THE DISCRETE FRÉCHET DISTANCE AND APPLICATIONS

by

Timothy Randall Wylie

A dissertation proposal submitted in partial fulfillment
of the requirements for the degree

of

Doctor of Philosophy

in

Computer Science

MONTANA STATE UNIVERSITY
Bozeman, Montana

January 28, 2013

TABLE OF CONTENTS

1. INTRODUCTION 1

1.1 Motivation 1
1.2 Overview 2

2. BACKGROUND WORK 6

2.1 Polygonal Curves 6
2.2 The Fréchet Distance 6
2.3 The Discrete Fréchet Distance 7
2.4 The Hausdorff Distance 10
2.5 Chain Pair Simplification 10

3. RELATED WORK 12

3.1 Protein Alignment and Comparison 12
3.2 Chain Pair Simplification 14
3.3 Map and Set-Chain Matching 14

4. CURRENT WORK 16

4.1 Alignment 16
4.1.1 Algorithm 17
4.1.2 Empirical Results 18
4.2 CPS-3F Heuristic 19
4.2.1 Algorithm 20
4.2.2 Empirical Results 21
4.3 CPS-3F$^+$ 21
4.3.1 The Moving Cost 22
4.3.2 CPS-3F$^+ \in$ P 23
4.3.3 Empirical Results 27
4.4 The Fréchet-based Protein Alignment & Comparison Toolkit 32

5. FUTURE WORK 33

5.1 Chain Pair Simplification 33
5.2 Set-chain Matching 34
5.3 Map Matching 37

TABLE OF CONTENTS – CONTINUED

6. CONCLUSION 39

6.1 Dissertation Plan 39

REFERENCES CITED 41

LIST OF TABLES

Table | Page

4.1 Alignment with 107j.a (Chain A) where all eight chains have 325 vertices, and the original work is [5] and our new algorithm, ALIGN, is [1]. 19

4.2 Comparison of Algorithm SIMPLIFY [1] and FIND-CPS-3F$^+$ with 107j.a (Chain A) of length 325. $\delta_1 = \delta_2 = 4$, and δ_3 set to the minimal value. The heuristic method is in [1] and the CPS-3F$^+$ results are in [7]. 30

4.3 Comparison of Algorithm SIMPLIFY [1] and FIND-CPS-3F$^+$ with 107j.a (Chain A) of length 325. $\delta_1 = \delta_2$, and δ_3 set to the minimal value. The heuristic method is in [1] and the CPS-3F$^+$ results are in [7]. 30

4.4 Comparison of Algorithm SIMPLIFY [1] and FIND-CPS-3F$^+$with 107j.a (Chain A) of length 325, and various δ_1, δ_2, and δ_3 set to simplify both chains to a similar length. The heuristic method is in [1] and the CPS-3F$^+$ results are in [7]. 31

LIST OF FIGURES

Figure Page

2.1 The relationship between the discrete and continuous Fréchet distance where o is the continuous and the dotted line between nodes is the discrete. (a) shows a case where the chains have fewer nodes and a larger discrete Fréchet distance, while (b) is the same path with more nodes, and thus provides a better approximation of the Fréchet distance... 9

4.1 The difference between the number of nodes and the moving cost. Suppose that both (a) and (b) are valid simplifications of two chains. They have the same moving cost, yet (a) only has four nodes in each of the simplified chains, but in (b) both chains have five nodes. ... 23

4.2 The rectangle $r_{i,j}$ constructed from subchains of A, B where $d(a_i, b_j) \leq \delta_3$. Here $S_A(a_i, \delta_1)$ contains the vertices a_{i-1} to a_{i+2}, and $S_B(b_j, \delta_2)$ contains the vertices b_{j-1} to b_{j+1}. Thus, $r_{i,j}$ is defined by the min and max node indices in each subchain... 24

5.1 An instance of the set matching problem in 2D with one possible solution of $k \geq 11$... 34

5.2 The difference between minimizing $|Q|$ and $|S'|$. Minimizing $|S'|$ gives $Q = \langle s_1, s_2, s_1 \rangle$ where $|S'| = 2$ and $|Q| = 3$, but minimizing $|Q|$ will yield $|Q| = 3$ whether it uses the sequence $\langle s_1, s_2, s_1 \rangle$ or $\langle s_1, s_2, s_3 \rangle$... 35

5.3 Example clause with three variables $c_i = (\bar{x}_1 \cup x_2 \cup \bar{x}_3)$ with assignments $x_1 = 0$, $x_2 = 0$, $x_3 = 1$... 37

ABSTRACT

Modern computational geometry plays a critical role across a vast number of diverse research fields where theoretical results for provably efficient algorithms are necessary. Many of these problems are based on matching geometric objects or finding paths through given points with polygonal curves. This work focuses on the study and application of polygonal curves with respect to the discrete Fréchet distance. We overview the finished work and outline the direction of future research for the completion of the doctoral research.

For protein structure alignment and comparison, a lot of work has been done using RMSD (Root Mean Square Deviation) as the distance measure, which has drawbacks under certain circumstances. Thus, the discrete Fréchet distance was recently applied to the problem of protein (backbone) structure alignment and comparison with promising results. Here, we present the first alignment algorithm based on the discrete Fréchet distance and compare with previous work.

For this problem, visualization is also important since protein backbone chains can have as many as 500∼600 α-carbon atoms, which constitute the vertices in the comparison. Even with an excellent alignment, the similarity of two polygonal chains can be difficult to visualize unless the chains are nearly identical. Thus, the chain pair simplification problem (CPS-3F) was proposed in 2008 to simultaneously simplify both chains with respect to each other under the discrete Fréchet distance. The complexity of CPS-3F is unknown, so we originally created a greedy backtracking heuristic (SIMPLIFY). Then we define a variation of CPS-3F, called the constrained CPS-3F problem (CPS-3F$^+$), and prove that it is polynomially solvable by presenting a dynamic programming solution, which we then prove is a factor-2 approximation for CPS-3F. We then compare CPS-3F$^+$ empirically with SIMPLIFY. Chain pair simplification based on the Hausdorff distance (CPS-2H) is known to be **NP**-complete, and we define the constrained version (CPS-2H$^+$) as another problem of interest.

Another area of our investigation of the discrete Fréchet distance is the map matching and set-chain matching problems. The map matching problem is to find a path in a graph with a minimal Fréchet distance to a given polygonal line. The set matching problem is similar, but rather than a graph, the goal is to find another polygonal curve with nodes from a given point set. We study the discrete map matching and discrete set-chain matching problems, and look at the complexity when given a maximal number of vertices or points allowed, and when the paths are unique.

Finally, most of the algorithms that we developed have also been implemented as a software library, named FPACT (The Fréchet-based Protein Alignment & Comparison Toolkit), providing the ability for others to align, compare, and simplify polygonal curves with the discrete Fréchet distance.

CHAPTER 1
INTRODUCTION

1.1 Motivation

Modern computational geometry plays a critical role across a vast number of diverse research fields. Theoretical results with efficient algorithms, whether heuristic, approximate, fixed-parameter tractable, etc., are necessary and applied in such disparate areas as protein structure alignment, Wi-Fi hotspot placement, pattern matching, computer vision, map routing and other GIS services, speech and handwriting processing, and a countless number of other applications.

Many of these problems are based on matching geometric objects and finding paths through designated points. The problems largely deal with, or can be abstracted to, polygonal curves, which we will focus on. The study of polygonal curves is not only imperative to many geometric applications, but some of these path problems are also fundamental, such as ordering, and are used to define complexity classes and completeness.

There are many measures that are commonly used with parametric curves. The Fréchet distance is often used when comparing two curves and can be described as a dissimilarity measure because the distance is a measure of how different the two curves are from each other [2]. When we examine polygonal curves, this calculation is much easier and can be done efficiently in $O(mn \log mn)$ time [3]. However, for many problems we are often only concerned with the nodes along the path and not the edges. For instance, our data may be sampled from a time series and thus we have ordering, but we may not be able to accurately infer information between the sampled data. This led to the discrete Fréchet distance, which was first defined in the early 1990s [4].

Despite being well-known, the discrete Fréchet distance has not been studied or applied in many areas where other measures – such as the standard Fréchet distance, root mean square deviation, and the Hausdorff distance – are considered the standard benchmarks. Motivated by some biological, visualization, and map routing applications, we utilize the discrete Fréchet distance and show many positive results. Further, we analyze many unique aspects of the discrete Fréchet distance.

1.2 Overview

In this section, we overview the problems and research that are covered in the proposal while highlighting many of our results.

The optimal alignment problem, as defined in [5], between two 3D chains under the discrete Fréchet distance takes $O(m^7 n^7 \log(m+n))$ time to solve [5]. Due to the high time complexity they proposed a heuristic method not dependent on the discrete Fréchet distance. In our first publication [1], we revisited the optimal alignment problem by proposing a possible PTAS heuristic algorithm in which all translations and rotations were based on the current discrete Fréchet distance of the two chains. We also showed that this was at worst a 2-approximation algorithm for the optimal alignment problem. The new algorithm provided better alignment results than the previous method for all empirical evaluations.

We also focused on another problem related to pairs of polygonal chains: simplification. Assuming two chains are optimally aligned, in what meaningful way can we simplify them with respect to each other? Can the chains be simplified such that their relationship maintains certain qualities of their similarity after simplification? Further, what are these qualities and how are they useful? To shed light on these questions, we utilized the Chain Pair Simplification (CPS) problem (Section 2.5) [6]. Specifically, CPS-3F – where the two chains and their comparison to each other are all simplified via the discrete Fréchet distance (we address other variations later).

The initial motivation for this problem was visualization of aligned protein backbones. Given that protein backbones can have as many as 500∼600 vertices (α-carbon atoms) in each chain, even with an optimal alignment, visualizing the similarity of two chains is difficult unless those chains are nearly identical. Our initial approach for CPS-3F was an efficient $O(n)$ greedy back-tracking algorithm [1]. This heuristic method was useful for efficiently handling pairs of extremely long chains. Using this heuristic with our alignment algorithm yielded positive results when empirically evaluated on protein backbone chains. However, the greedy nature of the method makes evaluating and controlling the simplification between the two chains difficult, even though the intra-chain simplifications are well-behaved.

Our next approach was a dynamic programming algorithm which we showed was at worst a 2-approximation to an optimal CPS-3F simplification [7]. We achieved this by defining the *moving cost* of the discrete Fréchet distance between two chains, which is a property controlled by weakly increasing integer sequences [8], but had never been defined or used as a measure on the discrete Fréchet distance. This measure is also unique to the discrete Fréchet distance, and thus is not a special case of the continuous Fréchet distance because it is infinite between continuous curves. By minimizing the moving cost, we could exploit the inter and intra chain relationships and we greatly improved on all of our previous empirical results. The dynamic programming algorithm is not as efficient though, with a complexity between $O(mn)$ and $O(m^2n^2)$ depending on the input simplification parameters. Further, chain pair simplification under the Hausdorff distance (CPS-2H) is **NP**-complete, and we are looking at the complexity of minimizing the moving cost of using the Hausdorff distance which we also believe will be **NP**-complete.

