MODELS AND ALGORITHMS FOR SORTING PERMUTATIONS WITH TANDEM DUPLICATION AND RANDOM LOSS

Von der Fakultät für Mathematik und Informatik
der Universität Leipzig
angenommene

DISSERTATION

zur Erlangung des akademischen Grades

DOCTOR RERUM NATURALIUM
(Dr. rer. nat.)

im Fachgebiet

INFORMATIK

Vorgelegt

von Dipl.-Math. Tom Hartmann

geboren am 21. Januar 1989 in Bergen

Die Annahme der Dissertation wurde empfohlen von:

1. Prof. Dr. Martin Middendorf, Universität Leipzig
2. Prof. Dr. Jean-Stéphane Varré, Université de Lille

Die Verleihung des akademischen Grades erfolgt mit Bestehen
der Verteidigung am 16.04.2019 mit dem Gesamtprädikat *magna cum laude*.

ABSTRACT

A central topic of evolutionary biology is the inference of phylogeny, i. e., the evolutionary history of species. A powerful tool for the inference of such phylogenetic relationships is the arrangement of the genes in mitochondrial genomes. The rationale is that these gene arrangements are subject to different types of mutations in the course of evolution. Hence, a high similarity in the gene arrangement between two species indicates a close evolutionary relation. Metazoan mitochondrial gene arrangements are particularly well suited for such phylogenetic studies as they are available for a wide range of species, their gene content is almost invariant, and usually free of duplicates. With these properties gene arrangements of mitochondrial genomes are modeled by permutations in which each element represents a gene, i. e., a specific genetic sequence. The mutations that shape the gene arrangement of genomes are then represented by operations that rearrange elements in permutations, so-called genome rearrangements, and thereby bridge the gap between evolutionary biology and optimization. Many problems of phylogeny inference can be formulated as challenging combinatorial optimization problems which makes this research area especially interesting for computer scientists. The most prominent examples of such optimization problems are the sorting problem and the distance problem. While the sorting problem requires a minimum length sequence of rearrangements that transforms one given permutation into another given permutation, i. e., it aims for a hypothetical scenario of gene order evolution, the distance problem intends to determine only the length of such a sequence. This minimum length is called distance and used as a (dis)similarity measure quantifying the evolutionary relatedness.

Most evolutionary changes occurring in gene arrangements of mitochondrial genomes can be explained by the tandem duplication random loss (TDRL) genome rearrangement model. A TDRL consists of a duplication of a consecutive set of genes in tandem followed by a random loss of one copy of each duplicated gene. In spite of the importance of the TDRL genome rearrangement in mitochondrial evolution, its combinatorial properties have rarely been studied. In addition, models of genome rearrangements which include all types of rearrangement that are relevant for mitochondrial genomes, i. e., inversions, transpositions, inverse transpositions, and TDRLs, while admitting computational tractability are rare. Nevertheless, especially for metazoan gene arrangements the TDRL rearrangement should be considered for the reconstruction of phylogeny. Realizing that a better understanding of the TDRL model is indispensable for the study of mitochondrial gene arrangements, the central theme of this thesis is to broaden the horizon of TDRL genome rearrangements with respect to mitochondrial genome evolution. For this purpose, this thesis provides combinatorial properties of the TDRL model and its variants

as well as efficient methods for a plausible reconstruction of rearrangement scenarios between gene arrangements. The methods that are proposed consider all types of genome rearrangements that predominately occur during mitochondrial evolution. More precisely, the main points contained in this thesis are as follows:

The distance problem and the sorting problem for the TDRL model are further examined in respect to circular permutations, a formal concept that reflects the circular structure of mitochondrial genomes. As a result, a closed formula for the distance is provided.

Recently, evidence for a variant of the TDRL rearrangement model in which the duplicated set of genes is additionally inverted have been found. Initiating the algorithmic study of this new rearrangement model on a certain type of permutations, a closed formula solving the distance problem is proposed as well as a quasilinear time algorithm that solves the corresponding sorting problem.

The assumption that only one type of genome rearrangement has occurred during the evolution of certain gene arrangements is most likely unrealistic, e. g., at least three types of rearrangements on top of the TDRL rearrangement have to be considered for the evolution metazoan mitochondrial genomes. Therefore, three different biologically motivated constraints are taken into account in this thesis in order to produce plausible evolutionary rearrangement scenarios. The first constraint is extending the considered set of genome rearrangements to the model that covers all four common types of mitochondrial genome rearrangements. For this 4-type model a sharp lower bound and several close additive upper bounds on the distance are developed. As a byproduct, a polynomial-time approximation algorithm for the corresponding sorting problem is provided that guarantees the computation of pairwise rearrangement scenarios that deviate from a minimum length scenario by at most two rearrangement operations. The second biologically motivated constraint is the relative frequency of the different types of rearrangements occurring during the evolution. The frequency is modeled by employing a weighting scheme on the 4-type model in which every rearrangement is weighted with respect to its type. The resulting NP-hard sorting problem is then solved by means of a polynomial size integer linear program. The third biologically motivated constraint that has been taken into account is that certain subsets of genes are often found in close proximity in the gene arrangements of many different species. This observation is reflected by demanding rearrangement scenarios to preserve certain groups of genes which are modeled by common intervals of permutations. In order to solve the sorting problem that considers all three types of biologically motivated constraints, the exact dynamic programming algorithm `CREx2` is proposed. `CREx2` has a linear runtime for a large class of problem instances. Otherwise, two versions of the `CREx2` are provided: The first version provides exact solutions but has an exponential runtime in the worst case and the second version provides approximated solutions efficiently. `CREx2` is evaluated by an empirical study for simulated artificial and real biological mitochondrial gene arrangements.

ARTISTICAL ABSTRACT

"Transforming ducklings into puffins." by Philipp Zins

To my grandmothers who were often in
my thoughts on this journey – you are missed.

ACKNOWLEDGMENTS

During the past five years of PhD student life, I was supported by many people who I now like to express my gratitude to.

First of all, I am genuinely thankful to my advisor Prof. Dr. Martin Middendorf for providing me with the excellent and inspiring scientific environment and the ongoing support, guidance, and advice. He taught me how to do research and generously shared his deep knowledge and expertise that continues to inspire me in personal life as well. I was indeed fortunate to have him as my advisor.

A gigantic thanks to all my current and past colleagues of the Swarm Intelligence and Complex Systems group, especially to Dr. Matthias Bernt and Dr. Nicolas Wieseke, for introducing me to fascinating research fields, always providing me with almost unsolvable riddles, sharing the very personal enthusiasm for discovery, and for being helpful coauthors and inspiring mentors. It has been a pleasure to work with you.

I had the great privilege to visit research groups around the world. First of all, I want to thank Prof. Dr. Yao-Ting Huang from the National Chung Cheng University and Yih-Chun Cheng for very informative and inspiring cooperation, for introducing me to Taiwanese culture, and for showing me the breathtaking beauty of Taiwanese nature. Furthermore, I want to thank Prof. Dr. Millie Pant and Dr. Sunil Kumar Jauhar from the Indian Institute of Technology Roorkee and Prof. Dr. Gabriel Valiente from the Technical University of Catalonia for hosting me and sharing their helpful thoughts.

A special thank you goes to Dr. Matthias Bernt, Dr. Nicolas Wieseke, Max Bannach, and Laura Schiele for proof-reading this thesis. I also thank Philipp Zins for providing me the artistical abstract illustrating what he thinks I am doing during the past five years.

I thank my wonderful girlfriend Julia Witzlack for always being by my side, loving, supporting, and believing in me. I would like to thank my family, especially my parents, Britta and Rene Hartmann, for their unconditional love and support, and for being there whenever I needed them.

Finally, I would like to gratefully acknowledge funding from the Leipzig University which granted me a three year doctoral scholarship; the German Israeli Foundation (GIF) through the project G-1051-407.4-2013; and the German Research Foundation (DFG) through MI439/14-1. My visit in Yao-Ting Huang's lab and the exchange program with the IIT Roorkee were funded by the German Academic Exchange Service (DAAD) through the project "Taiwan Summer Institute Programme" within 57342485 and the grant 57130298, respectively.

CONTENTS

PUBLICATIONS

Several parts of this thesis have already been published in the following publications:

1. T. Hartmann, A.-C. Chu, M. Middendorf, and M. Bernt (2018b). "Combinatorics of tandem duplication random loss mutations on circular genomes." In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 15.1, pp. 83–95

2. T. Hartmann, N. Wieseke, R. Sharan, M. Middendorf, and M. Bernt (2017). "Genome Rearrangement with ILP." In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 15.5, pp. 1585–1593

3. T. Hartmann, M. Middendorf, and M. Bernt (2018d). "Genome Rearrangement Analysis: Cut and Join Genome Rearrangements and Gene Cluster Preserving Approaches." In: *Comparative Genomics: Methods and Protocols*. Springer, pp. 261–289

4. T. Hartmann, M. Bernt, and M. Middendorf (2018a). "An Exact Algorithm for Sorting by Weighted Preserving Genome Rearrangements." In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics*. (in press)

5. T. Hartmann, M. Bernt, and M. Middendorf (2018c). "EqualTDRL: illustrating equivalent tandem duplication random loss rearrangements." In: *BMC Bioinformatics* 19.192

6. T. Hartmann, M. Bannach, and M. Middendorf (2018e). "Sorting Signed Permutations by Inverse Tandem Duplication Random Losses." In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics*. (in press)

A complete list of all my publications can be found at the end of this thesis.

1 INTRODUCTION

What do scientific fields like shuffling cards, scheduling, logistics, statistical physics as well as molecular evolution (and many more) have in common? Indeed, all these areas comprise remarkably rich combinatorial structures, e. g., elements of finite sets are either arranged sequentially representing specific orders or already ordered sequences are rearranged into new ones. In the corresponding contexts such ordered sequences of finite sets of elements, which are commonly called *permutations*, may represent, e. g., sets of cards that should be randomized (Aldous and Diaconis, 1986), shortest routes connecting a finite set of locations (Applegate et al., 2006), or the creation of a schedule for a number of employees (Burke et al., 2004). Besides applications in various scientific fields, the application of permutations in *Comparative Genomics* is a fascinating area of research and the main topic of this thesis.

In this prominent scientific field, the genetic information of different species – which may be represented by permutations – are comparatively analyzed to characterize similarities and differences in genomic features to obtain insights into evolutionary relations between the considered species (Xia, 2013). This principle is based on the fundamental idea that specific features of two closely related species are often conserved within the genomes of these organisms in the course of evolution (Hardison, 2003) which permits us to formulate hypotheses on evolutionary relations: a high similarity of common features indicates a close evolutionary relation between species. Since the genetic information in species is organized in a highly complex manner (Krebs et al., 2014) features vary from lower levels such as the genomic sequence (Brocchieri, 2001), i. e., the deoxyribonucleic acid (DNA), to higher levels such as the spatial structure that results from the condensed conformation of the DNA (Simonaitis and Swenson, 2018). A certain feature that has gotten a lot of attention for providing support for existing or new hypotheses for phylogenetic relationships among species is the arrangement of certain segments of DNA on the chromosomes of species. Those segments which are commonly called *genes* are of particular importance as they play a crucial role in the organisms cellular processes such as providing blueprints to construct essential biochemical materials.

That the arrangement of genes in genomes offers an opportunity to shed new light on the mechanisms of evolution has firstly been proposed in a pioneering series of articles by two of the leading Drosophilists Theodosius Dobzhansky and Alfred H. Sturtevant. The authors noticed that some sequences of genes had been flipped over in the DNA of certain *Drosophila* species collected from different geographic regions (Sturtevant and Beadle, 1936). These differences in the gene arrangements were further linked to the phenotype and

the reproduction rate of those individuals (Sturtevant and Dobzhansky, 1936a). The authors concluded that such genetic rearrangements could help in reconstructing a plausible path of evolution which later has been proposed for different species of *Drosophila* (Sturtevant and Dobzhansky, 1936b; Dobzhansky and Sturtevant, 1938). The milestone idea of their work was to realize that the considered genomes, which exhibit a striking resemblance in the genomic sequence, differ dramatically in the arrangement of their genes. This observation lead to genomes being modeled as permutations. The quantification of the evolutionary relatedness between two genomes was then done by a manual calculation of a minimum number of rearrangement operations between those permutations (Sturtevant and Novitski, 1941). Almost 40 years later, the first theoretical problem formulation for this procedure has been proposed by Watterson et al., 1982 bridging two – up to this time – predominantly independent research areas: the evolution of species and combinatorial optimization. More precisely, assuming that evolutionary events which change the arrangement of the genes (i. e., to reorder the elements in the permutations in the combinatorial context) are rare (Rokas and Holland, 2000) led to the conclusion that evolutionary scenarios which minimize their number are more likely to be close to reality (Fertin et al., 2009). This assumption connects evolutionary problems to combinatorial optimization by the principle of *maximum parsimony*, a widely used method that aims for an explanation of the considered data while requiring a minimum number of evolutionary events.

Taking this combinatorial background into account, the arrangement of unique genes of the considered genomes are represented by permutations and evolutionary events (which change the arrangement of these genes in the genomes) are represented by so-called *genome rearrangements*, i. e., operations which rearrange elements in permutations. In the light of combinatorics, for two given permutations and a set of considered genome rearrangements two fundamental genome rearrangement problems are then solved to acquire information on parsimonious scenarios of genome rearrangement. In particular, the two problems are the *sorting problem* and the *distance problem*. The sorting problem demands a parsimonious sequence of rearrangements that transforms one given permutation into another given permutation. The distance problem aims to determine only the number of rearrangements in such a sequence. From a biological perspective, the solution of the sorting problem is then interpreted as a hypothetical path of evolution and its length, i. e., the number of rearrangement events, acts as a (dis)similarity measure between genomes. The computational complexity of these problems depends on the types of the considered genome rearrangements. However, in many cases major computational challenges are faced even on permutations representing small gene arrangements, which makes this research area particularly interesting for computer scientists.

The gene arrangement of *mitochondrial* genomes which is available for a wide range of species is well suited for comparative analyses (Boore, 1999; San Mauro et al., 2005). Especially metazoan gene ar-

rangements can easily be represented by permutations as their gene content is nearly invariant among all animals. In the last decade, the mitochondrial gene arrangements in *Metazoa* have been proven to be a powerful source of information for the inference of phylogenetic relationships at different taxonomic levels (Boore, 2006). At least four types of genome rearrangements have to be considered in order to model mitochondrial gene arrangement evolution: *inversion*, *transposition*, *inverse transposition*, and *tandem duplication random loss (TDRL)* (Bernt et al., 2013b; Boore, 1999). Of the several mechanisms which have been proposed to explain gene order rearrangements in mitochondria, most evolutionary changes in mitochondrial gene arrangements can be explained by the TDRL rearrangement model (San Mauro et al., 2005). A TDRL consists of a duplication of a contiguous set of genes in tandem followed by a random loss of one copy of each duplicated gene. Moreover, the TDRL rearrangement was once described as the "most important rearrangement operation" for vertebrate mitogenomes (San Mauro et al., 2005). In spite of the importance of the TDRL genome rearrangement in mitochondrial evolution, its combinatorial properties have rarely been studied (Bernt, 2009). However, the mere fact that genome rearrangement models which include TDRL rearrangements while admitting computational tractability are scarce proves that a better understanding of the TDRL model is indispensable for the study of mitochondrial gene arrangements. Therefore, the central theme of this thesis is to broaden the horizon of the TDRL genome rearrangement model with respect to mitochondrial genome evolution. For this purpose this work provides combinatorial properties of the TDRL genome rearrangement and its variants as well as efficient methods for a biological feasible reconstruction of genome rearrangements between mitochondrial gene arrangements using all genome rearrangements that predominately occur during mitochondrial evolution.

STRUCTURE OF THIS THESIS

Chapter 2 provides a broad overview over the considerable progress which has been made in the development of computational methods solving the fundamental genome rearrangement problems during the last 30 years. In particular, for conceptual purposes Section 2.1 provides a brief introduction to molecular evolution. A key to the notation used in this thesis is provided within Section 2.2. Existing computational methods related to different types of (mitochondrial) genome rearrangements are reviewed in Section 2.3. Some concepts and ideas surveyed in this chapter have previously been discussed in Hartmann et al. (2018d).

The usual mitochondrial genome is organized in a single circular structure (Bernt et al., 2013b). Since the TDRL rearrangement is prevalent in such genomes it is important to consider the circularity in combinatorial analyses. For this purpose, the combinatorial properties of the TDRL rearrangement on circular permutations are investigated in

Chapter 3. More precisely, the distance problem and the sorting problem for circular permutations under the TDRL rearrangement model are studied in Section 3.1. The circularity of mitochondrial genomes entail practical consequences for biological applications which are further discussed in Section 3.2. The results obtained in Chapter 3 have already been covered to a large extent in Hartmann et al. (2018b) and Hartmann et al. (2018c).

In order to compute more realistic scenarios of rearrangements, several authors considered combinations of different types of genome rearrangements. Unfortunately, tracing gene arrangement evolution under combined genome rearrangement models often turns out to be computationally intractable. Nevertheless, in Chapter 4 a tractable genome rearrangement model, called the *inverse tandem duplication random loss rearrangement (iTDRL)* model which generalizes all major mitochondrial rearrangements is introduced. Evidence for the iTDRL model as evolutionary mechanism has recently been found by several authors, e. g., Jühling et al. (2011) and Shi et al. (2013). The algorithmic study of this new model of genome rearrangement is initiated in Section 4.1 in which it is proven that the sorting problem for permutations under iTDRLs can be solved in quasilinear time and that the corresponding distance problem can be solved in linear time. The first step towards a general and tractable model of genome rearrangements for mitochondrial evolution is made in Section 4.2. Most of the work presented in Chapter 4, including the main theorems and algorithmic ideas, was recently published in Hartmann et al. (2018e).

Besides TDRL rearrangements, three other types of genome rearrangements are assumed to be prevalent in the evolution of metazoan mitochondrial genomes (Bernt et al., 2013b). Chapter 5 is devoted to investigate the combined 4-type rearrangement model that considers all four types of mitochondrial genome rearrangements. More precisely, Section 5.1 recapitulates the notation used in Chapter 5. The distance problem for permutations under the 4-type rearrangement model is studied in Section 5.2. The theoretical findings result in a polynomial-time approximation algorithm for the corresponding sorting problem that guarantees to compute pairwise rearrangement scenarios that deviate from a parsimonious scenario by at most two rearrangement operations. In addition, the insight is gained that the general sorting problem with respect to the 4-type model is inconvenient for the inference of plausible reconstructions of mitochondrial gene order evolution. Methods that reconstruct pairwise rearrangement scenarios of mitochondrial gene arrangements under variants of the 4-type model have been developed in Section 5.3 and Section 5.4. Thereby, different biologically motivated constraints are taken into account in order to improve the plausibility of the reconstructed rearrangement scenarios between pairs of gene arrangements. The first biologically motivated constraint that has been considered is the relative frequency of the different types of rearrangements to occur during the evolution. This frequency is modeled by employing a weighting scheme on the 4-type rearrangement model in which every rearrangement is weighted with respect to its type. Section 5.3 studies the

corresponding sorting problem for permutations. In the same section, the sorting problem is tackled by the polynomial-size integer linear programming `GeRe-ILP`. The second biologically motivated constraint that has been taken into account in the reconstruction of pairwise rearrangement scenarios is, that certain subsets of genes are often found in close proximity in the gene arrangements of many different species (Krebs et al., 2014). This observation is reflected in Section 5.4 where the sorting problem for permutations under the (weighted) 4-type rearrangement model is studied under the additional constraint that certain groups of genes (which are modeled by common intervals of permutations) have to be preserved in pairwise rearrangement scenarios. In order to solve this problem, the exact dynamic programming algorithm `CREx2` is proposed. The accuracy of the `CREx2` reconstructions is analyzed in an empirical study with simulated artificial and real biological mitochondrial gene arrangements in Section 5.5. Algorithms `GeRe-ILP` and `CREx2` have been published in advance in (Hartmann et al., 2017) and (Hartmann et al., 2018a), respectively.

Lastly, a conclusion is drawn and opportunities for future research are outlined in Chapter 6.

2 BACKGROUND AND RELATED WORK

The research area of comparative genomics is exceedingly connected to aspects of evolutionary biology. In order to understand the major terms in gene order analysis, Section 2.1 gives an introduction to molecular biology and evolution. In particular, the structure of the genetic information of species and the way it is shaped during evolution is outlined. Since the main objective of this work is the provision of computationally methods which formulate hypotheses for the evolution of mitochondrial gene arrangements, the major differences between nuclear and mitochondrial genomes are described. However, readers who are interested in a more detailed introduction to molecular evolution are referred to Krebs et al. (2014). Section 2.2 formally introduces the notation used in this thesis. Particularly, the formal concepts for gene arrangements, genome rearrangements as well as the corresponding optimization problems are formalized. The computational and algorithmic study of gene arrangements and the mechanisms they are affected by has been initiated more than thirty years ago by Watterson et al. (1982). Over that period, a dramatic expansion of scientific work has lead this research area into a fascinating field. Section 2.3 aims to provide an overview over the achievements obtained by many outstanding researchers. The overview is by no means exhaustive and some topics are only being touched on briefly. A detailed study can be found in Fertin et al. (2009).

2.1 EVOLUTION AND MITOCHONDRIA IN A NUTSHELL

The main aspect of this thesis is combinatorially in the sense of providing efficient algorithms, finding evolutionary plausible mathematical models for fundamental genome rearrangement problems as well as providing theoretical results for them. However, most of the used concepts and ideas are motivated by empirical observations on the genetic information of certain species. Examples for such observations are the mechanisms that are assumed to shape gene arrangements of mitochondrial genomes, see Section 2.1.3. Therefore, this chapter gives a short introduction to the structure of DNA and molecular evolution.

This section is organized as follows: Section 2.1.1 gives an overview of the structure of DNA and its genetic aspects. Section 2.1.2 presents a brief review of mechanisms shaping the genetic information of species during evolution. The DNA of mitochondria serves as running example for an application of the theory and the algorithms which are presented in this thesis. Therefore, Section 2.1.3 aims to outline general aspects of mitochondrial genomes such as their origin and types of evolutionary processes driving their evolution. In

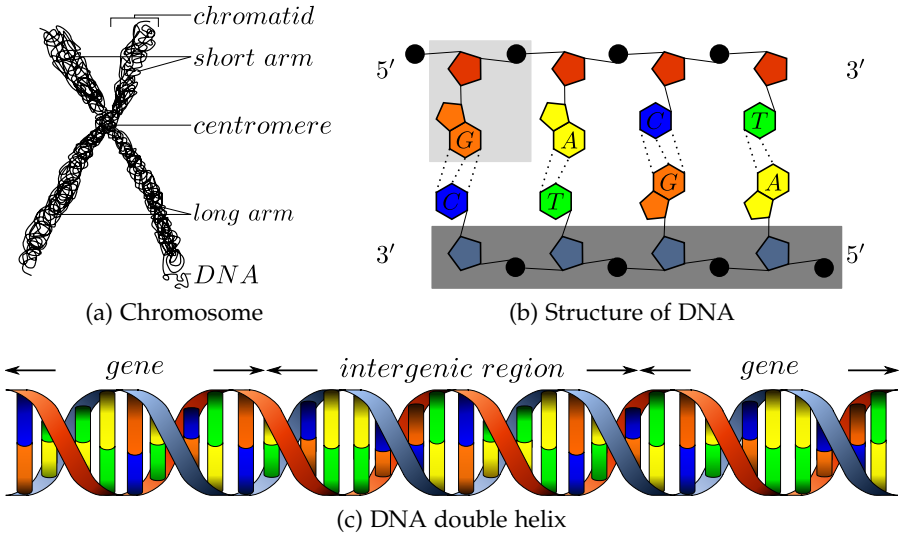
recent years mitochondrial gene arrangements have gotten a lot of attention for the reconstruction of phylogeny. Section 2.1.4 summarizes the merits and demerits of gene arrangements analyses for phylogeny inference.

2.1.1 *What is DNA?*

Every living organism on earth is made up of individual and identifiable cells. The number of cells of an organism varies from one cell in *bacteria* to trillions of cells, e. g., in the human body (Bianconi et al., 2013). Every cell itself has all characteristics of live: it arises, it metabolizes, it reproduces, and perishes eventually. The processes and the workings within a cell guaranteeing its viability are extremely complex and raise many interesting research questions in a variety of areas. It is, however, generally accepted that the blueprint for the construction and functionality of a cell (and therefore the organism) is encoded in one or more molecules of *deoxyribonucleic acid (DNA)* forming the hereditary basis of every living organism.

Physically, the DNA of an organism may be divided into a number of different DNA molecules called *chromosomes* which altogether make up the *genome* of an organism. Figure 2.1 (a) illustrates a typical appearance of a chromosome in an animal cell. The primary building block of the DNA is called *nucleotide*. It consists of three components: a single nitrogenous base of *adenine*, *cytosine*, *guanine*, or *thymine*; a sugar called deoxyribose; and at least one phosphate group, see Figure 2.1 (b) for an illustration. A long succession of nucleotides that are bonded together by covalent bonds between the phosphate and the sugar form a *polynucleotide strand* with an alternating sugar-phosphate backbone. The complementary nitrogenous bases of two separate polynucleotide strands tend to pair by hydrogen bounds: adenine and thymine are chemically attracted to each other as are cytosine and guanine, forming a double-stranded helically coiled macromolecule, the DNA (Watson and Crick, 1953). Figure 2.1 (c) illustrates the double helix shape of the DNA. The result of this process is that the DNA is composed of two strands which are the complement of each other with each strand uniquely distinguishable by its alternating sugar-phosphate backbone which implies a directionality from sugar to phosphate (called $3'$ end and $5'$ end) and vice versa. The order of the nucleotides, each abbreviated with the first letter of the nitrogenous base it contains (i. e., either A, C, G, or T), within the strand from the $5'$ end to the $3'$ end serve to represent the DNA of an organism as succession of letters, the *nucleic acid sequence* or *DNA sequence* for brevity. An example for a DNA sequence is given in Figure 2.1 (b).

Functionally, a genome is divided into *genes* which are consecutive sequences of DNA which encode the information to construct other molecules such as proteins or ribonucleic acids (e. g., tRNAs and rRNAs). It is worth mentioning that alternative concepts defining a gene as a combination of structural and functional components



(a) Chromosome (b) Structure of DNA

(c) DNA double helix

Figure 2.1: Schematic representations of DNA and its structure. (a) Typical structure of a eukaryotic chromosome; (b) DNA as a sequence of two bonded polynucleotide strands. Illustrated are a nucleotide (bright gray square); the nitrogenous bases adenine (yellow nine-sided shape), cytosine (blue hexagon), guanine (orange nine-sided shape), and thymine (green hexagon); the sugar deoxyribose (blue and red pentagon); a phosphate group (black filled circle); one sugar-phosphate backbone (dark gray square); and hydrogen bonds (dotted lines). The nucleic acid sequence of the exemplified DNA in 5′ end to the 3′ end direction is GACT and its complementary nucleic acid sequence is AGTC. (c) The double-stranded helically coiled shape of the DNA. Nucleotides and sugar-phosphate backbones are illustrated in the same color as in (b). Sequences of DNA that encode a specific function are called genes and the sequence between two genes is called intergenic region.

of a genome are subject of intense discussions, e. g., see Gerstein et al. (2007) and Stadler et al. (2009). A gene can appear on either of both strands of the DNA and each chromosome may contain a large number of genes, e. g., 19000 protein-encoding genes in human (Ezkurdia et al., 2014). The succession of the genes and their partition into chromosomes, commonly termed as its *gene order*, is known to have a crucial influence on an organism's cellular processes (Li and Reinberg, 2011). Sequences of DNA located between genes are called *intergenic regions*. There are indications that intergenic DNA has an influence on genes nearby, however most of its function is currently unknown (ENCODE Project Consortium, 2012).

Since the publishing of the pioneering work done by Woese et al. (1990) all life on earth is classified into the three different domains *Archaea*, *Bacteria*, and *Eukaryota* which form the highest taxonomic rank of an organism[1]. Based on their level of cellular organization the first two domains are also grouped together, forming the *prokaryotes*. Prokaryotes and eukaryotes are very different in various aspects, e. g.,

1 Intriguingly, since non of these three domains includes non-cellular life, it is still under debate whether or not (large) viruses should be considered as a fourth domain, as it has been proposed by, e. g., Nasir et al. (2012).

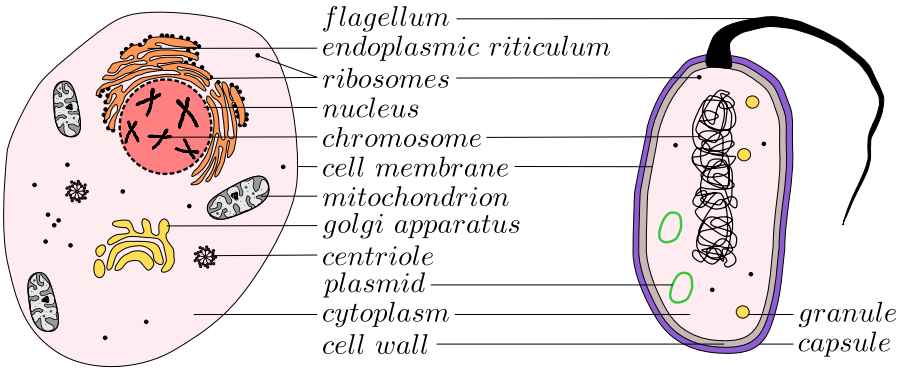


Figure 2.2: Schematic structure of a eukaryotic animal cell (left) and a prokaryotic cell (right).

see Figure 2.2 for a comparison of the structure of a eukaryotic cell and a prokaryotic cell. However, some primary differences that hold true for the majority of the cells are specified: eukaryotic cells contain a *nucleus*, a membrane-enclosed subunit within a cell that contains DNA whereas prokaryotes do not have a nucleus. A second difference is that prokaryotes are single-celled organisms whereas eukaryotic cells usually combine to form multicellular organisms. Lastly, the number and the structure of the chromosomes is prevalently different: while eukaryotes tend to have multiple chromosomes that are *linear*, i. e., each chromosome has two endpoints called *telomeres*, prokaryotes tend to possess a single *circular* chromosome, i. e., the chromosome forms a circular molecule which does not have telomeres. A genome containing only a single chromosome is called *unichromosomal* and a genome that contains multiple chromosomes is called *multichromosomal*.

The genome of a cell does not solely include nuclear DNA but also additional genomic DNA sequences which are separated from the nucleus, e. g., prokaryotic cells often contain small circular DNA molecules called *plasmids* and eukaryotes often exhibit organellar DNA of *chloroplasts* in plant and algal cells which derive energy from photosynthesis. Furthermore, nearly all eukaryotes have a subunit called *mitochondrion* that is responsible for supplying the cell with energy (Bernt et al., 2013b; McBride et al., 2006). As outlined in Chapter 1, the DNA sequence of the mitochondrion which is also called *mitochondrial genome*, and the arrangement of its genes is well suited for the study of evolutionary relationships of species and is the main application for the theory and the algorithms presented in the following chapters.

2.1.2 *Where there is DNA, there must be mutations!*

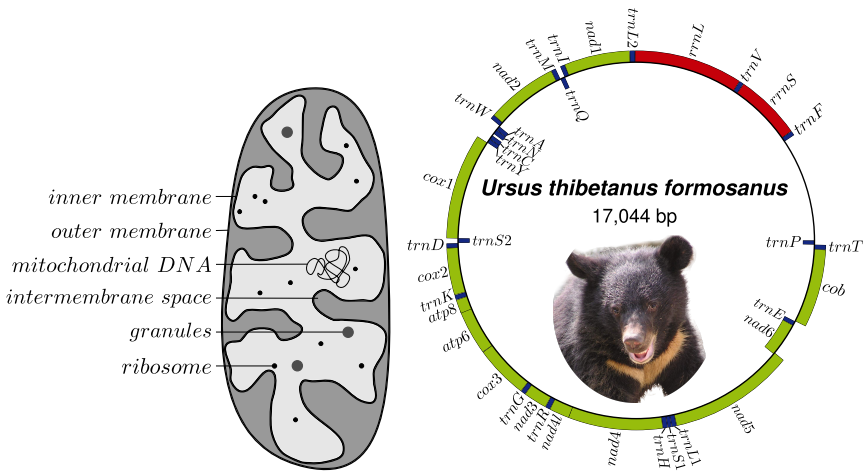
All living species are subject to various kinds of mutations that shape their genetic information. Many of these changes in the DNA induce an effect on the fitness of species in nature, e. g., they produce discernible changes in the phenotype of an organism (Sturtevant and Dobzhansky, 1936a), prevent some genes from functioning properly,

and cause death (Krebs et al., 2014). Occasionally, a mutation either has no effect on the functionality of a gene or it alters the product of a gene which leads to the survival of the cell. As a result, offspring with a slightly different genome is produced and the mutation may be passed on through the generations. This inaccuracy in the reproduction process as well as the biological fitness it implies on species, is the principle of *molecular evolution*.

The DNA is subject to various kinds of mutations which are classified by the extent of genetic information they affect. *Small-scale mutations* (commonly called *point mutations*) affect single nucleotides or a few nucleotides by either adding one or more extra nucleotides into the DNA (*insertion*), removing one or more nucleotides from the DNA (*deletion*) or exchanging a single nucleotide for another (*substitution*). The detection of these small-scale evolutionary events is the goal of *sequence alignment*, see Jones and Pevzner (2004) and Rosenberg (2009) for an introduction into this topic. *Large-scale mutations* modify the arrangement of genes or their quantity on the chromosomes by acting on large segments of DNA rather than single nucleotides. They are categorized into *genome rearrangements* (or *rearrangements* for brevity) which alter the organization of the genes in the genome, e. g, genes are moved to other positions or their complementary strand in the genome; and *content modifications* which change the quantity of genes on the chromosomes by adding, removing, or duplicating a gene (Dörr, 2016). The study of large-scale mutations is the subject of *gene order analysis*, e. g., see Fertin et al. (2009) for a recent overview.

Large-scale mutations are caused by a considerable amount of varying processes whereby the following two mechanisms are well established. As every living organism on earth, cells are mortal and therefore they must reproduce to ensure their survival. A significant part of the cell reproduction is the replication of its genome. Regardless of the type of the replication which again is fundamentally different in eukaryotes and prokaryotes, this process is not flawless, e. g., mechanisms such as slipped strand mispairing (Levinson and Gutman, 1987) and imprecise termination (Mueller and Boore, 2005) during replication are assumed to cause duplicated or partially missing genetic sequences in mitochondrial genomes. Another event that shapes the genetic information of species is an occasionally occurring break of chromosomes caused by some outside force, e. g., a chemical mutagen (Drake and Baltz, 1976) or oxidative damage (Harman, 1972). Finally, the repair process meant to prevent mutations is sometimes error-prone (Alexeyev et al., 2013) which leads to a change of the genome organization (Gredilla, 2011; Rodgers and McVey, 2016).

The genomes of species are not modified arbitrarily during the course of evolution, but rather certain types of rearrangements have been deduced from comparative analysis of closely related species. The types of observed rearrangements depend on the way they influence the genetic information. For example, one aspect in the classification of genome rearrangements is the number of chromosomes which are influenced by a rearrangement. Section 2.3 provides an overview on computational and algorithmic results in this area.



(a) Structure of a mitochondrion (b) Mitochondrial genome

Figure 2.3: (a) Structure of a mitochondrion; (b) modified image of the circular and unichromosomal mitochondrial genome of the Taiwanese black bear *Ursus thibetanus formosanus* (NCBI reference sequences database (Pruitt et al., 2007); accession: NC_009331.1); the image is generated by `OGDRAW` (Lohse et al., 2007); the genes are colored with respect to product they encode: proteins (green), rRNAs (red), and tRNAs (blue). (The image of *U. thibetanus* is published under Creative Commons licence.)

2.1.3 *What are mitochondria and how do they evolve?*

Mitochondria are small enclosed subunits within the cytoplasm of almost all eukaryotic cells. They are enclosed in a double-layered membrane, see Figure 2.3 (a) for an illustration of the structure of a mitochondrion. Mitochondria supply most of the *adenosine triphosphate (ATP)* that powers the cells metabolic activities by taking energy from the oxidation of food molecules (e. g., glucose) and oxygen (Alberts and Walter, 2003). This is the reason why the mitochondrion is called the *powerhouse of the cell*. In addition, the mitochondrion is also heavily involved in many other cellular processes such as the control of the cell cycle, cell growth, and the regulation of cellular metabolism (McBride et al., 2006). Initiated by pioneering works of Wallace et al. (1988) and Holt et al. (1988) which have described a direct connection between mutations of mitochondrial DNA and human genetic diseases, it is now widely believed that mitochondrial defects are implicated in age-dependent neurological diseases such as Alzheimer's disease (Wallace, 2005) and Parkinson's disease (Abou-Sleiman et al., 2006).

Intriguingly, mitochondria show the evolutionary connection between prokaryotes and eukaryotes. While mitochondria have many general similarities with certain bacteria such as *Rickettsiales proteobacteria* (Thrash et al., 2011), e. g., the unichromosomal circular structure of the genome and some genes which are clearly of prokaryotic origin (Gissi et al., 2008), some mitochondrial genes possess introns (i. e., specific regions inside a gene) that resemble eukaryotic nuclear genes (Rot et al., 2006; Vallès et al., 2008). It is almost certain that mitochon-

dria originated from free-living oxygen-metabolizing bacterial ancestors by a process called *endosymbiosis* (Gray, 1989; Krebs et al., 2014; Lang et al., 1999; Martijn et al., 2018; Thrash et al., 2011). In this process a bacterial cell was engulfed by a eukaryotic prototype. Escaping digestion, the bacterial cell dwelled within the cytoplasm of its host cell receiving shelter and nourishment in return for providing the ability to make use of oxygen to produce energy, see Lang et al. (1999) and Scheffler (2011) for a more detailed description. This symbiotic partnership is thought to has been established about 1.35 billion years ago (Alberts and Walter, 2003). However, the identity and nature of the mitochondrial ancestor is still controversial (Martijn et al., 2018).

Mitochondrial genomes are generally small genomes that encode a restricted number of functions. They are usually (but not always) circular unichromosomal and the total size of mitochondrial genomes can vary by more than an order of magnitude, e. g., the mitochondrial genomes of mammals are small in size of approximately 16.6 kilo base pairs (Chan, 2006) while fungal mitochondrial genomes are considerably larger with approximately $19 - 100$ kilo base pairs (Krebs et al., 2014). Mitochondrial genomes are extremely compact and their gene content is well conserved in animals, i. e., duplicated and missing genes are unusual (Boore and Brown, 1998; Oxusoff et al., 2018). Metazoan mitochondrial genomes typically encode 37 genes: 13 proteins (*atp6/8*, *cob*, *cox1-3*, *nad1-6*, *nad4l*), 2 ribosomal subunits of mitochondrial ribosomes (rRNAs) (*rrnL*, *rrnS*), and 22 transfer ribonucleic acid (tRNA) (*trnA-trnY*, *trnL1/L2*, *trnS1/S2*), see Figure 2.3 (b) for an illustration of the mitochondrial genome of the Taiwanese black bear *Ursus thibetanus formosanus*.

Different notations are commonly used to represent gene arrangements in the literature, see Section 2.2.1. Mitochondrial genomes are usually represented as a sequence of genes each assigned to a sign $+$ or $-$ with respect to their strandedness, see Example 2.1. In sequences that contain every gene exactly once; this is consistent with the concept of a (signed) permutation, see Section 2.2.1.

Example 2.1. *A potential representative of the mitochondrial genome of* U. thibetanus formosanus *illustrated in Figure 2.3 (b) is: cox1 -trnS2 trnD cox2 trnK atp8 atp6 cox3 trnG nad3 trnR nad4l nad4 trnH trnS1 trnL1 nad5 -nad6 -trnE cob trnT -trnP trnF rrnS trnV rrnL trnL2 nad1 trnI -trnQ trnM nad2 trnW -trnA -trnN -trnC -trnY.*

The nucleotide composition of both complementary polynucleotide strands has been found to differ asymmetrically in mitochondrial genomes, e. g., one strand has been reported to be rich in G (and T) the other is G (and T) lacking (Reyes et al., 1998). Due to different proportions of heavier nucleotides both strands have different masses, therefore the heavy strand is termed *H-strand* and the light strand is called *L-strand*.

During its evolution most of the genes of the mitochondria have been transferred to the nuclear DNA, thereby lacking genes that are necessary for independent life (Adams et al., 2000; Thorsness and Fox, 1990). Interestingly, those nuclear DNA sequences of mitochondrial

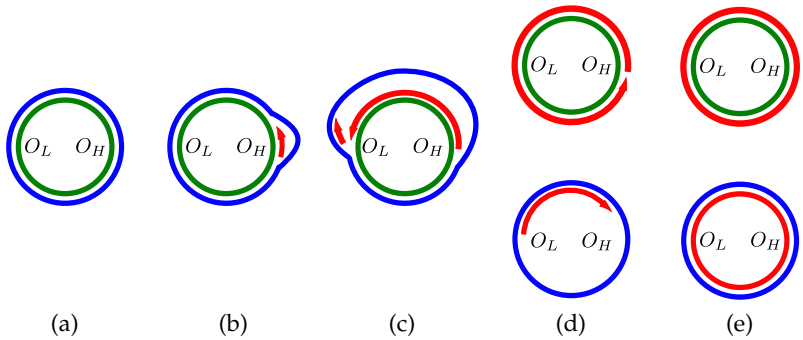


Figure 2.4: Schematic representation of the strand displacement model of mitochondrial replication. Illustrated are the H-strand (blue); L-strand (green); newly synthesized strands (red); and the heavy (light) strand origin O_H (respectively O_L). (a) Mitochondrial genome before replication; (b) H-strand synthesis has been initiated at O_H and proceeds towards O_L. Synthesis proceeds replacing the original H-strand, thereby displacing the original H-strand which remains single-stranded, forming a *displacement-loop (D-loop)*. (c) D-loop expands until it reaches O_L, where L-strand synthesis is initiated, proceeding in opposite direction to the H-strand synthesis. (d) H-strand synthesis finishes and segregates while L-strand synthesis has proceeded only a part of the strand (in mammals this part is one third of the way around the circle (Clayton, 1991)). (e) L-strand synthesis is finished after the segregation. Finally, the newly synthesized strands are sealed and form two daughter genomes of covalently closed circular DNA, where every genome possesses one parental strand as template.

origin can form a noticeable fraction of a species' nuclear genome (Hazkani-Covo and Graur, 2006). While the gene content in mitochondrial genomes is mostly conserved, there are extensive differences in the arrangement of the genes in distinct taxonomic groups across the *Metazoa* (Boore, 1999; Bernt et al., 2013b). Aguileta et al. (2014) verified that fungal mitochondrial gene orders also display a remarkable variation within the major fugal phyla.

Deviations from the typical genomic organization of a metazoan mitochondrial genome are rare and appear to be restricted to individual clades (Bernt et al., 2013b). Such aberrant genome structures include tRNA losses (Jühling et al., 2011), loss of protein-coding genes (Lavrov and Pett, 2016), disintegration of the mitochondrial genome into multiple small chromosomes (Shao et al., 2009), linear chromosomes (Kayal et al., 2011), and mitochondrial genomes composed of linear and circular chromosomes (Raimond et al., 1999). For a survey on aberrant mitochondrial genome structures the reader is referred to Bernt et al. (2013b).

A common phenomenon of mitochondria is its ability to replicate independently from the replication of the nuclear genome. The commonly accepted model for mitochondrial replication is the *strand displacement model* (Clayton, 1982; Clayton, 1991; Robberson et al., 1972; Shadel and Clayton, 1997), see Figure 2.4 for a brief depiction and Scheffler (2011) for a detailed description of the replication process.

It is also worth mentioning that alternative replication models, i. e., the *strand-couple-model* (Holt et al., 2000; Holt, 2009) and a model involving recombination presented in Pohjoismäki et al. (2010), have been proposed. For a critical discussion on the different models of mitochondrial replication see, e. g., Pohjoismäki and Goffart (2011).

A comparison of the mitochondrial genome of closely related species that exhibit different mitochondrial gene arrangements implies that at least five types of genome rearrangements, see also Figure 2.5, are relevant for the evolution of mitochondrial genomes (Bernt et al., 2013b; Boore, 1999):

- *Transposition* (e. g., Macey et al. (1997)): A continuous sequence of genes is moved to another position on the same chromosome.
- *Inversion or reversal* (e. g., Asakawa et al. (1995)): A continuous sequence of genes is reversed in the chromosome. In consequence, the gene order of the involved sequence is reversed and the direction of each affected gene is flipped.
- *Inverse transposition* (e. g., Boore et al. (1998)): A continuous sequence of genes is transposed to another position on the same chromosome where it is inserted inversely.
- *Tandem duplication random loss (TDRL)* (e. g., Boore (2000)): A continuous sequence of genes is duplicated in tandem followed by the random loss of one copy of each duplicated gene.
- *Inverse tandem duplication random loss (iTDRL)* (e. g., Jühling et al. (2011)): A continuous sequence of genes is duplicated in tandem in a way that the duplicated sequence is inverted, followed by the random loss of one copy of each duplicated gene.

Up to now, it is still not entirely clear whether the observed rearrangements deduced from the comparative analyses correspond to distinct molecular mechanisms. The reason is that the observed rearrangements can also be a product of a composition of several subsequent steps, e. g., the effect of an inverse transpositions can be obtained by a composite of a transposition and an inversion. Another example is the transposition model which can also be interpreted as a special case of the TDRL rearrangement model. However, convincing arguments have been found for the TDRL (San Mauro et al., 2005) and the iTDRL (Jühling et al., 2011) models by detecting remnants of such processes as intergenic regions at positions where the deleted genes would be expected (Bernt et al., 2013b). While the TDRL model is well established in the evolution of metazoan mitochondrial genomes, the support for the iTDRL rearrangement model has just recently been proposed, see Jühling et al. (2011). Consequently, the role of the differently types of rearrangements in mitochondrial gene order evolution is still not entirely clear and should be treated with caution. Moreover, the mechanisms leading to these rearrangements are subject of controversal discussions. While intra-mitochondrial recombination could provide plausible explanations for inversions, transpositions and inverse transpositions (Mueller and Boore, 2005), it fails to explain the

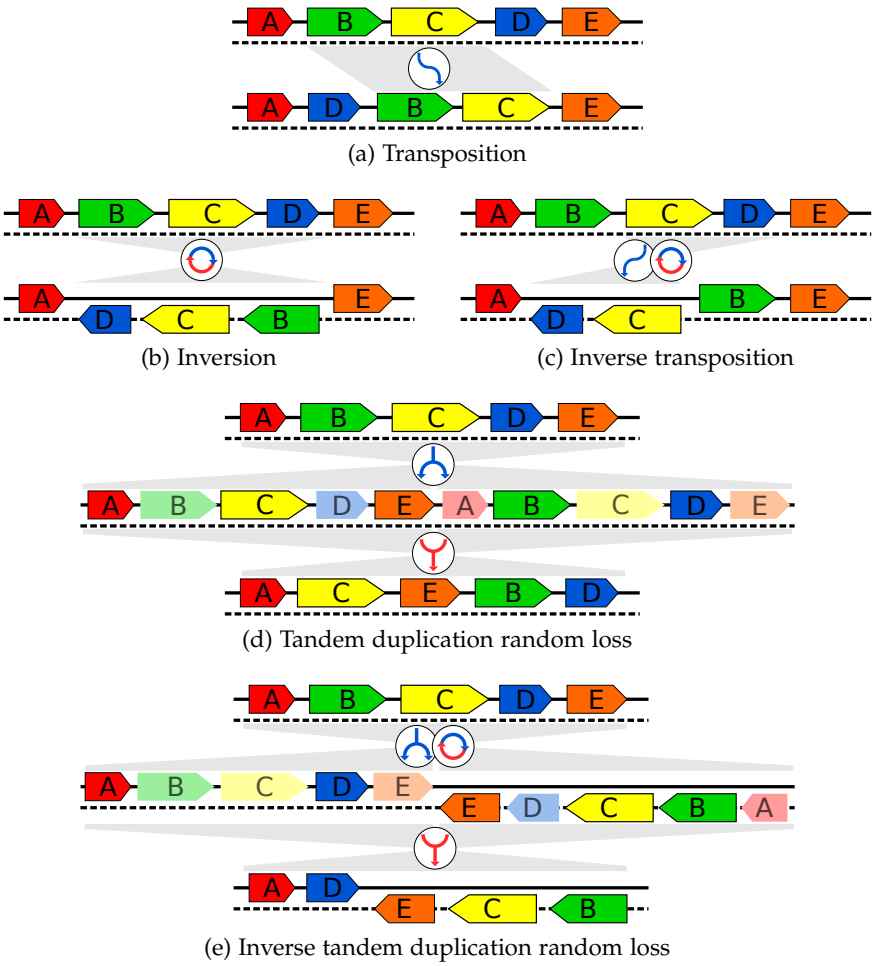
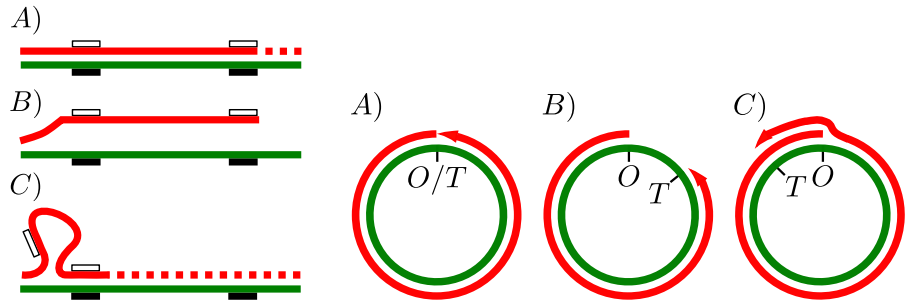


Figure 2.5: Elementary types of mitochondrial rearrangements exemplified for an artificial gene order of five genes (arrows A-E): (a) transposition; (b) inversion; (c) inverse transposition; (d) tandem duplication random loss; (e) and inverse tandem duplication random loss. The development of pseudogenes is represented by transparent genes without borders. The two DNA strands are illustrated by a continuous and a dashed black line and the orientation of a gene is represented by its location on one of the two strands.



(a) Slipped strand mispairing (b) Imprecise termination

Figure 2.6: Errors in the mitochondrial replication process which may result in duplicated genes: (a) slipped strand mispairing and (b) imprecise termination. Illustrated are one parental strand (green line) and the complementary newly synthesized strand (red line). (a) Ordinary replication process partially replicates the parental strand encountering genetic sequences with high similarity (black squares, the corresponding sequences on the complementary strand are illustrated by white squares) (A); replication suspends and the newly synthesized strand temporarily separates from parental template strand (B); the high sequence similarity allows the newly synthesized strand to pair to the parental strand at an upstream location and replication proceeds (dotted red line) leading to a tandem duplication of the slipped sequence (C). (b) In the ordinary replication process the origin (O) and termination (T) location of replication on the parental strand coincide, an identical replicate is produced (A); premature termination of replication leads to an incomplete synthesized strand lacking the sequence between T and O (B); missing the ordinary termination location followed by an alternative termination leads to a replicate with (partially) duplicated genetic sequences (C).

existence of pseudogenes (i. e., segments of DNA which are related to real genes and have lost its functionality as a result of disabling mutations) which require at least one duplication event (Macey et al., 1998).

A plausible explanation for the existence of pseudogenes is given by the tandem duplication random loss rearrangement model. In this model, the existence of pseudogenes is expected to represent evolutionary intermediate steps as briefly explained in the following. The TDRL rearrangement event is initiated by errors such as slipped strand mispairing (Levinson and Gutman, 1987) and imprecise termination (Mueller and Boore, 2005) during replication causing duplicated genes in mitochondrial genomes. Figure 2.6 illustrates both mechanisms, for a detailed explanation the reader is referred to Boore and Brown (1998) and Boore (2000). Alternative models such as dimerization (Raimond et al., 1999) and recombination (Awadalla et al., 1999; Lunt and Hyman, 1997; Mao et al., 2013) may also explain gene duplication in mitochondrial genomes, however, these models are still under debate. After the duplication, whichever copy of a duplicated gene that experienced the first disabling mutation is determined for turning into a pseudogene by subsequent accumulation of point mutations such as substitutions and deletions (Castellana et al., 2011). Strong selection pressure on the size and the gene num-

ber of the mitochondrial genome will then rapidly remove the non-functional pseudogenes (Wolstenholme, 1992). Remnants of such degenerating pseudogenes as intermediate steps fully satisfying predictions from the TDRL model have been detected for several groups of species, e. g., *Echinodermata* (Arndt and Smith, 1998), *Caecilian* (San Mauro et al., 2005), *Vertebrata* (Macey et al., 1997), and *Amphibia* (Xia et al., 2016), which suggests that TDRLs are a major mechanism in the evolution of metazoan mitochondrial genomes. Especially for vertebrate mitochondrial genomes, TDRL is considered to be the most important mechanism explaining gene order rearrangements, e. g., see San Mauro et al. (2005). Large scale analyses on the properties of mitochondrial gene order evolution of *Hymenoptera* (Dowton et al., 2009) and *Metazoa* (Bernt and Middendorf, 2011; Miklós and Hein, 2004) support this hypothesis by showing that transpositions, which can be interpreted as special cases of the TDRL model, are the majority of the reconstructed rearrangements. It has also been shown that three-fourths of the reconstructed rearrangements only affect tRNA genes, which suggests that the position of mitochondrial tRNAs is selectively neutral, i. e., a change in their position is adaptive with respect to the functionality of a gene (Brown, 1985; Dowton et al., 2009). Therefore, the position of these genes may not be informative to infer evolutionary information.

In a variant of the TDRL model it was suggested that the deletion process is not random for each single gene. Instead, genes belonging to the same transcript are lost jointly due to deleterious mutations in the promotor (Beckenbach, 2011; Lavrov et al., 2002). Motivated by evidence from comparative analyses (Jühling et al., 2011; Kong et al., 2009) and the fact that inverted duplications often occur in the control region of *Insecta* mitochondrial genomes (Liu et al., 2017), the inverse tandem duplication random loss (iTDRL) model has recently been proposed. Up to now, it is not entirely clear whether the iTDRL model corresponds to a distinct molecular mechanism or to a consecutive occurrence of TDRL and an inversion. However, additional support for the iTDRL model has recently been provided by Shi et al. (2013). In their work the authors suggested a special case of the iTDRL mechanism termed *dimer-mitogenome and non-random loss* in which the loss of a gene is not random but dependent on the polarity.

Since inversions and inverse transpositions cannot be explained by the TDRL model alone (recall that TDRLs cannot affect the orientation of a gene), there might be several different molecular mechanisms affecting the organization of mitochondrial genes. Another explanation may be given by iTDRL model which can explain both pseudogenization and modification gene orientation.

2.1.4 *Mitochondrial Gene Orders for Phylogeny Inference*

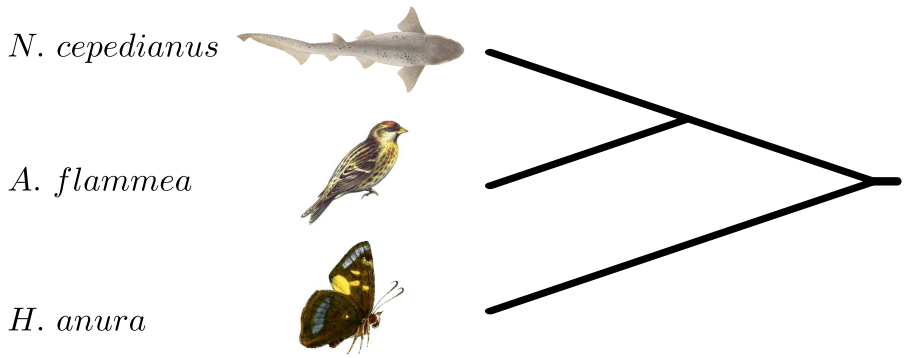
Understanding the origin of the rich diversity of living beings is one of the central problems of evolutionary biology. Although it may be impossible to unequivocally infer how species have evolved, mankind

is already able to reliably hypothesize on the evolutionary history of some species. A common form to represent such a hypothesis on the evolutionary history of a collection of organisms is a *phylogenetic tree* or for brevity a *phylogeny*. The information on evolutionary relationships that are necessary for the reconstruction of phylogenies are drawn from genetic features of contemporary species. Thereby, phylogenetic conclusions are based on the principle that the evolutionary history of species can be reconstructed from similarities between the considered organisms, i. e., a high similarity of some genetic features indicates a close evolutionary relation (Xia, 2013). Different features can be regarded for this task, e. g., morphology (i. e., the form and the structure of organisms), DNA sequences, and gene arrangements. An example of a genomic feature that is frequently used to infer phylogenies is the similarity of the DNA sequences of certain genes, e. g., the genes' coding for the large subunit of eukaryotic ribosomes. Many of the existing methods deduce good estimations of phylogenies based on sequence information. However, they suffer from the similar limitations such as the impact of *homoplasy*, i. e., multiple point mutations at the same position lead to species independently sharing a feature (e. g., similar DNA sequences) that is mistakenly interpreted as mutual information of ancestry. It becomes even more challenging to reconstruct deep evolutionary relationships as the impact of homoplasy becomes greater the further back one looks (Moret et al., 2002; Moret and Warnow, 2005; Nakhleh et al., 2001). Another limitation is the lack of genomic resources, e. g., through large variations in the size of DNA sequences as in *Crustacea* (Tan et al., 2018).

A feature that has been proven to be powerful for larger-scale comparisons that focus on ancient relationships (San Mauro et al., 2005) are mitochondrial gene arrangements (Bernt et al., 2013b; Boore, 2006). The benefit of using (mitochondrial) gene arrangements for phylogeny inference is that it considers the entire (mitochondrial) genome. This reflects organismal evolution, rather than just segments of the DNA sequence such as *cob* or *16S ribosomal RNA*, in combination with a simplified representation of the genome that allows to ignore point mutations. Instead, phylogenetic information is inferred only on the basis of gene content and gene order (Moret and Warnow, 2005). A common way to do this is to represent two genomes by their order of genes and to define an evolutionary distance measure that aims to approximate the actual number of evolutionary events. This method has the advantage that computationally hard problems that occur in the presence of gene and species trees can be avoided, e. g., see Felsenstein (2004) and Maddison (1997). Since the true number of evolutionary events cannot be inferred in consequence of homoplasy, the minimum number of evolutionary events that can transform one genome into another, i. e., the *edit distance*, is used instead. Given such an edit distance, phylogenetic trees can be obtained by a two-staged procedure: Evolutionary conclusions are drawn by computing pairwise distances between a collection of different gene orders. In the second step a phylogenetic tree has to be found that is consistent with these distances. Figure 2.7 demonstrates such a procedure using

	N. cepedianus	*A. flammea*	*H. anura*
Notorynchus cepedianus	0	1	12
Acanthis flammea	1	0	14
Hasora anura	11	12	0

(a) Pairwise edit distance matrix



(b) Phylogenetic tree

Figure 2.7: Minimalist example of phylogeny inference based on pairwise edit distances. (a) Pairwise edit distance matrix of the mitochondrial gene orders of *Notorynchus cepedianus* (NC_022731.1), *Acanthis flammea* (NC_027285.1), and *Hasora anura* (NC_027263.1). The corresponding gene orders were obtained from the NCBI RefSeq (Pruitt et al., 2007). The edit distances were computed by `CREx2`, see Chapter 5.4. `CREx2` considers the following types of rearrangements: transposition, inversion, inverse transposition, and TDRL. A close evolutionary relationship of *N. cepedianus* and *A. flammea* can be inferred by the observation that one rearrangement separates their gene orders. However, several rearrangements are necessary to obtain the gene order of *H. anura* from *N. cepedianus* or *A. flammea* (and vice versa) indicating a distant relationship. (b) A corresponding phylogenetic tree that is consistent with these observations. (The image of *N. cepedianus* and *H. anura* is published under Creative Commons licence with the former one being given from the Naturalis Biodiversity Center; the image of *A. flammea* is taken from Naumann (1900) "Naturgeschichte der Vögel Mitteleuropas".)

a minimalist example of three mitochondrial gene orders. Indeed, the example illustrated in Figure 2.7 is simplified and there may be cases in which there is no phylogenetic tree that is consistent with the pairwise distances. However, in such cases the pairwise distances may be interpreted as helpful hint indicating evolutionary relatedness in a framework that considers a variety of genomic features, e. g., see Perrin et al. (2015).

The usage of mitochondrial gene orders as a source of phylogenetic information has exceptional advantages:

- Mitochondrial gene orders are small in size and extremely compact as typically only 37 genes are encoded, see Section 2.1.3. Furthermore, mitochondrial gene orders have been determined for a wide range of species, e. g., currently more than 10^4 mito-

chondrial gene orders are available on the National Center for Biotechnology Information database (Pruitt et al., 2007).

- The gene content of mitochondrial genomes is nearly invariant and provides a unique data set that facilitates broad comparisons, e. g., almost all genes commonly found in metazoan mitochondrial genomes have *homologs* (i. e., genes that share the same ancestry) in mitochondrial genomes of plants, fungi, and protists (Boore and Brown, 1998).

- Large-scale mutations such as genome rearrangements are assumed to be – with respect to point mutations and the exception of certain taxa such as *Tunicata* (Iannelli et al., 2007) – rare genomic events because functional genomes must be maintained to ensure survivability. Therefore, the gene order of mitochondrial genomes is believed to evolve slowly (Rokas and Holland, 2000) potentially retaining the signal of ancient common ancestry (Boore and Brown, 1998).

- Different mitochondrial gene orders and the potential options of genome rearrangements affecting them are great in number which limits the impact of homoplasy (Moret and Warnow, 2005; Rokas and Holland, 2000), i. e., it supports the concept that gene orders are shared only as a result of common ancestry (Boore et al., 1995; Boore and Brown, 1998).

- The inheritance on metazoan mitochondrial genomes is strictly maternal with a single exception in unionid mussels (Breton et al., 2010), therefore recombination between parental genomes does not occur (Krebs et al., 2014; Scheffler, 2011) which limits the number of mechanisms that simultaneously shape metazoan mitochondrial gene orders.

Several studies using mitochondrial gene orders have been shown to be valuable for the support of phylogenetic hypotheses, e. g., for *Annelids* (Bleidorn et al., 2007), *Annomura* (Tan et al., 2018), *Cnidaria* (Kayal et al., 2015), *Coleoptera* (Yuan et al., 2016), and *Arthropoda* (Xu et al., 2006), characterizing unique gene rearrangements on ancient relationships, e. g., in birds (Härlid et al., 1997), marsupials (Janke et al., 1994), and echinoderms (De Giorgi et al., 1996; Scouras and Smith, 2001), and being a rich source of phylogenetic information even on more recent taxonomic levels, e. g., see Basso et al. (2017), Gan et al. (2018), Tan et al. (2017), and Weigert et al. (2016).

As pointed out in the groundbreaking work done by Darwin (1859), a single feature is not always sufficient for the classification of evolutionary relationships. Although mitochondrial gene orders have helped to broaden our understanding of phylogenetic relationships at different taxonomic levels, it has, at times, been disappointing to rely on gene orders to infer evolutionary relationships (Shao et al., 2003). An explanation for this outcome may be an apparent lack of an evolutionary signal in some lineages due to an extreme variety of

gene orders. For example, the mitochondrial gene orders in some lineages have been unchanged for long periods of time, e. g., human and shark share the same mitochondrial gene order (Boore, 2000). Therefore, no evolutionary signal may have been accumulated during the period of shared history. Another example is given by lineages such as *Mollusca* (Hoffmann et al., 1992; Yamazaki et al., 1997) in which mitochondrial gene orders can highly vary. Thereby, the evidence of relatedness might have been eroded by unusually high rates of molecular evolution leading to homoplasious gene orders (Bernt et al., 2013a; Oxusoff et al., 2018). Although the inference of phylogeny might not be resolved for some lineages, it is still believed that gene arrangements may serve as a model for interpreting broader aspects of genome evolution (Boore and Brown, 1998).

2.2 FORMAL BACKGROUND

The following sections provide an introduction to the mathematical objects that serve as formal models facilitating a computational comparison of gene orders. In particular, Section 2.2.1 focuses on formal models which represent biological gene arrangements. In nature one can often observe sets of genes that are similar in close proximity in several gene arrangements. One particularly interesting approach of gene order analysis aims to reconstruct gene order evolution while preserving such sets of genes. In Section 2.2.2 an introduction to this approach is given. The evolutionary mechanisms that are relevant for the evolution of metazoan mitochondrial gene arrangements are defined in Section 2.2.3. The solution of different combinatorial optimization problems can be utilized to infer phylogenetic information from gene order data. Section 2.2.4 describes the central combinatorial problems that are relevant for this thesis.

2.2.1 *Gene Orders and Permutations*

This section presents models that facilitate the study of gene orders on an abstract level. In order to do so, certain assumptions are imposed on the genomes that are modeled in this thesis: 1) Every genome contains exactly one copy of each gene; 2) genes do not overlap; and 3) all considered genomes contain the same set of genes. While these assumptions are very plausible for the representation of metazoan mitochondrial gene orders, they are too restrictive to model gene orders of nuclear genomes in which gene duplications can be spotted frequently (Shao and Moret, 2017).

Gene arrangements are commonly represented in the literature by permutations or adjacencies. The focus of this work lies in representing linear (or circular) unichromosomal genomes as a sequence of genomic markers that appear only once in the genome. With this assumption, such sequences are correctly modeled by permutations in which each element of the permutation represents one genetic marker in the chromosomes, usually a gene. Different kinds of permutations

can be regarded for this task. While linear chromosomes are modeled by linear permutations, circular chromosomes are represented by circular permutations. Another aspect that may be regarded is the double-stranded structure of DNA. This is done by adding a sign to each element of the permutation that represents the strandedness of the corresponding genetic marker. If the strandedness is considered in the representation of unichromosomal genomes, one speaks of *signed (linear/circular) permutations* and otherwise, i. e., if the representation of gene orientation is not of interest, *unsigned (linear/circular) permutations*. Another aspect that is taken into account by modeling gene orders is that chromosomes of species usually have no preferred reading direction, hence they may be read in both directions. If this information is regarded in the representation of gene orders, one speaks of *undirected permutations*, otherwise they are called *directed permutations*.

With a few exceptions, the representation of unichromosomal genomes as permutations is especially well suited for metazoan mitochondrial genomes, since these genomes commonly contain the same 37 non-duplicated genes, see Section 2.1.3. For this reason, the following section defines the concept of permutations.

An *(unsigned linear) permutation* π of size $n \in \mathbb{N}$ is a bijection of the set $[1:n]$ to itself, i. e., $\pi\colon [1:n] \to [1:n]$. Traditionally, an unsigned linear permutation π of size n is denoted by its *two-line* notation $\begin{pmatrix} 1 & 2 & \cdots & n \\ \pi(1) & \pi(2) & \cdots & \pi(n) \end{pmatrix}$. The first line of the two-line notation is always the same, therefore the classical notation is partially adopted in this thesis by representing an unsigned linear permutation π only by its bottom row $(\pi(1)\ \pi(2)\ \ldots\ \pi(n))$, where the i-th element of π with $i \in [1:n]$ is denoted by $\pi(i)$. The size of an unsigned linear permutation may be omitted if the context is clear. The set of all unsigned linear permutations of size n is denoted by $\mathcal{P}_n$. For every $\pi \in \mathcal{P}_n$ there is a unique permutation $\pi^{-1} \in \mathcal{P}_n$ that is called *inverse permutation* (of π) defined by $\pi^{-1}(j) = i$ if and only if $\pi(i) = j$, i. e., $\pi^{-1}(j)$ is the position of element j in π. The permutation $\iota := (1\ 2\ \ldots\ n)$ is called *identity*. The *composition* of two permutations π and σ of size n, denoted by $\pi \circ \sigma$, is defined by applying σ first followed by π which results in the permutation $(\pi(\sigma(1))\ \pi(\sigma(2))\ \ldots\ \pi(\sigma(n)))$, i. e., $(\pi \circ \sigma)(i) := \pi(\sigma(i))$ for all $i \in [1:n]$, see Example 2.2. The operation $\circ$ induces a group structure on the set $\mathcal{P}_n$ by satisfying associativity, i. e., $(\pi \circ \sigma) \circ \lambda = \pi \circ (\sigma \circ \lambda)$, the existence of the *neutral element* ι, i. e., for all $\pi \in \mathcal{P}_n$ it holds that $\pi \circ \iota = \pi$, and the existence of the inverse permutation as *inverse element*, i. e., for every permutation $\pi \in \mathcal{P}_n$ there exists a $\pi^{-1} \in \mathcal{P}_n$ such that $\pi \circ \pi^{-1} = \iota$. Therefore, the pair $(\mathcal{P}_n, \circ)$ is also called *symmetric group*.

As introduced in Section 2.1.1, the DNA of species is double-stranded and thereby facilitates the possibility that genes can be located on different DNA strands. Signed permutations make it easier to take the relative orientation of genes into account. Therefore, signed permutations constitute a biologically relevant structure to model gene orders (Fertin et al., 2009). Less formally, a signed

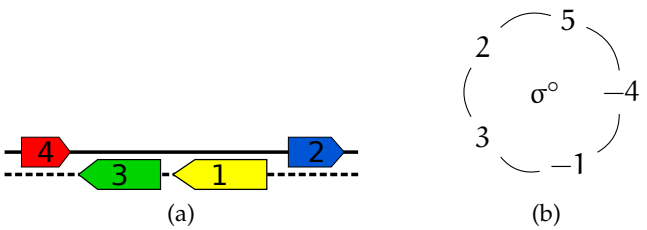


Figure 2.8: (a) Example of a unichromosomal gene order with a single linear chromosome. (b) Illustration of the signed circular permutation $\sigma^\circ = [(-1\ 3\ 2\ 5\ -4)]_\sim$. The circular illustration gives each representatives of σ° when read clockwise (respectively clockwise and counterclockwise) if π° is considered to be directed (respectively undirected), see Example 2.3.

linear permutation is an unsigned linear permutation where each element has an additional sign $+$ or $-$ indicating the strandedness of the gene it represents. Formally, a *(signed linear) permutation* π of size $n \in \mathbb{N}$ is a bijection $\pi\colon [-n\colon n] \setminus \{0\} \to [-n\colon n] \setminus \{0\}$ that satisfies $\pi(-i) = -\pi(i)$ for all $i \in [-n\colon n] \setminus \{0\}$, where $X \setminus Y$ denotes the *set difference*, i. e., the set of all elements that are in set X but not in set Y. Consequently, the two-line notation of a signed linear permutation π of size n is $\begin{pmatrix} -n & \dots & -2 & -1 & 1 & 2 & \dots & n \\ -\pi(n) & \dots & -\pi(2) & -\pi(1) & \pi(1) & \pi(2) & \dots & \pi(n) \end{pmatrix}$. In accordance to unsigned linear permutations, signed linear permutations are represented by $(\pi(1)\ \dots\ \pi(n))$ since the mapping of the negative elements $[-n\colon -1]$ is unambiguously given by $\pi(-i) = -\pi(i)$, see Example 2.2. The set of all signed linear permutations of size n is denoted by $s\mathcal{P}_n$. Obviously, it holds that $\mathcal{P}_n \subset s\mathcal{P}_n$. For the sake of simplicity the $+$ sign of an element of a signed permutation is usually omitted. The composition operation $\circ$ is defined for signed linear permutations analogously to unsigned linear permutations. The *inverse permutation* π^{-1} of a $\pi \in s\mathcal{P}_n$ is defined by $\pi^{-1}(j) = -i$ if and only if $\pi(i) = -j$, see Example 2.2. A group structure is induced on $s\mathcal{P}_n$ by the composition operation, therefore the pair $(s\mathcal{P}_n, \circ)$ is called the *hyperoctahedral group*. For more information about permutation group theory see, e. g., Bóna (2004).

Example 2.2. *Consider the linear gene order of the double-stranded chromosome illustrated in Figure 2.8 (a). This chromosome can be modeled by the signed linear permutation* $\mu = (4\ -3\ -1\ 2)$. *The inverse permutation of* μ *is* $\mu^{-1} = (-3\ 4\ -2\ 1)$, *since* $\mu \circ \mu^{-1} = (\mu(\mu^{-1}(1))\ \dots\ \mu(\mu^{-1}(4))) = (\mu(-3)\ \mu(4)\ \mu(-2)\ \mu(1)) = (-\mu(3)\ \mu(4)\ -\mu(2)\ \mu(1)) = (1\ 2\ 3\ 4)$. *An example for an unsigned linear permutation of size* 5 *is* $\pi = (4\ 2\ 1\ 5\ 3)$. *The inverse permutation of* π *is* $\pi^{-1} = (3\ 2\ 5\ 1\ 4)$, *because* $\pi \circ \pi^{-1} = (\pi(\pi^{-1}(1))\ \dots\ \pi(\pi^{-1}(5))) = (\pi(3)\ \pi(2)\ \pi(5)\ \pi(1)\ \pi(4)) = (1\ 2\ 3\ 4\ 5) = \iota$. *Now consider the two signed linear permutations* $\sigma = (-5\ 4\ -2\ 1\ -3)$ *and* $\sigma' = (-1\ 3\ 2\ 5\ -4)$. *The composition of* σ *and* σ' *is* $\sigma \circ \sigma' = (\sigma(\sigma'(1))\ \dots\ \sigma(\sigma'(5))) = (\sigma(-1)\ \sigma(3)\ \sigma(2)\ \sigma(5)\ \sigma(-4)) = (-\sigma(1)\ \sigma(3)\ \sigma(2)\ \sigma(5)\ -\sigma(4)) = (5\ -2\ 4\ -3\ -1)$.

As pointed out in Section 2.1.3, chromosomes are not linear in general, therefore circular permutations are used to model gene orders

of circular unichromosomal genomes, e. g., mitochondrial genomes. Intuitively, a circular permutation is the set of all permutations that become equivalent when the first and the last element of a (linear) permutation are considered to be adjacent. The following section defines the notion of (directed) circular permutations as it is usually done in mathematical literature, e. g., see Wielandt (1964).

The *shift* operation $\phi\colon s\mathcal{P}_n \to s\mathcal{P}_n$ is defined by $\pi = (\pi(1)\ \pi(2)\ \dots\ \pi(n)) \mapsto (\pi(2)\ \dots\ \pi(n)\ \pi(1))$. The *k-shift* $\phi^k(\pi)$ for a $k \in \mathbb{N}$ is recursively defined by $\phi^k := \phi \circ \phi^{k-1}$ and $\phi^1 := \phi$. Note that $\phi^k(\pi) = \pi$ for all $k \in \mathbb{N}$ that is a multiple of n, i. e., there exists an $m \in \mathbb{N}$ with $km = n$. Two permutations $\pi, \sigma \in s\mathcal{P}_n$ are equivalent or *shifts of each other*, denoted by $\pi \sim \sigma$, if and only if there exists a $k \in [1\colon n]$ such that $\phi^k(\pi) = \sigma$. Hence, $\sim$ induces equivalence classes on $s\mathcal{P}_n$. The equivalence class $\pi^\circ := [\pi]_\sim$ of $\sim$ on $s\mathcal{P}_n$ is called *(signed circular) permutation* of size n and the set of all signed circular permutations of size n is denoted by $s\mathcal{P}_n^\circ$, i. e., $s\mathcal{P}_n^\circ := \{[\pi]_\sim : \pi \in s\mathcal{P}_n\}$. In other words, a signed circular permutation π° is the set of all signed linear permutations that are equivalent with respect to $\sim$, i. e., $\pi^\circ = \{\pi, \phi(\pi), \dots, \phi^{n-1}(\pi)\}$. Usually signed circular permutations are also represented as signed linear permutations by cutting them at some position, these signed linear permutations $\pi \in \pi^\circ$ are called *representatives* of π°. The equivalence relation $\sim$ can be restricted to the set of all unsigned linear permutations and define the subset of all signed circular permutations whose elements are unsigned, i. e., the set $\mathcal{P}_n^\circ := \{[\pi]_\sim : \pi \in \mathcal{P}_n\}$ of all (unsigned circular) permutations of size n. Example 2.3 exemplifies the definitions concerning circular permutations and Figure 2.8 (b) illustrates a signed circular permutation.

Example 2.3. *Consider the signed linear permutation* $\sigma = (-1\ 3\ 2\ 5\ -4)$. *Permutations* σ *and* $(2\ 5\ -4\ -1\ 3)$ *are shifts of each other, since* $\phi^2(\sigma) = (2\ 5\ -4\ -1\ 3)$. *The signed circular permutation* $[\sigma]_\sim$ *is the set of all signed linear permutations that are shifts of* σ, *i. e., the set* $\{(-1\,3\,2\,5\,-4), (3\,2\,5\,-4\,-1), (2\,5\,-4\,-1\,3), (5\,-4\,-1\,3\,2), (-4\,-1\,3\,2\,5)\}$. *An illustration of* $[\sigma]_\sim$ *is shown in Figure 2.8 (b).*

Chromosomes have no prescribed reading direction, e. g., linear chromosomes can be read from left to right (or vice versa) and circular chromosomes can be read clockwise or counterclockwise. If this information is not considered, as it is done in the aforementioned definitions, then permutations are called *directed*. Directed permutations are usually used in the literature to model linear permutations (Fertin et al., 2009). On the other hand, if this information is taken into account, the corresponding permutations are called *undirected* and they are defined as follows: For an unsigned (respectively signed) linear permutation $\pi = (\pi(1) \dots \pi(n))$ the corresponding permutation where the order (and the sign of all elements) is reversed is defined as permutation $\overline{\pi}$ with $\overline{\pi}(i) = \pi(n+1-i)$ (respectively $\overline{\pi}(i) = -\pi(n+1-i)$) for all $i \in [1\colon n]$. Note that $\overline{\pi}$ is uniquely defined for all $\pi \in \mathcal{P}_n$ (respectively $\pi \in s\mathcal{P}_n$). If both reading directions of linear gene orders are taken into account, then unsigned (respectively signed) lin-

ear permutations are considered to be undirected. In other words, this means that the permutations π and $\overline{\pi}$ are equivalent. To define *undirected circular permutations*[2], the equivalence classed of the directed circular permutations are broadened such that an undirected circular permutation ϖ° contains $\{\pi, \ldots, \phi^{n-1}(\pi)\}$ and $\{\overline{\pi}, \ldots, \phi^{n-1}(\overline{\pi})\}$, i. e., $\varpi^\circ := \{\pi, \ldots, \phi^{n-1}(\pi), \overline{\pi}, \ldots, \phi^{n-1}(\overline{\pi})\}$. The set of all undirected signed (unsigned) circular permutations of size n is denoted by $\mathrm{us}\mathcal{P}_n^\circ$ (respectively $\mathrm{u}\mathcal{P}_n^\circ$), i. e., $\mathrm{us}\mathcal{P}_n^\circ := \{[\pi]_\sim \cup [\overline{\pi}]_\sim : \pi \in \mathrm{s}\mathcal{P}_n\}$ (respectively $\mathrm{u}\mathcal{P}_n^\circ := \{[\pi]_\sim \cup [\overline{\pi}]_\sim : \pi \in \mathcal{P}_n\}$).

Throughout this thesis the different kinds of permutations are considered. For each combinatorial problem studied in the following chapters, it is mentioned which type of permutation is used. To simplify the notation, the prefix "unsigned/signed linear/circular directed/undirected" may be omitted if it is apparent from the context.

The study of multichromosomal genomes in terms of permutations can be realized by the innovative idea of capping, e. g., see Hannenhalli and Pevzner (1995). However, the representation of gene orders as permutations is too restrictive to model genomes with unequal gene content and duplicated genes. These genomes are usually represented by sets of adjacencies of the considered genomic marker. The use of adjacencies has the advantage that it is well suited for a theoretical analysis of genome rearrangements that are based on the two operations to cut a chromosome into fragments and (re-)join such fragments into new chromosomes. For an introduction to this notation the reader is referred to, e. g., Fertin et al. (2009) and Hartmann et al. (2018d).

2.2.2 *Gene Clusters and Common Intervals*

Many groups of genes that can be observed in nature are found in close proximity in the genomes of different species (Krebs et al., 2014). Those groups of genes are called *gene clusters*. Motivated by this observation there has been a growing interest in the computational analysis of gene clusters that have been preserved throughout evolution. Several reasons may explain the persistence of such gene groups. Presumably, gene clusters are formed due to functional constraints or evolutionary inertia, e. g., clustered genes encode functionally associated proteins (Galperin and Koonin, 2000; Heber and Stoye, 2001a; Lathe 3rd et al., 2000; Sémon and Duret, 2006; Tamames, 2001), the prevalence of short rearrangements (Sankoff, 2002), and the restrictions of the replication mechanism (Tillier and Collins, 2000). Although there is strong evidence to suggest the preservation of certain gene clusters there is also the possibility that a gene cluster emerged or was not separated by chance.

Considering the information of gene clusters in comparative analyses motivated the exploration of interesting subjects such as the identification of functionally related gene groups and protein function pre-

2 These permutations are often called *genomic circular permutations*, e. g., see Fertin et al. (2009), Meidanis et al. (2000), and Solomon et al. (2003).

diction (Huynen et al., 2000; Mering et al., 2003; Overbeek et al., 1999; Tamames et al., 1997) and the computation of ancestral gene orders from the preserved gene clusters (Bernt et al., 2008a; Ouangraoua et al., 2011a) and similarity measures between genomes (Angibaud et al., 2006; Bergeron and Stoye, 2006). Another example, which is also performed in this thesis, is the computation of scenarios of rearrangements that regard the information on conserved gene clusters (Adam et al., 2007; Angibaud et al., 2007; Bernt et al., 2007). In the latter approach algorithms should enforce scenarios of rearrangements which preserve gene clusters in all intermediate gene orders. Such scenarios and the corresponding rearrangements are called *preserving*[3]. The fundamental idea of preserving rearrangement scenarios is that the universal presence of certain gene clusters in a collection of gene orders requires those clusters to be present in the gene orders of ancestral species. That means that they have not been separated during evolution (Fertin et al., 2009). The computation of gene cluster preserving scenarios of rearrangements is one of the main topics of this work. In particular, Section 5.4 presents the dynamic programming algorithm `CREx2` that computes a shortest scenario of rearrangements of type inversion, transposition, inverse transposition, and tandem duplication random loss between two given gene orders. The scenario obtained by `CREx2` does not break any common interval of both given gene orders. The following section provides a brief overview on relevant computational approaches on gene clusters.

Gene clusters are commonly modeled as sets of genes that fulfill some proximity constraints. A simple and formal model for gene clusters of gene orders are *common intervals* (Heber and Stoye, 2001b) of permutations. Common intervals represent groups of neighboring elements that are not necessarily in the same order or have the same orientation within (signed) permutations. This work exclusively focuses on common intervals to model gene clusters. However, it is worth mentioning that other models that regard positional constraints have already been studied. For example, the spatial structure of chromosome conformation have been considered in Simonaitis and Swenson (2018), Swenson et al. (2016), and Véron et al. (2011). Another example are gene teams which allow for gaps of a certain size between successive elements of a gene cluster, e. g., see Bergeron et al. (2002c), Luc et al. (2003), and Zhang and Leong (2008).

Consider a (possibly signed) linear permutation π of size n, an *interval* I of π is a non-empty subset of unsigned elements of π that forms a consecutive segment in π, i. e., for I there exists a unique pair (i, j) of indices with $1 \leqslant i \leqslant j \leqslant n$ such that $I = \{|\pi(x)| : i \leqslant x \leqslant j\}$. The set of all intervals of permutation π is denoted by $I(\pi)$. For a set of permutations $\Pi \subseteq s\mathcal{P}_n$ a *common interval* is a subset of (unsigned) elements of the permutations within Π that is an interval in each permutation of Π. With $C(\Pi)$ the set of all common intervals of a set of permutations Π is denoted, i. e., $C(\Pi) = \bigcap_{\pi \in \Pi} I(\pi)$. The set $[1 : n]$ of all elements of the permutations within Π and the singleton sets

3 Sometimes the term *perfect* is used instead of *preserving*, e. g., see Sagot and Tannier (2005) and Ouangraoua et al. (2010).

$\{1\}, \ldots, \{n\}$ are called *trivial common intervals*, see Example 2.4. Variations of the definition of common intervals for multichromosomal, circular gene orders, and for incorporating the orientation information of genes have been explored in Heber et al. (2011) and Heber and Stoye (2001a).

Example 2.4. *Consider* $\Pi = \{\pi, \iota\}$ *with* $\pi = (1\ -2\ 4\ 3\ -6\ 5)$. *The common intervals of* Π *are the trivial common intervals (i. e.,* $\{1,2,3,4,5,6\}$, $\{1\}$, $\{2\}$, $\{3\}$, $\{4\}$, $\{5\}$, $\{6\}$*) and* $\{2,3,4,5,6\}$, $\{1,2,3,4\}$, $\{3,4,5,6\}$, $\{2,3,4\}$, $\{1,2\}$, $\{3,4\}$, *and* $\{5,6\}$. *For example, the common interval* $\{3,4,5,6\}$ *is an interval of* π, *since for the unique pair of indices* $(3,6)$ *it holds that* $\{3,4,5,6\} = \{|\pi(x)| : 3 \leqslant x \leqslant 6\}$.

The computational analysis of common intervals of permutations has gained a lot of attention during the last decade. Uno and Yagiura (2000) have demonstrated that the common intervals of two permutations of size n can be computed in $\mathcal{O}(n+\beta)$ time, where $\beta \leqslant \binom{n}{2}$ is the number of all common intervals. Realizing the value of common intervals for gene order comparison, the algorithm proposed by Uno and Yagiura (2000) has been extended to arbitrary-sized sets of permutations by several authors that prove that the common intervals of a set of m permutations of size n can be computed in time $\mathcal{O}(mn+\beta)$ and $\mathcal{O}(n)$ space (Bergeron et al., 2008; Heber and Stoye, 2001a; Heber et al., 2011; Heber and Stoye, 2001b). Those algorithms are based on the fundamental insight that the set of all common intervals of a set of permutations can be obtained by a smaller generating subset of common intervals called *irreducible intervals* (Heber and Stoye, 2001a) and *strong intervals* (Bergeron et al., 2008). While the different algorithms have the same asymptotic runtime behavior, the approach by Bergeron et al. (2008) has the advantage of using only very basic data structures. In the following, both concepts are outlined.

Two common intervals I_1 and I_2 *overlap* if and only if $I_1 \cap I_2 \neq \emptyset$ and neither includes the other, i. e., neither $I_1 \subseteq I_2$ nor $I_2 \subseteq I_1$. A sequence $I_1, \ldots, I_j$ of common intervals of a set of permutations is called a *chain of common intervals (of length* j*)* if for all $i \in [1:j-1]$ the intervals I_i and I_{i+1} overlap. A common interval I is called *irreducible* if there is no chain $I_1, \ldots, I_j$ of common intervals with $j \geqslant 2$ such that $I = \bigcup_{i \in [1:j]} I_i$. The definition of irreducible common intervals is illustrated in Example 2.5. Heber and Stoye (2001a) observed that the set of irreducible intervals, which is smaller than n, generates the set of common intervals which is smaller than $\binom{n}{2}$. In the algorithm they proposed the irreducible common intervals are constructed first in time $\mathcal{O}(mn)$, followed by generating the set of common interval from the irreducible common intervals in time $\mathcal{O}(n+\beta)$.

Example 2.5. *Let* $\Pi = \{\pi_1, \pi_2, \pi_3\}$ *with* $\pi_1 = (1\ 2\ -3\ 4\ -5\ 6\ 7\ 8)$, $\pi_2 = (-8\ 4\ 5\ 6\ 7\ 1\ -2\ 3)$, *and* $\pi_3 = (1\ 2\ 3\ -8\ -7\ -4\ -5\ -6)$. *The common intervals of* Π *are* $\{1,2\}$, $\{2,3\}$, $\{4,5\}$, $\{5,6\}$, $\{1,2,3\}$, $\{4,5,6\}$, $\{4,5,6,7\}$, $\{4,5,6,7,8\}$, *and the trivial common intervals. The common intervals* $\{1,2\}$ *and* $\{2,3\}$ *(respectively* $\{4,5\}$ *and* $\{5,6\}$*) overlap because they do not have an empty intersection and neither is included in the other. Moreover, the sequences* $\{1,2\}, \{2,3\}$ *and* $\{4,5\}, \{5,6\}$ *are chains of common intervals of length*

2. *All common intervals of* Π *are irreducible except for* $\{1,2,3\}$ *(and* $\{4,5,6\}$*) for which there exists the chain* $\{1,2\}$, $\{2,3\}$ *(respectively* $\{4,5\}$, $\{5,6\}$*) such that* $\{1,2,3\} = \{1,2\} \cup \{2,3\}$ *(respectively* $\{4,5,6\} = \{4,5\} \cup \{5,6\}$*).*

A simple and general notion for generators of common intervals, which gave rise to strong common intervals, has been proposed by Bergeron et al. (2008). A common interval $I \in C(\Pi)$ is *strong* if every other common interval $J \in C(\Pi)$ is either disjoint to I, included in I, or includes I, i.e., $I \cap J = \emptyset$, $J \subseteq I$, or $I \subseteq J$. Since every two strong intervals are either disjoint or one includes the other, the strong intervals of Π form a hierarchy which is captured by the strong interval tree – an extension of the PQ tree data structure (Booth and Lueker, 1976; Parida, 2006). The strong interval tree (Bergeron et al., 2008; Bui-Xuan et al., 2005) is an important data structure for efficient preserving rearrangement analysis, since it can encode all $\mathcal{O}(n^2)$ common intervals of a set of permutations with $\mathcal{O}(n)$ nodes (Bérard et al., 2007). Consider a set of permutations $\Pi \subseteq s\mathcal{P}_n$ and a permutation $\lambda \in s\mathcal{P}_n$ that is *consistent* with Π, i.e., $C(\Pi) = C(\Pi \cup \lambda)$. The *strong interval tree (SIT)* of Π and λ, denoted by $T^{\lambda}(\Pi)$, is an ordered and rooted tree where the node set is the set of strong common intervals of Π, the edge set is defined by their minimal inclusion relation, and the child nodes of a node are ordered as the corresponding intervals in λ. The *degree of* a node N, denoted by $\deg(N)$, is the number of its child nodes. Let $\pi \in s\mathcal{P}_n$ be consistent with Π. For every inner node N of a SIT with the child nodes $N_1, \ldots, N_{\deg(N)}$ in this order, the *quotient permutation* of N (with respect to π) is the permutation $\pi_{|N}$ of size $\deg(N)$ that satisfies that $\pi_{|N}(i)$ precedes $\pi_{|N}(j)$ if and only if the interval N_i is to the left of the interval N_j in π for $i \neq j$. A quotient permutation $\pi_{|N}$ is *linear increasing* (*linear decreasing*) if $\pi_{|N} = (1\ 2\ \ldots\ \deg(N))$ (respectively $\pi_{|N} = (\deg(N)\ \ldots\ 2\ 1)$) holds. Permutation $\pi_{|N}$ is *prime* if it is neither linear increasing nor linear decreasing. Node N is *linear* (*prime*) with respect to π if $\pi_{|N}$ is linear increasing or linear decreasing (respectively prime). Observe that the following facts hold true for a strong interval tree $T^{\lambda}(\Pi)$: i) since λ is consistent with Π each node is an interval in λ, ii) the leaves are the single elements $1, \ldots, n$; and iii) the root is the set $\{1, \ldots, n\}$. Given a SIT $T^{\lambda}(\Pi)$, the common intervals of Π can be characterized in terms of the SIT: every union of consecutive children of a linear node and the union of all children of a prime node form a common interval of Π (Bérard et al., 2007; Bernt, 2009). An example illustrating a SIT for a set of permutations is shown in Figure 2.9 and Example 2.6.

Example 2.6. *Consider the set of permutations* Π *defined in Example 2.4, i.e.,* $\Pi = \{(1\ -2\ 4\ 3\ -6\ 5), \iota\}$*. The permutation* $\lambda = (1\ 2\ 3\ 4\ -6\ 5)$ *is consistent with* Π*, since every* $I \in C(\Pi)$ *is an interval in* λ*, hence* $C(\Pi) = C(\Pi \cup \lambda)$ *is implied. The strong common intervals of* Π *are* $\{1,2,3,4,5,6\}$, $\{1,2\}$, $\{3,4\}$, $\{5,6\}$, $\{1\}$, $\{2\}$, $\{3\}$, $\{4\}$, $\{5\}$, *and* $\{6\}$.

