\log(P_{ji})} = -\log(P_{ji}) \; .$$

Then the length of $path = ((v_1, v_2), (v_2, v_3), \ldots, (v_i, v_{i+1}))$ is

$$l(path) = \sum_{j=1}^{i} ew((v_j, v_{j+1})) = -\log(\prod_{j=1}^{i} P_{j+1,j}) \; .$$

4.3 Center Regions of Sets

We want to compute the center region of each almost invariant set. There are many ways to define such a center. We choose a center based on the length of shortest paths from each member of the set to the center.

We define the *center distance* of a vertex $v \in S$ of a set $S \subset V$ as

$$cdist(v) := \sum_{w \in S} dist(w, v) \; .$$

We call a set $S_c \subset V$ the *center* of the set S iff $cdist(v_c) \leq cdist(v)$ for all $v_c \in S_c$ and $v \in S$.

The center distance of a vertex can be calculated by one pass of the Dijkstra algorithm. Thus, the runtime is $O(|S| \log(|S|) + |E_S|)$. where $E_S = \{(u, v) \in E; u, v \in S\}$. The calculation of the center distance for all vertices of the set takes time $O(|S|(|S| \log(|S|) + |E_S|))$. Thus, the calculation of all centers takes no more time than $O(|V|(|V| \log(|V|) + |E|))$.

However, we do not calculate the center distance for all vertices of the set. Instead, we choose an arbitrary vertex and calculate its center distance. Then, we calculate the center distance of all neighbors of this vertex and progress with the neighbor with the smallest center distance and its neighbors. We terminate when no neighbor of the current vertex has a smaller center distance. This way we only compute the center distance of a small number of vertices.

4.4 Paths between Centers

We calculate the paths between all pairs of centers by using the generalized Dijkstra algorithm of Section 3. There are p centers and a total of $p * (p - 1)$ shortest paths. By applying the Dijkstra algorithm for each pair of centers results in a runtime of $O(p^2(|V| \log(|V|) + |E|))$.

5 Numerical Experiments

We implemented the described shortest path algorithms into the tool *Graph Algorithms for Dynamical Systems* (GADS). It has interfaces to the tools GAIO and PARTY and makes use of their fundamental procedures. Thus, GADS is like a layer around GAIO and PARTY with some own implementations of graph algorithms.

Our Implementation of Dijkstra The runtime of the Dijkstra algorithm depends on how the halo set is stored and manipulated. Our implementation of the Dijkstra algorithm uses binary heaps. Therefore, the running time of the algorithm is $O(|E|\log(|V|))$. This can asymptotically be improved to $O(|V|\log(|V|)+|E|)$ by the use of Fibonacci heaps. However, the operations on Fibonacci heaps involve larger constants and an implementation is more complex than an implementation of binary heaps. Furthermore, experiments usually show a similar runtime behavior for both realizations.

5.1 Examples

We now illustrate the meaning of shortest paths in the context of dynamical systems.

Pentane For the first example we use data from [DHFS00] in which a reversible stochastic matrix has been generated by simulations of an ODE-based model of the *n-pentane* molecule. Those computations made use of two dihedral angles as the essential variables. The boxes are lying in the plane defined by the two dihedral angles. This example has already been used in [DP03, FD03]. Figure 1 shows the transitions/edges between the boxes (left) and a shortest path between two boxes (right).

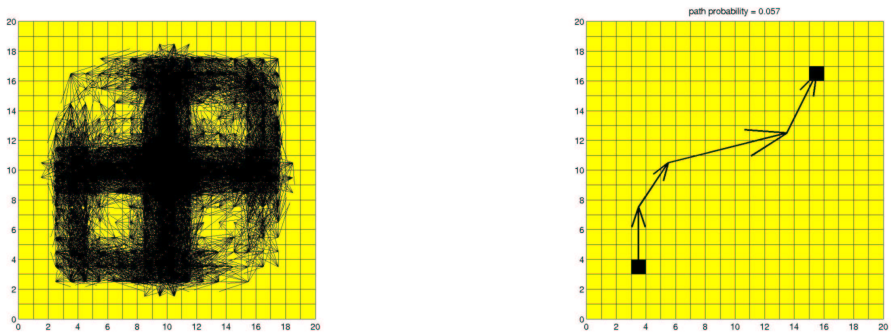


Figure 1: Left: All transitions between the boxes. Right: A shortest path between two boxes.

