AN ALGORITHM FOR FINDING OPTIMAL DESCENT TREES IN GENEALOGIES CONDITIONAL ON THE OBSERVED DATA

QIONG LI

An Algorithm for Finding Optimal Descent Trees in Genealogies Conditional on the Observed Data

by

© Qiong Li

A thesis submitted to the
School of Graduate Studies
in partial fulfilment of the
requirements for the degree of
Master of Science in Statistics

Department of Mathematics and Statistics
Memorial University

September 13, 2010

St. John's Newfoundland and Labrador

Abstract

There are many diseases caused by genes such as cystic fibrosis, hereditary spherocytosis and duchenne muscular dystrophy. Identifying the actual disease–causing genes is important since it may help prevent or avoid genetic disorder. Finding out which genes contribute to diseases is also helpful for understanding why some individuals are more inclined to have physical diseases than others. To do this, we should determine regions of chromosomes that are likely to contain the particular genes responsible for a given human disease. Genetic linkage analysis is developed as a statistical method that allows us to determine these regions of chromosomes.

Geneticists use pedigrees because they offer many advantages for genetic mapping regardless of the incidence of the genetic disease. One of this advantages is that study of pedigrees is quite powerful if the disease is rare, nevertheless there are many other aspects, like genetic homogeneity, the patterns of transmission, etc. These advantages make the study of pedigrees attractive. However, utilizing such pedigrees in genetic analysis is a computationally challenging task. Time complexity of some algorithms for genetic analysis is exponential in the size of the pedigree. Therefore, it is desirable to find a potentialy optimal subpedigree that connects the individuals with the disorder. The aim of this thesis is to study the methods of finding optimal subpedigrees conditional on the observed data in a large pedigree.

In chapter 1, we first provide background of finding optimal subpedigrees in a large original pedigree problem, in particular, one method to find such subpedigrees that uses graph theory techniques. Then, we introduce some terminologies in graph

theory related to the Steiner tree problem. At the end of this chapter, some basic ideas about statistical genetics are provided.

Chapter 2 concentrates on the description of constructing subpedigrees in a large pedigree based on the Steiner tree problem in graph theory. Two pieces of software, PedHunter and Miniped, that use the Steiner tree algorithm in constructing optimal subpedigrees are introduced. The study in this chapter also enables us to consider the most likely descent trees conditional on the observed data.

Chapter 3 is devoted to the study of algorithms for finding the most likely descent trees. We first present an algorithm to find the probability of every edge in all possible descent trees, and then reformulate this problem into a directed Steiner tree problem in graph theory. Furthermore, we provide an approximation algorithm to solve this directed Steiner tree problem.

The final chapter summarizes the results in this thesis and points out some problems for future study.

Acknowledgements

First and foremost, I would like to express my deepest appreciation to my supervisor, Professor J Concepcion Loredo-Osti, whose encouragement, guidance and support from the initial to the final level enabled me to develop an understanding of the amazing area of statistical genetics. Without his contributions of time and ideas, this thesis would not have been possible.

I warmly thank Professor Zhaozhi Fan for teaching me Survival analysis and Linear models, and giving me frequent encouragement. In addition, I have also benefited from many conversations with Professor Hong Wang, who has always been very generous in discussing my questions about probability. I am also indebted to Professor Xiaoqiang Zhao and Mrs. Zhao. Their kind help made my life in St. John's much more enjoyable.

I would like to take this opportunity to express my sincere thanks to Professors Kenneth Morgan and Mary Fujiwara for providing the Bowen–Conradi syndrome database and their very useful comments improve the quality of the manuscript.

I would like express my thanks to Professors Jie Xiao and Veeresh Gadag for their continuous encouragement. I also owe my sincere gratitude to Min Chen, Fang Fang, Jian Fang, Jianhui Feng, Yanjing He, Rui Hu, Yunqi Ji, Yu Jin, Luju Liu, Yi Tao, Haiyan Yang, Zhichun Zhai and Yuxiang Zhang for their help and support. My time at Memorial University was made enjoyable in large part due to many friends.

I greatfully acknowledge CIHR and MITACS for the funding support through the grant to my supervisor. I am also grateful to the School of Graduate Studies and

Department of Mathematics and Statistics for providing financial supports. Thanks also go to all staff members at the department for their help.

It is my great pleasure to thank Professors Yuehua Bu, Liying Kang and Weifan Wang for their constant encouragement.

Last, but not least, I deeply thank my parents and my parents-in-law. Most importantly, I would like to thank my husband, Yijun Lou, for helping me navigate through many years of study and research. He has supported me every day, both emotionally and practically. Their understanding and support have provided me the strength to keep going.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Variation of human physical traits (such as hair curl, tongue rolling and hitch–hiker's thumb) and diseases (such as cystic fibrosis, Huntington's disease and neurofibromatosis type–1) is now viewed as having some genetic influences. However, in most human inherited diseases, the gene or genes responsible for the disease as well as their locations remain unknown.

Searching for the particular genes responsible for some given disease is a critical step in the understanding of genetic diseases. Traditionally, finding a disease gene starts with linkage analysis. Genetic linkage analysis is a useful statistical method that is used to map disease genes and associate functionality of genes with their location on the chromosome. More specifically, the goal of linkage analysis is to find out the location of the gene relative to a genetic marker with known location, which has been genotyped in the nuclear family.

If offspring obtain some disease passed from parents along with a specific genetic marker, then it can be concluded that the genes which are responsible for the disease are located close to the genetic marker on the chromosome. In this case, the normal allele and the disease allele can be distinguished by seeking the occurrence of the disease in a pedigree. Therefore, a pedigree plays an important role in linkage analysis. Linkage analysis of pedigree data is also a powerful technique to find a candidate

region of the genome, which is likely to contain the gene responsible for the particular trait or disease under investigation. On the other hand, pedigree data can be used to predict the recurrence risk of a disease in future generations. In summary, pedigree analysis allows a better understanding of the transmission of a gene or genes in a family.

Successfully mapping the genes for complex traits requires large global population samples. If the populations in our study have better characteristics and their history can be established, we have more opportunities to design statistical strategies for mapping critical DNA regions. Isolated populations are such large global population samples.

Population isolates have already been proved to be an important and powerful resource for locating genes. A population isolate is a subpopulation without genetic interchange from other surrounding subpopulations over many generations because of the geographical and/or cultural isolation. Genetically, isolated populations possess many advantages for mapping inherited disorders. Foremost among these is the existence of multigenerational pedigrees including the affected individuals sharing ancestral haplotypes derived from a small number of founders. Here is a list of features of isolated populations versus outbred populations [44]:

- Higher prevalence for some diseases
- More inbreeding and opportunities to map recessive genes
- More uniform genetic background
- Better genealogical records
- Easier to standardize phenotype definitions
- Wider regions of linkage disequilibrium
- Closer to Hardy Weinberg equilibrium
- Less migration and more intact families
- More homogeneous environment
- Potentially more affected people

- More opportunities for replication
- Higher participation rate in studies

Although large extended human pedigrees from population isolates have great potential for linkage analysis, utilizing such pedigrees is computationally challenging. Figure 1.1 shows a large human pedigree. The pedigree size limits applicability of current linkage analysis software. For example, the Lander–Green–Kruglyak algorithm is frequently used for exact multipoint linkage analysis (see, e.g., [30, 32, 1, 20]). However, computer programs implementing this algorithm can not handle exactly large pedigrees with complex structures. Time and space complexity can increase exponentially with the size of a pedigree. Therefore, investigators have to reduce the complexity of these pedigrees by trimming, breaking or splitting a large pedigree into smaller fragments for multipoint linkage analysis, haplotype reconstruction and IBD (identical by descent) computations.

1.1 Related works

Many algorithms of simplifying the pedigree structure to fit the genetic analysis have been proposed recently (see, e.g., [2, 51, 34, 43, 17]). Each simplified pedigree should contain all affected individuals and at least one common ancestor.

1.1.1 Pedigree trimming

Geneticists are often forced to simplify pedigrees by dropping "irrelevant" members automatically. The goal of the pedigree trimming problem is to trim the uninformative branches in a complicated pedigree. PowerTrim and Mendel are two programs for solving the pedigree trimming problem provided by Thornton and Haines [51] and Lange and Sinsheimer [34], respectively.

Individuals or whole families satisfying the following conditions [51] are trimmed

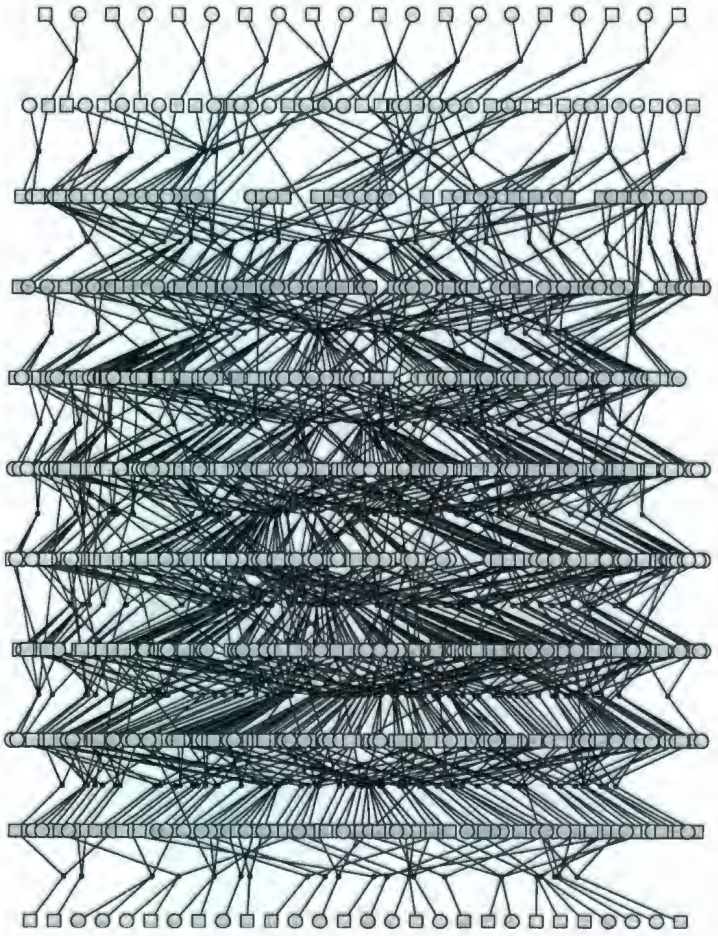


Figure 1.1: An initial pedigree from a population isolate. Pedigree drawn using software package Pedfiddler version 0.5.

from original pedigrees by PowerTrim.

- Families consisting of only a parent and a child.
- Individuals who have no parents and no children.
- Individuals who are unaffected and ungenotyped.
- Individuals who are ungenotyped, have no genotyped descendants, and have either an affected parent or an affected sibling.
- Founder individuals who are unaffected and ungenotyped and have only one child.
- Individuals or groups of individuals within a family who are unconnected to the proband.

Pedigree trimming algorithms potentially reduce the time consumed in linkage analysis by decreasing pedigree size.

1.1.2 Pedigree breaking

In 2001, Pankratz and Iturra [43] proposed a semiautomatic method to partition an entire pedigree into more manageable subunits. They used principles from factor analysis to split a large pedigree into discrete subgroups for fitting computational constraints. This method has been applied to the Genetic Analysis Workshop 12 Hutterite pedigree. In this analysis, twelve pedigrees were used to perform a genome wide linkage scan for IgE serum level. The result of two point linkage analysis is that two marker loci D1S3723 (LOD = 3.7) and D1S534 (LOD = 3.58) on chromosome 1 have strong evidence for linkage. There is a maximum LOD score of 4.58 on chromosome 1 at 143 cM from multipoint linkage analysis.

A number of other automatic algorithms for the pedigree splitting problem have recently been proposed to simplify linkage analysis (see [17, 3, 36, 29, 5]).

One of the most efficient algorithms for the pedigree breaking problem was provided by Falchi et al. [17]. Their algorithm is based on the maximum–cliques partitioning algorithm in graph theory. In what follows, we will introduce their ideas.

In a pedigree, every individual can be represented as a vertex in a graph. The edge between every two individuals is weighted by the pairwise measure of relatedness, such as their kinship coefficient or the number of meiotic steps separating them. This algorithm firstly creats a subgroup of related individuals or cliques using a maximum clique partitioning algorithm. Cliques of maximum size are iteratively searched and deleted from the graph until there are no more vertices left. Finally, they reconstruct pedigrees through keeping a small pedigree size in each clique. In the automatic pedigree breaking algorithm, the size and the structure of subpedigrees depend on a number of prespecified parameters: the maximum number of generation, the minimum clique size, the minimum kinship level and the maximum allowed number of non–genotyped subjects.

An earlier work for reducing the complexity of the pedigree was provided by Agarwala et al. [2]. They used the algorithm of the Steiner tree problem in graph theory to obtain a subpedigree to minimize calculation burden. However, the algorithm could not handle large pedigrees; Löschner et al. [37] discussed a faster algorithm to handle large pedigrees. In this thesis, we investigate a better algorithm considering more genetic information in observed data.

1.2 Introduction to graph theory

In this section, we present some definitions in graph theory and known results which are going to be used in the rest of this thesis (see, e.g., [14, 52]). We also provide a short introduction to complexity of computer algorithms in this section.

1.2.1 Definitions in graph theory

Definition 1.2.1 *A directed graph can be defined as a pair (V, E), where V is a finite nonempty set of vertices, and E is a finite set of arcs. The elements of E are ordered*

pairs of vertices.