One particularly interesting application is to use common intervals for the comparison of gene orders, see for instance Heber and Stoye (2001a). Of growing interest is the use of the strong interval tree data

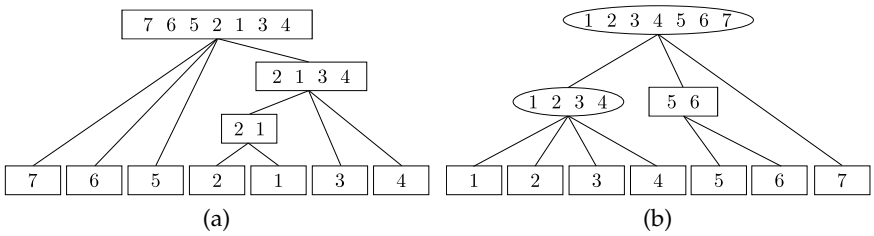


Figure 2.9: (a) SIT $T^{\sigma}(\Sigma)$ for $\Sigma = \{(7\ 6\ 5\ -2\ 1\ -3\ -4) = \sigma, \iota\}$. The set $C(\Sigma)$ is represented in $T^{\sigma}(\Sigma)$ since every union of consecutive children of each node is a common interval of Σ. Node $\{2,1\}$ is linear decreasing since its quotient permutation is $(2\ 1)$ and the quotient permutation $(1\ 2\ 3)$ of node $\{2,1,3,4\}$ implies that it is linear increasing. (b) SIT $T^{\iota}(\Pi)$ for $\Pi = \{\iota, (-7\ 1\ 3\ 2\ 4\ 6\ -5), (6\ -5\ -4\ 3\ -1\ 2\ 7), (-5\ -6\ 7\ 3\ 4\ 2\ -1)\}$. The trivial common intervals, $\{5,6\}$, and $\{1,2,3,4\}$ are the strong common intervals of Π. Prime nodes and linear nodes are represented by ellipses and rectangles, respectively. The root node and node $\{1,2,3,4\}$ are prime and node $\{5,6\}$ is linear increasing.

structure for the computation of scenarios of rearrangements that preserve common intervals, i. e., no genome rearrangement is allowed to break a common interval. Recently, an even more restrictive definition for the preservation of common intervals (of two gene orders) has been proposed by Ouangraoua et al. (2011b). In this work, the authors study the computation of a shortest scenario of rearrangements between two gene orders which ensures that every sub-scenario preserves the common intervals of the respective gene orders as well. Several of such optimization problems have been studied intensively, see Hartmann et al. (2018d) for a recent overview and Section 2.3 for a summary of certain fundamental results of those approaches. In addition to that, Section 5.4 presents an algorithm, called `CREx2`, that computes scenarios of rearrangements that preserve common intervals. `CREx2` considers all types of predominant mitochondrial rearrangements.

Recently, Rusu (2014b) proposed a unifying and efficient algorithmic framework for finding different types of common intervals, e. g., *nested intervals* (Blin et al., 2010; Hoberman and Durand, 2005) or *conserved intervals* (Bergeron and Stoye, 2006), for an arbitrarily-sized set of permutations. In contrast to algorithms that search for intervals directly in the permutations, their approach first extracts helpful information from the permutations followed by progressively computing the suitable intervals. The information extracted from the permutations is called *MinMax-profile* and it focuses on the minimum (and maximum) value that is located between each pair i and $i+1$ of elements, with $i \in [1:n-1]$, in a permutation, see Rusu (2014b) for a formal definition.

Moreover, the value of common intervals has also been recognized in other scientific areas. For example, common intervals are used for the design of crossover operators for genetic algorithms (Uno and Yagiura, 2000) that solve permutation problems like the prominent *traveling salesperson problem* (Held and Karp, 1970). Another interest-

ing aspect is, that finding the common intervals of permutations reverses the *consecutive ones problem* which asks for a permutation of a set X in which the elements of each set of a certain subset of the power set of X occur consecutively (Booth and Lueker, 1976; Heber and Stoye, 2001a). Recently, Pelletier and Rusu (2018) used common intervals to prove that the *Directed MinMax-Betweenness problem* (Rusu, 2016) which asks whether there is a permutation that agrees with a given MinMax-profile, is solvable in polynomial time for a certain class of MinMax-profiles.

Algorithms for the computation of generalized concepts of common intervals have been investigated in several ways. One example is the computation of common intervals (respectively their subclass of *conserved intervals*) of arbitrary strings (Didier, 2003; Rusu, 2014a; Schmidt and Stoye, 2004) (respectively Angibaud et al. (2009) and Bourque et al. (2005)). Another example is the computation of *gene teams* which generalize the common interval model to allow for gaps between genes in the same cluster that are smaller than a given constant (Luc et al., 2003). Gene teams can be computed in polynomial time (Béal et al., 2004; Bergeron et al., 2002c). For a broad overview on gene teams the reader is referred to Hoberman and Durand (2005).

2.2.3 *Mutations and Genome Rearrangements*

Mutations that change the arrangement of genes are modeled as genome rearrangements, i. e., operations that rearrange the elements of permutations. Inversion, transposition, inverse transposition, tandem duplication random loss (TDRL), and (possibly) inverse tandem duplication random loss (iTDRL) are assumed to be major mechanisms for the evolution of metazoan mitochondrial gene orders. The effect of those rearrangements on the gene order is illustrated in Figure 2.5, the five considered mutations are defined as follows: Let π be a (signed linear) permutation. A *rearrangement* ρ for π is an operation that, when applied to π, changes the position (and/or sign) of certain elements of π. The resulting permutation is denoted by $\rho \circ \pi$. The rearrangements that are considered in this thesis are characterized by the intervals of elements that they influence as described in the following (see also Figure 2.10).

- An *inversion* $\rho_I(X)$ for $\pi \in s\mathcal{P}_n$ is an operation that reverses the order and switches the sign of every element within an interval $X \in I(\pi)$. (Recall that $I(\pi)$ is the set of all intervals of π.) Formally, let $X = \{|\pi(i)|, \ldots, |\pi(j)|\}$ be an interval of $\pi = (\pi(1)\ \pi(2) \ldots \pi(n))$ and (i, j) the unique pair of indices with $1 \leqslant i \leqslant j \leqslant n$ such that $X = \{|\pi(x)| : i \leqslant x \leqslant j\}$, then $\rho_I(X) \circ \pi := (\pi(1) \ldots \pi(i-1)\ -\pi(j) \ldots -\pi(i)\ \pi(j+1) \ldots \pi(n))$.

- A *transposition* $\rho_T(X, Y)$ for $\pi \in s\mathcal{P}_n$ is an operation that switches the order of two disjoint intervals X and Y of π that are consecutive, i. e., $X, Y, X \cup Y \in I(\pi)$. Formally, let $X = \{|\pi(i)|, \ldots, |\pi(j)|\}$ and $Y = \{|\pi(j+1)|, \ldots, |\pi(k)|\}$ be two disjoint and consecutive intervals of $\pi = (\pi(1)\ \pi(2) \ldots \pi(n))$ and

(i,j) and $(j+1,k)$ the unique pairs of indices with $1 \leqslant i \leqslant j \leqslant k \leqslant n$, $i \neq k$, such that $X = \{|\pi(x)| : i \leqslant x \leqslant j\}$ and $Y = \{|\pi(x)| : j+1 \leqslant x \leqslant k\}$, then $\rho_T(X,Y) \circ \pi := (\pi(1) \dots \pi(i-1)\ \pi(j+1) \dots \pi(k)\ \pi(i)\ \dots\ \pi(j)\ \pi(k+1) \dots \pi(n))$.

- An *inverse transposition* $\rho_{iT}(X,Y)$ for $\pi \in s\mathcal{P}_n$ is an operation that switches the order of two disjoint and consecutive intervals X and Y of π and, in addition, it reverses the order and switches the sign of every element in X. Formally, let $X = \{|\pi(i)|, \dots, |\pi(j)|\}$ and $Y = \{|\pi(j+1)|, \dots, |\pi(k)|\}$ be two disjoint and consecutive intervals of $\pi = (\pi(1)\ \pi(2) \dots \pi(n))$ and (i,j) and $(j+1,k)$ the unique pairs of indices with $1 \leqslant i \leqslant j \leqslant k \leqslant n$, $i \neq k$, such that $X = \{|\pi(x)| : i \leqslant x \leqslant j\}$ and $Y = \{|\pi(x)| : j+1 \leqslant x \leqslant k\}$, then $\rho_{iT}(X,Y) \circ \pi := (\pi(1) \dots \pi(i-1)\ \pi(j+1) \dots \pi(k)\ -\pi(j) \dots -\pi(i)\ \pi(k+1) \dots \pi(n))$ and $\rho_{iT}(Y,X) \circ \pi := (\pi(1) \dots \pi(i-1)\ -\pi(k) \dots -\pi(j+1)\ \pi(i) \dots \pi(j)\ \pi(k+1) \dots \pi(n))$.

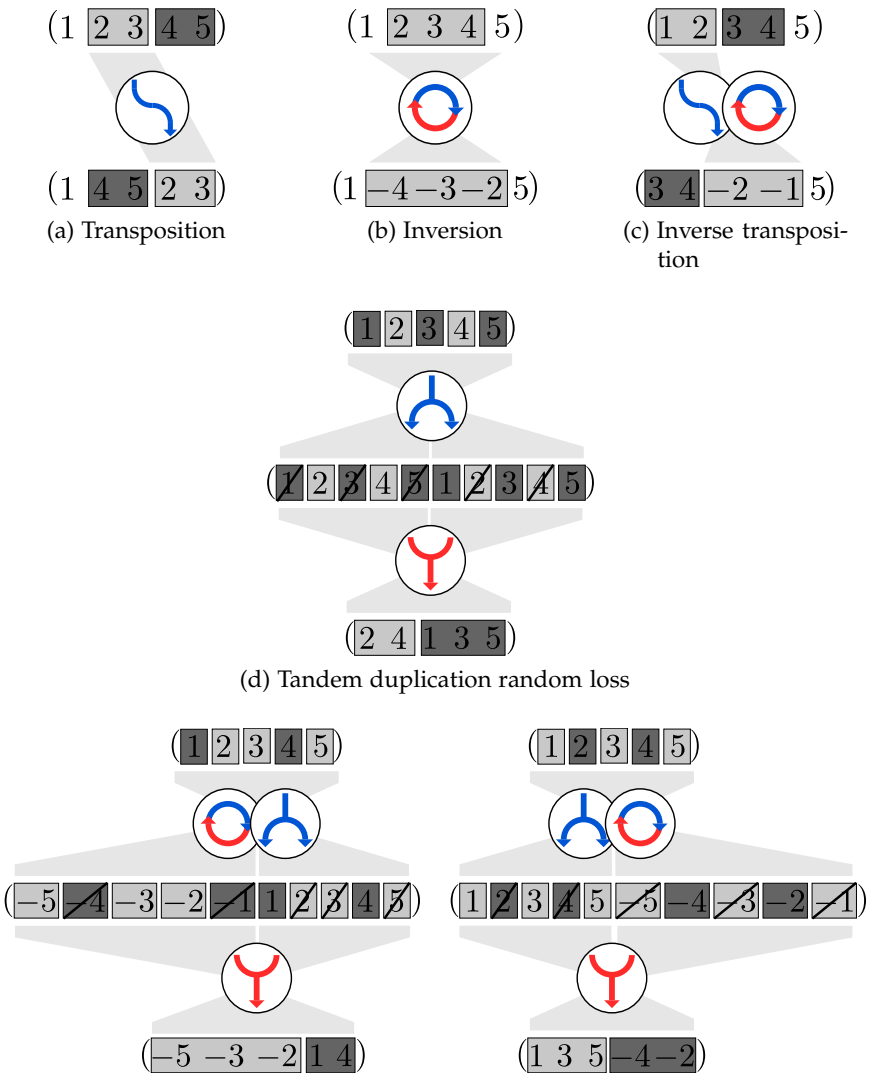
- Let X, Y be a bipartition of an interval of $\pi \in s\mathcal{P}_n$, i.e., $X \cup Y \in I(\pi)$ and $X \cap Y = \emptyset$. A *tandem duplication random loss* $\rho_{TDRL}(X,Y)$ for π is an operation that duplicates the interval $X \cup Y$ of π such that the duplicated interval is placed adjacently to the original one, followed by the loss of every element contained in Y (respectively X) in the left (respectively right) copy of the duplicated interval. The resulting permutation is denoted by $\rho_{TDRL}(X,Y) \circ \pi$.

- Let X, Y be a bipartition of an interval of $\pi \in s\mathcal{P}_n$, i.e., $X \cup Y \in I(\pi)$ and $X \cap Y = \emptyset$. A *left (respectively right) inverse tandem duplication random loss* $\rho_{liTDRL}(\ell,X,Y)$ (respectively $\rho_{riTDRL}(r,X,Y)$) for π is an operation that duplicates the interval $X \cup Y$ of π such that the duplicated interval is placed adjacently to the left (respectively right) of the original one, the order of all genes in the duplicated interval is reversed and the sign of every element is switched, followed by the loss of every element contained in Y in the left copy and every element contained in X in the right copy of the duplicated interval. The resulting permutation is denoted by $\rho_{liTDRL}(\ell,X,Y) \circ \pi$ (respectively $\rho_{riTDRL}(r,X,Y) \circ \pi$).

Note that the definitions of the aforementioned rearrangements can be restricted to unsigned linear permutations by ignoring the sign of each element of a permutation.

2.2.4 *Tracing Evolution and Rearrangement Problems*

The assumption that genome rearrangements occur rarely bridges the gap between the research areas of evolutionary biology and combinatorial optimization by the principle of *maximum parsimony*. Following this principle, the most succinct explanation is always considered to be the best or, in the context of genome rearrangements, a series of rearrangements that *explains* some given gene orders using a minimum number of rearrangements is more likely to be close to reality



(a) Transposition (b) Inversion (c) Inverse transposition

(d) Tandem duplication random loss

(e) Left inverse tandem duplication random loss (f) Right inverse tandem duplication random loss

Figure 2.10: Examples of rearrangements applied to the identity. Illustrated are the rearrangements (a) $\rho_T(\{2,3\},\{4,5\})$; (b) $\rho_I(\{2,3,4\}$; (c) $\rho_{iT}(\{1,2\},\{3,4\})$; (d) $\rho_{TDRL}(\{2,4\},\{1,3,5\})$; (e) $\rho_{liTDRL}(\ell,\{2,3,5\},\{1,4\})$; and (f) $\rho_{riTDRL}(r,\{1,3,5\},\{2,4\})$. Bright and dark gray squares illustrate the sets X and Y of $\rho_I(X)$, $\rho_{liTDRL}(\ell,X,Y)$, $\rho_{riTDRL}(r,X,Y)$, and $\rho_Z(X,Y)$ with $Z \in \{T, iT, TDRL\}$.

than another series of rearrangements with a greater number of rearrangement operations. Subsequently, the research field of genome rearrangements gives rise to a varying spectrum of fascinating and challenging algorithmic and combinatorial problems. This diversity of problems results from various aspects. For example, properties of the miscellaneous gene order representations (e. g., permutations or sets of adjacencies), the relevance of the strandedness of the genes (e. g., signed or unsigned), the importance of the orientation of the gene orders (e. g., directed or undirected), or some biological constraints (e. g., preservation of gene clusters). The set of rearrangements that are considered is of importance for the algorithmic problems and the biological relevance of the results. However, not every combination of these aspects results in a reasonable optimization problem, e. g., the use of TDRLs only on signed permutations makes no sense as TDRLs cannot change the orientation of a gene and only permutations having exactly the same gene orientations could be compared which is equivalent to comparing unsigned permutations.

Throughout this thesis, the set of possible rearrangements that are of interest (for a certain problem) is called a *(rearrangement) model*. Formally, a model $\mathcal{M}$ is a set of rearrangements that are considered for a certain problem. Examples are the models where only a single type of rearrangement operation is considered, e. g., $\mathcal{M}_{\mathrm{I}}$, $\mathcal{M}_{\mathrm{T}}$, $\mathcal{M}_{\mathrm{iT}}$, $\mathcal{M}_{\mathrm{TDRL}}$, and $\mathcal{M}_{\mathrm{iTDRL}}$ are the rearrangement models that include all inversions, transpositions, inverse transpositions, TDRLs and iTDRLs, respectively. Another example with a high biological relevance is the rearrangement model that considers all rearrangements which are frequently encountered in mitochondrial gene orders. Here, the model that is used to represent those rearrangements is denoted by $\mathcal{M}_{\text{4-type}}$ and it contains all predominant mitochondrial rearrangements, i. e., inversions, transpositions, inverse transpositions, and TDRLs. All models that are mentioned in this thesis as well as their definition are summarized in Table 2.1.

With regard to a general definition of the fundamental genome rearrangement problems, formal notions representing gene orders are denoted by $\xi, \xi_1, \ldots, \xi_k$ in the following. In particular, a $\xi \in \Xi$ serves as representation for (unsigned/signed undirected/directed linear/circular) permutations (or adjacencies) and Ξ denotes the set of all those gene order representatives. A considered rearrangement model is denoted by $\mathcal{M}$. Let ξ be a gene order. A *sequence* for ξ of length $t \in \mathbb{N}$ is a series of rearrangements $\rho_1, \ldots, \rho_t \in \mathcal{M}$ such that ρ_i is a rearrangement for $\rho_{i-1} \circ \ldots \circ \rho_1 \circ \xi$ for all $i \in [1:t]$. A sequence of length t is denoted as $(\rho_1, \ldots, \rho_t) \in \mathcal{M}^t$, where $\mathcal{M}^t$ is the t-fold Cartesian product on $\mathcal{M}$, i. e., $\mathcal{M}^t := \{(\rho_1, \ldots, \rho_t) : \rho_1, \ldots, \rho_t \in \mathcal{M}\}$. In order to simplify the notation the application of a sequence $S = (\rho_1, \ldots, \rho_t)$ for ξ (to ξ) is denoted by $S \circ \xi := \rho_t \circ \ldots \circ \rho_1 \circ \xi$. The length of a sequence S is denoted by $|S|$. If a sequence $S = (\rho_1, \ldots, \rho_t)$ for ξ_1 transforms ξ_1 into ξ_2, i. e., $S \circ \xi_1 = \xi_2$, then S is called a *scenario* for ξ_1 and ξ_2. The set of all scenarios for ξ_1 and ξ_2 under the model $\mathcal{M}$ is denoted by $\mathfrak{S}_{\mathcal{M}}(\xi_1, \xi_2)$.

Table 2.1: Summary of rearrangement models that are mentioned in this thesis and their definition. For every model $\mathcal{M}$ there is also a model $\mathcal{M}^P$ that preserves the common intervals of a given set of gene orders.

Model	Definition
$\mathcal{M}_I$	$\{\rho_I(X) : X \subseteq [1:n], X \neq \emptyset\}$
$\mathcal{M}_T$	$\{\rho_T(X,Y) : X,Y \subseteq [1:n], X,Y \neq \emptyset, X \cap Y = \emptyset\}$
$\mathcal{M}_{I;T}$	$\mathcal{M}_I \cup \mathcal{M}_T$
$\mathcal{M}_{iT}$	$\{\rho_{iT}(X,Y) : X,Y \subseteq [1:n], X,Y \neq \emptyset, X \cap Y = \emptyset\}$
$\mathcal{M}_{T;iT}$	$\mathcal{M}_T \cup \mathcal{M}_{iT}$
$\mathcal{M}_{I;T;iT}$	$\mathcal{M}_I \cup \mathcal{M}_T \cup \mathcal{M}_{iT}$
$\mathcal{M}_{TDRL}$	$\{\rho_{TDRL}(X,Y) : X,Y \subseteq [1:n], X \cup Y = [1:n], X \cap Y = \emptyset\}$
$\mathcal{M}_{iTDRL}$	$\{\rho_{liTDRL}(X,Y), \rho_{riTDRL}(X,Y) : X,Y \subseteq [1:n], X \cup Y = [1:n], X \cap Y = \emptyset\}$
$\mathcal{M}_{4\text{-type}}$	$\mathcal{M}_I \cup \mathcal{M}_T \cup \mathcal{M}_{iT} \cup \mathcal{M}_{TDRL}$

The problem of tracing the number of mutations that might have occurred during the evolution of two contemporary species motivates the formulation of the distance problem. This problem asks for the minimum number of rearrangements that are necessary to transform one given gene order into another given gene order as formally defined in the following.

Problem (Distance problem)**.** *The* distance problem *for two gene orders* ξ_1 *and* ξ_2 *under a considered rearrangement model* $\mathcal{M}$ *is to find the minimum number* $d_{\mathcal{M}}(\xi_1, \xi_2)$ *of rearrangements from* $\mathcal{M}$ *that are necessary to transform* ξ_1 *into* ξ_2*, i. e.,* $d_{\mathcal{M}}(\xi_1, \xi_2) := \min_{S \in \mathfrak{S}_{\mathcal{M}}(\xi_1, \xi_2)} |S|$.

The value $d_{\mathcal{M}}(\xi_1, \xi_2)$ is called the *distance* between ξ_1 and ξ_2 under $\mathcal{M}$ or, if the context is clear, the $\mathcal{M}$ *distance*. Observe that a distance between two gene orders under $\mathcal{M}$ provides just a lower bound on the actual number of mutations that might have occurred during the evolution of the considered species. This even holds true if the model $\mathcal{M}$ perfectly reflects the set of mutations that have affected the considered gene orders. The rationale behind this is that successive mutations can neutralize each other, e. g., two successive inversions that affect the same set of genes. Therefore, the information on such mutations is not preserved in the contemporary gene orders and, thus, these mutations cannot be revealed. Of particular interest for a rearrangement model $\mathcal{M}$ is the *diameter* $D_{\mathcal{M}}(\Xi)$ which is the maximum value of the $\mathcal{M}$ distance that can be obtained for a set Ξ of given gene orders, i. e., $D_{\mathcal{M}}(\Xi) := \max_{\xi_1, \xi_2 \in \Xi} d_{\mathcal{M}}(\xi_1, \xi_2)$. The reason is that the diameter allows to measure the variability of the $\mathcal{M}$ distance for a considered set Ξ.

The sorting problem is closely related to the distance problem. Instead of seeking a genomic distance, the sorting problem aims to find one particular scenario that uses a minimum number of considered rearrangements as defined in the following.

Problem (Sorting problem)**.** *The* sorting problem *for two gene orders* ξ_1 *and* ξ_2 *under a considered rearrangement model* $\mathcal{M}$ *is to find a scenario* $S \in \mathfrak{S}_{\mathcal{M}}(\xi_1, \xi_2)$ *such that* $S \in \arg\min_{S' \in \mathfrak{S}_{\mathcal{M}}(\xi_1,\xi_2)} |S'|$.

A scenario S that solves the sorting problem for ξ_1 and ξ_2 under $\mathcal{M}$ is called a *parsimonious scenario* (for ξ_1 and ξ_2 under $\mathcal{M}$).

It is worth pointing out that the sorting problem and the distance problem (for two gene orders under a certain model $\mathcal{M}$) are two distinct problems, whereby the sorting problem often turns out to be harder, see Section 2.3. However, both problems are not completely independent from each other. For example, it is not hard to see that if the sorting problem can be solved efficiently, then the $\mathcal{M}$ distance can be computed efficiently as well. On the other hand, if the $\mathcal{M}$ distance problem can be solved in polynomial time and the considered model contains a polynomial number of rearrangements, then the sorting problem can also be solved in polynomial time by iteratively searching for a rearrangement that decreases the $\mathcal{M}$ distance. However, the sorting problem can sometimes be solved faster than by testing all possible rearrangements, e. g., see the sorting problem for signed linear permutations under $\mathcal{M}_I$ in Section 2.3.1.

If permutations are considered to represent gene orders, these problems are usually studied for a single (directed/undirected signed/unsigned linear/circular) permutation π and the identity ι, which justifies the common term *"Sorting π by $\mathcal{M}$"*. The reason for this is that the number of rearrangements transforming one permutation into another is not dependent on the way genes are numbered, since the genes of both permutations can be renamed without changing the distance between the considered permutations. To see this, consider two permutations π, σ, and a model $\mathcal{M}$. It holds true that $d_{\mathcal{M}}(\iota, \pi^{-1} \circ \sigma) = d_{\mathcal{M}}(\pi^{-1} \circ \pi, \pi^{-1} \circ \sigma) = d_{\mathcal{M}}(\pi, \sigma) = d_{\mathcal{M}}(\sigma^{-1} \circ \pi, \sigma^{-1} \circ \sigma) = d_{\mathcal{M}}(\sigma^{-1} \circ \pi, \iota)$. Hence, the sorting (distance) problem is equivalent to searching for a minimum length scenario (respectively minimum $\mathcal{M}$ distance) that transforms the identity into a given (signed) permutation or vice versa. The underlying concept behind this transformation is the *left-invariance* of permutations, e. g., see Fertin et al. (2009) for more information. For all these reasons, the term *sorting (distance) problem for* Ξ *under* $\mathcal{M}$ is often used in this thesis, where Ξ denotes a considered set of permutations such as $\mathcal{P}_n$, $s\mathcal{P}_n$, $\mathcal{P}_n^\circ$, or $u\mathcal{P}_n^\circ$.

The rearrangement problem that is considered to be most important for reconstructing phylogenetic trees from gene orders is the median problem. It asks for a gene order minimizing the sum of some distance to a set of given gene orders. The aim of solving this problem is to identify a putative ancestral gene order, called a *median*, for the given set of gene orders as defined in the following.

Problem (Median Problem)**.** *The* median problem *for a set of* $k > 2$ *gene orders* $\xi_1, \ldots, \xi_k$ *under a rearrangement model* $\mathcal{M}$ *aims for a gene order* μ *such that* $\mu \in \arg\min_{\xi \in \Xi} \sum_{i \in [1:k]} d_{\mathcal{M}}(\xi, \xi_i)$*, where* Ξ *is the set of all (possible) gene orders.*

Interestingly, the solution of the case $k = 3$ is the foundation of many algorithms that reconstruct phylogenetic trees from gene orders, e. g.,

see Bourque and Pevzner (2002), Moret et al. (2001), and Zhang et al. (2009).

If a model considers more than one type of rearrangement, realistic reconstructions are computed in a parsimony framework by employing a weighting scheme (on the considered model) that reflects the likelihood of the occurrence of different rearrangement events during the evolution of different taxa. Weighted rearrangements have been explored in various aspects, e. g., using weights which respect to the number of genes that are affected by a rearrangement. Different approaches that are relevant for this thesis are summarized in Section 2.3. Another possibility is to use different weights for each type of rearrangement as explained in the following. Let $\mathcal{M}$ be a given model and $\mathfrak{T}_{\mathcal{M}}$ be the set of rearrangement types that are considered, e. g., for $\mathcal{M}_{\text{4-type}}$ it holds that $\mathfrak{T}_{\mathcal{M}} = \{I, T, iT, TDRL\}$. The miscellaneous rearrangements of $\mathcal{M}$ are assigned to *types* in order to classify their similarity. Therefore, the mapping $\mathtt{type}\colon \mathcal{M} \to \mathfrak{P}(\mathfrak{T}_{\mathcal{M}})$, where $\mathfrak{P}(X)$ is the powerset of a set X, assigns every rearrangement to a set of types. Note that the mapping $\mathtt{type}$ is eventually not bijective. The motivation for this definition is that some rearrangements can be of several types, e. g., the effect of a transposition can also be obtained by a TDRL and some iTDRLs have the same effect as inversions or inverse transpositions, see Example 2.7 for more details. The *weight* of a rearrangement in $\mathcal{M}$ is given by a weight function $\omega\colon \mathcal{M} \to \mathbb{R}_{>0}$ such that $\omega(\rho) := \min\{\omega_X : X \in \mathtt{type}(\rho)\}$, where ω_X denotes the given weight of a rearrangement of type X. Given such a weighting scheme ω, the *weight* of a sequence (scenario) $S = (\rho_1, \ldots, \rho_{|S|})$ for ξ_1 (and ξ_2) is defined by the sum of the weights of its rearrangements and, with a slight abuse of the notation, it is denoted by $\omega(S)$, i. e., $\omega(S) := \sum_{i \in [1:|S|]} \omega(\rho_i)$. However, considering a weight function ω generalizes the aforementioned problems leading to the *weighted distance problem*, the *weighted sorting problem*, and the *weighted median problem* by minimizing the weight of a scenario for the given gene orders instead of its length. Certainly, for equally weighted rearrangements, the sorting (respectively median) problem coincides with its weighted version.

Example 2.7. *Consider the TDRL* $\rho_{TDRL}(Y, X)$ *and the transposition* $\rho_T(X, Y)$ *for a permutation* $\pi \in s\mathcal{P}_n$. *If* X *is to the left of* Y *in* π *and* $X \cup Y \in I(\pi)$, *then* $\rho_{TDRL}(Y, X) \circ \pi = \rho_T(X, Y) \circ \pi$, *thus* $\{\mathsf{TDRL}, \mathsf{T}\} \subseteq \mathtt{type}(\rho_{TDRL}(Y, X))$. *As an example consider* $\pi = (1\ -3\ 4\ 2\ -5)$ *with* $X = \{3, 4\}$ *and* $Y = \{2, 5\}$ *for which it holds that* $\rho_{TDRL}(\{2, 5\}, \{3, 4\}) \circ \pi = (1\ 2\ -5\ -3\ 4) = \rho_T(\{3, 4\}, \{2, 5\}) \circ \pi$. *In a similar fashion, it holds that* $\rho_I(X) \circ \pi = \rho_{liTDRL}(X, \emptyset) \circ \pi = \rho_{riTDRL}(\emptyset, X) \circ \pi$ *and* $\rho_{iT}(X, Y) \circ \pi = \rho_{riTDRL}(Y, X) \circ \pi$.

The preservation of common intervals of a set of given gene orders is considered in this work in Section 5.4. Genome rearrangement problems that regard the preservation of common intervals are called *preserving genome rearrangement problems*. The preservation of common intervals is realized by restricting a considered model to contain only those rearrangements that do not break the common

intervals of the given set of gene orders. Formally, let $\mathcal{M}$ be a considered model, $\Pi \subseteq s\mathcal{P}_n$ and $\pi \in s\mathcal{P}_n$ be consistent to Π, then a rearrangement $\rho \in \mathcal{M}$ for π is *preserving* for Π if $\rho \circ \pi$ is consistent with Π, i. e., $C(\pi) = C(\{\rho \circ \pi\} \cup \Pi)$. Analogously, a sequence (scenario) $(\rho_1, \ldots, \rho_t) \in \mathcal{M}^t$ for π (and σ) is *preserving* for Π if for all $i \in [1:t]$ the permutations $\rho_i \circ \ldots \circ \rho_1 \circ \pi$ is consistent with Π. If a model $\mathcal{M}$ preserves a set of common intervals, it is denoted by $\mathcal{M}^p$. For example, the model $\mathcal{M}^p_{4\text{-type}}$ which contains all common mitochondrial rearrangements (note that iTDRLs are excluded) and preserves the common intervals of set of given gene orders, is studied in Section 5.4.

2.3 BACKGROUND ON GENOME REARRANGEMENTS

The following section gives a brief overview of algorithmics on genome rearrangements. In particular, Section 2.3.1, Section 2.3.2, and Section 2.3.3 review approaches considering rearrangement problems under $\mathcal{M}_I$, $\mathcal{M}_T$, and $\mathcal{M}_{iT}$, respectively. A particular focus is being put on combinatorics on the TDRL and the iTDRL rearrangement model in Section 2.3.4 and Section 2.3.5. In addition, approaches are outlined that preserve gene clusters which are represented by common intervals of permutations. The reason is that this work proposes new results for all of these aspects. The assumption that only one type of rearrangement has occurred during the evolution of certain gene arrangements is often unrealistic. For example, while TDRLs may explain the existence of pseudogenes in mitochondrial genomes that cannot be caused by inversions for example, TDRLs are unable to change orientation of genes which can be explained by inversions. Therefore, the development of approaches that consider a combination of different rearrangements is outlined in Section 2.3.6. Another highly active subfield of comparative genomics considers multichromosomal gene arrangements that eventually have an unequal gene content. Section 2.3.7 briefly outlines certain approaches for multichromosomal rearrangements and content modifications.

2.3.1 *Inversion*

The first combinatorially studied problems in the field of gene order analysis consider the inversion rearrangement. In this section, a brief overview on computational and combinatorial findings with respect to inversions is given. For more details the reader is referred to Fertin et al. (2009), Gusfield (1997), Pevzner (2000), and Setubal and Meidanis (1997).

Surprisingly, inversions have already been observed in fruit flies in Sturtevant and Beadle (1936), yet almost half a century has passed before the exploration of computational and combinatorial problems has been taken up in gene order analysis with respect to inversions. Creating the basis for decades of scientific work, Watterson et al. (1982) introduced the distance problem (and the sorting problem) for unsigned circular permutations and they described a greedy al-

gorithm that later turned out to be a 2-approximation (Hannenhalli and Pevzner, 1995). The idea of their algorithm was simple: for every element i from 1 to n of a given permutation π of size $n \in \mathbb{N}$ the algorithm moves element i to the i-th position if $\pi(i) \neq i$. Consequently, the algorithm requires at most $n-1$ steps. A second pioneering work has been presented by Sankoff (1992) which was the first that formalized the sorting problem for unsigned linear permutations. Subsequently, a greedy 2-approximation algorithm and a branch-and-bound approach with exponential runtime for the sorting problem for signed linear permutations has been presented in Kececioglu and Sankoff (1995).

A particularly intuitive notation that captures the similarities and dissimilarities of permutations has been introduced in Sankoff and Blanchette (1997). For two given (possibly signed) linear permutations π and σ a so-called *breakpoint* of π represents a pair of consecutive elements of π that are not consecutive in σ. More formally, let $\pi, \sigma \in \mathcal{P}_n$, a pair $(\pi(i), \pi(i+1))$ with $i \in [1:n-1]$ of consecutive elements of π is called a breakpoint of π (with respect to σ) if and only if neither $\pi(i)$ and $\pi(i+1)$ nor $\pi(i+1)$ and $\pi(i)$ are consecutive in σ. However, if $\pi, \sigma \in s\mathcal{P}_n$, then the definition is slightly modified as follows. A pair $(\pi(i), \pi(i+1))$ with $i \in [1:n-1]$ of consecutive elements of π is called a breakpoint of π (with respect to σ) if and only if neither $\pi(i)$ and $\pi(i+1)$ nor $-\pi(i+1)$ and $-\pi(i)$ are consecutive in σ. The number of breakpoints of a permutation π with respect to ι is denoted by $\mathrm{bp}(\pi)$. It is easy to see that the breakpoints of π (with respect to σ) can be obtained in linear time.

Linear permutations are either unsigned or signed. In the following paragraphs unsigned permutations are considered first. A single inversion can remove at most two breakpoints, hence an intuitive bound for the $\mathcal{M}_I$ distance of a permutation $\pi \in s\mathcal{P}_n$ and the identity $\iota \in s\mathcal{P}_n$ is given by $d_{\mathcal{M}_I}(\pi, \iota) \geqslant \lceil \mathrm{bp}(\pi)/2 \rceil$. Bafna and Pevzner recognized the profound value of the breakpoints in the theory of sorting permutations by inversions. For a given unsigned linear permutation, they represent the information of breakpoints in a graph which is called the breakpoint graph (Bafna and Pevzner, 1996). More precisely, the *breakpoint graph* of $\pi = (\pi(1) \ \ldots \ \pi(n)) \in \mathcal{P}_n$ is an edge-colored graph whose nodes are $\{0, \pi(1), \ldots, \pi(n), n+1\}$. Two nodes $\pi(i)$ and $\pi(j)$ are connected by a black edge if and only if $\pi(i)$ and $\pi(j)$ are adjacent in π and by a gray edge if they are adjacent in the identity permutation. Auxiliary nodes 0 and $n+1$ are appended to ensure that every node of $\{\pi(1), \ldots, \pi(n)\}$ has exactly two incident black and gray edges. Therefore, the breakpoint graph for π can completely be decomposed into edge-disjoint alternating cycles, where an *alternating cycle* is a cycle that has alternatingly colored edges. The number of edge-disjoint alternating cycles of a breakpoint graph of π is denoted by $\mathrm{ac}(\pi)$.

Making use of the breakpoint graph, Bafna and Pevzner (1996) showed that every inversion changes $\mathrm{bp}(\pi) - \mathrm{ac}(\pi)$ by at most one, which implies that $d_{\mathcal{M}_I}(\pi, \iota) \geqslant \mathrm{bp}(\pi) - \mathrm{ac}(\pi)$. They further proved that the diameter $D_{\mathcal{M}_I}(\mathcal{P}_n)$ is $n-1$, which is reached only by the *Gollan permutation* and its inverse, i. e.,

$(1\ 3\ 5\ 7\ \ldots\ n{-}1\ n\ \ldots\ 8\ 6\ 4\ 2)$ and $(1\ n\ 2\ n{-}1\ \ldots\ n/2\ \ n/2{+}1)$ (respectively $(1\ 3\ 5\ 7\ \ldots\ n\ n{-}1\ \ldots\ 8\ 6\ 4\ 2)$ and $(1\ n\ 2\ n{-}1\ \ldots\ \lceil n/2\rceil{+}1\ \lceil n/2\rceil)$) if n is even (respectively odd). In addition, they presented a 1.75-approximation algorithm with $\mathcal{O}(n^2)$ runtime for the sorting problem for unsigned linear permutations under $\mathcal{M}_I$. This method is based on finding a maximal alternating cycle decomposition using properties of the breakpoint graph. However, it turned out that finding such a decomposition is the main obstacle for efficient algorithms that optimally solve the sorting problem for unsigned linear permutations, since it has been shown to be NP-hard through a reduction from the Eularian cycle decomposition (Caprara, 1997b). Moreover, the sorting problem for inversions and unsigned linear permutations is not approximable within $1,0008$ unless $P = NP$ (Berman and Karpinski, 1999). Consequently, the sorting problem for unsigned circular permutations under $\mathcal{M}_I$ is NP-hard as well (Solomon et al., 2003). Those results motivate the numerous algorithmic studies on fast and constant factor approximation algorithms for the sorting problem for unsigned permutations under $\mathcal{M}_I$. Notable examples are a 1.5-approximation (Christie, 1998a), a $(1.4348+\epsilon)$-approximation (Caprara and Rizzi, 2002), a $(1.4193+\epsilon)$-approximation (Lin and Jiang, 2004), and the (up to now) best known approximation guarantee of $11/8$ (Berman et al., 2002). In addition, exact algorithms that exhibit a potentially exponential runtime have been presented: a branch and bound algorithm (Caprara et al., 1999) and two integer linear programming approaches (Caprara et al., 2000; Dias and Souza, 2007). The possibility to solve the distance problem for unsigned linear permutations under $\mathcal{M}_I$ with evolutionary algorithms has been explored in Silveira et al. (2017) (and references therein).

The concept of the breakpoint graph extends naturally to signed linear permutations. A signed permutation $\pi \in s\mathcal{P}_n$ can be transformed into an unsigned permutation $\pi' \in \mathcal{P}_{2n}$ by replacing every element $i \in [-n{:}n] \setminus \{0\}$ by the elements $2i-1$ and $2i$ (respectively $2i$ and $2i-1$) in that order if i is positive (respectively negative). Furthermore, it holds that inversions for π can be replaced by inversions for π' by replacing the set of affected elements as explained above, see Figure 2.11 for an example. Consequently, the sorting problem for signed linear permutations under $\mathcal{M}_I$ can be described as the sorting problem of transformed unsigned linear permutations under $\mathcal{M}_I$ (Bafna and Pevzner, 1996).

A breakthrough in the genome rearrangement analysis has been made with the milestone paper of Hannenhalli and Pevzner (1999) which showed that the sorting problem (and therefore the distance problem) for signed linear permutations under $\mathcal{M}_I$ can be solved in polynomial time. More precisely, they detected certain parameters in configurations of connected components in the breakpoint graph (called hurdle and fortress) of a considered $\pi \in s\mathcal{P}_n$ that separate the bound $bp(\pi) - ac(\pi)$ from the actual distance $d_{\mathcal{M}_I}(\pi, \iota)$. Based on their findings an exact algorithm with $\mathcal{O}(n^4)$ runtime has been presented. It is also worth mentioning that the sorting problem for signed

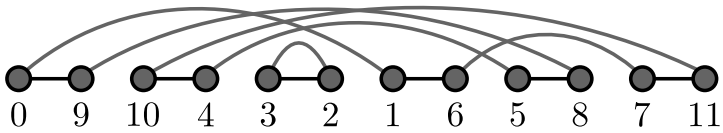


Figure 2.11: Breakpoint graph for the signed linear permutation $(5\ -2\ -1\ -3\ -4)$. Black edges (horizontal lines) represent the adjacent elements of π and gray edges the adjacent elements in the identity permutation ι. An alternating cycle in the breakpoint graph is formed by the nodes 3 and 2.

circular permutations under $\mathcal{M}_I$ is essentially equivalent to the analogous problem for linear permutations (Meidanis et al., 2000). Over the last two decades, the runtime of Hannenhalli and Pevzner's algorithm has been progressively improved by Berman and Hannenhalli (1996), Kaplan et al. (2000) and Kaplan and Verbin (2003), and Tannier and Sagot (2004) and Tannier et al. (2007) to $\mathcal{O}(n^{3/2}\sqrt{\log n})$. The currently best runtime has been presented by Swenson et al. (2010) which presented a simple and fast randomized algorithm that runs in time $\mathcal{O}(n \log n + \alpha n)$, where α is a data-dependent parameter. The authors concluded by extensive experiments on α that almost all permutations can be sorted in $\mathcal{O}(n \log n)$.

For the case that only the $\mathcal{M}_I$ distance for signed permutations is of interest, a linear time algorithm based on the breakpoint graph has been presented (Bader et al., 2001). Recently, lower and upper bounds on the average number of inversions that are needed to sort a signed linear permutation have been presented by Lima and Ayala-Rincon (2018). The diameter $D_{\mathcal{M}_I}(s\mathcal{P}_n)$ is $n+1$ for $n \geqslant 4$ (Christie, 1998b) and it is achieved by the permutation $(n\ n{-}1\ \ldots\ 1)$ (see also Knuth (1997)). The formal aspects for solving the sorting problem under $\mathcal{M}_I$ have been significantly simplified by Bergeron (2001) and Bergeron et al. (2004).

Finding the median of three or more unsigned (or signed) linear permutations under $\mathcal{M}_I$ has been shown to be NP-hard (Caprara, 1997a). However, near-optimal solutions can be obtained within an acceptable amount of time even for large permutations, e. g., see Rajan et al. (2010).

The computational exploration of the preservation of common intervals in gene order analysis with respect to inversions has been initiated by Figeac and Varré. In Figeac and Varré (2004) the sorting problem for $s\mathcal{P}_n$ under $\mathcal{M}_I^p$ has been introduced with the objective to produce more biologically relevant results. (Recall that $\mathcal{M}^p$ denotes the variant of a rearrangement model $\mathcal{M}$ that regards the preservation of common intervals.) The authors proved that the sorting problem for $s\mathcal{P}_n$ under $\mathcal{M}_I^p$ is NP-hard. However, it is fixed-parameter tractable (Bouvel et al., 2011), hence there are polynomial runtime algorithms for many relevant instances (Bérard et al., 2004; Bérard et al., 2007; Bérard et al., 2008), and algorithms with a polynomial average runtime (Bouvel et al., 2011). For example, algorithms have been presented which partition the set of problem instances into three subsets

of instances that can be solved in linear time, subquadratic time, and exponential time (Bérard et al., 2007). In the following, the three algorithms are described more precisely. Therefore, the definition of the strong interval tree (SIT) is recalled, followed by a characterization of the three subsets of problem instances in terms of the SIT. Finally, for every subset of problem instances it is outlined how the sorting problem for $s\mathcal{P}_n$ under $\mathcal{M}_I^p$ is solved by Bérard et al. (2007).

Let π and ι be two signed linear permutations of size n. Recall that the SIT of $\{\pi, \iota\}$ and π is the tree $T^\pi(\{\pi, \iota\})$ that has all strong common intervals of $\{\pi, \iota\}$ as nodes that are connected by an edge with respect to their minimal inclusion relation, and the nodes are ordered according to π. Recall also that for every inner node N of $T^\pi(\{\pi, \iota\})$ there exists a unique permutation (i. e., the quotient permutation $\iota_{|N}$) that reflects the order of the strong common intervals of π with respect to ι. An inner node is linear decreasing, linear increasing, or prime if and only if the corresponding quotient permutation is $(1\ 2\ \ldots\ \deg(N))$, $(\deg(N)\ \ldots\ 2\ 1)$, or neither of those, respectively. (Note that leaf nodes are always considered to be linear.) The set of all possible SITs is categorized into three subsets: A SIT is *unambiguous* if every prime node has a linear parent and *ambiguous* otherwise. SITs without prime nodes are called *definite*. For example, the SIT illustrated in Figure 2.9 (a) (respectively Figure 2.9 (b)) is definite (respectively ambiguous). Bérard et al. (2007) showed that the sorting problem for $s\mathcal{P}_n$ under $\mathcal{M}_I^p$ can be solved in linear time for problem instances with a definite SIT, those with an unambiguous SIT can be solved in subquadratic time, and instances with an ambiguous SIT have an exponential runtime in the worst case.

The key for solving the sorting problem for $\pi \in s\mathcal{P}_n$ and $\iota \in s\mathcal{P}_n$ under $\mathcal{M}_I^p$ is that an inversion $\rho_I(X)$ is preserving for $\{\pi, \iota\}$ (i. e., it does not break a common interval of π and ι) if and only if it is a node or a union of children of a prime node of $T^\pi(\{\pi, \iota\})$ (Bérard et al., 2007). Thus, linear nodes can only be reversed as a whole and the children of prime nodes can be rearranged freely. With respect to the SIT the sorting problem for $\pi \in s\mathcal{P}_n$ and $\iota \in s\mathcal{P}_n$ under $\mathcal{M}_I^p$ is to apply a minimum length sequence S of preserving inversions to π such that all quotient permutations in $T^{S\circ\pi}(\{\pi, \iota\})$ are transformed into the identity permutation, i. e., the nodes in $T^{S\circ\pi}(\{\pi, \iota\})$ become linear increasing. Bérard et al. (2007) proved that if a node N of $T^\pi(\{\pi, \iota\})$ is linear increasing and its parent node is linear decreasing (or vice versa), it holds that the inversion $\rho_I(N)$ is always part of any parsimonious sorting scenario, i. e., the inversions that correspond to nodes with a different *orientation* than their parent node define a parsimonious rearrangement scenario if the considered SIT is definite (Bérard et al., 2007). It is not difficult to see that this procedure can be implemented to run in linear time with respect to the number of nodes. Example 2.8 illustrates the algorithm for definite SITs. Interestingly, since the inversions in a parsimonious preserving scenario for a problem instance with a definite strong interval tree do not overlap, the order of the inversions in the scenario can arbitrary be changed. Hence, the set of all parsimonious scenarios can be obtained as well.

Example 2.8. *Consider the SIT* $T^{\sigma}(\Sigma)$ *with* $\Sigma = \{\sigma = (7\ 6\ 5\ -2\ 1\ -3\ -4), \iota\}$ *illustrated in Figure 2.9 (a). The SIT* $T^{\sigma}(\Sigma)$ *is definite since it does not contain prime nodes. A scenario for* σ *and* ι *is obtained by applying inversions* $\rho_I(\{7\})$, $\rho_I(\{6\})$, $\rho_I(\{5\})$, $\rho_I(\{4\})$, $\rho_I(\{3\})$, $\rho_I(\{1\})$, $\rho_I(\{1,2\})$, $\rho_I(\{1,2,3,4\})$, *and* $\rho_I(\{1,\ldots,7\})$ *to* σ.

For a problem instance with unambiguous SIT $T^{\pi}(\{\pi,\iota\})$ a method is needed to transform the quotient permutation of every prime node into the identity permutation by using a minimum number of inversions. Therefore, every element of a quotient permutations is assigned to the + (respectively -) sign if the corresponding child node is linear increasing (respectively decreasing). Since the children of a prime node N can freely be rearranged, the sorting problem for $\iota_{|N}$ and ι under $\mathcal{M}_I$ has to be solved. Note that this problem is an unconstrained sorting problem. Applying the obtained parsimonious scenarios S to π results in a definite SIT $T^{S\circ\pi}(\{\pi,\iota\})$ which can be processed as explained beforehand. The runtime of the algorithm for unambiguous SITs is dominated by solving the sorting problem for prime nodes which can be done for example by the algorithm presented in Tannier and Sagot (2004) in $O(n^{3/2}\sqrt{\log(n)})$ time.

Ambiguous SITs represent computationally hard problem instances, since the information on linear increasing/decreasing child nodes is unknown. Therefore, an exact solution can be obtained by considering both possible signs for every element of a prime node's quotient permutation and by applying the algorithm for unambiguous SITs. Assume that $\alpha \in \mathbb{N}$ prime nodes (with prime parent node) exist in a given SIT, the described algorithm for ambiguous SITs has an exponential runtime of $O(2^{\alpha}n^{3/2}\sqrt{\log(n)})$ in the worst case.

Algorithms for the median problem for $s\mathcal{P}_n$ under $\mathcal{M}_I^p$ that have a polynomial runtime for many relevant data sets have been proposed in Bernt et al. (2008b). In this work an exact algorithm – called `TCIP` – was presented. `TCIP` uses the bijective relations between consistent permutations and preserving inversions which yields a linear runtime for definite SITs. For ambiguous SITs, different unconstrained versions of the median problem have to be solved for the quotient permutations of prime nodes which significantly increases the runtime of `TCIP`. However, it has been empirically shown that median problems of random gene orders as well as organellar gene orders often have a definite SIT and `TCIP` has a good performance on such instances.

2.3.2 *Transposition*

Since transpositions cannot change the signs of elements, this section only considers unsigned permutations. The distance problem for $\mathcal{P}_n$ under $\mathcal{M}_T$ has been introduced in Bafna and Pevzner (1995). In their work, the authors gave lower and upper bounds for the $\mathcal{M}_T$ distance: $d_{\mathcal{M}_T}(\pi,\iota) \geqslant \lceil bp(\pi)/3 \rceil$ holds true since a transposition can reduce the number of breakpoints by at most 3. In addition, a quadratic runtime approximation algorithm with approximation factor 1.5 was

given. Up to now, only little is known about the diameter for transpositions: it lies between $\lfloor (n+1)/2 \rfloor$ (Bafna and Pevzner, 1995) and $2n/3$ (Eriksson et al., 2001). The algorithm by Bafna and Pevzner was simplified and the runtime was reduced to $\mathcal{O}(n^{3/2}\sqrt{\log n})$ (Hartman, 2003; Hartman and Shamir, 2006). Walter et al. (2000) also studied the distance problem for $\mathcal{P}_n$ under $\mathcal{M}_T$ and presented an approximation algorithm that is based on a very simple structure called *breakpoints diagram*, thereby yielding the approximation guarantee of 2.25. By using a variant of balanced binary trees to encode permutations, the runtime of the algorithm of Hartman and Shamir was improved to $\mathcal{O}(n \log n)$ (Feng and Zhu, 2007). Currently, the best known approximation algorithm runs in $\mathcal{O}(n^2)$ time and guarantees an approximation of 1.375 (Elias and Hartman, 2006). However, a set of heuristics that outperform this approximation factor on small sized permutations has been presented (Dias and Dias, 2013).

A longstanding open problem was to categorize the computational complexity of the distance problem for transpositions. This problem was successfully resolved by proving its NP-hardness (Bulteau et al., 2012). Interestingly, the median problem for $\mathcal{P}_n$ under $\mathcal{M}_T$ is also proven to be NP-complete (Bader, 2011).

2.3.3 *Inverse Transposition*

An inverse transposition can be mimicked by a transposition followed by an inversion of one of the transposed sequences. Therefore, authors usually study the sorting problem and the distance problem for $s\mathcal{P}_n$ ($\mathcal{P}_n$) under $\mathcal{M}_{I;T}$ instead of the $\mathcal{M}_{iT}$, e. g., see Brito et al. (2018) and Gu et al. (1999). Nevertheless, it is worth mentioning that both problems are different. The approaches on the $\mathcal{M}_{I;T}$ model are outlined in Section 2.3.6. If the number of genes affected by an inverse transposition is limited by two and the considered permutations are unsigned, then the sorting problem (and the distance problem) for $\mathcal{P}_n$ under $\mathcal{M}_{iT}$ is equivalent to the same problem under $\mathcal{M}_T$ which can be solved in time $\mathcal{O}(n \log n)$ (Dwork et al., 2001). However, apart from these cases many research questions concerning the algorithmical and computational aspects of inverse transpositions still need further investigation.

2.3.4 *Tandem Duplication Random Loss*

A generalization of the transposition rearrangement model is the TDRL rearrangement model. As well as for transpositions, TDRLs cannot affect the sign of an element of a permutation, therefore only unsigned permutations are considered in this section.

Combinatorial properties of the TDRL rearrangement were initially studied by Chaudhuri et al. (2006) for linear permutations. In their work, the authors presented a weight value of α^ℓ to weight a single TDRL that affects $\ell \in \mathbb{N}$ genes, where $\alpha \geqslant 1$ is a constant. The authors

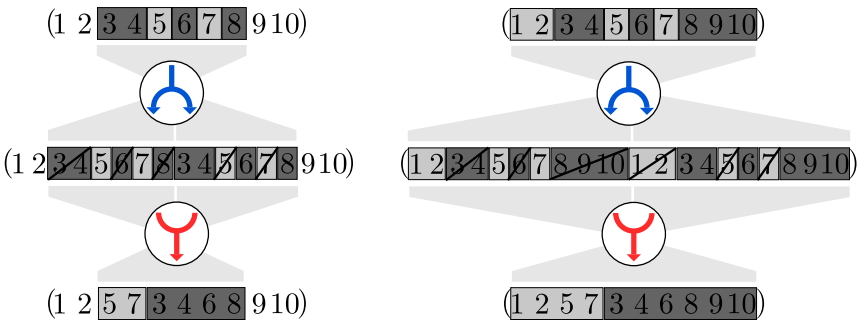
presented polynomial time algorithms that solve the sorting problem (and therefore the distance problem) for the cases $\alpha \geqslant 2$ and $\alpha = 1$.

For $\alpha \geqslant 2$ it was shown by Chaudhuri et al. (2006) that it is sufficient to consider TDRLs duplicating intervals of length two, hence the TDRL distance is exactly the Kentall-Tau distance which can be computed in time $\mathcal{O}(n \log n)$ (Dwork et al., 2001). The Kentall-Tau distance is also known as *bubble sort distance* since it is equivalent to the number of swaps that the bubble sort algorithm uses to sort one permutation into another. The bubble sort distance for a permutation $\pi \in \mathcal{P}_n$ that has to be rearranged to ι is $|\{(i,j) : j < i, \pi(j) > \pi(i)\}|$, e. g., see Knuth (1997).

For $\alpha = 1$ it was shown by Chaudhuri et al. (2006) that it is sufficient to consider TDRLs which copy the whole permutation. Since the length of the duplicated intervals has no influence on the weight of a TDRL, all TDRLs have a weight of 1 in this case. Furthermore, the TDRLs that duplicate the whole permutation implicitly cover TDRLs which duplicate a permutation only partially. This is because a partial permutation duplication TDRL $\rho_{\text{TDRL}}(L, R)$ can be mimicked by a whole permutation duplication TDRL where the elements before (respectively after) the duplicated interval $L \cup R$ are added to the set L (respectively R), see Figure 2.12 for an example. Also for circular permutations it is sufficient to consider TDRLs that duplicate the whole permutation if $\alpha = 1$. This case is covered in Chapter 3.

Chaudhuri et al. (2006) showed that the sorting problem and the distance problem for $\mathcal{P}_n$ under $\mathcal{M}_{\text{TDRL}}$ can be solved in polynomial time. Their approach is based on the insight that a TDRL is equivalent to one step of the classical radix sort algorithm, e. g., see Knuth (1997). The presented algorithm for computing a TDRL scenario for ι and $\pi \in \mathcal{P}_n$ and the $\mathcal{M}_{\text{TDRL}}$ distance is based on the notion of *maximal increasing substrings* of a permutation that is defined in the following for $\pi = (\pi(1) \ \ldots \ \pi(n)) \in \mathcal{P}_n$. A *subsequence* of π is a sequence $\pi(i_1)\pi(i_2)\ldots\pi(i_k)$ with $1 \leqslant i_1 < i_2 < \ldots < i_k \leqslant n$. When all elements in a subsequence S of π appear consecutively, then S is called a *substring* of π. A substring $S = \pi(i)\ldots\pi(k)$ (of π with $1 \leqslant i \leqslant k \leqslant n$) is called *increasing* if either $i = k$ or $\pi(j) < \pi(j+1)$ for all $j \in [i : k-1]$. An increasing substring is called *maximal* if and only if it cannot be extended into a longer increasing substring. The set of all maximal increasing substrings of a permutation π is denoted by $\mathcal{S}(\pi)$. Moreover, $|\mathcal{S}(\pi)|$ denotes the number of maximal increasing substrings of π.

The algorithm that transforms the identity permutation ι into π starts off by computing $\mathcal{S}(\pi)$ and consecutively numbering every maximal increasing substring of π from the left to the right. Then every element of the i-th maximal increasing substring is labeled with the binary representation of i. The identity permutation is rearranged into π by applying the radix sort algorithm to the binary representation of the maximal increasing substring index of an element. More precisely, in the k-th step of the algorithm a whole TDRL $\rho_{\text{TDRL}}(L, R)$ is applied, where an element is in L (respectively R) if the element has a 0 (respectively 1) at the k-th least significant digit in the binary representation of the element's maximal increasing substring index. Since a TDRL is



(a) Partial duplication TDRL (b) Whole duplication TDRL

Figure 2.12: (a) TDRL $\rho_{\text{TDRL}}(\{5,7\},\{3,4,6,8\})$ duplicating the permutation ι partially and (b) the corresponding TDRL $\rho_{\text{TDRL}}(\{1,2,5,7\},\{3,4,6,8,9,10\})$ that duplicates the whole permutation and yields the same result. For $\alpha = 1$ both TDRLs have a weight of 1. Bright and dark gray squares illustrate the sets L and R of a TDRL $\rho_{\text{TDRL}}(L,R)$, respectively.

applied for every digit of the binary representation of $|\mathcal{S}(\pi)|$, it holds that the $\mathcal{M}_{\text{TDRL}}$ distance is given by $d_{\mathcal{M}_{\text{TDRL}}}(\iota,\pi) = \lceil \log_2 |\mathcal{S}(\pi)| \rceil$. It is not hard to see that the $\mathcal{M}_{\text{TDRL}}$ distance can be computed in time $\mathcal{O}(n)$ and that the diameter $D_{\mathcal{M}_{\text{TDRL}}}(\mathcal{P}_n)$ is $\lceil \log_2 n \rceil$, which is achieved, e. g., for the permutation $(n \ \ldots \ 2 \ 1)$. Figure 2.13 (a) and Example 2.9 illustrate the algorithm presented by Chaudhuri et al. (2006).

Example 2.9. *Consider the permutation* $\pi = (5\ 2\ 4\ 3\ 1\ 6)$. *Permutation* π *has four maximal increasing substrings* 5, 2 4, 3, *and* 1 6, *which are indexed with* 1, 2, 3, *and* 4, *respectively. Since* $\lceil \log_2 |\mathcal{S}(\pi)| \rceil = \lceil \log_2 4 \rceil = 2$ *exactly two TDRLs are necessary to obtain* π *from* ι. *Every element of* $[1:6]$ *is assigned to a binary representation of the maximal increasing substring index it belongs to, i. e.,* 5 *is assigned to* 0 0, 2, 4 *are assigned to* 0 1, 3 *is assigned to* 1 0, *and* 1, 6 *are assigned to* 1 1. *The first TDRL* $\rho_{TDRL}(L,R)$ *that has to be applied (to* ι*) is characterized by the second digits of the assigned binary representations: if an element* e *has a* 0 *(respectively* 1*) at the last digit, then* $e \in L$ *(respectively* $e \in R$*). Consequently, it holds that* $3,5 \in L$ *and* $1,2,4,6 \in R$. *Analogously, the second TDRL is characterized by the first digits of the assigned binary representations, which gives* $\rho_{TDRL}(\{2,4,5\},\{1,3,6\})$. *For the inferred TDRLs holds* $\rho_{TDRL}(\{2,4,5\},\{1,3,6\}) \circ \rho_{TDRL}(\{3,5\},\{1,2,4,6\}) \circ \iota = \pi$, *see Figure 2.13 (a) for an illustration.*

While Chaudhuri et al. (2006) were interested in an algorithm that transforms the identity into a given permutation π, Bernt et al. (2011) investigated the opposite. Thereby, the algorithm by Bernt et al. (2011) is based on the notation of chains of permutations. A *chain* of a permutation $\pi \in \mathcal{P}_n$ is a maximal list $(e_1,\ldots,e_k)$ of elements of π such that either $k = 1$ or for all $i \in [1:k-1]$ it holds that $e_{i+1} = e_i + 1$ and $\pi^{-1}(e_i) < \pi^{-1}(e_{i+1})$. Thus, two elements e and $e+1$ of π are in the same chain if and only if element $e+1$ is positioned to the right of e in π, see Example 2.10 and Figure 2.13 (b). Clearly, each element of π belongs to exactly one chain. A permutation $\pi \in \mathcal{P}_n$ has at least 1 and at most n chains. These extremes are given by ι which has one

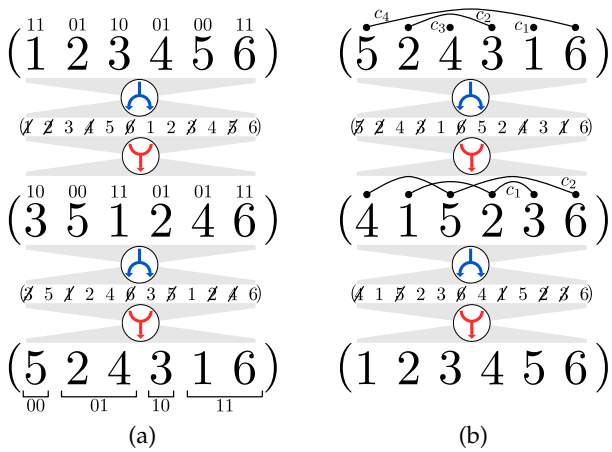


Figure 2.13: TDRL scenario of length 2 for (a) ι and $\pi = (5\ 2\ 4\ 3\ 1\ 6)$ constructed by the algorithm presented by Chaudhuri et al. (2006) and (b) π and ι obtained with the method from Bernt et al. (2011). (a) The maximum increasing substrings of π and the binary representation of their indices are depicted by horizontal lines at the bottom of the subfigure. The binary representation that is assigned to every element is shown on the top of every element. (b) The chains of π (and the intermediate permutation) are depicted above the permutations. Example 2.9 (respectively Example 2.10) clarifies how the corresponding TDRLs are obtained from the binary representations assigned to the elements (respectively the chains of the permutations).

chain, and $(n\ \ldots\ 2\ 1)$ which has n chains. The *number of chains* of a permutation π is denoted by $\vartheta(\pi)$. The chains of a permutation are strictly ordered as follows: $c < c'$ holds for two chains c and c' of π if and only if for all $e \in c$ and for all $e' \in c'$ it holds that $e < e'$.

Interestingly, there exists an one-to-one correspondence between the maximal increasing substrings and the chains of a permutation. In particular, a substring $S = \pi^{-1}(i)\,\pi^{-1}(i+1)\,\ldots\,\pi^{-1}(k)$ of π^{-1} is a maximal increasing substring of π^{-1} if and only if $(i, i+1, \ldots, k)$ is a chain of π. (A formal proof of this correspondence can be found in Bernt (2009).) Consequently, the authors proved that $\vartheta(\pi) = |\mathcal{S}(\pi^{-1})|$ and by the left-invariance of $\mathcal{P}_n$ it follows that $d_{\mathcal{M}_{\text{TDRL}}}(\pi, \iota) = \lceil \log_2 \vartheta(\pi) \rceil = \lceil \log_2 |\mathcal{S}(\pi^{-1})| \rceil = d_{\mathcal{M}_{\text{TDRL}}}(\iota, \pi^{-1})$.

Moreover, Bernt et al. (2011) presented an algorithm that transforms a permutation π into ι and uses a minimum number of TDRLs. Their algorithm is based on the insights that 1) ι is the only permutation of size n that contains only one chain, 2) $\vartheta(\pi)$ cannot be reduced by more than half by one TDRL, 3) a TDRL $\rho_{\text{TDRL}}(L, R)$ can connect successive chains depending on the sets L and R, and 4) a chain c cannot be split by a TDRL $\rho_{\text{TDRL}}(L, R)$ if all elements of c are entirely contained in either L or R. Therefore, a permutation π can be iteratively transformed into ι by the application of a so-called *restricted TDRL* that bisects the number of it chains. In this context, *bisecting* $\vartheta(\pi)$ means that the application of a TDRL ρ to π gives a permutation $\rho \circ \pi$ with $\vartheta(\rho \circ \pi) = \lceil \vartheta(\pi)/2 \rceil$. For a permutation π and total order $c_1 < c_2 < \ldots < c_{\vartheta(\pi)}$ of all its chains, the application of

the TDRL $\rho_{\text{TDRL}}(L, R)$ with $L := \{e \in c_i : \exists n \in \mathbb{N}_0 \text{ with } i = 2n + 1\}$ and $R := \{e \in c_i : \exists n \in \mathbb{N} \text{ with } i = 2n\}$ to π gives always a permutation $\rho_{\text{TDRL}}(L, R) \circ \pi$ that has half as many chains as π. Therefore, $\rho_{\text{TDRL}}(L, R)$ is a restricted TDRL and applying such a TDRL iteratively gives ι after $\lceil \log_2 \vartheta(\pi) \rceil$ steps. Figure 2.13 (b) and Example 2.10 illustrate the algorithm presented by Bernt et al. (2011).

Example 2.10. *Consider the unsigned permutation* $\pi = (5\ 2\ 4\ 3\ 1\ 6)$. *Permutation* π *has the four chains* $c_1 = (1)$, $c_2 = (2, 3)$, $c_3 = (4)$, *and* $c_4 = (5, 6)$. *Since* $\lceil \log_2 \vartheta(\pi) \rceil = \lceil \log_2 4 \rceil = 2$ *exactly two (restricted) TDRLs are necessary to transform* π *into* ι. *The first TDRL* $\rho_{TDRL}(L, R)$ *of such a scenario for* π *and* ι *contains all chains with odd (respectively even) index in* L *(respectively* R*), i. e.,* $\rho_{TDRL}(\{1, 4\}, \{2, 3, 5, 6\})$ *has to be applied to* π. *The resulting permutation* $\rho_{TDRL}(\{1, 4\}, \{2, 3, 5, 6\}) \circ \pi = (4\ 1\ 5\ 2\ 3\ 6)$ *has the two chains* $c_1 = (1, 2, 3)$ *and* $c_2 = (4, 5, 6)$. *The TDRL* $\rho_{TDRL}(\{1, 2, 3\}, \{4, 5, 6\})$ *transforms* $(4\ 1\ 5\ 2\ 3\ 6)$ *into* ι. *An illustration of this example is given in Figure 2.13 (b).*

The median problem for $\mathcal{P}_n$ under $\mathcal{M}_{\text{TDRL}}$ has been studied with respect to the Kendall-Tau distance in Dwork et al. (2001). The authors showed that it is NP-hard for more than three permutations. However, it is still unknown whether the median problem with $\alpha = 1$ under $\mathcal{M}_{\text{TDRL}}$ is solvable in polynomial time.

One exceptional property that characterizes the TDRL model is its asymmetry, i. e., the $\mathcal{M}_{\text{TDRL}}$ distance $d_{\mathcal{M}_{\text{TDRL}}}(\pi, \sigma)$ is in general not equal to $d_{\mathcal{M}_{\text{TDRL}}}(\sigma, \pi)$. Moreover, TDRLs are strongly asymmetric, hence the effect of a TDRL cannot be reversed with a single TDRL in general (Chaudhuri et al., 2006). More precisely, the only exceptions are TDRLs that only swap two adjacent intervals, i. e., they have the same effect as a transposition, and the TDRLs which leave a permutation unchanged. Since the number of these symmetric TDRLs is $\mathcal{O}(n^3)$ and the number of all possible TDRLs is 2^n, the probability of an asymmetric TDRL is exponential in the size n, if the loss of each TDRL in $\mathcal{M}_{\text{TDRL}}$ is considered to be uniformly at random.

In Bernt et al. (2011) the set of all sorting TDRLs, which are TDRLs that reduce the distance of one gene order towards another given gene order, has been investigated.

Different variants of the TDRL model have been studied. In Bouvel and Rossin (2009) one variant was studied in which the maximum number of genes that are allowed to be duplicated by a single TDRL is restricted. In particular, Bouvel and Rossin (2009) investigated the minimum number of TDRLs (each affecting only ℓ genes) that are necessary and sufficient to obtain any linear permutation from any other linear permutation and they proved that all permutations that can be obtained after a given number of such TDRLs define classes of pattern-avoiding permutations (which were further analyzed and enumerated in Bouvel and Pergola (2010)). Therefore, the authors used a modified weight function: the weight of a TDRL of width ℓ is 1 if $\ell \leqslant \beta$ and ∞ otherwise, where $\beta \in \mathbb{N} \cup \infty$ is a constant parameter. A tandem duplication variant has been suggested in Lavrov

et al. (2002) where the subsequent loss is not completely random but dependent on gene orientation or transcript structure.

The inverse operation of a TDRL is a riffle shuffle. In a *riffle shuffle* a deck of cards is split in two stacks followed by a riffle of both parts into a single stack again. The riffle shuffle distance of a permutation which is identical to the TDRL distance for the inverse of the permutation, was studied in, e. g., Bayer and Diaconis (1992). Some interesting results are available for this operation. For example, in their famous paper (see the chapter on card shuffling in Aigner et al. (2010)) Aldous and Diaconis (1986) analyzed the number of riffle shuffles necessary to randomize a deck of cards.

Mitochondrial genomes are commonly circular. Consequently, representing their gene order by circular permutations appears to be more sensible than using linear permutations. However, the TDRL model has not been studied for circular permutations up to now. In Chapter 3 this gap in the literature is closed by showing that the distance problem and the sorting problem for $\mathcal{P}_n^\circ$ under $\mathcal{M}_{\text{TDRL}}$ can be solved in polynomial time.

2.3.5 *Inverse Tandem Duplication Random Loss*

The inverse tandem duplication random loss rearrangement (iTDRL) has been investigated in terms of pattern-avoiding permutations in Baril and Vernay (2010). More precisely, the authors studied the right inverse tandem duplication random loss rearrangement (riTDRL) on unsigned linear permutations. In order to outline their work, the following definitions are crucial.

Let π be an unsigned linear permutation of size n. A *valley* of π is a position $i \in [2\colon n-1]$ with $\pi(i-1) > \pi(i) < \pi(i+1)$. The number of valleys of π is denoted by $\mathrm{val}(\pi)$. Let σ be an unsigned linear permutation of size m with $m \leqslant n$. Permutation σ is a *pattern* of π if there is a subsequence of π which is order-isomorphic to σ, i. e., if there is a subsequence $\pi(i_1)\,\pi(i_2)\,\ldots\,\pi(i_m)$ of π such that $1 \leqslant i_1 < \ldots < i_m \leqslant n$ and for all $\ell \in [1\colon m-1]$ it holds that $\pi(i_\ell) < \pi(i_{\ell+1})$ if and only if $\sigma(\ell) < \sigma(\ell+1)$. Otherwise, if σ is not a pattern of π, then π is said to *avoid* σ. Permutation π is called *alternating* if and only if each element of π is alternately less or greater than its preceding element, i. e., it holds that $\pi(1) > \pi(2) < \pi(3) > \ldots > \pi(n)$ (respectively $\pi(1) > \pi(2) < \pi(3) > \ldots < \pi(n)$) if n is even (respectively odd). See Example 2.11 for an illustration of these definitions.

Example 2.11. *Consider the unsigned linear permutation* $\pi = (4\ 1\ 3\ 2\ 6\ 7\ 5)$. *Since* $4 > 1 < 3$ *and* $3 > 2 < 6$, *the positions* 2 *and* 4 *are valleys of* π *and, thus,* $\mathrm{val}(\pi) = 2$. *The permutation* $\sigma = (1\ 3\ 4\ 2)$ *is a pattern of* π, *since the subsequence* $1\,2\,7\,5$ *of* π *is order-isomorphic to* σ. *Now consider the permutation* $\gamma = (6\ 3\ 7\ 1\ 4\ 2\ 5)$. *Permutation* γ *is alternating, since it holds that* $6 > 3 < 7 > 1 < 4 > 2 < 5$. *It is not difficult to see that there is no subsequence of* π *which is order-isomorphic to* γ. *Therefore, permutation* π *avoids* γ.

Let π be an unsigned linear permutation of size n. Baril and Vernay (2010) have proven that the distance for ι and π under the riTDRL model is $\lceil \log_2(val(\pi) + 1) \rceil + 1$. In addition, it has been shown that the corresponding sorting problem can be solved in time $\mathcal{O}(n \log n)$ by an algorithm that utilizes the reflected binary Gray code (Gray, 1953). Furthermore, they showed that every permutation that can be obtained from ι by the application of $k \in \mathbb{N}$ riTDRLs contains at most $2^{k-1} - 1$ valleys. Interestingly, the set of all these obtainable permutations forms the class of permutations that avoid the alternating permutations of size $2^k + 1$.

However, apart from this exceptional case, the iTDRL rearrangement has not been investigated computationally. In Chapter 4 this circumstance is addressed by showing that the sorting problem and the distance problem for signed linear permutations under $\mathcal{M}_{iTDRL}$ can both be solved in polynomial time.

2.3.6 *Mixed Rearrangement Models*

The assumption that only one type of rearrangement has occurred during the evolution of certain gene orders is most likely unrealistic, e. g., at least inversions, (inverse) transpositions, TDRLs, and potentially iTDRLs have to be considered for mitochondrial gene arrangements (Bernt et al., 2013b). This limitation is addressed by considering a combination of different types of rearrangements for the construction of combined rearrangement scenarios which is briefly outlined in the following.

The sorting problem for $\mathcal{P}_n$ under $\mathcal{M}_{I;T}$ has been investigated with exact algorithms having an exponential runtime (e. g., see Dias and Souza (2007), Sankoff (1992), and Sankoff et al. (1992)), with a machine learning approach (Silva et al., 2017), and approximation algorithms, e. g., see Walter et al. (1998). For signed linear permutations, approximation algorithms have been designed (e. g., Gu et al. (1999)). If the number of genes affected by a rearrangement of $\mathcal{M}_{I;T}$ is never greater than two, then the sorting problem (and therefore the distance problem) can be solved in polynomial time, e. g., see Oliveira et al. (2018b) and references therein. Eriksen (2001) studied a weighted variant of the sorting problem $\mathcal{P}_n$ under $\mathcal{M}_{I;T}$, where a transposition is always weighted twice as much as an inversion. The reason is that transpositions are favored in minimum-weight scenarios compared to inversions if the more powerful transposition is weighted less than twice as much as the less powerful inversion (Jiang and Alekseyev, 2011). Recently, two general heuristics that improve solutions for the sorting problem for $s\mathcal{P}_n$ under $\mathcal{M}_{I;T}$ have been presented (Brito et al., 2018).

The sorting problem for $s\mathcal{P}_n$ under $\mathcal{M}_{T;iT}$ has been investigated in Hartman and Sharan (2005), where a 1.5-approximation algorithm has been presented.

Computational approaches considering combinations of the rearrangement types inversion, transposition, and inverse transposition,

i. e., the model $\mathcal{M}_{I;T;iT}$, have been explored by several authors. The approximation algorithm from Walter et al. (1998) has been extended by Lin and Xue (2001) as to also include inverse transpositions. In their work, the authors presented a 1.75-approximation algorithm for the sorting problem for $s\mathcal{P}_n$ under $\mathcal{M}_{I;T;iT}$. Bader and Ohlebusch (2007) presented a quadratic time 1.5-approximation algorithm in which (inverse) transpositions are weighted more than and less as twice as much as inversions. For the case that (inverse) transpositions are weighted twice as much as inversions, a $(1+\epsilon)$-approximation algorithm for sorting problem for $\mathcal{P}_n$ has been presented (Lou and Zhu, 2010). Weighting those three types of rearrangements with respect to the number of affected elements and its type have been explored for signed linear permutations in Blanchette et al. (1996).

An integer linear programming formulation for the sorting problem for $\mathcal{P}_n$ under a definable model of weighted rearrangements (e. g., the model $\mathcal{M}_{4\text{-type}}$) has been presented in Lancia et al. (2015). The formulation uses an exponential number of variables (which is handled by column generation techniques) to solve the sorting problem for $\mathcal{P}_n$ under arbitrary sets of rearrangements. The presented model uses an upper bound on the length of the sought scenario which can be obtained, e. g., by a heuristic approach. The authors claim that it is easy to incorporate different costs for the different types of rearrangement operations that are used. However, for that case an explicit method to produce a good upper bound on the length of the sought scenario would be necessary.

By detecting patterns in the strong interval tree two algorithms have been described that heuristically compute preserving rearrangement scenarios: one algorithm in Parida (2006) and algorithm `CREx` in Bernt et al. (2007). While the former algorithm considers unsigned permutations, the latter algorithm uses signed permutations, is more general, and considers all preserving rearrangements of types inversion, transposition, inverse transposition, and tandem duplication random loss. In the last decade, the `CREx` heuristic has gotten a lot of attention. In particular, it has been used for the study of mitochondrial gene orders in more than a hundred scientific works. Therefore, the following section outlines the functional principle of the `CREx` heuristic. Consider two signed linear permutations π and ι of size $n \in \mathbb{N}$ and recall that every node of the strong interval tree (SIT) $T^{\pi}(\{\pi, \iota\})$ is either linear increasing, linear decreasing, or prime. The `CREx` heuristic infers an approximated preserving rearrangement scenario for two given signed linear permutations π and ι of size n under $\mathcal{M}^{p}_{4\text{-type}}$. In accordance with the sorting problem for $s\mathcal{P}_n$ under $\mathcal{M}^{p}_{I}$, a minimum length rearrangement scenario S of rearrangements from $\mathcal{M}^{p}_{4\text{-type}}$ is sought that applied to π transforms the SIT $T^{\pi}(\{\pi, \iota\})$ into $T^{S\circ\pi}(\{\pi, \iota\})$ which contains only linear increasing nodes. The basic principle of `CREx` is to detect patterns in the strong interval tree of the given permutations that reflect the four different rearrangement operations. For example, a transposition is indicated by an inner linear node that is linear decreasing while both of its two child nodes are linear increasing. While inversions and inverse transpositions also

lead to identifiable patterns in the strong interval tree, the occurrence of TDRLs is indirectly indicated by the presence of prime nodes. The CREx heuristic uses a stepwise approach: First, the SIT $T^{\pi}(\{\pi, \iota\})$ is constructed; Second, CREx identifies (in that order) transpositions, inverse transpositions, and inversions based on rearrangement patterns in the strong interval tree. In the third step, special care is taken when prime nodes occur in the strong interval tree. The reason for this is that an unconstrained sorting problem for a signed version of the prime node's quotient permutations and the identity permutation under $\mathcal{M}^{p}_{4\text{-type}}$ has to be computed and – up to now – it is unknown whether this problem can be solved efficiently. CREx gives an approximated solution for this problem by using only rearrangements of type TDRL and inversion. The reason is that TDRLs cannot toggle the sign of the involved elements, therefore inversions are used to equalize the signs of the elements of the signed variant of a prime node's quotient permutation. This is achieved by two variants which differ from each other in the order the TDRLs and inversions are applied, i. e., either TDRLs or inversions are applied first. However, the scenarios obtained by CREx are not guaranteed to be optimal and the sequences of rearrangements sorting the prime node's quotient permutation are often of limited biological reliability (Bernt, 2009).

Within this work several improvements for the CREx heuristic are presented: (i) in Section 5.2 an approximation algorithm is presented that significantly improves the prime node procedure of the CREx heuristic in a way that a given scenario of rearrangements which deviates from a parsimonious scenario in at most two rearrangements, and (ii) Chapter 5 also proposes two exact algorithms, namely GeRe-ILP and CREx2 that allow the incorporation or rearrangement weights for all four types of rearrangements. Thereby, both algorithms provide optimal solutions and have – in the worst case – an exponential runtime. However, CREx2 has a linear runtime for large classes of problem instances. In addition, it is shown that the solutions obtained by CREx2 are likely to improve the solutions of the CREx heuristic.

2.3.7 *Multichromosomal Rearrangements and Content Modifications*

Nuclear genomes of eukaryotic species usually contain multichromosomal gene orders. For the comparative analysis of multichromosomal gene orders more types of rearrangements have to be considered. Examples of those rearrangements are *fusions* merging two chromosomes, *fissions* splitting one chromosome into two chromosomes, and *translocations* allowing the exchange of genes between two chromosomes. For all these three types of rearrangements and inversions a polynomial time algorithm for the sorting problem has been presented (Hannenhalli and Pevzner, 1995).

Combined rearrangement models including transpositions are typically hard problems where known exact algorithms have an exponential runtime. Yancopoulos et al. (2005) and Bergeron et al. (2006) suggested the double-cut and join genome rearrangement (DCJ) which

cuts a (potentially multichromosomal) gene order at two different positions and rejoins the resulting fragments. The DCJ model (and other models that are based on cut and join operations such as single-cut and join (Bergeron et al., 2010), single-cut or join (Feijao and Meidanis, 2011), and multi-cut and join (Alekseyev and Pevzner, 2008)) have the benefit that they indirectly include all four major types of unichromosomal rearrangements (and also other rearrangements that are common in multichromosomal gene orders) while simplifying the computational complexity for both the sorting problem and the distance problem. For example, both problems for all mentioned cut and join rearrangements models can be solved in polynomial time. However, a drawback of the cut and join models for unichromosomal gene orders is that the intermediate gene orders (of a rearrangement scenario) are not bound to be unichromosomal, hence intermediate gene orders may consist of several chromosomes that can be both linear and circular. For a broad overview on cut and join rearrangements the reader is referred to Hartmann et al. (2018d). Interestingly, cut and join distances resemble the Hannenhalli-Pevzner-Theory of sorting permutations by inversions since they can be computed by counting cycles in the breakpoint graph (or its line graph), see Bergeron and Stoye, 2013.

Other aspects that arise in computationally comparisons of nuclear gene arrangements and cancer research are content modifications, i. e., large-scale mutations that change the quantity of genes on the chromosomes. Computational models of evolution in this area of comparative genomic mainly focus on comparisons of gene orders with unequal gene content and possibly multiple copies of genes. In accordance to models of genome rearrangements without duplicated genes, different assumptions are made on the structure of given gene orders and types of content modifications. Most problems with duplicated genes turn out to be computationally hard, e. g., the sorting problem for gene orders containing duplicated genes under $\mathcal{M}_I$ (Bryant, 2000; Chen et al., 2005). However, some of them can be solved in polynomial time, e. g., see Feijão et al. (2017). For a broad introduction to this branch of comparative genomics the reader is referred to El-Mabrouk and Sankoff (2012), Fertin et al. (2009), and Zeira and Shamir (2018).

3

TANDEM DUPLICATION RANDOM LOSSES ON CIRCULAR PERMUTATIONS

The tandem duplication random loss operation (TDRL) is a major factor of gene order evolution for mitochondrial genomes (Bernt et al., 2013b). For vertebrate mitogenomes it was even called the "most important rearrangement operation" (San Mauro et al., 2005). But TDRLs do not only occur in vertebrate mitogenome evolution, they have also been detected in the mitogenome evolution of species from other groups within the animal kingdom, e. g., in *Diplopoda* (Lavrov et al., 2002) and *Echinodermata* (Arndt and Smith, 1998) (see also Bernt and Middendorf (2011) for an overview). Formal studies of the TDRL rearrangement treat gene orders as linear unsigned permutations of the genes. However, the sorting problem and the distance problem for unsigned circular permutations under $\mathcal{M}_{\text{TDRL}}$ have not been studied up to now. This is especially important for mitochondrial gene orders, since mitochondrial genomes are commonly circular, see Section 2.1.3. Recall that the weight of a single TDRL rearrangement is α^{ℓ}, where $\ell \in \mathbb{N}$ is the number of duplicated genes and $\alpha \geqslant 1$ is a parameter. As in most other studies on the TDRL distance, the case $\alpha = 1$ is considered in this chapter as well. Therefore, it is sufficient to consider TDRLs which copy the whole genome (Chaudhuri et al., 2006).

In this chapter the combinatorics of the TDRL rearrangement on unsigned circular permutations is studied. In particular, the significant contributions of this chapter are:

- The set of equivalent TDRLs, which are TDRLs that when applied to the same circular permutation yield the same resulting circular permutation, is characterized (Theorem 3.1).
- The distance problem for two directed/undirected unsigned circular permutations of size n under the TDRL rearrangement model can be solved in time $\mathcal{O}(n)$ and the corresponding distance formula is given (Theorem 3.2/Theorem 3.3).
- The sorting problem for two directed/undirected unsigned circular permutations of size n under the TDRL rearrangement model can be solved in time $\mathcal{O}(n \log n)$ (Corollary 3.8/Corollary 3.10).
- The $\mathcal{M}_{\text{TDRL}}$ distance between two circular permutations is generally less by one or equal to the $\mathcal{M}_{\text{TDRL}}$ distance of the corresponding linear representatives. Hence, the $\mathcal{M}_{\text{TDRL}}$ distance might be overestimated if the circularity is neglected. The practical relevance of this result is further investigated (Section 3.2).

The chapter is organized as follows. In Section 3.1 the distance problem (and the sorting problem) for unsigned circular permuta-

tions under the TDRL rearrangement model is solved. The practical consequences of the theoretical results to mitochondrial gene order data is investigated in Section 3.2. In particular, the effect of neglecting the circularity of mitochondrial gene orders for the computation of the $\mathcal{M}_{\text{TDRL}}$ distance is investigated and the tandem duplication non-random loss rearrangement model (Lavrov et al., 2002; Beckenbach, 2011) is evaluated. A conclusion is given in Section 3.3.

3.1 SOLVING THE DISTANCE PROBLEM AND SORTING PROBLEM

This section investigates the distance problem and the sorting problem for unsigned circular permutations under $\mathcal{M}_{\text{TDRL}}$. In Section 2.2.1 it has been outlined that circular permutations are either directed or undirected, i. e., they can be read from either one or both directions. Both possible definitions are covered in this chapter: Section 3.1.1 to Section 3.1.4 consider circular permutations to be directed. What seems to be a restriction proves to be convenient for solving the considered problems for undirected circular permutations as it is done in Section 3.1.5.

3.1.1 *Basic Definitions and Preliminaries*

This section recalls and defines the formal definitions relevant for studying the TDRL rearrangement on unsigned circular permutations. For a detailed definition the reader is referred to Section 2.2.

Throughout the chapter the following notations are used. The set of positive integers is denoted by $\mathbb{N}$ and $\mathbb{N}_0$ refers to the set of non-negative integers, i. e., $\mathbb{N} := \mathbb{N}_0 \setminus \{0\}$. The composition of two functions f and g is denoted by $f \circ g$, i. e., $(f \circ g)(x) := f(g(x))$. The modulo operation that gives the remainder of a Euclidean division is denoted by $\bmod n$, i. e., $x \bmod n = a$ if and only if $\lfloor x/n \rfloor n + a = x$, with $x \in \mathbb{N}_0$, $n \in \mathbb{N}$, and $a \in [0:n-1]$.

Recall that an *unsigned linear permutation* (of *size* $n \in \mathbb{N}$) is a bijection $\pi : [1:n] \to [1:n]$ and it is denoted by $\pi = (\pi(1)\ \pi(2)\ \ldots\ \pi(n))$. A permutation represents a gene order of a linear chromosome and $\pi(i)$ corresponds to the i-th gene. $\mathcal{P}_n$ denotes the set of all unsigned linear permutations of size n. For every permutation π there exists a unique *inverse permutation* π^{-1} defined by $\pi^{-1}(j) = i$ if and only if $\pi(i) = j$. Hence, $\pi^{-1}(j)$ is the position of an element j in π. A TDRL $\rho_{\text{TDRL}}(L, R)$ for a permutation $\pi \in \mathcal{P}_n$ is defined by a bipartition (L, R) with $L \cup R \in I(\pi)$, where $I(\pi)$ is the set of all intervals of π, and L and R contain the elements that are kept in the left and right copy, respectively. Since $\alpha = 1$ it is considered that $L \cup R = [1:n]$, i. e., L and R form a bipartition of all elements of π. The set of all TDRLs is denoted by $\mathcal{M}_{\text{TDRL}}$. Equivalently, by its effects on a permutation π a TDRL $\rho_{\text{TDRL}}(L, R)$ (for π) can be defined as a permutation ρ such that for $\sigma = \rho \circ \pi = \rho(\pi)$ it holds that 1) if $e \in L, f \in R$ then $\sigma^{-1}(e) < \sigma^{-1}(f)$ and 2) if $e, f \in L$ or $e, f \in R$ then $\sigma^{-1}(e) < \sigma^{-1}(f)$ if and only if $\pi^{-1}(e) < \pi^{-1}(f)$. This means that the elements of L are moved in

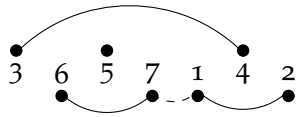


Figure 3.1: The chains (respectively circular chains) of $\pi = (3\ 6\ 5\ 7\ 1\ 4\ 2)$ are depicted as sequences of dots connected by continuous lines (respectively continuous lines and dashed lines).

front of the elements of R, and that the relative order of elements of L (respectively R) is not changed. To keep the notation simple, a TDRL is henceforth denoted by either $\rho(L, R)$ or ρ if the context is clear. For convenience the two notations $\rho \circ \pi$ and $\rho(\pi)$ for the application of a TDRL ρ to a permutation π are used interchangeably. Recall that the definition of a TDRL considers explicitly only whole genome duplications but implicitly also partial genome duplications. This is because a partial genome duplication TDRL $\rho(L, R)$ can be mimicked by a whole genome duplication TDRL where the elements before (respectively after) the duplicated interval are added to the set L (respectively R), see also Section 2.3.4. Recall that the $\mathcal{M}_{\text{TDRL}}$ distance is denoted by $d_{\mathcal{M}_{\text{TDRL}}}(\pi, \iota)$ and it holds that $d_{\mathcal{M}_{\text{TDRL}}}(\pi, \iota) := \min\{t \in \mathbb{N}_0 : \rho_t \circ \ldots \circ \rho_1 \circ \pi = \iota \text{ with } \rho_1, \ldots, \rho_t \in \mathcal{M}_{\text{TDRL}}\} = \lceil \log_2 \vartheta(\pi) \rceil$, where $\vartheta(\pi)$ is the number of chains of π, see Section 2.3.4. For an example see Figure 3.1 and Example 3.1.

