Application of Combinatorial Optimization Techniques in Genomic Median Problems

Maryam Haghighi

Thesis submitted to the Faculty of Graduate and Postdoctoral Studies
in partial fulfillment of the requirements for the degree of Doctorate in Mathematics[1]

Department of Mathematics and Statistics
Faculty of Science
University of Ottawa

[1]The Doctorate in Mathematics program is a joint program with Carleton University, administered by the Ottawa-Carleton Institute of Mathematics and Statistics

Abstract

Constructing the genomic median of several given genomes is crucial in developing evolutionary trees, since the genomic median provides an estimate for the ordering of the genes in a common ancestor of the given genomes. This is due to the fact that the content of DNA molecules is often similar, but the difference is mainly in the order in which the genes appear in various genomes. The mutations that affect this ordering are called genome rearrangements, and many structural differences between genomes can be studied using genome rearrangements.

In this thesis our main focus is on applying combinatorial optimization techniques to genomic median problems, with particular emphasis on the breakpoint distance as a measure of the difference between two genomes. We will study different variations of the breakpoint median problem from signed to unsigned, unichromosomal to multichromosomal, and linear to circular to mixed.

We show how these median problems can be formulated in terms of problems in combinatorial optimization, and take advantage of well-known combinatorial optimization techniques and apply these powerful methods to study various median problems.

Some of these median problems are polynomial and many are NP-hard. We find efficient algorithms and approximation methods for median problems based on well-known combinatorial optimization structures. The focus is on algorithmic and combinatorial aspects of genomic medians, and how they can be utilized to obtain optimal median solutions.

Dedicated to my mother and father, and to my husband.

Acknowledgments

Foremost, I owe my deepest gratitude to Dr. Sylvia Boyd for her supervision and guidance throughout my PhD. It is extremely difficult not to be motivated by her enthusiasm for research. Her expertise and continuous encouragement have been a tremendous help in completing my PhD.

It gives me great pleasure in acknowledging the support of Dr. David Sankoff; I learned a variety of topics in comparative genomics from him, and his feedback and resourcefulness contributed greatly to my work.

I thank all my friends and colleagues at the University of Ottawa, Department of Mathematics, SITE, and Dr. Sankoff's laboratory for their support. It has been a pleasure to work with them in such a positive and stimulating environment.

I am grateful for the support I received from various faculty members and administrative staff of the Department of Mathematics and Statistics at the University of Ottawa.

I am indebted to my family, my mother and my father, and my in-laws. I would not have been able to complete this chapter of my life without their continuous support and unconditional care and love.

Finally, I would like to thank my husband, Derek. This thesis would have remained a dream had it not been for his encouragement throughout writing it, and for his understanding, love, and patience throughout our life together.

Contents

1 Introduction and Background **1**

1.1 Introduction 1

1.2 Background 7

1.2.1 Genome rearrangement operations and distance measures . . . 8

1.2.2 Breakpoint distance 11

1.2.3 Phylogenetic analysis 13

1.2.4 Notes on complexity 17

1.2.5 A brief Introduction to the traveling salesman problem 18

1.2.6 Median analysis using combinatorial optimization 22

1.2.7 Other genomic distances 25

1.2.8 Median problems under other distances 33

1.2.9 Relations between various median problems 35

1.2.10 Implementations 37

1.2.11 Data 39

2 Linear Multichromosomal Breakpoint Median Problem **48**

2.1 Introduction 50

2.2 Definitions and Background 52

2.3 Transforming the multichromosomal median problem to mTSP 55

2.3.1 Median problem for unsigned genomes 55

2.3.2 Median problem for signed genomes 64
2.4 Computational experiments 67
2.4.1 Unichromosomal breakpoint median problem 69
2.5 Conclusions 70

3 Mixed and Circular Multichromosomal Breakpoint Median Problem 78

3.1 Introduction 80
3.2 Background 84
3.3 Multichromosomal BMP for circular unsigned genomes 87
3.3.1 Circular unsigned BMP with size restrictions 89
3.4 Multichromosomal BMP for mixed unsigned genomes 89
3.5 Partially signed genomes 92
3.5.1 BMP for circular partially signed genomes 93
3.5.2 BMP for mixed partially signed genomes 95
3.6 Conclusion 98

4 Approximation Algorithms for the Unichromosomal Breakpoint Median Problem 101

4.1 Introduction 103
4.2 A shrinking approach 106
4.2.1 BMP and relationships to TSP 106
4.2.2 Properties of the graph 108
4.2.3 Reducing the problem by shrinking 111
4.3 Improving the solutions obtained by the $(2-\frac{2}{q})$-approximation algorithm 116
4.3.1 A 2-opt style improvement on the $\frac{4}{3}$ approximation algorithm 118
4.3.2 A Lin-Kernighan style improvement 120
4.4 New approximation algorithms 121

4.4.1 A $\frac{7}{3}$ approximation algorithm based on cycle patching 121
4.4.2 A $\frac{5}{3}$-approximation algorithm based on MST 125
4.5 Conclusion 129

5 Heuristic methods for multichromosomal BMP based on detection of conserved groups of genes 134
5.1 Introduction 136
5.2 A heuristic method based on partitions 137
5.2.1 Applying clusters 140
5.3 Application of large deviation principle to genomic rearrangements . . 142
5.4 Conclusions 145

6 Conclusion 149
6.1 Future Work 152
6.1.1 Library of data 153

List of Tables

2.1 Performance for the multichromosomal breakpoint median problem on simulated data sets with a ratio of 30%, averages over 5 runs for each sample type of the simulated data. The optimality gap is calculated as the difference, as a percentage, between the reported median and the lower bound found by Concorde on the value of an optimal solution. . 73

2.2 Our method versus GRAPPA: unichromosomal breakpoint median problem. The results are for simulated data with a ratio of 30%. The times reported are averages over 5 runs for each sample type of the simulated data. 74

List of Figures

1.1 A possible high-level view of the Tree of Life [5]. 14

1.2 An example of a tree based on gene orders for fixed gene content. The vertices represent different genomes. Possible evolutionary rearrangement scenarios are marked on the edges of the tree. 15

1.3 An example of a median 16

1.4 A Hamilton cycle for the graph is shown in bold lines. 19

1.5 Transferring A Breakpoint Median Problem to a TSP instance. 24

2.1 BP-median graph constructed for three circular genomes $A = (1\ 2\ 6\ 5\ 4\ 3)$, $B = (2\ 1\ 4\ 5\ 6\ 3)$, and $C = (1\ 2\ 3\ 4\ 6\ 5)$. A breakpoint median is $M = (1\ 2\ 3\ 6\ 5\ 4)$. 56

2.2 An example of transforming a linear multichromosomal BMP to an rmTSP with one depot. The three genomes are $A = 1\ 2 \mid 3 \mid 4\ 5 \mid 6 \mid 7$, $B = 1\ 5\ 3 \mid 2\ 7 \mid 4\ 6$, and $C = 3\ 7\ 4 \mid 6 \mid 5\ 1\ 2$. The chromosomes in each genome are separated by vertical lines. 58

2.3 Transforming an rmTSP to TSP for r=3, m=5. The original BP-median graph is the graph presented in Figure 2.2 (without the depot). The edges between the depots and the original graph are grouped together based on their endpoint in the original graph. Each group has the same weight written besides the arrow. 61

2.4 An example of transforming a circular unichromosomal signed BMP for 3 genes to a TSP on 9 vertices: the solid unlabeled edges will receive weights according to the adjacencies, and the dotted edges have infinity as their weight. 65

3.1 An example of a perfect 2-matching for a graph of 9 vertices. The solid lines represent the graph and the dotted lines represent the perfect 2-matching. 85

4.1 BP-median graph constructed for three circular genomes $A = (1\ 2\ 6\ 5\ 4\ 3)$, $B = (2\ 1\ 4\ 5\ 6\ 3)$, and $C = (1\ 2\ 3\ 4\ 6\ 5)$. A breakpoint median is $M = (1\ 2\ 3\ 6\ 5\ 4)$. 107

4.2 Possible edge weights at a vertex v of the BP-median graph. The edges of weight 3 are not shown. 109

4.3 A saturated gene g_1 110

4.4 Impossible cases for a saturated gene g_1 in a BP-median graph 111

4.5 An example of the effect of shrinking the 0-edges on a BP-median graph. The three dashed edges with weight 0 are shrunk into one 0-edge. Edges with no weight shown have weight greater than 0. . . . 112

4.6 Case 1: if only uv is used 113

4.7 Case 2: if uvw is not used 114

4.8 An example of a median 117

4.9 An example of a 2-interchange: the edges ab and cd (in bold) are replaced with the edges ad and bc (in dashes). 119

4.10 cycle-patching 123

5.1 An example of three clusters with a max gap of 2. The target genes are shown as solid circles. 142

Chapter 1

Introduction and Background

1.1 Introduction

The area of comparative genomics has grown rapidly over the past few years as part of the interdisciplinary field of computational molecular biology. Part of this growth is due to the amount of data available through advances in technologies for extracting information from DNA. Generally speaking, comparative genomics deals with finding similarities and differences between genomes of different species and their evolutionary relationships, with the main goal of achieving a better understanding of the evolution of life on Earth.

Normally, phylogenetic trees are used to illustrate a possible evolutionary history of a set of species. Many of the methods that are applied in constructing these evolutionary trees are based on comparative genomics.

Genome rearrangement theory is an area of comparative genomics where the building blocks are genes and the structures of interest are chromosomes, abstracted in terms of a string of genes they contain. More precisely, a genome is all of the hereditary (or genetic) information of an organism which is usually encoded in the DNA (or, for some viruses, in RNA). The genome includes sections called genes that

may have a specified function in the organism. Most genomes are broken up into organized segments called chromosomes. Each chromosome includes several genes. If a genome contains only one chromosome, it is called unichromosomal. If it contains more than one chromosome, it is called multichromosomal. A chromosome can be circular (where the ordering of the genes in the chromosome follows a circular pattern) or linear (where there are two endpoints to the chromosome). In theory, we can have circular genomes, where all chromosomes are circular, linear genomes, where all chromosomes are linear, or mixed genomes, where some chromosomes are linear and some are circular (although mixed genomes are short-lived). Each gene on the DNA can have two orientations, going in the opposite directions. When the orientation of a gene in the chromosome is known, the gene is called a signed gene. When this orientation is unknown, the gene is called unsigned. If all the genes in a genome are signed (or unsigned), the genome is called signed (or unsigned). In practice, for most genomes, the direction of some genes are known and some are not. Therefore, we can have a partially-signed genome, where both signed and unsigned genes are allowed in the same genome. In reality, there may be several copies of each gene present in a genome. In this research, we only allow one copy of each gene to be present in each genome.

In the past few years, it has been shown that many problems in comparative genomics can be formulated as problems in combinatorial optimization. These methods assist in designing accurate and efficient frameworks to handle a huge amount of collected experimental data, and to extrapolate meaningful conclusions from the data.

In this research, our focus is on applying methods from combinatorial optimization to solve some of the problems that arise in comparative genomics and genome rearrangements. This allows us to take advantage of the knowledge, powerful algorithms and techniques, and the software available within the field of combinatorial optimization and apply them to our problems. We will focus mainly on the combi-

natorial aspects of this field, and will not study the biological details. Also, there has been a significant amount of work done on statistical studies related to genome rearrangement problems. However, these statistical studies are not part of our focus.

Several problems in comparative genomics can be formulated as combinatorial optimization problems. However, for the most part, we concentrate on a problem called the median problem. A main goal of comparative genomics is the reconstruction of the evolutionary events that explain the relations and the differences between several species. These relationships are often described through phylogenetic or evolutionary trees. Inferring evolutionary trees requires reconstructing a common ancestor of several given genomes. One way to model this common ancestor is using the notion of a median. The underlying assumption for a median is that the evolution occurs in the most parsimonious manner, i.e. the common ancestor represented by the median is such that it is the overall closest genome to all of its children. More precisely, a median of k given genomes is a genome such that the total distance between the median and the other k genomes is minimized. The problem of finding such a median of k genomes is called the k-median problem. Depending on the measure used for distance between two genomes, we can obtain different median problems. Some common distance measures include breakpoint, reversal, double cut-and-join, etc.. In this thesis, we work mainly with the breakpoint distance as it has the advantages of an easy computation along with intuitive interpretation and connections to combinatorial optimization problems. Moreover, breakpoint distance provides bounds on other distance measures, and thus the results found using the breakpoint distance and the constructed methods could also be applied to other distance measures.

There are some studies that have compared the medians constructed under different distance measures. These studies suggest that having a fair comparison between medians constructed by breakpoint versus other distance measures can be challenging, as there is no conclusive analytic framework for measuring the superiority of a median. Moreover, it has been shown that the biases in median reconstruction under

breakpoint and other distance measures, are of comparable size (see [2] for further details).

In this thesis, special emphasis is given to breakpoint median problems and different assumptions and formulations using combinatorial structures. As will be discussed, some of these problems are NP-hard, and thus it is considered highly unlikely that efficient algorithms will ever be found for them. Nevertheless, methods that can find close to optimal solutions, in the form of heuristics and approximation algorithms, are highly desired for large-scale problems on real-world data.

In particular, it has been shown that the breakpoint median problem is NP-hard for both signed and unsigned unichromosomal genomes ([14] and [45]). An interesting transformation of this NP-hard problem into a well-known problem from combinatorial optimization called the Traveling Salesman Problem is presented in [50]. The best approximation algorithm known for the k-median problem under breakpoint distance is by Caprara ([17]) where a $\frac{5}{3}$-approximation algorithm for the unichromosomal breakpoint k-median problem is given. For the multichromosomal breakpoint median problem, Tannier, Zheng and Sankoff ([58]) showed that the problem is NP-hard for linear genomes, and is polynomial for signed circular and mixed genomes.

The major contributions of this dissertation are as follows:

1) We provide the first mathematical framework that can handle both signed and unsigned variations of the NP-hard problem of multichromosomal breakpoint median for linear genomes, with demonstrated efficiency in an empirical study. In particular, we are able to solve the multichromosomal breakpoint median problem for genomes of 25,000 genes in less than 20 minutes with our method, whereas previous methods (which can mainly handle unichromosomal cases) are only able to solve problems with a few hundred genes (discussed in Chapter 2).

2) We present the first in-depth study of the multichromosomal breakpoint median problem for unsigned circular and mixed genomes, along with an algorithmic proof of the polynomial-time complexity for these problems. Previously, the com-

plexity of these problems was unknown (discussed in Chapter 3).

3) We present a first study of the breakpoint median problem for partially signed genomes. Despite the fact that it is quite natural to have this type of data from biological experiments, this type of problem has never been studied before. Moreover, we present an algorithmic proof that the breakpoint median problem for partially signed genomes for circular and mixed chromosomes is polynomial (discussed in Chapter 3).

4) We obtain novel approximation algorithms for the NP-hard problem of finding the breakpoint median for unichromosomal genomes, with improved worst-case performance ratios over Caprara's $\frac{5}{3}$-approximation in certain cases. We also provide a shrinking method that allows us to potentially greatly reduce the size of the input data for this problem (discussed in Chapter 4).

5) We study how clustering methods can be combined with different approaches to the median problem to improve the obtained median solution. We will propose partitioning methods that can be combined with various algorithms for finding the median, such as those discussed previously, and those presented in this thesis.

In more detail, the structure of this thesis is as follows:

The remainder of Chapter 1 contains the background survey and literature review of various genomic distances. We also provide detailed proofs for some of the facts that are commonly used in this field without actual formal proofs. This chapter gives a general flavor of the type of problems that exist with some of the known connections to combinatorial optimization. We will also study the relationship between the median problems under different distance measures, and obtain bounds on these problems. Since for the most part our focus in this thesis is on the breakpoint distance, we will provide the essential background for the breakpoint distance and breakpoint median required for the following chapters. Separately, we will provide some general background and literature review for other genomic distances and median problems, which will not be directly required for the upcoming chapters. Nonetheless, we believe

that an overview of other rearrangement distances and their related problems may assist in offering a more global picture of the position of breakpoint distance and breakpoint median, the focus of this dissertation, among other popular problems in the field.

Chapters 2, 3, 4, and 5 of this thesis consist of stand-alone papers. In the paper in Chapter 2, we study the breakpoint median problem for the multichromosomal linear genomes. We provide a new mathematical framework for this NP-hard problem. Our algorithm is the first formulation of its kind which provides us with a powerful technique to find very good solutions for large-scale multichromosomal breakpoint median problems. To test the effectiveness, accuracy, and speed of our approach, we perform an empirical study of this method on simulated and real-world data.

To obtain our method for the multichromosomal breakpoint median problem, we extend a known method for the unichromosomal breakpoint median problem. As mentioned, [50] provided a transformation of the unichromosomal breakpoint median problem into a well-known problem in combinatorial optimization known as the Traveling Salesman Problem (TSP). We extend this approach and provide a transformation of the multichromosomal breakpoint median problem into a multiple salesman TSP. By utilizing this transformation, we will provide new insight to the underlying graph and the structure of an optimal median corresponding to an optimal TSP tour. We then apply the information that we have discovered to construct our algorithm for the NP-hard problem of finding the multichromosomal breakpoint median. Next we take advantage of a well-known and powerful TSP solver called Concorde, and utilize it in solving our median problem. We demonstrate the usefulness of our method by testing it on simulated and real-world data, with a comparison to another available median solver.

Next, in the paper in Chapter 3, we study the multichromosomal breakpoint median problem for mixed and circular genomes and show that these are polynomial problems. To the best of our knowledge, this is the first comprehensive study of

the multichromosomal breakpoint median problem for mixed and circular unsigned genomes (only the signed version of these problems has been studied previously in [58]). We provide transformations from these problems into another well-known combinatorial optimization problem called a 2-matching. We also introduce the concept of partially signed genomes and illustrate how our methods for unsigned or signed genomes can be modified to handle partially signed genomes by using b-matchings.

After studying the multichromosomal case, we shift our focus to the breakpoint median problem for unichromosomal genomes. This is the more traditionally studied version of the median problem. In the paper in Chapter 4, we present an in-depth study of the properties of the unichromosomal breakpoint median problem and provide new approximation and heuristic algorithms for this problem. The first two algorithms provided are based on known tour improvement methods for TSP. Next, we apply other methods such as a cycle patching approach, and an approximation algorithm based on minimum spanning trees.

Finally, in Chapter 5, we present some ideas on how the approximation and heuristic algorithms presented in Chapters 2 and 3 can be further modified to suit additional requirements. In this chapter, we investigate how heuristic methods for breakpoint median problem can be combined with concepts from graph partitioning and clustering methods. We also present an idea to provide links between construction of phylogenies and probability theory, namely asymptotic behaviors of Dyck paths.

We conclude this dissertation with Chapter 6, in which we present a global summary of the results of previous chapters. In addition, we present some ideas and recommendations for further research and future work on this topic.

1.2 Background

This section contains the background survey and literature review of various genomic distances and their associated median problems. We also provide detailed proofs for

some of the facts that are commonly used in this field without actual formal proofs. Our primary focus in this thesis is on the breakpoint distance. Therefore, we start with the essential background for the breakpoint distance measure and breakpoint median problem that is required for the following chapters. We will also briefly review some concepts from combinatorial optimization and complexity theory which will serve as a basis for the rest of this thesis. This essential background is explained in Sections 1.2.1 through 1.2.6.

Separately, we will provide some general background and literature review for other genomic distances and median problems in Sections 1.2.7 through 1.2.11. These sections are not directly required for the upcoming chapters. Nonetheless, we believe that an overview of other rearrangement distances and their related problems may assist in offering an overall view of the position of breakpoint median problems among other median problems.

1.2.1 Genome rearrangement operations and distance measures

In 1938, Dobzhansky and Strutevant [57] published an article where they presented an evolutionary tree for Drosophila (fruit fly) with a rearrangement scenario using 17 reversals. Later on, it has been shown that genome rearrangement is a contributing factor in the evolution of different forms of life. For example, a comparison of mitochondrial genomes of cabbage and turnip revealed that almost 99% of their genes are identical [44]. The pioneer work of [57] was performed by eye, due to the small size of the data. Nowadays, with more complex data available, we need to have systematic and automated methods to be able to perform such analysis on large data sets. One way to compare the difference between a pair of genomes is to use some difference measure or distance. Here we will explain how a set of integers can be used to provide a mathematical model for such problems.

A genome containing n (unsigned) genes can be viewed as an ordered string of numbers $1, ..., n$. In this analogy, the possibility of duplication and circularity is not captured, however there are also models that consider duplication, etc. in the gene content. Since in the double-stranded DNA of a genome, some genes are located in one strand and are read in the direction of that strand, while others might be on the opposite strand and therefore in the opposite direction, it is very common to consider a polarity for the genes; hence we have signed genomes on the set $-1, +1, -2, +2, ..., -n, +n$ where each absolute value k of the genes in each signed genome only appears once (no repetition). From now on, we only consider unichromosomal, linear genomes, unless otherwise stated. Also, since results for signed versus unsigned genomes may vary in different measures, we will mention which of these are considered in each section. For example, $3, 2, 5, 1, 4$ is a representation of an unsigned genome on 5 genes, whereas, $2, -1, 5, 3, -4$ is an example of a signed genome on 5 genes. It is common not to write the "+" sign before genes in a signed genome.

During evolution, a DNA molecule replicates itself; however, there are usually inaccuracies in this process which are normally referred to as *mutations*. There are two types of mutations. The first type is at the level of nucleotides (building blocks of DNA) where a nucleotide may get deleted, inserted, or replaced by another nucleotide. Studying these events is the subject of *sequence alignment*. The second type of evolution might occur as a result of modification to the organization of the DNA sequence at a larger scale, i.e. operations that act on a genome at the level of DNA segments rather than nucleotides. During these operations a section of the sequence may get deleted, replaced, inverted, duplicated, etc.. These events are referred to as *rearrangements* and genome rearrangement studies these operations.

In general, the goal of genome rearrangement problems is to find a shortest set of events (operations) that transforms one genome to another. Different genome rearrangement problems consider various events, and different genomes. Also, finding a shortest sequence usually refers to finding a sequence with the least number of

events, however, there are variations of this problem in which a weight is assigned to each operation, and therefore, shortest would refer to minimum weight in that case.

Consider two genomes and an optimal (shortest) sequence of operations that transforms one genome to another. The length of such an optimal sequence is called the *distance* between the two genomes under that operation. The distance is often metric. A distance $d : S \times S \to R \;:\; (s,t) \mapsto r$ is *metric* if the following three conditions are satisfied:

1. For all $s,t \in S$, $d(s,t) \geq 0$ and $d(s,t) = 0$ if and only if $s = t$.
2. For all $s,t \in S$, $d(s,t) = d(t,s)$ (symmetry).
3. For all distinct $s,t,u \in S$, $d(s,t) \leq d(s,u) + d(u,t)$ (triangular inequality).

The models throughout this survey are considered deletion and duplication-free, i.e. each gene appears exactly once in the sequence. Therefore, a genome on n genes can be represented by a permutation on integers $1 \ldots n$. As usual, a permutation π is considered to be a bijection over $\{1, 2, \ldots, n\}$. Let π_i denote the image of i under the permutation π. Then, the permutation π can be denoted by $\begin{pmatrix} 1 & 2 & \cdots & n \\ \pi_1 & \pi_2 & \cdots & \pi_n \end{pmatrix}$.

Traditionally, in genome rearrangements only the second row is kept, i.e.

$\pi \quad = \quad (\pi_1 \pi_2 \ldots \pi_n)$.

Define a *composition* of two permutations π and σ, denoted by $\pi \circ \sigma$, by first applying σ and then applying π. For example, if $\pi = (4\ 2\ 1\ 3)$ and $\sigma = (1\ 2\ 4\ 3)$, then $\pi \circ \sigma = (4\ 2\ 3\ 1)$. The *identity* permutation is $\imath = (1\ 2\ \ldots n)$, and the *inverse* permutation π^{-1} of π is the permutation such that $\pi \circ \pi^{-1} = \imath$.

Without loss of generality, we can focus only on the distance of a permutation (genome) from the identity permutation. Consider two permutations π and σ. Then the distance between π and σ is equal to the distance between $\sigma^{-1} \circ \pi$ and the identity permutation. The problem of finding the distance from the identity is often referred to as *sorting*. Throughout this survey, whenever there is only one permutation mentioned

as the variable for a distance, we are assuming that the distance is considered from the identity.

1.2.2 Breakpoint distance

Breakpoint distance is based on comparing the two sequences of genes (not considering the orientation) of two genomes. Consider two unsigned genomes $A = a_1, ..., a_n$ and $B = b_1, ..., b_n$ on the same set of genes. The genes a_i and a_{i+1} are called *adjacent*. If two genes g and h are adjacent in A but not in B, then they determine a *breakpoint* in A (or B). The *breakpoint distance* between two genomes A and B, denoted by $bp(A, B)$, is equal to the number of breakpoints in A (or B). For example, if $A = 3\ 1\ 5\ 2\ 4$ and $B = 1\ 3\ 5\ 4\ 2$, then there are two breakpoints, one breakpoint between 1 and 5, and another breakpoint between 5 and 2. Therefore the breakpoint distance is equal to 2. Although the discussion in this section focuses on the breakpoint distance for unsigned genomes, note that similar definitions are valid for the breakpoint distance for signed genomes, and since all of the notations and results are valid for both cases, we will only focus on the unsigned case.

The breakpoint distance has the advantage that it can be computed very easily compared to other types of genomic distance measures. However, no operation is associated with the breakpoint distance. It has been mentioned in several references that the breakpoint distance is metric; however, we could not find any formal proof for it. Here we will present a short proof.

Lemma 1. *Breakpoint distance is metric.*

Proof. Let A and B be two genomes. By definition, breakpoint distance is non-negative and $bp(A, B) = 0$ if and only if $A = B$.

Also, clearly the number of breakpoints of A in comparison with B, is equal to the number of breakpoints of B in comparison with A. Therefore breakpoint distance is symmetric.

We need to show that breakpoint distance satisfies the triangular inequality. Without loss of generality, consider three genomes A, B, and C, of the same length on the same set of genes, and consider any pair of genes x and y. We have the following cases:

1) x, y are adjacent in A, B and C.

2) x, y are adjacent in both A and B, but not in C.

3) x, y are adjacent in A and in C, but not in B.

4) x, y are adjacent in B and in C, but not in A.

5) x, y are adjacent in A, but not in B and C.

6) x, y are adjacent in B, but not in A and C.

7) x, y are adjacent in C, but not in A and B.

8) x, y are not adjacent in A, B and C.

The table below shows the contribution to the breakpoint distances by the pair x, y for each of these cases.

Case	Contribution to bp(A,B)+bp(B,C)	Contribution to bp(A,C)
1	0	0
2	1	1
3	2	0
4	1	1
5	1	1
6	2	0
7	1	1
8	0	0

As can be seen from the table, the triangular inequality

$$bp(A, B) \ + \ bp(B, C) \geq bp(A, C)$$

is always valid. Thus, considering over all pairs of genes we see that the triangular inequality would be true, and therefore breakpoint distance is metric. □

1.2.3 Phylogenetic analysis

A *phylogeny* or an *evolutionary tree* for a set of N genomes is a tree with N leaf nodes such that each leaf node corresponds to one genome. Ideally, we want to construct a "correct" phylogeny for a given set of leaf nodes. A phylogeny is *correct* if the tree structure is same as the real-world evolutionary history. However, in practice, the real-world evolutionary history is usually unknown or only partially-known. Therefore, the goal is to find methods that can construct such a tree from a given set of data of the leaf nodes, with as little error as possible. A common assumption is that changes in gene order are a good indication for speciation, and therefore gene order can be used to construct the evolutionary history of species. In this assumption, for a given set of genomes, we wish to find their common ancestral genomes in the most parsimonious manner, that is by using the least number of rearrangement operations [46]. Given a set of genomes and the distance between each pair, the goal is to construct an evolutionary tree with minimum cost. The cost here is defined as the sum of the cost of the edges, where the cost of each edge is the pair-wise distance between its two endpoints [23].

This is based on assuming that chromosomal phylogenetic trees follow the evolution of species. While this is a common assumption, it should be noted that there is no conclusive evidence to confirm it. Nonetheless, building phylogenetic trees based on rearrangement analysis helps in providing information about the evolution of various species.

There is a close relationship between phylogenetic analysis and the Steiner tree problem. The *Steiner tree problem* on a subset U of the vertices of a graph is to find a minimum-cost tree T such that it contains the vertices U. The phylogenetic version

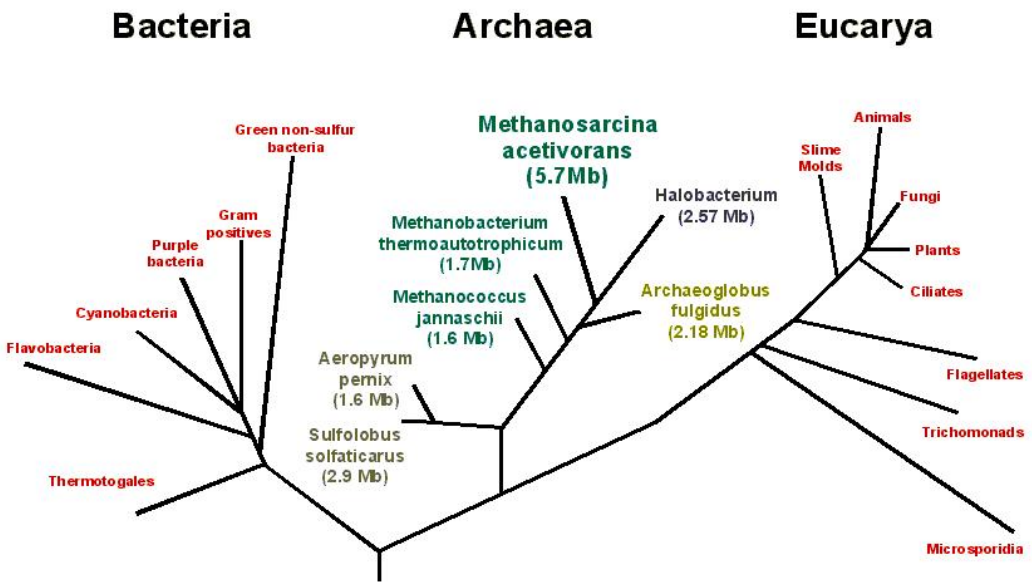


Figure 1.1: A possible high-level view of the Tree of Life [5].

of the Steiner tree problem can be defined as follows: Given a fixed phylogeny (tree) T and a set of permutations on n integers, such that each permutation corresponds to a leaf node of T, find a set of permutations corresponding to the internal nodes of T, such that the total weight of the tree is minimized. The total weight of a tree is $w(T) = \sum d(x, y)$ where $d(x, y)$ is the pair-wise distance between the two endpoints of each edge of T [55].

The Median problem

Constructing phylogenies can be very difficult, and therefore, restricted versions of the problem are often studied. One of the most common restrictions is trying to find one common ancestor of several given genomes. This problem is known as the *median problem* [54]. We focus on the median problem and present some of the major results for the median problem under different distance measures.

It is common to consider the median problem for three genomes, since its so-

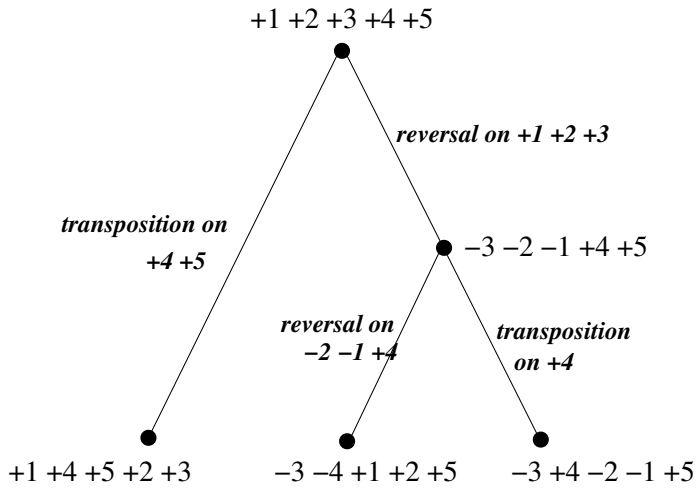


Figure 1.2: An example of a tree based on gene orders for fixed gene content. The vertices represent different genomes. Possible evolutionary rearrangement scenarios are marked on the edges of the tree.

lution can easily be generalized to the median problem of any size. Given three genomes G_1, G_2, and G_3, and a rearrangement distance d, the *3-median problem* is the problem of finding a genome M, called the *median*, such that the sum of distances between M and each other genome is minimized. The *median score* $S(M)$ is defined as $\sum_{i=1}^{3} d(M, G_i)$.