Definition 1.2.2 *The order of a graph $G = (V, E)$, denoted by $n = |V|$, is the number of vertices and the size denoted by $m = |E|$, is the number of arcs. An arc $e = (v, w)$ is said to be incident with vertices v and w, where v is the source and w is the target of arc e.*

In a directed graph, a vertex has two degrees. One degree is the number of arcs coming into the vertex and the other is the number of arcs going out of the vertex.

Definition 1.2.3 *The indegree of a vertex v in a directed graph $G = (V, E)$ is the number of arcs in G whose target is v, that is $indeg(v) = |\{(u, v)|(u, v) \in E\}|$. The outdegree of a vertex v is the number of arcs in G whose source is v, that is $outdeg(v) = |\{(v, w)|(v, w) \in E\}|$.*

Walks and paths in a graph are alternating sequences of vertices and arcs are such that each arc in the sequence is preceded by its source and followed by its target.

Definition 1.2.4 *A walk from vertex v_i to vertex v_j in a directed graph is an alternating sequence $[v_i, e_{i+1}, v_{i+1}, e_{i+2}, \ldots, v_{j-1}, e_j, v_j]$ of vertices and arcs in the graph such that $e_k = (v_{k-1}, v_k)$ for $k = i+1, \ldots, j$. A path is a walk with no repeated vertices and repeated arcs. The length of a path is the number of arcs in the sequence. A path from u to vertex v is a shortest path if it has minimum length of any path from u to v.*

Definition 1.2.5 *A path $[v_i, e_{i+1}, v_{i+1}, e_{i+2}, \ldots, v_{j-1}, e_j, v_j]$ is said to be closed if $v_i = v_j$. A cycle is a closed path of length at least one.*

An undirected graph is the particular case of a directed graph in which for every arc (u, v) of the graph, the reversed arc (v, u) also belongs to the graph.

Definition 1.2.6 *A graph $G = (V, E)$ is undirected if the elements of E are unordered pairs of vertices. Two vertices x, y of G are adjacent if xy is an edge of*

G. If all the vertices of G are pairwise adjacent, then G is complete. If a subgraph $G' = (V', E') \subseteq G$ and G' contains all the edges $xy \in E$ with $x, y \in V'$, then G' is an induced subgraph of G.

Definition 1.2.7 *A clique in an undirected graph $G = (V, E)$ is a subset of the vertex set $C \subseteq V$, such that the subgraph induced by C is complete.*

An important graph theoretical notion is the connected graph.

Definition 1.2.8 *An undirected graph $G = (V, E)$ is connected if for every pair of vertices $v, w \in E$, there is a path between v and w.*

Definition 1.2.9 *A directed graph is connected if the underlying undirected graph is connected.*

Definition 1.2.10 *A connected directed graph $G = (V, E)$ is said to be a tree if the underlying undirected graph has no cycles and there is a distinguished vertex $r \in V$, called the root of the tree, such that for all vertices $v \in V$, there is a path in G from the root r to vertex v.*

Theorem 1.2.1 *If T is a tree of order n, then T has $n - 1$ edges.*

The *clique partition problem* is defined as follows: Let G be a graph. A set $S = \{G_1, G_2, \ldots, G_k\}$, $k \geq 1$ of subgraphs of G is called a covering of G if $E(G) = \cup_{i=1}^{k} E(G_i)$. If each element of S is a clique, then S is called a clique cover of G. Clique partition is a clique cover S in which each edge belongs to exactly one member of S. The clique partition problem asks whether a given graph G has a clique partition of size at most k.

The *Steiner tree problem* in undirected graph is defined as follows: Given a undirected graph $G = (V, E)$ with a cost function c on the edges, and a subset of vertices $X \subseteq V$ (called terminals), the goal is to find a minimum cost tree that includes all

vertices in X. The cost of the tree is defined as the sum of the costs of the edges in the tree.

The Steiner tree problem is one of most well known combinatorial optimization problems. It was studied in various applications, including very large–scale integration designs, optical and wireless communication networks and phylogenetic tree reconstruction in biology. A network routing problem is an application of the Steiner tree problem in network design. In this problem, a communication server has to distribute the same data from a central server to several nodes in a network by selecting a minimum cost set of edges that connect the server to all nodes.

The Steiner tree problem has also been applied to reconstruct phylogenetic trees in biology. A phylogeny is a tree representing the evolutionary history. Reconstruction of a phylogenetic tree for a given set of species using protein sequences is a central problem in phylogeny. Each vertex of a phylogenetic tree represents a protein sequence for identifying the species. The protein sequence is identified with a vector from A^m (A is a finite alphabet). Edges represent mutations obtained from one sequence to another. Then, we have a graph $G = (A^m, E, c)$ where the cost function c is the Hamming distance between two sequences. The reconstruction of a phylogenetic tree aims to find a Steiner tree in G, where the set of given species corresponds to the set of terminals in the Steiner tree problem.

The *directed version of the Steiner tree problem* is defined as follows: Given a directed weighted graph $G = (V, A)$, a specified root $r \in V$, and a set of terminals $X \subseteq V$ ($|X| = k$), the objective is to find the minimum cost tree which is rooted at r and spans all the vertices in X (in other words, r should have a path to every vertex in X).

The *all shortest paths Steiner tree* is a Steiner tree on the all shortest paths graph with respect to a root r and a terminal set X, where the all shortest paths graph is a subgraph consisting only of those vertices and edges which from the shortest paths connecting r with elements of X.

1.2.2 Complexity of algorithms

The complexity of algorithms is a way to measure the quality of algorithms. It is used to compare different algorithms for solving the same problem. Complexity theory studies an algorithm's time and memory space as a function of the size of the input data (see, e.g., [25, 54]).

In computer science, *time complexity* of an algorithm quantifies the amount of time by an algorithm to run as a function of the size of the input to the problem. *Space complexity* is defined as memory space with respect to the size of the input data that this algorithm needs to store during its execution. The complexity of an algorithm is commonly expressed using the big O notation, meaning that if there are two functions $f(n)$, $g(n)$ and a constant $c > 0$ such that $f(n) \leq cg(n)$ for all sufficiently large n, then $f(n) = O(g(n))$.

Note that it is common practice to measure the complexity of the worst possible case for a given size of the input data. Therefore, we can see that most of algorithms work very quickly in practice although these algorithms have exponential complexity.

An algorithm is said to be *polynomial time* if its running time is upper bounded by a polynomial in the size of the input data for the algorithm. From a practical point of view, the polynomial algorithms have proved to be the most useful. If we have found a polynomial algorithm for a problem, we can say that this problem was well solved. Unfortunately, there is a very large class of problems which admits no known polynomial algorithm. Moreover, there are good reasons to believe that such a polynomial algorithm does not exist. We call that this class of problems NP–complete. Most problems arising in practice have a tendency to be NP–complete.

A decision problem is a question whose solution is either 'yes' or 'no'. For example, the problem "given two numbers x and y, does x evenly divide y?" is a decision problem. The answer can be either 'yes' or 'no'. We denote the class of all polynomial decision problems by P and the class of decision problems for which a 'yes' answer can be verified in polynomial time by NP. In fact, the question whether P = NP is an

outstanding question of complexity theory [18]. This problem has been recognized as one of the top three mathematical problems in the 21st century. It is also one of the most important open questions in computer science.

The complexity class *NP–complete* is a class of problems having two properties:

1. NP–complete problem is in NP.

2. If the problem can be solved in polynomial time, then so can every problem in NP.

We can see that if we could find a polynomial algorithm for such a NP–complete problem, we would prove P = NP.

The Steiner tree problem is NP–complete [19] which means unless P=NP there can not exist a polynomial time approximation scheme for the Steiner tree problem. For NP–complete problems, computer scientists do not try to solve these problems exactly, instead, the problems are often addressed by using approximation algorithms. Approximation algorithms are algorithms used to find approximate solutions to optimization problems. An algorithm is called a *polynomial time α–approximation algorithm* for a NP–complete problem if given any instant I of the problem, this algorithm can give a solution with cost $c(I) \leq \alpha c_{opt}(I)$, where $c_{opt}(I)$ is the cost of an optimal solution of I, and the running time of this algorithm is bounded by a polynomial in the size of the input I.

There exist many approximation algorithms for the Steiner tree problem, some of them are listed in Table 1.1.

For the directed Steiner tree problem, a polynomial time approximation algorithm for an acyclic graph is due to Zelikovsky [58]. He gave an approximation algorithm with an approximation ratio of $(2+lnk)^{i-1}k^{1/i}$ for any i. Charikar et al. [11] presented a better approximation ratio $i(i-1)k^{1/2}$ algorithm which runs in time $O(n^i k^{2i})$ for any $i > 0$, where k is the number of terminals.

Year	Approximation ratios	References
1980	2	Takahash and Matsuyama [48]
1993	1.833	Zelikovsky [56]
1994	1.746	Berman and Ramaiyer [6]
1997	1.734	Borchers and Du [7]
1996	1.693	Zelikovsky [57]
1997	1.666	Prömel and Steger [45]
1997	1.644	Karpinski and Zelikovsky [27]
1999	1.598	Hougardy and Prömel [23]
2005	1.55	Robins and Zelikovsky [46]

Table 1.1: Some algorithms for Steiner tree problem [15].

1.3 Necessary background in statistical genetics

Statistical analysis is used in many fields of genetic research. Many statistical and computational tools are developed to study genetics and molecular biology. In this section, we briefly introduce some elementary results from population genetics discussed in more detail in the references [10, 12, 16].

1.3.1 Preliminaries of statistical genetics

There are 46 chromosomes, comprised of 23 pairs in the nucleus of each normal human cell. Two of these are sex chromosomes–two paired Xs for a female and an X and a Y for a male. The remaining 22 homologous pairs of chromosomes are termed *autosomes*.

Each human chromosome has a short arm and long arm separated by a centromere. The ends of the chromosome are called telomeres. Each chromosome arm is divided into regions, or cytogenetic bands, that can be seen using a microscope and special stains. The cytogenetic bands are labeled p1, p2, q1, q2, etc., counting from the

centromere out toward the telomeres. At higher resolutions, sub–bands can be seen within the bands. The sub–bands are also numbered from the centromere out toward the telomere. For instance, the cytogenetic map location of a gene termed cystic fibrosis transmembrane conductance regulator (CFTR)(see, e.g., [41, 28]) is 7q31.2, which indicates that it is on chromosome 7, q arm, band 3, sub–band 1, and sub–sub–band 2 (Figure 1.2).

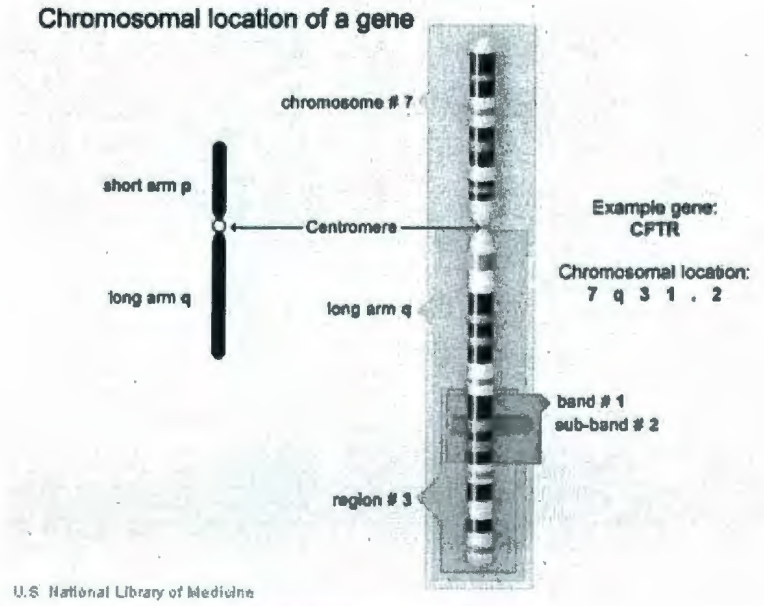


Figure 1.2: The location of the gene CFTR. Source:
http://ghr.nlm.nih.gov/handbook/illustrations/chromosomallocation

Any small segment of the DNA of the chromosome is known as a *locus*. Each locus can be occupied by one of several variants called *alleles*. Except for the sex chromosomes in males, it follows that there are two alleles at every locus. If one locus is related to a functional gene, the observable characteristic of the individual is the *phenotype*. For instance, a phenotype can be binary (affected or nonaffected) or quantitative (such as height or weight). A *genotype* consists of the pair of alleles found at a given locus, one inherited from the father and one from the mother. For

example, the DNA which codes for the antigens that determine an individual's ABO blood type resides on the long arm of chromosome 9. There are three alleles, A, B, and O. The genotype sets corresponding to these three alleles are AA, AO, BB, BO, OO, and AB.

Genotypes for which the two alleles are different, such as AO, BO or AB are known as *heterozygous* genotypes. If the two alleles are the same, such as AA, BB or OO in an individual, the individual is called *homozygous* at this locus. The phenotype A results from either the heterozygous genotype AO or the homozygous genotype AA; similarly, phenotype B results from either BO or BB. Alleles A and B both mask the presence of the O allele and are said to be *dominant* to it. Alternatively, O is *recessive* to A and B. Relative to one another, alleles A and B are *codominant.*

A gamete's sequence of alleles along a chromosome constitutes a *haplotype.* For example, the first locus contains alleles A and a with three possible genotypes AA, Aa, and aa; the second locus containing alleles B and b again gives three possible genotypes BB, Bb, and bb. Therefore, for a given individual, there are nine possible configurations for the genotypes at these two loci, as shown in the Table 1.2 below, which shows the possible haplotypes that an individual may carry.