In Figure 2 we illustrate the generalizations of shortest paths on this example. The left picture shows a shortest path between two sets of boxes. The center picture shows all paths which are up to 10% longer than a shortest path and the right picture shows all paths which are up to 20% longer.

In Figure 3 we illustrate the calculation of the dominant paths between almost invariant sets. We first computed a partition into 7 almost invariant sets. The left picture shows the partition by different colors for different parts. The partition has a cost value of 0.963, i.e. at each time step the probability that the pentane molecule will remain in the current almost invariant set is 0.963. After calculating the weight of the edges, we then calculated the centers of these sets. The centers are marked with a black box. The center picture shows the shortest paths between all pairs of centers, i.e. a total of 42 paths. Obviously, several of those paths go along the same transitions. Finally, the right picture additionally shows all paths between the centers which are up to 10% longer than the shortest path.

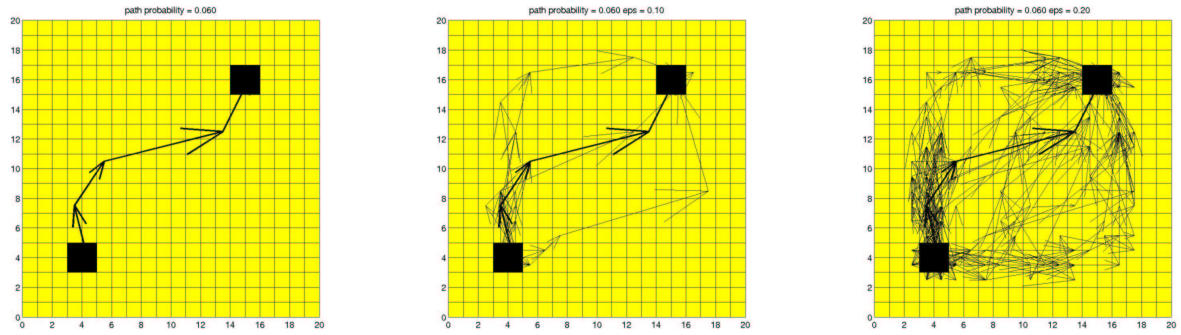


Figure 2: Left: A shortest path between two sets of boxes. Center: All paths which are up to 10% longer than a shortest path. Right: All paths which are up to 20% longer than a shortest path.

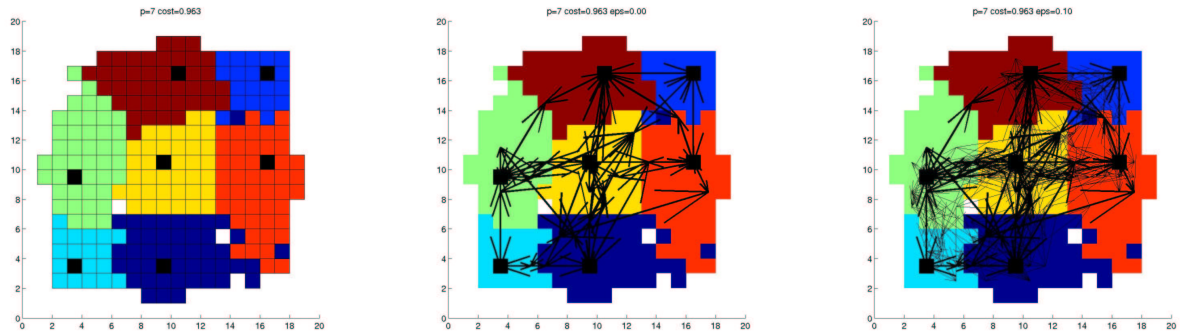


Figure 3: Almost shortest paths in the pentane-example. The colors illustrate a partition into 7 parts. Left: The centers of the sets. Center: The shortest paths between all pairs of centers. Right: All paths which are within 10% of the length of a shortest path.

Trialanine Figure 5 illustrates similar results on a simulation of a model of the *trialanine* molecule. The simulation was generated in vacuum using the Hybrid Monte Carlo method [BPCR93] with 550.000 steps with the GROMACS force field [BvdSvD95, LHvdS01] at a temperature of 750K. The integration of the subtrajectories of the HMC proposal step were realized with 1 fs timesteps of the Verlet integration scheme. This yielded an acceptance rate of about 99 %.