Example 3.1. *Consider the permutation* $\pi = (3\ 6\ 5\ 7\ 1\ 4\ 2)$. *The* $\vartheta(\pi) = 4$ *chains are shown in Figure 3.1. The* $\mathcal{M}_{TDRL}$ *distance of* π *is* $d_{\mathcal{M}_{TDRL}}(\pi, \iota) = \lceil \log_2 4 \rceil = 2$. *For TDRLs* $\rho_1(\{1, 2, 5\}, \{3, 4, 6, 7\})$ *and* $\rho_2(\{1, 2, 3, 4\}, \{5, 6, 7\})$ *it holds that* $\rho_1 \circ \pi = (5\ 1\ 2\ 3\ 6\ 7\ 4)$ *and* $\rho_2 \circ \rho_1 \circ \pi = \iota$. *This is the only possibility to transform* π *to* ι *with two TDRLs since there is only one parsimonious scenario for permutations where the number of chains is a power of two (Bernt et al., 2011).*

Two permutations $\pi, \sigma \in \mathcal{P}_n$ are *shifts of each other*, denoted by $\pi \sim \sigma$, if and only if there exists a $k \in [1:n]$ such that $\phi^k(\pi) = \sigma$, where the *k-shift* ϕ^k is recursively defined by $\phi^k := \phi^{k-1} \circ \phi$, $\phi^1 := \phi$, and the *shift operation* $\phi : \mathcal{P}_n \to \mathcal{P}_n$ is defined by $\pi = (\pi(1) \ldots \pi(n)) \mapsto (\pi(2) \ldots \pi(n)\ \pi(1))$. An equivalence class $\pi^\circ := [\pi]_\sim$ of $\sim$ on $\mathcal{P}_n$ is called *(unsigned directed) circular permutation*. The *set of all (unsigned directed) circular permutations* $\mathcal{P}_n^\circ$ is the set of the equivalence classes of $\sim$ on $\mathcal{P}_n$, i. e., $\mathcal{P}_n^\circ := \{[\pi]_\sim : \pi \in \mathcal{P}_n\}$. See Figure 3.2 for an illustration of an unsigned directed circular permutation. Each permutation $\pi \in \pi^\circ$ is called *representative* of π°. The representative π of π° which ends with a certain element $p \in [1:n]$ is denoted by π_p. Hence, $\pi_p \in \pi^\circ$ and $\pi_p^{-1}(p) = n$. In the domain of application, element p typically represents the replication origin and therefore it is called *origin*. Consider a circular permutation $\pi^\circ \in \mathcal{P}_n^\circ$ and two representatives π_p and π_q and let $m = \pi_p^{-1}(q)$ be the position of q in π_p. The two *partition strings* of π° for p with respect to q are the substrings of π_p (and also of π_q) defined as $P = \pi_p(1) \ldots \pi_p(m)$ and $Q = \pi_p(m+1) \ldots \pi_p(n)$ with $\pi_p(m) = q$ and $\pi_p(n) = p$ if $p \neq q$, otherwise, $Q = \emptyset$. See

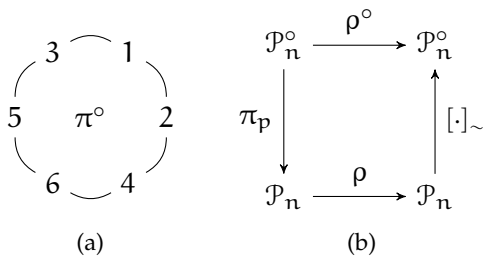


Figure 3.2: (a) Circular representation of $\pi^\circ = [(1\,2\,4\,6\,5\,3)]_\sim$ which gives the representatives in clockwise direction; (b) Diagram of the TDRL$^\circ$ definition as the composition of three operations.

Example 3.2 for an illustration of the definitions related to partition strings.

Example 3.2. *The permutations* $\pi_2 = (3\ 6\ 5\ 7\ 1\ 4\ 2)$ *and* $\pi_6 = (5\ 7\ 1\ 4\ 2\ 3\ 6)$ *are shifts of each other since* $\phi^2(\pi_2) = \pi_6$. *Let* π° *be the circular permutation with* $\pi_2, \pi_6 \in \pi^\circ$. *The partition strings of* π° *for* 2 *with respect to* 6 *are* $P = 3\,6$ *and* $Q = 5\,7\,1\,4\,2$.

A tandem duplication random loss for a circular permutation of size n, also called circular TDRL or *TDRL*$^\circ$ for short, is a mapping $\rho^\circ : \mathcal{P}_n^\circ \to \mathcal{P}_n^\circ$ defined by $\rho^\circ(L, R, p)$, where (L, R) is a bipartition of $[1:n]$ and $p \in [1:n]$ is the origin. The effect of a circular TDRL $\rho^\circ(L, R, p)$ on a circular permutation π° is defined as the result of the application of the TDRL $\rho(L, R)$ on the representative of π° that ends with p, i. e., $\rho^\circ(\pi^\circ) := [\rho(\pi_p)]_\sim$. The diagram depicted in Figure 3.2 (b) illustrates the definition. The definition is motivated by the process of imprecise termination (see Figure 2.6): if the replication of a circular genome misses the endpoint of replication (terminus) this results in a replicate where the part after the terminus (which might comprise the complete genome) is duplicated. Since in mitochondrial genomes the origin and terminus of each strand coincide, the term origin is used for the definition. The set of all circular TDRLs is denoted by $\mathcal{M}_{\text{TDRL}^\circ}$, i. e., $\mathcal{M}_{\text{TDRL}^\circ} := \{\rho^\circ(L, R, p) : L, R \subseteq [1:n], L \cup R = [1:n], L \cap R = \emptyset, p \in [1:n]\}$. Regardless of the chosen origin p the results of the circular duplications, i. e., the intermediate duplicated permutations, are equivalent. This can easily be understood by considering a permutation as a set of adjacencies which is the set $\{(\pi_p(i), \pi_p((i \bmod n) + 1)) : i \in [1:n]\}$ for the representative π_p. Note that the last and the first element are considered to be adjacent. By considering sets of adjacencies, it can be seen that a duplicate of a circular permutation has two copies of each element and two copies of each adjacency of the original permutation. Since the set of adjacencies is the same for each origin p, the corresponding circular duplicates are the same as well. However, the choice of the origin p determines the semantics of *left* and *right*, which is otherwise meaningless in a circular setting. Whereas all circular duplications are circularly equivalent, the corresponding TDRLs applied to the representatives may give different results for each of them.

To see that the formal model of circular TDRLs also covers partial duplication TDRL$^\circ$s (i.e., circular TDRLs where not all elements are

duplicated) consider a partial duplication TDRL° for π°. Hence, a TDRL° $\rho^\circ := \rho^\circ(L, R, p)$ with $L \cup R \in I(\pi_p)$ is considered, where π_p is a representative of π°. By definition it holds that $\rho^\circ \circ \pi^\circ = [\rho(L, R) \circ \pi_p]_\sim$. In Section 2.3.4, it is shown that TDRLs that duplicate the whole permutation implicitly also cover TDRLs that only duplicate a subset of the elements in the partition. Hence, the TDRL $\rho(L, R)$ for π_p can be mimicked by a TDRL $\rho'(L', R')$ that duplicates all elements of π_p. Thus, the equation $\rho^\circ \circ \pi^\circ = [\rho(L, R) \circ \pi_p]_\sim = [\rho'(L', R') \circ \pi_p]_\sim = \rho'^\circ \circ \pi^\circ$, where $\rho'^\circ := \rho'^\circ(L', R', p)$ is a circular TDRL, implies that partial duplication TDRL°s are implicitly covered in the circular model as well.

Analogously to the $\mathcal{M}_{\text{TDRL}}$ distance, the distance between $\pi^\circ \in \mathcal{P}_n^\circ$ and $\sigma^\circ \in \mathcal{P}_n^\circ$ under $\mathcal{M}_{\text{TDRL}^\circ}$ is $d_{\mathcal{M}_{\text{TDRL}^\circ}}(\pi^\circ, \sigma^\circ) := \min\{t \in \mathbb{N}_0 : \exists\, \rho_1^\circ, \ldots, \rho_t^\circ \in \mathcal{M}_{\text{TDRL}^\circ} \text{ such that } \rho_t^\circ \circ \ldots \circ \rho_1^\circ \circ \pi^\circ = \sigma^\circ\}$. For the ease of the notation, the $\mathcal{M}_{\text{TDRL}^\circ}$ *distance* between π° and $\iota^\circ := [\iota]_\sim$ is denoted by $d^\circ(\pi^\circ)$, i. e., $d^\circ(\pi^\circ) := d_{\mathcal{M}_{\text{TDRL}^\circ}}(\pi^\circ, \iota^\circ)$.

Example 3.3. *Consider the circular permutation π° that has the representatives from Example 3.2 and the TDRL° $\rho^\circ(\{1,2,5\},\{3,4,6,7\},6)$. The result of the application of ρ° to π° is the equivalence class of the result of the application of $\rho(\{1,2,5\},\{3,4,6,7\})$ to $\pi_6 = (5\ 7\ 1\ 4\ 2\ 3\ 6)$. Thus, $\rho^\circ(\pi^\circ) = [\rho(\pi_6)]_\sim = [(5\ 1\ 2\ 7\ 4\ 3\ 6)]_\sim$.*

Recall that a *chain* of a permutation π of size n is a list of maximal cardinality $(e_1, \ldots, e_k)$ of elements of π such that either $k = 1$ or for all $i \in [1 : k-1]$ it holds that $e_{i+1} = e_i + 1$ and $\pi^{-1}(e_i) < \pi^{-1}(e_{i+1})$. The following notion of circular chains is useful for studying the combinatorics of circular TDRLs. A *circular chain of a linear permutation* $\pi \in \mathcal{P}_n$ is a list of maximal cardinality of elements $(e_1, \ldots, e_k)$ with either $k = 1$, or $e_{i+1} = (e_i \bmod n) + 1$ and $\pi^{-1}(e_i) < \pi^{-1}(e_{i+1})$ for all $i \in [1 : k-1]$. The elements of each chain and of each circular chain form an interval in ι and they are in order from left to right in π. The only difference to the linear case is that n and 1 are also considered as adjacent in a circular context. The number of chains (respectively circular chains) of a permutation π is denoted by $\vartheta(\pi)$ (respectively $\vartheta^\circ(\pi)$). Figure 3.1 shows an example of the (circular) chains of a permutation.

3.1.2 *Properties of Circular Chains*

In this section, properties of circular chains are identified that are central to analyze the $\mathcal{M}_{\text{TDRL}^\circ}$ distance of circular permutations.

Since a circular permutation is defined as an equivalence class of linear permutations that are shifts of each other, it is of interest to understand the influence of shifts on the number of chains.

Proposition 3.1. *For $\pi \in \mathcal{P}_n$ with $n > 1$ it holds that.*

$$\vartheta(\phi(\pi)) = \begin{cases} \vartheta(\pi) + 1 & \text{if } \pi(1) = 1, \\ \vartheta(\pi) & \text{if } \pi(1) \notin \{1, n\}, \\ \vartheta(\pi) - 1 & \text{if } \pi(1) = n. \end{cases}$$

Proof. Let $\pi \in \mathcal{P}_n$ and $n > 1$. If $\pi(1) = 1$ then there exists a chain $c = (1, 2, \ldots)$. The shift moves 1 to the last position. Thereby it splits c into the two chains $(2, \ldots)$ and (1). Hence, the number of chains increases by one. For the case $\pi(1) \notin \{1, n\}$ there are chains $c = (\pi(1), \pi(1) + 1, \ldots)$ and $d = (\ldots, \pi(1) - 1)$. Then ϕ moves $\pi(1)$ from chain c to chain d. Since c still exists after $\pi(1)$ has left the chain, the number of chains does not change. If $\pi(1) = n$ there are chains $c = (n)$ and $d = (\ldots, n - 1)$. Shifting n to the last position connects c and d into the chain $(\ldots, n - 1, n)$. Thus, the number of chains is decreased by one. □

Proposition 3.1 shows that only shifts which move element 1 (respectively n) to the end of the linear permutation increase (respectively decrease) the number of chains by one, whereas all other shifts do not affect the number of chains. The reason for this is that the elements n and 1 are not considered to be adjacent in chains. Since they are adjacent in circular chains, the same property with circular chains does not hold. The actual difference between the number of chains and the number of circular chains is given by the following proposition.

Proposition 3.2. *For a permutation $\pi \in \mathcal{P}_n$ it holds that*

$$\vartheta^\circ(\pi) = \begin{cases} \vartheta(\pi) & \text{if } \pi^{-1}(1) \leqslant \pi^{-1}(n), \\ \vartheta(\pi) - 1 & \text{if } \pi^{-1}(n) < \pi^{-1}(1). \end{cases}$$

Proof. Let $\pi \in \mathcal{P}_n$ have chains $c_1, \ldots, c_{\vartheta(\pi)}$. From the definition of circular chains it follows that the elements e and $(e \bmod n) + 1$ are in the same circular chain if and only if either (i) $1 \leqslant e < n$ and they are contained in the same chain, or (ii) $e = n$ and $\pi^{-1}(n) < \pi^{-1}(1)$. Thus, if $\pi^{-1}(1) \leqslant \pi^{-1}(n)$ then $\vartheta(\pi) = \vartheta^\circ(\pi)$. Otherwise, for $\pi^{-1}(n) < \pi^{-1}(1)$ the two different chains c_i and c_j, with $1 \in c_i$, $n \in c_j$, $i, j \in [1 : \vartheta(\pi)]$, and $i \neq j$, form a single circular chain and therefore $\vartheta^\circ(\pi) = \vartheta(\pi) - 1$. □

For a linear permutation $\pi \in \mathcal{P}_n$ Proposition 3.2 states that the number of circular chains is smaller by one than the number of chains if n is to the left of 1 in π. Together with Proposition 3.1 this leads to the following result.

Proposition 3.3. *For $\pi \in \mathcal{P}_n$ it holds that $\vartheta^\circ(\pi) = \min_{\gamma \in \pi^\circ} \vartheta(\gamma)$.*

Proof. Two cases for $\pi \in \mathcal{P}_n$ are distinguished (as in Proposition 3.2).

i) For $\pi^{-1}(1) \leqslant \pi^{-1}(n)$ Proposition 3.2 shows $\rho^\circ(\pi) = \rho(\pi)$. By Proposition 3.1 for the iterative application of shifts it holds that $\vartheta(\phi^k(\pi))$ is equal to $\vartheta(\pi)$ for $0 \leqslant k < \pi^{-1}(1)$ and $\pi^{-1}(n) \leqslant k \leqslant n$, which means this holds as long as 1 is to the left of n. Otherwise, for $\pi^{-1}(1) \leqslant k < \pi^{-1}(n)$ the number of chains is increased by one, i. e., equation $\vartheta(\phi^k(\pi)) = \vartheta(\pi) + 1$ holds. Hence $\vartheta^\circ(\pi) = \min_{\gamma \in \pi^\circ} \vartheta(\gamma)$.

ii) For $\pi^{-1}(n) < \pi^{-1}(1)$ Proposition 3.2 shows $\vartheta^\circ(\pi) = \vartheta(\pi) - 1$. By Proposition 3.1 for the iterative application of shifts it holds

that $\vartheta(\phi^k(\pi)) = \vartheta(\pi)$ for $0 \leqslant k < \pi^{-1}(n)$ and $\pi^{-1}(1) \leqslant k \leqslant n$, which means this holds as long as n is to the left of 1. Otherwise, for $\pi^{-1}(n) \leqslant k < \pi^{-1}(1)$ equation $\vartheta(\phi^k(\pi)) = \vartheta(\pi) - 1$ holds. Thus, for this case it follows $\vartheta^\circ(\pi) = \min_{\gamma \in \pi^\circ} \vartheta(\gamma)$. □

The following corollary of Proposition 3.3 shows that the number of circular chains is the same for all permutations that are shifts of each other.

Corollary 3.1. *For permutations $\pi, \sigma \in \mathcal{P}_n$ that are shifts of each other, i.e., $\pi \sim \sigma$, it holds that $\vartheta^\circ(\sigma) = \vartheta^\circ(\pi)$.*

Proof. If $\pi \sim \sigma$ then $[\pi]_\sim = [\sigma]_\sim$. By Proposition 3.3 it follows that $\vartheta^\circ(\pi) = \min_{\gamma \in [\pi]_\sim} \vartheta(\gamma) = \min_{\gamma' \in [\sigma]_\sim} \vartheta(\gamma') = \vartheta^\circ(\sigma)$. □

Remark 3.1. *Corollary 3.1 implies that the number of circular chains of a directed circular permutation can be defined by $\vartheta^\circ(\pi^\circ) := \vartheta^\circ(\pi)$, where π is any representative of π°.*

The following example illustrates the results of Proposition 3.2 and Proposition 3.3.

Example 3.4. *For the circular permutation $\pi^\circ = [(2\ 4\ 6\ 5\ 3\ 1)]_\sim \in \mathcal{P}_6^\circ$ the representative $\pi_5 = (3\ 1\ 2\ 4\ 6\ 5)$ has element 1 at position $\pi_5^{-1}(1) = 2$. The representatives $\phi^0(\pi_5)$, $\phi^1(\pi_5)$, $\phi^5(\pi_5)$ have three chains and the representatives $\phi^2(\pi_5)$, $\phi^3(\pi_5)$, $\phi^4(\pi_5)$ have four chains, i.e., $\vartheta(\phi^0(\pi_5)) = \vartheta(\phi^1(\pi_5)) = \vartheta(\phi^5(\pi_5)) = 3$ and $\vartheta(\phi^2(\pi_5)) = \vartheta(\phi^3(\pi_5)) = \vartheta(\phi^4(\pi_5)) = 4$. Hence, $\vartheta^\circ(\pi^\circ) = 3$.*

The next corollary shows that, in order to compute the number of circular chains of a permutation π, it is sufficient to compute the number of chains for the representative which has element 1 at the first position. Alternatively, it is possible to consider the permutation that has element n at the last position.

Corollary 3.2. *Let $[\pi]_\sim = \pi^\circ \in \mathcal{P}_n^\circ$. For $\sigma = \pi_n$ and $\gamma = \phi^{n-1}(\pi_1)$ which are the permutations ending with n and starting with 1, respectively, it holds that: $\vartheta^\circ(\pi^\circ) = \vartheta(\sigma) = \vartheta(\gamma)$.*

Proof. Since $\sigma^{-1}(1) \leqslant \sigma^{-1}(n)$ and $\gamma^{-1}(1) \leqslant \gamma^{-1}(n)$ the corollary follows from Proposition 3.2. □

3.1.3 *Properties of TDRLs on Circular Permutations*

In this section, properties of the TDRL° operation are presented that are useful for the computation of the $\mathcal{M}_{\text{TDRL}^\circ}$ distance. The following theorem characterizes *equivalent* TDRL°s, which are TDRL°s that result in the same circular permutation when applied to the same circular permutation. Example 3.5 illustrates two such equivalent circular TDRLs.

Example 3.5. *Consider the circular permutation $\pi^\circ = [(2\,3\,4\,1\,5)]_\sim$ and the TDRL°s $\rho_1^\circ(\{3\},\{1,2,4,5\},1)$ and $\rho_2^\circ(\{2,4,5\},\{1,3\},4)$. The TDRL°s ρ_1° and ρ_2° are equivalent, since $\rho_1^\circ \circ \pi^\circ = [(3\ 5\ 2\ 4\ 1)]_\sim = [(5\ 2\ 4\ 1\ 3)]_\sim = \rho_2^\circ \circ \pi^\circ$. The remaining equivalent TDRLs of ρ_1° (and ρ_2°) are listed in Table 3.1.*

Let X be a set of elements and Y be a string. The *restriction* of Y to X, denoted by Y_X, is the string Y' obtained from Y by removing all elements which are not contained in X. The *concatenation* of two strings Y_1 and Y_2 is denoted by $Y_1\,Y_2$. A string Y_1 is called *prefix* (respectively *suffix*) of a string Y_2 if there is a string Y such that $Y_1\,Y = Y_2$ (respectively $Y\,Y_1 = Y_2$). A prefix (respectively suffix) with $Y \neq \emptyset$ is called *strict*. The set of elements of a string Y is denoted by $\mathcal{E}(Y)$.

Theorem 3.1. *Given $\pi^\circ \in \mathcal{P}^\circ_n$ with $n \in \mathbb{N}$, TDRL° $\rho^\circ_1(L_1, R_1, p_1)$, and TDRL° $\rho^\circ_2(L_2, R_2, p_2)$, let P and Q be the partition strings of π° for p_1 with respect to p_2. Then $\rho^\circ_1(\pi^\circ) = \rho^\circ_2(\pi^\circ)$ if and only if*

i) $\rho^\circ_1(\pi^\circ) = \rho^\circ_2(\pi^\circ) = \pi^\circ$ *or*

ii) *a)* $P_{L_1} = P_{L_2}$, $Q_{L_1} = Q_{R_2}$, $P_{R_1} = P_{R_2}$, *and* $Q_{R_1} = Q_{L_2}$ *or*

b) $P_{L_1} = P_{R_2}$, $Q_{L_1} = Q_{L_2}$, $P_{R_1} = P_{L_2}$, *and* $Q_{R_1} = Q_{R_2}$.

Proof. Let $\sigma^1 = \rho_1(\pi_{p_1}) = P_{L_1}\,Q_{L_1}\,P_{R_1}\,Q_{R_1}$ and $\sigma^2 = \rho_2(\pi_{p_2}) = Q_{L_2}\,P_{L_2}\,Q_{R_2}\,P_{R_2}$, where $\rho_1 = \rho_1(L_1, R_1)$, $\rho_2 = \rho_2(L_2, R_2)$, and $\pi_{p_1}, \pi_{p_2} \in \pi^\circ$. See Figure 3.3 for an illustration of this notation.

$\Leftarrow$) If (i) holds true then $\rho^\circ_1(\pi^\circ) = \rho^\circ_2(\pi^\circ)$ follows immediately. If (ii.a) holds true then

$$\begin{aligned}\rho^\circ_1(\pi^\circ) = [\sigma^1]_\sim = [P_{L_1}\,Q_{L_1}\,P_{R_1}\,Q_{R_1}]_\sim &\overset{(ii.a)}{=} [P_{L_2}\,Q_{R_2}\,P_{R_2}\,Q_{L_2}]_\sim \\ = [Q_{L_2}\,P_{L_2}\,Q_{R_2}\,P_{R_2}]_\sim = [\sigma^2]_\sim &= \rho^\circ_2(\pi^\circ).\end{aligned}$$

Similarly, if (ii.b) holds true then

$$\begin{aligned}\rho^\circ_1(\pi^\circ) = [\sigma^1]_\sim = [P_{L_1}\,Q_{L_1}\,P_{R_1}\,Q_{R_1}]_\sim &\overset{(ii.b)}{=} [P_{R_2}\,Q_{L_2}\,P_{L_2}\,Q_{R_2}]_\sim \\ = [Q_{L_2}\,P_{L_2}\,Q_{R_2}\,P_{R_2}]_\sim = [\sigma^2]_\sim &= \rho^\circ_2(\pi^\circ).\end{aligned}$$

$\Rightarrow$) Assume $[\sigma^1]_\sim = [P_{L_1}\,Q_{L_1}\,P_{R_1}\,Q_{R_1}]_\sim = [Q_{L_2}\,P_{L_2}\,Q_{R_2}\,P_{R_2}]_\sim = [\sigma^2]_\sim$. In the following, (i), (ii.a), and (ii.b) are proven for the cases in which exactly none, one, two, three, or four (denoted by case 1-5, respectively) strings of $\{P_{L_1}, Q_{L_1}, P_{R_1}, Q_{R_1}\}$ are empty.

1) In this Case (ii) follows directly since any two elements that are adjacent in σ^2 and belong to either Q_{L_2} or Q_{R_2} (respectively either P_{L_2} or P_{R_2}) necessarily both belong to either Q_{L_1} or Q_{R_1} (respectively either P_{L_1} or P_{R_1}). Otherwise, there would be elements from P (respectively Q) between these two elements in σ^1. Thus, $\sigma^1 \neq \sigma^2$, or one of the sets P_{L_1} or P_{R_1} (respectively Q_{L_1} or Q_{R_1}) needs to be empty.

2) First assume that $Q_{L_1} = \emptyset$. Then $Q_{R_1} = Q$ and it holds that $\sigma^1 = P_{L_1}\,P_{R_1}\,Q$. Since $[\sigma^1]_\sim = [\sigma^2]_\sim$ it is necessary that at least one string of $\{Q_{L_2}, P_{L_2}, Q_{R_2}, P_{R_2}\}$ is empty.

   a) If $P_{L_2} = \emptyset$ then $\sigma^2 = Q_{L_2}\,Q_{R_2}\,P$. Hence, $P_{L_1}\,P_{R_1} = P$ and $Q_{L_2}\,Q_{R_2} = Q$ since $[\sigma^1]_\sim = [\sigma^2]_\sim$. So $[\sigma^2]_\sim = [Q\,P]_\sim = \pi^\circ$ and (i) holds.

b) If $P_{R_2} = \emptyset$ then $\sigma^2 = Q_{L_2} P Q_{R_2}$. Hence, $P_{L_1} P_{R_1} = P$ and $Q_{R_2} Q_{L_2} = Q$ since $[\sigma^1]_\sim = [\sigma^2]_\sim$. So $[\sigma^2]_\sim = [Q P]_\sim = \pi^\circ$ and (i) holds.

c) If $Q_{L_2} = \emptyset$ then σ^2 is not equal to $P_{R_2} P_{L_2} Q$ but is a shift of it. Hence, $P_{L_1} P_{R_1} = P_{R_2} P_{L_2}$. This implies that either $P_{L_1} = P_{R_2}$ and $P_{R_1} = P_{L_2}$, or $P = P_{L_1} P_{R_1}$ by the argumentation that follows. Assume that P_{L_1} is a strict prefix of P_{R_2}, i. e., $P_{L_1} = P_{R_2}$ and $P_{R_1} = P_{L_2}$ does not hold. For the first element of P_{R_1} it holds that it has to occur in P to the right of all elements of P_{L_1}. This implies that $P_{L_1} P_{R_1} = P$. The case that P_{R_2} is a strict prefix of P_{L_1} is symmetric. In summary, (i) holds for $P = P_{L_1} P_{R_1}$ and (ii.b) holds for $P_{L_1} = P_{R_2}$ and $P_{R_1} = P_{L_2}$.

d) If $Q_{R_2} = \emptyset$ then $\sigma^2 = Q P_{L_2} P_{R_2}$. Hence, $P_{L_1} P_{R_1} = P_{L_2} P_{R_2}$ which is analogous to Case 2.c). It also follows that in this case either $P_{L_1} = P_{L_2}$ and $P_{R_1} = P_{R_2}$, or $P = P_{L_1} P_{R_1}$. In the first case (ii.a) holds and in the second case (i) holds.

The proof for the strings $\{P_{L_1}, P_{R_1}, Q_{R_1}\}$ can be done similarly.

3) First assume $Q_{L_1} = \emptyset$. In the following three cases are distinguished.

a) If $Q_{R_1} = \emptyset$ then $\sigma^1 = P_{L_1} P_{R_1}$. Hence, $[P_{L_1} P_{R_1}]_\sim = [P_{L_2} P_{R_2}]_\sim$ since $[\sigma^1]_\sim = [\sigma^2]_\sim$. This is a special case of Case 2.d).

b) If $P_{R_1} = \emptyset$ then $\sigma^1 = P Q$. Hence, (i) holds.

c) If $P_{L_1} = \emptyset$ then $\sigma^1 = P Q$. Hence, (i) holds.

The remaining combinations of two empty strings of $\{P_{L_1}, Q_{L_1}, P_{R_1}, Q_{R_1}\}$ follow in the same fashion.

4) Assume that exactly three strings of $\{P_{L_1}, Q_{L_1}, P_{R_1}, Q_{R_1}\}$ are empty. By definition it follows that σ^1 is a shift of P. Hence, (i) holds.

5) Since $\pi^\circ \in \mathcal{P}^\circ_n$ with $n \in \mathbb{N}$ all four strings of $\{P_{L_1}, Q_{L_1}, P_{R_1}, Q_{R_1}\}$ cannot be empty.

In summary, each case implies (i), (ii.a), or (ii.b) concluding the proof. □

See Figure 3.3 for an illustration of Theorem 3.1. The following corollary follows immediately from Theorem 3.1. It shows the relation between the two sets of elements that are kept in the left and the right copy in each of the two TDRL°s that have different origins but result in the same circular permutation.

Corollary 3.3. *If (ii) of Theorem 3.1 holds for two TDRL°s* $\rho^\circ_1, \rho^\circ_2 \in \mathcal{M}_{TDRL^\circ}$ *then either*

a) $L_2 = \mathcal{E}(Q_{R_1}) \cup \mathcal{E}(P_{L_1})$, $R_2 = \mathcal{E}(Q_{L_1}) \cup \mathcal{E}(P_{R_1})$ *or*

b) $R_2 = \mathcal{E}(Q_{R_1}) \cup \mathcal{E}(P_{L_1})$, $L_2 = \mathcal{E}(Q_{L_1}) \cup \mathcal{E}(P_{R_1})$.

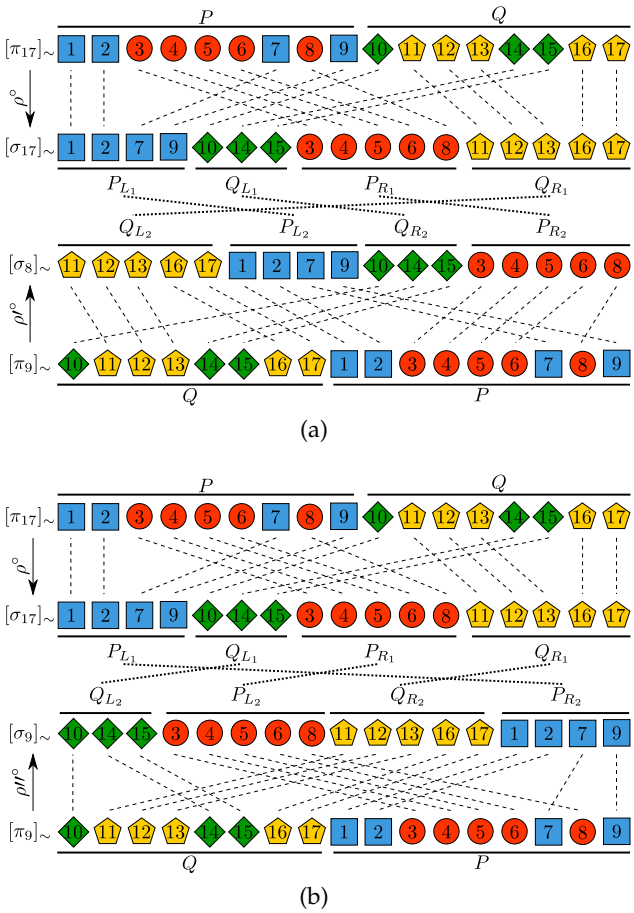


Figure 3.3: Illustration of Theorem 3.1 cases (ii.a) (subfigure (a)) and (ii.b) (subfigure (b)), where a TDRL° $\rho^\circ(L, R, p)$ is illustrated by TDRL $\rho(L, R)$ leading from $\pi_p \in \pi^\circ$ to $\rho(\pi_p) \in \sigma^\circ$; (a) TDRL° $\rho^\circ(L_1, R_1, p_1)$ with $L_1 = \{1, 2, 7, 9, 10, 14, 15\}$, $R_1 = \{3, 4, 5, 6, 8, 11, 12, 13, 16, 17\}$, $p_1 = 17$, and $\rho^\circ(\pi^\circ) = \sigma^\circ$ (upper part) and equivalent TDRL° $\rho'^\circ(L_2, R_2, p_2)$ with $L_2 = \{1, 2, 7, 9, 11, 12, 13, 16, 17\}$, $R_2 = \{3, 4, 5, 6, 8, 10, 14, 15\}$, and $p_2 = 9$ (bottom part); (b) TDRL° $\rho^\circ(L_1, R_1, p_1)$ (upper part) and equivalent TDRL° $\rho''^\circ(R_2, L_2, p_2)$ with L_i, R_i, and p_i, $i \in [1:2]$, as in (a) (bottom part). Symbols indicate to which substring the elements belong: square P_{L_1}, diamond Q_{L_1}, circle P_{R_1}, and pentagon Q_{R_1}. Dotted lines connect equal substrings, dashed lines connect equal elements.

Table 3.1: TDRL° $\rho'^{\circ}(L', R', p')$ that are equivalent to $\rho^{\circ}(L, R, p)$ applied to $\pi^{\circ} = [(2\ 3\ 4\ 1\ 5)]_{\sim}$, where $L = \{3, 5\}$, $R = \{1, 2, 4\}$, and $p = 5$. Hence, $\rho^{\circ}(\pi^{\circ}) = [(3\ 5\ 2\ 4\ 1)]_{\sim}$. Shown are representatives $\pi_{p'} \in \pi^{\circ}$, partition strings P and Q of π° for p with respect to p', L' and R', and the result of the application of the corresponding TDRL $\rho'(L', R')$ to $\pi_{p'}$ for all alternative origins $p' \in \{1, \ldots, 4\}$. For $p' \in \{1, 2\}$ the construction is according to Corollary 3.3.(a) and for the remaining origins $p' \in \{3, 4\}$ it is according to Corollary 3.3.(b).

p'	1	2	3	4
$\pi_{p'}$	(5 2 3 4 1)	(3 4 1 5 2)	(4 1 5 2 3)	(1 5 2 3 4)
P	2 3 4 1	2	2 3	2 3 4
Q	5	3 4 1 5	4 1 5	1 5
L'	$\{3\}$	$\{1, 4\}$	$\{2, 5\}$	$\{2, 4, 5\}$
R'	$\{1, 2, 4, 5\}$	$\{2, 3, 5\}$	$\{1, 3, 4\}$	$\{1, 3\}$
$\rho'(\pi_{p'})$	(3 5 2 4 1)	(4 1 3 5 2)	(5 2 4 1 3)	(5 2 4 1 3)

An example illustrating the statement of Corollary 3.3 is given in Table 3.1. The following corollary shows a property of two TDRL°s that have the same origin and result in the same permutation.

Corollary 3.4. *Let $\pi^{\circ} \in \mathcal{P}^{\circ}_{n}$ and let $\rho^{\circ}_{1}(L_1, R_1, p), \rho^{\circ}_{2}(L_2, R_2, p) \in \mathcal{M}_{TDRL^{\circ}}$. It holds that $\rho^{\circ}_{1}(\pi^{\circ}) = \rho^{\circ}_{2}(\pi^{\circ})$ if and only if*

i) $\rho^{\circ}_{1}(\pi^{\circ}) = \rho^{\circ}_{2}(\pi^{\circ}) = \pi^{\circ}$ *or*

ii) $\{L_1, R_1\} = \{L_2, R_2\}$.

Proof. Observe that the partition strings of π° for p with respect to p are $P = \pi_p$ and $Q = \emptyset$. Then the results follow by Theorem 3.1. □

Corollary 3.4 implies that two TDRL°s having the same origin and the same effect on a circular permutation either do not change the permutation, or both TDRL°s are equal in consideration of an exchange of the left and right copy, or both. Corollary 3.4 also implies that an exchange of the sets L and R of a TDRL° does not change its results, i. e., for $\pi^{\circ} \in \mathcal{P}^{\circ}_{n}$ and $\rho^{\circ}_{1}(L, R, p), \rho^{\circ}_{2}(R, L, p) \in \mathcal{M}_{TDRL^{\circ}}$ it holds that $\rho^{\circ}_{1}(\pi^{\circ}) = \rho^{\circ}_{2}(\pi^{\circ})$. However, the application of the corresponding TDRLs $\rho_1(L, R)$ and $\rho_2(R, L)$ to $\pi_p \in \pi^{\circ}$ results in two permutations $\rho_1(\pi_p)$ and $\rho_2(\pi_p)$ which differ by an $|L|$-shift or an $|R|$-shift, respectively. From a biological perspective, the results means that the knowledge of the origin of a TDRL° for gene order data alone does not allow to predict which genes were deleted in which copy. Therefore, the assumed TDRL° cannot be predicted uniquely.

The following corollary states that if a TDRL° changes the order of a circular permutation then there are exactly $2n$ TDRL°s – exactly two for each possible origin – that achieve the same effect.

Corollary 3.5. *Given $\pi^\circ \in \mathcal{P}_n^\circ$ and $\rho^\circ(L, R, p)$, consider $\sigma^\circ = \rho^\circ(\pi^\circ)$. For every $p' \in [1:n]$ one of the following two cases holds.*

i) If $\pi^\circ = \sigma^\circ$ there exist exactly $2n$ TDRL°s $\rho_i^\circ(L_i, R_i, p') \in \mathcal{M}_{TDRL^\circ}$, $i \in [1:2n]$ with $\rho_i^\circ(\pi^\circ) = \sigma^\circ$.

ii) If $\pi^\circ \neq \sigma^\circ$ there exist exactly two TDRL°s $\rho_1^\circ(L_1, R_1, p')$ and $\rho_2^\circ(L_2, R_2, p')$ with $\rho_1^\circ(\pi^\circ) = \rho_2^\circ(\pi^\circ) = \sigma^\circ$.

Proof. First it is shown that for a TDRL $\rho(L, R)$ and $\pi \in \mathcal{P}_n$ it holds that $\sigma = \rho(\pi) = \pi$ if and only if L is a prefix and R is a suffix of π. This is as otherwise there are $e, f \in [1:n]$ with $\pi^{-1}(e) < \pi^{-1}(f)$ and $e \in R$ and $f \in L$. Then $\sigma^{-1}(e) > \sigma^{-1}(f)$, and therefore $\sigma \neq \pi$.

Now assume a TDRL° $\rho'^\circ(L', R', p')$ exists with $\rho'^\circ(\pi^\circ) = \pi^\circ = \sigma^\circ$. Since $\rho'^\circ(\pi^\circ) = \pi^\circ$, it holds that $\rho'(\pi_{p'})$ is a shift of $\pi_{p'}$ for $\rho'(L', R')$. There are two possibilities: Either $\rho'(\pi_{p'}) = \pi_{p'}$, or $\rho'(\pi_{p'})$ is a shift of $\pi_{p'}$ that is different from $\pi_{p'}$. First assume $\rho'(\pi_{p'}) = \pi_{p'}$. Due to the first part of the proof, it holds that L is a prefix and R is a suffix of $\pi_{p'}$. Consequently, there exist $n+1$ choices for L and R. Now assume $\rho'(\pi_{p'})$ is a shift of $\pi_{p'}$ which is different from $\pi_{p'}$. Then, $\pi_{p'}$ can be written as $A\,B$, with both A and B non empty, such that $\rho'(\pi_{p'}) = B\,A$. This is only possible for $L = B$ and $R = A$ since otherwise not all elements of B precede A and not all elements of A follow B in $\rho'(\pi_{p'})$. There are $n-1$ other choices for L and R. Since no proper subsequence of $\pi_{p'}$ is both a prefix and a suffix of $\pi_{p'}$, it holds that $2n$ choices for L and R are distinct. That these are the only possible choices follows from the first part of the proof.

For (ii) Theorem 3.1 is implied with $\rho_1^\circ = \rho^\circ$ and $\rho_2^\circ(L_2, R_2, p')$, such that $\rho^\circ(\pi^\circ) = \rho_2^\circ(\pi^\circ)$. Because $\rho^\circ(\pi^\circ) \neq \pi^\circ$, this implies that Theorem 3.1.(ii) holds. By $\pi_p = P\,Q$, where P and Q are the partition strings of π° for p with respect to p', there are two possibilities for L_2 and R_2, which are determined by Theorem 3.1.(ii), cases (a) and (b). The possibility that is determined by Case (a) (respectively Case (b)) is denoted by $\rho_2(L_2, R_2)$ (respectively $\rho_3(L_3, R_3)$). The TDRL°s ρ_1° and ρ_2° of the statement of Corollary 3.5.(ii) are defined by $\rho_1^\circ = (L_3, R_3, p')$ and $\rho_2^\circ = (L_2, R_2, p')$. Proceeding by contradiction that ρ_1° and ρ_2° are distinct. Assume they are equal. Then $L_2 = L_3$ and $R_2 = R_3$. Thus, it follows by Theorem 3.1 that $P_L = P_{L_2} = P_{L_3} = P_R$. But $P_L = P_R$ can only hold when P is the empty string, which is not possible by the definition of partition strings. Hence, it concludes that ρ_1° and ρ_2° are indeed distinct. □

Corollary 3.5 implies that the number of equivalent TDRL°s can be determined as follows. For $\pi^\circ \in \mathcal{P}_n^\circ$ and $\rho^\circ(L, R, p) \in \mathcal{M}_{TDRL^\circ}$ the number of equivalent TDRL°s is $2n^2$ if $\rho^\circ(\pi^\circ) = \pi^\circ$ and it is $2n$ otherwise. Indeed, if $\rho^\circ(\pi^\circ) = \pi^\circ$ (respectively $\rho^\circ(\pi^\circ) \neq \pi^\circ$) for each of the n possible origins, there are $2n$ (respectively 2) choices of the bipartition (L, R). Note that the proof of Corollary 3.5 also implies a method to generate all circular TDRLs which are equivalent to a given TDRL°. This method is used to generate all equivalent TDRL° in the software program `EqualTDRL`, which has been presented in Hartmann et al. (2018c). `EqualTDRL` computes for two circular permutations that

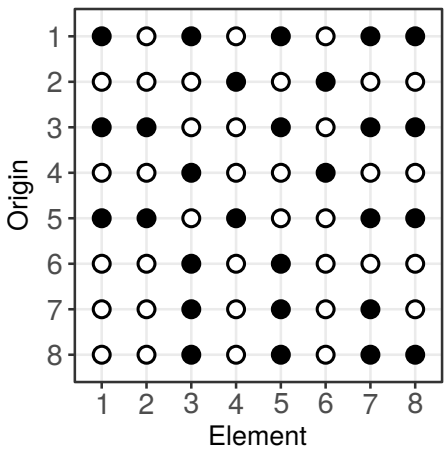


Figure 3.4: All circular TDRLs that rearrange $[(1\ 2\ 3\ 4\ 5\ 6\ 7\ 8)]_\sim$ into $[(1\ 2\ 4\ 6\ 3\ 5\ 7\ 8)]_\sim$. Each row illustrates the circular TDRLs $\rho_{\text{TDRL}^\circ}(L, R, p)$ and $\rho_{\text{TDRL}^\circ}(R, L, p)$, where p is the origin (y-axis) and an element (x-axis) is in the set L (respectively R) if the corresponding circle is filled with white color (respectively black color). The figure was created using `EqualTDRL`, see Hartmann et al. (2018c) for more information on this software.

differ by only one circular TDRL the set of all equivalent circular TDRLs, see Figure 3.4 and Example 3.6 for an illustration.

Example 3.6. *Consider the circular permutations* $\iota^\circ = [(1\ 2\ 3\ 4\ 5\ 6\ 7\ 8)]_\sim$ *and* $\pi^\circ = [(1\ 2\ 4\ 6\ 3\ 5\ 7\ 8)]_\sim$ *with* $d_{\mathcal{M}_{TDRL^\circ}}(\iota^\circ, \pi^\circ) = 1$. *All equivalent circular TDRLs that transform* ι° *into* π° *are:* $(\{2,4,6\},\{1,3,5,7,8\},1)$, $(\{1,2,3,5,7,8\},\{4,6\},2)$, $(\{3,4,6\},\{1,2,5,7,8\},3)$, $(\{1,2,4,5,7,8\},\{3,6\},4)$, $(\{3,5,6\},\{1,2,4,7,8\},5)$, $(\{1,2,4,6,7,8\},\{3,5\},6)$, $(\{1,2,4,6,8\},\{3,5,7\},7)$, $(\{1,2,4,6\},\{3,5,7,8\},8)$, *and all circular TDRLs that can be obtained from the listed TDRLs by interchanging the sets* L *and* R. *Figure 3.4 illustrates all those equivalent TDRLs.*

Recall that Corollary 3.5 demonstrates that it is not possible to reconstruct a TDRL° rearrangement from only the knowledge of the two circular permutations before and after the application of the TDRL°. This holds even if the origin is known as well. Consequently, for comparative analyses of gene orders additional data is necessary for evolutionary reconstructions of TDRL° rearrangements, e. g., DNA sequence information indicating remnants of genes. Corollary 3.5 implies the following corollary.

Corollary 3.6. *Given* $\pi^\circ \in \mathcal{P}^\circ_n$ *and* $\rho^\circ(L, R, p) \in \mathcal{M}_{TDRL^\circ}$, *consider* $\sigma^\circ = \rho^\circ(\pi^\circ)$. *For every* $p' \in [1:n]$ *there exist at least two TDRL°s* $\rho^\circ_1(L_1, R_1, p')$ *and* $\rho^\circ_2(L_2, R_2, p')$ *with* $\rho^\circ_1(\pi^\circ) = \rho^\circ_2(\pi^\circ) = \sigma^\circ$. *Equivalently, for every* $p' \in [1:n]$, *there exist at least two TDRLs* $\rho_i(L_i, R_i)$ *for* $i \in [1:2]$ *such that* $\rho_i(\pi_{p'}) \in \sigma^\circ$.

Circular TDRL scenarios can be mimicked by TDRL scenarios. Formally, this is stated in the following proposition which is used to determine the TDRL° distance in the following section.

Proposition 3.4. *Let $\pi^\circ \in \mathcal{P}^\circ_n$ and $\rho^\circ_i(L_i, R_i, p_i) \in \mathcal{M}_{TDRL^\circ}$, with $i \in [1:t]$ and $\rho^\circ_t \circ \cdots \circ \rho^\circ_1 \circ \pi^\circ = \sigma^\circ$. The following statements are true.*

i) For each $\pi \in \pi^\circ$, there exist a $\sigma \in \sigma^\circ$ and TDRLs $\rho'_i(L'_i, R'_i) \in \mathcal{M}_{TDRL}$, for all $i \in [1:t]$, such that $\rho'_t \circ \cdots \circ \rho'_1 \circ \pi = \sigma$.

ii) For each $\sigma \in \sigma^\circ$, there exist a $\pi \in \pi^\circ$ and TDRLs $\rho'_i(L'_i, R'_i) \in \mathcal{M}_{TDRL}$, for all $i \in [1:t]$, such that $\rho'_t \circ \cdots \circ \rho'_1 \circ \pi = \sigma$.

Proof. Consider first $t = 1$. Then either $\pi^\circ = \sigma^\circ$ or $\pi^\circ \neq \sigma^\circ$. Assume first $\pi^\circ = \sigma^\circ$. In this case, (i) and (ii) follow by $\pi = \sigma$ and $\rho'_1 = \rho'_1([1:n], \emptyset)$. Now assume $\pi^\circ \neq \sigma^\circ$. By Corollary 3.6 for all $p'_1 \in [1:n]$ there exists a TDRL $\rho'_1(L'_1, R'_1) \in \mathcal{M}_{TDRL}$ with L'_1 and R'_1 as in Corollary 3.3 such that $\rho'_1(\pi_{p'_1}) \in \sigma^\circ$. This proves (i) for $t = 1$. By Corollary 3.4 also the TDRL $\rho''_1(R'_1, L'_1)$ satisfies $\rho''_1(\pi_{p'_1}) \in \sigma^\circ$. Note that either $p'_1 \in L'_1$ or $p'_1 \in R'_1$. If $p'_1 \in R'_1$ then by construction p'_1 is the last element in $\pi_{p'_1}$ and by the definition of a TDRL it is the last element in $\rho'_1(\pi_{p'_1})$. Hence, since $\rho'_1(\pi_{p'_1}) \in \sigma^\circ$, it follows $\rho'_1(\pi_{p'_1}) = \sigma_{p'_1}$. Otherwise, $p'_1 \in L'_1$, and a similar argumentation implies that $\rho''_1(\pi_{p'_1}) = \sigma_{p'_1} \in \sigma^\circ$. Hence, for all $p'_1 \in [1:n]$ either $\rho'_1(\pi_{p'_1}) = \sigma_{p'_1}$ or $\rho''_1(\pi_{p'_1}) = \sigma_{p'_1}$ holds which proves (ii) for $t = 1$ with $\sigma = \sigma_{p'_1}$.

The result for $t > 1$ follows by iteration. Let $\pi^\circ_0 = \pi^\circ$ and $\pi^\circ_i = \rho^\circ_i(\pi^\circ_{i-1})$, $i \in [1:t]$, so that $\pi^\circ_t = \sigma^\circ$. Starting with $\pi = \pi^0 \in \pi^\circ_0$ (respectively $\pi^t = \sigma \in \sigma^\circ$) the proof for $t = 1$ implies that there exist TDRLs $\rho'_i \in \mathcal{M}_{TDRL}$, $i \in [1:t]$ with $\rho'_i(\pi^{i-1}) = \pi^i$ such that $\pi^i \in \pi^\circ_i$ (respectively $\pi^{i-1} \in \pi^\circ_{i-1}$). In particular, $\pi^t \in \sigma^\circ$ (respectively $\pi^0 \in \pi^\circ$). □

3.1.4 *Tandem Duplication Random Loss Distance on Directed Circular Permutations*

The following theorem states the formula for the $\mathcal{M}_{TDRL^\circ}$ distance for unsigned directed circular permutations.

Theorem 3.2. *For $\pi^\circ \in \mathcal{P}^\circ_n$ it holds that $d^\circ(\pi^\circ) = \lceil \log_2 \vartheta^\circ(\pi^\circ) \rceil$.*

Proof. The formal argument proceeds in several steps. In the first step $d^\circ(\pi^\circ) = \min_{\pi \in \pi^\circ} d_{\mathcal{M}_{TDRL}}(\pi, \iota)$ is proven. Let $t = \min_{\pi \in \pi^\circ} d_{\mathcal{M}_{TDRL}}(\pi, \iota)$ and $\rho_t \circ \ldots \circ \rho_1 \circ \pi = \iota$, with $\rho_i(L_i, R_i) \in \mathcal{M}_{TDRL}$ and $i \in [1:t]$, be a corresponding parsimonious TDRL scenario. Let $p_i = \pi^i(n)$ be the last element of permutation π^i which results from the application of the first $i \in [1:t]$ TDRLs, i.e., $\pi^i = \rho_i \circ \ldots \circ \rho_1 \circ \pi$. Thus, $\pi^0 = \pi$ and $\pi^t = \iota$. Then $\rho^\circ_t \circ \ldots \circ \rho^\circ_1 \circ \pi^\circ = \iota^\circ$, with $(\rho^\circ_1(L_1, R_1, p_0), \ldots, \rho^\circ_t(L_t, R_t, p_{t-1}))$ is a parsimonious scenario for π° under $\mathcal{M}_{TDRL^\circ}$. Hence, $d^\circ(\pi^\circ) \leqslant \min_{\pi \in \pi^\circ} d_{\mathcal{M}_{TDRL}}(\pi, \iota)$. Now consider a parsimonious scenario $(\rho^\circ_1, \ldots, \rho^\circ_t)$ with $\rho^\circ_1, \ldots, \rho^\circ_t \in \mathcal{M}_{TDRL^\circ}$ for π°. By Proposition 3.4 for $\iota \in \iota^\circ$ there exists a $\pi \in \pi^\circ$ and TDRLs $\rho'_i \in \mathcal{M}_{TDRL}$, $i \in [1:t]$, such that $\rho'_t \circ \cdots \circ \rho'_1 \circ \pi = \iota$. Hence, $\min_{\pi \in \pi^\circ} d_{\mathcal{M}_{TDRL}}(\pi, \iota) \leqslant d^\circ(\pi^\circ)$.

Altogether, $d^\circ(\pi^\circ) = \min_{\pi\in\pi^\circ} d_{\mathcal{M}_{\text{TDRL}}}(\pi,\iota)$ follows. Since $d_{\mathcal{M}_{\text{TDRL}}}(\pi,\iota) = \lceil \log_2(\vartheta(\pi))\rceil$ (Chaudhuri et al., 2006) and Proposition 3.3 the statement follows by

$$\begin{aligned}\min_{\pi\in\pi^\circ} d_{\mathcal{M}_{\text{TDRL}}}(\pi,\iota) &= \min_{\pi\in\pi^\circ}\lceil \log_2(\vartheta(\pi))\rceil = \lceil \log_2(\min_{\pi\in\pi^\circ}\vartheta(\pi))\rceil \\ &= \lceil \log_2(\vartheta^\circ(\pi))\rceil = \lceil \log_2(\vartheta^\circ(\pi^\circ))\rceil.\end{aligned}$$

□

Theorem 3.2 states that the distance problem for directed unsigned circular permutations can be solved by counting the circular chains of the considered permutation which can easily be done in linear time with respect to the size of the permutation. The following three corollaries follow directly from Theorem 3.2. Corollary 3.7 implies a simple method to compute the TDRL° distance by using the TDRL distance.

Corollary 3.7. *Let $[\pi]_\sim = \pi^\circ \in \mathcal{P}^\circ_n$. For $\sigma = \pi_n$ and $\gamma = \phi^{n-1}(\pi_1)$, i. e., the permutations in $[\pi]_\sim$ ending with n and starting with 1, respectively, it holds that $d^\circ(\pi^\circ) = d_{\mathcal{M}_{TDRL}}(\sigma,\iota) = d_{\mathcal{M}_{TDRL}}(\gamma,\iota)$.*

Proof. By Theorem 3.2 and Corollary 3.2 follows that $d^\circ(\pi^\circ) = \lceil \log_2 \vartheta^\circ(\pi^\circ)\rceil = \lceil \log_2 \vartheta(\sigma)\rceil = d_{\mathcal{M}_{\text{TDRL}}}(\sigma,\iota)$. The proof for γ is analogous. □

Corollary 3.8 shows that the sorting problem for unsigned directed circular permutations under $\mathcal{M}_{\text{TDRL}^\circ}$ can be solved in quasilinear time with respect to the size of the given permutation.

Corollary 3.8. *Let $[\pi]_\sim = \pi^\circ \in \mathcal{P}^\circ_n$ and let $S = (\rho(L_1,R_1),\ldots,\rho(L_t,R_t))$ with $t = d_{\mathcal{M}_{TDRL}}(\pi_n,\iota)$ be the parsimonious scenario for the representative $\pi_n \in \pi^\circ$ and ι which is obtained by the algorithm presented by Bernt et al. (2011). Consider the sequence $S' = (\rho^\circ_1,\ldots,\rho^\circ_t)$ for π°, where for all $i \in [1:t]$ it holds that $\rho^\circ_i = \rho^\circ(L_i,R_i,p_i)$ and p_i is the last element of permutation $\rho(L_{i-1},R_{i-1})\circ\ldots\circ\rho(L_1,R_1)\circ\pi_n$. It holds that S' is a parsimonious scenario for π° and ι°. Moreover, S' can be computed in time $\mathcal{O}(n\log n)$.*

Proof. Consider $[\pi]_\sim = \pi^\circ \in \mathcal{P}^\circ_n$, S, and S' as defined in the statement of the corollary. Since S can be obtained in time $\mathcal{O}(n\log n)$ (Bernt et al., 2011), it is not hard to verify that S' can also be obtained in time $\mathcal{O}(n\log n)$. By construction of S' it holds that $\rho(L_i,R_i)\circ\ldots\circ\rho(L_1,R_1)\circ\pi_n \in \rho^\circ_i\circ\ldots\circ\rho^\circ_1\circ\pi^\circ$ for all $i \in [1:t]$. Observe that this implies $S\circ\pi_n = \iota \in S'\circ\pi^\circ$. Consequently, $S'\circ\pi^\circ = \iota^\circ$ and S' is a scenario for π° and ι°. It remains to show that S' has a minimum length. By Corollary 3.7 it follows that $d^\circ(\pi^\circ) = d_{\mathcal{M}_{\text{TDRL}}}(\pi_n,\iota) = t$, hence S' is parsimonious. □

The following corollary of Theorem 3.2 gives the maximum value of the $\mathcal{M}_{\text{TDRL}^\circ}$ distance for all unsigned directed circular permutations, i. e., the diameter $D_{\mathcal{M}_{\text{TDRL}^\circ}}(\mathcal{P}^\circ_n)$.

Corollary 3.9. *The maximum value of the $\mathcal{M}_{TDRL^\circ}$ distance for all directed unsigned circular permutations of size n is $\lceil \log_2(n-1) \rceil$, i. e.,* $D_{\mathcal{M}_{TDRL^\circ}}(\mathcal{P}_n^\circ) = \lceil \log_2(n-1) \rceil$.

Proof. The corollary follows by:

$$\begin{aligned} D_{\mathcal{M}_{\mathrm{TDRL}^\circ}}(\mathcal{P}_n^\circ) &= \max_{\gamma^\circ,\sigma^\circ \in \mathcal{P}_n^\circ} d_{\mathcal{M}_{\mathrm{TDRL}^\circ}}(\gamma^\circ, \sigma^\circ) \\ &= \max_{\pi^\circ \in \mathcal{P}_n^\circ} d^\circ(\pi^\circ) \overset{\text{Thm. 3.2}}{=} \max_{\pi^\circ \in \mathcal{P}_n^\circ} \lceil \log_2 \vartheta^\circ(\pi^\circ) \rceil \\ &= \lceil \log_2 \max_{\pi^\circ \in \mathcal{P}_n^\circ} \vartheta^\circ(\pi^\circ) \rceil = \lceil \log_2(n-1) \rceil, \end{aligned}$$

where the first and the second equation follow by the definition of the diameter and the left-invariance of circular permutations, respectively. The last equation follows from the fact, that every circular permutation of size n has at most $n-1$ circular chains. □

3.1.5 *Tandem Duplication Random Loss Distance on Undirected Circular Permutations*

In the previous section it is shown that the $\mathcal{M}_{\mathrm{TDRL}^\circ}$ distance between two (directed circular) permutations is either less by one or equal to the $\mathcal{M}_{\mathrm{TDRL}}$ distance of the corresponding linear representatives. Hence, using an unfavorable choice of representatives may lead to an overestimation of the rearrangement distance in the circular case. Example 3.7 demonstrates that considering both reading directions of circular permutations, i. e., permutations are considered to be undirected, has a similar effect.

Example 3.7. *Consider the linear permutation $\pi = (3\ 2\ 1)$. Since $\vartheta(\pi) = 3$ it follows that $d_{\mathcal{M}_{TDRL}}(\pi, \iota) = \lceil \log_2 \vartheta(\pi) \rceil = \lceil \log_2 3 \rceil = 2$. If π is considered to be circular, i. e., $\pi^\circ := [\pi]_\sim$, then the number of circular chains of π° is 2. Theorem 3.2 implies that only one TDRL$^\circ$ is necessary to transform π° into the circular identity ι°. If both reading directions of π° are considered, then $\varpi^\circ := \{(3\ 2\ 1), (2\ 1\ 3), (1\ 3\ 2), (2\ 3\ 1), (3\ 1\ 2), (1\ 2\ 3)\}$. Observe that the identity permutation ι is already a representative of ϖ°. This implies that ϖ° is already identical to the (undirected circular) identity permutation ζ°. Hence, no TDRL$^\circ$ is necessary to transform ϖ° into ζ°.*

Example 3.7 illustrates that there is a difference of the $\mathcal{M}_{\mathrm{TDRL}^\circ}$ distance for directed circular permutations and the corresponding undirected circular permutations. Therefore, the $\mathcal{M}_{\mathrm{TDRL}^\circ}$ distance for undirected circular permutations is investigated in this section. In particular, it is shown that the distance problem and the sorting problem for undirected circular permutations of size n under $\mathcal{M}_{\mathrm{TDRL}^\circ}$ can be solved in time $\mathcal{O}(n)$ and $\mathcal{O}(n \log n)$, respectively. Moreover, it is shown that the $\mathcal{M}_{\mathrm{TDRL}^\circ}$ distance for an undirected circular permutation ϖ° is $\lceil \log_2 \vartheta^\circ(\pi) \rceil$, where π is a representative of ϖ° that has the minimum number of circular chains among all representatives of ϖ°.

In the following, properties of circular chains are identified that are crucial for analyzing the $\mathcal{M}_{\mathrm{TDRL}^\circ}$ distance for undirected circular

permutations. Recall that for a linear permutation $\pi = (\pi(1) \dots \pi(n))$ the permutation in which the order of all elements of π is reversed is denoted by $\overline{\pi}$, i. e., $\overline{\pi}(i) = \pi(n+1-i)$ for all $i \in [1:n]$. Consider a directed circular permutation π°. Then $\overline{\pi}^\circ$ denotes the circular permutation that has the opposite reading direction of π°, i. e., $\overline{\pi}^\circ := [\overline{\pi}]_\sim$ where π is a representative of π°. Permutation $\overline{\pi}^\circ$ is called the *mirrored* circular permutation of π°. Example 3.8 illustrates the definition of a mirrored circular permutation.

Example 3.8. *Consider the circular permutation* $\pi^\circ = [(3\ 4\ 2\ 1)]_\sim$. *Then,* $\overline{\pi}^\circ = [\overline{(3\ 4\ 2\ 1)}]_\sim = [(1\ 2\ 4\ 3)]_\sim = \{(1\ 2\ 4\ 3), (2\ 4\ 3\ 1), (4\ 3\ 1\ 2), (3\ 1\ 2\ 4)\}$.

The following proposition shows the connection between the number of circular chains of a directed circular permutation π° and its mirrored circular permutation $\overline{\pi}^\circ$.

Proposition 3.5. *Let* $\pi^\circ \in \mathcal{P}_n^\circ$ *with* $n > 1$, *then* $\vartheta^\circ(\pi^\circ) + \vartheta^\circ(\overline{\pi}^\circ) = n$.

Proof. Let π° be an unsigned directed circular permutation of size n with $n > 1$. The proposition is proven in two steps: First, it is shown by induction on n that $\vartheta^\circ(\pi) + \vartheta^\circ(\overline{\pi}) = n$ is true for all permutations $\pi \in \mathcal{P}_n$. Second, Remark 3.1 implies $\vartheta^\circ(\pi) = \vartheta^\circ(\pi^\circ)$ (respectively $\vartheta^\circ(\overline{\pi}) = \vartheta^\circ(\overline{\pi}^\circ)$) for all $\pi \in \pi^\circ$ (respectively $\overline{\pi} \in \overline{\pi}^\circ$).

Consider $n = 2$. It holds that either $\pi = (1\ 2)$ and $\overline{\pi} = (2\ 1)$ or vice versa. Since $\vartheta^\circ((1\ 2)) = \vartheta^\circ((2\ 1)) = 1$, the equation $\vartheta^\circ(\pi) + \vartheta^\circ(\overline{\pi}) = 2$ is satisfied, which proves the base case. For the induction step, assume that the equation $\vartheta^\circ(\sigma) + \vartheta^\circ(\overline{\sigma}) = t$ is satisfied for all $\sigma \in \mathcal{P}_t$ with $2 \leqslant t < n$. Now consider $\sigma \in \mathcal{P}_{n-1}$. Permutation σ can be transformed into a permutation π of size n by assigning element n to some position of σ. It follows from the definition of circular chains that adding element n can only influence the circular chains (or the circular chain) of σ that contain element 1 or $n-1$ and all other circular chains of σ (respectively $\overline{\sigma}$) are also circular chains of π (respectively $\overline{\pi}$). Without loss of generality, consider $\sigma^{-1}(1) < \sigma^{-1}(n-1)$. (Note that the case $\sigma^{-1}(1) > \sigma^{-1}(n-1)$ follows by reversing the roles of $\overline{\sigma}$ and σ.) Hence, it follows that $\overline{\sigma}^{-1}(n-1) < \overline{\sigma}^{-1}(1)$. Consequently, for permutation σ there exist either the two circular chains $(1, \dots)$ and $(\dots, n-1)$ or there exists one circular chain $(1, \dots, n-1)$, and for $\overline{\sigma}$ there exists the circular chain $(\dots, n-1, 1, \dots)$. There are three cases to add the element n to σ:

1) If n is assigned to the left of 1, then $\pi^{-1}(n) < \pi^{-1}(1) < \pi^{-1}(n-1)$. By definition, either the circular chain $(1, \dots)$ of σ becomes the circular chain $(n, 1, \dots)$ of π and circular chain $(\dots, n-1)$ of σ remains unchanged, i. e., $(n, 1, \dots)$ and $(\dots, n-1)$ are circular chains of π, or the circular chain $(1, \dots, n-1)$ of σ becomes the circular chain $(n, 1, \dots, n-1)$ of π. Thus, $\vartheta^\circ(\sigma) = \vartheta^\circ(\pi)$. For $\overline{\pi}$ it holds that $\overline{\pi}^{-1}(n-1) < \overline{\pi}^{-1}(1) < \overline{\pi}^{-1}(n)$. Hence, the circular chain $(\dots, n-1, 1, \dots)$ of $\overline{\sigma}$ is split into the circular chains $(\dots, n-1, n)$ and $(1, \dots)$ in $\overline{\pi}$. Hence, $\vartheta^\circ(\overline{\sigma}) = \vartheta^\circ(\overline{\pi}) - 1$.

2) If n is assigned between 1 and $n-1$, then it holds that $\pi^{-1}(1) < \pi^{-1}(n) < \pi^{-1}(n-1)$. Hence, the circular chains $(1, \dots)$

and $(\ldots, n-1)$ (or the circular chain $(1, \ldots, n-1)$) remain unchanged, i. e., they are circular chains of π, and a new circular chain $c = (n)$ of π is formed. Hence, $\vartheta^\circ(\sigma) = \vartheta^\circ(\pi) - 1$. For $\overline{\pi}$ it holds that $\overline{\pi}^{-1}(n-1) < \overline{\pi}^{-1}(n) < \overline{\pi}^{-1}(1)$, hence the circular chain $(\ldots, n-1, 1, \ldots)$ of $\overline{\sigma}$ becomes the circular chain $(\ldots, n-1, n, 1, \ldots)$ of $\overline{\pi}$, which implies $\vartheta^\circ(\overline{\sigma}) = \vartheta^\circ(\overline{\pi})$.

3) If n is assigned to the right of $n-1$, then $\pi^{-1}(1) < \pi^{-1}(n-1) < \pi^{-1}(n)$. By definition, either the circular chain $(1, \ldots)$ remains unchanged, i. e., it is a circular chain of π, and the circular chain $(\ldots, n-1)$ becomes $(\ldots, n-1, n)$ in π or the circular chain $(1, \ldots, n-1)$ of σ becomes the circular chain $(1, \ldots, n-1, n)$ of π. Thus, $\vartheta^\circ(\sigma) = \vartheta^\circ(\pi)$. For $\overline{\pi}$ it holds that $\overline{\pi}^{-1}(n-1) < \overline{\pi}^{-1}(1) < \overline{\pi}^{-1}(n)$, thus the circular chain $(\ldots, n-1, 1, \ldots)$ of $\overline{\sigma}$ is split into the circular chains $(\ldots, n-1, n)$ and $(1, \ldots)$ in $\overline{\pi}$. It follows that $\vartheta^\circ(\overline{\sigma}) = \vartheta^\circ(\overline{\pi}) - 1$.

From all three cases either $\vartheta^\circ(\sigma) = \vartheta^\circ(\pi)$ and $\vartheta^\circ(\overline{\sigma}) = \vartheta^\circ(\overline{\pi}) - 1$ (Case 1 and Case 3) or $\vartheta^\circ(\sigma) = \vartheta^\circ(\pi) - 1$ and $\vartheta^\circ(\overline{\sigma}) = \vartheta^\circ(\overline{\pi})$ (Case 2) is obtained. The equation $\vartheta^\circ(\pi) + \vartheta^\circ(\overline{\pi}) = n$ is deduced by combining the respective equations with the induction hypothesis, i. e., the equation $\vartheta^\circ(\sigma) + \vartheta^\circ(\overline{\sigma}) = n - 1$. By Remark 3.1 the statement follows. □

In addition to the statement from Proposition 3.5, there is another interesting connection between a directed circular permutation and its mirrored permutation. That is, if a TDRL° transforms a directed circular permutation π° into σ°, then there always exists another TDRL° that transforms $\overline{\pi}^\circ$ into $\overline{\sigma}^\circ$.

Lemma 3.1. *Let $\rho^\circ(L, R, p)$ be a TDRL° for the directed circular permutation $\pi^\circ \in \mathcal{P}_n^\circ$. Then $\rho^\circ(L, R, p) \circ \pi^\circ = \sigma^\circ$ if and only if $\rho^\circ(R, L, \pi_p(1)) \circ \overline{\pi}^\circ = \overline{\sigma}^\circ$.*

Proof. Let $\rho^\circ(L, R, p)$ be a TDRL° for π°. Moreover, let $L_1, \ldots, L_\ell$ and $R_1, \ldots, R_r$ be the elements of L and R ordered with respect to $\pi_p = (\pi_p(1), \ldots, \pi_p(n))$, i. e., for all $i, j \in [1:\ell]$ it holds that $\pi_p^{-1}(L_i) < \pi_p^{-1}(L_j)$ if and only if $i < j$, and for all $e, f \in [1:r]$ is holds that $\pi_p^{-1}(R_e) < \pi_p^{-1}(R_f)$ and if and only if $e < f$. It holds that $\rho^\circ(L, R, p) \circ \pi^\circ = [\rho(L, R) \circ \pi_p]_\sim = [(L_1 \ldots L_\ell R_1 \ldots R_r)]_\sim = \sigma^\circ$. Since the order of the elements in $\overline{\pi_p}$ are reversed with respect to π_p it follows that $\rho(R, L) \circ \overline{\pi_p} = (R_r \ldots R_1 L_\ell \ldots L_1)$. Observe that $\pi_p(1)$ is the first element of π_p. Hence, $\pi_p(1)$ is the last element of $\overline{\pi_p}$. Since $\overline{\pi_p} \in \overline{\pi}^\circ$ and each representative ending with a certain element is unique in $\overline{\pi}^\circ$, it follows that $\overline{\pi_p} = \overline{\pi}_{\pi_p(1)}$. Consequently, $\rho^\circ(R, L, \pi_p(1)) \circ \overline{\pi}^\circ = [\rho(R, L) \circ \overline{\pi}_{\pi_p(1)}]_\sim = [\rho(R, L) \circ \overline{\pi_p}]_\sim = [(R_r \ldots R_1 L_\ell \ldots L_1)]_\sim = \overline{\sigma}^\circ$. Thus, the implication from left to right is true. The other direction follows from this implication and the fact that $\overline{\overline{\pi}}^\circ = \pi^\circ$ for all $\pi \in \mathcal{P}_n^\circ$. □

Undirected and directed circular permutations are both considered in the remainder of this section. Therefore, the definition of unsigned undirected circular permutations are recalled in the following

paragraph. In addition, the difference in the formal definition of the TDRL° operation between both types of permutations is pointed out.

Recall that the set of all undirected circular permutations of size n is denoted by $u\mathcal{P}_n^\circ$, i. e., $u\mathcal{P}_n^\circ := \{\pi^\circ \cup \overline{\pi}^\circ : \pi^\circ \in \mathcal{P}_n^\circ\}$. To avoid any misunderstanding, undirected circular permutations are denoted by ϖ°, ς°, and the identity in $u\mathcal{P}_n^\circ$ is denoted by ζ°, i. e., $\zeta^\circ := \iota^\circ \cup \overline{\iota}^\circ$. As the focus in the previous sections of Chapter 3 is on directed permutations, the definition of the TDRL° rearrangements cannot directly be applied to undirected circular permutation as defined in Section 3.1.1. That is because for an undirected circular permutation $\varpi^\circ = \pi^\circ \cup \overline{\pi}^\circ$ of size n there exist exactly two representatives with a certain element $p \in [1:n]$ on the last position, namely $\pi_p \in \pi^\circ$ and $\overline{\pi}_p \in \overline{\pi}^\circ$. In contrast, for a directed circular permutation π° of size n, such a representative $\pi_p \in \pi^\circ$ is unique. Therefore, some adjustments have to be made on the TDRL° operation as explained in the following (see also Example 3.9): A TDRL° for an undirected circular permutation $\varpi^\circ \in u\mathcal{P}_n^\circ$ is defined as a mapping $\rho^\circ : u\mathcal{P}_n^\circ \to u\mathcal{P}_n^\circ$ recorded by $\rho^\circ(L, R, \pi)$, where (L, R) is a bipartition of $[1:n]$, $\pi \in \varpi^\circ$, and $\rho^\circ(L, R, \pi) \circ \varpi^\circ := [\rho(L, R) \circ \pi]_\sim \cup [\overline{\rho(L, R) \circ \pi}]_\sim$. It is not hard to see that the TDRL° operation for undirected circular permutations is essentially the same as for directed circular permutations. For that reason and the sake of simplicity, the notations for the TDRL° rearrangement on directed circular permutations are also used, in slight abuse of notation, for undirected circular permutations in the remainder of this section.

Example 3.9. *Consider the undirected circular permutation* $\varpi^\circ = [(1\ 2\ 3\ 5\ 4)]_\sim \cup [(4\ 5\ 3\ 2\ 1)]_\sim$. *The set of all representatives of* ϖ° *is* $\{(1\ 2\ 3\ 5\ 4), (2\ 3\ 5\ 4\ 1), (3\ 5\ 4\ 1\ 2), (5\ 4\ 1\ 2\ 3), (4\ 1\ 2\ 3\ 5), (5\ 3\ 2\ 1\ 4), (3\ 2\ 1\ 4\ 5), (2\ 1\ 4\ 5\ 3), (1\ 4\ 5\ 3\ 2), (4\ 5\ 3\ 2\ 1)\}$. *The application of the TDRL°* $\rho^\circ(\{1,3,4\},\{2,5\},(3\ 5\ 4\ 1\ 2))$ *to* ϖ° *results in* $\rho^\circ(\{1,3,4\},\{2,5\},(3\ 5\ 4\ 1\ 2)) \circ \varpi^\circ = [\rho(\{1,3,4\},\{2,5\}) \circ (3\ 5\ 4\ 1\ 2)]_\sim \cup [\overline{\rho(\{1,3,4\},\{2,5\}) \circ (3\ 5\ 4\ 1\ 2)}]_\sim = [(3\ 4\ 1\ 5\ 2)]_\sim \cup [(2\ 5\ 1\ 4\ 3)]_\sim$.

In the remainder of this section, it is shown that the distance as well as the sorting problem for undirected circular permutations can be solved by using the same ideas as in the directed case. Let $S = (\rho_1^\circ, \ldots, \rho_{|S|}^\circ)$ be a sequence of TDRL°s for an undirected circular permutation ϖ°. Consider that S is applied to ϖ° and let ς_i° denote the intermediate undirected circular permutation obtained after the application of ρ_i with $i \in [1:|S|-1]$. Note that it is possible that the linear representative of ς_i° has to be mirrored (and possibly shifted) before the next TDRL° ρ_{i+1}° from S can be applied. Those mirroring operations are no rearrangements. In fact, they are just used for technical reasons since a TDRL° is always applied on a linear directed representative. Now, a sequence S might, or might not, require those mirroring operations on intermediate permutations. Thus, a sequence for a directed circular permutation can always be represented as a sequence for the corresponding undirected circular permutations (see Lemma 3.2).

Moreover, if an undirected circular permutation $\varpi^\circ \in \mathcal{UP}_n^\circ$ contains a linear representative $\pi \in \pi^\circ$, then it also contains all linear variants from the directed circular permutation π° as well as from the mirrored directed circular permutation $\overline{\pi}^\circ$. In the following, it is shown that in order to solve the distance problem (respectively the sorting problem) for undirected circular permutations under $\mathcal{M}_{\text{TDRL}^\circ}$, it is sufficient to compute four parsimonious scenarios, one for each of the following directed circular permutations. Either π° has to be transformed into ι°, $\overline{\pi}^\circ$ has to be transformed into ι°, π° has to be transformed into $\overline{\iota}^\circ$, or $\overline{\pi}^\circ$ has to be transformed into $\overline{\iota}^\circ$. Whichever of these scenarios has the smallest length is a parsimonious scenario for the undirected case. In particular, this means that for any undirected circular permutation ϖ° there is always a parsimonious scenario that transforms π° into either ι° or $\overline{\iota}^\circ$, such that no mirroring operation is required at any intermediate step (see Proposition 3.6). To show this, two auxiliary lemmata are needed.

The following lemma shows that a scenario for two directed circular permutations can be translated almost one-to-one into a scenario for undirected circular permutations.

Lemma 3.2. *Let S be a scenario of TDRL°s for the directed circular permutations $\pi^\circ \in \mathcal{P}_n^\circ$ and $\sigma^\circ \in \mathcal{P}_n^\circ$. Then, there always exists a scenario S' of TDRL°s for the undirected circular permutations $\varpi^\circ = \pi^\circ \cup \overline{\pi}^\circ$ and $\varsigma^\circ = \sigma^\circ \cup \overline{\sigma}^\circ$ with $|S| = |S'|$.*

Proof. Let $S = (\rho_1^\circ, \ldots, \rho_{|S|}^\circ)$ be a scenario for π° and σ° under $\mathcal{M}_{\text{TDRL}^\circ}$ with $\rho_i^\circ := \rho_i^\circ(L_i, R_i, p_i)$ for all $i \in [1\colon|S|]$. For all $i \in [1\colon|S|]$ the circular permutation π_i° denotes the intermediate directed circular permutation that is obtained in the i-th step of the application of S, i.e., $\pi_i^\circ := \rho_i \circ \ldots \circ \rho_1 \circ \pi^\circ$ with $\pi_{|S|}^\circ = \sigma^\circ$ and $\pi_0^\circ := \pi^\circ$. Moreover, for all $i \in [0\colon|S|-1]$ let π_{p_i} denote the representatives of π_i° that ends with element π_i. Then, for the sequence $S' = (\varrho_1^\circ(L_1, R_1, \pi_{p_0}), \ldots, \varrho_{|S|}^\circ(L_{|S|}, R_{|S|}, \pi_{p_{|S|-1}}))$ it holds that $\pi_i^\circ \subseteq \varrho_i^\circ \circ \ldots \circ \varrho_1^\circ \circ \varpi^\circ$ for all $i \in [1\colon|S|]$. Note that $\pi_0^\circ \subseteq \varpi^\circ$ and $\pi_{|S|}^\circ = \sigma^\circ \subseteq \varsigma^\circ$. Therefore, S' is a scenario for the undirected circular permutations ϖ° and ς° under $\mathcal{M}_{\text{TDRL}^\circ}$. By construction, for each TDRL° of S a TDRL° of S' was constructed, and hence $|S| = |S'|$. □

The following proposition states that for every parsimonious scenario of TDRL°s for undirected permutations $\varpi^\circ = \pi^\circ \cup \overline{\pi}^\circ$ and $\zeta^\circ = \iota^\circ \cup \overline{\iota}^\circ$, there exists a parsimonious scenario S of the same length for the directed circular permutations π° and ι° (or π° and $\overline{\iota}^\circ$). An example for such a scenario can be found in Figure 3.5.

Proposition 3.6. *Let S be a parsimonious scenario of TDRL°s for the undirected circular permutations $\varpi^\circ = \pi^\circ \cup \overline{\pi}^\circ$ and $\zeta^\circ = \iota^\circ \cup \overline{\iota}^\circ$. Then there exists a scenario S' of TDRL°s for the directed circular permutations π° and ι° or π° and $\overline{\iota}^\circ$ with $|S| = |S'|$. Moreover, S' is a parsimonious scenario.*

Proof. The formal proof proceeds in two steps. First, it is shown that such a scenario S' exists. Second, it is shown that S' is parsimonious.

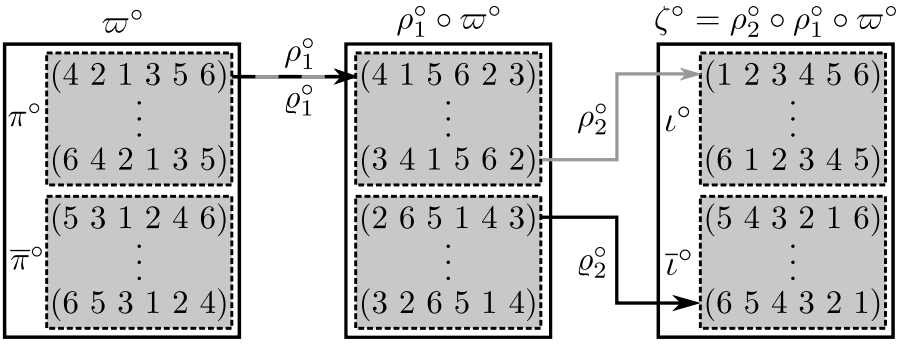


Figure 3.5: Scenario $S = (\varrho_1^\circ, \varrho_2^\circ)$ for the undirected circular permutations $\varpi^\circ = \pi^\circ \cup \overline{\pi}^\circ$ and $\zeta^\circ = \iota^\circ \cup \overline{\iota}^\circ$, where $\pi^\circ = [(4\ 2\ 1\ 3\ 5\ 6)]_\sim$, $\overline{\pi}^\circ = [(6\ 5\ 3\ 1\ 2\ 4)]_\sim$, $\varrho_1^\circ = \varrho^\circ_{\mathrm{TDRL}^\circ}(\{1,4,5,6\},\{2,3\},(4\ 2\ 1\ 3\ 5\ 6))$, and $\varrho_2^\circ = \varrho^\circ_{\mathrm{TDRL}^\circ}(\{3,4,5,6\},\{1,2\},(2\ 6\ 5\ 1\ 4\ 3))$. Illustrated are undirected circular permutations (continuous black square); directed circular permutations (gray square with dashed border); TDRL°s (black and gray arrows). The tail and the head of an arrow points out which representatives of the undirected circular permutations are involved in the TDRL° that is applied. Scenario S is illustrated by black arrows. By Theorem 3.3 it holds that $d_{\mathcal{M}_{\mathrm{TDRL}^\circ}}(\varpi^\circ, \zeta^\circ) = \min\{\lceil \log_2 \vartheta^\circ(\pi^\circ)\rceil, \lceil \log_2 \vartheta^\circ(\overline{\pi}^\circ)\rceil\} = \min\{\lceil \log_2 3\rceil, \lceil \log_2 3\rceil\} = 2$. Hence, scenario S is parsimonious. By Proposition 3.6 there exists a scenario S' of directed circular permutations π° and ι° (or π° and $\overline{\iota}^\circ$) that is parsimonious and has the same length as S. An example for such a scenario is $S' = (\rho_1^\circ, \rho_2^\circ)$ with $\rho_1^\circ = \rho^\circ_{\mathrm{TDRL}^\circ}(\{1,4,5,6\},\{2,3\},6)$ and $\rho_1^\circ = \rho^\circ_{\mathrm{TDRL}^\circ}(\{1,2\},\{3,4,5,6\},2)$. Scenario S' is illustrated by gray arrows.

Let $S = (\varrho_1^\circ, \ldots, \varrho_{|S|}^\circ)$ be a parsimonious scenario for ϖ° and ζ° such that $\varrho_i^\circ := \varrho_i^\circ(L_i, R_i, \pi_{i-1})$ is a TDRL° for $\varsigma_{i-1}^\circ := \varrho_{i-1}^\circ \circ \ldots \circ \varrho_1^\circ \circ \varpi^\circ$ for all $i \in [1:|S|]$ with $\varsigma_0^\circ = \varpi^\circ$ and $\varsigma_{|S|}^\circ = \zeta^\circ$. Note that for the first TDRL° $\varrho_1^\circ(L_1, R_1, \pi_0)$ it holds that $\pi_0 \in \varpi^\circ$ and either $\pi_0 \in \pi^\circ$ or $\pi_0 \in \overline{\pi}^\circ$.

In the following, a sequence $S' = (\rho_1^\circ, \ldots, \rho_{|S|}^\circ)$ for the directed circular permutation π° is constructed such that for all $i \in [1:|S|]$ it holds that $\rho_i^\circ \circ \ldots \circ \rho_1^\circ \circ \pi^\circ =: \sigma_i^\circ \subset \varsigma_i^\circ$.

The scenario S' is constructed iteratively for increasing $i = 1, \ldots, |S|$. Let $i = 1$ and assume first that $\pi_0 \in \pi^\circ$. It follows that the TDRL° $\rho_1^\circ(L_1, R_1, \pi_0(n))$ is the directed variant of $\varrho_1^\circ(L_1, R_1, \pi_0)$. In particular, since $\pi_0(n)$ is the last element of π_0, it holds by definition that the TDRL $\rho(L_1, R_1)$ is applied to π_0, i.e., $\rho_1^\circ(L_1, R_1, \pi_0(n)) \circ \pi^\circ = [\rho(L_1, R_1) \circ \pi_0]_\sim$. Note that the same TDRL is applied to π_0 in the undirected case. Thus, it follows that $\rho_1^\circ(L_1, R_1, \pi_0(n)) \circ \pi^\circ \subset \varsigma_1^\circ$.

Now assume that $\pi_0 \in \overline{\pi}^\circ$. Let $\rho^\circ(L_1, R_1, \pi_0(n)) \circ \overline{\pi}^\circ = \sigma_1^\circ$ be the directed permutation that is created when applying the directed variant of the first TDRL° from S (i.e., $\varrho_1^\circ(L_1, R_1, \pi_0)$) to the permutation $\pi_0 \in \overline{\pi}^\circ$. Then, $\sigma_1^\circ \subset \varsigma_1^\circ$. By Lemma 3.1 it follows that $\rho_1^\circ(R_1, L_1, \pi_0(1)) \circ \overline{\overline{\pi}}^\circ = \rho_1^\circ(R_1, L_1, \pi_0(1)) \circ \pi^\circ = \overline{\sigma}_1^\circ$. Now, since $\sigma_1^\circ \subset \varsigma_1^\circ$ also $\overline{\sigma}_1^\circ \subset \varsigma_1^\circ$ and thus, $\rho_1^\circ(R_1, L_1, \pi_0(1)) \circ \pi^\circ \subset \varsigma_1^\circ$.

Consequently, for both cases a TDRL° can be found that transforms π° into a directed permutation that is a subset of ς_1°. Therefore, the first TDRL° ρ_1° from S' is either $\rho^\circ(L_1, R_1, \pi_0(n))$ or $\rho^\circ(R_1, L_1, \pi_0(1))$

depending on the respective case. In addition, the intermediate directed permutation $\rho_1^\circ \circ \pi^\circ$ is either σ_1° or its mirror $\overline{\sigma}_1^\circ$.

The remaining TDRL°s of sequence S' are constructed iteratively for increasing $i = 2, \ldots, |S|$ by the following procedure which uses the same idea as for $i = 1$. If $\pi_{i-1} \in \sigma_{i-1}^\circ$, then $\rho_i^\circ := \rho_i^\circ(L_i, R_i, \pi_{i-1}(n))$. Otherwise, if $\pi_{i-1} \in \overline{\sigma}_{i-1}^\circ$, then $\rho_i^\circ := \rho_i^\circ(R_i, L_i, \pi_{i-1}(1))$. In both cases the next intermediate directed circular permutation is contained in ς_i°.

As a result of this procedure, the scenario S' is obtained which satisfies that $S' \circ \pi^\circ = \sigma_{|S|}^\circ \subset \varsigma_{|S|}^\circ = \zeta^\circ$. Since, $\zeta^\circ = \iota^\circ \cup \overline{\iota}^\circ$, it holds that either $S' \circ \pi^\circ = \iota^\circ$ or $S' \circ \pi^\circ = \overline{\iota}^\circ$. Recall that $\pi^\circ \in \varpi^\circ$, therefore S' is a scenario for π° and ι° or for π° and $\overline{\iota}^\circ$. By construction, for every TDRL° ϱ_i° from S a TDRL° ρ_i° from S' was constructed. Therefore, both scenarios have the same length, i. e., $|S| = |S'|$.

It remains to show that S' is parsimonious. By contraposition assume that S' is not parsimonious. Hence, there exists a scenario S'' for π° and ι° or π° and $\overline{\iota}^\circ$ such that $|S''| < |S'|$. By Lemma 3.2 there exists a scenario T of TDRL°s for ϖ° and ζ° with $|T| = |S''|$. However, this implies $|T| = |S''| < |S'| = |S|$, a contradiction to the assumption that S is parsimonious. Consequently, such a scenario S'' cannot exist and S' is parsimonious. □

The following theorem gives a closed formula for the $\mathcal{M}_{\text{TDRL}^\circ}$ distance for undirected circular permutations.

Theorem 3.3. *For an undirected circular permutation* $\varpi^\circ = \pi^\circ \cup \overline{\pi}^\circ \in \mathcal{UP}_n^\circ$ *holds:* $d_{\mathcal{M}_{TDRL^\circ}}(\varpi^\circ, \zeta^\circ) = \min\{\lceil \log_2 \vartheta^\circ(\pi^\circ) \rceil, \lceil \log_2 \vartheta^\circ(\overline{\pi}^\circ) \rceil\}$.

Proof. Let $\varpi^\circ \in \mathcal{UP}_n^\circ$ with $\varpi^\circ = \pi^\circ \cup \overline{\pi}^\circ$. Consider a parsimonious scenario S for ϖ° and ζ°. Since S is parsimonious, it has the minimum length, i. e., $d_{\mathcal{M}_{\text{TDRL}^\circ}}(\varpi^\circ, \zeta^\circ) = |S|$. By Proposition 3.6 there exists a parsimonious scenario S' for π° and ι° or π° and $\overline{\iota}^\circ$ with $|S| = |S'|$. If S' is a scenario for π° and ι°, then by Theorem 3.2 it holds that $|S'| = \lceil \log_2 \vartheta^\circ(\pi^\circ) \rceil$. Consequently, it follows $d_{\mathcal{M}_{\text{TDRL}^\circ}}(\varpi^\circ, \zeta^\circ) = \lceil \log_2 \vartheta^\circ(\pi^\circ) \rceil$ in this case.

Now, consider that $|S'|$ is a scenario for π° and $\overline{\iota}^\circ$. Recall that the number of rearrangements transforming π° into $\overline{\iota}^\circ$ is not dependent on the way the elements of both directed circular permutations are numbered. Therefore, the elements of both permutations can be renamed without changing the $\mathcal{M}_{\text{TDRL}^\circ}$ distance. Such a renaming is to rename the elements of each representative of $\overline{\iota}^\circ$ such that $\overline{\iota}^\circ$ becomes ι°. Formally, this renaming is to apply the linear permutation $\overline{\iota}$ to all representatives of $\overline{\iota}^\circ$. Applying the same renaming to π° transforms π° into $\overline{\pi}^\circ$. Therefore, it holds that $d_{\mathcal{M}_{\text{TDRL}^\circ}}(\pi^\circ, \overline{\iota}^\circ) = d_{\mathcal{M}_{\text{TDRL}^\circ}}(\overline{\pi}^\circ, \iota^\circ)$. By Theorem 3.2, it follows that $d_{\mathcal{M}_{\text{TDRL}^\circ}}(\overline{\pi}^\circ, \iota^\circ) = \lceil \log_2 \vartheta^\circ(\overline{\pi}^\circ) \rceil$. Consequently, in the second case it follows $d_{\mathcal{M}_{\text{TDRL}^\circ}}(\varpi^\circ, \zeta^\circ) = \lceil \log_2 \vartheta^\circ(\overline{\pi}^\circ) \rceil$.

The theorem follows by the fact that the $\mathcal{M}_{\text{TDRL}^\circ}$ distance is always the minimum of both cases. □

The following corollary immediately follows from Theorem 3.3. It shows that a parsimonious scenario of TDRL°s for a pair of undirected circular permutations of size n can be computed in quasilinear time with respect to n.

Corollary 3.10. *A parsimonious scenario for an undirected circular permutation $\varpi^\circ \in \mathcal{UP}_n^\circ$ and ζ° under $\mathcal{M}_{TDRL^\circ}$ can be computed in time $\mathcal{O}(n \log n)$.*

Proof. Let $\varpi^\circ \in \mathcal{UP}_n^\circ$ with $\varpi^\circ = \pi^\circ \cup \overline{\pi}^\circ$. By Theorem 3.3 the $\mathcal{M}_{\text{TDRL}^\circ}$ distance between ϖ° and ζ° is either $\lceil \log_2 \vartheta^\circ(\pi^\circ) \rceil$ or $\lceil \log_2 \vartheta^\circ(\overline{\pi}^\circ) \rceil$. Without loss of generality consider that the $\mathcal{M}_{\text{TDRL}^\circ}$ distance is obtained for $\lceil \log_2 \vartheta^\circ(\pi^\circ) \rceil$. Observe that the distance for the directed permutations $\pi^\circ \subset \varpi^\circ$ and $\iota^\circ \subset \zeta^\circ$ is also $\lceil \log_2 \vartheta^\circ(\pi^\circ) \rceil$. Therefore, let S be a parsimonious scenario for π° and ι° which can be obtained in $\mathcal{O}(n \log n)$ by the algorithm from Bernt et al. (2011). By Lemma 3.2 it holds that there exists a scenario S' for the undirected circular permutations $\varpi^\circ \in \mathcal{UP}_n^\circ$ and ζ° with $|S'| = |S|$ that can be constructed in time $\mathcal{O}(|S|)$ from S as explained in the proof of Lemma 3.2. Since $|S'| = |S| = \lceil \log_2 \vartheta^\circ(\pi^\circ) \rceil = d_{\mathcal{M}_{\text{TDRL}^\circ}}(\varpi^\circ, \zeta^\circ)$, it holds that S' is parsimonious.

Hence, scenario S' can be obtained in time $\mathcal{O}(|T| + n \log n)$. Since $|T| < n$, it follows that S' can be obtained in $\mathcal{O}(n \log n)$. □

The following corollary of Theorem 3.3 determines the diameter of the $\mathcal{M}_{\text{TDRL}^\circ}$ distance for the set of all undirected circular permutations.