	AA	Aa	aa
BB	AB/AB	AB/aB	aB/aB
Bb	AB/Ab	AB/ab or Ab/aB	aB/ab
bb	Ab/Ab	Ab/ab	ab/ab

Table 1.2: Ten possible diplotypes.

When a parent transmits a chromosome to a child, the phenomenon of recombination can often be observed, meaning that the chromosome differs from both of the corresponding homologous parental chromosomes. For example, consider two loci on the same chromosome, there are alleles A, a at the first locus and B, b at the

second one. Suppose an individual has inherited a haplotype AB from the father and ab from the mother. A gamete receiving a haplotype Ab during the meiosis is said to be recombinant, meaning that the two alleles come from different parents. The haplotype aB is also recombinant while AB and ab are non–recombinant.

Genetic loci which are physically close to one another on the chromosome tend to be transmitted together when passed on to offspring. Genes inherited in this way are said to be linked. For example, the genes in fruit flies affecting eye color and wing length are inherited together because they appear on the same chromosome. The recombination fraction θ is the probability that two loci become recombinant during meiosis. The recombination fraction is a measure of genetic linkage and is used in the creation of a genetic linkage map. Two loci are called linked when $\theta < 0.5$. Most loci on different chromosomes are unlinked, meaning that $\theta = 0.5$, and their segregation may be associated only if one modulates the transmission of the other.

Recombination or linkage study is critical for identifying the location of genes that cause genetic diseases. A *marker* is simply a place on a chromosome with an identifiable physical location on a chromosome whose inheritance can be followed. Markers are often used as tools for tracking the inheritance pattern of a gene that has not yet been identified but whose approximate location is known. If some disease is often transmitted to offspring along with specific marker alleles, then it would be possible to conclude that the genes are responsible for the disease.

The statistical estimate of whether two loci are likely to lie near each other on a chromosome and are therefore likely to be inherited together is called a LOD score. The LOD score is just a function of the likelihood ratio statistic for the hypothesis $H_0 : \theta = 0.5$ vs $H_1 : \theta \neq 0.5$, or a close variant of it, as we will see later. A LOD score of three or more is generally taken to indicate that the two loci are linked and are close to one another.

A *centimorgan* (cM) is a way of measuring the distance between genes on a chromosome. One cM is equal to a 1% chance that recombination will occur between two

loci.

Two genes G_1 and G_2 are *identical by descent* (IBD) if one is a physical copy of the other or if they are both physical copies of the same ancestral gene.

A *nuclear family* is a family group consisting of two parents and their common offspring.

1.3.2 Two point linkage analysis in experimental crosses

Linkage analysis critically relies on a pedigree in which both recombinant and non-recombinant gamete types can be counted. A segregating pedigree comprising an F_2 population, initiated with two contrasting inbred lines, has proven to be a powerful and efficient tool for linkage analysis.

Here, we describe the two point linkage analysis for F_2 population, initiated with two contrasting unrelated inbred lines. In this model, parents P_1 and P_2 are homologous for two alleles of each gene. They mated to generate F_1 individuals. Thus, all F_1 individuals are heterozygous at all genes. The F_1 individuals crossed with each other to generate the F_2 generation. A diagram illustrating this crossing procedure is illustrated in Figure 1.3.

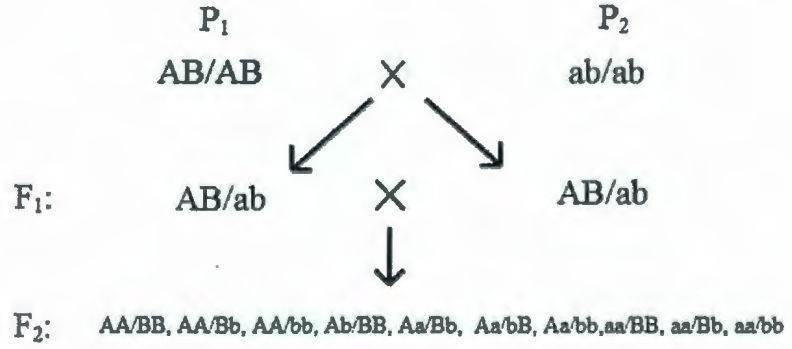


Figure 1.3: Two point linkage analysis of markers.

Consider two markers, **A** with two alleles A and a, and **B**, with two alleles B

and b. Parents P_1 and P_2 are homozygous for these two alleles. Parent P_1 generates haplotype AB during meioses, whereas parent P_2 generates haplotype ab. These two haplotypes are combined to form the heterozygous F_1. Then F_1 will generate four different gametes, AB, ab, Ab and aB. The recombination fraction between the two genes is denoted by θ. Thus, the nonrecombinant type has the frequency of $(1-\theta)/2$ and the recombinant type has the frequency of $\theta/2$. The observed numbers of these nine genotypes can be arrayed as

	AA	Aa	aa
BB	n_{22}	n_{12}	n_{02}
Bb	n_{21}	n_{11}	n_{01}
bb	n_{20}	n_{10}	n_{00}

The genotype frequencies for these two markers **A** and **B** can be arrayed as follows

	AA	Aa	aa
BB	$\frac{1}{4}(1-\theta)^2$	$\frac{1}{2}\theta(1-\theta)$	$\frac{1}{4}(\theta)^2$
Bb	$\frac{1}{2}\theta(1-\theta)$	$\frac{1}{2}[(1-\theta)^2+\theta^2]$	$\frac{1}{2}\theta(1-\theta)$
bb	$\frac{1}{4}(\theta)^2$	$\frac{1}{2}\theta(1-\theta)$	$\frac{1}{4}(1-\theta)^2$

where the AaBb cell includes two haplotypes Aa/Bb and Aa/bB, then we have missing data z, which are the number of the haplotype Aa/Bb. The missing data $z \sim binomial(n_{11}, r/2)$, where r is the expected number of recombinants for the genotype AaBb.

$$r(\theta) = \frac{0 \times \frac{1}{2} \times (1-\theta)^2 + 2 \times \frac{1}{2} \times \theta^2}{\frac{1}{2} \times (1-\theta)^2 + \frac{1}{2} \times \theta^2} = \frac{2\theta^2}{(1-\theta)^2+\theta^2}.$$

The likelihood function in the complete data is

$$L(\theta|\underline{Y}, z) = \frac{n!}{n_{22}!n_{12}!\ldots n_{10}!n_{00}!}[\tfrac{1}{4}(1-\theta)]^{n_{22}+n_{00}}[\tfrac{1}{2}\theta(1-\theta)]^{n_{12}+n_{21}+n_{01}+n_{10}}$$

$$[\tfrac{1}{4}\theta^2]^{n_{02}+n_{20}}[\tfrac{1}{2}(1-\theta)^2]^{n_{11}-z}[\tfrac{1}{2}\theta^2]^z + constant,$$

where n is the total number of all genotypes and $\underset{\sim}{Y} = (n_{22}, \ldots, n_{00})$.

The estimation of θ can be obtained by implementing the EM algorithm. Let z be the missing data and $n, n_{22}, \ldots, n_{00}$ be the observed data.

The log likelihood function becomes

$$\begin{aligned}\log L(\theta|\underset{\sim}{Y}, z) = {} & (n_{22} + n_{00}) \log[\frac{1}{4}(1-\theta)] + (n_{12} + n_{21} + n_{01} + n_{10}) \log[\frac{1}{2}\theta(1-\theta)] \\ & + (n_{02} + n_{20}) \log(\frac{1}{4}\theta^2) + (n_{11} + z) \log[\frac{1}{2}(1-\theta)^2] \\ & + z \log(\frac{1}{2}\theta^2) + constant.\end{aligned}$$

In the E step of the EM algorithm, we take the expectation of $\log L(\theta|\underset{\sim}{Y}, z)$ conditional on the observed data $\underset{\sim}{Y}$ and the current parameter θ_m. It is obvious that

$$E(n_{ij}|\underset{\sim}{Y}, \theta_m) = n_{ij}, \quad i = 0, 1, 2 \quad and \quad j = 0, 1, 2.$$

$$z_m = E(z|\underset{\sim}{Y}, \theta_m) = n_{11}\frac{r(\theta_m)}{2}.$$

Let

$$Q(\theta|\theta_m) = E[\log L(\theta|\underset{\sim}{Y}, z)|\underset{\sim}{Y} = y, \theta_m].$$

The M step of the EM algorithm maximizes the $Q(\theta|\theta_m)$ function by replacing z by z_m. Setting the first derivative

$$\begin{aligned}Q(\theta|\theta_m) = {} & \frac{n_{22} + n_{00}}{\theta - 1} + \frac{n_{12} + n_{21} + n_{01} + n_{10}}{\frac{1-2\theta}{\theta(1-\theta)}} + \frac{2(n_{02} + n_{20})}{\theta} \\ & + \frac{2(n_{11} + z_m)}{\theta - 1} + \frac{2z_m}{\theta}\end{aligned}$$

equals to 0 provides the unique stationary point of $Q(\theta|\theta_m)$. The solution of the resulting equations is

$$\theta_m = \frac{1}{2n}[n_{12} + n_{21} + n_{01} + n_{10} + 2(n_{02} + n_{20}) + 2z_m].$$

This procedure is repeated until the estimate converges to a stable value $\hat{\theta}$.

The MLE of θ can be used to determine the degree of linkage between the two markers. If $\hat{\theta}$=0, then there is evidence that the two markers are completely linked, in other words, heterozygous F_1 produces only nonrecombinant gametes. If $\hat{\theta}$=0.5, there is evidence that there is no linkage between the two markers, in other words, F_1 produces both recombinant and nonrecombinant haplotypes in equal proportions.

The hypotheses are:

$$\begin{cases} H_0 : \theta = 0.5 \\ H_1 : \theta < 0.5. \end{cases}$$

The test statistic used to detect linkage is the LOD score given by

$$LOD = \log_{10}\left(\frac{L(\hat{\theta}|\underset{\sim}{Y})}{L(\theta = 0.5|\underset{\sim}{Y})}\right).$$

LOD score of three or more is generally considered to indicate that two markers are close to each other on a chromosome.

Two point linkage analysis in F_2 population derived from two inbred lines were described above. However, this analysis based on inbred line is not appropriate for outbred species. Outbred populations have two significant characteristics that make their two point linkage analysis different from those in inbred line crossed. One characteristic is that the number of alleles and the inheritance mode of markers vary from locus to locus. The other characteristic is the uncertainty about linkage phases between different loci [55]. For more details about two point linkage analysis based on outbred line, we refer the readers to Lu et al. [39] and Maliepaard et al. [40].

Chapter 2

Human pedigrees within large genealogies and the Steiner tree problem in graph theory

2.1 Problem formulation

A pedigree is a set of individuals together with a full specification of all the relationship between them [50]. By examining a pedigree, the mode of inheritance of the disorder can be interpreted that allows a better understanding of the transmission of genes.

A form of this specification is to identify each individual's father and mother. Individuals without parents in a pedigree are *founders*; other individuals are *non–founders*. Individuals in a pedigree without offspring are referred to as *final* individuals. Pedigree members with offspring are *spouses*.

A standard diagram representation of a pedigree is shown in Figure 2.1. The diagram of representation of a pedigree makes it easier to visualize relationships within families, particularly in large extended genealogies. A pedigree can be expressed by a directed graph [33], such as in Figure 2.2(a) where the nodes denote pedigree members, and the arcs connect individuals to their offspring. A pedigree can also be expressed

by a marriage node graph [49], such as in Figure 2.2(b). Note that Figure 2.2(b) has two kinds of nodes, individual and marriage nodes and two kinds of arcs, connecting an individual to his marriages and connecting a marriage to the offspring of that marriage.