To facilitate research using the discrete Fréchet distance we created a set of open-source libraries to run any of our algorithms based on the discrete Fréchet distance. The FPACT (The Fréchet-based Protein Alignment & Comparison Toolkit) libraries were designed for easy access to the algorithms by being modular and format independent. The libraries are written and available in both C# and Python [9].

Another area of our investigation of the discrete Fréchet distance deals with the map matching and set-chain matching problems (Chapter 5). These problems are defined and analyzed based on the continuous Fréchet distance [10, 11]. We not only examine them based on the discrete Fréchet distance, but also generalize the problem definitions with new variations, and examine the complexities for the new variations of the continuous versions as well.

One application of the map matching problem is recreating the most likely path of a vehicle on a road network given noisy GPS tracking data. Our work extends this analogy to assume the GPS data may also be intermittent. Suppose that in certain areas we have no connection with the GPS until some other point in time– we still have a polygonal curve, but we can not depend on all edges of the line to be accurate location data. Or similarly, situations exist where a person may check in periodically, but keeping a constant connection is too costly due to coverage or power constraints.

The set-chain problem could be viewed as finding cellular towers to ensure coverage, given the route to travel and the maximum range of a tower. In the set-chain matching problem, there is a polygonal curve P, a set of points S, and an $\varepsilon > 0$ given. The problem is to find another polygonal curve Q such that the nodes of Q are points in S and the discrete Fréchet distance is $d_F(P, Q) \leq \varepsilon$. The map matching problem is similar, except instead of a set of points, we have a planar graph G and we must find a path through the graph that preserves the distance constraint. These are formally defined with our generalizations in Chapter 5.

We extend the problems in a few ways. The original works allow points/vertices to be used multiple times in Q, and they are only concerned whether such a path exists. We state the problems as minimization problems where we look at minimizing either the nodes of Q or the set of points/vertices from S/G used as nodes of Q. These still assume non-unique nodes in Q, so we also examine the problems given unique nodes, i.e., any point or vertex can only be used as a node in Q once. Note that when looking at unique nodes, minimizing $|Q|$ is equivalent to minimizing the set of points/vertices since they can only be used once.

The rest of the proposal is as follows. In Chapter 2 we first review some necessary background information. Then in Chapter 3 we discus the current related work. We then survey all of the research that has been done in Chapter 4, which is followed by the work that still needs to be finished in Chapter 5. Finally, Chapter 6 concludes and contains the dissertation plan and timeline.

CHAPTER 2
BACKGROUND WORK

Here, we briefly overview the background material necessary for the work presented. References are also given for a more complete and thorough treatment of the concepts being used.

2.1 Polygonal Curves

We first overview the parametric definition of polygonal curves since our research is based on an investigation of this fundamental geometric structure. We look at the natural parameterization of polygonal curves [3, 12, 8, 11].

Let $\alpha : [0, p] \to \mathbb{R}^d$ be a polygonal curve in $\mathbb{R}^d$, which consists of p line segments $\bar{\alpha}_i := \alpha|_{[i,i+1]}$ for $i \in \{0, 1, \ldots, p-1\}$. Each line segment $\bar{\alpha}_i$ is affine and parameterized by its natural parameterization, i.e., $\alpha(i+\lambda) = (1-\lambda)\alpha(i)+\lambda\alpha(i+1)$ for all $\lambda \in [0, 1]$.

2.2 The Fréchet Distance

The Fréchet distance was first defined by Maurice Fréchet in 1906 as a measure of similarity between two parametric curves [2]. Subsequently, it has become a standard measure between parametric curves used in many areas. The Fréchet distance is typically explained as the relationship between a person and a dog connected by a leash walking along the two curves and trying to keep the leash as short as possible. The maximum length the leash reaches is the value of the Fréchet distance. This common analogy has led to the term "dog-walking distance" sometimes being used.

We first define parameterized continuous curves and then formally give the standard definition for the Fréchet distance [3, 12].

Definition 1. *A continuous parameterized curve* $A \in \mathbb{R}^d$ *can be represented by a continuous mapping* $f : [a, b] \to \mathbb{R}^d$ *such that* $a, b \in \mathbb{R}$ *and* $a < b$*.*

Definition 2. *A monotone reparameterization* α *is a continuous non-decreasing function* $\alpha : [0, 1] \to [0, 1]$ *such that* $\alpha(0) = 0$ *and* $\alpha(1) = 1$*.*

Definition 3. *Given two curves,* A, B *in a metric space, the* ***Fréchet distance****,* $d_{\mathcal{F}}(A, B)$ *is defined as*

$$d_{\mathcal{F}}(A, B) = \inf_{\alpha,\beta} \max_{t \in [0,1]} \{d(A(\alpha(t)), B(\beta(t)))\}$$

where α, β *range over all monotone reparameterizations and* $d(\cdot, \cdot)$ *represents the Euclidean distance, and* inf *is the infimum.*

In the early 1990s, the Fréchet distance was applied to polygonal curves by Alt and Godau [13, 3]. With the restriction of polygonal lines, they proved the Fréchet distance can be found between two curves A, B efficiently with a time complexity of $O(mn \log mn)$ where $m = |A|, n = |B|$.

2.3 The Discrete Fréchet Distance

In 1994 Eiter and Mannila defined the discrete Fréchet distance as an approximation of the Fréchet distance to be used between two polygonal chains using only the nodes along the chains for the measurements [4]. They also referred to this discrete form as the coupling distance which is used synonymously. Furthermore, they proved the discrete version can be computed in $O(mn)$ time where m, n are the number of vertices in each polygonal chain. A rigorous look at the definition of the discrete Fréchet distance was also done by Mosig et al. in 2005 [8].

Given two paths, we define their discrete Fréchet distance as follows. (We use the graph-theoretic term "paths" instead of the geometric term "polygonal chains" here because our definition makes no assumption that the underlying space of points is geometric.) We use $d(a, b)$ to represent the Euclidean distance between two 3D points a and b, but it can be replaced with some other distance measure, depending on applications.

Definition 4. *Given a path $P = \langle p_1, \ldots, p_n \rangle$ of n vertices, a t-**walk** along P is a partitioning of P along the path into t disjoint non-empty subpaths $\{P_i\}_{i=1..t}$ such that $P_i = \langle p_{n_{i-1}+1}, \ldots, p_{n_i} \rangle$ and $0 = n_0 < n_1 < \cdots < n_t = n$.*

Definition 5. *Given two paths $A = \langle a_1, \ldots, a_m \rangle$ and $B = \langle b_1, \ldots, b_n \rangle$, a **paired walk** along A and B is a t-walk $\{A_i\}_{i=1..t}$ along A and a t-walk $\{B_i\}_{i=1..t}$ along B for some t, such that, for $1 \leq i \leq t$, either $|A_i| = 1$ or $|B_i| = 1$ (that is, either A_i or B_i contains exactly one vertex).*

Definition 6. *The **cost** of a paired walk $W = \{(A_i, B_i)\}$ along two paths A and B is*

$$d_F^W(A, B) = \max_i \max_{(a,b) \in A_i \times B_i} d(a, b).$$

Definition 7. *The **discrete Fréchet distance** between two paths A and B is*

$$d_F(A, B) = \min_W d_F^W(A, B).$$

*A paired walk that achieves the discrete Fréchet distance between two paths A and B is called a **Fréchet alignment** of A and B.*

If we revisit the dog-walking analogy, we consider the scenario in which a person walks along A and a dog along B. Intuitively, the definition of the paired walk is based on three cases:

1. $|B_i| > |A_i| = 1$: the person stays and the dog hops forward;

2. $|A_i| > |B_i| = 1$: the person hops forward and the dog stays;

3. $|A_i| = |B_i| = 1$: both the person and the dog hop forward.

The following figure shows the relationship between the discrete and continuous Fréchet distances. In Figure 2.1(a), we have the two chains $\langle a_1, a_2, a_3\rangle$ and $\langle b_1, b_2\rangle$, the continuous Fréchet distance between the two is the distance from a_2 to segment $\overline{b_1 b_2}$, i.e., $d(a_2, o)$. The discrete Fréchet distance is $d(a_2, b_2)$. The discrete Fréchet distance could be quite larger than the continuous distance. On the other hand, with enough sample points on the two chains, the resulting discrete Fréchet distance, i.e., $d(a_2, b)$ in Figure 2.1(b), closely approximates $d(a_2, o)$.

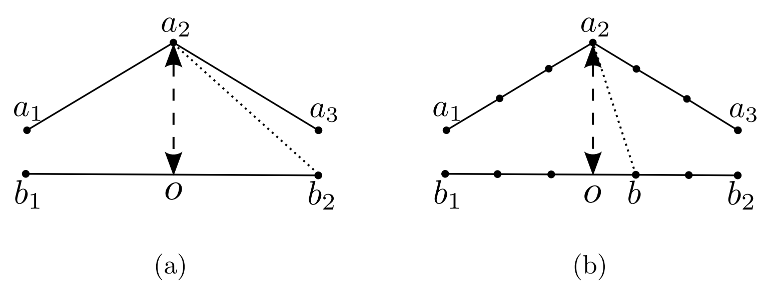

Figure 2.1: The relationship between the discrete and continuous Fréchet distance where o is the continuous and the dotted line between nodes is the discrete. (a) shows a case where the chains have fewer nodes and a larger discrete Fréchet distance, while (b) is the same path with more nodes, and thus provides a better approximation of the Fréchet distance.

With enough nodes the discrete Fréchet distance can closely approximate the continuous version, and with a standard dynamic programming approach, it is not hard to obtain the following theorem.

Theorem 1. *[4] The discrete Fréchet distance between two paths with m and n vertices respectively can be computed in $O(mn)$ time.*

In two dimensions, it was recently shown that subquadratic time is possible, but the difference is marginal [14]. The new algorithm still requires $O(\frac{mn \log \log n}{\log n})$ time.

2.4 The Hausdorff Distance

The Hausdorff distance was first defined by Felix Hausdorff in 1914 [15]. Since its introduction, the Hausdorff distance has become one of the most widely used similarity measures across many disciplines. Definition 8 is taken from [16].

Definition 8. *Let X and Y be two non-empty subsets of a metric space (M, d) where M is the space and d the distance measure. We define their Hausdorff distance $d_H(X, Y)$ by*

$$d_H(X, Y) = \max\{\sup_{x \in X} \inf_{y \in Y} d(x, y), \sup_{y \in Y} \inf_{x \in X} d(x, y)\},$$

where sup *represents the supremum and* inf *the infimum.*

2.5 Chain Pair Simplification

In 2008, the chain pair simplification problem in three dimensions under the discrete Fréchet distance was defined in order to allow better visualization of two polygonal chains [6]. The problem not only allows better visualization of the two chains in a simplified form, but it also keeps and exploits the characteristic similarities that exist between the chains. Although the problem does not necessarily need to be limited to only 3D space, we state the original decision problem as it was defined relating to protein backbone chains.