We used the high temperature of 750 K for the following reason: Simulation for 300 K in vacuum would lead to data whose conformation issues are perfectly described by the two central peptide angles Φ and Ψ. Instead, the time series from 750 K simulations show an additional feature: We observe transitions between substates of the torsion angle Ω (see Fig. 4) that we do not observe in 300 K simulations of this length. As one can see in Fig. 4, these transitions introduce additional features of metastability and thus result in a richer effective dynamics. We thus have three essential variables, the peptide backbone torsion angles Φ, Ψ and Ω; this fact can also be confirmed by means of successive PCCA [CWSE02]. The torsion angle Ω is separating two regions in the 3-dimensional state space by switching orientation from 0^o to 180^o, cf. again Fig. 4.

The metastable partitioning of the state space results in a partition into 5 sets as shown in Figure 5 (left). The calculation of the centers of the parts and of the dominant paths between the centers in Figure 5 (right) clearly shows that the

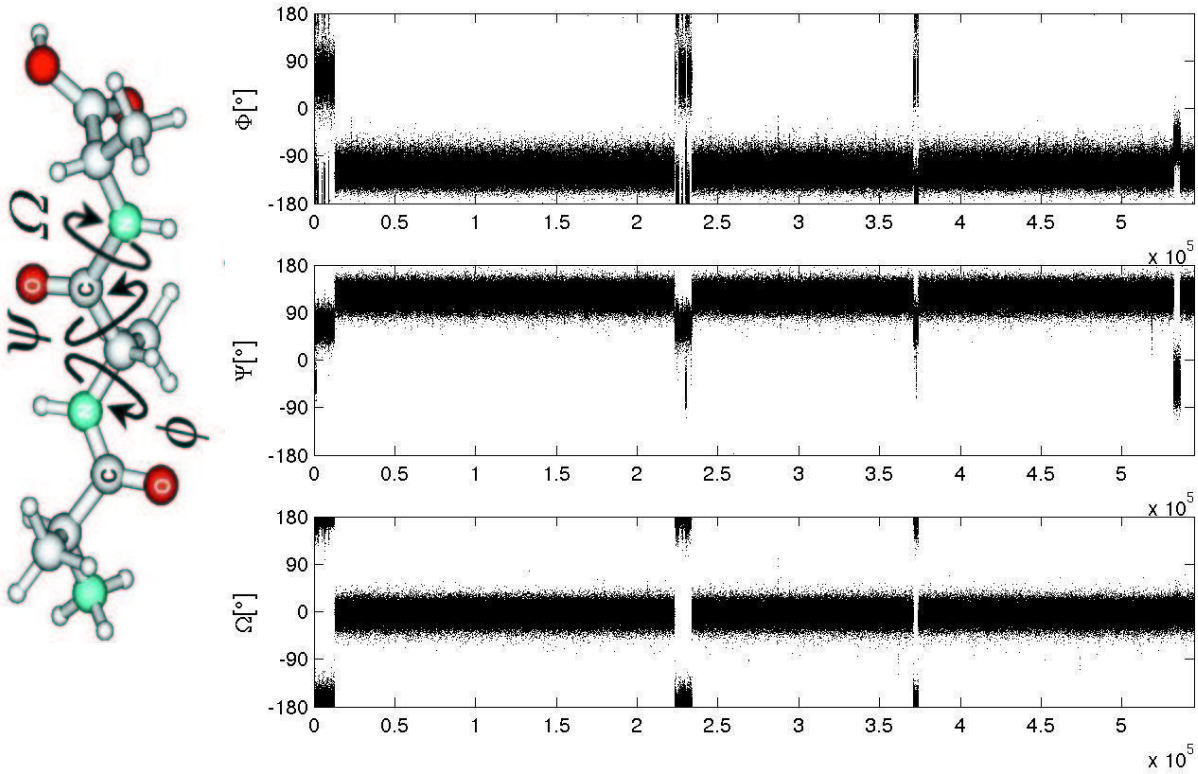


Figure 4: Simulation of the trialanine molecule. Left: The trialanine molecule shown in ball-and-stick representation and the three torsion angles Φ, Ψ and Ω. Right: A projection of the simulation data on the torsion angles Φ, Ψ and Ω clearly reveals metastable behavior.

transitions between the centers coincide with physical expectation. More precisely, the border between the sets S_2 and S_3 is playing a central role in bridging the energy barrier separating the different orientations of Ω. Note that the set S_3 allows transitions from the lower/left to the lower/right via periodicity.