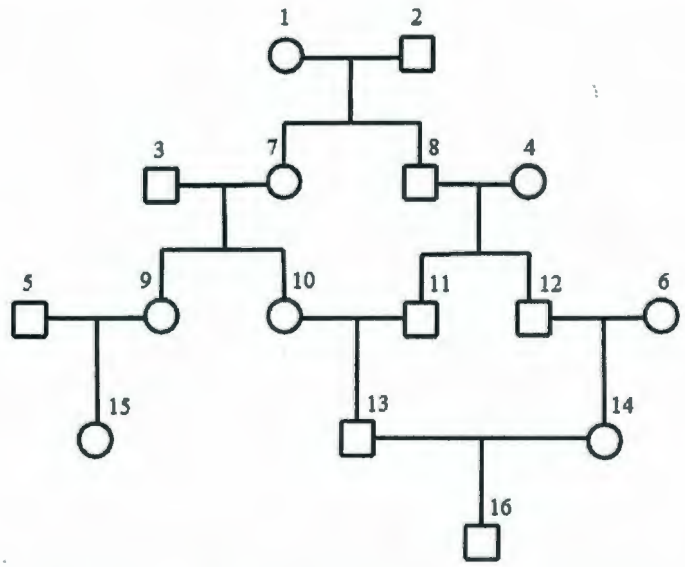
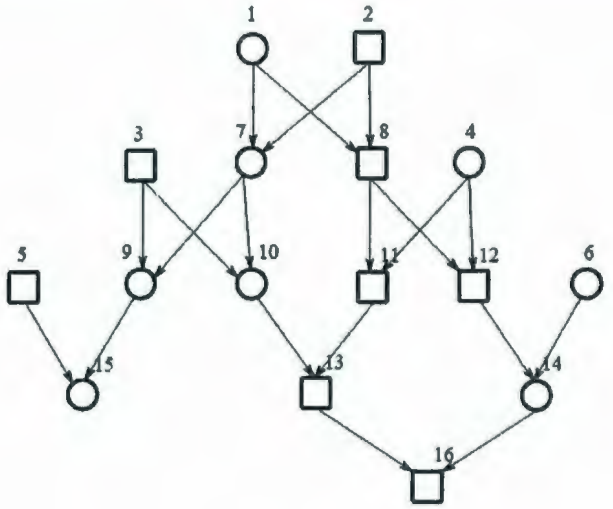


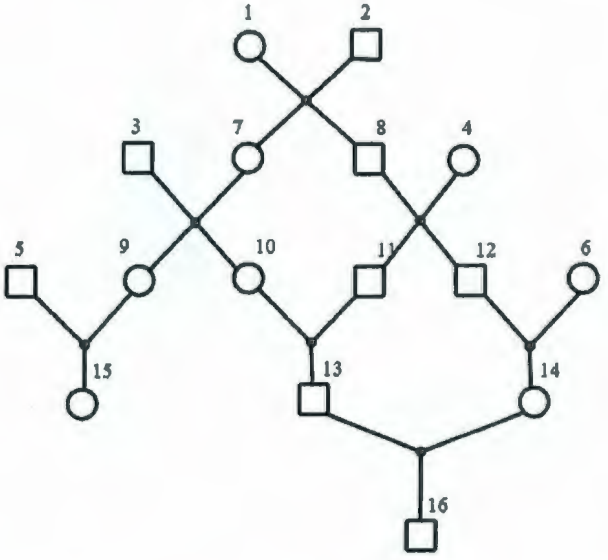
Figure 2.1: A standard diagram representation of a pedigree. The representations of a simple pedigree of 16 individuals. Females are represented by circles and males by squares. All members of one generation are shown in a row, with preceding generations above and later generations below. Individuals 1, 2, 3, 4, 5 and 6 are founders, while 15 and 16 are final individuals.

A graph representing a pedigree is called a *pedigree graph*. Any directed pedigree graph is a special graph having the following properties.

Firstly, suppose $G = (V, A)$ is a directed pedigree graph, then for any founder v in this pedigree graph, the indegree of v is $indeg(v) = 0$. For any vertex w that is not a final individual, the outdegree of w is $outdeg(w)$ which equals to the number of



(a)



(b)

Figure 2.2: In (a), a representation of a pedigree with nodes and directed edges (arcs) connecting individuals to their offspring and in (b), a marriage node graph with two kinds on nodes (individual and marriage) and two kinds of edges (connection between an individual and a marriage and connection between a marriage and offspring from the marriage).

children of the individual w. We know that each individual must have two and only two biological parents, although one or both of them may be unknown. Thus, for any vertex u that is not a founder, the indegree of u is $indeg(u) = 2$.

Secondly, a directed path p from an individual v to another individual u in a pedigree graph gives one way that u inherits genetic material from v. If there exists no directed path from v to u, then there is no way that u can inherit genetic material from u, and v is not an ancestor of u. For example, in Figure 2.2 (b), individual 15 does not receive genetic material directly from individual 8, although they have a common ancestor. In any directed pedigree graph, the edges are directed from a parent to a child, and it is biologically impossible to have a cycle in a directed pedigree graph. Hence, any directed pedigree graph is acyclic.

Finally, if incest is forbidden, no individual may marry his own sister or brother, then there can be no subgraph in the pedigree graph given in Figure 2.3(a). Also, no individual may marry one of his parents, thus there is no subgraph of the type represented in Figure 2.3(b) in a pedigree graph (see also [24]).

For a given set of people X, the set of *least common ancestors* of X is the set of people A such that any individual in A is an ancestor of every individual in X and no individual in A has a descendant who is also an ancestor of every individual in X, i.e., individuals in A have no descendance in A.

Parsimony is the principle that the simplest explanation that can explain the data is to be preferred. This principle is widely used in phylogenetic reconstruction. By this principle, ancestral relationships can be reconstructed by minimizing the total number of evolutionary steps to explain the given data.

Our problem is to find a subpedigree that simplifies the complexity of an original pedigree. Moreover, this subpedigree should retain most relevant features of the pedigree. By the principle of parsimony, two hypotheses can be considered. One is to minimize the number of meioses in our problem. The other is that a disease–causing allele passes from one individual u to another individual v in all shortest paths from

(a) (b)

Figure 2.3: (a) Brother-sister mating and (b) Child-parent mating.

u to v.

When we consider the first hypothesis, our problem is equivalent to the Steiner tree problem in graph theory. In this situation, the given pedigree graph is a directed weighted graph. In the corresponding directed weighted graph, a specified root can be one of the nearest common ancestors. Moreover, the set of terminals can be the set of all affected individuals or the set of the parents of all those affected individuals. Hence, minimizing the number of meioses is equivalent to minimizing the number of edges in a pedigree graph. By Theorem 1.2.1, any tree with m edges must have exactly $m+1$ vertices. Thus, the problem is to find a Steiner tree in a pedigree graph where all edges have a weight of 1. By this way, the obtained Steiner tree has a minimal size set of parent–child links with the fewest possible number of meioses. Furthermore, this Steiner tree also retains paths such that each affected child is potentially identical by descent for an allele from the least common ancestor.

When we consider the second hypothesis, we get an all shortest paths pedigree graph. Then we try to find a Steiner tree in this all shortest paths pedigree graph. The obtained all shortest paths Steiner tree guarantees that the disease–causing allele is passed from the least common ancestor to all affected individuals in a shortest path. This also can be seen as a dimension reduction strategy: first we find the all shortest paths (something that can be done efficiently) and, subsequently, find the Steiner trees on these all shortest paths subpedigrees. The overall "optimal" tree is selected amongst these Steiner trees.

Any of the Steiner trees or the all shortest paths Steiner tree represents a possible descent tree in which an allele is passed from one common ancestor to a given set of individuals. Moreover, this descent tree has a smaller number of individuals than the original pedigree. Hence, these subpedigrees simplify the complexity of the original pedigree and they are fit for linkage analysis.

To complete this setion, we give an example by showing all shortest paths Steiner trees and minimal Steiner trees in a complex pedigree. In 2003, Lamont et al. [31] gave a complete pedigree (see Figure 2.4) to study the Bowen–Conradi syndrome (BCS). As you can see, it is a very complex pedigree. To simplify this pedigree for linkage analysis, the authors computed an all shortest paths pedigree with respect to the terminal set and extracted a minimal Steiner tree via Miniped. For illustration, we adopt those results here. Figure 2.5 shows a minimal Steiner tree with respect to one least common ancestor, and Figure 2.6 shows an all shortest path Steiner tree of the same least common ancestor couple.

2.2 PedHunter and Miniped

In 1998, Agarwala et al. [2] developed software called PedHunter to analyze a complex pedigree based on the Steiner tree problem in graph theory. They applied a branch

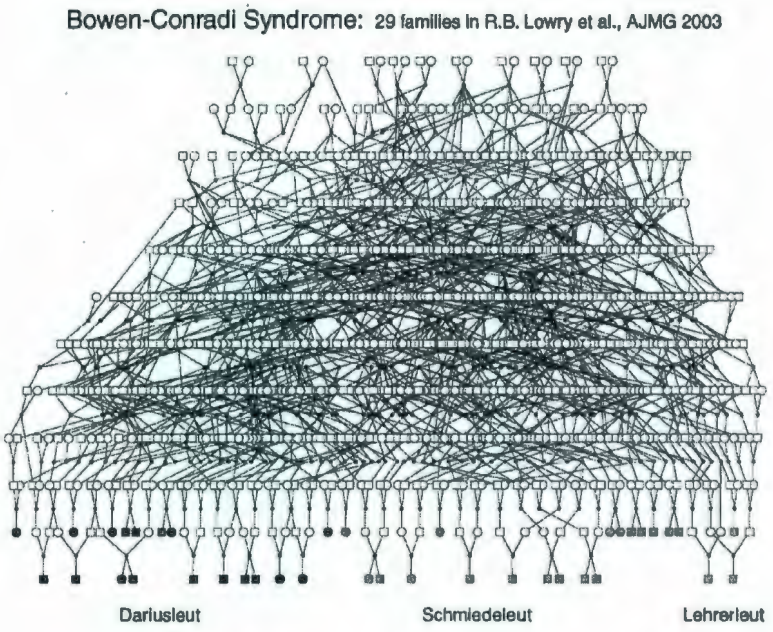


Figure 2.4: The complete pedigree for studying Bowen–Conradi syndrome. Affected individual are shown in color by subdivision on leut (red = Dariusleut, blue = Schmiedeleut, green = Lehrerleut). Source: Lowry et al. [38].

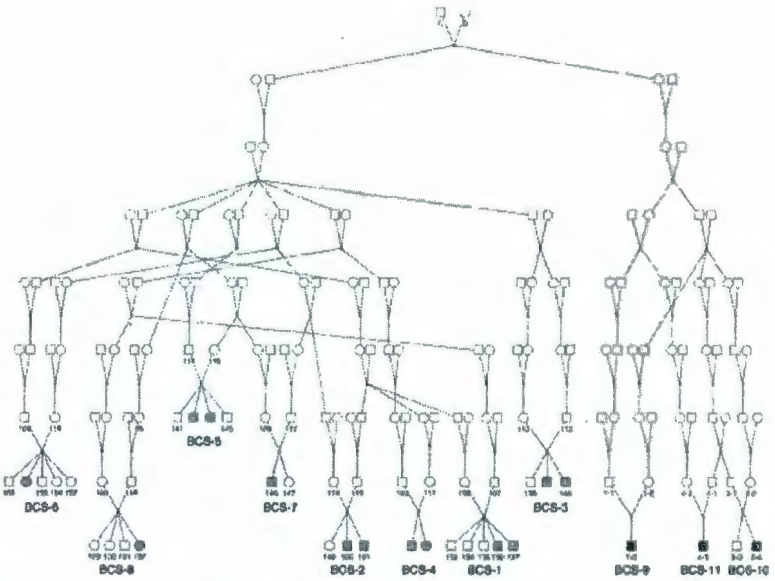


Figure 2.5: A minimal Steiner tree of affected individuals for whom there was genotype data. Source: Lamont et al. [31].

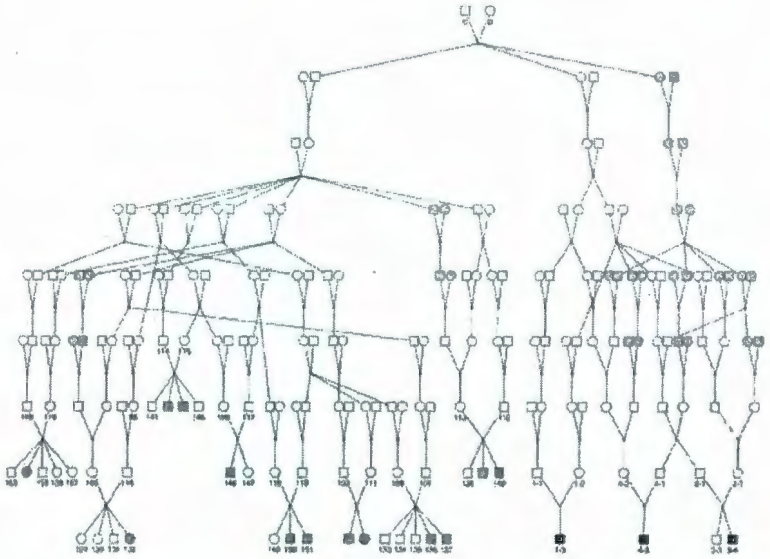


Figure 2.6: An all shortest path Steiner tree. Yellow symbols indicate individuals who are not in the minimal Steiner tree (see Figure 2.5). Source: Lamont et al. [31].

and bound algorithm to obtain Steiner trees and all shortest paths Steiner trees from an original large pedigree graph.

The branch and bound algorithm is by far the most widely used technique for finding optimal solutions of various combinatorial optimization problems. Initially, the algorithm searches a solution space to find an initial solution as the current best solution. Then, the solution space is split into two or more smaller sets whose union covers the solution space. For each of the smaller sets, it is checked whether the smaller set consists of a solution, in which case it is compared to the current best solution. If there is no solution in the smaller set, the bounding function for the smaller set will be calculated and compared to the current best solution. The recursion stops when the current solution is at least as good as any other solution. In the worst case, computation time of the algorithm is exponential in the number of edges of the input pedigree graph.

PedHunter has been applied to reduce the complexity of a pedigree graph for genetic analysis. We will illustrate some applications of PedHunter in the next section.

In 2004, Löschner et al. [37] also developed a software package called Miniped for finding the Steiner trees and all shortest paths Steiner trees from an original large complex pedigree. Miniped provides a fast heuristic to obtain exact solutions on all shortest paths pedigrees as well as approximate Steiner trees on the entire pedigree graph. The algorithm used in Miniped requires a small number of branching of the searched trees, so it is able to handle larger and more complex input pedigrees than PedHunter in much shorter time. The upper bound of the algorithm in Miniped for finding all shortest paths Steiner trees is $O(2^{|V|} \times |X| \times |E|)$. Table 2.1 shows the comparison between some results from Miniped and those from PedHunter [37].