Corollary 3.11. *The maximum value of the $\mathcal{M}_{TDRL^\circ}$ distance for all undirected circular permutations of size n is $\lceil \log_2 \lfloor n/2 \rfloor \rceil$, i.e., $D_{\mathcal{M}_{TDRL^\circ}}(\mathcal{UP}_n^\circ) = \lceil \log_2 \lfloor n/2 \rfloor \rceil$.*

Proof. The sought diameter can be expressed as:

$$\begin{aligned}
D_{\mathcal{M}_{\text{TDRL}^\circ}}(\mathcal{UP}_n^\circ) &= \max_{\varpi_1^\circ, \varpi_2^\circ \in \mathcal{UP}_n^\circ} d_{\mathcal{M}_{\text{TDRL}^\circ}}(\varpi_1^\circ, \varpi_2^\circ) \\
&= \max_{\varpi^\circ \in \mathcal{UP}_n^\circ} d_{\mathcal{M}_{\text{TDRL}^\circ}}(\varpi^\circ, \zeta^\circ) \\
&\overset{\text{Thm. 3.3}}{=} \max_{\pi^\circ \cup \overline{\pi}^\circ \in \mathcal{UP}_n^\circ} \min\{\lceil \log_2 \vartheta^\circ(\pi^\circ) \rceil, \lceil \log_2 \vartheta^\circ(\overline{\pi}^\circ) \rceil\} \\
&= \lceil \log_2 (\max_{\pi^\circ \cup \overline{\pi}^\circ \in \mathcal{UP}_n^\circ} \min\{\vartheta^\circ(\pi^\circ), \vartheta^\circ(\overline{\pi}^\circ)\}) \rceil \\
&= \lceil \log_2 (\max_{\pi^\circ \cup \overline{\pi}^\circ \in \mathcal{UP}_n^\circ} \frac{\vartheta^\circ(\pi^\circ) + \vartheta^\circ(\overline{\pi}^\circ) - |\vartheta^\circ(\pi^\circ) - \vartheta^\circ(\overline{\pi}^\circ)|}{2}) \rceil,
\end{aligned}$$

where the first (second) equation follows from the definition of the diameter (respectively the left-invariance of permutations). The application of Proposition 3.5 yields:

$$D_{\mathcal{M}_{\text{TDRL}^\circ}}(\mathcal{UP}_n^\circ) = \lceil \log_2 (\max_{\pi^\circ \cup \overline{\pi}^\circ \in \mathcal{UP}_n^\circ} \frac{n - |\vartheta^\circ(\pi^\circ) - \vartheta^\circ(\overline{\pi}^\circ)|}{2}) \rceil.$$

It is easy to verify that the maximum of $(n - |\vartheta^\circ(\pi^\circ) - \vartheta^\circ(\overline{\pi}^\circ)|)/2$ is obtained if the term $|\vartheta^\circ(\pi^\circ) - \vartheta^\circ(\overline{\pi}^\circ)|$ is minimized. Proposition 3.5 implies $|\vartheta^\circ(\pi^\circ) - \vartheta^\circ(\overline{\pi}^\circ)| \in [0 : n-1]$ if n is even and

$|\vartheta^\circ(\pi^\circ) - \vartheta^\circ(\overline{\pi}^\circ)| \in [1:n-1]$ if n is odd. Hence, the corollary follows by:

$$\begin{aligned} D_{\mathcal{M}_{\text{TDRL}^\circ}}(u\mathcal{P}_n^\circ) &= \begin{cases} \lceil \log_2((n-1)/2) \rceil & \text{if } n \text{ is odd,} \\ \lceil \log_2(n/2) \rceil & \text{if } n \text{ is even} \end{cases} \\ &= \lceil \log_2 \lfloor n/2 \rfloor \rceil. \end{aligned}$$

□

This section is concluded with Example 3.10 that illustrates how the algorithm from Bernt et al. (2011) can be used to compute a parsimonious scenario for directed and undirected circular permutations.

Example 3.10. *Consider the linear permutation* $\pi = (5\ 2\ 3\ 4\ 1)$. *Permutation* π *has the three chains* $c_1 = (1)$, $c_2 = (2,3,4)$, *and* $c_3 = (5)$, *hence* $\vartheta(\pi) = 3$ *and* $d_{\mathcal{M}_{TDRL}}(\pi, \iota) = 2$ *(Chaudhuri et al., 2006). The sequence* $S = (\rho_1(\{1,5\},\{2,3,4\}), \rho_2(\{1,2,3,4\},\{5\}))$ *is a parsimonious scenario for* π *and* ι *under* $\mathcal{M}_{TDRL}$ *in which scenario* S *is obtained as described in Section 2.3.4. Now consider* π *to be circular and directed, i.e., the circular permutation* $\pi^\circ = [\pi]_\sim$ *is considered. By Proposition 3.3 follows that* $\vartheta^\circ(\pi^\circ) = 2$, *hence only one TDRL*$^\circ$ *is needed to transform* π° *into* ι° *as* $d_{\mathcal{M}_{TDRL^\circ}}(\pi^\circ, \iota^\circ) = 1$ *(Theorem 3.2). The scenario* $S' = (\rho^\circ(\{1,2,3,4\},\{5\},4))$ *is one of the parsimonious scenarios for* π° *and* ι°. *If both reading directions of* π° *are of interest, then the undirected circular permutation* $\varpi^\circ = \pi^\circ \cup \overline{\pi}^\circ$ *is considered. By Proposition 3.5 it holds that* $\vartheta^\circ(\pi^\circ) + \vartheta^\circ(\overline{\pi}^\circ) = 5$. *Since* $\vartheta^\circ(\pi^\circ) = 2$ *it follows that* $\vartheta^\circ(\overline{\pi}^\circ) = 3$. *Theorem 3.3 implies that* $d_{\mathcal{M}_{TDRL^\circ}}(\varpi^\circ, \zeta^\circ) = \min\{1,2\} = 1$. *Therefore, a parsimonious scenario for the undirected circular permutations* ϖ° *and* ζ° *can be obtained from a parsimonious scenario for the directed circular permutations* π° *and* ι°. *A corresponding parsimonious scenario for* ϖ° *and* ζ° *that is obtained from* S' *is the scenario* $S'' = (\rho^\circ(\{1,2,3,4\},\{5\},(1\ 5\ 2\ 3\ 4)))$.

This section showed that considering undirected circular permutations, instead of directed circular permutations, can reduce the corresponding $\mathcal{M}_{\text{TDRL}^\circ}$ distance. However, this result can be neglected for practical applications in the case that only TDRL$^\circ$ rearrangements are considered by the following reasoning: The gene orientation of mitochondrial gene orders is commonly known, hence signed circular permutations are used to represent those gene orders. Since circular TDRLs cannot toggle the sign of an element of a permutation, the sign of every element in both considered permutations has to be identical in order to ensure the existence of a parsimonious scenario. However, changing the reading direction of one given circular permutation results in a circular permutation in which the sign of every element is toggled, hence the elements of the considered permutations have different signs and cannot be transformed into each other by TDRL rearrangements alone. For this reason, the following sections consider only directed circular permutations.

3.2 CONSEQUENCES FOR BIOLOGICAL APPLICATIONS

The results that were obtained in the Section 3.1 imply several practical consequences for biological applications:

The $\mathcal{M}_{\text{TDRL}^\circ}$ distance between two circular permutations is either less by one or equal to the $\mathcal{M}_{\text{TDRL}}$ distance of the corresponding linear representatives. Hence, using the $\mathcal{M}_{\text{TDRL}}$ distance for an unfavorable choice of representatives may lead to an overestimation of the rearrangement distance. However, Corollary 3.7 implies that the $\mathcal{M}_{\text{TDRL}}$ distance and the $\mathcal{M}_{\text{TDRL}^\circ}$ distance coincide if the $\mathcal{M}_{\text{TDRL}}$ distance is computed for a representative starting (or ending) with the same element as the target permutation. In the data bases that provide mitochondrial annotations, e. g., NCBI RefSeq (Pruitt et al., 2007), MitoZoa (Meo et al., 2012), and others, this property is not implemented and therefore might lead to an overestimation of the $\mathcal{M}_{\text{TDRL}}$ distance. It is worth mentioning that the data base MitoFish (Iwasaki et al., 2013) considers this property as all mitochondrial genomes in the data base start with gene *trnF*. Furthermore, in previous works on TDRLs (e. g., Chaudhuri et al. (2006) and Bernt et al. (2011)) the circular mitochondrial genome was not explicitly treated as a circular permutation but rather the representative which is contained in a data base was used. Therefore, in Section 3.2.1 the effect of neglecting the circularity of the mitochondrial genomes for the computation of the $\mathcal{M}_{\text{TDRL}}$ distance is investigated from both the theoretical as well as the empirical aspect: First, a formula for the probability to overestimate the $\mathcal{M}_{\text{TDRL}}$ distance due to an unfavorable choice of representatives is given. Second, the corresponding probability for this error is measured for the representatives of metazoan mitochondrial gene orders given in the NCBI RefSeq data base.

For every origin of a TDRL° there exist at least two different TDRL° rearrangements that result in the same circular gene order. Accordingly, for studying the predictions on the molecular mechanisms of TDRLs, e. g., the gene loss pattern and the gene intervals that are duplicated, the whole set of equivalent TDRL°s must be considered. The reason is that the set of equivalent TDRLs exhibits a variety of different loss patterns, allowing for different interpretations. Based on these results an evaluation of the tandem duplication non-random loss model (Lavrov et al., 2002) is performed in Section 3.2.2. The evaluation is based on a detailed analysis of two pairs of gene orders that have been used in the literature to argue for the tandem duplication non-random loss model.

3.2.1 *Rearrangement Distance Differences*

In this section, the probability of obtaining an overestimation of the $\mathcal{M}_{\text{TDRL}}$ distance due to an unfavorable choice of linear representatives of circular permutations is determined. To deduce this probability in Theorem 3.4, the Proposition 3.7 is needed. It is worth mentioning that the proof of Proposition 3.7 is done analogously to the proof of

Theorem 1.11 in Bóna (2004), which shows that the number of permutations of size n with r maximal increasing substrings is given by the Eulerian number $\left\langle {n \atop r} \right\rangle = A(n,r) = \sum_{i=0}^{r}(-1)^i\binom{n+1}{i}(r-i)^n$. (Recall that a *maximal increasing substring* of a permutation π is a maximal sequence $\pi(i_1)\dots\pi(i_m)$ of consecutive elements in π such that $\pi(i_j) < \pi(i_{j+1})$ for all $j \in [1:m-1]$.)

Proposition 3.7. *The number of permutations $\pi \in \mathcal{P}_n$ with $n \geqslant 2$ that have r maximal increasing substrings, element ℓ at the first position, and element k at the last position with $k < \ell$ is*

$$\sum_{i=0}^{r}(-1)^i\binom{n+1}{i}\sum_{L=0}^{r-i-1}\sum_{K=L+1}^{r-i}(K-L-1)^{\ell-k-1}(K-L)^{n-\ell+k-1}.$$

Proof. Consider x *compartments* with $x-1$ *bars* in between. Consider the assignment of n elements to the x compartments where the elements in the compartments are sorted in increasing order such that i) k is the largest element in the last non-empty compartment and ii) ℓ is the smallest element in the first non-empty compartment. Such an arrangement is called *valid*. Let $L, K \in [0:x-1]$ be the compartment containing ℓ and k, respectively. The elements can only be assigned to compartments in $[L:K]$. Furthermore, $L < K$ must hold since k and ℓ cannot be in the same compartment because $k < \ell$. Elements smaller than ℓ must not be assigned to compartment L and elements that are larger than k must not be assigned to compartment K. Because otherwise either ℓ is not the first element in L or k is not the last element in K. Thus, the $k-1$ elements that are smaller than k can be assigned to any of the $K-L$ compartments $[L+1:K]$, the $n-\ell$ elements that are larger than ℓ can be assigned to any of the $K-L$ compartments $[L:K-1]$, and the remaining $\ell-k-1$ elements that are smaller than ℓ and larger than k can be assigned to any of the $K-L-1$ compartments $[L+1:K-1]$. Hence, the number of valid arrangements with x bars is given by $\sum_{L=0}^{x}\sum_{K=L+1}^{x+1}(K-L-1)^{\ell-k-1}(K-L)^{n-\ell+k-1}$.

A bar is called *extraneous* if it is not immediately followed by another bar and if its removal results in a valid arrangement, i. e., where the elements of the remaining compartments are sorted. This is, bars to the right of empty compartments and bars that separate an increasing pair of elements are extraneous. Note that the number of valid arrangements for r compartments where no bar is extraneous is equal to $N_2(n,r,k,\ell)$. This number can be determined by counting the number of valid arrangements and removing those that have extraneous bars.

The $n+1$ *positions* of a permutation of size n are the spaces between consecutive elements of a permutation as well as the space preceding the first and following the last element.

For a set $S \subseteq [0:n]$ let $\mathcal{A}_S$ be the set of valid arrangements in r compartments where each position in S has an extraneous bar, i. e., for each arrangement in $\mathcal{A}_S$ it holds that the positions with extraneous bars are a superset of S. For $i = |S| \leqslant r-1$ it holds that $|\mathcal{A}_S| = \sum_{L=0}^{r-1-i}\sum_{K=L+1}^{r-i}(K-L-1)^{\ell-k-1}(K-L)^{n-\ell+k-1}$ because

such an arrangement can be obtained by inserting i bars at the positions that are in S into any valid arrangement with $r-1-i$ bars. There are $\binom{n+1}{i}$ choices for the elements of S.

Note that $\mathcal{A}_S$ is equal to the set of valid arrangements where the positions that have no extraneous bars is a subset of $[n+1] \setminus S$. Hence, $\mathcal{A}_\emptyset$ is the set of arrangements for which the set of positions without an extraneous bar is a subset of the $n+1$ positions. This set contains the sought set where none of the positions harbors an extraneous bar, but also other configurations which do have extraneous bars. Consequently, the size of the sought set can be determined using the Inclusion-Exclusion Principle (similarly to what is done in Bóna, 2004 Chapter 1.1) completing the proof. □

Given Proposition 3.7, a formula for the probability to overestimate the TDRL distance is given in Theorem 3.4.

Theorem 3.4. *Let π° be chosen uniformly at random from $\mathcal{P}_n^\circ$ with $n \geqslant 2$ and let $\pi \in \pi^\circ$ be a uniformly at random chosen representative of π°. The probability that $d_{\mathcal{M}_{TDRL^\circ}}(\pi^\circ, \iota^\circ) < d_{\mathcal{M}_{TDRL}}(\pi, \iota)$ is satisfied equals*

$$\left(\sum_{d=0}^{\lceil \log_2 n \rceil} \sum_{\substack{k,\ell=1 \\ k<\ell}}^{n} \sum_{i=0}^{2^d+1} (-1)^i \binom{n+1}{i} \sum_{\substack{L,K=0 \\ L<K}}^{2^d-i+1} (K-L-1)^{\ell-k-1} (K-L)^{n-\ell+k-1}\right) / n! \,.$$

Under the additional restriction that $d_{\mathcal{M}_{TDRL^\circ}}(\pi^\circ, \iota^\circ) = d$ the probability for $d_{\mathcal{M}_{TDRL^\circ}}(\pi^\circ, \iota^\circ) < d_{\mathcal{M}_{TDRL}}(\pi, \iota)$ equals

$$\left(\sum_{\substack{k,\ell=1 \\ k<\ell}}^{n} \sum_{i=0}^{2^d+1} (-1)^i \binom{n+1}{i} \sum_{\substack{L,K=0 \\ L<K}}^{2^d-i+1} (K-L-1)^{\ell-k-1} (K-L)^{n-\ell+k-1}\right) \Big/ \sum_{c=2^d+1}^{2^{d+1}} \left\langle {n \atop c} \right\rangle,$$

where $\left\langle {n \atop k} \right\rangle$ is the Eulerian number.

Proof. For the proof the number $N_1(n)$ of permutations π of size n with the following properties needs to be determined: i) $\pi^{-1}(n) < \pi^{-1}(1)$ and ii) $\vartheta(\pi) = 2^d + 1$, with $d \in [0 \colon \lceil \log_2 n \rceil]$. Property (i) guaranties that $N_1(n)$ enumerates only permutations where the number of chains differs from the number of circular chains. Hence, the former number is larger by one (see Proposition 3.2). Property (ii) guaranties that $N_1(n)$ counts only permutations for which this leads to a difference between the $\mathcal{M}_{\text{TDRL}}$ distance and the $\mathcal{M}_{\text{TDRL}^\circ}$ distance. Then

$$N_1(n) = \sum_{d=0}^{\lceil \log_2 n \rceil} \sum_{k=1}^{n-1} \sum_{\ell=k+1}^{n} N_2(n, 2^d+1, k, \ell),$$

where $N_2(n, r, k, \ell)$ gives the number of permutations π of size n that have r chains, element 1 at position ℓ (i. e., $\pi^{-1}(1) = \ell$), and element n at position k (i. e., $\pi^{-1}(n) = k$), with $1 \leqslant k < \ell \leqslant n$.

It is known that there exists a bijection between the maximal increasing substrings of a permutation π and the chains in π^{-1}. More precisely, the bijection is the functional inverse of π, i. e., $\pi(i)\pi(i+1)\ldots\pi(m)$ is a maximal increasing substring of π if and only if $(i, i+1, \ldots, m)$ is a chain of π^{-1} (see Proposition 3.4 in Bernt et al. (2011)).

Hence, $N_2(n, r, k, \ell)$ equals the number of permutations π^{-1} that have r maximal increasing substrings, element ℓ at the first position, and element k at the last position. Thus, by Proposition 3.7 follows:

$$N_2(n, r, k, \ell) = \sum_{i=0}^{r} (-1)^i \binom{n+1}{i} \sum_{L=0}^{r-i-1} \sum_{K=L+1}^{r-i} (K-L-1)^{\ell-k-1} (K-L)^{n-\ell+k-1}.$$

Hence,

$$N_1(n) = \sum_{d=0}^{\lceil \log_2 n \rceil} \sum_{k=1}^{n-1} \sum_{\ell=k+1}^{n} \sum_{i=0}^{2^d+1} (-1)^i \binom{n+1}{i} \sum_{L=0}^{2^d-i} \sum_{K=L+1}^{2^d+1-i} (K-L-1)^{\ell-k-1} (K-L)^{n-\ell+k-1}.$$

Dividing $N_1(n)$ by the number of all permutations of size n (i. e., $n!$) gives the first result. For the second result the number $N_1(n)$ with preassigned distance d has to be divided by the number of permutations that have a distance of d, i. e., by $\sum_{c=2^d+1}^{2^{d+1}} \left\langle {n \atop c} \right\rangle$. □

Theorem 3.4 presents a formula for the calculation of the probability to overestimate the $\mathcal{M}_{\text{TDRL}}$ distance. Figure 3.6 illustrates these probabilities (as stated in the first equation of Theorem 3.4) for all unsigned permutations up to a size of 37. The computations were done using the `GMP` software library (Granlund and the GMP development team, 2012). Figure 3.6 shows that the probabilities have maxima for permutation sizes n that are a power of two. Except for small sizes, virtually only permutations with a $\mathcal{M}_{\text{TDRL}^\circ}$ distance of $d = \lceil \log_2 n \rceil - 1$ contribute to the probability mass at these maxima. Towards the next larger permutation size that is a power of two, first the probabilities decrease and then increase again until the next maximum. The height of the maxima decreases with increasing size.

In order to find out if there exists a discrepancy between the theoretically computed probabilities presented in Figure 3.6 and the corresponding probabilities on mitochondrial gene order data, an experiment on metazoan mitochondrial gene orders is performed in the following paragraphs. The aim of the experiment is to investigate and evaluate this *reality gap* on the probability to derive an overestimated $\mathcal{M}_{\text{TDRL}}$ distance.

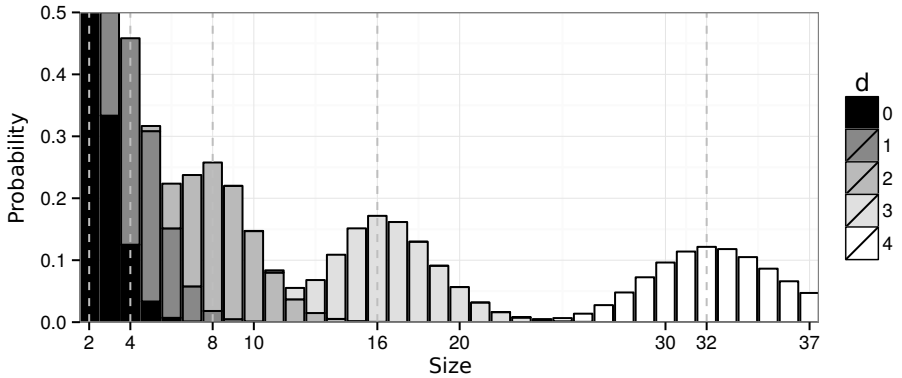


Figure 3.6: Probability that a uniformly chosen unsigned permutation π with size n and $\mathcal{M}_{\text{TDRL}^\circ}$ distance d has a $\mathcal{M}_{\text{TDRL}}$ distance of $d+1$. For clarity, values for $d \geqslant 5$ are omitted since the corresponding probabilities are smaller than 10^{-18} and therefore not visible in the figure.

Experiment

The NCBI RefSeq (Pruitt et al., 2007) contains mitochondrial genome sequences and their annotation. The experiments are based on the data of RefSeq release 89 which contains the data of more than 10^4 complete mitochondrial genomes. The NCBI RefSeq is the most up-to-date source for mitochondrial genomes and their annotation. However, the annotations that are presented contain a large number of errors such as genes which are annotated on the reverse complement strand or named incorrectly, e. g., see Boore et al. (2005) and Perseke et al. (2008). Since these errors may mislead the analyses performed in this thesis, a two step procedure has been used in order to obtain an improved data set. Firstly, all mitochondrial genomes of the metazoan species contained in the NCBI RefSeq release 89 have been re-annotated with an improved version[1] of `MITOS` (Bernt et al., 2013c). Note that `MITOS` is the standard tool for gene annotation of mitochondria. Secondly, only the 4900 mitochondrial genomes that contain the standard metazoan set of 37 genes were included in the data set. This data set contains 611 unique linear representative gene orders. For analyzing the $\mathcal{M}_{\text{TDRL}}$ distance only pairs of genomes can be considered where all genes have the same orientation. Altogether 14752 genome pairs satisfy this condition.

For all 14752 pairs of mitochondrial gene orders (π, σ) from the NCBI RefSeq it is checked if the $\mathcal{M}_{\text{TDRL}}$ distance for the representatives that are given in the data base is different from the $\mathcal{M}_{\text{TDRL}^\circ}$ distance. For the experiment, only pairs of gene orders have been considered where the $\mathcal{M}_{\text{TDRL}^\circ}$ distance is at most four. One reason for this is that a distance of four is close to the maximum $\mathcal{M}_{\text{TDRL}}$ distance for mitochondrial gene orders which is $\lceil \log_2 37 \rceil = 6$ by Chaudhuri et al. (2006). It is also close to the number of random TDRLs which are necessary to obtain a random gene order from a given gene order (Aldous and Diaconis, 1986). Thus, all pairs of

1 unpublished http://mitos2.bioinf.uni-leipzig.de

Table 3.2: Number of pairs of unique metazoan mitochondrial gene orders in the NCBI RefSeq that satisfy $d_{\mathcal{M}_{TDRL}}(\pi,\sigma) > d_{\mathcal{M}_{TDRL^\circ}}([\pi]_\sim,[\sigma]_\sim) = k$ with $k \leqslant 4$. Shown are the absolute and the relative frequency of the number of pairs.

k	1	2	3	4	Σ
absolute	1285	1310	188	11	2794
relative	0.09	0.09	0.01	$7 \cdot 10^{-4}$	0.19

mitochondrial gene orders (π,σ) have been considered for which $d_{\mathcal{M}_{TDRL^\circ}}([\pi]_\sim,[\sigma]_\sim) < d_{\mathcal{M}_{TDRL}}(\pi,\sigma)$ and $d_{\mathcal{M}_{TDRL^\circ}}([\pi]_\sim,[\sigma]_\sim) \leqslant 4$ holds.

Results

The empirical probabilities for the pairs of mitochondrial gene orders (π,σ) from the NCBI RefSeq for which it holds that the $\mathcal{M}_{TDRL}$ distance for the representatives that are given in the data base is different from the $\mathcal{M}_{TDRL^\circ}$ distance are specified in Table 3.2 for all $\mathcal{M}_{TDRL^\circ}$ distances $k \in [1:4]$.

Discussion

Some of the obtained pairs are well-known in the biological literature. For instance the standard chordate gene order, e. g., *Petromyzon marinus* (NC_001626), and the deep see gulper eel *Saccopharynx lavenbergi* (NC_005298) have been reported to differ by one large-scale TDRL genome rearrangement (Inoue et al., 2003). In this study the authors reported the biologically meaningful minimum $\mathcal{M}_{TDRL^\circ}$ distance which is different from the $\mathcal{M}_{TDRL}$ distance of the representatives given in the RefSeq. Another interesting example are the gene orders of unionoid bivalves *Lampsilis ornata* (NC_005335) and *Solenaia oleivora* (NC_022701). The mitogenome of *S. oleivora* was presented in Huang et al. (2013) without an analysis of its gene order. Since the scope of this section is mere one the number of obtained genome pairs rather than the genome pairs itself a specific listing of all obtained pairs is not presented in this thesis.

The percentage of gene order pairs with a difference between the $\mathcal{M}_{TDRL}$ distance and the $\mathcal{M}_{TDRL^\circ}$ distance is less than 10% for all $\mathcal{M}_{TDRL^\circ}$ distances $k \in [1:4]$. The fractions of gene order pairs with a difference between the $\mathcal{M}_{TDRL}$ distance and the $\mathcal{M}_{TDRL^\circ}$ distance are much higher than the theoretically computed probabilities with the exception of the distance for $k = 4$. The theoretically computed probabilities that have been computed according to Theorem 3.4 are $1.75 \cdot 10^{-31}$, $1.13 \cdot 10^{-20}$, $4.91 \cdot 10^{-10}$, and 0.047 for permutation size 37 and the $\mathcal{M}_{TDRL^\circ}$ distances 1, 2, 3, and 4, respectively. The respective percentages for the mitochondrial data are 0.09, 0.09, 0.01, and $7 \cdot 10^{-4}$, see Table 3.2.

One of the reasons for the differences is that the 611 unique gene orders in the RefSeq are not random, but from phylogenetically related species. Another reason is that for most mitochondrial genomes in the

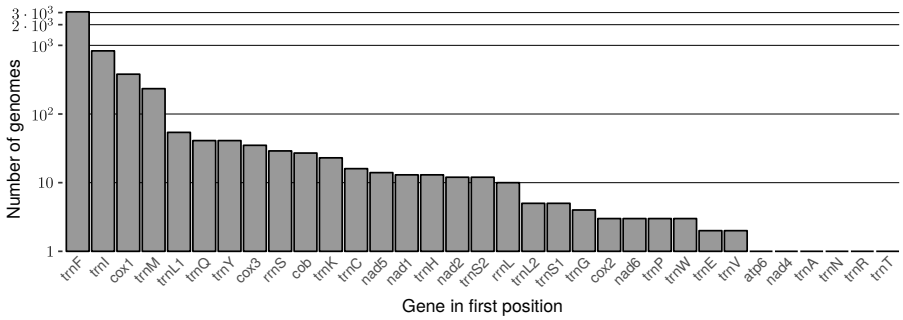


Figure 3.7: Number of mitochondrial gene orders with a specific gene at the first position in the annotation given by RefSeq release 89.

RefSeq the same gene is at the first position, see Figure 3.7. Therefore, it holds that the $\mathcal{M}_{\text{TDRL}}$ distance and the $\mathcal{M}_{\text{TDRL}^\circ}$ distance coincide which is implied by Corollary 3.7. In the data set, the most frequent first position is gene *trnF*. This gene is at the first position in 3079 of the considered 4900 mitochondrial genomes covering nearly all *Chordata* mitogenomes in the data base. Genes *trnI* and *trnM* are frequently chosen as the first position for the *Arthropoda* mitogenomes. The fourth gene that appears in more than 100 cases in the first position is *cox1* which is frequently the case in arthropod and mollusc mitogenomes. In most of these cases the gene chosen for the first position is adjacent to the large non-coding region of the mitogenome. This region is called the control region in chordates and the A+T rich region in invertebrates. This choice of the starting gene might just be a habit or may be selected because the control region contains many repetitions and is therefore difficult to sequence. In any case, this choice gives the RefSeq database implicitly some robustness against inaccuracies in the computation of $\mathcal{M}_{\text{TDRL}^\circ}$ distances. However, the fact that almost 20% of the compared gene order pairs have an overestimated $\mathcal{M}_{\text{TDRL}}$ distance points out that the circularity of the mitochondrial gene orders should be considered.

3.2.2 *Evaluation of the Tandem Duplication Non-Random Loss Model*

Theorem 3.1 showed that it is not always possible to uniquely reconstruct a TDRL only from the knowledge of the two circular gene orders before and after the application of the TDRL since there exist several TDRLs that can explain the change from one circular genome to another. More precisely, at least two different loss patterns exists for each representative (Corollary 3.5). This implies that it is not sufficient to consider the loss pattern of a single representative, even if its TDRL distance is minimum. This is particularly important if predictions are based on the loss pattern of the single representative.

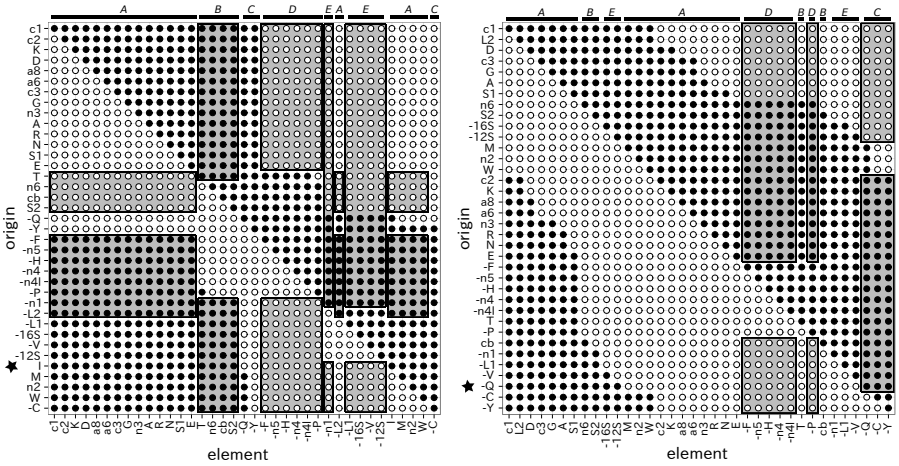
In the analyses of the TDRLs from *Limulus polyphemus* to *Narceus annularus* (Lavrov et al., 2002) and *Trichocera bimaculata* (with the same circular gene order as *Drosophila melanogaster*) to *Paracladura trichoptera* (Beckenbach, 2011) it has been suggested that the loss of the genes is non-random, but *specific* for gene orientation or transcript structure, i. e., genes with the same orientation or genes belonging to

the same transcript are lost jointly. For the TDRL that generated the *N. annularus* gene order from the putative panarthropod gene order (represented by *L. polyphemus*) the following scenario was suggested in Lavrov et al. (2002), see also Figure 3.8 (a). The complete genome was duplicated starting from the large non-coding region between *rrnS* and *trnI* which supposedly contains the regulatory signals. The subsequent loss of redundant genes follows the orientation of the genes, i. e., genes on the so called major coding strand are lost in one copy and those on the minor coding strand are lost in the other copy. The only exception to this rule is *trnC*. Under the assumption that two unidirectional transcript units exist that are controlled by signals in the control region (as in the rat mitogenomes where its three transcript promoters are located in the control region (Lavrov et al., 2002; Bernt et al., 2013b)), it has been suggested that the non-random loss is caused by the loss of the transcript promoters. For the rearrangement of the *T. bimaculata* gene order to the one of *P. trichoptera* a whole genome duplication followed by non-random loss of the genes of two out of five transcript units and the random loss of the remaining genes has been suggested (Beckenbach, 2011), see also Figure 3.8 (b). In contrast to the study of Beckenbach (2011) the finer grained transcript structure of *D. melanogaster* (which might be more relevant than the one of the rat used in Lavrov et al. (2002)), which consists of five parts, has been assumed.

In both papers, Lavrov et al. (2002) and Beckenbach (2011), only one possible TDRL was considered by the authors. In the following paragraphs the differences of the gene orders are discussed in consideration of all equivalent circular TDRLs. In both gene orders a single tRNA position has been corrected for an additional transposition of a tRNA that has been suggested in the literature. Therefore, for *N. annularus trnT* has been assumed at its putative position after the TDRL, i. e., before the transposition (as in Figure 2.B.2 of Lavrov et al. (2002)). For *P. trichoptera* the *trnI* has been omitted since no suggestion on the order of the TDRL and the transposition of *trnI* has been made in the literature. Figure 3.8 shows the equivalent circular TDRLs for the two comparisons.

The difference of the *P. trichoptera* mitochondrial genome with respect to the *T. bimaculata* gene order can be explained by 24 circular TDRLs (TDRLs with switched L and R are not counted separately here and in the following) where the genes of both transcript units *C* and *D* are lost in the same copy as in Beckenbach (2011), but also by 12 circular TDRLs where only the genes of one transcript unit are lost jointly in the same copy. In agreement with the prediction of Beckenbach (2011) for the linear gene orders, the loss of the genes is not specific for the genes in the transcript units *A*, *B*, and *E* in any of the equivalent TDRL°. For none of the circular TDRLs the loss is specific to gene orientation. However, if the two rRNAs are ignored there are 15 TDRLs with strand specific loss.

For the *Narceus* gene order in addition to the circular TDRL with duplication origin *trnI* (as in Lavrov et al. (2002)) also for the duplication origin *trnQ* the gene loss is – with the exception of *trnC* – specific



(a) N. annularus (b) P. trichoptera

Figure 3.8: The equivalent circular TDRLs for the putative panarthropod gene order to *N. annularus* (a) and for the *T. bimaculata* gene order to *P. trichoptera* (b) generated by `EqualTDRL` (Hartmann et al., 2018c). The five primary transcripts *A, B, C, D, E* according to Beckenbach (2011) are shown on the top. Boxes highlight transcript units that are deleted completely in one of the copies. Stars mark the duplication origins assumed in Beckenbach (2011) and Lavrov et al. (2002), respectively. Gene abbreviations are: one capital letter indicating the amino acid for the tRNAs, a: *atp*, n: *nad*, cb: *cob*, c: *cox*, and 12S and 16S for the small and large ribosomal subunit.

for the gene orientation. In all other TDRLs the loss is not specific to the gene orientation. There is also some similarity between the set of genes of the loss pattern and the set of genes of the mitochondrial transcript units of the fruit fly: A single circular TDRL that generates the *Narceus* gene order has the property that the genes of the transcript units *A*, *B*, *D*, *E* are lost jointly in one of the copies. Furthermore, the genes of three (two, and one, respectively) transcript units are lost in conjunction in 21 (13, and 2, respectively) of the equivalent circular TDRLs. Transcript unit *C* is lost differentially in all circular TDRLs, but when *trnC* is ignored (as in Lavrov et al. (2002)) then all but six of the circular TDRLs have a joint loss pattern. The special status of *trnC* in both examples is a remarkable coincidence. The genes belonging to the fruit fly transcript unit *A* are scattered throughout the *Narceus* mitogenome. In particular, the *trnL2* gene interrupts the transcript unit *E*. Since *trnL2* is known to be inverse transposed in Insects and *Crustacea* with respect to the remaining *Arthropoda* (Boore et al., 1998) this might indicate some differences between the transcription units of *Narceus* and the fruit fly.

Altogether, the scenarios presented in Lavrov et al. (2002) and Beckenbach (2011) overlap with the results presented in this section but due to the absence of any duplication remnants there is no evidence indicating which of the equivalent circular TDRL events actually took place. Hence, these results weaken the empirical evidence for the otherwise perfectly reasonable tandem duplication non-random loss

model that has been suggested in Lavrov et al., 2002; Beckenbach, 2011.

3.3 CONCLUSION

The tandem duplication random loss (TDRL) is an important rearrangement operation especially in mitochondrial gene orders. In this chapter combinatorial properties of the TDRL rearrangement model have been studied on circular permutations that represent circular gene orders. Thereby, previous works on the TDRL rearrangement have been significantly extended. In particular, two fundamental genome rearrangement problems have been studied on both, directed and undirected unsigned circular permutations. These are the sorting problem and the distance problem. Therefore, the notion of chains of a permutation that is crucial for studying TDRLs has been extended to circular permutations. It has been shown that the number of circular chains of a circular permutation equals the minimum number of chains of its representatives. This result connects the presented theory to the existing theory of TDRLs on linear permutations. The set of equivalent circular TDRLs (TDRL$^\circ$s) and its construction has been studied. It has been demonstrated that for every origin of a TDRL$^\circ$ there exist at least two different TDRL$^\circ$s that result in the same circular gene order. Based on these results it was shown that the $\mathcal{M}_{\text{TDRL}^\circ}$ distance between two directed circular permutations is either less by one or equal to the $\mathcal{M}_{\text{TDRL}}$ distance of the corresponding linear representatives. Hence, using the $\mathcal{M}_{\text{TDRL}}$ distance for an unfavorable choice of representatives may lead to an overestimation of the rearrangement distance. A formula for computing the probability of this error has been given. In addition, a closed formula for the $\mathcal{M}_{\text{TDRL}^\circ}$ distance between two unsigned undirected circular permutations has been deduced.

The empirical application of the results for metazoan mitochondrial gene orders has shown that the circularity of the genomes should be considered, since otherwise their distance is overestimated for a considerable fraction of the pairwise comparisons. The relevance of the theoretical findings was pointed out by a detailed analysis of two pairs of gene orders that have been used in the literature to argue for the tandem duplication non-random loss model. The analysis highlighted the importance to study the circular case explicitly. This is because the set of equivalent TDRLs exhibits a variety of different loss patterns, allowing for different interpretations as well.

In summary, the results presented in this chapter have the following practical consequences for biological applications:

- For studying the $\mathcal{M}_{\text{TDRL}^\circ}$ distance for gene orders with given gene orientation, it is sufficient to compute the $\mathcal{M}_{\text{TDRL}}$ distance for a pair of representatives of the gene orders that start with the same element. If otherwise the transcription direction of the genes is not given, then it is necessary to compute also the $\mathcal{M}_{\text{TDRL}^\circ}$ distance for the representative that can be obtained by

reversing the reading direction of the start gene order and shifting its gene order such that it starts with the same gene as the target gene order.

- For studying the potential loss pattern (e. g., random or non-random) the whole set of equivalent TDRL°s must be considered. This set can be obtained by the software tool `EqualTDRL` (Hartmann et al., 2018c) which computes and illustrates all equivalent TDRLs.

4

INVERSE TANDEM DUPLICATION RANDOM LOSSES ON LINEAR PERMUTATIONS

Gene order evolution of unichromosomal genomes, for example mitochondrial genomes, has been modeled mostly by four major types of genome rearrangements: inversions, transpositions, inverse transpositions, and tandem duplication random losses. Combined gene order rearrangement models that consider all those types of rearrangements while admitting computational tractability are rare, see Section 2.3. In particular, models that include transpositions motivate typically hard problems where known exact algorithms have an exponential worst case runtime. Therefore, Yancopoulos et al. (2005) and Bergeron et al. (2006) suggested the double cut and join genome rearrangement (DCJ) which cuts a (potentially multichromosomal) gene order at two different positions and rejoins the resulting fragments. The DCJ model has the advantage that it considers all four major types of unichromosomal genome rearrangements (and also other rearrangements that are common in multichromosomal gene orders) while simplifying the computational complexity for both the sorting problem and the distance problem. Furthermore, the DCJ rearrangement model allows the coexistence of multiple chromosomes, which may be linear or circular in the genomes. While such genome structures may be acceptable for some nuclear genomes, e.g., in *Bacteria* (Badrinarayanan et al., 2015), they are considered to be aberrant for metazoan mitochondrial genomes, since the usual metazoan mitochondrial genome is organized in a single circular chromosome (Bernt et al., 2013b). Consequently, there is a need for a tractable genome rearrangement model that includes all major types of rearrangements of mitochondria and excludes the coexistence of multiple chromosomes.

In this chapter such a rearrangement model, namely the inverse tandem duplication random loss (iTDRL) model, is suggested and initially studied on signed linear permutations. This rearrangement duplicates and inverts a segment of continuous genes of a gene order followed by the random loss of one of the redundant copies of each gene. The iTDRL rearrangement provides the advantage that it can mimic all major mitochondrial rearrangements: 1) an inversion and an inverse transposition can each be represented by at most two iTDRLs and 2) a transposition and a TDRL can each be represented by two iTDRLs. The iTDRL rearrangement has currently been proposed by several authors who suggest it to be a possible mechanism of mitochondrial gene order evolution. In particular, evidence for an iTDRL as evolutionary mechanism has been found in mitochondrial gene order comparisons on the walking stick *Ramulus hainanense* (Jühling et al., 2011), the tongue sole *Cynoglossus semilaevis* (Kong et al., 2009), and the flatfish *Crossorhombus azureus* (Shi et al., 2013). The iTDRL re-

arrangement is also motivated by the fact that inverted duplications often occur in the control region of *Insecta* mitochondrial genomes (Xiaochen et al., 2017). In this chapter, the algorithmic study of this model of genome rearrangement is initiated on linear signed permutations. In particular, the significant contributions of this chapter are:

- The distance problem for two signed linear permutations under the iTDRL model can be solved in linear time with respect to the size of the considered permutations (Corollary 4.2).
- A parsimonious sorting scenario for two signed linear permutations under the iTDRL model can be obtained in quasilinear time with respect to the size of the considered permutations (Algorithm 1).
- The $\mathcal{M}_{\text{iTDRL}}$ distance provides bounds on the minimum number of inversions, transpositions, inverse transpositions, and TDRLs that are necessary to transform one given signed permutation into another signed permutation (Corollary 4.2).

The chapter is organized as follows. Section 4.1 investigates the sorting problem and the distance problem for signed linear permutations under the iTDRL model. Section 4.2 shows that the distance problem under the iTDRL model provides bounds on the distance problem under all four major mitochondrial rearrangements. A conclusion is given in Section 4.3.

4.1 SOLVING THE DISTANCE AND THE SORTING PROBLEM

In the following, the distance and the sorting problem for unsigned linear permutations under the iTDRL model is investigated. Therefore, this section is divided into three parts: Section 4.1.1 recalls basic definitions and notations that are used throughout this chapter. The number of iTDRLs that are necessary and sufficient to obtain a permutation having a specific number of maximum increasing substrings from the identity permutation is investigated in Section 4.1.2. Computational results for the distance problem and the sorting problem under $\mathcal{M}_{\text{iTDRL}}$ are given in Section 4.1.3. This section also presents a quasilinear time algorithm for solving the sorting problem for arbitrary signed linear permutations.

4.1.1 *Basic Definitions and Preliminaries*

For the convenience of the reader, the formal definitions relevant for determining the $\mathcal{M}_{\text{iTDRL}}$ distance are recalled (for references see Section 2.2) in this section.

A *signed linear permutation* π of size $n \in \mathbb{N}$ is a bijection $\pi\colon [-n\colon n] \setminus \{0\} \to [-n\colon n] \setminus \{0\}$ such that $\pi(-i) = -\pi(i)$ for all $i \in [-n\colon n] \setminus \{0\}$. Observe that a signed permutation $\pi = (\pi(-n) \ \dots \ \pi(-1) \ \pi(1) \ \dots \ \pi(n))$ can always be represented by

$\pi = (\pi(1) \ \dots \ \pi(n))$, as the equation $\pi(-i) = -\pi(i)$ holds by definition. In this section, signed linear permutations are used as a formal model for gene orders in which each element represents a gene and the sign represents its strandedness. When the context is clear a signed permutation is called permutation and the $+$ sign of an element is omitted. The set of all signed permutations of size n is denoted by $s\mathcal{P}_n$. The identity permutation $(1\ 2\ \dots\ n)$ is denoted by ι. For a permutation $\pi = (\pi(1)\dots\pi(n))$ the corresponding permutation where the order and the sign of all elements is reversed is defined as permutation $\overline{\pi}$ with $\overline{\pi}(i) = -\pi(n+1-i)$ for all $i \in [1:n]$. Note that $\overline{\pi}$ is uniquely defined for every $\pi \in s\mathcal{P}_n$. Figure 4.1 illustrates an example of $\overline{\pi}$.

A *subsequence* of $\pi = (\pi(1)\dots\pi(n))$ is a sequence $\pi(i_1)\pi(i_2)\dots\pi(i_k)$ with $1 \leqslant i_1 < i_2 < \dots < i_k \leqslant n$. When all elements in a subsequence S of π appear consecutively, then S is called a *substring* of π. The set of all (unsigned) elements of a subsequence S is denoted by $\mathcal{E}(S)$. The first (last) element of S is denoted by f_S (respectively ℓ_S). A substring $S = \pi(i)\dots\pi(k)$ (of π with $1 \leqslant i \leqslant k \leqslant n$) is called *increasing* if either $i = k$ or $\pi(j) < \pi(j+1)$ for all $j \in [i:k-1]$. An increasing substring is called *maximal* when it cannot be extended into a longer increasing substring. The set of all maximal increasing substrings of a permutation π is denoted by $\mathcal{S}(\pi)$. With $|\mathcal{S}(\pi)|$ the number of maximal increasing substrings of π is denoted. The *maximal increasing substring decomposition* of π is the unique list of pairwise disjoint maximal increasing substrings $\tau_1\tau_2\dots\tau_{|\mathcal{S}(\pi)|}$ of π such that $\pi(1)\dots\pi(n) = \tau_1\tau_2\dots\tau_{|\mathcal{S}(\pi)|}$ and for all $1 \leqslant j \leqslant |\mathcal{S}(\pi)|$ it holds that $|\tau_j| \geqslant 1$. Example 4.1 illustrates the maximal increasing substring decomposition of a permutation.

Example 4.1. *Consider the permutation* $\pi = (1\ 2\ {-3}\ 4\ {-5}\ 6\ 7\ 9\ 8)$. *The set of maximal increasing substrings of* π *is* $\mathcal{S}(\pi) = \{1\,2, -3\,4, -5\,6\,7\,9, 8\}$ *and the maximal increasing substring decomposition of* π *is* $\tau_1\tau_2\tau_3\tau_4$ *with* $\tau_1 = 1\,2$, $\tau_2 = -3\,4$, $\tau_3 = -5\,6\,7\,9$, *and* $\tau_4 = 8$.

A convenient way to work with signed permutations of size n is to represent them as strings over the alphabet $[-n:n] \setminus 0$ of integers. A consequence of this is that a permutation can be represented as a concatenation of (character) disjoint substrings. This also holds for reversing the order and the sign of every element of a substring of a permutation, i.e., if $S = \pi(i_1)\dots\pi(i_k)$ is a substring of π, then $\overline{S}$ is the sequence for which $\overline{\pi}(i_j) = -\pi(i_k + i_1 - i_j)$ holds for all $j \in [1:k]$. Let S_1 and S_2 be two substrings of a permutation π such that $\mathcal{E}(S_1) \cap \mathcal{E}(S_2) = \emptyset$. By $S_1 \oplus S_2$ the sequence is denoted that is created by sorting the elements of S_1 and S_2 increasingly. Figure 4.1 illustrates the given definitions that are related to permutations and substrings.

An *inverse tandem duplication random loss* $\rho\colon s\mathcal{P}_n \to s\mathcal{P}_n$ is a mapping that processes an input π by taking two subsequences L and R (of π) with $\mathcal{E}(L) \cap \mathcal{E}(R) = \emptyset$ and $\mathcal{E}(L) \cup \mathcal{E}(R) = \mathcal{E}(\pi)$ that are outputted as $\rho \circ \pi = L\overline{R}$ or as $\rho \circ \pi = \overline{L}R$. In the first case ρ is called *right iTDRL* (*riTDRL*) and in the second case it is called *left iTDRL*

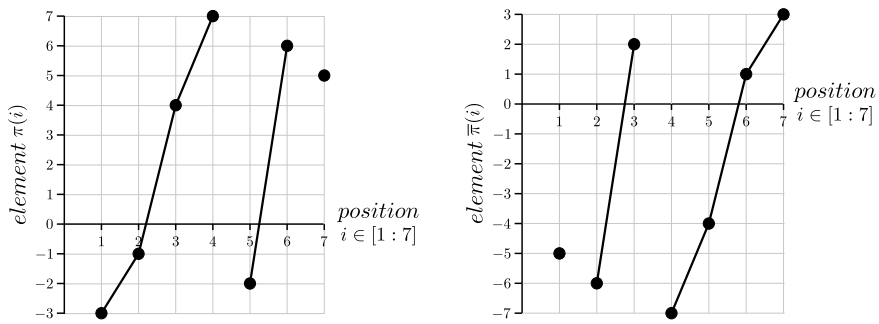


Figure 4.1: Illustration of $\pi = (-3\ -1\ 4\ 7\ -2\ 6\ 5)$ (a) and $\overline{\pi} = (-5\ -6\ 2\ -7\ -4\ 1\ 3)$ (b). Each dot represents an element of the corresponding permutation. Maximal increasing substrings are illustrated by continuous lines. The maximal increasing substring decomposition of π is $\pi = \tau_1\tau_2\tau_3$ with $\tau_1 = -3\,-1\,4\,7$, $\tau_2 = -2\,6$, and $\tau_3 = 5$. Consequently, $\mathcal{S}(\pi) = \{\tau_1, \tau_2, \tau_3\}$ and $|\mathcal{S}(\pi)| = 3$. The substring $-1\,4$ of π is increasing. While the sequence $-5\,2\,-4\,3$ is a subsequence of $\overline{\pi}$, the sequences $S_1 = -6\,2$ and $S_2 = -7\,-4\,1\,3$ are substrings of $\overline{\pi}$. For S_1 and S_2 it holds that $S_1 \oplus S_2 = -7\,-6\,-4\,1\,2\,3$. Intriguingly, $\overline{\pi}$ is point-symmetrically to π.

(*liTDRL*). Such a mapping is denoted as $\rho_{\text{riTDRL}}(r, \mathcal{E}(L), \mathcal{E}(R))$ for an riTDRL and $\rho_{\text{liTDRL}}(\ell, \mathcal{E}(L), \mathcal{E}(R))$ for an liTDRL. If the context is clear an iTDRL $\rho_{\text{riTDRL}}(r, \mathcal{E}(L), \mathcal{E}(R))$ (respectively $\rho_{\text{liTDRL}}(\ell, \mathcal{E}(L), \mathcal{E}(R))$) is written as $\rho(r, \mathcal{E}(L), \mathcal{E}(R))$ (respectively $\rho(\ell, \mathcal{E}(L), \mathcal{E}(R))$) or ρ for the sake of brevity. The set of all iTDRLs is denoted by $\mathcal{M}_{\text{iTDRL}}$. From a biological point of view an iTDRL can be seen as first applying a reversed tandem duplication to π, i. e., $\overline{\pi}$ is placed adjacently to the left (respectively right) of π resulting in a duplicated intermediate $\overline{\pi}\pi$ (respectively $\pi\overline{\pi}$), and to subsequently obtain a new permutation by random loss of one copy of every duplicated element. For an illustration of the iTDRL rearrangement see Section 2.2.3. The composition of two functions f and g is denoted by $f \circ g$, i. e., $(f \circ g)(x) := f(g(x))$.

Recall that the sorting problem of signed permutations under the iTDRL model aims to find a minimum length sequence of iTDRLs that transforms one given permutation into another given permutation. The $\mathcal{M}_{\text{iTDRL}}$ distance, denoted by $d_{\mathcal{M}_{\text{iTDRL}}}(\pi, \sigma)$, between two permutations π and σ is the minimum number of iTDRLs that is necessary to transform π into σ. Hence, it holds that $d_{\mathcal{M}_{\text{iTDRL}}}(\pi, \sigma) := \min\{t \in \mathbb{N}_0 : \rho_t \circ \ldots \circ \rho_1 \circ \pi = \sigma \text{ with } \rho_1, \ldots, \rho_t \in \mathcal{M}_{\text{iTDRL}}\}$. Observe that since $d_{\mathcal{M}_{\text{iTDRL}}}(\pi, \sigma) = d_{\mathcal{M}_{\text{iTDRL}}}(\pi^{-1} \circ \pi, \pi^{-1} \circ \sigma) = d_{\mathcal{M}_{\text{iTDRL}}}(\iota, \pi^{-1} \circ \sigma)$, where π^{-1} is the inverse permutation of π, it is sufficient to compute the $\mathcal{M}_{\text{iTDRL}}$ distance for ι and an arbitrary permutation from $s\mathcal{P}_n$. The following sections follow this notion.

4.1.2 *Structural Characterization of Permutations Generated by Repeated Application of iTDRLs*

In this section, the structure of permutations is characterized that can be generated by sequentially applying $k \in \mathbb{N}$ iTDRLs to the identity permutation ι. The characterization utilizes the number of *maximal increasing substrings* of a permutation. A lower bound on the (minimum) number of iTDRLs that are necessary to produce a permutation with a certain number of maximal increasing substrings is given in Proposition 4.1. A corresponding upper bound is given in Proposition 4.2. From the lower and upper bound the main theorem is derived in Theorem 4.1. The insights gained in this section are an important component in solving the sorting problem (and the distance problem) for signed linear permutations under the iTDRL model in the Section 4.1.3.

The Lower Bound

This subsection provides a lower bound on the number of required iTDRLs to generate a permutation from ι that has a certain number of maximal increasing substrings. For the proof of the lower bound in Proposition 4.1 the following four lemmata are needed.

Lemma 4.1. *Let π be a signed permutation of size n. Then $S \in \mathcal{S}(\pi)$ if and only if $\overline{S} \in \mathcal{S}(\overline{\pi})$.*

Proof. Let $\pi \in s\mathcal{P}_n$ and let $S = \pi(i)\dots\pi(j) \in \mathcal{S}(\pi)$ with $1 \leqslant i \leqslant j \leqslant n$. The fact that $S \in \mathcal{S}(\pi)$ implies that S is a maximal increasing substring of π, i. e., it holds that $\pi(k) < \pi(k+1)$ for all $i \leqslant k < j$. Since S is maximal it cannot be extended to the left or the right, i. e., either $i = 1$ (respectively $j = n$) or $i > 1$ (respectively $j < n$) and $\pi(i-1) > \pi(i)$ (respectively $\pi(j) > \pi(j+1)$). By the definition of $\overline{\pi}$ it holds that $\overline{S} = -\pi(j)\ -\pi(j-1)\ \dots\ -\pi(i)$ is a substring of $\overline{\pi}$. From $\pi(k) < \pi(k+1)$ for all $i \leqslant k < j$ it follows that $-\pi(k+1) < -\pi(k)$, hence $\overline{S}$ is an increasing substring of $\overline{\pi}$. If $i = 1$ and $j = n$ it directly follows that $\overline{S}$ cannot be extended in the respective direction. Hence, consider that $i > 1$ or $j < n$. Consequently, $\pi(i-1) > \pi(i)$ or $\pi(j) > \pi(j+1)$ holds which implies $-\pi(i-1) < -\pi(i)$ or $-\pi(j) < -\pi(j+1)$, respectively. Thus, $\overline{S}$ is a maximal increasing substring of $\overline{\pi}$. Consequently, $S \in \mathcal{S}(\pi)$ implies $\overline{S} \in \mathcal{S}(\overline{\pi})$. The other direction, i. e., if $\overline{S} \in \mathcal{S}(\overline{\pi})$ then $S \in \mathcal{S}(\pi)$, follows from this implication and the fact that $\overline{\overline{\pi}} = \pi$ and $\overline{\overline{S}} = S$. □

Lemma 4.1 shows that a substring S of a permutation π is maximal increasing if and only if its reversed substring $\overline{S}$ is maximal increasing in $\overline{\pi}$. The following corollary is an immediate consequence of Lemma 4.1. It shows that the number of maximal increasing substrings of π and $\overline{\pi}$ coincide.

Corollary 4.1. *For a signed permutation π of size n and its reversed permutation $\overline{\pi}$ it holds that $|\mathcal{S}(\pi)|=|\mathcal{S}(\overline{\pi})|$.*

Proof. By Lemma 4.1 it holds that $S \in \mathcal{S}(\pi)$ if and only if $\overline{S} \in \mathcal{S}(\overline{\pi})$. Hence, the equation $\mathcal{S}(\pi) = \{S_1, \dots, S_{|\mathcal{S}(\pi)|}\}$ holds if and only if $\mathcal{S}(\overline{\pi}) = \{\overline{S_1}, \dots, \overline{S_{|\mathcal{S}(\pi)|}}\}$. Consequently, $|\mathcal{S}(\pi)| = |\mathcal{S}(\overline{\pi})|$. □

Consider a string of integers S and a subsequence S' of S. The following lemma states that the number of maximal increasing substrings of S is always at least the number of maximal increasing substrings of S'.

Lemma 4.2. *Let S be a string of integers with the maximal increasing substring decomposition $S = S_1 \dots S_{|\mathcal{S}(S)|}$ and let S' be a subsequence of S. The following inequality holds:*

$$|\mathcal{S}(S')| \leqslant |\{i \in [1 : |\mathcal{S}(S)|] : \mathcal{E}(S') \cap \mathcal{E}(S_i) \neq \emptyset\}| \leqslant |\mathcal{S}(S)|.$$

Proof. Let $S' = S'_1 \dots S'_{|\mathcal{S}(S')|}$ be the maximal increasing substring decomposition of S'. The fact that every maximal increasing substring of S' contains at least one element, and the fact that S' is a subsequence of S (i. e., $\mathcal{E}(S') \subseteq \mathcal{E}(S)$) ensure that for every maximal increasing substring S'_i of S' with $i \in [1 : |\mathcal{S}(S')|]$ holds $1 = |\mathcal{S}(S'_i)| \leqslant |\{j \in [1 : |\mathcal{S}(S)|] : \mathcal{E}(S'_i) \cap \mathcal{E}(S_j) \neq \emptyset\}|$. Consequently,

$$\begin{aligned} |\mathcal{S}(S')| &= \sum_{S'_i \in \mathcal{S}(S')} |\mathcal{S}(S'_i)| \\ &\leqslant \sum_{S'_i \in \mathcal{S}(S')} |\{j \in [1 : |\mathcal{S}(S)|] : \mathcal{E}(S'_i) \cap \mathcal{E}(S_j) \neq \emptyset\}| \\ &= |\{i \in [1 : |\mathcal{S}(S)|] : \mathcal{E}(S') \cap \mathcal{E}(S_i) \neq \emptyset\}| \end{aligned}$$

gives the left inequality. Since $|\mathcal{S}(S)|$ is the maximum of the set $|\{i \in [1 : |\mathcal{S}(S)|] : \mathcal{E}(S') \cap \mathcal{E}(S_i) \neq \emptyset\}|$ the right inequality follows. □

The following lemma shows that the sum of the number of maximal increasing substrings over a set of signed permutations is always at least the number of maximal increasing substrings of a concatenation of subsequences of the given set of permutations.

Lemma 4.3. *Let $\pi_1, \dots, \pi_k \in s\mathcal{P}_n$ and let S_i be a non-empty subsequence of π_i for all $i \in [1 : k]$ such that $\mathcal{E}(S_i) \cap \mathcal{E}(S_j) = \emptyset$ for all $1 \leqslant i < j \leqslant k$. The following equation holds:*

$$|\mathcal{S}(S_1 S_2 \dots S_k)| = \sum_{i=1}^{k} |\mathcal{S}(S_i)| - K \leqslant \sum_{i=1}^{k} |\mathcal{S}(S_i)| \leqslant \sum_{i=1}^{k} |\mathcal{S}(\pi_i)|,$$

where $K = |\{j \in [1 : k-1] : \ell_{S_j} < f_{S_{j+1}}\}|$.

Proof. By Lemma 4.2 $|\mathcal{S}(S_i)| \leqslant |\mathcal{S}(\pi_i)|$ is obtained for all $i \in [1 : k]$, thus the last two inequalities hold. With $\tau^i_1 \dots \tau^i_{|\mathcal{S}(S_i)|}$ the maximal increasing substring decomposition of S_i is denoted for all $i \in [1 : k]$ and it holds that $\mathcal{S}(S_i) = \{\tau^i_1, \dots, \tau^i_{|\mathcal{S}(S_i)|}\}$. Now observe that all *internal* maximum increasing substrings of a S_i are present in $S_1 S_2 \dots S_k$ as well, i. e., for all S_i it holds that $\tau^i_k \in \mathcal{S}(S_1 S_2 \dots S_k)$ for all $k \in [2 : |\mathcal{S}(S_i)| - 1]$. Furthermore, $\tau^1_1 \in \mathcal{S}(S_1 S_2 \dots S_k)$ and $\tau^k_{|\mathcal{S}(S_k)|} \in \mathcal{S}(S_1 S_2 \dots S_k)$ since

the first and the last maximum increasing substring cannot be extended to the left and the right, respectively. Since two subsequences are pairwise disjoint it holds either $\ell_{S_i} > f_{S_{i+1}}$ or $\ell_{S_i} < f_{S_{i+1}}$ for each $i \in [1:k-1]$. If $\ell_{S_i} > f_{S_{i+1}}$ then $\tau^i_{|\mathcal{S}(S_i)|}$ cannot be extended to the right and τ^{i+1}_1 cannot be extended to the left. Hence, $\tau^i_{|\mathcal{S}(S_i)|}$ and τ^{i+1}_1 are counted separately in $|\mathcal{S}(S_1S_2\dots S_k)|$ as they are in $\sum_{i=1}^k |\mathcal{S}(S_i)| - K$. If $\ell_{S_i} < f_{S_{i+1}}$ then $\tau^i_{|\mathcal{S}(S_i)|}\tau^{i+1}_1$ forms an increasing substring in $S_1S_2\dots S_k$. Consequently, while $\tau^i_{|\mathcal{S}(S_i)|}$ and τ^{i+1}_1 are both counted separately in $\sum_{i=1}^k |\mathcal{S}(S_i)|$ only one string, i. e., $\tau^i_{|\mathcal{S}(S_i)|}\tau^{i+1}_1$, is counted in $|\mathcal{S}(S_1S_2\dots S_k)|$. Observe that this case is counted in K, which (in this case) reduces $\sum_{i=1}^k |\mathcal{S}(S_i)|$ by one. Altogether, the first equation of the lemma follows. □

The next and final lemma states that the application of an iTDRL to a permutation π always results in a permutation that has less than twice the number of maximal increasing substrings of π.

Lemma 4.4. *Let $\pi \in s\mathcal{P}_n$ with $\ell_\pi < 0 < f_\pi$. Then, for every iTDRL $\rho \in \mathcal{M}_{iTDRL}$ it holds that $|\mathcal{S}(\rho\circ\pi)| \leqslant 2|\mathcal{S}(\pi)| - 1$. Furthermore, if $|\mathcal{S}(\rho\circ\pi)| = 2|\mathcal{S}(\pi)| - 1$ then $\ell_{\rho\circ\pi} < 0 < f_{\rho\circ\pi}$.*

Proof. Let $\pi \in s\mathcal{P}_n$ with $\ell_\pi < 0 < f_\pi$ and let $\rho \in \mathcal{M}_{\text{iTDRL}}$. Then $\rho\circ\pi$ can be written as $\rho\circ\pi = \tau\tau'$ (respectively $\rho\circ\pi = \tau'\tau$), where τ is a subsequence of π and τ' is a subsequence of $\overline{\pi}$, if ρ is an riTDRL (respectively liTDRL). Corollary 4.1 implies that $|\mathcal{S}(\pi)| = |\mathcal{S}(\overline{\pi})|$ and by Lemma 4.2 holds that $|\mathcal{S}(\tau)| \leqslant |\mathcal{S}(\pi)|$ and $|\mathcal{S}(\tau')| \leqslant |\mathcal{S}(\overline{\pi})|$. Then, Lemma 4.3 implies $|\mathcal{S}(\tau\tau')| = |\mathcal{S}(\tau)| + |\mathcal{S}(\tau')| - K_1 \leqslant |\mathcal{S}(\tau)| + |\mathcal{S}(\tau')| \leqslant 2|\mathcal{S}(\pi)|$ and $|\mathcal{S}(\tau'\tau)| = |\mathcal{S}(\tau)| + |\mathcal{S}(\tau')| - K_2 \leqslant |\mathcal{S}(\tau)| + |\mathcal{S}(\tau')| \leqslant 2|\mathcal{S}(\pi)|$, where $K_1 = 1$ (respectively $K_2 = 1$) if $\ell_\tau < 0 < f_{\tau'}$ (respectively $\ell_{\tau'} < 0 < f_\tau$) and $K_1 = 0$ (respectively $K_2 = 0$) otherwise. Hence, if $|\mathcal{S}(\tau)| \leqslant |\mathcal{S}(\pi)| - 1$ and $|\mathcal{S}(\tau')| \leqslant |\mathcal{S}(\pi)| - 1$, then $|\mathcal{S}(\tau\tau')| \leqslant 2|\mathcal{S}(\pi)| - 2 < 2|\mathcal{S}(\pi)| - 1$ and $|\mathcal{S}(\tau'\tau)| \leqslant 2|\mathcal{S}(\pi)| - 2 < 2|\mathcal{S}(\pi)| - 1$. Consequently, it remains to consider the cases where $|\mathcal{S}(\tau)|, |\mathcal{S}(\tau')| \geqslant |\mathcal{S}(\pi)| - 1$ and at least one of $|\mathcal{S}(\tau)|$ or $|\mathcal{S}(\tau')|$ is $|\mathcal{S}(\pi)|$. More precisely, the following cases remain to be considered: i) $|\mathcal{S}(\tau)| = |\mathcal{S}(\tau')| = |\mathcal{S}(\pi)|$, ii) $|\mathcal{S}(\tau)| = |\mathcal{S}(\pi)| - 1$ and $|\mathcal{S}(\tau')| = |\mathcal{S}(\pi)|$, and iii) $|\mathcal{S}(\tau)| = |\mathcal{S}(\pi)|$ and $|\mathcal{S}(\tau')| = |\mathcal{S}(\pi)| - 1$. Let $\pi = \pi_1 \dots \pi_{|\mathcal{S}(\pi)|}$ be the maximal increasing substring decomposition of π. Then $0 < f_\pi$ (respectively $\ell_\pi < 0$) implies that π_1 (respectively $\pi_{|\mathcal{S}(\pi)|}$) contains only positive (respectively negative) elements. Hence, Lemma 4.1 implies that $\overline{\pi}_1$ (respectively $\overline{\pi}_{|\mathcal{S}(\pi)|}$) contains only positive (respectively negative) elements, where $\overline{\pi} = \overline{\pi}_1 \dots \overline{\pi}_{|\mathcal{S}(\pi)|}$ is the maximal increasing substring decomposition of $\overline{\pi}$. In the following, the statement is proven in the cases (i) and (ii). The proof for Case (iii) is similar to Case (ii).

(i): By Lemma 4.2 it holds that $\mathcal{E}(\tau) \cap \mathcal{E}(\pi_i) \neq \emptyset$ and $\mathcal{E}(\tau') \cap \mathcal{E}(\overline{\pi_i}) \neq \emptyset$ for all $i \in [1:|\mathcal{S}(\pi)|]$. Hence, since $\mathcal{E}(\tau) \cap \mathcal{E}(\pi_1) \neq \emptyset$ (respectively $\mathcal{E}(\tau) \cap \mathcal{E}(\pi_{|\mathcal{S}(\pi)|}) \neq \emptyset$) it holds that $f_\tau > 0$ (respectively $\ell_\tau < 0$). Analogously, $\mathcal{E}(\tau') \cap \mathcal{E}(\overline{\pi}_1) \neq \emptyset$ (respectively $\mathcal{E}(\tau') \cap \mathcal{E}(\overline{\pi}_{|\mathcal{S}(\pi)|}) \neq \emptyset$) implies that $f_{\tau'} > 0$ (respectively $\ell_{\tau'} < 0$). Hence, $\ell_\tau < 0 < f_{\tau'}$ (respectively $\ell_{\tau'} < 0 < f_\tau$) if ρ is an riTDRL

(respectively liTDRL). Consequently, by Lemma 4.3 it holds that $|\mathcal{S}(\tau\tau')| = |\mathcal{S}(\tau)| + |\mathcal{S}(\tau')| - K_1 = |\mathcal{S}(\pi)| + |\mathcal{S}(\pi)| - 1 = 2|\mathcal{S}(\pi)| - 1$ and $|\mathcal{S}(\tau'\tau)| = |\mathcal{S}(\tau)| + |\mathcal{S}(\tau')| - K_2 = |\mathcal{S}(\pi)| + |\mathcal{S}(\pi)| - 1 = 2|\mathcal{S}(\pi)| - 1$. Hence, $|\mathcal{S}(\rho \circ \pi)| = 2|\mathcal{S}(\pi)| - 1$ and $\ell_{\rho\circ\pi} < 0 < f_{\rho\circ\pi}$.

(ii): By Lemma 4.2 it holds that $\mathcal{E}(\tau') \cap \mathcal{E}(\overline{\pi_i}) \neq \emptyset$ for all $i \in [1 : |\mathcal{S}(\pi)|]$. Hence, since $\mathcal{E}(\tau') \cap \mathcal{E}(\overline{\pi}_1) \neq \emptyset$ (respectively $\mathcal{E}(\tau') \cap \mathcal{E}(\overline{\pi}_{|\mathcal{S}(\pi)|}) \neq \emptyset$) it holds that $f_{\tau'} > 0$ (respectively $\ell_{\tau'} < 0$). By Lemma 4.2 there exists an $i \in [1 : |\mathcal{S}(\pi)|]$ with $\mathcal{E}(\tau) \cap \mathcal{E}(\pi_i) = \emptyset$ and for all $j \in [1 : |\mathcal{S}(\pi)|] \setminus \{i\}$ it holds that $\mathcal{E}(\tau) \cap \mathcal{E}(\pi_i) \neq \emptyset$.

- Consider first that $i = 1$, then $\mathcal{E}(\pi_{|\mathcal{S}(\pi)|}) \cap \mathcal{E}(\tau) \neq \emptyset$, hence $\ell_\tau < 0$. Consequently, $\ell_\tau < 0 < f_{\tau'}$ and by Lemma 4.3 it holds $|\mathcal{S}(\tau\tau')| = |\mathcal{S}(\tau)| + |\mathcal{S}(\tau')| - K_1 = |\mathcal{S}(\pi)| - 1 + |\mathcal{S}(\pi)| - 1 = 2|\mathcal{S}(\pi)| - 2 < 2|\mathcal{S}(\pi)| - 1$. Analogously, Lemma 4.3 yields $|\mathcal{S}(\tau'\tau)| = |\mathcal{S}(\tau')| + |\mathcal{S}(\tau)| - K_2 \leqslant |\mathcal{S}(\pi)| - 1 + |\mathcal{S}(\pi)| = 2|\mathcal{S}(\pi)| - 1$ with $\ell_{\rho\circ\pi} = \ell_{\tau'\tau} = \ell_\tau < 0 < f_{\tau'} = f_{\tau'\tau} = f_{\rho\circ\pi}$.
- Consider now that $i = |\mathcal{S}(\pi)|$. Then, by Lemma 4.2 $\mathcal{E}(\pi_1) \cap \mathcal{E}(\tau) \neq \emptyset$, hence $f_\tau > 0$. Consequently, $\ell_{\tau'} < 0 < f_\tau$ and by applying Lemma 4.3 again $|\mathcal{S}(\tau'\tau)| = |\mathcal{S}(\tau')| + |\mathcal{S}(\tau)| - K_2 = |\mathcal{S}(\pi)| + |\mathcal{S}(\pi)| - 1 - 1 = 2|\mathcal{S}(\pi)| - 2 < 2|\mathcal{S}(\pi)| - 1$ is obtained. Analogously, Lemma 4.3 also provides $|\mathcal{S}(\tau\tau')| = |\mathcal{S}(\tau)| + |\mathcal{S}(\tau')| - K_1 \leqslant |\mathcal{S}(\pi)| - 1 + |\mathcal{S}(\pi)| = 2|\mathcal{S}(\pi)| - 1$ with $\ell_{\rho\circ\pi} = \ell_{\tau\tau'} = \ell_{\tau'} < 0 < f_\tau = f_{\tau\tau'} = f_{\rho\circ\pi}$.
- Finally, consider $i \in [2 : |\mathcal{S}(\pi)| - 1]$. Then, by Lemma 4.2 $\mathcal{E}(\pi_1) \cap \mathcal{E}(\tau) \neq \emptyset$ and $\mathcal{E}(\pi_{|\mathcal{S}(\pi)|}) \cap \mathcal{E}(\tau) \neq \emptyset$, hence $f_\tau > 0$ and $\ell_\tau < 0$ holds. Thus, $\ell_\tau < 0 < f_{\tau'}$ and $\ell_{\tau'} < 0 < f_\tau$ is implied. As before, using Lemma 4.3 yields $|\mathcal{S}(\tau\tau')| = |\mathcal{S}(\tau)| + |\mathcal{S}(\tau')| - K_1 = |\mathcal{S}(\pi)| - 1 + |\mathcal{S}(\pi)| - 1 = 2|\mathcal{S}(\pi)| - 2 < 2|\mathcal{S}(\pi)| - 1$ and $|\mathcal{S}(\tau'\tau)| = |\mathcal{S}(\tau')| + |\mathcal{S}(\tau)| - K_2 = |\mathcal{S}(\pi)| + |\mathcal{S}(\pi)| - 1 - 1 = 2|\mathcal{S}(\pi)| - 2 < 2|\mathcal{S}(\pi)| - 1$.

Altogether, either $|\mathcal{S}(\rho \circ \pi)| < 2|\mathcal{S}(\pi)| - 1$ or $|\mathcal{S}(\rho \circ \pi)| = 2|\mathcal{S}(\pi)| - 1$ and $\ell_{\rho\circ\pi} < 0 < f_{\rho\circ\pi}$ holds, which proves the statement. □

Given Lemma 4.1 to Lemma 4.4, the sought lower bound (on the number of required iTDRLs to generate a permutation from ι that has a certain number of maximal increasing substrings) can be proven as formulated in the following proposition.

Proposition 4.1. *For a permutation $\pi \in s\mathcal{P}_n$ that has been obtained from $\iota \in s\mathcal{P}_n$ by the application of $k \in \mathbb{N}$ iTDRLs it holds that either $|\mathcal{S}(\pi)| \leqslant 2^{k-1}$ or $|\mathcal{S}(\pi)| = 2^{k-1} + 1$ and $\ell_\pi < 0 < f_\pi$.*

Proof. The proposition is proven by induction on k. First consider the case $k = 1$. The application of a single iTDRL to ι yields a permutation with at most $2^{1-1} + 1 = 2$ maximal increasing substrings. This can be seen by the following argumentation that considers π' to be obtained by applying a single iTDRL $\rho \in \mathcal{M}_{\text{iTDRL}}$ (i.e., ρ is a riTDRL or a liTDRL) to ι. By the definition of an iTDRL π' can be written as $\pi' = \tau\tau'$ (respectively $\pi' = \tau'\tau$), where τ is a subsequence of ι and τ' is a subsequence of $\bar{\iota}$, in the case that ρ is an riTDRL (respectively

liTDRL). Corollary 4.1 implies $|\mathcal{S}(\iota)| = |\mathcal{S}(\bar{\iota})|$ and by Lemma 4.2 it holds that $|\mathcal{S}(\tau)| \leqslant |\mathcal{S}(\iota)|$ and $|\mathcal{S}(\tau')| \leqslant |\mathcal{S}(\iota)|$. Certainly, $|\mathcal{S}(\iota)| = 1$. Thus, by Lemma 4.3 $|\mathcal{S}(\pi')| = |\mathcal{S}(\tau\tau')| \leqslant |\mathcal{S}(\tau)| + |\mathcal{S}(\tau')| \leqslant 2$ (respectively $|\mathcal{S}(\pi')| = |\mathcal{S}(\tau'\tau)| \leqslant |\mathcal{S}(\tau')| + |\mathcal{S}(\tau)| \leqslant 2$) if ρ is an riTDRL (respectively liTDRL). Consequently, if one of τ or τ' is empty, then $|\mathcal{S}(\pi')| \leqslant 1$. If τ and τ' are not empty, then $|\mathcal{S}(\tau)| = 1$ and $|\mathcal{S}(\tau')| = 1$, and since ι contains only positive elements, all elements of τ (respectively τ') are positive (respectively negative). Thus, $\ell_{\tau'}, f_{\tau'} < 0$ and $\ell_\tau, f_\tau > 0$. Since $\ell_{\tau'} < f_\tau$ it follows by Lemma 4.3 that $|\mathcal{S}(\pi')| = |\mathcal{S}(\tau'\tau)| = |\mathcal{S}(\tau')| + |\mathcal{S}(\tau)| - 1 \leqslant 2 - 1 = 1$. Additionally, $\ell_{\pi'} < 0 < f_{\pi'}$ holds for the case that ρ is an riTDRL. Altogether, the statement holds for $k = 1$.

For the induction step, assume that σ is a permutation that has been obtained from ι by the application of $k-1$ iTDRLs and let π be a permutation obtained from σ by the application of a single iTDRL ρ. Then π can be written as $\pi = \tau\tau'$ (respectively $\pi = \tau'\tau$), where τ is a subsequence of σ and τ' is a subsequence of $\overline{\sigma}$, when ρ is an riTDRL (respectively liTDRL). Corollary 4.1 implies that $|\mathcal{S}(\sigma)| = |\mathcal{S}(\overline{\sigma})|$ and by Lemma 4.2 holds that $|\mathcal{S}(\tau)| \leqslant |\mathcal{S}(\sigma)|$ and $|\mathcal{S}(\tau')| \leqslant |\mathcal{S}(\overline{\sigma})|$. Then, Lemma 4.3 implies $|\mathcal{S}(\tau\tau')| = |\mathcal{S}(\tau)| + |\mathcal{S}(\tau')| - K_1 \leqslant |\mathcal{S}(\tau)| + |\mathcal{S}(\tau')| \leqslant 2|\mathcal{S}(\sigma)|$ and $|\mathcal{S}(\tau'\tau)| = |\mathcal{S}(\tau)| + |\mathcal{S}(\tau')| - K_2 \leqslant |\mathcal{S}(\tau)| + |\mathcal{S}(\tau')| \leqslant 2|\mathcal{S}(\sigma)|$, where $K_1 = 1$ (respectively $K_2 = 1$) if $\ell_\tau < 0 < f_{\tau'}$ (respectively $\ell_{\tau'} < 0 < f_\tau$) and $K_1 = 0$ (respectively $K_2 = 0$) otherwise. By the induction hypothesis $|\mathcal{S}(\sigma)| \leqslant 2^{k-2} + 1$ and if $|\mathcal{S}(\sigma)| = 2^{k-2} + 1$ then $\ell_\sigma < 0 < f_\sigma$. Therefore, $|\mathcal{S}(\tau)| \leqslant 2^{k-2} + 1$ and $|\mathcal{S}(\tau')| \leqslant 2^{k-2} + 1$. Hence, if $|\mathcal{S}(\tau)| \leqslant 2^{k-2}$ and $|\mathcal{S}(\tau')| \leqslant 2^{k-2}$, then $|\mathcal{S}(\tau\tau')| \leqslant 2^{k-1}$ and $|\mathcal{S}(\tau'\tau)| \leqslant 2^{k-1}$. Consequently, it remains to consider the cases where $|\mathcal{S}(\tau)|, |\mathcal{S}(\tau')| \geqslant 2^{k-2}$ and at least one of $|\mathcal{S}(\tau)|$ or $|\mathcal{S}(\tau')|$ is $2^{k-2} + 1$. Note that this implies that $|\mathcal{S}(\sigma)| = 2^{k-2} + 1$ and (by the induction hypothesis) it holds that $\ell_\sigma < 0 < f_\sigma$. Lemma 4.4 now implies that $|\mathcal{S}(\pi)| \leqslant 2|\mathcal{S}(\sigma)| - 1 = 2(2^{k-2} + 1) - 1 = 2^{k-1} + 1$. Moreover, by Lemma 4.4 it holds that $|\mathcal{S}(\pi)| = 2^{k-1} + 1$ which implies $\ell_\pi < 0 < f_\pi$.

Altogether, it holds either $|\mathcal{S}(\pi)| < 2^{k-1} + 1$ or $|\mathcal{S}(\pi)| = 2^{k-1} + 1$ and $\ell_\pi < 0 < f_\pi$, which proves the statement. □

The Upper Bound

In this subsection an upper bound is proven on the minimum number of iTDRLs that have to be applied to ι in order to produce a permutation with a certain number of maximal increasing substrings.

Consider a permutation π from $s\mathcal{P}_n$. The main idea to obtain the upper bound is to iteratively apply the inverse operation of an iTDRL to π that divides the number of maximal increasing substrings of π by two. The inverse operation of an iTDRL can be understood easily in the context of card shuffling as explained in the following. Consider a stack of cards that represents a given signed linear permutation π such that: 1) each element of π is represented by a single card; 2) a card in the stack is tidy (upside down) if the corresponding element of π has a positive (respectively negative) sign; and 3) the relative order of the cards is the same as the corresponding elements in π such that the topmost (bottommost) card in the stack corresponds to $\pi(1)$

(respectively $\pi(n)$). The inverted operation of an iTDRL can now be expressed as a variant of the riffle shuffle operation (see Section 2.3.4) proceeding in three steps: First, the stack of cards is split into two stacks. Second, one of both resulting stacks is flipped. Third, a riffle merges both parts to a single stack again. However, in order to half the number of the maximal increasing substrings of permutation π all three steps of the riffle shuffle variant are not performed randomly. Instead, all steps are performed with respect to the maximal increasing substrings of π. If $\mathcal{S}(\pi)$ is even (odd), then the stack is split after the card that corresponds to the last element of the $\mathcal{S}(\pi)/2$-th maximal increasing substring (respectively the last negative element of the $\lfloor \mathcal{S}(\pi)/2 \rfloor$-th maximal increasing substring). Furthermore, if the operation inverts an liTDRL (riTDRL), then the previously upper (respectively bottom) stack is inverted. Finally, a riffle halves the number of maximal increasing substrings of π by always merging cards that correspond to two maximal increasing substrings (each belonging to a different stack). Thereby, two maximal increasing substrings are merged into a new one such that the resulting substring is increasing. By iteratively applying this riffle shuffle variant the stack of cards is sorted or – in the context of permutations – permutation π is transformed into ι. In addition, a sequence of iTDRLs is indirectly created that transforms ι into π. The following paragraph formally defines an inverse operation of an riTDRL and an liTDRL as transformation T_1 and T_2, respectively.

Two transformations $T_i : s\mathcal{P}_n \to s\mathcal{P}_n$, $i \in [1:2]$, are defined to construct a permutation $T_i(\pi)$ from π which has the property that there always exists an iTDRL ρ such that $\rho \circ T_i(\pi) = \pi$. In other words, these transformations are the inverted iTDRL operations (which is proven in Lemma 4.5). Depending on whether the transformation is the inverse of an riTDRL or an liTDRL, a different transformation T_1 or T_2 is defined (and for each case it is also distinguished if the number of maximal increasing substrings of π is even or odd). For both constructions consider a signed linear permutation π of size n with the maximal substring decomposition $\pi = \pi_1 \dots \pi_{|\mathcal{S}(\pi)|}$. See also Figure 4.2 for examples of both transformations.

1) If $|\mathcal{S}(\pi)|$ is even, then $T_1(\pi) := \tau_1 \tau_2 \dots \tau_{|\mathcal{S}(\pi)|/2-1} \tau_{|\mathcal{S}(\pi)|/2}$, where

$$\begin{aligned}
\tau_1 &:= \pi_1 \oplus \overline{\pi_{|\mathcal{S}(\pi)|}}, \\
\tau_2 &:= \pi_2 \oplus \overline{\pi_{|\mathcal{S}(\pi)|-1}}, \\
&\vdots \\
\tau_{|\mathcal{S}(\pi)|/2-1} &:= \pi_{|\mathcal{S}(\pi)|/2-1} \oplus \overline{\pi_{|\mathcal{S}(\pi)|/2+2}}, \text{ and} \\
\tau_{|\mathcal{S}(\pi)|/2} &:= \pi_{|\mathcal{S}(\pi)|/2} \oplus \overline{\pi_{|\mathcal{S}(\pi)|/2+1}}.
\end{aligned}$$

If $|\mathcal{S}(\pi)|$ is odd, then $T_1(\pi) := \tau_1\tau_2\ldots\tau_{\lfloor|\mathcal{S}(\pi)|/2\rfloor}\tau_{\lceil|\mathcal{S}(\pi)|/2\rceil}$, where

$$\begin{aligned}
\tau_1 &:= \pi_1 \oplus \overline{\pi_{|\mathcal{S}(\pi)|}},\\
\tau_2 &:= \pi_2 \oplus \overline{\pi_{|\mathcal{S}(\pi)|-1}},\\
&\vdots\\
\tau_{\lfloor|\mathcal{S}(\pi)|/2\rfloor} &:= \pi_{\lfloor|\mathcal{S}(\pi)|/2\rfloor} \oplus \overline{\pi_{\lceil|\mathcal{S}(\pi)|/2\rceil+1}}, \text{ and}\\
\tau_{\lceil|\mathcal{S}(\pi)|/2\rceil} &:= \upsilon_{\lceil|\mathcal{S}(\pi)|/2\rceil} \oplus \overline{\kappa_{\lceil|\mathcal{S}(\pi)|/2\rceil}}
\end{aligned}$$

with $\upsilon_{\lceil|\mathcal{S}(\pi)|/2\rceil}$ (respectively $\kappa_{\lceil|\mathcal{S}(\pi)|/2\rceil}$) being the smallest substring of $\pi_{\lceil|\mathcal{S}(\pi)|/2\rceil}$ that contains all its negative (respectively positive) elements.

2) If $|\mathcal{S}(\pi)|$ is even, then $T_2(\pi) := \tau_1\tau_2\ldots\tau_{|\mathcal{S}(\pi)|/2-1}\tau_{|\mathcal{S}(\pi)|/2}$, where

$$\begin{aligned}
\tau_1 &:= \pi_{|\mathcal{S}(\pi)|/2+1} \oplus \overline{\pi_{|\mathcal{S}(\pi)|/2}},\\
\tau_2 &:= \pi_{|\mathcal{S}(\pi)|/2+2} \oplus \overline{\pi_{|\mathcal{S}(\pi)|/2-1}},\\
&\vdots\\
\tau_{|\mathcal{S}(\pi)|/2-1} &:= \pi_{|\mathcal{S}(\pi)|-1} \oplus \overline{\pi_2}, \text{ and}\\
\tau_{|\mathcal{S}(\pi)|/2} &:= \pi_{|\mathcal{S}(\pi)|} \oplus \overline{\pi_1}.
\end{aligned}$$

If $|\mathcal{S}(\pi)|$ is odd, then $T_2(\pi) := \tau_1\tau_2\ldots\tau_{\lfloor|\mathcal{S}(\pi)|/2\rfloor}\tau_{\lceil|\mathcal{S}(\pi)|/2\rceil}$, where

$$\begin{aligned}
\tau_1 &:= \kappa_{\lceil|\mathcal{S}(\pi)|/2\rceil} \oplus \overline{\upsilon_{\lceil|\mathcal{S}(\pi)|/2\rceil}},\\
\tau_2 &:= \pi_{\lceil|\mathcal{S}(\pi)|/2\rceil+1} \oplus \overline{\pi_{\lfloor|\mathcal{S}(\pi)|/2\rfloor}},\\
&\vdots\\
\tau_{\lfloor|\mathcal{S}(\pi)|/2\rfloor} &:= \pi_{|\mathcal{S}(\pi)|-1} \oplus \overline{\pi_2}, \text{ and}\\
\tau_{\lceil|\mathcal{S}(\pi)|/2\rceil} &:= \pi_{|\mathcal{S}(\pi)|} \oplus \overline{\pi_1}
\end{aligned}$$

with $\kappa_{\lceil|\mathcal{S}(\pi)|/2\rceil}$ (respectively $\upsilon_{\lceil|\mathcal{S}(\pi)|/2\rceil}$) being the smallest substring of $\pi_{\lceil|\mathcal{S}(\pi)|/2\rceil}$ that contains all its positive (respectively negative) elements.

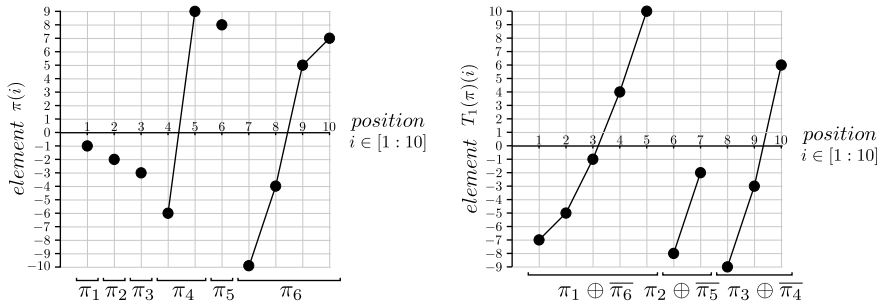
Four auxiliary lemmata are proven in the following. These lemmata are used in the proof of the main result of this subsection which is formulated in Proposition 4.2. The first lemma states the claim that the transformation T_1 (respectively T_2) is the inverse operation of an riTDRL (respectively liTDRL).

Lemma 4.5. *Let $\pi \in s\mathcal{P}_n$. The following statements are true:*

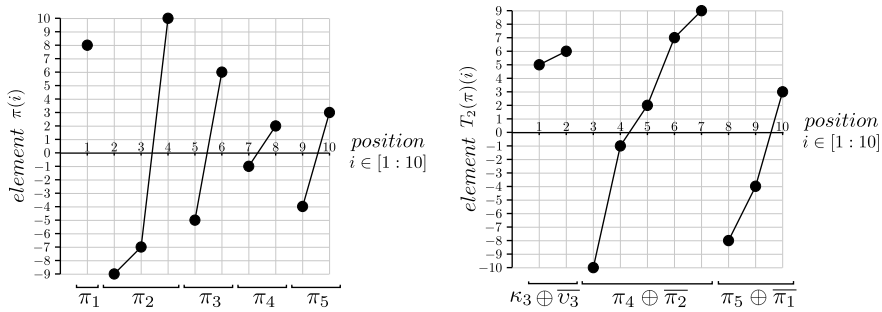
1) *There exists an riTDRL $\rho \in \mathcal{M}_{iTDRL}$ such that $\rho \circ T_1(\pi) = \pi$.*

2) *There exists an liTDRL $\rho \in \mathcal{M}_{iTDRL}$ such that $\rho \circ T_2(\pi) = \pi$.*

Proof. Let π be a signed linear permutation of size n and let $\pi = \pi_1\pi_2\ldots\pi_{|\mathcal{S}(\pi)|}$ be the maximal increasing substring decomposition of π. Then $T_1(\pi) = \tau_1\tau_2\ldots\tau_{|\mathcal{S}(\pi)|/2-1}\tau_{|\mathcal{S}(\pi)|/2}$ (respectively



(a) Case $T_1(\pi)$



(b) Case $T_2(\pi)$

Figure 4.2: Examples of the transformation T_1 (a) and T_2 (b) that is applied to $\pi = (-1\ -2\ -3\ -6\ 9\ 8\ -10\ -4\ 5\ 7)$ (a) and $\pi = (8\ -9\ -7\ 10\ -5\ 6\ -1\ 2\ -4\ 3)$ (b). The transformation $T_i(\pi)$, $i \in [1:2]$, is shown on the right in the respective subfigure. The notation is as in Figure 4.1. In addition, κ_3 (respectively υ_3) is the smallest substring of π_3 that contains all its positive (respectively negative) elements, i.e., $\kappa_3 = 6$ and $\upsilon_3 = -5$. For each permutation that is illustrated its maximal increasing substring decomposition is shown on the bottom of its illustration.