A common method used to reconstruct the evolutionary history of a given set is to use the solution to the median problem many times by conducting a Steinerization-based heuristic. In such a heuristic method, the median of three vertices is found, and then the median itself is used as one of the vertices along with two other vertices, to find a new median, over and over. Clearly, the quality of the solution depends on the choices made for the vertices, and the number of calls for finding the median. These variations lead to different heuristics for constructing phylogenies ([52] and [18]).

The following lemma mentioned in [28] gives a universal bound on the optimal median score, though no proof is given. Here we will present a simple proof for it.

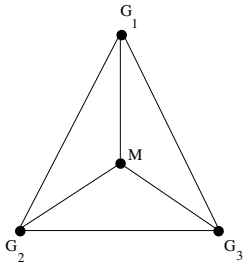


Figure 1.3: An example of a median

Lemma 2. *Let $S(M)$ be the median score for the median M of three given genomes G_1, G_2, and G_3, and a rearrangement distance d. Then*

$S(M) \geq \frac{d(G_1,G_2)+d(G_2,G_3)+d(G_1,G_3)}{2}$.

Proof. Using the triangular inequality,

$d(G_1, G_2) + d(G_2, G_3) + d(G_1, G_3) \leq$

$d(M, G_1) + d(M, G_2) + d(M, G_2) + d(M, G_3) + d(M, G_1) + d(M, G_3)$

$= 2S(M).$ □

It should be noted that the above lemma is only valid if the distance measures are metric. We have shown that the breakpoint distance is metric. It is widely believed that all other genomic distances such as the reversal distance and the DCJ are also metric, however, we were not able to find any mathematical proof of this fact.

The *breakpoint median problem*, introduced by Sankoff and Blanchette in 1997 ([50]) considered breakpoint as the distance measure for computing the median. They gave a reduction of this problem to the Traveling Salesman Problem, which we will study in detail in Section 1.2.6 . The problem is NP-hard for signed and unsigned permutations for genomes consisting of a single chromosome ([45] and [14]). In [46] Pe'er and Shamir give a $\frac{7}{6}$-approximation for signed genomes when finding the median of 3 genomes. In [17] Caprara gives a $\frac{5}{3}$-approximation for the unsigned case when finding the median of k genomes, $k \geq 3$.

1.2.4 Notes on complexity

In this section we present some definitions and concepts from computational complexity theory. For further details on this subject we refer the reader to [30].

A *decision problem* is a problem where the answer is yes or no. Any optimization problem with a numerical answer can be efficiently transformed into a decision problem. A decision problem is called a *polynomial-time* problem if it can be solved by an algorithm which takes an amount of time which is a polynomial function of the size of the problem instance. A decision problem is called *non-deterministic polynomial* (NP) if a yes answer to it can be verified in polynomial time by a non-deterministic Turing machine. Given two decision problems A and B, we say that A *is polynomial-time reducible to* B (written by $A \propto B$) if any instance X of A can be transformed in polynomial-time into an instance Y of B, such that the answer to X is yes if and only if the answer to Y is yes. A decision problem B is *NP-complete* if it is in the set of NP problems, and for every other problem A in the set of NP problems, we have $A \propto B$.

Note that a decision problem R that is in the set of NP problems can be proved to be NP-complete by showing $R \propto A$ for some known NP-complete problem A. Therefore, the class of NP-complete problems can all be transformed to each other, and solving any one of them is equivalent to solving all of them. It is widely believed that it is highly unlikely that a polynomial-time solution for the class of NP-complete problems exists. Nevertheless, many real-world problems are NP-complete and thus we need to find polynomial-time algorithms for finding near-optimal solutions for these NP-complete problems.

A combinatorial optimization problem R (a minimization or a maximization problem with a finite set of candidate solutions and solution values) has an *optimal* solution c^* if the value of the objective function for c^* is less than or equal in the case of minimization (or more than or equal in the case of maximization) to the value of

the objective function for any other feasible solution to R.

An algorithm A is a *k-approximation algorithm* for a minimization problem R for some constant $k \geq 1$ if for any given instance of the problem it finds a solution S where the value of the problem for the solution S is less than or equal to k times the value of the optimal solution to the problem. The number k is called the *performance guarantee* of the algorithm A.

If a problem is NP-complete, it is often desirable to find a polynomial-time k-approximation algorithm for it. The closer the value of k to 1, the better the approximation algorithm is in terms of its performance guarantee. It should be noted that the performance guarantee is a worst-case scenario, i.e. in general, a k-approximation algorithm may perform better in many instances and give solutions that are closer than k to the optimal value.

1.2.5 A brief Introduction to the traveling salesman problem

In this section we focus on some well-known problems from graph theory and combinatorial optimization and provide an introductory description of these problems which would be sufficient for the use in this research. Let $G = (V, E)$ be a graph with a set V of vertices and a set E of edges. A *Hamilton path* (or a Hamiltonian path) is a path of all vertices of the graph that contains each vertex exactly once. A *Hamilton cycle* (or a Hamiltonian cycle) is a cycle that contains each vertex exactly once except for the first vertex, which is also the last in the cycle. For example, in Figure 1.4 a Hamilton cycle is shown in bold lines. Obviously, not every graph has a Hamilton path or a Hamilton cycle. Also, notice that there may be more than one Hamilton cycle (or path) in a given graph. It is easy to see that a complete graph contains a Hamilton cycle. Hamilton path or cycle is named after the Irish mathematician, Sir William Rowan Hamilton who first developed a puzzle based on it in 1859 [32].

Suppose a salesman must visit n cities where each city is connected to all of

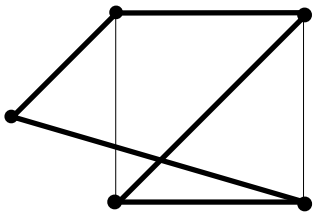


Figure 1.4: A Hamilton cycle for the graph is shown in bold lines.

the other cities by a road. The salesman must start from his home, visit all other cities exactly once and return to where he started. To minimize travel time, it is ideal to find a shortest route (each route can have a value assigned to it based on the cost of travel, distance between the two cities, etc.). This is an example of a *Traveling Salesman Problem* (TSP). An instance of this problem can be represented by a complete weighted graph where the vertices represent the cities, and there is an edge between two vertices if and only if there is a route between their corresponding cities. The weight of the edge represents the cost of taking the route between its endpoints. An *optimal tour* or an optimal solution to the Traveling Salesman Problem is a minimum weight Hamilton cycle in the graph. Clearly, the starting vertex is irrelevant to the length of an optimal tour. One can think of solving a TSP by considering all possible tours and finding the one with the minimum weight, starting from any vertex there are $n-1$ possible vertices for the second vertex, $n-3$ for the third vertex and so on. It is easy to see that in general there are $\frac{(n-1)!}{2}$ possible tours to check. Therefore, for large values of n a brute force method of checking all possibilities is not a realistic method. Rather than enumerating all possibilities, many algorithms for solving the TSP problem eliminate many of the possible tours without considering them explicitly.

From a computational standpoint, TSP belongs to the class of combinatorial optimization problems known as NP-complete. If one can find a polynomial-time algorithm (an algorithm that will be guaranteed to find an optimal solution in a

polynomial number of steps) for an NP-complete problem, then such algorithm exists for all other problems belonging to the NP-complete class of problems. To date, however, no one has found a polynomial-time algorithm for TSP. In fact, it has been conjectured that such algorithm does not exist. Although this theoretical question remains open, there has been much effort put into discovering efficient ways for finding near-optimal solutions for the TSP, making TSP to be one of the most intensely studied problems in combinatorial optimization. Many discrete optimization problems can be formulated using TSP, and therefore, studying TSP has turned to a general platform for finding methods that can be applied to these other problems. Aside from its theoretical importance, several real-world problems can be formulated by using TSP. Some examples include drilling circuit boards, applications in communication networks, various scheduling and route assignment problems.

Similar to many other optimization problems, generating a lower and an upper bound on the optimal value of a TSP instance can help to reduce the size of a search tree. Clearly, the total weight of any Hamilton cycle cannot be less than the optimal solution to the TSP. Therefore, constructing a feasible solution (a Hamilton cycle) can provide an upper bound for the optimal solution. There are several heuristic methods that provide such upper bounds. However, in general, there is no guarantee as to how close an upper bound is to the optimal solution. There are several heuristic algorithms for TSP that have been shown to find near-optimal solutions in many instances within reasonable time.

As with other optimization problems, a lower bound on the optimal value of a solution can be used. The main advantage of finding a lower bound is that if the lower and upper bounds become equal, optimality has been reached. Otherwise, the difference between the upper and the lower bound can be used to provide a measure of error for the heuristic solution. Lower bounds can be obtained by relaxations of the TSP, i.e. removing some of the constraints in order to obtain another problem (relaxation) which is easier to solve and its feasible solutions contain all feasible

solutions to the original problem (and hence containing the optimal solution as well). There are many different ways to find a relaxation of a TSP. Once a relaxation of a TSP is found and thus a lower bound is obtained, various techniques are used to tighten the difference between the lower and the upper bound, and therefore, get closer and closer to the optimal solution. One of the most common techniques is the cutting plane technique embedded in branch-and-cut algorithms. Loosely speaking, a branch-and-cut method finds a constraint that is violated by a relaxation and therefore it reduces the size of the polytope containing the optimal solution by "cutting" it through that plane.

Next, we will briefly introduce Concorde, a state of the art package for solving TSP. Concorde is a TSP solver written in ANSI C by David Applegate, Robert E. Bixby, Vasek Chvatal, and William J. Cook, and is freely available for academic use [3]. Concorde's TSP solver has been applied to obtain solutions for several large and difficult instances of the TSP, and is widely viewed as the best and fastest TSP solver currently available. Concorde's goal is solving symmetric TSP based on the branch-and-cut method. Concorde has been applied to many TSP problems on real-world data. In 2004, Concorde was used to find the optimal solution for the TSP of visiting all 24,978 cities in Sweden.

As explained before, finding the exact solution for a TSP involves by checking all possible tours is not a realistic method. To deal with large instances of the TSP and many other combinatorial optimization problems, linear programming (LP) methods have proven to be useful. Introduced by Dantzig, Fulkerson and Johnson in 1954, this method considers a relaxation of the LP for the original problem (that is the TSP, in our case). By using the inequalities of the relaxation and finding its solution at each step, we can then iteratively tighten the inequalities to obtain solutions that are closer to the solution for the TSP. This well-known approach is often referred to as the *cutting plane* method. A general cutting plane method starts by finding an optimal solution for the relaxation of the original LP. If this solution satisfies

the original problem, we have reached an optimal solution. Otherwise, there is an inequality in the original LP which is violated by this solution. This inequality is called a cut (or a cutting plane, since the space of the feasible solutions is contained in a polyhedron and each inequality represents a plane in the space). We can add this cut to the system to obtain a tighter relaxation of the original LP. Next, we repeat the same process by starting with this tighter relaxation. If we reach a step where a new cutting plane or a significant improvement to the objective function cannot be found, the problem can then get branched into two smaller minimization problems to be solved separately. This branching produces a binary tree (similar to that of the branch-and-bound methods). Various methods can be applied to reduce the number of branches that need to be considered. This is the well-known branch-and-cut method. Notice that the choice of the cut at each step can make a difference in the size of the tree and the number of steps involved in finding the optimal solution. Various methods have been developed to produce "good" cuts. Concorde is a state of the art implementation combining many of the most recent developments in this field for solving TSP. It applies branch-and-cut in an efficient way to obtain exact solutions. Where an optimal solution cannot be reached in reasonable time, Concorde reports a near-optimal solution (the best found so far). It also provides a bound on the worst case difference between the reported solution and the optimal solution.

1.2.6 Median analysis using combinatorial optimization

We will review two methods as examples of how combinatorial optimization techniques can be applied to solve median problems.

TSP formulation of breakpoint median

It can be shown that the breakpoint median problem is a special case of the traveling salesman problem [50]. Here we describe how the breakpoint median problem can

be reduced to the Traveling Salesman Problem (TSP) in the case of three unsigned genomes A, B and C over a set of n genes, each consisting of one circular chromosome.

Let G be a complete graph of n vertices such that each vertex represents one gene. For each edge uv let $adj(uv)$ be equal to the number of times the genes corresponding to u and v are adjacent (do not form a breakpoint) in genomes A, B, and C (so $adj(uv)$ can be 0,1,2, or 3). Let the weight of the edge uv, $w(uv)$, be equal to $3 - adj(uv)$. Then the solution to the TSP for the weighted graph G traces out a permutation of $\{1, 2, \ldots, n\}$ that provides an optimal solution to the breakpoint 3-Median problem. We consider all four possible cases for the weight of the edge uv.

1. $w(uv) = 0$, which means that u and v are adjacent in all three genomes. Therefore, using this edge will cost 0 for the lowest cost TSP solution.

2. $w(uv) = 1$, which means that u and v are adjacent in two out of the three genomes.

3. $w(uv) = 2$, which means that u and v are adjacent in only one genome.

4. $w(uv) = 3$, which means that u and v are not adjacent in any genome. This case gives the highest cost edge, and therefore, the optimal TSP solution would try to avoid choosing such an edge.

It is shown in [50] that, starting at any vertex of a cycle that is a TSP solution for the problem, we will have a genome that is the breakpoint median for the given three genomes. To better clarify this concept, we provide a simple example.

Suppose we want to find the breakpoint median of the following three genomes: $A = (1\ 2\ 3\ 4\ 5\ 6\ 7)$, $B = (4\ 3\ 5\ 1\ 2\ 7\ 6)$, and $C = (5\ 7\ 1\ 2\ 3\ 4\ 6)$.

We first calculate the value of $adj(uv)$ for all of the adjacent pairs of genes in these three genomes. We have

$adj(12) = 3$, $adj(23) = 2$, $adj(34) = 3$, $adj(45) = 1$, $adj(56) = 2$, $adj(67) = 2$, $adj(35) = 1$, $adj(51) = 1$, $adj(27) = 1$, $adj(17) = 2$, $adj(46) = 2$, and $adj(57) = 1$. The value of $adj(uv)$ for any other pair u and v is zero.

Then by using $w(uv) = 3 - adj(uv)$ we can find the weight of each edge. Figure

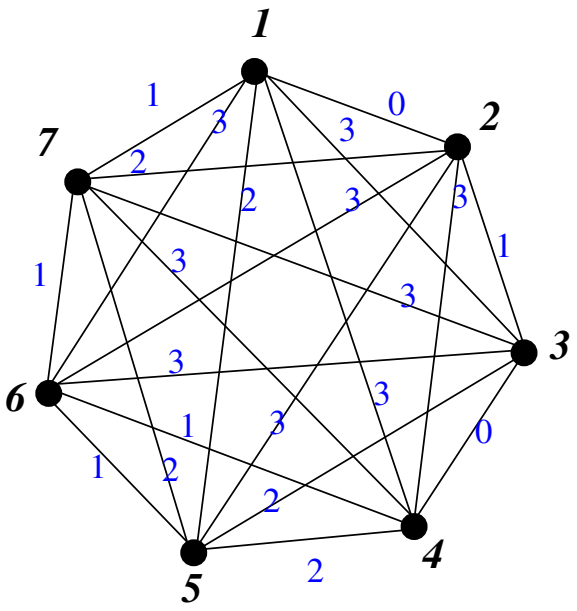


Figure 1.5: Transferring A Breakpoint Median Problem to a TSP instance.

1.5 shows the weighted graph of this problem.

As can be seen from the example, a TSP solution is 1234567 with the minimum cost equal to 6. Therefore, a breakpoint median for this problem is 1234567. Note that in this particular example the median happens to be one of the three genomes. As can be seen in this example, the TSP defined for the breakpoint median problem is not a metric TSP, that is, the triangular inequality is not valid for the weights of the edges in TSP (for example, look at the edges between vertices 1, 2, and 7).

Using matchings to obtain an approximation for the breakpoint median

Pe'er and Shamir in [46] give a fairly simple approximation for the breakpoint median problem on signed genomes. We will present a brief description of their method in this section.

They start by providing a $\frac{6}{5}$-approximation, and then improving it to obtain a $\frac{7}{6}$-approximation algorithm for the breakpoint median problem of three unsigned

unichromosomal genomes.

The $\frac{6}{5}$-approximation algorithm given in [46] is based on finding matchings on a graph. They first present a method to transfer a signed permutation on n integers to an unsigned permutation on $2n+2$ integers, where each adjacent pair of unsigned integers $(2i-1, 2i)$ represent a positive integer $+i$, and each pair of unsigned integers $(2i, 2i-1)$ represent a signed integer $-i$. Let V be set of $2n$ vertices corresponding to $2n$ integers. Next, define a *base matching* $M_b = \{(v_{2i}, v_{2i-1})\}_{i=0}^{i=n-1}$ on V (subscripts for the vertices are modulo $2n$). A perfect matching M on V is called a *Hamiltonian matching* with respect to the base matching M_b if and only if $M_b \cup M$ forms a Hamilton cycle (a cycle that visits every vertex exactly once). Pe'er and Shamir construct a Hamiltonian matching M_i corresponding to each of the three given genomes. Then, they show that a median corresponds to a Hamiltonian matching M such that the weight of M defined by $w(M) = \sum_i |M \setminus M_i|$ is minimized. They provide a $\frac{6}{5}$-approximation algorithm based on minimizing the weight of the corresponding matching.

After giving the $\frac{6}{5}$-approximation, Pe'er and Shamir ([46]) improve the ratio by a long case-by-case analysis and give a $\frac{7}{6}$-approximation for the breakpoint median problem of three signed genomes.

1.2.7 Other genomic distances

Although the research in this thesis deals solely with breakpoint distance, for completeness, we also describe other genomic distances in this section, and the related results.

Transposition distance

Transposition is an operation that moves a segment of a genome to another location. Transpositions are sometimes called translocations or insertions (note that the name

of this operation has nothing to do with algebraic transpositions). Since the result of a transposition is independent of whether a genome is signed or unsigned, we will only consider the case for the unsigned genomes.

The problem of sorting by transpositions was first introduced by Bafna and Pevzner in 1998 [7]. The complexity of this problem is shown to be NP-hard [15] and the best approximation solution known so far is due to Elias and Hartman [26] with ratio $\frac{11}{8}$. Also the maximum transposition distance lies between $\lfloor \frac{n+1}{2} \rfloor$ (Bafna and Pevzner [7]) and $\frac{2n}{3}$ (Eriksson et al. [27]).

We start by obtaining bounds for transposition distance based on breakpoints. A transposition can decrease the number of breakpoints by at most three [7]. For example

$(1 \ \ldots \ i-1 \quad j \quad j+1 \ \ldots \ k-1 \quad i \quad i+1 \ \ldots \ j-2 \quad j-1 \quad k \ \ldots \ n)$

can be transformed to the following by one transposition:

$(1 \ \ldots \ i-1 \quad i \quad i+1 \ \ldots \ j-2 \quad j-1 \quad j \quad j+1 \ \ldots \ k-1 \quad k \ \ldots \ n)$

Therefore we have the following bound:

$$bp(\pi) \leq 3 \times td(\pi).$$

There are two types of breakpoints: *ascents* and *descents*. Ascents are breakpoints such that $\pi_{i+1} > \pi_i$ and descents are breakpoints such that $\pi_{i+1} < \pi_i$. Let $des(\pi)$ denote the number of descents in a permutation π. Each transposition can decrease the number of descents by at most 2. Therefore, we have a lower bound of $\frac{des(\pi)}{2}$ for the transposition distance. However in this case the order matters, and therefore the lower bound as shown in [27] is

$$td(\pi) \geq max\left(\frac{des(\pi)}{2}, \frac{des(\pi^{-1})}{2}\right).$$

The identity permutation is the only permutation with no breakpoint. It is only natural to think that a sorting strategy could be to "repair" the breakpoints while

keeping the adjacencies. Christie [22] showed that for any permutation π there exists an optimal sorting sequence by transpositions such that all the adjacencies of π are preserved.

The above bounds are all lower bounds for transposition distance. Bafna and Pevzner [7] provided an upper bound for transposition distance $td(\pi) \leq \frac{3}{4}bp(\pi)$. Later, Eriksson et al. [27] improved this upper bound to the following:

$$td(\pi) \leq \begin{cases} \lceil \frac{2}{3}bp(\pi) \rceil & if \quad n < 9 \\ \lfloor \frac{2bp(\pi)-2}{3} \rfloor & if \quad n \geq 9. \end{cases}$$

In 1998, Bafna and Pevzner [7] gave the first polynomial-time approximation for sorting by transpositions with an approximation ration of $\frac{3}{2}$. Following their method, alternative simpler approximation algorithms were found but they all had the same $\frac{3}{2}$ ratio (for example see [22]). Later, Elias and Hartman [26] improved the approximation ratio to $\frac{11}{8}$ by using a huge case by case analysis of over 80,000 configurations.

Reversal distance

A *reversal* or an *inversion*, denoted by $\rho(i, j)$, occurs when the segment of the genome between gene i and gene j is reversed and the DNA strands are exchanged. Reversals were the first genome rearrangement operations proposed in biology by Dobzhansky and Sturtevant in 1938 [57]. Because of its biological importance, reversal is one of the most studied operations in this field. As usual, *reversal distance* refers to the minimum number of reversals needed to transform one genome to another. *Sorting by reversals* is the problem of finding a shortest sequence of reversals that transforms a genome to the identity permutation. Caprara [18] showed that sorting by reversals for unsigned genomes is NP-hard. However, sorting by reversals for signed genomes has been shown to be polynomial [33].

First, we look at the unsigned reversal distance and describe some bounds for it based on breakpoints. Let $rd(\pi)$ denote the reversal distance of a permutation π (from the identity). The following result is mentioned as an observation by Bafna and Pevzner in [6], though no detailed proof is given. Here we will provide a proof for this result.

Lemma 3. *$bp(\pi) \leq 2 \times rd(\pi)$.*

Proof. Consider genome $\imath = A = g_1\, g_2 \ldots g_n$, and a reversal occurring between g_i and g_j. After the reversal we have $g_1 \ldots g_{i-1}\, g_j\, g_{j-1} \ldots g_{i+1}\, g_i\, g_{j+1} \ldots g_n = B_1$.

If we compare A and B_1 there are two breakpoints formed at $g_{i-1}g_j$ and g_ig_{j+1}. Now consider a reversal on genome B_1. There are three cases: the new reversal is completely independent from the previous one i.e. they do not share an end, or the new reversal shares one end with the previous one, or the new reversal has both ends shared with the previous one. Observe that in case 1, two new breakpoints are formed. In case 2, one new breakpoint is formed, and in case 3, no new breakpoint is formed. Therefore after each reversal, at most two new breakpoints are formed.

By induction, $bp(\pi) \leq 2 \times rd(\pi)$.

□

In 1995 Kececioglu and Sankoff [36] used matchings to obtain lower bounds for the reversal distance. Let $G = (V, E)$ be the graph whose vertex set V is the set of breakpoints of π. There is an edge between two vertices of V if and only if their corresponding breakpoints can be eliminated by one reversal. Let m be the number of vertices in a maximum cardinality matching of G. Then, it is shown in [36] that

$$rd(\pi) \geq \left\lceil \frac{m}{2} + \frac{2(bp(\pi) - m)}{3} \right\rceil.$$

The authors of [36] also presented the following upper bound for the unsigned reversal distance:

$$rd(\pi) \leq bp(\pi) - 1.$$

There are various approximations and exact algorithms for computing the unsigned reversal distance. Kececioglu and Sankoff [36] developed two algorithms for the problem of reversal distances; a greedy approximation algorithm, that finds a solution close to optimal in $O(n^2)$ time for n-element permutations, and a branch-and-bound exact algorithm, that finds an optimal solution in $O(mL(n,n))$, where m is the size of the branch-and-bound search tree, and $L(n,n)$ is the time to solve a linear program of n variables and n constraints. The greedy algorithm guarantees a solution that uses no more than twice the minimum number of reversals. The lower and upper bounds of the branch-and-bound algorithm use maximum-weight matchings, shortest paths, and linear programming.

Bafna and Pevzner in [6] have improved the approximation performance ratio to $\frac{7}{4}$ for the case of unsigned permutations.

Later, Berman et al. [10] were able to obtain an algorithm with the approximation ratio of $\frac{11}{8}$.

Some interesting variations of the reversal distance problem that have been studied include:

- The block interchange distance: introduced by Christie (1996) [21]. The distance can be computed in $O(n)$, and the sorting problem can be computed in $O(nlogn)$.

- Prefix reversals: introduced by Goodman (1975) [25], in prefix reversals the only valid reversals are those acting only on the beginning of the permutation, i.e. $\rho(1,j)$ for all j. Complexity of prefix reversal distance is unknown, however, there is a 2-approximation algorithm by Fischer and Ginzinger (2005) [29].

- Weighted reversals: introduced by Pinter and Skiena (2002) [47], this operation

assigns a weight to each reversal depending on the likelihood of it happening based on biological arguments. In such models, short reversals are given smaller weights since they are more likely to happen in a genome, compared with long reversals. Therefore, in such problems the objective is to minimize the cost function based on the weighted distance. The complexity of this problem is unknown, however, there are various approximations available based on different values of distance assigned [8].

- Fixed-length reversals: introduced by Chen and Skiena (1996) [20], only considers sorting by reversals with a given fixed length. The complexity of this problem is unknown.

- Bounded reversals: introduced by Heath and Vergara (2003) [34], only considers reversals with lengths less than or equal to a given fixed value k. In the case of $k = 3$ the problem is referred to as short swaps. The complexity of this problem is unknown, however, for short swaps there exists a 2-approximation algorithm [34].

Next, we look into the problem of finding reversal distance for signed genes. A signed reversal $\rho(i, j)$ applied to a permutation π for $1 \leq i \leq j \leq n$ reverses the order and the sign of the genes between i and j. For example a reversal on genes $i = +2$ and $j = +5$ of the genome $+1, +2, -3, -4, +5, -6$ reverses the order of the genes (and the signs) that are between these two positions, i.e. $+1, -5, +4, +3, -2, -6$. Hannenhalli and Pevzner in 1999 [33] gave a polynomial-time algorithm for solving the problem of sorting signed permutations by reversals. Later, there have been several improvements to this solution. Almost all of these improvements take advantage of a graph structure commonly referred to as the breakpoint graph for signed permutations (see [28] for further details).

Bafna and Pevzner in [6] considered the problem of reversals of signed permutations and used a method (similar to what they used for the unsigned version) that

guarantees a performance ratio of $\frac{3}{2}$.

Hannenhalli and Pevzner [33] showed that the signed problem is of polynomial complexity. They also found a linear-time algorithm for sorting signed reversals.

Double Cut-and-Join distance

The *Double Cut-and-Join* (DCJ) operation was first proposed by Yancopoulos et al. [61] in 2005. This operation gives a unifying view of other rearrangement operations such as reversals, and transpositions. Each DCJ operation can be viewed as several reversals and transpositions. The general idea of a DCJ is to cut the genome in two places, and join the pieces that are created together. The mathematical simplicity of formulating DCJ which leads to a polynomial complexity for calculating DCJ distance, has made DCJ to be a very popular operation to study. DCJ is generally defined on signed genomes. If a genome is unsigned, we can assume all positive signs for its genes. Since the introduction of DCJ in 2005, different ways of expressing it have been published which all lead to the same result.

Bergeron et al. gave a nice mathematical formulation of DCJ. They introduced a bipartite graph, called the *adjacency graph*, for two signed genomes A and B such that the vertices are the set of adjacencies and telomeres of A and B, and two edges share a vertex if they have a common head or tail in their corresponding adjacencies or telomeres. They showed that each DCJ operation can change the number of odd paths in the adjacency graph by 2, 0, or -2, and thus were able to prove that the DCJ distance between A and B is equal to $n - C + \frac{I}{2}$, where n is the number of genes, C is the number of cycles in the bipartite adjacency graph, and I is the number of odd paths [9]. By using this result, they provided a greedy algorithm that gives an optimal sequence of DCJ operations to transfer a genome A to another genome B in $O(|A| + |B|)$.

Combined operations

In the real-world, several operations will happen on a genome. Therefore, from a biological point of view it is desired to study combinations of different operations. Compared to studying single operations, combined operations are more difficult and less studied. The most common combination considered is transpositions and reversals.

We start by studying some combined rearrangement operations defined for unsigned genomes. In 1998 Walter et al. [59] introduced the problem of computing the genomic distance when reversals and transpositions are allowed together. Although the complexity of this problem is not known, they used a simple method to obtain a 3-approximation for the problem based on lower bounds on the maximum number of breakpoints that can be removed by a reversal, or by a transposition. Rahman et al. [48] were able to give a $(4 - \frac{2}{k})$-approximation, where k is the approximation ratio of the algorithm used for finding a maximal decomposition of a breakpoint graph.

Next, we review combined operations for signed genomes. Studying reversals and transpositions for signed genome as combined operations was first mentioned in [49]. Later, Blanchette et al. [11] developed a software called DERANGE which is based on providing parameters for the weights of reversals and transpositions, based on probability of their occurrence. According to their experiments, the most logical weights are 1 for reversals and 2 for transpositions when compared to real-world data. Since then, several variations of these parameters have been studied. Although the complexity of the problem is not known, approximation algorithms exist for different variations of the parameters. For example, Walter et al. [59] gives a 2-approximation algorithm when the weight of both reversals and transpositions is equal to 1.

Comparison chart

The following chart provides a summary of the results found so far on different genomic distances (for linear unichromosomal case).

Operation	**Distance**	**Best Approx For Distance**	**Sorting**	**Best Approx for Sorting**
Breakpoint	Polynomial	$O(n)$	-	-
Transposition	NP-hard	unknown	NP-hard	$\frac{11}{8}$
Reversal Unsigned	NP-hard	unknown	NP-hard	$\frac{11}{8}$
Reversal Signed	Polynomial	$O(n)$	Polynomial	$o(n^{\frac{3}{2}})$
DCJ	Polynomial	$O(n)$	Polynomial	$O(n)$

1.2.8 Median problems under other distances

Although the research in this thesis only deals with the median problem for breakpoint distance, for completeness we provide here some background for the median problem using other distance functions.

Reversal median

The reversal median problem has been a popular subject for research, although only the signed version has been studied. One reason could be that as mentioned in the previous chapter, signed reversal distance can be computed in polynomial time, whereas the unsigned case is NP-hard. Therefore, one might assume that the signed reversal median problem can be easier to investigate. However, the signed reversal median problem is NP-hard and APX-hard ([18]). A problem is APX-hard if every APX problem (i.e. an optimization problem that has a constant-factor approximation) can

be reduced to it. If a problem is APX-hard, then it is considered highly unlikely we will find a polynomial-time approximation algorithm for the problem, for every fixed constant factor.

Caprara [16] gives a $\frac{4}{3}$-approximation for the reversal median problem. He also proposes an integer programming formulation for the reversal median problem, and an exact solution obtained by a branch-and-bound method which only works well for small instances. Generally, if the number of genes is larger than 50 most exact algorithms fail [28], and therefore, several heuristics have been studied. Most of these heuristic methods follow a greedy strategy of taking one of the given genomes as the median, and then trying to modify it to bring it closer to the real median by reducing its distance from the other genomes (for example see [12] and [35]).

DCJ median

The problem of finding the median based on DCJ distance is more recent compared to the reversal median and the breakpoint median. However, since DCJ distance provides a lower bound for the reversal distance, the bounds and methods for DCJ median can also be used for the reversal median problem [28]. The DCJ median problem is proven to be NP-hard and APX-hard for the unichromosomal case.