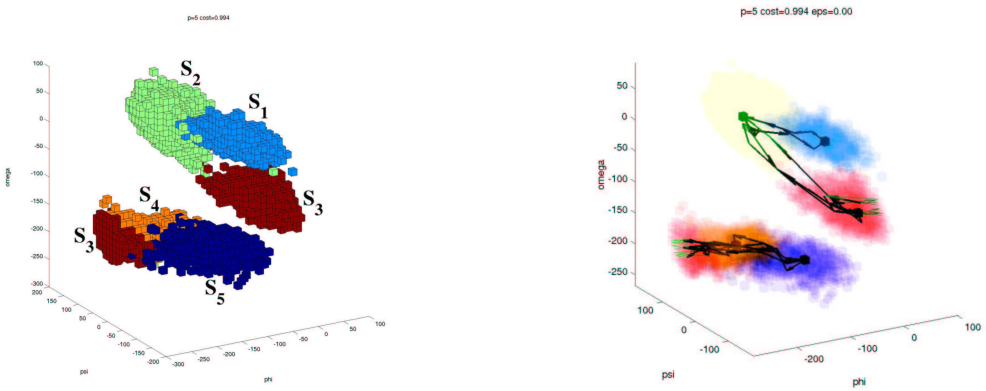


Figure 5: Analysis of trialanine. Left: The 3-dimensional state space is partitioned into 5 almost invariant sets. Right: Shortest path between all pairs of centers. Note that we are in a periodic scenario such that transitions from S_1 and S_2 to S_4 and S_5 are also integrated through S_3.

Again, we also computed paths which are only slightly longer than the shortest

paths. Figure 6 (left) shows all paths which are up to 5% (left), and 10% (right), longer. Location and aggregation of these paths again coincides nicely with physical intuition.

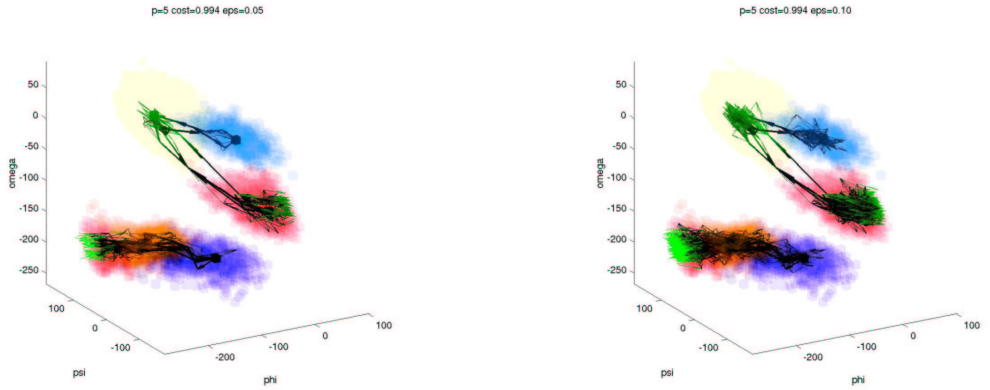


Figure 6: Analysis of trialanine (cont.). Left: All paths which are up to 5% longer than the shortest paths. Right: All paths which are up to 10% longer than the shortest paths.

Finally, the authors want to emphasize that the transition paths computed herein can only be discrete approximation of the potentially smooth *optimal* transition pathways. Obviously, they will be the coarser approximations the coarser the underlying box partition of state space is. However, there are several techniques that are optimized for computing the *optimal* smooth pathway *if* good enough coarse approximations are already known. Thus, the technique presented herein can be very helpful to compute good initial pathways for iterative pathway refinement up to high accuracy. In addition, our method will yield immediate information about the width of the pathways by computing more than only the shortest paths.

Fast–Slow Dynamics in a Double Well Potential As another example we consider the dynamics of the SDE

$$\begin{aligned} \dot{x} &= -D_x V(x,y) + \sigma \dot{W}_1 \\ \varepsilon \dot{y} &= -D_y V(x,y) + \sqrt{\varepsilon}\sigma \dot{W}_2 \end{aligned}$$

with a potential function $V(x,y)$, standard Brownian motions W_1 and W_2 and a scaling parameter $0 < \varepsilon << 1$ that introduces a time scale separation between the x and the y variable. This system has been considered in detail in [SWHH04]. The system has an *invariant measure* with density proportional to $\exp(-\beta V(x,y))$, and the almost invariant sets of this system are determined by the prominent local minima of the potential function, independent from ε. However, it seems reasonable to expect that transition paths between the minima do depend on the scaling parameter.