$T_1(\pi) = \tau_1\tau_2\ldots\tau_{\lfloor|\mathcal{S}(\pi)|/2\rfloor}\tau_{\lceil|\mathcal{S}(\pi)|/2\rceil})$ in the case that $|\mathcal{S}(\pi)|$ is even (respectively odd). It follows that

$$\begin{aligned} S_1 &= \pi_1\pi_2\ldots\pi_{|\mathcal{S}(\pi)|/2-1}\pi_{|\mathcal{S}(\pi)|/2} \text{ and} \\ S_1' &= \overline{\pi_{|\mathcal{S}(\pi)|}}\,\overline{\pi_{|\mathcal{S}(\pi)|-1}}\ldots\overline{\pi_{|\mathcal{S}(\pi)|/2+2}}\,\overline{\pi_{|\mathcal{S}(\pi)|/2+1}}, \end{aligned}$$

are disjoint subsequences of $T_1(\pi)$ in the case that $|\mathcal{S}(\pi)|$ is even. If otherwise $|\mathcal{S}(\pi)|$ is odd, then it follows that

$$\begin{aligned} S_1 &= \pi_1\pi_2\ldots\pi_{\lfloor|\mathcal{S}(\pi)|/2\rfloor}\upsilon_{\lceil|\mathcal{S}(\pi)|/2\rceil} \text{ and} \\ S_1' &= \overline{\pi_{|\mathcal{S}(\pi)|}}\,\overline{\pi_{|\mathcal{S}(\pi)|-1}}\ldots\overline{\pi_{\lceil|\mathcal{S}(\pi)|/2\rceil+1}}\,\overline{\kappa_{\lceil|\mathcal{S}(\pi)|/2\rceil}} \end{aligned}$$

are disjoint subsequences of $T_1(\pi)$.

Furthermore, $T_2(\pi) = \tau_1\tau_2\ldots\tau_{|\mathcal{S}(\pi)|/2-1}\tau_{|\mathcal{S}(\pi)|/2}$ (respectively $T_2(\pi) = \tau_1\tau_2\ldots\tau_{\lfloor|\mathcal{S}(\pi)|/2\rfloor}\tau_{\lceil|\mathcal{S}(\pi)|/2\rceil}$) in the case that $|\mathcal{S}(\pi)|$ is even (respectively odd). The sequences

$$\begin{aligned} S_2 &= \overline{\pi_{|\mathcal{S}(\pi)|/2}}\,\overline{\pi_{|\mathcal{S}(\pi)|/2-1}}\ldots\overline{\pi_2}\,\overline{\pi_1} \text{ and} \\ S_2' &= \pi_{|\mathcal{S}(\pi)|/2+1}\pi_{|\mathcal{S}(\pi)|/2+2}\ldots\pi_{|\mathcal{S}(\pi)|-1}\pi_{|\mathcal{S}(\pi)|} \end{aligned}$$

are disjoint subsequences of $T_2(\pi)$ in the case that $|\mathcal{S}(\pi)|$ is even. If otherwise $|\mathcal{S}(\pi)|$ is odd, then it follows that the sequences

$$S_2 = \overline{\upsilon_{\lceil|\mathcal{S}(\pi)|/2\rceil}}\,\overline{\pi_{\lfloor|\mathcal{S}(\pi)|/2\rfloor}}\ldots\overline{\pi_2}\,\overline{\pi_1} \text{ and}$$
$$S_2' = \kappa_{\lceil|\mathcal{S}(\pi)|/2\rceil}\pi_{\lceil|\mathcal{S}(\pi)|/2\rceil+1}\ldots\pi_{|\mathcal{S}(\pi)|-1}\pi_{|\mathcal{S}(\pi)|}$$

are disjoint subsequences of $T_2(\pi)$.

Note that $\pi = S_i\overline{S_i'}$ holds (in the respective case $i \in [1:2]$) and S_i, $\overline{S_i'}$ are disjoint subsequences of $T_i(\pi)$ that together include all elements of $T_i(\pi)$. Hence, for the riTDRL (respectively liTDRL) $\rho_{\text{riTDRL}}(r, \mathcal{E}(S_1), \mathcal{E}(S_1'))$ (respectively $\rho_{\text{liTDRL}}(\ell, \mathcal{E}(S_1), \mathcal{E}(S_1'))$) it holds that $\rho_{\text{riTDRL}} \circ T_1(\pi) = \pi$ (respectively $\rho_{\text{liTDRL}} \circ T_2(\pi) = \pi$). □

The transformations $T_1(\pi)$ and $T_2(\pi)$ have been designed such that the following lemma holds.

Lemma 4.6. *Let π be a signed permutation of size n. The decomposition of the permutation $T_i(\pi)$, $i \in [1:2]$, into strings $\tau_1\tau_2\ldots\tau_t$ (where t is as in the respective case) is a maximal increasing substring decomposition.*

Proof. Let $\pi \in s\mathcal{P}_n$ and let $\pi = \pi_1\ldots\pi_{|\mathcal{S}(\pi)|}$ be the maximal increasing substring decomposition of π. For the proof it is sufficient to show that for each $j \in [1:t-1]$ it holds that the last element of τ_j is larger than the first element of τ_{j+1}, i.e., $\ell_{\tau_j} > f_{\tau_{j+1}}$. In the following the proof is described for the case that $i = 1$ (in the case that $i = 2$ the proof can be done analogously).

In this case the lemma has to be shown for $T_1(\pi) := \tau_1\tau_2\ldots\tau_{t-1}\tau_t$, where $t = |\mathcal{S}(\pi)|/2$ (respectively $t = \lceil|\mathcal{S}(\pi)|/2\rceil$) in the case that $|\mathcal{S}(\pi)|$ is even (respectively odd). Since all elements of two sequences X and Y are sorted increasingly in $X \oplus Y$ it follows that every τ_j with $j \in [1:t]$ is an increasing substring. By the construction of $T_1(\pi)$ it holds that $\tau_j = \pi_j \oplus \overline{\pi_{t+1-j}}$ for $j \in [1:t]$. Hence, a τ_j always contains all elements of π_j. Consequently, $\ell_{\tau_j} \geqslant \ell_{\pi_j}$ and $f_{\tau_j} \leqslant f_{\pi_j}$ holds for all $j \in [1:t]$. The fact that π_j and π_{j+1} are two maximal increasing substrings, i.e., $\ell_{\pi_j} > f_{\pi_{j+1}}$ for $j \in [1:|\mathcal{S}(\pi)|-1]$, implies $\ell_{\tau_j} \geqslant \ell_{\pi_j} > f_{\pi_{j+1}} \geqslant f_{\tau_{j+1}}$ for all $j \in [1:t-1]$. Therefore, $\tau_1, \ldots, \tau_t$ are maximal proving the lemma. □

The following lemma shows that the application of the transformation T_1 and T_2 to a permutation π results in a permutation that has half as many maximal increasing substrings as π.

Lemma 4.7. *Let $\pi \in s\mathcal{P}_n$ with $|\mathcal{S}(\pi)| > 1$. Then $|\mathcal{S}(T_i(\pi))| = \lceil|\mathcal{S}(\pi)|/2\rceil$ holds for all $i \in [1:2]$.*

Proof. Let $\pi \in s\mathcal{P}_n$ with $|\mathcal{S}(\pi)| > 1$. Consider the case that $|\mathcal{S}(\pi)|$ is even. By the construction of $T_i(\pi) = \tau_1\ldots\tau_{|\mathcal{S}(T_i(\pi))|}$ with $i = 1,2$ it holds that two maximal increasing substrings of π always form a new increasing substring in $T_i(\pi)$, hence $|\mathcal{S}(T_i(\pi))| \leqslant |\mathcal{S}(\pi)|/2$. By Lemma 4.6 it holds that every τ_i of $T_i(\pi)$ is also maximal, and hence $|\mathcal{S}(T_i(\pi))| \geqslant |\mathcal{S}(\pi)|/2$. Altogether, $|\mathcal{S}(T_i(\pi))| = |\mathcal{S}(\pi)|/2$ if $|\mathcal{S}(\pi)|$ is even.

Now consider that $|\mathcal{S}(\pi)|$ is odd. By the construction of $T_i(\pi) = \tau_1 \dots \tau_{|\mathcal{S}(T_i(\pi))|}$ with $i = 1, 2$ it holds that $\tau_1, \dots, \tau_{|\mathcal{S}(T_i(\pi))|-1}$ (respectively $\tau_2, \dots, \tau_{|\mathcal{S}(T_i(\pi))|}$) of $T_i(\pi)$ are always formed by two maximal increasing substrings of π and $\tau_{|\mathcal{S}(T_i(\pi))|}$ (respectively τ_1) is formed by one maximal increasing substring of π if $i = 1$ (respectively $i = 2$). Hence, $|\mathcal{S}(T_i(\pi))| \leqslant \lceil |\mathcal{S}(\pi)|/2 \rceil$. By Lemma 4.6 it holds that every τ_i of $T_i(\pi)$ is also maximal, hence $|\mathcal{S}(T_i(\pi))| \geqslant \lceil |\mathcal{S}(\pi)|/2 \rceil$. Altogether, $|\mathcal{S}(T_i(\pi))| = \lceil |\mathcal{S}(\pi)|/2 \rceil$ if $|\mathcal{S}(\pi)|$ is odd. □

Consider a signed permutation π with at least two maximal increasing substrings such that the first (last) element of π is positive (respectively negative). The following lemma states that the application of the transformation T_i, $i \in [1:2]$, to π preserve this structure, i. e., the first (last) element of $T_i(\pi)$ is positive (respectively negative).

Lemma 4.8. *Let π be a signed permutation of size n with $|\mathcal{S}(\pi)| > 1$, $|\mathcal{S}(\pi)|$ odd, and $\ell_\pi < 0 < f_\pi$. For all $i \in \{1, 2\}$ it holds that $\ell_{T_i(\pi)} < 0 < f_{T_i(\pi)}$.*

Proof. Let $\pi \in s\mathcal{P}_n$ and let $\pi = \pi_1 \dots \pi_{|\mathcal{S}(\pi)|}$ be the maximal increasing substring decomposition of π with $|\mathcal{S}(\pi)| > 1$ and $|\mathcal{S}(\pi)|$ is odd. The fact that $f_\pi > 0$ (respectively $\ell_\pi < 0$) implies that π_1 (respectively $\pi_{|\mathcal{S}(\pi)|}$) contains only positive (respectively negative) elements. Consequently, $\overline{\pi_1}$ (respectively $\overline{\pi_{|\mathcal{S}(\pi)|}}$) contains only negative (respectively positive) elements. Therefore, $\pi_1 \oplus \overline{\pi_{|\mathcal{S}(\pi)|}}$ (that is the leftmost maximal increasing substring in $T_1(\pi)$) contains only positive elements. Analogously, $\pi_{|\mathcal{S}(\pi)|} \oplus \overline{\pi_1}$ (the rightmost maximal increasing substring in $T_2(\pi)$) contains only negative elements and, thus, $f_{T_1(\pi)} > 0$ and $\ell_{T_2(\pi)} < 0$. By definition it holds that the rightmost (respectively leftmost) maximal increasing substring of $T_1(\pi)$ (respectively $T_2(\pi)$) is $\upsilon_{\lceil |\mathcal{S}(\pi)| \rceil} \oplus \overline{\kappa_{\lceil |\mathcal{S}(\pi)| \rceil}}$ (respectively $\kappa_{\lceil |\mathcal{S}(\pi)| \rceil} \oplus \overline{\upsilon_{\lceil |\mathcal{S}(\pi)| \rceil}}$). Since $\kappa_{\lceil |\mathcal{S}(\pi)| \rceil}$ (respectively $\upsilon_{\lceil |\mathcal{S}(\pi)| \rceil}$) contains only positive (respectively negative) elements, it holds that $\overline{\kappa_{\lceil |\mathcal{S}(\pi)| \rceil}}$ (respectively $\overline{\upsilon_{\lceil |\mathcal{S}(\pi)| \rceil}}$) contains only negative (respectively positive) elements. Therefore, $\upsilon_{\lceil |\mathcal{S}(\pi)| \rceil} \oplus \overline{\kappa_{\lceil |\mathcal{S}(\pi)| \rceil}}$ (respectively $\kappa_{\lceil |\mathcal{S}(\pi)| \rceil} \oplus \overline{\upsilon_{\lceil |\mathcal{S}(\pi)| \rceil}}$) contains only negative (respectively positive) elements, which implies $\ell_{T_1(\pi)} < 0$ and $f_{T_2(\pi)} > 0$. □

Given Lemma 4.5 to Lemma 4.8, the sought upper bound (on the minimum number of iTDRLs that have to be applied to ι in order to produce a permutation with a certain number of maximal increasing substrings) can be proven as formulated in the following proposition.

Proposition 4.2. *Let $\pi \in s\mathcal{P}_n \setminus \{\iota\}$ such that $|\mathcal{S}(\pi)| = 2^{k-1} + 1$ and $\ell_\pi < 0 < f_\pi$ or $|\mathcal{S}(\pi)| \leqslant 2^{k-1}$ for a $k \in \mathbb{N}$. It holds that permutation π can be obtained by applying k iTDRLs to ι.*

Proof. Let $\pi \in s\mathcal{P}_n \setminus \{\iota\}$. The proposition is proven by induction on k. For the base case assume $k = 1$. There exist the two cases: i) $|\mathcal{S}(\pi)| \leqslant 2^{1-1} = 2^0 = 1$ or ii) $|\mathcal{S}(\pi)| = 2^{1-1} + 1 = 2^0 + 1 = 2$ and $\ell_\pi < 0 < f_\pi$. Consider Case (i). Since every permutation has at least one maximal increasing substring the equation $|\mathcal{S}(\pi)| = 1$ holds. Since π is unequal to ι it follows that can be written as $\pi = (\pi(1) \dots \pi(j)\ \pi(j+1) \dots \pi(n))$,

where $\pi(1) < \ldots < \pi(j) < 0 < \pi(j+1) < \ldots < \pi(n)$ and $j \in [1:n-1]$. Hence, for the liTDRL $\rho_{\text{liTDRL}}(\ell, \{|\pi(1)|, \ldots, |\pi(j)|\}, \{\pi(j+1), \ldots, \pi(n)\})$ it holds that $\rho_{\text{liTDRL}} \circ \iota = \pi$. Now consider Case (ii). Since $0 < f_\pi$ (respectively $\ell_\pi < 0$) it holds that the left (respectively right) maximal increasing substring of π contains only positive (respectively negative) elements. Hence, π can be written as $\pi = (\pi(1) \ldots \pi(j)\ \pi(j+1) \ldots \pi(n))$, where $\pi(j+1) < \ldots < \pi(n) < 0 < \pi(1) < \ldots < \pi(j)$ and $j \in [1:n-1]$. Then the equation $\rho_{\text{riTDRL}} \circ \iota = \pi$ is satisfied for the riTDRL $\rho_{\text{riTDRL}}(r, \{\pi(1), \ldots, \pi(j)\}, \{|\pi(j+1)|, \ldots, |\pi(n)|\})$.

For the induction step let $k > 1$. It is sufficient to consider the following two cases: i) $|\mathcal{S}(\pi)| \leqslant 2^{k-1}$ or ii) $|\mathcal{S}(\pi)| = 2^{k-1} + 1$ and $\ell_\pi < 0 < f_\pi$. Lemma 4.7 implies $|\mathcal{S}(T_i(\pi))| \leqslant 2^{k-2}$ if Case (i) holds and $|\mathcal{S}(T_i(\pi))| \leqslant \lceil (2^{k-1}+1)/2 \rceil = 2^{k-2} + 1$ if Case (ii) holds. Furthermore, Lemma 4.8 ensures that $\ell_{T_i(\pi)} < 0 < f_{T_i(\pi)}$ if Case (ii) holds. By Lemma 4.5, π can be obtained by applying a single riTDRL (respectively an liTDRL) to $T_1(\pi)$ (respectively $T_2(\pi)$) and by the induction hypothesis $T_i(\pi)$ with $i \in [1:2]$ can be obtained by applying $k-1$ iTDRLs to ι. Hence, π can be obtained by applying k iTDRLs to ι. □

Remark 4.1. *Observe, that the proof of Proposition 4.2 shows that the permutation π in Proposition 4.2 can always be obtained from ι by a single iTDRL (i. e., riTDRL or liTDRL) followed by $k-1$ riTDRLs.*

The Main Theorem

The following theorem characterizes permutations π that have a certain number of maximal increasing substrings with respect to the number of iTDRLs that are necessary and sufficient to obtain π from the identity permutation ι.

Theorem 4.1. *Let $\pi \in s\mathcal{P}_n \setminus \{\iota\}$ be such that either $|\mathcal{S}(\pi)| = 2^{k-2} + 1$ and $\ell_\pi > 0$ or $f_\pi < 0$, $2^{k-2} + 1 < |\mathcal{S}(\pi)| \leqslant 2^{k-1}$, or $|\mathcal{S}(\pi)| = 2^{k-1} + 1$ and $\ell_\pi < 0 < f_\pi$ for a $k \in \mathbb{N}$. It holds that k iTDRLs are necessary and sufficient in order to obtain π from ι.*

Proof. Let $\pi \in s\mathcal{P}_n \setminus \{\iota\}$ be such that either $2^{k-2} + 1 = |\mathcal{S}(\pi)|$ and $\ell_\pi > 0$ or $f_\pi < 0$, $2^{k-2} + 1 < |\mathcal{S}(\pi)| \leqslant 2^{k-1}$, or $|\mathcal{S}(\pi)| = 2^{k-1} + 1$ and $\ell_\pi < 0 < f_\pi$ for a $k \in \mathbb{N}$. Proposition 4.1 shows that at least k iTDRLs are necessary to obtain π from ι. Proposition 4.2 shows that k iTDRLs are sufficient to obtain π from ι. Altogether, the theorem follows. □

Given a permutation $\pi \in s\mathcal{P}_n$, Theorem 4.1 determines the minimum number of iTDRLs that are necessary to obtain π from ι, i. e., it immediately implies the $\mathcal{M}_{\text{iTDRL}}$ distance as proven in the following section.

4.1.3 *Inverse Tandem Duplication Random Loss Distance on Signed Linear Permutations*

This section considers the distance problem and the sorting problem for signed linear permutations under the iTDRL rearrangement

model. The following corollary of Theorem 4.1 states a closed formula for the $\mathcal{M}_{\text{iTDRL}}$ distance for signed linear permutations.

Corollary 4.2. *The $\mathcal{M}_{iTDRL}$ distance for ι and any signed linear permutation $\pi \in s\mathcal{P}_n \setminus \{\iota\}$ is given by*

$$d_{\mathcal{M}_{iTDRL}}(\iota, \pi) = \begin{cases} \log_2(|\mathcal{S}(\pi)| - 1) + 1 & \text{if } \exists\, k \in \mathbb{N} \text{ with } |\mathcal{S}(\pi)| = 2^{k-1} + 1 \\ & \text{and } \ell_\pi < 0 < f_\pi, \\ \lceil \log_2 |\mathcal{S}(\pi)| \rceil + 1 & \text{else.} \end{cases}$$

Proof. Let $\pi \in s\mathcal{P}_n \setminus \{\iota\}$. Then either $|\mathcal{S}(\pi)| = 2^{k-2} + 1$ and $\ell_\pi > 0$ or $f_\pi < 0$, $2^{k-2} + 1 < |\mathcal{S}(\pi)| \leqslant 2^{k-1}$, or $|\mathcal{S}(\pi)| = 2^{k-1} + 1$ and $\ell_\pi < 0 < f_\pi$ holds for a $k \in \mathbb{N}$. Hence, it holds either $2^{k-2} < |\mathcal{S}(\pi)| \leqslant 2^{k-1}$ or $|\mathcal{S}(\pi)| = 2^{k-1} + 1$ and $\ell_\pi < 0 < f_\pi$. Theorem 4.1 implies that k iTDRLs are necessary and sufficient to obtain π from ι, i. e., $d_{\mathcal{M}_{\text{iTDRL}}}(\iota, \pi) = k$. Consider first $2^{k-2} < |\mathcal{S}(\pi)| \leqslant 2^{k-1}$. This implies $k - 2 < \log_2 |\mathcal{S}(\pi)| \leqslant k - 1$ and hence $k - 1 \leqslant \lceil \log_2 |\mathcal{S}(\pi)| \rceil \leqslant k - 1$. Consequently, it holds that $d_{\mathcal{M}_{\text{iTDRL}}}(\iota, \pi) = k = \lceil \log_2 |\mathcal{S}(\pi)| \rceil + 1$. Now consider $|\mathcal{S}(\pi)| = 2^{k-1} + 1$. Then $|\mathcal{S}(\pi)| - 1 = 2^{k-1}$ implies $\log_2(|\mathcal{S}(\pi)| - 1) = k - 1$. Consequently, $d_{\mathcal{M}_{\text{iTDRL}}}(\iota, \pi) = k = \log_2(|\mathcal{S}(\pi)| - 1) + 1$. □

Note that Corollary 4.2 implies that the $\mathcal{M}_{\text{iTDRL}}$ distance of a signed permutation π of size n can be computed by calculating the number of its maximal increasing substrings. Certainly, this can be done in time $\mathcal{O}(n)$. The following example illustrates both cases of the $\mathcal{M}_{\text{iTDRL}}$ distance that are distinguished in Corollary 4.2.

Example 4.2. *Consider the permutation $\pi = (-3\ -1\ 5\ -2\ 4)$. Permutation π has the two maximal increasing substrings $\pi_1 = -3\ -1\ 5$ and $\pi_2 = -2\ 4$, hence $|\mathcal{S}(\pi)| = 2$. The maximal increasing substring decomposition of π is $\pi_1\pi_2$. Since $f_\pi = -3 < 0$ the second case that is distinguished in Corollary 4.2 is satisfied, which implies that $d_{\mathcal{M}_{iTDRL}}(\iota, \pi) = \lceil \log_2 |\mathcal{S}(\pi)| \rceil + 1 = 2$. Thus exactly two iTDRLs are necessary to obtain π from ι. For a parsimonious scenario that transforms ι to π see Example 4.3 and Figure 4.3. For an example of a permutation that satisfies the first case of Corollary 4.2, consider the permutation $\sigma = (4\ 2\ -3\ -5\ \ -6\ -1)$. Permutation σ has the five maximal increasing substrings $\sigma_1 = 4$, $\sigma_2 = 2$, $\sigma_3 = -3$, $\sigma_4 = -5$, and $\sigma_5 = -6\ -1$. Consequently, $|\mathcal{S}(\sigma)| = 5$. In addition, the first (last) element of σ is positive (respectively negative), i. e., $\ell_\sigma = -1 < 0 < 4 = f_\sigma$, and the equation $|\mathcal{S}(\sigma)| = 2^{k-1} + 1$ is ensured for $k = 3$. By Corollary 4.2 it holds that the $\mathcal{M}_{iTDRL}$ distance is given by $d_{\mathcal{M}_{iTDRL}}(\iota, \sigma) = \log_2(|\mathcal{S}(\sigma)| - 1) + 1 = 3$.*

The following corollary determines the diameter of the $\mathcal{M}_{\text{iTDRL}}$ distance for the set of all signed linear permutations.

Corollary 4.3. *The diameter for the set of all directed signed linear permutations $s\mathcal{P}_n$ under $\mathcal{M}_{iTDRL}$ is given by: $D_{\mathcal{M}_{iTDRL}}(s\mathcal{P}_n) = \lceil \log_2 n \rceil + 1$*

Proof. By definition of the diameter and the left-invariance of signed linear permutations it holds that:

$$D_{\mathcal{M}_{\text{iTDRL}}}(s\mathcal{P}_n) = \max_{\gamma, \sigma \in s\mathcal{P}_n} d_{\mathcal{M}_{\text{iTDRL}}}(\gamma, \sigma) = \max_{\pi \in s\mathcal{P}_n} d_{\mathcal{M}_{\text{iTDRL}}}(\iota, \pi).$$

It follows from Corollary 4.2 that $d_{\mathcal{M}_{\text{iTDRL}}}(\iota, \pi)$ is $\log_2(|\mathcal{S}(\pi)| - 1) + 1$ if there exists a $k \in \mathbb{N}$ with $|\mathcal{S}(\pi)| = 2^{k-1} + 1$ and $\ell_\pi < 0 < f_\pi$, and $\lceil \log_2 |\mathcal{S}(\pi)| \rceil + 1$ otherwise. The fact that $\lceil \log_2 |\mathcal{S}(\pi)| \rceil + 1 \geqslant \log_2(|\mathcal{S}(\pi)| - 1) + 1$ implies:

$$\begin{aligned} D_{\mathcal{M}_{\text{iTDRL}}}(s\mathcal{P}_n) &= \max_{\pi \in s\mathcal{P}_n} d_{\mathcal{M}_{\text{iTDRL}}}(\iota, \pi) = \max_{\pi \in s\mathcal{P}_n} \lceil \log_2 |\mathcal{S}(\pi)| \rceil + 1 \\ &= \lceil \log_2 \max_{\pi \in s\mathcal{P}_n} |\mathcal{S}(\pi)| \rceil + 1 = \lceil \log_2 n \rceil + 1. \end{aligned}$$

The last equation follows by the fact that π is a permutation of size n which implies that $|\mathcal{S}(\pi)| \in [1:n]$. □

Corollary 4.3 states that the $\mathcal{M}_{\text{iTDRL}}$ diameter for a signed linear permutation of size $n \in \mathbb{N}$ is obtained if its number of maximal increasing substrings is n which is the case for, e. g., $(n \ \ldots \ 2 \ 1)$ and $(-1 \ -2 \ \ldots \ -n)$.

Motivated by the tractability of the $\mathcal{M}_{\text{iTDRL}}$ distance, the following paragraphs study the sorting problem of signed linear permutations under the iTDRL model. Recall that the *sorting problem* for a permutation π under $\mathcal{M}_{\text{iTDRL}}$ is to find a scenario $S \in \mathfrak{S}_{\mathcal{M}_{\text{iTDRL}}}(\iota, \pi)$ such that $S \in \arg\min_{S' \in \mathfrak{S}_{\mathcal{M}_{\text{iTDRL}}}(\iota,\pi)} |S'|$. Observe that this definition of the sorting problem indirectly covers the problem to find a shortest scenario $\rho_1(d_1, L_1, R_1), \ldots, \rho_t(d_t, L_t, R_t) \in \mathcal{M}_{\text{iTDRL}}$ of iTDRLs between two arbitrary permutations $\pi, \sigma \in s\mathcal{P}_n$ (i. e., $\rho_t \circ \ldots \circ \rho_1 \circ \pi = \sigma$), since such a sequence can be obtained by firstly finding $\rho'_t \circ \ldots \circ \rho'_1 \circ \iota = \pi^{-1} \circ \sigma$ with $\rho'_i(d'_i, L'_i, R'_i) \in \mathcal{M}_{\text{iTDRL}}$ and subsequently computing the sought iTDRLs $\rho_1, \ldots, \rho_t$ by $d_1 = d'_1$, $L_i = \{\pi(j) \colon j \in L'_i\}$, and $R_i = \{\pi(j) \colon j \in R'_i\}$ for all $i \in [1:t]$.

In the following, an algorithm is presented that solves the sorting problem for signed linear permutations under the iTDRL rearrangement model. Recall that according to Remark 4.1 there is always a solution of the considered sorting problem that contains at most one liTDRL. Therefore, the following algorithm computes for a given permutation $\pi \in s\mathcal{P}_n$ a sequence $\rho_1, \ldots, \rho_t \in \mathcal{M}_{\text{iTDRL}}$ of iTDRLs such that $t := d_{\mathcal{M}_{\text{iTDRL}}}(\iota, \pi)$, $\rho_t \circ \ldots \circ \rho_1 \circ \iota = \pi$, and either $\rho_1, \ldots, \rho_t$ are riTDRLs or ρ_1 is an liTDRL and $\rho_2, \ldots, \rho_t$ are riTDRLs. The pseudo code of the algorithm can be found in Algorithm 1. See Example 4.3 for an illustration of the algorithm. The main idea of Algorithm 1 is to iteratively apply the inverse operation of an iTDRL. In particular, transformation T_1 or (once in the last step) T_2 is iteratively applied to the given permutation π to obtain a permutation $T_i(\pi)$ that has at most half as many maximal increasing substrings as π. By that process a minimum length sequence S of transformations (of T_1 or T_2) is obtained in reversed order, i. e., S transforms π into ι. Subsequently, the sought minimum length sequence of iTDRLs transforming ι into π is obtained by computing the inverting iTDRL for every transformation in S and reversing the relative order of all computed iTDRLs. Since Algorithm 1 uses exactly $d_{\mathcal{M}_{\text{iTDRL}}}(\iota, \pi)$ iTDRLs to construct the sought sequence of iTDRLs, it solves the sorting problem for a given permutation π under $\mathcal{M}_{\text{iTDRL}}$ exactly.

Algorithm 1 : Pseudo code of sorting by iTDRLs

Data : $\pi \in s\mathcal{P}_n$
Result : $(\rho_1, \ldots, \rho_t) \in \mathfrak{S}_{\mathcal{M}_{\text{iTDRL}}}(\iota, \pi)$ such that $t = d_{\mathcal{M}_{\text{iTDRL}}}(\iota, \pi)$

1 **if** $\pi == \iota$ **then**
2 | **return** $\emptyset$;
3 **if** $\exists\, h \in \mathbb{N}_0 : |\mathcal{S}(\pi)| = 2^h + 1$ *and* $\ell_\pi < 0 < f_\pi$ **then**
4 | $t = \log_2(|\mathcal{S}(\pi)| - 1) + 1$;
5 **else**
6 | $t = \lceil \log_2 |\mathcal{S}(\pi)| \rceil + 1$;
7 **for** $j \leftarrow t, \ldots, 1$ **do**
8 | $\pi = \pi_1 \ldots \pi_{|\mathcal{S}(\pi)|}$;
9 | **if** $j == 1$ *and* $f_\pi < 0$ **then** // Application T_2
10 | | $\pi \leftarrow \overline{\upsilon_1} \oplus \kappa_1 = T_2(\pi)$;
11 | | $\rho_j \leftarrow \rho_j(\ell, \mathcal{E}(\overline{\upsilon_1}), \mathcal{E}(\kappa_1))$;
12 | | **continue**;
13 | **if** $|\mathcal{S}(\pi)|$ *is even* **then** // Application T_1
14 | | $\pi \leftarrow \pi_1 \oplus \overline{\pi_{|\mathcal{S}(\pi)|}} \ldots \pi_{|\mathcal{S}(\pi)|/2} \oplus \overline{\pi_{|\mathcal{S}(\pi)|/2+1}} = T_1(\pi)$;
15 | | $\rho_j \leftarrow \rho_j(r, \mathcal{E}(\pi_1 \ldots \pi_{|\mathcal{S}(\pi)|/2}), \mathcal{E}(\overline{\pi_{|\mathcal{S}(\pi)|}} \ldots \overline{\pi_{|\mathcal{S}(\pi)|/2+1}}))$;
16 | **else**
17 | | $\pi \leftarrow \pi_1 \oplus \overline{\pi_{|\mathcal{S}(\pi)|}} \ldots \upsilon_{\lceil |\mathcal{S}(\pi)|/2 \rceil} \oplus \overline{\kappa_{\lceil |\mathcal{S}(\pi)|/2 \rceil}} = T_1(\pi)$;
18 | | $\rho_j \leftarrow \rho_j(r, \mathcal{E}(\pi_1 \ldots \upsilon_{\lceil |\mathcal{S}(\pi)|/2 \rceil}), \mathcal{E}(\overline{\pi_{|\mathcal{S}(\pi)|}} \ldots \overline{\kappa_{\lceil |\mathcal{S}(\pi)|/2 \rceil}}))$;
19 | **continue**;
20 **return** $(\rho_1, \ldots, \rho_t)$;

Let $\pi \in s\mathcal{P}_n$. The case $\pi = \iota$ (i. e., the sorting sequence of iTDRLs is empty) is handled in lines 1 to 2. If otherwise $\pi \neq \iota$ then $\rho_t, \ldots, \rho_1 \in \mathcal{M}_{\text{iTDRL}}$ with $t = d_{\mathcal{M}_{\text{iTDRL}}}(\iota, \pi)$ are iteratively computed in the lines 3 to 19. By Corollary 4.2 either $d_{\mathcal{M}_{\text{iTDRL}}}(\iota, \pi) = \lceil \log_2 |\mathcal{S}(\pi)| \rceil + 1$ or $d_{\mathcal{M}_{\text{iTDRL}}}(\iota, \pi) = \log_2(|\mathcal{S}(\pi)| - 1) + 1$ and both cases are handled in lines 3 to 6. For every $j \in [t : 1]$ the maximal increasing substring composition of π is computed in Line 8 and – depending on j and whether $|\mathcal{S}(\pi)|$ is even or odd – either T_1 or T_2 is applied to π in lines 13 to 19 or lines 9 to 12. More precisely, if $j = 1$ and $f_\pi < 0$ (i. e., π is exactly one maximal increasing substring that contains negative and possibly positive elements) then T_2 is applied to π in Line 10 and in Line 11 the corresponding liTDRL (which exists since Lemma 4.5) is computed. Otherwise, i. e., either $j > 1$ or $j = 1$ and $f_\pi > 0$, the permutation π is substituted by $T_1(\pi)$ and the corresponding riTDRL $\rho_j \in \mathcal{M}_{\text{iTDRL}}$ is constructed as defined in the proof of Lemma 4.5. This iterative procedure gives the scenario $(\rho_1, \ldots, \rho_t)$ for ι and π which is returned in Line 20.

Example 4.3. *Consider the permutation* $\pi = (-3\ -1\ 5\ -2\ 4)$ *that is investigated in Example 4.2. Recall that the maximal increasing substring decomposition of* π *is* $\pi_1\pi_2$*, where* $\pi_1 = -3 -1\ 5$ *and* $\pi_2 = -2\ 4$*, and that* $d_{\mathcal{M}_{iTDRL}}(\iota, \pi) = 2$*. Thus, exactly two iTDRLs are necessary to obtain* π *from* ι*. To obtain these iTDRLs, Algorithm 1 computes two transformations of* T_1 *or* T_2 *that are necessary (and sufficient) to transform* π *into* ι*. By Algorithm 1 the first transformation that has to be applied to* π *is*

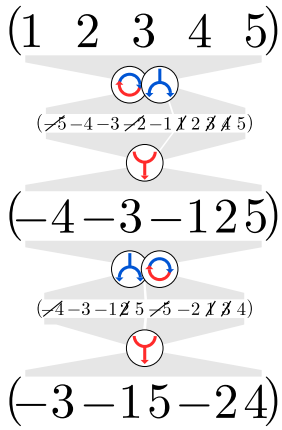


Figure 4.3: Parsimonious sorting scenario for ι and $(-3\ -1\ 5\ -2\ 4)$ under $\mathcal{M}_{\text{iTDRL}}$ that is computed by Algorithm 1. The first iTDRL that is applied to ι is the liTDRL $\rho_{\text{liTDRL}}(\ell, \{1,3,4\}, \{2,5\})$. The second iTDRL that is applied to the immediate permutation $(4\ 3\ 1\ 2\ 5)$ is the riTDRL $\rho_{\text{riTDRL}}(r, \{3,1,5\}, \{2,4\})$. Example 4.3 illustrates the computation of both iTDRLs.

T_1, *which gives* $T_1(\pi) = \pi_1 \oplus \overline{\pi_2} = (-4\ -3\ -1\ 2\ 5)$. *The corresponding iTDRL that reverses the application of* T_1 *is* $\rho_{riTDRL}(r, \{3,1,5\}, \{2,4\})$. *The permutation* $T_1(\pi)$ *has one maximal increasing substring and it holds that* $f_{T_1(\pi)} = -4 < 0$. *By Algorithm 1 the next transformation that has to be applied to* $T_1(\pi)$ *is* T_2. *The application of* T_2 *to* $T_1(\pi)$ *gives* $T_2(T_1(\pi)) = \overline{\upsilon_1} \oplus \kappa_1 = \overline{-4\,-3\,-1} \oplus 2\,5 = 1\,3\,4 \oplus 2\,5 = (1\ 2\ \ldots\ n)$. *The iTDRL that reverses this transformation is* $\rho_{liTDRL}(\ell, \{1,3,4\}, \{2,5\})$. *Consequently, the sequence* $(\rho_{liTDRL}(\ell, \{1,3,4\}, \{2,5\}), \rho_{riTDRL}(r, \{3,1,5\}, \{2,4\}))$ *transforms* ι *into* π, *see Figure 4.3 for an illustration.*

For a runtime analysis of Algorithm 1 consider $\pi \in s\mathcal{P}_n$. Certainly, the verification whether $\pi = \iota$ (Line 1), the computation of $d_{\mathcal{M}_{\text{iTDRL}}}(\iota, \pi)$ (lines 3–6), the computation of the maximal increasing substring decomposition (Line 8), the construction of $T_1(\pi)$ and $T_2(\pi)$ (lines 14, 17, and Line 10), and the construction of ρ_j (lines 11, 15, 18) can be done in time $\mathcal{O}(n)$. Therefore, lines 8 to 19 are executed in time $\mathcal{O}(n)$ and they are executed at most $\lceil \log_2 |\mathcal{S}(\pi)| \rceil + 1$ times. Since $|\mathcal{S}(\pi)| \leqslant n$ it follows that Algorithm 1 has a runtime in $\mathcal{O}(n \log n)$.

Algorithm 1 is implemented in `C++` and it is freely available on `http://pacosy.informatik.uni-leipzig.de/spitdrl`.

4.2 IMPACT ON A GENERAL MODEL FOR MITOCHONDRIAL EVOLUTION

The assumption that only one type of rearrangement, e. g., only iTDRLs, has been occurring during the evolution of mitochondrial gene orders is certainly unrealistic. Instead, mitochondrial genome comparisons provide strong evidence (see Section 2.1) that at least four major types of rearrangements are relevant for the evolution of mitochondrial genomes: inversions, transpositions, inverse transpositions, and tandem duplication random losses (Bernt et al., 2013b; Boore, 1999). The rearrangement model $\mathcal{M}_{4\text{-type}}$ that contains all these

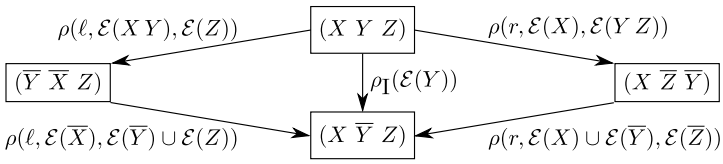


Figure 4.4: Inversion mimicked by either a sequence of two iTDRLs or a single iTDRL. The application of a single rearrangement is illustrated by a black arrow. A permutation $(\pi(1)\ \ldots\ \pi(n))$ with $\pi(1)\,\pi(2)\,\ldots\,\pi(n) = XYZ$, i. e., X, Y, and Z are substrings of π, is denoted by $(X\ Y\ Z)$. The inversion $\rho_I(\mathcal{E}(Y))$ applied to (XYZ) gives the permutation $(X\overline{Y}Z)$. The inversion can be mimicked by $\rho(\ell, \mathcal{E}(XY), \mathcal{E}(Z)$ followed by $\rho(\ell, \mathcal{E}(\overline{X}), E(\overline{Y}) \cup \mathcal{E}(Z))$ and $\rho(r, \mathcal{E}(X), \mathcal{E}(YZ))$ followed by $\rho(r, \mathcal{E}(X) \cup \mathcal{E}(\overline{Y}), \mathcal{E}(\overline{Z}))$. Observe that if X (or Z) is empty, then the second iTDRL in both sequences is an identity mapping. Hence, the first iTDRL in the respective sequence and the inversion have the same effect.

four types of rearrangements is called the 4-type rearrangement model. Although the results in Section 5.2 show that the distance problem and the sorting problem (for unsigned linear permutations) under $\mathcal{M}_{\text{4-type}}$ are tractable for large classes of permutations, a definite proof for all unsigned linear permutations has not been proposed. However, the scenarios that are obtained by solving the respective problem for the same permutations under $\mathcal{M}_{\text{iTDRL}}$ can be used to obtain approximate solutions. This section demonstrates a method to obtain such approximate solutions. In particular, Section 4.2.1 shows that the $\mathcal{M}_{\text{iTDRL}}$ distance for signed permutations provides bounds on the $\mathcal{M}_{\text{4-type}}$ distance. Furthermore, this section imparts how an approximated sequence of 4-type rearrangements can be obtained from a parsimonious scenario of iTDRL rearrangements.

4.2.1 *Bounding the Distance Problem under Major Mitochondrial Rearrangements*

Recall the formal definitions of the four major rearrangement operations relevant for mitochondrial gene orders given in Section 2.2.3. Nevertheless, these definitions shall be briefly recalled here.

Let π be a signed linear permutation, and let L and R be two character disjoint subsequences of π such that $\mathcal{E}(L) \cup \mathcal{E}(R) = \mathcal{E}(\pi)$. Recall that an interval $X \in I(\pi)$ is defined as a set of (unsigned) elements that occur consecutively in π. Moreover, let X and Y be two intervals of π, i. e., $X, Y \in I(\pi)$ and $X \cup Y$ or $Y \cup X$ is an interval of π. The *inversion* $\rho_I(X)$ applied to π reverses the order and it toggles the sign of every element of X. The *inverse transposition* $\rho_{iT}(X, Y)$ applied to π exchanges the order of X and Y and, in addition, it reverses the order and toggles the sign of every element in X. The *TDRL* $\rho_{TDRL}(\mathcal{E}(L), \mathcal{E}(R))$ applied to π duplicates π in tandem, followed by the loss of all elements of L (respectively R) in the left (respectively right) copy of the duplicated intermediate. A *transposition* $\rho_T(X, Y)$ applied to π swaps the order of X and Y. It is not hard to see that a transposition is a special case of the TDRL rearrangement.

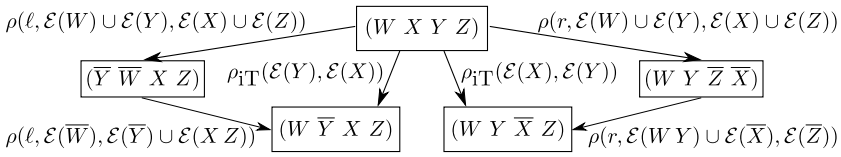


Figure 4.5: Inverse transposition mimicked by either a sequence of two iTDRLs or a single iTDRL. Notation is as in Figure 4.4. The application of $\rho_{\mathrm{iT}}(\mathcal{E}(Y), \mathcal{E}(X))$ (respectively $\rho_{\mathrm{iT}}(\mathcal{E}(X), \mathcal{E}(Y))$) to a permutation $(W\,X\,Y\,Z)$, where W, X, Y, and Z denote consecutive substrings, generates the permutation $(W\,\overline{Y}\,X\,Z)$ (respectively $(W\,Y\,\overline{X}\,Z)$). The same output permutation can be obtained by the application of $\rho(\ell, \mathcal{E}(W) \cup \mathcal{E}(Y), \mathcal{E}(X) \cup \mathcal{E}(Z))$ followed by $\rho(\ell, \mathcal{E}(\overline{W}), \mathcal{E}(\overline{Y}) \cup \mathcal{E}(X\,Z))$ (respectively $\rho(r, \mathcal{E}(W) \cup \mathcal{E}(Y), \mathcal{E}(X) \cup \mathcal{E}(Z))$ followed by $\rho(r, \mathcal{E}(W\,Y) \cup \mathcal{E}(\overline{X}), \mathcal{E}(\overline{Z}))$). Observe that if W (respectively Z) is empty, then the second iTDRL in the respective sequence is the identity mapping and, thus, the inverse transposition has the same effect as the first iTDRL of the corresponding sequence.

Consider a minimum length scenario S of iTDRLs that is computed by Algorithm 1. Scenario S can be used to obtain an approximate solution of the sorting problem under $\mathcal{M}_{4\text{-type}}$. To see this, the fundamental idea is to realize the connection between iTDRLs and 4-type rearrangements:

1) every iTDRL can be mimicked by one or two 4-type rearrangements, and

2) every 4-type rearrangement can be mimicked by one or two iTDRLs.

On one hand, every iTDRL has either the same effect as an inversion (illustrated in Figure 4.4) or an inverse transposition (illustrated in Figure 4.5), or it can be mimicked by both, a TDRL followed by an inversion, as well as a TDRL followed by an inverse transposition (illustrated Figure 4.6). On the other hand, an inversion and an inverse transposition have either the same effect as an iTDRL, or they can be mimicked by two iTDRLs, see Figure 4.4 and Figure 4.5, respectively. A TDRL (and therefore also transposition) can always be mimicked by two iTDRLs (illustrated Figure 4.7).

The fact that every iTDRL can be mimicked with at most two 4-type rearrangements implies that the $\mathcal{M}_{4\text{-type}}$ distance is less than twice the $\mathcal{M}_{\mathrm{iTDRL}}$ distance, i. e., $d_{\mathcal{M}_{4\text{-type}}}(\pi, \sigma) \leqslant 2d_{\mathcal{M}_{\mathrm{iTDRL}}}(\pi, \sigma)$, where $\pi, \sigma \in s\mathcal{P}_n$, and $d_{\mathcal{M}_{4\text{-type}}}(\pi, \sigma)$ denotes the $\mathcal{M}_{4\text{-type}}$ rearrangement distance for π and σ. In addition, the fact that every 4-type rearrangement can be mimicked by at most two iTDRLs implies $d_{\mathcal{M}_{\mathrm{iTDRL}}}(\pi, \sigma) \leqslant 2d_{\mathcal{M}_{4\text{-type}}}(\pi, \sigma)$. Dividing the latter inequality by two gives $d_{\mathcal{M}_{\mathrm{iTDRL}}}(\pi, \sigma)/2 \leqslant d_{\mathcal{M}_{4\text{-type}}}(\pi, \sigma)$. Combining this inequality with $d_{\mathcal{M}_{4\text{-type}}}(\pi, \sigma) \leqslant 2d_{\mathcal{M}_{\mathrm{iTDRL}}}(\pi, \sigma)$ gives the following proposition.

Proposition 4.3. *For each pair of signed linear permutations of size* $n \in \mathbb{N}$ π *and* σ *it holds that:*

$$\frac{d_{\mathcal{M}_{iTDRL}}(\pi, \sigma)}{2} \leqslant d_{\mathcal{M}_{4\text{-type}}}(\pi, \sigma) \leqslant 2d_{\mathcal{M}_{iTDRL}}(\pi, \sigma).$$

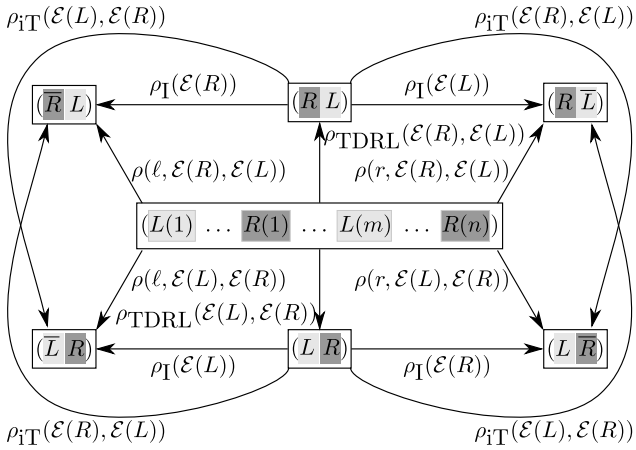


Figure 4.6: Inverse tandem duplication random loss mimicked by TDRL and inversion or inverse transposition. The notation is as in Figure 4.4, where in addition $L = L(1)\dots L(m)$ and $R = R(1)\dots R(n)$, $n, m \in \mathbb{N}$, are disjoint subsequences of a permutation of size $n+m$. The iTDRL $\rho(d,\mathcal{E}(L),\mathcal{E}(R))$ (respectively $\rho(d,\mathcal{E}(R),\mathcal{E}(L))$) can be replaced by applying (to the same permutation) the TDRL $\rho_{\mathrm{TDRL}}(\mathcal{E}(L),\mathcal{E}(R))$ (respectively $\rho_{\mathrm{TDRL}}(\mathcal{E}(R),\mathcal{E}(L))$), followed by the inversion $\rho_{\mathrm{I}}(\mathcal{E}(R))$ if $d = r$, or $\rho_{\mathrm{I}}(\mathcal{E}(L))$ if $d = \ell$ (respectively $\rho_{\mathrm{I}}(\mathcal{E}(R))$ if $d = \ell$ or $\rho_{\mathrm{I}}(\mathcal{E}(L))$ if $d = r$). Alternatively, the iTDRL $\rho(d,\mathcal{E}(L),\mathcal{E}(R))$ (respectively $\rho(d,\mathcal{E}(R),\mathcal{E}(L))$) can be mimicked by applying the TDRL $\rho_{\mathrm{TDRL}}(\mathcal{E}(R),\mathcal{E}(L))$ (respectively $\rho_{\mathrm{TDRL}}(\mathcal{E}(L),\mathcal{E}(R))$), followed by the inverse transposition $\rho_{\mathrm{iT}}(\mathcal{E}(L),\mathcal{E}(R))$ if $d = \ell$, or $\rho_{\mathrm{iT}}(\mathcal{E}(R),\mathcal{E}(L))$ if $d = r$ (respectively $\rho_{\mathrm{iT}}(\mathcal{E}(R),\mathcal{E}(L))$ if $d = \ell$, or $\rho_{\mathrm{iT}}(\mathcal{E}(L),\mathcal{E}(R))$ if $d = r$).

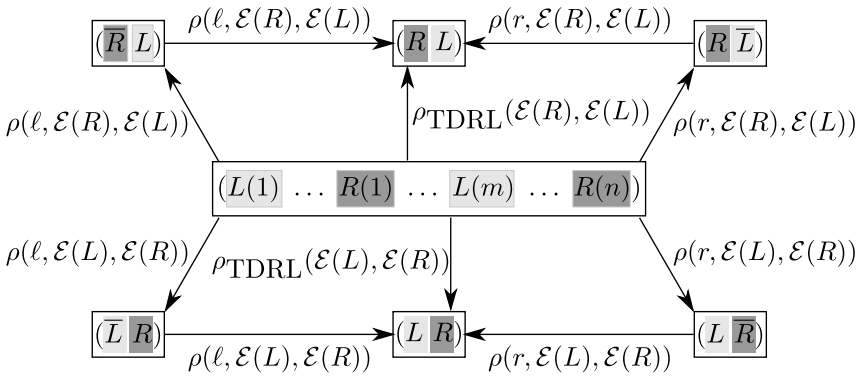


Figure 4.7: TDRL mimicked by two iTDRLs. Notation is as in Figure 4.6. The TDRL $\rho_{\mathrm{TDRL}}(\mathcal{E}(L),\mathcal{E}(R))$ (respectively $\rho_{\mathrm{TDRL}}(\mathcal{E}(R),\mathcal{E}(L))$) can be mimicked by iteratively applying two times $\rho(d,\mathcal{E}(L),\mathcal{E}(R))$ (respectively $\rho(d,\mathcal{E}(R),\mathcal{E}(L))$), where $d \in \{\ell, r\}$.

Proposition 4.3 states that the $\mathcal{M}_{\text{iTDRL}}$ distance is a 2-approximation for the $\mathcal{M}_{\text{4-type}}$ distance. In addition, an approximated sequence of 4-type rearrangements sorting π to σ can also be obtained by replacing every iTDRL by either an inversion, an inverse transposition, a TDRL and an inversion, or a TDRL and an inverse transposition as explained in figures 4.4 to 4.7. The following example illustrates such a replacement.

Example 4.4. *Consider the scenario for ι and $\pi = (-3\ -1\ 5\ -2\ 4)$ that is illustrated in Figure 4.3. Both iTDRLs can be replaced by a TDRL and an inversion or an inverse transposition. For example, the iTDRL $\rho_{liTDRL}(\ell, \{1,3,4\}, \{2,5\})$ can be mimicked by $\rho_{TDRL}(\{1,3,4\}, \{2,5\})$ followed by $\rho_I(\{1,3,4\})$. An example for a sequence of 4-type rearrangements that mimics the second iTDRL $\rho_{riTDRL}(r, \{3,1,5\}, \{2,4\})$ is $\rho_{TDRL}(\{4,2\}, \{3,1,5\})$ followed by $\rho_{iT}(\{4,2\}, \{3,1,5\})$. Combining both replacements yields the scenario $S = (\rho_{TDRL}(\{1,3,4\}, \{2,5\}), \rho_I(\{1,3,4\}), \rho_{TDRL}(\{4,2\}, \{3,1,5\}), \rho_{iT}(\{4,2\}, \{3,1,5\}))$ for ι and π under $\mathcal{M}_{\text{4-type}}$. However, scenario S is not parsimonious. This can be seen since the sequence $S' = (\rho_I(\{1,2,3\}), \rho_{TDRL}(\{3,1,5\}, \{2,4\}))$ is also a scenario for ι and π under $\mathcal{M}_{\text{4-type}}$, and the fact that there does not exist a single 4-type rearrangement that can transform ι into π. For more information on algorithms that compute such parsimonious scenarios see Chapter 5.*

4.3 CONCLUSION

In this chapter, the problem has been studied of computing a minimum length scenario (and its length) of inverse tandem duplication random loss rearrangements (iTDRLs) that are necessary to transform one given signed linear permutation into another given signed permutation. This problem is interesting since such a scenario allows to draw conclusions on the evolution of gene orders of unichromosomal genomes, e. g., mitochondrial gene orders. The reason is that signed linear permutations are commonly used as a formal model for such gene orders. In addition, the iTDRL rearrangement has currently been suggested to be a potential evolutionary mechanism in mitochondrial genomes. It was shown that the $\mathcal{M}_{\text{iTDRL}}$ distance – the minimum number of iTDRLs needed to transform one permutation into another – can be computed in linear time. Moreover, it was shown that a corresponding scenario can be obtained in quasilinear time. In addition, a closed formula has been determined for the maximum $\mathcal{M}_{\text{iTDRL}}$ distance for two permutations of a certain size. It was proven that every type of major mitochondrial rearrangement (respectively every iTDRL) can be mimicked by at most two iTDRLs (respectively major mitochondrial rearrangements). Taking advantage of this characteristic and the fact that the distance problem with respect to iTDRLs is computationally tractable, it has been shown that the $\mathcal{M}_{\text{iTDRL}}$ distance is a 2-approximation for the $\mathcal{M}_{\text{4-type}}$ distance, which is the minimum number of inversions, transpositions, inverse transpositions, and TDRLs necessary to transform one given permutation into another given permutation.

5

ALGORITHMS FOR SORTING BY MITOCHONDRIAL REARRANGEMENTS

The assumption that only one type of rearrangement has been occurring during the evolution of certain gene orders is most likely unrealistic. This holds especially for mitochondrial genomes where the major forces that shape gene arrangements are inversions, transpositions, inverse transpositions, and tandem duplication random losses (TDRLs) (Boore, 1999). In order to compute realistic scenarios of gene order evolution, it is crucial to base reconstructions on a suitable rearrangement model that reflects the prevalent biological conditions. Such conditions may be the frequency of different types of evolutionary mechanisms, the preservation of certain groups of genes, and the evolutionary mechanisms itself. The use of a suitable rearrangement model becomes even more important when parsimonious scenarios of mitochondrial gene order evolution are used to provide evidence for phylogenetic hypothesis as it is often done in the biological literature, e. g., see Bleidorn et al. (2007) and Tan et al. (2018) and many other references listed in Section 2.1.3.

However, despite the fact that a better understanding of the rearrangement model that considers all predominant mitochondrial genome rearrangements (i. e., the model $\mathcal{M}_{4\text{-type}}$) is indispensable for the study of mitochondrial gene orders, only little is known about it. In particular, the fundamental genome rearrangement problems (see Section 2.2.4) under $\mathcal{M}_{4\text{-type}}$ have not be investigated up to now and only two algorithms provide parsimonious rearrangement scenarios for pairs of gene orders under $\mathcal{M}_{4\text{-type}}$. Those algorithms are the `CREx` heuristic (Bernt et al., 2007) and the integer linear program proposed in Lancia et al. (2015). While the scenarios obtained by `CREx` are not guaranteed to be optimal, it preserves certain gene clusters of the given gene orders. However, the heuristic assumes that all rearrangement types are equally likely which may limit the biological practicality especially since analyses from Bernt and Middendorf (2011) indicate that the different types of rearrangements in *Protostomia* (and also *Metazoa*) occur with significant differences: 20% inversions, 80% transpositions, 10% inverse transpositions, and 10% TDRLs. The integer linear program that has been proposed in Lancia et al. (2015) considers neither gene orientation, i. e., it considers pairs of unsigned permutations, nor the preservation of gene clusters. In addition, it uses an exponential number of variables which limits the approach to be used for problem instances of small size. (See Section 2.3.6 for more details on both algorithms.) Consequently, the existing techniques to compute rearrangement scenarios under $\mathcal{M}_{4\text{-type}}$ either reflect biological conditions of mitochondrial genomes only to a limited degree or are computationally demanding.

This chapter addresses several of the latter limitations by investigating the fundamental genome rearrangement problems for pairs of signed permutations under the $\mathcal{M}_{4\text{-type}}$ rearrangement model. Section 5.2 investigates the sorting problem (and the distance problem) for signed linear permutations under $\mathcal{M}_{4\text{-type}}$. It is shown that the $\mathcal{M}_{4\text{-type}}$ model is only insufficiently suitable to obtain biological relevant rearrangement scenarios. This is because parsimonious rearrangement scenarios contain almost entirely rearrangements of type TDRL. As a consequence, two biological motivated variants of the sorting problem under $\mathcal{M}_{4\text{-type}}$ are studied in Section 5.3 and Section 5.4. The first variant considers individual weights for each type of rearrangement to reflect different relative likelihoods for inversions, transpositions, inverse transpositions, and TDRLs. Recall that for a weighted rearrangement model, the sorting problem becomes the *weighted sorting problem* where a minimum weight scenario is sought instead of a scenario having the minimum length. Likewise, the distance problem becomes the *weighted distance problem*, see Section 2.2.4 for a formal definition. In Section 5.3 the weighted sorting problem (i. e., the first variant of the sorting problem) is solved exactly by using integer linear programming. The second variant of the sorting problem extends the first one by explicitly enforcing genome rearrangement scenarios to preserve certain groups of genes that occur in both considered gene orders in close proximity. Those groups of genes are represented formally by the notion of common intervals of permutations, see Section 2.2.2. The exact dynamic programming algorithm `CREx2` that solves this problem efficiently for large classes of problem instances is proposed in Section 5.4. To show that `CREx2` is able to reliably reconstruct gene order rearrangement scenarios, it is evaluated on artificial and biological gene order data sets in Section 5.5. In particular, the fundamental contributions of this chapter are:

- A sharp lower bound and several close additive upper bounds on the $\mathcal{M}_{4\text{-type}}$ distance are proposed for the first time (Theorem 5.1).
- An approximation algorithm solving the sorting problem for signed linear permutations under $\mathcal{M}_{4\text{-type}}$ is described. The algorithm guarantees to compute a rearrangement scenario that deviates from a parsimonious scenario in at most two rearrangements (Algorithm 2).
- The polynomial-size integer linear program `GeRe-ILP` is described (Section 5.3). `GeRe-ILP` solves the weighted sorting problem for signed linear permutations under $\mathcal{M}_{4\text{-type}}$ in which every type of rearrangement can be weighted arbitrarily.
- The tool `CREx2` is described (Algorithm 3). `CREx2` solves the weighted sorting problem for signed linear permutations under $\mathcal{M}^{p}_{4\text{-type}}$, where a weight can be assigned to each type of rearrangement and the common intervals of the considered permutations are preserved.

The chapter is organized as follows. Section 5.1 recalls the most important definitions that are used throughout the chapter. The sorting problem and the distance problem for signed linear permutations under $\mathcal{M}_{4\text{-type}}$ is investigated in Section 5.2. Section 5.3 proposes the integer linear program `GeRe-ILP`. The algorithm `CREx2` is presented in Section 5.4 and evaluated on simulated and biological gene order data in Section 5.5. Conclusions are drawn in Section 5.6.

5.1 BASIC DEFINITIONS AND PRELIMINARIES

In this section, the formal definitions relevant for investigating the characteristics of the $\mathcal{M}_{4\text{-type}}$ model under several biological conditions are briefly recalled (for an extensive formal introduction with references and examples see Section 2.2).

In the following, *signed linear permutations* are used as a formal model for gene orders in which each element represents a unique gene and the sign represents its orientation. Recall that the set of all signed permutations of size n is denoted by $s\mathcal{P}_n$. An *interval* of a permutation π is a non-empty set of (unsigned) elements that are consecutive in π. $I(\pi)$ is the set of all intervals of π.

Evolutionary events that change the order of genes of some considered unichromosomal genomes are modeled by rearrangement operations. In particular, this chapter considers the major rearrangement operations that occur in metazoan mitochondrial genomes: inversions, transpositions, inverse transpositions, and tandem duplication random losses (TDRLs). Let π be a signed linear permutation. Furthermore, let X and Y denote disjoint and consecutive intervals of π, i. e., $X, Y \in I(\pi)$, $X \cup Y \in I(\pi)$, and $X \cap Y = \emptyset$. An *inversion* $\rho_I(X)$ for π reverses the order and it switches the sign of every element within an interval X. A *transposition* $\rho_T(X, Y)$ for π switches the order of the intervals X and Y of π. An *inverse transposition* $\rho_{iT}(X, Y)$ for π switches the order of the intervals X and Y of π and, in addition, it reverses the order and switches the sign of every element in X. Let L and R be a bipartition of $[1:n]$, i. e., $L \cup R = [1:n]$ and $L \cap R = \emptyset$. Finally, a *tandem duplication random loss* $\rho_{TDRL}(L, R)$ for π duplicates π such that the duplicate is placed adjacently to π, followed by the loss of every element contained in L (respectively R) in the left (respectively right) copy of π. Observe that this TDRLs definition reflects the fact that every TDRL that duplicates a permutation only partially can easily be represented by a TDRL that duplicates the complete permutation as explained in Section 2.3.4.

Gene clusters of a set of gene orders are modeled by common intervals of a set of (signed) permutations. Consider a set of permutations $\Pi \subseteq s\mathcal{P}_n$. A *common interval* of Π is a subset of (unsigned) elements of the permutations within Π that is an interval in each permutation of Π. With $C(\Pi)$ the set of all common intervals of Π is denoted. Recall that the preservation of common intervals in the construction of parsimonious rearrangement scenarios (for two given permutations) is realized by restricting the considered model to contain only those

rearrangements that do not break any common interval of the considered permutations. Such rearrangements are called *preserving* and the genome rearrangement problems that are defined for the preservation of common intervals are called *preserving genome rearrangement problems*. This chapter considers the preserving sorting problem under the model $\mathcal{M}^{p}_{4\text{-type}}$ which includes all preserving inversions, transpositions, inverse transpositions, and TDRLs in Section 5.4.

The *weight* of a rearrangement in $\mathcal{M} \in \{\mathcal{M}_{4\text{-type}}, \mathcal{M}^{p}_{4\text{-type}}\}$ is given by a weight function $\omega : \mathcal{M} \to \mathbb{R}_{>0}$. In this chapter, the weight of a rearrangement is determined by its type $Z \in \{I, T, iT, TDRL\}$ and the corresponding weight is denoted by ω_Z. The *weight* of a sequence (scenario) S for two permutations π and σ is given by the sum of the weights of its rearrangements. A scenario for π and σ with minimum weight is called *parsimonious*.

5.2 EXPLORING THE 4-TYPE REARRANGEMENT MODEL

This section considers the distance problem (and also the sorting problem) for two signed permutations under $\mathcal{M}_{4\text{-type}}$. Despite the fact that this rearrangement model represents the major rearrangement events that occur during the evolution of metazoan mitochondrial gene orders (see Section 2.1.3), both problems have – up to now – not been investigated formally. However, as the introduction outlines, a better understanding of the $\mathcal{M}_{4\text{-type}}$ rearrangement model is indispensable for the study of mitochondrial gene orders. Therefore, Section 5.2.1 presents a sharp lower and several close upper bounds on the distance problem for signed linear permutations under $\mathcal{M}_{4\text{-type}}$. An approximation algorithm that satisfies those upper bounds is proposed in Section 5.2.2. The insights gained in this section are important for biological applications as they show that a parsimonious rearrangement scenario under $\mathcal{M}_{4\text{-type}}$ predominantly contains rearrangements of type TDRL. The practical consequences of the theoretical results for mitochondrial gene order data are discussed in Section 5.2.3.

5.2.1 *Bounding the 4-type Rearrangement Distance*

In this section, the $\mathcal{M}_{4\text{-type}}$ distance for signed linear permutations is investigated. For this task the identity permutation ι and an arbitrary permutation π are considered instead of two arbitrary permutations. Recall that this is sufficient due to the left-invariance of linear permutations (see Section 2.2.1). As the main result of this section, bounds on the $\mathcal{M}_{4\text{-type}}$ distance are determined by a characterization of permutations that can be generated by sequentially applying $k \in \mathbb{N}$ 4-type rearrangements to the identity permutation ι. This characterization utilizes the number of *maximal increasing substrings* of a permutation, which is an interval of the considered permutation that cannot be extended to the left or the right without violating the condition that – from left to right – all elements increase.

The chain of evidence presented in this section is performed by using similar techniques as in Section 4.2: A lower bound on the (minimum) number of $k \in \mathbb{N}$ 4-type rearrangements that are necessary to produce a permutation with a certain number of maximal increasing substrings is determined in Proposition 5.1. A corresponding upper bound is proposed in Proposition 5.2. Finally, the bounds on the $\mathcal{M}_{4\text{-type}}$ distance are derived by combining the lower and the upper bound in Theorem 5.1.

The Lower Bound

This subsection provides a lower bound on the number of required 4-type rearrangements that transform the identity permutation ι into a permutation with a certain number of maximal increasing substrings. To determine this bound, it is helpful to investigate to which extent the different types of 4-type rearrangements can influence the maximal increasing substrings of permutation. The following lemma gains such insights for inversions and inverse transpositions. It states that the number of *descents*, i. e., the number of positions $i \in [1:n-1]$ of a permutations $\pi \in s\mathcal{P}_n$ for which $\pi(i) > \pi(i+1)$, can be increased by at most one, using an inversion, or by at most two, using an inverse transposition. It is worth mentioning that the proof of the following lemma is similar to the proof of Proposition 9.3 from Bóna (2004).

Lemma 5.1. *The following statements are true:*

i) No inversion can increase the number of descents of a signed linear permutation by more than one.

ii) No inverse transposition can increase the number of descents of a signed linear permutation by more than two.

Proof. Let $\pi = (\pi(1)\ \pi(2)\ \dots\ \pi(n))$ be a signed linear permutation of size $n \in \mathbb{N}$.

(i): Let $X = \pi(i+1)\ \dots\ \pi(j)$, $1 \leqslant i+1 \leqslant j \leqslant n$ denote the substring of π that is affected by an inversion ρ_I, i. e.,

$$\rho_I \circ \pi = (\pi(1)\ \dots\ \pi(i)\ -\pi(j)\ \dots\ -\pi(i+1)\ \pi(j+1)\ \dots\ \pi(n)).$$

It is easy to see that the only positions in which an *ascent* (i. e., a position $1, \dots, n-1$ that is not a descent) can be turned into a descent may be positions i and j. Note that all ascents within X remain unchanged. In order to increase the number of descents by two, such a change has to occur in both positions i and j. This can only be the case if the following inequalities hold:

$$\begin{aligned} \pi(i) &< \pi(i+1), & \pi(i) &> -\pi(j), \\ \pi(j) &< \pi(j+1), & -\pi(i+1) &> \pi(j+1). \end{aligned}$$

Observe that the left (right) inequalities satisfy that positions i and j are ascents (respectively descents) in π (respectively $\rho_I \circ \pi$). However, this is not possible since it would imply:

$$\pi(i) < \pi(i+1) < -\pi(j+1) < -\pi(j) < \pi(i).$$

Thus, an inversion can increase the number of descents by at most one.

(ii): Let $X = \pi(i+1) \dots \pi(j)$ and $Y = \pi(j+1) \dots \pi(k)$ with $1 \leqslant i+1 \leqslant j < j+1 \leqslant k \leqslant n$ denote the consecutive substrings of π that are affected by an inverse transposition ρ_{iT}, i. e., permutation $\rho_{iT} \circ \pi$ is either:

$$(\pi(1) \dots \pi(i) \; -\pi(k) \dots -\pi(j+1) \; \pi(i+1) \dots \pi(j) \; \pi(k+1) \dots \pi(n)) \text{ or}$$
$$(\pi(1) \dots \pi(i) \; \pi(j+1) \dots \pi(k) \; -\pi(j) \dots -\pi(i+1) \; \pi(k+1) \dots \pi(n)).$$

For the sake of a clear argument, only the first equation is considered. (The proof for the second equation is fully analogous.) The only positions in which an ascent can be turned into a descent by an inverse transposition are positions i, j, and k. In order to increase the number of descents by three, such a change would have to occur in each of the three positions. This can only be the case if the following inequalities hold:

$$\begin{aligned} \pi(i) &< \pi(i+1), & \pi(i) &> -\pi(k), \\ \pi(j) &< \pi(j+1), & -\pi(j+1) &> \pi(i+1), \\ \pi(k) &< \pi(k+1), & \pi(j) &> \pi(k+1). \end{aligned}$$

However, this is not possible since it would imply:

$$\pi(i) < \pi(i+1) < -\pi(j+1) < -\pi(j) < -\pi(k+1) < -\pi(k) < \pi(i).$$

Thus, an inverse transposition can increase the number of descents by at most two and the lemma follows. □

Since every descent of a permutation $\pi \in s\mathcal{P}_n$ is the position of the last element of a maximal increasing substring of π, it is easy to verify that the number of descents of π is one less than the number of maximal increasing substrings of π. Hence, the following corollary is an immediate consequence of Lemma 5.1.

Corollary 5.1. *The following statements are true:*

i) No inversion can increase the number of maximal increasing substrings of a signed linear permutation by more than one.

ii) No inverse transposition can increase the number of maximal increasing substrings of a signed linear permutation by more than two.

Using Corollary 5.1, the sought lower bound on the number of required 4-type rearrangements is determined in the following proposition. To understand Proposition 5.1, recall that the number of maximal increasing substrings of a permutation π is denoted by $|\mathcal{S}(\pi)|$.

Proposition 5.1. *For a permutation $\pi \in s\mathcal{P}_n$ that has been obtained from $\iota \in s\mathcal{P}_n$ by the application of $k \in \mathbb{N}_0$ 4-type rearrangements it holds that $|\mathcal{S}(\pi)| \leqslant 2^k$.*

Proof. The proposition is proven by induction on k. First consider the case $k \leqslant 1$. If $k = 0$ then the statement follows trivially from $|\mathcal{S}(\pi)| =$

$|\mathcal{S}(\iota)| = 1 = 2^0$. Now consider $k = 1$. If π is obtained by the application of an inversion or an inverse transposition then there exist two indices i and j with $1 \leqslant i \leqslant j \leqslant n$ such that $0 < \pi(1) < \ldots < \pi(i-1)$, and $\pi(i) < \ldots < \pi(j) < 0 < \pi(j+1) < \ldots < \pi(n)$. Hence, if $i = 1$ then it holds that $|\mathcal{S}(\pi)| = 1$ and, otherwise, it follows $|\mathcal{S}(\pi)| = 2$. Consequently, $|\mathcal{S}(\pi)| \leqslant 2$. Now consider that π is obtained by a TDRL. Then π can be expressed as $\tau\tau'$, where τ and τ' are disjoint subsequences of ι. Then, $|\mathcal{S}(\pi)| = |\mathcal{S}(\tau\tau')| \leqslant 2|\mathcal{S}(\iota)| = 2$ is obtained from Lemma 4.3. The base case follows by the fact that the transposition is a special case of the TDRL rearrangement.

For the induction step assume that $k \geqslant 2$. Now consider a permutation σ that has been obtained from ι by the application of $k-1$ rearrangements from $\mathcal{M}_{\text{4-type}}$. The permutation obtained from σ by the application of a single 4-type rearrangement ρ is denoted by π. There are four cases that have to be considered depending on the type of ρ. If ρ is an inversion (respectively inverse transposition) then Corollary 5.1.(i) (respectively Corollary 5.1.(ii)) implies $|\mathcal{S}(\pi)| = |\mathcal{S}(\rho \circ \sigma)| \leqslant |\mathcal{S}(\sigma)| + 1 \leqslant 2^{k-1} + 1 < 2^k$ (respectively $|\mathcal{S}(\pi)| = |\mathcal{S}(\rho \circ \sigma)| \leqslant |\mathcal{S}(\sigma)| + 2 \leqslant 2^{k-1} + 2 \leqslant 2^k$). If ρ is a TDRL, then there are two disjoint substrings τ and τ' of σ such that π can be expressed as $\tau\tau'$. By Lemma 4.3 it holds that $|\mathcal{S}(\pi)| = |\mathcal{S}(\rho \circ \sigma)| = |\mathcal{S}(\tau\tau')| \leqslant |\mathcal{S}(\tau)| + |\mathcal{S}(\tau')| \leqslant |\mathcal{S}(\sigma)| + |\mathcal{S}(\sigma)| = 2^k$. The statement follows by the fact that a transposition is a special case of a TDRL. □

The Upper Bound

In this subsection, several upper additive bounds are given on the minimum number of 4-type rearrangements that have to be applied to ι in order to produce a permutation with a certain number of maximal increasing substrings.

The main idea is to obtain these upper bounds by iteratively applying certain transformations to a given permutation π in order to reduce the number of its maximal increasing substrings. The result of this procedure is to obtain a sequence S of transformations that transform π into ι. These transformations are not chosen arbitrarily. Instead, transformations that invert 4-type rearrangements are applied. The idea is that a sequence of 4-type rearrangements that transforms ι into π can then be obtained from S by reversing its order and replacing every transformation by its corresponding inverted 4-type rearrangement.

The required transformations can be determined easily for inversions, transpositions, and inverse transpositions since these operations are reversible, i.e., $\rho \circ \rho \circ \pi = \pi$ if ρ is an inversion, transposition, or inverse transposition for π. A transformation that can invert a TDRL rearrangement is a *riffle shuffle* (Bayer and Diaconis, 1992). In the following, the focus lies on a certain riffle shuffle that is able to bisect the number of maximal increasing substrings of a permutation. More precisely, the transformation $T\colon s\mathcal{P}_n \to s\mathcal{P}_n$ is defined to construct a permutation $T(\pi)$ from π which has two properties:

1) there always exists a TDRL ρ_{TDRL} such that $\rho_{\text{TDRL}} \circ T(\pi) = \pi$ (see Lemma 5.2), and 2) $|\mathcal{S}(T(\pi))| = \lceil |\mathcal{S}(\pi)|/2 \rceil$ (see Lemma 5.4).

The following notations are needed. The *maximal increasing substring decomposition* of a permutation π is the unique list of pairwise disjoint maximal increasing substrings $\tau_1\tau_2\ldots\tau_{|\mathcal{S}(\pi)|}$ of π such that $\pi(1)\ldots\pi(n) = \tau_1\tau_2\ldots\tau_{|\mathcal{S}(\pi)|}$ and for all $1 \leqslant j \leqslant |\mathcal{S}(\pi)|$ it holds that $|\tau_j| \geqslant 1$. Let X be a subsequence of a permutation. The notation $\mathcal{E}(X)$ is used to refer to the set of absolute values of all elements of X. Consider two disjoint subsequences X and Y of a permutation π. By $X \oplus Y$ the sequence is denoted that is created by sorting the elements X and Y increasingly. For examples of the aforementioned definitions see Example 5.1, Figure 5.1, and Section 2.3.4.

Consider a signed linear permutation π of size n with the maximal substring decomposition $\pi = \pi_1\ldots\pi_{|\mathcal{S}(\pi)|}$. The transformation T is defined depending on whether the number of maximal increasing substrings of π is even or odd (see also Figure 5.1). If $|\mathcal{S}(\pi)|$ is even, then $T(\pi) := \tau_1\tau_2\ldots\tau_{|\mathcal{S}(\pi)|/2-1}\tau_{|\mathcal{S}(\pi)|/2}$, where

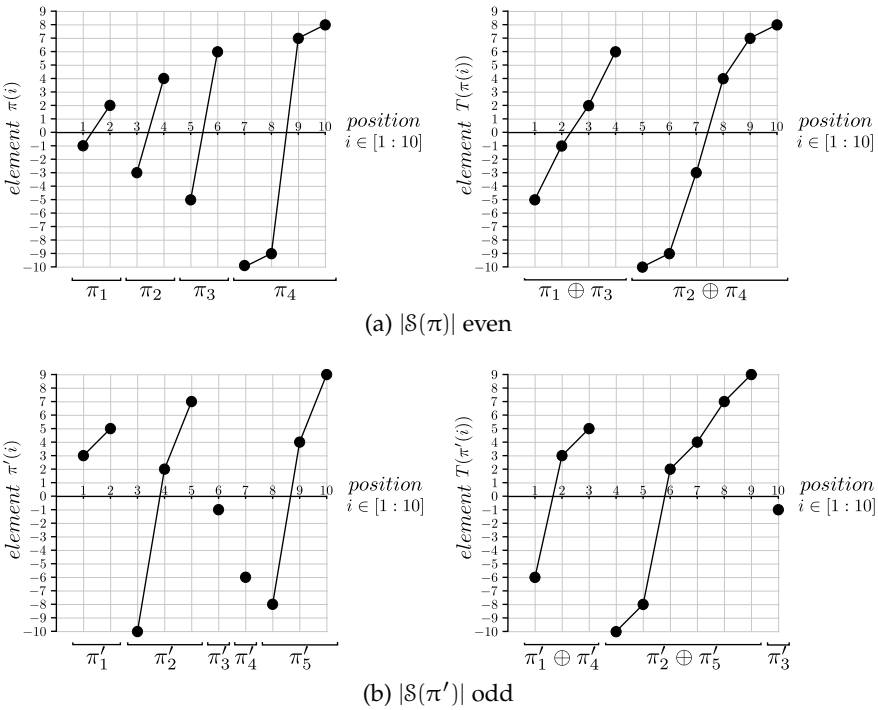
$$\begin{aligned}
\tau_1 &:= \pi_1 \oplus \pi_{|\mathcal{S}(\pi)|/2+1},\\
\tau_2 &:= \pi_2 \oplus \pi_{|\mathcal{S}(\pi)|/2+2},\\
&\vdots\\
\tau_{|\mathcal{S}(\pi)|/2-1} &:= \pi_{|\mathcal{S}(\pi)|/2-1} \oplus \pi_{|\mathcal{S}(\pi)|-1}, \text{ and}\\
\tau_{|\mathcal{S}(\pi)|/2} &:= \pi_{|\mathcal{S}(\pi)|/2} \oplus \pi_{|\mathcal{S}(\pi)|}.
\end{aligned}$$

If $|\mathcal{S}(\pi)|$ is odd, then $T(\pi) := \tau_1\tau_2\ldots\tau_{\lfloor|\mathcal{S}(\pi)|/2\rfloor}\tau_{\lceil|\mathcal{S}(\pi)|/2\rceil}$, where

$$\begin{aligned}
\tau_1 &:= \pi_1 \oplus \pi_{\lceil|\mathcal{S}(\pi)|/2\rceil+1},\\
\tau_2 &:= \pi_2 \oplus \pi_{\lceil|\mathcal{S}(\pi)|/2\rceil+2},\\
&\vdots\\
\tau_{\lfloor|\mathcal{S}(\pi)|/2\rfloor} &:= \pi_{\lfloor|\mathcal{S}(\pi)|/2\rfloor} \oplus \pi_{|\mathcal{S}(\pi)|}, \text{ and}\\
\tau_{\lceil|\mathcal{S}(\pi)|/2\rceil} &:= \pi_{\lceil|\mathcal{S}(\pi)|/2\rceil}.
\end{aligned}$$

Example 5.1. *Consider the permutation* $\pi = (-1\ 2\ -3\ 4\ -5\ 6\ -10\ -9\ 7\ 8)$. *The maximal increasing substring decomposition of* π *is* $\pi_1\pi_2\pi_3\pi_4$, *where* $\pi_1 = -1\ 2$, $\pi_2 = -3\ 4$, $\pi_3 = -5\ 6$, *and* $\pi_4 = -10\ -9\ 7\ 8$. *Since* $|\mathcal{S}(\pi)| = 4$ *is even, it follows that* $T(\pi) = \tau_1\tau_2$ *with* $\tau_1 = \pi_1 \oplus \pi_3 = -5\ -1\ 2\ 6$ *and* $\tau_2 = \pi_2 \oplus \pi_4 = -10\ -9\ -3\ 4\ 7\ 8$. *Now consider the permutation* $\pi' = (3\ 5\ -10\ 2\ 7\ -1\ -6\ -8\ 4\ 9)$ *which has the maximal increasing substring decomposition* $\pi'_1\pi'_2\pi'_3\pi'_4\pi'_5$, *where* $\pi'_1 = 3\ 5$, $\pi'_2 = -10\ 2\ 7$, $\pi'_3 = -1$, $\pi'_4 = -6$, *and* $\pi'_5 = -8\ 4\ 9$. *Since* $|\mathcal{S}(\pi')| = 5$ *is odd, it follows that* $T(\pi') = \tau'_1\tau'_2\tau'_3$, *where* $\tau'_1 = \pi'_1 \oplus \pi'_4 = -6\ 3\ 5$, $\tau'_2 = \pi'_2 \oplus \pi'_5 = -10\ -8\ 2\ 4\ 7\ 9$, *and* $\tau'_3 = \pi'_3 = -1$. *For an illustration see Figure 5.1.*

The following three auxiliary lemmata are needed to show the upper bound in Proposition 5.2. The first lemma states that the introduced transformation T (the riffle shuffle) is an inverse operation of a TDRL rearrangement.



(a) $|\mathcal{S}(\pi)|$ even

(b) $|\mathcal{S}(\pi')|$ odd

Figure 5.1: Illustration of Example 5.1: (a) π (left) and $T(\pi)$ (right); (b) π' (left) and $T(\pi')$ (right). Each dot represents an element (y-axis) of a permutation and its position (x-axis). Maximal increasing substrings are illustrated by continuous lines. For every permutation that is illustrated its maximal increasing substring decomposition is shown on the bottom of the respective subfigure.

Lemma 5.2. *For every signed linear permutation π of size $n \in \mathbb{N}$, the following statement is true. There exists a TDRL $\rho_{TDRL} \in \mathcal{M}_{4\text{-}type}$ such that $\rho_{TDRL} \circ T(\pi) = \pi$.*

Proof. Let $\pi \in s\mathcal{P}_n$ and let $\pi = \pi_1\pi_2 \dots \pi_{|\mathcal{S}(\pi)|}$ be the maximal increasing substring decomposition of π. Then, by definition of $T(\pi)$ it holds that $T(\pi) = \tau_1\tau_2 \dots \tau_{|\mathcal{S}(\pi)|/2-1}\tau_{|\mathcal{S}(\pi)|/2}$ (respectively $T(\pi) = \tau_1\tau_2 \dots \tau_{\lfloor|\mathcal{S}(\pi)|/2\rfloor}\tau_{\lceil|\mathcal{S}(\pi)|/2\rceil}$) in the case of $|\mathcal{S}(\pi)|$ being even (respectively odd). It follows that

$$L = \pi_1\pi_2 \dots \pi_{|\mathcal{S}(\pi)|/2-1}\pi_{|\mathcal{S}(\pi)|/2} \text{ and}$$
$$R = \pi_{|\mathcal{S}(\pi)|/2+1}\,\pi_{|\mathcal{S}(\pi)|/2+1} \dots \pi_{|\mathcal{S}(\pi)|-1}\,\pi_{|\mathcal{S}(\pi)|}$$

are disjoint subsequences of $T(\pi)$ in the case that $|\mathcal{S}(\pi)|$ is even. If otherwise $|\mathcal{S}(\pi)|$ is odd, then it follows that

$$L = \pi_1\pi_2 \dots \pi_{\lfloor|\mathcal{S}(\pi)|/2\rfloor}\pi_{\lceil|\mathcal{S}(\pi)|/2\rceil} \text{ and}$$
$$R = \pi_{\lceil|\mathcal{S}(\pi)|/2\rceil+1}\,\pi_{\lceil|\mathcal{S}(\pi)|/2\rceil+2} \dots \pi_{|\mathcal{S}(\pi)|-1}\pi_{|\mathcal{S}(\pi)|}$$

are disjoint subsequences of $T(\pi)$. Note that in both cases $\pi = LR$ holds true and L, R are disjoint subsequences of $T(\pi)$ with $L \cup R$ containing all elements of $T(\pi)$. Hence, for the TDRL $\rho_{TDRL}(\mathcal{E}(L), \mathcal{E}(R))$ it holds that $\rho_{TDRL} \circ T(\pi) = \pi$. □

Observe that the proof of Lemma 5.2 also defines the TDRL that, when applied to $T(\pi)$, reverses the effect of the transformation T on π:

For a signed permutation π with the maximal increasing substring decomposition $\pi = \pi_1 \ \dots \ \pi_{|\mathcal{S}(\pi)|}$ it holds that $\rho_{\text{TDRL}}(\mathcal{E}(L), \mathcal{E}(R)) \circ T(\pi) = \pi$, where $L = \pi_1 \ \dots \ \pi_{\lceil |\mathcal{S}(\pi)|/2 \rceil}$ and $R = \pi_{\lceil |\mathcal{S}(\pi)|/2 \rceil + 1} \ \dots \ \pi_{|\mathcal{S}(\pi)|}$. The following example illustrates this observation.

Example 5.2. *Consider the permutation* $\pi' = (3\ 5\ {-10}\ 2\ 7\ {-1}\ {-6}\ {-8}\ 4\ 9)$ *that is illustrated in Figure 5.1 (b). The example shows that the maximum increasing substring decomposition of* π' *is* $\pi'_1\pi'_2\pi'_3\pi'_4\pi'_5$ *with* $\pi'_1 = 3\,5$, $\pi'_2 = -10\,2\,7$, $\pi'_3 = -1$, $\pi'_4 = -6$, *and* $\pi'_5 = -8\,4\,9$. *In addition, it shows that* $T(\pi') = (-6\ 3\ 5\ {-10}\ {-8}\ 2\ 4\ 7\ 9\ {-1})$. *By Lemma 5.2 it follows that for the TDRL* $\rho_{TDRL}(\mathcal{E}(L), \mathcal{E}(R))$ *in which* $L = \pi'_1\pi'_2\pi'_3 = 3\,5\,{-10}\,2\,7\,{-1}$ *and* $R = \pi'_4\pi'_5 = -6\,{-8}\,4\,9$ *it holds that* $\rho_{TDRL}(\mathcal{E}(L), \mathcal{E}(R)) \circ T(\pi) = \pi$.

Transformation T is designed such that the following lemma holds.

Lemma 5.3. *Let* π *be a signed permutation of size* n. *For the permutation* $T(\pi)$ *the respective decomposition into strings* $\tau_1\tau_2\dots\tau_t$ *(where* t *is as in the respective case) is a maximal increasing substring decomposition.*

Proof. Let $\pi \in s\mathcal{P}_n$ and let $\pi = \pi_1 \dots \pi_{|\mathcal{S}(\pi)|}$ be the maximal increasing substring decomposition of π. For the proof it is sufficient to show that for each $j \in [1:t-1]$ it holds that the last element ℓ_{τ_j} of τ_j is larger than the first element $f_{\tau_{j+1}}$ of τ_{j+1}, i.e., $\ell_{\tau_j} > f_{\tau_{j+1}}$. By construction $T(\pi) := \tau_1\tau_2\dots\tau_{t-1}\tau_t$ with $t = |\mathcal{S}(\pi)|/2$ (respectively $t = \lceil |\mathcal{S}(\pi)|/2 \rceil$) in the case that $|\mathcal{S}(\pi)|$ is even (respectively odd). Since all elements of two sequences X and Y are sorted increasingly in $X \oplus Y$ it follows that every τ_j with $j \in [1:t]$ is an increasing substring. By the construction of $T(\pi)$ it holds that $\tau_j = \pi_j \oplus \pi_{|\mathcal{S}(\pi)|/2+j}$ for $j \in [1:\lfloor |\mathcal{S}(\pi)|/2 \rfloor]$ and $\pi_{\lceil |\mathcal{S}(\pi)|/2 \rceil} = \tau_{\lceil |\mathcal{S}(\pi)|/2 \rceil}$ if $|\mathcal{S}(\pi)|$ is odd. Hence, a τ_j always contains all elements of π_j for all $j \in [1:t]$. Consequently, $\ell_{\tau_j} \geqslant \ell_{\pi_j}$ and $f_{\tau_j} \leqslant f_{\pi_j}$ holds true for all $j \in [1:t]$. The fact that π_j and π_{j+1} are two maximal increasing substrings, i.e., $\ell_{\pi_j} > f_{\pi_{j+1}}$ for $j \in [1:|\mathcal{S}(\pi)|-1]$, implies $\ell_{\tau_j} \geqslant \ell_{\pi_j} > f_{\pi_{j+1}} \geqslant f_{\tau_{j+1}}$ for all $j \in [1:t-1]$. Therefore, $\tau_1, \dots, \tau_t$ are maximal, which proves the statement. □

Finally, the following lemma shows that the application of the transformation T to a permutation π results in a permutation $T(\pi)$ that has half as many maximal increasing substrings as π.

Lemma 5.4. *Let* $\pi \in s\mathcal{P}_n$ *with* $|\mathcal{S}(\pi)| > 1$. *It holds that* $|\mathcal{S}(T(\pi))| = \lceil |\mathcal{S}(\pi)|/2 \rceil$.

Proof. Let $\pi \in s\mathcal{P}_n$ with $|\mathcal{S}(\pi)| > 1$. Consider the case that $|\mathcal{S}(\pi)|$ is even. By the construction of $T(\pi) = \tau_1 \dots \tau_{|\mathcal{S}(T(\pi))|}$ it holds that two maximal increasing substrings of π always form a new increasing substring in $T(\pi)$, hence $|\mathcal{S}(T(\pi))| \leqslant |\mathcal{S}(\pi)|/2$. By Lemma 5.3 it holds that every τ_i of $T(\pi)$ is also maximal, and hence $|\mathcal{S}(T(\pi))| \geqslant |\mathcal{S}(\pi)|/2$. Altogether, $|\mathcal{S}(T(\pi))| = |\mathcal{S}(\pi)|/2$ if $|\mathcal{S}(\pi)|$ is even.

Now consider that $|\mathcal{S}(\pi)|$ is odd. By the construction of $T(\pi) = \tau_1 \dots \tau_{|\mathcal{S}(T(\pi))|}$ it holds that $\tau_1, \dots, \tau_{\lfloor |\mathcal{S}(T(\pi))|/2 \rfloor}$ of $T(\pi)$ are always formed by two maximal increasing substrings of π and $\tau_{\lceil |\mathcal{S}(T(\pi))|/2 \rceil}$ is formed by one maximal increasing substring of π. Hence, $|\mathcal{S}(T(\pi))| \leqslant \lceil |\mathcal{S}(\pi)|/2 \rceil$. By Lemma 5.3 it holds that every maximal

increasing substring of $T(\pi)$ is also maximal, hence $|\mathcal{S}(T(\pi))| \geqslant \lceil |\mathcal{S}(\pi)|/2 \rceil$. Altogether, $|\mathcal{S}(T(\pi))| = \lceil |\mathcal{S}(\pi)|/2 \rceil$ if $|\mathcal{S}(\pi)|$ is odd. □

For a clear presentation in the following paragraphs, the notation $\Upsilon(\pi)$ denotes the set of all elements of a permutation π of size n that have a negative sign, i. e., $\Upsilon(\pi) := \{|\pi(i)| : \pi(i) < 0, i \in [1:n]\}$. Observe that $\Upsilon(\pi)$ contains the absolute value of every negative element of π. The following proposition states the upper bounds on the minimum number of 4-type rearrangements that have to be applied to ι in order to produce a permutation with a certain number of maximal increasing substrings.