A $\frac{4}{3}$-approximation for the reversal median problem given by Caprara [16] also acts as a $\frac{4}{3}$-approximation for the DCJ median problem. In fact, Caprara uses the NP-hardness of the DCJ median problem and the fact that it is a lower bound for the reversal median problem, in order to prove the NP-hardness of the reversal median problem (although, the terminology for DCJ and the specific introduction of this operation did not exist at that time and was introduced years later by Yancopoulos et al. [61]). All of these NP-hardness proofs take advantage of reducing the median problem to a breakpoint graph decomposition.

Xu and Sankoff [60] use a decomposition to a multiple breakpoint graph (MBG)

(a multiple breakpoint graph can be viewed as an extension of several copies of the breakpoint graph joined together through their "common" vertices) and search for a vertex matching in that graph that maximizes the number of the alternating color cycles. They give a list of the so-called adequate subgraphs that assists in rapid decomposition of a MBG into smaller subgraphs for which the median can be found faster. This provides a considerable speedup in finding the median, although finding the adequate subgraphs of larger sizes can be a difficult task on its own.

Comparison chart

The following chart provides a summary of the results found so far on the median for different genomic distances (for linear unichromosomal case).

Operation	**Median**	**Best Approximation**
Breakpoint-signed	NP-hard	$\frac{7}{6}$
Breakpoint-unsigned	NP-hard	$\frac{5}{3}$
Transposition	unknown	unknown
Reversal-unsigned	unknown	unknown
Reversal-signed	NP-hard	4/3
DCJ	NP-hard	4/3

1.2.9 Relations between various median problems

Using different distance measures, it is possible to obtain relationships between their corresponding median problems. In this section we show how breakpoint median problem (BMP) can be applied to find lower and upper bounds for other median problems, such as the famous NP-hard problem of finding the median under reversal distance. Recall that in Lemma 3 we proved the following result on breakpoint (bp) and reversal (rd) distances: $bp(\pi) \leq 2 \times rd(\pi)$.

Also, an upper bound for the unsigned reversal distance presented in [36] is $rd(\pi) \leq bp(\pi) - 1$. Combining these bounds we have

$$\frac{1}{2}bp(\pi) \leq rd(\pi) \leq bp(\pi) - 1. \tag{1.1}$$

Note that both bounds in (1.1) can be tight; as illustrated by the example $\pi =$ 1 2 4 3 5, where $bp(\pi) = 2$ and $rd(\pi) = 1$.

We also provided a proof for the following lower bound for *median score* $S(M) = \sum_{i=1}^{3} d(M, G_i)$ where $d(M, G_i)$ is the distance between the median M and the genome G_i.

$$S(M) \geq \frac{d(G_1, G_2) + d(G_2, G_3) + d(G_1, G_3)}{2}. \tag{1.2}$$

Since Inequality 1.2 is valid for both breakpoint and reversal distances, by combining it with inequalities (1.1) we have the following:

$$\frac{1}{2}S_{bp}(M) \leq S_{rd}(M) \leq S_{bp}(M) - 1, \tag{1.3}$$

where $S_{bp}(M)$ and $S_{rd}(M)$ are the median scores obtained by using the breakpoint distance and the reversal distance, respectively. This relationship provides an effective tool for obtaining lower and upper bounds for the reversal median score from the breakpoint median score.

Several studies have been performed on the expected number of rearrangement operations between two random genomes. In [53], the authors showed that for two random genomes of length n, the expected reversal distance is $n - O(\frac{1}{2}\log n)$. Similarly, it can showed that the expected number of breakpoints between two random genomes is $\log n$ and therefore, the expected breakpoint distance between two random genome of length n is $n - \log n$. Therefore, we can claim that the gaps in inequalities (1.3) can not be too large for large enough genomes. Hence, inequalities (1.3) provide good lower and upper bounds to the reversal median problem. These bounds can be

effectively utilized to improve the performance of other methods that find the reversal median (e.g. Siepel [56] gave an exact algorithm for the reversal median problem which is based on finding lower and upper bounds for the reversal median).

1.2.10 Implementations

There are various software packages designed for solving some of the problems discussed in previous chapters. Here we provide a short list of the most popular packages along with their main purpose.

- BPAnalysis, 1997 ([13]): BreakPoint Analysis (BPAnalysis) [50] is designed to calculate the breakpoint distance and the breakpoint median. BPAnalysis uses an iterative heuristic method that calculates all search trees and their leaf nodes, and within each iteration solves an instance of a TSP at an internal node, until convergence. The approach works well in finding solutions for not too large data sets, however, due to the NP-hardness of TSP, can be slow for larger inputs.

- GRAPPA, 2001 ([31]): Genome Rearrangement Analysis under Parsimony and other Phylogenetic Algorithms (GRAPPA) is a modification of BPAnalysis that focuses on breakpoint and reversal distance calculation and finding breakpoint and reversal medians by linear time algorithms (for the signed case). It produces similar results to BPAnalysis in the case of breakpoints, however due to better data structure and some algorithm engineering and parallel running, it has a higher speed than BPAnalysis.

- MGR, 2002 ([40]): Multiple Genome Rearrangement (MGR) is a tool written by Guillaume Bourque for constructing phylogenies based on reversals. It focuses on finding the reversal median for both unichromosomal and multichromosomal cases, however, only certain types of reversals, those that bring one genome closer to the other two, are considered in this method [12]. Although

theoretically this may not be the case for a median in general, the authors in [12] argue that MGR-Median performs well on simulated data and outperforms GRAPPA after ratio 0.2 for the simulation. MGR can be applied for finding reversal median for unichromosomal circular, unichromosomal linear (signed) and multichromosomal (linear unsigned) genomes. Also, an improved version of MGR has been developed and available on a web server called webMGR [38].

- MedRByLS, 2007 ([39]): Median Reversal By Local Search is based on the work in [35], and finds the median of a set of genomes by starting at an arbitrary one and performing a series of reversals in a greedy method (with a certain probability) to find the ancestral genome. After a pre-defined number of steps, the program outputs the best median found so far.

- MedITas, 2008: Median Solver by Iterated Tabu Search (MedITas), based on the work presented in [37], is an improvement to MedRByLS such that the local search is enhanced by a tabu search. This method produced results that are as good as the ones produced by MedRByLs for simulated data, and in some real-world instances tested, MedITas significantly outperformed MedRByLS and even found new solutions to previously unsolved cases.

- ASMedian,2008 ([4]): Adequate Subgraph Median (ASMedian), based on the results in [60], finds the DCJ median. The package can exactly solve the DCJ median problem by using a set of adequate subgraphs to reduce the search tree of a branch-and-bound method and with taking advantage of different matchings available in the breakpoint graph. The package is designed for both unichromosomal and multichromosomal case.

1.2.11 Data

Simulations

In general, most of the methods for calculating genomic distance, finding medians, or performance of various algorithms related to these areas are tested on simulated data, as real-world data required for these studies is generally harder to find, contain errors, or is in the progress of being developed. Therefore, simulation plays an important role for performance measurement and analysis.

Generating a set of random data is essential for simulated studies. A typical set of genes for data simulation generated in previous studies consists of set of integers with sizes ranging from 50 to 400 genes (see for example [23], [43],[60]). Also, for generating the genomic operations which occurred on the set of genes, the common assumption is an occurrence of 4 to 40 genomic operations. There are various assumptions for generating random sets of genes. A common assumption is to sample from a uniform distribution. Another standard model is the Nadeau-Taylor model [43] for genome evolution which considers a Poisson distribution for the number of operations that occurred on a genome.

Using randomly-generated data has the advantage that any of the proposed algorithms and methods can be tested over a set of data with minimal efforts required in obtaining the data set. However, an important disadvantage of simulation is that like any other simulations, the algorithms and methods may fail to perform as good on real-world data as they do on simulated data. Nonetheless, simulation continues to be an important source for testing algorithms.

A popular method in simulating genomic operations is using a ratio r of the number of operations over the total number of genes (or markers). Using this ratio has the advantage that one can eliminate the effect of the size of the sample on the results. However, by using a ratio, a complete comparison of the performance of algorithms in terms of their running-time might be harder to obtain, specially for

NP-hard problems.

Real-world data

Due to the disadvantages of simulated data, it is often desired to test the methods and algorithms on real-world data. However, obtaining real-world data can be challenging. Different factors that contribute to this problem include limited/restricted access to data obtained from genomic laboratories, copyright issues, error occurrence due to laboratory experiments, and having incomplete/partial/ambiguous data which gets improved or changes as laboratory experiments are conducted.

So far, different methods for calculating genomic distance, or finding medians have been tested on data from some mammals, as well as some plants, and bacteria. Some wide-range studies include human-cow-cat comparison (for this and other comparisons see [41]), human-mouse-rat comparison in [24], and human-cat-ferret comparison [19]. Other studies use a particular set of real-world data and find the median score, or calculate the distance from the found median to the real ancestor (if known). It is also common to build or modify a particular phylogeny based on the results obtained from median calculations. For example, [1] uses the information drawn from [42] to find a phylogeny for a mammalian data set by using DCJ operations.

Bibliography

[1] Adam, Z., and Sankoff, D.: The ABCs of MGR with DCJ, Evolutionary Bioinformatics Journal, 4: 6974, 2008.

[2] Adam, Z., and Sankoff, D.: A statistically fair comparison of ancestral genome reconstructions based on breakpoint and rearrangement distances, Journal of Computational Biology, 17(9): 1299-1314, 2010.

[3] Applegate, D., Bixby, R., Chvatal, V., and Cook, W.: Concorde TSP Solver, www.tsp.gatech.edu/concorde

[4] ASMedian Homepage: http://sites.google.com/site/andrewweixu/Home/software

[5] A high level view of tree of life: http://www.broadinstitute.org/

[6] Bafna, V., and Pevzner, P.A.: Genome rearrangements and sorting by reversals, Proceedings of the 34th Symposium on Foundations of Computer Science, 148-157, 1993.

[7] Bafna, V., and Pevzner, P.A.: Sorting by transpositions, SIAM Journal on Discrete Mathematics, 11:224-240, 1998.

[8] Bender, M.A., Ge, D., He, S., Hu, H., Pinter, R., Skiena, S., and Swidan, F.: Improved bounds on sorting by length-weighted reversals, Journal of Computer and System Sciences, 74(5):744-774, 2008.

[9] Bergeron, A., Mixtacki, J., and Stoye, J.: A unifying view of genome rearrangements, Proceedings of the Sixth International Workshop on Algorithms in Bioinformatics (WABI), Lecture Notes in Computer Science, 4175:163-173, 2005.

[10] Berman, P., Hannenhalli, S., and Karpinski, M.: 1.375-approximation algorithm for sorting by reversals, Proceedings of the Tenth Annual European Symposium on Algorithms, Lecture Notes in Computer Science, 2461:200-210, 2002.

[11] Blanchette, M., kunisawa, T., and Sankoff, D.: Parametric genome rearrangement, Gene, 172:GC11-GC17, 1996.

[12] Bourque, G., and Pevzner, P.A.: Genome-scale evolution: reconstructing gene orders in ancestral species, Genome Research, 12:26-36, 2002. http://grimm.ucsd.edu/MGR/

[13] BPAnalysis Homepage: http://www.mcb.mcgill.ca/ blanchem/software.html

[14] Bryant, D.: The complexity of the breakpoint median problem, technical report CRM-2579, Centre de Recherches de Mathematiques, Universite de Montreal, 1998.

[15] Bulteau,L., Fertin,G., and Rusu, I.: Sorting by Transpositions Is Difficult, Lecture Notes in Computer Science, Proceedings of International Colloquium on Automata Languages and Programming(ICALP), (1):654-665, 2011.

[16] Caprara, A.: Formulations and hardness of multiple sorting by reversals, Proceedings of the Third Annual International Conference on Computational Molecular Biology (RECOMB), S. Istrail, P. Pevzner, and M. Waterman eds., ACM, 84,93, 1999.

[17] Caprara, A.: Additive bounding, worst-case analysis and the breakpoint median problem, SIAM Journal on Optimization, 13:508-519, 2002.

[18] Caprara, A.: The reversal median problem, INFORMS Journal on Computing, 15 (1):93-113, 2003.

[19] Cavagna, P., Menotti, A., and Stanyon, R.: Genomic homology of the domestic ferret with cats and humans, Journal of Mammalian Genome, Springer 11 (10) 866-870, 2000.

[20] Chen, T., and Skiena, S.: Sorting with fixed-length reversals, Discrete Applied Mathematics, 71:269-295, 1996.

[21] Christie, D.A.: Sorting permutations by block-interchanges, Information Processing Letters, 60:165-169, 1996.

[22] Christie, D.A.: Genome rearrangement problems, PhD Thesis, University of Glaskow, Scotland, 1998.

[23] Cosner, M.E., Jansen, R.K., Moret, B.M.E., Raubeson, L.A., Wang, L., Warnow, T., and Wyman S.: A new fast heuristic for computing the breakpoint phylogeny and experimental phylogenetic analyses of real and synthetic data, ISMB-00 Proceedings, 2000.

[24] Denwey, C., and Pachter, L.: Genome sequencing of the brown Norway rat yields insights into mammalian evolution, Rat Genome Sequencing Consortium, Nature, 428:493-521, 2004.

[25] Dweighter, H. (Goodman, J.E.): Elementary problems and solutions, problem E2569, American Mathematical Monthly, 82(1):1010, 1975.

[26] Elias, I., and Hartman, T.: A 1.375-approximation algorithm for sorting by transpositions, IEEE/ACM Transactions on Computational Biology and Bioinformatics, 3(4):369-379, 2006.

[27] Eriksson, H., Eriksson, K., Karlander, J., Svensson, L., and Wastlund, J.: Sorting a bridge hand, Discrete Mathematics 241:289-300, 2001.

[28] Fertin, G., Labarre, A., Rusu, I., Tannier, E., and Vialette, S.: Combinatorics of genome rearrangements, The MIT Press, 2009.

[29] Fischer, J., and Ginzinger, S.W.: A 2-approximation algorithm for sorting by prefix reversals, Proceedings of the Fourth Workshop on Algorithms in Bioinformatics (WABI), I.Jonassen and J. Kim eds., Lecture Notes in Computer Science, 3669:415-425, 2005.

[30] Garey, M., and Johnson, D.: Computers and interactibility - A guide to the theory of NP-completeness, W.H. Freeman and company, 1979.

[31] GRAPPA Homepage: http://www.cs.unm.edu/ moret/GRAPPA

[32] Gutin, G., and Punnen, A.P. (eds.): The traveling salesman problem and its variations, Kluwer, Dordrecht, 2002.

[33] Hannenhalli, S., and Pevzner, P.: Transforming cabbage into turnip (polynomial algorithm for sorting signed permutations by reversals), Proceedings of 27th Annual ACM Symposium on the Theory of Computing: 178-187, 1995.

[34] Heath, L.S., and Vergara, J.P.C.: Sorting by short swaps, Journal of Computational Biology, 10(2):775-789, 2003.

[35] Interian, Y., and Durret, R.: Genomic midpoints: computation and evolutionary implications,www.math.cornell.edu/ durrett/mdpt/midpoint07.pdf, 2005.

[36] Kececioglu, J.D., Sankoff, D.: Exact and approximation algorithms for sorting by reversals, with application to genome rearrangement, Algorithmica 13(1/2): 180-210, 1995.

[37] Lenne, R., Solnon, C.,Stutzle, T., Tannier, E., and Birattari, M.: Reactive stochastic local search algorithms for the genomic median problem, Proceedings of the Eighth European Conference on Evolutionary Computation in Combinatorial Optimization (EVOCOP), Lecture Notes in Computer Science, 4972:266-276, 2008.

[38] Lin, C.H., Zhao, H., Lowcay,S.H., Shahab,A., and Bourque, G.: webMGR: an online tool for multiple genome rearrangement, Bioinformatics, 26(3):408-410, 2010.

[39] MedRByLS Homepage: http://www.cam.cornell.edu/ interian/codes.html

[40] MGR Homepage: http://grimm.ucsd.edu/MGR/

[41] Murphy. W.J, Frnicke, L., O'Brien, S.J., and Stanyon, R.: The origin of human chromosome 1 and its homologs in placental mammals, Genome Research 13: 1880-1888, 2003.

[42] Murphy W.J., Larkin D.M., Wind A.E., Bourque G., Tesler G., Auvil L., Beever J.E., Chowdhary B.P., Galibert F., Gatzke L., Hitte C., Meyers S.N., Milan D., Ostrander E.A., Pape G., Parker H.G., Raudsepp T., Rogatcheva M.B., Schook L.B., Skow L.C., Welge M., Womack J.E., OBrien S.J., Pevzner P.A., and Lewin H.A.: Dynamics of Mammalian Chromosome Evolution Inferred from Multispecies Comparative Maps, Science, 309:6137, 2005.

[43] Nadeau J.H., and Taylor B.A.: Lengths of chromosomal segments conserved since divergence of man and mouse, Proceedings of the National Academy of Sciences USA, 81: 814818, 1984.

[44] Palmer, J.D., and Herbon, L.A.: Plant mitochondrial DNA evolves rapidly in structure but slowly in sequence, Journal of Mol. Evolution, 28: 87-97, 1988.

[45] Pe'er, I., and Shamir, R.: The median problems for breakpoints are NP-complete, Electronic Colloquium on Computational Complexity, technical reports 71, 1998.

[46] Pe'er, I., and Shamir, R.: Approximation algorithms for the median problem in the breakpoint model, in Sankoff, D., and Nadeau, J.H. (eds.), Comparative Genomics: 225-241, Kluwer Dordrecht, 2000.

[47] Pinter, R.Y., and Skiena, S.: Genomic sorting with length-weighted reversals, Proceedings of the Thirteenth International Conference on Genome Informatics, 13:103-111, 2002.

[48] Rahman, A., Shatabda, S., and Hasan, M.: Approximation algorithm for sorting by reversals and transpositions, Proceedings of the First Workshop on Algorithms and Computation (WALCOM), 97-108, 2007.

[49] Sankoff, D.: Edit distance for genome comparison based on non-local operations, Proceedings of the Third Annual Symposium on Combinatorial Pattern Matching (CPM), Lecture Notes in Computer Science, 644:122-1135, 1992.

[50] Sankoff, D. and Blanchette, M.: The median problem for breakpoints in comparative genomics, in Proceedings of the Third International Computing and Combinatorics Conference, T.Jiang and D.T.Lee eds, Lecture Notes in Computer Science 1276:251-263, Springer-Verlag, 1997.

[51] Sankoff, D.: Genome rearrangement with gene families, Bioinformatics, 16(11):909-917, 1999.

[52] Sankoff, D. and Blanchette, M.: Multiple genome rearrangement and breakpoint phylogeny, Journal of Computational Biology, 5:555570, 1998.

[53] Sankoff, D., and Haque, L.: The distribution of genomic distance between random genomes, Journal of Computational Biology, 13-5:1005-1012, 2006.

[54] Sankoff, D. and Nadeau, J.H.: Conserved synteny as a measure of genomic distance, Discrete Applied Mathematics, 71:247-257, 1996.

[55] Sankoff, D., Sundaram, G., and Kececioglu, J.: Steiner points in the space of genome rearrangements, International Journal of Foundations of Computer Science, 7:1-9, 1996.

[56] Siepel, A.C.: Exact algorithms for the reversal median problem, Master's thesis, University of New Mexico, Albuquerque, NM, 2001.

[57] Sturtevant, A.H., and Dobzhansky, T.: Inversions in the third chromosome of wild races of Drosophila pseudoobscura, and their use in the study of the history of the species, Proceedings of National Academy of Science of USA, 22:448-450, 1936.

[58] Tannier, E., Zheng, C., and Sankoff, D.: Multichromosomal median and halving problems under different genomic distances, BMC Bioinformatics, 10:120, 2009.

[59] Walter, M.E.M.T., Dias, Z., and Meidanis, J.: Reversal and transposition distance of linear chromosomes, Proceedings of the Fifth International Symposium on String Processing and Information Retrieval (SPIRE), IEEE Computer Society Press, 96-102, 1998.

[60] Xu A.W., and Sankoff, D.: Decompositions of multiple breakpoint graphs and rapid exact solutions to the median problem, Lecture Notes in Bioinformatics, 5251:25-37, Proceedings of the 8th international workshop on Algorithms in Bioinformatics (WABI), 2008.

[61] Yancopoulos S., Attie O., Friedberg R.: Efficient sorting of genomic permutations by translocation, inversion, and block interchange, Bioinformatics, 21:33406, 2005.

Chapter 2

Linear Multichromosomal Breakpoint Median Problem

• Maryam Haghighi and Sylvia Boyd, A Fast Method for Large-scale Multichromosomal Median Problem, Proceedings of the ACM International Conference on Bioinformatics Computational Biology and Biomedicine, BCB 2011, Chicago, Il, USA, August 1-3, 2011, ACM 2011, ISBN 978-1-4503-0796-3.

• Sylvia Boyd and Maryam Haghighi, A Fast Method for Large-scale Multichromosomal Median Problem, Journal of Bioinformatics and Computational Biology, 2011(accepted).

Work of Maryam Haghighi under the supervision of Sylvia Boyd. Mathematical formulations and proofs are joint work of Maryam Haghighi and Sylvia Boyd. Algorithm implementation and empirical work is by Maryam Haghighi.

Abstract

We provide a computationally realistic mathematical framework for the NP-hard problem of the multichromosomal breakpoint median for linear genomes that can be used in constructing phylogenies. A novel approach is provided that can handle both signed and unsigned cases of the multichromosomal breakpoint median problem. Our method provides an avenue for incorporating biological assumptions (whenever available) such as the number of chromosomes in the ancestor, and thus, it can be tailored to obtain a more biologically-relevant picture of the median. We demonstrate the usefulness of our method by performing an empirical study on both simulated and real data with a comparison to other methods.

2.1 Introduction

Constructing phylogenies can be very challenging, and therefore, restricted versions of the problem are often studied. One of the most common restrictions is to find one common ancestor of several given genomes. This problem can be modeled by using the notion of a median, where one searches for a genome that is close to several other genomes. Once we have a method to find the median, we can then use it in an iterative manner to construct the evolutionary tree (see [16] for examples of such construction). A fundamental question in building phylogenies is how far apart two species are from each other. This idea can be captured under the notion of *distance* between two genomes. There are several popular measures for distance, such as breakpoint, reversal, double cut-and-join, etc.. For measuring distance, we focus on breakpoint distance as it provides an intuitive link between the order of genes in each genome and how far two genomes could be from each other. It has been argued that the reversal distance is a more biologically accurate representation of what can happen in nature [5]. However, there are some disadvantages, such as the fact that finding the reversal distance for unsigned genomes is NP-hard, whereas the breakpoint distance can be easily computed in both signed and unsigned cases [10].

In general, eukaryotic cells (cells with nuclei) have several linear chromosomes (as opposed to prokaryotic cells which have circular chromosomes). Most of the research on the median problem involves the simplified unichromosomal case. The multichromosomal median problem is less studied due to its conceived theoretical difficulty compared to the unichromosomal case. However, the linear multichromosomal median problem is the more accurate model for the eukaryotes, which includes all complex organisms. In this paper our focus is on finding solutions to the linear multichromosomal breakpoint median problem (BMP). To the best of our knowledge, our method is the first practical method that provides solutions for both signed and unsigned linear multichromosomal BMP with the ability of considering several

possibilities for the number of chromosomes in the desired solution.

The unichromosomal BMP is NP-hard for signed and unsigned genomes ([6] and [14]). In 2009, Tannier, Zheng and Sankoff ([19]) showed that for the case of multichromosomal genomes where circular and mixed genomes are allowed, the BMP can be solved in polynomial time. They also showed that if we only allow linear chromosomes, the median problem becomes NP-hard. In this paper we focus on the case where only linear chromosomes are allowed, i.e. the NP-hard case. As previously mentioned, the assumption of allowing only linear chromosomes is the relevant case for all eukaryotes. In this analysis, we consider multichromosomal genomes where each gene is present exactly once in every genome. We first define the framework in which we can mathematically represent the median problem. Sankoff and Blanchette [16] gave a reduction of the unichromosomal BMP to another well-known problem, the Traveling Salesman Problem (TSP). In this paper we extend this method and provide a novel approach for a transformation from the multichromosomal BMP to the multiple salesmen TSP. We study the case for both signed and unsigned linear genomes. Then, we apply a second transformation from the multiple salesmen TSP to the usual TSP. The subject of several books in the past few years, TSP is arguably the most intensively studied problem in combinatorial optimization [12]. Therefore, a transformation of multichromosomal BMP to an instance of TSP opens the door to the vast knowledge and tools available for solving TSP which we can apply for finding the median. In particular, we take advantage of a software package called Concorde for solving TSP [2]. We demonstrate the usefulness of our method by presenting the results of an empirical study on both simulated and real-world data, with a comparison to another method.

2.2 Definitions and Background

We first present the definitions for unsigned genomes. An *unsigned gene* g is a sequence of DNA where the orientation is unknown. A set of unsigned genes form an unsigned *genome*. We can represent an unsigned genome on n genes by a string of unsigned integers $1, 2, \ldots, n$ which represents the ordering of the genes in the genome. This string can be broken into segments, representing the *chromosomes* of the genome. Chromosomes can be *circular* or *linear*. Circular chromosomes have a circular gene ordering (which will be represented with brackets around the segment). A linear chromosome has two extremities, called the *telomeres* of the chromosome. For example, $G = (3\ 6\ 10\ 1)\ |\ (2\ 4\ 5)\ |\ (7\ 9\ 8)$ represents an unsigned genome G on 10 genes with three circular chromosomes, where the chromosome are separated from each other by a vertical line. $H = 1\ 2\ 4\ 3$ has one linear chromosome with 4 unsigned genes. Gene 1 and gene 3 are the telomeres of H. If all of the chromosomes in a genome are circular, the genome is called *circular*, and if all of the chromosomes are linear, the genome is called *linear*. If some chromosomes are linear and some are circular, the genome is called *mixed*.

Given an unsigned genome A, we say two genes are *adjacent* in A if they are adjacent in the gene ordering. For example, in G above, 2 and 4 are adjacent, 7 and 8 are adjacent, and 3 and 10 are not adjacent. In H above, 4 and 3 are adjacent, 1 and 3 are not.

Next we present the definitions for signed genomes. In this case, the orientation of each gene is known, and a gene g is a sequence of DNA with two extremities called the tail and the head, denoted by g_t and g_h respectively. A *signed genome* is a sequence of oriented genes. As in the unsigned case, we can represent a signed genome on n genes by a string of integers $1, 2, \ldots, n$ broken into segments representing the chromosomes, where each integer will be given a sign (+ or -) representing the orientation of the gene. In this notation, we let $+g$ represent the gene g in the orientation $g_h g_t$, and we

let $-g$ represents the gene in the orientation $g_t g_h$. For example, consider the following signed genome C with 5 genes:

$$C = (-3 - 4 + 1) \ | \ +5 - 2$$

Genome C has two chromosomes, one is circular and one is linear. We could also represent C by writing each gene as its ordered tail and head extremities:

$$C = (3_t 3_h \ 4_t 4_h \ 1_h 1_t) \ | \ 5_h 5_t \ 2_t 2_h$$

Note that for each linear or circular chromosome, there are two equivalent strings where one is obtained from the other by reversing the order and switching the signs of all the genes.

For signed genomes, the adjacencies are not defined on the genes, but instead on the extremities of the genes, namely the heads and tails of the genes. We say an extremity of gene u and an extremity of gene v are adjacent if they are adjacent in the ordering of the genome. Thus, there are four possible adjacencies between two genes u and v depending on the direction of the genes: $u_h v_h$, $u_h v_t$, $u_t v_h$, or $u_t v_t$. The telomeres are sets that contain one element, i.e. u_h, u_t, v_h or v_t.

For example, in the genomes C above, the adjacencies are 3_t and 1_t, 3_h and 4_t, 4_h and 1_h, and 5_t and 2_t. The telomeres of the genome are the gene extremities at the ends of linear chromosomes. For example, in the genome C above, 5_h and 2_h are telomeres.

We now define breakpoint and breakpoint distance, for both signed and unsigned genomes. Consider two genomes A and B on the same set of n genes, where both are either signed or both are unsigned. If two genes (or two gene extremities, in the case of signed genomes) are adjacent in A, but not in B, then we say they determine a *breakpoint*. Note that the usual notion of breakpoint distance is defined on unichromosomal genomes, where the *breakpoint distance* between A and B is defined as the number of breakpoints in A (or B). This can be calculated as $d(A, B) = n - a(A, B)$ for two circular genomes, where $a(A, B)$ represents the number of common adjacencies between genomes A and B.

As described in [19], the breakpoint distance between two (signed or unsigned) multichromosomal genomes G and H on the same set of n genes can be defined as

$$d(G, H) = n - a(G, H) - \frac{e(G, H)}{2} \tag{2.1}$$

where $a(G, H)$ is the number of common adjacencies between G and H, and $e(G, H)$ is the number of common telomeres of G and H. Such a definition counts one breakpoint for a fusion or a fission of two linear chromosomes (for two unsigned multichromosomal genomes, if each genome contains a linear chromosome with exactly one gene g, then g contributes twice to the number of common telomeres of the two genomes). Note that we can equivalently think of the breakpoint distance between two multichromosomal genomes G and H as the number of breakpoints between G and H plus half the number of times a gene g is a telomere in one of the two genomes, but not the other.

As an example of using Equality (2.1), consider the genome C from before, and the following genome D with 3 chromosomes on 5 genes:

$D = (-3 - 4) \mid +5 - 1 - 2$

Then $a(C, D) = 1$, $e(C, D) = 2$, and $d(C, D) = 3$.

Given three genomes A, B, and C, the *breakpoint median problem* (BMP) is the problem of finding a genome M, called the *median*, such that the sum of the breakpoint distances between M and each other genome is minimized.

One of the most well-known problems in combinatorial optimization is the *Traveling Salesman Problem (TSP)* (see [12] for background on the TSP). Consider a complete weighted graph. A *Hamilton cycle* is a cycle of the graph that visits every vertex exactly once. A minimum-weight Hamilton cycle is a Hamilton cycle such that the sum of the edge-weights of the cycle is minimized. An optimal solution to the TSP calls for finding such a minimum-weight Hamilton cycle. Due to the wide range of applications and its theoretical appeal, there is a huge amount of research on the

TSP (see [11] and [12] for example).

The TSP is known to be NP-hard. Currently the best k-approximation algorithm known for the TSP, when the costs satisfy the triangle inequality, is the algorithm due to Christofides [8] for which $k = \frac{3}{2}$. However, for the general TSP, a k-approximation algorithm for any constant k would imply P=NP, and thus it is considered highly unlikely to exist [12].

A generalization of the well-known TSP is to consider multiple salesmen. A *Multiple Traveling Salesman Problem* (mTSP) is an assignment of exactly m salesmen to a set of vertices of a graph such that all salesmen start and end their journey at a fixed vertex called the *depot* and each other vertex gets visited exactly once by exactly one salesman. The goal of the mTSP is to minimize the total cost of all of the routes. We define the *r-to-m Multiple Traveling Salesman Problem* (rmTSP) as a variation of the mTSP in which at least r and at most m salesmen are used. Due to several real-life applications, the mTSP has been the subject of several studies [4].

2.3 Transforming the multichromosomal median problem to mTSP

2.3.1 Median problem for unsigned genomes

In this section we describe our solution method for solving the linear multichromosomal BMP in the case where the genomes are unsigned. In 1997, Sankoff and Blanchette [16] showed that the unichromosomal circular BMP can be reduced to the TSP in the case of three unsigned genomes A, B and C over a set of n genes. In [17] the same authors used a branch-and-bound method to solve this TSP. Although useful for many problems in combinatorial optimization, branch-and-bound is not usually an effective method for solving the TSP.