	All shortest paths Steiner tree		
	$\|V\|$	$\|Steiner\|$	Time M/P
1	146	95	0/67
2	101	78	0/0
3	148	105	0/0
4	141	86	0/0
5	142	95	0/90
6	135	80	0/12700
7	160	86	0/n.a.
8	153	86	0/13
9	141	80	0/22700
10	158	88	0/>1 day
11	142	89	0/30800

Table 2.1: Miniped versus PedHunter. Column 2 and 3 show the numbers of vertices in the starting graph and the Steiner tree, respectively. Column 4 shows the computation times with Miniped (M) versus PedHunter (P) in seconds. Source: Löschner et al. [37].

2.3 Applications of algorithms to reduce the complexity of pedigrees in linkage analysis

To extract the full available information from both parametric and nonparametric linkage analysis, geneticists increase the number of loci and the number of alleles per locus. In this situation, the slow running times become more severe. Therefore, simplifying the pedigree is necessary so that linkage analysis computations can be carried out within a reasonable amount of time. In fact, simplifying the pedigree is a very efficient tool for linkage analysis. In this section, we will give some applications of algorithms to reduce the complexity of pedigrees in linkage analysis which also

serve as the motivation of this thesis.

To study the disorder Amish microcephaly (MCPHA), Rosenberg et al. [47] found an all shortest paths subpedigree using the software PedHunter in the Amish Genealogy Database. They concluded that MCPHA must be associated with another genetic locus and localized the gene MCPHA to the region of 3 cM or 2 Mb on chromosome 17q25. In 2000, Johnston et al. [26] implemented linkage analysis of nemaline myopathies (NM) among the Old Order Amish. In their study, 33 nuclear families were connected, and all shortest paths Steiner pedigrees were obtained by PedHunter. They identified an ~2 cM interval on chromosome region 19q13.4 that was homozygous in all affected individuals. They also found the Amish nemaline myopathy (the unique form of NM observed in the Old Order Amish) is inherited in an autosomal recessive pattern.

2.3.1 Bowen–Conradi syndrome

Bowen–Conradi syndrome is a very rare inherited lethal autosomal recessive disorder, and it is characterized by intrauterine growth retardation, severe psychomotor retardation, micrognathia, microcephaly, flexion contractures, rockerbottom feet and low birth weight. The average life span of the affected infants is 13 months. Most affected infants died early within the first 2 years of life. Those children who survived longer showed extreme growth failure in height, weight and head circumference. The frequency of BCS in the Hutterite population is estimated to be 1 in 355 live births [38].

The Hutterite population is a genetically isolated population living on the North American prairies since the late 1800s [22]. The Hutterites immigrated to South Dakota in 1870s and formed three different branches of Hutterites: the Schmiedeleut, the Lehrerleut, and the Dariusleut. Approximately 40,000 descendents are from 89 founders [42]. About 28,000 Hutterites live in Canada and about 12,000 live in the USA. The Hutterite population provides advantages for identification of a number of

genes. Over 30 autosomal recessive genetic disorders were recognized in this population.

In 2005, Lamont et al. [31] studied the genetic mapping of the Bowen–Conradi syndrome (BCS) through extracting less complex pedigrees, such as all shortest paths Steiner pedigrees. In the study, 14 BCS patients in nine nuclear families were diagnosed. In order to estimate the marker allele frequencies and define a disease–associated haplotype, blood samples were collected from 42 individuals including all 18 parents, eight BCS patients, one grandparent and 15 unaffected children.

Genealogical relationships among the families were used to increase the information for linkage analysis. From a genealogical database of over 38,000 individuals, a pedigree of 647 individuals was obtained. In order to do two point linkage analysis on chromosomes 12 and 16, 17 and 13 markers, respectively, these 11 nuclear families were genotyped for the genome–scan markers.

However, a complex pedigree of 647 individuals was computationally intractable. Firstly, 16 least common ancestors were identified by PedHunter. Moreover, the pedigree containing the all shortest paths between the least common ancestors and the set of 22 parents were obtained. The result showed that there were 11 all shortest paths pedigrees that ranged from 196 to 279 individuals in size. In order to further reduce complexity, a minimal pedigree for the 11 all shortest paths pedigrees was obtained using PedHunter and Miniped, although, PedHunter was unable to handle the subpedigrees with more than 10 terminals. Figure 2.6 and Figure 2.5 show an all shortest path Steiner tree and a minimal Steiner tree for the same common ancestors, respectively.

Linkage analysis of the minimal pedigrees gave significant evidence for linkage to three adjacent chromosome 12 markers (D12S374, D12S391, and GATA91H01) while there was no significant evidence for linkage to markers on chromosome 16. Table 2.2 shows LOD scores at zero recombination between BCS locus and D12S391 for all the Steiner tree pedigrees and all shortest paths pedigrees.

At least common ancestors	$\|V\|$ of ASP pedigrees	$\|V\|$ of ST pedigrees	LOD scores at the marker position		
			D12S374	D12S391	GATA91H01
A or B	263	199	2.27	4.15	2.75
C or D	196	166	2.99	4.97	2.38
E or F	246	171	3.35	4.44	1.93
G or H	239	171	2.27	4.38	2.60
I or J	272	185	2.58	4.85	3.14
K	254	190	3.15	4.65	2.87
L	276	189	2.47	4.71	3.28
M	263	201	2.64	3.82	2.48
N	279	220	2.81	3.27	3.43
O or P	261/277	183	2.63	4.63	2.23

Table 2.2: LOD scores at the marker position for all the minimal Steiner tree pedigrees (ST pedigrees) and all shortest paths pedigrees (ASP pedigrees). $|V|$ denotes the number of people in a pedigree.

Lamont et al. [31] also reported the mapping of the BCS locus in the Hutterite population to a 3.5 cM segment in chromosome region 12p13.3, under the assumption that all Hutterite BCS patients were homozygous for the same mutation inherited from a common ancestor.

Chapter 3

Optimal descent trees conditional on the observed data

In this chapter, we will develop a new algorithm to find an optimal subpedigree from a large genealogy. This new algorithm has an additional benefit in that it uses more information from the given data. The improvement allows us to use more information from those affected individuals. In contrast to PedHunter and Miniped, where all edges have the same weights, the directed weighted graph in our algorithm has a different weight for each edge. Through assigning a different weight to each edge of a pedigree graph in our algorithm, we can get a more reasonable subpedigree using a Steiner tree algorithm.

Our aim is to find an optimal subpedigree from an original large complex genealogy to reduce the complexity of algorithms for genetic analysis. However, two questions exist under this aim: (1) How to determine which subpedigree from the solutions is best? (2) What is the standard for the best subpedigree used in genetic analysis?

In practice, we do not know the paths of the disease–causing allele that flows down from an original carrier to the affected individuals. This implies that the actual tree in which the disease–causing allele passes down from a founder to the affected individuals is not known. Furthermore, in a large genealogy, such as the Old Order

Amish, Hutterites or Iceland, more than one common ancestor who qualifies very well as the original carrier can be found. Therefore, we may not even know which founder is the actual original carrier.

In a pedigree graph, some individuals and edges would be much more likely in the subpedigree than others. For example, if an edge has a much larger portion in all possible descent trees than another alternative edge, then it is certainly preferable in the subpedigree. In other words, it is more reasonable to choose a descent tree that contains the paths in which the disease–causing allele most likely passes from a common ancestor to the affected individuals.

In order to find the most likely descent tree, we want to assign different weights to the edges in an original pedigree graph. Each edge is weighted by the probability of this edge being in all possible descent trees. Therefore, the probability of each descent tree is a product of its edge weights. A most likely descent tree is a tree that has a maximized probability over the set of all possible descent trees. Now, it is easy to show that this problem is equivalent to the following combinational optimization problem:

Let $\Gamma(r, X)$ denote the set of all descent trees with the root r and the terminal set X. Each edge e in the pedigree graph is assigned a weight $w(e) \in [0, 1]$. We try to find an optimal tree $T_{opt}^{r,X}$ such that

$$P_r(T_{opt}^{r,X}) = \arg\max_{T \in \Gamma(r,X)} \prod_{e \in T} w(e). \tag{3.1}$$

In this combinational optimization problem, we will meet two problems. One is how to find the weight of every edge in an original pedigree graph. In other words, we should know how to calculate the probability of each edge in all the possible descent trees. The other problem is how to solve the optimization problem given the weight of each edge in an original pedigree graph.

Firstly, we will give a method to determine the weight of each edge in an original pedigree graph. As a concrete example, let us consider the original pedigree graph

with the affected individuals 15 and 16 in Figure 2.2(b). Figure 3.1 shows the seven possible descent trees of Figure 2.2(b).

Given the affected individuals, what is the probability of an existing possible descent tree? Now, let us calculate the probability of each tree in the seven possible descent trees given the affected individuals 15 and 16 in Figure 3.1. Firstly, let us calculate the probability of tree T_1 given the affected individuals 15 and 16, denoted by $P_r(T_1|\{15,16\})$. Given the individual 15 in the descent tree T_1, then edge $e_{5,15}$ could not be in T_1 since individual 5 is not a common ancestor. That means there is no path from individual 5 to the other affected individual 16. Therefore, edge $e_{9,15}$ has a probability of 1 in tree T_1. Since a gene is inherited from the father or the mother with the same probabilities, then individual 9 may receive the gene from his mother or his father. So we deduce that edge $e_{3,9}$ has a probability of $\frac{1}{2}$ in tree T_1. Next, note that there exists only one path from individual 3 to the other affected individual 16. That implies each of these edges $e_{3,10}$, $e_{10,13}$ and $e_{13,16}$ has a probability of 1 in tree T_1. Thus, the probability of tree T_1 given the individual 15 is equal to $\frac{1}{2} \cdot (1)^4 = \frac{1}{2}$. Using the same argument, we can get the probability of T_1 given the individual 16 is equal to $(\frac{1}{2})^3$. Then

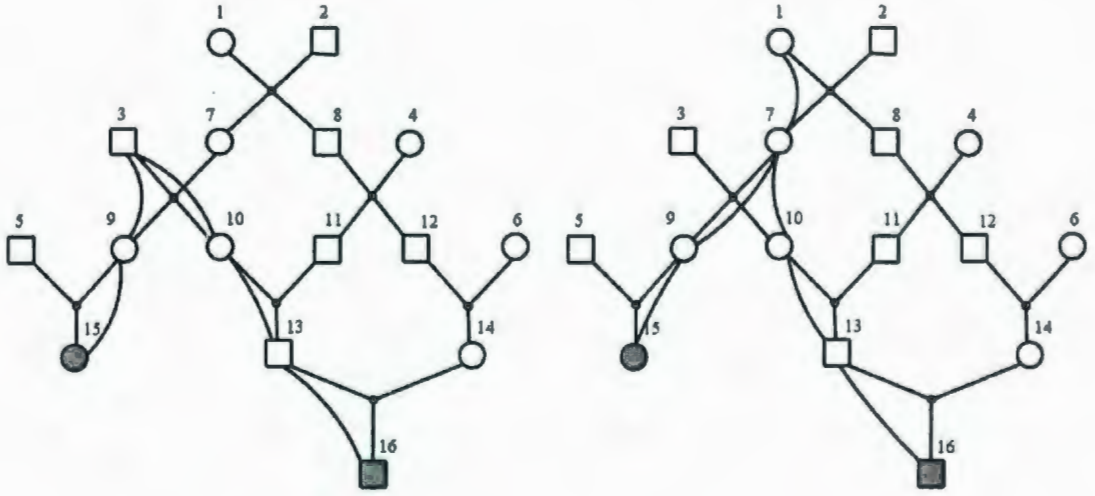
$$P_r(T_1|\{15,16\}) = \frac{1}{2}(\frac{1}{2}) + \frac{1}{2}(\frac{1}{2})^3 = \frac{5}{16}.$$

In the similar manner, we can obtain the following probabilities:

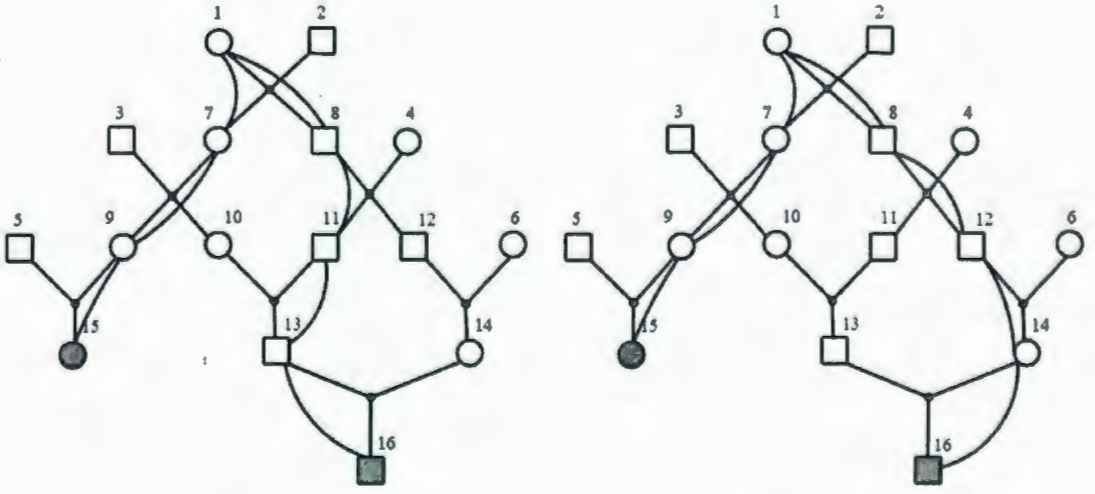
$$P_r(T_2|\{15,16\}) = p(T_5|\{15,16\}) = (\frac{1}{2})^4 + (\frac{1}{2})^5 = \frac{3}{32}.$$

$$P_r(T_3|\{15,16\}) = p(T_6|\{15,16\}) = (\frac{1}{2})^5 + (\frac{1}{2})^4 = \frac{3}{32}.$$

$$P_r(T_4|\{15,16\}) = p(T_7|\{15,16\}) = (\frac{1}{2})^5 + (\frac{1}{2})^3 = \frac{5}{32}.$$



(a) T_1 (b) T_2 or T_5



(c) T_3 or T_6 (d) T_4 or T_7

Figure 3.1: The seven possible descent trees of Figure 2.2(b). In (b)–(d), six descent trees are presented. Three descent trees T_2, T_3 and T_4 are exactly shown in (b)–(d). Other three descent trees, T_5, T_6 and T_7, can be obtained through replacing the root, individual 1 by individual 2 in trees T_2, T_3 and T_4, respectively.