Proposition 5.2. *Let $\pi \in s\mathcal{P}_n$ with $2^{k-1} < |\mathcal{S}(\pi)| \leqslant 2^k$ for a $k \in \mathbb{N}_0$. The following statements are true:*

i) Permutation π can be obtained by applying $k+2$ rearrangements from $\mathcal{M}_{4\text{-}type}$ to ι.

ii) If there exist $\ell, m \in \mathbb{N}$ such that $\Upsilon(\pi) = [\ell : m]$, then permutation π can be obtained by applying $k+1$ rearrangements from $\mathcal{M}_{4\text{-}type}$ to ι.

iii) If $\Upsilon(\pi) = \emptyset$, then π can be obtained by applying k rearrangements from $\mathcal{M}_{4\text{-}type}$ to ι

Proof. Let π be a signed permutation of size n as specified in the statement. Note that all three statements of the proposition are proven simultaneously by induction on k. For the base case consider $k = 0$. It follows that $2^{-1} < |\mathcal{S}(\pi)| \leqslant 2^0$. Therefore, $|\mathcal{S}(\pi)|$ is equal to 1. Consider first that $\pi = \iota$, hence $\Upsilon(\pi) = \emptyset$. In this case no rearrangement is needed to transform ι into π which proves the base case for Statement (iii). Now consider that $\pi \neq \iota$. Hence, for π it holds that $\pi(1) < \ldots < \pi(i) < 0 < \pi(i+1) < \ldots < \pi(n)$ with $1 \leqslant i \leqslant n$. If there exist $\ell, m \in \mathbb{N}$ such that $\Upsilon(\pi) = \{|\pi(1)|, \ldots, |\pi(i)|\} = [\ell : m]$, then it holds that $\rho_{iT}(\Upsilon(\pi), [1:\ell-1]) \circ \iota = \pi$ if $\ell \neq 1$ and $\rho_I(\Upsilon(\pi)) \circ \iota = \pi$ otherwise. Consequently, π can be obtained by the application of one 4-type rearrangement (inversion or inverse transposition) which proves the base case for Statement (ii). Finally, if such indices ℓ and m do not exist, then Corollary 4.2 implies $d_{\mathcal{M}_{iTDRL}}(\iota, \pi) = 1$. Since every iTDRL can replaced by a TDRL and an inversion or an inverse transposition (see Figure 4.6), π can be obtained by applying two 4-type rearrangements to ι which proves the base case for Statement (i).

For the induction step let $k \geqslant 1$. Consider a permutation π such that $2^{k-1} < |\mathcal{S}(\pi)| \leqslant 2^k$. Consider first that $\Upsilon(\pi) = \emptyset$. By Chaudhuri et al. (2006) it holds that $d_{\mathcal{M}_{TDRL}}(\iota, \pi) = \lceil \log_2 |\mathcal{S}(\pi)| \rceil = k$. Hence, π can be obtained from ι by the application of k TDRL rearrangements. Statement (iii) follows by the fact that $\mathcal{M}_{TDRL} \subset \mathcal{M}_{4\text{-type}}$. Now consider that $\Upsilon(\pi) \neq \emptyset$. Lemma 5.4 implies that $|\mathcal{S}(T(\pi))| = \lceil |\mathcal{S}(\pi)|/2 \rceil$ and thus, it holds that $2^{k-2} < |\mathcal{S}(T(\pi))| \leqslant 2^{k-1}$. Note that, if there exist $\ell, m \in \mathbb{N}$ with $\Upsilon(\pi) = [\ell : m]$, then $T(\pi)$ also satisfies this property, since transformation T does not influence the sign of an element of π. By the induction hypothesis it holds that $T(\pi)$ can be obtained by applying $k+1$ (respectively k) rearrangements from $\mathcal{M}_{4\text{-type}}$ to ι (in the

case that such ℓ and m exist). Lemma 5.2 states that there always exists a TDRL $\rho_{\text{TDRL}} \in \mathcal{M}_{\text{4-type}}$ such that $\rho_{\text{TDRL}} \circ T(\pi) = \pi$. Consequently, π can be obtained by the application of $k+2$ (respectively $k+1$) rearrangements from $\mathcal{M}_{\text{4-type}}$ which proves Statement (i) (respectively Statement (ii)). □

Observe that the proof of Proposition 5.2 shows that the permutation π in Proposition 5.2 can always be obtained from ι by: 1) a TDRL followed by an inversion (or inverse transposition) followed by k TDRLs, 2) an inversion or inverse transposition followed by k TDRLs if for π exist $\ell, m \in \mathbb{N}$ such that $\Upsilon(\pi) = [\ell : m]$, or 3) by k TDRLs if $\Upsilon(\pi) = \emptyset$.

Bounding the $\mathcal{M}_{\text{4-type}}$ Distance

In the following, the lower and upper bounds on the $\mathcal{M}_{\text{4-type}}$ distance are provided as formulated in the following theorem.

Theorem 5.1. *Let π be a signed permutation of size $n \in \mathbb{N}$. For the $\mathcal{M}_{\text{4-type}}$ distance it holds that:*

$$\lceil \log_2 |\mathcal{S}(\pi)| \rceil \leqslant d_{\mathcal{M}_{\text{4-type}}}(\iota, \pi) \leqslant \begin{cases} \lceil \log_2 |\mathcal{S}(\pi)| \rceil & \text{if } \pi \in \mathcal{P}_n, \\ \lceil \log_2 |\mathcal{S}(\pi)| \rceil + 1 & \text{if } \exists\, \ell, m \in \mathbb{N} : \Upsilon(\pi) = [\ell : m] \\ \lceil \log_2 |\mathcal{S}(\pi)| \rceil + 2 & \text{else.} \end{cases}$$

Proof. Let $\pi \in s\mathcal{P}_n$. There exists a unique $k \in \mathbb{N}_0$ with $2^{k-1} < |\mathcal{S}(\pi)| \leqslant 2^k$. Therefore, Proposition 5.1 implies that $d_{\mathcal{M}_{\text{4-type}}}(\iota, \pi) \geqslant k$. By $|\mathcal{S}(\pi)| \leqslant 2^k$ it follows that $\lceil \log_2 |\mathcal{S}(\pi)| \rceil \leqslant k \leqslant d_{\mathcal{M}_{\text{4-type}}}(\iota, \pi)$.

By $2^{k-1} < |\mathcal{S}(\pi)|$ it holds that $k - 1 < \log_2 |\mathcal{S}(\pi)|$ which implies $k \leqslant \lceil \log_2 |\mathcal{S}(\pi)| \rceil$. Proposition 5.2 implies that $d_{\mathcal{M}_{\text{4-type}}}(\iota, \pi) \leqslant k'$, where $k' = k$ if $\Upsilon(\pi) = \emptyset$; $k' = k+1$ if for π exist $\ell, m \in \mathbb{N}$ such that $\Upsilon(\pi) = [\ell : m]$; and $k' = k+2$ otherwise. Combining the inequalities $k \leqslant \lceil \log_2 |\mathcal{S}(\pi)| \rceil$ and $d_{\mathcal{M}_{\text{4-type}}}(\iota, \pi) \leqslant k'$ (in the respective case) proves the theorem. □

Several mentionable facts are implied by Theorem 5.1. First of all, the bounds on the $\mathcal{M}_{\text{4-type}}$ distance of a permutation π that are determined in Theorem 5.1 can be computed by calculating the number of maximal increasing substrings of π. Certainly, this can be done in linear time with respect to the number of elements of π. The $\mathcal{M}_{\text{4-type}}$ distance of a permutation π is identical to the $\mathcal{M}_{\text{TDRL}}$ distance if π does not contain any elements with a negative sign (i. e., $\Upsilon(\pi) = \emptyset$). Otherwise, i. e., $\Upsilon(\pi) \neq \emptyset$, the bounds that are proposed in Theorem 5.1 are very close in the sense that the deviation from the actual $\mathcal{M}_{\text{4-type}}$ distance is at most two. Finally, Theorem 5.1 implies that a parsimonious sorting scenario for signed linear permutations under $\mathcal{M}_{\text{4-type}}$ almost entirely contains rearrangements of type TDRL. The practical consequences of this insight for biological applications are further discussed in Section 5.2.3.

5.2.2 *Approximation Algorithm for Sorting By Mitochondrial Rearrangements*

In the following, a quasilinear time approximation algorithm for the sorting problem for two given permutations under $\mathcal{M}_{4\text{-type}}$ is proposed which guarantees the bounds that are formulated in Theorem 5.1.

Recall that the proof of Proposition 5.2 implies that for a given signed linear permutation π there always exists a sequence of 4-type rearrangements which contains at most one inversion (or inverse transposition) and a certain number of TDRLs (from which two might be transpositions). The following algorithm computes for a given signed linear permutation π a scenario $(\rho_1, \ldots, \rho_t)$ that is of one of the following types:

1) $t = \lceil \log_2 |\mathcal{S}(\pi)| \rceil$ and $\rho_2, \ldots, \rho_t$ are TDRLs and ρ_1 is either an inversion or an inverse transposition,

2) $t = \lceil \log_2 |\mathcal{S}(\pi)| \rceil$ and $\rho_1, \ldots, \rho_t$ are TDRLs,

3) $t = \lceil \log_2 |\mathcal{S}(\pi)| \rceil + 1$ and $\rho_2, \ldots, \rho_t$ are TDRLs and ρ_1 is either an inversion or an inverse transposition, and

4) $t = \lceil \log_2 |\mathcal{S}(\pi)| \rceil + 2$ and $\rho_1, \rho_3, \ldots, \rho_t$ are TDRLs and ρ_2 is an inversion.

The pseudo code of the algorithm can be found in Algorithm 2 and an example illustrating the algorithm is given by Example 5.3.

The main idea of Algorithm 2 is to generate a sequence S of transformations that transforms π into ι by the following procedure: First, transformation T is iteratively applied to π in order to produce a permutation π' that has two maximal increasing substrings. If π' can be transformed into ι by the application of one inversion or inverse transposition, then S is obtained by applying the respective rearrangement. Otherwise, the transformation T is applied once more to π' which results into a permutation π'' that has exactly one maximal increasing substring. It follows that (with respect to the set $\Upsilon(\pi'')$) either no additional transformation, one inverse transposition, or an inversion followed by transformation T is applied to π'' in order to obtain S. In all those cases, the sought scenario for ι and π under $\mathcal{M}_{4\text{-type}}$ is obtained from S by computing the inverting 4-type rearrangements for every transformation in S and reversing the relative order of all computed rearrangements. In the following, Algorithm 2 is described in more detail.

Let π be a signed permutation of size $n \in \mathbb{N}$. It is easy to see that for π exactly one of the following cases applies: (i) $\Upsilon(\pi) = \emptyset$, (ii) for $\Upsilon(\pi)$ exist $\ell, m \in \mathbb{N}$ such that $\Upsilon(\pi) = [\ell : m]$, or (iii) neither Case (i) nor Case (ii) holds. Since applying the transformation T to π does not influence the signs of the elements of π, the respective case that applies for π also applies for permutation $T(\pi)$. (This observation is silently used in the following description of Algorithm 2.)

Algorithm 2 : Pseudo code of sorting by 4-type rearrangements

Data : $\pi \in s\mathcal{P}_n$
Result : $(\rho_1, \ldots, \rho_t) \in \mathfrak{S}_{\mathcal{M}_{4\text{-type}}}(\iota, \pi) : (t - d_{\mathcal{M}_{4\text{-type}}}(\iota, \pi)) \in [0:2]$

```
if π == ι then
    return ∅;
j ← 0;
while π ≠ ι do
    j ← j + 1;
    π = π_1 ... π_|S(π)|;
    Υ(π) ← {|π(i)| : π(i) < 0, i ∈ [1:n]};
    if |S(π)| = 1 then
        if ∃ ℓ, m ∈ [2:n] : Υ(π) = [ℓ:m] then
            π ← ρ_iT(Υ(π), [1:ℓ−1]) ∘ π;
            ρ_j ← ρ_iT(Υ(π), [1:ℓ−1]);
        else
            π ← ρ_I(Υ(π)) ∘ π;
            ρ_j ← ρ_I(Υ(π));
        continue;
    if |S(π)| > 1 then
        if |S(π)| = 2, π(1) > 0, ∃ ℓ, m ∈ [1:n] : Υ(π) = [ℓ:m]
         then
            if ∃ ℓ, m with 2 ⩽ ℓ ⩽ m ⩽ n:
            π = (1 ... ℓ−1 −m ... −ℓ m+1 ... n) then
                π ← ρ_I(Υ(π)) ∘ π;
                ρ_j ← ρ_I(Υ(π));
                continue;
            if ∃ o, p, q with 1 ⩽ o < p < q ⩽ n:
            π = (1 ... o−1 p ... q −(p−1) ... −o q+1 ... n)
             then
                π ← ρ_iT(Υ(π), {p, ..., q}) ∘ π;
                ρ_j ← ρ_iT(Υ(π), {p, ..., q});
                continue;
            if ∃ o, p, q with 1 ⩽ o < p < q ⩽ n:
            π = (1 ... o−1 −q ... −p o ... p−1 q+1 ... n) then
                π ← ρ_iT(Υ(π), {o, ..., p−1}) ∘ π;
                ρ_j ← ρ_iT(Υ(π), {o, ..., p−1});
                continue;
        π ← π_1 ⊕ π_⌈|S(π)|/2⌉+1 ... π_⌈|S(π)|/2⌉ = T(π);
        ρ_j ← ρ_TDRL(E(π_1 ... π_⌈|S(π)|/2⌉), E(π_⌈|S(π)|/2⌉+1 ... π_|S(π)|));
    continue;
return (ρ_j, ..., ρ_1);
```

If $\pi = \iota$, i.e., the sorting sequence of 4-type rearrangements that transforms ι into π is empty, then an empty scenario is returned in lines 1 to 2. If otherwise $\pi \neq \iota$, then a counter j is initialized with 0 in Line 3. The sorting scenario for ι and π is iteratively computed in the while loop (lines 4 to 35) as long as $\pi \neq \iota$. In each iteration of the while loop the counter j is incremented (Line 5), the maximal increasing substring decomposition of π is computed (Line 6), and the set $\Upsilon(\pi)$ is generated (Line 7). If π has more than two maximal increasing substring, i.e., $|\mathcal{S}(\pi)| > 1$, then π is always substituted by $T(\pi)$ (Line 33) and the corresponding inverting TDRL (which exists by Lemma 5.2) is computed (Line 34). This process is repeated until $|\mathcal{S}(\pi)| = 2$ which requires $\lceil \log_2 |\mathcal{S}(\pi)| \rceil - 1$ iterations (see Lemma 5.4).

If π has two maximal increasing substrings in which the left maximal increasing substring contains only positive elements (which is implied by $\pi(1) > 0$), and there exist two indices ℓ and m with $1 \leqslant \ell \leqslant m \leqslant n$ such that $\Upsilon(\pi) = [\ell : m]$, then it is checked whether π can be transformed into ι by using one inversion (lines 18 to 22) or one inverse transposition (lines 23 to 32). If one of these conditions (lines 18, 23, or 28) is satisfied for π, then the respective rearrangement is applied to π transforming it into ι (lines 20, 25, 30) and the corresponding inverting rearrangement is computed (lines 21, 26, 31). Hence, the condition of the while loop becomes false and the algorithm returns in Line 36 a scenario of Type 1. If π cannot be transformed into ι by an inversion or inverse transposition, i.e., none of the conditions in lines 18, 23 or 28 are satisfied, then π is substituted by $T(\pi)$ once more (lines 33, 34) resulting into a permutation that has exactly one maximal increasing substring.

If now Case (i) applies to π, then $\pi = \iota$. Hence, the condition of the while loop becomes false and the algorithm returns in Line 36 a scenario of Type 2.

Now consider that Case (i) does not apply to π. Since $|\mathcal{S}(\pi)| = 1$ it follows that $\pi(1) < \ldots < \pi(i) < 0 < \pi(i+1) < \ldots < \pi(n)$ for an $i \in [1 : n-1]$.

If Case (ii) applies to the initial π, then either $1 \in \{|\pi(1)|, \ldots, |\pi(i)|\} = \Upsilon(\pi)$ or $1 \notin \Upsilon(\pi)$. Note that the former (latter) condition implies that π can be expressed as $(-i \ \ldots \ -1 \ i+1 \ \ldots \ n)$ (respectively $(-i \ \ldots \ -\ell \ 1 \ \ldots \ \ell-1 \ i+1 \ \ldots \ n)$ with $\ell \in [2 : i-1]$). Hence, if the former is true, then π is substituted by $\rho_I(\Upsilon(\pi)) \circ \pi$ (Line 13), the corresponding inversion (i.e., $\rho_I(\Upsilon(\pi))$) is computed (Line 14), and it holds that $\rho_I(\Upsilon(\pi)) \circ \pi = \iota$. If, otherwise, the latter is true, then π is substituted by $\rho_{iT}(\Upsilon(\pi), [1 : \ell-1]) \circ \pi$ (Line 10), the corresponding inverse transposition (i.e., $\rho_{iT}(\Upsilon(\pi), [1 : h-1])$) is computed (Line 11), and it holds that $\rho_{iT}(\Upsilon(\pi), [1 : \ell-1]) \circ \pi = \iota$. Hence, if Case (ii) holds true, then the condition of the while loop becomes false and the algorithm returns in Line 36 a scenario of Type 3.

It remains to consider Case (iii). Hence, neither Case (i) nor Case (ii) is satisfied. Since $|\mathcal{S}(\pi)| = 1$ and the expression in Line 9 is false, lines 13 to 14 are executed. Hence, π is substituted by $\rho_I(\Upsilon(\pi)) \circ \pi$ (Line 13), the corresponding inversion (i.e., $\rho_I(\Upsilon(\pi))$) is computed (Line 14), and it holds that $\rho_I(\Upsilon(\pi)) \circ \pi$ is a permutation that has

exactly two maximal increasing substrings and $\Upsilon(\rho_I(\Upsilon(\pi)) \circ \pi) = \emptyset$. Consequently, in the next iteration the lines 33 to 34 are executed which transform π into ι (Line 33) and the corresponding inverting TDRL (Line 34) is computed. Subsequently, the algorithm returns in Line 36 a scenario of Type 4.

The following example illustrates the application of Algorithm 2.

Example 5.3. *Consider the permutation* $\pi = (1\ -4\ -3\ -2\ 5)$ *which has the maximum increasing substrings* 1 *and* $-4\,-3\,-2\,5$. *Since* $\pi(1) = 1 > 0$ *and* $\Upsilon(\pi) = \{2,3,4\} = [2\colon 4]$ *the condition in Line* 17 *of Algorithm 2 is satisfied. Likewise the condition in Line* 18 *is satisfied and thus, the inversion* $\rho_I(\{2,3,4\})$ *is applied to* π *resulting into* ι *and the rearrangement* $\rho_I(\{2,3,4\})$ *is stored in Line* 21. *Subsequently, the scenario* $(\rho_I(\{2,3,4\}))$ *of Type* 1 *is returned.*

Now consider $\pi = (1\ 3\ 5\ 2\ 4)$. *Permutation* π *contains the maximal increasing substrings* $1\,3\,5$ *and* $2\,4$. *Since* $\Upsilon(\pi) = \emptyset$ *it follows that* $\pi \in \mathcal{P}_n$ *and thus, Theorem 5.1 implies that* $d_{\mathcal{M}_{4\text{-}type}}(\iota, \pi) = 1$. *Hence, only one* 4*-type rearrangement is necessary to obtain* π *from* ι. *To obtain this rearrangement, Algorithm 2 applies transformation* T *to* π. *The corresponding* 4*-type rearrangement that reverses the application of* T *is* $\rho_{TDRL}(\mathcal{E}(1\,3\,5), \mathcal{E}(2\,4)) = \rho_{TDRL}(\{1,3,5\},\{2,4\})$. *Since* $T(\pi) = (1\ 2\ 3\ 4\ 5) = \iota$ *the scenario* $(\rho_{TDRL}(\{1,3,5\},\{2,4\}))$ *of Type* 2 *is returned.*

Consider $\pi = (-3\ 4\ 5\ -2\ 1)$. *Permutation* π *contains the maximal increasing substrings* $-3\,4\,5$ *and* $-2\,1$. *Furthermore, it holds that* $\Upsilon(\pi) = \{2,3\}$ *and thus,* $\Upsilon(\pi) = [2\colon 3]$. *By Theorem 5.1 it follows that* π *can be obtained from* ι *by two* 4*-type rearrangements. To obtain a sequence of* 4*-type rearrangements of this length, Algorithm 2 computes a sequence of transformations that transform* π *into* ι. *The first transformation that is applied to* π *is* T *which gives* $T(\pi) = (-3\ -2\ 1\ 4\ 5)$. *The corresponding* 4*-type rearrangement that reverses the application of* T *is* $\rho_{TDRL}(\{3,4,5\},\{1,2\})$. *Since* $\Upsilon(\pi) = \Upsilon(T(\pi)) = [2\colon 3]$ *the inverse transposition* $\rho_{iT}(\{2,3\},\{1\})$ *is applied to* $T(\pi)$. *Note that* $\rho_{iT}(\{2,3\},\{1\})$ *can invert its effect if it is applied to* $\rho_{iT}(\{2,3\},\{1\}) \circ T(\pi)$. *Since* $\rho_{iT}(\{2,3\},\{1\}) \circ T(\pi) = \iota$ *Algorithm 2 returns the scenario* $(\rho_{iT}(\{2,3\},\{1\}), \rho_{TDRL}(\{3,4,5\},\{1,2\}))$ *of Type* 3.

Finally, consider $\pi = (-4\ 5\ -1\ 2\ 3)$ *which contains the maximal increasing substrings* $-4\,5$ *and* $-1\ 2\ 3$. *In addition, it holds that* $\Upsilon(\pi) = \{1,4\}$. *By Theorem 5.1 it follows that* π *can be obtained from* ι *by three* 4*-type rearrangements. In order to obtain a sequence of transformations that transform* π *into* ι, *Algorithm 2 first applies* T *to* π *which gives* $(-4\ -1\ 2\ 3\ 5)$. *Then* $\rho_I(\{1,4\}))$ *is applied to* $(-4\ -1\,2\,3\,5)$ *resulting into* $(1\,4\,2\,3\,5)$. *Observe that* $(1\,4\,2\,3\,5)$ *has two maximal increasing substrings and that* $\Upsilon((1\,4\,2\,3\,5)) = \emptyset$. *Therefore, the transformation* T *is applied to* $(1\ 4\ 2\ 3\ 5)$ *which results in* $(1\ 2\ 3\ 4\ 5)$. *Reversing all transformations and its relative order gives the scenario* $(\rho_{TDRL}(\{1,4\},\{2,3,5\})), \rho_I(\{1,4\})), \rho_{TDRL}(\{4,5\},\{1,2,3\})))$ *of Type* 4.

For a runtime analysis of Algorithm 2 consider $\pi \in s\mathcal{P}_n$. It is not hard to see that the evaluations in lines 1, 4, 8 – 9, 16 – 19, 23 – 24, and 28 – 29, the computation of the maximal increasing substring decomposition (Line 6), the computation of $\Upsilon(\pi)$ (Line 7), the replacement of π (lines 10, 13, 20, 25, 30, and 33), and the construction of ρ_j (lines 11,

14, 21, 26, 31, 34) can be done in time $\mathcal{O}(n)$. Therefore, the lines 4 to 35 are executed in time $\mathcal{O}(n)$. By Theorem 5.1 these lines are executed at most $\lceil \log_2 |\mathcal{S}(\pi)| \rceil + 2$ times. Since $|\mathcal{S}(\pi)| \leqslant n$ it follows that Algorithm 2 has a runtime in $\mathcal{O}(n \log n)$.

Algorithm 2 is implemented in C++ and is freely available from `http://pacosy.informatik.uni-leipzig.de/sp4type`. It is also worth mentioning that if a TDRL has the same effect as a transposition (partial duplication TDRL), then the implementation of Algorithm 2 always returns a transposition (respectively an equivalent partial duplication TDRL).

5.2.3 *Consequences for Biological Applications*

The results from Section 5.2.1 imply several consequences for biological applications:

A parsimonious scenario for two gene orders under $\mathcal{M}_{4\text{-type}}$ is strongly dominated by rearrangements of type TDRL. This is due to the fact, that the different types of rearrangement operations have a major difference in their *rearrangement power*: while inversions (inverse transpositions and even transpositions) can change the number of maximal increasing substrings of a permutation that represents a given gene order at most by one (respectively two), TDRLs are capable of modifying this number by a factor of two. Hence, TDRLs are able to rearrange a gene order in a much greater extent than inversions or (inverse) transpositions.

But unfortunately, long sequences of TDRL rearrangements are often of limited biological reliability for a variety of reasons. On the one hand, this is because almost all TDRLs (of the constructed scenarios that are obtained by Algorithm 2) affect the whole gene order, while in real world biological scenarios rearrangements tend to occur more frequently close to the replication origin and affect only a small number of genes (Fonseca and Harris, 2008). On the other hand, the frequencies of the different types of mitochondrial rearrangements that have been proposed in the literature (e. g., see Bernt et al. (2013b)) are not reflected by the scenarios that are computed with Algorithm 2.

Another insight gained from the preceding section is that the $\mathcal{M}_{4\text{-type}}$ distance for two gene orders with n genes grows asymptotically not faster than $\log n$. Therefore, the $\mathcal{M}_{4\text{-type}}$ distance is a rough evolutionary similarity measure compared to other rearrangement distances that grow asymptotically not faster than n (e. g., the $\mathcal{M}_{\mathrm{I}}$ distance). Consequently, the usefulness of the $\mathcal{M}_{4\text{-type}}$ distance for phylogeny reconstruction might be limited by the small maximum distance which grows very slowly for an increasing gene order size.

Both insights indicate that the 4-type rearrangement model (with equally weighted rearrangements) is less valuable for inferring reliable evolutionary rearrangements scenarios. Therefore, in the remainder of this chapter two variations of the model $\mathcal{M}_{4\text{-type}}$ are proposed in order to improve the reliability of the inferred rearrangement scenarios. The main idea of both variants is to restrict the effect of TDRL

rearrangements in the constructed scenarios. This can be done by reducing the number of TDRL rearrangements or/and the number of genes that are affected by a TDRL.

The first approach, which is covered in Section 5.3, allows to restrict the number of the TDRL rearrangements by employing a weighting scheme on $\mathcal{M}_{4\text{-type}}$. The idea is to weight a rearrangement from $\mathcal{M}_{4\text{-type}}$ by its type such that the weights reflect the likelihood of a rearrangement to occur during evolution.

The second approach extends the first one by using biological constraints to additionally restrict the amount of genes that can be influenced by a rearrangement. The idea is to use the common interval framework enforcing the rearrangements to act locally on smaller subsets of the genes. This results in (preserving) scenarios that grow asymptotically not faster than the number of genes of the considered gene orders. This second approach leads to the algorithm `CREx2` that is well suited for studying mitochondrial gene order evolution, see Section 5.4 and Section 5.5.

5.3 INTEGER LINEAR PROGRAMMING FOR SORTING BY WEIGHTED REARRANGEMENTS

The different types of genome rearrangements which shape the gene arrangement of unichromosomal genomes do not occur with similar frequencies during evolution (Bernt and Middendorf, 2011). For example, the predominant rearrangement event that is relevant for chloroplast genomes of plants is the inversion (Cosner et al., 1997) while for the case of mitochondrial genomes transpositions are the predominant (Bernt et al., 2013b). Consequently, for analyzing the phylogeny of such genomes, it is crucial to consider a suited model that reflects the different relative likelihoods for rearrangements to occur during the evolution of different taxa. The relative likelihood of a rearrangement in the computation of genome rearrangement scenarios is commonly influenced by employing a weighting scheme on the considered rearrangement model. The idea is that if a rearrangement is unlikely to occur then it is assigned with a high weight relative to a rearrangement which is assumed to occur more often during evolution. Given a weighted rearrangement model, the weighted sorting problem is then to find a rearrangement scenario for some given gene orders in which the total weight of all rearrangements that are used is minimum, see Section 2.2.4 for a formal definition.

Different weighting schemes have been proposed for various combinations of genome rearrangements. Examples are weighting inversions and transpositions with respect the number of affected genes (e. g., Bender et al. (2008) and Oliveira et al. (2018b)) or weighting inversions, transpositions, and inverted transpositions with respect to their type (e. g., Blanchette et al. (1996) and Oliveira et al. (2018a)), see also Section 2.3 for an overview. However, with a few exceptions (e. g., Bader and Ohlebusch (2007) and Lancia et al. (2015)) algorithms for solving the sorting problem usually assume fixed weights for the

different types of rearrangements and – up to now – no algorithm has been presented that includes all predominant mitochondrial rearrangements while allowing rearrangements to be weighted differentially. This lack of literature is addressed in this section by proposing an integer linear program that solves the weighted sorting problem for signed linear permutation under $\mathcal{M}_{4\text{-type}}$.

Unfortunately, in contrast to the sorting problem under $\mathcal{M}_{4\text{-type}}$ which appears to be tractable, the weighted sorting problem under $\mathcal{M}_{4\text{-type}}$ is NP-hard. This can be seen by the fact that one of its subproblems is the sorting problem of unsigned linear permutations under $\mathcal{M}_T$ which has been proven to be NP-hard (Bulteau et al., 2012). Therefore, one cannot realistically hope for an efficient exact algorithm. However, one method which is sufficient for obtaining exact solutions for such problems is an integer linear program (ILP). Using ILP, an optimization problem is described by means of 1) a set of binary variables $V_1, \ldots, V_m$; 2) a set of linear constraints of the form $\sum_{i=1}^{m} a_{ij} V_i \geqslant b_j$ with $a_{ij}, b_j \in \mathbb{R}$ that constrain the variables; and 3) a linear objective function $\sum_{i=1}^{m} c_i V_i$ with $c_i \in \mathbb{R}$ that transforms the model to an optimization problem. An exact solution for the optimization problem represented by an ILP is then given by an assignment of the binary variables V_1 to V_n that minimize (or sometimes maximize) the objective function while satisfying the given constraints. An assignment is usually obtained by an optimizer such as the `IBM ILOG™ Optimizer` or `Gurobi` (Gurobi Optimization, 2018). For an in-depth introduction to theoretical and practical aspects of ILPs see for example Bertsimas and Tsitsiklis (1997) and Schrijver (1998).

In the following Section, the polynomial size integer linear program `GeRe-ILP` (Hartmann et al., 2017) is proposed for solving the weighted sorting problem for signed linear permutation under $\mathcal{M}_{4\text{-type}}$. `GeRe-ILP` uses $\mathcal{O}(n^3)$ variables and $\mathcal{O}(n^3)$ constraints for a permutation of size $n \in \mathbb{N}$. It provides an exact minimum weight rearrangement scenario for a signed permutations π and the identity permutation ι with arbitrary weights for the four different types of rearrangement operations.

In favor of a clear notation, the constraints of the following integer linear program are often specified as Boolean operations. Those operation can be modeled easily by linear constraints (which are not further mentioned in this thesis) as stated in the following proposition.

Proposition 5.3. *Let $X_1, X_2, A_1, \ldots, A_n$, and $B_1, \ldots, B_m$ be binary variables with $n + m > 0$ and $n, m \in \mathbb{N}_0$ of a given integer linear program. Furthermore, let $N \in \mathbb{R}$ be such that $N > |X_1 - X_2| \geqslant 0$. The implication*

$$A_1 \wedge \ldots \wedge A_n \wedge \neg B_1 \wedge \ldots \wedge \neg B_m \Rightarrow X_1 = X_2$$

is satisfied by using the following inequalities as linear constraints:

$$\begin{aligned} N(1 - A_1) + \ldots + N(1 - A_n) + NB_1 + \ldots + NB_m + X_1 &\geqslant X_2, \\ N(1 - A_1) + \ldots + N(1 - A_n) + NB_1 + \ldots + NB_m + X_2 &\geqslant X_1. \end{aligned}$$

Proof. Consider $A_i = 1$ and $B_j = 0$ for all $i \in [1:n]$, $j \in [1:m]$. The first constraint implies $X_1 \geqslant X_2$ and the second implies $X_2 \geqslant X_1$ and, thus, $X_1 = X_2$. Observe that both constraints are always satisfied if there exists at least one A_i with $A_i = 0$ or one B_j with $B_j = 1$. This is because $N > |X_1 - X_2| \geqslant 0$. □

For the following section it is essential to introduce the notation of a bounding position of a rearrangement (see also Example 5.4). Let π be a permutation and X, Y be two disjoint consecutive intervals of π, i.e., $X, Y, X \cup Y \in I(\pi)$ and $X \cap Y$. Consider that a rearrangement $\rho_Z \in \mathcal{M}_{4\text{-type}}$ with $Z \in \{I, T, iT\}$ is applied to π. The position of the leftmost element of π that is affected by ρ_Z is called the *left bounding position*. If the position of the rightmost element of π that is affected by ρ_Z is $r \in [1:n]$, then $r+1$ is called *right bounding position*. For a inversion it is assumed that the *middle bounding position* coincides with the right bounding position. If a transposition $\rho_T(X, Y)$ (or a inverse transposition $\rho_{iT}(X, Y)$) is applied to π, then the position of the leftmost element of the rightmost interval of $\{X, Y\}$ is called the *middle bounding position*.

Example 5.4. *Consider the permutation* $(4\ {-6}\ 2\ 3\ {-1}\ 5)$ *and the intervals* $X = \{2, 3, 6\}$ *and* $\{1, 5\}$ *of* π. *The inversion* $\rho_I(X)$ *for* π *transforms* π *into* $(4\ {-3}\ {-2}\ 6\ {-1}\ 5)$. *The leftmost (rightmost) element of* π *that is affected by* $\rho_I(X)$ *is* -6 *(respectively* 3*). Therefore, the left (right) bounding position of* $\rho_I(X)$ *is* 2 *(respectively* 5*). Since* $\rho_I(X)$ *is an inversion, the middle bounding position of* $\rho_I(X)$ *is* 5 *as well. The inverse transposition* $\rho_{iT}(Y, X)$ *for* π *transforms* π *into* $(4\ {-5}\ 1\ {-6}\ 2\ 3)$. *The leftmost (rightmost) element of* π *that is affected by* $\rho_{iT}(Y, X)$ *is* -6 *(respectively* 5*). Consequently, the left (right) bounding position of* $\rho_{iT}(Y, X)$ *is* 2 *(respectively* 7*). Interval* Y *is the rightmost interval of* $\{X, Y\}$ *and the leftmost element of* Y *is* -1. *Hence, the middle bounding position of* $\rho_{iT}(Y, X)$ *is* 5.

5.3.1 *Integer Linear Programming* `GeRe-ILP`

For a permutation $\pi \in s\mathcal{P}_n$ `GeRe-ILP` determines a rearrangement scenario and maintains variables that represent the intermediate permutations π_k with $k \in [1:t]$. In the following, $k \in [1:t]$ is the index of intermediate permutations and $i, j \in [1:n+1]$ denote elements. Note that to each permutation of size n an auxiliary element $n+1$ is added at the right end of the permutation in order to model the right bounding position in some cases.

The permutations are encoded by binary variables P_{ijk} which hold the information if element i is to the left of element j in permutation π_k. Binary variables O_{ik} encode the sign of element $\pi_k(i)$. Formally,

$$P_{ijk} = \begin{cases} 1 & \text{if } \pi_k^{-1}(i) < \pi_k^{-1}(j), \\ 0 & \text{otherwise,} \end{cases} \qquad \text{(ILP 1)}$$

$$O_{ik} = \begin{cases} 1 & \text{if } \pi_k(\pi_k^{-1}(i)) > 0, \\ 0 & \text{otherwise.} \end{cases} \tag{ILP 2}$$

Note that the variables P_{ijk} and O_{ik} are fixed for $k = 0$ and $k = t$ such that $\pi_0 = \pi$ and $\pi_t = \iota$. Furthermore, it is worth mentioning that the position and the sign of the auxiliary element $n+1$ is fixed for all intermediate permutations by adding the constraints $O_{n+1k} = 0$ and $P_{in+1k} = 1$ for all $k \in [0:t]$ and $i \in [1:n]$. The variables P_{ijk} for $k = [2:t-1]$ are deduced from the variables for $k-1$ by constraints. For each $k \in [1:t]$, there are four binary variables I_k, T_k, iT_k, and $TDRL_k$ that determine if an inversion, transposition, inverse transposition or a tandem duplication random loss takes place, i. e., $Z_k = 1$ if and only if $\rho_Z \circ \pi_{k-1} = \pi_k$ for a $\rho_Z \in \mathcal{M}_{\text{4-type}}$ with $Z \in \{I, T, iT, TDRL\}$. The following constraint guarantees that for all $k \in [1:t]$ it holds that exactly one of the variables I_k, T_k, iT_k, and $TDRL_k$ is one:

$$I_k + T_k + iT_k + TDRL_k = 1. \tag{ILP 3}$$

In the following, TDRL rearrangements are encoded differently from the remaining considered types of rearrangements. The variables and constraints that encode TDRL rearrangements are introduced first. Assume a TDRL $\rho_{TDRL}(X, Y) \in \mathcal{M}_{TDRL}$ transforms π_k into π_{k+1}. Then, ρ_{TDRL} is encoded by binary variables FS_{ik} such that $FS_{ik} = 0$ if and only if element i of π_k is included in set X. This is ensured by the following constraints that are set for all $i, j \in [1:n]$, $k \in [1:t-1]$:

$$TDRL_k \wedge FS_{ik} = FS_{jk} \Rightarrow P_{ijk} = P_{ijk+1} \wedge P_{jik} = P_{jik+1}, \tag{ILP 4}$$

$$TDRL_k \wedge FS_{ik} \neq FS_{jl} \Rightarrow P_{ijk} \neq P_{ijk+1} \wedge P_{jik} \neq P_{jik+1}. \tag{ILP 5}$$

A TDRL cannot toggle the sign of an element i. Hence, for all $i \in [1:n]$, $k \in [1:t-1]$ the following constraint is set:

$$TDRL_k \Rightarrow O_{ik} = O_{ik+1}. \tag{ILP 6}$$

Observe that the constraints (ILP 4) to (ILP 6) are sufficient to model TDRL rearrangements. Therefore, the constraints in the remainder of this section consider the case that no TDRL rearrangement transforms π_k into π_{k+1}, i. e., either $I_k = 1$, $T_k = 1$, or $iT_k = 1$ for a $k \in [1:t-1]$. This is satisfied by the term "$\neg TDRL_k$" which occurs in several of the following constraints.

Inversions, transpositions, and inverse transpositions are encoded by three binary variables B^{ℓ}_{ik}, B^{m}_{ik}, and B^{r}_{ik} that encode the left, middle, and right bounding position of the rearrangement such that a variable is one if position i is the respective bounding position in π_k. For every of those rearrangements, there exist exactly one left, one

middle, and one right bounding position. This is guaranteed by the following equation that is set for all $k \in [1:t]$:

$$\sum_{i=1}^{n+1} B_{ik}^{\ell} = \sum_{i=1}^{n+1} B_{il}^{r} = \sum_{i=1}^{n+1} B_{ik}^{m} = 1. \tag{ILP 7}$$

Constraint (ILP 8) guarantees that for each rearrangement operation the left bounding position is smaller than the middle bounding position. For transpositions and inverse transpositions the middle bounding position has to be to the left of the right bounding position, as it is guaranteed by Constraint (ILP 9) and Constraint (ILP 10), respectively. Constraint (ILP 11) guarantees that for inversions the middle and the right bounding position are identical. In order to satisfy those implications, the following constraints are added for all $i, j \in [1:n+1]$, $k \in [1:t]$:

$$\neg \mathrm{TDRL}_k \wedge B_{ik}^{\ell} \wedge B_{jk}^{m} \Rightarrow P_{ijk}, \tag{ILP 8}$$

$$T_k \wedge B_{ik}^{m} \wedge B_{jk}^{r} \Rightarrow P_{ijk}, \tag{ILP 9}$$

$$\mathrm{iT}_k \wedge B_{ik}^{m} \wedge B_{jk}^{r} \Rightarrow P_{ijk}, \tag{ILP 10}$$

$$I_k \wedge B_{ik}^{m} \wedge B_{jk}^{r} \Rightarrow i = j. \tag{ILP 11}$$

The bounding positions define two intervals which contain the rearranged elements, see Figure 5.2. In particular, the elements from B^{ℓ} to B^{m} (but excluding B^{m}) form the *left interval* and the elements from B^{m} to B^{r} form the *right interval*. For inverse transpositions one of the intervals is inverted. This is controlled with the binary variables C_k^{ℓ} and C_k^{r}:

$$C_k^{x} = \begin{cases} 0, & \text{interval } x \in \{\ell, r\} \text{ is not inverted,} \\ 1, & \text{interval } x \in \{\ell, r\} \text{ is inverted.} \end{cases} \tag{ILP 12}$$

Inversions are defined such that the right interval is empty (since $B^{m} = B^{r}$) and therefore only the left interval has to be inverted which is guaranteed by Constraint (ILP 13). Constraint (ILP 14) guarantees that transpositions do not invert any interval. In the case of an inverse transposition exactly one of the two intervals is inverted. This is guaranteed by Constraint (ILP 15). Altogether the following constraints are added for all $k \in [1:t]$:

$$I_k \Rightarrow C_k^{\ell} \wedge \neg C_k^{r}, \tag{ILP 13}$$

$$T_k \Rightarrow C_k^{\ell} + C_k^{r} = 0, \tag{ILP 14}$$

$$\mathrm{iT}_k \Rightarrow C_k^{\ell} + C_k^{r} = 1. \tag{ILP 15}$$

To test if an element $i \in [1:n]$ is either within the left interval, within the right interval, or outside of both intervals, four auxiliary variables are defined as follows. Two binary variables L_{ik}^{m} and L_{ik}^{r} are introduced to test if element i is to the left of B^{m} and if i is to the left of B^{r}, respectively. This is guaranteed by adding Constraint (ILP 16)

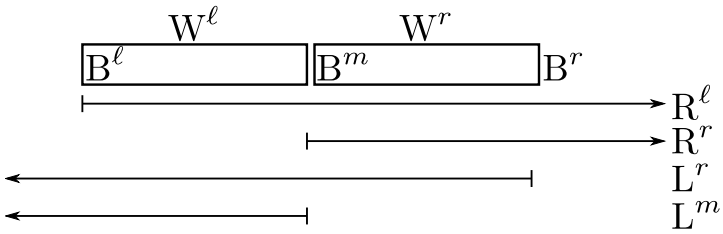


Figure 5.2: The ranges where L^m, L^r, R^ℓ, and R^m are 1 (arrows), and the resulting left interval W^ℓ and right interval W^r that are affected by the rearrangement (boxes).

for all $i \in [1:n+1]$, $k \in [1:t]$ and constraints (ILP 17) to (ILP 18) for all $i, j \in [1:n+1]$, $k \in [1:t]$:

$$\neg \mathrm{TDRL}_k \wedge B^x_{ik} \Rightarrow \neg L^x_{ik} \qquad (x \in \{m, r\}) \qquad \text{(ILP 16)}$$
$$\neg \mathrm{TDRL}_k \wedge P_{ijk} \wedge B^x_{jk} \Rightarrow L^x_{ik} \qquad (x \in \{m, r\}) \qquad \text{(ILP 17)}$$
$$\neg \mathrm{TDRL}_k \wedge P_{jik} \wedge B^x_{jk} \Rightarrow \neg L^x_{ik} \qquad (x \in \{m, r\}) \qquad \text{(ILP 18)}$$

Auxiliary binary variables R^ℓ_{ik} and R^m_{ik} are introduced to test if an element $i \in [1:n+1]$ is to the right of or equal to B^ℓ and if i is to the right of or equal to B^m, respectively. This is ensured by adding Constraint (ILP 19) for all $i \in [1:n+1]$, $k \in [1:t]$ and constraints (ILP 20) to (ILP 21) for all $i, j \in [1:n+1]$, $k \in [1:t]$:

$$\neg \mathrm{TDRL}_k \wedge B^x_{ik} \Rightarrow R^x_{ik}, \qquad (x \in \{\ell, m\}) \qquad \text{(ILP 19)}$$
$$\neg \mathrm{TDRL}_k \wedge P_{jik} \wedge B^x_{jk} \Rightarrow R^x_{ik}, \qquad (x \in \{\ell, m\}) \qquad \text{(ILP 20)}$$
$$\neg \mathrm{TDRL}_k \wedge P_{ijk} \wedge B^x_{jk} \Rightarrow \neg R^x_{ik}. \qquad (x \in \{\ell, m\}) \qquad \text{(ILP 21)}$$

Observe the difference between R^ℓ and R^m (which include B^ℓ, B^m and B^m, respectively), and L^m and L^r (which exclude B^m and B^r, B^m, respectively).

Binary variables W^ℓ_{ik} and W^r_{ik} are one if and only if an element $i \in [1:n+1]$ is within the left interval or the right interval, respectively (see Figure 5.2). This is guaranteed by setting the following constraints for all $i \in [1:n+1]$, $k \in [1:t]$:

$$\neg \mathrm{TDRL}_k \wedge R^\ell_{ik} \wedge L^m_{ik} \Rightarrow W^\ell_{ik}, \qquad \text{(ILP 22)}$$
$$\neg \mathrm{TDRL}_k \wedge R^\ell_{ik} \wedge \neg L^m_{ik} \Rightarrow \neg W^\ell_{ik}, \qquad \text{(ILP 23)}$$
$$\neg \mathrm{TDRL}_k \wedge L^m_{ik} \wedge \neg R^\ell_{ik} \Rightarrow \neg W^\ell_{ik}, \qquad \text{(ILP 24)}$$
$$\neg \mathrm{TDRL}_k \wedge R^m_{ik} \wedge L^r_{ik} \Rightarrow W^r_{ik}, \qquad \text{(ILP 25)}$$
$$\neg \mathrm{TDRL}_k \wedge R^m_{ik} \wedge \neg L^r_{ik} \Rightarrow \neg W^r_{ik}, \qquad \text{(ILP 26)}$$
$$\neg \mathrm{TDRL}_k \wedge L^r_{ik} \wedge \neg R^m_{ik} \Rightarrow \neg W^r_{ik}. \qquad \text{(ILP 27)}$$

The following constraints ensure that the order of two elements i, j is reversed if either both elements are in the inverted interval (guaranteed by Constraint (ILP 28)) or one element is in the left interval and the other element in the right interval (guaranteed by

Constraint (ILP 29)). Both constraints are added for all $i \in [1:n+1]$, $k \in [1:t-1]$.

$$\neg\text{TDRL}_k \wedge C_k^x \wedge W_{ik}^x \wedge W_{jk}^x \Rightarrow P_{ijk+1} = 1 - P_{ijk} (x \in \{\ell, r\}) \quad \text{(ILP 28)}$$
$$\neg\text{TDRL}_k \wedge W_{ik}^\ell \wedge W_{jk}^r \Rightarrow P_{ijk+1} = 1 - P_{ijk} \quad \text{(ILP 29)}$$

The order of two elements i and j of a permutation π_k is not affected by a rearrangement that is applied to π_k if 1) at least one of both elements is not between bounding position or 2) both elements are in the interval that is not inverted. Implication 1 is ensured by adding constraints (ILP 30) to (ILP 31) and Implication 2 is guaranteed by setting Constraint (ILP 32), both for all $i, j \in [1:n+1]$, $k \in [1:t-1]$.

$$\neg\text{TDRL}_k \wedge \neg W_{ik}^\ell \wedge \neg W_{ik}^r \Rightarrow P_{ijk+1} = P_{ijk} \quad \text{(ILP 30)}$$
$$\neg\text{TDRL}_k \wedge \neg W_{jk}^\ell \wedge \neg W_{jk}^r \Rightarrow P_{ijk+1} = P_{ijk} \quad \text{(ILP 31)}$$
$$\neg\text{TDRL}_k \wedge \neg C_k^x \wedge W_{ik}^x \wedge W_{jk}^x \Rightarrow P_{ijk+1} = P_{ijk} \quad (x \in \{\ell, r\}) \quad \text{(ILP 32)}$$

It remains to incorporate the effects of inversions, transpositions, and inverse transpositions on the signs of the elements. Constraint (ILP 33) guarantees the change of the sign of every element within the inverted interval. Elements which are neither in the inverted interval nor in the left interval or right interval do not change the sign. This is guaranteed by Constraint (ILP 34) and Constraint (ILP 35), respectively. The constrains (ILP 33) to (ILP 35) are added for all $i, j \in [1:n+1]$, $k \in [1:t-1]$.

$$\neg\text{TDRL}_k \wedge C_k^x \wedge W_{ik}^x \Rightarrow O_{ik+1} = 1 - O_{ik} \quad (x \in \{\ell, r\}) \quad \text{(ILP 33)}$$
$$\neg\text{TDRL}_k \wedge \neg C_k^x \wedge W_{ik}^x \Rightarrow O_{ik+1} = O_{ik} \quad (x \in \{\ell, r\}) \quad \text{(ILP 34)}$$
$$\neg\text{TDRL}_k \wedge \neg W_{ik}^\ell \wedge \neg W_{ik}^r \Rightarrow O_{ik+1} = O_{ik} \quad \text{(ILP 35)}$$

The objective is to minimize the weight of the resulting rearrangement scenario, i. e.,

$$\min \sum_{k=1}^{t} (I_k \omega_I + T_k \omega_T + iT_k \omega_{iT} + \text{TDRL}_k \omega_{\text{TDRL}}), \quad \text{(ILP 36)}$$

where ω_I, ω_T, ω_{iT}, and ω_{TDRL} denote the respective weight of a rearrangement of type inversion (I), transposition (T), inverse transposition (iT), and tandem duplication random loss (TDRL).

For n elements and distance t, the ILP model has $\mathcal{O}(tn^2)$ variables and constraints. The number of variables is due to the variables P_{ijk} that maintain the order information of elements i and j in the (intermediate) permutation π_k. Since the number of elements i and j is n and $1 \leqslant k \leqslant t \leqslant n$, the model uses $\mathcal{O}(n^3)$ many variables in the worst case. The number of constraints is $\mathcal{O}(n^3)$ since some constraints are set for all i, j, and k, e. g., see Constraint (ILP 8).

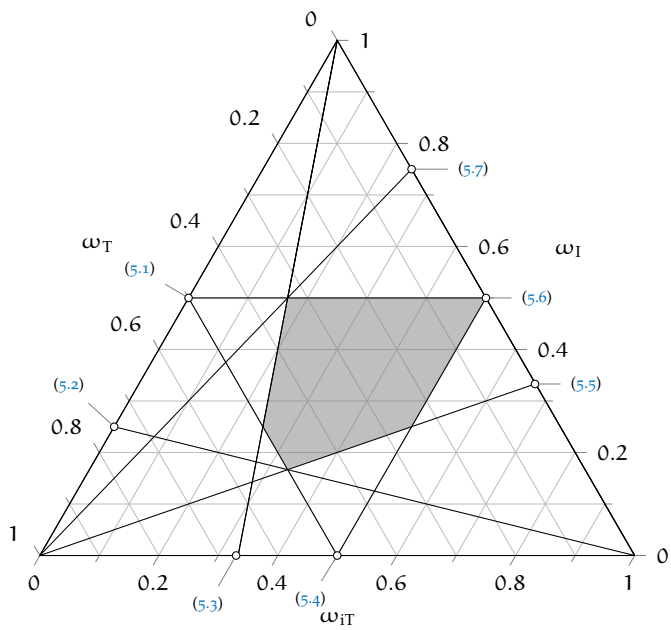


Figure 5.3: Weighting schemes for inversions, transpositions, and inverse transpositions in which a single rearrangement cannot be replaced with a sequence of other rearrangements without violating the parsimony criterion (gray area). Black lines correspond to weighting schemes in which equality holds for one of the inequalities (5.1) - (5.7).

5.3.2 *Implementation*

Given a permutation π and a maximum number of rearrangements t, *GeRe-ILP* gives either a minimum-weight scenario for π and ι using t rearrangements from $\mathcal{M}_{4\text{-type}}$ or it returns the information that the model is unfeasible, i. e., such a scenario does not exist.

As proposed for the iTDRL rearrangement in Section 4.2, single rearrangements can in certain cases be mimicked by a sequence of other rearrangements. For the case that only weighted inversion, transpositions, and inverse transpositions are considered, Bernt et al. (2013d) have analyzed all possible alternatives to replace a single rearrangement. The authors proved that all three types of rearrangements may be part of a parsimonious scenario if the following inequalities are satisfied (see also Figure 5.3):

$$\omega_T < \omega_I + \omega_{iT}, \quad (5.1) \qquad \omega_{iT} < 2\omega_I, \quad (5.5)$$
$$\omega_T < 3\omega_I, \quad (5.2) \qquad \omega_I < \omega_T + \omega_{iT}, \quad (5.6)$$
$$\omega_T < 2\omega_{iT}, \quad (5.3) \qquad \omega_I < 3\omega_{iT}. \quad (5.7)$$
$$\omega_{iT} < \omega_I + \omega_T, \quad (5.4)$$

It is easy to see that if transpositions, inversions, and inverse transpositions are weighted equally, than a single rearrangement cannot be replaced by single rearrangement of another type. Combining this with the facts that a single inversion (also inverse transposition) can-

not replace a TDRL and that a transposition can be seen as a special TDRL, it follows that if `GeRe-ILP` is iteratively executed for increasing values of t, then the first found solution is an optimal solution. In the case that at least one rearrangement can be replaced by a sequence of other rearrangements, e. g., at least one of inequalities (5.1) to (5.7) is violated, an upper bound on the maximum length of the scenario t is obtained by $t = \mathrm{obj}/\min\{\omega_I, \omega_T, \omega_{iT}, \omega_{TDRL}\}$, where obj denotes the objective value of the first solution that is found. In contrast to the case where no rearrangement can be mimicked, the ILP is solved for all $k \in [0:t]$, which gives an optimal solution. In this case, the objective value of the best feasible solution is always added as an upper bound on the objective value in the succeeding executions of the ILP.

Although `GeRe-ILP` can exactly solve the weighted sorting problem for two signed permutations under $\mathcal{M}_{4\text{-type}}$, it is unrealistic that it can be applied to complete mitochondrial gene orders that are separated by several rearrangements. The reason for this is its exponential runtime behavior. However, it can be used for moderate-sized gene orders, e. g., the order of mitochondrial protein-coding genes, as explored in Hartmann et al. (2017). In this work another approach is investigated: `GeRe-ILP` is used as a subroutine of the `CREx2` algorithm to solve computational hard (but in many cases sufficiently smaller) subproblems in a common interval framework as proposed in the following section.

5.4 SORTING BY WEIGHTED PRESERVING REARRANGEMENTS

Often times in nature one can observe sets of genes that are similar in close proximity in several genomes (Lathe 3rd et al., 2000). Such sets of genes are called gene clusters. Gene clusters might have been formed by functional constraints, evolutionary inertia, or by chance. Considering that maintaining functional genomes might have inhibited the destruction of certain gene clusters, it is most likely that those gene clusters are also present in the ancestral genomes. Based on that idea, algorithms should enforce scenarios of rearrangements which do not break those gene clusters in all (intermediate) gene orders. Such scenarios and the corresponding rearrangements are called *preserving*. A simple and formal concept to model clusters of genes in genomes is to consider common combinatorial structures between gene orders. In this chapter, gene clusters of unichromosomal genomes are modeled by common intervals. Common intervals are intervals of consecutive genes that appear in the considered gene orders. For a broad overview on computational approaches regarding common intervals see Section 2.2.2.

Rearrangement problems that account for common intervals have already been studied. The idea to make use of common intervals for the comparison of gene orders has been presented in Heber and Stoye (2001b) and Heber and Stoye (2001a). In particular, the distance problem and the sorting problem for preserving inversions was studied intensively (Bérard et al., 2004; Bergeron et al., 2002a; Figeac and Varré,

2004; Swenson et al., 2009; Swenson and Moret, 2009). The main idea of the presented algorithms, e. g., Bergeron et al. (2008) and Heber et al. (2011), is to use a generating subset of the common intervals as outlined in Section 2.2.2. In Figeac and Varré (2004) the distance problem for preserving inversions was shown to be NP-hard. Nevertheless, by using the strong interval tree data structure (Bergeron et al., 2008) which encodes all common intervals efficiently, a fixed-parameter tractable algorithm with linear runtime has been proposed for all instances for which the common intervals are organized in a certain linear structure (Bérard et al., 2007; Bérard et al., 2008). By detecting patterns in the strong interval tree, two algorithms have been described that heuristically compute rearrangement scenarios: one algorithm in Parida (2006) and algorithm `CREx` in Bernt et al. (2007). The latter algorithm is more general and considers the $\mathcal{M}^{p}_{4\text{-type}}$ model that contains all preserving rearrangements that are predominant in metazoan mitochondrial genomes.

The preservation of common intervals also provides significant advantages for reconstructing evolutionary scenarios of mitochondrial gene orders:

Recall that the $\mathcal{M}_{4\text{-type}}$ model is insufficiently suited for reconstructing pairwise mitochondrial gene order scenarios as parsimonious scenarios contain almost entirely tandem duplication random losses that affect the whole gene order (see Section 5.2.3). In a common interval framework, TDRLs (and also the other types of rearrangements) are enforced to act locally on smaller subsets of genes in order to ensure the preservation of the common intervals. Note that this also reflects the fact that short rearrangements are found more often than long ones (Lefebvre et al., 2003).

The insights gained in Section 5.2.3 also show that the $\mathcal{M}_{4\text{-type}}$ distance proves to be only a rough genomic measure. This especially holds if the $\mathcal{M}_{4\text{-type}}$ distance (which grows asymptotically not faster than $\log_2 n$ for gene orders having n genes) is compared to other genomic (dis)similarity measures that grow asymptotically not faster than n, e. g., the $\mathcal{M}_{I}$ distance. However, by using the common interval framework the corresponding $\mathcal{M}^{p}_{4\text{-type}}$ distance grows asymptotically not faster than n as well. This is because a rearrangement can potentially be applied for every non-overlapping common interval which are of number $\mathcal{O}(n)$.

For all these reasons, this section investigates the weighted sorting problem for two given signed linear permutations under the $\mathcal{M}^{p}_{4\text{-type}}$ rearrangement model. This model considers the types of rearrangements inversion (I), transposition (T), inverse transposition (iT), and tandem duplication random loss (TDRL) that preserve the common intervals of the two given permutations. In addition, each rearrangement is considered to be weighted with respect to its type $Z \in \{I, T, iT, TDRL\}$. This section proposes an algorithm, called `CREx2`, that improves the `CREx` heuristic in the following aspects:

i) `CREx2` allows the incorporation of rearrangement weights, and

ii) CREx2 is able to provide an exact solution even in the case that the common intervals are organized in a non-linear structure.

CREx2 has a linear runtime if the common intervals are organized in a linear structure. Otherwise, if the common intervals are organized in a non-linear structure then CREx2 either produces an approximated solution in time $\mathcal{O}(n^2 \log n)$, where n is the size of the input permutations, or an exact solution within exponential runtime. In addition it is shown empirically for simulated and biological data sets that CREx2 computes results of high accuracy.

The chapter is organized as follows. Definitions that are necessary for the following formal analysis are given in Section 5.4.1. Theoretical results are shown in Section 5.4.2 and Section 5.4.3. CREx2 is presented in Section 5.4.4. An evaluation of the accuracy of the CREx2 results is performed on simulated and real biological gene order data sets in Section 5.5. A conclusion is given in Section 5.6.

5.4.1 *Common Intervals and Strong Interval Trees*

Recall the formal definitions of common intervals of a set of (signed) permutations and the strong interval tree that efficiently represents the complete set of common intervals. Nevertheless, these definitions should be recalled here (for references and examples see Section 2.2.2).

An interval of a permutation π is a set of unsigned elements that occur consecutively in π. A common interval of a set of permutations $\Pi \subseteq s\mathcal{P}_n$ is an interval that occurs in each permutation of Π. The set of all common intervals of Π is denoted by $C(\Pi)$. A permutation π is *consistent* to Π if $C(\Pi) = C(\Pi \cup \pi)$. The common intervals of Π can be represented elegantly by using a generating subset of the common intervals called the *strong common intervals*. A common interval $I \in C(\Pi)$ is *strong* if every other common interval $J \in C(\Pi)$ is either disjoint, included in I, or includes I, i. e., $I \cap J = \emptyset$, $J \subseteq I$, or $I \subseteq J$. It is not hard to see that every two strong intervals do not overlap, i. e., they are either disjoint or one includes the other. Therefore, the strong intervals of Π form a hierarchy which allows to represent them efficiently. The representation used for this purpose is the strong interval tree (SIT). The SIT is the central data structure for efficient preserving rearrangement analysis. The main reason is that the SIT can be computed in linear time and represents the common intervals (which can be quadratic in number) in linear space (Bérard et al., 2007). More formally, consider a permutation $\lambda \in s\mathcal{P}_n$ that is consistent to Π. The *strong interval tree* $T^\lambda(\Pi)$ of Π and λ is an ordered and rooted tree in which the nodes correspond to the strong common intervals of Π. Two nodes N_1 and N_2 of $T^\lambda(\Pi)$ are connected by an edge if one includes the other (without loss of generality $N_1 \subset N_2$) and there is no strong common interval $N \in C(\Pi)$ with $N_1 \subset N \subset N_2$. The child nodes of a node of $T^\lambda(\Pi)$ are ordered as the corresponding intervals in λ. Let $\pi \in s\mathcal{P}_n$ be consistent with Π and N be an inner node of $T^\lambda(\Pi)$ with child nodes $N_1, \ldots, N_{\deg(N)}$, where $\deg(N)$ denotes

the number of child nodes N. The *quotient permutation* of N (with respect to π) is the permutation $\pi_{|N}$ which satisfies that $\pi_{|N}(i)$ precedes $\pi_{|N}(j)$ if and only if the interval N_i is to left of the interval N_j in π for $i \neq j$. A quotient permutation $\pi_{|N}$ is *linear increasing* (*linear decreasing*) if $\pi_{|N} = (1 \dots \deg(N))$ (respectively $\pi_{|N} = (\deg(N) \dots 1)$). A quotient permutation $\pi_{|N}$ is called *prime* if it is neither linear increasing nor linear decreasing. Node N is *linear* (*prime*) with respect to π if $\pi_{|N}$ is linear increasing or linear decreasing (respectively prime). Consequently, for a permutation π, there are three types of nodes in the SIT $T^{\lambda}(\Pi)$, namely linear nodes, leaf nodes, and prime nodes. If all inner nodes of a SIT are linear (with respect to π), then the SIT is called *linear*. Otherwise, the SIT is called *prime*. Both types of SITs are considered in this chapter because both occur in biological applications. For example, pairwise comparisons of metazoan mitochondrial gene orders often correspond to instances with linear SITs (Bernt and Middendorf, 2011), this is different for fungal mitochondrial gene orders where the investigation performed in Hartmann et al. (2018a) shows that instances with linear SIT occur in less than 5% of the cases.

Recall the definition of the different types of rearrangements that are assumed to be predominant in the evolution of mitochondrial gene orders, namely inversion (I), transposition (T), inverse transposition (iT), and tandem duplication random loss (TDRL), given in Section 2.2.3. In this section, two special cases of inverse transpositions $\rho_{iT}(X, Y)$ for a permutation $\pi = (\pi(1) \ \dots \ \pi(n))$ with two consecutive disjoint intervals X and Y are considered: *prefix inverse transpositions* ($\rho_{piT}(X, Y)$), where $X = \{|\pi(1)|, \dots, |\pi(n-1)|\}$, $Y = \{|\pi(n)|\}$ and *suffix inverse transpositions* ($\rho_{siT}(X, Y)$), where $X = \{|\pi(2)|, \dots, |\pi(n)|\}$ and $Y = \{|\pi(1)|\}$. Similarly to inverse transpositions, also a special case of TDRL rearrangement is necessary for technical reasons. A TDRL $\rho_{TDRL}(L, R)$ for π is called *minimal* if for all proper subsets $L' \subset L$, $R' \subset R$ it holds that $\rho_{TDRL}(L', R) \circ \pi \neq \rho_{TDRL}(L, R) \circ \pi \neq (L, R') \circ \pi$. It is not hard to see that for each TDRL $\rho_{TDRL}(L, R)$ there exists a uniquely defined minimal TDRL $\rho_{TDRL}(L', R')$ with $L' \subseteq L$ and $R' \subseteq R$, see Example 5.5.

Example 5.5. *Consider the set* $\Pi = \{(1 \ -2 \ 4 \ 3 \ -6 \ 5), \iota\}$ *and the permutation* $\lambda = (1 \ 2 \ 3 \ 4 \ -6 \ 5)$ *consistent to* Π *that are considered in Example 2.4 in Section 2.2.2. For equally weighted rearrangements, i.e.,* $\omega_I = \omega_{iT} = \omega_T = \omega_{TDRL}$, *a parsimonious scenario for* $(1 \ -2 \ 4 \ 3 \ -6 \ 5)$ *and* ι *that is consistent for* Π *is* $(\rho_T(\{3\}, \{4\}), \rho_I(\{2\}), \rho_{iT}(\{6\}, \{5\}))$ *and* $\omega(S) = 3$. *A TDRL for* λ *is* $\rho_{TDRL}(\{1, 2, 4, 5\}, \{3, 6\})$ *and the corresponding minimal TDRL is* $\rho_{TDRL}(\{4, 5\}, \{3, 6\})$.

For the following section, it is necessary to extend the definition of a strong interval tree as done in the following. The idea is to assign a sign to each node of a given SIT which represents the orientation of the corresponding strong common interval with respect to a target permutation. Let $\Pi \subseteq s\mathcal{P}_n$ and $\lambda, \pi \in s\mathcal{P}_n$ be consistent with Π. The *signed strong interval tree* (sSIT) of Π, λ, and π, denoted by $T^{\lambda}(\pi, \Pi)$, is the strong interval tree of Π and λ, where nodes that are linear with respect to π and leaves have an additional sign that is determined

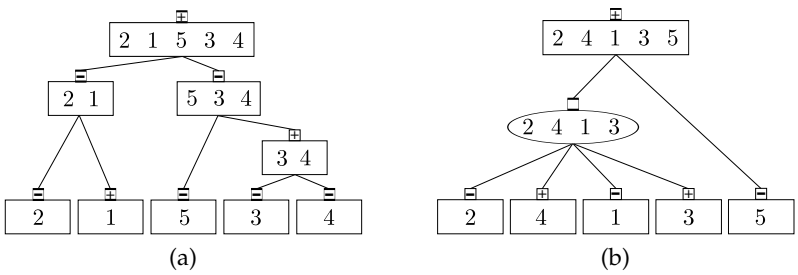


Figure 5.4: (a): Linear sSIT $T^{\lambda}(\iota, \{\lambda, \iota\})$ with $\lambda = (-2\ 1\ -5\ -3\ -4)$ and $\iota = (1\ 2\ 3\ 4\ 5)$. Linear (prime) nodes are represented by rectangles (respectively ellipses). The sign of a node is shown at the top of the rectangles. The common intervals of $\{\lambda, \iota\}$ are $\{1,2\}$, $\{3,4\}$, $\{3,4,5\}$, $\{1,2,3,4,5\}$, $\{1\}$, $\{2\}$, $\{3\}$, $\{4\}$, and $\{5\}$ and each of these intervals is strong as well. The node $\{5,3,4\}$ is linear decreasing since $\iota_{|\{5,3,4\}} = (2\ 1)$ and the node $\{3,4\}$ is linear increasing since $\iota_{|\{3,4\}} = (1\ 2)$. The signed quotient permutation of node $\{5,3,4\}$ is $\hat{\pi}_{|\{5,3,4\}} = (-2\ 1)$. (b): Prime sSIT $T^{\lambda'}(\iota, \{\lambda', \iota\})$ with $\lambda' = (-2\ 4\ -1\ 3\ -5)$. The interval $\{1,3,4\}$ is a prime-sibling.

as follows: 1) a linear inner node N gets the sign $+$ (respectively $-$) if $\pi_{|N}$ is linear increasing (respectively decreasing) with respect to π, 2) a leaf node gets the sign $+$ if the corresponding element has the same sign in π and λ and the sign $-$ otherwise. Note that no sign is assigned to a prime node. The sign of a linear node or leaf node N is denoted by $\mathrm{sign}(N)$. With $-\mathrm{sign}(N)$ is the opposite sign of N is denoted, i. e., $-\mathrm{sign}(N) = +$ (respectively $-\mathrm{sign}(N) = -$) if $\mathrm{sign}(N) = -$ (respectively $\mathrm{sign}(N) = +$). The *signed quotient permutation* $\hat{\pi}_{|N}$ of N with respect to π is the quotient permutation $\pi_{|N}$ in which each element is assigned the sign of its corresponding child node, i. e., the i-th element of $\hat{\pi}_{|N}$ is assigned to the sign of the child node N_i. Observe that if the sign of a child node is unknown, then the signed quotient permutation is partially signed. An interval X of λ is called a *prime-sibling* with respect to π and Π if X is a union of child nodes of a prime node in $T^{\lambda}(\pi, \Pi)$. Figure 5.4 gives examples for the definitions related to signed strong interval trees.

The preservation of common intervals of a set of permutations Π in a sequence of rearrangements is formally defined as follows. A rearrangement $\rho \in \mathcal{M}_{4\text{-type}}$ for a permutation π that is consistent to Π is *preserving* for Π if $\rho \circ \pi$ is consistent with Π, i. e., $C(\Pi) = C(\{\rho \circ \pi\} \cup \Pi)$. Analogously, a sequence (scenario) $(\rho_1, \ldots, \rho_t)$ of 4-type rearrangements for π and σ is *preserving* for Π if for all $i \in [1:t]$ the permutations $\rho_i \circ \ldots \circ \rho_1 \circ \pi$ is consistent with Π.

For the convenience of the reader, the variation of the sorting problem that is considered in this section is recalled in the following. The *weighted preserving sorting problem* under $\mathcal{M}^{p}_{4\text{-type}}$ is to find for the set of rearrangements $\mathcal{M}^{p}_{4\text{-type}}$, a weight function $\omega : \mathcal{M}^{p}_{4\text{-type}} \to \mathbb{R}_{>0}$, and signed permutations $\lambda, \pi \in s\mathcal{P}_n$ parsimonious preserving scenario for λ and π. The weighted preserving sorting problem can also be formulated in terms of the sSIT: Find a parsimonious preserving scenario S for λ and π of 4-type rearrangements such that the sSIT $T^{\lambda}(\pi, \{\pi, \lambda\})$

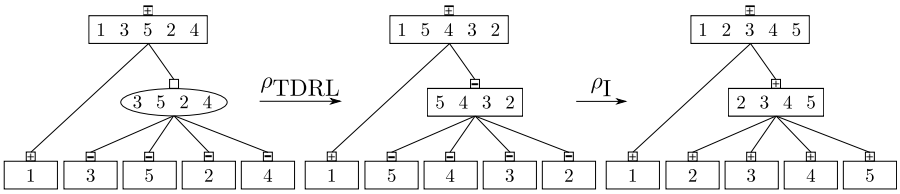


Figure 5.5: Parsimonious scenario $S = (\rho_{\text{TDRL}}(\{4,5\},\{2,3\}), \rho_{\text{I}}(\{2,3,4,5\})$ for $\lambda = (1\ -3\ -5\ -2\ -4)$ and ι that is preserving for $\{\lambda, \iota\}$. Scenario S is a solution of the weighted preserving sorting problem defined by λ, π, $\mathcal{M}^{p}_{4\text{-type}}$, and the rearrangement weights $\omega_{\text{I}} = \omega_{\text{T}} = \omega_{\text{iT}} = \omega_{\text{TDRL}} = 1$. Illustrated are from the left to the right $T^{\lambda}(\iota,\{\lambda,\iota\})$, $T^{\rho_{\text{TDRL}}\circ\lambda}(\iota,\{\lambda,\iota\})$, and $T^{\rho_{\text{I}}\circ\rho_{\text{TDRL}}\circ\lambda}(\iota,\{\lambda,\iota\})$. Observe that ρ_{TDRL} transforms the prime node $\{2,3,4,5\}$ in $T^{\lambda}(\iota,\{\lambda,\iota\})$ into a linear node in $T^{\rho_{\text{TDRL}}\circ\lambda}(\iota,\{\lambda,\iota\})$ that has a negative sign.

is transformed into $T^{S\circ\lambda}(\pi,\{\pi,\lambda\})$ in which all prime nodes become linear and each node has sign $+$. See Figure 5.5 for an example of a parsimonious preserving scenario.

5.4.2 *Generalized Preserving Rearrangements*

In this section some theoretical results for preserving scenarios are shown. The insights gained in this section are crucial for solving the weighted preserving sorting problem. In particular, the rearrangements that preserve the common intervals of a given set of permutations are determined. Throughout the section the following notations are used. A given set of signed linear permutations of size n is denoted by Π. Moreover, $\lambda \in s\mathcal{P}_n$ and $\pi \in s\mathcal{P}_n$ are consistent with Π.

The following proposition is adapted from Bérard et al. (2007). It is one reason for the success of the sSIT data structure for computing preserving rearrangements.

Proposition 5.4 (Bérard et al., 2007)**.** *Let* I *be an interval of a permutation* $\pi' \in \Pi$*. Then,* $I \in C(\Pi)$ *if and only if* I *is a node of* $T^{\lambda}(\pi,\Pi)$ *or the union of consecutive child nodes of a linear node of* $T^{\lambda}(\pi,\Pi)$*.*

Proposition 5.4 is the foundation for specifying rearrangements that preserve common intervals in terms of the sSIT. Such a specification has been presented for inversions in Bérard et al., 2007. The following theorem generalizes this specification.

Theorem 5.2. *Let* $S = (\rho_1,\ldots,\rho_t)$ *be a sequence for* λ, $\lambda_i := \rho_i \circ \ldots \circ \rho_1 \circ \lambda$ *with* $i \in [1\colon t]$*, and* $\lambda_0 := \lambda$*. Then,* S *is preserving for* Π *if and only if for all* $j \in [0\colon t-1]$ *each linear node in* $T^{\lambda_j}(\pi,\Pi)$ *is a linear node in* $T^{\lambda_{j+1}}(\pi,\Pi)$*.*

Proof. Assume that S is preserving for Π, i.e., for all $j \in [1\colon t]$ the permutation λ_j is consistent with Π. Let $I \in C(\Pi)$ be a node in $T^{\lambda_j}(\pi,\Pi)$ with $j \in [0\colon t-1]$. By the definition of the sSIT, the strong interval I is also a node in $T^{\lambda_{j+1}}(\pi,\Pi)$, since the nodes in $T^{\lambda_j}(\pi,\Pi)$ and $T^{\lambda_{j+1}}(\pi,\Pi)$ are the strong intervals of Π. Now, let $I \in C(\Pi)$ be a linear node in $T^{\lambda_j}(\pi,\Pi)$ with child nodes $I_1,\ldots,I_{\deg(I)}$ in that

order. The order of the child nodes of I in $T^{\lambda_{j+1}}(\pi, \Pi)$ is either $I_1, \ldots, I_{\deg(I)}$ or its reverse, i. e., $I_{\deg(I)}, \ldots, I_1$. This can be seen by the following argumentation. By contradiction assume that the order of the child nodes of I in $T^{\lambda_{j+1}}(\pi, \Pi)$ is neither $I_1, \ldots, I_{\deg(I)}$ nor $I_{\deg(I)}, \ldots, I_1$. Hence, there exist two child nodes I_i and I_{i+1} of I that are not consecutive in $T^{\lambda_{j+1}}(\pi, \Pi)$ whereas they are consecutive in $T^{\lambda_j}(\pi, \Pi)$. By Proposition 5.4 the set $I_i \cup I_{i+1}$ is a common interval of $\Pi \cup \lambda_j$, since it is a union of consecutive child nodes of a linear node, i. e., $I_i \cup I_{i+1} \in C(\Pi \cup \lambda_j)$. In addition, by Proposition 5.4 it holds that $I_i \cup I_{i+1} \notin C(\Pi \cup \lambda_{j+1})$, since I_i and I_{i+1} are neither consecutive child nodes of a linear node in $T^{\lambda_{j+1}}(\pi, \Pi)$ nor is $I_i \cup I_{i+1}$ a node in $T^{\lambda_{j+1}}(\pi, \Pi)$. The first fact holds since I_i and I_{i+1} are not consecutive in $T^{\lambda_{j+1}}(\pi, \Pi)$. The latter fact holds since $I_i \cup I_{i+1}$ is not a node in $T^{\lambda_j}(\pi, \Pi)$ and since $T^{\lambda_j}(\pi, \Pi)$ and $T^{\lambda_{j+1}}(\pi, \Pi)$ have the same nodes, i. e., the strong common intervals of Π. Therefore, it holds that $I_i \cup I_{i+1} \in C(\Pi \cup \lambda_j)$ and $I_i \cup I_{i+1} \notin C(\Pi \cup \lambda_{j+1})$ which contradicts the assumption that S is preserving for Π since $C(\Pi) = C(\Pi \cup \lambda_j) \neq C(\Pi \cup \lambda_{j+1}) = C(\Pi)$, i. e., λ_{j+1} is not consistent with Π. Consequently, the order of the child nodes of I in $T^{\lambda_{j+1}}(\pi, \Pi)$ is either unchanged, i. e., $I_1, \ldots, I_{\deg(I)}$, or reversed, i. e., $I_{\deg(I)}, \ldots, I_1$. In both cases, I is linear in $T^{\lambda_{j+1}}(\pi, \Pi)$ proving the implication from left to right.

Assume that for all $j \in [0 : t-1]$ it holds that each linear node N in $T^{\lambda_j}(\pi, \Pi)$ is a linear node in $T^{\lambda_{j+1}}(\pi, \Pi)$. Consider a common interval $I \in C(\Pi \cup \lambda_j)$ with $j \in [0 : t-1]$. By Proposition 5.4 I is a node in $T^{\lambda_j}(\pi, \Pi)$ (i. e., I is a strong interval) or I is a union of consecutive child nodes of a linear node of $T^{\lambda_j}(\pi, \Pi)$. If I is a node in $T^{\lambda_j}(\pi, \Pi)$, then I is also a node in $T^{\lambda_{j+1}}(\pi, \Pi)$, since (by the definition of a sSIT) the nodes of a sSIT $T^{\sigma_1}(\sigma_2, \Sigma)$, with $\Sigma \subseteq s\mathcal{P}_n$ and $\sigma_1, \sigma_2 \in s\mathcal{P}_n$ consistent with Σ, are the strong common intervals of Σ which are not influenced by σ_1 or σ_2. If I is a union of consecutive child nodes of a linear node N in $T^{\lambda_j}(\pi, \Pi)$, then $I \in C(\Pi \cup \lambda_{j+1})$ since consecutive child nodes of a linear node in $T^{\lambda_j}(\pi, \Pi)$ are also consecutive in $T^{\lambda_{j+1}}(\pi, \Pi)$. This can be seen by the following argumentation. Since N is linear in $T^{\lambda_j}(\pi, \Pi)$ and $T^{\lambda_{j+1}}(\pi, \Pi)$ the order of the child nodes of N is either unchanged, i. e., N is linear increasing (respectively decreasing) in $T^{\lambda_j}(\pi, \Pi)$ and $T^{\lambda_{j+1}}(\pi, \Pi)$, or reversed, i. e., N is linear increasing (decreasing) in $T^{\lambda_j}(\pi, \Pi)$ and linear decreasing (respectively increasing) in $T^{\lambda_{j+1}}(\pi, \Pi)$. Note that no other case satisfies that N is linear in $T^{\lambda_j}(\pi, \Pi)$ and $T^{\lambda_{j+1}}(\pi, \Pi)$. If the order of child nodes of N is unchanged, then obviously consecutive child nodes of N in $T^{\lambda_j}(\pi, \Pi)$ are also consecutive in $T^{\lambda_{j+1}}(\pi, \Pi)$. If the order of child nodes of N is reversed, then each two child nodes N_1, N_2 of N that are consecutive in $T^{\lambda_j}(\pi, \Pi)$ (in that order) are consecutive in $T^{\lambda_{j+1}}(\pi, \Pi)$ in the reversed order. In both cases it holds that $I \in C(\Pi \cup \lambda_{j+1})$. Consequently, for all $j \in [0 : t-1]$ λ_j is consistent with Π, i. e., S is preserving for Π. □

The following corollary of Theorem 5.2 shows that a preserving sequence S for λ retains the consecutiveness of the child nodes of

linear nodes of $T^{\lambda}(\pi, \Pi)$. With other words, consecutive child nodes of a linear node in $T^{\lambda}(\pi, \Pi)$ stay consecutive in $T^{S \circ \lambda}(\pi, \Pi)$.

Corollary 5.2. *Let $S = (\rho_1, \ldots, \rho_t)$ be a sequence for λ that is preserving for Π, $\lambda_i := \rho_i \circ \ldots \circ \rho_1 \circ \lambda$ with $i \in [1:t]$, and $\lambda_0 := \lambda$. For each linear node N in $T^{\lambda_j}(\pi, \Pi)$ with $j \in [0:t-1]$, it holds that consecutive child nodes of N are consecutive in $T^{\lambda_{j+1}}(\pi, \Pi)$.*

Proof. Let N be a linear node in $T^{\lambda_j}(\pi, \Pi)$, $j \in [1:t-1]$, with child nodes $N_1, \ldots, N_{\deg(N)}$ in that order. By Theorem 1 it holds that N is a linear node in $T^{\lambda_{j+1}}(\pi, \Pi)$. Hence, the order of the child nodes of N in $T^{\lambda_{j+1}}(\pi, \Pi)$ is either $N_1, \ldots, N_{\deg(N)}$ or its reverse, i. e., $N_{\deg(N)}, \ldots, N_1$. Consequently, for all $i \in [1: \deg(N) - 1]$ it holds that the consecutive child nodes N_i, N_{i+1} (of N in $T^{\lambda_j}(\pi, \Pi)$) are also consecutive in $T^{\lambda_{j+1}}(\pi, \Pi)$ (either in the same order, i. e., N_i, N_{i+1}, or in the reversed order, i. e., N_{i+1}, N_i). □

By Corollary 5.2 a preserving rearrangement can change the sSIT only as follows:

1) the order of the child nodes of a linear node N is reversed and the sign of N is toggled,

2) the sign of a leaf node is toggled, and

3) the order of the child nodes of a prime node is permuted.

In Case 3 the consequences for a prime node N are that either N becomes linear and gets a corresponding sign or N remains prime (and has no sign).

The following corollary of Theorem 5.2 specifies the preserving rearrangements for several types of rearrangements in terms of the sSIT.

Corollary 5.3. *Let ρ be a rearrangement for λ that is an inversion, transposition, inverse transposition, or a minimal TDRL. Rearrangement ρ is preserving for Π if and only if one of the following cases holds:*

i) $\rho = \rho_I(X)$, *where X is a prime-sibling with respect to π and Π or* $\rho = \rho_Z(X, Y)$ *with $Z \in \{T, iT, \text{(minimal) TDRL}\}$, where X and Y are prime-siblings with respect to π and Π;*

ii) $\rho = \rho_I(X)$, *where X is a linear node in $T^{\lambda}(\pi, \Pi)$;*

iii) $\rho = \rho_T(X, Y)$, *where X and Y are the only child nodes of a linear node $X \cup Y$ in $T^{\lambda}(\pi, \Pi)$;*

iv) $\rho = \rho_{iT}(X, Y)$, *where Y is the first or last child of a linear node $X \cup Y$ in $T^{\lambda}(\pi, \Pi)$.*

Proof. The formal argument proceeds in two steps. The implication from left to right is proven first, subsequently the opposite direction is proven.

Assume that ρ is a rearrangement for λ that is preserving for Π and ρ is an inversion, a transposition, an inverse transposition, or a minimal TDRL. Since ρ is preserving $T^{\lambda}(\pi, \Pi)$ and $T^{\rho \circ \lambda}(\pi, \Pi)$ have

the same nodes. Therefore, ρ can change only the order of the child nodes of some nodes, change the sign of nodes, or add a sign to a node that was prime in $T^{\lambda}(\pi, \Pi)$ and becomes linear in $T^{\rho\circ\lambda}(\pi, \Pi)$.

If ρ is an inversion $\rho_I(X)$ and X contains a single element, then X is a leaf in $T^{\lambda}(\pi, \Pi)$ and the Case (ii) holds. Otherwise, ρ changes the order of (at least) two elements and therefore changes the order of at least two child nodes of some node. Let N be a highest node in $T^{\lambda}(\pi, \Pi)$ for which the order of its child nodes is changed, i. e., the order of the child nodes of all predecessors of N is not changed. Let $N_1, \ldots, N_{\deg(N)}$ be the child nodes of N in $T^{\lambda}(\pi, \Pi)$ in that order. From the possible types of rearrangements it can be seen that for all nodes N' which are not within the subtree with root N of $T^{\lambda}(\pi, \Pi)$, the order of the child nodes is not changed. Moreover, if ρ is a transposition, an inverse transposition, or a minimal TDRL, then for each child node N_i, $i \in [1\colon \deg(N)]$ one of the following cases holds: 1) $N_i \subset X$, 2) $N_i \subset Y$, 3) $N_i \cap X = \emptyset$ and $N_i \cap Y = \emptyset$. Similarly, if ρ is an inversion for each child node N_i, $i \in [1\colon \deg(N)]$ one of the following cases holds: $N_i \subset X$ or $N_i \cap X = \emptyset$. For the inversion $\rho_I(X)$, the transposition $\rho_T(X, Y)$, and the inverse transposition $\rho_{iT}(X, Y)$ it holds that X is an interval and therefore it is of the form $X = N_i \cup \ldots \cup N_j$ for $1 \leqslant i \leqslant j \leqslant \deg(N)$. Similarly, for $\rho_T(X, Y)$, and $\rho_{iT}(X, Y)$ it holds that Y is an interval and therefore it is of the form $Y = N_k \cup \ldots \cup N_\ell$ for $1 \leqslant k \leqslant \ell \leqslant \deg(N)$ where either $j < k$ or $\ell < i$ holds. For the minimal TDRL $\rho_{TDRL}(X, Y)$ it holds that $X \cup Y$ is an interval and therefore it is of the form $X \cup Y = N_i \cup \ldots \cup N_j$ for $1 \leqslant i \leqslant j \leqslant \deg(N)$.

Assume that N is a prime node. If ρ is an inversion $\rho_I(X)$ it follows that X is a prime sibling. Similarly, if ρ is a transposition $\rho_T(X, Y)$, or an inverse transposition $\rho_{iT}(X, Y)$ it follows that X and Y are prime siblings and if ρ is a minimal TDRL $\rho_{TDRL}(X, Y)$ then $X \cup Y$ is a prime sibling.

Now assume that N is a linear node. By Corollary 5.2 it follows that consecutive child nodes of N in $T^{\lambda}(\pi, \Pi)$ are also consecutive in $T^{\rho\circ\lambda}(\pi, \Pi)$.

If ρ is an inversion $\rho_I(X)$, then for $X = N_i \cup \ldots \cup N_j$ it follows that $i = 1$ and $j = \deg(N)$. The reason is that for $i > 1$ ($j < \deg(N)$) the consecutiveness on child nodes N_{i-1} and N_i (respectively N_j and N_{j+1}) would be violated.

If $\rho = \rho_T(X, Y)$, then N must have exactly two child nodes. Therefore, either $X = N_1$ and $Y = N_2$ or $X = N_2$ and $Y = N_1$ holds. To see this assume $\deg(N) \geqslant 3$. Consider $X = N_i \cup \ldots \cup N_j$ and $Y = N_k \cup \ldots \cup N_\ell$ for the case $j < k$ (the case $\ell < i$ is fully analogous). Since ρ is a transposition N_j and N_k are consecutive, i. e., $j = k - 1$ must hold. Thus, one of the following cases holds true $j \geqslant 2$ or $k < \deg(N)$. In the former case either $N_{j-1} \notin X$ (which implies that ρ would violate the consecutiveness of N_{j-1} and N_j) or $N_{j-1} \in X$ (which implies that ρ would destroy the consecutiveness of N_j and N_k). The latter case can be shown analogously.

If $\rho = \rho_{iT}(X, Y)$, then for $X = N_i \cup \ldots \cup N_j$ and $Y = N_k \cup \ldots \cup N_\ell$ either $k = \ell = \deg(N)$ or $1 = k = \ell$ must hold, i. e., Y contains the elements of either only N_1 or node $N_{\deg(N)}$. To see this, consider the

case $j < k$ first. Since ρ is an inverse transposition $j = k - 1$ must hold true. Then $k < \ell$ is not possible because the consecutiveness of N_j and N_k would be violated by ρ. Now, $k = \ell < \deg(N)$ is not possible because ρ would violate the consecutiveness of N_ℓ and $N_{\ell+1}$. Now $1 < i$ is not possible because ρ would violate the consecutiveness of N_{i-1} and N_i. Hence, $i = 1$, $j = \deg(N) - 1$ and $k = \ell = \deg(N)$ must hold. Similarly, it can be shown for case $\ell < i$ that $1 = k = \ell$, $i = 2$ and $j = \deg(N)$ must be true.

The case that ρ is a minimal TDRL $\rho_{\text{TDRL}}(X, Y)$ with $X \cup Y = N_i \cup \ldots \cup N_j$ for $1 \leqslant i \leqslant j \leqslant \deg(N)$ remains. Assume that there exist $i' < i'' < i'''$ with $N_{i'} \subset X$, $N_{i'''} \subset X$, and $N_{i''} \subset Y$. Then ρ would destroy the consecutiveness of $N_{i'}$ and $N_{i''}$. Similarly, there cannot exist $i' < i'' < i'''$ with $N_{i'} \subset Y$, $N_{i'''} \subset Y$, and $N_{i''} \subset X$. Thus, there must exist an i' with $i \leqslant i' < j$ and $X = N_i \cup \ldots \cup N_{i'}$ and $Y = N_{i'+1} \cup \ldots \cup N_j$. Hence, ρ is a transposition $\rho_T(X, Y)$. As in the proof of Case (ii), it follows that X and Y are the only child nodes of a node $X \cup Y$ in $T^\lambda(\pi, \Pi)$. Consequently, the implication from left to right is true.

Theorem 5.2 shows that ρ is preserving if the following property $(*)$ holds: linear nodes in $T^\lambda(\pi, \Pi)$ are linear in $T^{\rho \circ \lambda}(\pi, \Pi)$. Let ρ be a rearrangement for which one of the cases (i)–(iv) holds. It remains to shows that property $(*)$ holds for ρ. If (i) holds, then ρ changes only the order of the child nodes of a prime node of $T^\lambda(\pi, \Pi)$. Hence property $(*)$ holds. It is not hard to show that in all the cases (ii)–(iv) property $(*)$ holds as well. □

To describe the consequences of Corollary 5.3 for finding parsimonious preserving scenarios, the following definition is needed. Consider a rearrangement $\rho \in \mathcal{M}^p_{\text{4-type}}$ that is preserving for Π. Therefore, rearrangement ρ can be expressed as $\rho_I(X)$ or $\rho_Z(X, Y)$ with $Z \in \{I, T, iT, \text{(minimal) TDRL}\}$. Rearrangement ρ *acts* on a node N of $T^\lambda(\pi, \Pi)$ if X (respectively $X \cup Y$) is a node in $T^\lambda(\pi, \Pi)$ or is a union of child nodes of a node N in $T^\lambda(\pi, \Pi)$. Corollary 5.3 shows that each of the considered rearrangements acts on a node of $T^\lambda(\pi, \Pi)$. Another consequence of Corollary 5.3 is that a preserving rearrangement of type $Z \in \{I, T, iT, \text{(minimal) TDRL}\}$ that acts on a linear node N is uniquely determined, i. e., if there exist two rearrangements ρ and ρ' of type Z that act on N, then $\rho = \rho'$. More precisely, if ρ is a preserving rearrangement that acts on a linear node N of $T^\lambda(\pi, \Pi)$, then the following observations are implied by Corollary 5.3:

1) If ρ is an inversion $\rho_I(X)$, then $N = X$. Hence, the effect of applying ρ to λ in the sSIT $T^\lambda(\pi, \Pi)$ is that it reverses the order of subtree rooted at node N. In addition, the sign of N and all signed nodes below N is toggled.

2) If ρ is a transposition $\rho_T(X, Y)$, then $X = N_1$ and $Y = N_2$, where N_1 and N_2 are the only child nodes of N. Hence, the effect of applying ρ to λ in the sSIT $T^\lambda(\pi, \Pi)$ is that it swaps the subtrees rooted at N_1 and N_2. In addition, the sign of N is toggled.