Definition 9 (The Chain Pair Simplification (CPS) Problem)**.**

Instance: *Given a pair of 3D chains A and B, with lengths $O(m), O(n)$ respectively, an integer $K > 0$, and three real numbers $\delta_1, \delta_2, \delta_3 > 0$.*

Problem: *Does there exist a pair of chains A', B', each of at most K vertices, such that the vertices of A', B' are from A, B respectively, and $d_1(A, A') \leq \delta_1$, $d_2(B, B') \leq \delta_2$, $d_F(A', B') \leq \delta_3$?*

When $d_1 = d_2 = d_F$, the problem is called CPS-3F since all three distance measures are the discrete Fréchet distance. When $d_1 = d_2 = d_H$ (the Hausdorff distance), the problem is called CPS-2H since two of the distances are Hausdorff. CPS-2H was proven to be NP-complete [6], but the complexity of CPS-3F is unknown.

CHAPTER 3
RELATED WORK

We now survey the research relavant to our work. Given the distinct application fields, we cover the related work in the categories: protein alignment and comparison, chain pair simplification, and map and set-chain matching.

3.1 Protein Alignment and Comparison

The comparison and simplification of polygonal chains have been well studied in several fields including computer vision, bioinformatics, computational geometry, and parametric curve approximations [17, 18, 19]. Within structural biology, polygonal chain similarity is one of the central problems of protein research. In general, it is believed that a protein's structure implies its function, and thus to compare the functionality of proteins their structures must be compared [20]. This is known to be true for certain situations, especially with homologous traits between proteins, and in general the empirical evidence between proteins is in agreement [20, 21]. The structure is defined by the α-carbon atoms of the residues (amino acids) along the backbone of each chain. These atoms represent the vertices that constitute our 3D polygonal chains.

Since the structure of the protein is intrinsically related to its function, there have been many software systems designed for protein structure alignment and comparison in the last couple of decades. A few of the more well-known systems are SSAP [22], DALI [23, 24], CATH [25], CE [26], SCOP [27], MAMMOTH [28], ProteinDBS [29] and 3D-BLAST [30]. None of these systems use the discrete Fréchet distance, and the majority of the work previously done on protein global structure alignment and protein local structure alignment uses the RMSD distance measure. Given two m-vectors $V_1 = \langle u_1, u_2, ..., u_m \rangle$ and $V_2 = \langle v_1, v_2, ..., v_m \rangle$, RMSD is defined as:

$$RMSD(V_1, V_2) = \sqrt{\frac{\sum_i (u_i - v_i)^2}{m}}.$$

This gives an average pair-wise distance along the two vectors which provides some insight into the similarity of the two chains, but the reliance on m shows one of the major drawbacks of using RMSD. The comparison hinges on the necessity that the two vectors be the same length and that the vertices at a given index in each chain be pairwise similar. If we modified the chains at all we could receive very different RMSD values. Suppose we are given two chains C_1, C_2 with m vertices, and we then add some vertices on C_1 and C_2 by alternatively duplicating/repeating some different vertices in C_1 and C_2 to obtain C_1', C_2', then $RMSD(C_1', C_2')$ could be dramatically different from $RMSD(C_1, C_2)$, even though geometrically C_1' and C_2' are just as close as C_1 and C_2 are. This suggests that a measure independent of the number of vertices or a pair-wise alignment would be a better indication on the similarity of the two chains.

To achieve a more accurate measure of similarity between two protein structures, Jiang et al. proposed using the discrete Fréchet distance for the protein backbone comparison [5]. The two main problems they addressed were the alignment of the two chains, and then the comparison itself. They showed that the optimal alignment problem, as defined in [5], between two 3D chains under the discrete Fréchet distance takes $O(n^7 m^7 \log(n + m))$ time to solve [5]. Due to the high time complexity they proposed a heuristic method not dependent on the discrete Fréchet distance. We compare the results with our algorithm and show an improvement in alignment.

3.2 Chain Pair Simplification

While a lot of work has been done on single chain simplification [6], there is little comparable work with dual simplification, even outside of research with the Fréchet distance. Most research related to simplification focuses on the continuous Fréchet distance. This work includes partial alignment [31] or the Fréchet distance between curves with shortcuts [32] that ignore a set number of sections of the curve that are not well aligned.

3.3 Map and Set-Chain Matching

With respect to map matching, the problem of finding a path in a graph given a polygonal line was first posed by Alt et. al. [11] as follows: Let $G = (V, E)$ be an undirected connected planar graph with a given straight-line embedding in $\mathbb{R}^2$ and a polygonal line P, find a path π in G which minimizes the Fréchet distance between P and π. They give an efficient algorithm which runs in $O(pq \log q)$ time and $O(pq)$ space where p is the number of line segments of P and q is the complexity of G, but it also allowed vertices and edges to be visited multiple times.

The recent work by Maheshwari et al. improved the running time for the case of a complete graph [10]. The original algorithm would decide it in $O(pk^2 \log k)$ where k is the number of vertices in the graph, and their new algorithm solves it in $O(pk^2)$. The work by Maheshwari et al. changed the problem to trying to find a polygonal curve through a subset of points to minimize the Fréchet distance to some curve P. They define the set-chain matching problem as: Given a point set S and a polygonal curve P in $\mathbb{R}^d$ ($d \geq 2$), find a polygonal curve Q with its vertices chosen from S, which has a minimum Fréchet distance to P.

Another well-known problem that we reduce from is planar 3-SAT [33]. Planar 3-SAT is a version of the standard 3-SAT problem where the variables and clauses can be represented by nodes in a graph, and that graph has a planar embedding. Planar 3-SAT is also **NP**-complete and more convenient for proving properties of geometric problems [33, 34, 35].

CHAPTER 4
CURRENT WORK

In this chapter we outline most of the research that we have already done. This begins with an investigation into the alignment and simplification of protein backbone chains. We also provide some of our empirical results. These highlight the algorithm contributions with some heuristics and approximation algorithms based on greedy and dynamic programming methods.

4.1 Alignment

The optimal (global structure-structure) alignment problem is formally defined as follows.

Definition 10. *Given two 3D polygonal chains A and B, a transformation class T, and a distance measure $d(-)$, find a transformation $\tau \in T$ such that $dist(A, \tau(B))$ is minimized.*

Of course, in our case T contains both rotation and translation, and $d = d_F$.

Let $A = \langle a_1, a_2, ..., a_m\rangle$ and $B = \langle b_1, b_2, ..., b_n\rangle$. It was shown that the optimal alignment problem under the discrete Fréchet distance can be solved in $O(n^7 m^7 \log(n+m))$ time [5]. This is impractical in use, so Jiang et al. presented a heuristic method which focuses on first aligning the center a of A and the center b of B. (Given a 3D chain C of n vertices, the coordinates of each vertex c_i of C is really a vector $\vec{c_i}$, the center c corresponds to $\vec{c} = \frac{\sum_i \vec{c_i}}{n}$.) Then a rotation is performed such that $\triangle a_1 a a_m$ and $\triangle b_1 b b_n$ are on the same plane. Finally, some local improvements are performed until the discrete Fréchet distance cannot be further improved. While this algorithm is still slower compared to some of the known software (like ProteinDBS [29]), it can improve the accuracy in many situations [5].

4.1.1 Algorithm

We use a slightly different idea here. We can prove that if we first move B such that b_1 is located exactly at a_1 and subsequently obtain an optimal solution, then this solution is a factor-2 approximation for the optimal alignment problem (when a_1 does not necessarily collide with b_1). Of course, a factor-2 approximation may not be accurate enough for many biological applications. Therefore, while colliding b_1 at a_1 is our starting point, our algorithm goes beyond that. Our complete (heuristic) algorithm is as follows.

Algorithm ALIGN(A, B):
Input: Two polygonal chains $A = \langle a_1, ..., a_m \rangle$ and $B = \langle b_1, ..., b_n \rangle$.

1. Translate B so that $d(b_1, a_1) = 0$.
2. Let β be the midpoint of $\langle a_m, b_n \rangle$. Rotate B around the axis line (a_1, β) so that $d(a_m, b_n)$ is minimized. Let $a_i \in A$ and $b_j \in B$ be the two vertices such that $d(a_i, b_j) = d_F(A, B)$.
3. Initialize $O^*(A, B) \longleftarrow d_F(A, B)$.
4. Loop until no improvement of $O^*(A, B)$ is made.
   (a) Rotate until no improvement of $O^*(A, B)$ is made.
      i. Let γ be the midpoint between a_1, b_1. Let μ be the midpoint between a_i, b_j.
      ii. Rotate B around the axis line (γ, μ) by θ such that $-180 \leq \theta \leq 180$ and $|\theta|$ is the largest angle which results in $d_F(A, B) < O^*(A, B)$.
      iii. Update $O^*(A, B) \longleftarrow d_F(A, B)$ and update a_i, b_j accordingly.
   (b) Translate until no improvement of $O^*(A, B)$ is made.
      i. Translate B along the vector $\overrightarrow{b_j a_i}$ by δ such that δ is the largest value which results in $d_F(A, B) < O^*(A, B)$.

ii. Update $O^*(A, B) \longleftarrow d_F(A, B)$ and update a_i, b_j accordingly.

5. Return $A, B, O^*(A, B)$.

While we are unable to prove that this algorithm is a PTAS for the optimal alignment problem, we believe that for practical data it is almost a PTAS. Now, we give some evidence that, when translating B such that b_1 collides with a_1 and obtaining subsequently an optimal solution (with b_1 sticking at a_1), in fact gives us a factor-2 approximation for the optimal alignment problem.

Lemma 1. *Given two 3D polygonal chains A, B of length m, n respectively such that the optimal $d_F(A, B) = \epsilon$, an optimal transformation τ aligning A and $\tau(B)$ such that $d(a_1, \tau(b_1)) = 0$ gives a 2-approximation for the optimal alignment problem.*

4.1.2 Empirical Results

In [5], rigorous studies are performed regarding comparing protein backbone 107j.a with the other seven chains from the Protein Database (PDB): 1hfj.c, 1qd1.b, 1toh, 4eca.c, 1d9q.d, 4eca.b, 4eca.d. These seven chains were reported to be similar to 107.j by the ProteinDBS software (which takes a few seconds searching the whole PDB, which contained over 30,000 protein backbones at that time). Using the discrete Fréchet distance as distance measure, while taking much longer (close to one minute for each pair), the heuristic algorithm in [5] reported that 3 of the 7 chains are in fact not really similar to 107.j. ProteinDBS subsequently updated their webpage for this. Here, in Table 4.1, we simply compare our ALIGN algorithm with that of [5]. We mostly focus on accuracy. All distances are measured in angstroms.