To explore the behavior of transition paths with varying ε, we made experiments with $\varepsilon = 0.005$ (Figure 7), $\varepsilon = 0.05$ (Figure 8) and $\varepsilon = 0.5$ (Figure 9). Using sample paths of the SDE, we derived the transition matrix based on a box collection with box size adapted to the average potential in each box.

The left pictures in the figures show a partition of the resulting state space into 2 parts each. For all three values of ε the partitions look almost identical and reflect the two almost invariant sets given by the potential. However, by computing the dominant transition paths we find that the paths are very different. Therefore, we

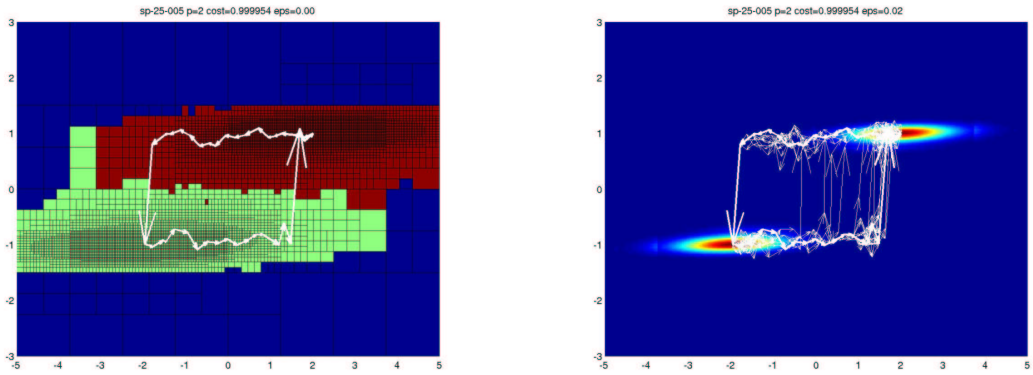


Figure 7: Example of fast–slow dynamics in a double well potential with $\varepsilon = 0.005$.

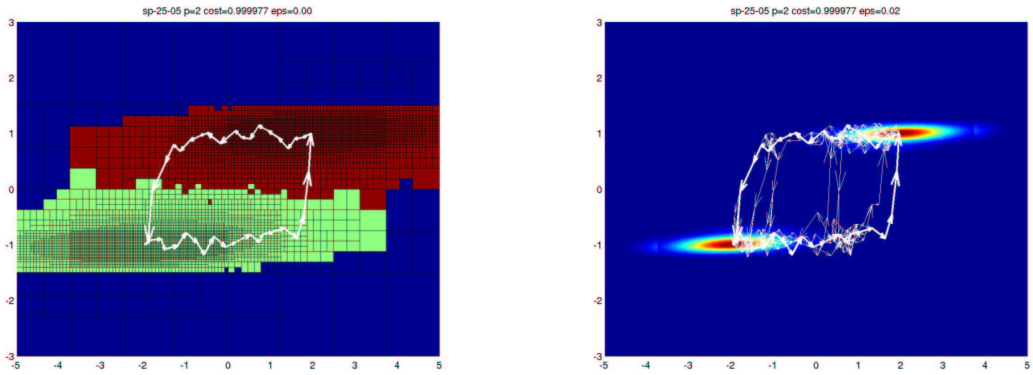


Figure 8: Example of fast–slow dynamics in a double well potential with $\varepsilon = 0.05$.

also included the shortest paths between the centers of the two sets into the pictures. The paths clearly show that the transition between the almost invariant sets depend on the amount of time scale separation. The right pictures in those figures show the invariant measure together with all paths which are at most 2% longer than the shortest paths.