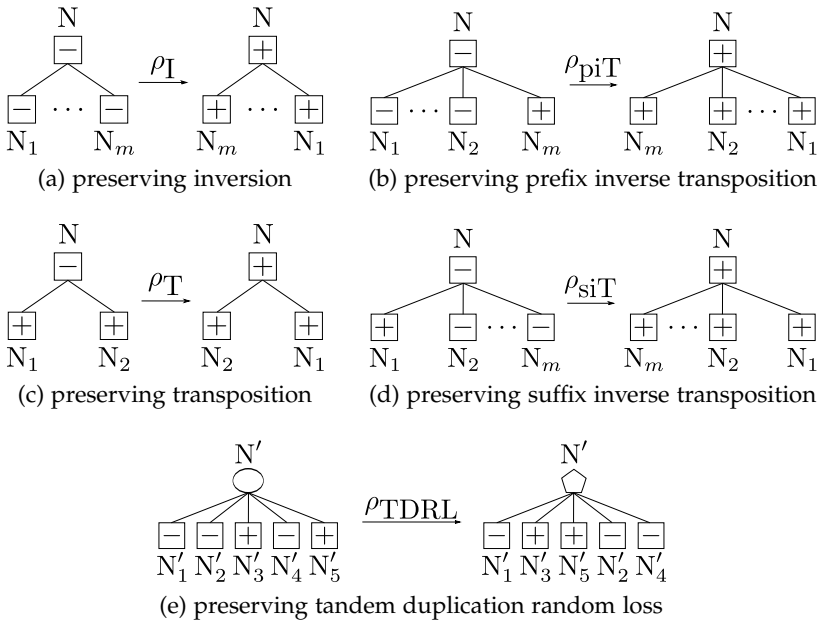


Figure 5.6: Preserving rearrangements (a) ρ_{I}; (b) ρ_{piT}; (c) ρ_{T}; and (d) ρ_{siT} that act on a linear node N. An example of a preserving rearrangement ρ_{TDRL} (e) that acts on a prime node N' which is illustrated by an ellipse. Node N, its child nodes $N_1, \ldots, N_m$, and the child nodes $N'_1, \ldots, N'_5$ of N' are represented by their signs. Illustrated is the case in which all child nodes of N and N' are linear. If a child node is prime, then it has no sign. A pentagon illustrates that the node N' can either remain prime or it becomes linear (and gets a corresponding sign) by the application of the TDRL. Observe that the rearrangement ρ_{TDRL} does not change the signs of the child nodes of N'.

3) If ρ is an inverse transposition $\rho_{\mathrm{iT}}(X, Y)$, then it is either a prefix inverse transposition or a suffix inverse transposition. The effect in the sSIT $T^{\lambda}(\pi, \Pi)$ of applying ρ to λ is that the rightmost (leftmost) child node of N is moved to the leftmost (respectively rightmost) position and the relative order of remaining child nodes of N is reversed in the case that ρ is a prefix (respectively suffix) inverse transposition. In addition, the sign of N and the signed nodes below N (except the last (first) child node of N and all nodes in the subtree it is the root of) are toggled if ρ is a prefix (respectively suffix) inverse transposition.

4) If ρ is a minimal TDRL $\rho_{\mathrm{TDRL}}(X, Y)$, then it has the same effect as a transposition.

In addition, if N is a prime node, then the order of its child nodes and their signs can be changed arbitrarily. Consequently, the uniqueness described above does not necessarily hold for preserving rearrangements that act on a prime node of a sSIT. Figure 5.6 illustrates the effects of preserving rearrangements from $\mathcal{M}^{\mathrm{p}}_{\text{4-type}}$ that act on a node of a sSIT. The figure illustrates also the effect on the signs of the nodes.

For the following proposition, it is necessary to introduce the notation of a strict subsequence of a sequence or rearrangements. Con-

sider a sequence of rearrangements $S = (\rho_1, \ldots, \rho_t)$, $t \in \mathbb{N}$, for a permutation π, then a sequence $S' = (\rho_i, \ldots, \rho_j)$ with $1 \leqslant i \leqslant j \leqslant t$ is called a *strict subsequence* of S. The following proposition shows that there exist parsimonious preserving scenarios that have a specific structure.

Proposition 5.5. *Consider scenarios for λ and π that consist only of rearrangements of types I, T, iT, and minimal TDRL that are preserving for Π. There exists such a scenario $S = (\rho_1, \ldots, \rho_t)$ that is parsimonious and in which each rearrangement ρ_i, $i \in [1:t]$, acts on a node of $T^\lambda(\pi, \Pi)$ and for each node N of $T^\lambda(\pi, \Pi)$ the rearrangements that act on N are a strict subsequence of S. Moreover, for each order of the nodes of $T^\lambda(\pi, \Pi)$ there exists such a parsimonious scenario in which the subsequences of rearrangements that act on the different nodes have the same relative order as their nodes.*

Proof. Observe, there exists always a scenario for λ and π that is preserving for Π and consists only of inversions. Hence, there also exists a parsimonious scenario for λ and π that is preserving for Π. Let $S' = (\rho_1, \ldots, \rho_t)$ be such a parsimonious scenario. By Corollary 5.3 it holds that each rearrangement acts on a node in $T^\lambda(\pi, \Pi)$. Consider two rearrangements ρ_i and ρ_{i+1}, $i \in [1:t-1]$, of S' that act on different nodes N_i and N_{i+1} of $T^\lambda(\pi, \Pi)$. It holds that a parsimonious preserving scenario $S'' := (\rho_1, \ldots, \rho_{i-1}, \rho'_{i+1}, \rho'_i, \rho_{i+2}, \ldots, \rho_t)$ for π and λ exists such that i) either $\rho'_i = \rho_i$ or ρ'_i and ρ_i are of the same type, $\rho'_i(Y, X)$, and $\rho_i(X, Y)$ and ii) either $\rho'_{i+1} = \rho_{i+1}$ or ρ'_{i+1} and ρ_{i+1} are of the same type, $\rho'_{i+1}(Y, X)$, and $\rho'_i(X, Y)$. This can be seen by the following argumentation. Since the nodes N_i and N_{i+1} are different it holds that either $N_i \cap N_{i+1} = \emptyset$, $N_i \subset N_{i+1}$, or $N_{i+1} \subset N_i$.

Consider first the case $N_i \cap N_{i+1} = \emptyset$. Due to the hierarchical structure of the nodes of $T^\lambda(\pi, \Pi)$, it is easy to see that ρ_i (respectively ρ_{i+1}) changes only the order of nodes in a subtree rooted at N_i (respectively N_{i+1}). Since $N_i \cap N_{i+1} = \emptyset$ both subtrees are disjoint. Therefore, the order of the child nodes of N_i (respectively N_{i+1}) is unchanged by the application of ρ_{i+1} (respectively ρ_i). Consequently, ρ_{i+1} is a rearrangement for λ_{i-1} and ρ_i is a rearrangement for $\rho_{i+1} \circ \lambda_{i-1}$ and $\rho_{i+1} \circ \rho_i \circ \lambda_{i-1} = \rho_i \circ \rho_{i+1} \circ \lambda_{i-1}$. Therefore, it holds that S'' is a scenario for λ and π in this case.

Now consider the case $N_i \subset N_{i+1}$. Assume that N_i is a child node of N_{i+1}. Since ρ_{i+1} acts on N_{i+1} it can only change the relative order of its child nodes or it can inverse some of its child nodes. If ρ_{i+1} does not inverse N_i then clearly both sequences of rearrangements (ρ_i, ρ_{i+1}) and (ρ_{i+1}, ρ_i) have the same effect. Now consider the case that ρ_{i+1} inverses node N_i. If ρ_i is an inversion, transposition or inverse transposition, then it is clear that both sequences (ρ_i, ρ_{i+1}) and (ρ_{i+1}, ρ_i) have the same effect. If $\rho_i(X, Y)$ is a minimal TDRL, then the sequences (ρ_i, ρ_{i+1}) and (ρ'_{i+1}, ρ_i) have the same effect were $\rho'_{i+1} = (Y, X)$, i.e., the elements that are kept in the left copy and in right copy are exchanged. If N_i is a successor of N_{i+1} but not a child node the proof is similar. The remaining case $N_{i+1} \subset N_i$ can be carried out analogously to the case $N_i \subset N_{i+1}$.

It is not hard to see that the proposition follows by an iterative application of the described interchange of to pairs of neighbored rearrangements that act on different nodes. □

Proposition 5.5 shows that there exist parsimonious preserving scenarios that consist of consecutive strict subsequences which act on different nodes of a given sSIT. Algorithm CREx2 computes for given permutations λ and π such a parsimonious scenario S for which holds that: 1) S is preserving for $\{\pi, \lambda\}$ and 2) the strict subsequences of S that act on different nodes of $T^{\lambda}(\pi, \{\pi, \lambda\})$ have the same relative order as the bottom-up order of the nodes of $T^{\lambda}(\pi, \{\pi, \lambda\})$. Algorithm CREx2 is presented in Section 5.4.4.

5.4.3 *Weighted Preserving Rearrangements*

The various types of rearrangements occur during the evolution of mitochondrial gene orders of diverse taxa with different likelihoods. In order to compute rearrangement scenarios which reflect these likelihoods, it may be useful to consider a weighting scheme for the different types of rearrangements. In this section, the problem is investigated to compute the weights of preserving parsimonious scenarios of rearrangements that are weighted by their type.

Throughout the section the following notations are used. A set of signed linear permutations of size n is denoted by Π. Permutations $\pi \in s\mathcal{P}_n$ and $\lambda \in s\mathcal{P}_n$ are consistent with Π. With N a node of the sSIT $T^{\lambda}(\pi, \Pi)$ is denoted. Moreover, $\mathcal{M}^{p}_{4\text{-type}\,|\,N}$ denotes the set of all preserving rearrangements of type inversion (I), transposition (T), inverse transposition (iT), or (minimal) TDRL that act on N. For N and a sign $s \in \{+, -\}$ let $\Omega(N, s)$ denote the *minimum weight of a preserving scenario* S for λ and $S \circ \lambda$ such that all nodes in the subtree rooted at N in $T^{S\circ\lambda}(\pi, \Pi)$ have sign s. Observe that when N is the root of the sSIT, the value $\Omega(N, +)$ is the minimal weight of a preserving scenario for λ and π. In the following, it is proven that the values $\Omega(N, s)$ can easily be computed if N is a leaf node or a linear node and otherwise, i. e., N is a prime node, the computation of $\Omega(N, s)$ is NP-hard.

Weighted Preserving Rearrangements Acting on a Leaf Node

Consider the case that N is a leaf node. By Corollary 5.3 the sign of a leaf node can only be modified by an inversion. Therefore, in this case $\Omega(N, s) = \omega_I$ if $s \neq \text{sign}(N)$ and, otherwise, $\Omega(N, s) = 0$.

Weighted Preserving Rearrangements Acting on a Linear Node

Consider that N is a linear node. In the following, it is shown that if N is a linear node, then only a small constant number of different weight values for parsimonious preserving scenarios have to be considered. This is due to the facts, that every type of preserving rearrangement which acts on a linear node N is uniquely determined (Corollary 5.3) and that (under the set of considered rearrangements) a preserving

scenario with more than 3 rearrangements cannot be parsimonious (Proposition 5.6).

Let $\Omega_i(N, s)$, for $i \in [0:3]$, be auxiliary weight functions which give the minimum weight of a scenario S that contains i rearrangements that all act on node N such that all nodes in the subtree rooted at N in $T^{S\circ\lambda}(\pi, \Pi)$ have the sign s. In the following, some specific $\Omega_i(N, s)$, $i \in [0:3]$, are described and the corresponding sequences of length one, two, and three are illustrated in Figure 5.6.(a)–(d), Figure 5.7.(a), and Figure 5.7.(b), respectively. Subsequently, Proposition 5.6 shows that $\Omega(N, s)$ is the minimum of these four weight functions. Therefore, the Figure 5.6.(a)–(d) and Figure 5.7 illustrate the sequences that act on a linear node N and can be parsimonious.

Recall that $\Omega(N_i, \pm)$, $i \in [1: \deg(N)]$, denotes the minimum weight of a preserving scenario for the child node N_i of N and that $\text{sign}(N)$ is the sign of node N. Furthermore, let $s \in \{-, +\}$ be the desired sign. Note that any rearrangement ρ from $\mathcal{M}^{p}_{\text{4-type}|N}$ toggles the sign of the linear node N, since ρ acts on N. Therefore, for the application of an even (respectively odd) number of rearrangements it cannot hold that the sign of N in $T^{S\circ\lambda}(\pi, \Pi)$ is s if $\text{sign}(N) \neq s$ (respectively $\text{sign}(N) = s$). Hence, if $i \in \{1, 3\}$ and $\text{sign}(N) = s$, then $\Omega_i(N, s) = \infty$, and if $i \in \{0, 2\}$ and $\text{sign}(N) \neq s$, then $\Omega_i(N, -s) = \infty$.

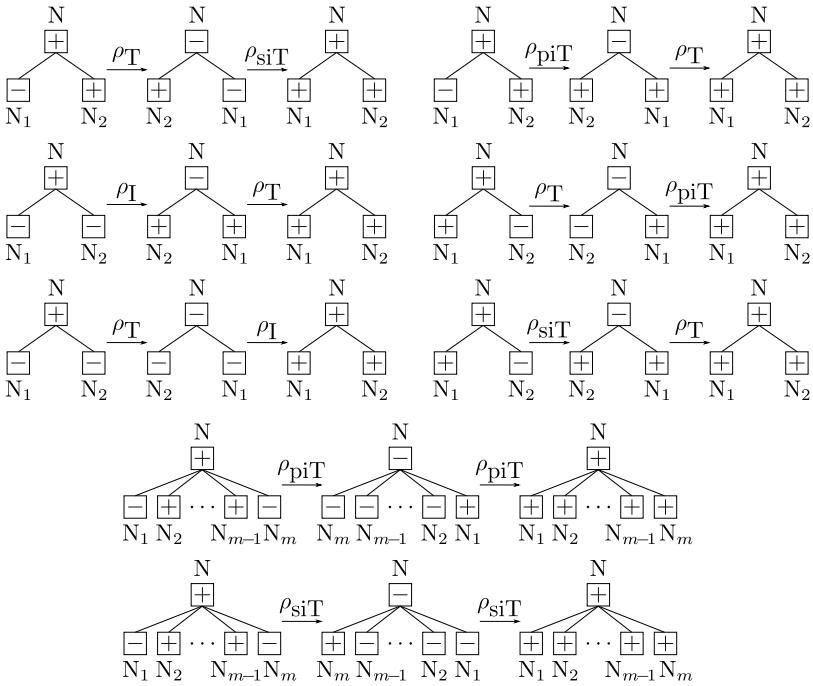
No rearrangement: If $\text{sign}(N) = s$, then one option is to apply no rearrangement that acts on N and apply rearrangements only to its $\deg(N)$ child nodes $N_1, \ldots, N_{\deg(N)}$ in order to adjust their signs to s. The weight of such a scenario is given by $\Omega_0(N, s) = \sum_{i=1}^{\deg(N)} \Omega(N_i, s)$. If $\text{sign}(N) \neq s$ then $\Omega_0(N, s) = \infty$.

One rearrangement: If $\text{sign}(N) \neq s$, then one possibility is to apply one rearrangement that acts on N satisfying that N and all its child nodes have the sign s. By Corollary 5.3 the weights have to be considered for all rearrangements that are illustrated in Figure 5.6. More precisely, the weight $\Omega_1(N, s)$ has to consider the weight of every type of preserving rearrangement plus the weight to realize the corresponding signs of the child nodes. Thus, $\Omega_1(N, s) = \min\{\mathcal{K}_{1,1}, \mathcal{K}_{1,2}, \mathcal{K}_{1,3}\}$, where

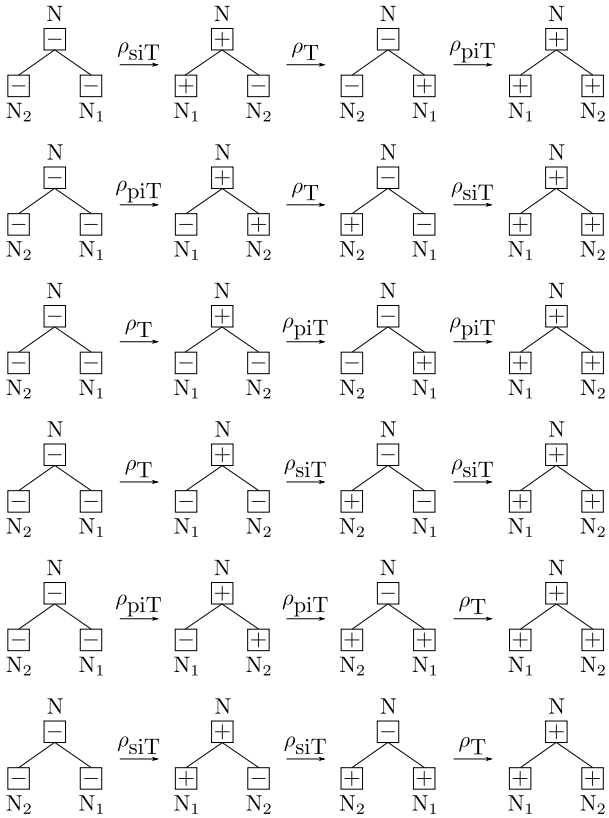
$$\begin{aligned}
\mathcal{K}_{1,1} &:= \omega_I + \sum_{i=1}^{\deg(N)} \Omega(N_i, -s), \\
\mathcal{K}_{1,2} &:= \omega_T + \Omega(N_1, s) + \Omega(N_2, s), \\
\mathcal{K}_{1,3} &:= \omega_{iT} + \min \left\{ \begin{matrix} \Omega(N_1, s) + \sum_{i=2}^{\deg(N)} \Omega(N_i, -s), \\ \Omega(N_{\deg(N)}, s) + \sum_{i=1}^{\deg(N)-1} \Omega(N_i, -s) \end{matrix} \right\}.
\end{aligned}$$

Note that $\mathcal{K}_{1,1}$, $\mathcal{K}_{1,2}$, and $\mathcal{K}_{1,3}$ is the weight of applying an inversion, transposition, and inverse transposition, respectively. If $\text{sign}(N) = s$ then $\Omega_1(N, s) = \infty$.

Two rearrangement: If $\text{sign}(N) = s$, then instead of applying rearrangements only to the child nodes of N (no rearrangement case), there is also the possibility to apply two rearrangements that act on N in order to change the signs of its child nodes simultaneously.



(a)



(b)

Figure 5.7: Examples of possibly parsimonious sequences of two (a) and three (b) rearrangements $\rho_{\mathrm{I}}, \rho_{\mathrm{T}}, \rho_{\mathrm{piT}}, \rho_{\mathrm{siT}} \in \mathcal{M}^{\mathrm{P}}_{\text{4-type}|\mathrm{N}}$ that act on a node N and transform N and its child nodes $\mathrm{N}_1, \ldots, \mathrm{N}_m$ into an order where all nodes have sign $+$. It is assumed that N is linear and its child nodes are linear or leaves (each node is represented by a square with the sign of the node).

Therefore, $\Omega_2(N,s) = \min\{\mathcal{K}_{2,1}, \mathcal{K}_{2,2}, \mathcal{K}_{2,3}\}$, where the weight of applying an inversion and a transposition is given by

$$\mathcal{K}_{2,1} := \begin{cases} \omega_I + \omega_T + \Omega(N_1, -s) + \Omega(N_2, -s) & \text{if } \deg(N) = 2, \\ \infty & \text{otherwise,} \end{cases}$$

the weight of applying two successive inverse transpositions of the same type (i. e., either piT or siT) is given by

$$\mathcal{K}_{2,2} := 2\omega_{iT} + \Omega(N_1, -s) + \sum_{i=2}^{\deg(N)-1} \Omega(N_i, s) + \Omega(N_{\deg(N)}, -s),$$

and the weight of applying a transposition and an inverse transposition is given by

$$\mathcal{K}_{2,3} := \begin{cases} \omega_T + \omega_{iT} + \min\left\{\begin{smallmatrix} \Omega(N_1,-s)+\Omega(N_2,s), \\ \Omega(N_1,s)+\Omega(N_2,-s) \end{smallmatrix}\right\} & \text{if } \deg(N) = 2, \\ \infty & \text{otherwise.} \end{cases}$$

If $\operatorname{sign}(N) \neq s$ then $\Omega_2(N,s) = \infty$.
Three rearrangement: If $\operatorname{sign}(N) \neq s$, only an application of one transposition and two inverse transpositions can be parsimonious. Therefore,

$$\Omega_3(N,s) = \begin{cases} \omega_T + 2\omega_{iT} + \Omega(N_1, -s) + \Omega(N_2, -s) & \begin{smallmatrix} \text{if } \deg(N) = 2, \\ \deg(N_1), \deg(N_2) > 2, \\ \omega_I > \omega_T + 2\omega_{iT}, \end{smallmatrix} \\ \infty & \text{otherwise.} \end{cases}$$

If $\operatorname{sign}(N) = s$ then $\Omega_3(N,s) = \infty$.

The following proposition states that it is sufficient to consider only $\Omega_0(N,s), \ldots, \Omega_3(N,s)$ for different weight values for preserving parsimonious scenarios.

Proposition 5.6. *Let $\Pi \subseteq s\mathcal{P}_n$, $\pi, \lambda \in s\mathcal{P}_n$ consistent with Π, and N be a linear node of $T^\lambda(\pi, \Pi)$. The set of possible rearrangements are all rearrangements of type I, T, iT that are preserving for Π with given positive weights ω_I, ω_T, and ω_{iT}, respectively. Let $s \in \{+,-\}$ be a given sign. The total weight for a parsimonious (preserving) scenario S that transforms λ into a permutation $S \circ \lambda$ such that all nodes within the subtree with root N in $T^{S\circ\lambda}(\pi, \Pi)$ are linear nodes with sign s is given by:*

$$\sum_{\rho \in S} \omega(\rho) = \begin{cases} \min\{\Omega_0(N,s), \Omega_2(N,s)\} & \textit{if } s = \textit{sign}(N), \\ \min\{\Omega_1(N,s), \Omega_3(N,s)\} & \textit{otherwise.} \end{cases}$$

Proof. Let $N_1, \ldots, N_{\deg(N)}$ be the child nodes of N. It follows from Corollary 5.3.(ii)–(iv) that only a few specific cases for rearrangements are possible since N is a linear node, e. g., an inverse transposition can only be a suffix inverse transposition or a prefix inverse transposition. For a clear representation, the following notations are used in the proof: The uniquely defined rearrangement ρ_Z of type $Z \in \{I, T, siT, piT\}$ that acts on N is denoted by its type Z. Consequently, a scenario $(\rho_{Z_1}, \ldots, \rho_{Z_t})$ is written as $(Z_1, \ldots, Z_t)$, where

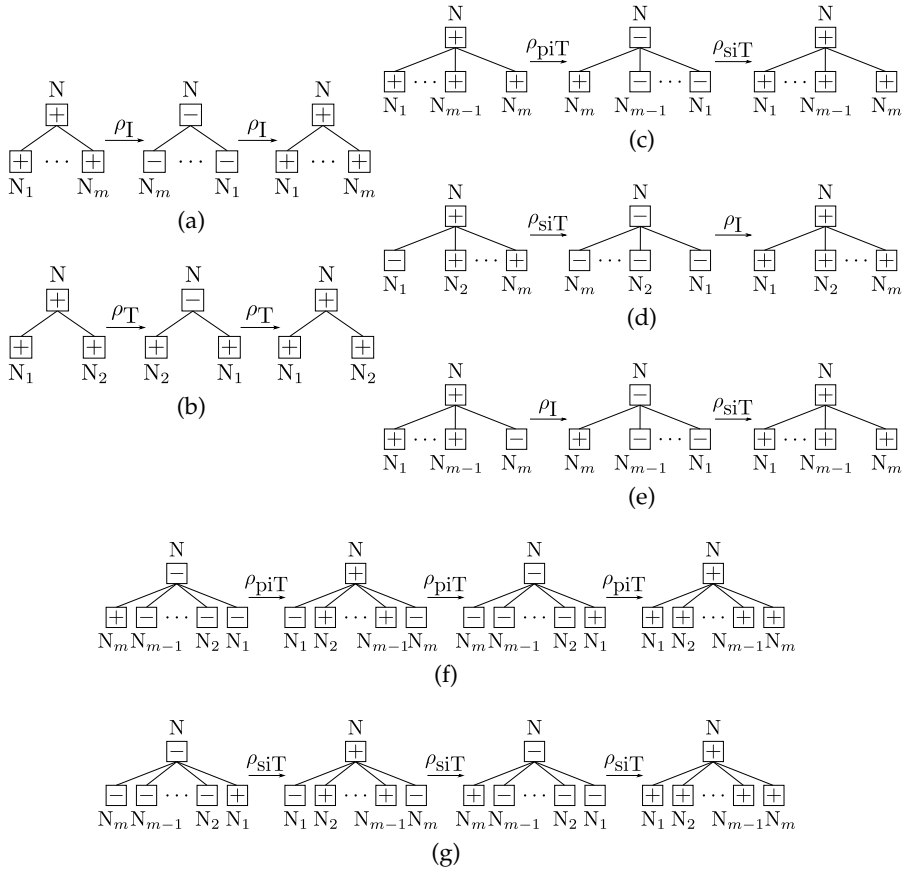


Figure 5.8: Examples of sequences of rearrangements $\rho_I, \rho_T, \rho_{siT}, \rho_{piT} \in \mathcal{M}^{P}_{4\text{-type}|N}$ that are not parsimonious: (a) (ρ_I, ρ_I); (b) (ρ_T, ρ_T); (c) (ρ_{piT}, ρ_{siT}); (d) (ρ_{siT}, ρ_I); (e) (ρ_I, ρ_{siT}); (f) $(\rho_{piT}, \rho_{piT}, \rho_{piT})$; and (g) $(\rho_{siT}, \rho_{siT}, \rho_{siT})$. (The notation is as in Figure 5.7.)

$Z_1, \ldots, Z_t \in \{I, T, siT, piT\}$. Two sequences of rearrangements S and S' for a permutation π are called *type-equivalent* if S and S' consists of exactly the same number of types of rearrangements (suffix and prefix inverse transpositions are both classified as inverse transpositions) and it holds that $S \circ \pi = S' \circ \pi$.

Proposition 5.5 shows that it can be assumed that all rearrangements that act on N form a strict subsequence in S. Two types of rearrangements occur in S: i) rearrangements that act on nodes within one of the subtrees where a child node of N is the root, and ii) rearrangements that act on N. Since the relative order of rearrangements that act on different nodes does not matter, it is assumed that all rearrangements of (i) are applied first. Hence, S can be expressed as $S_1\, S_2$, where S_1 is a sequence of rearrangements in (i) and S_2 is a sequence of rearrangements in (ii). Also, it can be assumed that after the application of S_1 to λ each child node of N is linear and therefore has a sign. Observe that – without loss of generality – the following proof considers only the case that $s = +$. (The proof for $s = -$ can be obtained by exchanging $+$ and $-$ in the proof shown below.) In the following paragraphs, it is determined which sequences of rearrangements can possibly form S_2 by proving the five cases where exactly either none, one, two, three, or at least four (denoted by Case 1 to 5, respectively) rearrangements are contained in S_2. Before performing the case anal-

ysis, the following two observations are made: A) Sequence S_2 cannot contain one of the following subsequences (I, I), (T, T), (piT, siT), and (siT, piT) since in each case the application of the second rearrangement removes the effect of the first one, see figures 5.8.(a) – (c). B) If a sequence S of rearrangements contains a strict subsequence that is not parsimonious then S is not parsimonious.

1) Consider S_2 is empty, i. e., $|S_2| = 0$. This case is only possible when N has sign s and each child node of N has sign s after the application of $S_1 = S$, i. e., in $T^{S \circ \lambda}(\pi, \Pi)$. The total weight $\Omega_0(N, +)$ for a parsimonious sequence that transforms all nodes within the subtree with root N to s were no rearrangement acts on N is $\sum_{i=1}^{\deg(N)} \Omega(N_i, +)$.

2) Consider $|S_2| = 1$. As a direct consequence of Corollary 5.3, it can be seen that figures 5.6.(a) – (d) illustrate the only possible cases. Let S denote a parsimonious sequence that transforms all nodes within the subtree rooted at N into nodes with sign $+$ and that contains exactly one rearrangement acts on N. It follows that the total weight $\Omega_1(N, +)$ for such a parsimonious sequence S is given by $\min\{\mathcal{K}_{1,1}, \mathcal{K}_{1,2}, \mathcal{K}_{1,3}\}$.

3) Consider $|S_2| = 2$. Sequences of applying one inversion and one inverse transposition (i. e., (siT, I), (I, piT), (I, siT), and (piT, I)) cannot be parsimonious by the following argumentation. Assume that S_2 is such a sequence and it is a strict subsequence of a parsimonious sequence S, then the weight of S_2 must be less than the weight of an inversion that acts on the one child node with the sign $-$ and changes its sign to $+$ (see Figure 5.8.(d) and Figure 5.8.(e)). This would imply $\omega_I + \omega_{iT} \leqslant \omega_I$, which is not possible since $\omega_{iT} > 0$. Considering this and Observation (A) it follows that $S_2 \in \{(I, T), (T, I), (piT, piT), (siT, siT), (T, piT), (siT, T), (T, siT), (piT, T)\}$. Hence, the total weight $\Omega_2(N, +)$ for a parsimonious sequence S which changes the signs of all nodes within the subtree with root N to $+$ and contains exactly two rearrangements act on N is $\min\{\mathcal{K}_{2,1}, \mathcal{K}_{2,2}, \mathcal{K}_{2,3}\}$.

4) Consider $|S_2| = 3$. Consider first that sequence S_2 contains a transposition. Corollary 5.3 implies that N has only two child nodes. It is not hard to see that each combination of signs and order of the two child nodes of N can be sorted with one inversion and one transposition. Therefore, S_2 cannot contain an inversion and a transposition (in addition to a third rearrangement).

   It is also not hard to see that each combination of signs and order of the two child nodes can be sorted with one suffix inverse transposition and one transposition or with one prefix transposition and one transposition if $\deg(N_1) = 2$ or $\deg(N_2) = 2$. A sequence with one transposition and two inverse transpositions can be replaced by a sequence that contains only one inversion

if $\omega_I < \omega_T + 2\omega_{iT}$. For this reason, a sequence with one transposition and two inverse transpositions might be parsimonious if $\omega_I > \omega_T + 2\omega_{iT}$, $\deg(N_1) > 2$, and $\deg(N_2) > 2$.

The remaining possible sequences that contain at least one transposition are (T, siT, T) and (T, piT, T). Note that (T, siT, T) (respectively (T, piT, T)) is type-equivalent to (T, T, piT) (respectively (T, T, siT)). By Observation (B) it holds that (T, siT, T) and (T, piT, T) cannot be parsimonious, since they contain the non-parsimonious strict subsequence (T, T). By Observation (B), S_2 cannot end with one of the following subsequences (siT, I), (I, piT), (I, siT), and (piT, I).

The sequences (piT, piT, piT) and (siT, siT, siT) (illustrated in Figure 5.8.(f) and Figure 5.8.(g), respectively) cannot be parsimonious, since they can be replaced by the sequences (siT) and (piT) that have a smaller weight as $\omega_{piT} = \omega_{siT} = \omega_{iT}$.

It remains to consider the sequences (I, piT, piT), (I, siT, siT), (piT, piT, I), and (siT, siT, I). It is not hard to see that these sequences have the same effect as (siT, siT) and (piT, piT).

Consequently, the weight $\Omega_3(N, +)$ for a parsimonious sequence S which transforms the signs of all nodes within the subtree with root N to $+$ and contains exactly three rearrangements acts on N is either $\omega_T + 2\omega_{iT} + \Omega(N_1, -) + \Omega(N_2, -)$ if $\deg(N) = 2$, $\deg(N_1) > 2$, $\deg(N_2) > 2$, and $\omega_I > \omega_T + 2\omega_{iT}$ or ∞, otherwise.

5) Consider $|S_2| \geqslant 4$. By Observation (B) S_2 must end with a parsimonious sequence. By cases 1–4, it holds that S_2 can only end with (piT, piT, T), (siT, siT, T), (T, piT, piT), (T, siT, siT), (siT, T, piT), or (piT, T, siT). Note that (piT, piT, T) (respectively (siT, siT, T) and (siT, T, piT)) is type-equivalent to (T, piT, piT) (respectively (T, siT, siT) and (piT, T, siT)).

Now consider all the scenarios of four rearrangements that end with (piT, piT, T), (siT, siT, T), or (siT, T, piT), i. e., (Z, piT, piT, T), (Z, siT, siT, T), and (Z, siT, T, piT) with $Z \in \{T, I, piT, siT\}$.

All sequences that contain three inverse transpositions cannot be parsimonious since either two inverse transpositions of the same type occur subsequently or suffix inverse transposition and a prefix inverse transposition occur subsequently after replacing (siT, siT) (respectively (piT, piT)) with its type-equivalent sequence (piT, piT) (respectively (siT, siT)).

It is not hard to see that (T, piT, piT, T) (respectively (T, siT, siT, T) and (T, siT, T, piT)) is type-equivalent to (siT, siT, T, T) (respectively (piT, piT, T, T) and (piT, T, T, piT)). Therefore, sequences of S_2 with four rearrangements that start with a transposition cannot be parsimonious.

Furthermore, it is easy to verify that (I, piT, piT, T) (respectively (I, siT, siT, T) and (I, siT, T, piT)) is type-equivalent to

(piT, piT, I, T) (respectively (siT, siT, I, T)). Since the subsequences (piT, I, T) and (siT, I, T) are non-parsimonious.

Hence, no scenario with four rearrangements is parsimonious. Observation (B) implies that no scenario with more than four rearrangements can be parsimonious.

Hence, $\sum_{\rho \in S} \omega(\rho) = \min\{\Omega_0(N,s), \Omega_1(N,s), \Omega_2(N,s), \Omega_3(N,s)\}$. The proposition follows by the fact that any preserving rearrangement that acts on N toggles the sign of N (which is implied by Corollary 5.2). □

Weighted Preserving Rearrangements acting on a Prime Node

Consider that N is a prime node. If N is a prime node, then a preserving rearrangement that acts on N can arbitrarily change the order of the child nodes of N. Therefore, the set of possibly parsimonious scenarios cannot be reduced as it is done in the linear case.

Recall that $\pi_{|N}$ denotes the quotient permutation of N with respect to π and that $\Omega(N_i, s)$ is the minimum weight of a sequence S_i^s of preserving rearrangements that transforms the considered sSIT $T^{\lambda}(\pi, \Pi)$ into the sSIT $T^{S_i^s \circ \lambda}(\pi, \Pi)$ in which all nodes of the subtree rooted at the child node $N_i \in \{N_1, .., N_{\deg(N)}\}$ of N are linear and have the sign $s \in \{+, -\}$. Hence, the weight to rearrange the subtree rooted at a child node N_i to a certain sign s has the weight $\Omega(N_i, s)$. Recall that the signed quotient permutation $\hat{\pi}_{|N}$ has sign s assigned to its i-th element if the node N_i has sign s. Apparently, making a decision for the sign for every child node N_i determines a signed quotient permutation $\hat{\pi}_{|N}$ and the corresponding weight for such a decision is determined by $\sum_{i=1}^{\deg(N)} \Omega(N_i, \text{sign}(\hat{\pi}_{|N}(i)))$. Let $\hat{\Pi}_{|N}$ denote the set of all signed quotient permutations that can be obtained by every possible sign combination. In order to compute the value $\Omega(N, +)$ (respectively $\Omega(N, -)$) a signed quotient permutation $\hat{\pi}_{|N} \in \hat{\Pi}_{|N}$ has to be found such that a weight minimum scenario S for $\hat{\pi}_{|N}$ and ι (respectively $\bar{\iota} := (-\deg(N) \ \ldots \ -1)$) minimizes the sum of its weight $\omega(S)$ plus the weight to determine the considered signed quotient permutation. Formally, the desired weights are given by:

$$\Omega(N,+) = \min_{\hat{\pi}_{|N} \in \hat{\Pi}_{|N}} \min_{S \in \mathfrak{S}_{\mathcal{M}_{4\text{-type}}}(\hat{\pi}_{|N}, \iota)} \omega(S) + \sum_{i=1}^{\deg(N)} \Omega(N_i, \text{sign}(\hat{\pi}_{|N}(i))),$$

$$\Omega(N,-) = \min_{\hat{\pi}_{|N} \in \hat{\Pi}_{|N}} \min_{S \in \mathfrak{S}_{\mathcal{M}_{4\text{-type}}}(\hat{\pi}_{|N}, \bar{\iota})} \omega(S) + \sum_{i=1}^{\deg(N)} \Omega(N_i, \text{sign}(\hat{\pi}_{|N}(i))).$$

Hence, to determine $\Omega(N, s)$, $s \in \{+, -\}$, every sign combination has to be considered. Since the unconstrained problem to find a weight minimum scenario for a given signed quotient permutation and ι (or $\bar{\iota}$) under $\mathcal{M}_{4\text{-type}}$ is already a NP-hard optimization problem (Bulteau et al., 2012), it is not hard to see that the computation of $\Omega(N, s)$, $s \in \{+, -\}$ is NP-hard as well.

Algorithm CREx2, which is presented in the following section, handles the computation of $\Omega(N, s)$ by Algorithm 2 (see Section 5.2) or

GeRe-ILP (see Section 5.3). While latter algorithm is able to provide exact solution within an exponential runtime, the former algorithm provides approximated solutions efficiently.

5.4.4 *Dynamic Programming Algorithm* CREx2

In this section, the algorithm CREx2 is presented. CREx2 solves the weighted preserving sorting problem for a given signed linear permutation $\pi \in s\mathcal{P}_n$ under $\mathcal{M}^{p}_{4\text{-type}}$. Thereby, it considers weights for each type of rearrangement.

CREx2 is a dynamic programming algorithm that computes the weights $\Omega(N, s)$ with three different routines depending on whether a node N is a leaf, linear, or prime. For leaf nodes and linear nodes N the weights $\Omega(N, s)$ are computed directly as stated in Proposition 5.6. The weights for prime nodes are computed by two algorithms that make a different trade-off between exactness and runtime: Algorithm 2 runs efficiently but gives only approximate results. An adjusted version of algorithm GeRe-ILP (see Section 5.3) makes the opposite trade-off by finding the optimum weights $\Omega(N, s)$ but paying a penalty in runtime and memory requirements.

Algorithm 3 shows the pseudocode of CREx2. In the following, it is described how CREx2 determines $\Omega(N, s)$ and implicitly also a corresponding scenario.

Algorithm 3 : Pseudocode of CREx2 algorithm.

Data : Node N of $T^{\lambda}(\pi, \{\pi, \lambda\})$
Result : $\Omega(N, +), \Omega(N, -)$

1 $\Omega(N, +) \leftarrow \infty$; $\Omega(N, -) \leftarrow \infty$;
2 **for** $N_i \in \{N_1, \ldots, N_{\deg(N)}\}$ **do** // compute child weights
3 | $\Omega(N_i, +), \Omega(N_i, -) \leftarrow$ CREx2(N_i);
4 **if** N *is a leaf node* **then** // case leaf node
5 | $\Omega(N, \text{sign}(N)) \leftarrow 0$;
6 | $\Omega(N, \text{-sign}(N)) \leftarrow \omega_I$;
7 | **return** $(\Omega(N, +), \Omega(N, -))$;
8 **if** N *is linear* **then** // case linear node
9 | $\Omega(N, \text{sign}(N)) \leftarrow \min\{\Omega_0(N, \text{sign}(N)), \Omega_2(N, \text{sign}(N))\}$;
10 | $\Omega(N, \text{-sign}(N)) \leftarrow \min\{\Omega_1(N, \text{-sign}(N)), \Omega_3(N, \text{-sign}(N))\}$;
11 **else** // case prime node
12 | $\Omega(N, \pm) \leftarrow$ PrimeNodeAlgorithm$(\pi_{|N}, \pm)$;
13 **return** $(\Omega(N, +), \Omega(N, -))$

CREx2 is called for the root node of the sSIT $T^{\lambda}(\pi, \{\pi, \lambda\})$. Recursive function calls (Line 3) for each child node N_i, $i \in [1 \colon \deg(N)]$, of a node N pre-compute the necessary weights $\Omega(N_i, \pm)$. The base case of the recursion are the leaf nodes (lines 4 – 7). Note that in the considered rearrangement model the sign of a leaf node N can only be modified by an inversion (Corollary 5.3). Hence, for a leaf N $\Omega(N, s) = \omega_I$ if $s \neq \text{sign}(N)$ and $\Omega(N, s) = 0$ otherwise (lines 5 – 7).

For an inner node N (lines 8 – 12) the value of $\Omega(N, s)$ can be computed from a parsimonious preserving scenario S that transforms N

to a linear node with sign s. In addition, the weight of S plus the weight to change the signs of the child nodes of N to s (using preserving scenarios) must be minimum. The cases of a linear node N (lines 8 – 10) and a prime node N (lines 11 – 12) are handled differently by `CREx2`. This is described in more detail in the following.

Consider first the case that N is a linear node. By Proposition 5.6 the minimum weight $\Omega(N, s)$ for a linear node N can be computed as: $\Omega(N, s) = \min\{\Omega_0(N, s), \Omega_2(N, s)\}$ if $s = \text{sign}(N)$ and $\Omega(N, s) = \min\{\Omega_1(N, s), \Omega_3(N, s)\}$ otherwise (lines 9 – 10).

Now, consider the case that N is a prime node. `CREx2` is able to handle this case in lines 11 to 12 by two different algorithms which are henceforth called *prime node algorithms*: The first prime node algorithm that can be used to compute an approximated value of $\Omega(N, \pm)$ (and the corresponding rearrangement scenario) is Algorithm 2. In this case, the signed quotient permutation is chosen such that the sum of the weights to sort the corresponding child nodes of the prime node is minimum. Note that this choice is locally the optimal decision. However, in some cases this decision might be unfavorable.

The second prime node algorithm is an adjusted version of the integer linear program `GeRe-ILP` (see Section 5.3) which can compute $\Omega(N, \pm)$ optimally. In order to be used by `CREx2`, the ILP formulation of `GeRe-ILP` has been adjusted as follows:

1) `GeRe-ILP` computes $\Omega(N, s)$ and finds a sufficient signed quotient permutation $\hat{\pi}_{|N}$ simultaneously to a weight minimum scenario for $\hat{\pi}_{|N}$ and ι if $s = +$ (or $\hat{\pi}_{|N}$ and $\bar{\iota}$ if $s = -$). Therefore, `GeRe-ILP` uses as an additional input the $\Omega(N_i, s)$ of the child nodes $N_1, \ldots, N_{\deg(N)}$ of N which have been pre-computed by recursive function calls. This is done by removing the Constraint (`ILP 2`) in Section 5.3.1 for all O_{i0} with $i \in [1:n]$. The minimum weight, which includes the weight of the scenario acting on N plus the weight to realize the corresponding signs of the child nodes, is then computed by the following objective function that replaces Constraint (`ILP 36`).

$$\begin{aligned} \min\{ & \sum_{k=1}^{t} (I_k \omega_I + T_k \omega_T + iT_k \omega_{iT} + TDRL_k \omega_{TDRL}) \\ & + \sum_{i=1}^{\deg(N)} ((1 - O_{i0})\Omega(N_i, +) + O_{i0}\Omega(N_i, -))\} \end{aligned} \tag{ILP 37}$$

2) In order to handle the exponential runtime behavior of `GeRe-ILP`, a time limit $\mathcal{L}$ can be set by the user of `CREx2`. If the runtime of `GeRe-ILP` reaches $\mathcal{L}$ either best solution that has been found so far is returned or, if no solution has been found, Algorithm 2 is applied to give an approximate fallback solution. Clearly, in both cases the solution might not be exact. Note, if parameter $\mathcal{L}$ is used the total runtime of `CREx2` can exceed $\mathcal{L}$ since $\mathcal{L}$ is a runtime bound only for handling a single prime node N.

For a runtime analysis of CREx2 let $\lambda, \pi \in s\mathcal{P}_n$ and $T^{\lambda}(\pi, \{\pi, \lambda\})$ be a sSIT of $\{\pi, \lambda\}$, λ, and π. By recursion, Algorithm 3 is called once for each of the at most $\mathcal{O}(n)$ nodes of $T^{\lambda}(\pi, \{\pi, \lambda\})$. For a leaf node the weights $\Omega(N, \pm)$ can be computed in constant time. For linear nodes at most a small constant number of weight values are evaluated using only the given weights and the weights of the subtrees rooted at the child nodes. Therefore, in the case that the sSIT is linear, only a constant amount of time is necessary per node of the sSIT. Hence, the CREx2 algorithm has a runtime of $\mathcal{O}(n)$ for solving the weighted preserving sorting problem if the given sSIT is linear. If the sSIT is prime, the runtime of CREx2 is dependent on the prime node algorithm that is used in Line 12. If Algorithm 2 is used to obtain an approximated solution of the weights $\Omega(N, \pm)$, then a scenario for a prime node is computed in time $\mathcal{O}(n \log n)$. Therefore, the total runtime of CREx2 is in $\mathcal{O}(n^2 \log n)$ in this case. If otherwise GeRe-ILP is used to compute the weights $\Omega(N, \pm)$ exactly, then the runtime of CREx2 is dominated by the introduced variant of GeRe-ILP which is exponential in the worst case.

Experimental results for CREx2 when applied to artificial and mitochondrial gene order data sets are presented in Section 5.5. The results prove that CREx2 is suitable to reconstruct scenarios of genome rearrangements between gene orders with a high accuracy.

Algorithm CREx2 is implemented in C++ using Gurobi Optimizer 8.1 (Gurobi Optimization, 2018) and is freely available on http://pacosy.informatik.uni-leipzig.de/crex2.

5.5 EVALUATION

Naturally, the question arises how exact the constructed scenarios of CREx2 are. Addressing this question, this section analyses the performance of CREx2 empirically on simulated as well as on real mitochondrial gene order data sets. Recall that two variants of CREx2 have been proposed in Section 5.4.4. The first variant (henceforth called CREx2-ILP) computes exact solutions but has an exponential runtime in the worst case. The second variant (henceforth called CREx2-APP) computes approximated solutions efficiently. In this section, the accuracy of the rearrangement scenarios that are computed with CREx2-APP are analyzed empirically in a simulation for many different models of genome rearrangements. Thereby, it is shown that the reconstruction accuracy of the rearrangement scenarios computed with CREx2-APP is dependent on properties of the strong interval tree of the considered problem instance. If the strong interval tree is linear, then CREx2-APP computes scenarios of high reconstruction accuracy. This accuracy is reduced as the number of prime nodes in the strong interval tree increases. In addition, biologically useful simulation parameters are identified for which CREx2-APP gives scenarios of high accuracy. Experiments on simulated gene orders are performed in order to determine rearrangement weights that maximize the reconstruction accuracy of the CREx2-APP results with respect to metazoan

mitochondrial gene orders. A large-scale comparison of the results of CREx (Bernt et al., 2007) and CREx2-APP on the complete set of all metazoan mitochondrial gene orders from the NCBI RefSeq release 89 is performed. In addition, the scenarios obtained by CREx2-APP and CREx2-ILP are compared with each other on the set of all currently available *Chordata*, *Ecdysozoa*, and *Lophotrochozoa* mitochondrial gene orders.

5.5.1 *CREx2 on Simulated Gene Order Data Sets*

In this section, the accuracy of the rearrangement scenarios returned by CREx2 is analyzed in a large study on simulated gene order data. Therefore, each data set is obtained by applying $t \in [1:10]$ randomly chosen rearrangements from $\mathcal{M}_{4\text{-type}}$ to the identity permutation of size $n \in \{37, 100\}$. In order to simulate various models of genome rearrangements, the type of a chosen rearrangement is determined with respect to a given probability vector $(p_I, p_T, p_{iT}, p_{TDRL})$, where p_X denotes the probability of a rearrangement of type $X \in \{I, T, iT, TDRL\}$. For example, the probability vector $(0, 0, 0, 1)$ always results into TDRLs and for the vector $(0.3, 0.3, 0.3, 0.1)$ an inversion, transposition, and inverse transposition is chosen with probability 0.3 while 0.1 is the probability to choose a TDRL. Subsequently, inversions (transpositions and inverse transpositions) are chosen uniformly at random from the set of all inversions (respectively transpositions and inverse transpositions), i. e., from $\mathcal{M}_I$ (respectively $\mathcal{M}_T$ and $\mathcal{M}_{iT}$). To obtain a random TDRL $\rho_{TDRL}(L, R)$ a two step procedure is used: First, a consecutive substring (of the permutation the TDRL is supposed to be applied to) is chosen at random. This substring represents the duplicated interval of a TDRL rearrangement. Second, for every element of the interval it is chosen uniformly at random if it is kept in the left copy L or the right copy R. Clearly, elements that are outside of the interval are not affected by the TDRL. By this procedure a scenario of rearrangements S for a considered probability vector is obtained as well as a permutation $\pi := S \circ \iota$. Permutations π and ι are then used as input for CREx2 with the aim to reconstruct scenario S. Let T denote the scenario obtained with CREx2. In order to compare S and T the measures *recall* and *precision* are used which are defined as $|S \cap T|/|T|$ and $|S \cap T|/|S|$, respectively. Note that $|S_1|$ denotes the length of a scenario S_1 and the intersection $S_1 \cap S_2$ of two scenarios S_1 and S_2 is the set of rearrangements that occur in S_1 and S_2. It is not hard to see that recall and precision are well defined if $|S|, |T| > 0$ which is always satisfied if π is unequal to the identity permutation. For that reason, a permutation π (and the corresponding simulated scenario) is only included to a data set if $\pi \neq \iota$. Less formally, recall measures the completeness of the reconstructed rearrangement scenario and precision measures its exactness.

For the first experiment a large data set has been generated for permutations of size $n = 100$. For each $t \in [1:10]$, 1000 permutations have been generated with respect to each of the probability

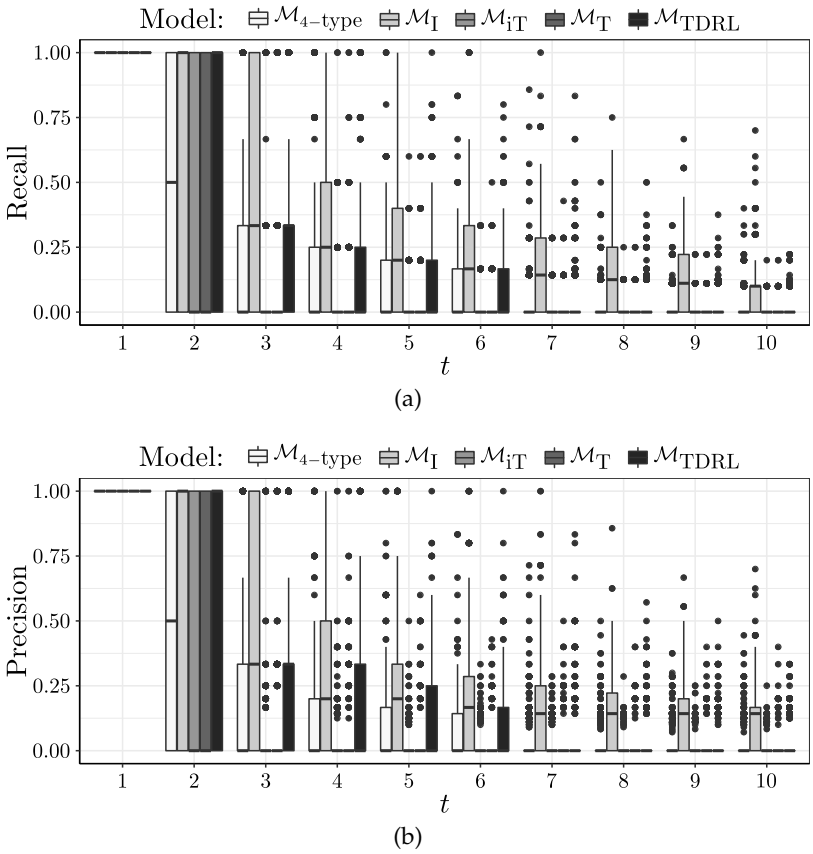


Figure 5.9: Recall (a) and precision (b) of the results of `CREx2-APP` applied to the data sets generated by the application of $t \in [1:10]$ rearrangements to ι of size $n = 100$. Gray scales of the box plots indicate the rearrangement models that have been considered for simulation: all 4-type rearrangements with the probability vector $(0.25, 0.25, 0.25, 0.25)$ ($\mathcal{M}_{4\text{-type}}$); inversions only ($\mathcal{M}_{\mathrm{I}}$); inverse transpositions only ($\mathcal{M}_{\mathrm{iT}}$); transpositions only ($\mathcal{M}_{\mathrm{T}}$); and TDRLs only ($\mathcal{M}_{\mathrm{TDRL}}$). Each box plot has been generated based on the results of 1000 problem instances.

vectors $(1, 0, 0, 0)$ (i. e., the model $\mathcal{M}_{\mathrm{I}}$), $(0, 1, 0, 0)$ (i. e., the model $\mathcal{M}_{\mathrm{T}}$), $(0, 0, 1, 0)$ (i. e., the model $\mathcal{M}_{\mathrm{iT}}$), $(0, 0, 0, 1)$ (i. e., the model $\mathcal{M}_{\mathrm{TDRL}}$) and $(0.25, 0.25, 0.25, 0.25)$ (i. e., the model $\mathcal{M}_{4\text{-type}}$). Hence, 50 data sets each containing 1000 problem instances were generated.

Recall that `CREx2-ILP` has an exponential runtime in the worst case. In order to perform the following experiments in a reasonable amount of time, this section considers `CREx2-APP` only. In addition, all types of rearrangements that are considered by `CREx2-APP` are weighted equally in the first experiment.

Figure 5.9 shows box plots of recall and precision for all data sets. In addition, the respective average values of recall and precision are illustrated in Figure 5.10. For $t = 1$ the figures show that the simulated scenario is always reconstructed by `CREx2-APP`. This is not a surprising result as the application of a single inversion (transposition, inverse transposition) always results in a linear strong interval tree (SIT), i. e., a SIT without prime nodes, and `CREx2-APP` is able so solve such SITs exactly. Moreover, the application of a TDRL may result into a prime SIT which can be resolved by a single TDRL that

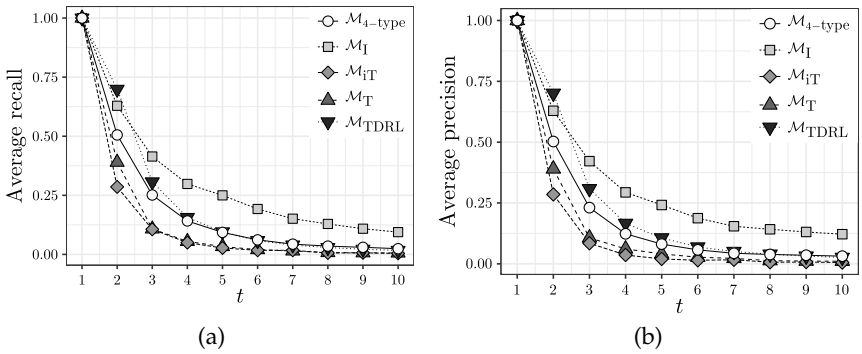


Figure 5.10: Average values of recall (a) and precision (b) of the box plots illustrated in Figure 5.9 for $t \in [1:10]$ and the rearrangement models $\mathcal{M}_{4\text{-type}}$, $\mathcal{M}_{\mathrm{I}}$, $\mathcal{M}_{\mathrm{iT}}$, $\mathcal{M}_{\mathrm{T}}$, and $\mathcal{M}_{\mathrm{TDRL}}$. Notation is as in Figure 5.9.

can be reconstructed exactly with `CREx2-APP`. In addition, the results for $t = 1$ show that the different rearrangement cases of `CREx2-APP` are implemented correctly. For increasing t the values of recall and precision drop significantly. Scenarios simulated under the model $\mathcal{M}_{\mathrm{I}}$ are reconstructed with higher accuracy. For $t = 2$ the majority of scenarios is reconstructed correctly (i. e., recall and precision are both 1) for the data sets that has been simulated with respect to $\mathcal{M}_{\mathrm{I}}$ or $\mathcal{M}_{\mathrm{TDRL}}$. More precisely, 629 and 670 scenarios have been reconstructed correctly in the former and latter case, respectively. For the data set generated using all four types of genome rearrangements with equally probability, 489 scenarios were obtained correctly. One third (390) of the simulated scenarios were obtained correctly for the data sets that have been simulated under the model $\mathcal{M}_{\mathrm{T}}$. Less than one third (285) scenarios were obtained correctly for the data sets that have been generated with respect to $\mathcal{M}_{\mathrm{iT}}$. At least one rearrangement scenario has been reconstructed correctly for all $t < 6$ (respectively $t \leqslant 6$ and $t < 8$) for the data sets that have been constructed with respect to each rearrangement model (respectively the models that consider TDRLs and $\mathcal{M}_{\mathrm{I}}$). It is worth mentioning that for $t = 10$ at least one rearrangement has correctly been reconstructed in all data sets. More precisely, in 598 (respectively 56, 41, 146, and 204) scenarios for data sets with $t = 10$ at least one rearrangement has correctly been reconstructed for the data set constructed with respect to $\mathcal{M}_{\mathrm{I}}$ (respectively $\mathcal{M}_{\mathrm{T}}$, $\mathcal{M}_{\mathrm{iT}}$, $\mathcal{M}_{\mathrm{TDRL}}$, and $\mathcal{M}_{4\text{-type}}$).

The main reason for the low accuracy of the reconstructed `CREx2-APP` scenarios for larger t is the presence of prime nodes in the strong interval trees of the considered permutations. One reason is that the number of SITs containing at least one prime node increases significantly for increasing t. Table 5.1 illustrates the amount of SITs that contain at least one prime node for each data set. It can be seen that with the exception of $\mathcal{M}_{\mathrm{I}}$ more than 99% of the problem instances result into a prime SIT for all considered rearrangement models with $t \geqslant 4$. Even for $t = 2$, more than half of the problem instances have a prime SIT. For $t = 1$ prime SITs are observed only for data sets that

Table 5.1: Number of simulated problem instances that have at least one prime node in the corresponding strong interval tree for all combinations of $t \in [1:10]$ and the rearrangement models $\mathcal{M}_{4\text{-type}}$, $\mathcal{M}_{I}$, $\mathcal{M}_{iT}$, $\mathcal{M}_{T}$, and $\mathcal{M}_{TDRL}$.

t	$\mathcal{M}_{4\text{-type}}$	$\mathcal{M}_{I}$	$\mathcal{M}_{iT}$	$\mathcal{M}_{T}$	$\mathcal{M}_{TDRL}$
1	225	0	0	0	921
2	633	0	453	601	988
3	873	282	873	919	998
4	970	559	983	988	999
5	995	769	994	998	1000
6	999	886	1000	1000	1000
7	1000	951	1000	1000	1000
8	1000	993	1000	1000	1000
9	1000	997	1000	1000	1000
10	1000	998	1000	1000	1000

are generated with respect to models that consider TDRL rearrangements. In particular, 225 (respectively 921) SITs that contain at least one prime node were obtained. Note that the former value reflects the fact that a TDRL is chosen with probability 0.25 and that all prime nodes for $t = 1$ are caused by TDRLs. For the 79 problem instances were no prime node occurs in the data set generated with respect to $\mathcal{M}_{TDRL}$ it applies that the randomly chosen TDRL is actually a transposition rearrangement. Another reason for the assumption that the presence of prime nodes indicates low values of recall and precision is the observation that the values of recall and precision are very large for linear SITs as illustrated in Figure 5.11. The figure presents recall and precision for all data sets (and models) in relation to the number of prime nodes that occur in the corresponding SITs. It shows that if a SIT does not contain any prime node, then the rearrangement scenarios that are computed with `CREx2-APP` exhibit very large values of recall and precision. More precisely, 5898 (75%) of all 8155 problem instances that have a linear SIT were reconstructed correctly, i. e., recall and precision are both 1. Computed scenarios in which more than the half of its rearrangements are reconstructed correctly (i. e., recall and precision are both larger than 0.5) are obtained for 6332 (78%) problem instances. Moreover, at least one rearrangement of a reconstructed scenario is computed correctly (i. e., recall and precision are both larger than 0) for 7170 (88%) problem instances. Observe that the values of recall and precision in the data sets generated with respect to $\mathcal{M}_{TDRL}$ for prime node free instances correspond to the cases were a TDRL has the same effect as a transposition rearrangement. Figure 5.11 also shows that the values of recall and precision are much worse for problem instances which have a prime SIT. In this case, only 2401 (5, 7%) scenarios of the 41845 problem instances with a prime SIT were reconstructed correctly. Moreover, for problem instances with a

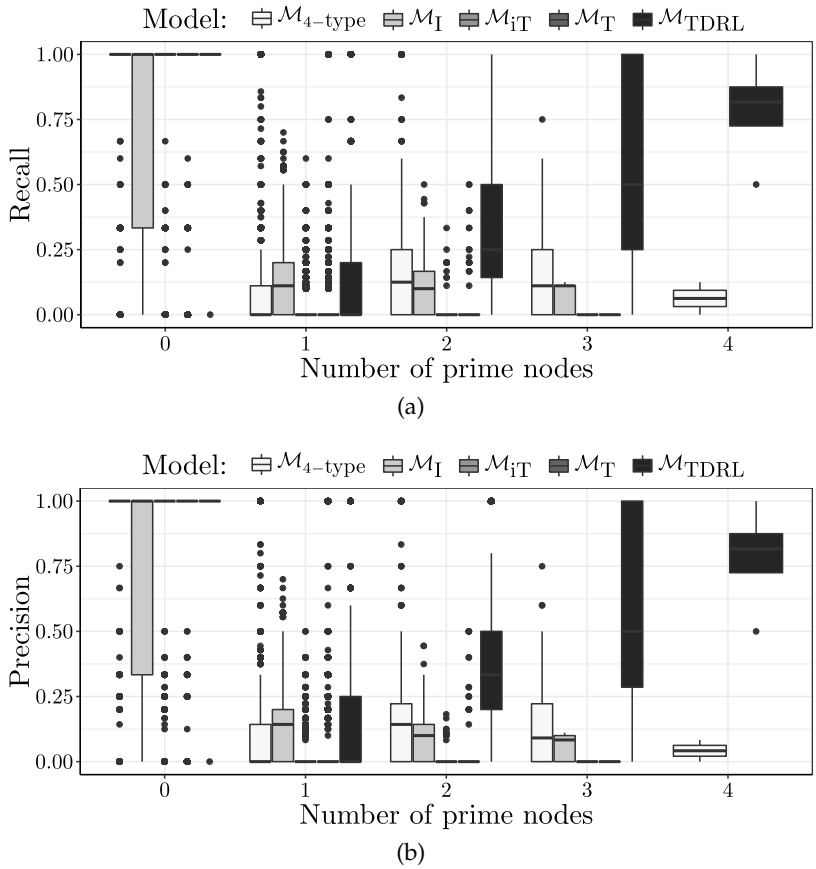


Figure 5.11: Recall (a) and precision (b) of the results of `CREx2-APP` in relation to the number of prime nodes in the corresponding strong interval tree for the data sets generated with respect to the rearrangement models $\mathcal{M}_{4\text{-type}}$, $\mathcal{M}_{\text{I}}$, $\mathcal{M}_{\text{iT}}$, $\mathcal{M}_{\text{T}}$, and $\mathcal{M}_{\text{TDRL}}$. Notation is as in Figure 5.9.

prime SIT, half of the rearrangements were reconstructed correctly in 2855 (6,8%) scenarios and at least one rearrangement of a computed scenario was reconstructed correctly for 34841 (83,3%) problem instances. It seems counter-intuitive that the values for recall and precision increase for an increasing number of prime nodes for data sets that have been generated with respect to $\mathcal{M}_{\text{TDRL}}$. `CREx2-APP` handles prime nodes with Algorithm 2 which mostly uses rearrangements of type TDRL. However, the results of recall and precision for SITs with an increasing number of prime nodes should be interpreted with caution as the number of problem instances with a larger number of prime nodes decreases significantly. For example, only 6 of the 41845 problem instances have a SIT with 4 prime nodes and only one of these 6 scenarios was reconstructed correctly. Hence, the rightmost box plot in both subfigures of Figure 5.11 may be deceptive.

The first results of `CREx2-APP` on simulated gene order data do clearly not shape up well. However, it should be noted that `CREx2-APP` is able to reconstruct at least parts of the simulated rearrangement scenarios for many data set across the different models that have been considered for the simulations. It is also worth mentioning that `CREx2-APP` computes rearrangement scenarios of high accuracy if problem instances have strong interval trees without prime nodes.

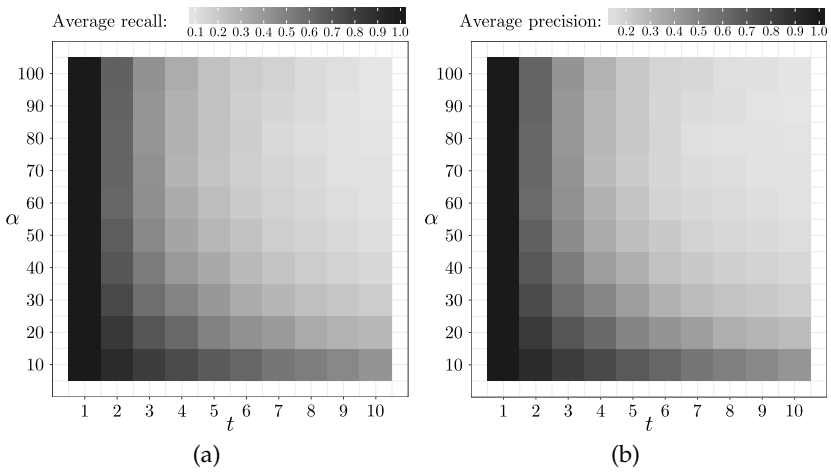


Figure 5.12: Average recall (a) and average precision (b) of CREx2-APP applied to the data sets generated by the application of $t \in [1:10]$ rearrangements that affect at most $\alpha \in \{10, 20, \ldots, 100\}$ elements to ι of size $n = 100$. For each combination of t and α 1000 problem instances were simulated with respect to each rearrangement model of $\mathcal{M}_{4\text{-type}}$, $\mathcal{M}_{I}$, $\mathcal{M}_{iT}$, $\mathcal{M}_{T}$, and $\mathcal{M}_{TDRL}$. The averages for each combination of t and α are computed including the results for all five rearrangement models.

Hence, the absence of prime nodes is a good indicator for the accuracy of the scenarios that are computed with CREx2-APP.

The following experiment shows that the values of recall and precision of the scenarios computed with CREx2-APP are much better if a certain additional biologically motivated constraint is imposed on the considered rearrangement models. More precisely, this constraint is the number of elements that can be affected by a rearrangement. Considering that rearrangements can only influence a limited number of elements is reasonable, as rearrangements tend to occur in metazoan mitochondrial genomes more frequently close to the replication origin, e. g., see Fonseca and Harris (2008). Another indicator for rearrangements of limited size is given by the analysis performed in Bernt and Middendorf (2011) on *Metazoa* mitochondrial gene orders. In their work the authors show that around 75% of rearrangements influence only tRNAs and a vast majority of those rearrangements influence only a single gene. For all these reasons, the simulation method was modified in the following experiment: Each of the t rearrangements that are randomly chosen affect at most α elements. The remaining aspects of the simulation such as the considered probability vectors, randomly selecting $t \in [1:10]$ rearrangements with respect to a probability vector as well as applying these rearrangements iteratively to the identity permutation of size $n = 100$ are kept unchanged. For the following experiment, the parameter α is explored for all values in $\{10, 20, \ldots, 100\}$. It is worth mentioning that the unrestricted case that has been analyzed in the first experiment is obtained for $\alpha = 100$. As in the previous experiment, a data set always contains 1000 problem instances and the experiment was performed for each combination of $\alpha \in \{10, 20, \ldots, 100\}$, $t \in [1:10]$, and all five

probability vectors representing the rearrangement models $\mathcal{M}_X$ with $X \in \{I, T, iT, TDRL, 4\text{-type}\}$.

Figure 5.12 illustrates the average values of recall and precision for all combinations of α and t. The illustrated averages are computed including the results of `CREx2-APP` for all five different rearrangement models. The figure shows that the values of recall and precision are much better for smaller values of α. For example, the recall for $t = 10$ and $\alpha = 100$, i. e., the worst unrestricted case, increases from 0.09 up to 0.43 for $\alpha = 10$. A similar increase can be seen for the corresponding precision values. Another example that is worth mentioning is the case were every rearrangement affects at most the half of a permutation's elements, i. e., $\alpha = 50$. In this case an average recall of 0.36 implies that more than a third of the simulated rearrangements for $\alpha = 50$ were reconstructed correctly.

One reason for the significantly improved reconstruction accuracy of `CREx2-APP` for smaller values of α can be found in the number of prime nodes: Shorter rearrangements act more often on different intervals of a permutation resulting in SITs that have a smaller number of prime nodes. As rearrangement scenarios with prime SITs are computationally hard to reconstruct, the accuracy of the results obtained by `CREx2-APP` improve with a decreasing number of prime nodes. However, the experiment shows that for these biologically more relevant restricted rearrangement models, `CREx2-APP` provides results with a much higher accuracy compared to the unconstrained rearrangement models.

The first two experiments on simulated gene order data sets that are performed in this thesis have been performed in a similar fashion for the `CREx` heuristic (Bernt et al., 2007) in Bernt (2009). While the major experimental settings are the same two important differences can be observed: 1) The analysis performed in this thesis excludes the identity permutation in each data set. The reason is that including these permutations biases the values of recall (and precision) towards 1, since an empty scenario can always be reconstructed exactly. In Bernt (2009) the identity permutations are included in the data sets. 2) In Bernt et al. (2007) a different probability vector has been used for simulating rearrangement scenarios that consider all 4-type rearrangements. While in Bernt (2009) the probability vector $(0.3, 0.3, 0.3, 0.1)$ is used, this thesis considers the vector $(0.25, 0.25, 0.25, 0.25)$. Hence, more TDRLs are applied during the simulations in this thesis. Since TDRL rearrangements often results in problem instances with a prime SIT, a direct consequence is that the data sets that have been generated with respect to all 4-type rearrangements contain 395 (4,5%) more problem instances with a prime SIT compared to Bernt (2009). Clearly, as the two differences outlined above penalize the value of recall and precision in the experiments presented in this thesis, a direct comparison of `CREx` and `CREx2-APP` on the first two experiments put the results that are presented here at a slight disadvantage. Hence, the values of recall and precision presented in Figure 4.9 of Bernt (2009) are slightly better across the whole experimental data set. In addition, it is worth mentioning that the gen-

eral tendency of recall and precision are similar and the values for the data sets that have been generated with respect to $\mathcal{M}_{4\text{-type}}$ are almost equal. Thus, when considering equally weighted types of rearrangements, `CREx2-APP` is competitive with `CREx`. Moreover, `CREx2-APP` also provides a great advantage in respect of the parsimony assumption: The scenarios computed by `CREx2-APP` contain in general less rearrangements than the scenarios of `CREx`. This aspect is explored in more detail in Section 5.5.2 on the complete set of metazoan mitochondrial gene orders.

Compared to the `CREx` heuristic another benefit of `CREx2-APP` is its ability to include rearrangement weights for every type of rearrangement. However, the choice of such weights is a difficult task which is crucial for finding reliable scenarios of gene order evolution. Moreover, up to now no preferred weighting scheme has been proposed for mitochondrial gene orders. The following experiment aims to explore the set of different weighting schemes and their influence on the reconstruction accuracy of `CREx2-APP` with respect to metazoan mitochondrial genomes.

In the following experiment, two data sets were simulated by iteratively applying $t \in [1:6]$ rearrangements to the identity permutation of size $n = 37$. For the first (second) data set the rearrangements were chosen randomly as explained above with respect to the probability vector $(0.2, 0.6, 0.1, 0.1)$ (respectively $(2/9, 2/3, 1/9, 0)$). For each t it was ensured that the identity permutation is not included in a data set and exactly 1000 problem instances were generated for each of the two data sets. The permutation size 37 has been chosen as it is the typical number of genes of metazoan mitochondrial genomes, see Section 2.1.3. The probability vector $(0.2, 0.6, 0.1, 0.1)$ was chosen according to the analysis on metazoan mitochondrial gene orders presented in Bernt and Middendorf (2011). Realizing the special role of linear SITs for the reconstruction accuracy of `CREx2-APP`, the second data set contains only problem instances having a linear strong interval tree. Therefore, the second probability vector was chosen in order to keep the relation between inversions, transpositions, and inverse transpositions as in the first probability vector while excluding TDRL rearrangements. Indeed, even for this probability vector the simulated rearrangement scenarios may result in a problem instance with a prime SITs, especially for increasing values of t. In such a case the simulation of the problem instance is repeated until a problem instance with linear SIT is generated. As the simulation almost always resulted in a problem instance with prime SIT for $t \geqslant 7$, t has been chosen to be less than 7. The maximum number of elements that can be affected by a rearrangement is set to $\alpha = 37$, i. e., the unconstrained case is considered. In sum, two data sets each containing 6 sets of 1000 problem instances were generated such that the first data set (henceforth denoted by $\mathcal{A}$) contains problem instances with linear and prime SITs and the second data set (henceforth denoted by $\mathcal{B}$) contains problem instances that have a linear SIT only.

Both simulated data sets were analyzed with `CREx2-APP` using a large number of different weighting schemes. In the following, a

weighting scheme is denoted by $(\omega_I, \omega_T, \omega_{iT})$, where ω_I (respectively ω_T, ω_{iT}) denotes the weight of an inversion (respectively transposition, inverse transposition). Note that a weighting scheme does not provide a weight for TDRL rearrangements as TDRLs can only appear in reconstructions of problem instances with a prime SIT and `CREx2-APP` solves these prime nodes by Algorithm 2 which does not consider rearrangement weights. For the experiment the weighting scheme that weights all types of rearrangements equally, i. e., $(1/3, 1/3, 1/3)$, and all weighting schemes $(x/10, y/10, z/10)$ with $x, y, z \in [1:8]$ and $x + y + z = 10$ were selected. The chosen weighting schemes were selected with the objective to provide a comprehensive view on the large set of all possible weighting schemes.

Figure 5.13 illustrates the values of recall and precision for all weighting schemes as well as for both data sets. It can be seen that recall and precision are much better for data set $\mathcal{B}$. This is not surprising as the previous experiments already showed that scenarios with a prime SIT are more unlikely to be reconstructed correctly. It can also be seen that the recall and precision for both data sets is significantly smaller for 9 weighting schemes (bright gray columns). It is not hard to see that all these weighting schemes violate at least one of the equations (5.1) – (5.3) presented on Page 139. For these 9 weighing schemes a transposition can always be replaced by a sequence of inversions and/or inverse transpositions. As `CREx2-APP` considers the weighting scheme in linear nodes of the corresponding SIT, transpositions can only occur for problem instances with prime SIT. Since the simulated scenarios consists by approximately 60% of transpositions, the values of recall and precision for these 9 weighting schemes are less than for the remaining ones. Figure 5.13 also shows that the difference in recall and precision for the remaining weighting schemes is relatively small. This can be seen in the average of recall and precision for a weighting scheme including the values for all t. For example, the largest of these averages is 0.313 for the weighting scheme $(0.3, 0.3, 0.4)$ and the average for the fifth largest-valued weighing scheme $(1/3, 1/3, 1/3)$ is 0.309 for the results of data set $\mathcal{A}$. Note that these averages of recall and precision imply that `CREx2-APP` is able to correctly reconstruct approximately a third of all simulated rearrangements. This fraction can even increase if rearrangements are bounded in size and the corresponding strong interval trees are free of prime nodes. An example for the latter is given by the average values of recall and precision that include the results of data set $\mathcal{B}$ for all t and a certain weighing scheme: the largest two averages values are 0.553 and 0.547 for the weighting scheme $(0.3, 0.3, 0.4)$ and $(1/3, 1/3, 1/3)$, respectively. Hence, in this case more than half of the simulated rearrangements were reconstructed correctly. Intriguingly, the six weighting schemes that show the largest values of recall and precision in both data sets $\mathcal{A}$ and $\mathcal{B}$ are all weighting schemes within the area where a single rearrangement cannot be replaced with a sequence of rearrangements of another type without violating the parsimony criterion. Figure 5.3 provides an illustration of this area.

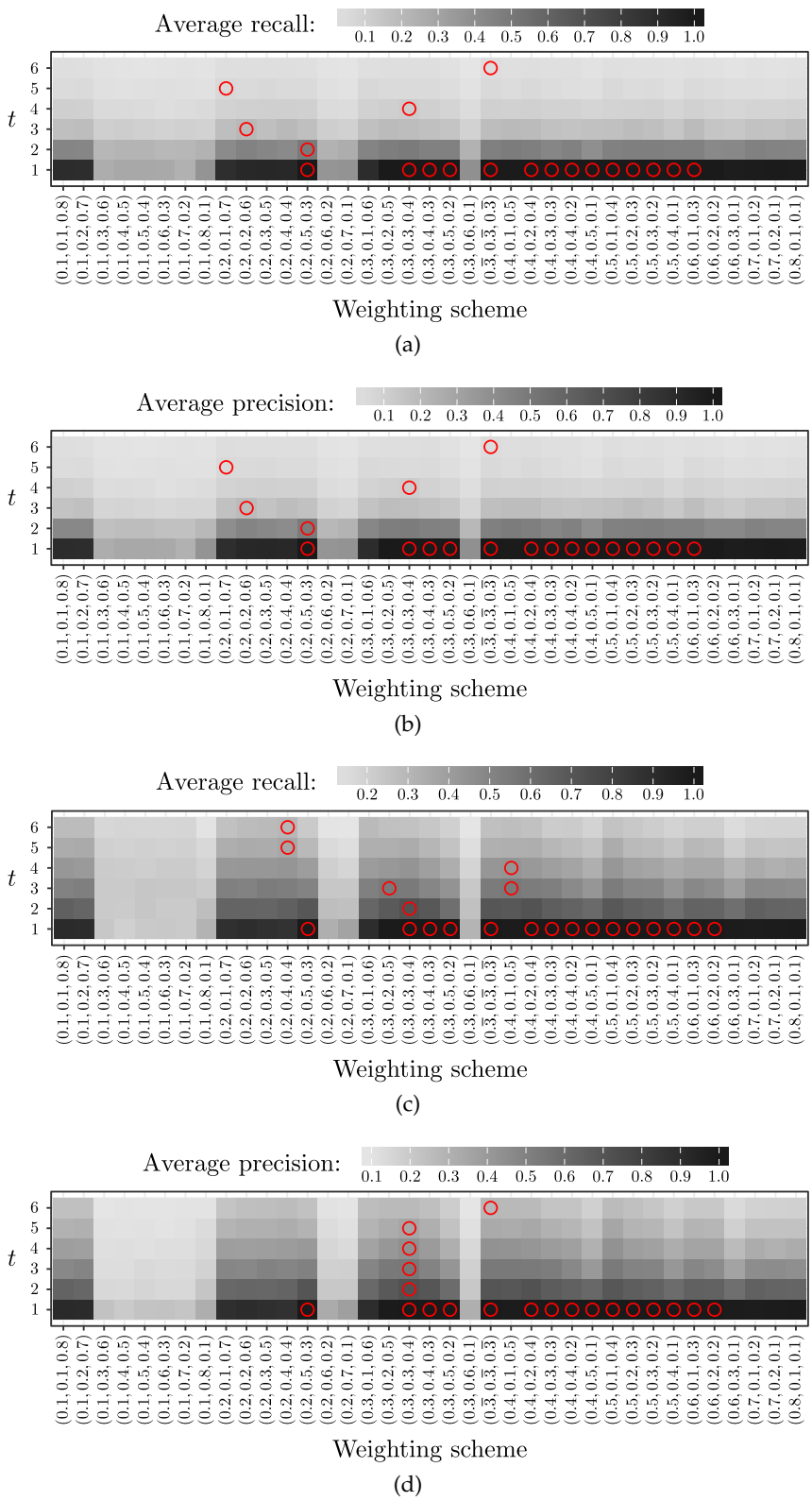


Figure 5.13: Average recall and average precision of the results of CREx2-APP on data sets $\mathcal{A}$ and $\mathcal{B}$. The values of recall and precision belonging to data set $\mathcal{A}$ are illustrated in (a) and (b), respectively. For data set $\mathcal{B}$ the corresponding values of recall and precision are given in (c) and (d), respectively. For each combination of $t \in [1:6]$ and all weighting schemes an average was computed including the results of all 1000 problem instances. A red circle indicates the averages that are maximum for t.

Consequently, the results clearly show that if the data set is simulated with respect to the probability vector $(0.2, 0.6, 0.1, 0.1)$ or $(2/9, 2/3, 1/9, 0)$, then the best reconstructions are obtained by `CREx2-APP` for weighting schemes within the area illustrated in Figure 5.3, e. g., for $(0.3, 0.3, 0.4)$ and $(1/3, 1/3, 1/3)$.

While the difference in the accuracy of the reconstructions is relatively small across most weighting schemes, the fractions of the different types of rearrangements that have been used for the reconstruction show more variation. Figure 5.14 shows those variations by depicting the average fraction of the different types of rearrangements in the reconstructions for both data sets $\mathcal{A}$ and $\mathcal{B}$. Figure 5.14 (a) shows that the fraction of TDRLs in the reconstructions is almost constant with 0.38 for all weighting schemes. Clearly, the amount of TDRLs correlates with the number of prime nodes of the corresponding SITs that are handled by Algorithm 2 in `CREx2-APP`. As the number of prime nodes in the data set is independent of the considered weighting scheme, the number of TDRLs used for the reconstructions stays constant. Since Algorithm 2 also uses inverse transpositions and transpositions, the fraction of these types of rearrangements is always larger than 0 for all weighting schemes. Another observation is that also inversions are used for all weighting schemes. The reasoning is that the sign of a single leaf node in the SIT can only be changed by an inversion rearrangement. Interestingly, Figure 5.14 (a) also shows that the fraction of a considered rearrangement event is increased as soon as its weight is decreased. For example, the fraction of inversions in the reconstructed scenarios of data set $\mathcal{A}$ increases from 0.14 for the weighting scheme $(0.8, 0.1, 0.1)$ to 0.49 for $(0.1, 0.3, 0.6)$. This effect can also be observed slightly weakened for transpositions and inverse transpositions. For transpositions (inverse transpositions) the fraction increases from 0.09 for $(0.1, 0.8, 0.1)$ (respectively 0.03 for $(0.1, 0.1, 0.8)$) to 0.35 for $(0.4, 0.1, 0.5)$ (respectively 0.28 for $(0.3, 0.6, 0.1)$). Figure 5.14 (b) illustrates the corresponding fractions for data set $\mathcal{B}$. It can be seen that no TDRL rearrangement has been used for the reconstructions. This is due to the fact that data set $\mathcal{B}$ consists of problem instances with no prime nodes in the corresponding SITs. Moreover, the effect that the fraction of a considered rearrangement event is increased for decreasing values of the respective weight can also be observed in a greater extent. For example, the fraction of inverse transpositions increases from 0 for the weighting scheme $(0.1, 0.1, 0.8)$ to 0.63 for $(0.3, 0.6, 0.1)$. In accordance to the discussion on Figure 5.13, all weighting schemes with small values of recall and precision do not contain transpositions, e. g., the scheme $(0.1, 0.6, 0.3)$.

The results of this experiment show that different weighting schemes have a significant influence on the reconstruction accuracy of `CREx2` as well as on the fractions of rearrangements that are used for the reconstruction. This influence is reduced for an increasing number of prime nodes.

The computation of all 766000 problem instances that were analyzed in this section has been performed within 56 minutes and 8 sec-

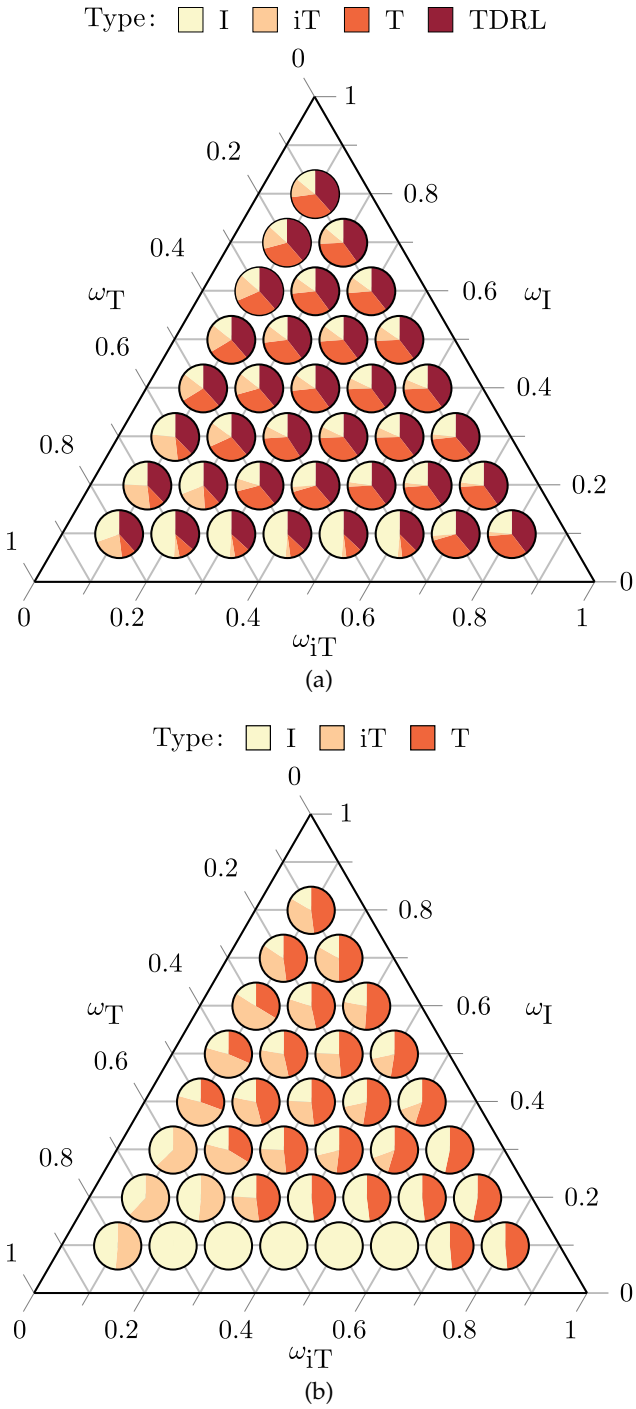


Figure 5.14: Average fractions of inversions (I), inverse transpositions (iT), transpositions (T), and tandem duplication random losses (TDRL) in the reconstructed scenarios of `CREx2-APP` for data sets $\mathcal{A}$ (a) and $\mathcal{B}$ (b). Each pie chart represents the average fractions of the different rearrangement types for a certain weighting scheme. The averages were computed including the results of `CREx2-APP` for all $t \in [1:6]$. The center of a pie chart in the ternary plot represents the corresponding weighting scheme. For a clear representation the results for the weighting scheme $(1/3, 1/3, 1/3)$ are omitted.

onds on a laptop with a 2.10 GHz processor. That is 0.0044 seconds for the reconstruction of a single rearrangement scenario on average.

5.5.2 *CREx2 on Mitochondrial Gene Order Data Sets*

In this section, gene orders of metazoan mitochondrial genomes are utilized to evaluate the performance of CREx2 on biological data sets.

For the first experiment, algorithms CREx2-APP and CREx (Bernt et al., 2007) have been applied to the set of all unique metazoan mitochondrial gene orders that were obtained as described in Section 3.2.1. The obtained set of 4900 metazoan gene orders that contain the standard set of 37 genes contains 532 unique gene orders that are used for the experiment.

Comparing all 532 gene orders pairwisely yields 282492 problem instances from which 36324 (14.76%) have a linear strong interval tree. As the experiments in Section 5.5.1 reveal, the phylogenetic reliability of the CREx2-APP results (and also the CREx results, see Bernt (2009)) is likely to be low on this data set. The reason for this lies in the small number of problem instances with a corresponding linear strong interval tree. Therefore, the aim of the following experiment is merely to demonstrate the difference of the results of CREx2-APP and CREx on biological data sets. All computations were performed on a laptop with a 2.1 GHz processor and the results of both algorithms on the data set were obtained in less than 12 minutes. That is a runtime of 0.0013 seconds (CREx) and in 0.0025 seconds (CREx2-APP) for a problem instance on average.

Figure 5.15 summarizes the results of the application of both algorithms to the data set. Box plots for the length of all reconstructed rearrangement scenarios are shown in Figure 5.15 (a). Roughly speaking, it can be seen that the scenarios of CREx2-APP are on average shorter than the scenarios constructed by CREx. More precisely, the length of 215468 (76%) (respectively 63580 (23%) and 3444 (1%)) scenarios that are constructed by CREx2-APP is less than (respectively equal to and greater than) the length of the corresponding scenario computed by CREx. If a scenario that is computed by CREx2-APP is shorter (larger) than the corresponding scenario of CREx, then it is shorter (respectively larger) by 3.71 (respectively 1.06) rearrangements on average. Altogether, the average distance of the results that are computed by CREx2-APP (respectively CREx) is 7.61 (respectively 10.44). Hence, on the data set CREx2-APP uses almost three rearrangements less than CREx on average.

Figure 5.15 (b) illustrates the absolute frequency of all four types of rearrangements for the scenarios computed with CREx and CREx2-APP. On one hand, the figure shows that CREx2-APP uses 1.28 times more TDRLs than CREx. On the other hand, the scenarios computed with CREx contain 2.77 times more inversions and 1.37 times more inverse transpositions than the corresponding scenarios computed with CREx2-APP. It can also be seen that the absolute frequency of transpositions are almost equal for both algorithms. The large differences for

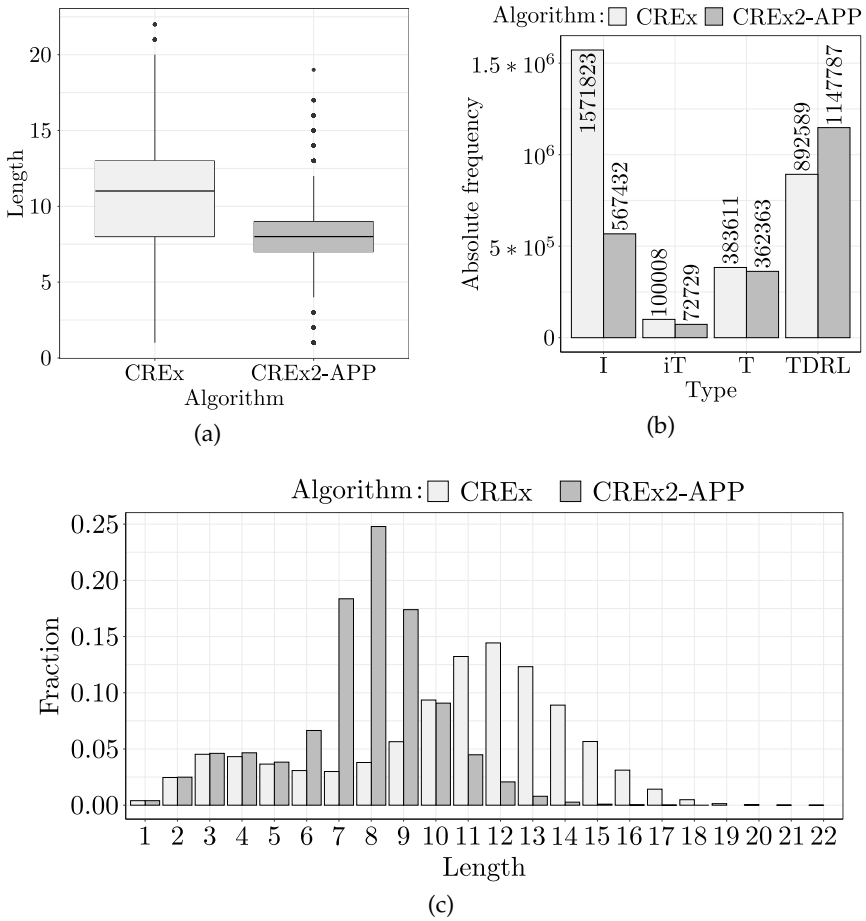


Figure 5.15: Results of the application of `CREx2-APP` and `CREx` (Bernt et al., 2007) to the data set consisting of 532 unique metazoan gene orders. (a) Box plots of the length of the scenarios that were computed by `CREx` (bright gray) and `CREx2-APP` (dark gray). (b) Absolute frequency of inversions (I), inverse transpositions (iT), transpositions (T), and tandem duplication random losses (TDRL) in the scenarios that are computed by both algorithms. (c) The fraction of the computed scenarios of both algorithms in relation to its length.

the absolute frequency of TDRLs and inversions can be explained by the different methods that both algorithms use for handling prime nodes in the corresponding SITs. While the method in `CREx` solves the corresponding sorting problem by using sequences of TDRLs and inversions, `CREx2-APP` uses all four types of genome rearrangements in which TDRLs are used prevalently. An interesting difference of the results of `CREx` and `CREx2-APP` can be seen by considering the relative fraction of the different types of rearrangements in the computed scenarios. For the scenarios computed with `CREx` transpositions, inversions, inverse transpositions, and TDRLs occur with the fraction 0.13, 0.54, 0.03, and 0.30, respectively. The respective fractions for the scenarios that are computed with `CREx2-APP` are 0.17, 0.27, 0.03, and 0.53. Hence, in the scenarios computed with `CREx2-APP` transpositions occur with a slightly higher relative fraction while the relative fraction of inverse transpositions is equal for both algorithms. The relative fractions of inversions and TDRLs seems to be inversely related:

the scenarios obtained with CREx are dominated by inversions while TDRLs dominate the scenarios obtained with CREx2-APP.