Protein Chain (B)	RMSD [5]	$d_F(A,B)$ [5]	$d_F(A,B)$ [1]
1hfj.c	0.27	1.01	0.95
1qd1.b	2.81	22.90	22.65
1toh	2.91	35.09	22.06
4eca.c	1.10	6.01	5.55
1d9q.d	2.88	22.18	20.87
4eca.b	1.09	5.76	5.64
4eca.d	1.45	5.92	5.71

Table 4.1: Alignment with 107j.a (Chain A) where all eight chains have 325 vertices, and the original work is [5] and our new algorithm, ALIGN, is [1].

4.2 CPS-3F Heuristic

Here we try to solve the CPS-3F problem with a practical solution. It is known that the greedy method does not always work even for simplifying a single chain under the discrete Fréchet distance, with some counterexample presented in [6]. Here, we use a greedy backtracking method. Our ideas are as follows: (1) While greedy does not always work, for protein backbones we have the implicit condition that for all possible i $d(a_i, a_{i+1}) \approx 3.7$ to 3.8 (angstroms), i.e., the neighboring α-carbon atoms in a protein backbone have an almost uniform length. With this condition a lot of counterexamples do not hold anymore. (2) To mend possible holes of the algorithm, when we are stuck at a certain point (using the greedy method), we backtrack some (constant number of) steps and re-try the greedy method again. While it is not known whether this algorithm leads to an optimal solution, it works well for practical protein data, some of which are to be presented in Section 5.

We first show a simple lemma which helps us to determine whether the input could lead to an infeasible solution. Certainly, this lemma also implies that having an almost optimal alignment of A and B, resulting in minimizing $d_F(A,B)$, is crucial for the success of the simplification algorithm.

Lemma 2. *Given two 3D polygonal chains A and B, if a solution (A', B') for CPS-3F is found with $d_F(A,A') \leq \delta_1$, $d_F(B,B') \leq \delta_2$ and $d_F(A',B') \leq \delta_3$, then $d_F(A,B) \leq \delta_1 + \delta_2 + \delta_3$.*

4.2.1 Algorithm

Let $\mathcal{B}(b,\delta)$ be a ball centered at point b with radius δ. Our heuristic algorithm for CPS-3F, which assumes that A and B are almost optimally aligned, is as follows.

Algorithm SIMPLIFY$(A, B, K, \delta_1, \delta_2, \delta_3)$:
Input: Two polygonal chains $A = \langle a_1, ..., a_m\rangle$ and $B = \langle b_1, ..., b_n\rangle$, a positive integer K, and three positive constants $\delta_1, \delta_2, \delta_3$.
Output: Two simplified chains $A' = \langle a'_1, ..., a'_K\rangle$ and $B' = \langle b'_1, ..., b'_K\rangle$.

1. Run the algorithm ALIGN(A, B).
2. If $d_F(A,B) > \delta_1 + \delta_2 + \delta_3$, report 'no valid solution' and exit.
3. Initialize $a'_1 \longleftarrow a_1, b'_1 \longleftarrow b_1, i \leftarrow 1, j \leftarrow 1$.
4. Loop until $i = j = K$.
   (a) Let $\langle a_{i,1}, a_{i,2}, ..., a_{i,p}(= a_I)\rangle$ be the maximal subsequence of A which is inside $\mathcal{B}(a'_i, \delta_1)$ and let $\langle b_{j,1}, b_{j,2}, ..., b_{j,q}(= b_J)\rangle$ be the maximal subsequence of B which is inside $\mathcal{B}(b'_j, \delta_2)$. (Note that $a'_i = a_{i,p'}$ for some $p' \leq p$ and $b'_j = b_{j,q'}$ for some $q' \leq q$.)
   (b) Let $\langle a_{I+1}, a_{I+2}, ..., a_{I+s}\rangle$ be the maximal subsequence of A which is inside $\mathcal{B}(a_{I+s'}, \delta_1)$ and let $\langle b_{J+1}, b_{J+2}, ..., b_{J+t}\rangle$ be the maximal subsequence of B which is inside $\mathcal{B}(b_{J+t'}, \delta_2)$, with $s' \leq s, t' \leq t$.
   (c) If $d(a_{I+s'}, b_{J+t'}) \leq \delta_3$, then
      i. $I \longleftarrow I + s$, $J \longleftarrow J + t$,
      ii. $a'_{i+1} \longleftarrow a_{I+s'}$, $b'_{j+1} \longleftarrow b_{J+t'}$,
      iii. $i \leftarrow i + 1, j \leftarrow j + 1$.

(d) Else if $d(a'_i, b_{J+t'}) \leq \delta_3$, then
   i. $J \longleftarrow J+t$, $b'_{j+1} \longleftarrow b_{J+t'}$,
   ii. $j \leftarrow j+1$.
(e) Else if $d(a'_{I+s'}, b_j) \leq \delta_3$, then
   i. $I \longleftarrow I+s$, $a'_{i+1} \longleftarrow a_{I+s'}$,
   ii. $i \leftarrow i+1$.
(f) Else backtrack by successively letting a'_i be $a_{i,p'-1}, a_{i,p'-2}, ..., a_{i,1}$ and letting b'_j be $b_{j,q'-1}, b_{j,q'-2}, ..., b_{j,1}$, and loop over Steps (a) through (e). If neither i nor j can be incremented over these pairs of a'_i and b'_j, exit with a report 'no valid solution'.

The algorithm returns two simplified chains where $|A'| = |B'| = K$ whether they could be simplified more or not. Thus, it is possible for consecutive nodes of A', B' to be equal, e.g. $a_i = a_j = a_r$ where $a_i, a_j \in A', a_r \in A$. This duplication can easily be taken out if the desire is that the chains be less than or equal to K rather than only equal.

4.2.2 Empirical Results

The results for the heuristic algorithm are in Tables 4.2 and 4.3. The setup and values are discussed in detail in Section 4.3.3, so they can be easily compared with the results from our approximation algorithm.

4.3 CPS-3F$^+$

The greedy nature of our heuristic method makes evaluating and controlling the simplification between the two chains difficult, and far from optimal. To improve these results, we define a new metric on the discrete Fréchet distance called the *moving cost*. Using the moving cost, we define a dynammic programming algorithm that minimizes the new measure, and we prove is a 2-approximation algorithm of the optimal CPS-3F solution.

4.3.1 The Moving Cost

Here, we use the theory of weakly increasing integers [8] with simultaneous simplification to define what we call the moving cost of the alignment between two chains.

Definition 11. *The* ***moving cost*** *of a paired walk* $W = \{(A_i, B_i)\}$ *is*

$$m_c^W(A_i, B_i) = \max\{|A_i|, |B_i|\} \tag{4.1}$$

The moving cost of a paired walk W *between* A *and* B *is*

$$m_c^W(A, B) = \sum_{i=1}^{t} m_c^W(A_i, B_i). \tag{4.2}$$

The moving cost for A and B is the sum of the number of "hops" the man or dog make along the two chains. However, when they both move at once, this only counts as a single move. In other words, it is the number of pairs of points, or matched points, between the chains used in calculating the discrete Fréchet distance.

In Figure 4.1 we show a very simple example of two chains that can be simplified in two possible ways. In Figure 4.1(a) the moving cost is six and the number of nodes for each chain is four, and in Figure 4.1(b) the moving cost is still six, yet the number of nodes is now five in each chain.

We can prove some nice properties of the moving cost, such as the complexity being polynomial and its ability to approximate the number of vertices. As we make use of later, $\max(|A|, |B|) \leq m_c^W(A, B) \leq |A| + |B| - t$ for a paired t-walk W along A and B. This is the motivation for our variant of CPS-3F and CPS-2H.

Definition 12 (The Constrained Chain Pair Simplification (CPS$^+$) Problem)**.**

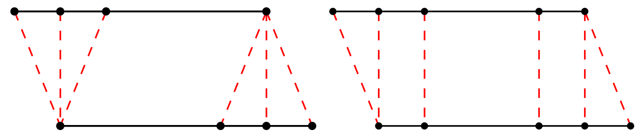

Figure 4.1: The difference between the number of nodes and the moving cost. Suppose that both (a) and (b) are valid simplifications of two chains. They have the same moving cost, yet (a) only has four nodes in each of the simplified chains, but in (b) both chains have five nodes.

Instance: *Given a pair of 3D chains A and B, with lengths $O(m), O(n)$ respectively, an integer $K' > 0$, and $\delta_1, \delta_2, \delta_3 \in \mathbb{R}^+$.*

Problem: *Does there exist a pair of chains A', B' where the vertices are from A, B, respectively, such that for some paired walk W between A', B', $m_c^W(A', B') \leq K'$, and $d_1(A, A') \leq \delta_1$, $d_2(B, B') \leq \delta_2$, $d_F(A', B') \leq \delta_3$?*

When $d_1 = d_2 = d_F$, we call the problem CPS-3F$^+$, and when $d_1 = d_2 = d_H$, the problem is denoted CPS-2H$^+$.

4.3.2 CPS-3F$^+$ $\in$ P

In this section we present a polynomial time solution for CPS-3F$^+$. Several versions of the single chain simplification problem were addressed and shown to be polynomially solvable by Bereg et al. [6]. However, CPS-2H (where the Hausdorff distance is used for $d(A, A')$ and $d(B, B')$) was shown to be **NP**-complete and thus it is believed that the Fréchet version might be as well. The solution presented here proves that under the discrete Fréchet distance, the constrained chain pair simplification problem (CPS-3F$^+$) is polynomially solvable when the dimension is fixed. The algorithm

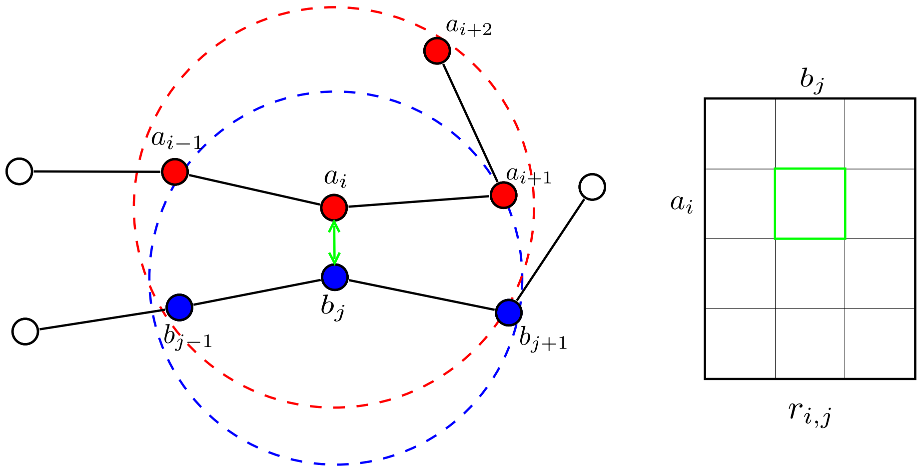

Figure 4.2: The rectangle $r_{i,j}$ constructed from subchains of A, B where $d(a_i, b_j) \leq \delta_3$. Here $S_A(a_i, \delta_1)$ contains the vertices a_{i-1} to a_{i+2}, and $S_B(b_j, \delta_2)$ contains the vertices b_{j-1} to b_{j+1}. Thus, $r_{i,j}$ is defined by the min and max node indices in each subchain.

returns the optimal K', specified in the definition of the decision problem, which is equal to

$$m_c(A', B') = \min_W m_c^W(A', B'), \tag{4.3}$$

among all feasible W. We now define several necessary terms and data structures.