The transformation in [16] is as follows. Let G be a complete undirected graph

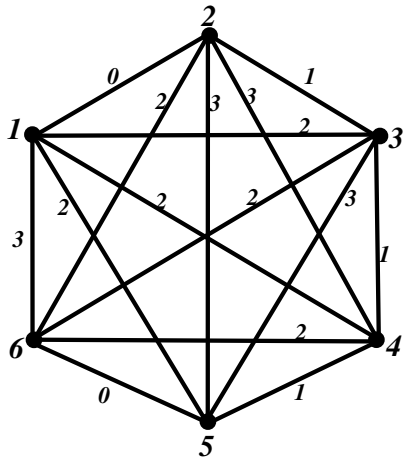


Figure 2.1: BP-median graph constructed for three circular genomes A = (1 2 6 5 4 3), B = (2 1 4 5 6 3), and C = (1 2 3 4 6 5). A breakpoint median is M = (1 2 3 6 5 4).

of n vertices such that each vertex represents one gene. For each edge uv let $adj(uv)$ be equal to the number of times the genes corresponding to u and v are adjacent (do not form a breakpoint) in genomes A, B, and C (so $adj(uv)$ can be 0, 1, 2, or 3). Let the weight of the edge uv, $w(uv)$, be equal to $3 - adj(uv)$. Then the solution to the TSP for the weighted graph G traces out a permutation of $\{1, 2, \ldots, n\}$ that provides an optimal solution to the BMP. We call the weighted graph G the *BP-median graph*. Note that in this method, if we use an edge uv in the final TSP solution, this corresponds to genes u and v being adjacent in the median genome M. The weight $w(uv)$ in the BP-median graph (i.e. $3-adj(uv)$) represents the number of times u and v are not adjacent in three genomes A,B and C. Thus using the edge uv in the TSP solution contributes exactly $w(uv)$ to the sum $d(M, A) + d(M, B) + d(M, C)$ which we are trying to minimize, i.e. the sum of the distance between the median and the other genomes.

Since the goal of the TSP is to find a minimum cost Hamilton cycle, the solution

given by the TSP optimizes the use of adjacencies available in the three genomes. Hence, starting at any vertex of a cycle that is a TSP solution for the problem, we will have a genome that is the breakpoint median for the given three genomes. Figure 2.1 provides the BP-median graph for a simple example of three circular unichromosomal genomes $A = (1\ 2\ 6\ 5\ 4\ 3)$, $B = (2\ 1\ 4\ 5\ 6\ 3)$, and $C = (1\ 2\ 3\ 4\ 6\ 5)$. The edge weights are assigned by using $w(uv) = 3 - adj(uv)$. It can be seen from the example that an optimal TSP solution is 1236541 with the minimum cost equal to 6. Therefore, a breakpoint median for this problem is $M = (1\ 2\ 3\ 6\ 5\ 4)$, and $d(M, A) + d(M, B) + d(M, C) = 6$.

We first provide a transformation from the unsigned linear multichromosomal BMP to an instance of an rmTSP.

Construct the graph G for the genomes the same way as described for the BP-median graph, i.e. G is a complete graph of n vertices corresponding to n genes where the weight of an edge uv is equal to $3-adj(uv)$. Then add a vertex d called the *depot* and add an edge between d and every other vertex g of the graph. Let $\tau(g)$ be equal to the number of times g is a telomere in one of the three genomes. Then, the edge dg has a weight $w(dg) = \frac{3-\tau(g)}{2}$.

Figure 2.2 provides an example of such a transformation for the following three linear multichromosomal genomes A,B and C, where the chromosomes are separated by vertical lines: $A = 1\ 2 \mid 3 \mid 4\ 5 \mid 6 \mid 7$, $B = 1\ 5\ 3 \mid 2\ 7 \mid 4\ 6$, and $C = 3\ 4\ 1 \mid 5\ 2 \mid 7\ 6$.

An rmTSP solution to this graph corresponds to a set of say C cycles, with $r \leq C \leq m$, starting and ending at the depot such that each non-depot vertex belongs to exactly one cycle, with the total cost minimized. Once we obtain such a solution, we can delete the depot and have C disjoint paths (r$\leq C \leq$ m) covering all vertices of the graph such that the overall cost is minimized. Each path will correspond to a linear chromosome in the solution to the multichromosomal BMP. We claim this is the solution to the BMP, and that this solution has cost equal to the cost of the rmTSP solution. To see this, first consider an edge uv in the rmTSP graph where u and v are

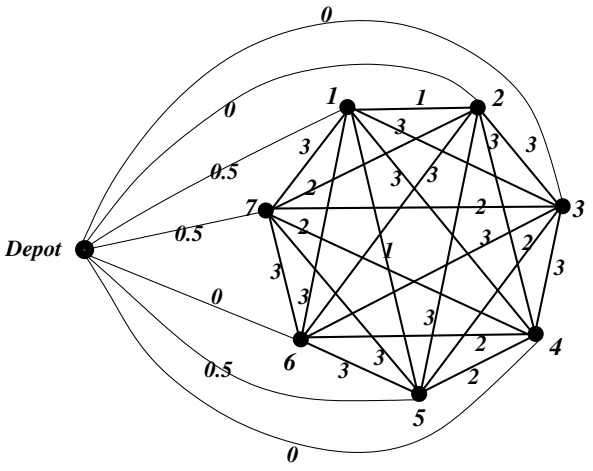


Figure 2.2: An example of transforming a linear multichromosomal BMP to an rmTSP with one depot. The three genomes are $A = 1\ 2 \mid 3 \mid 4\ 5 \mid 6 \mid 7$, $B = 1\ 5\ 3 \mid 2\ 7 \mid 4\ 6$, and $C = 3\ 7\ 4 \mid 6 \mid 5\ 1\ 2$. The chromosomes in each genome are separated by vertical lines.

gene (non-depot) vertices. As we described before for the unichromosomal circular BMP, if we use the edge uv in the final rmTSP solution, this corresponds to u and v being adjacent in the final BMP solution, and the rmTSP solution contributes exactly $w(uv) = 3 - adj(uv)$ to the BMP solution cost $d(M, A) + d(M, B) + d(M, C)$ that we are trying to minimize. Next, consider an edge Dg in the rmTSP graph, where D is a depot vertex and g is a gene (non-depot) vertex. The weight $w(Dg) = \frac{3-\tau(g)}{2}$ represents half the number of times g is not a telomere in A, B, or C. If we use the edge Dg in the final rmTSP solution, this corresponds to gene g being a telomere in the final BMP solution, and the rmTSP solution contributes exactly $w(Dg) = \frac{3-\tau(g)}{2}$ to the BMP solution value of $d(M, A) + d(M, B) + d(M, C)$.

Algorithm 1: Transformation of an unsigned multichromosomal BMP to an rmTSP

1. Construct the BP-median graph G.

2. Add a depot vertex d.

3. Add an edge between the depot d and all vertices g in G corresponding to the genes, such that the weight of such edge is equal to $w(dg) = \frac{3-\tau(g)}{2}$ where $\tau(g)$ is the number of times g is a telomere in one of the chromosomes of the three genomes.

4. Find a solution to the rmTSP on this graph.

Next we show that an rmTSP can be transformed to a TSP. Several methods for such a transformation exist (see [4] for a survey of the results). We provide the most straight-forward transformation in this context.

Algorithm 2: Transformation of rmTSP to TSP

1. Include m copies of the depot, d_1,...,d_m.

2. Add edges from each depot to all non-depot vertices $v_1, \ldots, v_n$. An edge $d_i v_j$ has the same weight as its corresponding edge dv_j in the extended BP-median graph.

3. Add an edge between each pair of depots.

4. Assign the weight ∞ to the edges between $d_1, \ldots, d_{r-1}$ depots and from these $r-1$ depots to all other depots.

5. Assign the weight 0 to the edges between $d_r, \ldots, d_m$ depots.

6. All other edges in the graph (edges with $v_1, \ldots, v_n$ vertices as endpoints) have weights equal to the weights assigned as per the extended BP-median graph.

7. Find a TSP solution for this newly formed graph.

Figure 2.3 provides a sketch of the transformation mentioned in Algorithm 1 for $r = 3$ and $m = 5$.

In order to minimize the total cost, a TSP solution to the expanded graph explained above will be forced to not choose the edges with weight ∞. Therefore, a TSP tour visits at least $r - 1$ depots (the $r - 1$ depots with costs ∞ between them) and for the remaining depots, it will require between 1 and $m - r + 1$ visits. Thus it provides a solution to the rmTSP problem.

A TSP solution on this graph visits every vertex exactly once while minimizing the total sum of the edges it goes through. Therefore, those edges with weight ∞ are not included in the TSP solution. So for the $r-1$ depots for which the incident edges from depots all have value infinity (d_1 and d_2 in Figure 2.3), a TSP solution must visit those depots by using two edges that have one end in the original BP-median graph. The other depots (d_3, d_4 and d_5 in Figure 2.3) may get visited by using the edges between them (edges of weight 0), or the edges with one end in the original BP-median graph, or a combination of these two types of edges. Therefore, a TSP solution has to visit the subgraph containing the depots (the left side of Figure 2.3) at least r times and at most m times. So we have obtained a solution to the rm-TSP.

By combining the transformation from the unsigned linear multichromosomal BMP to an instance of an rmTSP and Algorithm 1, we have reduced the unsigned linear multichromosomal BMP to a TSP. Once a solution to the TSP with r to m visits to the depots is obtained, we delete the depots and the adjacent edges. Deleting the depots will result in at least r and at most m disjoint paths covering all vertices exactly once. A TSP solution has minimized the total cost on the edges. Therefore, after removal of the depots, this solution traces a set of minimum cost disjoint paths covering the vertices of the graph. Each of the these disjoint paths gives the sequence of genes in one linear chromosome of the median. So we have obtained a median with at least r and at most m linear chromosomes.

For example, if we consider our previous example of A, B and C, our optimal

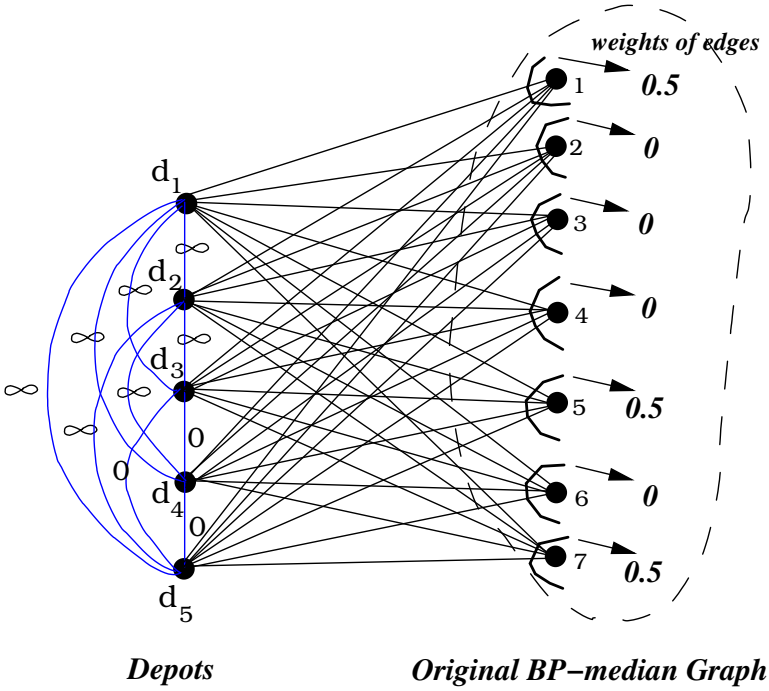


Figure 2.3: Transforming an rmTSP to TSP for r=3, m=5. The original BP-median graph is the graph presented in Figure 2.2 (without the depot). The edges between the depots and the original graph are grouped together based on their endpoint in the original graph. Each group has the same weight written besides the arrow.

TSP tour for the TSP problem shown in Figure 2.3 is d_1 3 7 4 d_5 d_3 2 d_2 5 1 d_4 6, which gives the optimal median solution $M = 3\ 7\ 4 \mid 2 \mid 5\ 1 \mid 6$.

It should be noted that there is a one-to-one correspondence between the solutions of the multichromosomal BMP and the TSP formed. As explained above, every linear multichromosomal BMP can be solved by transforming it into a TSP. On the other hand, every optimal TSP tour provides an ordering that minimizes the total cost of the edges of the graph and therefore, it provides the order of genes in a multichromosomal breakpoint median. Also, the length of the TSP tour is equal to the sum of the breakpoint distances between the median and the other genomes.

Special cases for various numbers of chromosomes

Note that our solution method for the linear multichromosomal BMP allows us to choose the range (r to m) of the number of linear chromosomes which we will allow in the BMP solution. Thus, we can extend our method to include several special cases of the median problem that may be useful in real-world applications. For example, the common ancestor of the grass family in 52.5 million years ago has 12 chromosomes, while the children have anywhere from 5 to 12 chromosomes: sorghum has 10, rice has 12, Brachypodium distachyon has 5 and wheat lineage has 7 chromosomes [15]. Therefore, in such examples it may be appropriate for the algorithm to set a maximum for the number of the chromosomes of the ancestral genome, where that maximum is defined by some parameters or biological data such as the largest number of chromosomes in the children. In such cases, we would use an rmTSP with $r = 1$ (or $r =$ minimum number of chromosomes required) and m equals the maximum.

Other cases, such as enforcing the ancestor to have exactly k chromosomes might be desired in cases when k is given by some other methods (statistical, laboratory, etc.), but the sequence of genes in the median is unknown. In this case we would use $r = m = k$ in the rmTSP model.

Linear unichromosomal BMP

We must point out that if we only add one depot (set $r = m = 1$ in our transformation), we are using exactly one salesman. This corresponds to having one chromosome, i.e. the unichromosomal BMP.

There is a key difference between how the unichromosomal BMP for a linear versus a circular chromosome is modeled that often goes unnoticed. The transformation provided in [16] from the unichromosomal BMP to the TSP is valid for circular unichromosomal genomes. Though not specifically mentioned in the literature, it is commonly assumed that the case for linear unichromosomal genomes is similar to the

circular case, in the sense that the circular median can be found and then "cut" to obtain a linear median. However, this assumption is not necessarily true. The main problem is that once a circular median in the form of a TSP tour is obtained, it is unclear where to "cut" the tour to obtain a path corresponding to a linear median. Moreover, an optimal TSP tour corresponding to a solution to the circular BMP, may not contain the optimal solution to the linear case (an optimal path covering all vertices of the graph exactly once) at all. For instance, in the example shown in Figure 2.1, a circular median is $M = (1\ 2\ 3\ 6\ 5\ 4)$ with an optimal cost of 6. This is an optimal solution to the circular unichromosomal BMP and an optimal TSP tour. The optimal solution to the linear unichromosomal BMP for this example is $P = 1\ 2\ 3\ 4\ 5\ 6$ with cost 3. However, it is impossible to obtain an optimal path P from M. Indeed, the best path that can be obtained from M by removing one edge, would have a cost of at least 4, which is higher that the cost of P. Therefore, the optimal solution M to the circular unichromosomal BMP may not yield to an optimal solution for the linear version of the problem.

To fix this problem, we propose the following solution. Use our method for the linear multichromosomal BMP, with the restriction that there is exactly one chromosome in each genome. Then, the unichromosomal case is just a special case of the multichromosomal BMP. Specifically, add a depot vertex to the BP-median graph, and add the edges between the depot and all other vertices with the edge weight of $\frac{3-\tau(g)}{2}$. Solve the TSP on this extended graph where an optimal tour starts and ends at the depot. Then remove the depot to obtain a path of minimum cost that goes through all vertices. This path corresponds to the sequence of the genes in the solution for the linear unichromosomal BMP.

2.3.2 Median problem for signed genomes

The transformation described in the previous section considers unsigned genomes. In this section we provide a method to transform a signed unichromosomal circular BMP to a TSP instance and we then generalize the result to include the linear multichromosomal case, and thereby reduce the linear multichromosomal breakpoint median problem on signed genomes to a TSP. Recall that the breakpoint distance between two signed multichromosomal genomes can be defined using Equality(2.1), and that in signed genomes, there are four possible adjacencies between two genes u and v depending on the direction of the genes: $u_h v_h$, $u_h v_t$, $u_t v_h$, or $u_t v_t$. The telomeres are sets that contain one element, i.e. u_h, u_t, v_h or v_t.

Consider the signed BMP on three unichromosomal circular genomes A, B, and C where each genome has n (signed) genes. We construct a *signed BP-median graph* G with $3n$ vertices. Each gene g in the chromosome corresponds to three vertices g_h (representing the head of the gene), g_t (representing the tail of the gene) and g_m in the graph. Both edges $g_h g_m$ and $g_m g_t$ have weights equal to zero. All the other edges with g_m as an endpoint have weights equal to infinity. Also, all edges between the head and the tail of the same gene, $g_h g_t$, have weights equal to infinity. See Figure 2.4. A positive (negative) gene in a genome indicates that in a Hamilton cycle of G, the head (tail) is visited before the tail (head). All other edges are formed as in the BP-median graph, and the weights are similarly assigned based on the adjacencies in the signed genomes: for example, the weight $w(g_t f_h)$ on the edge between the tail of gene g and the head of gene f is $3 - adj(g_t f_h)$ where $adj(g_t f_h)$ is the number of times tail of g and head of f are adjacent.

The existence of a vertex g_m between the head and the tail of each gene g is necessary to ensure that an optimal tour for the TSP includes g_h if g_t is picked (and vice-versa) by traveling through g_m. Otherwise, if g_m is not included in the model, it is possible to have a case where one endpoint of a gene is visited and it is not followed

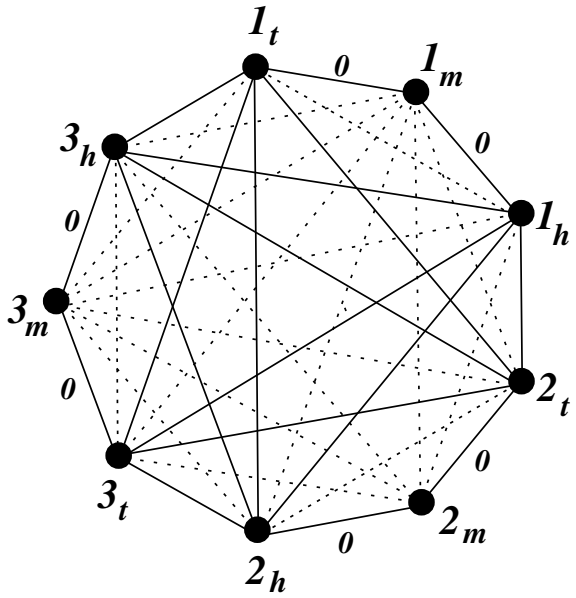


Figure 2.4: An example of transforming a circular unichromosomal signed BMP for 3 genes to a TSP on 9 vertices: the solid unlabeled edges will receive weights according to the adjacencies, and the dotted edges have infinity as their weight.

by the other endpoint. Note that the edges of weight ∞ in the signed BP-Median graph G ensure that the edges $g_t g_m$ and $g_m g_t$ will be included in the optimal TSP solution for every gene g.

Similar to the unsigned case, a solution to the TSP on the signed BP-median graph provides a solution to the BMP simply by taking the vertices in the order they are visited in the TSP solution, ignoring the g_m vertices. For example, in Figure 2.4, the TSP tour $1_t1_m1_h2_h2_m2_t3_t3_m3_h1_t$ corresponds to the genome $(1_t1_h\ 2_h2_t\ 3_t3_h)$.

In the multichromosomal case, similar to what was described previously for the unsigned case, the problem can be transformed to an rmTSP.

Algorithm 3: Transformation of a signed multichromosomal BMP to an rmTSP

1. Construct the signed BP-median graph.

2. Add a depot vertex d.

3. Add an edge between the depot d and all vertices g corresponding to the heads and tails, such that the weight of such edge is equal to $w(dg) = \frac{3-\tau(g)}{2}$ where $\tau(g)$ is the number of times g is a telomere in one of the chromosomes of the three genomes.

4. Add an edge of weight ∞ between the depot and all vertices g_m.

5. Find a solution to the rmTSP on this graph.

A solution to the rmTSP described above has at least r and at most m visits to the depot. Let P be such a solution. After deleting the depot, we are left with at least r and at most m disjoint paths covering all (non-depot) vertices of the graph. Once again, if we ignore the g_m vertices, each path corresponds to a chromosome in the solution to the signed multichromosomal BMP. By combining Algorithm 1 and Algorithm 2 a multichromosomal BMP for signed genomes can be transformed to a TSP.

Similar to Section 2.3.1 for unsigned genomes, our proposed method for signed genomes can handle different requirements on the number of linear chromosomes.

As mentioned in [13], since typically the number of salesmen m for an rmTSP is much less than the number of vertices n of the graph, the addition of m depots and therefore expanding the graph to $n + m$ vertices is not a substantial increase, when $m \ll n$. Also, in theory, the mTSP on $n + m$ vertices should be a computationally easier problem to solve than the TSP on n vertices [13]. From a biological point of view, this is also accurate, since the m depots correspond to m chromosomes, and typically, the number of chromosomes is far less than n, the number of genes.

2.4 Computational experiments

We tested our method on randomly generated datasets using uniform distribution. For this purpose, we randomly generated sets of genomes with various numbers of chromosomes. To simulate the occurrence of breakpoints, randomly selected genes in each genome were swapped. The number of gene swaps corresponding to the occurrence of breakpoints were assigned by setting a ratio $k = \frac{m}{n} \times 100$ where m is the number of breakpoints compared to the identity, and n is the total number of genes present in the genome. We performed our tests for a range of ratios between 10% and 90%, for the number of genes ranging from 10 to 10,000. We also tested this method on both signed and unsigned genomes with various numbers of chromosomes: A fixed chromosome number for all genomes, and a lower and upper bound for the number of chromosomes allowed in each genome. After generating these random datasets, we created the BP-median graph for each test. We then used our transformation to add the depots to the graph based on the number of chromosomes, and found the input for the rmTSP. The next step was to apply the second algorithm to tranform the rmTSP to a TSP. Once we obtained an instance of the TSP corresponding to our linear multichromosomal BMP, we applied Concorde [2] to solve the TSP. Concorde is a sophisticated TSP solver that combines many of the most recent advances on TSP research to find exact or provably nearly-exact solutions for a TSP. It uses the linear programming formulation of the TSP and applies advanced branch-and-cut steps to move towards the optimal solution. Concorde starts from a feasible solution (not necessarily optimal) and applies cutting-plane methods to move towards the optimal solution. It also finds a lower bound on the value of the optimal solution. If a solution is obtained with value equal to the lower bound, then it is an optimal solution. If the optimal solution cannot be reached, Concorde provides the best found solution and a bound indicating its worst-case difference from optimality. The output from Concorde is in the form of a sequence of vertices that gives the TSP tour. Based on

our method, we can translate this tour back into the sequence of the genes present in the median. We used the C programming language, and all the tests were performed on an Intel Xeon 3.2 GHz with 3.2 GB of memory and the GCC compiler for Linux. The reported results are for $k = 30\%$ and they are averages over 5 runs for each sample type of the simulated data. There was no noticeable difference in performance time and optimality gap within various ratios. The optimality gap is calculated as the worst-case difference, as a percentage, between the reported median and an optimal solution (i.e. the difference, as a percentage, between the reported median value and the lower bound found by Concorde on the optimal solution value). Note that an optimality gap of 0 indicates that an optimal solution was found. Table 2.1 provides a summary of the results of applying our method for the multichromosomal BMP. The average time to find the optimal median in the case of 3 chromosomes in each genome was 127.54 seconds. Note that we were successful in obtaining an optimal median in all cases where $n < 10000$. For $n \geq 10000$, the worst-case accuracy of 7% gap from the optimal solution was for 25000 genes per genome spread over 23 chromosomes.

For real-world datasets we tested our method on the human, cat, and mouse gene data available from [20]. This dataset consists of 114 common genes (markers) contained in several chromosomes. Our method was able to find an optimal breakpoint median of this dataset in less than one minute. The parameters chosen were $r = 19$ and $m = 23$ (The number of chromosomes for cat, mouse and human are 19, 20, and 23, respectively). An optimal median was found with 20 chromosomes.

As mentioned in other studies (for example see [23]) and to the best of our knowledge, aside from our current work, there are no other large-scale linear multi-chromosomal median solvers for breakpoint distance. We should mention that there are other tools available for finding the median of multichromosomal genomes using other distance functions. These tools include MGR [5], GRIMM [20], MGRA [1], and

ASMedian [21]. Among these packages, MGR and GRIMM are primarily based on reversal distance and can only handle genomes of significantly smaller size (in the range of 1000 or fewer genes). MGRA can handle larger data, however, it is based on a scenario called "2-break" that is used to approximate other rearrangement scenarios such as reversals. ASMedian, solves the median problem under Double Cut-and-Join (DCJ) distance. A recent version of ASMedian, called ASMedian-linear [22] has been developed for linear chromosomes under DCJ distance and tested on large datasets of up to 5000 genes, again using the DCJ distance measure. The reported running times for ASMedian-linear are very fast. One drawback is that it is possible to have circular chromosomes in the solution found by ASMedian-linear. As mentioned, none of these tools are based on breakpoint distance and therefore, we were not able to compare them to our method.

2.4.1 Unichromosomal breakpoint median problem

The *circular unichromosomal breakpoint median problem* can be defined in cases where each genome has only one circular chromosome. There have been more studies on the unichromosomal case compared to the multichromosomal median problem. Therefore, we decided to test our method for the unichromosomal case as well so we could compare its performance with other available packages, given that we know of no other methods for large-scale linear multichromosomal breakpoint median problem. For this purpose, we focused on the circular unichromosomal BMP for signed and unsigned genomes. This problem is known to be NP-hard ([6], [7]). In Section 2.3 we discussed the transformation proposed in [16] for transforming the unichromosomal BMP for unsigned genomes, as well as our adaptation of this method for signed genomes, into an instance of TSP. We use this transformation and apply the TSP solver Concorde [2] to obtain a solution to the TSP, and then translate this solution to the median. We tested our method on both real-world and synthetic datasets of

circular unichromosomal genomes, each containing 10 to 10,000 genes for both signed and unsigned cases. For all of our test data, we were able to obtain an optimal solution. The time it took for our method to reach optimality ranged from less than 1 second to about 5 minutes for the largest dataset on an Intel Xeon 3.2 GHz with 3.2 GB of memory running Linux. The times reported are averages over 5 runs for each sample type of the simulated data. The optimal median is found using the optimal TSP solution from Concorde. Total average time over all samples for our method was 39.38 seconds.

For real-world datasets, we tested our method on the human, fruit fly, and sea urchin mtDNA data which was used in [18]. This dataset contained 33 signed genes over the three genomes. We also tested the Campanulaceae cpDNA dataset which was considered a more challenging dataset studied in [9]. This is a dataset of 13 genomes each with 105 genes. In both cases, our method was able to find optimal solutions in a matter of seconds.

A well-known genomic median solver is GRAPPA [3]. We compared the performance of our method to GRAPPA. For smaller genomes (less than 60 genes per genome) GRAPPA was faster than our method, however, once the genome size exceeded 60 genes, our method outperformed GRAPPA in terms of speed. For $n \geq 1000$, GRAPPA was unable to find any median even after one hour of running time. Also, in terms of accuracy, our method obtained provably optimal results for all test cases, whereas GRAPPA was, on average, 20% away from optimal in the test cases where it was applied and able to find a solution. Table 2.2 provides the results for the unichromosomal BMP.

2.5 Conclusions

In this paper, a novel approach for solving the breakpoint median problem on signed and unsigned multichromosomal genomes is presented. The focus of our framework

is on the NP-hard problem of finding a multichromosomal breakpoint median for linear genomes. Our method is based on constructing a complete graph that has all the genes as vertices and the edge weights representing the breakpoints. We then obtain a multiple salesman TSP by adding a depot. Next, we converted this graph into an rmTSP. The parameters r and m (minimum and maximum number of salesmen/chromosomes) can be used to capture different biological assumptions on the number of chromosomes in a common ancestor. By setting these parameters, we can use our method to test hypotheses on the number of chromosomes of a common ancestor, even when the known genomes or the available data provide less chromosomes. It should be noted that the parameters r and m only need to be specified if desired. If such information is unknown or we do not want to restrict our solutions to particular parameters, we can still use the same method without any restrictions. In such cases, we can set r to be the minimum number of chromosomes (say 1) and m to be the maximum number of chromosomes possible.

Also, our method can be easily extended to include more than 3 genomes. For example, if we want to find the median of K genomes, it suffices to set the weights as $w(gh) = K - adj(gh)$ instead of $3 - adj(gh)$ for genes g and h, and $w(dg) = \frac{K-\tau(g)}{2}$ instead of $\frac{3-\tau(g)}{2}$ for a depot d and a gene g in our TSP transformation.

The other advantage of this method is its computational efficiency in terms of both the running time and also accuracy of finding optimal results. In terms of the running time, the presented method allowed us to compute the median in realistic time limits on datasets which can appropriately resemble real-world data. In all cases up to 5000 genes an optimal solution is found. For datasets of size 10,000 the optimality gap is 2% and in the largest dataset that was tested, $n = 25000$ with 23 chromosomes, the worst case difference between the solution we found and the optimal solution value is 7% and this solution was found in less than 20 minutes.

One drawback of this method is the assumption of containing equal gene content in the genomes. This is a very common assumption in the field in order to reduce the

general problem into manageable datasets. Ideally, models should allow for unequal numbers of genes in the genomes. We are planning to extend our method to include cases where unequal gene content can be considered, and we believe that such an extension of our framework can be done. For future work, including several copies of the genes may also lead to a more realistic model.

We should point out that finding a median is a combinatorial optimization problem that is only a rough approximation of a common ancestor. With more genomic data becoming available and different biological hypotheses tested on known common ancestors, it would be very interesting to compare the results of the median obtained by our method to the sequence of a known common ancestor. Such comparison could shed light on appropriate parameters and constraints that need to be considered in computing medians in the future.

# of Genes Per Genome	# of Chromosomes Per Genome	Time (seconds)	Median Score	Optimality Gap
10	3	0	15	0%
100	3	5.12	182	0%
500	3	18.49	873	0%
1000	3	42.17	1802	0%
5000	3	125.19	9247	0%
10000	3	574.26	5274	0%
10	$3 \leq n \leq 5$	0	18	0%
100	$3 \leq n \leq 5$	9.07	192	0%
500	$3 \leq n \leq 5$	28.31	943	0%
1000	$3 \leq n \leq 5$	189.06	2403	0%
5000	$3 \leq n \leq 5$	170.41	10684	0%
10000	$3 \leq n \leq 5$	681.01	14278	2%
25000	n=23	1091.72	31361	7%

Table 2.1: Performance for the multichromosomal breakpoint median problem on simulated data sets with a ratio of 30%, averages over 5 runs for each sample type of the simulated data. The optimality gap is calculated as the difference, as a percentage, between the reported median and the lower bound found by Concorde on the value of an optimal solution.

Number of genes per genome	**Time for our method (sec.)**	**Time for GRAPPA (sec.)**	**Median Score our method**	**Median Score GRAPPA**	**Optimal Median**
10	0.01	0	11	14	11
20	0.05	0	33	43	33
30	0.07	0.02	44	56	44
50	0.18	0.04	76	91	76
60	0.33	0.28	96	104	96
70	0.49	3.2 min	113	121	113
80	0.76	12.5 min	137	148	137
90	1.08	48.7 min	142	153	142
100	1.34	$>$ 60 min	168	172	168
150	2.02	$>$ 60 min	259	267	259
200	3.88	$>$ 60 min	351	352	351
500	10.74	$>$ 60 min	861	869	861
1000	28.12	$>$ 60 min	1743	N/A	1743
1500	22.27	$>$ 60 min	2640	N/A	2640
2000	39.31	$>$ 60 min	3485	N/A	3485
3000	54.07	$>$ 60 min	5259	N/A	5259
5000	83.54	$>$ 60 min	8719	N/A	8719
10000	328.14	$>$ 60 min	10751	N/A	10751

Table 2.2: Our method versus GRAPPA: unichromosomal breakpoint median problem. The results are for simulated data with a ratio of 30%. The times reported are averages over 5 runs for each sample type of the simulated data.