3.1 An algorithm to obtain the weight of each edge in a pedigree graph

In this section, we will give a Monte Carlo algorithm to calculate (or properly said to estimate with pre-specified degree of precision) the weight for every edge in an original pedigree graph. Let R be the set of common ancestors and I be the set of individuals who are founders but not common ancestors. For example, in Figure 3.1, $R = \{1, 2, 3\}$ and $I = \{4, 5, 6\}$. For any individual v, we use $P_1(v)$ to denote the mother of v and $P_2(v)$ to denote the father of v. $c(v)$ denotes the set of all children of the individual v. V denotes the set of all individuals in a pedigree graph and let $M = V - X$, where X is the terminal set.

Algorithm 1

Step 1: Initialize by letting $c'(v) = 0$ and $n(vt) = 0$ for any individual v and any terminal t.

Step 2: Pick up one terminal t from X, and set $X = X \backslash \{t\}$.

Step 3: Pick up an individual v from M, and let $M = M \backslash \{v\}$.

Step 4: Pick up a child u from $c(v)$, and set $c(v) = c(v) \backslash \{u\}$.

Step 5: If there exists a path from u to t, then let $n(vt) = n(vt) + 1$ and $c'(v) = c'(v) \cup \{u\}$.

Step 6: If $c(v) \neq \emptyset$, return to Step 4.

Else if $M \neq \emptyset$, return to Step 2.

Else if $X \neq \emptyset$, return to Step 1.

Else stop.

The purpose of the algorithm 1 is to get the values of $n(vt)$ and $c'(v)$. These two values will be used in our main algorithms.

Main algorithms (Calculating the weight of each edge)

Step 1: Randomly pick up one terminal from X, and denote a as this chosen terminal. Let $X = X \backslash \{a\}$ and $X' = X \backslash \{a\}$.

Step 2: Initialize by letting $v = a$ and $S_0^{(a)} = \{v\}$, $i = 1$, $P = \emptyset$.

Step 3: **If** $P_1(v) \in I$ or $P_2(v) \in I$, suppose $P_1(v) \in I$, then $w(e^{(a)}_{v,P_2(v)}) = 1$.

Let $S_i^{(a)} = S_j^{(a)} \cup \{P_2(v)\}$ with $a \in S_j^{(a)}$, $j = 0, \ldots, i-1, i = i+1$.

Set $P = P \cup \{P_2(v)\}$.

Else $w(e^{(a)}_{v,P_1(v)}) = w(e^{(a)}_{v,P_2(v)}) = \frac{1}{2}$.

Then let $S_i^{(a)} = S_j^{(a)} \cup \{P_1(v)\}$ with $a \in S_j^{(a)}$, $j = 0, \ldots, i-1, i = i+1$.

Let $S_i^{(a)} = S_j^{(a)} \cup \{P_2(v)\}$ with $a \in S_j^{(a)}$, $j = 0, \ldots, i-1, i = i+1$.

Set $P = P \cup \{P_1(v), P_2(v)\}$.

Step 4: If $P \neq \emptyset$, then pick up one element from P.

Let u denote this element and $P = P \backslash \{u\}$.

If $u \in R$, return to Step 4.

Else set $u = v$ and return to Step 2.

Else go to Step 5.

Step 5: Randomly pick up one terminal from X'. Let t denote this terminal and $X' = X' \backslash \{t\}$.

Step 6: Pick up a root from R, and let m' denote this root. Set $R = R \backslash \{m'\}$ and $B_{0,t}^{(a)} = \{m'\}$, $m = m'$, $h = 1$, $Q = \emptyset$.

Step 7: Pick up a child q from $c'(m)$. Let $c'(m) = c'(m) \backslash \{q\}$, $w(e^{(a)}_{m,q,t}) = \frac{1}{n(mt)}$, $B_{h,t}^{(a)} = B_{j,t}^{(a)} \cup \{q\}$, $m' \in B_{j,t}^{(a)}$, $j = 0, \ldots, h-1, h = h+1$ and $Q = Q \cup \{q\}$.

Step 8: If $c'(m) \neq \emptyset$, then return to Step 7.

Else go to Step 9.

Step 9: If $Q \cap \{t\} \neq \{t\}$, then pick up one element $p \neq t$ from Q.

Let $Q = Q \backslash \{p\}$, $m = p$ and return to Step 7.

Else If $R \neq \emptyset$, then return to Step 6.

Else If $X' \neq \emptyset$, then return to Step 5.

Else If $X \neq \emptyset$, then return to Step 1.

Else output $S_i^{(a)}$, $B_{i,t}^{(a)}$ and $w(e_{v,u}^{(a)})$.

From the algorithms, for a root r and a terminal t, we can obtain the values of

$S_i^{(a)}$, $B_{i,t}^{(a)}$ and $w(e_{v,u}^{(a)})$, where u and v are two individuals, and a is a terminal that has been picked up in Step 1 of the Main algorithms. Before giving a formula to calculate the probability of every edge, we firstly introduce some notations. Let

$N_{uv,r}^{(a)}$ be the number of $S_i^{(a)}$ satisfying $uv, r \in S_i^{(a)}$,

$N_{r,t}^{(a)}$ be the number of $B_{i,t}^{(a)}$ satisfying $t, r \in B_{i,t}^{(a)}$,

$n_{uv,r,t}^{(a)}$ be the number of $B_{i,t}^{(a)}$ satisfying $uv, t, r \in B_{i,t}^{(a)}$,

and $N_r^{(a)}$ be the number of $S_i^{(a)}$ satisfying $r \in S_i^{(a)}$.

For any edge $e_{u,v}$, u is the parent of v, we denote the probability of edge $e_{u,v}$ in all possible descent trees as $w(e_{u,v})$. Now, we give the formula to calculate $w(e_{u,v})$.

$$w(e_{u,v}) = \sum_{a \in T} \{w(e_{v,u}^{(a)}) \times [\sum_{r \in R} (N_{vu,r}^{(a)} \times \prod_{t \in T-\{a\}} N_{r,t}^{(a)})]$$

$$+ \sum_{t \in T-\{a\}} (w(e_{uv,t}^{(a)}) [\sum_{r \in R} (n_{uv,r,t}^{(a)} \times N_r^{(a)} \times \prod_{t' \in T-\{a,t\}} N_{r,t'}^{(a)})])\}. \qquad (3.2)$$

In the concrete example in Figure 3.1, we can get the weights of all edges in a pedigree graph using the formula (3.2) as follows:

$$w(e_{15,9}) = \frac{1\times(1+3+3)+1\times(1+3+3)}{2\times 7} = 1.$$
$$w(e_{15,5}) = \frac{0}{2\times 7} = 0.$$
$$w(e_{9,3}) = \frac{\frac{1}{2}\times 1+1\times 1}{2\times 7} = \frac{3}{28}.$$
$$w(e_{9,7}) = \frac{\frac{1}{2}\times(3+3)+1\times(3+3)}{2\times 7} = \frac{9}{14}.$$
$$w(e_{10,3}) = \frac{1\times 1+\frac{1}{2}\times 1}{2\times 7} = \frac{3}{28}.$$

In the similar way, we can calculate the weights of the remaining. They turn out to be

$$w(e_{13,11}) = w(e_{12,8}) = w(e_{11,8}) = w(e_{10,7}) = w(e_{16,14}) = \tfrac{3}{14}.$$
$$w(e_{8,1}) = w(e_{14,12}) = w(e_{8,2}) = \tfrac{2}{7}.$$
$$w(e_{12,4}) = w(e_{11,4}) = w(e_{14,6}) = 0.$$
$$w(e_{7,1}) = w(e_{7,2}) = \tfrac{11}{14}.$$
$$w(e_{16,13}) = \tfrac{15}{28}. \qquad w(e_{13,10}) = \tfrac{9}{28}.$$

From these results, we can see that $w(e_{15,9}) = 1$, meaning that edge $e_{15,9}$ has a probability of 1 in all possible descent trees. In other words, any possible descent tree must contain edge $e_{15,9}$. Again, since edge $e_{9,3}$ has a smaller weight than edge $e_{9,7}$, then edge $e_{9,7}$ is more possible in an optimal descent tree than edge $e_{9,3}$.

3.2 An approximation algorithm to find optimal trees given a set of terminals and a root

In this section, we will focus on solving the optimization problem (3.1) for a pedigree graph with the given edge weights. This optimization problem (3.1) will be reformulated into the Steiner tree problem through a transformation. Firstly, we take the logarithm of the weight of each edge. Since the weight of each edge is greater than zero, let us define $\log(0) = \lim_{x \to 0} \log(x) = -\infty$. Hence we can minimize the sum of the negative value of the logarithm instead of maximizing the product in problem (3.1). Now, the optimization problem (3.1) is equivalent to the following one:

$$P_r(T_{opt}^{r,X}) = \min_{T \in \Gamma(r,X)} \sum_{e \in T} -\log w(e). \tag{3.3}$$

Obviously, optimization problem (3.3) is a directed Steiner tree problem in graph theory. We can see that $-\log w(e)$ is the cost of edge e in the directed Steiner tree

problem. Thus we can solve this optimization problem using a directed graph Steiner tree algorithm. By now, just a few algorithms for the directed Steiner tree problem have been developed. Fortunately, we can use the approximation algorithm provided by Charikar et al. [11] to solve the directed Steiner tree problem for our purpose. The time complexity of this approximation algorithm is $O(|V|^i \cdot |X|^{2i})$, where i is the number of generations in a pedigree graph. In what follows, we will describe the main idea of this approximation algorithm.

Without loss of generality, we can assume that for every pair of vertices u and v, there exists an edge $e_{u,v}$ with a cost that equals to the shortest path distance from u to v in an original pedigree graph. We call such a graph as a *transitive* graph. Let $c(T)$ denote the cost of a tree T and $k(T)$ denote the number of terminals in T.

Definition 3.2.1 *The density of a tree T, denoted by $d(T)$, is the ratio of the cost of the tree to the number of terminals in T. In other words, $d(T) = c(T)/k(T)$.*

The density of a tree can be interpreted as the average cost of connecting a terminal to the root.

Definition 3.2.2 *An l–lever tree is a tree where no leaf is more than l edges away from the root, where l is a positive integer.*

From the Definition 3.2.2, we can see that the pedigree graph with i generations is an i–lever tree.

Approximation algorithm (Finding a Steiner tree in a pedigree graph.) $A_i(k, r, X)$
Output: A tree T rooted at r spanning $k(T)$ terminals which has the minimum cost in a pedigree graph.

Step 1. $T \leftarrow \emptyset$.

Step 2. While $k > 0$

(a) $T_{BEST} \leftarrow \emptyset$;

(b) For each vertex $v \in V$, and each k', $1 \leq k' \leq k$

(1) $T' \leftarrow A_{i-1}(k', v, X) \cup \{(r, v)\}$;

(2) If $d(T_{BEST}) > d(T')$ then $T_{BEST} \leftarrow T'$.

(c) $T \leftarrow T \cup T_{BEST}$; $k \leftarrow k - |X \cap V(T_{BEST})|$; $X \leftarrow X - V(T_{BEST})$.

Step 3. Return T.

$A_1(k, v, X)$ is to find k terminals which are closest to the root and connect them to the root using shortest paths. $A_i(k, v, X)$ repeatedly finds a vertex v and a number k' with $1 \leq k' < k$ such that the density of tree $T_{i-1}(k', v, X) \cup \{r, v\}$ is the least among all trees of this form.

3.3 An example to execute the approximation algorithm

In this section, we will execute the approximation algorithm to find Steiner trees in a directed weighted pedigree graph. As a result, a Steiner tree will be found through the execution of the approximation algorithm in Figure 2.2(a) with the affected individuals 15 and 16. Let us use W to denote the costs of edges in Problem (3.3). Therefore, we can get all the costs of edges in Figure 2.2(a) as follows:

$$W(e_{13,11}) = W(e_{12,8}) = W(e_{11,8}) = W(e_{10,7}) = W(e_{16,14}) = -\log \tfrac{3}{14}.$$
$$W(e_{8,1}) = W(e_{14,12}) = W(e_{8,2}) = -\log \tfrac{2}{7}.$$
$$W(e_{12,4}) = W(e_{11,4}) = W(e_{14,6}) = -\log 0 = \infty.$$
$$W(e_{7,1}) = W(e_{7,2}) = -\log \tfrac{11}{14}.$$
$$W(e_{16,13}) = -\log \tfrac{15}{28}. \qquad W(e_{13,10}) = -\log \tfrac{9}{28}.$$

In the approximation algorithm, we have an assumption that every graph is a transitive graph. Thus, we firstly find the transitive graph corresponding to Figure 2.2(a).