Given two polygonal chains $A = \langle a_1, a_2, ..., a_m \rangle$, $B = \langle b_1, b_2, ..., b_n \rangle$, and constraints $\delta_1, \delta_2, \delta_3 \in \mathbb{R}^+$, we can design a dynamic programming algorithm to find the optimal moving cost K'. First we let $\mathcal{D} = \{(a_i, b_j) |\ a_i \in A,\ b_j \in B \text{ and } d(a_i, b_j) \leq \delta_3\}$. This is the set of all pairs of nodes between the two chains which are at a distance of at most δ_3 from each other. Then we can define a matrix $\mathcal{C}$ of size $m \times n$, which in any cell $\mathcal{C}_{i,j}$, contains the minimum number, K', of pairs $(a_k, b_l) \in \mathcal{D}$, which given δ_1, δ_2, and δ_3 simplify A and B via CPS-3F$^+$ from (a_1, b_1) up to (a_i, b_j).

In order to maintain $\mathcal{C}$, we need another data structure $\mathcal{R}$ and some other helpful definitions. We define $S_X(x_i, \delta)$ as the maximal continuous subchain containing x_i on the polygonal chain X such that all the vertices on this subchain are contained in the sphere centered at x_i and with radius δ. Now let $r_{i,j}$ be the rectangle on $\mathcal{C}$ defined as $\langle \min(S_A(a_i, \delta_1)), \max(S_A(a_i, \delta_1)), \min(S_B(b_j, \delta_2)), \max(S_B(b_j, \delta_2))\rangle$ such that $(a_i, b_j) \in \mathcal{D}$. Here, min and max refer to the minimum or maximum indexed element within $S_X(x_i, \delta)$. For every pair in $\mathcal{D}$, we envision the corresponding rectangles as being overlayed on $\mathcal{C}$. A rectangle $r_{i,j}$ covers all the cells of $\mathcal{C}$ that are analogous to the vertices in $S_A(a_i, \delta_1) \cup S_B(b_j, \delta_2)$ as shown in Figure 4.2.

For convenience we also define the set of all rectangles that a cell in $\mathcal{C}$ belongs to: $\mathcal{Q}_{k,l} = \{r_{i,j} | a_k \in A, b_l \in B$ and $\min(S_A(a_i, \delta_1)) \leq a_k \leq \max(S_A(a_i, \delta_1))$ and $\min(S_B(b_j, \delta_2)) \leq b_l \leq \max(S_B(b_j, \delta_2))\}$.

Now we define $\mathcal{R}$ as a matrix of sets where the matrix is of size m by n, and $\mathcal{R}$ provides information needed to fill out $\mathcal{C}$ by storing a list of rectangles for each cell. $\mathcal{R}_{i,j}$ contains a set of rectangles (dynamic array) which pertain to the number of coverings (rectangles) still viable at any (i, j) relating to the number already calculated for $\mathcal{C}_{i,j}$. These are computed by the recurrences in Equations 4.4 and 4.5, which are shown along with the initial conditions for the relations.

Initial Conditions: $\mathcal{Q}_{1,1} \neq \emptyset$, $\mathcal{R}_{1,1} = \mathcal{Q}_{1,1}$, and $\mathcal{C}_{1,1} = 1$.

$$\mathcal{C}_{i,j} = \min_{(k,l)\in\{(i\text{-}1,j),(i,j\text{-}1),(i\text{-}1,j\text{-}1)\}} \begin{cases} \mathcal{C}_{k,l}, & \text{if } \mathcal{Q}_{i,j} \cap \mathcal{R}_{k,l} \neq \emptyset \\ \mathcal{C}_{k,l} + 1, & \text{if } \mathcal{Q}_{i,j} \cap \mathcal{R}_{k,l} = \emptyset,\ \mathcal{Q}_{i,j} \neq \emptyset,\ \mathcal{R}_{k,l} \neq \emptyset \\ NULL, & \text{if } \mathcal{Q}_{i,j} = \emptyset \end{cases} \tag{4.4}$$

$$\mathcal{R}_{i,j} = \bigcup_{(k,l)\in\{(i\text{-}1,j),(i,j\text{-}1),(i\text{-}1,j\text{-}1)\}} \begin{cases} \mathcal{R}_{k,l} \cap \mathcal{Q}_{i,j}, & \text{if } \mathcal{C}_{i,j} = \mathcal{C}_{k,l},\ \mathcal{R}_{k,l} \cap \mathcal{Q}_{i,j} \neq \emptyset \\ \mathcal{Q}_{i,j}, & \text{if } \mathcal{C}_{i,j} = \mathcal{C}_{k,l} + 1,\ \mathcal{R}_{k,l} \neq \emptyset, \\ & \mathcal{R}_{k,l} \cap \mathcal{Q}_{i,j} = \emptyset \end{cases} \tag{4.5}$$

The idea is to find the minimum covered xy-monotone increasing path from (a_1, b_1) to (a_m, b_n) which corresponds to $\mathcal{C}_{1,1}$ to $\mathcal{C}_{m,n}$. This is the minimum path by basic dynamic programming with all feasible options explored. If we visited a cell that was not covered, that would mean one of the nodes is not covered by a pair in $\mathcal{D}$. By finding this covered walk, one guarantees that every column and every row is covered by at least one rectangle which means all of the nodes of A and B are covered.

The increasing xy-monotone path is necessary in the recurrence due to the definition of the discrete Fréchet distance. Without the requirement of a monotonically increasing path this would be using the weak discrete Fréchet distance.

We first characterize the optimal substructure of CPS-3F$^+$ as an optimization problem given our definitions, and then show this yields the optimal solution for K' and thus decides CPS-3F$^+$. All theorems have been proven, though this work has been omitted for brevity.

Theorem 2. *Optimal substructure of CPS-3F$^+$:*
Let $A = \langle a_1, \ldots, a_m \rangle$ and $B = \langle b_1, \ldots, b_n \rangle$ be two polygonal chains, $\delta_1, \delta_2, \delta_3 \in \mathbb{R}^+$, and let $Z_i = \langle z_1, \ldots, z_i \rangle$ such that every z_j is a rectangle, be any CPS-3F$^+$ solution.

1. *If (a_k, b_l) is covered by z_i where $(k, l) \in \{(m\text{-}1, n), (m, n\text{-}1), (m\text{-}1, n\text{-}1)\}$, then Z_i is a CPS-3F$^+$ solution for A_k, B_l.*

2. *If (a_k, b_l) is covered by z_{i-1} where $(k, l) \in \{(m\text{-}1, n), (m, n\text{-}1), (m\text{-}1, n\text{-}1)\}$, then Z_{i-1} is a CPS-3F$^+$ solution for A_k, B_l.*

3. *If (a_k, b_l) is not covered by z_i or z_{i-1} where $(k, l) \in \{(m\text{-}1, n), (m, n\text{-}1), (m\text{-}1, n\text{-}1)\}$, then $\nexists$ a CPS-3F$^+$ solution for A_k, B_l.*

Theorem 3. *Constrained chain pair simplification, under the discrete Fréchet distance, is polynomially solvable, i.e. CPS-3F$^+$ $\in$ $\boldsymbol{P}$.*

Corollary 1. *Constrained chain pair simplification gives a factor 2-approximation to the chain pair simplification problem under the discrete Fréchet distance, i.e., CPS-3F$^+$ provides a 2-approximation of CPS-3F.*

The time complexity is largely dependent on δ_1, δ_2, and δ_3 because they define the size and number of rectangles. We allow δ_1, δ_2, and δ_3 to be absorbed in the complexity because their values do not guarantee a specific number of rectangles to be considered, nor how large a given rectangle is. We can easily bound the complexity between $O(mn)$ and $O(m^2n^2)$. If the values of δ_1, δ_2, and δ_3 are small then any cell will only have a small constant number of rectangles to consider and the algorithm runs in $O(mn)$ time, which is the case for most protein related data.

The space complexity also has similar bounds, requiring a minimum of $O(mn)$ space and a maximum of $O(m^2n^2)$ space if $\mathcal{Q}$ is used naïvely and built beforehand. The recurrences themselves only require two rows of data for either $|A|$ or $|B|$, so the space complexity is linear to the size of the smaller chain (WLOG $O(n)$). However, this would require calculating $\mathcal{Q}_{i,j}$ at every step for the cell, which as discussed, could be expensive if the delta values are large.

4.3.3 Empirical Results

We now present some results comparing our previous heuristic method SIMPLIFY [1] and the 2-approximation solution of CPS-3F$^+$. We present the results for chains with a similar length and then look at dissimilar chains of various lengths in order to vary the amount of simplification per chain.

We note that in the result sections, the RMSD values were taken from ProteinDBS, and thus the alignment length, or coverage, is not the full length of each chain [29]. This is especially true when discussing chains of different lengths in 4.4. This makes a straightforward comparison between the chains using both RMSD and the discrete Fréchet distance difficult. However, the results are mainly to compare CPS-3F$^+$ to our previous algorithm SIMPLIFY [1], and thus the coverage is not listed.

Using the same format as our previous results, we set $\delta_1 = \delta_2$ for simplicity and to ensure chains A', B' will have a similar reduced length since nearly all are the same length initially. δ_3 is set to the minimum integer value that will reduce the chains via CPS-3F$^+$ given δ_1, δ_2. The comparison tables in both cases are using the protein backbone 107j.a (protein A) and comparing it with seven other chains from the Protein DataBank: 1hfj.c, 1qd1.b, 1toh, 4eca.c, 1d9q.d, 4eca.b, 4eca.d. These seven chains were reported to be similar to 107j.a by the ProteinDBS software [29] (this took a few seconds searching the whole PDB, which contained over 30,000 protein backbones at that time). Previously, [5] used a heuristic algorithm based on the discrete Fréchet distance and showed that three of the seven chains were not actually similar to 107j.a, and ProteinDBS has subsequently updated their page to reflect this. The protein chain 107j.a and all but one of the seven chains have 325 nodes along the backbone.