Bibliography

[1] Alekseyev, M.A., and Pevzner, P.A.: Breakpoint graphs and ancestral genome reconstructions, Genome Research, 19:943-957, 2009.

[2] Applegate, D., Bixby, R., Chvatal, V., and Cook, W.: Concorde TSP Solver, www.tsp.gatech.edu/concorde

[3] Bader, D.A., Moret, B.M.E., Warnow, T., Wyman, S.K., and Yan., M.: GRAPPA (Genome Rearrangements Analysis under Parsimony and other Phylogenetic Algorithms), www.cs.unm.edu/ moret/GRAPPA/

[4] Bektas, T.: The multiple Traveling Salesman Problem: an overview of formulations and solution procedures, Elsevier-Omega 34, 209-219, 2006.

[5] Bourque, G., and Pevzner, P.A.: Genome-scale evolution: reconstructing gene orders in ancestral species, Genome Research, 12:26-36, 2002, http://grimm.ucsd.edu/MGR/

[6] Bryant, D.: The complexity of the breakpoint median problem, technical report CRM-2579, Centre de Recherches de Mathematiques, Universite de Montreal, 1998.

[7] Caprara, A.: Additive bounding, worst-case analysis and the breakpoint median problem, SIAM Journal on Optimization, 13:508-519, 2002.

[8] Christofides, N.: Worst case analysis of a new heuristic for the traveling salesman problem, Report 388, Graduate School of Industrial Administration, Carnegie Mellon University, Pittsburgh, 1976.

[9] Cosner, M.E., Jansen, R.K., Moret, B.M.E., Raubeson, L.A., Wang, L.S., Warnow, T., and Wyman, S.: An empirical comparison of phylogenetic methods on chloroplast gene order data in Campanulaceae, Comparative Genomics (DCAF-2000), 104-115, 2000.

[10] Fertin, G., Labarre, A., Rusu, I., Tannier, E., and Vialette, S.: Combinatorics of genome rearrangements, The MIT Press, 2009.

[11] Gutin, G., and Punnen, A.P. (eds.): The traveling salesman problem and its variations, Kluwer, Dordrecht, 2002.

[12] Lawler, E., Lenstra, J., Rinnooy Kan, A., and Shmoys, D.: The traveling salesman problem–a guided tour of combinatorial optimization, Wiley, Chichester, 1985.

[13] Orloff, C.S.: Routing a fleet of m vehicles to/from a central facility, Networks, 4, 147-162, 1974.

[14] Pe'er, I., and Shamir, R.: The median problems for breakpoints are NP-complete, Electronic Colloquium on Computational Complexity, technical reports 71, 1998.

[15] Salse, J., Bolot, S., Throude, M., Jouffe, V., Piegu, B., Quraishi, U.M., Calcagno, T., Cooke, R., Delseny, M., and Feuillet, C.: Identification and characterization of shared duplications between rice and wheat provide new insight into grass genome evolution, The Plant Cell, 20:11-24, 2008.

[16] Sankoff, D., and Blanchette, M.: The median problem for breakpoints in comparative genomics, in Proceedings of the Third International Computing and

Combinatorics Conference, T.Jiang and D.T.Lee eds, Lecture Notes in Computer Science 1276:251-263, Springer-Verlag, 1997.

[17] Sankoff, D., and Blanchette, M.: Multiple genome rearrangement and breakpoint phylogeny, Journal of Computational Biology, 5:555-570, 1998.

[18] Sankoff, D., Sundaram, G., and Kececioglu, J.: Steiner points in the space of genome rearrangements, International Journal of Foundations of Computer Science 7:1-9, 1996.

[19] Tannier, E., Zheng, C., and Sankoff, D.: Multichromosomal median and halving problems under different genomic distances, BMC Bioinformatics, 10:120, 2009.

[20] Tesler,G.: GRIMM: genome rearrangements web server, Bioinformatics, 18-3: 492-493, 2002, http://grimm.ucsd.edu/GRIMM/

[21] Xu, A.W., and Sankoff, D.: Decompositions of multiple breakpoint graphs and rapid exact solutions to the median problem, Lecture Notes in Computer Science, 5251:25-37, 2008.

[22] Xu, A.W.: The median problems on linear multichromosomal genomes: Graph representations and fast exact solutions, Journal of Computational Biology, 17-9:1195-1211, 2010.

[23] Zhang, M., Arndt, W., and Tang, J.: An exact median solver for the DCJ distance, Proceedings of the 14th Pacific Symposium on Biocomputing, 138-149, 2009.

Chapter 3

Mixed and Circular Multichromosomal Breakpoint Median Problem

• Sylvia Boyd and Maryam Haghighi, Mixed and Circular Multichromosomal Breakpoint Median Problem, to be submitted to SIAM Journal on Discrete Mathematics (SIDMA), 2011.

Work of Maryam Haghighi under the supervision of Sylvia Boyd. All results are joint work of Maryam Haghighi and Sylvia Boyd.

Abstract

We study the problem of finding the breakpoint median for multichromosomal unsigned genomes that contain circular or mixed (circular and linear) chromosomes. We show that these problems are polynomial. We also study restricted versions of the multichromosomal breakpoint median problem where the number of genes allowed in each chromosome of the median must satisfy a lower bound. We show that this restricted version of the problem is NP-hard. Moreover, we introduce the notion of partially signed genomes, where both signed and unsigned genes may exist in the same genome. We provide the first study of the breakpoint median problem for partially signed genomes and show that this problem is also polynomial for both circular and mixed multichromosomal genomes.

3.1 Introduction

There has been extensive research on genome analysis of gene orders and their application in constructing evolutionary trees. Generally speaking, the distance $d(A, B)$ between two genomes A and B is defined as a measure of how far apart the two genomes are from each other in terms of gene ordering. The question of finding the common ancestor of a given set of genomes is one of the fundamental problems required for building phylogenies. The notion of a median is often used as a way to model this problem. Formally, given a set G of m genomes, the *median* is a genome that minimizes the total sum of the distances from all other genomes, i.e. the median is a genome M such that $\sum_{g_i \in G}(d(M, g_i),\ 1 \leq i \leq m)$ is minimized. The most common version of this problem, sometimes referred to as the *3-median problem*, involves finding the median of three given genomes. From now on, we focus on the 3-median problem and we refer to it as the *median problem*. As we will see later, this is not a significant restriction since all of our results can easily be extended to the median of any number of genomes.

In this paper, we concentrate on finding the median of multichromosomal circular and mixed (linear and circular) genomes under breakpoint distance. The unichromosomal breakpoint median problems are known to be NP-hard ([1] and [4]). However, in [6] it was shown that, contrary to the unichromosomal case, the multichromosomal breakpoint median problem for circular and mixed signed genomes is polynomial. In this paper, we will show that a similar result holds for unsigned genomes, i.e. we will provide the first proof that the multichromosomal breakpoint median problem for circular and mixed unsigned genomes is polynomial. Furthermore, we will introduce the problem of finding the multichromosomal breakpoint median for mixed and circular genomes where the number of chromosomes is restricted, and show that this problem is NP-hard, i.e. adding the size restrictions totally changes the complexity of the problem. In addition, we will present the notion of partially signed genomes (where

both signed and unsigned genes are allowed in the same genome), and provide the first study of the breakpoint median problem on circular and mixed partially signed genomes. We will prove that this is a polynomial-time problem.

We first present the formal definitions for unsigned genomes. An *unsigned gene* g is a sequence of DNA where the orientation is unknown. A set of unsigned genes form an unsigned *genome*. We can represent an unsigned genome on n genes by a string of unsigned integers $1, 2, \ldots, n$ which represents the ordering of the genes in the genome. This string can be broken into segments, representing the *chromosomes* of the genome. Chromosomes can be *circular* or *linear*. Circular chromosomes have a circular gene ordering (which will be represented with brackets around the segment). A linear chromosome has two extremities, called the *telomeres* of the chromosome, and will be represented by bars around the segment. For example, G = (4 5 2 6) | 3 1 | represents an unsigned genome G on 6 genes with one circular chromosome and one linear chromosome which has telomeres 1 and 3. If all of the chromosomes in a genome are circular, the genome is called *circular*, and if all of the chromosomes are linear, the genome is called *linear*. If some chromosomes are linear and some are circular, the genome is called *mixed*.

Given an unsigned genome A, we say two genes are *adjacent* in A if they are adjacent in the gene ordering. For example, in G above, 5 and 2 are adjacent, 4 and 6 are adjacent, 5 and 6 are not adjacent, 6 and 3 are not adjacent, and 3 and 1 are adjacent.

Next we present the definitions for signed genomes. In this case, the orientation of each gene is known, and a gene g is a sequence of DNA with two extremities called the tail and the head, denoted by g_t and g_h respectively. A *signed genome* is a sequence of oriented genes. As in the unsigned case, we can represent a signed genome on n genes by a string of integers $1, 2, \ldots, n$ broken into segments representing the chromosomes, where each integer will be given a sign (+ or −) representing the orientation of the gene. In this notation, we let $+g$ represent the gene g in the orientation $g_h g_t$, and we

let $-g$ represents the gene in the orientation $g_t g_h$. For example, consider the following signed genome C with 5 genes:

$$C = (-3 - 4 + 1) \mid \ + 5 - 2 \mid.$$

Genome C has two chromosomes, one is circular and one is linear. We could also represent C by writing each gene as its ordered tail and head extremities:

$$C = (3_t 3_h \ 4_t 4_h \ 1_h 1_t) \mid 5_h 5_t \ 2_t 2_h \mid.$$

For signed genomes, the adjacencies are not defined on the genes, but instead on the extremities of the genes, namely the heads and tails of the genes. We say an extremity of gene u and an extremity of gene v are adjacent if they are adjacent in the ordering of the genome. Thus, there are four possible adjacencies between two genes u and v depending on the direction of the genes: $u_h v_h$, $u_h v_t$, $u_t v_h$, or $u_t v_t$. The telomeres are sets that contain one element, i.e. u_h, u_t, v_h or v_t. For example, in the genomes C above, the adjacenies are 3_t and 1_t, 3_h and 4_t, 4_h and 1_h, and 5_t and 2_t. The telomeres of the genome are the gene extremities at the ends of linear chromosomes. For example, in the genome C above, 5_h and 2_h are telomeres.

As previously mentioned, the distance between two genomes is a measure of how far apart they are from each other. There are various definitions for distance. We focus on the breakpoint distance measure. Consider two genomes A and B on the same set of n genes, where both are either signed or both are unsigned. If two genes (or two gene extremities, in the case of signed genomes) are adjacent in A, but not in B, then we say they determine a *breakpoint*. Note that the usual notion of breakpoint distance is defined on unichromosomal genomes, where the *breakpoint distance* between A and B is defined as the number of breakpoints in A (or B). This can be calculated as $d(A, B) = n - a(A, B)$ for two circular genomes, where $a(A, B)$ represents the number of common adjacencies between genomes A and B.

As described in [6], the breakpoint distance between two (signed or unsigned) multichromosomal genomes G and H on the same set of n genes can be defined as

$$d(G, H) = n - a(G, H) - \frac{e(G, H)}{2} \tag{3.1}$$

where $a(G, H)$ is the number of common adjacencies between G and H, and $e(G, H)$ is the number of common telomeres of G and H. Such a definition counts one breakpoint for a fusion or a fission of two linear chromosomes. We can equivalently think of the breakpoint distance between two multichromosomal genomes G and H as the number of breakpoints between G and H plus half the number of times a gene g is a telomere in one of the two genomes, but not the other.

Note that for two unsigned multichromosomal genomes, if they each contain a linear chromosome with exactly one gene g, then g contributes twice to the number of common telomeres of the two genomes. Also, if they each contain a circular chromosome of two genes (a b), then we have two adjacencies between a and b. Note that a circular chromosome of exactly one gene is assumed not to occur.

As an example of using Equality (3.1), consider the the following genomes C and D:

$$C = (-3 - 4 + 1) \mid +5 - 2 \mid, \quad D = (-3 - 4) \mid -5 \mid -1 - 2 \mid.$$

Then $a(C, D) = 1$, $e(C, D) = 2$, and $d(C, D) = 3$.

Given three genomes A, B, and C, the *breakpoint median problem* (BMP) is the problem of finding a genome M, called the *median*, such that the sum of the breakpoint distances between M and each other genome is minimized.

Let M be a potential candidate for a median of genomes A, B, and C. The *median score* of M is defined by $d(M, A) + d(M, B) + d(M, C)$. Clearly, the (optimal) median has the minimum median score.

The unichromosomal BMP is NP-hard for signed and unsigned genomes ([1] and [4]). In 2009, the authors of [6] provided the first theoretical proof that the multi-

chromosomal BMP on circular or mixed signed genomes can be solved in polynomial time. This was contrary to the common belief that in general, median problems are NP-hard. In this paper, we will provide the first proof that multichromosomal BMP on circular or mixed unsigned genomes is also polynomial. Our proof is based on the well-known perfect 2-matching problem and provides us with a polynomial-time algorithm to solve the multichromosomal BMP on circular or mixed genomes. Moreover, we will define the multichromosomal BMP for partially signed genomes, i.e. where some of the genes are signed and some are unsigned, and prove that this problem is polynomial for circular or mixed genomes. To the best of our knowledge, this is the first study of partially signed genomes.

3.2 Background

Let $G = (V, E)$ be a weighted undirected graph. A *matching* in G is a set of edges such that no two edges have a vertex in common. A *perfect matching* in G is a matching that covers all vertices of G. A *perfect 2-matching* M on G is a multi-subgraph $M = (V, E')$ of G which uses each edge of G 0, 1, or 2 times, and such that every vertex $v \in V$ is incident with exactly 2 edges in M. For a weighted graph G, a *minimum weight 2-matching* is a perfect 2-matching such that the sum of the edge weights of the edges present in the perfect 2-matching is minimum. Note that if an edge is used twice in the 2-matching, then its weight $w(e)$ is added to the total weight twice. A solution to the perfect 2-matching problem on a graph gives a collection of disjoint cycles in G such that every vertex of G appears in exactly one cycle and the total sum of the edge weights present in the perfect 2-matching solution is minimized. Note that some of these cycles may have length 2, i.e. consist of an edge used twice.

Finding a minimum weight Hamilton cycle in a graph is one of the most famous problems in combinatorial optimization and is referred to as the Traveling Salesman Problem (TSP). Note that every Hamilton cycle (and therefore every TSP tour) is a

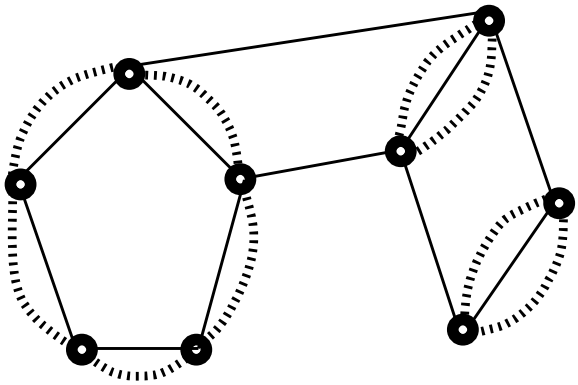


Figure 3.1: An example of a perfect 2-matching for a graph of 9 vertices. The solid lines represent the graph and the dotted lines represent the perfect 2-matching.

perfect 2-matching. However, every perfect 2-matching is not necessarily a Hamilton cycle, i.e. there may be more than one cycle present in a perfect 2-matching. Clearly, the perfect 2-matching problem is a relaxation of the TSP, i.e. the minimum weight (cost) of a perfect 2-matching provides a lower bound for the optimal solution to the TSP. For general graphs, a perfect 2-matching may not always exist, however, for complete graphs, there is always a perfect 2-matching. Figure 3.1 shows an example of a perfect 2-matching for a graph of 9 vertices, where the graph is shown by the solid lines and the dotted lines show a perfect 2-matching.

Finding a minimum weight perfect 2-matching on graphs is a well-studied problem in combinatorial optimization and several algorithms for such problems exist (for example, see [3] and [5]). In general, a minimum weight perfect 2-matching for a graph with n vertices can be found in $O(n^3)$ time [5].

Another closely related concept on graphs is a 2-factor. A *2-factor* of a graph G is a perfect 2-matching that uses every edge at most once, i.e. has no cycles of length 2. Thus, a 2-factor consists of a set of disjoint cycles of length three or more covering all vertices of the graph. The *minimum weight 2-factor problem* (also called the *2-factor problem*) in a weighted graph gives a 2-factor such that the total sum of

the edge weights present in the 2-factor solution is minimized. An optimal 2-factor, if it exists, can be found in polynomial time [5].

An extension of 2-matchings are b-matchings. For a weighted graph $G = (V, E)$ the *minimum weight perfect b-matching problem* asks for a subgraph $H = (V, E_0)$ such that every vertex $v \in V$ is incident with exactly b_i edges where b_i is the i-th entry of a vector $b \in \mathbb{R}^V$, and the sum of the edge weights is minimized. This b-matching problem can be solved in polynomial time [5].

We now briefly review the method presented in [6] for mixed signed genomes. Let A, B, and C be three genomes with mixed chromosomes. To show that the multichromosomal BMP for signed mixed genomes is polynomial, the authors of [6] build a graph G with the following vertices: Add a vertex for each gene extremity (head or tail). Add a supplementary vertex t_x for each gene extremity x. For each pair of gene extremities x and y from different genes, add an edge xy to the graph such that the weight of xy is equal to $adj(xy)$, the number of times x and y are adjacent in each of the three genomes. For each vertex x corresponding to a gene extremity, add an edge between x and t_x weighted by half the number of times x is a telomere in each of the three genomes. Add an edge of weight 0 between every pair t_x and t_y. In [6] it is shown that the maximum weight perfect matching on this graph has a weight equal to the minimum median score. For the case of circular genomes, the authors suggested to restrict the graph G to the extremities to constrain the results for the circular signed genomes.

The result presented in [6] serves as a proof of the mixed multichromosomal BMP for signed genomes to be polynomial. To the best of our knowledge, our research is the first study of mixed and circular multichromosomal BMP for unsigned genomes. We prove that these problems are polynomial.

3.3 Multichromosomal BMP for circular unsigned genomes

We will reduce the multichromosomal BMP for circular unsigned genomes into a minimum weight 2-matching problem, thus showing that an optimal solution can be found for it in polynomial time.

Let A, B, and C be three multichromosomal circular genomes on the same set of n genes. The *BP-median graph* G ([6] and [2]) is defined to be a complete weighted graph where each vertex represents one gene, and the weight of each edge uv is $w(uv) = 3 - adj(uv)$ where $adj(uv)$ is the number of times the two genes corresponding to the vertices u and v are adjacent in each of the three genomes.

Theorem 1. *An optimal solution to the multichromosomal BMP for circular unsigned genomes can be found in polynomial time.*

Proof. Let A, B, and C be three multichromosomal circular unsigned genomes on the same set of n genes. Construct the BP-median graph G. Let M be an optimal solution to the perfect 2-matching problem on G. Such an optimal solution consists of a set of disjoint cycles (some possibly of size 2) covering all vertices of the graph such that the total cost of the edges present in M is maximized. Let $M = \{C_1 \cup C_2 \cup \ldots \cup C_j\}$ where each C_i $(1 \leq i \leq j)$ is a cycle and for all $v \in V$ there exists exactly one C_i such that $v \in C_i$.

We construct a solution to the multichromosomal circular BMP corresponding to M such that each cycle C_i in M corresponds to one circular chromosome in the median, i.e. the vertices of each cycle appear in the same order as their corresponding genes in the circular median. We claim such a median is an optimal one. Let uv be an edge in C_i, for any $1 \leq i \leq j$. Based on our construction, $w(uv) = 3 - adj(uv)$. So uv contributes exactly $w(uv)$ to the median score $d(M, A) + d(M, B) + d(M, C)$ which we are trying to minimize. This is equivalent to $w(uv)$ as it represents the number of

times u and v are not adjacent in three genomes A, B, and C.

Since an optimal 2-matching minimizes the total sum of the edge weights present in M, this solution also minimizes the median score, and thus provides us with an optimal solution for the multichromosomal BMP for the three circular genomes.

On the other hand, let M be an optimal solution to the multichromosomal BMP on three circular genomes A, B, and C. Suppose M consists of j circular chromosomes $\{C_1, C_2, \ldots, C_j\}$ such that each gene belongs to exactly one circular chromosome. Let u and v be two adjacent genes in the optimal median M. Then, $w(uv) = 3 - adj(uv)$ represents the number of times u and v are not adjacent in the three genomes. If uv is in the optimal median M, it contribute $w(uv)$ to the median score. Construct the BP-median graph G. We claim that each circular chromosome in the median M corresponds to a cycle in the optimal 2-matching for the BP-median graph G. For two adjacent genes u and v in M, the edge $w(uv)$ is in exactly one of the circular chromosomes the optimal median M, corresponding to exactly one cycle in the BP-median graph G, and vice-versa. Since an optimal solution M to the median problem minimizes the median score, it also minimizes the sum of the edge weights $w(x_i y_j)$ for each edge $x_i y_j$ that belongs to exactly one $C_i \in \{C_1, C_2, \ldots, C_j\}$. Thus, $\{C_1, C_2, \ldots, C_j\}$ corresponds to a solution for the minimum weight 2-matching problem on the BP-median graph G.

So, there is a one-to-one correspondence between the solutions of the multichromosomal circular BMP and the constructed 2-matching problem. We know that an optimal solution to the 2-matching problem can be found in $O(n^3)$ time. Therefore, an optimal solution to the multichromosomal BMP for circular genomes can also be found in $O(n^3)$ time. □

It should be noted that the above construction is valid for the median of any k genomes, $k \geq 3$. We only need to modify the edge weights to $w(uv) = k - adj(uv)$ instead of $w(uv) = 3 - adj(uv)$, and the rest follows.

3.3.1 Circular unsigned BMP with size restrictions

So far, we have studied the problem without any restrictions in terms of the size of the solution, i.e. there are no restrictions on the number of circular chromosomes of the median. In this section, we will introduce this restricted version of the problem, and study the problem of finding medians with a given number of circular chromosomes.

Note that in general, there are no restrictions on the number of cycles in an optimal solution to the 2-matching problem. Therefore, there will be no restrictions on the number of potential circular chromosomes found for the corresponding median.

We introduce the idea of finding a k-chromosome median. A *k-chromosome median* of a set of genomes is a multichromosomal median that has k chromosomes (circular or linear). We proved that the multichromosomal circular BMP can be solved in polynomial time, however, as soon as the restriction of having exactly k chromosomes for the median is introduced, the problem becomes NP-hard.

Theorem 2. *Finding a k-chromosome breakpoint median for circular (signed or unsigned) genomes is NP-hard.*

Proof. If $k = 1$, the statement turns into the usual BMP where we are looking for a median with one chromosome. This is an NP-hard problem in both linear and circular, signed and unsigned cases ([1] and [4]). Since the statement is NP-hard for the reduced version of $k = 1$, it is NP-hard for the general problem. □

3.4 Multichromosomal BMP for mixed unsigned genomes

Next, we focus on solving the multichromosomal BMP for mixed unsigned genomes. In this case, all of the given genomes A, B and C, and the median M may contain both circular and linear chromosomes. Once again, we will provide a transformation

from the multichromosomal BMP for mixed unsigned genomes to a minimum weight 2-matching problem. Note that since it is possible to have linear chromosomes, we must consider a transformation suitable to handle the telomeres.

Let A, B, and C be three multichromosomal mixed unsigned genomes on the same set of n genes. We start by constructing a BP-median graph, similar to the case for circular genomes. So let the BP-median graph G_1 be a complete weighted graph on n vertices where each vertex represents one gene, and the weight of each edge uv is $w(uv) = 3 - adj(uv)$ where $adj(uv)$ is the number of times the two genes corresponding to the vertices u and v are adjacent in each of the three genomes. We call these edges to be *adjacency-type* edges. Let G_2 be another complete weighted graph on n vertices such that for each vertex $v_i \in V(G_1)$ there is a corresponding vertex $t_{v_i} \in V(G_2)$, and all edge weights in G_2 are equal to zero. Next add an edge between each pair of vertices v of G_1 and the corresponding t_v of G_2. The weight of an edge vt_v is equal to $\frac{3-\tau(v)}{2}$, where $\tau(v)$ is the number of times v is a telomere in any of the three genomes A, B, and C. We call such edges vt_v *telomere-type* edges, and the graph on $2n$ vertices obtained by this construction to be the *mixed BP-median graph*. We can think of the mixed BP-median graph G as the BP-median graph G_1 and its mirror image G_2 with the n added edges between the two subgraphs.

Theorem 3. *An optimal solution to the multichromosomal BMP for mixed unsigned genomes can be found in polynomial time.*

Proof. Let A, B, and C be three mixed unsigned genomes on the same set of n vertices, and let G be the corresponding mixed BP-median graph. We will show that for every solution to the minimum weight 2-matching problem on G, there exists a corresponding median of A, B, and C such that the weight of the 2-matching is equal to the median score, and vice-versa. Therefore, an optimal solution to the minimum weight 2-matching problem on the mixed BP-graph provides an optimal solution to the multichromosomal BMP for the mixed unsigned genomes.

Let P be a solution to the minimum weight perfect 2-matching problem on G. We first show how to obtain a median M from P of the same cost. Since P is a perfect 2-matching, it consists of a set of disjoint cycles (some possibly of size 2) covering all vertices of G. Remove all vertices of the form t_v (the vertices that belong to G_2, the mirrored image of G_1) along with all edges that have such vertices as an endpoint. Let P' be the set of edges left from P after this removal. So P' forms a set of disjoint paths and cycles in G_1 that cover all the vertices. Let M be the median obtained by forming circular chromosomes from the cycles and linear chromosomes from the paths in the obvious way. We claim that the median score of M is the same as the cost of P.

To see this, first consider an edge uv that belongs to the subgraph G_1 in the mixed BP-median graph G. If we use the edge uv in the minimum weight 2-matching solution, uv will remain in P'. This corresponds to u and v being adjacent in the final BMP solution M, and the minimum weight 2-matching solution contributes exactly $w(uv) = 3 - adj(uv)$ to the BMP solution cost $d(M, A) + d(M, B) + d(M, C)$, which is equal to the median score.

Next, consider an edge vt_v such that $v \in G_1$ and $t_v \in G_2$. The weight $w(vt_v) = \frac{3-\tau(v)}{2}$ represents half the number of times v is not a telomere in A, B, or C. If this edge is present in the 2-matching solution P, this corresponds to gene v being a telomere in the corresponding BMP solution M which contributes exactly $w(uv) = \frac{3-\tau(v)}{2}$ to the BMP solution value of $d(M, A) + d(M, B) + d(M, C)$. All other edges in P not accounted for yet have weight 0, therefore, the weight of P equals the median score of M. Therefore, the optimal solution to the minimum weight perfect 2-matching problem on the mixed BP-median graph G provides an optimal median for the genomes A, B, and C.

Conversely, we show how a median M can be transformed to a perfect 2-matching solution P on the corresponding mixed BP-median graph G such that the median score of M equals the weight of P. We will show how the median M traces out cycles

of a perfect 2-matching P in G. Each circular chromosome $C_1 = (v_i \dots v_j)$ in M corresponds to a cycle C_1 in G_1 which we include in P. For each linear chromosome $L = |v_i \dots v_j|$ in M, take the corresponding path in G_1, and add the edges $v_i t_{v_i}$, $v_j t_{v_j}$, and $t_{v_i} t_{v_j}$ to form a cycle C_L in G. Finally, let T be the set of vertices t_i that belong to G_2 and are not yet present in any cycles. Let C_T be the cycle consisting of all vertices of the set T exactly once. We know that such a cycle exists since G_2 is a complete graph. Moreover, since all the edges in G_2 have weights equal to zero, the sum of the edge weights of C_T is also equal to zero. We know that now we have obtained disjoint cycles that cover all vertices of G exactly once. These cycles provide a perfect 2-matching P on G. Similar to above, we can show the median score of M is equal to the weight of P.

We showed that every solution to the 2-matching problem on G provides us with the order of genes in a corresponding median M of A, B, and C, and vice-versa such that their costs are equal. Therefore, the optimal solution to the minimum weight 2-matching problem on the mixed BP-graph provides an optimal solution to the multichromosomal BMP for the mixed genomes.

We know that an optimal solution to a minimum weight perfect 2-matching problem on G can be found in $O((2n)^3) = O(n^3)$ time where n is the number of genes in the genome. Thus, an optimal solution to the multichromosomal BMP for mixed genomes can also be found in $O(n^3)$ time. □

3.5 Partially signed genomes

In some cases we may have incomplete data and be able to have the direction of only some of the genes and not all of them. Therefore, it is worthwhile to consider the possibility of having signed and unsigned genes present at the same time in our model. Also, from a theoretical point of view, we show that there are interesting connections between the median problem for partially signed genomes and some well-

known combinatorial objects. In this section we will explore such possibilities. We will define what we call partially signed genomes, and will show that the multichromosomal BMP for circular and mixed partially signed genomes is polynomial. To the best of our knowledge, this problem has not previously been studied, despite its practical usefulness.

Let Σ be the set of n genes where $S \in \Sigma$ is the subset of signed genes and $U \in \Sigma$ is the subset of unsigned genes. Let A, B, and C be three genomes on the set Σ. For example, $A = (-1\ 5\ +2\ 4\ 3)$, $B = (3\ +1\ +2\ 5\ 4)$, and $C = (4\ -2\ -1\ 3\ 5)$ are three unichromosomal circular genomes with 5 genes $\Sigma = \{1,\ 2,\ 3,\ 4,\ 5\}$ such that $S = \{1,\ 2\}$ is the signed set and $U = \{3,\ 4,\ 5\}$ is the set of unsigned genes. Note that the signed genes may have different signs (+ or −) in each genome. The breakpoint distance can be defined similar to the previous sections. In the above example, the breakpoint distance between B and C is $d(B, C) = n - adj(B, C) = 5 - 3 = 2$ and the number of breakpoints between B and C is also 2. The breakpoint median problem on partially signed genomes is defined in a similar fashion.

We will show that the construction presented in the previous section can be easily modified to handle partially signed genomes.

3.5.1 BMP for circular partially signed genomes

Let A, B, and C be three circular partially signed genomes. Our aim is to find a solution to the breakpoint median problem for these circular partially signed genomes. We assume that if a gene is signed in A, B, and C, then it will be signed in the median M, and vice-versa. Same assumption holds for the unsigned genes. Without this assumption, the definition of the breakpoint distance would not be clear. We construct a *partially signed BP-median graph* G (PS-BP-median), a weighted graph with the following vertices: Add a vertex s for each $u \in U$, the set of unsigned genes. We call such a vertex a *regular* vertex. For each $s \in S$, the set of signed genes, add

two *extremity* vertices, s_t, and s_h representing the tail and the head of s. Add an edge between all pairs of vertices except between s_t and s_h, the tail and head of the same gene. The weight of each edge xy where x and y are regular or extremity vertices is $w(xy) = 3 - adj(xy)$. Note that here $adj(xy)$ can be determined regardless of x or y being a regular or an extremity vertex.

Both edges $g_h g_m$ and $g_m g_t$ have weights equal to zero. All the other edges with g_m as an endpoint have weights equal to infinity. Also, all edges between the head and the tail of the same gene, $g_h g_t$, have weights equal to infinity.

Theorem 4. *The multichromosomal BMP for circular partially signed genomes is polynomial.*

Proof. Let A, B, and C be three circular partially signed genomes, and let G be the corresponding PS-BP-median graph. We will define a perfect b-matching problem on G and show that for every perfect b-matching on G, there exists a corresponding median of A, B, and C such that the weight of the perfect b-matching is equal to the median score, and vice-versa. Therefore, an optimal solution to the perfect b-matching problem on the PS-BP-graph provides an optimal solution to the multichromosomal BMP for the circular partially signed genomes. We define a perfect b-matching problem on G as follows. Every vertex v in G is assigned a value b_v such that for every regular vertex v, $b_v = 2$, and for every extremity vertex v, $b_v = 1$. Now let R be any perfect b-matching on G. Therefore, the number of edges in R incident with a vertex v of G is exactly b_v. Note that R consists of a set of disjoint cycles and paths covering all vertices of G. Let C be the set of all cycles and P be the set of all paths in R. Note that the ends of the paths in R are exactly the extremity vertices. For each pair of extremities s_t and s_h add the edge $s_t s_h$ with weight 0. Adding these edges turns all the paths in P into cycles (each cycle may contain more than one of the paths). Let C' be the set of cycles obtained in this way.