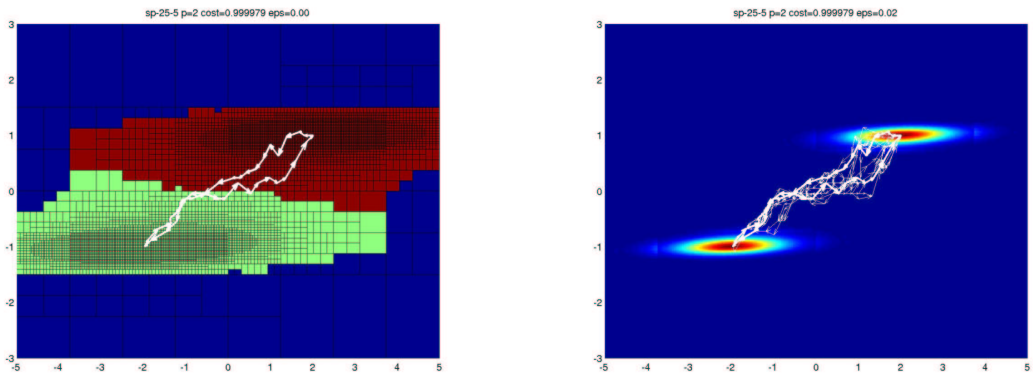


Figure 9: Example of fast–slow dynamics in a double well potential with $\varepsilon = 0.5$.

6 Conclusion

We presented a novel approach to transition pathway approximation between almost invariant sets of molecular dynamical systems based on graph techniques. This approach allows to extract transition pathways from a set oriented description of the dynamical system. Therefore, a transition graph based on a decomposition of state space which requires an exploration of the entire state space has to be computed. This disadvantage is counterbalanced by the possibility to compute almost invariant sets, their centers and the transition pathways simultaneously from the same data. In contrast, other available techniques have to know about possible initial and target centers in advance.

It should be obvious that the shortest path in the graph crucially depends on some small number, i. e., on the small transition probabilities on the critical edges in the graph. Since these numbers will have to be approximated numerically we will have to face the problem of sensitivity of *the* shortest path on numerical precision. This is one of the main reasons for the approximation of all $(1+\epsilon)$-shortest paths in the graph. If ϵ could be adjusted to numerical precision then the family of almost shortest paths will include all possible shortest paths indistinguishable by numerical precision. This is an aspect of further investigation.

It will also be a subject of further exploration whether the proposed algorithm can be integrated into other techniques in order to compute good initial guesses for smooth optimal transition pathways to be identified by further optimization.

References

[BCDG02a] Peter G. Bolhuis, D. Chandler, C. Dellago, and Phillip L. Geissler. Transition Path Sampling. *Advances in Chemical Physic*, 123:1, 2002.

[BCDG02b] Peter G. Bolhuis, D. Chandler, C. Dellago, and Phillip L. Geissler. Transition Path Sampling: Throwing Ropes over Mountain Passes in the Dark. *Annual Review in Physical Chemistry*, 53:291, 2002.

[BPCR93] A. Brass, B. J. Pendleton, Y. Chen, and B. Robson. Hybrid Monte Carlo simulations theory and initial comparison with molecular dynamics. *Biopolymers*, 33:1307–1315, 1993.

[BvdSvD95] H.J.C. Berendsen, D. van der Spoel, and R. van Drunen. Gromacs: A message-passing parallel molecular dynamics implementation. *Comp. Phys. Comm.*, 91:43–56, 1995.

[CLR90] T.H. Cormen, C.E. Leiserson, and R.L. Rivest. *Introduction to Algorithms*. MIT Press, 1990.

[CWSE02] Frank Cordes, Marcus Weber, and Johannes Schmidt-Ehrenberg. Metastable Conformations via successive Perron-Cluster Cluster Analysis of dihedrals. Technical Report 02-40, Konrad-Zuse-Zentrum für Informationstechnik Berlin, November 2002.

[DDJS98] P. Deuflhard, M. Dellnitz, O. Junge, and Ch. Schütte. Computation of essential molecular dynamics by subdivision techniques. In Deuflhard et. al., editor, *Computational Molecular Dynamics: Challenges, Methods, Ideas*, LNCSE 4, 1998.

[DFJ01] M. Dellnitz, G. Froyland, and O. Junge. The algorithms behind GAIO – set oriented numerical methods for dynamical systems. In

Ergodic Theory, Analysis, and Efficient Simulation of Dynamical Systems, pages 145–174, 2001.

[DH97] M. Dellnitz and A. Hohmann. A subdivision algorithm for the computation of unstable manifolds and global attractors. *Numerische Mathematik*, 75:293–317, 1997.

[DHFS00] P. Deuflhard, W. Huisinga, A. Fischer, and Ch. Schütte. Identification of almost invariant aggregates in reversible nearly uncoupled Markov chains. *Lin. Alg. Appl.*, 315:39–59, 2000.