For the CPS-3F$^+$ algorithm, all chains are assumed to be aligned, and we use the alignments from our previous algorithm ALIGN [1]. In Table 4.2 we fixed $\delta_1 = \delta_2 = 4$ since the distance between two α-carbon atoms in the backbone is approximately $\approx$ 3.7 to 3.8 (angstroms). This value ensures that we will be simplifying the chains a minimal amount. We can see that we get an approximate reduced length of 1/3 which is what we would expect (since this distance will only use the neighboring nodes). The optimal algorithm allows for δ_3 to be much smaller than the heuristic because it can simplify the chains with a value often less than $d_F(A, B)$, and hence $d_F(A', B')$ is a lower value.

In Table 4.3 we vary δ_1 and δ_2 for different amounts of simplification and again set δ_3 to the minimum integer value that allows for simplification via CPS-3F$^+$. We keep $\delta_1 = \delta_2$ for simplicity and to ensure a similar reduced size for both chains. Here we have a more dramatic difference in δ_3 and in $d_F(A', B')$ because of the greater simplification possibilities between A, A' and B, B' since δ_1, δ_2 are much larger. This demonstrates how CPS-3F$^+$ is able to simplify the two chains simultaneously while highlighting the similarities between the two chains. This is especially noticeable in that the discrete Fréchet distance between the simplified chains, $d_F(A', B')$, is drastically less than that of the original chains, $d_F(A, B)$.

We can see that the optimal results far exceed the heuristic approximation. If we look at 4eca.c in Table 4.3, the difference between the heuristic (11.73) and the optimal (2.90) is dramatic. The optimal δ_3 for CPS-3F$^+$ is 3 to 4 times smaller than the heuristic in general, and the discrete Fréchet distance between A' and B' is smaller than the original distance between A, B.

The heuristic algorithm only allowed for a constant number of backtracking steps which resulted in both chains being simplified to a similar number of vertices. With CPS-3F$^+$, we can see that the chains can vary greatly in the amount they simplify in order to have a minimum moving cost.

One aspect of chain pair simplification we have not exploited is simplifying the chains differently. Here, we look at chains that vary in length, are not aligned as well with the base chain, and subsequently have a large discrete Fréchet distance. Table 4.4 shows these results. The values for δ_1 and δ_2 were chosen in an attempt to simplify both chains to a similar size via CPS-3F$^+$. This allows us to pull out the similarities of two chains that may be vastly different without simplification, yet still have some subset of nodes that align and compare well. For visualization purposes, it lets us see the overall subset similarity structure of the two chains. This method could prove

Protein Chain (B)	$\|B\|$	RMSD [5]	$d_F(A,B)$	δ_1	δ_2	δ_3 [1]	$\|A'\|$ [1]	$\|B'\|$ [1]	K' [1]	$d_F(A',B')$ [1]	δ_3 [7]	$\|A'\|$ [7]	$\|B'\|$ [7]	K' [7]	$d_F(A',B')$ [7]
1hfj.c	325	0.27	0.95	4	4	1	109	109	109	0.95	1	109	109	109	0.95
1qd1.b	325	2.81	22.65	4	4	47	110	109	110	24.96	21	117	126	150	20.70
1toh	325	2.91	22.06	4	4	60	109	110	110	23.39	21	149	130	178	20.54
4eca.c	325	1.10	5.55	4	4	20	109	109	109	7.96	6	110	111	111	5.97
1d9q.d	297	2.88	20.87	4	4	43	109	108	109	23.68	20	130	127	166	19.86
4eca.b	325	1.09	5.64	4	4	17	109	109	109	7.51	5	110	111	111	4.89
4eca.d	325	1.45	5.71	4	4	18	109	109	109	7.82	5	111	113	113	4.94

Table 4.2: Comparison of Algorithm SIMPLIFY [1] and FIND-CPS-3F$^+$ with 107j.a (Chain A) of length 325. $\delta_1 = \delta_2 = 4$, and δ_3 set to the minimal value. The heuristic method is in [1] and the CPS-3F$^+$ results are in [7].

Protein Chain (B)	$\|B\|$	RMSD [5]	$d_F(A,B)$	δ_1	δ_2	δ_3 [1]	$\|A'\|$ [1]	$\|B'\|$ [1]	K' [1]	$d_F(A',B')$ [1]	δ_3 [7]	$\|A'\|$ [7]	$\|B'\|$ [7]	K' [7]	$d_F(A',B')$ [7]
1hfj.c	325	0.27	0.95	12	12	4	28	28	28	3.77	1	26	26	26	0.95
1qd1.b	325	2.81	22.65	15	15	33	16	17	17	22.64	12	21	23	24	11.94
1toh	325	2.91	22.06	16	16	34	18	16	18	28.51	13	22	19	22	12.80
4eca.c	325	1.10	5.55	12	12	12	28	28	28	11.73	3	27	27	27	2.90
1d9q.d	297	2.88	20.87	15	15	27	19	21	21	23.73	13	22	24	26	12.99
4eca.b	325	1.09	5.64	12	12	8	28	28	28	7.81	3	26	26	26	2.94
4eca.d	325	1.45	5.71	12	12	11	28	28	28	10.01	3	32	32	32	2.99

Table 4.3: Comparison of Algorithm SIMPLIFY [1] and FIND-CPS-3F$^+$ with 107j.a (Chain A) of length 325. $\delta_1 = \delta_2$, and δ_3 set to the minimal value. The heuristic method is in [1] and the CPS-3F$^+$ results are in [7].

Protein Chain (B)	$\lvert B\rvert$	RMSD	$d_F(A,B)$	δ_1	δ_2	δ_3 [1]	$\lvert A'\rvert$ [1]	$\lvert B'\rvert$ [1]	K' [1]	$d_F(A',B')$ [1]	δ_3 [7]	$\lvert A'\rvert$ [7]	$\lvert B'\rvert$ [7]	K' [7]	$d_F(A',B')$ [7]
3ntx.a	322	2.14	10.04	10	10	22	35	40	40	16.21	5	39	39	39	4.91
1wls.a	316	2.18	11.97	15	13	32	16	25	25	20.50	6	22	22	22	5.99
2eq5.a	215	2.72	22.35	8	6	39	53	43	53	23.47	19	58	53	66	18.91
2zsk.a	219	2.85	21.92	12	8	30	27	31	31	24.60	17	38	34	43	16.90
1zq1.a	363	3.01	23.38	10	12	40	36	37	37	28.30	19	51	53	56	18.47
3jq0.a	457	11.52	27.36	6	9	52	71	54	71	30.75	26	65	70	80	25.67
2fep.a	273	3.33	24.55	20	17	27	13	13	13	25.00	10	10	11	11	9.94

Table 4.4: Comparison of Algorithm SIMPLIFY [1] and FIND-CPS-3F$^+$with 107j.a (Chain A) of length 325, and various δ_1, δ_2, and δ_3 set to simplify both chains to a similar length. The heuristic method is in [1] and the CPS-3F$^+$ results are in [7].

useful when finding nodes in each chain that match well together, i.e. they have a low moving cost and small discrete Fréchet distance.

The heuristic method SIMPLIFY [1] does not find similar optimal simplifications, and also ends up with a much higher moving cost and δ_3. The discrete Fréchet distance, consequently, is also much higher. As in our previous results, for both SIMPLIFY and CPS-3F$^+$, we picked δ_1 and δ_2, and then report the smallest integer value of δ_3 that worked for the respective algorithm.

The disparity between the number of vertices and the moving cost (K') is lessened if the chains simplify in a similar fashion. When δ_1 and δ_2 are large, but δ_3 is small, it allows for these larger "hops" to be made, and thus the simplified chains are similar in length, and the moving cost is a closer approximation to K. Using larger than minimum values for δ_3, we allow for greater flexibility in the simplification and it will yield a lower moving cost.

4.4 The Fréchet-based Protein Alignment & Comparison Toolkit

To facilitate research using the discrete Fréchet distance we have also created a set of libraries to run any of our algorithms based on the discrete Fréchet distance. The FPACT (The Fréchet-based Protein Alignment & Comparison Toolkit) libraries were designed for easy access to the algorithms by being modular and protein file format independent. The toolkit includes methods and classes such as the discrete Fréchet distance, ALIGN [1], SIMPLIFY [1], versions of CPS-3F$^+$ optimized for space or time efficiency [7], the CPS-3F$^+$ backtracking algorithm [7], and some other utility functions. The libraries will be updated with any future algorithms or results as well. All libraries are written and available in both C# and Python with Numpy.

We have also implemented a simple web-based application which uses these libraries. The web-based application runs within the Silverlight framework, and can be used in any browser supporting the Silverlight or Moonlight runtime. The software is available to the public for general use, thus providing the ability to align, compare, and simplify protein backbones with the discrete Fréchet distance without directly using the libraries [9].

FPACT, the web application, and relevant documentation about the research, can be found at the website http://www.cs.montana.edu/∼timothy.wylie/frechet/.

CHAPTER 5
FUTURE WORK

We now cover our ongoing and future research to conclude the scope of the dissertation. There are some open problems related to chain pair simplification that need to be answered, and then we move into a related area of path finding algorithms based on the discrete Fréchet distance.

Our recent work on the discrete versions of the map matching and set-chain matching problems are more theoretical and seek to prove the complexity of the problems. For the problems that are polynomially solvable, we also prove the optimal substructure and give a dynamic programming solution.

5.1 Chain Pair Simplification

In [7] we introduced the idea of the *moving cost* of two polygonal curves compared under the discrete Fréchet distance (Section 4.3.1). We then defined the constrained chain pair simplification problem (Definition 12), and then proved that under the discrete Fréchet distance, CPS-3F$^+$ is polynomially solvable. We plan to address two open problems posed by this work.

The complexity of CPS-3F is unknown, and it is easy to see that every solution with a minimum moving cost is not a CPS-3F solution, but does every solution of CPS-3F have a minimum moving cost? This is most likely not true, but needs to be proven or shown with a counter-example.

CPS-2H was proven to be **NP**-complete [6], and we need to look at the complexity of CPS-2H$^+$. We believe this problem will also be **NP**-complete. If proving this becomes infeasible, evaluating CPS-1H^{+} may prove fruitful. If no positive results come from this investigation, then finding a decent approximation algorithm or heuristic method will be a priority.

5.2 Set-chain Matching

Here, we look at the discrete version of a problem originally defined in [10] with the continuous Fréchet distance, and then look at the variations of this problem that are also of interest. Figure 5.1 shows a simple instance of the problem.

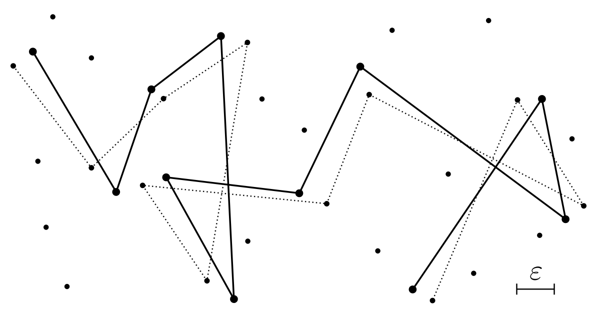

Figure 5.1: An instance of the set matching problem in 2D with one possible solution of $k \geq 11$.