Let M be the median obtained by forming circular chromosomes from the cycles

of C and C' in the obvious way. We claim that the median score of M is the same as the cost of P. Consider an edge uv that belongs to the PS-BP-median graph G. Assume we use the edge uv in the b-matching solution R. This corresponds to u and v being adjacent in the final BMP solution, and the b-matching solution contributes exactly $w(uv) = 3-adj(uv)$ to the BMP solution cost $d(M, A)+d(M, B)+d(M, C)$, which is equal to the median score. All added edges (the edges between two extremities) have weight 0, and hence their addition does not affect the median score of M. Therefore, the optimal solution to the perfect b-matching problem on the PS-BP-median graph G provides an optimal median for the circular partially signed genomes A, B, and C.

Similarly, we can show how any median M can be transformed to a perfect b-matching solution R on the corresponding PS-BP-median graph G such that the median score of M equals the weight of R.

We showed that the multichromosomal BMP for circular partially signed genomes can be transformed into a perfect b-matching problem on the PS-BP-median graph with the corresponding weight as the median score, and vice-versa. Since the perfect b-matching problem can be solved in polynomial time [5], the multichromosomal BMP for circular partially signed genomes is polynomial. □

3.5.2 BMP for mixed partially signed genomes

In this section, we will study the breakpoint median problem for multichromosomal mixed partially signed genomes, i.e. we are allowed to have circular and linear genomes in our solution. We will show that this problem is polynomial. The overall method is similar to the method for circular partially signed genomes, except that we allow linear genomes.

Let A, B, and C be three mixed partially signed genomes. Once again, we assume that if a gene is signed in A, B, and C, then it will be signed in the median M, and

vice-versa. The same assumption holds for the unsigned genes.

Theorem 5. *The multichromosomal BMP for mixed partially signed genomes is polynomial.*

Proof. Let A, B, and C be three mixed partially signed genomes, and let G_1 be the corresponding PS-BP-median graph with n vertices. We refer to the edges of G_1 as *adjacency-type* edges. Let G_2 be a complete weighted graph on n vertices such that for each vertex $v_i \in V(G_1)$ there is a corresponding vertex $t_{v_i} \in V(G_2)$, and all edge weights in G_2 are equal to zero. Next add an edge between each pair of vertices v of G_1 and the corresponding t_v of G_2. The weight of an edge vt_v is equal to $\frac{3-\tau(v)}{2}$, where $\tau(v)$ is the number of times v is a telomere in any of the three genomes A, B, and C. We call such edges vt_v *telomere-type* edges, and the graph G on $2n$ vertices obtained by this construction to be the *mixed PS-BP-median graph*.

We will define a b-matching on G and show that for every solution to this b-matching problem on G, there exists a corresponding median of A, B, and C such that the weight of the b-matching is equal to the median score, and vice-versa. Therefore, an optimal solution to the b-matching problem on the mixed PS-BP-graph provides an optimal solution to the multichromosomal BMP for the mixed partially signed genomes.

Define a b-matching on G as follows. For vertices in G_1, let every regular vertex v have a b_v value of 2, and every extremity vertex v_h or v_t have a b_v value of 1. For all vertices v in G_2, let the b_v value be equal to 2. Let R be a solution to this b-matching on G. Therefore, the number of edges in R incident with a vertex v of G is exactly b_v. We first show how to obtain a median M from R of the same weight. Note that R consists of a set of disjoint cycles and paths covering all vertices of G. Remove all vertices of the form t_v (the vertices that belong to G_2) along with all edges that have such vertices as an endpoint. Let R' be the set of edges left from R after this removal. So R' forms a set of disjoint paths and cycles in G_1 that cover all the vertices. Now

we add an edge of weight 0 between each pair of vertices v_t and v_h, corresponding to the tail and the head of the same gene ("glue" the head and the tail of each signed gene together). Let G' be the obtained graph. Let M be the median obtained by forming circular chromosomes from the cycles in G' and linear chromosomes from the paths in G' in the obvious way. We claim that the median score of M is the same as the cost of R.

To see this, first consider an edge uv that belongs to the subgraph G_1 in the mixed PS-BP-median graph G. If we use the edge uv in the perfect b-matching solution, uv will remain in G'. This corresponds to u and v being adjacent in the final BMP solution M, and the perfect b-matching solution contributes exactly $w(uv) = 3 - adj(uv)$ to the BMP solution cost $d(M, A) + d(M, B) + d(M, C)$, which is equal to the median score.

Next, consider an edge vt_v such that $v \in G_1$ and $t_v \in G_2$. The weight $w(vt_v) = \frac{3-\tau(v)}{2}$ represents half the number of times v is not a telomere in A, B, or C. If this edge is present in the perfect b-matching solution R, this corresponds to gene v being a telomere in the corresponding BMP solution M which contributes exactly $w(uv) = \frac{3-\tau(v)}{2}$ to the BMP solution value of $d(M, A) + d(M, B) + d(M, C)$. All other edges in R not accounted for yet, and all edges added later by gluing the head and tails have weight 0, therefore, the weight of R equals the median score of M. Therefore, the optimal solution to the perfect b-matching problem on the mixed PS-BP-median graph G provides an optimal median for the genomes A, B, and C.

Conversely, we show how a median M can be transformed to a perfect b-matching solution R on the corresponding mixed BP-median graph G such that the median score of M equals the weight of R. We will show how the median M traces out cycles and paths of a perfect b-matching R in G. Each circular chromosome $C_1 = (v_i \dots v_j)$ in M where C_1 does not contain any signed gene, corresponds to a cycle C_1 in G_1 which we include in R. If a circular chromosome $C_1 = (v_i \dots v_j)$ in M where C_1 contains any unsigned gene $v_k = v_{k_h} v_{k_t}$, then v_k corresponds to two non-adjacent

extremity vertices in G_1. In G, this corresponds to a set of paths in G, by following the vertices corresponding to the entries in C_1 in their circular order, but skipping from v_{k_h} to v_{k_t} (where there is no edge) for every signed gene v_k. This matches the requirement for the b-matching of having $b_v = 1$ for the extremity vertices and $b_v = 2$ for the regular vertices in G_1. Now consider a linear chromosome $L_1 = |v_i \dots v_j|$ in M. In G_1, take the same set of edges for L_1 as described for C_1, except for the telomeres v_i and v_j of L_1, add the edges $v_i t_{v_i}$, $v_j t_{v_j}$, and $t_{v_i} t_{v_j}$. Finally, let T be the set of vertices t_i that belong to G_2 and are not present in the cycles and paths obtained above. Let C_3 be the cycle covering all vertices of the set T exactly once. This assigns a b_v of 2 to all such vertices. We know that such a cycle exists since G_2 is a complete graph. Moreover, since all the edges in G_2 have weights equal to zero, the sum of the edge weights of C_3 is also equal to zero. We have now obtained disjoint cycles and paths that cover all vertices of G exactly once with corresponding b_v values for the perfect b-matching. Similar to before, the median score of M is equal to the weight of the perfect b-matching solution R.

We showed that an optimal solution to the b-matching problem on G corresponds to an optimal solution to the median problem, with equal costs for the b-matching and the median score. We know that b-matching can be solved in polynomial time [5], therefore, the multichromosomal BMP for mixed partially signed genomes is polynomial. □

3.6 Conclusion

In this paper, we settled the complexity of the multichromosomal breakpoint median problem for unsigned genomes where circular or mixed (circular and linear) chromosomes are allowed. We provided polynomial-time algorithms for these problems, thus proving that these are polynomial-time problems. These algorithms were obtained by transforming these problems into instances of well-known polynomial-time problems

in combinatorial optimization.

In addition, we introduced the idea of k-chromosome median as a median genome containing k chromosomes. We showed that finding a k-chromosome breakpoint median is NP-hard.

We then introduced partially signed genomes, where we can have both signed and unsigned genes in the same genome. We proved that the breakpoint median problem for circular or mixed partially signed genomes is polynomial. We proved this by providing a transformation into b-matchings, taking advantage of an innovative construction of the PS-BP-median graph.

In addition to answering the theoretical complexity problems, in this paper we provided several new ideas on how various matching problems can be manipulated to obtain results for median problems. From a computational point of view, the most significant advantage of these transformations from median problems to matchings is the potential to use known methods for finding optimal 2-matchings and b-matchings in graphs to obtain optimal medians. Further empirical studies on the performance of such algorithms could be of interest.

From a theoretical point of view, typically, median problems are considered to be NP-hard. Therefore, its remarkable that the multichromosomal breakpoint median problem for circular and mixed genomes, whether unsigned, or partially signed, are all polynomial time problems.

Bibliography

[1] Bryant, D.: The complexity of the breakpoint median problem, technical report CRM-2579, Centre de Recherches de Mathematiques, Universite de Montreal, 1998.

[2] Haghighi, M. and Boyd, S.: A Fast Method for Large-scale Multichromosomal Median Problem, Proceedings of the ACM International Conference on Bioinformatics Computational Biology and Biomedicine ACM-BCB, Chicago, Il, 2011.

[3] Lovasz, L., and Plummer, M.D.: Matching Theory, volume 121 of North-Holland Mathematics Studies, Elsevier, 1986.

[4] Pe'er, I., and Shamir, R.: The median problems for breakpoints are NP-complete, Electronic Colloquium on Computational Complexity, technical reports 71, 1998.

[5] Schrijver, A., Combinatorial Optimization: Polyhedra and Efficiency, volume 24 of Algorithms and Combinatorics, Springer, 2003.

[6] Tannier, E., Zheng, C., and Sankoff, D.: Multichromosomal median and halving problems under different genomic distances, BMC Bioinformatics, 10:120, 2009.

Chapter 4

Approximation Algorithms for the Unichromosomal Breakpoint Median Problem

• Sylvia Boyd and Maryam Haghighi, Approximation Algorithms for the Unichromosomal Breakpoint Median Problem, Technical Report, University of Ottawa, 2011.

Work of Maryam Haghighi under the supervision of Sylvia Boyd. All results are joint work of Maryam Haghighi and Sylvia Boyd.

Abstract

We study several approximation algorithms for the unichromosomal breakpoint median problem. The best known result on approximation algorithms for the unichromosomal breakpoint median problem is by Caprara where he provides a $\frac{5}{3}$-approximation algorithm for the problem of finding the breakpoint median of k unichromosomal genomes. It is also known that there is a very simple idea that gives a $(2 - \frac{2}{q})$-approximation algorithm, which is a better performance ratio than Caprara's for the median of 3,4, and 5 genomes.

In this paper, we begin with a general shrinking method that may greatly reduce the size of the problem is first introduced. Next, we investigate some new ideas for the breakpoint median problem which may potentially lead to improved solutions in the future. We provide a novel polynomial-time approximation algorithm based on a cycle-patching idea for the breakpoint median problem of 3 genomes with the ratio of $\frac{7}{3}$. Although this ratio of the performance guarantee is inferior to the other ones, it has the advantage of having a very easy and intuitive interpretation, and a straightforward proof which potentially could be improved. We also present another original approximation algorithm for the breakpoint median of 3 genomes based on minimum spanning trees, which achieves a worst-case performance ratio of $\frac{5}{3}$.

4.1 Introduction

One of the main goals of genome rearrangements is the reconstruction of the evolutionary events between different species. The first step towards such constructions is usually the inference of a common ancestor of three or more organisms. There are many methods available for inference of the common ancestor from gene order data. Distance-based phylogenetic reconstructions are based on the notion of *distance*, for which the purpose is to capture how far apart two genomes are from each other. The common ancestor is often modeled using the notion of a *median*. There are several popular measures for distance, such as breakpoint, reversal, double cut-and-join, etc.. For measuring distance, we focus on breakpoint distance as it provides an intuitive link between the order of genes in each genome and how far two genomes could be from each other. We will study the problem of reconstructing the median from gene order data under the breakpoint distance. Here we only focus on the genomes with one chromosome, i.e. the unichromosomal genomes.

Consider two unsigned genomes $A = a_1, ..., a_n$ and $B = b_1, ..., b_n$ on the same set of n genes. The genes a_i and a_{i+1} are called *adjacent* and they represent one *adjacency* on A. If two genes g and h are adjacent in A but not in B (that is, gh or hg do not appear in B), then they determine a *breakpoint*. Here, we only consider what is sometimes referred to as *circular* chromosomes, i.e. we assume that in a genome $A = a_1, ..., a_n$, the last element a_n is adjacent to the first element a_1.

The *breakpoint distance* between two genomes A and B is equal to the number of breakpoints in A (or B). It can easily be shown that breakpoint distance is metric as follows. Consider each adjacency ab as an unordered set of two elements $\{a, b\}$. Let S be a multiset of all sets corresponding to the adjacencies of genome A. Let R be the corresponding multiset for genome B. Then the breakpoint distance between A and B is equivalent to the size of the symmetric difference between S and R [4]. Symmetric difference is metric. Therefore, breakpoint distance is metric.

An alternative way to calculate the breakpoint distance is by using $d(A, B) = n - a(A, B)$, where $a(A, B)$ represents the number of common adjacencies between genomes A and B.

Given a set G of k genomes, the *k-median problem* is the problem of finding a genome M, called *median*, that minimizes the total sum of the distances from all other genomes, i.e. the median is a genome M such that $\sum_{g_i \in G}(d(M, g_i),\ 1 \leq i \leq k)$ is minimized. The most common version of this problem is when $k = 3$, i.e. the median of three genomes. Wherever there is no confusion, we will use median problem instead of 3-median problem. If the distance measure used is the breakpoint distance, then the corresponding median problem is sometimes called the *breakpoint median problem*. The sum $\sum_{g_i \in G}(d(M, g_i),\ 1 \leq i \leq k)$ is often referred to as the *median score*, the value we are trying to minimize.

In this analysis, we only consider unichromosomal unsigned genomes where each gene is present exactly once in every genome. Each genome can be represented by a string of n integers (genes) $1, ..., n$. We refer to this version of the breakpoint median problem as BMP.

It has been shown that finding the breakpoint median of three (signed or unsigned) unichromosomal genomes is NP-hard ([2] and [13]). In 2000, Pe'er and Shamir gave a $\frac{7}{6}$-approximation for the signed 3-median problem [14]. In 2002, Caprara gave a $\frac{5}{3}$-approximation for the unsigned k-median problem [3]. For finding a point closest to q given points in a metric space, it is possible to get a $(2 - \frac{2}{q})$-approximation simply by taking the best among them [6]. This leads to a better approximation than Caprara's $\frac{5}{3}$ approximation for $q = 3, 4, 5$. In particular, it gives a $\frac{4}{3}$ approximation for $q = 3$.

To the best of our knowledge, the best result for the BMP in the general case of k-median is still the $\frac{5}{3}$-approximation by Caprara [3]. He formulates the problem as an integer linear program and defines a feasible dual solution for the relaxation of the LP. He then uses a very involved algebraic argument in an additive bounding approach

to prove the ratio of $\frac{5}{3}$. However, finding an intuitive interpretation of deriving the dual solution for his method proves to be non-easy. Also, Caprara showed that for his algorithm, the ratio is tight, even for $k = 3$, i.e. the approximation guarantee cannot be improved using his method. Therefore we are interested in finding other approximation algorithms for the BMP that provide a better bound, or at least have the possibility of improvement in their bound.

We will first discuss a general approach of shrinking that allows us to reduce the size of the genomes involved, and thus is very useful from the practical viewpoint of dealing with large genomes. This shrinking method can be applied on any data before applying any algorithm for solving the BMP. We give a detailed proof that shrinking does not affect the median, and therefore, the median found for the shrunk problem can be easily transferred to a median of the original problem.

We then discuss methods for improving the solutions found by the $(2 - \frac{2}{q})$-approximation which more fully take the relationship of the genomes into account. These methods are based on using solution improvement methods for the Traveling Salesman Problem [9].

Finally, we investigate some new avenues for the BMP problem. In Section 4.4.1 we will provide a novel polynomial-time approximation algorithm for BMP with the ratio of $\frac{7}{3}$. Although the ratio of the performance guarantee is not the best, the main advantages of this method are having a very easy and intuitive interpretation, and a straightforward proof which could potentially be improved and extended. This algorithm is based on a cycle-patching idea which is a direct and fast approach. We also introduce a second new approximation algorithm for the BMP in Section 4.4.2. This approximation algorithm is based on using minimum spanning trees and achieves the performance ratio of $\frac{5}{3}$ for BMP on three genomes. Again, it is possible that the ideas from this algorithm could lead to improved solutions in the future.

4.2 A shrinking approach

We will first study the properties of the complete weighted graph obtained from BMP with edge weights 0,1,2,3. Next we investigate the possibility of shrinking some of the edges of the graph.

4.2.1 BMP and relationships to TSP

Define the *breakpoint median graph* (BP-median graph) G for three genomes A, B, and C as follows. Let G be a complete graph of n vertices such that each vertex represents one gene. For each edge uv let $adj(uv)$ be equal to the number of times the genes corresponding to u and v are adjacent (do not form a breakpoint) in genomes A, B, and C (so $adj(uv)$ can be 0, 1, 2, or 3). Let the weight of the edge uv, $w(uv)$, be equal to $3 - adj(uv)$. Note that the BP-median graph can be easily generalized to k genomes, by replacing $3 - adj(uv)$ in the edge weights with $k - adj(uv)$. Unless specified, we will focus on the BP-median graph for three genomes.

In 1997 Sankoff and Blanchette [15] showed that the BMP can be reduced to the Traveling Salesman Problem (TSP) in the case of three unsigned circular genomes A, B and C over a set of n genes. They showed that the solution to the TSP for the BP-median graph G traces out an ordering of $\{1, 2, \ldots, n\}$ that provides an optimal solution to the BMP. In [16] the same authors used a branch-and-bound method to solve this TSP. Although popular in combinatorial optimization, branch-and-bound is not the fastest method for solving the TSP. For a weighted graph, an optimal solution to the TSP, calls for finding a minimum weight Hamilton cycle (i.e. a cycle that visits every vertex exactly once, and the sum of the edge weights on the cycle is minimized). An instance of the TSP is called *metric* if the edge weights of the graph satisfy the triangle inequality.

Since the goal of the TSP is to find a minimum cost Hamilton cycle, the solution given by the TSP optimizes the use of adjacencies available in the three genomes.

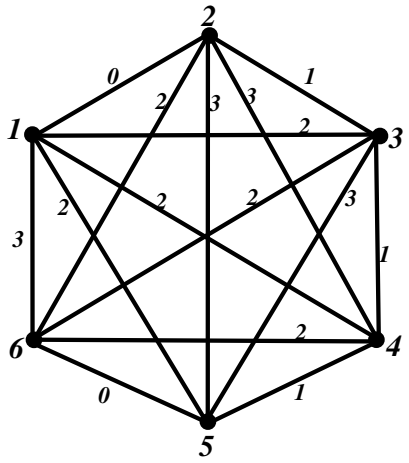


Figure 4.1: BP-median graph constructed for three circular genomes A = (1 2 6 5 4 3), B = (2 1 4 5 6 3), and C = (1 2 3 4 6 5). A breakpoint median is M = (1 2 3 6 5 4).

Hence, starting at any vertex of a cycle that is a TSP solution for the problem, we will have a genome that is the breakpoint median for the given three genomes. Figure 4.1 provides the graph for a simple example of three circular unichromosomal genomes A = (1 2 6 5 4 3), B = (2 1 4 5 6 3), and C = (1 2 3 4 6 5). The edge weights are assigned by using $w(uv) = 3 - adj(uv)$. It can be seen from the example that an optimal TSP solution is 1236541 with the minimum cost equal to 6. Therefore, a breakpoint median for this problem is M = (1 2 3 6 5 4), and $d(M, A)+d(M, B)+d(M, C) = 6$. As can be seen in this example, the TSP defined for the breakpoint median problem is not a metric TSP, that is, the triangular inequality is not valid for the weights of the edges in TSP (for example, look at the edges between vertices 4, 5, and 6).

The TSP is known to be NP-hard. Currently the best c-approximation algorithm known for the metric symmetric TSP is the algorithm due to Christofides [5] for which $c = \frac{3}{2}$. However, for the general non-metric TSP, a c-approximation algorithm for

any constant c would imply P=NP, and thus it is considered highly unlikely to exist [9].

This should not be discouraging for our problem, as the non-metric TSP instance for BMP is a very specific instance with special weights on the edges, namely a TSP with weights 0,1,2,3. Therefore, we first investigate the properties of the underlying graph. By doing so, we are able to develop a shrinking method which potentially greatly reduces the size of the problem. Note that for the case of specific edge weights, there is one approximation result of which we are aware. For TSP with weights 1 and 2, Papadimitriou and Yannakakis [12] present a $\frac{7}{6}$-approximation algorithm. They also showed that this special case of the TSP is MAX SNP-hard (implying that it is unlikely that a polynomial-time approximation algorithm exists whereby any ratio can be achieved, i.e. there is a limit to how close we can get to 1). Since TSP with weights 1 and 2 is embedded in TSP with weights 0,1,2, and 3, their result on complexity also holds for our problem. Note that a TSP problem with weights 1 and 2 is metric, whereas our problem is not, indicating it is in a harder class of problems to approximate.

4.2.2 Properties of the graph

As before, construct the BP-median graph G of n vertices (representing genes) such that the weight of the edge uv, $w(uv)$, is $3 - adj(uv)$ where $adj(uv)$ can be 0,1,2, or 3 depending on the number of times the genes corresponding to u and v are adjacent in the three circular genomes. The following lemma describes a characteristic of the vertices on any BP-median graph.

Lemma 4. *For each vertex v in a BP-median graph G, $\sum_{u \in V(G)} adj(vu) = 6$.*

Proof. For each gene v there are exactly two possible adjacent locations (to the left and right side of v). Since we have three genomes, there is a total of 6 possible

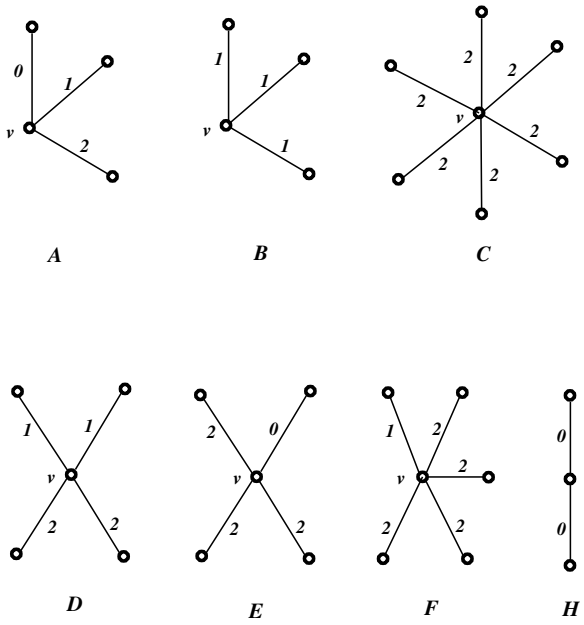


Figure 4.2: Possible edge weights at a vertex v of the BP-median graph. The edges of weight 3 are not shown.

adjacent locations beside each gene v. Therefore, for a vertex v in the graph we have $\sum_{u \in V(G)} adj(vu) = 6$. □

An immediate consequence of Lemma 4 is that there are not too many possible patterns of edge weights in the BP-median graph G. Thus, we have the following important theorem which will be used throughout the paper.

Theorem 6. *Any vertex v of a BP-median graph, must belong to one of the 7 classes shown in Figure 4.2, where the edges of weight 3 are not shown.*

Proof. The 7 classes for vertices of a BP-median graph G are direct results of Lemma 4 obtained by observing that an edge uv in G of weight 0,1,2, and 3 corresponds to having the $adj(uv)$ equal to 3,2,1, and 0, respectively. Since from Lemma 4 the sum of $adj(uv)$ at any vertex v must be 6, the only possibilities are those shown in Figure 4.2, where the edges of weight 3 are not shown. □

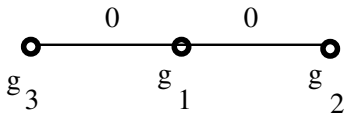


Figure 4.3: A saturated gene g_1

A *common subsequence* of length i of k genomes is a string of genes $g_1\ g_2\ \ldots g_{i+1}$ that appear in the exact same (or reverse) order in all k genomes. Each pair of consecutive genes g and h in a common subsequence lead to an edge gh of weight zero in the BP-median graph, and vice-versa. Therefore, there is a one-to-one correspondence between paths of 0-edges in G and common subsequences of the 3 genomes.

Lemma 5. *If genomes A, B, and C are not all identical, then the zero edges form disjoint paths in the BP-median graph G.*

Proof. Suppose the statement is false. Therefore, there exists at least a cycle of 0-edges. Let $C = v_1\ v_2\ \ldots v_i\ v_1$ be such a cycle. So the genes corresponding to the vertices of this cycle must appear in the same (or reverse) order in all 3 genomes. That is we have a common subsequence $v_1\ v_2\ \ldots v_i$. Since v_iv_1 is also a 0-edge, v_iv_1 must appear in all 3 genomes. This results in either having more than one chromosome if the number of vertices is greater than i, (which is a contradiction to our assumption of having one chromosome per genome), or it implies that there are i vertices in total, i.e. all three genomes are identical, that is a contradiction. Therefore, there cannot be a cycle of 0-edges. □

We call a gene g_1 *saturated* if there exists two genes g_2 and g_3 such that $g_3g_1g_2$ appears in the same order (or reverse) in all three genomes. In the graph, a saturated gene corresponds to a vertex incident to two edges of weight 0 (see Figure 4.3). We have the following corollary, directly from Theorem 6.

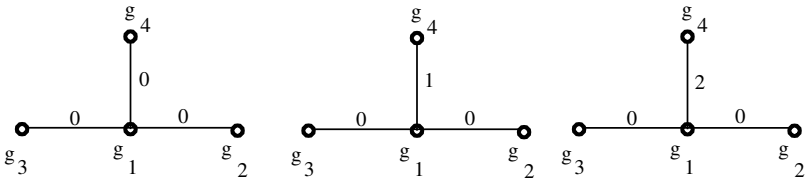


Figure 4.4: Impossible cases for a saturated gene g_1 in a BP-median graph

Corollary 1. *If a node g_1 is saturated, then there are exactly two edges incident to g_1 with weight 0 (the ones that made it saturated) and every other edge that has g_1 as an endpoint must have weight 3.*

It follows from Corollary 1 that none of the cases shown in Figure 4.4 can occur.

4.2.3 Reducing the problem by shrinking

Different variations of shrinking approaches for problems of combinatorial optimization have been studied in various contexts. In this section we study the effects of shrinking 0-edges in our graph. We first define what we mean by shrinking.

The main idea behind shrinking is that if a subsequence $P = g_1 g_2 ... g_k$ appears in the same (or exactly in reverse) order in all of the genomes, then we can prove (see Theorem 7 below) that a median must include this common subsequence. Therefore, we may replace this subsequence or *shrink* it to $P = g_1 g_k$. Equivalently, in the graph G, if there are consecutive vertices $v_1, v_2, \ldots, v_k$ such that they lie on a path of 0-edges, then we can *shrink* this path to $v_1 v_k$. During shrinking, we delete v_2,...,v_{k-1} and all of the edges that have v_2,...,v_{k-1} as an endpoint, and we add an edge $v_1 v_k$ of weight 0. Figure 4.5 shows an example of a graph before and after shrinking.

We can then find the median for this reduced version of the problem. To find the median, we can construct the BP-median graph G for the reduced sequences and find an optimal TSP tour for G. The optimal TSP tour of G gives the median for the

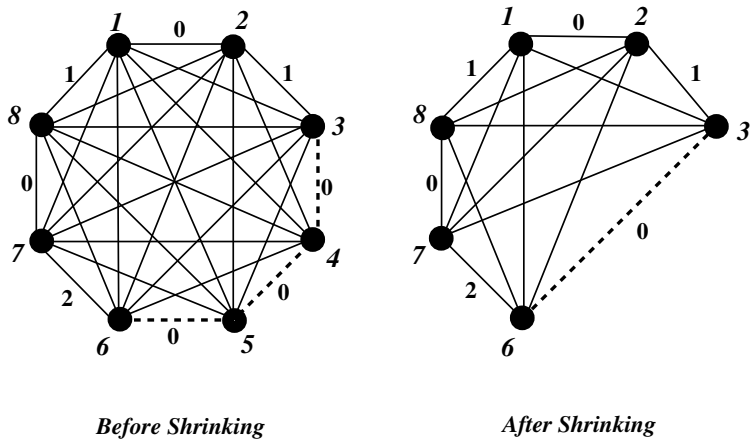


Figure 4.5: An example of the effect of shrinking the 0-edges on a BP-median graph. The three dashed edges with weight 0 are shrunk into one 0-edge. Edges with no weight shown have weight greater than 0.

reduced problem. Once the median for this reduced problem is found, we unshrink the sequences back to find the median of the original problem. Although, intuitively this operation may seem logical, it does not have an obvious mathematical proof. Here we present a detailed proof that shows shrinking works. We first show that shrinking a common subsequence of size 2 (a path of size 2) to one 0-edge will not affect the median. Then we prove by induction that the result is valid for a common subsequence of any size.

Lemma 6. *Suppose there is a common subsequence uvw that appears in the same order (or reverse) in all three genomes. The process of shrinking this subsequence to an edge uw of weight 0 will not affect the median.*

Proof. Suppose uvw is a common subsequence. We basically prove the lemma by proving that there is an optimal tour for G that uses uvw. Proof by contradiction. Suppose not, that is, let T be the optimal tour and assume T does not use uvw, and any tour T' that does is not optimal. We have two cases:

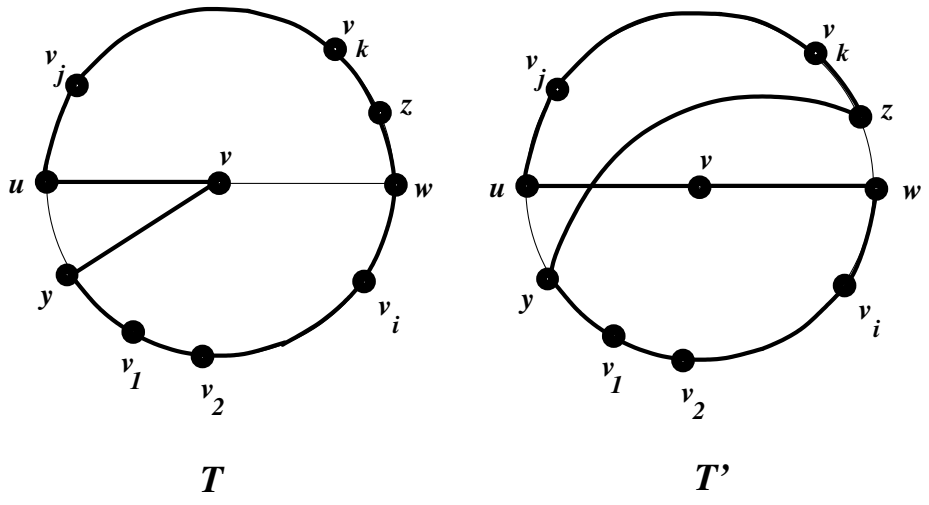


Figure 4.6: Case 1: if only uv is used

Case 1: T uses exactly one of the zero edges uv or vw. Without loss of generality, suppose T uses uv of weight zero, $T = uvyv_1v_2...v_iwzv_k...v_ju$. Then, let T' be the tour shown in Figure 4.6, that is,

$T' = uvwv_iv_{i-1}...v_2v_1yzv_k...v_ju$.

Then, cost(T') = cost(T) + cost(yz) + cost(vw) - cost(wz) - cost(vy).