3.3.1 The transitive graph corresponding to Figure 2.2(a)

To construct the transitive graph corresponding to Figure 2.2(a), for every pair of vertices u and v, we need to obtain a cost that equals the shortest path distance

between u and v in the original pedigree graph. For example, for the vertices 1 and 13, there exist two paths P_1 $(1 \to 8 \to 11 \to 13)$ and P_2 $(1 \to 7 \to 10 \to 13)$. We then calculate the costs of these two paths as follows:

The cost of path P_1 is equal to

$$W(e_{13,11}) + W(e_{11,8}) + W(e_{8,1}) = -\log\frac{3}{14} - \log\frac{3}{14} - \log\frac{2}{7} = -\log\frac{9}{686}.$$

The cost of path P_2 is equal to

$$W(e_{13,10}) + W(e_{10,7}) + W(e_{7,1}) = -\log\frac{9}{28} - \log\frac{3}{14} - \log\frac{11}{14} = -\log\frac{297}{5488}.$$

Since $-\log\frac{297}{5488} < -\log\frac{9}{686}$, we have $W(e_{13,1}) = -\log\frac{297}{5488}$. In the similar way, we can obtain the costs of other pairs of vertices. They turn out to be

$W(e_{9,1}) = W(e_{9,2}) = -\log\frac{99}{196}$. $W(e_{10,1}) = W(e_{10,2}) = -\log\frac{33}{196}$.

$W(e_{11,1}) = W(e_{11,2}) = -\log\frac{3}{49}$. $W(e_{12,1}) = W(e_{12,2}) = -\log\frac{3}{49}$.

$W(e_{13,1}) = W(e_{13,2}) = -\log\frac{297}{5488}$. $W(e_{14,1}) = W(e_{14,2}) = -\log\frac{6}{343}$.

$W(e_{15,1}) = W(e_{15,2}) = -\log\frac{99}{196}$. $W(e_{16,1}) = W(e_{16,2}) = -\log\frac{4455}{153664}$.

$W(e_{13,3}) = -\log\frac{27}{784}$. $W(e_{15,3}) = -\log\frac{3}{28}$. $W(e_{16,3}) = -\log\frac{405}{21951}$.

$W(e_{15,7}) = -\log\frac{9}{14}$. $W(e_{13,7}) = -\log\frac{27}{392}$. $W(e_{16,7}) = -\log\frac{405}{10976}$.

$W(e_{13,8}) = -\log\frac{9}{196}$. $W(e_{14,8}) = -\log\frac{3}{49}$. $W(e_{16,8}) = -\log\frac{135}{5488}$.

$W(e_{16,10}) = -\log\frac{135}{784}$. $W(e_{16,11}) = -\log\frac{45}{392}$. $W(e_{16,12}) = -\log\frac{3}{49}$.

$W(e_{16,4}) = W(e_{13,4}) = W(e_{16,4}) = W(e_{16,6}) = \infty$.

Figure 3.2 shows the transitive graph corresponding to Figure 2.2(a) with affected individuals 15 and 16.

After finding the transitive graph, we then prune the transitive graph to decrease the complexity of the transitive graph.

3.3.2 The pruned pedigree graph

In order to decrease the complexity of the transitive graph, we will remove as many vertices and edges as possible without changing the size of a Steiner tree in the original graph.

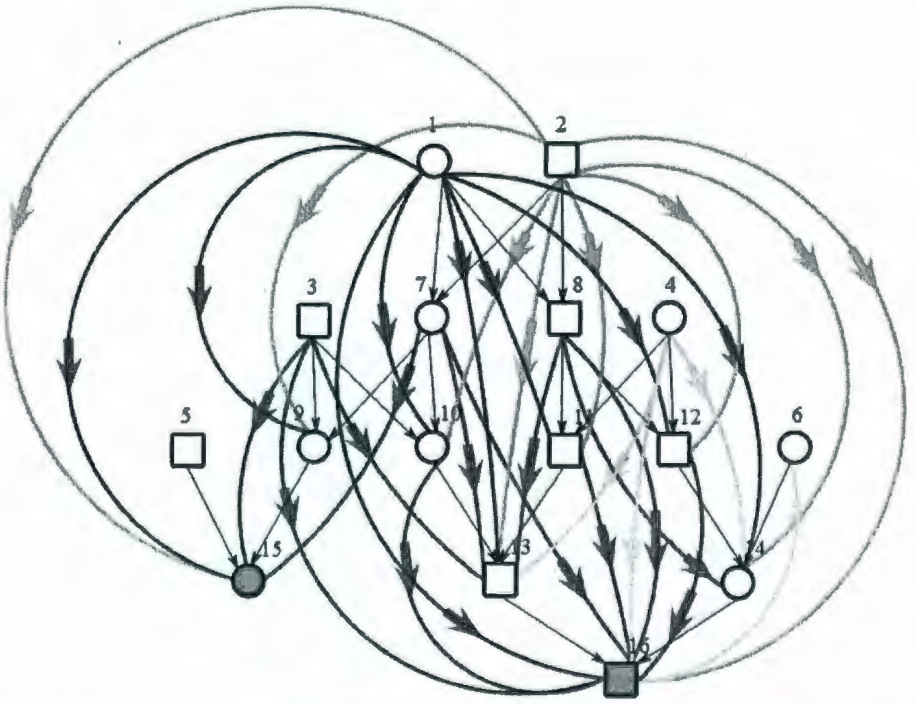


Figure 3.2: The transitive graph corresponding to Figure 2.2(a). All edges are added into Figure 2.2(a) by different colors. The edges from different generations are given different colors. Some edges that will be pruned are distinguished by different colors.

Firstly, since individual 1 and individual 2 are founders and they are married, so if there is a Steiner tree T rooted at individual 1, then tree T' obtained by replacing the individual 1 by individual 2 in T is also a Steiner tree. Thus, all the edges from vertex 2 to any other vertex can be removed from the transitive graph. That means all green edges can be removed from Figure 3.2. Secondly, when we execute the approximation algorithm, we never choose any edge with a cost of ∞ since any tree which consists of this kind of edge is not a Steiner tree. For example, in Figure 3.2, edges $e_{14,4}$, $e_{13,4}$, $e_{16,4}$ and $e_{16,6}$ have a cost of ∞, then we delete these four edges from the transitive graph. Therefore, we remove all yellow edges from Figure 3.2. Figure 3.3 shows the final pruned transitive graph in which the approximation algorithm will be executed.

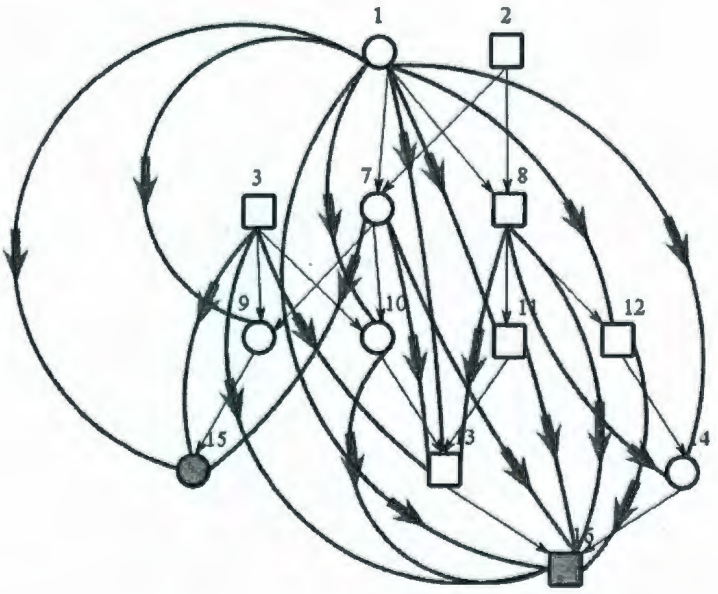


Figure 3.3: The final pruned transitive graph.

3.3.3 The implementation of the approximation algorithm in the pruned transitive graph

The approximation algorithm $A_i(k, r, X)$ mentioned in Section 3.2 outputs a tree T rooted at r spanning terminals X which has the minimum cost. In Figure 3.3, the approximation algorithm $A_i(2, 1, \{15, 16\})$ will give a Steiner tree rooted at individual 1 with the terminals $\{15, 16\}$.

- For every vertex and every k', $0 < k' < 3$, the Step 2 (b) of the approximation algorithm is iterated until a tree T_{BEST} is presented. After the implementation of the process in Figure 3.3, we have that $d(T_{BEST}) = -\log \frac{99}{196}$. Figure 3.4(a) shows the T_{BEST} obtained in this process.

- After the iteration (b) in Step 2, the algorithm moves to (c) of Step 2. Then, the approximation algorithm $A_i(1, 1, \{16\})$ will be executed. After the implementation of the algorithm $A_i(1, 1, \{16\})$, we have that $d(T_{BEST}) = -\log \frac{4455}{153664}$. Figure 3.4(b) shows the T_{BEST} obtained in this process.

- After the implementation of the whole approximation algorithm, we obtain a Steiner tree rooted at individual 1 spanning the terminals {15, 16}. The Steiner tree obtained from the algorithm is shown in Figure 3.1(b).

By these steps, we get the Steiner tree T_2 or T_5 (Figure 3.1(b)) which is rooted at individual 1 with affected individuals 15 and 16. In addition, there exists exactly only one tree T_1 (Figure 3.1(a)) rooted at individual 3 with affected individuals 15 and 16. At last, let us compare the costs of these two trees with different roots.

The cost of tree T_1 is equal to

$$\begin{aligned} W(T_1) &= W(e_{15,9}) + W(e_{9,3}) + W(e_{10,3}) + W(e_{13,10}) + W(e_{16,13}) \\ &= -\log 1 - \log \frac{3}{28} - \log \frac{3}{28} - \log \frac{9}{28} - \log \frac{15}{28} = -\log \frac{1215}{614656}. \end{aligned}$$

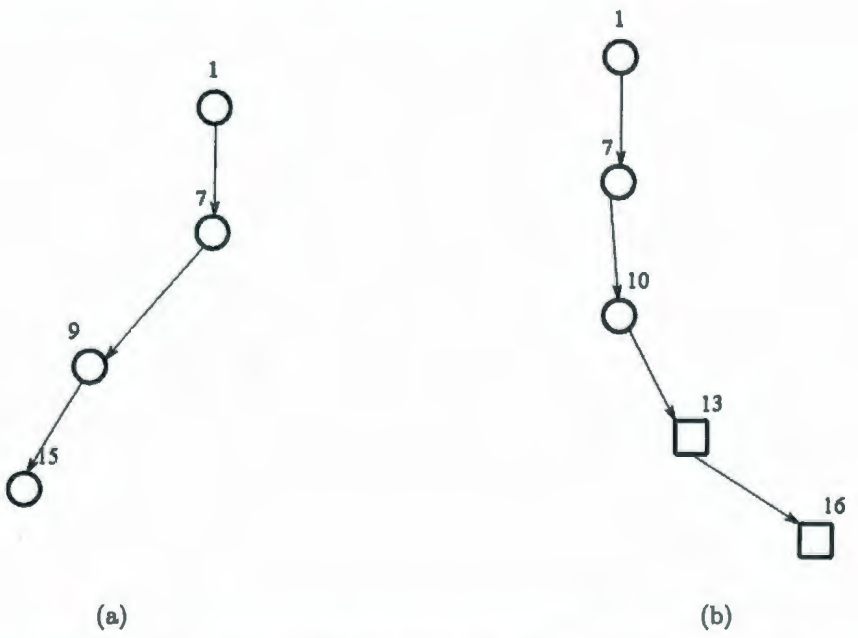


Figure 3.4: T_{BEST} obtained from the approximation algorithm.

The cost of tree T_2 or tree T_5 is equal to

$$\begin{aligned} W(T_2 \ \ or \ \ T_5) &= W(e_{15,9}) + W(e_{9,7}) + W(e_{7,1}) + W(e_{10,7}) + W(e_{13,10}) + W(e_{16,13}) \\ &= -\log 1 - \log \frac{9}{14} - \log \frac{11}{14} - \log \frac{3}{14} - \log \frac{9}{28} - \log \frac{15}{28} \\ &= -\log \frac{40095}{2151296}. \end{aligned}$$

Since $-\log \frac{1215}{614656} > -\log \frac{40095}{2151296}$, tree T_1 has more costs than those for tree T_2 or tree T_5.

Chapter 4

Conclusions and future work

This chapter presents the main conclusions of this thesis and the possible future work. First, we summarize the main results. Then we suggest some future research directions.

4.1 Research summary

A powerful approach for identifying genes is to study genetically isolated populations. Well characterized isolates provide excellent study samples for linkage mapping. The utility of population isolates eliminates the possible confounding effects of genetic background and the impacts of environmental heterogeneity. Therefore, we focused our study on developing a new approach from a large and complex genealogy. The new approach is to maximize the genealogical information for linkage analysis while minimizing the computational burden. In this thesis, we investigated an approach to construct an optimal subpedigree from an original large pedigree based on the Steiner tree problem in graph theory. Firstly, we presented an algorithm to find the probability of each edge among all possible descent trees. Then we assigned the weight of the probability obtained from this algorithm to each edge. Last, we provided an approximation algorithm to solve the directed Steiner tree problem. The results in

this thesis can be used to find a most likely tree through the information that the affected individuals are known.

4.2 Future work

Strewn throughout this thesis there are many natural questions which arise and warrant consideration. In this section, we discuss these possible directions.