The set-chain matching problem is equivalent to the map matching problem for complete graphs. Essentially, there is more freedom because any choice of points is possible. In a graph, however, you are restricted by the neighbors at any given vertex. We first formally define the problem and its variations. It is important to note that, as in the continuous version, we make no requirements that P or Q be planar. Further, for problem classification we mainly focus on the reachable points (defined below). For discussion, we will refer to the number of nodes in a polygonal chain as the "size" of the chain and will be denoted as $|A|$ for a polygonal chain A.

Definition 13 (The Discrete Set-Chain Matching Problem)**.**

Instance: *Given a point set S, a polygonal curve P in $\mathbb{R}^d$ $(d \geq 2)$, an integer $K \in \mathbb{Z}^+$, and an $\varepsilon > 0$.*

Problem: *Does there exist a polygonal curve Q with vertices chosen from S' where $S' \subseteq S$, such that $T \leq K$ and $d_F(P, Q) \leq \varepsilon$?*

T is defined in two ways. When we are minimizing the number of nodes in the chain, $T = |Q|$, and if we are minimizing the points used then $T = |S'|$.

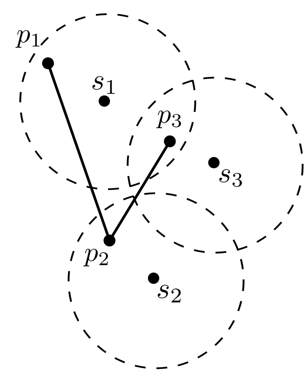

Figure 5.2: The difference between minimizing $|Q|$ and $|S'|$. Minimizing $|S'|$ gives $Q = \langle s_1, s_2, s_1 \rangle$ where $|S'| = 2$ and $|Q| = 3$, but minimizing $|Q|$ will yield $|Q| = 3$ whether it uses the sequence $\langle s_1, s_2, s_1 \rangle$ or $\langle s_1, s_2, s_3 \rangle$.

Now define $S_\varepsilon = \{s \in S | p \in P \text{ and } d(p, s) \leq \varepsilon\}$ as the reachable points because these are the points s that can be reached by p within the ε given. Here, we look at four variations of this problem. They vary whether or not there is a uniqueness constraint on $s \in S$ being used as a node in Q (if points may be used more than once), and whether our goal is to minimize the size of the chain Q or of the set S'. We therefore distinguish the problems as Unique/Non-unique(U/N) Set-Chain(S) Matching(M) with a k Subset/Chain(S/C). The variants are thus NSMS-k, NSMC-k, and USM-k. When looking at unique nodes, minimizing $|Q|$ is equivalent to minimizing the set of points used, $|S'|$, since they can only be used once, so we do not separate the cases.

Theorems 4 and 5 are fairly straightforward to prove along with the recurrence relation shown in Equation 5.1, although this work has yet to be published.

$$M[i,j] = \min \begin{cases} M[i,j-1], & \text{if } d(s_i,p_j) \leq \varepsilon, M[i,j-1] \neq \emptyset \\ \min_{k=1}^{|S|}(M[k,j-1]) + 1, & \text{if } d(s_i,p_j) \leq \varepsilon, M[i,j-1] = \emptyset \\ \emptyset, & \text{if } d(s_i,p_j) > \varepsilon \end{cases} \tag{5.1}$$

Theorem 4 (Optimal Substructure of NSMC-k)**.** *Let $P = \langle p_1, ..., p_n \rangle$ be a polygonal chain, and $S = \{s_1, ..., s_m\}$ be a set of points such that there exists a $Q = \langle q_1, ..., q_k \rangle$ through a set $S' \subseteq S$ which is a minimal sequence such that $d_F(P,Q) \leq \varepsilon$.*
(1) If $d(p_{n-1}, q_k) \leq \varepsilon$ and $d(p_{n-1}, q_{k-1}) > \varepsilon$, then Q_k is an optimal solution for P_{n-1}.
(2) If $d(p_{n-1}, q_{k-1}) \leq \varepsilon$, then Q_{k-1} is an optimal solution for P_{n-1}.
(3) If $d(p_{n-1}, q_k) > \varepsilon$, then Q_{k-1} is an optimal solution for P_{n-1}.

Theorem 5. *The Non-unique Set-Chain Matching (NSMC-k) problem minimizing $|Q|$ is in **P**.*

Theorem 6 is proven by a reduction from planar 3-SAT [33] and was printed in a short abstract and presented over the summer [36] of 2012, but has not been officially published. An example of this last reduction is shown in Figure 5.3.

Theorem 6. *The Unique Set-Chain Matching (USM-k) problem is **NP**-complete.*

In the original work dealing with the continuous Fréchet distance, the authors looked at the equivalent of the NSMC problem without a minimization constraint, i.e., they were only concerned whether a path could be found. Thus, the complexities of the other variations that we have defined are still unknown.

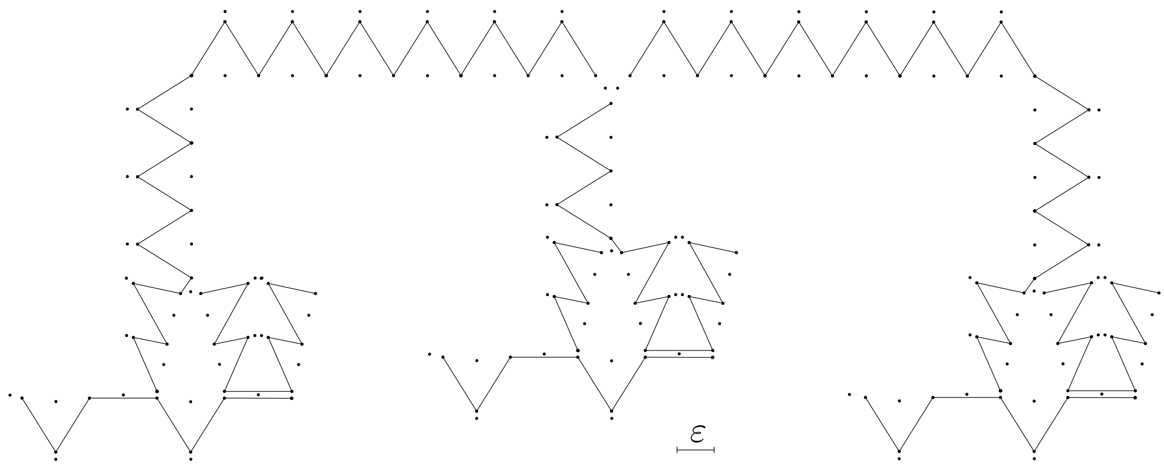

Figure 5.3: Example clause with three variables $c_i = (\bar{x}_1 \cup x_2 \cup \bar{x}_3)$ with assignments $x_1 = 0$, $x_2 = 0$, $x_3 = 1$.

5.3 Map Matching

We now turn our attention to the related map matching problem [11] where the goal is to find a path through a graph rather than a set of points. We first define the problem formally for the discrete Fréchet distance, and then discuss the related variants of the problem which we will address.

Definition 14 (The Discrete Map Matching Problem)**.**
Instance: *Given a simple connected planar graph* $G = (V, E)$ *with a straight-line embedding in* $\mathbb{R}^2$*, a polygonal curve* P *in* $\mathbb{R}^d$ $(d \geq 2)$*, an integer* $K \in \mathbb{Z}^+$*, and an* $\varepsilon > 0$.
Problem: *Does there exist a path* Q *in* G *with the polygonal curve formed by its edges using vertices chosen from* V' *where* $V' \subseteq V$*, such that* $T \leq K$ *and* $d_F(P, Q) \leq \varepsilon$*?*

Again, T is defined in two ways. When we are minimizing the size of the chain, $T = |Q|$, and if we are minimizing the vertices in the graph used then $T = |V'|$. We look at the analogous versions of the set-chain matching problems for each of these: NMMC-k, NMMS-k, and UMM-k. Again we note that when the nodes are unique the two minimization problems are equivalent.

For the NMMC-k problem, the difference in the recurrence can be seen in Equation 5.2, which shows the added difficulty in the map matching versions by restricting the movement to only the neighbor set, $N(v_i)$, for a vertex $v_i \in V$.

$$M[i,j] = \min \begin{cases} M[i,j\text{-}1], & \text{if } d(v_i,p_j) \leq \varepsilon, v_i \in N(v_{i\text{-}1}), M[i,j\text{-}1] \neq \emptyset \\ \min\limits_{k \in N(v_i)} (M[k,j\text{-}1]) + 1, & \text{if } d(v_i,p_j) \leq \varepsilon, v_i \in N(v_{i\text{-}1}), M[i,j\text{-}1] = \emptyset \\ \emptyset, & \text{if } d(v_i,p_j) > \varepsilon \text{ or } v_i \notin N(v_{i\text{-}1}) \end{cases} \tag{5.2}$$

Theorem 7. *The Non-unique Map Matching (NMMC-k) problem minimizing* $|Q|$ *is in* ***P***.

Similar to the Set-chain matching problem the original work, based on the continuous Fréchet distance, considered the equivalent of the NMMC problem without a minimization constraint. Again, the complexities of the other variations that we defined are unknown for the continuous Fréchet distance.

CHAPTER 6
CONCLUSION

This proposal has covered our work that has already been published as well as the research that is awaiting publication or is unfinished.

We now conclude the proposal by highlighting the open questions and future research to be done, and outline the plan and timeline to accomplish these goals. This is an approximate outline for the work that needs to be done in order to finish the dissertation and meet the research and time requirements.

6.1 Dissertation Plan

1. Begin writing the dissertation, taking into consideration the feedback from the comprehensive presentation and the doctoral committee's proposal input.

2. Prove whether a CPS-3F solution must have a minimum moving cost or give a counter-example demonstrating this is not the case.

3. Examine CPS-2H^{+} and look at proving the complexity of the problem. If this becomes infeasible, evaluating CPS-1H^{+} may prove fruitful. If no positive results come from this investigation, then finding a decent approximation algorithm or heuristic method will be a priority.

4. Additional research contributions to the submitted journal paper, which has passed reviews, but is awaiting more material for acceptance. These contributions should come directly from the previous two items.

5. Prove the complexities of the variations of the discrete map and set-chain matching problems defined in Sections 5.2 and 5.3.