By Corollary 1 every edge incident to v has weight 3 except uv and vw which have weights 0. Therefore,

cost(T') = cost(T) + cost(yz) - cost(wz) - 3.

Since T is an optimal tour and T' is not optimal by assumption, we must have that cost(T') > cost(T). Therefore, cost(yz) > cost(wz) + 3. Considering that the weights are $0, 1, 2$ or 3, this gives a contradiction.

Case 2: T uses neither of the zero edges. Let $T = u...bw...zvy...au$ and let T' be a (not optimal) tour such that $T' = u...ba...yz...wvu$ (see Figure 4.7).

Then,

cost(T')= cost(T) - cost(yv) - cost(vz) + cost(yz) +cost(uv) + cost(vw) - cost(wb) - cost(ua) + cost(ab)

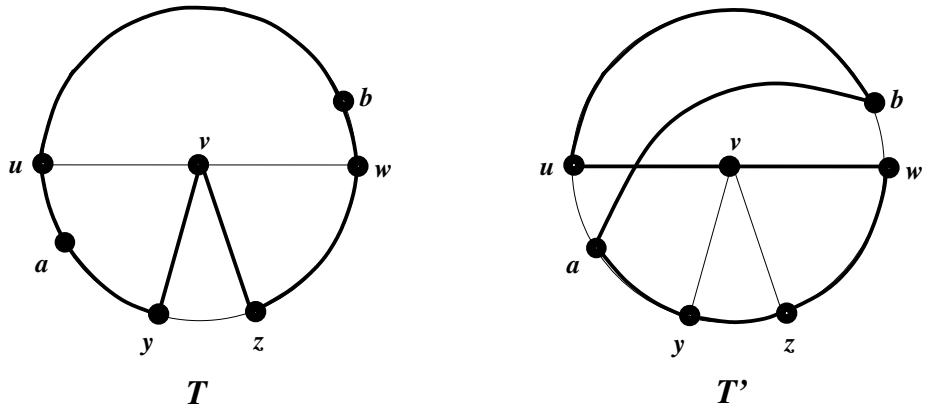


Figure 4.7: Case 2: if uvw is not used

= cost(T) - 3 -3 + cost(yz) - cost(wb) - cost(ua) + cost(ab)

= cost(T) - 6 + cost(yz) - cost(wb) - cost(ua) + cost(ab)

> cost(T).

So cost(yz) + cost(ab) > cost(wb) + cost(ua) + 6.

Once again since the edge weights are $0, 1, 2, 3$, this gives a contradiction.

Therefore, there is an optimal tour in the graph that uses both of the zero edges uv and vw. We can shrink uvw to uw of weight 0, and find the optimal solution M' for this shrunk graph. Then unshrink uw back to uvw. Since the cost of uv and vw is zero in the original graph, this replacement will not affect the total cost and the optimality of the solution. Therefore, we can obtain an optimal solution M for the original graph, of the same cost as M'. There is a clear one-to-one correspondence between the optimal solutions to the original and the shrunk graph. □

We can extend the above lemma to a common subsequence of any size.

Theorem 7. *Suppose there is a common subsequence $uv_1v_2...v_iw$ that appears in the same order (or reverse) in all three genomes. The process of shrinking this subsequence to uw of weight 0 in the BP-median graph will not affect the median.*

Proof. Proof by induction on the number of vertices in the common subsequence.

Lemma 6 provides the basis of the induction. Suppose the statement is true for any common subsequence of length k, $k \geq 2$. We show that it will be true for a common subsequence of length $k+1$. Let $uv_1v_2...v_iw$ be a common subsequence of length $k+1$. Since $v_1v_2...v_iw$ is a common subsequence of length k, by the induction's hypothesis we can shrink it to v_1w without affecting the median. So let v_1w be the edge of weight 0 replacing $v_1v_2...v_iw$. Then we have a common subsequence of uv_1w such that uv_1 and v_1w are both of weight 0. Using Lemma 6, we can shrink this common subsequence to an edge uw of weight 0 without affecting the median. Therefore, the common subsequence $uv_1v_2...v_iw$ of length $k+1$ can be shrunk to uw of weight 0 without affecting the median. □

A *matching* in a graph is a set of edges such that no two edges have a vertex in common.

Lemma 7. *After shrinking all of the common subsequences, the 0 edges form a matching of the graph.*

Proof. This follows directly from Lemma 5. □

Lemma 8. *After shrinking all of the common subsequences, the value of a TSP tour in the graph G with n vertices is at least $\lceil n/2 \rceil$.*

Proof. By Lemma 7 the edges of value 0 form a matching in G. Therefore, at most half of the edges of a TSP tour can have an edge weight of 0, and the other half must have an edge weight of at least 1. Thus the value of a TSP tour cannot be less than $\lceil n/2 \rceil$. □

It may seem an obvious extension to try to shrink single 0-edges too. However, we do not allow shrinking of single 0-edges as it would result in losing structural information about the graph. If a single 0-edge uv is shrunk to a vertex v', it is not possible to unambiguously define the weights of the edges incident to v', and whether

those edges came from edges incident to u or incident to v; and therefore, we cannot unshrink the graph.

We must emphasize that typically considerable amount of data is in the form of common subsequences, and therefore, the shrinking method presented here could potentially lead to significant reduction in the size of the data that needs to be analyzed. Prior to this, for the case of reversal distance, some similar strategies on reducing the size of the data have been investigated and proved to be useful [7].

4.3 Improving the solutions obtained by the $(2-\frac{2}{q})$-approximation algorithm

It has been shown that taking the best permutations among q given permutations yields a $(2-\frac{2}{q})$-approximation algorithm [6]. This is based on a general result on finding the point closest to q given points in any metric space. This is an improvement over Caprara's $\frac{5}{3}$-approximation algorithm for the k-median problem for $k = 3, 4$, and 5. In particular, for the case of $q = 3$ this yields a $\frac{4}{3}$-approximation algorithm. We show how it can be combined with tour improvement methods for the TSP to obtain a $\frac{4}{3}$-approximation algorithm for the BMP on three unichromosomal genomes with improved solutions.

Figure 4.8 provides a general scheme of the relationships between the median and the three genomes A, B, and C. Recall that the breakpoint distance is metric, and thus the triangular inequality holds.

There is a strong well-known lower bound for the median score.

Lemma 9. *Let $S(M)$ be the median score for the median M of three given genomes A, B, and C, and a metric rearrangement distance d. Then*

$$\frac{d(A,B) + d(B,C) + d(A,C)}{2} \leq S(M).$$

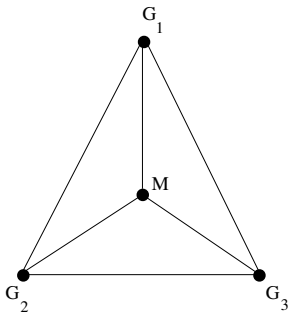


Figure 4.8: An example of a median

Proof. Using the triangular inequality,

$d(A,B)+d(B,C)+d(A,C) \leq d(A,M)+d(M,B)+d(B,M)+d(M,C)+d(A,M)+d(M,C) = 2S(M)$. □

The following provides a proof that taking the closest genome to the other two, among a set of three given genomes, gives a worst case ratio of $\frac{4}{3}$ for the 3-median problem.

Theorem 8. *For the 3-median problem with a metric distance d, taking the closest (to the other two) of the three genomes as the solution has a worst case ratio of* $\frac{4}{3}$.

Proof. Let A, B and C be three given genomes. Without loss of generality, let us assume that A is the closest to the other two, i.e. $d(A,B) \leq d(B,C)$ and $d(A,C) \leq d(B,C)$. Then, $d(A,B)+d(A,C) \leq 2\ d(B,C)$, or equivalently, $3\ d(A,B)+3\ d(A,C) \leq 2\ d(A,B) + 2\ d(A,C) + 2\ d(B,C)$.

On the other hand, by using the bound provided in Lemma 9, we have $2(d(A,B)+d(A,C)+d(B,C)) \leq 4S(M) = 4(d(A,M)+d(B,M)+d(C,M))$. By combining these inequalities we have

$$3\ d(A,B) + 3\ d(A,C) \leq 4(d(A,M) + d(B,M) + d(C,M)).$$

Then, since we take A as the median solution, we have the following for the median

score:

$$d(A,A)+d(A,B)+d(A,C) = d(A,B)+d(A,C) \leq \frac{4}{3}(d(A,M)+d(B,M)+d(C,M)).$$

□

Although this $\frac{4}{3}$-approximation method of taking the best among the given input is very easy and fast, it has some disadvantages. First of all, it does not take advantage of any of the given information about the input genomes. Moreover, in practice, it is sometimes needed to construct a median M that can be different from the given input.

As mentioned, a similar approach yields to a $(2-\frac{2}{q})$-approximation algorithm for the median of q genomes. This outperforms Caprara's $\frac{5}{3}$-approximation up to $q \leq 5$.

In order to be able to take advantage of the information within the order of genes in the given genome, and thus potentially improving the result obtained by this $(2-\frac{2}{q})$-approximation algorithm, we propose two methods, explained in the following sections. These methods combine the simple idea of "taking the best among the data" as the solution and then perform steps to achieve further improvements.

4.3.1 A 2-opt style improvement on the $\frac{4}{3}$ approximation algorithm

We start by explaining a well-known heuristic algorithm for TSP called *2-opt* [9]. This method is based on starting with a Hamilton cycle in the graph, i.e. a feasible solution (but not necessarily optimal) to the TSP. We then try to improve this tour T by a process called *2-interchange*. The 2-opt method takes every pair of non-adjacent edges e_1 and e_2 in T (so the two edges do not share a vertex), and deletes them. So T is broken into two paths, say T_1 and T_2. There is exactly one way that we can combine the two paths T_1 and T_2 to obtain a new tour T' (where $T' \neq T$). This is called 2-interchange. If the cost of T' is less than the cost of T, the 2-opt method replaces

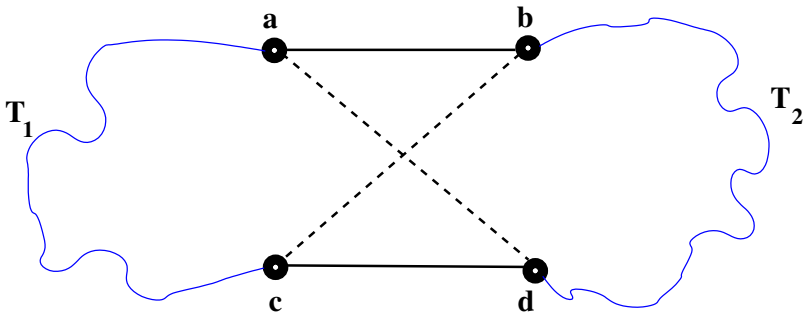


Figure 4.9: An example of a 2-interchange: the edges ab and cd (in bold) are replaced with the edges ad and bc (in dashes).

T with T' and repeats the same process of 2-interchange. If the cost of T' is greater than T for every pair of non-adjacent edges, then T can not be improved by the 2-opt method, and is called *2-optimal*. Figure 4.9 gives an example of a 2-interchange in a tour.

Recall that taking the best (closest to the other genomes) among the given set of q genomes yields a $(2 - \frac{2}{q})$-approximation algorithm. We propose the following algorithm to obtain a median of q genomes.

Algorithm 1: An improvement on the $(2-\frac{2}{q})$-approximation algorithm for the q-median problem based on 2-opt

1. Construct the BP-median graph.
2. Find the genome G_j among the q given genomes that has the minimum $\sum_{G_i}(d(G_j, G_i), 1 \leq i \leq q)$. Let $M = G_j$.
3. Find the tour T in the BP-median graph that corresponds to M.
4. Perform 2-interchange operations on T until it cannot be further improved.

Note that 2-opt can be performed in polynomial time, therefore Algorithm 1

provides a $\frac{4}{3}$-approximation for the 3-median problem. Also, as previously discussed, Algorithm 1 has the advantage that it guarantees to find a solution for the median problem that is at most $(2 - \frac{2}{q})$ away from the optimal. It therefore outperforms Caprara's $\frac{5}{3}$ result up to $q \leq 5$. Furthermore, it uses the information on the ordering of the genes in the input data to perform 2-interchanges and thus potentially improves the result from simply taking the best among the input as the solution.

4.3.2 A Lin-Kernighan style improvement

Lin and Kernighan [10] developed a variable k-opt heuristic algorithm for TSP which has been shown to work extremely well in practice. There has been several modifications to the original Lin-Kernighan algorithm (LK). For the details of the LK method and further developments we refer the reader to [9], [10], and [17]. We propose a heuristic that uses the LK method to obtain a median solution.

Algorithm 2: An improvement on the $(2-\frac{2}{q})$-approximation algorithm for the q-median problem based on LK

1. Construct the BP-median graph.
2. Find the genome G_j among the q given genomes that has the minimum $\sum_{G_i}(d(G_j, G_i), 1 \leq i \leq q)$. Let $M = G_j$.
3. Find the tour T in the BP-median graph that corresponds to M.
4. Perform LK heuristic to improve T.

Note that once again, Algorithm 2 provides a $\frac{4}{3}$ guarantee for the 3-median problem. The only disadvantage is that the LK heuristic is not a polynomial-time algorithm, and in fact could potentially not terminate. To avoid such a case, we can add a termination condition based on time, as this algorithm (and in general,

LK method) continually improves the best solution found so far. As mentioned, LK methods have been shown to perform extremely well in practice [9].

4.4 New approximation algorithms

4.4.1 A $\frac{7}{3}$ approximation algorithm based on cycle patching

Let $G = (V, E)$ be a weighted graph. The *minimum cost 2-factor problem* asks for a subgraph $H = (V, E')$ of G such that every vertex $v \in V$ is incident with exactly 2 edges and the sum of the edge costs is minimum. Every edge of G appears at most once in a 2-factor. A 2-factor is also called a disjoint vertex cycle cover. A solution to the 2-factor problem on a graph G gives a collection of cycles in G such that every vertex of G appears in exactly one cycle, while the total cost of the edges of the cycles is minimized. Note that every Hamilton cycle (a cycle that includes all vertices of a graph) and therefore every TSP tour (a minimum cost Hamilton cycle) is a 2-factor. But not every 2-factor is Hamiltonian, as there may be more than one cycle present in a 2-factor. Clearly, the cost of a 2-factor provides a lower bound for the optimal solution to the TSP. Note that for complete graphs there is always a 2-factor. The 2-factor problem has been studied to a great degree (for example see [17]), and an optimal (minimum cost) 2-factor, if it exists, can be found in polynomial time [17].

The authors of [12] gave a cycle-patching approach for the TSP with weights 1 and 2. By using this approach they first obtained an $\frac{11}{9}$-approximation, and then improved the ratio to $\frac{7}{6}$. Note that our graph may have edges of weight 0 and 3 as well as 1 and 2. Therefore, without any knowledge on the properties of our graph, we cannot directly use the cycle-patching method described in [12]; the minimum cost tour of a general graph with weights 0,1,2,3 can be of size zero. However, the results of Section 4.2 enable us to rule out this possibility. In fact, we use the lower bound we found in Lemma 8 and combine it with a cycle-patching method to derive a new

algorithm that works on our graph. Here we describe this approach.

Given a complete graph with weights 0, 1, 2, and 3 corresponding to a BMP, we first shrink all common subsequences (i.e. paths of 0-edges) according to Section 4.2. Let us suppose after all the shrinking is done, we are left with a graph that has n vertices. Start with an optimum 2-factor of this graph. Note this contains at most $\frac{n}{3}$ cycles. Consider any two cycles. Since the 0-edges are disjoint (Lemma 7), every cycle must contain at least one edge of value greater than zero. We can patch two cycles together by using these non-zero edges. In the first patching, two edges of weight at least 1 get replaced by two edges of weight at most 3. So there is a maximum increase in cost of 4. From now on there is at least one edge of weight 3 in a cycle (otherwise, without loss of generality, just increase the cost of the edge that was recently added to the cycle to 3). Then by using this edge in the patching, the maximum increase in the cost is 2 for each patching (see Figure 4.10). We continue the patching one after another until a Hamilton cycle T is formed. If we let opt(TSP) represent the cost of an optimal solution, using Lemma 8 we have

Total cost of T $\leq$ cost of 2-factor + 4 + 2 $(\frac{n}{3} - 2)$

$\leq$ cost of 2-factor + $\frac{2n}{3}$

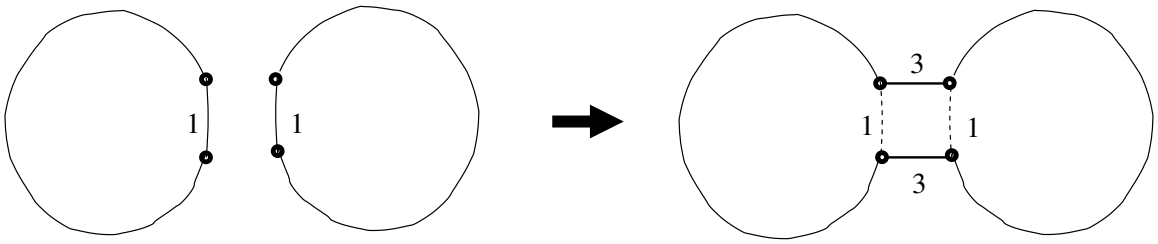
$\leq$ opt(TSP) + $\frac{4}{3}$ $(\frac{n}{2})$ $\leq$ opt(TSP) + $\frac{4}{3}$ opt(TSP) $\leq$ $\frac{7}{3}$ opt(TSP).

The running time of this algorithm is dominated by the running time of finding an optimal 2-factor which is currently $O(n^{2.5})$ [11].

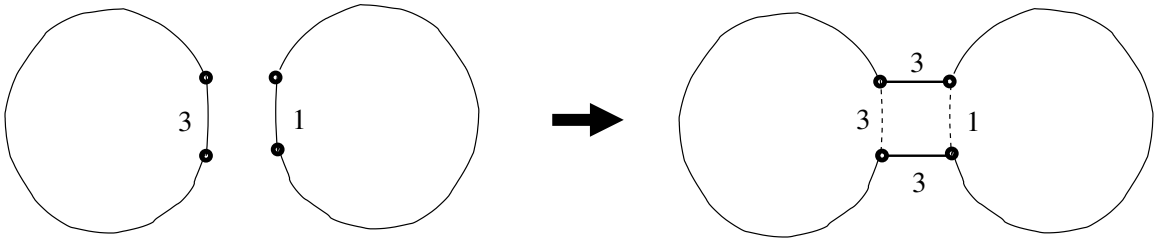
Note that this algorithm can easily be generalized to the median problem for $L > 3$ genomes, i.e. edge weights 0,1,...,L. In such case, the first patching will cost at most 2L-2 and every other patching will cost at most L-1. So for the TSP tour T found we have

Total cost of T $\leq$ cost of 2-factor + (2L - 2) + (L-1)$(\frac{n}{3}$ - 2)

$\leq$ cost of 2-factor + $\frac{2(L-1)}{3}$ $(\frac{n}{2})$ $\leq$ $\frac{2L+1}{3}$ opt(TSP).



First patching



After the first patching

Figure 4.10: cycle-patching

Improvements and extensions

One possible improvement to this algorithm is to start with a 2-factor that has less than $\frac{n}{3}$ cycles. In fact, the upper bound of $\frac{n}{3}$ on the number of cycles in the graph is based on the worst case scenario that each cycle has 3 vertices. If there is a way to improve this bound, i.e. to start with larger cycles in the 2-factor, then the performance of the cycle-patching can be improved. This relates to the problem of finding triangle-free 2-factors in a graph. Authors of [12] mentioned that they used results from [8] for finding triangle-free 2-factors and therefore proposed a $\frac{7}{6}$-approximation for TSP with weights 1 and 2. However, they did not provide a proof of their method. In 2009, the authors of [1] resolved this issue and proved that the claim in [12] is indeed true. So far, all of the results in other works have been focused on TSP with weights 1 and 2. If the triangle-free 2-factor can also be obtained for TSP with 0,1,2,3 weights, then our algorithm could achieve a better ratio (a ratio of 2 instead of $\frac{7}{3}$).

Another possibility for future work is performing a more complex patching such that the cost at each step is less, as is done in [12], although this appears to be difficult when weight 0 edges are involved. We have been able to extend our approach presented in this paper and find an approximation ratio of $\frac{13}{9}$ for most cases (which would result in an improvement over the $\frac{5}{3}$ ratio found by Caprara [3]), however, one case involving 0 edges is proving to be challenging.

It should be noted that the $\frac{7}{3}$ bound we have given is only an upper bound and it is not necessarily tight. It is based on the assumption that the lowest weight edges only exist in the 2-factor, and every other edge used in the cycle-patching will be expensive. The assumption we made about the increase of 2 in the cost for each patching (after the first one) is only a worst-case analysis. In reality, if there are edges of weight larger than 1 in both cycles, each patching would cost less, and thereby, the approximation can result in a better solution. We expect that our algorithm would

perform very well on real data and we are in the process of performing empirical computational studies to test the actual performance of this algorithm.

4.4.2 A $\frac{5}{3}$-approximation algorithm based on MST

As mentioned before, the best known algorithm for solving (unsigned) BMP is due to Caprara. This algorithm is a $\frac{5}{3}$-approximation with a complexity of $O(n^4)$ for the k-median problem. Here we provide a new heuristic algorithm for solving the BMP in the case of 3-median. Our algorithm has a worst-case ratio of $\frac{5}{3}$ and complexity of $O(n^3)$. Thus, it improves the previous results in terms of complexity for the case of $k = 3$, the most well-studied case of the median problem.

As previously showed, solving the BMP corresponds to solving a TSP on the BP-median graph where each sequence of zero edges is shrunk into one 0-edge. Let G be such a graph, i.e. the breakpoint median graph combined with shrinking all paths of 0-edges according to Section 4.2. Then, since all paths of 0-edges are shrunk, Theorem 6 tells us that the only possible edge weights at any vertex v of G are the patterns A,B,C,D,E and F shown in Figure 4.2.

We need to review some results from graph theory which we will use later in the construction of our algorithm. A *minimum spanning tree* (MST) on a graph G is a tree that contains all vertices of G such that the total sum of the edge weights within the tree is minimized. The following lemma is a well-known result for TSP (see [9]).

Lemma 10. *MST is a lower bound for TSP.*

Lemma 11. *Given a spanning tree T on a graph G and an edge e in T, there is always an edge e_l not in T that joins two leaves of T, and such that if we remove e from T and add e_l, we get a different spanning tree of G.*

Proof. Let $e = uv$ be the edge that is being removed. We follow the vertices u and v until we reach two leaves of T, say x and y (note that $x \neq y$, since there is no cycle

in a tree). Add the edge xy to T. We obtain a unique cycle in T. Now removing the edge e results in a new spanning tree T'. □

We refer to the operation described in Lemma 11 as a *swap* on an edge e, i.e. removal of an edge of a spanning tree T and replacing it with another edge not in T that joins two leaves of T. Note that a swap on an edge e decreases the degree of the endpoints of e by 1, and creates two new vertices of degree 2 in T.

Lemma 12. *In any tree with maximum vertex degree 3, there is a degree 3 vertex that is only adjacent to at most one degree 3 vertex (other than itself).*

Proof. Let T be a tree with maximum degree 3. Let v be an arbitrary vertex of degree 3 in T. If the lemma is not true, we must have v being adjacent to two degree 3 vertices, say u and w. Now, if u or w are not adjacent to any other vertex of degree 3, we are done. Otherwise, let us assume u and w are adjacent to another degree 3 vertex, and so on. That is, let us assume $v_0, v_1, v_2, \ldots, v_k$ is a longest path of degree 3 vertices in T. Each vertex except v_0 and v_k in this path is adjacent to two other degree 3 vertices. Now, either v_0 or v_k satisfy the lemma and we are done, or otherwise, v_k must be adjacent to another degree 3 vertex. Moreover, since T is acyclic, v_k cannot be adjacent to $v_0, v_1, v_2, \ldots, v_{k-1}$ since that would create a cycle in the tree T. Therefore, there is at least one degree 3 vertex (v_0 or v_k) that is adjacent to at most one other degree 3 vertex. □

The main idea behind our algorithm is to start with an MST, and then use swaps to decrease the degrees of all vertices to at most 2, thus creating a new spanning tree T' that is a Hamilton path. We will do this in such a way that the cost of T' is at most $\frac{5}{3}$ of the cost of the MST. Let G be a BP-median graph where all paths of 0-edges are shrunk as previously described. The following steps describe our algorithm:

Algorithm 1: Solving a BMP by using a MST:

1. Find an MST T for G, and let $T' = T$.

2. While there exists an edge e of weight 3 in T' incident to a vertex of degree larger than 2, use a swap to remove e from T'. (Note that at the end of this step, for any edge of weight 3 in T', the degrees of u and v are at most 2.)

3. While there exists a vertex v of degree larger than 2 in T' which is not a type B vertex (see Figure 4.2), use swaps to remove all but the cheapest two edges of T' at v. This decreases the degree of v to 2. Mark the removed edges as *used*. Note that the only vertices remaining in T' of degree larger than 2 are type B, and they have exactly degree 3.

4. For each vertex v of type B in T': Using Lemma 12, there is such a vertex x that is adjacent to at most one other vertex y of degree 3. Apply a swap to remove the edge xy if such a y exists, or to remove any edge at x if y does not exist. Mark all three edges at x that have been removed, as *used*.

5. At this point T' has only vertices of degree 1 and 2, and therefore, it is a Hamilton path from a vertex u to some vertex v. Add the edge vu to obtain a Hamilton cycle C.

Note that an edge is marked *used* at most once throughout the algorithm. Next, we prove that the cost of the Hamilton cycle C obtained from this algorithm has an upper bound of $\frac{5}{3}$ of the optimal TSP.

Theorem 9. *In the above algorithm,*

$$cost(C) \leq \frac{5}{3}\, cost(opt\ \ TSP) + 3.$$

Proof. Consider a vertex v of degree greater than 2 in T' when it is processed by the algorithm. The edges incident to v in T' are a subset of those incident to v in T. For each edge e that is swapped, the edge e_l that is added may be of weight 3 in the

worst case. Thus, each of these swaps costs $3 - w(e)$ in the worst case, for each edge e removed. Since at Step 2, only 3-edges are swapped, by the end of Step 2 we have cost(T')=cost(T).

At Step 3, we will only need to remove edges of weight 2, and each swap adds 1 to the cost of T'. So, at the end of Step 3, we have

cost(T') = cost(T) + $\frac{1}{2}$(cost of edges marked as used)

$\leq$ cost(T)+ $\frac{1}{2}$(cost(T) - number of 1-edges in T)

Therefore, at the end of Step 3, all of the vertices of degree greater than 2 that are left in T' are of degree 3 and are of type B, i.e. all edges incident to these vertices have weight equal to 1 in T'. Also, none of the edges in T' are marked as used.

At Step 4, each swap would cost at most 2, since the weights of the edges that are being removed (edges incident to type B vertices) are 1. Therefore, each swap increases the cost of T' by at most 2. Because of the way we chose v, after the edge e incident to v is swapped, neither of the other two edges incident to v is incident to a vertex of degree 3 in the new T' (recall Lemma 12). So we will not need to mark them as *used* for another vertex of degree 3, later at this step. Assuming k swaps are performed at Step 4, we increase the cost of T' by $2k$, but the set of disjoint 1-edges marked as *used* adds up to a total cost of $3k$. So for the final cost of T' we have

total cost of T'= cost of T' at the end of Step 3 + $2k$

$\leq$ cost(T) + $\frac{1}{2}$(cost of T - number of 1-edges in T) +$2k$

= cost(T) + $\frac{1}{2}$(cost of T - number of 1-edges in T) +$\frac{2}{3}(3k)$

$\leq$ cost(T) + $\frac{1}{2}$(cost of T - number of 1-edges in T) +$\frac{2}{3}$(number of 1-edges in T)

= $\frac{3}{2}$ cost(T) + $\frac{1}{6}$(number of 1-edges in T)

$\leq \frac{5}{3}$ cost(T).

Since T is an MST, from Lemma 10, cost(T) is a lower bound for TSP, and we have

total cost of $T' \leq \frac{5}{3}$ cost(opt TSP).

At Step 5, we add one edge to the MST, and thus, in the worst-case, the cost is increased by 3. Therefore, an upper bound for the algorithm is $\frac{5}{3}$ cost(opt TSP) + 3. □

Note that the time complexity of the algorithm using the MST is $O(n^3)$ which is better compared to Caprara's algorithm by at least an order of n.

4.5 Conclusion

The purpose of the current study was to examine the NP-hard problem of finding the breakpoint median for unichromosomal genomes, to investigate the properties of the underlying structure of such problems, and to derive new algorithms for finding the median.

We provided a general shrinking method that may greatly reduce the size of the problem. This could potentially speed up the running times specially when genomes of larges size are considered. We provided a formal proof that this shrinking method does not affect the optimal solution, i.e. the optimal solution to the shrunk problem can easily be translated back into an optimal solution to the original BMP.

Next, we investigated the possibility of applying tour improvement methods from TSP to the BMP. We combined this approach with a simple algorithm of selecting the best genome among the given set of genomes to be the median. This resulted in a $\frac{4}{3}$-approximation algorithm in the case of applying a 2-opt style algorithm, and a $\frac{4}{3}$ performance guarantee in the case of combining with a Lin-Kernighan approach. Both methods outperform Caprara's $\frac{5}{3}$ ratio for the k-median problem when $k = 3, 4, 5$.

We also considered a cycle patching approach and showed that it provides a $\frac{7}{3}$ approximation algorithm for BMP. Although the performance guarantee is not as good as the other methods (e.g. the 2-opt and the LK approaches with the worst case guarantees of $\frac{4}{3}$), this cycle patching is not known to be tight, and thus, it can

potentially perform better and the bound could be improved.

Furthermore, we provided an original approximation algorithm with a performance ratio of $\frac{5}{3}$, based on using minimum spanning trees. This algorithm has a time complexity of $O(n^3)$, which is better than the best known result of Caprara by at least an order of n. This performance guarantee is also not known to be tight, so again it can potentially perform better and the bound could be improved.

The findings in this study add to the growing body of literature on median problems, and confirm that combinatorial optimization techniques can provide strong tools for designing algorithms for BMP. This research provides new avenues for finding an optimal or near-optimal solutions to the BMP, and may also serve as a base for future studies on performance improvement of the NP-hard problem of finding the unichromosomal breakpoint median.

Bibliography

[1] Baburin, A.E., Della Croce, F., Gimadi, E.K., Glazkov, Y.V., and Paschos, V.Th.: Approximation algorithms for the 2-peripatetic salesman problem with edge weights 1 and 2, Discrete Applied Mathematics, 157(9):1988-1992, 2009.

[2] Bryant, D.: The complexity of the breakpoint median problem, technical report CRM-2579, Centre de Recherches de Mathematiques, Universite de Montreal, 1998.

[3] Caprara, A.: Additive bounding, worst-case analysis and the breakpoint median problem, SIAM Journal on Optimization, 13:508-519, 2002.

[4] Cheung, K.: Private communications, 2009.

[5] Christofides, N.: Worst case analysis of a new heuristic for the traveling salesman problem, Report 388, Graduate School of Industrial Administration, Carnegie Mellon University, Pittsburgh, 1976.

[6] Gusfield, D.: Algorithms on strings, trees and sequences: computer science and computational biology, Cambridge University Press, 1997.

[7] Hannenhalli, S. and Pevzner, P.: To cut ... or not to cut (applications of comparative physical maps in molecular evolution), Proceedings of the Seventh Annual ACM-SIAM Symposium on Discrete Algorithms -Atlanta, GA, 304-313, 1996.

[8] Hartvigsen, D.B.: Extensions of matching theory, Ph.D. Thesis, Carnegie-Mellon University, 1984.

[9] Lawler, E., Lenstra, J., Rinnooy Kan, A., and Shmoys, D.: The Traveling Salesman Problem–A Guided Tour of Combinatorial Optimization, Wiley, Chichester, 1985.

[10] Lin, S., and Kernighan, B.W.: An effective heuristic algorithm for the traveling salesman problem, Operations Research, 21: 498-516, 1973.