4.2.1 Using more information from the observed data

The observed data from a complete large genealogy provide information about the genetic relatedness of individuals in a pedigree, their medical records, the pattern of inheritance of a genetic disorder and the number of meiotic steps separating them. The more information that can be used in statistical analysis, more reasonable subpedigrees can be found. In 2004, Falchi et al. [17] investigated an approach that reconstructed sets of subpedigrees by the use of clique partitioning algorithms. They assigned a weight to each edge on the basis of the pairwise measure of biological relationship between two individuals, such as their kinship coefficient or the number of meiotic steps separating them. Their study enables us to assign a weight as kinship coefficient or the number of meiotic steps to each edge. Then we can use a directed Steiner tree algorithm to get a subpedigree. If we can do that, more information would be considered in constructing the subpedigrees.

4.2.2 Using special properties of a pedigree graph

Since the Steiner tree problem in graph theory is NP–complete and no algorithm for any NP–complete problem is known to run in polynomial time, we provided a good approximation algorithm to a previously intractable problem. However, a pedigree graph has some special properties as a graph.

If an individual is a founder in a directed pedigree graph, then the indegree of this individual is zero. If an individual is not a founder, then the indegree of this individual is two because any individual has two parents. In addition, it is biologically impossible to have a cycle in a directed pedigree graph. Hence, any directed pedigree graph is a connected acyclic graph. These special properties of a pedigree graph enable us to find a better algorithm to solve the directed Steiner tree problem in a special graph. We believe that these special properties of a pedigree graph can help us find a better algorithm with low time complexity. As a result, we can handle larger and more complex pedigrees.

Bibliography

[1] G. R. Abecasis, S. S. Cherny, W. O. Cookson and L. R. Cardon, Merlin–rapid analysis of dense genetic maps using sparse gene flow trees, *Nat. Genet.*, **30** (2002), 97–101.

[2] R. Agarwala, L. G. Biesecker, K. A. Hopkins, C. A. Francomano and A. A. Schäffer, Software for constrcting and verifying pedigree within large genealogies and an application to the old order Amish of lancaster country, *Genome Research*, **8** (1998), 211–221.

[3] J. Andrade, M. Andersen, A. Sillén, C. Graff and J. Odeberg, The use of Grid computing to drive data–intensive genetic research, *European Journal of Human Genetics*, **15** (2007), 694–702.

[4] J. Armistead, S. Khatkar, B. Meyer, B. L. Mark, N. Patel, G. Coghlan, R. E. Lamont, S. Liu, J. Wiechert, P. A. Cattini, P. Koetter, K. Wrogemann, C. R. Greenberg, K. D. Entian KD, T. Zelinski and B. Triggs-Raine, Mutation of a gene essential for ribosome biogenesis, EMG1, causes Bowen-Conradi syndrome, *Am. J. Hum. Genet*, **84** (2009), 728–739.

[5] C. Bellenguez, C. Ober and C. Bourgain, A multiple splitting approach to linkage analysis in large pedigrees identifies a linkage to asthma on chromosome 12, *Genet. Epidemiol.*, **33** (2008), 207–217.

[6] P. Berman and V. Ramaiyer, Improved approximation algorithms for the Steiner tree problem, *J. Algorithms*, **17** (1994), 381–408.

[7] A. Borchers and D. Z. Du, The k–Steiner ratio in graphs, *SIAM Journal on Computing*, **26** (1997), 857–869.

[8] P. Bowen and G. J. Conradi, Syndrome of skeletal and genitourinary anomalies with unusual facies and failure to thrive in Hutterite sibs, *Birth. Edfects. Orig. Artic. Ser.*, **12** (1976), 101–108.

[9] J. H. Camin and R. R. Sokal. A method of deducing branching sequences in phylogeny, *Evolution*, **19** (1972), 311–326.

[10] L. L. Cavalli-Sforza and W. F. Bodmer, *The Genetics of Human Populations*, Freeman, San Francisco, 1971.

[11] M. Charikar, C. Chekuri, T. Cheung, Z. Dai, A. Goel, S. Guha and M. Li, Approximation algorithms for directed Steiner problems, *Journal of Algorithms*, **33** (1999), 73–91.

[12] J. F. Crow and M. Kimura, *An Introduction to Population Genetics Theory*, Harper and Row, New York, 1970.

[13] A. Dagklis, C. Fazi, C. Sala, V. Cantarelli, C. Scielzo, R. Massacane, D. Toniolo, F. Caligaris-Cappio, K. Stamatopoulos and P. Ghia, The immunoglobulin gene repertoire of low–count chronic lymphocytic leukemia (CLL)–like monoclonal B lymphocytosis is different from CLL: diagnostic implications for clinical monitoring, *Blood*, **114** (2009), 26–32.

[14] R. Diestel, *Graph Theory*, 2nd ed., Springer, New York, 2000.

[15] D. Z. Du and X. D. Hu, *Steiner Tree Problems in Computer Communication Networks*, World Scientific, 2008.

[16] R. C. Elandt-Johnson, *Probability Models and Statistical Methods in Genetics*, Wiley, New York, 1971.

[17] M. Falchi, P. Forabosco, E. Mocci, C. C. Borlino, A. Picciau, E. Virdis, I. Persico, D. Parracciani, A. Angius and M. Pirastu, A genomewide search using an original pairwise sampling approach for large genealogies identifies a new locus for total and low–density lipoprotein cholesterol in two genetically differentiated isolates of Sardinia, *Am. J. Hum. Genet.*, **75** (2004), 1015–1031.

[18] L. Fortnow, The status of the P versus NP problem, *Communications of the ACM*, **52** (2009), 78–86.

[19] M. R. Garey and D. S. Johnson, *Computers and Intractability: a Guide to the Theory of NP–completeness*, Framan, San Francisco, 1978.

[20] D. F. Gudbjartsson, T. Thorvaldsson, A. Kong, G. Gunnarsson and A. Ingolfsdottir, Allegro version 2, *Nat. Genet.*, **37** (2005), 1015–1016.

[21] M. P. Hassell and G. C. Varley, New inductive population model for insect parasites and its bearing on biological control, *Nature*, **223** (1969), 1133–1137.

[22] J. Hostetler, History and relevance of the Hutterite population for genetic studies, *Am. J. Med. Genet.*, **22** (1985), 453–462.

[23] S. Hougardy and H. J. Prömel, A 1.598 approximation algorithm for the Steiner problem in graphs, In: Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA'99 (1999), 448–453.

[24] J. R. John and S. Patti, Qualitative and quantitative pedigree analysis: graph theory, computer software, and case Studies, *Bioscene*, **21** (1995), 12–22.

[25] D. Jungnickel, *Graphs, Networks and Algorithms*, Second Edition, Springer, New York, 2005.

[26] J. J. Johnston, R. I. Kelley, T. O. Crawford, D. H. Morton, A. R. Agarwala, T. Koch, A. A. Schäffer, C. A. Francomano and L. G. Biesecker, A novel nemaline myopathy in the Amish caused by a mutation in troponin T, *Am. J. Hum. Genet.*, **67** (2000), 814–821.

[27] M. Karpinski and A. Zelikovsky, New approximation algorithms for the Steiner tree problems, *Journal of Combinatorial Optimization*, **1** (1997), 47–65.

[28] B. Kerem and E. Kerem, The molecular basis for disease variability in cystic fibrosis, *Eur. J. Hum. Genet.*, **4** (1996), 65–73.

[29] A. V. Kirichenko, N. M. Belonogova, Y. S. Aulchenko and T. I. Axenovich, PedStr software for cutting large pedigrees for haplotyping, IBD computation and multipoint linkage analysis, *Ann. Hum. Genet.*, **73** (2009), 527–531.

[30] L. Kruglyak, M. J. Daly, M. P. Reeve-Daly and E. S. Lander, Parametric and nonparametric linkage analysis: a unified multipoint approach, *Am. J. Hum. Genet.*, **58** (1996), 1347–1363.

[31] R. E. Lamont, J. Loredo-Osti, N. M. Roslin, J. Mauthe, G. Coghlan, E. Nylen, D. Frappier, A. M. Innes, E. G. Lemire, R. B. Lowry, C. R. Greenberg, B. L. Triggs-Raine, K. Morgan, K. Wrogemann, T. M. Fujiwara and T. Zelinski, A locus for Bowen–Conradi syndrome maps to chromosome region 12p13.3, *American Journal of Medical Genetics*, **132A** (2005), 136–143.

[32] E. S. Lander and P. Green, Construction of multilocus genetic linkage maps in humans, *Proc. Natl. Acad. Sci. USA*, **84** (1987), 2363–2367.

[33] K. Lange and R. C. Elston, Extension to pedigree analysis. I. Likelihood calculations for simple and complex pedigree, *Human Heredity*, **25** (1975), 95–105.

[34] K. Lange and J. S. Sinsheimer, The pedigree triming problem, *Hum. Hered.*, **58** (2004), 108–111.

[35] Woei-Jyh Lee, T. I. Pollin, J. R. O'Connell, R. Agarwala and A. A. Schäffer, PedHunter 2.0 and its usage to characterize the founder structure of the Old Order Amish of Lancaster County, *BMC Medical Genetics*, **11** (2010).

[36] F. Liu, A. Kirichenko, T. I. Axenovich, C. M. van Duijn and Y. S. Aulchenko, An approach for cutting large and complex pedigrees for linkage analysis, *Eur. J. Hum. Genet.*, **16** (2008), 854–860.

[37] K. Löschner, K. Morgan and J C. Loredo-Osti, Miniped finding Steiner trees in large complex pedigrees, 2004, Technical report, http://www.math.mun.ca/ loredoos/software/

[38] R. B. Lowry, A. M. Innes, F. P. Bernier, D. R. McLeod, C. R. Greenberg, A. E. Chudley, B. Chodirker, S. L. Marles, M. J. Crumley, J. C. Loredo-Osti, K. Morgan and T. M. Fujiwara, Bowen–Conradi syndrome: a clinical and genetic study, *Am. J. Med. Genet.*, **120A** (2003), 423–428.

[39] Q. Lu, Y. H. Cui, R. L. Wu, A multilocus likelihood approach to joint modeling of linkage, parental diplotype and gene order in a full-sib family, *BMC Genetics*, **5** (2004), 20.

[40] C. Maliepaard, J. Jansen and J. W. van Ooijen, Linkage analysis in a full-sib family of an outbreeding plant species: overview and consequences for applications, *Genet. Res.*, **70** (1997), 237–250.

[41] I. McIntosh and G. R. Cutting, Cystic fibrosis transmembrane conductance regulator and the etiology and pathogenesis of cystic fibrosis, *FASEB J.*, **6** (1992), 2775–2782.

[42] V. L. Nimgaonkar, T. M. Fujiwara, M. Dutta, J. Wood, K. Gentry, S. Maendel, K. Morgan and J. Eaton, Low prevalence of psychoses among the Hutterites, an isolated religious community, *Am. J. Psychiatry*, **157** (2000), 1065–1070.

[43] V. S. Pankratz and S. J. Iturria, A pedigree partitioning approach to quantitative trait loci mapping of IgE serum level in the GAW12 Hutterite data, *Genet. Epidemiol.*, **21** (2001), S258–S263.

[44] L. Peltonen, A. Palotie and K. Lange, Use of population isolates for mapping complex traits, *Nature Reviews*, **1** (2000), 182–190.

[45] H. J. Prömel and A. Steger, RNC-approximation algorithms for the Steiner problem, Proceedings of STACS 1997, 559–570.

[46] G. Robins and A. Zelikovsky. Tighter bounds for graph Steiner tree approximation, *SIAM Journal on Discrete Mathematics*, **19** (2005), 122–134.

[47] M. J. Rosenberg, R. Agarwala, G. Bouffard, J. Davis, G. Fiermonte, M. S. Hilliard, T. Koch, L. M. Kalikin, I. Makalowska, D. H. Morton, E. M. Petty, J. L. Weber, F. Palmieri, R. I. Kelley, A. A. Schäffer and L. G. Biesecker, Mutant deoxinucleotide carrier is associated with congenital microcephaly, *Nature Genetics*, **32** (2002), 175–179.

[48] H. Takahashi and A. Matsuyama, An approximate solution for the Steiner problem in graphs, *Math. Japonica*, **24** (1980), 573–577.

[49] A. Thomas, *Data Structures, Methods of Approximation and Optimal Computation for Pedigree Analysis*, Ph.D. thesis, Cambridge University, 1985.

[50] E. A. Thompson, *Pedigree Analysis in Human Genetics*, The Johns Hopkins University Press, Baltimore, 1986.

[51] T. A. Thornton and J. L. Haines, PowerTrim: An automated decision support algorithm for preprocessing family–based genetic data, *Am. J. Hum. Genet.*, **72** (2003), 1280–1281.

[52] G. Valiente, *Algorithms on Trees and Graphs*, Spinger, New York, 2002.

[53] T. Varilo and L. Peltonen, Isolates and their potential use in complex gene mapping efforts, *Curr. Opin. Genet. Dev.*, **14** (2004), 316–323.

[54] V. V. Vazirani, *Approximation Algorithms*, Springer, New York, 2001.

[55] R. L. Wu, C.-X. Ma and G. Casella, *Statistical Genetics of Quantitative Traits: Linkage, Maps, and QTL*, Springer, New York, 2007.

[56] A. Zelikovsky, An 11/6 approximation algorithm for the network Steiner problem, *Algorithmica*, **9** (1993), 463–470.

[57] A. Zelikovsky, Better approximation bounds for the network and Euclidean Steiner tree problems, Technical Report, CS-96-06, University of Virginia, Charlottesville, VA, USA.

[58] A. Zelikovsky, A series of approximation algorithms for the acyclic directed Steiner Tree problem, *Algorithmica*, **18** (1997), 99–110.