The fraction of the computed scenarios relative to its length is illustrated in Figure 5.15 (c). The figure shows that the number of scenarios with a small length (i. e., length 1 to 5) is almost equal for both algorithms. Scenarios of length 6 to 9 are computed to a much greater extent by CREx2-APP than by CREx. Apart from that, CREx computes a larger number of scenarios of a length greater than 9 than CREx2-APP.

The results show that CREx2-APP computes in general shorter rearrangements than the CREx heuristic. For the data set that contains a high number of problem instances with a prime SIT, the scenarios obtained by CREx2-APP tend to be dominated by rearrangements of type TDRL. In addition, it is shown that CREx2-APP is well-suited for large scale comparisons of metazoan mitochondrial gene orders.

The aim of the last experiment that is performed in this section is to compare the performance of both variants of CREx2, i. e., CREx2-APP and CREx2-ILP, on biological data. Recall that CREx2-ILP has an exponential runtime in the worst case. Therefore, it is not feasible to analyze the complete set of metazoan mitochondrial gene orders with CREx2-ILP. For this reason, three taxonomical subsets of all metazoan mitochondrial gene orders are utilized in order to perform the experiments in a reasonable amount of time. In particular, the considered phylogenetic groups are *Chordata*, *Ecdysozoa*, and *Lophotrochozoa*. While *Lophotrochozoa*, *Ecdysozoa*, and *Deuterostomia* are major subtaxa of *Bilateria* (which itself is a major subtaxa of the kingdom *Metazoa*), *Chordata* is a phylum within *Deuterostomia*. The three taxonomic groups are chosen with respect to the diversity of their gene orders which is relatively low for *Chordata* and comparably high for *Lophotrochozoa* and *Ecdysozoa*, e. g., see Bernt et al. (2013a) and Podsiadlowski et al. (2009).

For all gene orders that are considered, the highly variable positions of tRNAs were excluded. Hence, each gene order in the data sets contains 13 protein-encoding genes and two ribosomal genes. Excluding the tRNAs is done in accordance to the presumption that mitochondrial tRNAs are selectively neutral, see Dowton et al. (2009) and Section 2.1.3. The resulting data sets consists of 11 *Chordata*, 63 *Ecdysozoa*, and 43 *Lophotrochozoa* unique gene orders. Taking the fact into consideration that the 4900 metazoan mitochondrial gene orders in the RefSeq release 89 contain 3220 (66%) chordates, 1422 (29%) ecdysozoan, and 222 (0.05%) lophotrochozoan gene orders, it can be seen that the number of different gene orders among the considered gene groups is highly different. While for *Chordata* only a few different gene orders can be found, the relative number of different gene orders is much greater in *Ecdysozoa* and even more in *Lophotrochozoa*. However, this effect might also be caused by uneven sampling. Nevertheless, the diversity of the gene orders within the different data sets is also supported by the number of linear strong interval trees for pairwise gene order comparisons. In particular, more than half (67.3%) of the pairwise comparisons of the *Chordata* mitochondrial gene orders have no prime node in the corresponding strong interval

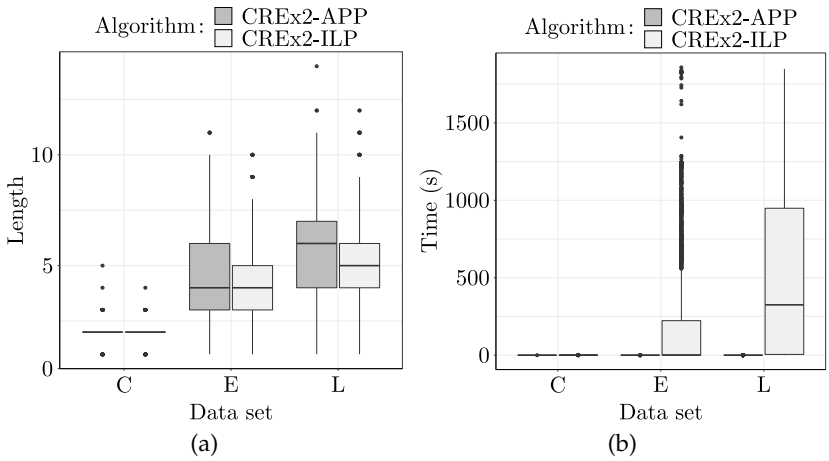


Figure 5.16: Results of the application of `CREx2-ILP` (bright gray) and `CREx2-APP` (dark gray) to pairs of gene orders from the *Chordata* (C), *Ecdysozoa* (E), and *Lophotrochozoa* (L) data set. Box plots showing (a) the length of the constructed scenarios and (b) the corresponding runtimes of both algorithms for all three data sets.

trees. On the contrary, only a minor fraction of the strong interval trees of the *Ecdysozoa* (29.07%) and *Lophotrochozoa* (9.04%) data sets have a linear strong interval tree.

The computation of all pairwise rearrangement scenarios for each of the three data sets with `CREx2-APP` and `CREx2-ILP` was performed on a single core of an AMD Opteron 2435 with 2.6 GHz. All pairwise scenarios were computed with equally weighted rearrangements which is in accordance with the results obtained in Section 5.5.1. To limit the runtime of `CREx2-ILP` a time limit of 300 seconds was used for a single prime node, i. e., $\mathcal{L} = 300$. Recall that if a computation of a prime node exceeds 300 seconds, then the solution obtained by `CREx2-ILP` might not be exact.

Within the time limit 110 (100%), 3351 (85,79%), and 1030 (57.03%) of the pairwise comparisons were solved exactly by `CREx2-ILP` for the *Chordata*, *Ecdysozoa*, and *Lophotrochozoa* data set, respectively. Hence, `CREx2-ILP` was able to exactly compute more than 88% of all comparisons. Figure 5.16 (a) shows box plots for the lengths of the scenarios obtained by both algorithms. It can be seen that the lengths of the scenarios obtained by both algorithms are almost equivalent for the *Chordata* data set. For the *Ecdysozoa* and *Lophotrochozoa* data sets it is shown that `CREx2-ILP` produces shorter scenarios than `CREx2-APP`. In particular, the average length of a scenario is 3.9 (respectively 5.2) for `CREx2-ILP` and 4.2 (respectively 5.6) for `CREx2-APP` for the *Ecdysozoa* (respectively *Lophotrochozoa*) data set. Recall that the relative fraction of prime nodes is much higher in the *Lophotrochozoa* data set than in the *Ecdysozoa* data set and that the *Chordata* data set has a much smaller fraction than both other data sets. As both algorithms are identical for linear nodes in the strong interval trees, the difference in the scenario lengths increases for an increasing number of prime nodes. Overall, `CREx2-ILP` produces shorter rearrangements scenar-

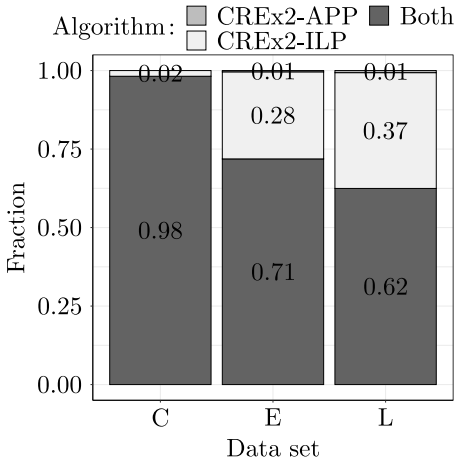


Figure 5.17: Fractions of pairwise comparisons of gene orders from the *Chordata* (C), *Ecdysozoa* (E), and *Lophotrochozoa* (L) data set for which the shortest scenario was produced either with `CREx2-ILP` (bright gray), `CREx2-APP` (gray), or both algorithms (dark gray).

ios than `CREx2-APP` on average. However, the price for this is a significantly increased runtime as shown in Figure 5.16 (b).

Figure 5.17 shows the fractions of problem instances for which one of the algorithms yields a smaller, an equal, or a larger scenario length. It can be seen that almost all scenarios (108 of 110) obtained by `CREx2-APP` and `CREx2-ILP` have the same length for gene order pairs of the *Chordata* date set. For the remaining two gene order pairs of the *Chordata* date set `CREx2-ILP` computed scenarios that use one rearrangement less than the corresponding scenarios obtained by `CREx2-APP`. The figure also shows that in the 3906 pairwise comparisons (i. e., the *Ecdysozoa* data set) the resulting scenarios of `CREx2-ILP` and `CREx2-APP` are equal for 2807 (71%) instances, for 1083 (28%) instances `CREx2-ILP` provides shorter scenarios than `CREx2-APP`, and for 16 (1%) instances `CREx2-APP` provides shorter scenarios than `CREx2-ILP`. It may seems surprising that `CREx2-APP` is able to produce shorter rearrangement scenarios than `CREx2-ILP`. However, this case can occur if `CREx2-ILP` is used with a time limit $\mathcal{L}$ for solving a prime node in the considered SIT. In particular, this case can occur if $\mathcal{L}$ is exceeded and an unfavorable signed quotient permutation is chosen by the fallback method, see Section 5.4.4. On average, a scenario obtained with `CREx2-APP` is shorter (longer) than the scenario obtained with `CREx2-ILP` by a factor of 1.69 (respectively 1.13). For the *Lophotrochozoa* data set, both algorithms obtained solutions with an equal length for 1128 (62%) gene order pairs and for 666 (37%) gene order pairs `CREx2-ILP` produced a shorter scenario than `CREx2-APP`. If a scenario of `CREx2-ILP` is shorter (larger) than the corresponding scenario computed by `CREx2-APP`, then the average difference is 1.2 (respectively 2.1) rearrangements. Finally, it can be seen that the fraction of instances for which both algorithms generate solutions of the same length becomes smaller as the relative number of prime nodes in the data set increases.

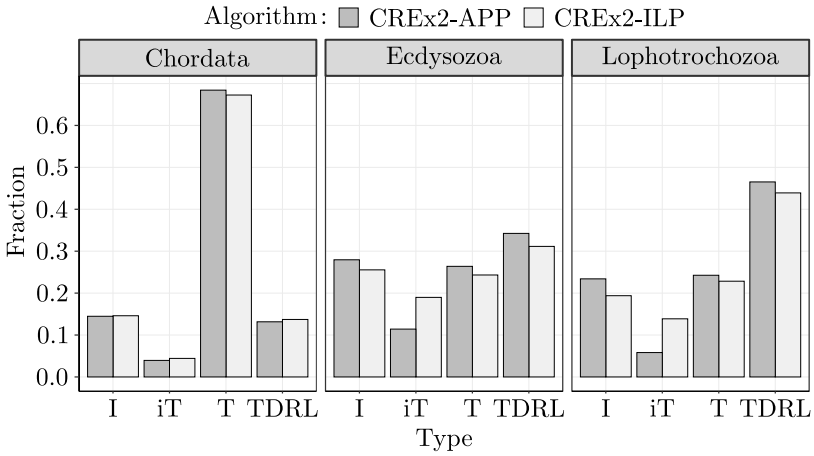


Figure 5.18: Distribution of inversions (I), inverse transpositions (iT), transpositions (T), and tandem duplication random losses (TDRL) in the scenarios obtained by the application of `CREx2-ILP` (bright gray) and `CREx2-APP` (dark gray) to the *Chordata*, *Ecdysozoa*, and *Lophotrochozoa* data set.

Figure 5.18 shows the distribution of the different types of rearrangements for both algorithms and all three data sets. It is shown that the distributions of the different types of rearrangements in the computed scenarios are not substantially different for both algorithms. While the distribution is almost equivalent for the *Chordata* data set, the distributions for the ecdysozoan and lophotrochozoan data sets reveal that `CREx2-ILP` uses a smaller number of inversions, transpositions, and TDRLs and a larger number of inverse transpositions. It is worth mentioning that the *Chordata* data set is dominated by transposition rearrangements. This is not the case in the other two data sets in which TDRLs are used predominantly.

The results of the experiments on biological data show that the solutions of `CREx2` are likely to improve the solutions of the `CREx` heuristic with respect to the parsimony assumption. In addition, for almost 80% of the gene order comparisons both variants of `CREx2` compute rearrangement scenarios that have the same length and approximately the same distribution of the different types of 4-type rearrangements. For an increasing number of prime nodes in the considered data set `CREx2-ILP` computes shorter scenarios than `CREx2-APP`. However, to obtain these solution `CREx2-ILP` needs significantly more runtime. The two variants of `CREx2` are intended to be utilized for different purposes. While `CREx2-APP` is well-suited for large-scale comparisons of mitochondrial gene orders, `CREx2-ILP` gives the possibility to obtain exact rearrangement scenarios for a small-sized data sets.

5.6 CONCLUSION

This chapter has investigated the sorting problem and the distance problem for signed linear permutations under the genome rearrangement model $\mathcal{M}_{4\text{-type}}$ that considers the four types of rearrangements relevant for evolution of metazoan mitochondrial genomes, i. e., in-

version, transposition, inverse transposition, and tandem duplication random loss (TDRL). It has been shown that the distance problem can sufficiently be approximated such that the rearrangement distance is larger by at most 2. A corresponding algorithm that computes rearrangement scenarios for two given gene orders while achieving this approximation factor has been developed. It has been shown that the general sorting problem under $\mathcal{M}_{4\text{-type}}$ is less valuable for the reconstruction of biologically meaningful rearrangement scenarios of mitochondrial gene order evolution. The reason is that TDRLs are predominantly used in parsimonious rearrangement scenarios which contradicts with the literature on mitochondrial evolution, e. g., see Section 2.1.3. To address this problem, methods for two biological motivated variants of the sorting problem under $\mathcal{M}_{4\text{-type}}$ have been presented. The first variant has considered that a weighting scheme is employed on the $\mathcal{M}_{4\text{-type}}$ model weighting every rearrangement by its type. The resulting weighted sorting problem has been solved by means of an integer linear program. The second variant has extended the first one by explicitly enforcing rearrangement scenarios between gene orders to preserve certain gene clusters, i. e., groups of genes that occur in both considered gene orders in close proximity. Those clusters are represented formally by the notion of common intervals of permutations. The exact dynamic programming algorithm `CREx2` that solves the corresponding problem efficiently for large classes of problem instances has been proposed. `CREx2` is based on the strong interval tree data structure and can compute an exact solution in linear runtime for gene order pairs for which the corresponding strong interval tree is organized in a linear structure. For pairs of gene orders with a non-linear strong interval tree two variants of `CREx2` have been provided: `CREx2-APP` computes approximated rearrangement scenarios efficiently and `CREx2-ILP` computes exact solutions but has a worst case exponential runtime. Thereby, a significant improvement of the heuristic algorithm `CREx` (Bernt et al., 2007) has been provided.

An empirical evaluation of `CREx2` has been conducted on artificial and biological gene order data. The results have shown that `CREx2` is able to compute parsimonious rearrangement scenarios for most pairs of gene orders, even if only a small fraction has a linear strong interval tree. It has empirically been demonstrated that `CREx2` is able to reconstruct gene order rearrangement scenarios with reliable accuracy, especially in the case of problem instances with a linear strong interval tree. Biologically motivated parameters, properties of the strong interval tree, and different weighting schemes that increase the accuracy of the `CREx2` reconstructions have been determined. In addition, it has been demonstrated how different weighting schemes for the types of rearrangements influence the fractions of these types in the constructed solutions.

Algorithm `CREx2-APP` has successfully been applied to the complete metazoan mitochondrial gene order data set. Thereby allowing a comparison of `CREx` and `CREx2` which has indicated that the solutions obtained by `CREx2` are likely to improve the solutions of the `CREx`

heuristic. Furthermore, a comparison of `CREx2-APP` and `CREx2-ILP` has been performed showing the advantages and disadvantages of both methods: `CREx2-APP` facilitates an efficient exploration of gene order comparisons for large-sized sets of gene orders, but in some cases it violates the parsimony principle by computing approximated rearrangement scenarios. `CREx2-ILP` provides exact solutions, but has a significantly increased runtime.

6 CONCLUSION

Understanding the evolutionary history and relationships among living beings is one of the central problems in evolutionary biology. A powerful approach to address this problem is to compare the genetic information of a pair of species. Especially animal mitochondrial genomes have been shown to be valuable for supporting various phylogenetic hypotheses. The reason is that these genomes exhibit beneficial characteristics such as a small size, a considerable conservation of their genes, and, most often, the absence of gene duplicates. With these characteristics, gene orders of metazoan mitochondrial genomes can be modeled formally by permutations. Representing evolutionary events on mitochondrial genomes as rearrangement operations on permutations allows to study mitochondrial evolution by theoretical analyses. Two central algorithmic problems for such analyses are the sorting problem and the distance problem. The sorting problem asks for the shortest sequence of rearrangements that transforms one permutation into another permutation. The distance problem aims to find the length of such a shortest rearrangement sequence. In this thesis both of these fundamental genome rearrangement problems have been studied under various rearrangement models. The main focus was set on the tandem duplication random loss (TDRL) genome rearrangement model which has been proven to be indispensable for understanding the evolution of metazoan mitochondrial genomes. Thereby, previous works on the TDRL rearrangement model have been extended significantly.

The first problem analyzed in this work is the distance problem for circular permutations under the TDRL rearrangement model. This problem is relevant since the usual mitochondrial genome is organized in a single circular structure and TDRL rearrangements are common in such genomes. It has been shown that the TDRL distance, i. e., the solution of the distance problem under the TDRL model, between two circular permutations is either less by one or equal to the TDRL distance of the corresponding linear representatives. Moreover, it has been shown that this difference is even larger if the circular permutations have no pre-defined reading direction. The results have revealed that using an unfavorable choice of linear representatives may lead to an overestimation of the TDRL distance. A formula for computing the probability of this error has been provided. It has been demonstrated empirically on metazoan mitochondrial gene orders that the circularity of the genomes should be considered, since otherwise the TDRL distance is overestimated for a considerable fraction of the pairwise gene order comparisons. Furthermore, combinatorial properties of the TDRL rearrangement model on circular permutations have led to a characterization of sets of equivalent TDRLs, i. e., TDRLs that have the same effect when applied to the same permuta-

tion. The relevance of the theoretical findings was pointed out by a detailed analysis of two pairs of gene orders that have been used in the literature to argue for the tandem duplication non-random loss model. The analysis has highlighted the importance of explicitly studying the circular case.

The inverse tandem duplication random loss (iTDRL) rearrangement model, a variant of the TDRL model, has currently been suggested to be a potential evolutionary mechanism in mitochondrial genomes. In this work, the distance problem and the sorting problem with respect to iTDRLs have been studied. It has been shown that the iTDRL distance can be computed in linear time, and that a corresponding scenario that solves the sorting problem can be obtained in quasilinear time. Thereby, the combinatorial exploration of the iTDRL model on signed linear permutations has been initialized. One characteristic that makes the iTDRL model particularly interesting is that it can mimic all types of rearrangements that are predominant in metazoan mitochondrial genomes, i. e., inversions, transpositions, inverse transpositions, and TDRLs. Using this benefit, it has been shown that the iTDRL distance is a 2-approximation on the minimum number of inversions, transpositions, inverse transpositions, and TDRLs that are necessary to transform one given permutation into another.

The last part of this thesis has been devoted to investigate the 4-type rearrangement model. This model is especially attractive as it considers all major rearrangement operations that are relevant for metazoan mitochondrial genomes, i. e., inversions, transpositions, inverse transpositions, and TDRLs. The distance problem for signed linear permutations under the 4-type model has been studied. It has been shown that the 4-type distance can be approximated in linear time such that the discrepancy from the actual distance is not more than two. As a byproduct, a quasilinear approximation algorithm has been obtained for the corresponding sorting problem. The algorithm guarantees to compute rearrangement scenarios that differ from minimum length scenarios by at most two rearrangement operations. Thereby, an important step towards an efficient and exact resolution of the sorting problem under the 4-type model has been made. Based on these results, the insight has been gained that the general sorting problem under the 4-type model is less valuable for the inference of plausible reconstructions of mitochondrial gene order evolution. This is due to the fact that parsimonious rearrangement scenarios contain predominantly rearrangements of type TDRL which is – up to now – not supported by the literature on mitochondrial evolution. As a consequence, two biologically motivated variants of the sorting problem, that are more relevant for mitochondrial evolution, have been investigated.

The first variant considers a weighting scheme on the 4-type rearrangement model in which each rearrangement is weighted with respect to its type. Since the corresponding sorting problem is NP-hard, one cannot hope for an efficient algorithm that solves all problem instances. Therefore, the polynomial-size integer linear program

GeRe-ILP has been proposed. GeRe-ILP provides an exact solution, but has an exponential runtime in the worst case. The second variant extents the first one by enforcing scenarios of rearrangements to preserve certain gene clusters of the input gene orders. In this work, those gene clusters were formally modeled by common intervals of the input permutations. To solve the corresponding sorting problem, the exact dynamic programming algorithm CREx2 has been proposed. For two given signed linear permutations and a weight value for every type of rearrangement, CREx2 is able to compute a scenario of 4-type rearrangements that has a minimum weight and preserves the common intervals of the considered permutations. In addition, CREx2 provides exact solutions within linear runtime for a large class of problem instances in which the common intervals are organized in a linear structure. Thereby, a significant improvement of the heuristic algorithm CREx (Bernt et al., 2007) has been provided. For the case that the common intervals are organized in a non-linear structure two variants of CREx2 have been developed, each making a different trade-off between exactness and runtime: CREx2-APP runs efficiently but gives only approximate results. CREx2-ILP makes the opposite trade-off by finding exact solutions but lagging behind in terms of runtime and memory requirements.

The accuracy of the CREx2 has been analyzed empirically in a study for simulated artificial and real metazoan mitochondrial gene orders. For the simulated data set, it has been demonstrated empirically that CREx2 is able to reconstruct a significant fraction of the simulated rearrangement events for different rearrangement models. However, the reliability of the CREx2 reconstructions drops if the common intervals of a considered problem instance are organized in a non-linear structure. A comparison of CREx2-APP and the CREx heuristic on both data sets has demonstrated that CREx2 is able to compute competitive rearrangement scenarios and allows to efficiently analyze large data sets of hundreds of gene orders.

The topics studied in this thesis are far from being exhaustively explored. Instead, the results presented in this thesis pave the way for many avenues of further research – to mention only two: The effect of considering the circular structure of mitochondrial gene orders has only been analyzed for the TDRL model in this thesis. A particularly interesting research question that needs further investigation is the study of the fundamental genome rearrangement problems for circular permutations under the iTDRL or the 4-type rearrangement model. Another challenging and still unresolved research question, for which this work may serve as a starting point, aims for the computational complexity of the median problem under the TDRL model.

BIBLIOGRAPHY

Abou-Sleiman, P. M., M. M.-K. Muqit, and N. W. Wood (2006). "Expanding insights of mitochondrial dysfunction in Parkinson's disease." In: *Nature Reviews Neuroscience* 7.3, p. 207.

Adam, Z., M. Turmel, C. Lemieux, and D. Sankoff (2007). "Common intervals and symmetric difference in a model-free phylogenomics, with an application to streptophyte evolution." In: *Journal of Computational Biology* 14.4, pp. 436–445.

Adams, K. L., D. O. Daley, Y.-L. Qiu, J. Whelan, and J. D. Palmer (2000). "Repeated, recent and diverse transfers of a mitochondrial gene to the nucleus in flowering plants." In: *Nature* 408.6810, p. 354.

Aguileta, G., D. M. De Vienne, O. N. Ross, M. E. Hood, T. Giraud, E. Petit, and T. Gabaldón (2014). "High variability of mitochondrial gene order among fungi." In: *Genome Biology and Evolution* 6.2, pp. 451–465.

Aigner, M., G. M. Ziegler, K. H. Hofmann, and P. Erdos (2010). *Proofs from the Book*. Vol. 274. Springer.

Ajana, Y., J.-F. Lefebvre, E. R. M. Tillier, and N. El-Mabrouk (2002). "Exploring the set of all minimal sequences of reversals – an application to test the replication-directed reversal hypothesis." In: *Proc. 2nd Int'l Workshop on Algorithms in Bioinformatics (WABI 2002)*. Springer, pp. 300–315.

Alberts B., Johnson A. Lewis J. Raff M. Roberts K. and P. Walter (2003). *Molecular biology of the cell (4th ed.)* Garland Science.

Aldous, D. and P. Diaconis (1986). "Shuffling cards and stopping times." In: *The American Mathematical Monthly* 93.5, pp. 333–348.

Alekseyev, M. A. and P. A. Pevzner (2008). "Multi-break rearrangements and chromosomal evolution." In: *Theoretical Computer Science* 395.2-3, pp. 193–202.

Alexeyev, M., I. Shokolenko, G. Wilson, and S. LeDoux (2013). "The maintenance of mitochondrial DNA integrity – critical analysis and update." In: *Cold Spring Harbor Perspectives in Biology* 5.5, a012641.

Angibaud, S., G. Fertin, I. Rusu, and S. Vialette (2006). "How pseudo-boolean programming can help genome rearrangement distance computation." In: *Proc. 4th Int'l Workshop on Comparative Genomics (RECOMB-CG 2006)*. Springer, pp. 75–86.

Angibaud, S., G. Fertin, I. Rusu, A. Thévenin, and S. Vialette (2007). "A pseudo-boolean programming approach for computing the breakpoint distance between two genomes with duplicate genes." In: *Proc. 5th Int'l Workshop on Comparative Genomics (RECOMB-CG 2007)*. Springer, pp. 16–29.

— (2009). "On the approximability of comparing genomes with duplicates." In: *Journal of Graph Algorithms and Applications* 13.1, pp. 19–53.

Applegate, D. L., R. E. Bixby, V. Chvàtal, and W. J. Cook (2006). *The traveling salesman problem: a computational study*. Princeton University Press.

Arndt, A. and M. J. Smith (1998). "Mitochondrial gene rearrangement in the sea cucumber genus Cucumaria." In: *Molecular Biology and Evolution* 15.8, pp. 1009–1016.

Asakawa, S., H. Himeno, K.-I. Miura, and K. Watanabe (1995). "Nucleotide sequence and gene organization of the starfish Asterina pectinifera mitochondrial genome." In: *Genetics* 140.3, pp. 1047–1060.

Awadalla, P., A. Eyre-Walker, and J. M. Smith (1999). "Linkage disequilibrium and recombination in hominid mitochondrial DNA." In: *Science* 286.5449, pp. 2524–2525.

Bader, D. A., B. M. E. Moret, and M. Yan (2001). "A linear-time algorithm for computing inversion distance between signed permutations with an experimental study." In: *Proc. 7th Workshop on Algorithms and Data Structures (WADS 2001)*. Springer, pp. 365–376.

Bader, M. (2011). "The transposition median problem is NP-complete." In: *Theoretical Computer Science* 412.12-14, pp. 1099–1110.

Bader, M. and E. Ohlebusch (2007). "Sorting by weighted reversals, transpositions, and inverted transpositions." In: *Journal of Computational Biology* 14.5, pp. 615–636.

Badrinarayanan, A., T. B. Le, and M. T. Laub (2015). "Bacterial chromosome organization and segregation." In: *Annual Review of Cell and Developmental Biology* 31, pp. 171–199.

Bafna, V. and P. A. Pevzner (1996). "Genome rearrangements and sorting by reversals." In: *SIAM Journal on Computing* 25.2, pp. 272–289.

Bafna, V. and P. Pevzner (1995). "Sorting permutations by tansponsitions." In: *Proc. 6th ACM-SIAM Symposium on Discrete Algorithms (SODA 1995)*. Society for Industrial and Applied Mathematics, pp. 614–623.

Baril, J.-L. and R. Vernay (2010). "Whole mirror duplication-random loss model and pattern avoiding permutations." In: *Information Processing Letters* 110.11, pp. 474–480.

Basso, A., M. Babbucci, M. Pauletto, E. Riginella, T. Patarnello, and E. Negrisolo (2017). "The highly rearranged mitochondrial genomes of the crabs Maja crispata and Maja squinado (Majidae) and gene order evolution in Brachyura." In: *Scientific reports* 7.1, p. 4096.

Bayer, D. and P. Diaconis (1992). "Trailing the dovetail shuffle to its lair." In: *The Annals of Applied Probability* 2.2, pp. 294–313.

Béal, M.-P., A. Bergeron, S. Corteel, and M. Raffinot (2004). "An algorithmic view of gene teams." In: *Theoretical Computer Science* 320.2-3, pp. 395–418.

Beckenbach, A. T. (2011). "Mitochondrial genome sequences of Nematocera (lower Diptera): evidence of rearrangement following a complete genome duplication in a winter crane fly." In: *Genome Biology and Evolution* 4.2, pp. 89–101.

Bender, M. A., D. Ge, S. He, H. Hu, R. Y. Pinter, S. Skiena, and F. Swidan (2008). "Improved bounds on sorting by length-weighted reversals." In: *Journal of Computer and System Sciences* 74.5, pp. 744–774.

Bérard, S., A. Bergeron, and C. Chauve (2004). "Conservation of combinatorial structures in evolution scenarios." In: *Proc. 2nd Int'l Workshop on Comparative Genomics (RECOMB-CG 2004)*. Springer, pp. 1–14.

Bérard, S., A. Bergeron, C. Chauve, and C. Paul (2007). "Perfect sorting by reversals is not always difficult." In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 4, pp. 4–16.

Bérard, S., C. Chauve, and C. Paul (2008). "A more efficient algorithm for perfect sorting by reversals." In: *Information Processing Letters* 106.3, pp. 90–95.

Bérard, S., A. Chateau, C. Chauve, C. Paul, and E. Tannier (2009). "Computation of perfect DCJ rearrangement scenarios with linear and circular chromosomes." In: *Journal of Computational Biology* 16.10, pp. 1287–1309.

Bergeron, A. (2001). "A very elementary presentation of the Hannenhalli-Pevzner theory." In: *Proc. 12th Symposium on Combinatorial Pattern Matching (CPM 2001)*. Springer, pp. 106–117.

Bergeron, A. and J. Stoye (2006). "On the similarity of sets of permutations and its applications to genome comparison." In: *Journal of Computational Biology* 13.7, pp. 1340–1354.

— (2013). "The genesis of the DCJ formula." In: *Models and Algorithms for Genome Evolution*. Springer, pp. 63–81.

Bergeron, A., S. Heber, and J. Stoye (2002a). "Common intervals and sorting by reversals: a marriage of necessity." In: *Bioinformatics* 18.Suppl 2.

Bergeron, A., C. Chauve, T. Hartman, and K. St-Onge (2002b). "On the properties of sequences of reversals that sort a signed permutation." In: *Proceedings of JOBIM* 2, pp. 99–108.

Bergeron, A., S. Corteel, and M. Raffinot (2002c). "The algorithmic of gene teams." In: *Proc. 2nd Int'l Workshop Algorithms in Bioinformatics (WABI 2002)*. Springer, pp. 464–476.

Bergeron, A., J. Mixtacki, and J. Stoye (2004). "Reversal distance without hurdles and fortresses." In: *Proc. 15th Symposium on Combinatorial Pattern Matching (CPM 2004)*. Springer, pp. 388–399.

— (2006). "A unifying view of genome rearrangements." In: *Proc. 6th Int'l Workshop Algorithms in Bioinformatics (WABI 2006)*. Springer, pp. 163–173.

Bergeron, A., C. Chauve, F. De Montgolfier, and M. Raffinot (2008). "Computing common intervals of k permutations, with applications to modular decomposition of graphs." In: *SIAM Journal on Discrete Mathematics* 22.3, pp. 1022–1039.

Bergeron, A., P. Medvedev, and J. Stoye (2010). "Rearrangement models and single-cut operations." In: *Journal of Computational Biology* 17.9, pp. 1213–1225.

Berman, P. and S. Hannenhalli (1996). "Fast sorting by reversal." In: *Proc. 7th Symposium on Combinatorial Pattern Matching (CPM 1996)*. Springer, pp. 168–185.

Berman, P. and M. Karpinski (1999). "On some tighter inapproximability results." In: *Proc. 26th Int'l Colloquium on Automata, Languages, and Programming (ICALP 1999)*. Springer, pp. 200–209.

Berman, P., S. Hannenhalli, and M. Karpinski (2002). "1.375-approximation algorithm for sorting by reversals." In: *Proc. 1st European Symposium on Algorithms (ESA 2002)*. Springer, pp. 200–210.

Bernt, M. and M. Middendorf (2011). "A method for computing an inventory of metazoan mitochondrial gene order rearrangements." In: *BMC Bioinformatics*. Vol. 12. 9. BioMed Central, S6.

Bernt, M., D. Merkle, and M. Middendorf (2006). "Genome rearrangement based on reversals that preserve conserved intervals." In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 3.3, pp. 275–288.

Bernt, M., D. Merkle, K. Ramsch, G. Fritzsch, M. Perseke, D. Bernhard, M. Schlegel, P. F. Stadler, and M. Middendorf (2007). "CREx: inferring genomic rearrangements based on common intervals." In: *Bioinformatics* 23.21.

Bernt, M., D. Merkle, and M. Middendorf (2008a). "An algorithm for inferring mitogenome rearrangements in a phylogenetic tree." In: *Proc. 6th International Workshop on Comparative Genomics (RECOMB-CG 2008)*. Springer, pp. 143–157.

— (2008b). "Solving the preserving reversal median problem." In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 5.3, pp. 332–347.

Bernt, M., K.-Y. Chen, M.-C. Chen, A.-C. Chu, D. Merkle, H.-L. Wang, K.-M. Chao, and M. Middendorf (2011). "Finding all sorting tandem duplication random loss operations." In: *Journal of Discrete Algorithms* 9.1, pp. 32–48.

Bernt, M., C. Bleidorn, A. Braband, J. Dambach, A. Donath, G. Fritzsch, A. Golombek, H. Hadrys, F. Jühling, K. Meusemann, et al. (2013a). "A comprehensive analysis of bilaterian mitochondrial genomes and phylogeny." In: *Molecular Phylogenetics and Evolution* 69.2, pp. 352–364.

Bernt, M., A. Braband, B. Schierwater, and P. F. Stadler (2013b). "Genetic aspects of mitochondrial genome evolution." In: *Molecular Phylogenetics and Evolution* 69.2, pp. 328–338.

Bernt, M., A. Donath, F. Jühling, F. Externbrink, C. Florentz, G. Fritzsch, J. Pütz, M. Middendorf, and P. F. Stadler (2013c). "MITOS: improved de novo metazoan mitochondrial genome annotation." In: *Molecular Phylogenetics and Evolution* 69.2, pp. 313–319.

Bernt, M., N. Wieseke, and M. Middendorf (2013d). "On Weighting Schemes for Gene Order Analysis." In: *Proc. 5th German Conference on Bioinformatics (GCB 2013)*, pp. 14–23.

Bernt, Matthias (2009). "Gene order rearrangement methods for the reconstruction of phylogeny." PhD thesis. University Leipzig.

Bertsimas, D. and J. N. Tsitsiklis (1997). *Introduction to linear optimization*. Vol. 6. Athena Scientific Belmont, MA.

Bhatia, S., P. Feijão, and A. R. Francis (2016). "Position and content paradigms in genome rearrangements: the wild and crazy world of permutations in genomics." In: *arXiv preprint arXiv:1610.00077*.

Bianconi, E., A. Piovesan, F. Facchin, A. Beraudi, R. Casadei, F. Frabetti, L. Vitale, M. C. Pelleri, S. Tassani, F. Piva, et al. (2013). "An estimation of the number of cells in the human body." In: *Annals of Human Biology* 40.6, pp. 463–471.

Blanchette, M., T. Kunisawa, and D. Sankoff (1996). "Parametric genome rearrangement." In: *Gene* 172.1, GC11–GC17.

Bleidorn, C., I. Eeckhaut, L. Podsiadlowski, N. Schult, D. McHugh, K. M. Halanych, M. C. Milinkovitch, and R. Tiedemann (2007). "Mitochondrial genome and nuclear sequence data support Myzostomida as part of the annelid radiation." In: *Molecular Biology and Evolution* 24.8, pp. 1690–1701.

Blin, G., D. Faye, and J. Stoye (2010). "Finding nested common intervals efficiently." In: *Journal of Computational Biology* 17.9, pp. 1183–1194.

Bóna, M. (2004). *Combinatorics of permutations*. Chapman and Hall/CRC.

Boore, J. L. (1999). "Animal mitochondrial genomes." In: *Nucleic Acids Research* 27.8, pp. 1767–1780.

— (2000). "The duplication/random loss model for gene rearrangement exemplified by mitochondrial genomes of deuterostome animals." In: *Comparative Genomics: Empirical and analytical approaches to gene order dynamics, map alignment and the evolution of gene families*. Springer, pp. 133–147.

— (2006). "The complete sequence of the mitochondrial genome of Nautilus macromphalus (Mollusca: Cephalopoda)." In: *BMC Genomics* 7.1, p. 182.

Boore, J. L. and W. M. Brown (1998). "Big trees from little genomes: mitochondrial gene order as a phylogenetic tool." In: *Current Opinion in Genetics & Development* 8.6, pp. 668–674.

Boore, J. L., T. M. Collins, D. Stanton, L. L. Daehler, and W. M. Brown (1995). "Deducing the pattern of arthropod phytogeny from mitochondrial DNA rearrangements." In: *Nature* 376.6536, pp. 163–165.

Boore, J. L., D. V. Lavrov, and W. M. Brown (1998). "Gene translocation links insects and crustaceans." In: *Nature* 392.6677, p. 667.

Boore, J. L., J. R. Macey, and M. Medina (2005). "Sequencing and comparing whole mitochondrial genomes of animals." In: *Methods in Enzymology*. Vol. 395. Elsevier, pp. 311–348.

Booth, K. S. and G. S. Lueker (1976). "Testing for the consecutive ones property, interval graphs, and graph planarity using PQ-tree algorithms." In: *Journal of Computer and System Sciences* 13.3, pp. 335–379.

Bourque, G. and P. A. Pevzner (2002). "Genome-scale evolution: reconstructing gene orders in the ancestral species." In: *Genome Research* 12.1, pp. 26–36.

Bourque, G., Y. Yacef, and N. El-Mabrouk (2005). "Maximizing synteny blocks to identify ancestral homologs." In: *Proc. 3rd Int'l Workshop on Comparative Genomics (RECOMB-CG 2005)*. Springer, pp. 21–34.

Bouvel, M. and E. Pergola (2010). "Posets and permutations in the duplication-loss model: Minimal permutations with d descents." In: *Theoretical Computer Science* 411.26-28, pp. 2487–2501.

Bouvel, M. and D. Rossin (2009). "A variant of the tandem duplication-random loss model of genome rearrangement." In: *Theoretical Computer Science* 410.8-10, pp. 847–858.

Bouvel, M., C. Chauve, M. Mishna, and D. Rossin (2011). "Average-case analysis of perfect sorting by reversals." In: *Discrete Mathematics, Algorithms and Applications* 3.03, pp. 369–392.

Braga, M. D. V., M.-F. Sagot, C. Scornavacca, and E. Tannier (2008). "Exploring the solution space of sorting by reversals, with experiments and an application to evolution." In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 5.3, pp. 348–356.

Breton, S., D. T. Stewart, S. Shepardson, R. J. Trdan, A. E. Bogan, E. G. Chapman, A. J. Ruminas, H. Piontkivska, and W. R. Hoeh (2010). "Novel protein genes in animal mtDNA: a new sex determination system in freshwater mussels (Bivalvia: Unionoida)?" In: *Molecular Biology and Evolution* 28.5, pp. 1645–1659.

Brito, K. L., A. R. Oliveira, U. Dias, and Z. Dias (2018). "Heuristics for the Sorting Signed Permutations by Reversals and Transpositions Problem." In: *Proc. 5th Int'l Conference on Algorithms for Computational Biology (AlCoB 2018)*. Springer, pp. 65–75.

Brocchieri, L. (2001). "Phylogenetic inferences from molecular sequences: review and critique." In: *Theoretical Population Biology* 59.1, pp. 27–40.

Brown, W. M. (1985). "The mitochondrial genome of animals." In: *Molecular Evolutionary Genetics*.

Bryant, D. (2000). "The complexity of calculating exemplar distances." In: *Comparative Genomics*. Springer, pp. 207–211.

Bui-Xuan, B.-M., M. Habib, and C. Paul (2005). "Revisiting T. Uno and M. Yagiura's algorithm." In: *Proc. 14th Int'l Symposium on Algorithms and Computation (ISAAC 2006)*. Springer, pp. 146–155.

Bulteau, L., G. Fertin, and I. Rusu (2012). "Sorting by transpositions is difficult." In: *SIAM Journal on Discrete Mathematics* 26.3, pp. 1148–1180.

Burke, E. K., P. De Causmaecker, G. V. Berghe, and H. Van Landeghem (2004). "The state of the art of nurse rostering." In: *Journal of Scheduling* 7.6, pp. 441–499.

Caprara, A (1997a). "Formulations and complexity of multiple sorting by reversals." In: *Proc. 3rd Int'l Conference on Computational Molecular Biology (RECOMB 1999)*, pp. 84–93.

Caprara, A. (1997b). "Sorting by reversals is difficult." In: *Proc. 1st Int'l Conference on Computational Molecular Biology (RECOMB 1997)*. ACM, pp. 75–83.

Caprara, A. and R. Rizzi (2002). "Improved approximation for breakpoint graph decomposition and sorting by reversals." In: *Journal of Combinatorial Optimization* 6.2, pp. 157–182.

Caprara, A., G. Lancia, and S. K. Ng (2000). "Fast practical solution of sorting by reversals." In: *Proc. 11th ACM-SIAM Symposium on Discrete Algorithms (SODA 2000)*. Society for Industrial and Applied Mathematics, pp. 12–21.

Caprara, G., A. Lancia, and S.-K. Ng (1999). "A column-generation based branch-and-bound algorithm for sorting by reversals." In: *Mathematical Support for Molecular Biology* 47, p. 213.

Cardazzo, B., S. Minuzzo, G. Sartori, A. Grapputo, and G. Carignani (1998). "Evolution of mitochondrial DNA in yeast: gene order and structural organization of the mitochondrial genome of Saccharomyces uvarum." In: *Current Genetics* 33.1, pp. 52–59.

Castellana, S., S. Vicario, and C. Saccone (2011). "Evolutionary patterns of the mitochondrial genome in Metazoa: exploring the role of mutation and selection in mitochondrial protein-coding genes." In: *Genome Biology and Evolution* 3, pp. 1067–1079.

Chan, D. C. (2006). "Mitochondria: dynamic organelles in disease, aging, and development." In: *Cell* 125.7, pp. 1241–1252.

Chaudhuri, K., K. Chen, R. Mihaescu, and S. Rao (2006). "On the tandem duplication-random loss model of genome rearrangement." In: *Proc. 17th ACM-SIAM Symposium on Discrete Algorithms (SODA 2006)*. Society for Industrial and Applied Mathematics, pp. 564–570.

Chen, T. and S. S. Skiena (1996). "Sorting with fixed-length reversals." In: *Discrete Applied Mathematics* 71.1-3, pp. 269–295.

Chen, X., J. Zheng, Z. Fu, P. Nan, Y. Zhong, S. Lonardi, and T. Jiang (2005). "Assignment of orthologous genes via genome rearrangement." In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 2.4, pp. 302–315.

Cheng, Y.-C., T. Hartmann, P.-Y. Tsai, and M. Middendorf (2016). "Population based ant colony optimization for reconstructing ECG signals." In: *Evolutionary Intelligence* 9.3, pp. 55–66.

Christie, D. A. (1998a). "A 3/2-Approximation Algorithm for Sorting by Reversals." In: *Proc. 9th ACM-SIAM Symposium on Discrete Algorithms (SODA 1998)*, pp. 244–252.

— (1998b). "Genome rearrangement problems." PhD thesis. University of Glasgow.

Clayton, D. A. (1982). "Replication of animal mitochondrial DNA." In: *Cell* 28.4, pp. 693–705.

— (1991). "Replication and transcription of vertebrate mitochondrial DNA." In: *Annual Review of Cell Biology* 7.1, pp. 453–478.

Cosner, M. E., R. K. Jansen, J. D. Palmer, and S. R. Downie (1997). "The highly rearranged chloroplast genome of Trachelium caeruleum (Campanulaceae): multiple inversions, inverted repeat expansion and contraction, transposition, insertions/deletions, and several repeat families." In: *Current Genetics* 31.5, pp. 419–429.

Darwin, C. (1859). *On the Origin of Species by Means of Natural Selection, or the Preservation of Favored Races in the Struggle for Life*. Murray.

De Giorgi, C., A. Martiradonna, C. Lanave, and C. Saccone (1996). "Complete sequence of the mitochondrial DNA in the sea urchin Arbacia lixula: conserved features of the Echinoid mitochondrial genome." In: *Molecular Phylogenetics and Evolution* 5.2, pp. 323–332.

Dias, U. and Z. Dias (2013). "Heuristics for the transposition distance problem." In: *Journal of Bioinformatics and Computational Biology* 11.05, p. 1350013.

Dias, Z. and C. C. de Souza (2007). "Polynomial-sized ILP models for rearrangement distance problems." In: *Proc. 2nd Brazilian Symposium on Bioinformatics (BSB 2007)*, pp. 74–85.

Didier, G. (2003). "Common intervals of two sequences." In: *Proc. 3rd Int'l Workshop on Algorithms in Bioinformatics (WABI 2003)*. Springer, pp. 17–24.

Dobzhansky, T. and A. H. Sturtevant (1938). "Inversions in the chromosomes of Drosophila pseudoobscura." In: *Genetics* 23.1, pp. 28–64.

Dörr, D. (2016). "Gene family-free genome comparison." PhD thesis. Bielefeld University.

Dowton, M., S. L. Cameron, J. I. Dowavic, A. D. Austin, and M. F. Whiting (2009). "Characterization of 67 mitochondrial tRNA gene rearrangements in the Hymenoptera suggests that mitochondrial tRNA gene position is selectively neutral." In: *Molecular Biology and Evolution* 26.7, pp. 1607–1617.

Drake, J. W. and R. H. Baltz (1976). "The biochemistry of mutagenesis." In: *Annual Review of Biochemistry* 45.1, pp. 11–37.

Dwork, C., R. Kumar, M. Naor, and D. Sivakumar (2001). "Rank aggregation methods for the web." In: *Proc. 10th Int'l Conference on World Wide Web (WWW 2001)*. ACM, pp. 613–622.

ENCODE Project Consortium (2012). "An integrated encyclopedia of DNA elements in the human genome." In: *Nature* 489.7414, p. 57.

El-Mabrouk, N. and D. Sankoff (2012). "Analysis of gene order evolution beyond single-copy genes." In: *Evolutionary Genomics*. Springer, pp. 397–429.

Elias, I. and T. Hartman (2006). "A 1.375-approximation algorithm for sorting by transpositions." In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 3.4.

Eriksen, N. (2001). "(1+ε)-Approximation of Sorting by Reversals and Transpositions." In: *Proc. 1st Int'l Workshop on Algorithms in Bioinformatics (WABI 2001)*. Springer, pp. 227–237.

Eriksson, H., J.and Svensson L. Eriksson K.and Karlander, and J. Wästlund (2001). "Sorting a bridge hand." In: *Discrete Mathematics* 241.1-3, pp. 289–300.

Ezkurdia, I., D. Juan, J. M. Rodriguez, A. Frankish, M. Diekhans, J. Harrow, J. Vazquez, A. Valencia, and M. L. Tress (2014). "Multiple evidence strands suggest that there may be as few as 19000 human protein-coding genes." In: *Human Molecular Genetics* 23.22, pp. 5866–5878.

Feijao, P. and J. Meidanis (2011). "SCJ: a breakpoint-like distance that simplifies several rearrangement problems." In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 8.5, pp. 1318–1329.

Feijão, P., A. Mane, and C. Chauve (2017). "A Tractable Variant of the Single Cut or Join Distance with Duplicated Genes." In: *Proc. 15th Int'l Workshop on Comparative Genomics (RECOMB-CG 2017)*. Springer, pp. 14–30.

Felsenstein, J. (2004). *Inferring phylogenies*. Vol. 2. Sinauer associates Sunderland.

Feng, J. and D. Zhu (2007). "Faster algorithms for sorting by transpositions and sorting by block interchanges." In: *ACM Transactions on Algorithms* 3.3, p. 25.

Fertin, G., A. Labarre, I. Rusu, E. Tannier, and S. Vialette (2009). *Combinatorics of genome rearrangements*. MIT press.

Figeac, M. and J.-S. Varré (2004). "Sorting by reversals with common intervals." In: *Proc. 4th Int'l Workshop on Algorithms in Bioinformatics (WABI 2004)*. Springer, pp. 26–37.

Fonseca, M. M. and D. J. Harris (2008). "Relationship between mitochondrial gene rearrangements and stability of the origin of light strand replication." In: *Genetics and Molecular Biology* 31.2, pp. 566–574.

Galperin, M. Y. and E. V. Koonin (2000). "Who's your neighbor? New computational approaches for functional genomics." In: *Nature Biotechnology* 18.6, p. 609.

Galvão, G. R., O. Lee, and Z. Dias (2015). "Sorting signed permutations by short operations." In: *Algorithms for Molecular Biology* 10.1, p. 12.

Galvao, G. R., C. Baudet, and Z. Dias (2017). "Sorting circular permutations by super short reversals." In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 14.3, pp. 620–633.

Gan, H. M., M. H. Tan, Y. P. Lee, M. B. Schultz, P. Horwitz, Q. Burnham, and C. M. Austin (2018). "More evolution underground: Accelerated mitochondrial substitution rate in Australian burrowing freshwater crayfishes (Decapoda: Parastacidae)." In: *Molecular Phylogenetics and Evolution* 118, pp. 88–98.

Gerstein, M. B., C. Bruce, J. S. Rozowsky, D. Zheng, J. Du, J. O. Korbel, O. Emanuelsson, Z. D. Zhang, S. Weissman, and M. Snyder (2007). "What is a gene, post-ENCODE? History and updated definition." In: *Genome Research* 17.6, pp. 669–681.

Gissi, C., F. Iannelli, and G. Pesole (2008). "Evolution of the mitochondrial genome of Metazoa as exemplified by comparison of congeneric species." In: *Heredity* 101.4, p. 301.

Graham, G. J. (1995). "Tandem genes and clustered genes." In: *Journal of Theoretical Biology* 175.1, pp. 71–87.

Granlund, T. and the GMP development team (2012). *GNU MP: The GNU Multiple Precision Arithmetic Library*. 5.0.5. http://gmplib.org/.

Gray, F. (1953). *Pulse code communication*. US Patent 2,632,058.

Gray, M. W. (1989). "Origin and evolution of mitochondrial DNA." In: *Annual Review of Cell Biology* 5.1, pp. 25–50.

Gredilla, Ricardo (2011). "DNA damage and base excision repair in mitochondria and their role in aging." In: *Journal of Aging Research* 2011.

Gu, Q.-P., S. Peng, and H. Sudborough (1999). "A 2-approximation algorithm for genome rearrangements by reversals and transpositions." In: *Theoretical Computer Science* 210.2, pp. 327–339.

Gurobi Optimization, LLC (2018). *Gurobi Optimizer Reference Manual*. URL: http://www.gurobi.com.

Gusfield, D. (1997). *Algorithms on strings, trees and sequences: computer science and computational biology*. Cambridge University Press.

Hannenhalli, S. and P. A. Pevzner (1995). "Transforming men into mice (polynomial algorithm for genomic distance problem)." In: *Proc. 36th Symposium on Foundations of Computer Science (FOCS 1995)*. IEEE, pp. 581–592.

— (1999). "Transforming cabbage into turnip: polynomial algorithm for sorting signed permutations by reversals." In: *Journal of the ACM* 46.1, pp. 1–27.

Hao, F.-C., M. Zhang, and H. W. Leong (2017). "A 2-Approximation Scheme for Sorting Signed Permutations by Reversals, Transpositions, Transreversals, and Block-Interchanges." In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics*.

Hardison, R. C. (2003). "Comparative genomics." In: *PLoS Biology* 1.2, e58.

Härlid, A., A. Janke, and U. Arnason (1997). "The mtDNA sequence of the ostrich and the divergence between paleognathous and neognathous birds." In: *Molecular Biology and Evolution* 14.7, pp. 754–761.

Harman, D. (1972). "The biologic clock: the mitochondria?" In: *Journal of the American Geriatrics Society* 20.4, pp. 145–147.

Hartman, T. (2003). "A simpler 1.5-approximation algorithm for sorting by transpositions." In: *Proc. 14th Symposium on Combinatorial Pattern Matching (CPM 2003)*. Springer, pp. 156–169.

Hartman, T. and R. Shamir (2006). "A simpler and faster 1.5-approximation algorithm for sorting by transpositions." In: *Information and Computation* 204.2, pp. 275–290.

Hartman, T. and R. Sharan (2005). "A 1.5-approximation algorithm for sorting by transpositions and transreversals." In: *Journal of Computer and System Sciences* 70.3, pp. 300–320.

Hartmann, T., N. Wieseke, R. Sharan, M. Middendorf, and M. Bernt (2017). "Genome Rearrangement with ILP." In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 15.5, pp. 1585–1593.

Hartmann, T., M. Bernt, and M. Middendorf (2018a). "An Exact Algorithm for Sorting by Weighted Preserving Genome Rearrangements." In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics*. (in press).

Hartmann, T., A.-C. Chu, M. Middendorf, and M. Bernt (2018b). "Combinatorics of tandem duplication random loss mutations on

circular genomes." In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 15.1, pp. 83–95.

Hartmann, T., M. Bernt, and M. Middendorf (2018c). "EqualTDRL: illustrating equivalent tandem duplication random loss rearrangements." In: *BMC Bioinformatics* 19.192.

Hartmann, T., M. Middendorf, and M. Bernt (2018d). "Genome Rearrangement Analysis: Cut and Join Genome Rearrangements and Gene Cluster Preserving Approaches." In: *Comparative Genomics: Methods and Protocols*. Springer, pp. 261–289.

Hartmann, T., M. Bannach, and M. Middendorf (2018e). "Sorting Signed Permutations by Inverse Tandem Duplication Random Losses." In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics*. (in press).

Hazkani-Covo, E. and D. Graur (2006). "A comparative analysis of numt evolution in human and chimpanzee." In: *Molecular Biology and Evolution* 24.1, pp. 13–18.

He, Y. and T. Chen (2003). "A new approximation algorithm for sorting of signed permutations." In: *Journal of Computer Science and Technology* 18.1, pp. 125–130.

Heber, S. and J. Stoye (2001a). "Algorithms for finding gene clusters." In: *Proc. 1st Int'l Workshop on Algorithms in Bioinformatics (WABI 2001)*. Springer, pp. 252–263.

— (2001b). "Finding all common intervals of k permutations." In: *Proc. 12th Symposium on Combinatorial Pattern Matching (CPM 2001)*. Vol. 2089. LNCS, pp. 207–218.

Heber, S., R. Mayr, and J. Stoye (2011). "Common intervals of multiple permutations." In: *Algorithmica* 60.2, pp. 175–206.

Held, M. and R. M. Karp (1970). "The traveling-salesman problem and minimum spanning trees." In: *Operations Research* 18.6, pp. 1138–1162.

Hoberman, R. and D. Durand (2005). "The incompatible desiderata of gene cluster properties." In: *Proc. 3rd Int'l Workshop on Comparative Genomics (RECOMB-CG 2005)*. Springer, pp. 73–87.

Hoffmann, R. J., J. L. Boore, and W. M. Brown (1992). "A novel mitochondrial genome organization for the blue mussel, Mytilus edulis." In: *Genetics* 131.2, pp. 397–412.

Holt, I. J. (2009). "Mitochondrial DNA replication and repair: all a flap." In: *Trends in Biochemical Sciences* 34.7, pp. 358–365.

Holt, I. J., A. E. Harding, and J. A. Morgan-Hughes (1988). "Deletions of muscle mitochondrial DNA in patients with mitochondrial myopathies." In: *Nature* 331.6158, p. 717.

Holt, I. J., H. E. Lorimer, and H. T. Jacobs (2000). "Coupled leading-and lagging-strand synthesis of mammalian mitochondrial DNA." In: *Cell* 100.5, pp. 515–524.

Huang, X. C., J. Rong, Y. Liu, M. H. Zhang, Y. Wan, S. Ouyang, C. H. Zhou, and X. P. Wu (2013). "The complete maternally and paternally inherited mitochondrial genomes of the endangered freshwater mussel *Solenaia carinatus* (Bivalvia: Unionidae) and implications for Unionidae taxonomy." In: *PLoS ONE* 8.12, e84352.

Huynen, M., B. Snel, W. Lathe 3rd, and P. Bork (2000). "Predicting protein function by genomic context: quantitative evaluation and qualitative inferences." In: *Genome Research* 10.8, pp. 1204–1210.

Iannelli, F., F. Griggio, G. Pesole, and C. Gissi (2007). "The mitochondrial genome of Phallusia mammillata and Phallusia fumigata (Tunicata, Ascidiacea): high genome plasticity at intra-genus level." In: *BMC Evolutionary Biology* 7.1, p. 155.

Inoue, J. G., M. Miya, K. Tsukamoto, and M. Nishida (2003). "Evolution of the Deep-Sea Gulper Eel Mitochondrial Genomes: Large-Scale Gene Rearrangements Originated Within the Eels." In: *Molecular Biology and Evolution* 20.11, pp. 1917–1924.

Iwasaki, W., T. Fukunaga, R. Isagozawa, K. Yamada, Y. Maeda, T. P. Satoh, T. Sado, K. Mabuchi, H. Takeshima, M. Miya, and M. Nishida (2013). "MitoFish and MitoAnnotator: A mitochondrial genome database of fish with an accurate and automatic annotation pipeline." In: *Molecular Biology and Evolution* 30.11, pp. 2531–2540.

Janke, A., G. Feldmaier-Fuchs, W. K. Thomas, A. Von Haeseler, and S. Pääbo (1994). "The marsupial mitochondrial genome and the evolution of placental mammals." In: *Genetics* 137.1, pp. 243–256.

Jerrum, M. R. (1985). "The complexity of finding minimum-length generator sequences." In: *Theoretical Computer Science* 36, pp. 265–289.

Jiang, S. and M. A. Alekseyev (2011). "Weighted Genomic Distance Can Hardly Impose a Bound on the Proportion of Transpositions." In: *Proc. 15th Int'l Conference on Computational Molecular Biology (RECOMB 2011)*, pp. 124–133.

Jones, N. C. and P. A. Pevzner (2004). *An introduction to bioinformatics algorithms*. MIT press.

Jühling, F., J. Pütz, M. Bernt, A. Donath, M. Middendorf, C. Florentz, and P. F. Stadler (2011). "Improved systematic tRNA gene annotation allows new insights into the evolution of mitochondrial tRNA structures and into the mechanisms of mitochondrial genome rearrangements." In: *Nucleic Acids Research* 40.7, pp. 2833–2845.

Kaplan, H. and E. Verbin (2003). "Efficient data structures and a new randomized approach for sorting signed permutations by reversals." In: *Proc. 14th Symposium on Combinatorial Pattern Matching (CPM 2003)*. Springer, pp. 170–185.

Kaplan, H., R. Shamir, and R. E. Tarjan (2000). "A faster and simpler algorithm for sorting signed permutations by reversals." In: *SIAM Journal on Computing* 29.3, pp. 880–892.

Kayal, E., B. Bentlage, A. G. Collins, M. Kayal, S. Pirro, and D. V. Lavrov (2011). "Evolution of linear mitochondrial genomes in medusozoan cnidarians." In: *Genome Biology and Evolution* 4.1, pp. 1–12.

Kayal, E., B. Bentlage, P. Cartwright, A. A. Yanagihara, D. J. Lindsay, R. R. Hopcroft, and A. G. Collins (2015). "Phylogenetic analysis of higher-level relationships within Hydroidolina (Cnidaria: Hy-

drozoa) using mitochondrial genome data and insight into their mitochondrial transcription." In: *PeerJ* 3, e1403.

Kececioglu, J. and D. Sankoff (1995). "Exact and approximation algorithms for sorting by reversals, with application to genome rearrangement." In: *Algorithmica* 13.1-2, p. 180.

Knuth, D. E. (1997). *The art of computer programming: sorting and searching*. Vol. 3. Pearson Education.

Kong, X., X. Dong, Y. Zhang, W. Shi, Z. Wang, and Z. Yu (2009). "A novel rearrangement in the mitochondrial genome of tongue sole, Cynoglossus semilaevis: control region translocation and a tRNA gene inversion." In: *Genome* 52.12, pp. 975–984.

Krebs, J. E., B. Lewin, E. S. Goldstein, and S. T. Kilpatrick (2014). *Lewin's Genes XI*. Jones & Bartlett Publishers.

Labarre, A. (2006). "New bounds and tractable instances for the transposition distance." In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 3.4.

Lancia, G., F. Rinaldi, and P. Serafini (2015). "A unified integer programming model for genome rearrangement problems." In: *Proc. 3rd Int'l Conference on Bioinformatics and Biomedical Engineering (IWBBIO 2015)*. Springer, pp. 491–502.

Lang, B. F., G. Burger, C. J. O'kelly, R. Cedergren, G. B. Golding, C. Lemieux, D. Sankoff, M. Turmel, and M. W. Gray (1997). "An ancestral mitochondrial DNA resembling a eubacterial genome in miniature." In: *Nature* 387.6632, pp. 493–497.

Lang, B. F., M. W. Gray, and G. Burger (1999). "Mitochondrial genome evolution and the origin of eukaryotes." In: *Annual Review of Genetics* 33.1, pp. 351–397.

Lathe 3rd, W. C., B. Snel, and P. Bork (2000). "Gene context conservation of a higher order than operons." In: *Trends in Biochemical Sciences* 25.10, pp. 474–479.

Lavrov, D. V. and W. Pett (2016). "Animal mitochondrial DNA as we do not know it: mt-genome organization and evolution in nonbilaterian lineages." In: *Genome Biology and Evolution* 8.9, pp. 2896–2913.

Lavrov, D. V., J. L. Boore, and W. M. Brown (2002). "Complete mtDNA sequences of two millipedes suggest a new model for mitochondrial gene rearrangements: duplication and nonrandom loss." In: *Molecular Biology and Evolution* 19.2, pp. 163–169.

Lefebvre, J.-F., N. El-Mabrouk, E. Tillier, and D. Sankoff (2003). "Detection and validation of single gene inversions." In: *Bioinformatics* 19.suppl_1, pp. i190–i196.

Levinson, G. and G. A. Gutman (1987). "Slipped-strand mispairing: a major mechanism for DNA sequence evolution." In: *Molecular Biology and Evolution* 4.3, pp. 203–221.

Li, G. and D. Reinberg (2011). "Chromatin higher-order structures and gene regulation." In: *Current Opinion in Genetics & Development* 21.2, pp. 175–186.

Lima, T. A. de and M. Ayala-Rincon (2018). "On the average number of reversals needed to sort signed permutations." In: *Discrete Applied Mathematics* 235, pp. 59–80.

Lin, G.-H. and G. Xue (2001). "Signed genome rearrangement by reversals and transpositions: models and approximations." In: *Theoretical Computer Science* 259.1-2, pp. 513–531.

Lin, G. and T. Jiang (2004). "A further improved approximation algorithm for breakpoint graph decomposition." In: *Journal of Combinatorial Optimization* 8.2, pp. 183–194.

Liu, X., H. Li, Y. Cai, F. Song, J.-J. Wilson, and W. Cai (2017). "Conserved gene arrangement in the mitochondrial genomes of barklouse families Stenopsocidae and Psocidae." In: *Frontiers of Agricultural Science and Engineering* 4.3, pp. 358–365.

Lohse, M., O. Drechsel, and R. Bock (2007). "OrganellarGenomeDRAW (OGDRAW): a tool for the easy generation of high-quality custom graphical maps of plastid and mitochondrial genomes." In: *Current Genetics* 52.5-6, pp. 267–274.

Lou, X.-W. and D.-M. Zhu (2010). "Sorting unsigned permutations by weighted reversals, transpositions, and transreversals." In: *Journal of Computer Science and Technology* 25.4, pp. 853–863.

Luc, N., J.-L. Risler, A. Bergeron, and M. Raffinot (2003). "Gene teams: a new formalization of gene clusters for comparative genomics." In: *Computational Biology and Chemistry* 27.1, pp. 59–67.

Lunt, D. H. and B. C. Hyman (1997). "Animal mitochondrial DNA recombination." In: *Nature* 387.6630, p. 247.

Macey, J. R., A. Larson, N. B. Ananjeva, Z. Fang, and T. J. Papenfuss (1997). "Two novel gene orders and the role of light-strand replication in rearrangement of the vertebrate mitochondrial genome." In: *Molecular Biology and Evolution* 14.1, pp. 91–104.

Macey, J. R., J. A. Schulte, A. Larson, and T. J. Papenfuss (1998). "Tandem duplication via light-strand synthesis may provide a precursor for mitochondrial genomic rearrangement." In: *Molecular Biology and Evolution* 15.1, pp. 71–75.

Maddison, W. P. (1997). "Gene trees in species trees." In: *Systematic Biology* 46.3, pp. 523–536.

Mao, M., A. D. Austin, N. F. Johnson, and M. Dowton (2013). "Coexistence of minicircular and a highly rearranged mtDNA molecule suggests that recombination shapes mitochondrial genome organization." In: *Molecular Biology and Evolution* 31.3, pp. 636–644.

Martijn, J., J. Vosseberg, L. Guy, P. Offre, and T. J.-G. Ettema (2018). "Deep mitochondrial origin outside the sampled alphaproteobacteria." In: *Nature* 557.7703, p. 101.

Martin, W. and R. G. Herrmann (1998). "Gene transfer from organelles to the nucleus: how much, what happens, and why?" In: *Plant Physiology* 118.1, pp. 9–17.

McBride, H. M., M. Neuspiel, and S. Wasiak (2006). "Mitochondria: more than just a powerhouse." In: *Current Biology* 16.14, R551–R560.

Meidanis, J., M. E. M. T. Walter, and Z. Dias (2000). *Reversal distance of signed circular chromosomes*. Relatório Técnico IC-00-23. University of Campinas, Brazil.

Meo, P. D'Onorio de, M. D'Antonio, F. Griggio, R. Lupi, M. Borsani, G. Pavesi, T. Castrignano, G. Pesole, and C. Gissi (2012). "MitoZoa

2.0: A database resource and search tools for comparative and evolutionary analyses of mitochondrial genomes in Metazoa." In: *Nucleic Acids Research* 40.Database issue, pp. D1168–1172.

Mering, C. von, M. Huynen, D. Jaeggi, S. Schmidt, P. Bork, and B. Snel (2003). "STRING: a database of predicted functional associations between proteins." In: *Nucleic Acids Research* 31.1, pp. 258–261.

Miklós, I. and J. Hein (2004). "Genome rearrangement in mitochondria and its computational biology." In: *Proc. 2nd Int'l Workshop on Comparative Genomics (RECOMB-CG 2004)*. Springer, pp. 85–96.

Moret, B. M. E. and T. Warnow (2005). "Advances in phylogeny reconstruction from gene order and content data." In: *Methods in Enzymology*. Vol. 395. Elsevier, pp. 673–700.

Moret, B. M. E., L.-S. Wang, T. Warnow, and S. K. Wyman (2001). "New approaches for reconstructing phylogenies from gene order data." In: *Bioinformatics* 17.suppl_1, S165–S173.

Moret, B. M. E., U. Roshan, and T. Warnow (2002). "Sequence-length requirements for phylogenetic methods." In: *Proc. 2nd Int'l Workshop on Algorithms in Bioinformatics (WABI 2002)*. Springer, pp. 343–356.

Morrison, C.-L., A.-W. Harvey, S. Lavery, K. Tieu, Y. Huang, and C.-W. Cunningham (2002). "Mitochondrial gene rearrangements confirm the parallel evolution of the crab-like form." In: *Proceedings of the Royal Society of London B: Biological Sciences* 269.1489, pp. 345–350.

Mueller, R. L. and J. L. Boore (2005). "Molecular mechanisms of extensive mitochondrial gene rearrangement in plethodontid salamanders." In: *Molecular Biology and Evolution* 22.10, pp. 2104–2112.

Nakhleh, L., B. M. E. Moret, U. Roshan, K. St. John, J. Sun, and T. Warnow (2001). "The accuracy of fast phylogenetic methods for large datasets." In: *Proc. 7th Pacific Symposium on Biocomputing (PSB 2002)*. World Scientific, pp. 211–222.

Nasir, A., K. M. Kim, and G. Caetano-Anolles (2012). "Giant viruses coexisted with the cellular ancestors and represent a distinct supergroup along with superkingdoms Archaea, Bacteria and Eukarya." In: *BMC Evolutionary Biology* 12.1, p. 156.

Oliveira, A. R., K. L. Brito, Z. Dias, and U. Dias (2018a). "Sorting by Weighted Reversals and Transpositions." In: *Proc. 11th Brazilian Symposium on Bioinformatics (BSB 2018)*. Springer, pp. 38–49.

Oliveira, A. R., G. Fertin, U. Dias, and Z. Dias (2018b). "Sorting signed circular permutations by super short operations." In: *Algorithms for Molecular Biology* 13.1, p. 13.

Ouangraoua, A., A. Bergeron, and K. M. Swenson (2010). "Ultra-perfect sorting scenarios." In: *Proc. 8th Int'l Workshop on Comparative Genomics (RECOMB-CG 2010)*. Springer, pp. 50–61.

Ouangraoua, A., E. Tannier, and C. Chauve (2011a). "Reconstructing the architecture of the ancestral amniote genome." In: *Bioinformatics* 27.19, pp. 2664–2671.

Ouangraoua, A., A. Bergeron, and K. M. Swenson (2011b). "Theory and practice of ultra-perfection." In: *Journal of Computational Biology* 18.9, pp. 1219–1230.

Overbeek, R., M. Fonstein, M. D'souza, G. D. Pusch, and N. Maltsev (1999). "The use of gene clusters to infer functional coupling." In: *Proceedings of the National Academy of Sciences* 96.6, pp. 2896–2901.

Oxusoff, L., P. Prea, and Y. Perez (2018). "A complete logical approach to resolve the evolution and dynamics of mitochondrial genome in bilaterians." In: *PloS One* 13.3, e0194334.

Palmer, J. D. and L. A. Herbon (1988). "Plant mitochondrial DNA evolved rapidly in structure, but slowly in sequence." In: *Journal of Molecular Evolution* 28.1-2, pp. 87–97.

Parida, L. (2006). "A PQ Framework for Reconstructions of Common Ancestors & Phylogeny." In: *Proc. 10th Int'l Workshop on Comparative Genomics (RECOMB-CG 2006)*. Springer, pp. 141–55.

Pelletier, L. and I. Rusu (2018). "Common intervals and permutation reconstruction from MinMax-betweenness constraints." In: *Journal of Discrete Algorithms* 49, pp. 8–26.

Perrin, A., J.-S. Varré, S. Blanquart, and A. Ouangraoua (2015). "Procars: Progressive reconstruction of ancestral gene orders." In: *BMC Genomics* 16.5, S6.

Perseke, M., G. Fritzsch, K. Ramsch, M. Bernt, D. Merkle, M. Middendorf, D. Bernhard, P. F. Stadler, and M. Schlegel (2008). "Evolution of mitochondrial gene orders in echinoderms." In: *Molecular Phylogenetics and Evolution* 47.2, pp. 855–864.

Pevzner, P. (2000). *Computational molecular biology: an algorithmic approach*. MIT press.

Podsiadlowski, L., A. Braband, T. H. Struck, J. von Döhren, and T. Bartolomaeus (2009). "Phylogeny and mitochondrial gene order variation in Lophotrochozoa in the light of new mitogenomic data from Nemertea." In: *BMC Genomics* 10.1, p. 364.

Pohjoismäki, J. L.-O. and S. Goffart (2011). "Of circles, forks and humanity: topological organisation and replication of mammalian mitochondrial DNA." In: *Bioessays* 33.4, pp. 290–299.

Pohjoismäki, J. L.-O., S. Goffart, R. W. Taylor, D. M. Turnbull, A. Suomalainen, H. T. Jacobs, and P. J. Karhunen (2010). "Developmental and pathological changes in the human cardiac muscle mitochondrial DNA organization, replication and copy number." In: *PLoS One* 5.5, e10426.

Pruitt, K. D., T. Tatusova, and D. R. Maglott (2007). "NCBI reference sequences (RefSeq): A curated non-redundant sequence database of genomes, transcripts and proteins." In: *Nucleic Acids Research* 35.Database issue, pp. D61–5.

Raimond, R., I. Marcadé, D. Bouchon, T. Rigaud, J.-P. Bossy, and C. Souty-Grosset (1999). "Organization of the large mitochondrial genome in the isopod Armadillidium vulgare." In: *Genetics* 151.1, pp. 203–210.

Rajan, V., A. W. Xu, Y. Lin, K. M. Swenson, and B. M. E. Moret (2010). "Heuristics for the inversion median problem." In: *BMC Bioinformatics* 11.1, S30.

Reyes, A., C. Gissi, G. Pesole, and C. Saccone (1998). "Asymmetrical directional mutation pressure in the mitochondrial genome of mammals." In: *Molecular Biology and Evolution* 15.8, pp. 957–966.

Robberson, D. L., H. Kasamatsu, and J. Vinograd (1972). "Replication of mitochondrial DNA. Circular replicative intermediates in mouse L cells." In: *Proceedings of the National Academy of Sciences* 69.3, p. 737.

Rodgers, K. and M. McVey (2016). "Error-prone repair of DNA double-strand breaks." In: *Journal of Cellular Physiology* 231.1, pp. 15–24.

Rokas, A. and P. W.H. Holland (2000). "Rare genomic changes as a tool for phylogenetics." In: *Trends in Ecology & Evolution* 15.11, pp. 454–459.

Rosenberg, M. S. (2009). *Sequence alignment: methods, models, concepts, and strategies*. Univ of California Press.

Rot, C., I. Goldfarb, M. Ilan, and D. Huchon (2006). "Putative cross-kingdom horizontal gene transfer in sponge (Porifera) mitochondria." In: *BMC Evolutionary Biology* 6.1, p. 71.

Rusu, I. (2014a). "Extending common intervals searching from permutations to sequences." In: *Journal of Discrete Algorithms* 29, pp. 27–46.

— (2014b). "MinMax-Profiles: A unifying view of common intervals, nested common intervals and conserved intervals of K permutations." In: *Theoretical Computer Science* 543, pp. 90–111.

— (2016). "Permutation reconstruction from MinMax-Betweenness constraints." In: *Discrete Applied Mathematics* 207, pp. 106–119.

Sagot, M.-F. and E. Tannier (2005). "Perfect sorting by reversals." In: *Proc. 11th Int'l Computing and Combinatorics Conference (COCOON 2005)*. Springer, pp. 42–51.

San Mauro, D., D. J. Gower, R. Zardoya, and M. Wilkinson (2005). "A hotspot of gene order rearrangement by tandem duplication and random loss in the vertebrate mitochondrial genome." In: *Molecular Biology and Evolution* 23.1, pp. 227–234.

Sankoff, D. (1992). "Edit distance for genome comparison based on non-local operations." In: *Proc. 3rd Symposium on Combinatorial Pattern Matching (CPM 1992)*. Springer, pp. 121–135.

— (2002). "Short inversions and conserved gene clusters." In: *Proc. 17th ACM/SIGAPP Symposium On Applied Computing (SAC 2002)*. ACM, pp. 164–167.

Sankoff, D. and M. Blanchette (1997). "The median problem for breakpoints in comparative genomics." In: *Proc. 3rd Int'l Computing and Combinatorics Conference (COCOON 1997)*. Springer, pp. 251–263.

Sankoff, D., G. Leduc, N. Antoine, B. Paquin, B. F. Lang, and R. Cedergren (1992). "Gene order comparisons for phylogenetic inference: Evolution of the mitochondrial genome." In: *Proceedings of the National Academy of Sciences* 89.14, pp. 6575–6579.

Scheffler, I. E. (2011). *Mitochondria*. John Wiley & Sons.

Schmidt, T. and J. Stoye (2004). "Quadratic time algorithms for finding common intervals in two and more sequences." In: *Proc. 15th Symposium on Combinatorial Pattern Matching (CPM 2004)*. Springer, pp. 347–358.

Schrijver, A. (1998). *Theory of linear and integer programming*. John Wiley & Sons.

Scouras, A. and M. J. Smith (2001). "A novel mitochondrial gene order in the crinoid echinoderm Florometra serratissima." In: *Molecular Biology and Evolution* 18.1, pp. 61–73.

Sémon, M. and L. Duret (2006). "Evolutionary origin and maintenance of coexpressed gene clusters in mammals." In: *Molecular Biology and Evolution* 23.9, pp. 1715–1723.

Setubal, J. C. and J. Meidanis (1997). *Introduction to computational molecular biology*. 04; QH506, S4. PWS Pub.

Shadel, G. S. and D. A. Clayton (1997). "Mitochondrial DNA maintenance in vertebrates." In: *Annual Review of Biochemistry* 66.1, pp. 409–435.

Shao, M. and B. M. E. Moret (2017). "On computing breakpoint distances for genomes with duplicate genes." In: *Journal of Computational Biology* 24.6, pp. 571–580.

Shao, R., M. Dowton, A. Murrell, and S. C. Barker (2003). "Rates of gene rearrangement and nucleotide substitution are correlated in the mitochondrial genomes of insects." In: *Molecular Biology and Evolution* 20.10, pp. 1612–1619.

Shao, R., E. F. Kirkness, and S. C. Barker (2009). "The single mitochondrial chromosome typical of animals has evolved into 18 minichromosomes in the human body louse, Pediculus humanus." In: *Genome Research* 19.5, pp. 904–912.

Shi, W., X.-L. Dong, Z.-M. Wang, X.-G. Miao, S.-Y. Wang, and X.-Y. Kong (2013). "Complete mitogenome sequences of four flatfishes (Pleuronectiformes) reveal a novel gene arrangement of L-strand coding genes." In: *BMC Evolutionary Biology* 13.1, p. 173.

Siepel, A. C. (2003). "An algorithm to enumerate sorting reversals for signed permutations." In: *Journal of Computational Biology* 10.3-4, pp. 575–597.

Silva, F. A. M. da, A. R. Oliveira, and Z. Dias (2017). *Machine Learning Applied to Sorting Permutations by Reversals and Transpositions*. Relatório Técnico IC-PFG-17-03. University of Campinas, Brazil.

Silveira, L. Â. da, J. L. Soncco-Álvarez, and M. Ayala-Rincón (2017). "Parallel genetic algorithms with sharing of individuals for sorting unsigned genomes by reversals." In: *Proc. 10th IEEE Congress on Evolutionary Computation (CEC 2017)*. IEEE, pp. 741–748.

Simonaitis, P. and K. M. Swenson (2018). "Finding local genome rearrangements." In: *Algorithms for Molecular Biology* 13.1, p. 9.

Solomon, A., P. Sutcliffe, and R. Lister (2003). "Sorting circular permutations by reversal." In: *Proc. 9th Workshop on Algorithms and Data Structures (WADS 2003)*. Springer, pp. 319–328.

Stadler, P. F., S. J. Prohaska, C. V. Forst, and D. C. Krakauer (2009). "Defining genes: a computational framework." In: *Theory in Biosciences* 128.3, p. 165.

Sturtevant, A. H. and G. W. Beadle (1936). "The relations of inversions in the X chromosome of Drosophila melanogaster to crossing over and disjunction." In: *Genetics* 21.5, pp. 554–604.

Sturtevant, A. H. and T. Dobzhansky (1936a). "Geographical distribution and cytology of "sex ratio" in Drosophila pseudoobscura and related species." In: *Genetics* 21.4, pp. 473–490.

— (1936b). "Inversions in the third chromosome of wild races of Drosophila pseudoobscura, and their use in the study of the history of the species." In: *Proceedings of the National Academy of Sciences* 22.7, pp. 448–450.

Sturtevant, A. H. and E. Novitski (1941). "The homologies of the chromosome elements in the genus Drosophila." In: *Genetics* 26.5, pp. 517–541.

Swenson, K. M. and B. M. E. Moret (2009). "Inversion-based genomic signatures." In: *BMC Bioinformatics* 10.1, S7.

Swenson, K. M., Y. To, J. Tang, and B. M. E. Moret (2009). "Maximum independent sets of commuting and noninterfering inversions." In: *BMC Bioinformatics* 10.1, S6.

Swenson, K. M., V. Rajan, Y. Lin, and B. M. E. Moret (2010). "Sorting signed permutations by inversions in $\mathcal{O}(n \log n)$ time." In: *Journal of Computational Biology* 17.3, pp. 489–501.

Swenson, K. M., P. Simonaitis, and M. Blanchette (2016). "Models and algorithms for genome rearrangement with positional constraints." In: *Algorithms for Molecular Biology* 11.1, p. 13.

Tamames, J. (2001). "Evolution of gene order conservation in prokaryotes." In: *Genome Biology* 2.6, research0020–1.

Tamames, J., G. Casari, C. Ouzounis, and A. Valencia (1997). "Conserved clusters of functionally related genes in two bacterial genomes." In: *Journal of Molecular Evolution* 44.1, pp. 66–73.

Tan, M. H., H. M. Gan, Y. P. Lee, G. C.-B. Poore, and C. M. Austin (2017). "Digging deeper: new gene order rearrangements and distinct patterns of codons usage in mitochondrial genomes among shrimps from the Axiidea, Gebiidea and Caridea (Crustacea: Decapoda)." In: *PeerJ* 5, e2982.

Tan, M. H., H. M. Gan, Y. P. Lee, S. Linton, F. Grandjean, M. L. Bartholomei-santos, A. D. Miller, and C. M. Austin (2018). "ORDER within the chaos: Insights into phylogenetic relationships within the Anomura (Crustacea: Decapoda) from mitochondrial sequences and gene order rearrangements." In: *Molecular Phylogenetics and Evolution*.

Tannier, E. and M.-F. Sagot (2004). "Sorting by reversals in subquadratic time." In: *Proc. 15th Symposium on Combinatorial Pattern Matching (CPM 2004)*. Springer, pp. 1–13.

Tannier, E., A. Bergeron, and M.-F. Sagot (2007). "Advances on sorting by reversals." In: *Discrete Applied Mathematics* 155.6-7, pp. 881–888.

Thorsness, P. E. and T. D. Fox (1990). "Escape of DNA from mitochondria to the nucleus in Saccharomyces cerevisiae." In: *Nature* 346.6282, p. 376.

Thrash, J. C., A. Boyd, M. J. Huggett, J. Grote, P. Carini, R. J. Yoder, B. Robbertse, J. W. Spatafora, M. S. Rappé, and S. J. Giovannoni (2011). "Phylogenomic evidence for a common ancestor of mitochondria and the SAR11 clade." In: *Scientific Reports* 1, p. 13.

Tillier, E. R. M. and R. A. Collins (2000). "Genome rearrangement by replication-directed translocation." In: *Nature Genetics* 26.2, p. 195.

Uno, T. and M. Yagiura (2000). "Fast algorithms to enumerate all common intervals of two permutations." In: *Algorithmica* 26.2, pp. 290–309.

Vallès, Y., K. M. Halanych, and J. L. Boore (2008). "Group II introns break new boundaries: presence in a bilaterian's genome." In: *PLoS One* 3.1, e1488.

Vergara, J. P. C. (1997). "Sorting by bounded permutations." PhD thesis. Virginia Tech.

Véron, A. S., C. Lemaitre, C. Gautier, V. Lacroix, and M.-F. Sagot (2011). "Close 3D proximity of evolutionary breakpoints argues for the notion of spatial synteny." In: *BMC Genomics* 12.1, p. 303.

Wallace, D. C. (2005). "A mitochondrial paradigm of metabolic and degenerative diseases, aging, and cancer: a dawn for evolutionary medicine." In: *Annual Review of Genetics* 39, pp. 359–407.

Wallace, D. C., G. Singh, M. T. Lott, J. A. Hodge, T. G. Schurr, A.-M. Lezza, L. J. Elsas, and E. K. Nikoskelainen (1988). "Mitochondrial DNA mutation associated with Leber's hereditary optic neuropathy." In: *Science* 242.4884, pp. 1427–1430.

Walter, M. E. M. T., Z. Dias, and J. Meidanis (1998). "Reversal and transposition distance of linear chromosomes." In: *Proc. 5th Int'l Symposium on String Processing and Information Retrieval (SPIRE 1998)*. IEEE, pp. 96–102.

— (2000). "A new approach for approximating the transposition distance." In: *Proc. 7th Int'l Symposium on String Processing and Information Retrieval (SPIRE 2000)*. IEEE, pp. 199–208.

Watson, J. D. and F. H. C. Crick (1953). "Molecular structure of nucleic acids." In: *Nature* 171.4356, pp. 737–738.

Watterson, G. A., W. J. Ewens, T. E. Hall, and A. Morgan (1982). "The chromosome inversion problem." In: *Journal of Theoretical Biology* 99.1, pp. 1–7.

Weigert, A., A. Golombek, M. Gerth, F. Schwarz, T. H. Struck, and C. Bleidorn (2016). "Evolution of mitochondrial gene order in Annelida." In: *Molecular Phylogenetics and Evolution* 94, pp. 196–206.

Wielandt, H. (1964). *Finite Permutation Groups*. Academic Press.

Wieseke, N., T. Hartmann, M. Bernt, and M. Middendorf (2015). "Cophylogenetic reconciliation with ILP." In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 12.6, pp. 1227–1235.

Woese, C. R., O. Kandler, and M. L. Wheelis (1990). "Towards a natural system of organisms: proposal for the domains Archaea, Bacteria, and Eucarya." In: *Proceedings of the National Academy of Sciences* 87.12, pp. 4576–4579.

Wolstenholme, D. R. (1992). "Animal mitochondrial DNA: structure and evolution." In: *International Review of Cytology*. Vol. 141. Elsevier, pp. 173–216.

Xia, X. (2013). "What is Comparative Genomics?" In: *Comparative Genomics*. Springer, pp. 1–20.

Xia, Y., Y. Zheng, R. W. Murphy, and X. Zeng (2016). "Intraspecific rearrangement of mitochondrial genome suggests the prevalence of the tandem duplication-random loss (TDLR) mechanism in Quasipaa boulengeri." In: *BMC Genomics* 17.1, p. 965.

Xiaochen, L., L. Hu, C. Yao, S. Fan, J.-J. Wilson, and C. Wanzhi (2017). "Conserved gene arrangement in the mitochondrial genomes of barklouse families Stenopsocidae and Psocidae." In: *Frontiers of Agricultural Science and Engineering* 4.3, pp. 358–365.

Xu, W., D. Jameson, B. Tang, and P. G. Higgs (2006). "The relationship between the rate of molecular evolution and the rate of genome rearrangement in animal mitochondrial genomes." In: *Journal of Molecular Evolution* 63.3, pp. 375–392.

Yamazaki, N., R. Ueshima, J. A. Terrett, S. Yokobori, M. Kaifu, R. Segawa, T. Kobayashi, K. Numachi, T. Ueda, K. Nishikawa, et al. (1997). "Evolution of pulmonate gastropod mitochondrial genomes: comparisons of gene organizations of Euhadra, Cepaea and Albinaria and implications of unusual tRNA secondary structures." In: *Genetics* 145.3, pp. 749–758.

Yancopoulos, S., O. Attie, and R. Friedberg (2005). "Efficient sorting of genomic permutations by translocation, inversion and block interchange." In: *Bioinformatics* 21.16, pp. 3340–3346.

Yuan, M.-L., Q.-L. Zhang, L. Zhang, Z.-L. Guo, Y.-J. Liu, Y.-Y. Shen, and R. Shao (2016). "High-level phylogeny of the Coleoptera inferred with mitochondrial genome sequences." In: *Molecular Phylogenetics and Evolution* 104, pp. 99–111.

Zeira, R. and R. Shamir (2018). "Genome Rearrangement Problems with Single and Multiple Gene Copies: A Review."

Zhang, M. and H. W. Leong (2008). "Gene team tree: A compact representation of all gene teams." In: *Proc. 6th Int'l Workshop on Comparative Genomics (RECOMB-CG 2008)*. Springer, pp. 100–112.

Zhang, M., W. Arndt, and J. Tang (2009). "An exact solver for the DCJ median problem." In: *Proc. 14th Pacific Symposium on Biocomputing (PSB 2009)*. World Scientific, pp. 138–149.

CURRICULUM VITÆ

PERSONAL INFORMATION

Name:	Dipl.-Math. Tom Hartmann
Born on:	January 21, 1989
in:	Bergen, Germany

EDUCATION

2014–Present	Research assistant at the Leipzig University, Germany
2008–2014	Student of Mathematics at the Leipzig University, Germany
2007	General Certificate of Education (Abitur), Bismarck-Gymnasium Genthin, Germany

SCIENTIFIC COOPERATIONS

Germany	Prof. Martin Middendorf – Leipzig University Dr. Nicolas Wieseke – Leipzig University Dr. Matthias Bernt – Helmholtz Centre for Environmental Research - UFZ
Israel	Prof. Roded Sharan – Tel-Aviv University
India	Prof. Millie Pant – Indian Institute of Technology Roorkee
Taiwan	Prof. Pei-Yun Tsai – National Central University Prof. Yao-Ting Huang – National Chung Cheng University

April 25, 2019

LIST OF PUBLICATIONS

Cheng, Y.-C., T. Hartmann, P.-Y. Tsai, and M. Middendorf (2016). "Population based ant colony optimization for reconstructing ECG signals." In: *Evolutionary Intelligence* 9.3, pp. 55–66.

Hartmann, T., N. Wieseke, R. Sharan, M. Middendorf, and M. Bernt (2017). "Genome Rearrangement with ILP." In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 15.5, pp. 1585–1593.

Hartmann, T., M. Bernt, and M. Middendorf (2018a). "An Exact Algorithm for Sorting by Weighted Preserving Genome Rearrangements." In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics*. (in press).

Hartmann, T., A.-C. Chu, M. Middendorf, and M. Bernt (2018b). "Combinatorics of tandem duplication random loss mutations on circular genomes." In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 15.1, pp. 83–95.

Hartmann, T., M. Bernt, and M. Middendorf (2018c). "EqualTDRL: illustrating equivalent tandem duplication random loss rearrangements." In: *BMC Bioinformatics* 19.192.

Hartmann, T., M. Middendorf, and M. Bernt (2018d). "Genome Rearrangement Analysis: Cut and Join Genome Rearrangements and Gene Cluster Preserving Approaches." In: *Comparative Genomics: Methods and Protocols*. Springer, pp. 261–289.

Hartmann, T., M. Bannach, and M. Middendorf (2018e). "Sorting Signed Permutations by Inverse Tandem Duplication Random Losses." In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics*. (in press).

Wieseke, N., T. Hartmann, M. Bernt, and M. Middendorf (2015). "Cophylogenetic reconciliation with ILP." In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 12.6, pp. 1227–1235.

SELBSTÄNDIGKEITSERKLÄRUNG

Hiermit erkläre ich, die vorliegende Dissertation selbständig und ohne unzulässige fremde Hilfe angefertigt zu haben. Ich habe keine anderen als die angeführten Quellen und Hilfsmittel benutzt und sämtliche Textstellen, die wörtlich oder sinngemäß aus veröffentlichten oder unveröffentlichten Schriften entnommen wurden, und alle Angaben, die auf mündlichen Auskünften beruhen, als solche kenntlich gemacht. Ebenfalls sind alle von anderen Personen bereitgestellten Materialien oder erbrachten Dienstleistungen als solche gekennzeichnet.

Leipzig, 25. April 2019

Tom Hartmann