6. If possible, extend these results to the continuous Fréchet versions of the map and set-chain matching problems.

7. Submit the work on the matching problems to a conference for publication.

8. Finish writing the dissertation.

9. Present and defend the dissertation in August, 2013 for graduation.

REFERENCES CITED

[1] Tim Wylie, Jun Luo, Binhai Zhu. A practical solution for aligning and simplifying pairs of protein backbones under the discrete Fréchet distance. *Proceedings of the 2011 international conference on Computational science and its applications - Volume Part III*, ICCSA'11, pages 74–83, Berlin, Heidelberg, 2011. Springer-Verlag.

[2] Maurice Fréchet. Sur quelques points du calcul fonctionnel. *Rendiconti del Circolo Matematico di Palermo (1884 - 1940)*, 22(1):1–72, December 1906. doi: 10.1007/BF03018603.

[3] Helmut Alt, Michael Godau. Computing the Fréchet distance between two polygonal curves. *International Journal of Computational Geometry and Applications*, 5:75–91, 1995.

[4] Thomas Eiter, Heikki Mannila. Computing discrete Fréchet distance. Technical Report CD-TR 94/64, Information Systems Department, Technical University of Vienna, 1994.

[5] Minghui Jiang, Ying Xu, Binhai Zhu. Protein structure-structure alignment with discrete Fréchet distance. *Journal of Bioinformatics and Computational Biology*, 6(1):51–64, 2008.

[6] Sergey Bereg, Minghui Jiang, Wencheng Wang, Boting Yang, Binhai Zhu. Simplifying 3d polygonal chains under the discrete Fréchet distance. *Proceedings of the 8th Latin American conference on Theoretical informatics*, LATIN'08, pages 630–641, Berlin, Heidelberg, 2008. Springer-Verlag.

[7] Tim Wylie, Binhai Zhu. A polynomial time solution for protein chain pair simplification under the discrete Fréchet distance. *Proceedings of the 2012 International Symposium on Bioinformatics Research and Applications*, ISBRA'12, pages 287–298, Berlin, Heidelberg, 2012. Springer-Verlag.

[8] Axel Mosig, Michael Clausen. Approximately matching polygonal curves with respect to the Fréchet distance. *Computational Geometry: Theory and Applications*, 30(2):113–127, Feb 2005.

[9] Tim Wylie. FPACT: The Fréchet-based protein alignment & comparison toolkit, 2012. http://www.cs.montana.edu/∼timothy.wylie/frechet.

[10] Anil Maheshwari, Jrg-Rdiger Sack, Kaveh Shahbaz, Hamid Zarrabi-Zadeh. Staying close to a curve. *Proceedings of the 23rd Annual Canadian Conference on Computational Geometry*, CCCG'11, 2011. August 10-12, 2011.

[11] Helmut Alt, Alon Efrat, Günter Rote, Carola Wenk. Matching planar maps. *J. Algorithms*, 49(2):262–283, November 2003.

[12] Boris Aronov, Sariel Har-Peled, Christian Knauer, Yusu Wang, Carola Wenk. Fréchet distance for curves, revisited. *Proceedings of the 14th conference on Annual European Symposium - Volume 14*, ESA'06, pages 52–63, London, UK, 2006. Springer-Verlag.

[13] Helmut Alt, Michael Godau. Measuring the resemblance of polygonal curves. *Proceedings of the eighth annual symposium on Computational geometry*, SoCG'92, pages 102–109, New York, NY, USA, 1992. ACM.

[14] Pankaj K. Agarwal, Rinat Ben Avraham, Haim Kaplan, Micha Sharir. Computing the discrete Fréchet distance in subquadratic time. *CoRR*, abs/1204.5333, 2012.

[15] Felix Hausdorff. *Grundzge der mengenlehre.* Von Veit, Leipzig, 1914.

[16] J.R. Munkres. *Topology.* Prentice Hall, Incorporated, 2000.

[17] Helmut Alt, Bernd Behrends, Johannes Blömer. Approximate matching of polygonal shapes (extended abstract). *Proceedings of the 7th Annual Symposium on Computational Geometry*, SoCG '91, pages 186–193, New York, NY, USA, 1991. ACM.

[18] Helmut Alt, Christian Knauer, Carola Wenk. Matching polygonal curves with respect to the Fréchet distance. *Proceedings of the 18th Annual Symposium on Theoretical Aspects of Computer Science*, STACS '01, pages 63–74, London, UK, UK, 2001. Springer-Verlag.

[19] Carola Wenk. *Shape Matching in Higher Dimensions.* Doctoral Dissertation, Freie Universitaet Berlin, 2002.

[20] S. B. Needleman, C. D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48:443–453, 1970.

[21] C A Mauzy, M A Hermodson. Structural homology between rbs repressor and ribose binding protein implies functional similarity. *Protein Science*, 1(7):843–849, 1992.

[22] W. Taylor, C. Orengo. Protein structure alignment. *Journal of Molecular Biology*, 208:1–22, 1989.

[23] L. Holm, C. Sander. Protein structure comparison by alignment of distance matrices. *Journal of Molecular Biology*, 233:123–138, 1993.

[24] Liisa Holm, Jong Park. Dalilite workbench for protein structure comparison. *Bioinformatics*, 16(6):566–567, 2000.

[25] C.A. Orengo, A.D. Michie, D.T. Jones, M.B. Swindells, J.M. Thornton. Cath: A hierarchic classification of protein domain structures. *Structure*, 1(5):1093–1108, 1997. Using secondary structures to measure the geometry of a protein.

[26] I. Shindyalov, P. Bourne. Protein structure alignment by incremental combinatorial extension (ce) of the optimal path. *Protein Engineering*, 11:739–747, 1998.

[27] Loredana Lo Conte, Bart Ailey, Tim J. P. Hubbard, Alexey G. Murzin, Cyrus Chothia. Scop: a structural classification of proteins database. *Nucleic Acids Research*, 28:257–259, 2000.

[28] C. Strauss A. Oritz, O. Olmea. Mammoth (matching molecular models obtained from theory): an automated method for model comparison. *Protein Science*, 11:2606–2621, 2002.

[29] Chi-Ren Shyu, Pin-Hao Chi, Grant Scott, Dong Xu. Proteindbs: a real-time retrieval system for protein structure comparison. *Nucleic Acids Research*, 32(Web Server issue):W572–575, 2004.

[30] Jinn-Moon M. Yang, Chi-Hua H. Tung. Protein structure database search and evolutionary classification. *Nucleic Acids Research*, 34(13):3646–3659, 2006.

[31] Kevin Buchin, Maike Buchin, Yusu Wang. Exact algorithms for partial curve matching via the Fréchet distance. *Proceedings of the twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '09, pages 645–654, Philadelphia, PA, USA, 2009. Society for Industrial and Applied Mathematics.

[32] Anne Driemel, Sariel Har-Peled. Jaywalking your dog: computing the Fréchet distance with shortcuts. *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '12, pages 318–337. SIAM, 2012.

[33] David Lichtenstein. Planar Formulae and Their Uses. *SIAM Journal on Computing*, 11(2):329–343, 1982.

[34] Alexander Wolff. A simple proof for the NP-hardness of edge labeling. Technical Report W-SPNPH-00, Institut für Mathematik und Informatik, Universität Greifswald, 2000.

[35] Minghui Jiang, Sergey Bereg, Zhongping Qin, Binhai Zhu. New bounds on map labeling with circular labels. *Proceedings of the 15th international conference on Algorithms and Computation*, ISAAC'04, pages 606–617, Berlin, Heidelberg, 2004. Springer-Verlag. LNCS 3341. `doi:http://dx.doi.org/10.1007/978-3-540-30551-4_53`.

[36] Tim Wylie, Binhai Zhu. Discretely following a curve. *Short Abstract for Computational Geometry: Young Researchers Forum*, CG:YRF'12, pages 33–34, 2012.

[37] Minghui Jiang. *Map Labeling with Circles.* Doctoral Dissertation, Montana State University, 2005.

[38] Hee-Kap Ahn, Christian Knauer, Marc Scherfenberg, Lena Schlipf, Antoine Vigneron. Computing the discrete Frchet distance with imprecise input. Otfried Cheong, Kyung-Yong Chwa, Kunsoo Park, editors, *Proceedings of the 21st International Symposium on Algorithms and Computation (ISAAC 2010)*, Volume 6507 series *Lecture Notes in Computer Science*, pages 422–433. Springer-Verlag, Berlin/Heidelberg, Germany, 2010.

[39] Richard Cole. Slowing down sorting networks to obtain faster sorting algorithms. *Journal of the ACM*, 34(1):200–208, January 1987.

[40] Xiaohua Xu, Zhu Wang. Wireless coverage via dynamic programming. Yu Cheng, DoYoung Eun, Zhiguang Qin, Min Song, Kai Xing, editors, *Wireless Algorithms, Systems, and Applications*, Volume 6843 series *Lecture Notes in Computer Science*, pages 108–118. Springer Berlin Heidelberg, 2011.

[41] Gautam K. Das, Robert Fraser, Alejandro Lòpez-Ortiz, Bradford G. Nickerson. On the discrete unit disk cover problem. *Proceedings of the 5th international conference on WALCOM: algorithms and computation*, WALCOM'11, pages 146–157, Berlin, Heidelberg, 2011. Springer-Verlag.

[42] Rashmisnata Acharyya, Manjanna B., Gautam K. Das. Unit disk cover problem. *CoRR*, abs/1209.2951, 2012.

[43] T. K. Vintsyuk. Speech discrimination by dynamic programming. *Cybernetics*, 4(1):52–57, 1968. Russian Kibernetika 4(1):81-88 (1968).

[44] Fumitada Itakura. Minimum prediction residual principle applied to speech recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 23(1):67–72, 1975.

[45] Hiroaki Sakoe, Seibi Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 26(1):43–49, 1978.

[46] Lawrence Rabiner, Biing-Hwang Juang. *Fundamentals of speech recognition.* Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993.

[47] Meinard Müller. *Information Retrieval for Music and Motion.* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2007.

[48] R. Sriraghavendra, Karthik K., C. Bhattacharyya. Fréchet distance based approach for searching online handwritten documents. *Proceedings of the Ninth International Conference on Document Analysis and Recognition - Volume 01*, ICDAR '07, pages 461–465, Washington, DC, USA, 2007. IEEE Computer Society.

[49] R. Sriraghavendra. *Language-neutral Algorithms for Partial Search on Online Handwritten Text*. Doctoral Dissertation, Indian Institute of Science, Jul 2007.

[50] Simina Vasilache, Nazanin Mirshahi, Soo-Yeon Ji, James Mottonen, Donald J. Jacobs, Kayvan Najarian. A signal processing method to explore similarity in protein flexibility. *Advances in Bioinformatics*, 2010, 2010.

[51] Binhai Zhu. Protein local structure alignment under the discrete Fréchet distance. *Journal of Computational Biology*, 14(10):1343–1351, 2007.

[52] Maike Buchin. *On the Computability of the Fréchet Distance Between Triangulated Surfaces*. Doctoral Dissertation, Free University Berlin, Institute of Computer Science, 2007.