[DJ99] M. Dellnitz and O. Junge. On the approximation of complicated dynamical behavior. *SIAM J. Numer. Anal.*, 36(2):491–515, 1999.

[DP03] M. Dellnitz and R. Preis. Congestion and almost invariant sets in dynamical systems. In *Symbolic and Numerical Scientific Computation (Proc. of SNSC'01)*, LNCS 2630, pages 183–209. Springer, 2003.

[FD03] G. Froyland and M. Dellnitz. Detecting and locating near-optimal almost-invariant sets and cycles. *SIAM Journal on Scientific Computing*, 24(6):1839–1863, 2003.

[FSDC01] A. Fischer, Ch. Schütte, P. Deuflhard, and F. Cordes. Hierarchical uncoupling-coupling of metastable conformations. Technical report, ZIB, 2001.

[FW84] M.I. Freidlin and A.D. Wentzell. *Random Perturbations of Dynamical Systems*. Springer, New York, 1984. Series in Comprehensive Studies in Mathematics.

[LHvdS01] E. Lindahl, B. Hess, and D. van der Spoel. Gromacs 3.0: A package for molecular simulation and trajectory analysis. *J. Mol. Mod.*, 7:306–317, 2001.

[OE96] R. Olender and Ron Elber. Calculation of classical trajectories with a very large time step: Formalism and numerical examples. *J. Chem. Phys.*, 105:9299–9315, 1996.

[PPD04] K. Padberg, R. Preis, and M. Dellnitz. Integrating multilevel partitioning with hierarchical set-oriented methods for the analysis of dynamical systems. Preprint 152, 2004.

[Pre00] R. Preis. *Analyses and Design of Efficient Graph Partitioning Methods*. Heinz Nixdorf Institut Verlagsschriftenreihe, 2000. Dissertation, Universität Paderborn, Germany.

[Pre04] R. Preis. *GADS - Graph Algorithms for Dynamical Systems; User Manual*. University of Paderborn, 2004. In preparation.

[SWHH04] Ch. Schütte, J. Walter, C. Hartmann, and W. Huisinga. An averaging principle for fast degrees of freedom exhibiting long-term correlations. *SIAM Multiscale Modeling and Simulation*, 2(3):501–526, 2004.

[WWVE02] E. Weinan, R. Weiqing, and E. Vanden-Eijnden. String method for the study of rare events. *Phys. Rev. B*, 66:052301, 2002.

Recent Preprints in this Series

140 *B. Nestler,* A 3D parallel simulator for crystal growth and solidification in complex alloy systems, *2004*

141 *D. Danilov, B. Nestler,* Phase-field simulations of solidfication in binary and ternary systems using a finite element method, *2004*

142 *K. Matthies, C. E. Wayne,* Wave Pinning in Strips, *2004*

143 *H. Garcke, B. Nestler, B. Stinner,* A Phase Field Model for the Solidification Process in Multicomponent Alloys, *2004*

144 *H. Garcke, B. Nestler, A. A. Wheeler,* Modelling of microstructure formation and interface dynamics, *2004*

145 *B. Nestler,* Diffuse Interface Model for Microstructure Evolution, *2004*

146 *Ch. Eck, H. Emmerich,* Models for Liquid Phase Epitaxy, *2004*

147 *S. Arnrich, S. Luckhaus,* A Mathematical Model for Phase Transitions in Crystals, *2004*

148 *S. Arnrich,* Lower Semicontinuity of the Surface Energy Functional - An Alternative Proof, *2004*

149 *S. Jansen, S. Luckhaus,* The Continuum Limit of a discrete energy functional - A first example with Phase Transitions, *2004*

150 *T. Blesgen, U. Weikard,* A Sharp Interface Model for Phase Transitions in Crystals with Linear Elasticity, *2004*

151 *T. Blesgen, U. Weikard,* A Segregation Principle for Ternary Systems and its Application to a Sharp Interface Model, *2004*

152 *M. Dellnitz, K. Padberg, R. Preis,* Integrating Multilevel Graph Partitioning with Hierarchical Set Oriented Methods for the Analysis of Dynamical Systems, *2004*

153 *M. Dellnitz, M. Hessel,* Topological Entropy and Almost Invariant Sets, *2004*

154 *M. Dellnitz, R. Preis, Ch. Schütte,* Dominant Paths Between Almost Invariant Sets of Dynamical Systems, *2004*

To get these preprints, please visit
`http://www.mathematik.uni-stuttgart.de/~mehrskalen/`