[11] Papadimitriou, C.H., and Steiglitz, K.: Combinatorial optimization: Algorithms and Complexity, Prentice-Hall, Englewood Ciffs, NJ., 1982.

[12] Papadimitriou, C.H., and Yannakakis, M.: The traveling salesman problem with distances one and two, Mathematics of Operations Research, 18:111, 1993.

[13] Pe'er, I., and Shamir, R.: The median problems for breakpoints are Np-complete, Electronic Colloquium on Computational Complexity, technical reports 71, 1998.

[14] Pe'er, I., and Shamir, R.: Approximation algorithms for the median problem in the breakpoint model, in Sankoff, D., and Nadeau, J.H. (eds.), Comparative Genomics: 225-241, Kluwer Dordrecht, 2000.

[15] Sankoff, D. and Blanchette, M.: The median problem for breakpoints in comparative genomics, in Proceedings of the Third International Computing and Combinatorics Conference, T.Jiang and D.T.Lee eds, Lecture Notes in Computer Science 1276:251-263, Springer-Verlag, 1997.

[16] Sankoff, D. and Blanchette, M.: Multiple genome rearrangement and breakpoint phylogeny, Journal of Computational Biology, 5:555570, 1998.

[17] Schrijver, A., Combinatorial Optimization: Polyhedra and Efficiency, volume 24 of Algorithms and Combinatorics, Springer, 2003.

[18] Tannier, E., Zheng, C., and Sankoff, D.: Multichromosomal median and halving problems under different genomic distances, BMC Bioinformatics, 10:120, 2009.

Chapter 5

Heuristic methods for multichromosomal BMP based on detection of conserved groups of genes

• Maryam Haghighi, Application of Clusters for Median Problems: Heuristic Methods, Technical Report, University of Ottawa, 2011.

Work of Maryam Haghighi under the supervision of Sylvia Boyd. This chapter is work of Maryam Haghighi with minor editorial comments of Sylvia Boyd.

Abstract

In this paper we investigate some concepts that can be combined with other available methods for finding the breakpoint median of a given set of genomes. These ideas may help us to obtain a more realistic picture of the median. In particular, we propose the idea of having conserved groups of genes in the input data required to stay together in the median. We briefly review some clustering concepts, and analyze how different clustering methods can be applied jointly with algorithms for finding the breakpoint median. These provide us with new heuristic algorithms for the breakpoint median problems. Moreover, we propose some possible relationships between Dyck paths, random graphs, and phylogenetic studies.

5.1 Introduction

There has been evidence that certain groups of genes may transfer together throughout the evolution or from one species to another [3]. This has given rise to the notion of genetic linkage, conserved clusters, and conserved segments. The formal definition of these terms may vary in the literature (e.g. see [10] and [12]). Nonetheless, the idea that some groups of gene appear to "stay together" remains the common belief behind this concept. Significant amounts of biological research has been performed on detecting such groups/clusters. There is evidence that "intra-chromosomal" rearrangements (change in the location of a gene within the same chromosome) happen more often than "interchromosomal" rearrangements (change in the location of a gene from one chromosome to another) [3].

We will show how clustering methods can be combined with methods for finding the median (representing a common ancestor of a set of genomes). We present a heuristic algorithm based on detection of conserved groups of genes, and utilizing them into obtaining a partitioning of the vertices representing those genes in a graph. Next, we apply a similar approach to [6], to obtain an estimate ordering of the genes in the common ancestor (median).

Furthermore, we propose a connection between large deviation principle and genome rearrangements. In particular, we present how Dyck paths may be used to model various stages of a genome throughout a set of rearrangement operations.

We must emphasize that the focus here is on novel ideas that can be applied to improve our knowledge of genomic rearrangements, and although we have not yet tested any of these heuristic methods on data, we believe that they represent a fresh point of view on the subject matter, and are worth further investigated.

5.2 A heuristic method based on partitions

As we mentioned, there is some evidence that if the two genes belong to the same chromosome in a genome, then there is a higher probability that the two genes exist in the same chromosome in the ancestral species. The idea is supported by the fact that large changes in the content of a chromosome, and the jumping of a sequence of genes from one chromosome to another is not a common evolutionary process compared to gene swaps within the same chromosome [3].

A model for the common ancestor of several genomes is the so called notion of a *median*. Given a set of genomes, the *median problem* asks for finding a *median* genome M such that the total sum of the "distances" between M and every other genome is minimized. There are many possible distance functions commonly used to measure how far one genome is from another. A popular distance measure is the *breakpoint distance* introduced in [11].

We focus on the *multichromosomal linear breakpoint median problem* (BMP) as defined as follows. Given three multichromosomal genomes A, B, and C on the same set of n genes, BMP asks to find a *median* genome M such that the total sum of the breakpoint distances between M and every other genome is minimized.

We propose a heuristic method for BMP that combines the approach presented in [6] for finding an optimal solution to the BMP, with special attention to having certain groups of genes appearing together in the median solution.

One of the most well-known problems in combinatorial optimization is the *Traveling Salesman Problem (TSP)* (see [5] for background on the TSP). Consider a complete weighted graph. A *Hamilton cycle* is a cycle of the graph that visits every vertex exactly once. A minimum-weight Hamilton cycle is a Hamilton cycle such that the sum of the edge-weights of the cycle is minimized. An optimal solution to the TSP calls for finding such a minimum-weight Hamilton cycle. The TSP is known to be NP-hard [5].

Given two multichromosomal genomes G and H on the same set of n genes, the *breakpoint distance* between G and H, denoted by $d(G, H)$ is defined as

$$d(G, H) = n - a(G, H) - \frac{e(G, H)}{2} \tag{5.1}$$

where $a(G, H)$ is the number of common adjacencies between G and H, and $e(G, H)$ is the number of common telomeres of G and H [13].

In [6], a transformation of the BMP into a *Multiple Traveling Salesman Problem* (mTSP) is given. An mTSP is an assignment of exactly m salesmen to a set of vertices of a graph such that all salesmen start and end their journey at a fixed vertex called the *depot* and each other vertex gets visited exactly once by exactly one salesman. The goal of the mTSP is to minimize the total cost of all of the routes. Define the *r-to-m Multiple Traveling Salesman Problem* (rmTSP) as a variation of the mTSP in which at least r and at most m salesmen are used [6].

A *BP-median graph* G is a complete graph of n vertices such that n vertices correspond to n genes where the weight of an edge uv between each of these n vertices is equal to $3 - adj(uv)$, where $adj(uv)$ is the number of times u and v are adjacent in each of the three genomes. The *extended BP-median graph* G' for three multichromosomal linear genomes A, B, and C is the BP-median graph G with the addition of a vertex d called the *depot* with an edge between d and every other vertex g of the graph G. Let $\tau(g)$ be equal to the number of times g is a telomere in one of the three genomes. Then, the edge dg has a weight $w(dg) = \frac{3-\tau(g)}{2}$. In [6] it is shown that an optimal solution to the mTSP on this extended BP-median graph corresponds to an optimal breakpoint median with m chromosomes for the genomes A, B, and C.

As previously mentioned, it is believe that "intra-chromosomal" rearrangements are more likely to occur than "interchromosomal" rearrangements [3]. Let A, B, and C be three genomes on the same set Σ of n genes. We define a *conserved group* of genes S to be a set of genes $S = \{i, \ldots, j\}$, $|S| < n$, such that the elements of S

always stay together in each genome, i.e. $S = \{i, \ldots, j\}$ can be permuted among themselves in each genome, but an ordering of genes that contain S does not have a gene $g \notin S$ separating the members of S. Moreover, we assume that each conserved group is maximal, i.e. it is not a subset of a larger conserved group. Let $S_1, \ldots, S_k$ be k conserved groups of genes such that $S_i \cap S_j = \emptyset$, for all $1 \leq i, j \leq k$. Two conserved groups S_i and S_j are called *adjacent conserved groups* if they belong to the same chromosome in one of the genomes A, B, or C. Let C_{ij} be a positive integer corresponding to the cost of having S_i and S_j as adjacent conserved groups. This cost may be determined independently through biological information, statistical studies etc.. Moreover, for a gene g in Σ that does not belong to any conserved group S_i, let $C_g(S_i)$ be the cost of having g adjacent to the conserved group S_i, where adjacency is defined by being adjacent to an element of S_i. We will present an approach to use this knowledge on conserved groups and their associated costs in constructing the median.

For conserved groups $S_1, S_2, \ldots, S_m$, we propose a method to generate m mutually exclusive complete subgraphs $H_1, H_2, \ldots, H_m$ of the BP-median graph G as follows. Let each conserved group S_i, $1 \leq i \leq m$, correspond to a complete weighted graph H_i, $1 \leq i \leq m$, such that the vertices in H_i represent the genes of S_i, and the weight of an edge (uv) in H_i is equal to $3 - adj(uv)$ (same as the weights used in the BP-median graph). Next, add a depot vertex d. For each subgraph H_i, add an edge between d and all vertices v in H_i, with a weight $w(dv) = \frac{3-\tau(v)}{2}$. Add an edge vw between a vertex v in H_i and a vertex w in H_j such that the weight of vw is equal to the cost C_{ij} of having S_i (corresponding to H_i) and S_j (corresponding to H_j) as adjacent conserved groups. For all genes g not in any S_i, add a vertex g, called a *single* vertex, to the graph. Add an edge between each single vertex g and all vertices of H_i, for all $1 \leq i \leq m$, with the weight equal to the cost of having g adjacent to the conserved group S_i. Finally, add an edge between each pair of single vertices g_1 and g_2 such that $w(g_1g_2) = 3 - adj(g_1g_2)$. Let R be the graph obtained by this method.

Find an mTSP solution T on R. Such a solution contains m visits to the depot by m salesmen, where the vertices of R (except depot) are covered by exactly one salesman. Delete the depot, and obtain m disjoint paths covering the remaining vertices. Use each path to obtain an ordering of the genes corresponding to one chromosome of the median.

The intuition behind this heuristic algorithm is to have the weights between H_is relatively higher than the weights for edges within each H_i. Therefore, it would be preferable for the mTSP solution to keep the vertices in each H_i visited continuously, before leaving H_i. By allowing the edges (though with relatively higher weights) between two subgraphs H_i and H_j, we will maintain the possibility of having more than one conserved group of genes in the same chromosome. Furthermore, the edges between the single vertices guarantee that it is possible to obtain a solution containing a conserved group of gene, and a single vertex, in the same chromosome.

We must point out a limitation of this heuristic algorithm. Following this method, it is possible to obtain a path (chromosome) such that a single vertex g appears within the conserved group of genes $S_i = 1, 2, \ldots, i$,that is, it is possible to obtain a path such as $1, 2, \ldots, g, \ldots, i$. Clearly, such a solution seems to be contradictory to our notion of conserved groups of genes. However, in practice, conserved groups of genes are mostly viewed as groups that are "approximately" together, i.e. it is possible to have a few other genes within them that don't belong to our set of conserved genes.

5.2.1 Applying clusters

The limitations of the heuristic algorithm proposed in the previous section motivates us to study the well-known notion of clustering, whereby, slight variations in the conserved groups of genes are allowed. There are various clustering methods available, and different clustering methods may yield very different results. In general, clustering methods can be divided into three main approaches: the partition-based approach,

the hierarchical approach, and the pattern-based approach.

Hierarchical approaches either start with one gene and subsequently merge until clusters are formed, or start with one cluster including all genes and divide until a criteria for stopping is reached. Pattern-based algorithms look for determining a subset of objects (here genes) that exhibit similar behavior across a subset of conditions. Partition-based algorithms include various approaches such as trying to minimize a sum of distances from cluster centers, specifying a geometric structure and reaching convergence, using statistical distributions, or using graphs to model the partition under certain thresholds.

There are various methods that find gene clusters under different conditions such as allowing a gap. One of the most common methods is considering a *max-gap cluster*. A *max-gap cluster* is a region on the chromosome containing a maximal set of homolog genes, where the number of non-homologous genes allowed between pairs of adjacent homologs is at most a fixed parameter g. This fixed parameter is called the (maximum allowed) *gap*. There has been several studies on simulation and also the probability of observing such clusters by chance [7].

Although the concept of "closeness" is essential in detecting clusters, different definitions of proximity exist. Some definitions put a restriction on the total size (total number of conserved genes in the cluster) or the length of a cluster (the total number of the genes in the cluster from one end to the other), whereas some others restrict the allowable proportion of "outsider" genes within a conserved segment/cluster. Let G be the set of all possible genes and S be the subset of G containing our target set of genes, i.e. the conserved genes in a cluster. A *cluster* C is a subset of G such that $S \subseteq C$. The compactness of a cluster C is often measured by the gaps observed between the elements of S in C. Most definitions restrict the size of the largest gap observed in a cluster. The *max gap* is the maximum length of a gap (allowed) in a cluster. Some cluster definitions require the gene order to be exactly the same, while others ignore the gene order and only consider the content. Here, we disregard the

Figure 5.1: An example of three clusters with a max gap of 2. The target genes are shown as solid circles.

gene order in the cluster and have no restrictions on the number of elements allowed in a cluster. For example, Figure 5.1 shows three clusters where the max gap is set at 2. The first cluster is of size 7 and length 10, the second cluster is a singleton of size 1 and length 1, and the third cluster is of size 2 and length 4. In this definition, clusters will never overlap. Therefore, a gene is not considered part of two distinct clusters (otherwise, the clusters can be merged).

Clearly, the length of a cluster of size m with an allowed gap of g is at least m and at most $g(m-1)+m$. Several algorithms have been developed to find such clusters (for example, see [1], and [9]). These algorithms may be applied to detect the clusters. Using these clusters, we can construct a modified BP-median graph, and then apply our method of transforming the multichromosomal breakpoint median problem into an instance of mTSP on this modified BP-median graph.

5.3 Application of large deviation principle to genomic rearrangements

Large deviation theory deals with the decay of the probability of increasingly unlikely events. In this section we briefly explore some possible links between the large deviation theory and the construction of phylogenies.

We now momentarily focus our attention on some concepts from probability theory. Large deviations can be thought of as the study of rare events and asking how rare they are and what happens when they do occur. The first results in large

deviations are based on the work of Harald Cramér. In a simple context of coin tossing, beginning with a fair coin, after a large number of tosses we expect to observe an equal proportion of heads and tails. However, there is a small probability that this equal proportion does not happen, i.e. the proportion of heads or tails is not $\frac{1}{2}$. The large deviation problem focuses on computing how quickly this probability (of not having equal proportions) decays as the number of coin tosses increases.

Furthermore, there are interesting connections between large deviations and random graphs. Consider the canonical Erdos-Renyi model for generating random graphs (see [4] for more details). Such a graph G has n vertices, and it gets updated as follows. At the starting point, G is a complete graph. The canonical Erdos-Renyi model allows to keep each edge of G independently with a fixed probability p. The behavior of this graph as n grows has been the subject of several studies. There are various results for the density of this graph and its different subgraphs (e.g. number of triangles, cliques, connected components etc.). Moreover, there are various results on the thresholds where changes occur in a canonical Erdos-Renyi graph [4]. Recall the breakpoint median graph G, and let us assume we apply an Erdos-Renyi model on G, i.e. we assign a value to each edge e of G corresponding to the probability that the weight of e does not change after a rearrangement operation r. Then, the available results on the asymptotic behaviour of the Erdos-Renyi graph can be used to obtain more information on the behaviour of the breakpoint median graph after a sequence of rearrangement operations.

We now briefly review some results from probability theory, and the asymptotic behavior of Dyck paths. We next propose a new structure that links the evolution of a given genome to the asymptotic behavior of a Dyck path, under certain conditions.

The oldest and one of the most important results for large deviations is the following theorem, known as Cramér's theorem for partial sums of independent identically-distributed random variables. It essentially states that the probability of large deviations from the mean value for the average of the partial sums of such variables decays

exponentially. That is, given independent identically-distributed random variables X_i, $i \geq 1$, the average of their partial sums converges in probability to the mean of the probability distribution of X_i, $i \geq 1$. Moreover, the rate of the exponential decay is given in the form of a Legendre transform.

Theorem 10. *[2] Let $X_1, X_2, \ldots$ be independent identically distributed random variables with mean μ, and let $\varphi(\lambda) = \log(\mathbb{E}(e^{\lambda X}))$ for $\lambda \in \mathbb{R}$ be less than infinity, where $\mathbb{E}$ represents the expected value. Let $S_n = \sum_{i=1}^{n} X_i$ denote the partial sums. Then, for any $x > \mu$ we have*

$$\lim_{n\to\infty} \frac{1}{n} \log \mathbb{P}\left\{\frac{1}{n} S_n \geq x\right\} = -\varphi^*(x),$$

where $\varphi^*(x)$ is the Legendre transform of φ, given by $\varphi^*(x) := \sup_{\lambda\in\mathbb{R}} \{\lambda x - \varphi(\lambda)\}$.

A *Dyck path* is a lattice path in $\mathbb{Z} \times \mathbb{Z}$ consisting of steps $(1, 1)$ and $(1, -1)$ connecting diagonal lattice points such that it never goes below the x-axis. Dyck paths are very well-studied objects that appear in various areas of combinatorics and probability. The number of all Dyck paths that start at $(0, 0)$ and finish at $(2n, 0)$ is the n-th Catalan number. The steps in the form of $(1, 1)$ are called *North-East* steps and the steps in the form of $(1, -1)$ are called *South-East* steps. The *maximum height* h of a Dyck path is defined as the largest coordinate $y = h$ that a Dyck path reaches. The *length* of a Dyck path is the length of the distance that the path travels on the x-axis, i.e. the length of a Dyck path that starts at $(0, 0)$ and ends at $(k, 0)$ is k.

Cramer's Theorem can be viewed as explaining the asymptotic behavior of random walks. In [8], this theorem is generalized to obtain a similar result for Dyck paths.

Theorem 11. *[8] Let h denote the maximum height of Dyck path with length $2N$, where N is a positive integer. Let $n \ll N \ll n^2$, and let x be a positive real number. Then,*

$$\lim_{N\to\infty} \frac{N}{2n^2} \log \mathbb{P}\{h > xn\} \to -x^2.$$

We propose using the idea of constructing a Dyck path corresponding to the evolutionary stages of a genome G. Let G be a given genome and let $(0, 0)$ correspond to G at the starting time $t = 0$. Let each North-East step correspond to one rearrangement operation r (for example one reversal) on G, and let each South-East step correspond to the inverse of a previous rearrangement r on G. For example, if $G = (1\ 2\ 3\ 4\ 5\ 6)$ and r is the reversal between 2 and 4, then the North-East step gives $(1\ 4\ 3\ 2\ 5\ 6)$ and the subsequent South-East step gives $(1\ 2\ 3\ 4\ 5\ 6)$. Note that the North-East and the South-East steps may occur in any order. Then, if several rearrangement operations happen on G (representing the passage of time t), the asymptotic behavior of G may be studied by studying the asymptotic behavior of the corresponding Dyck path. For example, the asymptotic behaviour of the maximum height of the Dyck path can be viewed as an estimate for the farthest distance we can get from G by having the rearrangements occurring in no particular order. In particular, by using the above theorem, at time $t = 2N$, where $n \ll N \ll n^2$, the probability of having a maximum height greater than xn approaches to $exp(\frac{-2n^2x^2}{N})$. These type of asymptotic studies could potentially be useful when estimating the end result of a series of rearrangement operations over a long period of time.

5.4 Conclusions

We presented an idea that related large deviation principle and Dyck paths to phylogenetic studies. This relationship could be investigated further. In particular, we could take advantage of the results available on the asymptotic behaviour of Dyck paths, and also, asymptotic behaviour of random graphs, to analyze the asymptotic behaviour of genomes under several rearrangements. In particular, an application of large deviation for evolutionary studies may prove to be interesting.

We also proposed a novel idea for further improvements in finding the breakpoint median of multichromosomal linear genomes. This scheme is based on the detection

of conserved groups of genes. We illustrated how these conserved groups of genes can be used in developing specific breakpoint median graphs. We discussed how clustering methods are related to the detection of conserved groups of genes, and how they provide a higher degree of freedom for adjustment of the allowed gaps within the conserved segments of the genome. This could potentially be helpful in dealing with median problems arising from specific groups of datasets, where there is previous knowledge available on partial segments of the median, or segments which are highly desired to stay together within a median solution. Further investigation of merging clustering methods with method for finding median solutions may be valuable in providing a more accurate picture of the median as an estimate for a common ancestor.

Bibliography

[1] Bocker, S., Jahn, K., Mixtachi, J., and Stoye, J.: Computation of Median Gene Clusters, Journal of Computational Biology, 16(8):1085-1099, 2009.

[2] Cramer, H.: Random variables and probability distributions, third edition, Cambridge University Press, 1970.

[3] Ehrlich, J., Sankoff, D. and Nadeau, J.H.: Synteny conservation and chromosome rearrangements during mammalian evolution, Genetics 147, 289296, 1997.

[4] Erdos, P., and Renyi, A.: On the evolution of random graphs, Magyar Tud. Akad., Mat. Kutato Int. Kozl, 5: 17-61, 1960.

[5] Gutin, G., and Punnen, A.P. (eds.): The traveling salesman problem and its variations, Kluwer, Dordrecht, 2002.

[6] Haghighi, M. and Boyd, S.: A Fast Method for Large-scale Multichromosomal Median Problem, Proceedings of the ACM International Conference on Bioinformatics Computational Biology and Biomedicine ACM-BCB, Chicago, Il, 2011.

[7] Hoberman, R., Sankoff, D., and Durand, D.: The Statistical Significance of Max-Gap Clusters, In Proceedings of the 2nd RECOMB Workshop on Comparative Genomics, Lagergren, ed., Lecture Notes in Bioinformatics, Springer-Verlag. October 2004.

[8] Kousha, T.: Asymptotic behavior and moderate deviation principle for the maximum of a Dyck path, Workshop on Positivity, Field's Institute, Toronto, Canada, August 2011.

[9] Ling, X., He, X., and Xin, D.: Detecting gene clusters under evolutionary constraint in a large number of genomes, Bioinformatics 25 (5): 571-577, 2009.

[10] Nadeau J.H., and Taylor B.A.: Lengths of chromosomal segments conserved since divergence of man and mouse, Proceedings of the National Academy of Sciences USA, 81: 814818, 1984.

[11] Sankoff, D. and Blanchette, M.: The median problem for breakpoints in comparative genomics, in Proceedings of the Third International Computing and Combinatorics Conference, T.Jiang and D.T.Lee eds, Lecture Notes in Computer Science 1276:251-263, Springer-Verlag, 1997.

[12] Sankoff, D., Parent, M.-N. and Bryant, D.: In Comparative Genomics, D. Sankoff and J.H. Nadeau, eds. Kluwer Academic, Dordrecht, pp. 299-306, 2000.

[13] Tannier, E., Zheng, C., and Sankoff, D.: Multichromosomal median and halving problems under different genomic distances, BMC Bioinformatics, 10:120, 2009.

Chapter 6

Conclusion

This dissertation has investigated several techniques from combinatorial optimization that can be applied to find solutions to the breakpoint median problem and its variations. We started in Chapter 1 by reviewing some of the current results in the area. The main focus was in mathematical and combinatorial aspects of the problem, and how questions from computational biology can be formulated into problems from combinatorial optimization. We provided a brief review of the best known results with a focus on results on the calculation of various genomic distances and different median problems.

In the paper in Chapter 2, we studied the problem of multichromosomal breakpoint median for linear genomes. In particular, we provided a novel and practical mathematical structure for the NP-hard problem of finding the multichromosomal breakpoint median for linear genomes. As we have mentioned, most of the research in the literature has been focused on the unichromosomal median problem. The findings from Chapter 2 make several contributions to advancing the few results that were previously available on the multichromosomal breakpoint median problem for linear genomes. The multichromosomal breakpoint median problem for linear genomes is an NP-hard problem. Thus, approximation algorithms and heuristic methods can be ap-

plied to find optimal or near-optimal solutions. As mentioned, we presented a heuristic method based on transforming the multichromosomal BMP on linear genomes to an instance of multiple salesman TSP. We then used a transformation of the multiple salesman TSP into (the usual) TSP. Since TSP is a well-studied problem in combinatorial optimization, this transformation opens the door to all the techniques that are available for solving TSP to be applied to finding the median. We used this method combined with the TSP solver Concorde ([1]) to obtain solutions for the multichromosomal BMP for linear genomes. We demonstrated the efficiency of our approach by performing an empirical study on both simulated and real data with a comparison to other methods. Based on our results, we can conclude that our method is a very powerful technique as we were able to solve median problems involving genomes with 25,000 genes in less than 20 minutes, whereas previous methods were only able to solve median problems for genomes with a few hundred genes. Moreover, our method proved to be extremely efficient in obtaining optimal solutions in almost all cases.

In the paper in Chapter 3 we continued our research on the multichromosomal breakpoint median problem by focusing on the two other possible cases, namely, the circular and mixed (circular and linear) genomes. We provided a novel mathematical framework for the unsigned version of these problems and obtained the first proof that the multichromosomal BMP for circular and mixed unsigned genomes is polynomial. A similar result for signed genomes was previously obtained in [4]. Therefore, combining our result with those of [4], we can now recognize that the multichromosomal BMP for circular and mixed genomes, signed or unsigned, is a polynomial-time problem. We also defined the concept of partially signed genomes, a possible case arising from incomplete real-world data. We showed how we are able to modify our model for the median problem to suit having partially signed genomes, and be able to find the breakpoint median in this case. As a consequence of this model, we see that the multichromosomal BMP for circular and mixed partially signed genomes is also polynomial.

The focus in Chapter 4 was the breakpoint median problem for unichromosomal genomes. In this chapter we provided different approximation and heuristic algorithms for the unichromosomal breakpoint median problem. These algorithms are mainly based on the underlying structure of the constructed graphs and take advantage of some known results from graph theory. We provided details of a shrinking method which can reduce the size of the problem. This can be especially effective in light of having larger amount of real-world or simulated data becoming available. For the median of three genomes, we presented improved solutions by combining various techniques for TSP. The underlying structure present in the breakpoint median problem was studied and was subsequently utilized to derive two new approximation algorithms, the first one based on a patching idea, and the second one based on minimum spanning trees. The findings in this chapter add to a growing body of literature on the approximation results for the median problem. It is our belief that the ideas and methods presented here would do well in an empirical testing. In particular, we expect that the method based on the Lin-Kernighan method for TSP would be extremely successful, given its great success for TSP.

Detection of clusters and conserved groups of genes (groups of genes that may transfer together throughout the evolution) has been the subject of several studies. In the paper in Chapter 5, we concentrated on how some of the methods presented in the previous chapters can be modified further based on specific information on the data, such as having conserved groups of genes. In this chapter we briefly looked into some of the definitions and available methods for detection of these clusters and conserved groups of genes. We presented a heuristic method for the multichromosomal breakpoint median problem, derived from the method presented in Chapter 2 for this problem combined with a clustering method. This can potentially lead to finding more accurate results for the median problem and enhance our understanding of how these various methods can be merged together. Furthermore, we briefly introduced some ideas in combining results from the probability theory with the median problem.

Although the focus of this research is in combinatorial optimization, we felt that such connections are worth to explore and can provide basis for further research, especially considering the known connections between Dyck paths and combinatorics. Taken together, these findings suggest that perhaps the best way to model and solve the median problem is a multi-level approach combining all of these methods.

6.1 Future Work

There were a number of important limitations which need to be considered in this research.

First of all, the current study has only examined the case of having no deletions, insertions, or duplications, all of which do happen in real-world genomes. In all of our models we have considered the problem of having exactly one copy of each gene present in each genome. A more realistic framework would be to consider the possibilities of having multiple copies of various genes in the genomes. Due to its difficulty, the multiple-copied problem is much less-studied, even in its simplest form. Even though the single-copy problem is very crucial as a building block for rearrangement problems, it is worthwhile to consider variations such as including multiple copies, as this is the case in most advanced organisms.

Sankoff in [6] proposed a way to select, from the duplicated copies of genes, the common ancestor gene such that the distance between the reduced genomes (exemplar genomes) is minimized. This method was tested by a general branch-and-bound approach. The authors of [3] computed the exemplar breakpoint distance by using a divide-and-conquer method. It has been shown that calculating exemplar breakpoint distance is NP-complete [2].

We strongly believe that the multiple-copied breakpoint median problem can be formulated using methods similar to Chapter 2 with the addition of requirements for multiple copies. Multiple copies of a gene may be viewed as several visits to a vertex,

and therefore, may be modeled by assigning weights to the vertices corresponding to the copies. Multivisit multisalesman TSP can then be utilized to find solutions for the multiple-copied linear multichromosomal BMP.

Further research could also concentrate in generalizing the methods of Chapter 4 to obtain better approximation ratios and easier and simpler theoretical proofs for cases where the median for more than 3 genomes is considered. Some of the proposed approximation algorithms in Chapter 4 are not tight, i.e. in practice, they may perform much better than their worst-case ratio. Therefore, an experimental study to compare the actual performance of the various approximation algorithms proposed could be very valuable. Such an experimental analysis may also show that some of these algorithms perform better on certain types of data, whereas some have a better performance in others. Also, the performance comparison can be studied in terms of both the running time, and also how close the medians found are to optimal. The notion of an optimal median can also be further investigated, i.e. how close an optimal or a near-optimal median is to real-world medians.

Also, it would be interesting to assess the effects of combining the methods presented in Chapter 5, with the framework presented in Chapter 2. An empirical study of comparing the current results of Chapter 2, with the heuristics proposed in Chapter 5 would be very interesting.

6.1.1 Library of data

There is an increasing interest in genome rearrangement, including finding genomic distance and genomic medians, and as mentioned in the previous section, the methods and algorithms in this field are tested on simulated and real-world data. Although various genome databases exist, to the best of our knowledge, no uniform database for genome rearrangement data exists.

We should mention that most material available in genome rearrangement liter-

ature make a reference to the genome browsers they used to obtain the data from, however, these genome browsers are continuously progressing and changing as new data become available, or modifications are made to previously obtained data due to laboratory errors. Therefore, unless authors provide the specific data used for testing their methods, it makes it nearly impossible to check the performance of different methods under the same data set used by the authors. This problem also makes it difficult to draw a fair comparison between various methods and algorithms. The same problem holds for simulated data, as different papers use similar but somewhat slightly different methods to generate simulated data.

Hence, there is an obvious need for a public library of genome rearrangement data. Such libraries exist for famous problems such as TSP [5].

Here we provide some basic criteria that should be considered for making such a public library.

1)All of the data available from the library should have a uniform format. This may prove to be difficult to obtain, as different researchers and different softwares use slightly different formats. However, it seems that a notation similar to strings or arrays for permutations is somewhat popular.

2)It should be possible to generate different inputs from the available data, for example it should be possible to generate unsigned genomes from the signed version.

3)The contribution and the source of the data must be clearly stated, and copyright rules must be followed.

4)There should be no further modifications to the data after including it in the library

5)The library may contain both simulated, and real-world data, with proper identifications.

We strongly believe that creation of such a library of data could be extremely helpful in advancing research in genome rearrangements studies.

Bibliography

[1] Applegate, D., Bixby, R., Chvatal, V., and Cook, W.: Concorde TSP Solver, www.tsp.gatech.edu/concorde

[2] Bryant, D.: The complexity of calculating exemplar distances, In D. Sankoff and J.Nadeau, editors, Comparative Genomics: Empirical and Analytical Approaches to Gene Order Dynamics, Map Alignment, and the Evolution of Gene Families, 207-212, Kluwer Acad. Pub., 2000.

[3] Nguyen, C.T., Tay, Y.C., and Zhang, L.: Divide-and-conquer approach for the exemplar breakpoint distance, Bioinformatics, 21(10):2171-2176, 2005.

[4] Tannier, E., Zheng, C., and Sankoff, D.: Multichromosomal median and halving problems under different genomic distances, BMC Bioinformatics, 10:120, 2009.

[5] Library of Travelling Salesman Problems maintained by Reinelt, G.: http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/

[6] Sankoff, D.: Genome rearrangement with gene families, Bioinformatics, 16 (11):909-917, 1999.